A Longitudinal View at the Adoption of Multipath TCP

Tanya Shreedhar, Danesh Zeynali, Oliver Gasser, Nitinder Mohan, and Jörg Ott

Abstract—Multipath TCP (MPTCP) extends traditional TCP to enable simultaneous use of multiple connection endpoints at the source and destination. MPTCP has been under active development since its standardization in 2013, and more recently in February 2020, MPTCP was upstreamed to the Linux kernel. In this paper, we provide an in-depth analysis of MPTCPv0 in the Internet and the first analysis of MPTCPv1 to date. We probe the entire IPv4 address space and an IPv6 hitlist to detect MPTCP-enabled systems operational on port 80 and 443. Our scans reveal a steady increase in MPTCPv0-capable IPs, reaching 13k+ on IPv4 (2× increase in one year) and 1k on IPv6 (40× increase). MPTCPv1 deployment is comparatively low with ≈ 100 supporting hosts in IPv4 and IPv6, most of which belong to Apple. We also discover a substantial share of seemingly MPTCP-capable hosts, an artifact of middleboxes mirroring TCP options. We conduct targeted HTTP(S) measurements towards select hosts and find that middleboxes can aggressively impact the perceived quality of applications utilizing MPTCP. Finally, we analyze two complementary traffic traces from CAIDA and MAWI to shed light on the real-world usage of MPTCP. We find that while MPTCP usage has increased by a factor of 20 over the past few years, its traffic share is still quite low.

I. INTRODUCTION

Despite significant advances in Internet infrastructure and connectivity, TCP's connectivity model has remained largely unchanged over the last 30 years. Recent advances in network technologies have led to the rise of multi-homed devices, e.g., smartphones, with access to more than one networking interface. Multipath TCP (MPTCP) is an extension to TCP that allows endpoints to simultaneously utilize multiple interfaces for concurrent or backup data transmissions [1]. Standardized in early 2013, MPTCP has shown better resource utilization, higher aggregated throughput, and resilience to network failures in numerous research studies published over the years [2]–[6].

Due to the performance benefits of MPTCP compared to TCP, several known organizations have incorporated the protocol within their products and services. Apple uses MPTCP in its iOS devices to enhance the user experience surrounding its system services, e.g., Siri, Music, Maps, Wi-Fi Assist [7]. In 2019, Apple provided APIs to third-party developers for making use of MPTCP in non-system iOS applications. Korea Telecom, in partnership with Samsung, uses MPTCP to provide Gigabit speeds over Wi-Fi and LTE to its customers [8]. In February 2020, MPTCPv1 was upstreamed to Linux and is now available to all users running Linux 5.6 or newer [9].

Despite significant interest in improving the protocol [1], [2], [10] , the current state of MPTCP deployment in the Internet remains largely unexplored in research. We attribute this gap partially to the influence of middleboxes on the

accuracy of such studies. The Internet is proliferated with a wide spectrum of specialized appliances and systems known as middleboxes that meddle with user traffic before it reaches the target [11]. The intended operation of middleboxes is to offer valuable benefits, e.g., firewalls drop unintentional packets and proxies improve the performance of connection setup. However, certain middleboxes interact quite poorly with connections containing TCP header extensions. While some may strip the packet of any header additions before relaying it to the next hop, others might block the connection altogether [12]. Since MPTCP relies on TCP extensions for signaling, it is also susceptible to such middleboxes in the Internet. MPTCP designers incorporate several mechanisms into the protocol specification that allows the protocol to fall back to regular TCP for data transfers, if the connection is affected by middleboxes [13]. Despite that, middleboxes continue to hinder MPTCP studies, since scanning tools leverage the connection establishment mechanism to interact with targets and thus remain vulnerable to side-effects of middleboxes. In a study from 2015, the authors wanted to analyze the deployment of MPTCP in the Internet, it later became clear that the results include false-positives due to middleboxes echoing MPTCP options for non-MPTCP hosts [14]. However, despite significant measurement challenges, assessing the adoption of MPTCP in-the-wild is still pertinent since the protocol can only be employed if there is sufficient server-side support in the Internet.

This paper extends our previous study [15] and presents a broad and multi-faceted assessment of MPTCPv0 and MPTCPv1. We study both the *infrastructure*, in terms of MPTCP-capable IPv4 and IPv6 addresses, and the *traffic share* at two geographically diverse vantage points. We identify and remove middleboxes affecting MPTCP operation in-the-wild from our scans to avoid false-positives. Furthermore, we also investigate if such middleboxes also negatively impact MPTCP application traffic. Specifically, we make the following contributions in our paper.

I. We regularly probe the entire IPv4 address space and an IPv6 hitlist [16] for MPTCPv0 support since July 2020 using ZMap. Our scans target HTTP (port 80) and HTTPS (port 443) since they make up the largest traffic share in the Internet [17], [18]. We find that our scans are affected mainly by middleboxes that echo TCP extensions, indicating that traditional scanning methods are still ineffective in accurately evaluating the deployment of MPTCP. We also observe that the number of IPs reported to support MPTCP *without* replayed options increased fourfold over IPv4 port 443 compared to 2015 [14].

II. We present the first analysis of the MPTCPv1 ecosystem to date and regularly perform measurements since March 2021 on ports TCP/80 and TCP/443. Its support on both IPv4 and IPv6 remains almost non-existent before October 2021 and remains low compared to MPTCPv0 afterwards. In a case-study, we extend our measurement period to February 2022, and we find that Apple has added MPTCPv1 support to some of its hosts and is now dominating its deployment in IPv4 and IPv6.

III. We scrutinize targets that reportedly support MPTCP in our ZMap scans for middleboxes by using Tracebox [19] and make two significant discoveries. First, some MPTCP-capable hosts in the IPv4 are *transient*; indicating experimental connotations attached to MPTCP usage in-the-wild. Second, we identify several middleboxes that interact in a much more complicated fashion than just echoing with MPTCP packets. Despite that, we observe a growing adoption of MPTCP reaching 16.5k/13.5k and 1195/1184 over port 80/443 on IPv4 and IPv6, respectively. Compared to December 2020, MPTCP support has grown substantially: 2× in IPv4 and 40× in IPv6.

IV. We initiate parallel HTTP(S) GET requests using MPTCP and regular TCP towards IPs identified in our ZMap and Tracebox measurements. Our results show that a majority of *truly* MPTCP-capable servers are indifferent to the choice of a transport protocol for connection establishment. However, IP addresses affected by middleboxes take longer to successfully establish a connection using MPTCP vs. TCP, hinting at the potential impact of middleboxes on MPTCP's perceived quality.

V. We analyze the usage of MPTCP for data transfers over the Internet by investigating four years of inter-domain traffic collected by CAIDA and MAWI. Our findings show that MPTCP data usage is still quite low compared to TCP (peaking at 0.4%), primarily due to the lack of widespread MPTCP support among clients, servers, and applications. We find that Apple—a vocal supporter of MPTCP—is responsible for almost all MPTCP traffic that we observe. Moreover, in early 2022 we can find the first data being exchanged over MPTCPv1 in-the-wild. Finally, we observe a steadily rising popularity of MPTCP for large data transfers.

To foster reproducibility, we plan to publish our latest datasets and scripts similar to the previous iteration of this work [20]. We also continuously perform MPTCPv0 and MPTCPv1 scans and publish the results at https://mptcp.io.

II. MULTIPATH TCP: AN OVERVIEW

Multipath TCP (MPTCP), standardized by the IETF in 2013, is a multipath extension to TCP. It utilizes multiple interfaces of a given device to create multiple subflows and transmits data using these multiple subflows concurrently. This leads to better resource utilization, higher aggregated throughput, and resilience to network failures [1], [13]. The subflows can be added and removed throughout the MPTCP lifecycle. However, both end-hosts must be MPTCP enabled to

Fig. 1: MPTCPv0 and MPTCPv1 connection establishment process between two MPTCP-capable endpoints [13], [23]. The `MP_CAPABLE` option is slightly different for the both versions. While MPTCPv0 SYN includes Bob's sender's key (shown in green dashed box), MPTCPv1 SYN is sent without the key and Bob's key is sent in the final ACK.

support multipath transfer. Data from the unmodified applications are scheduled on one of the underlying TCP subflows by the MPTCP scheduler block based on the scheduling policy [10]. The default `minSRTT` scheduler [21] sends data on the available subflow with the lowest smoothed round trip time (SRTT) to the receiver. MPTCP's *coupled congestion control* mechanisms balance packet congestion over all the subflows and minimize additional reordering delays [22].

