MPTCP Performance over Heterogeneous Subpaths

Vivek Adarsh
Department of Computer Science
University of California, Santa Barbara
vivek@cs.ucsb.edu

Paul Schmitt
Department of Computer Science
Princeton University, NJ
pschmitt@cs.princeton.edu

Elizabeth Belding
Department of Computer Science
University of California, Santa Barbara
ebelding@cs.ucsb.edu

***Abstract*—Today's smartphones are equipped with both Wi-Fi and cellular interfaces, creating usage opportunities for protocols such as Multi-path TCP (MPTCP), which enable devices to use more than one interface concurrently. One of the biggest hurdles in implementing MPTCP is the heterogeneity in performance characteristics that exists across multiple interfaces. This makes the selection of primary interface of paramount importance, as this interface is also used for DNS resolution. In this work, we explore performance and IP reachability over real world networks. Our findings indicate that widespread MPTCP deployment faces significant obstacles. In particular, we perform controlled and real world experiments over multiple paths with differing loss rates and round trip latencies to assess the effect of primary path selection, and the range of issues that arise from selecting the under-performing path. Using results from our experiments, we show how heterogeneous paths can adversely affect MPTCP performance, especially when one path is lossy.**

I. Introduction

The challenge to meet 5G throughput goals has motivated today's mobile devices to use both Wi-Fi and cellular networks simultaneously. In order to take advantage of multiple available networks, researchers have created MPTCP, a transport protocol designed to achieve better throughput and resource utilization by enabling the simultaneous use of several IP-addresses/interfaces [1]. A unique challenge facing widespread MPTCP usage is the considerable performance differences commonly observed between cellular and Wi-Fi networks [2].

Prior work has explored the performance of MPTCP in a variety of contexts. While much of that work has focused on implementation and measurement of MPTCP performance, few studies have focused on the default MPTCP scheduler, especially from the perspective of heterogeneous subpaths. Today, path heterogeneity plagues the performance of various networks, such as Wi-Fi and cellular networks. In addition, cellular networks traditionally have significantly higher latency and loss compared to Wi-Fi [2]. As studied by [3], single path TCP performs better than MPTCP over HTTP/2, especially when page size is small or when network transfer is not the bottleneck. Furthermore, mobile users should not enable Multi-path TCP without thoughtful consideration, because it may lead to higher energy and cellular data consumption without providing significant quality of experience (QoE) improvements [3]. Our goal in this study is to understand MPTCP performance on paths with differing characteristics. In particular, we focus on paths of varying latency and loss rates, and assess MPTCP performance in these contexts. We seek to answer the following questions:

∗ *What is the impact of choosing a particular interface over another to act as the primary, particularly when DNS resolution is a concern?*
∗ *How does interface selection impact MPTCP availability and server reachability?*
∗ *How does MPTCP perform when paths have differing performance characteristics?*
∗ *When is single path TCP a better choice than MPTCP?*

We begin our study by measuring packet round-trip times (RTTs) for the Tranco top 10K websites [4] over different interfaces: Wi-Fi and LTE. RTT is a crucial parameter in network performance; the default MPTCP scheduler makes path selection decisions that rely heavily on the RTT of each path. Different interfaces can produce vastly different RTTs for the same web servers, in part due to server replication within CDNs. Next, server reachability is critical in assessing whether MPTCP can be implemented in a broad range of scenarios. Additionally, it helps us understand the importance of selecting the primary interface. Informed by our findings, we perform controlled in-lab experiments that vary network latency and loss conditions to examine the effect on metrics such as download times and page load time. Lastly, we replicate the in-lab test conditions to investigate MPTCP performance in the real world.

Our experiments indicate that selecting cellular as the primary interface can be detrimental, because of two reasons: a) RTT observed on the cellular path can be considerably higher than on Wi-Fi, and b) IP addresses resolved on the cellular interface have a lower chance of reachability than those resolved on Wi-Fi. Using empirical data, we show that the default MPTCP scheduler can under-perform when paths are disproportionately lossy. We compare MPTCP with single path TCP to study how small flows suffer more in comparison.

The rest of the paper is organized as follows. We provide a brief introduction to MPTCP and discuss related work in Section II. In Section III, we present our methodology and results from the real world survey. In Section IV we describe our controlled experiment testbed, methodology used and results. Real world test results are then presented in Section V. Finally, we conclude our analysis in Section VI.

II. Background

In this section, we present a brief overview of MPTCP and the default scheduler. We then discuss some related work to understand how our study explores a less-analyzed aspect of MPTCP performance.

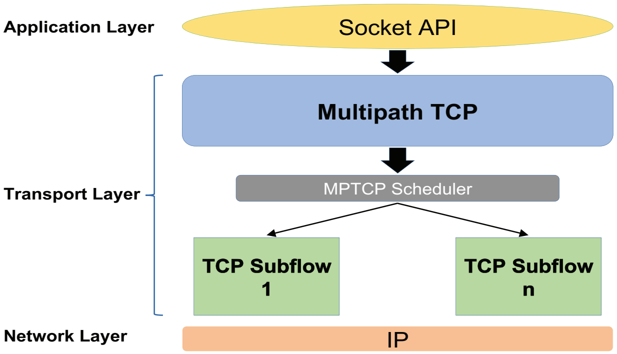


Fig. 1: Multi-path TCP architecture.

