DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Measuring and Analysing Multipath TCP support on the Internet

Florian Aschenbrenner

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Measuring and Analysing Multipath TCP support on the Internet

Messen und Analysieren der Multipath-TCP-Unterstützung im Internet

Author:	Florian Aschenbrenner
Supervisor:	Prof. Dr.-Ing. Jörg Ott
Advisor:	Dr. Nitinder Mohan
Submission Date:	14.10.2020

Acknowledgments

I would like to thank my adivisor Dr. Nitinder Mohan for supporting, motivating and guiding me through our research. Furthermore, I would like to express my gratitude towards the MPTCP team for creating and maintaining a long overdue extension, even though they face so many obstacles, as we were not the only ones that encounter middleboxes.

Abstract

MPTCP is an extension for TCP that utilizes multiple network paths to a single target. The extension requires no changes in the application layer, but still provides a better usage of network resources and a faster reaction to network failures than its regular version. Our motivation is to reveal the current support for Multipath TCP in the wild before MPTCP get released in the official Linux kernel. For that, we issued multiple ZMAP scans on IPv4, each receiving replies of ~64M hosts on port 80 and ~44M hosts on port 443. In this thesis, we also present a new method to further distinguish between MPTCP capable hosts and middleboxes on a large scale using Tracebox. This method discovered 3,708 MPTCP capable hosts on port 80 and another 2,519 hosts on port 443. Additionally, we analyzed an external dataset for IPv6 there we estimated 200 MPTCP capable hosts on port 80 and 173 hosts with MPTCP support on port 443. While performing these scans, we noticed a strong influence of middleboxes on our scans. Our Tracebox results point to one IP in China, which was responsible for removing our MPTCP options 148 times during our scans. We also had to face ~190K middleboxes that mirror our TCP options, which deny us from determining whether the end host supports MPTCP. Furthermore, we analyzed the performance of MPTCP for the end-user with HTTP GET requests. While we do not find that middleboxes gravely affect performance of MPTCP-based HTTP and HTTPS traffic, we found almost no performance difference between MPTCP vs TCP traffic over the Internet.

Contents

1 Introduction

1.1 Motivation

While the infrastructure of the Internet is constantly evolving, the almost 40 year old TCP protocol is still similar to its initial version [8]. MPTCP aims to extend TCP in a way to utilize multiple paths for a single connection [23]. This extension promises “higher throughput and improved resilience to network failures” [23]. Apple has adopted to MPTCP at an early stage [6] and can verify those advantages with MPTCP [1]. It is even planned that MPTCP is going to be part of the upstream Linux kernel [14]. Although Apple and Linux support the deployment of MPTCP, MPTCP usage in the wild is still mostly unexplored as the latest research on the deployment prior to this thesis was carried out in an early stage of MPTCP, in 2015 [18]. A host with MPTCP enabled will not experience the advantages of MPTCP if the target does not also support MPTCP. This underlines the importance of analyzing the infrastructure of MPTCP.

In this thesis we will analyze our scans for MPTCP support on the entire IPv4 Space and an external dataset for IPv6, show the organizations behind our found MPTCP servers as well as extensive information about the hosts and present our methodology with multiple scanning tools for detecting MPTCP hosts on a large scale.

1.2 Research Questions

In this thesis we want to investigate and answer the following research questions about MPTCP.

RQ1: How many HTTP and HTTPS servers already support MPTCP over the Internet?
During this research, we want to determine the number of MPTCP capable hosts reachable in the IPv4 space. This should give us a decent understanding of the current infrastructure of MPTCP. Additionally, we want to find these numbers for both HTTP and HTTPS to learn what MPTCP capable hosts support more.

RQ2: How does MPTCP adoption look like in the IPv6 space?

There is no previous research for MPTCP deployment in IPv6. We give the first insight into the MPTCP deployment status of the IPv6 space. Similar to the previous research question, we analyze both HTTP and HTTPS to determine the support for these protocols.

RQ3: How much impact do middleboxes have on MPTCP scanning techniques?
There are reports that even native MPTCP struggles with middleboxes [11]. We have also witnessed that MPTCP scanning techniques are heavily affected by middleboxes [7]. The question arises, how many hosts can not be identified due to middleboxes? Furthermore, how many hosts have been previously wrongly identified? And is it possible for a technique to counteract false-positives from middleboxes?

RQ4: What is the performance of MPTCP for application traffic over the Internet?
The last research question revolves around the performance of MPTCP on the web. How long does it take for a user to establish or tear down a connection then MPTCP is enabled and then it is disabled? Can we spot any differences in timings and how drastic is the difference for the user?

1.3 Contribution

We contribute towards detecting and analyzing MPTCP hosts on a large scale. For this we made a tool chain that will be explained in detail in chapter 3. Furthermore, we analyze the performance for end users with MPTCP enabled and compare it with the performance of regular TCP.

1.4 Outline

In the 2nd chapter 'Related Work', we recap previous research in MPTCP deployment and also explain MPTCP in general. Chapter 3 'Methodology' regards the tools that were used during the research and the methodology behind our data collection. The following chapter 4 'Results' is about our analysis of the data collected during our research. Afterwards, we conclude our research with a summary, limitations of the thesis and future work in chapter 5.

2 Related Work

2.1 MPTCP

TCP was designed to retransmit failed packets on another route to avoid losing packets [8]. However, the infrastructure of the Internet has changed since TCP was first developed. Nowadays devices can have several interfaces, which could be utilized [8]. MPTCP aims to extend the reliable protocol to enable hosts to create various connections simultaneously via different routes to the end host [23] and to use other interfaces for establishing redundant connections in case of a network failure. The extension promises advantages like "higher throughput and improved resilience to network failure" [23], while applications in the user space do not need any changes [23].

Figure 2.1 shows the process of initiating a MPTCP connection. First a MP_CAPABLE option, along with a 64-bit MPTCP key generated by the client, is sent to the server within an TCP SYN packet. If the server supports MPTCP, it

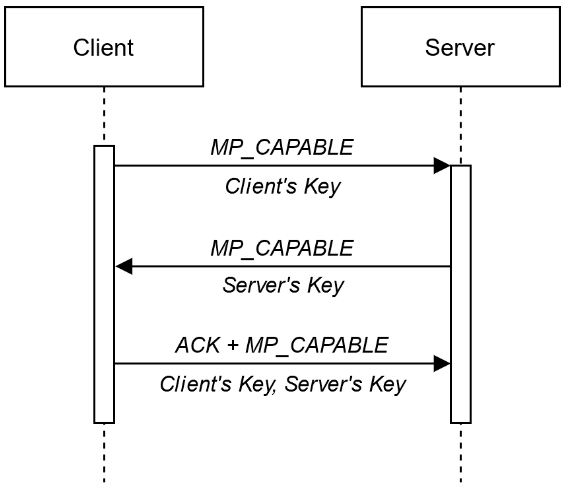


Figure 2.1: Initiation of an MPTCP Connection between two MPTCP enabled machines [23]

replies in the SYN/ACK with an MP_CAPABLE option and their own MPTCP key. The connection is finally established by the ACK packet that contains both keys and the MP_CAPABLE option. While the MP_CAPABLE option is primarily used to indicate that a machine supports MPTCP, keys are "used to authenticate the endpoints when new subflows will be set up" [23].

2.2 Measuring MPTCP in the Wild

As MPTCP still struggles with multiple issues to get into the upstream Linux Kernel [14], we look forward to installing MPTCP on our local machines without installing a custom kernel. While the benefits of MPTCP are many in theory, almost little to no attention has been paid till date to understand the scope of MPTCP adoption over the internet. At this point another study about the deployment of MPTCP is due. The most recent study aiming to discover the adoption of MPTCP over the Internet dates back to 2015 from Mehani et. al [18]. Mehani et. al tried to determine the MPTCP Capable hosts of the Alexa Top 1M list. In order to obtain their results, they probe every host in that list on port 80. The authors use ZMAP (described in section 3.1) for their probing in order to obtain a list of IP addresses that answer with an MPTCP capable option. Afterwards they perform a handshake to each IP address, which is retrieved in the previous step. The purpose of this handshake is to get other IP addresses under which a host is also available. In the end, their results disagree with previous data as most of their classified MPTCP clients are located in China [7]. They concluded that this was not the correct way to classify clients that support MPTCP [7]. Now our paper aims to correct their methodology and show results of the advanced stage of MPTCP. Here is a summary of the differences between our ZMAP scans and theirs:

- different measurement location (Norway vs Germany)
- different scan time (2015 vs 2020)
- analysis from a single scan vs analysis from multiple scans
- single port vs multiple ports (Port 80 vs Port 80/443)
- larger scale (Alexa Top 1M list vs "Entire Public IPv4 Space" [20])

'Revealing Middlebox Interference with Tracebox' [11] also contributes towards actively measuring MPTCP support. While they did not perform any real measurements, they presented a tool that allows to detect middleboxes in a path. Similar to the popular Traceroute [12] tool, Tracebox sends packets with an increasing TTL to retrieve hosts along the path to a target [11]. Only if a router conforms to the current suggestion to reply with the original ICMP packet, Tracebox is able to analyze the changes in the TCP options. This often allows the user to pinpoint the hop, there TCP options were added or removed. The reason why one might not be able to detect a middlebox is due to devices not replying with a packet, that carry the complete IP header of the packet.

3 Methodology

This chapter describes the process of data acquisition for the following research of MPTCP deployment in IPv4 and IPv6 as well as a brief look at performance of MPTCP in the web. Figure 3.1 outlines our scanning routine. We start with no input for ZMAP [20], which probes hosts for their TCP options. We end up with a list of IPs and their corresponding TCP options. Tracebox [26] is then issued for certain IPs to retrieve us information about the changes in TCP options along the path to the targets. We use this information to filter some IPs and then pipe the remaining IPs straight into Geoiplookup [17], Pyasn [3] and Curl [24]. Geoiplookup finds the location for each IP and Curl serves us for measuring the performance of MPTCP on HTTP GET requests [7]. Pyasn delivers us the ASN to each IP, which PeeringDB [21] will use to gather additional data about the AS. The last tool left in our tool chain are the Caida IXPs, which we use to associate a popularity to an AS.