We focus here on the MPTCP connection establishment procedure as our measurement approach utilizes it inherently. We refer inclined readers to previous work [1], [10] for more details on MPTCP machinery, features, and design choices. Figure 1 shows the MPTCPv0 and MPTCPv1 connection establishment process between MPTCP-enabled client and server. The MPTCP handshake mechanism is derived from the TCP three-way handshake. In addition, MPTCP hosts use a random 64-bit sequence as *keys* to authenticate themselves when setting up new subflows [13], [23]. Moreover, every packet in the handshake signals MPTCP support through the `MP_CAPABLE` option. The `MP_CAPABLE` option also includes the version supported by the host. The connection establishment process differs for MPTCPv0 and MPTCPv1. For MPTCPv0, the client (in our example Bob) initiates a connection by sending a SYN packet containing its key (highlighted by green) and the `MP_CAPABLE` option to the server (Alice). For MPTCPv1, Bob's sender's key is not sent in the SYN packet. In case the host supports multiple versions, the highest version number is sent in the `MP_CAPABLE` option. If the server also supports MPTCP, it replies back with a SYN-ACK including the `MP_CAPABLE` option and its own key. The receiver's `MP_CAPABLE` option will indicate the version number it supports. According to the specification [13], [23], both key values in SYN and SYN-ACK are individually

referred to as Bob's and Alice's sender's key. In the first stage of our study, we use `MP_CAPABLE` SYN packets to probe for hosts which reply with an `MP_CAPABLE` option in the SYN-ACK; recording their IP address and sender's key value (for more details see §IV-A). For both MPTCPv0 and MPTCPv1, Bob finally establishes the connection by sending an ACK with both keys and the `MP_CAPABLE` option. This allows regular MPTCP data transmissions between the two parties.

III. RELATED WORK

Scanning the activity of different protocols in the Internet has been a long-lasting interest within the network measurement research community [24], [25]. In early 2008, Heidemann *et al.* [26] systematically probed a subset of 1% of IPv4 address space with ICMP pings. The state of active scanning research was pushed forward significantly by ZMap [27], which allows researchers to scan the entire IPv4 address space in less than an hour. Several works have since used the tool to investigate the deployment of different protocols and applications in the Internet, e.g., liveness [28], TCP initial window [29], and QUIC [30]. Others have looked into passive data traces for a different viewpoint on deployment measurements. Richter *et al.* [31] studied the IPv4 activity as observed from within the Akamai network and show that comparative active scanning studies miss up to 40% of the hosts that contact the CDN. Qian *et al.* [32] analyzed TCP behavior from multiple vantage points within a large tier-1 ISP. Maghsoudlou *et al.* [33] used active and passive scanning to analyze and characterize the traffic from port 0. Wan *et al.* [34] discussed several factors that can impact the quality of scanning results e.g., geo-location, losses, blocking, etc. We circumvent these biases to the best of our abilities by following best practices and scanning continuously for six months. Please refer to §IV-A for our detailed measurement methodology.

Transport Protocol Deployment in the Internet. There is a substantial body of work that has looked into the deployment of transport layer protocols over the Internet, the majority of which has been highly focused on QUIC [35]. QUIC is a recently standardized transport protocol that has seen widespread deployment [36]. Rüth *et al.* [30] performed ZMap scans over IPv4 to check for QUIC support in the wild. They further built a scanning tool using `quic-go`[1] to investigate QUIC. The results indicate that QUIC's traffic over the Internet is continuously increasing and upwards of $\approx 8\%$ with $\approx 98\%$ of that traffic going to/from the Google AS [30]. A more recent study [37] checks for the deployment of different QUIC versions by developing ZMap modules for both IPv4 and IPv6. They also developed `QScanner`, a scanning tool to analyze the various deployments in detail. Madariaga *et al.* [38] analyze the adoption of QUIC from measurements in user-space taken by mobile end-user devices. The authors developed an Android framework to perform network flow measurements through passive monitoring of active connections. There are works that have measured other important aspects of the

protocol, such as DNS over QUIC [39] and web censorship measurements of HTTP/3 over QUIC [40].

The closest work to ours dates back to 2015 [14]. Mehani *et al.* proposed a scanning mechanism that probed every host on port 80 of the Alexa Top 1M with ZMap and classified IP addresses that responded with `MP_CAPABLE` as supporting MPTCP. Their results indicate that less than 0.1% of scanned targets support MPTCP, with a majority located in China. However, the accuracy of the work was later found to be low as it falsely recognized middleboxes that echoed unknown TCP extensions as MPTCP hosts [41]. The authors later published an errata and tracked the non middlebox-affected MPTCP deployment for several months in 2015. In a previous iteration of this work [15], the authors extend the above methodology to identify middleboxes affecting MPTCP correctly, hence providing the most accurate picture of *true* MPTCPv0 deployment. The authors observed a steady growth in MPTCP-enabled IPs that support HTTP and HTTPS, primarily driven by Apple. This work provides the first long-term multi-faceted analysis of MPTCP v0 and MPTCP v1 deployment in the Internet. Additionally, we identify the *true* support for MPTCP over the two most popular services in the Internet, HTTP and HTTPS, over both IPv4 and IPv6.

IV. ACTIVE INTERNET SCANS

To identify support for Multipath TCP in the Internet, we actively scan for MPTCP options (both *version 0* and *version 1*) over the IPv4 and IPv6 address space. Our study probes the entire IPv4 address space over port 80 (≈ 74M unique responsive IPs) and port 443 (≈ 52M unique responsive IPs). In IPv6, we use the *IPv6 hitlist* [16] to probe both port 80 and port 443 due to the size of the address space. We find 746k and 544k responsive IPv4 and IPv6 addresses, respectively. Our results in this work extend our previous measurement study [15] and are drawn from over 18 months of data collection (July 2020 – December 2021). We share our dataset and results on the website: https://mptcp.io

A. Methodology

We use ZMap [42] to rapidly enumerate IPv4 and IPv6 addresses. To identify MPTCP hosts (both MPTCPv0 and MPTCPv1), we leverage the initial handshake mechanism. However, since the MPTCP options sent during the connection establishment procedure differ for both versions (as discussed in §III), we perform separate periodic scans for each version. As shown in Figure 1, in MPTCPv0, we send a SYN with the `MP_CAPABLE` option for version number 0 along with a static sender's key. On the other hand, for MPTCPv1, we send the SYN with the `MP_CAPABLE` option for version number 1 only (i.e., we do not send a static sender's key in our MPTCPv1 scans). We record the SYN-ACK responses for scans of both versions. As illustrated in Figure 1, a legitimate MPTCP host (both versions) will reply back to an MPTCP SYN with a SYN-ACK containing the `MP_CAPABLE` option, version number and its own sender's key. If the target's SYN-ACK response includes these values, we classify it as *potentially MPTCP-capable* hosts. Previous research has shown that the

---

[1]https://github.com/lucas-clemente/quic-go

(a) IPv4 addresses in our scans from July 2020 – December 2021.

(b) IPv6 addresses in our scans from August 2020 – December 2021.

Fig. 2: Unique (a) IPv4 and (b) IPv6 addresses in our ZMap scans for MPTCPv0 over port 80 and port 443 over 18-months that returned a different MPTCP key than the one sent in SYN. The "consistent" counts the common hosts that are consistently available across three month moving average window.

accuracy of identifying MPTCP hosts via ZMap can be low due to middleboxes that replay or strip packets with TCP extensions [12], [14], [41]. As shown in previous work [15], such middleboxes particularly affect MPTCPv0 as the sender's key value is repeated back to the host—resulting in a large number of false positives. We discuss and highlight this issue further in §IV-C. Such ill-interaction between middleboxes and MPTCPv0 operation primarily motivated MPTCP protocol designers to remove the sender's key from SYN during connection establishment [13]. To further improve the correctness of our analysis for MPTCPv0, we probe *potentially MPTCP-capable* hosts with the well-known middlebox-detection tool Tracebox [19]. Tracebox allows us to detect the presence of MPTCP options modifications on the path, revealing IPs that *truly* support MPTCP. For our MPTCPv1 scans, we discard SYN-ACK responses that simply mirror our MPTCP option sent in the SYN. For both MPTCPv0 and MPTCPv1, we filter out responses that do not include the expected version number in MPTCP options.

Before conducting active measurements, we incorporate proposals by Partridge and Allman [43] and Dittrich *et al.* [44]. We follow best scanning practices [27] by limiting our probing rate, maintaining a blocklist, and using dedicated servers with informing rDNS names, websites, and abuse contacts. Furthermore, we diligently complied to any emails from organizations asking for their networks to be blocklisted.

B. Finding MPTCP Support In-The-Wild

1) ZMap Scan Results: Our scans targeted almost 60M responsive addresses on port 80 and 50M on port 443 in IPv4. Of these, about 300k addresses for MPTCPv0 and 200k addresses for MPTCPv1 responded with the MP_CAPABLE flag in their SYN-ACK (*potential MPTCP*) – making $\approx$5.0% and $\approx$6.0% of total responsive hosts on port 80 and 443 for MPTCPv0 and $\approx$3.3% and $\approx$4.6% of total responsive hosts on port 80 and 443 for MPTCPv1, respectively.

MPTCPv0: Figure 2a provides a month-wise distribution of our scanning results of MPTCPv0 over the IPv4 address space over port 80 and port 443 for our 18 months scan period from July 2020 to December 2021. Note that we were unable to perform any IPv4 port 443 scans in September 2020 due to infrastructural reasons and hence we omit the port 80 results for that month. Figure 2b provides a month-wise distribution of our scanning results of MPTCPv0 over the IPv6 address space over port 80 and port 443 for our 18 months scan period from August 2020 to December 2021. The bars indicate the unique IP addresses that return a different MPTCP key than the one sent in SYN. The "consistent" numbers represent the hosts that are consistently available across the past three-month moving average window. We show the "consistent" hosts to highlight the stability of MPTCP support over a three-month interval and to isolate any possible effects that we might observe due to transient hosts.

