Yarrpbox: Detecting Middleboxes at Internet-Scale

FAHAD HILAL, Max Planck Institute for Informatics, Germany
OLIVER GASSER, Max Planck Institute for Informatics, Germany

The end-to-end principle is one of the foundations of the original Internet architecture. This principle is put to the test by middleboxes, i.e., devices which change important parts of a packet in transit. Middleboxes can have beneficial effects, such as a lower handshake time, but also make it more difficult to deploy newly developed protocols, such as TLS 1.3 and QUIC. Therefore, it is important to have a good understanding of the middlebox ecosystem in the Internet.

In this paper, we present results from a multi-faceted middlebox analysis study. We develop Yarrpbox, a tool to efficiently perform middlebox detection measurements on an Internet-scale. With Yarrpbox, we perform IPv4-wide middlebox detection and find that nearly 10% of paths are affected by a total of 5.8k middlebox devices. We perform the first IPv6 study to date, uncovering a lower prevalence of middleboxes in IPv6. Moreover, we show that the location of a vantage point can have an effect on the results, leading to up to 600 more detected middleboxes. Additionally, we characterize middleboxes by mapping them to vendors and resolving aliases. Finally, we release Yarrpbox as open-source software and make data and analysis code publicly available.

CCS Concepts: • **Networks** → *Network monitoring*; *Transport protocols*; **Middle boxes / network appliances**; **Network measurement**; *Public Internet*.

Additional Key Words and Phrases: Middlebox detection, middlebox interference, Internet measurement, stateless

ACM Reference Format:
Fahad Hilal and Oliver Gasser. 2023. Yarrpbox: Detecting Middleboxes at Internet-Scale. *Proc. ACM Netw.* 1, 1, Article 4 (June 2023), 23 pages. https://doi.org/10.1145/3595290

1 INTRODUCTION

A central notion of the traditional Internet architecture is the end-to-end principle. It states that packets should remain unaltered while in transit between the two endpoints of a communication. Routers on the path limit themselves to tasks such as route selection and packet forwarding, while the endpoints attend to the complexities associated with secure and reliable delivery of packets.

However, these strict textbook descriptions have nowadays been put into question. In the last few decades, people have called for adding complexity to the middle of the network. One such example are "middleboxes", i.e., intermediary devices manipulating traffic for purposes other than the standard functions of an IP router [9]. These middleboxes may not always be separate physical devices but could also exist as embedded functions of a single network device.

Compared to routers or switches, middleboxes are known to inspect, filter or even alter packets traversing them. While this is in clear violation of the traditional end-to-end principle, the Internet has seen an increased deployment of these middleboxes owing to the value they bring. Studies have shown that middleboxes now also penetrate cellular networks [36, 61]. Also, in corporate

Authors' addresses: Fahad Hilal, Max Planck Institute for Informatics, , Germany, fhilal@mpi-inf.mpg.de; Oliver Gasser, Max Planck Institute for Informatics, , Germany, oliver.gasser@mpi-inf.mpg.de.

2834-5509/2023/6-ART4
https://doi.org/10.1145/3595290

settings, they are as numerous as standard networking appliances [55]. They seem to have cemented their place in the Internet ecosystem with 25% of tested paths found to be containing at least one middlebox [35]. In the current Internet, they fulfill a multitude of tasks such as thwarting of attacks, censoring or monitoring users, address space expansions, or balancing resources.

Apart from breaking the end-to-end principle, the deployment of middleboxes comes with additional caveats. They introduce hidden points of failure, thus complicating the debugging of networks. Moreover, they stand in the way of innovations and improvements to protocols and their extensions. This entails that protocol designers have to ensure that any new protocols (e.g., TLS 1.3 [14, 34, 41, 53, 58] or QUIC [10, 17, 17, 18, 38, 54, 64]) or extensions of existing protocols (e.g., MPTCP [4, 27, 28, 45]) are middlebox-aware. In order to make protocols middlebox-aware, a good understanding of middlebox behavior in the Internet is necessary.

A lot of research has gone into detecting middleboxes in the wild [1, 11, 12, 20, 31, 37, 59, 60, 66]. One promising approach is Tracebox [20]. Tracebox is an extension to the network topology mapping tool traceroute. In addition to tracing the path to a destination, Tracebox also looks for modifications on the path in order to identify middleboxes. It does so by inspecting the quoted packet in the returned ICMP messages. However, its major limitation is the need for state maintenance and its sequential probing. This slows down probing and has the potential to trigger undesired side-effects like ICMP rate-limiting [30]. These shortcomings limit the feasibility to use Tracebox in Internet-scale measurements. In this work, we perform large-scale measurements by probing around 15M IP addresses. Based on the completion times of previous measurements that use Tracebox [20, 22], we estimate such measurements with Tracebox to take in excess of 4 months. This is particularly undesirable as some middleboxes have been seen to exhibit periods of inactivity. A recent study pointed to half the detected middleboxes as displaying dynamic behavior [22].

In this work, we aim to improve upon Tracebox to obtain a middlebox census at Internet-scale in a much shorter period of time. Specifically, our main contributions are:

- **Yarrpbox:** We develop Yarrpbox, a stateless measurement tool for Internet-scale middlebox detection. Yarrpbox is over 300 times faster than Tracebox and can conduct large-scale scans in under 10 hours. It detects packet-rewriting middleboxes that make modifications to the IP and transport layer headers. To foster further middlebox detection studies, we make Yarrpbox available at:

  github.com/yarrpbox/yarrpbox

- **Internet-scale IPv4 measurements:** We perform the first Internet-scale IPv4 measurements to date and find that about 10% of paths are affected by a total of 5.8k middleboxes.
- **First IPv6 measurements:** We also perform the first analysis of middlebox deployment in IPv6, where we see a lower percentage of affected paths (0.2%).
- **Middlebox characterization:** We leverage third-party datasets in an in-depth study to characterize middleboxes. We resolve aliases of middlebox IPs and find 132 multiple-alias middlebox devices. Moreover, we find that more than 25% of middleboxes are deployed in only 10 ASes.
- **Influence of vantage point location:** Finally, we perform measurements from 8 different geographic locations on 6 continents, finding that paths with middleboxes on them range from 6.4% to 29% and from 0.05% to 0.15% for our IPv4 and IPv6 measurements, respectively.

2 BACKGROUND AND RELATED WORK

Generally, the end-to-end principle states, that a packet should remain unaltered while in transit and on-path hops should not modify it. There are, however, a couple of header fields, which can or even must be modified by routers.

The most well known ones are the IPv4 TTL and IPv6 Hop Limit (HL) fields. These fields ensure that packets are not circulating infinitely and must therefore be decremented at each hop [6, 19, 50]. Due to the changing TTL value, the IPv4 checksum field also needs to be recomputed at each hop. This is not the case for IPv6, as it does not have a checksum field. Furthermore, Quality of Service (QoS) related fields such as the Differentiated Services (DS) field for IPv4 (formerly known as ToS [51]) and the Traffic Class (TC) field for IPv6 have no guarantee of remaining untouched in transit. However, barring these, most other fields from both the IP and the transport headers hold an end-to-end significance and should therefore not be modified. In our analysis we ignore modifications of TTL, HL, DS, and TC fields, unless otherwise explicitly stated.

Two protocols which are essential for traceroute in general and thus also for Yarrpbox are ICMP [49] and ICMPv6 [15]. One important component of both protocols is the TTL (we use the term TTL for both the IPv4 TTL as well as the IPv6 hop limit) exceeded message. This message is sent by a router to the source address of a packet upon receiving a packet with an expiring TTL (i.e., it would be decremented to 0 when forwarding the packet). Yarrpbox uses these returned TTL exceeded messages and inspects the quoted original packet to detect middlebox modifications.

An important consideration for this work is the size of the quoted packet, i.e., the TTL exceeded packet sent by the router which contains (part of) the original packet. For ICMPv6, the quoted packet should return as much of the original packet as possible, without exceeding the minimum IPv6 MTU of 1280 bytes [19]. For IPv4, the original RFC 792 [49] states that only the IPv4 header plus the following 8 bytes should be returned in a quoted packet. This recommendation is updated in RFC 1812 [6], which—similar to ICMPv6—recommends to quote as much as possible from the original packet, without exceeding 576 bytes.

Another important consideration for this study is the rate-limiting behavior of ICMP error messages [30]. Rate-limiting is important as it determines as to how often TTL Exceeded error messages, which are central to this study, would be sent by routers along paths. As ICMPv6 is a central building block for IPv6 and can therefore not be simply blocked, routers are required to perform rate limiting of ICMPv6 error messages. For IPv4, rate-limiting is generally not as aggressive as the IPv6 equivalent of the protocol [15].

Many techniques have been put forward to study middlebox interference (alterations to end-to-end headers). Medina et al. [44] propose "tbit" and use it to investigate the interactions between transport protocols and middleboxes. They send TCP probes to Web servers and analyze the responses to see how e.g., the ECN, IP, and TCP options are influenced. They find interferences to about one-third of Web servers, when sending the Record Route or Timestamp option.

Honda et al. design "TCPExposure" [35], a client-server application with custom TCP implementations at both ends. This allows for exact interference detection between client and server, but it also requires control over both endpoints.

Craven et al. [16] propose "TCP HICCUPS" as an extension tightly integrated into TCP implementations to achieve in-band middlebox detection. In addition to control over both endpoints, it also calls for updates to the TCP stack.

