Simulation and Experimental Evaluation of Multipath Congestion Control Strategies

Thomas Dreibholz, Hakim Adhari, Martin Becke, Erwin P. Rathgeb
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstraße 29, 45326 Essen, Germany
{dreibh,hakim.adhari,martin.becke,rathgeb}@iem.uni-due.de

***Abstract*—The need for service resilience is leading to a steadily growing number of multi-homed Internet sites. In consequence, this results in a growing demand for utilising multiple Internet accesses simultaneously, in order to improve application payload throughput during normal operation. Multi-path Transport Layer protocol extensions – like Multi-Path TCP (MPTCP) for TCP and Concurrent Multipath Transfer for SCTP (CMT-SCTP) – allow applications to make use of such network topologies.**

However, since TCP – which constitutes the basis of most Internet applications – and its congestion control procedures have been designed under the assumption of single-homed sites, fairness issues may arise by the usage of multipath transport. These issues are addressed by advanced congestion control approaches, which have already been examined by simulations. However, real-life network measurements are missing. In this paper, we perform an experimental proof-of-concept evaluation of several multipath congestion control strategies, which are currently under discussion within the IETF in the context of MPTCP as well as CMT-SCTP. Particularly, we validate effects that have been observed in simulations, in order to trigger further discussions on multipath congestion control. Also, our goal is to provide insight into the different approaches to support the ongoing IETF standardisation activities on multipath transport protocols.[1][2][3][4]

Keywords: Multipath Transfer, Congestion Control, Performance Analysis, Simulation, Measurement

I. Introduction

In the last years, communication devices such as laptops and smart phones have become more and more common. One of the important features of these devices is the availability of more than one network interface (e.g. W-LAN and UMTS), providing multiple network carriers and access technologies to be part of the communication. In addition to other goals which could benefit from the existence of multiple IP addresses, such as providing mobility or increasing availability, it seems to be a natural evolution to aggregate bandwidths in order to achieve throughput benefits here. This is denoted as *load sharing*. To be able to support load sharing for end-to-end transport, multiple approaches are already available. Currently under discussion in the IETF are end-to-end Transport Layer protocols and protocol extensions, such as Multipath TCP (MPTCP [1]) and Concurrent Multipath Transfer for SCTP (CMT-SCTP [2]).

One of the issues related with load sharing on the Transport Layer is the congestion control (CC) mechanism used. The standard CC mechanisms used by TCP [3] or by SCTP [4] are working well for *single path* data transfer. However, this is not the case any more for *multipath* data transfer. New approaches are requested here. In fact, using the single-path approaches with multiple paths [2] results in unfairness when paths share a common bottleneck link [5]. Here, it is obvious that there is a need for a CC mechanism with multipath data transfer in mind. Two different strategies – our own CMT/RP CC [6] for CMT-SCTP as well as MPTCP CC [7] for MPTCP – are currently discussed within the IETF.

Our goal is to evaluate these CC mechanisms and to analyse their behaviour in scenarios with dissimilar paths – as introduced in [8]–[10] – which are challenging scenarios for efficient multipath transport but very common in realistic Internet setups. A first step of this work has been performed in [6]. However – and in contradiction to [6], where only simulative results have been presented – this paper extends this work with real testbed measurements. In particular, we also validate our simulation model by comparing the results of simulations and testbed experiments.

This paper is structured as follows: first, we introduce the existing CC mechanisms for multipath transfer. After that, the simulation as well as the testbed topologies are described and the used parameters are provided. This is followed by the evaluation of the CC mechanisms behaviour in the relation with the chosen topologies. Our evaluation is based on CMT-SCTP. However, the results are generic and may be adapted to other protocols (particularly MPTCP) as well. Finally, we conclude our work and give an overview of future goals for multipath-aware CC mechanisms.

II. Congestion Control Basics

CC is a topic which has become a facet of daily life for Internet users. In such a network, a user is provided, in most of the cases, with a *best effort* service [11]. This means that the network does its best to deliver the data and – at the same time – avoid congestion. Both goals are provided by CC. In the following, we will first introduce the CC basics in general and in a second step present the CC mechanisms designed for multipath transfer.

[1]Parts of this work have been funded by the German Research Foundation (Deutsche Forschungsgemeinschaft – DFG).

[2]Parts of this work have been funded within the scope of the G-Lab project by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung – BMBF).

[3]The authors would like to thank Michael Tüxen for his friendly support.

[4]The authors would like to thank the reviewers for their helpful comments.

A. Classical Congestion Control

TCP as well as SCTP [4] both use window-based CC. That is, the sender is maintaining the maximum allowed amount of outstanding data. This amount is stored as *congestion window* variable c. Once transmitted data is acknowledged, c may be increased. In the case of congestion, c is decreased. This specific change behaviour is denoted as *Additive Increase, Multiplicative Decrease (AIMD)*.