A. Multi-Path TCP Overview

MPTCP is a TCP extension that enables concurrent transmission of data from one end-to-end connection over multiple paths. For instance, on a smartphone, MPTCP allows applications to simultaneously send and receive data over multiple interfaces, such as Wi-Fi and cellular, by establishing one TCP subflow over each interface [1]. Once a subflow is established, it can be used by the MPTCP scheduler for transmission of data. MPTCP is designed to provide a variety of benefits, including better resource utilization, higher throughput and smoother reaction to failures, and it is likely to work better with paths that have comparable qualities. Figure 1 shows the architecture of Multi-path TCP.

When multiple subflows are available to send data, the default scheduler [5] will transmit the data on the subflow with the shortest round trip time. As soon as a segment is ready to be transmitted, the scheduler chooses the path with the minimum RTT, out of all subflows whose congestion window is not yet full. If there is more than one such path, then the scheduler develops a systematic inclination towards one of the interfaces and continues to transmit data on that particular subflow, until the subflow's congestion window becomes full. The interested reader can learn more about MPTCP in RFC 6824 [1].

B. Related Work

Previous work has examined MPTCP in a mobile context. For instance, [6] studied the impact of mobility on MPTCP, while [7] proposed different MPTCP modes to be used by mobile devices for cellular/Wi-Fi handover. However, neither work explored path heterogeneity with regard to lossy subflows. [8] presented a measurement study that compares single path TCP to MPTCP. Arzani et al. [9] studied the effect of the scheduler design on performance by using different scheduler algorithms, while others have compared various congestion control algorithms [10][11][12][13]. Little prior work studying the effect of selecting the primary resolution interface exists. Yang et al. [14] proposed an alternative scheduler that chooses subflows based on an estimation of how much more traffic they can handle before becoming congested. Their approach considers scenarios with very large transfers in a network with a small amount of buffering. Another scheduling algorithm was proposed in [15] to avoid out-of-order segments. However, the authors do not explain how to remove a segment from a TCP buffer once it is retransmitted from another subflow.

Kuhn et al. [16] proposed a delay-aware packet scheduler, which is evaluated only through ns-2 simulations. Their method examines path heterogeneity in stable CWND and delay conditions only. Closest to our work is [17], which measured MPTCP performance over cellular networks and Wi-Fi. This study focuses on varying numbers of subflows and detailed statistics, such as out-of-order delivery and round trip times, but does not take into account lossy subflows. Lim et al. proposed a scheduler that monitors the available bandwidth on each subflow and send buffers; however, it does not take advantage of the loss rate information on each individual subflow.

III. Latency and Reachability Survey

The MPTCP scheduler relies on round rip latencies for path selection. Hence, in a mobile context, it is imperative to understand the typical RTT differences between interfaces. Furthermore, MPTCP can be impossible to implement in situations where the server IP address is unreachable over one of the subpaths. We study these parameters in our real world survey to understand the obstacles MPTCP faces for widespread deployment.

A. Methodology

Since the default MPTCP scheduler relies heavily on round-trip times for path selection, our first goal is to study the RTT difference of each potential path to the path's corresponding web server. To do so we conduct a survey to explore latency and reachability for each of these interfaces to the Tranco top 10K websites [4]. Our testbed consists of a Samsung Galaxy S5 phone tethered to a Lenovo Thinkpad laptop via USB. We access the Wi-Fi network through the tethered phone instead of the built-in interface in the laptop to ensure we account for the performance overhead added through tethering and to maintain consistency across all the interfaces. We use the T-Mobile network for LTE services. The tests were performed after midnight so as to avoid high-usage times, and we ensured the signal strength on the devices was strong. Latencies were collected through Hping3 by averaging the RTT of 10 packets sent to each target IP address, through each interface.

B. Results

As a first step in our study, we perform DNS resolution of the Tranco top 10K websites [4] on each interface (Wi-Fi and LTE) using the tethered phone. Web servers can resolve to different IPs over different interfaces. This is because resolution depends on how the ISP routes the request so as to return the address of the desirable content delivery network (CDN). For instance, cellular operators are likely to embed web servers and CDNs within their core network in order to provide faster response times to user web requests. Omitting websites that do not resolve, the resulting sample

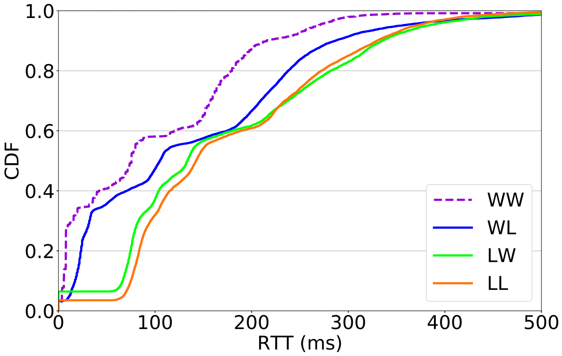


Fig. 2: CDF of Tranco top 10K web-servers' RTT, resolved using different interfaces.

size after DNS resolution is 9756 and 9638 for Wi-Fi and LTE, respectively, with an overlap of 58.74% in IP addresses. We then conduct RTT tests to the obtained web servers using Hping3. Hping3 uses TCP packets to ping the servers. There are four possible DNS resolution/latency combinations. They are:

1) ***WW***: Wi-Fi interface using address resolved on Wi-Fi DNS
2) ***WL***: Wi-Fi interface using address resolved on LTE DNS
3) ***LW***: LTE interface using address resolved on Wi-Fi DNS
4) ***LL***: LTE interface using address resolved on LTE DNS

Figure 2 shows the cumulative distribution function (CDF) for the average round-trip time for 10 TCP packets sent to each target server collected over each interface. We observe that the RTT deviation between *WW* and *WL* is around 40ms, while the mean RTT deviation is about 75ms between *WW* and websites accessed through LTE (*LL* and *LW*). It is interesting to note that in about 60% of websites, *LL* under-performs in comparison to *LW*. This result is quite surprising: web servers resolved over LTE, which are likely CDN servers within the cellular network infrastructure, incur larger delays. We speculate that this happens because servers become unresponsive soon after DNS resolution. Possible reasons for this behavior can be attributed to load balancing across several IP addresses associated with a web server and continual Denial of Service attacks [18][19]. In addition, LTE should be able to access the servers resolved on Wi-Fi. It is clear from figure 2 that the use of the web servers resolved over the cellular interface will likely yield larger delays, and hence adversely affect user experience with longer RTTs.