Detal et al. [20] introduce a new traceroute-style middlebox detection tool with "Tracebox". Similar to traceroute, Tracebox sends out probes with increasing TTL. However, in addition to using the ICMP Time Exceeded messages for identifying the replying hop, they are also leveraged for detecting middlebox interferences. Although Tracebox is server independent, it performs sequential and stateful probing making it slow and susceptible to undesired side effects such as ICMP rate-limiting and IDS alarms. In this work, we leverage ideas from Tracebox to build Yarrpbox, a stateless, destination randomizing, high-speed middlebox detection tool.

The original Tracebox was later extended to the mobile domain through TraceboxAndroid [59]. Building on Tracebox-style middlebox detection efforts, Zullo et al. [66] apply these ideas to the detection of NATs. Edeline and Donnet [23] use Tracebox to analyze the causes for the transport-level ossification. They categorize observed interferences by the impact they have on TCP traffic. For instance, 75% of interferences are found to be benign (e.g., DSCP modifications). Modifications to fields such as TCP checksum (15%) are classified as being somewhere in between on the scale of benign[1] and path-impairing. The final category involves impairments with the potential to negatively affect TCP, e.g., modifying the TCP sequence number or TCP options.

Moreover, Hesmans et al. [31] design a click-based middlebox tool called “MBTest”. They use it for the experimental evaluation of interactions between middleboxes and the Linux TCP stack. They find that middleboxes have a noticeable impact on TCP options.

Finally, many other works also study middlebox interactions with DNS, HTTP, and HTTPS [12, 37, 52, 60].

3 YARRPBOX

As discussed in Section 2, there are existing tools aimed at middlebox detection. However, those tools fall short either due to the speed at which detection can be performed or the inability to lend themselves to a middlebox census at Internet-scale. To address these issues, we propose Yarrpbox. As the name suggests, it draws inspiration from the state-of-the-art middlebox detection tool Tracebox. Tracebox, however, is severely restricted by state keeping on outgoing probes. Although path specific parallelism is an obvious improvement, the speed enhancement it would provide would be achieved at the cost of burdening paths and potentially triggering IDS alarms and ICMP rate limiting, which we want to avoid in middlebox detection. With Yarrpbox, we aim to create a tool that allows for rapid scanning, detects as many modifications as possible, while also avoiding the triggering of ICMP rate limiting. These goals are fulfilled through a stateless design, the use of hashes encoded in the packet, and a randomized approach to probing, respectively. Compared to other stateless scanning tools such as ZMap or Yarrp, Yarrpbox is the first of its kind to enable Internet-scale *middelbox detection*. In the following, we present design decisions, challenges, and implementation details of Yarrpbox and compare it with Tracebox.

3.1 Implementation

Yarrpbox is built on top of Yarrp [7], benefiting from many of its features. Yarrpbox is stateless, allowing for high-speed probing, and randomizes the order of targeted destination addresses, reducing the risk of ICMP rate limiting like Yarrp. Like Yarrp, it also follows Paris traceroute techniques [5]. Yarrpbox can send different probe types: TCP SYN and TCP ACK for IPv4; TCP SYN, TCP ACK, UDP, and ICMPv6 for IPv6. Similar to Yarrp, Yarrpbox uses offline post-processing of scan output files to perform the middlebox detection.

Yarrpbox uses the following steps for its middlebox detection: (1) Yarrpbox crafts probes and encodes state information such as the destination IP, timing information, and TTL within the probe packet. Header fields that are not used to store any state are set to fixed values across all probes. This allows us to identify changes of these fields by inspecting the received values in a post-processing step. (2) Yarrpbox sends out TTL-limited probes to randomized destination addresses. (3) Yarrpbox receives TTL Exceeded replies, extracts header values from them, and writes all relevant information to an intermediate text file. (4) In a post-processing step, we run the actual middlebox detection: We check for mismatching header values between the quoted packet in the TTL Exceeded response and our sent packets. In case of a mismatch, we detect a middlebox interference and pinpoint it to the hop by inspecting the encoded TTL in the quoted packet.

[1]Correctly updated TCP checksums in response to, for instance, a NOP addition would be classified as benign.

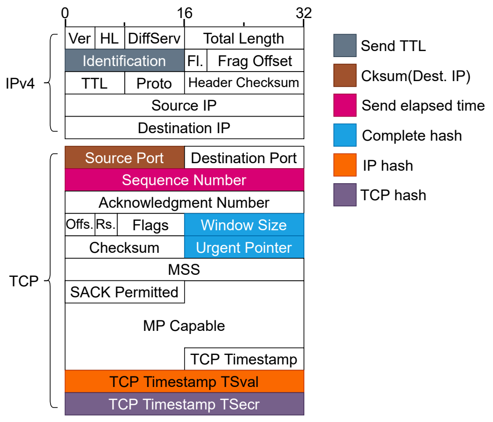

Fig. 1. Fields for state and hash encoding in IPv4.

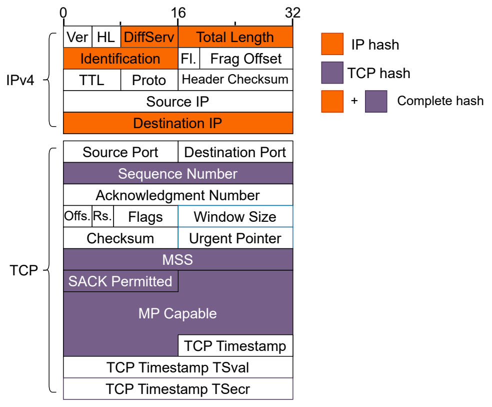

Fig. 2. Fields over which hashes are computed.

In the following we provide details on steps 1, 3, and 4. We omit step 2, as it uses the same technique as Yarrp [7].

3.1.1 Packet Crafting. Yarrpbox encodes two different types of values in the packet: (1) state information and (2) configurable header values. The state information is the same as for Yarrp and includes the originating TTL, checksum of the target IP address and time elapsed since the start of probing. Configurable header values can be specified on the command line, e.g., to change the default MSS or send a Window Scale TCP extension. In the following we describe the packet crafting separately for IPv4 and IPv6.

IPv4: Yarrpbox adheres to the choices that Yarrp makes for fields where the state is encoded (allowing us to follow Paris traceroute methods, recovering state of partially quoted packets, and adopting choices already accepted by the research community). This means that for IPv4 probes, Yarrpbox stores the above-mentioned state information entirely in the IP and TCP headers (cf. Figure 1). This presents challenges for some cases of middlebox interferences. More specifically, detecting interferences to the state carrying IP ID, TCP sequence number, and the source port fields through a simple comparison is not possible. To remedy this, Yarrpbox calculates three additional hashes. The first of these hashes—the “IP hash”—is over fields from the IP header. These include the IP ID, Differentiated Services, IP total length, and the destination IP fields. Fields in addition to the IP ID are chosen to further enhance the reliability of detection over fields identified to be extensively modified in prior measurement studies [23]. The second hash—the “TCP hash”—is computed over TCP header fields including the state carrying sequence number and the TCP options. The final hash—the “complete hash”—is calculated over the concatenation of fields used in the IP hash and the TCP hash. We use the efficient non-cryptographic XXHash hash function [13] to generate a 4-byte hash.

Next, we choose appropriate header fields to store the three hashes. We refrain from storing them in the payload, for reasons similar to the ones that dictate our choice for the encoding of state for IPv4 probes. We choose two 4-byte fields and two 2-byte fields (i.e., 12 bytes in total) from the IP and TCP headers to store the three hashes. The IP hash is placed into the Timestamp TCP option’s TSval field, the TCP hash goes into the TSecr field of the same option. Finally, we split the complete hash between two fields from the TCP header: Two bytes are encoded into the TCP receiver window, while the other two bytes are inserted into the urgent pointer field (without setting the urgent flag). We select these fields as they are not changed by non-middlebox devices and have seen a low number of interferences in previous studies [23]. The redundant complete hash facilitates the detection of interferences to the set of hashed fields if the Timestamp TCP option is removed. This allows us to narrow down modifications to the state carrying fields plus the hash storage fields, if no modifications to the fields that are both hashed over and also assigned fixed values occur. Figure 1 shows a schema of state encoding fields in IPv4. The fields over which the hashes are computed are depicted in Figure 2.

We can detect the modifications of header fields (DiffServ, IP total length, destination IP, and TCP options) which are hashed over by direct comparisons with assigned values (or of computed checksum with stored checksum in source port in case of the destination IP). This means that a mismatch between extracted and computed hashes (computed over fields from the extracted ICMP TTL exceeded message quote) must be down to either the IP ID, the sequence number, or the fields that store the hash. For instance, consider a scenario where the IP hash and the complete hash extracted from the Timestamp TSval, and the combination of the urgent pointer and receiver window fields, respectively, do not match the corresponding computed hashes over the respective fields mentioned above. If none of the fields from the set of destination IP, DiffServ, IP total length, sequence number and TCP options have been modified, then either the IP ID has been modified or the Timestamp TSval (IP hash storage) has been modified, or any or both of the fields from the combination of receiver window and urgent pointer (complete hash storage) have been modified. The sequence number would not have been modified as that would cause the extracted TCP hash to be different from the computed TCP hash as well. A similar scenario includes computed TCP hash and complete hash not matching their extracted counterparts, with no modifications to the hash input fields assigned fixed values. The inference here is that the sequence number, the TSecr (TCP hash storage), or a combination of receiver window and urgent pointer are modified. Another scenario could be where all the three extracted hashes fail to match the corresponding computed hashes, while the fixed value fields were not identified as modified. In this case, the modification may come from the IP ID, the sequence number, the TSval, the TSecr, or the combination of urgent pointer and the receiver window.