The variation of c depends on which phase the CC is situated in. The first one is called *slow start* phase [12]. Here, c may grow on every acknowledgement, leading to an exponential growth as long as $c \leq s$, where s denotes the so-called *slow-start threshold* variable. The slow start phase is followed by the *congestion avoidance* phase. Here, i.e. for $c > s$, a linear growth of c is applied. An additional counter p (denoted as *partial acknowledgements*) is also recommended. It maintains the amount of previous acknowledgements. An increase of c is possible for $p \geq c$; on advance, p is reset.

In case of a possible packet loss (signalled by the reception of three duplicate acknowledgements), a so-called Fast Retransmission (Fast RTX; see [12]) is performed by halving s, setting $c = s$ and retransmitting the lost segment. Usually, Fast RTX is combined with so-called Fast Recovery (see also [12]), which means to forbid growing c until the retransmitted segment has been acknowledged. Further losses of the same segment are assumed as a sign of severe congestion. Therefore, its retransmission is triggered by expiration of the segment's Retransmission Timer. It is therefore denoted as Timer-Based Retransmission (Timer-Based RTX). At that time, s is halved but c minimized.

In most state-of-the art CC implementations, the congestion window c and slow start threshold s are stored in bytes. In some older implementations, packets are used instead of bytes. It has to be noted here that the Maximum Transmission Unit (MTU) is not necessarily the same for different network types. These differences may lead to different byte amounts of data for the same setting of c (i.e. a counting in packets would hide this fact).

Similar to the MTU, which is a limit of the amount of data on the Data Link Layer – another limit, called Maximum Segment Size (MSS) exists on the Transport Layer and specifies the largest amount of payload data that can be sent on a single segment. It is e.g. 1,460 bytes for TCP or 1,452 bytes for SCTP using IPv4 over a 100BaseTX Ethernet interface with an MTU of 1,500 bytes.

B. Congestion Control for Multi-Path Transfer

1) Plain CMT Congestion Control: Clearly, the naïve approach for multipath CC is to assume disjoint paths and therefore to just apply classical CC for each path independently – as for CMT-SCTP introduced in [2]. Then, for each path P, there are independent congestion window c_P, slow-start threshold s_P and partial acknowledgements p_P variables.

On α newly acknowledged bytes on path P in a fully-utilised congestion window, c_P is adapted as follows:

$$c_P = c_P + \begin{cases} \min\{\alpha, \text{MSS}_P\} & (c_P \leq s_P) \\ \text{MSS}_P & (c_P > s_P \wedge p_P > c_P) \end{cases}.$$

SCTP as well as state-of-the-art TCP implementations apply so-called Appropriate Byte Counting [13], i.e. c_P is only advanced by the minimum of the acknowledged bytes α and MSS_P the MSS of P in slow start (i.e. $c_P \leq s_P$), as well as only by MSS_P in congestion avoidance (i.e. $c_P > s_P$).

On retransmission on path P, s_P and c_P are adapted as follows:

$$\begin{aligned} s_P &= \max\{c_P - \frac{1}{2} * c_P, 4 * \text{MSS}_P\}, \\ c_P &= \begin{cases} s_P & \text{(Fast RTX)} \\ \text{MSS}_P & \text{(Timer-Based RTX)} \end{cases}. \end{aligned}$$

In result, having n paths sharing a single bottleneck, a multipath flow gets n times the bandwidth of a concurrent single-path flow – which is quite unfair [5], [6].

2) CMT/RP Congestion Control: In order to deal with the unfairness problem of CMT-SCTP, the idea of resource pooling (RP) [14] can be applied. That is, a collection of resources is supposed to behave like a single pooled resource. In the case of multipath transfer, the set of all paths should behave like a single one. RP should fulfil the following goals [7] here:

1) A CMT flow should get at least as much bandwidth as a single-homed flow via the best path.
2) A CMT flow should not take more capacity on a shared bottleneck path than a single-homed flow via the same bottleneck.
3) A CMT flow should balance congestion on all of its paths.