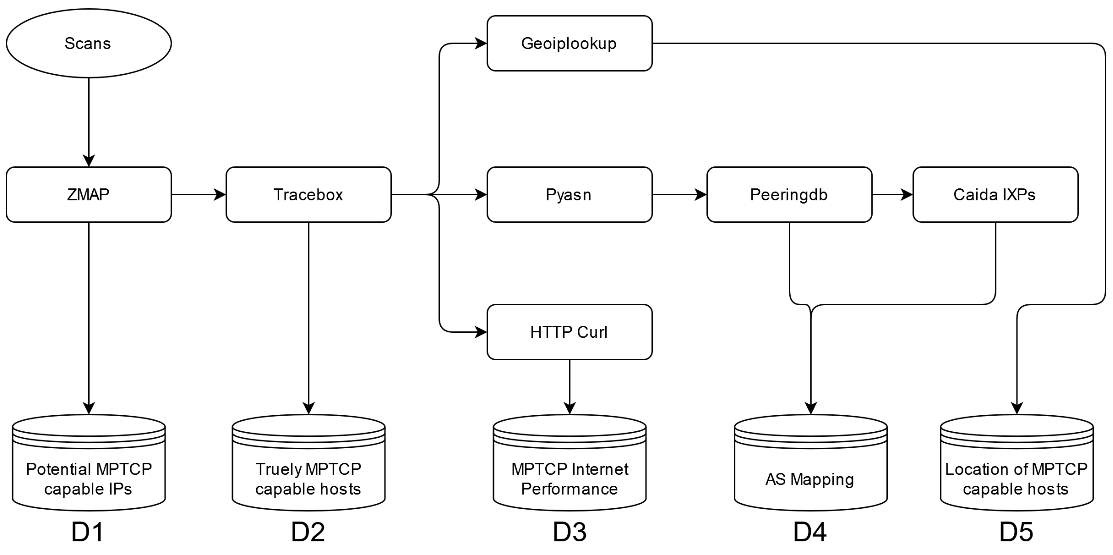

Scans
ZMAP
Tracebox
Geoiplookup
Pyasn
Peeringdb
Caida IXPs
HTTP Curl
Potential MPTCP capable IPs
Truely MPTCP capable hosts
MPTCP Internet Performance
AS Mapping
Location of MPTCP capable hosts
D1
D2
D3
D4
D5

Figure 3.1: Tool Chain used for data acquisition during our research

3.1 ZMAP

Our tool chain in figure 3.1, begins with ZMAP [22], a tool that allows us to probe each host in the IPv4 space for MPTCP support and results in the dataset D1 in figure 3.1. “ZM(AP) is a fast single packet network scanner designed for Internet-wide network surveys.” [22] However, the default version of ZMAP proved to be too ineffective for MPTCP analysis. Our approach of scanning MPTCP support is to send and read TCP options to obtain a preliminary list of hosts that support MPTCP. A fork of ZMAP from the Chair of Network Architectures and Services of TUM [20], allowed us to define the probe arguments sent by each probe. From now on, if we refer to 'ZMAP' we really refer to the fork of ZMAP that allows to send and read TCP options. With ZMAP we scanned the “entire public IPv4 space” [20] multiple times with a MPTCP capable option and our own forged MPTCP key. Overall, we successfully scanned 74M unique servers on port 80 and 52M unique servers on port 443 on IPv4 and each scan lasted several days. We distinguish the scanned hosts into three groups:

1. Not MPTCP Capable hosts
2. MPTCP Capable hosts with the same MPTCP Sender's Key
3. MPTCP Capable hosts with a different MPTCP Sender's Key

The reason for separating the groups beyond being MPTCP capable and not MPTCP capable will be explained later in section 4.1.1. While we perform all measurements in IPv4, we review an external dataset for IPv6 [13] in section 4.1.2. The only difference between our IPv4 and Gassner's IPv6 data acquisition is the input to ZMAP. The authors [13] feed ZMAP a list of IPv6 addresses to determine their target hosts, while we scan the “entire IPv4 space” [22] specifically exclude blacklisted servers from our scans. In order to compare our results with Gassner's we first had to clean their dataset. Multiple ZMAP scans were often issued for a single IPv6 address. We cleanse this by first distinguishing between the different groups and uniquifying them over their IPv6 address. For further processing, ZMAP leaves us in this first step with IP addresses of hosts which replied us with a different MPTCP key (D1) as shown in figure 3.1.

3.2 Tracebox

We feed the final IPv4 addresses from ZMAP directly into Tracebox [26] as depicted in figure 3.1. Tracebox is launched towards each host, which has replied on a ZMAP probe with a Different Sender's Key and an MPTCP capable Option. As discussed in section 2.1, Tracebox is able to detect changes in the path to our target host. We utilize this to check whether the TCP options we received on ZMAP originate from the end host or if there was an interfering middlebox on the way. Tracebox leaves us with a list of hosts, that we classify as truly MPTCP capable hosts.

3.3 Geoiplookup

"(G)eoiplookup uses the GeoIP library and database to find the Country that an IP address or hostname originates from" [17]. This tool therefore allows us to map the location of each IP, which Tracebox classified as MPTCP capable, to their location.

3.4 AS Analysis

The goal of this step is, to figure out as much as possible about the organizations behind the hosts in the different categories. This is a step, that we had to perform multiple times throughout our research in order to learn about the differences in organizations between the different classifications.

3.4.1 Pyasn

Before we can start any further AS analysis, we first have to figure out to which AS a host belongs to. Pyasn [3] is our tool of choice for mapping each IP to one Autonomous System Number, i.e. mapping each host to one organization.

3.4.2 PeeringDB

PerringDB [21] is our main source for AS data acquisition. While we do not use their CLI for more than downloading their entire database to our scanning setup, it still served its purpose. With the database at our local machine, we were able to query large amounts of ASNs to gather additional data. We queried

PeeringDB for the Name, Location and Network Type of all ASes, which we retrieved from Pyasn.

3.4.3 Caida IXPs

The last piece of information for the AS Analysis delivers ASRanks Caida [4]. We want to assess the popularity of ASes for each distinguished group. Caida IXPs allows us to quickly research the Rankings for our ASes from Pyasn without any passive data or measurements on our side.

3.5 HTTP Curl

The last tool, HTTP Curl, in our tool chain is for measuring the performance. We send an HTTP GET request to IPs that we classified earlier in the tool chain. The requests were performed multiple times for MPTCP enabled and MPTCP disabled on an AWS machine in Frankfurt (Germany). This allows us to analyze the different timings it took for our machine, when MPTCP was enabled and when a regular TCP connection was used.

4 Results

4.1 MPTCP Availability with ZMAP Analysis

We start our analysis by recreating the measurement setup of the previous research on MPTCP deployment [18]. In the following sections we first evaluate port 80 and port 443 ZMAP scans on IPv4 and IPv6. In those sections we try to provide an insight to the current deployment situation of MPTCP exclusively with ZMAP results. We present the organizations behind hosts that we found with ZMAP, which supposedly support MPTCP, and showcase why ZMAP is not an accurate tool for correctly identifying MPTCP support over the Internet.

4.1.1 IPv4 Adoption

Table 4.1 summarizes the output of all of our scans in the IPv4 space. We first examine our results on port 80. The first approach to classify a host as MPTCP capable was to examine the TCP options of a host and check for the MP_CAPABLE option. We only perform a ZMAP scan with the first two packets in figure 2.1. The first packet is our forged probe packet from ZMAP. This packet carries an MP_CAPABLE option to notify the server that we want to initiate a MPTCP connection. Afterwards, we wait until the server responds with the second packet of figure 2.1. We could assume that the server is supporting MPTCP, if we receive a reply that carries an MP_CAPABLE option. If we

Label	Date	Targets	MP Capable	Diff. Sender's Key	Port
A	06.07.2020	64,464,026	192,740 (0.30%)	3,642 (0.006%)	80
B	13.07.2020	64,256,224	194,032 (0.30%)	3,712 (0.006%)	80
C	16.07.2020	63,954,786	179,578 (0.28%)	3,741 (0.006%)	80
D	10.08.2020	58,807,697	201,687 (0.34%)	4,158 (0.007%)	80
X	06.08.2020	47,898,813	211,143 (0.44%)	36,623 (0.076%)	443
Y	27.08.2020	42,490,650	198,170 (0.47%)	33,666 (0.079%)	443

Table 4.1: Results of ZMAP Scans on IPv4

classify a host as MPTCP capable by only checking the MPTCP capable flag as the previous MPTCP deployment paper [18] did, we would now define ∼193K (∼0.30%) as truly MPTCP capable of scan 'A' (table 4.1). If we compare our results of MPTCP capable options with the authors of [18] in 2015, we could say that the MPTCP capable option has increased from 0.08% to around 0.30% from 2015 to 2020. But this is an unfair comparison as the authors only performed ZMAP scans on the Alexa Top 1M list.

Table 4.1 also shows that most of these hosts are unlikely to support MPTCP. For example, if we take a closer look at scan 'A' in table 4.1, we can see that from those ∼193K hosts, only ∼4K hosts reply with a MPTCP key different from the key we initially sent them. The probability that a MPTCP server generates the exact same 64-bit key as the client is quite low. It begs the question whether one can tell if a host is MPTCP capable if the MPTCP capable flag is not accurate? After some time an erratum for the former paper was published and a new way to "correctly distinguish between real Multipath TCP enabled servers and middleboxes that echo Multipath TCP options" [7] was found. This method for true MPTCP support is a check for both, did the host reply with an MPTCP capable flag and did it send a different sender's key.

With that method in mind, we examine our data from scan 'A' in table 4.1 once again. ∼64M targets replied to our probe, ∼193K are MPTCP capable and out of those ∼193K, ∼4K reply with their own MPTCP Key. This approach simply filters middleboxes, that mirror TCP options. It turns out that most replies with an MP_CAPABLE option originate from a middlebox that mirrors TCP options. This method shows surprising results for port 443 scans. While the amount of replying targets is generally higher on port 80, we can see that the amount of hosts with a MPTCP capable flag almost stay the same for port 80 and port 443. *ZMAP measurements would indicate that hosts reachable on port 443 are more likely to support MPTCP.* If we take the amount of hosts that sent us a different sender's key as MPTCP capable hosts, we can see almost 10 times more MPTCP capable hosts on port 443 than on port 80. This also implies that the amount of encountered middleboxes, which mirror our MPTCP options, is considerably lower on port 443. We will see later whether MPTCP hosts actually favor port 443 on this scale.