IPv6: Again, Yarrpbox follows Yarrp's choice of storing the state information. For IPv6, state information is thus stored in the header and the payload. This choice is driven by the difference in packet quoting behavior for IPv4 and IPv6 [15, 49]. Encoding state information in the payload allows for easier and more versatile detection of interferences in IPv6. Since the header fields are not used to carry state or the integrity data, more header fields can be assigned configured values across all probes. The only header fields in IPv6 carrying any state are the source port (for UDP and TCP probes) and the ICMPv6 identifier. They both carry the checksum of the destination IP field. In order to detect modifications to the source port, TCP and UDP probes carry the 4-byte XXHash [13] within the payload alongside other state information. The same hash is calculated over the identifier field of the ICMPv6 header. Additionally, for the purpose of verification of the integrity of the encoded state information, a hash over the payload is also appended. In the event of a mismatch between the hash computed over the payload and the encoded hash, Yarrpbox ignores the response. All main header fields except the source IP, the destination IP and the hop limit from the IPv6 header can be configured at runtime. Similarly, except for the source port from the TCP header and UDP header, and the identifier from the ICMPv6 header, several fields are configurable. We refer the interested reader to Appendix B for more details on IPv6 encoding.

3.1.2 Response Parsing. The responses of value for Yarrpbox, like Tracebox, are ICMP Time Exceeded messages. As discussed in Section 2, these quote the packet that lead to the generation of the Time Exceeded message. The quote size differs for IPv4 and IPv6 owing to different recommendations [6, 49]. According to the recommendation we expect IPv4 quotes to be either "partial

	IPv4		IPv6	
Tool	MB Targ.	Non-MB Targ.	MB Targ.	Non-MB Targ.
Yarrpbox	15	2	5	2
Tracebox	10	3	4	1

Table 1. Middlebox IPs detected by Yarrpbox and Tracebox, with IPv4 and IPv6 targets observed with and without middleboxes on the path.

quotes" (e.g., first 28 bytes) or "full quotes" (entire packet). For IPv6, the recommendation is to always send full quotes. Yarrpbox is designed to apply appropriate parsing to both types of quotes.

During response parsing, we extract the encoded state, integrity, as well as other header field values. Subsequently, we perform integrity checks besides comparisons between expected and observed values of other header fields.

3.1.3 File Writing. File writing is the last component of Yarrpbox which is altered from Yarrp's default implementation to incorporate middlebox detection functionality. During probing, Yarrpbox first writes responses to an intermediate yrp file, in the order of the received response packets. This file includes extracted per-packet state, e.g., the encoded TTL, the IP address of the destination, and the elapsed time. Moreover, Yarrpbox also writes the observed and expected header field values in addition to several boolean flags. This is done live, i.e, during the course of probing. These flags indicate if a particular header field has been modified.

Note that due to randomized probing, entries for each hop to a destination are intermixed with other responses in the yrp output file. The next step is to ensure that all replying hops towards the same target are merged. Therefore, Yarrpbox performs an *offline reconstruction* once the probing completes and can output the results to a text or JSON file.

3.2 Comparison with Tracebox

We compare our tool with Tracebox by performing small-scale scans to two types of targets: targets with and without middlebox-affected paths. We use 1000 targets for each category and run comparative measurements for IPv4 and IPv6.

As Tracebox limits the scanning rate to 1 packet per second and can not be reliably run in parallel[2], we also run Yarrpbox at a rate of 1 pps for the comparative analysis[3]. Each individual Tracebox scan to our target list of 1000 IPs took about 34 hours to complete. At this rate, Tracebox would take approximately 58.2 years to perform our large-scale measurements with target lists as large as 15M targets. We carry out such measurements in under 10 hours with Yarrpbox.

The results of the comparative analysis are shown in Table 1. For IPv4, Yarrpbox does not detect TCP checksum modifications. Now, this may not be considered a major shortcoming as modifications to this field are a consequence of alterations to other fields within the TCP and IP headers. However, we find Tracebox reporting a large number of middlebox IPs that modify the TCP checksum without there being any modifications to fields that the checksum is computed over. We manually inspected several Tracebox traces and find these checksum modifications to result from the removal of TCP options while the packet is also being partially quoted by the responding hop. Tracebox does not apply appropriate parsing and correct interference detection logic in the face

[2]When run in parallel, we discover Tracebox can not properly assign response packets to the correct instance and incorrectly identifies destination IP modifications. To ensure proper response parsing, we thus run Tracebox sequentially.
[3]Although Yarrpbox is built for high throughput, large-scale scanning, we nevertheless lower its sending rate for comparison reasons.

of partially quoted TCP options. This results in this hop (due to the TCP checksum modification) and the first subsequent hop which fully quoted the packet (due to TCP option removals) to be counted as separate middleboxes. To address this incorrect middlebox identification, we remove these double-counted IPs from the Tracebox measurements. As a result, we find Yarrpbox to perform as well as Tracebox. Moreover, Yarrpbox also extracts a similar number of replies, as Tracebox, towards both sets of targets.

For IPv6, Yarrpbox does not detect middleboxes that remove its state carrying payload. However, we observe that there are several cases of payload removal reported by Tracebox. We again manually inspect these cases and observe replies with payload and option removals without any changes to the TCP checksum. These replies are in fact IPv6 partial quotes which are not quoting the options and the payload. We conclude that Tracebox applies flawed interference detection logic to IPv6 partial quotes reporting unquoted options and the payload as being removed. We remove such IPs from Tracebox analysis and find the performance of Yarrpbox to be on par with Tracebox. Similar to our IPv4 results, the replies extracted by Yarrpbox are comparable to Tracebox's.

To analyze the memory and CPU usage, we run[4] Yarrp and Yarrpbox for 265.5k targets and Tracebox for 200 targets, finding the first two to exhibit identical memory usage with a 4.8% increase in CPU usage in Yarrpbox due to additional MB detection computations. Although Tracebox has a lower CPU usage overall, it uses 1000x more CPU resources than Yarrpbox when normalized by the number of probed targets.

To summarize, Yarrpbox can successfully detect all modifications detected by Tracebox, with the exception of the destination IP, the TCP checksum (over IPv4 only), and alterations to the IPv6 payload. Furthermore, Yarrpbox makes Internet-scale middlebox detection feasible for the first time.

4 INTERNET-SCALE MEASUREMENTS

In this section, we detail our measurement setup and discuss the ethical considerations guiding us in our measurement methodology. We then report our findings from the Internet-scale measurement campaigns conducted from our university vantage point located in Europe.

4.1 Measurement Setup

We use Yarrpbox to scan the IPv4 as well as the IPv6 Internet. All of these scans are performed from a single vantage point located in Europe. For both protocols, we first perform small-scale scans in order to determine a suitable scanning rate (based on the number of ICMP/ICMPv6 replies we receive, missing replies in our traces, and the number of detected middlebox IPs we have the highest confidence in). We find that 20kpps and 5kpps to be suitable choices for our large-scale IPv4 and IPv6 measurements, respectively.

For IPv4, we perform TCP SYN and TCP ACK scans using Yarrp's "entire" mode. In this mode, a single random IP address from each /24 is probed. For IPv6, we perform TCP SYN, TCP ACK, ICMPv6, and UDP scans. As Yarrp does not support an entire mode in IPv6, we generate a list of random IPv6 addresses representing different BGP prefixes. A pre-specified number of addresses are randomly generated for each BGP announced IPv6 prefix. During the generation of an IPv6 address, we ensure that we avoid generating an IPv6 address for a less-specific BGP-announced prefix, that falls in one of its more-specific BGP-announced prefixes. This ensures that we do not misattribute middlebox interferences of more-specific prefix targets to less-specific ones. Such misattribution is undesirable due to potentially different routing strategies, AS locations, and middlebox interferences, towards the more-specific prefix compared to its less-specific prefix. We

[4]System specs: AMD Epyc 7251, 8 cores, 16 threads, 32 GB RAM, Debian 10.

Date	Scan	Replies	Traces	Hop IPs	Total Interf.	Interf. excl. DS/TC	MB IPs	HC MB IPs
Dec 22, 2021	IPv4 SYN	89.4M	11.9M	628.4k	6.4M	759.5k	16814	5808
Dec 22, 2021	IPv4 ACK	88.8M	11.9M	606.5k	6.2M	636.6k	8914	4215
Feb 10, 2022	IPv6 SYN	92.4M	14.4M	282.3k	7.4M	197.9k	8241	4262
Feb 10, 2022	IPv6 ACK	94.6M	14.4M	305.6k	8.2M	25.1k	7583	3810
Feb 10, 2022	IPv6 UDP	94.1M	14.4M	316.8k	50.1M	50.7k	10759	6030
Feb 15, 2022	ICMPv6	98.7M	14.4M	304.9k	8.7M	4	2	0

Table 2. Overview of large-scale measurements from the university vantage point.

use the BGP routing table from February 9, 2022 published by CAIDA [8], consisting of under 160k BGP announced IPv6 prefixes. From it, we generate 100 addresses per prefix and end up with just under 16M IPv6 addresses. Using a prefix-based approach for IPv6 instead of a hitlist-based approach [29], we can better compare its results to the IPv4 results.