CMT/RP [5] version 1 – shortly CMT/RPv1 – is our initial approach to apply RP to CMT-SCTP. In this case, the slow start threshold is used as a metric to describe the capacity of a path P. The *slow-start threshold ratio* $\hat{s}_P$ is defined as:

$$\hat{s}_P = \frac{s_P}{\sum_i s_i}. \tag{1}$$

On α acknowledged bytes on path P in a fully-utilised congestion window, c_P is adapted according to the slow-start threshold ratio $\hat{s}_P$ of P as follows:

$$c_P = c_P + \begin{cases} \lceil \hat{s}_P * \min\{\alpha, \text{MSS}_P\} \rceil & (c_P \leq s_P) \\ \lceil \hat{s}_P * \text{MSS}_P \rceil & (c_P > s_P \wedge p_P > c_P) \end{cases}.$$

On retransmission on path P, s_P and c_P are adapted as follows:

$$\begin{aligned} s_P &= \max\{\lceil c_P - \frac{1}{2} * \sum_i c_i \rceil, \lceil \hat{s} * 4 * \text{MSS}_P \rceil, \text{MSS}_P\}, \\ c_P &= \begin{cases} s_P & \text{(Fast RTX)} \\ \text{MSS}_P & \text{(Timer-Based RTX)} \end{cases}. \end{aligned}$$

CMT/RPv1 CC works well for similar paths [5], i.e. paths having almost the same characteristics (bandwidth, delay, error rate). In fact, CMT/RPv1 assumes comparable slow start thresholds s_i in the computation of the ratio $\hat{s}_P$ of a path P. However, this may be difficult in case of dissimilar paths [9], [10].

CMT/RPv2 instead is based on the *increase factor* $\hat{i}_P$, which represents the current bandwidth share of P on the total bandwidth of the flow. It is defined as:

$$\hat{i}_P = \frac{\frac{c_P}{\text{RTT}_P}}{\sum_i \frac{c_i}{\text{RTT}_i}}.$$

$\hat{i}$ is used to adapt c_P as follows on α acknowledged bytes on path P in a fully-utilised congestion window:

$$c_P = c_P + \begin{cases} \lceil \hat{i} * \min\{\alpha, \text{MSS}_P\} \rceil & (c_P \leq s_P) \\ \lceil \hat{i} * \text{MSS}_P \rceil & (c_P > s_P \wedge p_P > c_P). \end{cases}$$

For reducing c_P on a packet loss on path P, the *decrease factor* $\hat{d}_P$ is applied. $\hat{d}$ represents the factor by which the bandwidth of P should be reduced in order to halve the total bandwidth of the flow. Let us consider the example of two paths P_1 (10 Mbit/s) and P_2 (2 Mbit/s). A loss on P_1 leads to $\hat{d}_1 = \frac{1}{2} * \frac{12}{10} = 0.6$; a loss on P_2 to $\hat{d}_2 = \frac{1}{2} * \frac{12}{2} = 3.0$. That is, $\hat{d}_P$ is defined as follows:

$$\hat{d}_P = \max\{\frac{1}{2}, \frac{1}{2} * \frac{\sum_i \frac{c_i}{\text{RTT}_i}}{\frac{c_P}{\text{RTT}_P}}\}.$$

Using $\hat{d}_P$, s_P and c_P are adapted as follows:

$$\begin{aligned} s_P &= \max\{c_P - \lceil \hat{d}_P * c_P \rceil, 1 * \text{MSS}_P\}, \\ c_P &= \begin{cases} s_P & \text{(Fast RTX)} \\ \text{MSS}_P & \text{(Timer-Based RTX)} \end{cases}. \end{aligned}$$

This means that the new setting of c_P tries to halve the total bandwidth, with a lower bound of MSS_P.

3) MPTCP-Like Congestion Control: Another CC based on the RP principle is the CC of MPTCP [7], [15]. In contradiction to CMT/RP, which tries to halve the *total* congestion window/total bandwidth on a packet loss on path P, MPTCP CC works differently: it behaves exactly like standard TCP or SCTP by only halving the *path* congestion window c_P. In order to avoid an unfair bandwidth allocation, the congestion window growth behaviour is adapted: a per-flow *aggressiveness factor* $\hat{a}$ is used to bring the increases and decreases of c_P into equilibrium.

The MPTCP CC introduced in [7] is based on packets instead of bytes. However, the CC of most state-of-the-art TCP implementations as well as of SCTP (which is used in our testbed setup as well as in our simulation environment for the evaluation) count bytes instead. Consequently, we had to port the MPTCP CC accordingly to also use bytes instead of packets. That is, on α acknowledged bytes on path P in a fully-utilised congestion window, our MPTCP-like CC adapts c_P as follows:

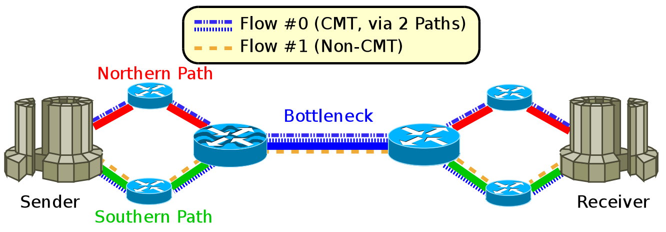


Figure 1. Bottleneck Scenario

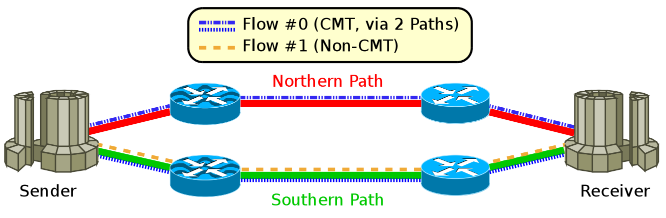


Figure 2. Disjoint Paths Scenario

$$c_P = c_P + \begin{cases} \min\left\{\left\lceil \frac{c_P * \hat{a} * \min\{\alpha, \text{MSS}_P\}}{\sum_i c_i} \right\rceil, \min\{\alpha, \text{MSS}_P\}\right\} \\ \quad (c_P \leq s_P) \\ \min\left\{\left\lceil \frac{c_P * \hat{a} * \text{MSS}_P}{\sum_i c_i} \right\rceil, \text{MSS}_P\right\} \\ \quad (c_P > s_P \wedge p_P \geq c_P) \end{cases}.$$

$\hat{a}$ denotes the per-flow *aggressiveness factor*, defined as:

$$\hat{a} = \left(\sum_i c_i\right) * \frac{\max_i \left\{\frac{c_i/\text{MSS}_i}{(\text{RTT}_i)^2}\right\}}{\left(\sum_i \frac{c_i/\text{MSS}_i}{\text{RTT}_i}\right)^2}.$$

III. Simulation and Testbed Setup

In order to evaluate the described CC mechanisms, two basic scenarios have been considered. The first topology, which is illustrated in Figure 1, consists of two paths (Northern and Southern Path) that have a common bottleneck link. This scenario is therefore denoted as *bottleneck scenario*. The other scenario, which is depicted in Figure 2, uses two disjoint paths. It is therefore denoted as *disjoint path scenario*. In both cases, Flow #0 is a CMT-SCTP flow (trying to utilise both paths) and Flow #1 is a standard SCTP flow (only using the Southern Path). These scenarios have been established in a simulation environment as well as a real testbed setup.

The simulation environment has used the OMNeT++-based INET framework with the CMT-SCTP model introduced in [16] and the SimProcTC [17] tool-chain for parametrisation and result post-processing. The testbed setup consists of x86_64-based FreeBSD 8.2 machines running the latest SCTP development kernel. In order to apply bandwidth limitations, Dummynet [18], [19] – which is part of the FreeBSD kernel – is used on the routers.

Otherwise specified, the following configuration parameters have been used for both, simulation and testbed measurement:

- The sender has been saturated (i.e. it has tried to transmit as much data as possible); the message size has been

1,452 bytes at an MTU of 1,500 bytes. All messages have used unordered delivery.

- The send and receive buffer sizes have been set to 1,000,000 bytes for the bottleneck scenario and to 5,000,000 bytes for the disjoint path scenario (i.e. they are sufficiently large for these scenarios). Buffer Splitting according to [8], [9] as well as the state-of-the-art SCTP features NR-SACK [20] and SACK Immediately [21] have been used.
- On the routers, RED queues have been configured, using the parameters MinTh=30, MaxTh=90 and MaxP=10% (following the recommendations by [22]). All other queues have been FIFO queues with a capacity of 100 packets.
- The link delay between the core routers has been δ=10 ms.

The measurement as well as the simulation runtime has been 300 s, after a transient phase of 20 s. Each run has been repeated at least 20 times in order to ensure a sufficient statistical accuracy. The results plots show the average values and their corresponding 95% confidence intervals.

IV. Performance Analysis

A. Bottleneck Scenario

In the first scenario, we examine the performance of the different CC strategies in the bottleneck scenario illustrated in Figure 1. Figure 3 presents the resulting application payload throughput of the simulation (in Subfigure 3(a)) and the corresponding measurement (in Subfigure 3(b)) for varying the bottleneck bandwidth ρ_{B}. Flow #0 (i.e. F=0 – shown by a solid line) represents the CMT-SCTP flow, Flow #1 (i.e. F=1 – shown by a dashed line) the non-CMT reference flow.

The curves for plain CMT CC (Γ=cmt; i.e. non-coupled CCs) – as shown by curves 1 and 2 – make the initial problem clear: the CMT-SCTP Flow #0, assuming two independent paths, occupies twice the bandwidth of the non-CMT Flow #1. That is, the fairness issue – motivating the research on new CC strategies – is clearly visible in the simulation as well as in the measurement results. The slightly lower throughput of the CMT-SCTP flow – leading in result to a small improvement for the non-CMT reference flow – in the measurement is assumed to be a result of implementation differences on optional parts like burst mitigation and CMT fast recovery handling. As part of future work, it is planned to update the kernel to the state of the art in CMT-SCTP research.