We now analyze MPTCP capable hosts throughout our IPv4 scans. Figure 4.1 visualizes the amount of hosts gained each IPv4 scan for each port. Port 80 scans have a steady growth of hosts, which reply with a different sender's key. While the focus of this thesis is not on the growth of MPTCP deployment, we observe it slightly in our measurements. The number of hosts responding to our

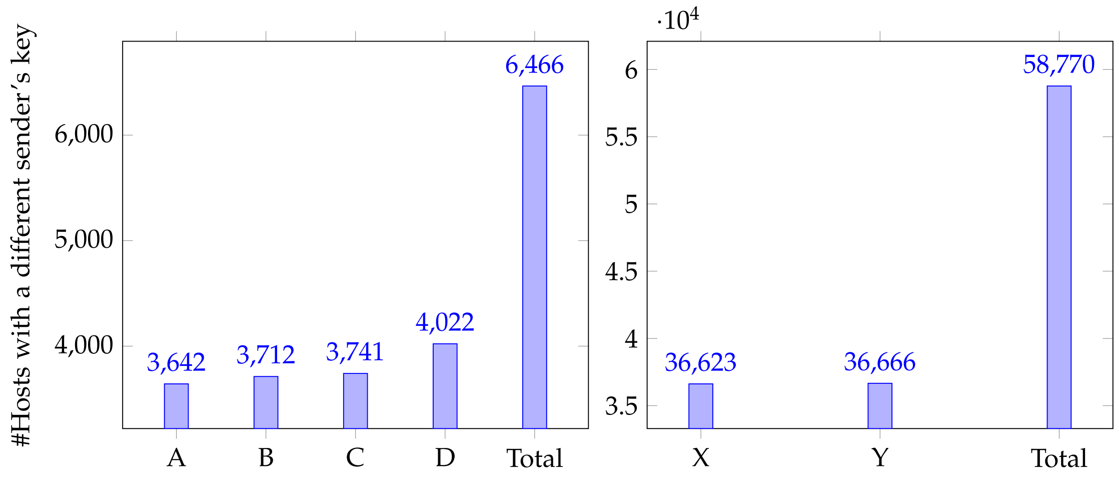


Figure 4.1: Overview of Hosts with a different Sender's Key on all IPv4 ZMAP Scans

scans is steadily decreasing, which is likely due to organizations blacklisting our vantage point as wrongly marking it as an attacker. However, within roughly one month we can observe a growth of about 400 hosts with a different sender's key on our port 80 scans. This means that the MPTCP deployment is making an observable progress, at least if we look at the current classification method. For port 443 we do not regard growth as we lack datasets. Figure 4.1 also shows that we constantly discover new MPTCP servers. We detected a total of 6,466 hosts with a different sender's key on port 80 and another 58,770 on port 443. The reason for the low share between our two port 443 scans will be explained later in this section.

AS Mapping Table 4.2 reveals the top 10 ASes, which hold hosts that replied to us with a different sender's key and the MPTCP capable option, on any of our port 80 scans. That 'Apple CDN' is the leading AS here, gives us some confidence in what we are doing, as Apple is known for deploying MPTCP in various fields e.g. for creating backups and for Siri services [2]. Location wise, we observe that ASes housing MPTCP servers are spread around the world. However, the location of an AS does not show the location of the hosts that replied to us and there is a large share of hosts not listed in this top 10. For those reasons, we postpone the discussion about the location to section 4.2. Another interesting find in table 4.2 is the network type of each AS. The Autonomous Systems within the top 10 either provide content or internet service. It makes

Amount	ASN	Name	Rank	Location	Network Type
1,451	6185	Apple CDN	13577	US	Content
673	61157	PlusServer GmbH	1368	DE	Content
234	714	Apple Inc.	6330	US	Content
213	7922	Comcast	27	US	Cable/DSL/ISP
204	5607	Sky Broadband	5033	UK	Cable/DSL/ISP
203	4766	KT Corporation (Korea Telecom)	47	KR	Cable/DSL/ISP
196	20115	Charter Communications	98	US	Cable/DSL/ISP
119	8551	Bezeq International	249	IL	Cable/DSL/ISP
81	3462	HiNet	255	TW	Cable/DSL/ISP
77	3216	VEON Group	23	RU	NSP
3055	-	Others	-	-	-

Table 4.2: AS Mapping for Port 80 with a different sender's key on IPv4

sense for the top three ASes to serve content, due to the bandwidth aggregation advantages of MPTCP [23].

Another interesting column is the rank for each AS, which we collect from Caida IXPs [4]. However, these ranks are not determined "by traffic, revenue, users, or any other non-topological metric" [4] — instead they are calculated with the following algorithm: "The customer cone of AS X is defined as the set of ASes that X can reach using p2c links" [15]. While this should give us a rough idea about the popularity of the AS, we still should take the AS ranks with caution. Nevertheless, many of the ASes here are highly ranked ASes like 'Comcast' on Rank 27 or 'VEON Group' on Rank 23. Surprisingly the ASes of Apple are on the lower spectrum of Caida's AS ranking.

As shown in table 4.3, most hosts with a different sender's key originate either from 'A1 Telekom Austria AG' with 33,288 hosts or from 'XS4ALL Internet B.V.' with 1,236 hosts. This may raise some doubts about the accuracy of the current agreed method on determining MPTCP capable hosts. Beside those first two, we can see that port 80 and port 443 show six common autonomous systems. This means that it is likely that our target MPTCP hosts, support both HTTP and HTTPS. However, 'Apple CDN', 'Sky Broadcast', 'Bezeq International' and 'VEON Group' from port 80 do not reoccur in this top 10. 'Apple CDN' reached the 27th place with 79 hosts, 'Bezeq International' dropped to the 65th place with 23 hosts and 'VEON Group' managed to still place 24th with 87 hosts.

Amount	ASN	Name	Rank	Location	Network Type
33,288	8447	A1 Telekom Austria AG	132	AT	Cable/DSL/ISP
1,236	3265	XS4ALL Internet B.V.	2223	NL	Cable/DSL/ISP
557	61157	PlusServer GmbH	1368	DE	Content
421	7922	Comcast	27	US	Cable/DSL/ISP
403	714	Apple Inc.	6630	US	Content
363	4760	HKT Limited	2885	HK	Cable/DSL/ISP
340	4766	KT Corporation (Korea Telecom)	47	KR	Cable/DSL/ISP
330	1136	KPN-Netco	805	NL	NSP
307	3462	HiNet	255	TW	Cable/DSL/ISP
276	20115	Charter Communications	98	US	Cable/DSL/ISP
21,249	-	Others	-	-	-

Table 4.3: AS Mapping for Port 443 with a different Sender's Key on IPv4

4.1.2 IPv6 Adoption

While there already is a study about the deployment of MPTCP in the wild for IPv4 dating back almost 5 years ago, no other study has been carried out for the IPv6 space. So the data and resulting analysis presented in this thesis provide new insights about the deployment status in the IPv6 space. As already mentioned in chapter 3, this data does not originate from us but from an almost identical setup.

Like in IPv4, we first take a closer look at one specific scan 'a' from table 4.4. We see that this port 80 scan has completed for 883K unique targets. Out of these 883K only 122 targets replied with an MPTCP capable option. While the number of MPTCP capable hosts on IPv6 is only 0.014%, we have a surprising 0.005% that sent us a different sender's key. So only ∼64% of those hosts, which reply with an MPTCP capable flag, there affected by a middlebox that just

Label	Date	Targets	MP Capable	Diff. Sender's Key	Port
a	09.08.2020	883,058	122 (0.014%)	40 (0.004%)	80
b	19.08.2020	884,402	183 (0.021%)	102 (0.012%)	80
c	21.08.2020	886,471	193 (0.022%)	105 (0.012%)	80
d	03.09.2020	1,149,771	286 (0.025%)	185 (0.016%)	80
x	06.08.2020	681,861	315 (0.046%)	239 (0.035%)	443
y	27.08.2020	897,102	848 (0.095%)	762 (0.085%)	443

Table 4.4: Results of ZMAP scans on IPv6 [13]

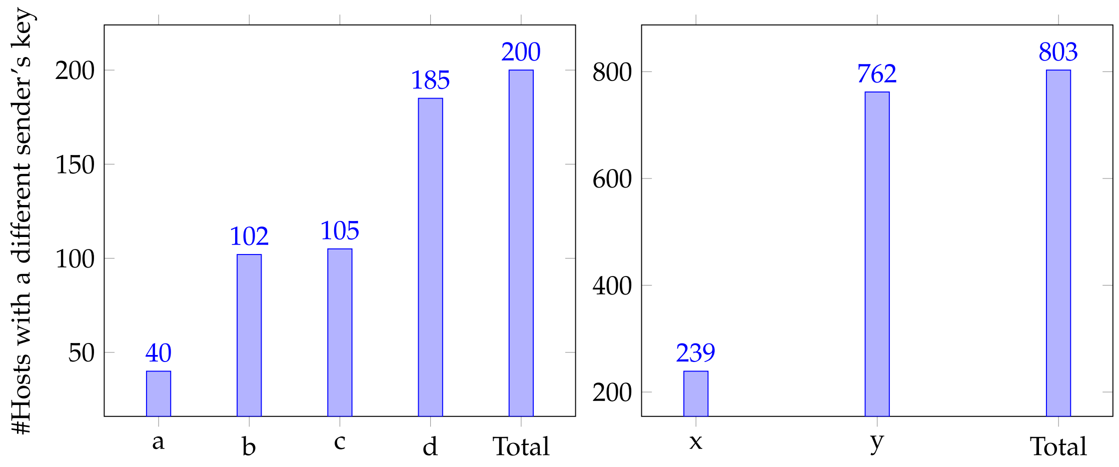


Figure 4.2: Overview of Hosts with a different Sender's Key on all IPv6 ZMAP Scans

mirrors TCP options.

The targets in table 4.4 for IPv6 for both port 80 and port 443 should not change too much as the input for ZMAP is, as described in chapter 3, more or less static. This means that table 4.4 also includes the changes that the targets applied during the beginning and end of the scanning period. We now examine the number of hosts with a different sender's key throughout our scans. Figure 4.2 shows an ongoing growth in MPTCP deployment in the IPv6 space. From mere 40 hosts with a different sender's key in their TCP options to 185 hosts within less than a month shows a remarkable development in adoption of MPTCP. IPv6 scans on port 443 also indicate a trend towards supporting MPTCP. We start with 239 MPTCP hosts on the beginning of August and end up with 762 at the end of August.