(a) IPv4 addresses in our scans from March – December 2021

(b) IPv6 addresses in our scans from April – December 2021

Fig. 3: Unique (a) IPv4 and (b) IPv6 addresses in our ZMap scans for MPTCPv1 port 80/443 that returned MPTCP key in response. The "consistent" counts the common hosts that are consistently available across three month moving average window.

At first glance, it might seem that MPTCPv0 support is increasing consistently during our measurement period, with the highest number of supported addresses reaching ≈25k and ≈121k on port 80/443 in October 2021, respectively. Moreover, the support for MPTCPv0 on port 443 seems much higher than on port 80 in both IPv4 and IPv6. However, we also observe that a large percentage of *potential MPTCP* hosts are inconsistently active across our scanning period, signaling at the existence of transient hosts. For IPv4, the consistently available IPs hover around ≈50–80% and ≈25–50% on port 80 and port 443, respectively. On the other hand, compared to IPv4, we find a very small number of IPv6 addresses responding with the `MP_CAPABLE` option. However, the support has increased from our previous observations [15] and remains stable over time since early 2021.

MPTCPv1: Figures 3a and 3b show the unique IPv4 and IPv6 addresses in our ZMap scans for MPTCPv1 for port 80/443 from April – December 2021 that returned a non-mirrored MPTCPv1 response. To assert MPTCPv1 support, we check the MPTCP version number and filter out all hosts that replay the same MPTCP options as we sent in the SYN packet. Similar to the MPTCPv0 results in Figure 2, "consistent" count the common IPs that are consistently available across a three-month moving window.

We observe a large difference in the number of hosts that support MPTCP between v0 and v1. Out of 200K hosts that respond to our scans, MPTCPv1 hosts in both IPv4 and IPv6 hovered close to 100. In fact, we found that MPTCPv1 support was almost non-existent until October 2021, after which we observe a large spike in both IPv4 and IPv6. On closer analysis, we find Apple to be the primary contributor to this support (we explore this trend further in §VI-A). For months prior to October 2021, MPTCPv1 had minimal support in the Internet, which came mostly from experimental networks. Also, note that the percentage of consistent responsive hosts is much larger in MPTCPv1 compared to MPTCPv0 for both IPv4 and IPv6 – hovering around 80–90%. Finally, we find that the magnitude of responsive MPTCPv1 IPv4 and IPv6 hosts is quite similar (≈ 120 in IPv4 vs. ≈ 80 in IPv6), which is in stark contrast to MPCTPv0, where the chasm between

(a) IPv4 Port 443   (b) IPv6 Port 443

Fig. 4: Hamming weight distribution of sender's keys received from potential MPTCPv0-capable hosts in our ZMap scans.

the two is much larger (≈ 100k vs. ≈ 2k).

2) Impact of middleboxes on correctness of scans: To examine if our scans are affected by interfering middleboxes, we analyze the MPTCP sender's key we receive from targets in their SYN-ACK response. A *true* MPTCP host generates a random 64-bit sequence to use as the key (see §III). According to the central limit theorem, the sum of independent random variables tends toward a normal distribution. In that case, the sum of all bits in the sender's key—i.e., the Hamming weight—should follow the normal distribution $\mathcal{N}(32, 16)$. Figure 4 shows the Hamming weight distribution of the sender's key from potential MPTCPv0 hosts on port 443 in IPv4 and IPv6. We find that a large number of sender's keys do not follow the normal distribution. In fact, the Hamming weight 16, i.e., the exact Hamming weight of the key that we send in our SYN probes, is heavily over-represented. This indicates a prevalence of middleboxes that mirror MPTCP options sent in SYN in our ZMap scans. On port 443, the phenomenon is much more prominent in IPv4, where almost 80% of the sender's keys are mirrored compared to ≈8% of keys on IPv6. On port 80 (not shown), we find that middlebox interference is even more elevated, with almost 90% and 30% of received sender's keys identified as being mirrored for IPv4 and IPv6, respectively. Note that MPTCPv1 hosts in our scans are unlikely to be affected by mirroring middleboxes since the sender's key is

Fig. 5: Unique IPv4 addresses that returned MPTCPv0 options in our ZMap scans over port 80 and 443 categorized by the sender's key.

no longer sent in the SYN for this protocol version (refer to §IV-A). On the other hand, for MPTCPv0, since we receive a legitimate sender's key along with an `MP_CAPABLE` option in the SYN-ACK response (as per protocol's expectation), such responses can result in false-positives in our resulting analysis. Furthermore, for our MPTCPv1 scans, we filter out responses that simply replay the MPTCP options sent in the SYN and do not explicitly report support for version 1 in SYN-ACK.

We now analyze the share of hosts that are affected by middleboxes mirroring MPTCPv0 options in our measurements. Figure 5 shows the aggregate number of unique potential MPTCP targets scanned over IPv4 for which the sender's key was mirrored (in orange) and different from ours (in blue) for ports 80 and 443. It is evident that middleboxes affect ZMap scans significantly as many hosts on both ports have mirrored keys. Interestingly, we find that the presence of middleboxes is far greater on port 80 than on port 443, as port 80 has 96% of hosts with mirrored keys compared to 81% on port 443. In contrast, 6484 and 42294 hosts send back different sender's keys on ports 80 and 443, respectively. For IPv6, we received different sender's key responses from 31 IP addresses on port 80 and 157 on port 443 (not shown).

The result is quite intriguing as it hints at HTTPS having far more support for MPTCPv0 than HTTP over both IPv4 and IPv6. We also investigate whether any hosts that are middlebox-affected on port 80 are MPTCP-capable on port 443, but we find no such intersection. This leads us to believe the following contrasting possibilities. First, HTTPS traffic is end-to-end encrypted at the application layer; it is possible that a large number of middleboxes do not modify the transport layer options of user traffic. This results in a much smaller percentage of hosts that are affected by middleboxes that inject replayed TCP extensions. Second, the result may still include non-MPTCP end-hosts, which are affected by middleboxes that also modify the sender's key value of SYN-ACK packets.

Takeaway — Our ZMap scans reveal significant support for MPTCP over IPv4 compared to IPv6. MPTCPv1 support on both IPv4 and IPv6 was almost non-existent before October 2021 and remains to be minimal compared to MPTCPv0. However, ZMap includes a large share of seemingly MPTCP-capable addresses, where middleboxes are mirroring TCP options, which significantly impacts the accuracy of MPTCPv0 scans. For IP addresses that responded with different options, we find potential MPTCP support to lean more towards HTTPS than HTTP for both IPv4 and IPv6.

C. Finding Interfering Middleboxes

While our ZMap analysis in the previous section filters out hosts that are affected by middleboxes simply mirroring TCP options, it still does not entirely capture the true state of MPTCP deployment in the Internet. First, our filtering mechanism assumes that all hosts which reply with the same sender's key as ours are middlebox-affected and *do not* support MPTCP. However, this excludes the possibility of legitimate MPTCP hosts whose MPTCP options in the SYN-ACK are either stripped or overwritten by middleboxes – thus resulting in false negatives. Second, our analysis may also include false positives due to middleboxes that may perform complex operations on packets with extended TCP options, e.g., modifying sender's keys. Note that the assumption does not affect MPTCPv1 ZMap scans as the sender's key is excluded in the MPTCPv1 SYN and, therefore, cannot be modified by middleboxes. Therefore, we detect the presence of interfering middleboxes by running Tracebox [19] towards targets that sent the `MP_CAPABLE` option in our MPTCPv0 ZMap scans.

Methodology. Similar to our ZMap methodology, we issue Tracebox requests with the `MP_CAPABLE` option towards a target address. In the reply, we receive responses from intermediate routers on the path, including any modifications made. Overall, we observe the following different behaviors in Tracebox responses.

I. Only the target IP modifies the `MP_CAPABLE` option.

II. An intermediate hop modifies the `MP_CAPABLE` option.

III. The target was unresponsive or the query timed out.

Based on these three categories, we classify MPTCP support as follows. Since *category I* responses are caused by IP addresses updating MPTCP options with their own sender's key in the SYN-ACK response; we classify such targets as *truly* MPTCP-capable. Targets in *category II* are clearly affected by middleboxes on the path and hence tagged as *middlebox-affected*. Lastly, we classify hosts in *category III* as *unreachable*.

True MPTCP support in the Internet. We check if any of the end-hosts that mirror our MPTCP key in ZMap (mirrored key hosts in Figure 5) truly support MPTCP. Interestingly, we did not *any* end-host among these that sent back the `MP_CAPABLE` flag (indicating MPTCP support) to our Tracebox probe for both IPv4 and IPv6. This confirms that our ZMap analysis does not lead to any false negatives, and checking for mirrored MPTCP options in SYN-ACK is an effective first step in filtering out middleboxes.