Table I shows the percentage of unresponsive servers during the latency test. We define unresponsive as those servers that never send a response back to our pings. We study reachability of servers to understand whether MPTCP is possible to use in all cases (assuming web servers were MPTCP enabled). If a server is unreachable over a particular interface, which MPTCP uses as one of its subflows, then MPTCP is unusable. In other words, it is no better than using single path TCP. From Table I we note that the Wi-Fi interface produces far fewer unresponsive servers than the LTE interface; more than 4.5% of the web servers resolved on LTE are unresponsive. We posit that this is due to unreachable servers deployed in the cellular core behind NAT and interference by middleboxes. On the other hand, servers resolved using Wi-Fi are more likely to be reachable via other interfaces. Note that more than 40% of the web servers either resolve differently or do not resolve at all depending on the choice of interface. This is noteworthy because there is a high probability that either of the interfaces (i.e., Wi-Fi or LTE) will be unable to establish an end-to-end path with a web server resolved over the other path. The takeaway from these observations is that in the presence of unreachable servers, the end-to-end path is not available. In other words, MPTCP will be impossible to use.

Ping Interface	Resolution Interface	Percentage
Wi-Fi	LTE	**4.57**%
Wi-Fi	Wi-Fi	3.26%
LTE	LTE	3.50%
LTE	Wi-Fi	3.59%

TABLE I: Percentage of unresponsive servers.

IV. Controlled in-lab experiments

As shown in Section III, a considerable disparity exists between the round-trip times to web servers through cellular and Wi-Fi interfaces. This disparity is likely to influence the path selection process and the resulting performance. In this section, we study the effect of varying latency and loss rates on performance metrics, as an indication of how the MPTCP scheduler will perform across heterogeneous links.

A. *Testbed Setup*

Figure 3 illustrates the testbed used in our second set of experiments, where we study the performance of MPTCP by manually setting loss and latency parameters in our controlled environment. The testbed consists of a wired server and client, both housed at our research facility. The client is connected to the server through two interfaces via switches, resulting in two paths between the devices. The client and server are both Lenovo ThinkCenter M910T machines (Ubuntu Linux 16.04 with MPTCP Kernel implementation version v0.92) configured with Intel Core i7-7700 processor (3.6GHz) and paired with 64GB of DDR4 RAM. Each machine comes with one integrated Intel Gigabit Ethernet interface. For the purpose of our experiment, we install an additional TP-Link Gigabit Ethernet card on both machines. We disable the wireless interfaces on the client and server. The switches are both

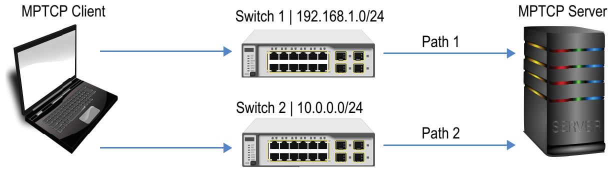


Fig. 3: Experimental setup.

Linksys WRT1200AC dual-band routers, running OpenWrt version 15.05.1.

B. Methodology

The motivation of our controlled experiments is to generate baseline results that can be used as a reference for the real world experiments in Section V. For in-lab testing, we configure our lab server as an HTTP server, running Apache2 on port 80. To evaluate page load time, we first cache the Tranco top 1000 websites on our local lab server. We then establish an MPTCP connection from the client to the server in our testbed, wherein the client fetches each of the cached websites. The experiment is repeated 100 times for each website and the average page load time is calculated. While this approach does not exactly translate to fetching live webpages, the results serve as a reference for the real world experiments. Hence, our results are an approximation of actual page load times. Note that our testbed does not reflect the true RTT between the client and server. Therefore, the controlled experiments consider a broad range of inter-path RTT difference.