AS Mapping We now turn our attention towards analyzing the organizations behind those MPTCP capable hosts. Table 4.5 shows the top 10 ASes of hosts that reply with a different sender's key. 'Apple CDN' makes up 70% of all MPTCP capable hosts in these scans. Surprisingly the second most important AS only holds 7 hosts that support MPTCP. Thus, Apple is the only big fish for MPTCP capable hosts in IPv6 and furthermore, a lot of ASes support MPTCP, just on a smaller scale. However, this result could also be due to the IPv6 Hitlist leaning towards Apple hosts. We also want to highlight two ASes in this table: 'OVHcloud' and 'Linode AS63949', which are known for their cloud hosting

Amount	ASN	Name	Rank	Location	Network Type
140	6185	Apple CDN	13577	US	Content
7	63949	Linode AS63949	6704	US	Content
7	16276	OVHcloud	3634	FR	Content
4	7922	Comcast	27	US	Cable/DSL/ISP
4	12876	Scaleway	10842	FR	Content
3	5607	Sky Broadband	5033	UK	Cable/DSL/ISP
2	9808	Guangdong Mobile Communication Co. Ltd.	309	CH	Not Disclosed
2	4134	China Telecom	102	CH	NSP
2	40428	Pandora Media, Inc	23211	Content	
2	3320	Deutsche Telekom	19	DE	NSP
27	-	Others	-	-	-

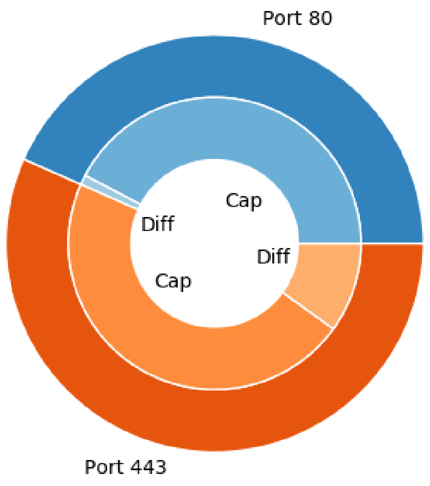
Table 4.5: AS Mapping for Port 80 with a different Sender's Key on IPv6

Amount	ASN	Name	Rank	Location	Network Type
404	8447	A1 Telekom Austria AG	132	AT	Cable/DSL/ISP
227	3265	XS4ALL Internet B.V.	2223	NL	Cable/DSL/ISP
78	6185	Apple CDN	13577	US	Content
16	63949	Linode AS63949	6704	US	Content
12	8767	M-net Telekommunikations GmbH	8767	DE	Cable/DSL/ISP
6	16276	OVHcloud	3634	FR	Content
5	51184	fonira Telekom	5684	AT	Cable/DSL/ISP
5	1136	KPN-Netco	805	NL	NSP
4	7922	Comcast	27	US	Cable/DSL/ISP
4	5607	Sky Broadband	5033	UK	Cable/DSL/ISP
42	-	Others	-	-	-

Table 4.6: AS Mapping for Port 443 with a different Sender's Key on IPv6

services. The high share of MPTCP support on these ASes could be due to us scanning hosts, there their customers decided to install the custom MPTCP kernel on. While bigger companies often take precautions in their firewalls to restrict external scanning, VMs should focus on reachability for their users and are therefore more likely to respond to us.

The same data for port 443 is shown in table 4.6. Once again 'A1 Telekom Austria AG' and 'XS4ALL Internet B.V.' hold most of the MPTCP capable hosts. In third place is the AS 'Apple CDN' with 78 hosts. The percentage of Apple's share is still lower on port 443 (45%) than on port 80 (70%) even if we eliminate the first two ISP ASes. Other ASes reappear with even more hosts than on port 80, like 'Linode AS63949' as it holds 7 hosts on port 80 and now 16 hosts on port 443. While the distribution is dominated by the first two Internet Service



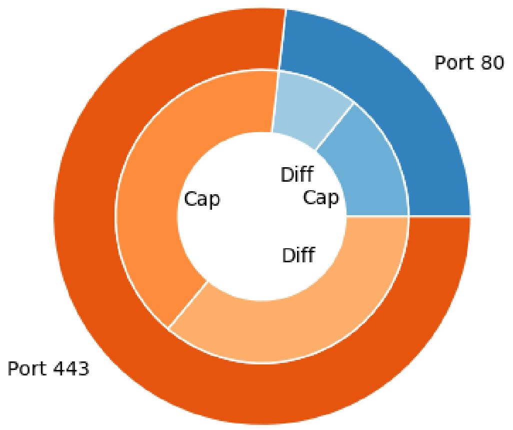


Figure 4.3: Distribution of Hosts with a mirrored Key on Ports and IPv4 (left) and IPv6 (right)

Providers, we observe a flatter curve on the hosts followed by those.

IPv4 vs IPv6

One last question in this ZMAP chapter remains: How does the deployment of MPTCP compare on IPv4 with the current state in IPv6? For that we first examine figure 4.3. The orange section of the circles represent hosts from port 443, while the blue sections depict results from port 80. We first examine the distribution for HTTP and HTTPS support on IPv4 and IPv6. The amount of IPv4 hosts, which reply with an MPTCP capable option, is almost identical on port 80 and port 443. IPv6 on the other hand shows that port 443 clearly dominates the MPTCP capable option. Around 3/4 of the entire IPv6 hosts with a MPTCP capable option originate from a port 443 scan.

The amount of encountered middleboxes (inferred by hosts which replay the MPTCP key sent by the vantage point) is noticeably lower on IPv6 for both ports. However, these middleboxes seem to prefer port 80, as the percentages of hosts, that reply with our sender's key, is generally higher on port 80. The tiny share of port 80 IPv4 hosts, that reply with a different sender's key, also becomes apparent in the left pie chart. Overall, we see a MPTCP trend in port 443 for IPv6 and a trend in middleboxes that mirror our MPTCP key in IPv4.

If we look back to the tables (4.2, 4.3, 4.5 and 4.6) with the different ASes, we

Targets	MP Capable	Diff. Sender's Key	Port	Version
74,246,491	250,356	6,466	80	IPv4
52,478,964	276,250	58,770	443	IPv4
1,195,331	316	200	80	IPv6
934,239	907	803	443	IPv6

Table 4.7: Superset Overview of ZMAP Scans for IPv4 and IPv6

find a lot of ASes in common. 'Apple CDN' dominates both IPv4 and IPv6 AS port 80 lists. Some leading ASes like the 'PlusServer GmbH' on IPv4 port 80 are completely missing on IPv6 port 80. Furthermore, the drop of dominance of the top ASes is higher on IPv6 port 80 than on IPv4 port 80. The top two ASes, the Internet Service Providers 'A1 Telekom Austria AG' and 'XS4ALL Internet B.V.', stay on top of both port 443 AS lists. Otherwise, both lists share again many ASes in common. The last thing to point out in this section is, that we collected more data in the IPv4 space than we received on the IPv6 space. If we take the two scans that there performed on time slots close to each other, the scans 'D' (from table 4.1) and 'd' (from table 4.4) for example. We see that our IPv4 scan 'D' collected data from 64M hosts, while the IPv6 data only originates from 1M hosts. For this reason, we only investigate our measurements over the IPv4 space in the following sections.

4.2 Diving deeper into the scans

The previous section provides a general understanding of the status of MPTCP deployment. However, while analyzing we doubted the correctness of the results obtained from ZMAP. Especially at port 443, there we found ISPs that dominate the AS tables 4.3 and 4.6. Furthermore, since a large percentage of the ZMAP MPTCP capable servers do not support MPTCP considering they replayed our key, we wanted to shed some light on the following questions.

1. Do servers, which sent us a different key, truly support MPTCP?
2. Who were these middleboxes that intercept MPTCP packet headers and add their own on user traffic?

In this section, we investigate an answer to the first question. For a more detailed look into the origins of ZMAP TCP options we use Tracebox [26] - a tool that allows us to see changes for each hop in TCP options from our vantage

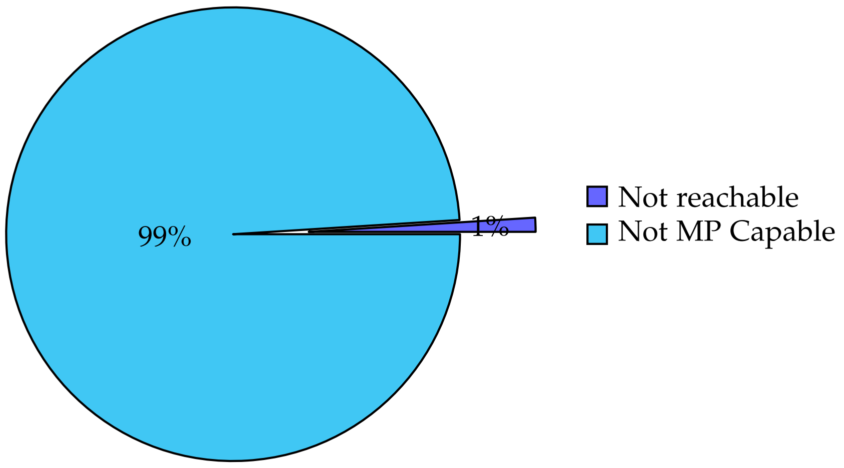


Figure 4.4: Tracebox Results of Hosts with the same Sender's Key and an MP_CAPABLE Option

point till the end host. We probe important hosts from ZMAP results to learn more about the middleboxes interfering with MPTCP options and to verify that the MPTCP options originated from the end host.

Before we start to examine the supposed MPTCP capable hosts from ZMAP, we first analyze what output we get in Tracebox for machines, that sent us a MPTCP capable flag and the same MPTCP Key as we sent them. Figure 4.4 proves the theory that a host with the same sender's key does not truly support MPTCP [7]. *We have not a single MPTCP capable flag in the last hop of the Tracebox results for these hosts.* The Tracebox tool discovers that 99% of the end hosts have no MPTCP capable flag whatsoever and the other 1% did not respond within 64 hops. It clearly states that most hosts classified by the previous investigative study about analyzing support for MPTCP over the internet [18] are not MPTCP capable at all.

We now proceed with hosts there ZMAP has reached its capabilities. Hosts that replied to us not only with an MPTCP capable option but also with a different MPTCP sender's key. On the left side of figure 4.5, we see the Tracebox results for port 80 and the right pie chart depicts the results for port 443 of the supposedly MPTCP capable machines. First we take a closer look at the port 80 results on the left. 3,708 hosts of the ZMAP results indeed are truly MPTCP capable machines. However, we also got 45 hosts that were affected by a middlebox close to the end host. Another 155 hosts were not MPTCP capable,

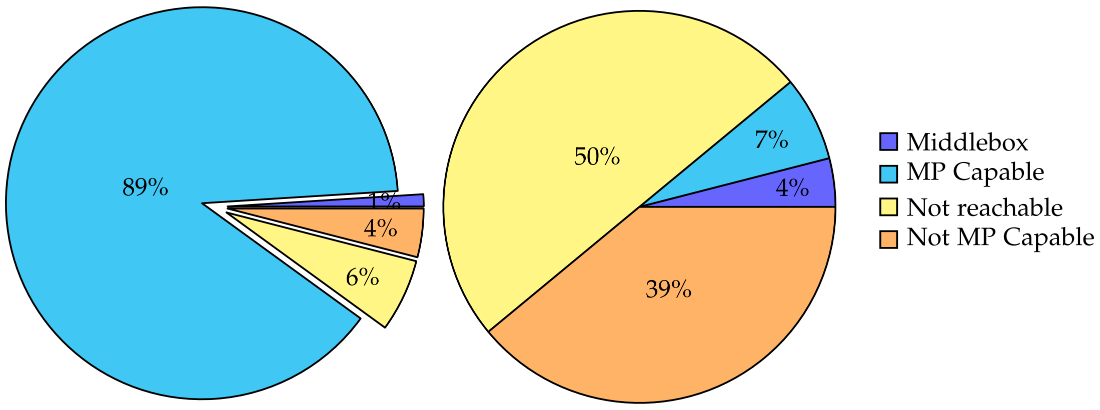


Figure 4.5: Tracebox Results of ZMAP MPTCP capable Hosts on Port 80 (left) and Port 443 (right)

so they were also affected by some kind of middlebox in the path between them and our scanning machine. For the remaining 250 hosts we got no results, as we could not reach the desired target host within 64 hops. After all, we observe that ZMAP gives us a few false-positives on port 80. But while the current method is not completely accurate, the 5% that were identified incorrectly are probably acceptable.