In TCP packets we include the MSS (set to 536 bytes for IPv4, 1220 bytes for IPv6), SACK-Permitted, MP CAPABLE, and the timestamping TCP options.

4.2 Ethical Considerations

Before we conduct our Yarrpbox measurements, we incorporate proposals by Partridge and Allman [46] and Kenneally and Dittrich [39]. We follow best measurement practices [21] by limiting our probing rate, using a well-established blocklist, and making use of dedicated servers and AWS instances as measurement vantage points. These vantage points communicate the scientific nature of our measurements with an informing rDNS name, a website providing more information, and contact details to reach out to us in case of issues. During our measurements, we did not receive any complaints or requests for being added to our blocklist.

4.3 Result Overview

We apply similar analyses to the results of both the IPv4 as well as the IPv6 large-scale scans. First, we provide overview statistics. Then, we add in-depth analyses on the completeness of traces, the location error and the resulting sets of varying confidence, the most frequent interferences, and the detection of middleboxes with multiple aliases.

In Table 2, we show an overview of our IPv4 and IPv6 scans. When looking at IPv4 and IPv6 separately, we can see that we get a relatively similar number of replies, traces, and unique hop IPs. The number of interferences, however, differs greatly between scans, which we detail in following sections. The 7th column shows the number of interferences excluding "benign" modifications such as the IPv4's DiffServ (DS) and IPv6's Traffic Class (TC) fields. Those fields in fact allow for modifications on the path and have no end-to-end semantic. Therefore, we do not report on any DS and TC modifications in the remainder of the analysis, unless explicitly stated. The two rightmost columns contain the number of middlebox IPs and those in the highest confidence set, which we detail in the upcoming Section 4.4. Furthermore, we focus on two scans for IPv6, i.e., the TCP SYN and UDP scan, which show the most interferences excluding TC.

Trace Gaps: Next, we analyze the trace characteristics themselves, starting with the completeness of traces. To this end, we compute the largest gap in each of our traces. The largest trace gap is determined by simply computing the maximum difference between the next and the previous replying hop number in a trace. A largest trace gap of 1 is the smallest possible gap for a trace where only different hop numbers reply.[5] We report this case as "No Gap" in our results. Similarly, a

[5]Note that although a largest trace gap of 1 might seem like there is a gap in the trace, there is not.

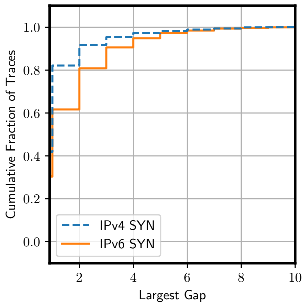

Fig. 3. Largest trace gaps for IPv4 and IPv6 SYN scans.

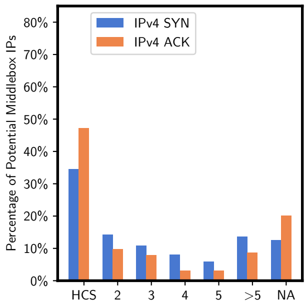

Fig. 4. Location error for IPv4 SYN and ACK scans.

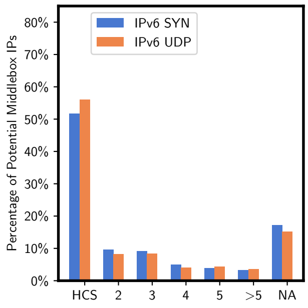

Fig. 5. Location error for IPv6 SYN and UDP scans.

trace with the computed largest trace gap of 2 has a single hop gap. This could happen, for instance, when the next replying hop after hop 5 is hop 7. To aid comprehension, we report this as “Largest Gap = 1” in our results. Please note, that there could be cases where the computed gap is 0. This happens when there is only one replying hop number in the complete trace. We find this to be very *rare*.

Figure 3 shows the distribution of largest trace gaps. As the distribution is similar for all IPv4 scans, we only show the SYN scan. The same is true for IPv6 scans. About 41% of IPv4 traces are complete, i.e., without any gaps, and more than 80% of traces are either gap-free or the largest gap therein equals 1. This as well as the scans from multiple vantage points (see Section 5.1) shows that Yarrpbox’s randomized and load-distributive probing works well in IPv4. In IPv6 the situation differs, with around 30% of complete traces, and more than 60% of traces having a gap size not larger than 1. The lower IPv6 trace completion underlines, that aggressive ICMPv6 rate limiting [15] remains a challenge in IPv6.

4.4 Locating Middleboxes

Overall, we find 16.8k middlebox (MB) IPs in IPv4 and 8.2k MB IPs in IPv6 (cf. Table 2). When looking at probe types other than TCP SYN, we find differing numbers. In IPv4, we find only about half the number of MB IPs with our TCP ACK scan compared to the TCP SYN scan. In IPv6, the ACK scan is comparable to the SYN scan, with the UDP scan finding 2.5k more MBs. As expected, the ICMPv6 scan elicits the fewest interferences and thus also MB IPs. This underlines the value of using different protocols when investigating middlebox behavior in the Internet.

Location Error: Next, we leverage the trace gap metric and middlebox interferences on the path, to locate middleboxes with varying confidence levels. We use the term “potential middlebox IPs” for hop IPs where we see middlebox modifications. We then use the location error metric to classify MB IPs into different confidence levels. The location error denotes the distance to a previous replying hop which is quoting at least as much of the original packet as the potential middlebox hop[6]. This ensures that we exclude previous replying hops with a small ICMP quote size, from which we can not accurately determine middlebox interferences. Potential middlebox IPs with a location error of 1 constitute the highest confidence middlebox IP set (HCS).

Figure 4 shows the location error of IPv4 MB IPs. We find a similar trend for both IPv4 scan types, although a higher percentage of the ACK scan falls into the highest confidence set. Consequently,

[6]We find that computing the location error as the distance to a previously replying hop that quotes until the modified field, is computationally infeasible.

we can exactly locate 35% of middleboxes for TCP SYN scan and 48% for TCP ACK in IPv4. Although the ACK scan reports a higher percentage of IPs in the most confident middlebox IP set, the absolute number of IPs in the same set is higher by about 1500 IPs for the SYN scan. The prefix as well as the AS count is also higher for the SYN scan. Detal et al. [20] also use a slightly different location error metric, that searches from the interference reporting hop until the first previous full quote hop. They apply this only to look at how accurately they locate MSS modifying IPs. They locate about 10% of such middlebox IPs with an error of 1. Our approach uses a stricter metric and we apply it to middleboxes making any modification.

Next, we analyze the location error for our IPv6 measurements, as shown in Figure 5. In IPv6, we can accurately locate a higher percentage of potential middleboxes, i.e., they fall into the HCS. The SYN and UDP scans show a similar distribution, with 52% and 55% of MB IPs in the HCS, respectively.

Highest Confidence Middlebox IPs: We also report the number of middlebox IPs in the highest confidence sets. In IPv4, the SYN scan is able to uncover 5.8k HC MB IPs, whereas with the ACK scan we find 4.2k IPs. Of those addresses, only 1477 are visible as HC MB IPs in both scans (25% for SYN and 35% for ACK). In IPv6, we find that the UDP scan—with around 6k IPs—uncovers the most highest confidence middlebox IPs. Both IPv6 TCP scans reveal about 4k IPs, whereas the ICMPv6 scan does not reveal a single MB IP in the highest confidence set. We find an overlap of only 750 HC MB IPs between TCP SYN and UDP in IPv6 (17% for SYN and 12% for UDP). When inspecting the MB-affected paths, we find that IPv4 paths are much more likely to be affected by MBs (9.9%) compared to IPv6 (0.2%).

AS Distribution: After establishing the set of highest confidence middlebox IPs, we investigate the ASes they are mapped to. Figure 6 shows the AS distribution for the IPv4 and IPv6 SYN scans. In total, we find HC MB IPs in about 1.6k IPv4 ASes and 1.3k IPv6 ASes. 80% of all HC MB IPs are located in the top 500 ASes (by number of middleboxes), whereas the top 10 ASes cover more than 25% of these. Moreover, we report the top 5 ASes for each scan in Table 3. The ASes from the top 5 are found to be registered across 4 continents namely in Europe, North America, South America, and Asia. For both IPv4 scans, AS 7018 (AT&T) is found to contain most of the highest confidence middlebox IPs. Other top ASes in IPv4 are the Polish ISP Orange Polska (AS 5617), the Swedish Tier 1 AS Arelion (AS 1299)—previously called Telia Carrier—, the German Tier 1 Deutsche Telekom (AS 3320), Microsoft (AS 8075), and the South American ISP Claro (AS 4230).

In IPv6, the picture looks quite different than in IPv4. The top ASes are two Indian ISPs: Idea Cellular (AS 45271) and Bharti Airtel (AS 45609). Other top ASes are Arelion (AS 1299), the Chinese research network CERNET (AS 23910), the US transit provider Hurricane Electric (AS 6939), and the Japanese Tier 1 NTT (AS 2914). The only AS in the top 5 in both IPv4 and IPv6 is Arelion.