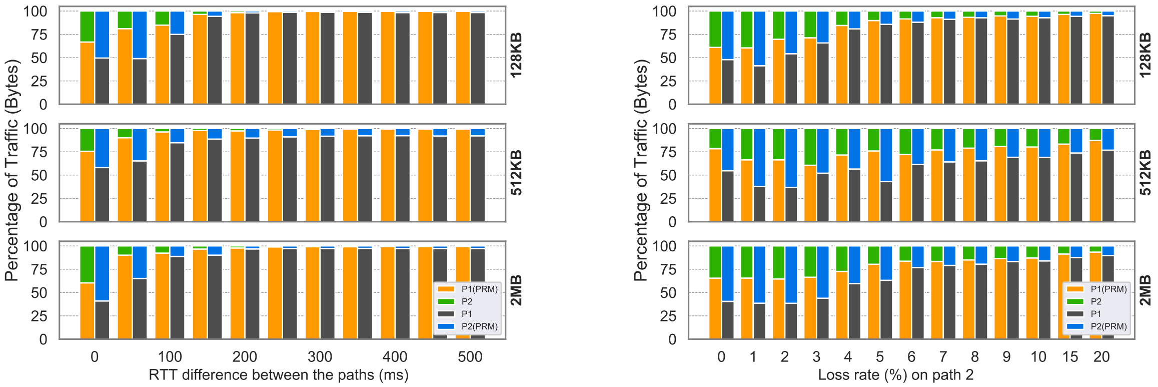
Next, we set up a 2-path MPTCP connection between the client and server in our testbed. The paths are set up on two different subnets. Our aim is to structure these paths so as to emulate two different interfaces on a device, e.g. Wi-Fi and LTE. To study the performance of web traffic over MPTCP, we choose various file sizes for measurement: 128KB, 256KB, 512KB, 1MB and 2MB. To study a variety of network conditions, we vary either the latency or the loss rate of path 2, keeping the other parameter constant. We use the Linux Traffic Control `tc` command for this purpose. During our survey in Section III-B, we found the median RTT to be around 20ms and loss rates consistently about 0.1% for Wi-Fi. Therefore, we initiate both the paths with 20ms round-trip time and 0.1% loss rate. To study the effect of differing path latency, we increase the RTT to 50ms and in each subsequent experiment, we increase it in increments of 50ms, to a maximum of 500ms on path 2. To study the effect of loss rate, we initialize both paths with no loss and 20ms of RTT. Thereafter, we increment the loss by 0.5% on path 2, to a maximum loss rate of 10%. We record measurements for 15% and 20% loss rates as well.

As our goal is to understand the effect on performance when a higher RTT path is chosen instead of the lower RTT path as the primary interface, we initialize these experiments by manually specifying the primary path at the beginning of the experiment. That is, we run two sets of experiments. First, path 1 is set as the primary path and second, path 2 is configured as the primary path. This approach is analogous to MPTCP choosing the primary interface for DNS resolution in a real world scenario. As we explained in Section III, the primary interface chosen for DNS resolution plays an important role in reachability of servers. Note that MPTCP can and will, in subsequent stages, choose which path to send traffic through, a decision dictated by the MPTCP scheduler. We measure goodput by running `iperf` [20] in client-server mode. In order to specify file transfer size, corresponding to our short and long flows, the *-n* flag in `iperf` is assigned accordingly. Each set of experiments is run 100 times for every file size. At the server side, we capture the traffic using `tcpdump` [21].

C. Evaluation

In this section we present the results of our study on increasing RTT and loss rate on the range of traffic sent on each path. Figures 4(a) and (b) show the percentage of traffic sent over the primary and secondary paths as latency and loss differ. In figure 4(a), we see that as the RTT increases on path 2, the traffic is directed almost exclusively on the lower RTT path 1. This is because the algorithm with which the default scheduler initiates path selection gives higher priority to paths with lower RTT [22]. However, in figure 4(b), we see that path 2 still carries traffic, even though the loss rate increases. For short flows like these, the subflows may never exit their slow-start phases, which explains why path 2 is still significantly used.

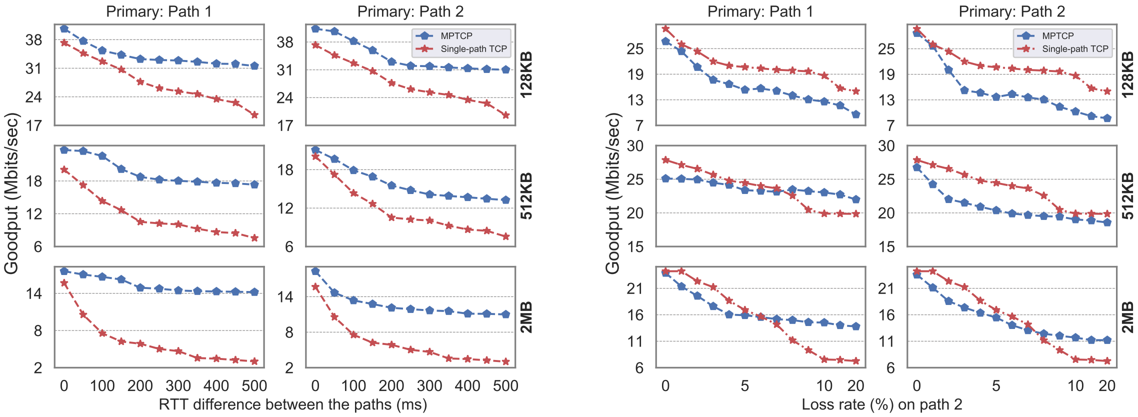
Figure 5(a) shows the achieved goodput for 128KB, 512KB and 2MB file sizes as the RTT on path 2 increases. Due to



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

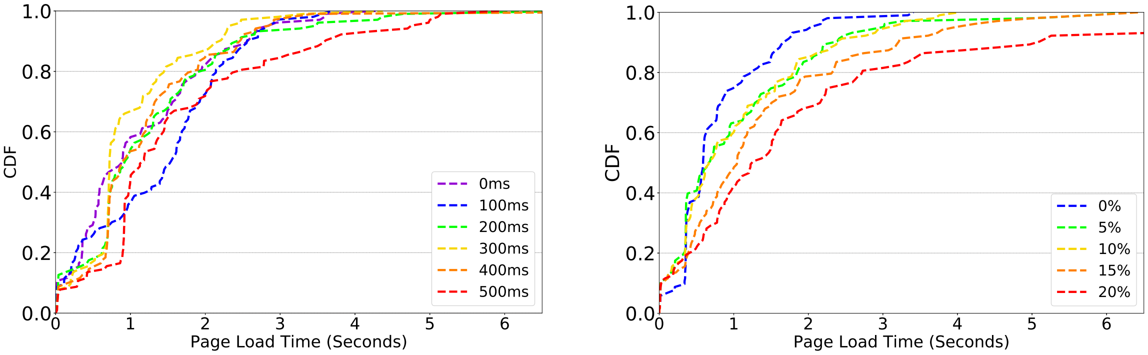
Fig. 4: Fraction of traffic carried on each path for controlled iPerf experiments.