We now turn our attention towards the rather suspicious results of port 443. The right side of Figure 4.5 looks quite different from the results of port 80 on the left side. Sadly 16,844 targets were not reachable anymore, and we can therefore not determine whether they support MPTCP or not. We decided to not include them in our data set of truly MPTCP capable hosts. 1,263 hosts fall victim to a middlebox near their end host. 13,040 hosts replied to Tracebox that they do not support MPTCP at all. They sent us an MPTCP capable option and a different MPTCP key in ZMAP, still Tracebox reports a big portion of them as non MPTCP capable hosts. Only 2,519 of the initial 33,666 MPTCP classified hosts truly support MPTCP. *These results prove that the ZMAP results of port 443 were considerably misclassified.*

We now take a look at the location of these truly MPTCP capable hosts. Figure 4.6 helps us to visualize the whereabouts of our Tracebox results. Note that we do not show the location of the AS connected to a host, but the host itself by issuing a Geoiplookup [17] for each of those hosts. We end up with a map that unsurprisingly favors the US, as Apple is based in the US. After Germany and

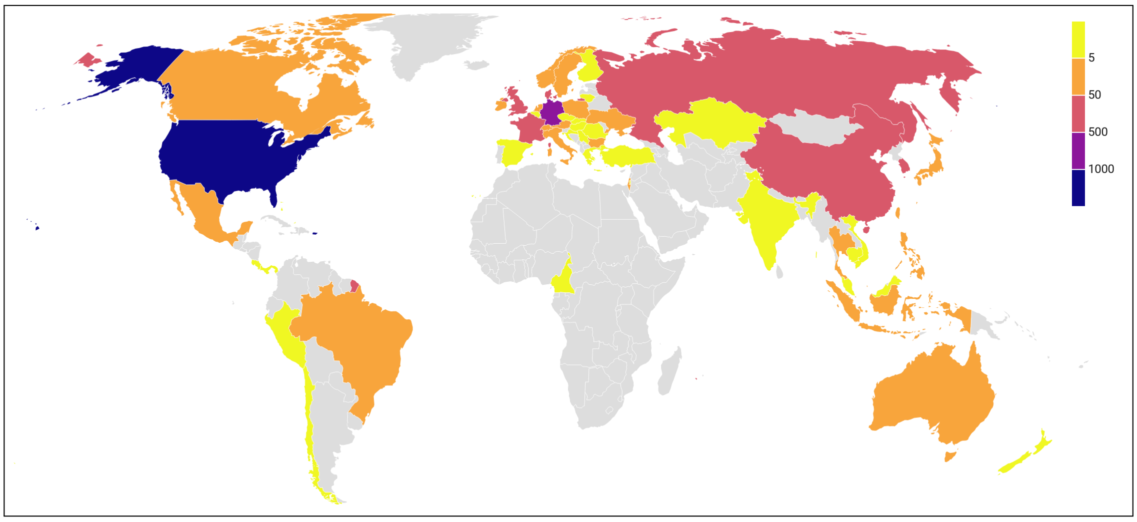


Figure 4.6: Location of our proposed truly MPTCP capable Hosts on IPv4 [10]

Amount	ASN	Name	Rank	Location	Network Type
673	61157	PlusServer GmbH	1368	DE	Content
198	7922	Comcast	27	US	Cable/DSL/ISP
194	4766	KT Corporation (Korea Telecom)	47	KR	Cable/DSL/ISP
182	714	Apple Inc.	6330	US	Content
182	20115	Charter Communications	98	US	Cable/DSL/ISP
78	6185	Apple CDN	13577	US	Content
76	3216	VEON Group	23	RU	NSP
70	5607	Sky Broadband	5033	UK	Cable/DSL/ISP
68	786	Janet	804	UK	Educational/Research
66	14265	TelePacific Communications	618	US	NSP
1921	-	Others	-	-	-

Table 4.8: Truly MPTCP Capable Hosts of Scan 'D' on Port 80

the US the drop to the next highest MPTCP capable country is quite high. But we see that we could scan MPTCP hosts from almost all over the world.

Table 4.8 denotes the ASes of our truly MPTCP capable hosts on port 80. These are ASes, which replied to ZMAP and Tracebox in the last hop with an MP_CAPABLE option and a different sender's key. If we compare this table to table 4.2, we can derive the ASes that there connected to hosts, that either were influenced by a middlebox or went out of our reach. The previous table 4.2 describes results from all scans, while this table only yields results from scan 'D' (table 4.1). We now discuss the organizations behind truly MPTCP capable

Amount	ASN	Name	Rank	Location	Network Type
497	61157	PlusServer GmbH	1368	DE	Content
156	4766	KT Corporation (Korea Telecom)	47	KR	Cable/DSL/ISP
151	7922	Comcast	27	US	Cable/DSL/ISP
147	20115	Charter Communications	98	US	Cable/DSL/ISP
101	714	Apple Inc.	6630	US	Content
77	6185	Apple CDN	13577	US	Content
50	14265	TelePacific Communications	618	US	NSP
47	20001	Charter Communications (20001)	308	US	Cable/DSL/ISP
44	786	Janet	804	UK	Educational/Research
43	3216	VEON Group	23	RU	NSP
1206	-	Others	-	-	-

Table 4.9: Truly MPTCP Capable Hosts of Scan 'Y' on Port 443

hosts. 'PlusServer GmbH' is the first on the list and makes up 18% of hosts that support MPTCP on port 80. While 'Apple Inc.' only dropped from 234 in the ZMAP table to 182 in the Tracebox table, 'Apple CDN' dropped from 1,451 hosts to 78 hosts. This huge drop can have multiple reasons. The first and most obvious is that some hosts of 'Apple CDN' do not truly support MPTCP and were affected by a middlebox. Another reason for a lower share could be that scan 'D' simply has not received many replies from 'Apple CDN' and they instead resulted from another scan. And last but not least, hosts from 'Apple CDN' could have been in the 6% from figure 4.5, and we just could not reach them with our Tracebox measurements.

Table 4.9 summarizes the ASes behind the truly MPTCP capable hosts found with Tracebox. Figure 4.5 shows that the classification method on ZMAP scans for port 443 there not accurate at all. Unsurprisingly 'A1 Telekom Austria AG' and 'XS4All Internet B.V.', which dominated the former AS table 4.3, have vanished from the top 10. Like in the previous port 80 table 4.2, the 'PlusServer GmbH' AS still stays on top. But unlike the drop of 'Apple CDN' on port 80, it first appeared in this AS table for port 443. 'Apple Inc.' fell a few spots though. Overall we see a lot of familiar ASes like 'VEON Group' or 'Comcast' that already appeared on previous tables. We are satisfied with this table as it shows most of the assumed ASes that hold MPTCP capable hosts, while it filters ASes like 'A1 Telekom Austria AG' that supposedly do not truly support MPTCP.

IP	Appearance	Location	ASN	AS Name	Network Type
59.46.60.146	133	CN	4134	China Telecom	NSP
62.216.144.122	3	GB	15412	Global Cloud Xchange (f.k.a. FLAG Telecom)	NSP
93.125.20.68	2	BY	60330	Belarusian Cloud Technologies	Not Disclosed
122.96.66.98	2	CN	4837	China Unicom	NSP
221.183.55.86	1	CN	9808	Guangdong Mobile Communication Co. Ltd.	Not Disclosed
183.221.156.178	1	CN	9808	Guangdong Mobile Communication Co. Ltd.	Not Disclosed
176.52.248.123	1	ES	12956	Telxius Cable	Cable/DSL/ISP
94.142.98.240	1	US	12956	Telxius Cable	Cable/DSL/ISP
221.6.9.134	1	CN	4837	China Unicom	NSP
93.125.20.67	1	BY	60330	Belarusian Cloud Technologies	Not Disclosed

Table 4.10: MPTCP Option Changes in a Tracebox Port 80 Scan for same Sender's Key

4.3 Encountered middleboxes

From figure 4.4 and figure 4.5, it is evident that middleboxes severely affect the outcome of our ZMAP scans. In this section, we analyze middleboxes that temper with our sent MPTCP options before they reach the target host. As previously discussed, Tracebox works by sending packets with increasing TTL to the target until reached. Tracebox then waits for the time-exceeded messages to get back and hopefully the router replied with an ICMP that contains an IP header. Tracebox then "compare(s) the quoted packet sent back in an ICMP time-exceeded by an intermediate router with the original one" [11]. This means that if a network device replies with an ICMP that quotes the entire IP packet, Tracebox can proceed to analyze the differences between the packet we sent and the replied packet. In this section, we examine those middleboxes that there not detected in the last hop (near the target server) but somewhere in the path to it.