Applying one of the RP-based CCs – i.e. CMT/RPv1 (Γ=cmtrpv1; curves 3 and 4), CMT/RPv2 (Γ=cmtrpv2; curves 5 and 6) or MPTCP-like (Γ=like-mptcp; curves 7 and 8) – solves the fairness issue reasonably well in the simulation as well as in the measurement. The small throughput difference between the two flows is caused by the small delay of the setup (i.e. δ=10 ms) in combination with the need to probe *all* paths. That is, the smallest useful congestion window size $c_{\min}$ is the smallest path MTU (sPMTU) of any of the association's paths. Then, for each path, there is at least a stop-and-wait transmission with one sPMTU per path RTT possible. In our setup, for ρ_{B}=9 Mbit/s, δ=10 ms and RTT$\approx$25 ms:

$$\frac{1,000\text{ ms}}{25\text{ ms}}\text{ packet/s} * \underbrace{1,500\text{ B/packet}}_{\text{MTU}} * 8\text{ bit/B} = 0.48\text{ Mbit/s}.$$

This corresponds to the simulations results. As reasoned above, the FreeBSD kernel CMT-SCTP behaves slightly less aggressive – leaving more room for the reference flow.

A solution for the stop-and-wait challenge is proposed by [23]: by temporary blocking transmission on a path, the mechanism *RP Path Blocking* reduces the throughput when a path's congestion window is on its lower limit $c_{\min}$. Since this mechanism is not yet supported by FreeBSD kernel CMT-SCTP, a further analysis has to be made as future work.

In summary, it can be shown that the new CC strategies solve the general fairness issue – in simulation as well as in reality. However, a common bottleneck just represents the worst case scenario. Furthermore, the new CC strategies must also improve the performance in the better cases – i.e. for disjoint paths. Therefore, we have also examined such a scenario.

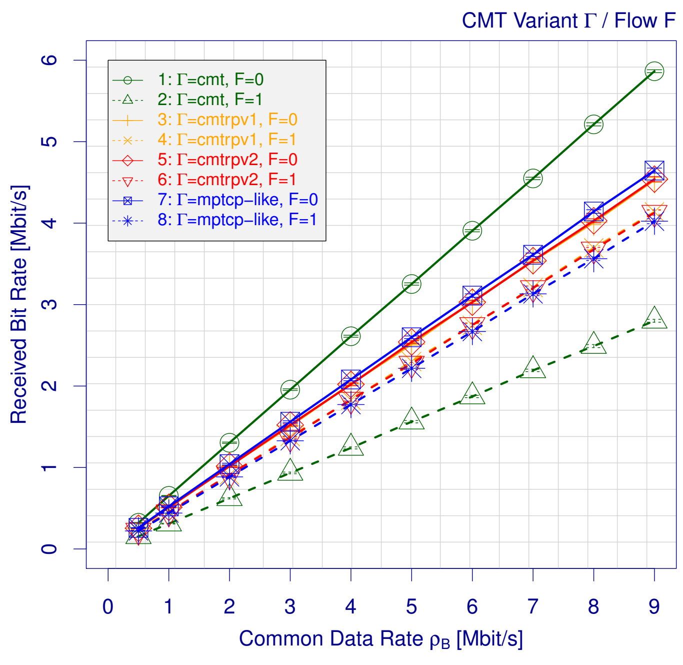
B. Disjoint Paths Scenario

The disjoint paths scenario uses the setup illustrated in Figure 2. That is, the Northern Path is exclusively utilised by the CMT-SCTP flow while the Southern Path is shared by both flows. Figure 4 presents the application payload throughput results of simulation (in Subfigure 4(a)) and measurement (in Subfigure 4(b)) for varying the bandwidth of the shared Southern Path ρ_{S}. The Northern Path bandwidth is fixed at ρ_{N}=2.5 Mbit/s.