We continue our analysis with Tracebox responses from hosts that respond with different sender's keys in ZMap. We observe that a large percentage of targets do not respond to our Tracebox queries and are therefore classified as *unreachable*. This behavior is slightly more predominant in IPv4 than in IPv6, primarily due to the different target sets (Internet-wide in IPv4, hitlist-based in IPv6). In IPv6, we only see unreachable hosts on port 443 ($\approx 82\%$), where the majority of targets are located in the same prefix of a Dutch ISP. These IPv6 targets

Fig. 6: Tracebox analysis for *consistently* responsive IPv4 addresses over port 80/443. The blue region denotes IP addresses that *truly* support MPTCP, red (barely visible) are IP addresses affected by middleboxes on the path, and orange are unreachable.

Fig. 7: MPTCP (v0 and v1) support for HTTP and HTTPS in IPv4.

respond to our Tracebox queries with *Destination Unreachable (administratively prohibited)*, which hints at blocking of our queries by the ISP. In IPv4, the number of *unreachable* targets is significantly higher on port 443 ($\approx 90\%$) than on port 80 ($\approx 48\%$). As a result, we only target addresses that were *consistently reachable*, responded with MP_CAPABLE and sent a different MPTCP sender's key in SYN-ACK at least once in the last three months of our measurement period, i.e., from October to December 2021. This precaution deflates the number of *unreachable* hosts from further analysis as it removes transient hosts that are only active for short periods of time. In total, we ran Tracebox towards ≈ 20k addresses for port 80 and 48k addresses for port 443 in IPv4. In IPv6, we sent Tracebox queries to 1267 and 1585 addresses for TCP/80 and TCP/443, respectively. We confirm that the number of *truly* MPTCP-capable addresses remains comparable before and after pruning transient IPs. Figure 6 shows the overview of our Tracebox analysis for IPv4.

Despite reducing the input dataset, we observe that a large share of IPv4 targets over port 443 are still *unreachable* ($\approx 72\%$) and the absolute number of responsive addresses is similar on both port 80 and 443. Contrary to our initial assessment based on the ZMap results, we find *true* MPTCP support in IPv4 to be slightly higher on port 80 (16.5k hosts) than on port 443 (13.5k hosts). We also find that the *true* support has almost doubled within the year 2021 in IPv4, as our previous observations found only 7.5k hosts and 6.9k hosts to *truly* support MPTCP over port 80 and 443, respectively, at the end of December 2020 [15]. Similarly, in IPv6, we find that 1195 and 1184 hosts *truly* support MPTCP over TCP/80 and TCP/443, respectively. This is an increase of almost $40\times$ in MPTCP support compared to December 2020 (31 on TCP/80 and 27 on TCP/443).

Our results also show a limited interference of middleboxes affecting MPTCP support in IPv4. In IPv4, we find 9 and 7 hosts that are affected by middleboxes on the path on TCP/80 and TCP/443, respectively. For all recorded instances, the middleboxes stripped the MPTCP options from the SYN packet. Note that the result is in contrast to our previous observations [15] in December 2020 where we found 402 and 1.3k *middlebox-affected* end-hosts on port 80 and 443,

including 6 *truly* MPTCP hosts. While we are unsure of the exact reason behind the significant decrease in hosts affected by middleboxes, we find that none of the *middlebox-affected* end-hosts in our previous analysis responded to our recent scans in late 2021. As a result, we likely attribute those short-lived end-hosts to enthusiasts, system tinkerers, or researchers that may have used MPTCP for short time periods. Note that since MPTCP options are stripped for the *middlebox-affected* end-hosts, we cannot accurately assess the true support for MPTCP within this group. On the other hand, similar to our past observations, we do not find a single middlebox-affected target address in IPv6.

We attempt to investigate the deployment nature of middleboxes that impact MPTCP traffic using Nmap [45] fingerprinting. Unfortunately, this did not lead to fruitful results due to the following hindrances. First, the majority of middleboxes that impact our study do not respond to Tracebox probes, and hence we are unable to identify their IP address. Second, for the handful of middleboxes that we positively identified, the accuracy of Nmap is too low to confidently identify their hardware and OS characteristics. However, we find that most middleboxes we identified in our latest scans in 2021 belonged to three different ISPs operating in the USA, UK and Asia. We leave the thorough analysis of middleboxes in the Internet and their impact on TCP extensions to future work.

MPTCP Port Overlap. Since an MPTCP-capable machine can offer different services concurrently, we now examine the overlap between TCP/80 and TCP/443 end-hosts. Figure 7 shows the port breakup of all MPTCPv0 and MPTCPv1 capable IPv4 addresses throughout our study over either port 80/443. As shown in the figure, most IPv4 MPTCP hosts provide services over both ports simultaneously. It must be noted that this result is in stark contrast to our previous observations made in December 2020, in which we found that the majority of IPv4 addresses provided complementary services over either of the two ports. At the end of December 2021, we find that 80.6% and 66.9% of IPv4 addresses support both HTTP and HTTPS simultaneously over MPTCPv0 and MPTCPv1, respectively. Hosts that only support HTTPS are in the minority in MPTCPv0 ($< 1\%$) as more addresses tend to provide services over HTTP (18.5%). On the other hand, HTTPS-only support is more popular over MPTCPv1 (22.4%) compared to HTTP-only support (10.7%). The picture is different for IPv6, where more than 77% of addresses

Fig. 8: Overlap between MPTCPv0 and MPTCPv1 support in IPv4 and IPv6.

(a) MPTCPv0   (b) MPTCPv1

Fig. 9: Geographic distribution of *truly* MPTCP-capable IPv4 addresses verified by Tracebox. The bars show counts of unique IPs over both port 80 and 443 (including common IPs). The numbers below the x-axis denote the MPTCP-capable IPs serving over port 80 and 443 in that country.

ASN	#Port80	#Port443	Rank	Country	Owner
9269	6461	6370	364	HK	HK Broadband
6185	1347	1344	13577	US	Apple Inc.
61157	674	534	1368	DE	Plus Server
1221	529	456	76	AU	Telstra Corp.
18618	390	390	3915	US	West Central
18943	353	352	3533	US	Yelcot Tele.
7922	419	209	27	US	Comcast
11976	232	231	2360	US	Fidelty Comm.
202870	404	2	16712	IT	Dimensione
15129	369	1	4034	US	Geneseo Tele.

TABLE I: Top 10 Autonomous Systems for *truly* MPTCP-enabled hosts in IPv4.

ASN	#Port80	#Port443	Rank	Country	Owner
6185	98	98	14234	US	Apple Inc.
396986	2	2	16570	US	Bytedance
206293	0	3	22645	DE	ProIO GmbH
714	1	2	6949	US	Apple Inc.
3209	0	2	221	DE	Vodafone

TABLE II: Top 5 Autonomous Systems for MPTCPv1 hosts in IPv4.

ASN	#Port80	#Port443	Rank	Country	Owner
6185	1163	1163	14234	US	Apple Inc.
63949	4	3	7042	US	Linode
4811	3	3	2034	CN	China Tele.
714	2	2	6949	US	Apple Inc.
201155	2	2	26184	CH	EmbeDD

TABLE III: Top 5 Autonomous Systems for *truly* MPTCPv0-enabled hosts in IPv6.

support MPTCPv0 on both ports and 100% of addresses support MPTCPv1 on both ports.

MPTCP Version Overlap. Considering the increasing adoption trend of MPTCPv0 and MPTCPv1 within 2021 (as shown in §IV-B), we now take a deeper look into understanding the deployment nature of both versions. Figure 8 shows the overlap between end-hosts that support MPTCPv0 and MPTCPv1. For this analysis, we combine end-targets that were active on either TCP/80 or TCP/443 in our measurements. We observe that a large majority of the end-hosts on IPv4 and IPv6, i.e., 87.68% and 93.25% respectively, only support the earlier experimental MPTCPv0 protocol. Despite being available in the default Linux kernel, just 10.65% of IPv4 addresses provide services only on MPTCPv1. On the other hand, all IPv6 end-hosts that support MPTCPv1 also support MPTCPv0—with all of these addresses belonging to Apple (AS6185). Similarly, we find that a fraction of IPv4 addresses support both MPTCPv0 and MPTCPv1 simultaneously (1.66%). Further investigation reveals that the majority of these overlapping MPTCP targets in IPv4 belong to Apple (AS6185) and Telecom Italia (AS3269).

> *Takeaway* — Using Tracebox we find that the *true* support for MPTCP is much lower compared to numbers reported by ZMap, around 16.5k and 13.5k for IPv4 and 1195 and 1184 for IPv6 over HTTP and HTTPS, respectively. Compared to our last observations in December 2020, MPTCP support has grown significantly ($\approx 2\times$ in IPv4 and $\approx 40\times$ in IPv6) in the year 2021. Despite its experimental nature, MPTCPv0 is still the more popularly deployed version on IPv4 and IPv6 and a majority of MPTCP hosts (both v0 and v1) provide services over port 80 and 443 simultaneously. We also find a few MPTCP end-hosts to be middlebox-affected over IPv4, while IPv6 MPTCP support remains largely unaffected.