During the research period, we issued a total of three Tracebox scans. Two for the latest port 80 scan 'D' (table 4.1), once for IPs that replied with the same sender's key and MP_CAPABLE and once for hosts with a different sender's key and MP_CAPABLE. The last scan was issued for all hosts in port 443 scan 'Y' (table 4.1) that sent us a different sender's key in ZMAP. Surprisingly we did not notice any accumulations of hops right after or at the middlebox in the Tracebox scans for hosts with a different sender's key. We expected to see an accumulation at the port 443 scan, as it was heavily affected by middleboxes in Austria. However, it appears that there was no close router that replies with an ICMP packet to those middleboxes. The port 80 scan with the same sender's key showed us an accumulation at one specific IP. Table 4.10 shows that of 197,529 Tracebox results in this scan 146 paths were affected by middleboxes, which were not detected within the last hop. 133 paths were affected by a single

Listing 4.1: Tracebox Extract with a Middlebox Hop

```
tracebox to 201.187.168.166 (201.187.168.166): 64 hops max
1: 131.159.25.253 0ms
2: 131.159.252.6 0ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
   IP::DestinationIP
3: 129.187.0.149 0ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
4: 188.1.37.89 1ms IP::TTL IP::CheckSum
5: 188.1.145.229 9ms IP::TTL IP::CheckSum
6: 213.248.97.40 9ms IP::TTL IP::CheckSum
7: 62.115.37.19 9ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
8: 62.115.120.213 2ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
   IP::DestinationIP
9: 159.226.254.46 62ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
   IP::DestinationIP
10: 195.219.87.169 59ms TCP::CheckSum IP::TTL IP::CheckSum
    IP::DestinationIP [Extra headers: ICMPExtension
    ICMPExtensionObject ICMPExtensionMPLS ]
11: 101.4.117.121 68ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
    IP::DestinationIP
12: 59.46.60.146 14ms -TCP IP::TotalLength IP::TTL IP::CheckSum
    IP::DestinationIP -TCPOptionMaxSegSize -TCPOptionPad
    -TCPOptionMPTCPCapable -TCPOptionWindowScale +RawLayer
13: 213.140.39.161 17ms [PARTIAL] +PartialTCP IP::DiffServicesCP
    IP::TTL IP::CheckSum
14: 186.148.63.46 82ms [PARTIAL] +PartialTCP IP::TTL IP::CheckSum
15: *
16: 117.253.201.222 9164ms [PARTIAL] +PartialTCP IP::TTL
    IP::CheckSum IP::DestinationIP
17: 201.187.168.166 237ms TCP::SrcPort TCP::DstPort TCP::SeqNumber
    TCP::AckNumber TCP::Flags TCP::WindowsSize
    TCP::CheckSum IP::TTL IP::CheckSum IP::SourceIP IP::DestinationIP
    -TCPOptionWindowScale
```

middlebox. 59.46.60.146 is either the middlebox, which changes MPTCP options or there is a middlebox, that messes with our MPTCP options, right in front of this IP. Geoiplookup maps the location of this IP to China, so the middlebox is somewhere between the routing of Germany (our vantage point) and China. Furthermore, this IP did not appear in any hop of the port 80 Tracebox scan results with a different sender's key.

The listing 4.1 shows a typical Tracebox result for a ZMAP host that replied with the same sender's key and the MP_CAPABLE option set, there this IP was along its path. At hop 12 we identify this IP and also the removal of the MP_CAPABLE option. We can not determine where exactly the option was removed due to the previous hop (11) not replying with the IP header of the packet. However, we can restrict it to somewhere in between hop 10 (EU), hop 11 (CN) and hop 12 (CN). Other results, in which this IP appears, only show Chinese hops before the MP_CAPABLE option is removed. This indicates that the middlebox around this IP is indeed located in China. For further information about this hop, like the operating system, we issued an NMAP [16] towards that IP and listed the results in the Appendix (chapter 6).

4.4 Effect of the middleboxes on the performance

While in the previous sections, we investigate the adoption of the MPTCP protocol in servers from the IPv4 and IPv6 space, in this section, we analyze the quality of user experience if they interact with these servers with and without MPTCP enabled. It has been reported in the past that the user can likely experience longer latencies while connecting to MPTCP capable servers with MPTCP options than with regular TCP [5]. This discrimination's primary reason can be interruptions due to middleboxes on path, which can place MPTCP tagged packets on lower-priority queues or reroute them to the destination via longer parallel paths.

To investigate this, we set up MPTCP capable VMs in AWS cloud located in Frankfurt (Germany). We collect the list of all truly MPTCP capable servers operating on port 80 and port 443 from our analysis in section 4.2. We then initiate HTTP GET requests over curl [24] to all these servers and collect four timing components: connect time, pre-transfer time, time-to-first-byte and total completion time. Figure 4.7 shows the breakup of the time we record while connecting to the servers. HTTP connect refers to time it took to complete HTTP handshake (time_connect) and pre-transfer time is the time it took to exchange

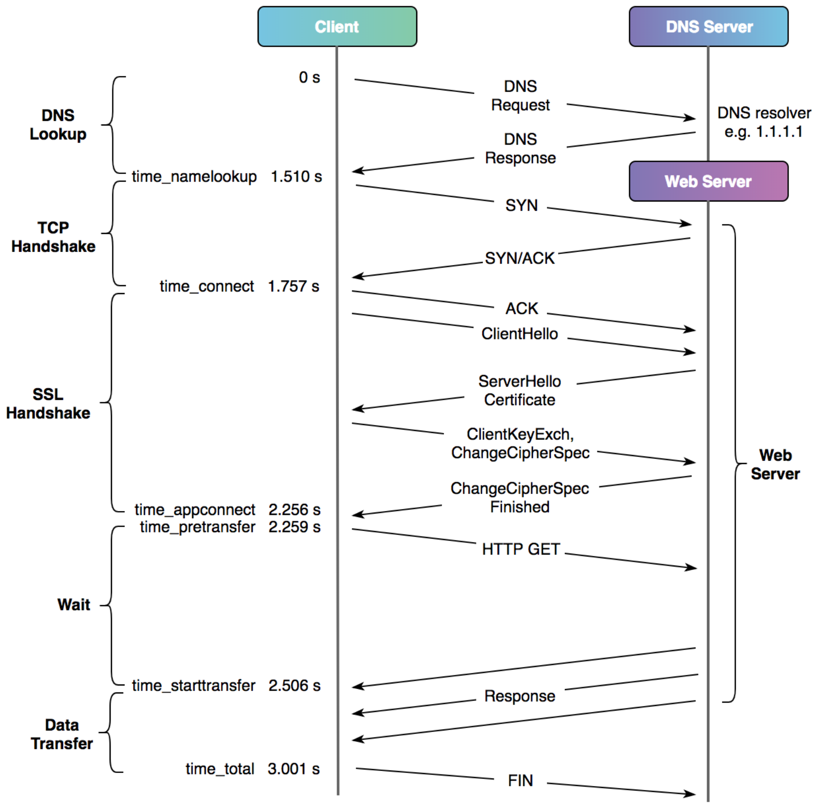
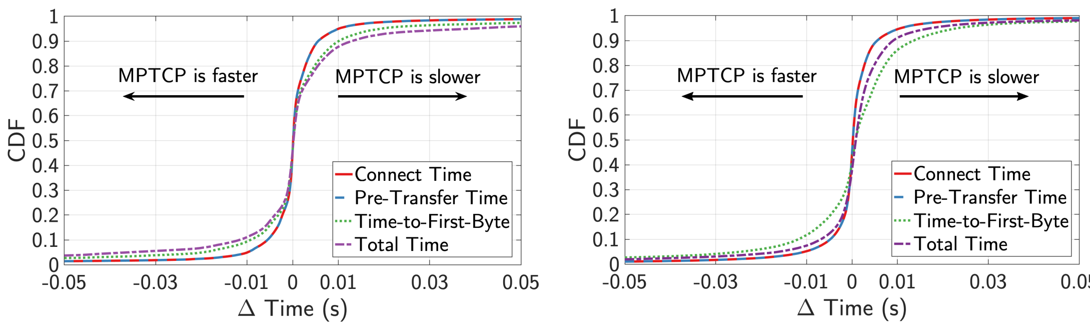


Figure 4.7: Timings of an HTTP GET Request [9]

security keys over SSL with the server (time_pretransfer). Time-to-first-byte is the time when the first packet was received at the VM and total time refers to the time at which FIN packet was sent to the server (time_total). We initiate two parallel tests connecting all servers, one with VM in which MPTCP was disabled (regular TCP) and another in which MPTCP was enabled. Overall, we targeted ∼6,5K endpoints over port 80 and ∼59K servers over port 443. We executed multiple runs of the tests which lasted almost two weeks.

4.4.1 Results

Firstly, we were only able to establish connection with ∼5,2K servers over port 80 (80%) and ∼16K over port 443 (27%). The rest of our connection requests were dropped by middleboxes (mostly firewalls) associated with the servers. For



(a) Port 80 (b) Port 443

Figure 4.8: Performance of HTTP GET Requests to MPTCP Capable Port 80/443 Servers

the rest, we calculate the Δ time difference between connections over MPTCP and over regular TCP for all time components. Figure 4.8 shows the CDF of our results. Distribution leaning towards the left side of the graph indicates that connections over MPTCP were faster than over TCP and a distribution on the right shows vice versa.

From Figure 4.8 it is clear that there is no discernible difference in connections over MPTCP and TCP protocols for both port 80 and port 443 (as evident from quite symmetric distributions on upper and lower half of the plot centered around origin). For almost 90% of the servers, it took the VM almost a similar time to connect and break the connection. For the rest, the differences did not favor any protocol but could be accounted due to changes in network conditions. Interestingly, we can observe that while connecting to servers over port 443, the time-to-first-byte is slightly larger than the total time which might seem illogical. Such a distribution can be explained by servers for which connection timed out (threshold set as 20 seconds) and therefore did not report a total time value but only TTFB. The presence of such servers was higher in port 443 dataset than in port 80.

4.4.2 Connectivity towards servers affected by middleboxes

While the above section shows that the underlying network does not treat application traffic over MPTCP any differently than over TCP, it is also possible that the servers in the dataset above were not affected by middleboxes that alter MPTCP traffic. To understand how meddlesome middleboxes would affect MPTCP traffic, we selected ∼450 servers from our Tracebox analysis

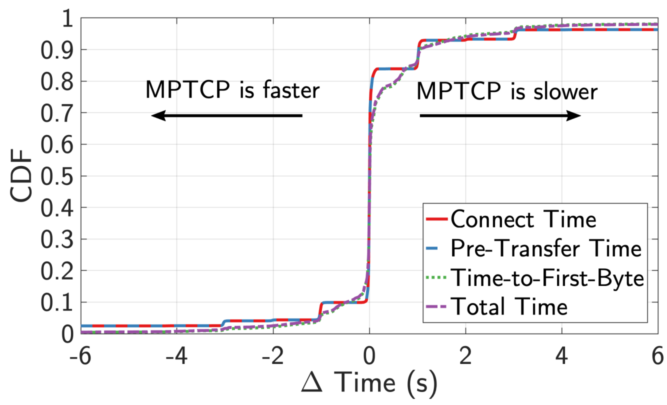


Figure 4.9: Performance of HTTP GET Requests to Hosts affected by Middleboxes

in section 4.2 for which middleboxes on path altered MPTCP options in the packets. Please note that ZMAP marked all these servers as MPTCP capable but Tracebox revealed none of these actually supported MPTCP protocol. We ran similar HTTP GET request test as mentioned above but increased our number of runs to 30. The results are shown in Figure 4.9

4.4.3 Connectivity towards known middleboxes

It can be observed from the figure 4.9 that middleboxes on path to servers clearly treat MPTCP traffic differently than TCP traffic. For almost 30% of the servers, MPTCP enabled connections take longer for all time components whereas only 10% servers saw a longer TCP connect time. Notice the x-axis limit scale difference of figure 4.9 compare to figure 4.8. Middleboxes attached to 10% servers can extend the time-to-first byte by greater than 1 second which can be significantly high for real operations. We further probe middleboxes' impact on MPTCP traffic comparing the time-to-live (TTL) values achieved by Tracebox queries with and without MPTCP options. We notice almost no significant changed in the underlying path carrying MPTCP options compared to TCP. We conclude that these middleboxes probably place MPTCP tagged traffic in lower priority queues which impacts the time packets take to reach the endpoint. While this section simply analyzes the impact of middleboxes on application traffic over MPTCP, further analysis is necessary to understand the type, location and prevalence of these devices at large over the Internet to fully understand the adoption of MPTCP.

