Performance Evaluation of Multipath TCP Linux Implementations

Amanpreet Singh, Carmelita Görg
Communication Networks
University of Bremen
Bremen, Germany
{aps|cg@comnets.uni-bremen.de}

Andreas Timm-Giel
Institute of Communication Networks
Hamburg University of Technology
Hamburg, Germany
{timm-giel@tuhh.de}

Michael Scharf, Thomas-Ralf Banniza
Bell Labs Germany
Alcatel-Lucent
Stuttgart, Germany
{Michael.scharf|Thomas-Rolf.Banniza@alcatel-lucent.com}

I. INTRODUCTION

Nowadays, wireless devices are often equipped with multiple interfaces such as fixed network, wireless LAN, UMTS, WiMAX, Bluetooth and so on. These enhancements allow for deploying multipath transport to increase throughput, improve resilience and balance congestion in the network [1]. Theoretically, it is possible to transport data simultaneously over more than one interface in order to increase the transfer speed. In practice, however, legacy transport protocols such as TCP only support a single end-to-end connection over one interface. In order to do so, a data flow has to be split into multiple subflows which are then transferred over the different interfaces. Each subflow contains a subset of the packets which form the data stream. At the receiving station, these subflows have again to be reassembled before being handed over to the application.

II. Multipath Transport of Application Data Streams

This work is focused on the Multipath TCP (MPTCP) [2] which is a set of extensions for TCP that allows spreading of a single TCP flow across multiple subflows. One of the design goals for such a solution is that each subflow should appear to the network as a normal TCP flow.

MPTCP operations need additional signalling information in the TCP segments, either in the TCP options field or payload. Issues to be negotiated are the multipath capability and the currently available interfaces. Further, it must be tracked to which connection each subflow belongs. Finally, data sequence numbers, an accumulative data acknowledgement and a data FIN is required at the connection level.

The MPTCP proposal uses a dual sequence number space. Each subflow has its own sequence space that identifies bytes within a subflow as if it was running alone. There is also a data (or connection level) sequence space, which allows reordering at the (aggregated) connection level. Each segment carries both subflow and data sequence numbers. Retransmissions are driven only by the subflow sequence number; hence MPTCP avoids problems due to connection level reordering of packets. The challenge to be met is that the different links over which the subflows are transported may have different throughputs and delays, so the arrival order of the packets at the receiver is unpredictable. In MPTCP, congestion control is coupled across paths, so as to ensure fairness without needing to detect shared bottlenecks [3].

The beforementioned topic of multipath TCP transmission is currently under investigation at the IETF where different solutions have been published as drafts: Multipath TCP (MPTCP [1]) [2][3], Payload Multi-connection Transport (PLMT) [4] and Multiple Connection TCP (MCTCP) [5]. The solutions differ in the in the structure of the protocol stack as well as signalling:

- In MPTCP, each multipath TCP subflow looks to the network as a normal TCP flow, with the only difference that it carries new TCP options for MPTCP signalling. It is a kernel-space solution where the existing TCP stack is modified.
- PLMT encodes all the signalling information in the payload of TCP connections. It operates as an additional protocol layer in the user space on top of existing TCP stacks so that the latter do not need to be modified. In this work, it is not further considered.
- MCTCP is a hybrid variant that encodes control information, as far as possible, in the payload of the TCP connections. It uses the TCP option field for the connection setup messages (SYN/ACK for MPCapability and Join messages). It is transparent in the single-path case.

A comparison of the three variants for multipath TCP was also discussed in [6].

This work compares the performance of the MPTCP and MCTCP protocol in a dynamically changing network subjected to varying delay and packet loss rate. MPTCP's Linux kernel implementation is currently open source [7]. The development of the MCTCP concept and its implementation in the Linux kernel as well as user space has been done within the German-Lab Phase-2 project NETCOMP [8].

[1] In this context, MPTCP is the name of one multipath TCP solution, not of the general multipath concept.

III. Investigated Scenario

For the initial comparisons of the two above mentioned variants for multipath transport a direct connected topology was chosen for the local testbed, depicted in Figure 1.

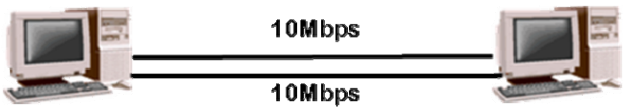

Figure 1: Testbed setup for direct connected nodes

IV. Results

Comparison of MPTCP and MCTCP performances for the direct connected scenario is depicted in Figure 2. The server rate for the depicted results is 10Mbps. For the test runs, 3 events are scheduled that introduces packet losses or additional delay over one or both the links. The packet losses are first introduced on both the links after 10s from the start of the test run, for duration of 10s (event1 – losses on both the links). After another 10s, packet losses are configured again for duration of 10s but this time only on one of the links (event2 – losses on link1). After another 10s, the link delay on one of the links is increased to 100ms from 10ms for duration of 10s (event3 – 100ms delay on link2). In the depicted results, 4 different cases are considered for both multipath protocol variants with respect to the introduced packet losses, i.e. 0, 2, 5 or 10% (refer Figure 2).