D. Geo-distribution of MPTCP-capable hosts

We now shed some light on the physical deployment locations and operational zones of end-hosts that *truly* support MPTCP. Figure 9a visualizes the top 5 countries with the most MPTCPv0 capable host densities while Figure 9b shows the top-4 countries with the most MPTCPv1 capable hosts on IPv4, arranged in decreasing fashion. We use the MaxMind database [46] for our analysis and only show country-level breakup since more fine-granular IP geo-location suffers from accuracy shortcomings [47]–[49]. Table I provides further insights, as it shows the top 10 ASes with most IPv4 MPTCPv0 hosts on port 80 and 443. Similarly, Table II shows the top-5 ASes with MPTCPv1 hosts on both ports in IPv4. The tables also list the associated organization name, country and AS rank, which we obtain from CAIDA's AS database [50].

We find that close to half of all IPv4 MPTCP hosts are using MPTCPv0 and are deployed in Hong Kong, totaling more than 6000 unique addresses over port 80 and port 443. Comparing our results to our observations drawn in December 2021 [15], we find that the majority of MPTCP hosts in Hong Kong started operation in the year 2021 and are hosted by a major ISP in the region, Hong Kong Broadband (AS9269). However, HK Broadband has almost negligible MPTCPv1 support in IPv4, which is still being dominated by the USA. Furthermore, the USA also comes close second as the largest supporter of either MPTCPv0 or MPTCPv1, with almost 5.8k unique MPTCP-capable hosts. Table I shows that within the US, Apple has the largest deployment of MPTCP servers operational on both port 80 and 443, totaling close to 3000 unique IPv4 addresses. The result is unsurprising since Apple has been publicly known to use MPTCP for several iOS services, e.g., Siri, Music, Maps, and has recently allowed third-party developers to utilize MPTCP for non-system-native apps [7]. We also find that Apple has recently started supporting the newly standardized MPTCPv1, even migrating its existing MPTCPv0 hosts to MPTCPv1 (we investigate this further in §VI-A). The third-largest support for MPTCP over IPv4 comes from Germany, mainly due to servers hosted by Plus Server, a major cloud hosting company in the region. We also observe that many network operators and ISPs across the globe (specifically in the US) are utilizing MPTCP within their networks to enhance several of their client-facing services. Interestingly, Korea Telecom, which publicly announced exploiting MPTCP to provide Gigabit speeds over Wi-Fi and LTE [8] fails to be among the top-10 ASNs with MPTCP-capable hosts (with 205 unique MPTCP addresses). Interestingly, we also observe from Figure 9a that in certain deployments such as Dimensione (AS202870) and Geneseo Telecom (AS15129), MPTCP deployment favors one port over the other, showcasing an organization's tendency to utilize MPTCP for serving specific application traffic.

In Table III we show the AS distribution of *truly* MPTCPv0-capable IPv6 addresses. Compared to IPv4, MPTCPv0 support in IPv6 is completely dominated by Apple deployment with more than 1k addresses. The rest of the top 5 ASes in IPv6 only has a handful of MPTCPv0 capable hosts. We omit the top 5 table for IPv6 MPTCPv1, as it only contains a single entry, namely Apple with 84 IPs. Overall, we find that the current MPTCP deployment in IPv4 and IPv6 spans more than 80 countries across the globe.

> *Takeaway* — The largest support for MPTCP in IPv4 comes from Hong Kong Broadband, which significantly favors MPTCPv0 over MPTCPv1. The second largest MPTCP deployment in IPv4, and largest in IPv6, is within the USA— backed by Apple and other ISPs in the country that use MPTCP to enhance their services.

E. Middlebox Impact on MPTCP Perceived Quality

Our analysis in §IV-C revealed a prevalence of middleboxes that modify extensions MPTCP relies on. Previous research has shown that certain middleboxes, such as firewalls or load balancers, manipulate packets that do not fit pre-defined rule

Fig. 10: HTTP(S) GET timing deltas over MPTCP and TCP towards: *truly*-MPTCP targets on port 80 (a) and port 443 (b); MPTCP targets affected by middleboxes via Tracebox (c); and targets affected by mirroring middleboxes (d). Please note the different x-axis time scale.

sets, e.g., by marking them low-priority or forwarding them on longer paths [12]. In this section, we want to answer whether middleboxes treat MPTCP application traffic differently from regular TCP traffic.

We investigate this by initiating HTTP(S) GET requests using MPTCP from AWS in Germany towards IPv4 addresses that are marked *potential*-MPTCP in §IV-B. We conduct the same measurements over regular TCP from the same data center in parallel. For each successful GET response, we record (1) the TCP handshake time (a.k.a. connect time), (2) the TLS handshake time, (3) time to first byte (TTFB), and (4) the total completion time (roughly equates to website load time). We run each measurement set, composed of 10+ runs, for almost two weeks. Overall, $\approx 80\%$ and $\approx 27\%$ targets responded to our GET requests on port 80 and 443, respectively.

Figures 10a and 10b show the distribution of Δ time difference between responses from *truly* MPTCP IPs identified in §IV-C. Keep in mind that these targets are *not* affected by middleboxes on the path. Δ values less than zero denote targets that are faster using MPTCP, while $\Delta > 0$ are hosts that are faster over TCP. Values centered around zero indicate that both protocols perform similarly. The symmetric upper and lower distributions in Figure 10a shows that the clients observe no discernible difference using (MP)TCP if connecting to targets that support MPTCP over port 80. MPTCP-capable targets on port 443 (shown in Figure 10b) show similar results for all timing values except completion time, for which the distribution tilts slightly in favor of TCP.

We now investigate the impact of middleboxes on MPTCP traffic. In Figure 10c we show the responses from MPTCP-capable targets found to be affected by middleboxes. As can be observed, middleboxes treat MPTCP application traffic

(a) MPTCP traffic share. (b) MPTCP absolute traffic.

Fig. 11: MPTCP traffic over time captured by CAIDA monitors on direction-A. (a) shows the share of MPTCP flows and bytes (compared to TCP) and (b) shows their absolute values across time. The gap is due to missing data for 2017.

Fig. 12: MPTCP traffic share for bytes and flows at MAWI's samplepoint-F.

differently. For ≈30% of *all* timing values, MPTCP is slower than TCP while TCP is slower for only 10% of measurements. Notice the difference in x-axis ticks of Figure 10c and Figure 10a; indicating that middleboxes can expand TTFB and load time of MPTCP connections by several seconds. Likely, the MPTCP client falls back to TCP before initiating a data transfer for these targets since middleboxes strip away MPTCP options from the header [13]. As a result, such middleboxes only affect the TCP handshake phase, which also justifies large connect time values recorded for these targets. However, not all middleboxes have a deleterious impact on MPTCP traffic, as seen in Figure 10d. The result shows that middleboxes that simply replay unknown TCP extensions (mirroring middleboxes) have no discernible effect on MPTCP traffic. Keep in mind that data transfers over these connections end up using TCP since none of the end-targets in this group were found to support MPTCP.

> *Takeaway* — We observe no significant difference in HTTP(S) GET responses when using MPTCP over TCP from *truly* MPTCP-capable servers. However, we find that certain middleboxes can aggressively delay MPTCP connections, whereas TCP remains largely unaffected.

V. MPTCP INTERNET TRAFFIC SHARE

We quantify the real-world MPTCP traffic share by analyzing two traffic traces from geographically diverse vantage points: (1) four years of traffic (from 2015 to 2019) on a Tier 1 ISP backbone link in North America (CAIDA traces [51]) and (2) eight years of traffic (from 2014 to 2022) captured at the uplink of a Japanese university network (MAWI traces [52])
CAIDA. The CAIDA dataset includes bidirectional traffic captured at an Equinix data center connected to an ISP backbone link (we only consider single direction "*dir-A*" traffic in our analysis). For 2015 and 2016, the monitor captures traffic of the ISP backbone connecting Chicago and Seattle, while for 2018 and 2019, the backbone links New York and São Paolo. The dataset includes a one-hour trace per month for four months of 2015 and 2016 each, ten months for 2018 and January 2019. No data is available for 2017 and after January 2019 since the monitored links have been upgraded to 100 Gbps and exceed capturing capacity.

MAWI. The MAWI dataset includes traffic captured at samplepoint-F, a 1 Gbps transit link of the WIDE working group to an upstream ISP. We analyze 15-minute captures of the third Thursday of each month, from January 2014 to December 2021. This selection allows for better comparison between months, ensuring that weekday traffic is analyzed.

Generally, both CAIDA and MAWI datasets are anonymized, disallowing us to identify participating endpoints accurately. This is sufficient to understand the popularity of MPTCP in real-world Internet traffic. However, we analyzed a small set of unanonymized MAWI traces to better understand communication endpoints and compare them to our active measurements. We remove all flows with less than five packet exchanges to prevent possible scanning traffic from influencing our study.

A. MPTCP Traffic Characteristics

Figures 11a and 12 show the share of MPTCP flows and bytes over TCP at the CAIDA and MAWI vantage points, respectively. We observe that the MPTCP share remains relatively consistently low in the CAIDA dataset, making up only 0.00006% of TCP byte and 0.0003% of TCP flow traffic. However, there is an apparent uptick in MPTCP flow share at the start of 2018 that increases as the year progresses, reaching 0.005%. Interestingly, the trend is mostly missing on MPTCP byte share, indicating a simultaneous rise of TCP traffic on the link. By the end of 2018 (and the beginning of 2019), both MPTCP flow and byte share within the CAIDA dataset escalate significantly and peak at 0.02% and 0.002%, respectively. Figure 11b paints the complementary picture of the dataset in absolute numbers. The bars (attached to the left y-axis) denote the aggregate amount of MPTCP bytes, and the line (to the right y-axis) shows the mean of MPTCP flows over four years. We observe a ≈ 8.6× jump in MPTCP bytes from 2016–2018 and an increase of 64% within 2018–2019. However, the concurrent increase in the number of MPTCP flows hints that MPTCP is largely being used for short-lived small transfers. Unfortunately, we cannot analyze the after-effects of MPTCP upstreaming in Linux at the beginning of 2020 from the CAIDA dataset as no trace data is available beyond 2019. Hence we turn our attention to the MAWI traces.