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

Fig. 5: Achieved goodput for web downloads (higher is better).



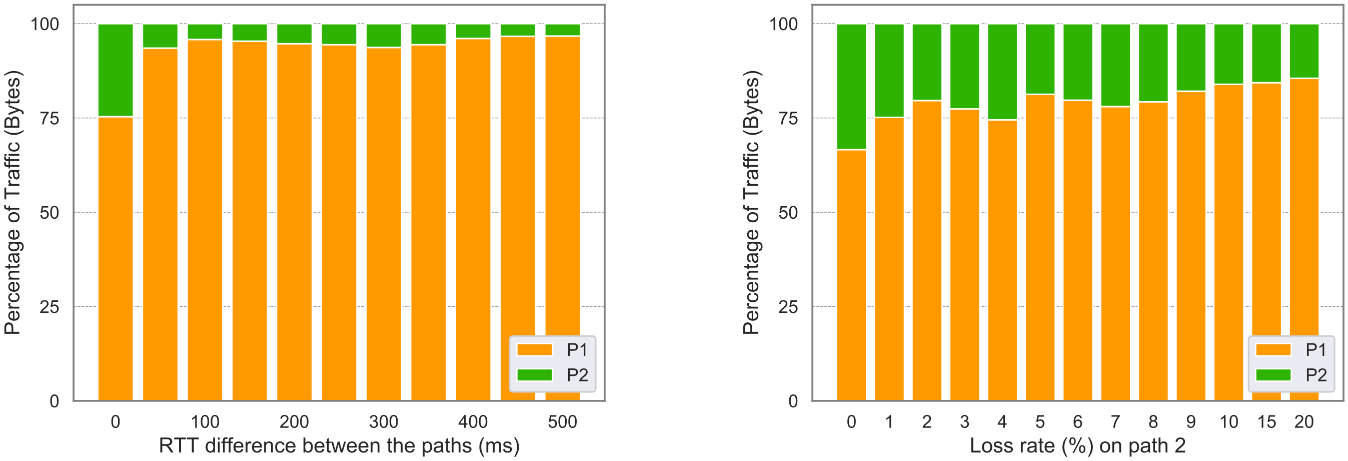
(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

Fig. 6: CDF of page load time (lower is better).

space restrictions, we have omitted flows sizes of 256KB and 1MB from these plots. We see a steady decrease in goodput until an RTT difference between 200ms and 250ms, after which the goodput flattens. A similar pattern is observed in file sizes of 512KB and 2MB as well. This behavior can be explained by figure 4(a) which indicates that, once the inter-path RTT exceeds 200ms, the bulk of traffic is carried on path 1 due to its lower RTT. It is clear from figure 5(a) that after 200ms, additional increases in RTT on path 2 cease to have any significant effect on the goodput. We observe that MPTCP performs better than single path TCP on all occasions, regardless of the choice of primary path. Next, figure 5(b) shows the achieved goodput with respect to increasing loss rate on path 2. When we compare MPTCP with single path TCP in figure 5(b), we observe that for the short flows of size 128KB and 512KB, single path TCP outperforms MPTCP. For flow sizes this small, subflows can still be in their slow-start phase when the download is complete. Additionally, for the 2MB file size, we observe that TCP performs as good as MPTCP, if not slightly better. This result tells us how important the initial path selection process is, particularly because a considerable fraction of web traffic is small flows [23].

Page load time directly affects user experience, and is represented in figure 6. Figure 6(a) indicates that the page load time does not necessarily increase as the RTT increases on path 2. We observe that as the RTT increases on path 2, the bulk of the traffic is carried on path 1, which is associated with lower latency. However, as we increase the loss rate on path 2 as shown in figure 6(b), page load time gradually increases. To better understand this result, we study the fraction of traffic carried on each path in figure 7(a). We notice that while the vast majority of traffic traverses path 1 as the RTT on path 2 increases, this does not hold true as the loss rate on path 2 increases. In figure 7(b), even though path 2 is more lossy, it still carries an appreciable amount of traffic, over 20% in most cases. As a result, page load times slowly degrade as we approach 20% loss rate on path 2. This is because TCP's estimation of a path's RTT is not affected by the packet loss on that particular path; the MPTCP scheduler ignores RTT for re-transmitted and lost packets. Stated otherwise, it does not take into account path loss. Consequently, this adversely impacts path selection, and hence performance.



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

Fig. 7: Fraction of web traffic carried on each path for controlled experiments.

V. Real world experiments

We next examine the performance of the default scheduler in a more realistic scenario. This enables us to gauge whether the conditions identified in Section IV actually occur in practical settings. Wherever possible, we replicate the test cases used in Section IV in order to establish a fair comparison.

A. *Testbed Setup*

For this set of experiments, we deploy two MPTCP enabled machines on a popular cloud service provider. Our testbed consists of a server, located in Virginia, and a client that is situated in California. We keep the system configuration similar to our controlled experiments, i.e., Ubuntu Linux 16.04 with MPTCP Kernel implementation version v0.92. The client communicates with the server over the Internet through two different wired interfaces, each connected to a different subnet to maintain isolation of routes. We observe an average round-trip time of 61ms with a standard deviation of ±3 ms between the client and server on both paths.