Middlebox Location on the Path: We also study if HC MB IPs are located in the source AS (AS of the probing machine located in our university network), the destination AS (AS of the target IP) or any other AS besides these. For IPv4, we find that more than two thirds of these IPs (69.8%) are located in the destination AS, 0.4% are in the source AS, and other ASes contain 29.8%. We further investigate the middleboxes located in other ASes by computing a normalized path location metric based on the hop number. We observe that on average the HC MB IPs are located very close to the end of the observed path (average/median path location of 88.8%/92.8%) which is consistent with previous work [23]. Similar analysis for IPv6 reveals, that other ASes contain nearly 58% of the HC MB IPs. Destination ASes contribute about 42% and similar to the IPv4 SYN scan the source AS contains less than 0.5% of the HC MB IPs. The normalized path location analysis reveals similar results to the IPv4 SYN scan, with HC MB IPs being present very close to the end of the observed path (average/median path location of 83.3%/87.5%). To summarize, the vast majority of middleboxes are located in the destination AS itself or very close to the destination.

Scan	#1 AS	#2 AS	#3 AS	#4 AS	#5 AS
IPv4 SYN	7018 (9%)	5617 (3.3%)	1299 (3.3%)	3320 (1.8%)	8075 (1.7%)
IPv4 ACK	7018 (12%)	1299 (4.6%)	3320 (2.5%)	8075 (2.4%)	4230 (2.4%)
IPv6 SYN	45271 (5.8%)	1299 (4.7%)	23910 (3.2%)	6939 (2.3%)	45609 (2.1%)
IPv6 UDP	45609 (3.4%)	45271 (2.3%)	6939 (2.1%)	1299 (2%)	2914 (1.7%)

Table 3. Top 5 ASes and the percentage of highest confidence middlebox IPs in them, for IPv4 and IPv6 scans.

AS Categories: To better understand what type of ASes middleboxes are usually deployed in, we classify ASes with the ASdb dataset [65]. This dataset contains categories and subcategories for each AS. For IPv4, we find the top 2 categories to be "Computer and Information Technology" (1312 out of 1641 ASes) followed by "Education and Research" (165 ASes). Out of the 1312 ASes in the first category over 1200 end up in the ISP subcategory. On the IPv6 side, same categories make the top 2. 1176 out of 1329 ASes map to the former while the latter sees under 85 ASes. Similar to our IPv4 results, a large chunk (1116 out of 1176 ASes) are ISPs. This confirms the picture seen in the top 5 ASes that the majority of MB ASes are indeed ISPs, transit providers, and research networks.

4.5 Middlebox Modifications

After we have now identified middleboxes, we investigate what kind of interferences these perform.
DiffServ & Traffic Class: The by-far largest number of interferences concern the Differentiated Services in IPv4 (88.1%) and the Traffic Class in IPv6 (97.3%). Since modifying these is part of normal network operations and these fields do not carry any end-to-end meaning, we do not consider these as actual middlebox modifications. Note that we do not classify IPs modifying these as middleboxes, but still report the numbers to allow for comparison with previous work [20, 23] (see Appendix C).
Options: The second-most common modification in IPv4 is the "IP ID/Timestamp TSval/Receiver Window or Urgent Pointer" category. We find 584k modifications for this category. Moreover, we also find TCP options being modified. The Timestamp option's TSval field is observed to experience the most modifications (50k). We manually investigate the value of this field using the collected pcaps but do not find any pattern in the newly assigned TSval values. Also, over 41k NOP option additions (make the top interference at some vantage points, see Section 5.3) are seen while MP CAPABLE option is found to be the most removed of the set options with 38k instances. We find that NOP addition and MP CAPABLE removal overlap in 98% percent of the cases. This is in line with related work [4, 45], which finds MPTCP deployment to be heavily affected by middleboxes. Additionally, we find 5.2k IP ID and 3.8k sequence number modifications. Finally, in 1.5k cases the TCP timestamp option is removed.
Length & Offset: We also observe interferences with the IP total length (3.8k) and TCP data offset (464) fields. These generally result from option removals and additions. However, we observe a large mismatch between option removals and additions, and the modifications to these length fields. This is due to the length-preserving replacement of options through a series of NOPs instead of simple removals. Further, there is a mismatch between the IP total length and TCP data offset modifications. This is due to the higher visibility of the former field owing to its presence in both partial as well as full quotes. In fact, we find 88% of all IP total length modifications in partial quotes.
SYN vs. ACK: The IPv4 ACK scan triggers fewer interferences compared to SYN. Another difference between the two scans is the much lower number of TCP option removals and additions, and sequence number modifications. The ACK scan reports about 1000 times lower numbers for both categories. Modifications to the MSS data field are also found to be about 700 times lower. This

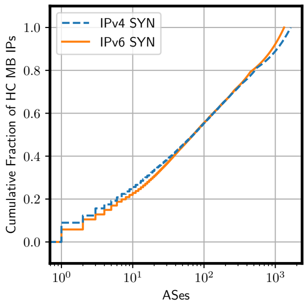

Fig. 6. AS distribution of HC MB IPs for IPv4 and IPv6 scans.

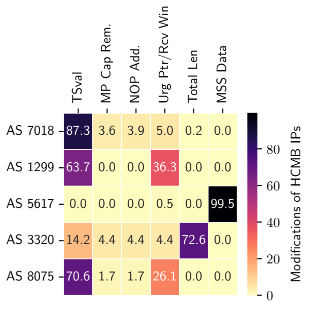

Fig. 7. Modifications by HC MB IPs from top 5 IPv4 ASes.

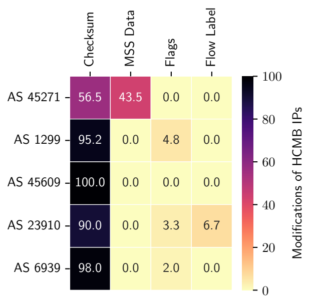

Fig. 8. Modifications by HC MB IPs from top 5 IPv6 ASes.

shows that middleboxes modify packets depending on whether the connection seems to already have been established (TCP ACK) or is currently in the handshake phase (TCP SYN).

Top IPv4 ASes: In order to understand modification patterns by ASes, we analyze the most common modifications by the top 5 ASes. Figure 7 shows a heatmap with the modifications of top IPv4 ASes. Generally, we observe that top ASes focus on one or two prominent types of modifications. This is especially the case for AS 7018 and AS 5617, with 87.3% TSval and 99.5% IP total length modifications, respectively. For AS 3320, IP total length modifications dominate with 72.6%. Upon investigating this case more closely, we find that 99.8% of modified total length values are set to 17408 bytes. We reached out to the operator, who confirmed the issue and is currently looking into it. Moreover, we analyze the MSS modifications by AS 5617 and find that those increase our original MSS of 536 to values beyond 1400 bytes, with 1460 being the most prominent choice (80.9%).

IPv6: For all four of our IPv6 scans, Traffic Class is by far the most modified field with 97% (see Appendix C for a full list). As already stated, we consider this a benign modification and not indicative of middlebox behavior.

Checksum: With 58k modifications in IPv6, the TCP checksum is observed to be most interfered with by middleboxes. Any TCP checksum alteration is expected to be a consequence of interference with other fields within the TCP header, the pseudo-header, or the payload. However, we also find instances where the TCP checksum is simply altered without any modifications elsewhere.

TCP Options: The TCP checksum modification is followed by interference with TCP options. Similar to our IPv4 scans, there is an overlap between MP CAPABLE removals and NOP additions. In fact, all 27.5k MP CAPABLE removals happen by replacement through a series of NOPs. The Timestamp and the SACK-Permitted options are next among the interfered options with 21k occurrences. Overall, for all three of these set TCP options, we find more deletions than modifications to the option data fields. However, the MSS option is an exception to this, where we find only modifications and no deletions. With IPv6 ACK probes, we find no interference of any sort with TCP options, again underlining the prominent interference of middleboxes in the handshake process.

Header Fields: Within the main transport and IP headers, we identify modifications to fields such as the TCP sequence number, the flow label and the payload length. The most interesting of these is the payload length modification occurring 2.2k times, with the TCP data offset field remaining untouched. This suggests that there are modifications exclusively to the payload length field.

TCP vs. UDP vs. ICMPv6: For our IPv6 UDP scans, the source port is found to be the most modified (except TC). We also witness modifications to the UDP checksum and the flow label field. For ICMPv6, TC modifications are dominating (8.6M), indicating that ICMPv6 traffic is more likely

to being treated differently compared to TCP or UDP. We find only 4 payload length modifications for ICMPv6, but none to ICMPv6 header fields.
Top IPv6 ASes: We show the modification distribution of the top 5 IPv6 ASes in Figure 8. For all top ASes, the most prominent modification is the TCP checksum modification, ranging from 56.5% to 100%. For ASes 6939, 1299, and 23910 we find that the checksum is changed predominantly to zero, which hints at tunneling deployments [25]. For the remaining two Indian ISP ASes the checksum value zero is not dominating: in AS 45609 we see 83 distinct checksum values, with no emerging pattern, whereas in AS 45271 80% of checksum values are set to 48395.
Other Oddities: While studying the detected interferences, we find some peculiar cases. One such observation is the IPv4 total length field being set to an extremely large value of 17408 bytes. Modifying a critical field which indicates the size of a packet is especially concerning as it can lead to incorrect parsing of responses. We find that this value might be due to a byte endianness issue of a middlebox, i.e., 17408=`0x4400` vs. 68=`0x0044`. Upon further investigation, we identify 134 MB IPs from the HCS to perform this exact modification. 82 of these 134 IPs are registered with AS 3320, i.e., Deutsche Telekom (German Tier 1 provider). The remaining IPs mapped to a total of 43 other ASes. We reached out to Deutsche Telekom, who confirmed the issue.