Fig. 13: MPTCP flow size distribution at MAWI's samplepoint-F showing top 1, top 5, and top 50% of flows compared to all MPTCP traffic.

Fig. 14: Exponentially weighted moving average of TCP and MPTCP flow count in MAWI captures.

From Figure 12 we observe that the share of MPTCP traffic flows captured by MAWI stays relatively constant over time, making up less than 0.1% of all TCP traffic flows. The share of MPTCP bytes is even smaller, until the end of 2019, as we begin to see it increase substantially, peaking at upwards of 0.4% in June 2020 and February 2021. Interestingly, the number of MPTCP flows remains low and does not increase. We further investigate this phenomenon by looking at the flow size distribution over time. To convey this distribution, we show the traffic share of the top flow, the top five flows, and the top 50% of flows in Figure 13. If all flows had the same size, the green top 50% line would be at 0.5. Around the end of 2019, we see a drastic change in flow size distributions. At times, a single flow makes up 50% of all MPTCP traffic, and the top five flows make up almost all of MPTCP traffic. This indicates that MPTCP is starting to be used and carries actual data. We also evaluate the duration of these "elephant flows" and find that they last $\approx$ 30s. The relatively short duration also explains the dips in the top 5 in 2020 and 2021, as seen in Figure 12. If an elephant MPTCP flow is not present within MAWI's 15-minute daily capturing window, its distribution and traffic share drops considerably.

To further analyze the number of flows, we process all Thursdays recorded by MAWI from 2017 until the end of 2021. Figure 14 shows the exponentially weighted moving average (EWMA)[2] of the number of MPTCP and TCP flows. As can be seen, the number of TCP flows is relatively stable during the analysis period. However, the number of MPTCP flows fluctuates, and we find two peaks and valleys in each year between 2017 and 2020. These fluctuations reflect the nature of the MAWI dataset, which is an academic network capture and can be affected by the presence of students and staff on campus. At the beginning of 2020 (vertical red line), the COVID-19 pandemic leads to a switch to remote working and remote learning, which drastically impacts the TCP flows, but even more so the MPTCP flows. The number of flows decreases, and the trend is not comparable to that of previous years. The result is not surprising, as our active scans show that mobile applications on iOS are the main MPTCP

clients in the wild (see §IV-C). At the end of 2021, many lockdown restrictions were lifted in Japan, and we see an increase in MPTCP and TCP flows. This finding is consistent with previous work investigating the effects of the COVID-19 pandemic on Internet traffic [18], [53]–[58]. Furthermore, we also find that all captured MPTCP traffic between 2017 and 2021 in our MAWI dataset is for MPTCPv0. However, we do observe a handful of MPTCPv1 flows in 2022, all of which were to/from Apple servers (cf. §VI-A).

B. MPTCP Application Usage

To better understand the applications used in MPTCP traffic, we map transport port numbers for MAWI and CAIDA traces to well-known port numbers used for specific services [59]. Additionally, we leverage Apple's list of ports used in their services to identify Apple service traffic [60]. We are able to successfully map all flows to well-known ports in MAWI; except for one flow with both high ports and two flows with reserved value zero as the source port. The latter could be attributed to misconfigured devices [33]. For CAIDA, we find more than 80% of source ports in the well-known range and a majority of destination ports as ephemeral, indicating that the link mainly carries server-to-client upstream traffic.

Overall, we observe six different applications utilizing MPTCP in MAWI: HTTPS, HTTP, Ident, SMB, Siri, and RDP. However, the overwhelming majority of all MPTCP traffic is HTTPS traffic, whose lowest share is 99.5%. On the other hand, the application mix in the CAIDA dataset is more diverse than MAWI, as we find 15+ services using MPTCP, including HTTPS, HTTP, Spamtrap, and Microsoft services. However, HTTPS traffic eclipses all other applications, similar to MAWI, with 99.91% being its lowest share. Moreover, other than very small traces of Siri in 2018, we did not discover any other instances of Apple services using MPTCP in both datasets. Finally, using non-anonymized MAWI traces for select days in 2021, we find that all MPTCP traffic in that year is directed toward Apple servers.

Takeaway — The MPTCP traffic share remains consistently low over time. Since mid-2019, MPTCP traffic has increased steadily and now includes larger flows, indicating

---

[2] $EWMA(t) = a \times x(t) + (1-a) \times EWMA(t-1)$, with $a = 0.2$

Fig. 15: MPTCP IPv4 hosts within ASes owned by Apple (measurement period extended to February 2022).

that MPTCP has started to see actual deployment. With more than 99% of all MPTCP traffic, HTTPS over MPTCPv0 is the dominant application using MPTCP in the wild. We can also observe a drop in MAWI MPTCP traffic during the COVID-19 pandemic, confirming findings by related work.

VI. Discussion

A. Case-Study: A Deeper Bite of the Apple

Our analysis in §IV-D shows that Apple continues to be one of the most prominent supporters of MPTCP and has a significant deployment of servers that support MPTCPv0 and MPTCPv1 for services on port 80 and port 443 in IPv4 and IPv6. We also show through our MAWI traffic analysis in §V that Apple also actively utilizes MPTCP to enhance the performance of its system services, e.g., Siri. As such, we now provide a deeper look into the MPTCP support within Apple, explicitly understanding how Apple is integrating the newly standardized MPTCPv1 within its infrastructure.

Figure 15 shows the IPv4 hosts within ASes owned by Apple (i.e., ASN6185 & ASN714) that reportedly support MPTCPv0 and MPTCPv1 in our ZMap scans. Since we started scanning for MPTCPv1 in our ZMap scans in March 2021, we analyze the period beginning in April 2021 and extend our end date to February 2022 (totaling *ten* months). Firstly, we find that Apple's support over MPTCPv0 remains relatively consistent throughout our analysis period, with almost 1500+ hosts supporting MPTCPv0 in our ZMap scans. We also observe that a large majority of these hosts (exceeding 90%) provide services over both port 80 and port 443—further supporting our port breakup analysis in Figure 7. Interestingly, the plot shows that despite MPTCPv1 being available in the default Linux kernel 5.6 since March 2020 [9], Apple did not utilize MPTCPv1 for its services until October 2021, i.e., almost 20 months after its availability. However, MPTCPv1 support within Apple remained limited compared to MPTCPv0 ($\lesssim$ 100 MPTCPv1 hosts), which remained consistent until the end of January 2022. In February 2022, we observe a significant uptick in MPTCPv1 support within Apple, increasing almost 11× ($\approx$ *1400* hosts), approximately equaling MPTCPv0 support in the same month. Similarly, in IPv6, we find that Apple had no MPTCPv1 deployment until September 2021, with $\approx$ 1200 MPTCPv0 IPs. In October 2021, Apple added 90 dual-protocol MPTCPv1 IPs, with the bulk of 1100 additional IPs being added in February 2022.

Fig. 16: Added support and migration between MPTCPv0 and MPTCPv1 within Apple ASes (*note the different y-axis scale for both versions*). The measurement period extends to February 2022.

The sudden support for MPTCPv1 within Apple is quite peculiar and can be due to multiple reasons. For one, Apple may be migrating its existing MPTCPv0 infrastructure to MPTCPv1. The simultaneous support for both versions, however, makes this assumption unlikely. Similarly, Apple might be employing both MPTCPv0 and MPTCPv1 for different (non-overlapping) services. We investigate the reason behind the event further by analyzing the support migration in consecutive months between the two MPTCP versions within Apple infrastructure (see Figure 16). The bottom half of the figure denotes hosts that supported MPTCPv0 on either TCP/80 or TCP/443 in the previous month and now added support for MPTCPv1. Similarly, the top half of the figure denotes hosts that supported MPTCPv1 in the previous month and now support MPTCPv0. The dashed region denotes "migrated" hosts, i.e., hosts that no longer support the previously supported MPTCP version. On the other hand, the "support" numbers denote hosts that did not support one of the two versions in the previous months but now show support for both MPTCPv0 and MPTCPv1. The plot shows a direct correlation with Figure 15. We find that the minor uptick in MPTCPv1 support within Apple in October 2021 over IPv4 and IPv6 is due to MPTCPv0 hosts that also started supporting MPTCPv1. In the following few months, we observed that many IPv4 hosts that supported either MPTCPv0 or MPTCPv1 in the previous month began supporting the other version. Note that the trend is largely missing from IPv6. Interestingly, we find that only 11 IPv4 hosts migrated their support from MPTCPv1 to MPTCPv0 (we did not find any migration in the opposite direction or within IPv6). On the other hand, the result shows that the significant uptick in MPTCPv1 support within Apple in February 2022 is due to IPv4 and IPv6 hosts that previously supported MPTCPv0 also starting to support MPTCPv1 ($\approx$96% of total MPTCPv1 hosts this month also support MPTCPv0).