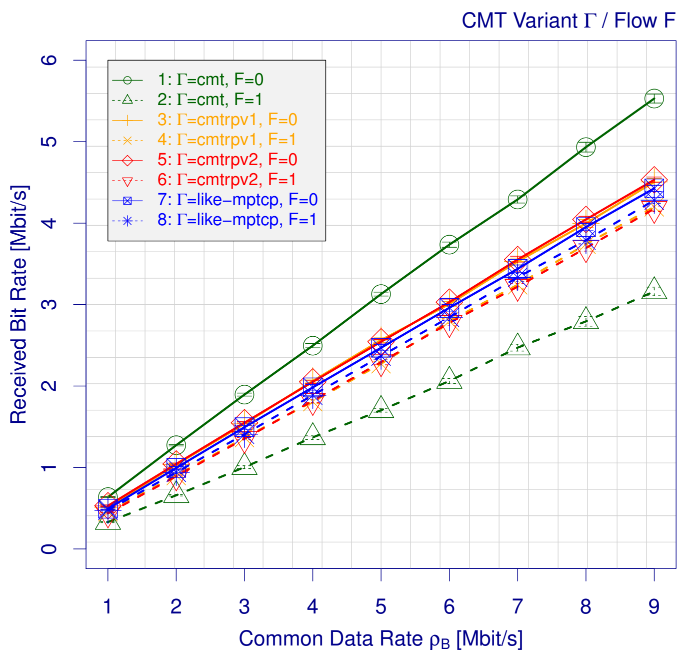
The plain CMT CC performance (curves 1 and 2) represents the baseline performance: none of the flows should achieve a lower throughput than the reference flow (curve 2) here. Also, none of the flows should get a higher throughput than the CMT-SCTP flow here (curve 1), since this would mean a too-aggressive bandwidth occupation. Since the bandwidth of the Northern Path remains constant at ρ_{N}=2.5 Mbit/s, the two curves are parallel – with a difference of about 2.5 Mbit/s.

CMT/RPv1 CC does not achieve this performance goal: while the CMT-SCTP flow throughput falls below the lower baseline (curve 3) for $\rho_{\mathrm{S}}\geq$7 Mbit/s, the reference flow exceeds the upper baseline at the same settings. As reasoned for the simulations in [6], the dissimilarity of the paths (here: different bandwidths) make the slow-start thresholds incomparable (see also Equation 1). That is, as validated by the measurements, CMT/RPv1 is not a useful CC strategy when paths become dissimilar.

CMT/RPv2 CC (curves 5 and 6) as well as MPTCP-like CC (curves 7 and 8) are able to cope with dissimilar paths, as already expected from the simulation results by [6]. However, the behaviour for a rising Southern Path bandwidth ρ_{S} is interesting: at about $\rho_{\mathrm{S}}\geq$4 Mbit/s (CMT/RPv2) or $\rho_{\mathrm{S}}>$6 Mbit/s (MPTCP-like), the throughput of the reference flow exceeds the performance achieved by the CMT-SCTP flow. Nevertheless, both flows achieve a performance improvement in