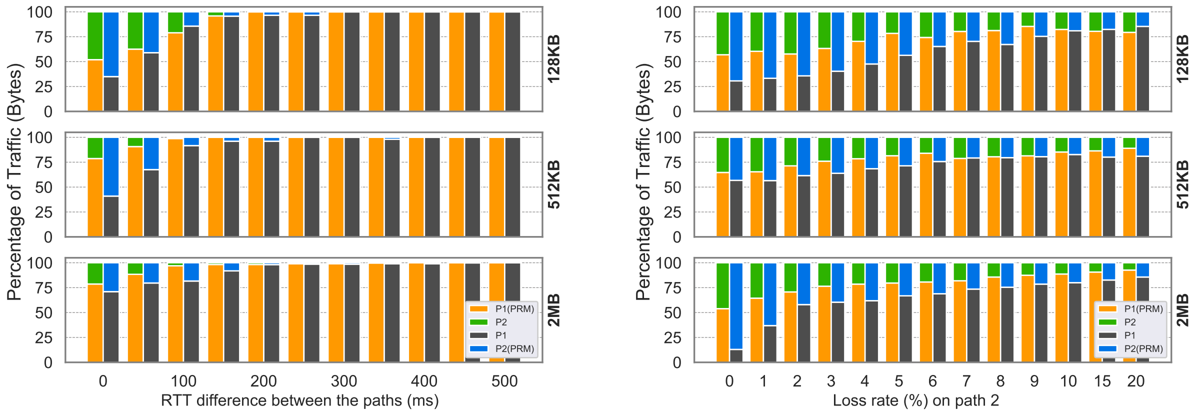
B. *Methodology*

In this section, we examine the performance of the default MPTCP scheduler for simple web downloads using `iPerf`. According to [24], about 58% of today's global Internet traffic is attributed to video streaming. Consequently, content streaming is typically preceded by (for instance, metadata files) and accompanied by (change in user preferences) small download sessions, that are characteristically short flows. This suggests that a bottleneck in such download sessions could result in poor quality of experience (QoE) at the user end. Therefore, we again explore the download performances of 128KB, 256KB, 512KB, 1MB and 2MB file sizes using `iPerf`. Similar to Section IV, we vary either the latency or loss rate on path 2, keeping the other parameter steady. Inter-path latency is varied from 50ms through 500ms, with increments of 50ms. Loss rates are varied from 1% through 10%, including 15% and 20%. Furthermore, we manually specify the primary path at the start of each experiment in order to study the effect on performance when a higher RTT or more lossy path is initially selected. To determine the fraction of Internet traffic carried over each subpath, we capture network traces at the server side using `tcpdump`.

Then, we investigate the distribution of the page load times for Tranco top 1K websites [4]. For evaluation purposes, we cache the top 1000 webpages on our server located in Virginia, along with their associated web objects. We then establish an MPTCP connection from the client to the server. We run the experiment 100 times for each webpage, and calculate the average page load time while replicating the path characteristics in Section IV, i.e., we vary the latency and loss rates on path 2.

C. *Evaluation*

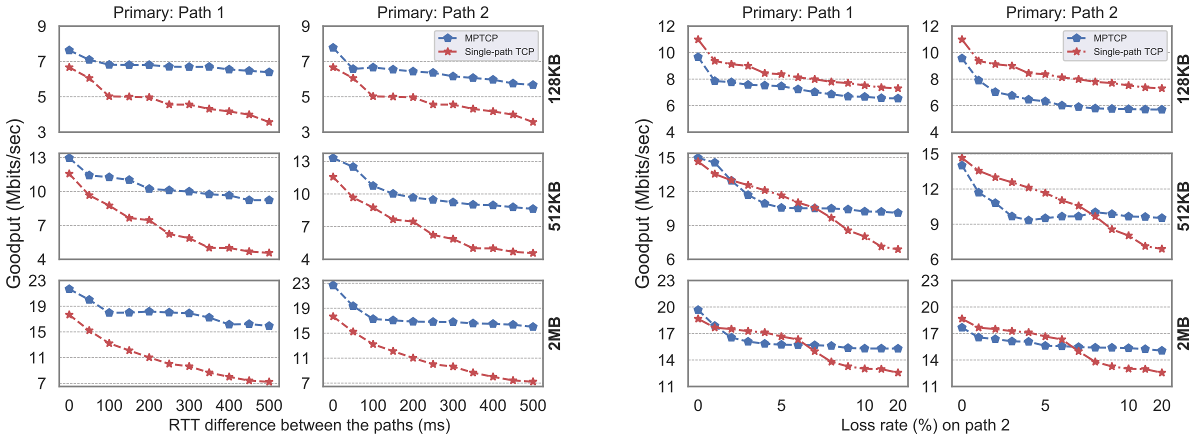
In this section, we present the results of our real world experiments. We first study the effect of varying path characteristics on web downloads for short flows. Figure 8 shows the percentage of traffic sent over the two subpaths as the path characteristics (latency and loss rate) change. We observe in figure 8(a) that traffic on path 2 becomes negligible as inter-path latency increases. An expected behavior, this can be attributed to lower RTT on path 1, which is the single greatest deciding factor in the default scheduler. On the other hand, while loss rates increase on path 2, the decision to select a viable path becomes less deterministic, as we hypothesized. We see a similar trend in figures 8(b) and 4(b), which indicates that subflows fail to exit their slow-start phase before the download is complete, resulting in considerable traffic over the lossier path. The implications of that behavior, however, become more noticeable in our study of goodput.