B. Factors Affecting Future MPTCP Deployment

In this work, we showcase that MPTCP's adoption and support globally have been slow but steady. Originating as a

research project with ambitions to maximize throughput within datacenter environments [6], MPTCP has evolved significantly and is now primarily used in commercial (mobile-focused) networks—thanks to its capabilities to multiplex over multiple heterogeneous paths. However, we observe that the rise in *infrastructure* size is not yet reflected in MPTCP's *traffic share* compared to TCP in the Internet (see §V). We also find that MPTCPv0 is highly susceptible to middleboxes in the Internet – however, this shortcoming seems to have been successfully plugged by the introduction of MPTCPv1. This makes the potential future for MPTCP "somewhat attractive" to the Internet community, provided the state of the transport protocol standards remains unchanged. For instance, as we highlight in §III, there is a significant interest in the community to support and utilize QUIC for supporting service operations in the Internet. Thanks to its flexibility, the protocol allows operators to experiment with inherent functionality to best fit the needs of different use-cases [61]. Additionally, being end-to-end encrypted, QUIC remains largely unaffected by middleboxes in the Internet (unlike TCP). As a result, several researchers are actively working on providing similar multipath capabilities to QUIC. As of now, there are three different multipath proposals for QUIC: [62], [63] and [64] – each proposing different features to support multipath. While no specific multipath over QUIC proposal has yet been selected for standardization, thanks to the inherent disagreements on core design [65], the authors of all three proposals wrote a common draft [66] which has a high likelihood of being standardized. Considering the popularity of QUIC, the potential standardization of MP-QUIC may threaten the deployment and continued usage of MPTCP. As such, we envision the widespread adoption of MPTCP to be possible only if it is embraced by service-providing organizations, in addition to Apple, for supporting real applications in the Internet.

C. Limitations of our Methodology

Our methodology does not capture client-side MPTCP deployments, including MPTCP proxy solutions that work only when the client establishes an MPTCP connection. The passive data analysis (see §V) only focuses on MPTCP support and not MPTCP usage. We plan to plug these limitations in a future study, with MPTCPv1 results being already available at `https://mptcp.io`.

VII. Conclusion

This paper presented the first broad multi-faceted assessment on MPTCPv0 and MPTCPv1 deployment in the Internet. We studied both the *infrastructure*, by probing the entire IPv4 address space and an IPv6 hitlist for MPTCP-capable IPs, and *traffic share* at two geographically diverse vantage points. We identified middleboxes that impact both MPTCP scanning attempts and user traffic during the course of our study, hence providing the most accurate picture of *true* MPTCP deployment to date. We observed a steady growth in MPTCP-enabled IPs that support HTTP and HTTPS in our 18-month investigation period, reaching $\approx$ 13k+ and 1k in December 2021 for IPv4 and IPv6, respectively. The growth is primarily driven by Apple and ISPs across the globe that rely on the protocol to enhance their services.

VIII. Acknowlegment

References

[1] O. Bonaventure, M. Handley, and C. Raiciu, "An Overview of Multipath TCP," vol. 37, no. 5, p. 17, 2012. [Online]. Available: http://hdl.handle.net/2078.1/114081

[2] T. Shreedhar, N. Mohan, S. K. Kaul, and J. Kangasharju, "QAware: A Cross-Layer Approach to MPTCP Scheduling," in *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, 2018, pp. 1–9. [Online]. Available: https://doi.org/10.23919/IFIPNetworking.2018.8696843

[3] N. Mohan, T. Shreedhar, A. Zavodavoski, O. Waltari, J. Kangasharju, and S. K. Kaul, "Redesigning mptcp for edge clouds," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 675–677. [Online]. Available: https://doi.org/10.1145/3241539.3267738

[4] S. Ferlin, O. Alay, O. Mehani, and R. Boreli, "Blest: Blocking estimation-based mptcp scheduler for heterogeneous networks," in *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, 2016, pp. 431–439.

[5] X. Corbillon, R. Aparicio-Pardo, N. Kuhn, G. Texier, and G. Simon, "Cross-layer scheduler for video streaming over mptcp," in *Proceedings of the 7th International Conference on Multimedia Systems*, ser. MMSys '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2910017.2910594

[6] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving datacenter performance and robustness with multipath tcp," in *Proceedings of the ACM SIGCOMM 2011 Conference*, ser. SIGCOMM '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 266–277. [Online]. Available: https://doi.org/10.1145/2018436.2018467

[7] Apple. (2017) Use Multipath TCP to create backup connections for iOS. [Online]. Available: https://support.apple.com/en-us/HT201373

[8] Android Authority. (2015) South Korea's KT launches 1.17Gbps GiGA LTE. [Online]. Available: https://www.androidauthority.com/kt-launches-1gbps-giga-lte-617147/

[9] Netdev Group. (2020) MPTCP Linux kernel upstream. [Online]. Available: https://git.kernel.org/pub/scm/linux/kernel/git/netdev/net-next.git/commit/?id=f870fa0b5768842cb4690c1c11f19f28b731ae6d

[10] 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*, 2014, pp. 27–32. [Online]. Available: https://doi.org/10.1145/2630088.2631977

[11] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, "Making middleboxes someone else's problem: Network processing as a cloud service," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, p. 13–24, aug 2012. [Online]. Available: https://doi.org/10.1145/2377677.2377680

[12] B. Hesmans, F. Duchene, C. Paasch, G. Detal, and O. Bonaventure, "Are tcp extensions middlebox-proof?" in *Proceedings of the 2013 Workshop on Hot Topics in Middleboxes and Network Function Virtualization*, ser. HotMiddlebox '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 37–42. [Online]. Available: https://doi.org/10.1145/2535828.2535830

[13] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch, "RFC 8684: TCP Extensions for Multipath Operation with Multiple Addresses." [Online]. Available: https://www.rfc-editor.org/rfc/rfc8684.txt

[14] O. Mehani, R. Holz, S. Ferlin, and R. Boreli, "An early look at multipath tcp deployment in the wild," in *Proceedings of the 6th International Workshop on Hot Topics in Planet-Scale Measurement*, ser. HotPlanet '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 7–12. [Online]. Available: https://doi.org/10.1145/2798087.2798088

[15] F. Aschenbrenner, T. Shreedhar, O. Gasser, N. Mohan, and J. Ott, "From Single Lane to Highways: Analyzing the Adoption of Multipath TCP in the Internet," in *Proceedings of the 2021 IFIP Networking Conference*, 2021. [Online]. Available: https://doi.org/10.23919/IFIPNetworking52078.2021.9472785

[16] O. Gasser, Q. Scheitle, P. Foremski, Q. Lone, M. Korczyński, S. D. Strowes, L. Hendriks, and G. Carle, "Clusters in the expanse: Understanding and unbiasing ipv6 hitlists," in *Proceedings of the Internet Measurement Conference 2018*, ser. IMC '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 364–378. [Online]. Available: https://doi.org/10.1145/3278532.3278564

[17] M. Trevisan, D. Giordano, I. Drago, M. M. Munafò, and M. Mellia, "Five Years at the Edge: Watching Internet From the ISP Network," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 561–574, 2020.

[18] A. Feldmann, O. Gasser, F. Lichtblau, E. Pujol, I. Poese, C. Dietzel, D. Wagner, M. Wichtlhuber, J. Tapiador, N. Vallina-Rodriguez, O. Hohlfeld, and G. Smaragdakis, "The Lockdown Effect: Implications of the COVID-19 Pandemic on Internet Traffic," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1–18. [Online]. Available: https://doi.org/10.1145/3419394.3423658

[19] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet, "Revealing middlebox interference with tracebox," in *Proceedings of the 2013 Conference on Internet Measurement Conference*, ser. IMC '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1–8. [Online]. Available: https://doi.org/10.1145/2504730.2504757

[20] F. Aschenbrenner, T. Shreedhar, O. Gasser, N. Mohan, and J. Ott. (2021) Dataset: From Single Lane to Highways: Analyzing the Adoption of Multipath TCP in the Internet. [Online]. Available: https://doi.org/10.14459/2021mp1610028

[21] 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*, 2014, pp. 27–32. [Online]. Available: https://doi.org/10.1145/2630088.2631977

[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 *USENIX NSDI*, 2012.

[23] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "RFC 6824: TCP Extensions for Multipath Operation with Multiple Addresses." [Online]. Available: https://www.rfc-editor.org/rfc/rfc6824.txt

[24] T. Dai and H. Shulman, "SMap: Internet-Wide Scanning for Spoofing," in *Annual Computer Security Applications Conference*, ser. ACSAC. New York, NY, USA: Association for Computing Machinery, 2021, p. 1039–1050. [Online]. Available: https://doi.org/10.1145/3485832.3485917

[25] L. Izhikevich, R. Teixeira, and Z. Durumeric, "LZR: Identifying Unexpected Internet Services," in *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, M. Bailey and R. Greenstadt, Eds. USENIX Association, 2021, pp. 3111–3128. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/izhikevich

[26] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, "Census and Survey of the Visible Internet," in *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 169–182. [Online]. Available: https://doi.org/10.1145/1452520.1452542

[27] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-Wide Scanning and Its Security Applications," in *Proceedings of the 22nd USENIX Conference on Security*, ser. SEC'13. USA: USENIX Association, 2013, p. 605–620.