We see a similar unexpected modification for the IPv6 payload length field. In our IPv6 SYN scans, we observe over 2k cases wherein the field is set to 21504 (21504=`0x5400` vs. 84=`0x0054`). We find 10 potential middlebox IPs reporting this modification. We are able to map 6 of these to the highest confidence set and find them to be located across AS 24362 (CERNET2 IX at University of Science and Technology of China), AS 24348 (CERNET2 IX at Tsinghua University), AS 24360 (CERNET2 IX at Zhengzhou University), AS 24364 (CERNET2 IX at Shanghai Jiaotong University) and AS 138375 (China Education and Research Network CERNET).[7] Similarly, over 97% of the IP payload length modifications of our UDP scan have the field set to 11264 (11264=`0x2C00` vs. 44=`0x002C`). The IPs reporting these are the same as the ones seen in the IPv6 SYN scan.

We also observe the data field within the MSS option being assigned to strange MSS sizes. This happens for scans over IPv4 and IPv6, with the effect being far more pronounced with SYN compared to ACK probes. For our IPv6 probes, whenever the MSS is modified, it is always being lowered[8]. The most common of these newly assigned values is 1164 which contributes about 45% to the observed MSS alterations. We also see MSS values as low as 1000, 900 and even 40, although few and far between. Unlike IPv6, we find both increasing as well as the decreasing MSS values in IPv4. Over 75% of MSS modifications in IPv4 see the MSS being increased to 578 bytes.

4.6 Alias Resolution

Our analysis thus far only looks at middlebox IPs. However, it is possible that multiple such IPs could belong to the same middlebox. In order to determine if this is indeed the case, we perform alias resolution, i.e., the process of identifying multiple IP addresses belonging to the same device. To this end, we use three different publicly available alias datasets: CAIDA's ITDK Midar dataset [40], the Speedtrap dataset [42], and the SNMPv3 dataset [2]. We apply these three datasets to the highest confidence middlebox IPs to perform alias resolution for our IPv4 and IPv6 scans.
Alias Results: We use the Midar dataset to identify the number of multiple alias IPv4 middleboxes in the wild. We are able to map 86.1% of all HC MB IPv4 addresses to this dataset. The ITDK dataset, however, contains many singleton alias sets, i.e., where a device has just a single IP address associated to it. When ignoring these, only 25.8% of addresses can be mapped to non-singleton (NS)

[7]Similar modifications have been linked to the Great Firewall of China by previous studies [24, 26, 63].
[8]Increasing MSS values could lead to larger packets being dropped due to a lack of on-path fragmentation in IPv6.

alias sets. Our IPv4 scans reveal 223 non-singleton aliases. On average 2.4 IPs in these sets can be matched to middleboxes; the average size of NS alias sets (including MB and non-MB IPs) is 4.3.

With the IPv6-only Speedtrap dataset, we map a large number of middlebox IPv6 addresses (75.2%). Similar as with Midar, if we ignore singleton alias set, the share drops to 4.3%. These 30 devices have an average of 2.2 MB IPs per device; the average size of these NS alias sets (including MB and non-MB IPs) is 3.3.

The SNMPv3 dataset obtains aliases by using responses from unsolicited SNMPv3 requests [2]. This dataset differs from Midas and Speedtrap, as it contains IPv4, IPv6, and dual-stack alias sets. We find that about 11% of IPs from the combined IPv4 and IPv6 highest confidence set map to alias sets within the dataset. Over 10% of IPs map to non-singleton alias sets within the dataset, highlighting the different natures of these datasets. We find 132 middlebox alias sets, with an average 2.4 MB IPs per NS alias set. The average size of NS alias sets with at least one MB (including MB and non-MB IPs) is 92.2 IPs. This indicates, that these matched MB alias sets have many associated IPs and could therefore be core routers. Moreover, 38.6 % of these devices are IPv4-only middleboxes, 43.2% are IPv6-only, and 18.2% are dual-stack middleboxes. We summarize our findings in Appendix D.
Middlebox Vendors: The SNMPv3 dataset also has vendor information, which we use to map middleboxes (incl. singletons) to vendors. We find that of the 2189 devices, 54.2% are Cisco devices, 20.1% are Juniper, and 6.9% are Huawei.

5 INFLUENCE OF PROBING LOCATION

Until this point, all our findings are from scans from a single vantage point located inside our university network. This provides us only with a limited picture of middlebox deployment in the Internet. To address this, we perform similar measurements from 8 geographically distributed vantage points (VPs). We leverage AWS and choose VPs in North America (US East and US West), South America (Brazil), Europe (Germany and Sweden), Asia (India), Oceania (Australia), and Africa (South Africa). Due to the lower available resources and to avoid being blocked by AWS, we reduce our scanning rate to 1kpps to perform SYN scans for both IPv4 and IPv6. We also repeat the SYN scans at 1kpps from our university VP in Europe for comparison purposes.

5.1 Result Overview

For our IPv4 scans we see a large fluctuation in the number of replies, ranging from 92.8M (university VP) to 172.1M (South Africa). Similarly, we also see differences when it comes to the number of interferences without DiffServ modifications with more than twice as many interferences from South Africa (467.1k) compared to Brazil (222.5k). For our IPv6 scans, the fluctuation is not as pronounced as in IPv4. The replies received range from 99.2M (university VP) to 148.9M (South Africa). We see the lowest number of interferences (excl. TC) from Australia (44.8k) and the highest from our university VP (205.5k). To summarize, the VP location has a substantial effect on the number of replies as well as middlebox interferences. We refer the interested reader to Appendix E for more details on vantage point differences.

Next, we investigate the effect of location on trace gaps, i.e., how many missing responses we find in traces. More complete traces means more traces wherein all probed hops reply with ICMP messages. This, in turn, allows us to accurately detect the hops at which the interference takes place. In IPv4, the completeness of traces varies between 46.7% (Brazil) to 91.1% (US East), depending on the VP. The range is even larger in IPv6, stretching from 0.01% (US West) to 83.3% (US East) (see Appendix F). Comparing trace gaps to the absolute number of traces per VP shows that the number of traces is not a good indicator of trace completion.

VP	HC MB IPs	ASN #1	ASN #2	ASN #3
India	(3.4k, 1.8k)	(5617, 45609)	(56402, 16509)	(3320, 45271)
Germany	(3.6k, 2.4k)	(5617, 45609)	(3320, 45271)	(7018, 12252)
Brazil	(3.3k, 1.8k)	(5617, 45609)	(7018, 12252)	(3320, 45271)
US West	(3.8k, 1.6k)	(5617, 45609)	(3320, 12252)	(7018, 2914)
South Africa	(3.5k, 1.6k)	(56402, 45609)	(7018, 16509)	(3320, 12252)
Australia	(3.2k, 1.1k)	(5617, 45609)	(7018, 1299)	(3320, 55644)
Sweden	(3.6k, 1.5k)	(5617, 45609)	(56402, 12252)	(3320, 45271)
US East	(3.7k, 1.8k)	(56402, 45609)	(5617, 1299)	(3320, 9498)
University	(3.5k, 2.1k)	(5617, 45609)	(1299, 45271)	(7018, 12252)

Table 4. Number of (IPv4, IPv6) highest confidence middlebox IPs (HC MB IPs) and top three MB ASes/VP.

5.2 Locating Middleboxes

One of the findings central to this work is the number of IPs that end up in the highest confidence set. As already discussed, we categorize potential middlebox IPs into sets of varying confidence by applying the location error metric. The potential middlebox IPs for which the location error is 1 make up the highest confidence middlebox IP set. Table 4 shows the number of such IPs and the top ASes they belong to for all 9 VPs for IPv4 and IPv6. We find the number of HC IPs reported by all 9 VPs to be fairly consistent for our IPv4 scans, differing by at most around 600 middlebox IPs.
Top ASes: The top IPv4 AS for 7 out of 9 VPs is AS 5617 (Orange Polska, also seen in Section 4.4). Interestingly, at the South Africa and US East VP we see most MB modifications by AS 56402. This AS is a major ISP in Iran, which could hint at side-effects of MB interference for censorship [3, 47]. This AS is also among the top two in India and Sweden, and the majority of modifications are related to the Urgent Pointer + Receiver window and TCP Timestamp TSval classes. Furthermore, similar to our single VP measurements, we see Tier 1 ASes such as Deutsche Telekom (AS 3320), AT&T (AS 7018), and Arelion (1299) among the top three at multiple VPs.

As seen in Table 4, the top AS distribution is very homogeneous in IPv6 as well. The top AS for all VPs is AS 45609 (Bharti Airtel), the Indian ISP we already know from our single VP measurements. Other top ASes are the Indian ISP Idea Cellular (AS 45271), the Peruvian ISP Claro Peru (AS 12252), the Japanese Tier 1 AS NTT (AS 2914), the Swedish Tier 1 AS Arelion (AS 1299), another sibling AS of Bharti Airtel (AS 9498), the major UK ISP Vodafone (AS 55644), and Amazon (16509). The Amazon AS could be prominently present in our measurements due to the choice of our VP platform.
Middlebox-affected Paths: In order to quantify the middlebox impact, we analyze the paths affected by MBs. For IPv4, the percentage of MB-affected paths ranges from 6.4% at the US East VP to over 29% at the Australia VP. The findings for the two European VPs are similar, with both reporting around 20% middlebox-affected paths. Contrary to the US East VP, the US West VP reports double the number of affected paths. For IPv6, the percentage of MB-affected paths never touches 1%. It ranges from 0.05% at the VP located in Australia to 0.15% at our university VP. This again underlines that VP selection has a large effect on MB affectedness.