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

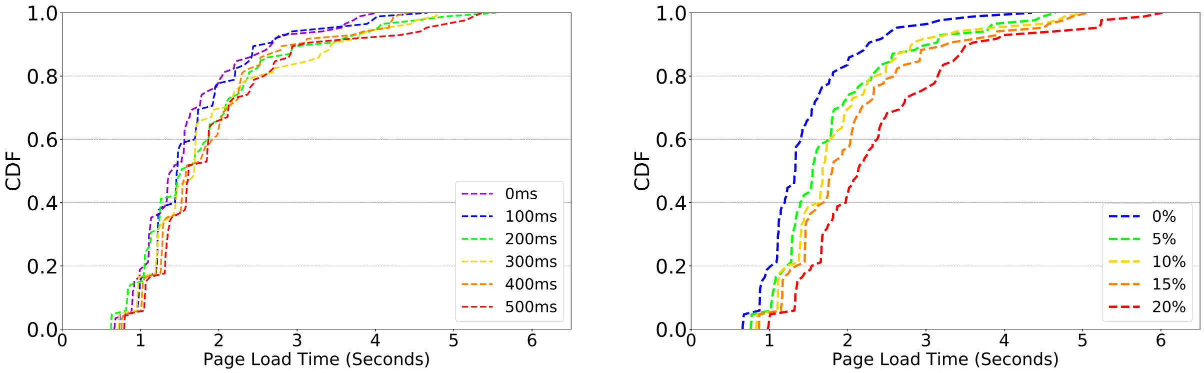
Fig. 8: Fraction of traffic carried on each path for real world iPerf experiments.



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

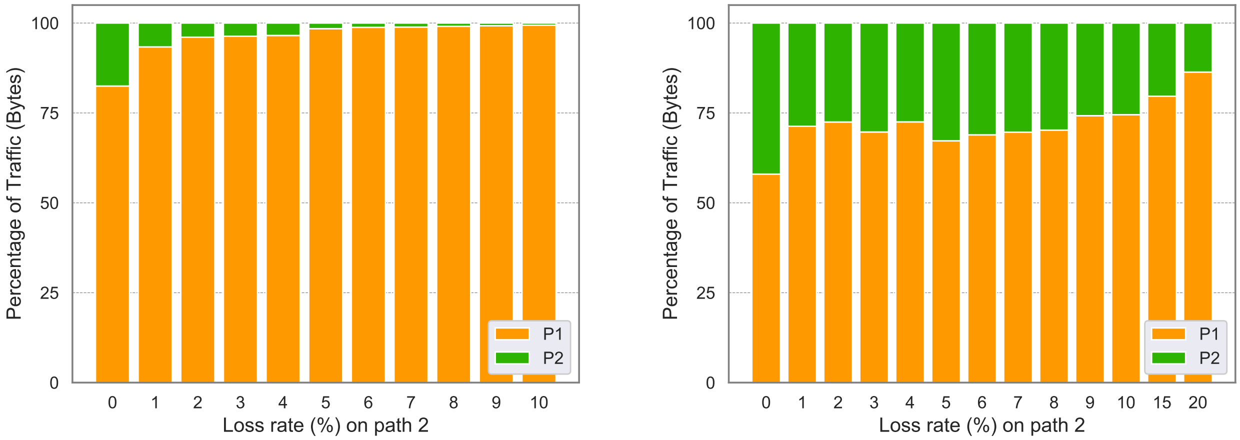
Fig. 9: Achieved goodput for web downloads (higher is better).



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

Fig. 10: CDF of page load time (lower is better).



(a) RTT on path 2 increases

(b) Loss rate on path 2 increases

Fig. 11: Fraction of web traffic carried on each path for real world experiments.

Figure 9 illustrates the goodput for 128KB, 512KB and 2MB file sizes as the path properties become more heterogeneous. In figure 9(a) we notice that the difference in goodput between MPTCP and single path TCP widens as the inter-path RTT increases. This is supported by figure 8(a), which demonstrates that after an inter-path latency difference of 150ms, traffic is almost exclusively carried on the lower latency path 1, enabling it to achieve better goodput than the slower single path TCP. In contrast, figure 9(b) shows that single path TCP indeed surpasses MPTCP with uneven lossy paths, which confirms our findings from the in-lab experiments (figure 5(b)). For instance, in the case of the 128KB file, MPTCP almost always performs worse than single path TCP. For 512KB and 2MB files, single path TCP achieves better goodput until 6% and 8% loss rates, respectively. This is a significant finding since it is less likely for a path to have loss rates as high as 6%–8% under normal operating conditions. Stated otherwise, single path TCP almost always performs better than MPTCP while downloading short flows. In practice, this could mean the difference between an instant video playback versus laggy, pause-filled video content. This result informs us of how crucial the initial path selection process is, as this can adversely affect the end user experience.

Another essential factor that affects user experience is the webpage load time. As shown in figure 10(a), the detrimental effect of increasing RTT on path 2 is subtly absorbed by MPTCP since path 1 carries the bulk of the traffic (figure 11(a)). Instead, page load time is adversely affected when loss rates are introduced on path 2. Given the path selection criteria for the default MPTCP scheduler, which only considers minimum RTT, it is not surprising to observe that more than 25% of the total traffic is carried on path 2, even though it suffers from a significant 10% packet loss. Correspondingly, there is a steady increase in page load time in figure 11(b) as path 2 becomes lossier. Since cellular networks tend to be more lossy than Wi-Fi [2], the path selection process becomes highly critical. Web surfing and video streaming are common activities for an average mobile user and could easily experience a decline in QoE. The results shown here indicate the limitations of the default scheduler, and the factors that need to be incorporated in order to improve mobile performance.