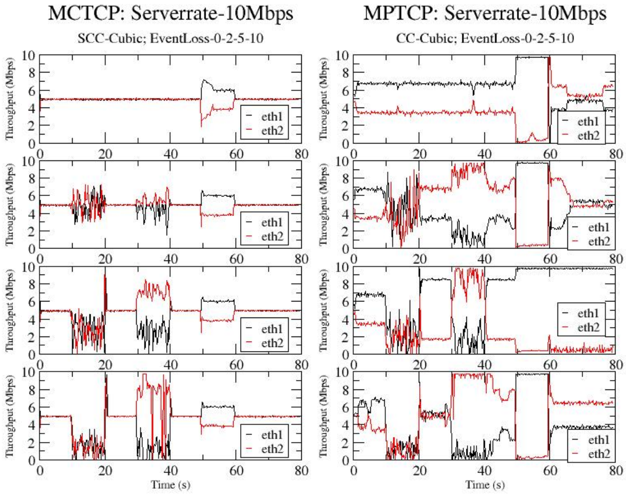

Figure 2: MPTCP and MCTCP comparison results for direct connected scenario

Further, tests were also done for higher server rates for the directly connected scenario. In addition, different scenarios with an extended network topology were also studied. Since, the local testbed can be limited due to the availability of the hardware; virtualization solutions offered within the German-Lab platform were also investigated in this work, e.g. Topology Management Tool [9].

V. Conclusion

There are different proposed multipath TCP solutions within the IETF MPTCP WG which span different design choices (option, payload and hybrid) for exchange of multipath signalling information. All the MPTCP solution variants claim to adhere to the requirement goals of the multipath TCP architecture, i.e. they increase the throughput, are more resilient and move congestion away from the congested paths while offering reliable, in-order transport being transparent to applications.

In this work, the performance of MPTCP and MCTCP solution variants were investigated with respect to different transmission parameters such as throughput and delay. Results obtained from a local testbed reveal that MCTCP can schedule the traffic equally on the available subflows, thereby reducing congestion on a single path. In addition, the MCTCP solution is robust to dynamic changes in the network such as variations in the packet loss or end-to-end delay. On the other hand, the MPTCP solution is also able to open multiple subflows whenever possible and hence increases the overall throughput or reduces the congestion on a single path. But, scheduling of traffic on the subflows is not equal even though the subflows experience same network characteristics, i..e, delay, packet loss and bandwidth.

The number of subflows which can be opened for a MPTCP flow and the criteria when to close an underperforming subflow needs to be further investigated. In addition, the fairness of multipath TCP solutions needs to investigated. With respect to multipath-aware applications, specific APIs to control the setup and operation of a multipath TCP connection need to be also developed. Both MPTCP and MCTCP implementation is a work in progress and hence evolving to provide a better solution for future communication demands.

VI. Acknowledgement

The authors are supported by the German-Lab project (http://www.german-lab.de/), a research project funded by the German Federal Ministry of Education and Research (BMBF).

VII. References

[1] A. Ford and C. Raiciu and S. Barre and J. Iyengar, Architectural Guidelines for Multipath TCP Development, RFC 6182 (work in progress), IETF, March 2011

[2] A. Ford and C. Raiciu and M. Handley, TCP Extensions for Multipath Operation with Multiple Addresses, draft-ietf-mptcp-multiaddressed-03 (work in progress), IETF, March 2011

[3] C. Raiciu and M. Handley and D. Wischik, Coupled Congestion Control for Multipath Transport Protocols, draft-ietf-mptcp-congestion-03 (work in progress), IETF, April 2011

[4] A. Singh, M. Scharf, PayLoad Multi-connection Transport using Multiple Addresses, draft-singh-mptcp-plmt-00 (work in progress), IETF, August 2010

[5] M. Scharf, Multi-Connection TCP (MCTCP) Transport, draft-scharf-mptcp-mctcp-01 (work in progress), IETF, July 2010

[6] M. Scharf, T.-R. Banniza, A. Singh, P. Schefczik and A. Timm-Giel, Evaluation and Prototyping of Multipath Protocol Mechanisms, 10th Würzburg Workshop on IP: Joint ITG, ITC, and Euro-NF Workshop "Visions of Future Generation Networks" (EuroView2010), 2010

[7] MPTCP Linux Implementation Downloads https://scm.info.ucl.ac.be/trac/mptcp/downloads

[8] Network Computing, http://www.german-lab.de/phase-2/netcomp/

[9] Topology Management Tool, http://tomato.german-lab.de/