[28] S. Bano, P. Richter, M. Javed, S. Sundaresan, Z. Durumeric, S. J. Murdoch, R. Mortier, and V. Paxson, "Scanning the Internet for Liveness," *SIGCOMM Comput. Commun. Rev.*, vol. 48, no. 2, p. 2–9, may 2018. [Online]. Available: https://doi.org/10.1145/3213232.3213234

[29] J. Rüth, C. Bormann, and O. Hohlfeld, "Large-Scale Scanning of TCP's Initial Window," in *Proceedings of the 2017 Internet Measurement Conference*, ser. IMC '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 304–310. [Online]. Available: https://doi.org/10.1145/3131365.3131370

[30] J. Rüth, I. Poese, C. Dietzel, and O. Hohlfeld, "A First Look at QUIC in the Wild," in *Passive and Active Measurement*, R. Beverly, G. Smaragdakis, and A. Feldmann, Eds. Cham: Springer International Publishing, 2018, pp. 255–268.

[31] P. Richter, G. Smaragdakis, D. Plonka, and A. Berger, "Beyond Counting: New Perspectives on the Active IPv4 Address Space," in *Proceedings of the 2016 Internet Measurement Conference*, ser. IMC '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 135–149. [Online]. Available: https://doi.org/10.1145/2987443.2987473

[32] F. Qian, A. Gerber, Z. M. Mao, S. Sen, O. Spatscheck, and W. Willinger, "TCP Revisited: A Fresh Look at TCP in the Wild," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 76–89. [Online]. Available: https://doi.org/10.1145/1644893.1644903

[33] "Zeroing in on Port 0 Traffic in the Wild," in *Passive and Active Measurement*, O. Hohlfeld, A. Lutu, and D. Levin, Eds. Cham: Springer International Publishing, 2021, pp. 547–563.

[34] G. Wan, L. Izhikevich, D. Adrian, K. Yoshioka, R. Holz, C. Rossow, and Z. Durumeric, "On the Origin of Scanning: The Impact of Location on Internet-Wide Scans," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 662–679. [Online]. Available: https://doi.org/10.1145/3419394.3424214

[35] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar *et al.*, "The Quic Transport Protocol: Design and Internet-Scale Deployment," in *ACM SIGCOMM*, 2017.

[36] J. Iyengar and M. Thomson, "RFC 9000: QUIC: A UDP-Based Multiplexed and Secure Transport." [Online]. Available: https://www.rfc-editor.org/rfc/rfc9000.txt

[37] J. Zirngibl, P. Buschmann, P. Sattler, B. Jaeger, J. Aulbach, and G. Carle, "It's over 9000: Analyzing Early QUIC Deployments with the Standardization on the Horizon," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 261–275. [Online]. Available: https://doi.org/10.1145/3487552.3487826

[38] D. Madariaga, L. Torrealba, J. Madariaga, J. Bermúdez, and J. Bustos-Jiménez, "Analyzing the Adoption of QUIC From a Mobile Development Perspective," in *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, ser. EPIQ '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 35–41. [Online]. Available: https://doi.org/10.1145/3405796.3405830

[39] M. Kosek, T. V. Doan, M. Granderath, and V. Bajpai, "One to Rule Them All? A First Look at DNS over QUIC", booktitle="Passive and Active Measurement," O. Hohlfeld, G. Moura, and C. Pelsser, Eds. Cham: Springer International Publishing, 2022, pp. 537–551.

[40] K. Elmenhorst, B. Schütz, N. Aschenbruck, and S. Basso, "Web Censorship Measurements of HTTP/3 over QUIC," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 276–282. [Online]. Available: https://doi.org/10.1145/3487552.3487836

[41] O. Bonaventure. (2015) Measuring the adoption of Multipath TCP is not so simple... [Online]. Available: http://blog.multipath-tcp.org/blog/html/2015/10/27/adoption.html

[42] TUM. (2015) ZMap: The Internet Scanner. [Online]. Available: https://github.com/tumi8/zmap

[43] C. Partridge and M. Allman, "Ethical Considerations in Network Measurement Papers," *Commun. ACM*, vol. 59, no. 10, p. 58–64, sep 2016. [Online]. Available: https://doi.org/10.1145/2896816

[44] D. Dittrich *et al.*, "The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research," *US DHS*, 2012.

[45] G. Lyon. (2020) Nmap. [Online]. Available: https://nmap.org/

[46] MaxMind. (2020) GeoLite2 Free Geolocation Data. [Online]. Available: https://dev.maxmind.com/geoip/geoip2/geolite2/

[47] I. Livadariu, T. Dreibholz, A. S. Al-Selwi, H. Bryhni, O. Lysne, S. Bjørnstad, and A. Elmokashfi, "On the Accuracy of Country-Level IP Geolocation," in *Proceedings of the Applied Networking Research Workshop*, ser. ANRW '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 67–73. [Online]. Available: https://doi.org/10.1145/3404868.3406664

[48] Q. Scheitle, O. Gasser, P. Sattler, and G. Carle, "HLOC: Hints-based geolocation leveraging multiple measurement frameworks," in *2017 Network Traffic Measurement and Analysis Conference (TMA)*, 2017, pp. 1–9.

[49] Y. Shavitt and N. Zilberman, "A Geolocation Databases Study," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 10, pp. 2044–2056, 2011.

[50] CAIDA. (2020) ASRank. [Online]. Available: https://asrank.caida.org/

[51] CAIDA, "Anonymized Internet Traces Dataset," 2020. [Online]. Available: https://www.caida.org/data/passive/passive_dataset.xml

[52] MAWI Working Group, "MAWI Working Group Traffic Archive," 2020. [Online]. Available: http://mawi.wide.ad.jp/mawi/

[53] A. Feldmann, O. Gasser, F. Lichtblau, E. Pujol, I. Poese, C. Dietzel, D. Wagner, M. Wichtlhuber, J. Tapiador, N. Vallina-Rodriguez, O. Hohlfeld, and G. Smaragdakis, "A Year in Lockdown: How the Waves of COVID-19 Impact Internet Traffic," *Communications of the ACM Research Highlights*, vol. 64, Jun. 2021. [Online]. Available: https://doi.org/10.1145/3465212

[54] H. Chang, M. Varvello, F. Hao, and S. Mukherjee, "Can You See Me Now? A Measurement Study of Zoom, Webex, and Meet," in *Proceedings of the 21st ACM Internet Measurement Conference*, ser. IMC '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 216–228. [Online]. Available: https://doi.org/10.1145/3487552.3487847

[55] M. Karamollahi, C. Williamson, and M. Arlitt, "Zoomiversity: A Case Study of Pandemic Effects on Post-secondary Teaching and Learning," in *Passive and Active Measurement*, O. Hohlfeld, G. Moura, and C. Pelsser, Eds. Cham: Springer International Publishing, 2022, pp. 573–599.

[56] T. Böttger, G. Ibrahim, and B. Vallis, "How the Internet Reacted to Covid-19: A Perspective from Facebook's Edge Network," in *Proceedings of the ACM Internet Measurement Conference*, ser. IMC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 34–41. [Online]. Available: https://doi.org/10.1145/3419394.3423621

[57] M. Candela and A. Prado, "Italian Operators' Response to the COVID-19 Pandemic," *SIGCOMM Comput. Commun. Rev.*, vol. 51, no. 1, p. 26–31, mar 2021. [Online]. Available: https://doi.org/10.1145/3457175.3457180

[58] A. Lutu, D. Perino, M. Bagnulo, E. Frias-Martinez, and J. Khangosstar, "A Characterization of the COVID-19 Pandemic Impact on a Mobile Network Operator Traffic," 10 2020, pp. 19–33.

[59] IANA, "Service Name and Transport Protocol Port Number Registry," 2020. [Online]. Available: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[60] Apple, "TCP and UDP ports used by Apple software products," 2020. [Online]. Available: https://support.apple.com/en-us/HT202944

[61] M. Kosek, T. Shreedhar, and V. Bajpai, "Beyond quic v1: A first look at recent transport layer ietf standardization efforts," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 24–29, 2021.

[62] Q. D. Coninck and O. Bonaventure, "Multipath Extensions for QUIC." [Online]. Available: https://datatracker.ietf.org/doc/html/draft-deconinck-quic-multipath

[63] C. Huitema, "QUIC Multipath Negotiation Option." [Online]. Available: https://datatracker.ietf.org/doc/html/draft-huitema-quic-mpath-option

[64] Y. L. et al., "Multipath Extensions for QUIC." [Online]. Available: https://datatracker.ietf.org/doc/html/draft-liu-multipath-quic

[65] Q. De Coninck, "The packet number space debate in multipath quic," *arXiv preprint arXiv:2112.01068*, 2021.

[66] Y. L. et al., "Multi-path Extension for QUIC." [Online]. Available: https://datatracker.ietf.org/doc/html/draft-lmbdhk-quic-multipath