5.3 Middlebox Modifications

Next, we compare the type of middlebox interferences when executing measurements from different VPs. The traversal of more distinct paths by our probes allows us to get a more thorough view of middlebox behavior. At all our vantage points IPv4's DiffServ field and IPv6's TC field are most altered (93%–99% for IPv4; 95%–99% for IPv6). As with previous analysis, we do not consider these modifications middlebox behavior and therefore ignore them in the rest of the analysis.

We see the most IPv4 modifications from the South Africa VP (over 460k). At 6 out of 9 VPs, the category "IP ID/TSval/Rcv or Urg Ptr" makes up most modifications (≥ 48%). At the Germany, US East, and our university VPs the MP CAPABLE removal and NOP additions are most prevalent.

Excluding the TC in IPv6, the highest number of other interferences are reported by our university VP. In fact, the European VPs again dominate and are in the top 4 reporting at least 104k and at the most 197.9k such interferences. The most common modifications are similar across VPs and comparable to our findings from the single VP measurements (cf. Section 4.5 for more details).

Another interesting observation is the modification to the Flow Label field within the IPv6 header. We find only 4 VPs that report any such modification (2k–3.3k modifications). Three of these are in Europe and one is in the US.

5.4 ICMP Quoting Behavior

Using traces from all VPs, we provide a more comprehensive picture of ICMP quoting behavior in the wild. Since RFCs recommend different quote sizes [6, 49], it is important to have a good understanding of the current state of ICMP quoting behavior.

Our analysis of all IPv4 ICMP replies across all 9 VPs reveals that 6 out of the 9 VPs see a majority of full quotes (54%–71%). For the remaining 3 VPs we see a majority of partial quotes (56%–60%). If instead we investigate quoting behavior per replying *hop IP*, the picture changes. Now we get a very consistent quoting behavior across all VPs, with a majority of partial quotes (≈61%). Compared to previous work [43] this is an increase in full quoting IPs of 26.7 percentage points within 17 years (cf. Appendix A for details).

6 DISCUSSION AND CONCLUSION

The Past and Future of Middleboxes: Even though during the inception of the Internet, the end-to-end principle was considered important, it soon became clear, that this principle would be put in danger by the advent of middleboxes. Network operators and other players began to deploy middleboxes, which in their view *optimize* certain aspects of Internet traffic. In addition to being used by oppressive countries for censorship, middleboxes can in fact have legitimate reasons. Network operators might want to reduce latencies by deploying proxies [62, 67], rewrite the MSS field to ensure proper tunneling [48, 56], or offer additional capabilities within the WAN (e.g., MPTCP) [4, 45]. Due to abundant use cases, new middleboxes will still be deployed in the future. Thus, as our results underline, *middleboxes are here to stay* and thus it is imperative to have a timely middlebox detection tool like Yarrpbox to get a thorough understanding of their deployment.

Publication of Yarrpbox and Reproducibility: We publish Yarrpbox to aid in large-scale middlebox detection [33]. This ensures that researchers, operators, and protocol designers can quickly assess the middlebox deployment at scale. Moreover, we make analysis code and data publicly available to aid in reproducibility of our work [32].

Limitations: Although Yarrpbox improves upon Tracebox through its stateless and load-distributive probing, the tool has some limitations. Firstly, it can not pinpoint all modifications (e.g., NATs [57]). This is down to encoding state within main headers of the outgoing probe. As such, our numbers on detected interferences as well as the number of middleboxes should be treated as a lower bound. Moreover, Yarrpbox is not designed to identify load balancers that merely inspect the packets, or redundancy eliminators that operate on our state carrying payload, which makes detecting these types of middleboxes infeasible. Also, the detected middlebox location (except when we have HC IPs) can only be approximate. This is not exclusive to our solution but is also a shortcoming of Tracebox [20] and stems from missing responses. Although our collected traces show good completion, ensuring entirely gapless traces is still challenging.

Conclusion: In this paper we presented Yarrpbox, a tool to conduct Internet-scale middlebox detection. With Yarrpbox, we ran multiple large-scale measurement campaigns and found a total of 5.8k middleboxes in IPv4. Overall, nearly 10% of paths in our measurements are affected by middlebox interferences. Moreover, we conducted the first middlebox analysis of the IPv6 Internet to date, finding a lower prevalence of interferences compared to IPv4. We also characterized middleboxes by vendors and aliases. Finally, we published Yarrpbox as well as our analysis toolchain.

ACKNOWLEDGMENTS

We thank Nitinder Mohan and Kevin Vermeulen for their comments on the paper and the authors of Yarrp for providing the basis of Yarrpbox. We also thank the anonymous reviewers as well as our shepherd for their constructive feedback.

REFERENCES

[1] Ahmed Abujoda and Panagiotis Papadimitriou. 2015. MIDAS: Middlebox discovery and selection for on-path flow processing. In *2015 7th international conference on communication systems and networks (COMSNETS)*. IEEE, 1–8.

[2] Taha Albakour, Oliver Gasser, Robert Beverly, and Georgios Smaragdakis. 2021. Third time's not a charm: exploiting SNMPv3 for router fingerprinting. In *Proceedings of the 21st ACM Internet Measurement Conference*. 150–164.

[3] Simurgh Aryan, Homa Aryan, and J Alex Halderman. 2013. Internet censorship in Iran: A first look. In *3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)*.

[4] Florian Aschenbrenner, Tanya Shreedhar, Oliver Gasser, Nitinder Mohan, and Jörg Ott. 2021. From Single Lane to Highways: Analyzing the Adoption of Multipath TCP in the Internet. In *IFIP Networking Conference 2021*.

[5] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. 2006. Avoiding traceroute anomalies with Paris traceroute. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. 153–158.

[6] F. Baker (Ed.). 1995. Requirements for IP Version 4 Routers. RFC 1812 (Proposed Standard). https://doi.org/10.17487/RFC1812 Updated by RFCs 2644, 6633.

[7] Robert Beverly. 2016. Yarrp'ing the Internet: Randomized High-Speed Active Topology Discovery. In *Proceedings of the 2016 Internet Measurement Conference*. 413–420.

[8] CAIDA. 2022. Routeviews Prefix to AS mappings Dataset (pfx2as) for IPv4 and IPv6. https://www.caida.org/catalog/datasets/routeviews-prefix2as/.

[9] B. Carpenter and S. Brim. 2002. Middleboxes: Taxonomy and Issues. RFC 3234 (Informational). https://doi.org/10.17487/RFC3234

[10] Sapna Chaudhary, Prince Sachdeva, Abhijit Mondal, Sandip Chakraborty, and Mukulika Maity. 2022. YouTube over Google's QUIC vs Internet Middleboxes: A Tug of War between Protocol Sustainability and Application QoE. *arXiv preprint arXiv:2203.11977* (2022).

[11] David Choffnes, Phillipa Gill, and Alan Mislove. 2017. An empirical evaluation of deployed dpi middleboxes and their implications for policymakers. In *Proc. of TPRC*.

[12] Taejoong Chung, David Choffnes, and Alan Mislove. 2016. Tunneling for transparency: A large-scale analysis of end-to-end violations in the Internet. In *Proceedings of the 2016 Internet Measurement Conference*. 199–213.

[13] Yann Collet. [n. d.]. xxHash. https://cyan4973.github.io/xxHash/.

[14] Lucian Constantin. 2017. You Can Now Help Identify Middleboxes Holding Back TLS 1.3 Adoption. https://securityboulevard.com/2017/12/can-now-help-identify-middleboxes-holding-back-tls-1-3-adoption/.

[15] A. Conta, S. Deering, and M. Gupta (Ed.). 2006. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443 (Internet Standard). https://doi.org/10.17487/RFC4443 Updated by RFC 4884.

[16] Ryan Craven, Robert Beverly, and Mark Allman. 2014. A middlebox-cooperative TCP for a non end-to-end Internet. *ACM SIGCOMM Computer Communication Review* 44, 4 (2014), 151–162.

[17] Quentin De Coninck and Olivier Bonaventure. 2017. Multipath QUIC: Design and Evaluation. In *Proceedings of the 13th international conference on emerging networking experiments and technologies*. 160–166.

[18] Quentin De Coninck, François Michel, Maxime Piraux, Florentin Rochet, Thomas Given-Wilson, Axel Legay, Olivier Pereira, and Olivier Bonaventure. 2019. Pluginizing QUIC. In *Proceedings of the ACM Special Interest Group on Data Communication*. 59–74.

[19] S. Deering and R. Hinden. 2017. Internet Protocol, Version 6 (IPv6) Specification. RFC 8200 (Internet Standard). https://doi.org/10.17487/RFC8200

[20] Gregory Detal, Benjamin Hesmans, Olivier Bonaventure, Yves Vanaubel, and Benoit Donnet. 2013. Revealing middlebox interference with tracebox. In *Proceedings of the 2013 conference on Internet measurement conference*. 1–8.