5 Conclusion

5.1 Summary

This section answers the research question from the Introduction, Chapter 1.

RQ1: How many HTTP and HTTPS servers already support MPTCP over the Internet?
The total amount of MPTCP capable hosts found with ZMAP on IPv4 add up to 6,466 hosts on port 80 as table 4.7 shows. We issued Tracebox for our latest port 80 scan 'D' (table 4.1), and reviewed the results in section 4.2. Tracebox filters 4,158 hosts down to 3,708 hosts, which we classify as truly MPTCP capable hosts on IPv4 port 80. For port 443 ZMAP reported a total of 58,770 unique IP addresses, that replied with a different sender's key and an MP_CAPABLE option. Scan 'Y' (table 4.1) holds 36,666 hosts with these properties. Out of those 36,666 hosts, only 2,519 hosts turned out to truly support MPTCP.

IPv4 port 80:	*3,708 MPTCP capable hosts*
IPv4 port 443:	*2,519 MPTCP capable hosts*

RQ2: How does MPTCP adoption look like in the IPv6 space?
For IPv6 we only have data from ZMAP, due to the long runtime of Tracebox and our short time period. The results in section 4.1.2 give us nevertheless a rough understanding of the MPTCP deployment situation in IPv6. All analyzed scans tally up to 200 MPTCP capable hosts on port 80. ZMAP reported 803 hosts on port 443 as MPTCP capable. However, 631 of them are very likely to not support MPTCP as these hosts originate from either the AS 'A1 Telekom Austria AG' or 'XS4ALL Internet B.V.'. As these ASes did not survive the Tracebox filter on IPv4, we assume that they would also be filtered in IPv6. This leaves us with 173 MPTCP capable hosts on port 443. We can observe that MPTCP support on IPv6 ($\sim$0.01% of total targets on port 80) is a lot less than on IPv4 ($\sim$0.30% of total targets on port 80).

IPv6 port 80:	*200 MPTCP capable hosts*
IPv6 port 443:	*173 MPTCP capable hosts*

RQ3: How much impact do middleboxes have on MPTCP scanning techniques?
We started with an ZMAP scan, which was heavily affected by different middleboxes. Each scan we had to face ∼190K middleboxes, that mirrored our TCP options. Furthermore, we noticed two ASes on port 443 that together had 34,524 hosts of 58,770 with MPTCP enabled and a different sender's key. In section 4.2 both ASes turned out to be influenced by middleboxes. We could also determine that ∼5% of our results on port 80 and ∼11% on port 443 may not have MPTCP support even though they reply with an MP_CAPABLE option and a different sender's key. In section 4.3 we could even identify one IP in China that was around a middlebox that removed our MPTCP options constantly.

RQ4: What is the performance of MPTCP for application traffic over the Internet?
There are reports that a longer route is chosen if MPTCP is enabled [5]. However, our results in section 4.4 show no perceptible time differences for the user between an HTTP GET Request via regular TCP and MPTCP. While investigating the performance of MPTCP, we noticed that middleboxes that change MPTCP options, influence the performance negatively.

5.2 Limitations and lessons learned

During the course of the research, we identified several pointers, which could be improved in the next iteration of the work. In this section, we enlist all such shortcomings and discuss various lessons learned in the process. The first limitation is due to the location of our single setup. All ZMAP results on IPv4 that were presented during the thesis originate from a single machine located in Munich (Germany). This probably hindered many truly MPTCP capable hosts from being part of our research. Because MPTCP capable hosts that had interfering middleboxes in either the Tracebox path or the ZMAP path are not in our final MPTCP dataset. We therefore can not determine the absolute number of MPTCP capable hosts even if we scan the entire IPv4 space [22].

Another limitation in our research is the time it took for ZMAP/Tracebox scans to finish. These scans took several days to complete and therefore we could not scan more frequently. Furthermore, our IP was often tagged as 'potential attack' on abuseIP DB [25], which disallowed us to run more scans in order to not get blacklisted. This limitation has resulted in non-linear data collection.

During our ZMAP scans we got various emails from multiple companies, who did not want to be part of the research and demanded us to blacklist them for future scans. This is intelligible and we of course expanded our blacklist for them. We also hosted our own webpage at `http://waves.cm.in.tum.de/` and at the IP of the scanning machine informing about our MPTCP measurements and that we do not want to cause any harm. However, at the end of our research period we got an IP range that we did not send any probes towards. Additionally, we assume that many companies blocked our IP on their firewalls instead of explicitly asking for the other way around. Even if one would change the IP of their scanning machine for every single scan into a completely different IP range, there would still be organizations that block one's IP within the first few probes in their IP range. Due to this, we see no practical way to completely get rid of this limitation or even to mitigate the impact any further.

The last limitation is in the middlebox section. "(W)hen (T)racebox receives a Full ICMP, it is able to detect more changes such as the TCP acknowledgement number, TCP window, removal/addition of TCP options, payload modifications, etc." [11] We see this as a limitation due to the large number of middleboxes that did not reply us with a full ICMP and not even with an IP. Hence, we can only determine the approximate number of middleboxes in the path between us and the target. The identification of middleboxes did not work for us as expected and need to be explored in much finer detail in the future.

5.3 Future Work

Almost all limitations mentioned in section 5.2 can be subject of Future Work. While there will never be a large scale scan there no middlebox is encountered, one could reduce the hops with multiple vantage points throughout the world. This would result in a more comprehensive dataset for truly MPTCP capable hosts. Another constraint that the next research should improve on, is the frequency and an extended time period for the scans. Multiple machines that work on different IP ranges would not be difficult to set up. One can then also double these machines and use one half for port 80 and the other for port 443. Scans on a daily or weekly basis over an extended period would be great to verify the growth of MPTCP support. Little is known about the middleboxes that temper with MPTCP options. To identify the middleboxes with Tracebox turned out to be not that easy. This could have many reasons and we look forward to someone who figures out how to identify them correctly.

Another important topic is how real application using MPTCP will be affected by middleboxes. This can be evaluated by interacting with end-hosts using tools like MPScapy [19] to see how additional machinery of MPTCP, e.g. subflow addition/removal, works with MPTCP capable servers over the Internet.

Many questions on the detection of MPTCP capable hosts in both the IPv4 space and the IPv6 space are still open for discussion, and we are looking forward to any further research that will be done in this field.

6 Appendix

6.1 Vantage Points

For all measurements besides the performance measurements, we used the following Linux kernel in Munich (Germany):

```
Linux version 4.19.126.mptcp (root@debian-amd64)
(gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1))
#20200611235134 SMP Thu Jun 11 23:55:29 UTC 2020
```

In combination with the following MPTCP version:

```
MultiPath TCP v.0.95
```

We performed all performance measurements with the following Linux kernel in Frankfurt (Germany):

```
Linux version 4.19.126.mptcp (root@debian-amd64)
(gcc version 6.3.0 20170516 (Debian 6.3.0-18+deb9u1))
#20200611235134 SMP Thu Jun 11 23:55:29 UTC 2020
```

With the same MPTCP version as the previous machine:

```
MultiPath TCP v.0.95
```

6.2 ZMAP

For our ZMAP scans, we used the TCP SYN scan probe module [20] with the following probe arguments:

```
0x020400010402080a0f0f0f0f000000000303001e0c00810c0c0c0c0c0c01
```

saddr	optionshex	optionstext	mptcpkey
145.90.226.129	0x020405b40402080a035535f40f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
181.200.97.25	0x0204000104020101010101010101010101011e0c00810c0c0c0c0c0c0c01	MSS-SACK-N-N-N-N-N-N-N-N-N-N-N-N-N-MPTCP-N-	0x0c0c0c0c0c0c0c0c
72.45.184.172	0x020405b40402080a3988b9c60f0f0f0f01030302	MSS-SACK-TS-N-WS-	–
193.246.9.140	0x020405b40402080a3e8431370f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
24.14.243.146	0x020405b40402080a068cc4270f0f0f0f01030306	MSS-SACK-TS-N-WS-	–
40.67.135.206	0x020405b40402080a947168c20f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
23.7.91.103	0x020405b40402080a05f983ee0f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
172.67.94.82	0x0204057801010402010303 0a	MSS-N-N-SACK-N-WS-	–
23.5.45.152	0x020405b40402080a5235f05a0f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
23.76.243.93	0x020405b40402080a96a97c2f0f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
193.190.135.129	0x020405b40402080a50d6c3b90f0f0f0f01030307	MSS-SACK-TS-N-WS-	–
188.23.185.239	0x020405ac0402080a2619f9d30f0f0f0f010303061e0c0081d20fa49300bda23d	MSS-SACK-TS-N-WS-MPTCP-	0xd20fa49300bda23d

Table 6.1: Extract of important Fields of a ZMAP Scan Result

6.3 Tracebox

For our Tracebox scans we used the latest version of the master branch v0.4.4 [26]. We wrote our own script to utilize threads on an input list of IPs and wrote our own analyzing scripts for the output. For an example output please refer to section 4.3.