VI. Discussion and Conclusion

The results of our measurement study point to a few important findings:

Round-trip Times. Our results in Section III show that round-trip times on cellular are substantially higher than on Wi-Fi. In addition, majority of web servers resolved on cellular have lower round trip times on Wi-Fi than on cellular.

Reachability. The primary interface is responsible for DNS resolution in MPTCP. Resolving IPs over cellular can be detrimental since a large number of servers were unreachable through either Wi-Fi or cellular itself. MPTCP clearly cannot be used those scenarios. In order for MPTCP to be successful, CDNs deployed by commercial providers may need to be modified to be reachable from the outside (i.e., not behind NAT). However, this proposition is in contrast to how content is delivered on the modern Internet. These observations hint at the low viability of a full scale MPTCP deployment.

Heterogeneous Paths. Our measurement study shows that diversity in loss rates on paths is ignored when selecting the best path. This weakness is embedded inside the default scheduler because it considers only RTT as a metric for path selection. We realize that it is difficult to implement MPTCP with heterogeneous subpaths. From an institutional level, near-homogeneous network conditions are needed on all subpaths for MPTCP to utilize its full potential. For MPTCP to achieve optimal performance, it should take a broader view of path performance and, at a minimum, also consider loss rate.

References

[1] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," *RFC 6824*, 2013.

[2] J. Sommers and P. Barford, "Cell vs. Wi-Fi: On the Performance of Metro Area Mobile Connections," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*. ACM, 2012, pp. 301–314.

[3] B. Han, F. Qian, and L. Ji, "When Should We Surf the Mobile Web using both Wi-Fi and Cellular?" in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. ACM, 2016, pp. 7–12.

[4] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A Research-oriented Top Sites Ranking Hardened against Manipulation," in *Proceedings of the 26th Annual Network and Distributed System Security Symposium*. Internet Society, 2019.

[5] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental Evaluation of Multipath TCP Schedulers," in *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop*. ACM, 2014, pp. 27–32.

[6] C. Raiciu, D. Niculescu, M. Bagnulo, and M. J. Handley, "Opportunistic Mobility with Multipath TCP," in *Proceedings of the 6th International Workshop on MobiArch*. ACM, 2011, pp. 7–12.

[7] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring Mobile/Wi-Fi Handover with Multipath TCP," in *Proceedings of the 2012 ACM SIGCOMM workshop on Cellular Networks: Operations, Challenges, and Future Design*. ACM, 2012, pp. 31–36.

[8] S. Deng, R. Netravali, A. Sivaraman, and H. Balakrishnan, "Wi-Fi, LTE, or Both?: Measuring Multi-homed Wireless Internet Performance," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 181–194.

[9] B. Arzani, A. Gurney, S. Cheng, R. Guerin, and B. T. Loo, "Impact of Path Characteristics and Scheduling Policies on MPTCP Performance," in *28th International Conference on Advanced Information Networking and Applications Workshops (WAINA), 2014*. IEEE, 2014, pp. 743–748.

[10] R. Khalili, N. Gast, M. Popovic, U. Upadhyay, and J.-Y. Le Boudec, "MPTCP is Not Pareto-Optimal: Performance Issues and a Possible Solution," in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. ACM, 2012, pp. 1–12.

[11] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, Implementation and Evaluation of Congestion Control for Multipath TCP." in *NSDI*, vol. 11, 2011, pp. 8–8.

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

[13] K. Noda and Y. Ito, "Proposal of Novel MPTCP Congestion Control to Suppress QoS Fluctuation for WebQoE Improvement," in *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*. IEEE, 2018, pp. 1–3.

[14] F. Yang, P. Amer, and N. Ekiz, "A Scheduler for Multipath TCP," in *2013 22nd International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2013, pp. 1–7.

[15] F. Yang, Q. Wang, and P. D. Amer, "Out-of-Order Transmission for In-Order Arrival Scheduling for Multipath TCP," in *2014 28th International Conference on Advanced Information Networking and Applications Workshops*. IEEE, 2014, pp. 749–752.

[16] N. Kuhn, E. Lochin, A. Mifdaoui, G. Sarwar, O. Mehani, and R. Boreli, "DAPS: Intelligent delay-aware packet scheduling for multipath transport," in *2014 IEEE International Conference on Communications (ICC)*. IEEE, 2014, pp. 1222–1227.

[17] Y.-C. Chen, 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," in *Proceedings of the 2013 Conference on Internet Measurement Conference*. ACM, 2013, pp. 455–468.

[18] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-based Application Layer DoS Attacks on Web Servers in the Presence of Sampling," *Computer Networks*, vol. 121, pp. 25–36, 2017.

[19] C.-A. Staicu and M. Pradel, "Freezing the Web: A Study of Redos Vulnerabilities in Javascript-based Web Servers," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 361–376.

[20] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "iPerf," 2006.

[21] V. Jacobson, C. Leres, and S. McCanne, "Tcpdump," *Lawrence Berkeley Laboratory, Berkeley, CA*, vol. 143, 1989.

[22] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How Hard Can it be? Designing and Implementing a Deployable Multipath TCP," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 29–29.

[23] P. Jurkiewicz, G. Rzym, and P. Borylo, "Flow Length and Size Distributions in Campus Internet Traffic," *CoRR*, vol. abs/1809.03486, 2018. [Online]. Available: http://arxiv.org/abs/1809.03486

[24] FCC, "The Mobile Internet Phenomena Report 2019," https://www.sandvine.com/hubfs/downloads/phenomena/2019-mobile-phenomena-report.pdf, September 2019.