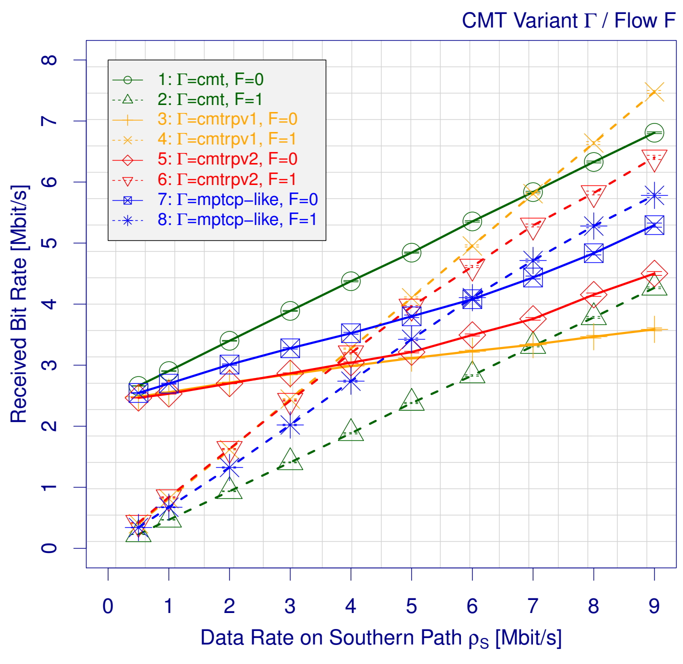


(a) Simulation

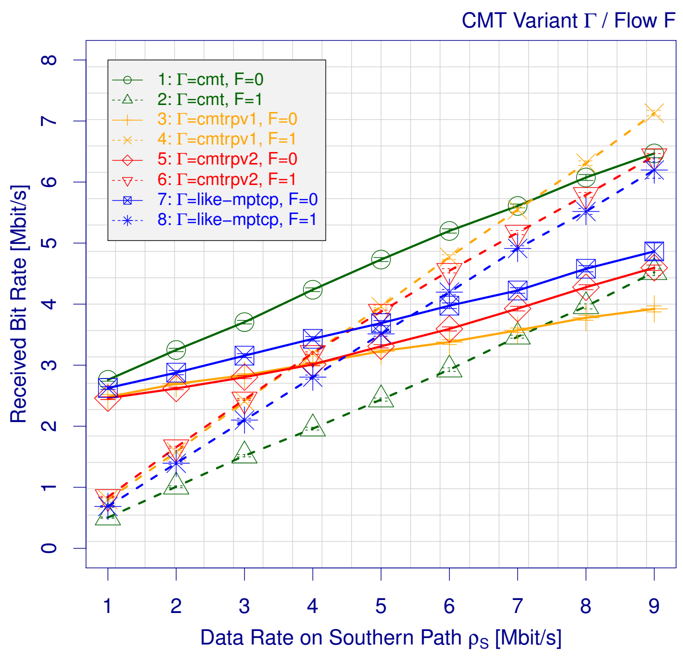


(b) Testbed Measurement

Figure 3. Results for the Bottleneck Scenario



(a) Simulation



(b) Testbed Measurement

Figure 4. Results for the Disjoint Paths Scenario

comparison to a single-path scenario without the Northern Path (where each flow may only occupy a bandwidth of $\frac{\rho_S}{2}$). That is, this effect – first observed by the simulations in [6] – has now also been validated in a real network setup.

Note, that this performance result is certainly useful for the network provider (both users get a better performance) and for the non-CMT flow user (significant performance improvement for free). However, from the perspective of the CMT flow user, there is an increased maintenance overhead (due to the handling of multiple paths) for only a small performance improvement. Therefore, the notion of "fairness" in presence of multipath transfer is to be discussed in more detail as part of the ongoing and future work.

In summary, it can be shown that the new CC strategies are also useful in disjoint path scenarios – in simulation as well as in reality.

V. Conclusions

With the steadily growing number of multi-homed Internet sites, multipath transport is becoming an increasingly hot topic in research as well as in the IETF standardisation discussions. Appropriate congestion control strategies are an important subject of these discussions. While there are several approaches for the multipath transfer protocol extensions under standardisation (i.e. MPTCP and well as CMT-SCTP), a comparison of these strategies with both, simulations as well as measurements, has not been made before.

In this paper, we have presented the results of a proof-of-concept evaluation for the relevant multipath congestion control approaches by comparing simulation results and lab measurements. We have shown that CMT/RPv2 as well as MPTCP-like congestion control are useful approaches, delivering the expected performance. However, as part of future work, there is a need to further formalise the fairness criteria of multipath transport, in order to provide a more fine-granular classification. Also, a broader range of Internet setups has to be evaluated, e.g. by using the G-Lab environment [24]. Particularly, these evaluations have to cover the impact of delay, with appropriate mechanisms to handle small RTTs, e.g. by RP Path Blocking as suggested in [23]. Furthermore, we are also going to contribute our results into the ongoing IETF standardisation process of MPTCP and CMT-SCTP [25].

References

[1] A. Ford, C. Raiciu, M. Handley, S. Barré, and J. R. Iyengar, "Architectural Guidelines for Multipath TCP Development," IETF, Informational RFC 6182, Mar. 2011, ISSN 2070-1721.

[2] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent Multipath Transfer using SCTP Multihoming over Independent End-to-End Paths," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 951–964, Oct. 2006, ISSN 1063-6692.

[3] J. B. Postel, "Transmission Control Protocol," IETF, Standards Track RFC 793, Sept. 1981, ISSN 2070-1721.

[4] R. R. Stewart, "Stream Control Transmission Protocol," IETF, Standards Track RFC 4960, Sept. 2007, ISSN 2070-1721.

[5] T. Dreibholz, M. Becke, J. Pulinthanath, and E. P. Rathgeb, "Applying TCP-Friendly Congestion Control to Concurrent Multipath Transfer," in *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, Perth/Australia, Apr. 2010, pp. 312–319, ISSN 1550-445X.

[6] T. Dreibholz, M. Becke, H. Adhari, and E. P. Rathgeb, "On the Impact of Congestion Control for Concurrent Multipath Transfer on the Transport Layer," in *Proceedings of the 11th IEEE International Conference on Telecommunications (ConTEL)*, Graz/Austria, June 2011, pp. 397–404, ISBN 978-953-184-152-8.

[7] C. Raiciu, M. Handley, and D. Wischik, "Practical Congestion Control for Multipath Transport Protocols," University College London, London/United Kingdom, Tech. Rep., 2009.

[8] T. Dreibholz, M. Becke, E. P. Rathgeb, and M. Tüxen, "On the Use of Concurrent Multipath Transfer over Asymmetric Paths," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Miami, Florida/U.S.A., Dec. 2010, ISBN 978-1-4244-5637-6.