6.4 Geoiplookup

Here is the list of countries to the choropleth map in figure 4.6:

Country	Hosts	Percentage
US	1821	45.49%
DE	907	22.66%
KR	210	5.25%
GB	159	3.97%
FR	150	3.74%
RU	106	2.65%
CN	101	2.52%
HK	57	1.42%
DK	55	1.37%
TW	45	1.12%
Other	392	9.80%

Table 6.2: Location of truly MPTCP capable Hosts

6.5 NMAP

Result of an NMAP OS scan towards the Chinese IP from section 4.3:

```
Nmap scan report for 59.46.60.146
Host is up (0.29s latency).
Not shown: 987 closed ports
PORT     STATE    SERVICE        VERSION
21/tcp   open     ftp
| fingerprint-strings:
|   GenericLines, SMBProgNeg:
|     220 RSR7708-SRCMI-X FTP server <version 1.00> ready.
|     Command not implemented.
|   Help:
|     220 RSR7708-SRCMI-X FTP server <version 1.00> ready.
|     214- The following commands are recognized (* =>is unimplemented).
|     214- USER PORT STOR MSAM* RNTO NLST MKD CDUP OPTS*
|     214- PASS PASV APPE* MRSQ* ABOR* SITE* XMKD* XCUP* AUTH*
|     214- ACCT* TYPE MLFL* MRCP* DELE SYST RMD STOU* PBSZ*
|     214- SMNT* STRU* MAIL* ALLO* CWD STAT XRMD* SIZE PROT*
|     214- REIN* MODE* MSND* REST XCWD* HELP PWD MDTM EPRT
|     214- QUIT RETR MSOM* RNFR LIST NOOP* XPWD* FEAT* EPSV
|     Direct comments to ftp-bugs@RSR7708-SRCMI-X.
|   NULL:
|_    220 RSR7708-SRCMI-X FTP server <version 1.00> ready.
22/tcp   open     ssh            (protocol 2.0)
| fingerprint-strings:
|   NULL:
|_    SSH-2.0-RGOS_PK3223
| ssh-hostkey:
|_  512 d7:68:b7:08:34:dd:eb:ff:bc:b2:32:e7:0f:db:fa:4c (DSA)
23/tcp   filtered telnet
25/tcp   filtered smtp
42/tcp   filtered nameserver
80/tcp   filtered http
135/tcp  filtered msrpc
139/tcp  filtered netbios-ssn
443/tcp  filtered https
445/tcp  filtered microsoft-ds
593/tcp  filtered http-rpc-epmap
2009/tcp open     tcpwrapped
8080/tcp filtered http-proxy
2 services unrecognized despite returning data. If you know the
service/version, please submit the following fingerprints at
```

```
https://nmap.org/cgi-bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port21-TCP:V=7.60%I=7%D=10/8%Time=5F7EFCF1%P=x86_64-pc-linux-gnu%r(NULL
SF:,36,"220\x20RSR7708-SRCMI-X\x20FTP\x20server\x20<version\x201\.00>\x20r
SF:eady\.\r\n")%r(GenericLines,54,"220\x20RSR7708-SRCMI-X\x20FTP\x20server
SF:\x20<version\x201\.00>\x20ready\.\r\n502\x20Command\x20not\x20implement
SF:ed\.\r\n")%r(Help,286,"220\x20RSR7708-SRCMI-X\x20FTP\x20server\x20<vers
SF:ion\x201\.00>\x20ready\.\r\n214-\x20The\x20following\x20commands\x20are
SF:\x20recognized\x20\(\*\x20=>is\x20unimplemented\)\.\r\n214-\x20\x20\x20
SF:\x20USER\x20\x20\x20\x20PORT\x20\x20\x20\x20STOR\x20\x20\x20\x20MSAM\*\
SF:x20\x20\x20RNTO\x20\x20\x20\x20NLST\x20\x20\x20\x20MKD\x20\x20\x20\x20\
SF:x20CDUP\x20\x20\x20\x20OPTS\*\r\n214-\x20\x20\x20\x20PASS\x20\x20\x20\x
SF:20PASV\x20\x20\x20\x20APPE\*\x20\x20\x20MRSQ\*\x20\x20\x20ABOR\*\x20\x2
SF:0\x20SITE\*\x20\x20\x20XMKD\*\x20\x20\x20XCUP\*\x20\x20\x20AUTH\*\r\n21
SF:4-\x20\x20\x20\x20ACCT\*\x20\x20\x20TYPE\x20\x20\x20\x20MLFL\*\x20\x20\
SF:x20MRCP\*\x20\x20\x20DELE\x20\x20\x20\x20SYST\x20\x20\x20\x20RMD\x20\x2
SF:0\x20\x20\x20STOU\*\x20\x20\x20PBSZ\*\r\n214-\x20\x20\x20\x20SMNT\*\x20
SF:\x20\x20STRU\*\x20\x20\x20MAIL\*\x20\x20\x20ALLO\*\x20\x20\x20CWD\x20\x
SF:20\x20\x20\x20STAT\x20\x20\x20\x20XRMD\*\x20\x20\x20SIZE\x20\x20\x20\x2
SF:0PROT\*\r\n214-\x20\x20\x20\x20REIN\*\x20\x20\x20MODE\*\x20\x20\x20MSND
SF:\*\x20\x20\x20REST\x20\x20\x20\x20XCWD\*\x20\x20\x20HELP\x20\x20\x20\x2
SF:0PWD\x20\x20\x20\x20\x20MDTM\x20\x20\x20\x20EPRT\x20\r\n214-\x20\x20\x2
SF:0\x20QUIT\x20\x20\x20\x20RETR\x20\x20\x20\x20MSOM\*\x20\x20\x20RNFR\x20
SF:\x20\x20\x20LIST\x20\x20\x20\x20NOOP\*\x20\x20\x20XPWD\*\x20\x20\x20FEA
SF:T\*\x20\x20\x20EPSV\x20\r\n214\x20Direct\x20comments\x20to\x20ftp-bugs@
SF:RSR7708-SRCMI-X\.\r\n")%r(SMBProgNeg,54,"220\x20RSR7708-SRCMI-X\x20FTP\
SF:x20server\x20<version\x201\.00>\x20ready\.\r\n502\x20Command\x20not\x20
SF:implemented\.\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port22-TCP:V=7.60%I=7%D=10/8%Time=5F7EFCF1%P=x86_64-pc-linux-gnu%r(NULL
SF:,15,"SSH-2\.0-RGOS_PK3223\r\n");
Device type: switch|specialized|printer
Running (JUST GUESSING): D-Link embedded (93%), Cisco embedded (86%),
HW group embedded (85%), NetOptics embedded (85%), Oki embedded (85%)
OS CPE: cpe:/h:dlink:des-7210 cpe:/h:cisco:sg_300 cpe:/h:oki:b711
Aggressive OS guesses: D-Link DES-7210 switch (93%),
D-Link DGS-3610 switch (93%), Cisco SG200 or SG300 switch (86%),
HW group HWg-STE Ethernet thermometer (85%), NetOptics iBypass switch (85%),
Oki B711 printer (85%), Oki B930 printer (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 12 hops

TRACEROUTE (using port 1720/tcp)
HOP RTT       ADDRESS
```

```
1   0.28 ms   gate-il11-bb1.in.tum.de (131.159.25.253)
2   0.79 ms   nz-csr1-2wr-bb2.in.tum.de (131.159.252.6)
3   0.90 ms   cr-gar1-be2-147.x-win.dfn.de (188.1.37.89)
4   0.89 ms   cr-gar1-be2-147.x-win.dfn.de (188.1.37.89)
5   10.84 ms  GM-FF-FRK-F-1.163.chinatelecomeurope.com (80.81.194.33)
6   10.86 ms  GM-FF-FRK-F-1.163.chinatelecomeurope.com (80.81.194.33)
7   ...
8   288.04 ms 202.97.48.217
9   255.39 ms 202.97.80.210
10  255.40 ms 202.97.80.210
11  301.10 ms 219.148.216.34
12  319.35 ms 59.46.60.146

OS and Service detection performed. Please report any incorrect
results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 820.65 seconds
```

List of Figures

List of Tables

Bibliography

[1] Apple. *Boost performance and security with modern networking*. 2020. URL: https://developer.apple.com/videos/play/wwdc2020/10111.

[2] Apple. *Use Multipath TCP to create backup connections for iOS*. 2017. URL: https://support.apple.com/en-us/HT201373.

[3] H. Asghari. *pyasn*. 2020. URL: https://github.com/hadiasghari/pyasn.

[4] *ASRank CAIDA*. 2020. URL: https://asrank.caida.org/.

[5] bhesmans. *mpsocks*. 2020. URL: https://github.com/bhesmans/mpsocks.

[6] O. Bonaventure. *Apple Music on iOS13 uses Multipath TCP through load-balancers*. 2019. URL: http://blog.multipath-tcp.org/blog/html/2019/10/27/apple_music_on_ios13_uses_multipath_tcp_through_load_balancers.html.

[7] O. Bonaventure. *Measuring the adoption of Multipath TCP is not so simple...* 2015. URL: http://blog.multipath-tcp.org/blog/html/2015/10/27/adoption.html?highlight=easy (visited on 09/18/2020).

[8] O. Bonaventure, M. Handley, and C. Raiciu. *An Overview of Multipath TCP*. 2012. URL: https://www.usenix.org/system/files/login/articles/login1210_bonaventure.pdf.

[9] P. Cornwell. *A Question of Timing*. 2018. URL: https://blog.cloudflare.com/a-question-of-timing/.

[10] Datawrapper. *Create interactive maps with Datawrapper*. 2020. URL: https://www.datawrapper.de/maps/.

[11] 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*. 2013, pp. 1–8.

[12] L. DEVLOPPERS. "traceroute (8)-Linux man page." In: *URL: http://linux. die. net/man/8/traceroute (visité le 25/11/2015)* ().

[13] O. Gasser, Q. Scheitle, P. Foremski, Q. Lone, M. Korczynski, S. D. Strowes, L. Hendriks, and G. Carle. "Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists." In: *Proceedings of the 2018 Internet Measurement Conference*. Boston, MA, USA: ACM, 2018. DOI: 10.1145/3278532.3278564.

[14] C. Jonathan. *Upstreaming multipath TCP*. 2019. URL: `https://lwn.net/Articles/800501/`.

[15] M. Luckie, B. Huffaker, A. Dhamdhere, V. Giotsas, and K. Claffy. "AS relationships, customer cones, and validation." In: *Proceedings of the 2013 conference on Internet measurement conference*. 2013, pp. 243–256.

[16] G. Lyon. *NMAP*. 2020. URL: `https://nmap.org/`.

[17] T. Mather. *geoiplookup(1) - Linux man page*. 2006. URL: `https://linux.die.net/man/1/geoiplookup`.

[18] 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*. 2015, pp. 7–12.

[19] Neohapsis. *mptcp-abuse*. 2014. URL: `https://github.com/Neohapsis/mptcp-abuse`.

[20] C. of Network Architectures and Services. *ZMap: The Internet Scanner*. 2015. URL: `https://github.com/tumi8/zmap`.

[21] peeringdb. *PeeringDB*. 2020. URL: `https://github.com/peeringdb/peeringdb/`.

[22] T. Z. Project. *ZMap: The Internet Scanner*. 2020. URL: `https://github.com/zmap/zmap`.

[23] C. Raiciu, U. P. of Bucharest, M. Handley, U. C. London, O. Bonaventure, and U. catholique de Louvain. *TCP Extensions for Multipath Operation with Multiple Addresses*. 2013. URL: `https://tools.ietf.org/html/rfc6824`.

[24] D. Stenberg. *curl.1 the man page*. 2020. URL: `https://curl.haxx.se/docs/manpage.html`.

[25] M. Studios. *AbuseIPDB making the internet safer, one IP at a time*. 2020. URL: `https://www.abuseipdb.com/`.

[26] tracebox. *Tracebox*. 2018. URL: `https://github.com/tracebox/tracebox`.