[9] H. Adhari, T. Dreibholz, M. Becke, E. P. Rathgeb, and M. Tüxen, "Evaluation of Concurrent Multipath Transfer over Dissimilar Paths," in *Proceedings of the 1st International Workshop on Protocols and Applications with Multi-Homing Support (PAMS)*, Singapore, Mar. 2011, pp. 708–714, ISBN 978-0-7695-4338-3.

[10] T. Dreibholz, R. Seggelmann, M. Tüxen, and E. P. Rathgeb, "Transmission Scheduling Optimizations for Concurrent Multipath Transfer," in *Proceedings of the 8th International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNeT)*, vol. 8, Lancaster, Pennsylvania/U.S.A., Nov. 2010, ISSN 2074-5168.

[11] M. Welzl, *Network Congestion Control: Managing Internet Traffic*. Chichester, West Sussex/United Kingdom: John Wiley & Sons, 2005, ISBN 978-0-470-02528-4.

[12] M. Allman, V. Paxson, and E. Blanton, "TCP Congestion Control," IETF, Standards Track RFC 5681, Sept. 2009, ISSN 2070-1721.

[13] M. Allman, "TCP Congestion Control with Appropriate Byte Counting (ABC)," IETF, RFC 3465, Feb. 2003, ISSN 2070-1721.

[14] D. Wischik, M. Handley, and M. B. Braun, "The Resource Pooling Principle," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 5, pp. 47–52, Oct. 2008, ISSN 0146-4833.

[15] C. Raiciu, M. Handley, and D. Wischik, "Coupled Multipath-Aware Congestion Control," IETF, Network Working Group, Internet Draft Version 07, July 2011, draft-ietf-mptcp-congestion-07, work in progress.

[16] T. Dreibholz, M. Becke, J. Pulinthanath, and E. P. Rathgeb, "Implementation and Evaluation of Concurrent Multipath Transfer for SCTP in the INET Framework," in *Proceedings of the 3rd ACM/ICST International Workshop on OMNeT++*, Torremolinos, Málaga/Spain, Mar. 2010, ISBN 978-963-9799-87-5.

[17] T. Dreibholz, X. Zhou, and E. P. Rathgeb, "SimProcTC – The Design and Realization of a Powerful Tool-Chain for OMNeT++ Simulations," in *Proceedings of the 2nd ACM/ICST International Workshop on OMNeT++*, Rome/Italy, Mar. 2009, pp. 1–8, ISBN 978-963-9799-45-5.

[18] M. Carbone and L. Rizzo, "Dummynet Revisited," Dipartimento di Ingegneria dell'Informazione, Università di Pisa, Pisa/Italy, Tech. Rep., May 2009.

[19] M. Becke, T. Dreibholz, E. P. Rathgeb, and J. Formann, "Link Emulation on the Data Link Layer in a Linux-based Future Internet Testbed Environment," in *Proceedings of the 10th International Conference on Networks (ICN)*, St. Maarten/Netherlands Antilles, Jan. 2011, pp. 92–98, ISBN 978-1-61208-002-4.

[20] P. Natarajan, N. Ekiz, E. Yilmaz, P. D. Amer, and J. R. Iyengar, "Non-Renegable Selective Acknowledgments (NR-SACKs) for SCTP," in *Proceedings of the 16th IEEE International Conference on Network Protocols (ICNP)*, Orlando, Florida/U.S.A., Oct. 2008, pp. 187–196, ISBN 978-1-4244-2506-8.

[21] M. Tüxen, I. Rüngeler, and R. R. Stewart, "SACK-IMMEDIATELY Extension for the Stream Control Transmission Protocol," IETF, Individual Submission, Internet Draft Version 08, Oct. 2011, draft-tuexen-tsvwg-sctp-sack-immediately-08.txt, work in progress.

[22] S. Floyd, "RED: Discussions of Setting Parameters," Nov. 1997.

[23] T. Dreibholz, "Evaluation and Optimisation of Multi-Path Transport using the Stream Control Transmission Protocol," Habilitation Treatise, University of Duisburg-Essen, Faculty of Economics, Institute for Computer Science and Business Information Systems, Apr. 2012.

[24] M. Kleis, J. Müller, A. Siddiqui, and M. Becke, "Evaluating a Future Internet Cross-Layer Composition Prototype," in *Proceedings of the 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, Shanghai/People's Republic of China, Apr. 2011.

[25] M. Becke, T. Dreibholz, J. R. Iyengar, P. Natarajan, and M. Tüxen, "Load Sharing for the Stream Control Transmission Protocol (SCTP)," IETF, Network Working Group, Internet Draft Version 02, July 2011, draft-tuexen-tsvwg-sctp-multipath-02.txt, work in progress.