[21] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2013. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *22nd USENIX Security Symposium (USENIX Security 13)*. 605–620.

[22] Korian Edeline and Benoit Donnet. 2017. A first look at the prevalence and persistence of middleboxes in the wild. In *2017 29th International Teletraffic Congress (ITC 29)*, Vol. 1. IEEE, 161–168.

[23] Korian Edeline and Benoit Donnet. 2019. A bottom-up investigation of the transport-layer ossification. In *2019 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 169–176.

[24] Roya Ensafi, Philipp Winter, Abdullah Mueen, and Jedidiah R Crandall. 2015. Analyzing the Great Firewall of China Over Space and Time. *Proc. Priv. Enhancing Technol.* 2015, 1 (2015), 61–76.

[25] M. Eubanks, P. Chimento, and M. Westerlund. 2013. IPv6 and UDP Checksums for Tunneled Packets. RFC 6935 (Proposed Standard). https://doi.org/10.17487/RFC6935

[26] Oliver Farnan, Alexander Darer, and Joss Wright. 2016. Poisoning the well: Exploring the great firewall's poisoned dns responses. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*. 95–98.

[27] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. 2013. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6824 (Experimental). https://doi.org/10.17487/RFC6824 Obsoleted by RFC 8684.

[28] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch. 2020. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 8684 (Proposed Standard). https://doi.org/10.17487/RFC8684

[29] Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczyński, Stephen D Strowes, Luuk Hendriks, and Georg Carle. 2018. Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists. In *Proceedings of the Internet Measurement Conference 2018*. 364–378.

[30] Hang Guo and John Heidemann. 2018. Detecting ICMP rate limiting in the Internet. In *International Conference on Passive and Active Network Measurement*. Springer, 3–17.

[31] Benjamin Hesmans, Fabien Duchene, Christoph Paasch, Gregory Detal, and Olivier Bonaventure. 2013. Are TCP extensions middlebox-proof?. In *Proceedings of the 2013 workshop on Hot topics in middleboxes and network function virtualization*. 37–42.

[32] Fahad Hilal. 2023. Yarrpbox Measurement Data. https://doi.org/10.17617/3.EVDWIT

[33] Fahad Hilal and Oliver Gasser. 2023. Yarrpbox. https://github.com/yarrpbox/yarrpbox.

[34] Ralph Holz, Jens Hiller, Johanna Amann, Abbas Razaghpanah, Thomas Jost, Narseo Vallina-Rodriguez, and Oliver Hohlfeld. 2020. Tracking the deployment of TLS 1.3 on the Web: A story of experimentation and centralization. *ACM SIGCOMM Computer Communication Review* 50, 3 (2020), 3–15.

[35] Michio Honda, Yoshifumi Nishida, Costin Raiciu, Adam Greenhalgh, Mark Handley, and Hideyuki Tokuda. 2011. Is it still possible to extend TCP?. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. 181–194.

[36] Hyunwook Hong, Hyunwoo Choi, Dongkwan Kim, Hongil Kim, Byeongdo Hong, Jiseong Noh, and Yongdae Kim. 2017. When cellular networks met IPv6: Security problems of middleboxes in IPv6 cellular networks. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 595–609.

[37] Shan Huang, Félix Cuadrado, and Steve Uhlig. 2017. Middleboxes in the Internet: a HTTP perspective. In *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 1–9.

[38] J. Iyengar (Ed.) and M. Thomson (Ed.). 2021. QUIC: A UDP-Based Multiplexed and Secure Transport. RFC 9000 (Proposed Standard). https://doi.org/10.17487/RFC9000

[39] Erin Kenneally and David Dittrich. 2012. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *Available at SSRN 2445102* (2012).

[40] Ken Keys, Young Hyun, Matthew Luckie, and Kim Claffy. 2012. Internet-scale IPv4 alias resolution with MIDAR. *IEEE/ACM Transactions on Networking* 21, 2 (2012), 383–399.

[41] Hyunwoo Lee, Doowon Kim, and Yonghwi Kwon. 2021. TLS 1.3 in Practice: How TLS 1.3 Contributes to the Internet. In *Proceedings of the Web Conference 2021*. 70–79.

[42] Matthew Luckie, Robert Beverly, William Brinkmeyer, and kc claffy. 2013. Speedtrap: internet-scale IPv6 alias resolution. In *Proceedings of the 2013 conference on Internet measurement conference*. 119–126.

[43] David Malone and Matthew Luckie. 2007. Analysis of ICMP quotations. In *International Conference on Passive and Active Network Measurement*. Springer, 228–232.

[44] Alberto Medina, Mark Allman, and Sally Floyd. 2004. Measuring interactions between transport protocols and middleboxes. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. 336–341.

[45] Olivier Mehani, Ralph Holz, Simone Ferlin, and Roksana Boreli. 2015. An early look at multipath TCP deployment in the wild. In *Proceedings of the 6th international workshop on hot topics in planet-scale measurement*. 7–12.

[46] Craig Partridge and Mark Allman. 2016. Ethical considerations in network measurement papers. *Commun. ACM* 59, 10 (2016), 58–64.

[47] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. 2017. Global measurement of {DNS} manipulation. In *26th USENIX Security Symposium (USENIX Security 17)*. 307–323.

[48] C. Perkins. 1996. IP Encapsulation within IP. RFC 2003 (Proposed Standard). https://doi.org/10.17487/RFC2003 Updated by RFCs 3168, 6864.

[49] J. Postel. 1981. Internet Control Message Protocol. RFC 792 (Internet Standard). https://doi.org/10.17487/RFC0792 Updated by RFCs 950, 4884, 6633, 6918.

[50] J. Postel. 1981. Internet Protocol. RFC 791 (Internet Standard). https://doi.org/10.17487/RFC0791 Updated by RFCs 1349, 2474, 6864.

[51] K. Ramakrishnan, S. Floyd, and D. Black. 2001. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168 (Proposed Standard). https://doi.org/10.17487/RFC3168 Updated by RFCs 4301, 6040, 8311.

[52] Ram Sundara Raman, Mona Wang, Jakub Dalek, Jonathan Mayer, and Roya Ensafi. 2022. Network Measurement Methods for Locating and Examining Censorship Devices. In *Proceedings of the 18th International Conference on Emerging Networking Experiments And Technologies*.

[53] E. Rescorla. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446 (Proposed Standard). https://doi.org/10.17487/RFC8446

[54] Jan Rüth, Ingmar Poese, Christoph Dietzel, and Oliver Hohlfeld. 2018. A First Look at QUIC in the Wild. In *International Conference on Passive and Active Network Measurement*. Springer, 255–268.

[55] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. 2012. Making middleboxes someone else's problem: Network processing as a cloud service. *ACM SIGCOMM Computer Communication Review* 42, 4 (2012), 13–24.

[56] W. Simpson. 1995. IP in IP Tunneling. RFC 1853 (Informational). https://doi.org/10.17487/RFC1853

[57] Pyda Srisuresh, Bryan Ford, Senthil Sivakumar, and Saikat Guha. 2009. *NAT Behavioral requirements for ICMP*. Technical Report.

[58] Nick Sullivan. 2017. Why TLS 1.3 isn't in browsers yet. https://blog.cloudflare.com/why-tls-1-3-isnt-in-browsers-yet/.

[59] Valentin Thirion, Korian Edeline, and Benoit Donnet. 2015. Tracking middleboxes in the mobile world with traceboxandroid. In *International Workshop on Traffic Monitoring and Analysis*. Springer, 79–91.

[60] Gareth Tyson, Shan Huang, Felix Cuadrado, Ignacio Castro, Vasile C Perta, Arjuna Sathiaseelan, and Steve Uhlig. 2017. Exploring http header manipulation in-the-wild. In *Proceedings of the 26th International Conference on World Wide Web*. 451–458.

[61] Zhaoguang Wang, Zhiyun Qian, Qiang Xu, Zhuoqing Mao, and Ming Zhang. 2011. An untold story of middleboxes in cellular networks. *ACM SIGCOMM Computer Communication Review* 41, 4 (2011), 374–385.

[62] Nicholas Weaver, Christian Kreibich, Martin Dam, and Vern Paxson. 2014. Here be web proxies. In *International Conference on Passive and Active Network Measurement*. Springer, 183–192.

[63] Fenglu Zhang, Chaoyi Lu, Baojun Liu, Haixin Duan, and Ying Liu. 2022. Measuring the Practical Effect of DNS Root Server Instances: A China-Wide Case Study. In *International Conference on Passive and Active Network Measurement*. Springer, 247–263.

[64] Johannes Zirngibl, Philippe Buschmann, Patrick Sattler, Benedikt Jaeger, Juliane Aulbach, and Georg Carle. 2021. It's Over 9000: Analyzing Early QUIC Deployments with the Standardization on the Horizon. In *Proceedings of the 21st ACM Internet Measurement Conference*. 261–275.

[65] Maya Ziv, Liz Izhikevich, Kimberly Ruth, Katherine Izhikevich, and Zakir Durumeric. 2021. ASdb: a system for classifying owners of autonomous systems. In *Proceedings of the 21st ACM Internet Measurement Conference*. 703–719.

[66] Raffaele Zullo, Antonio Pescapé, Korian Edeline, and Benoit Donnet. 2017. Hic sunt NATs: Uncovering address translation with a smart traceroute. In *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 1–6.

[67] Raffaele Zullo, Antonio Pescapé, Korian Edeline, and Benoit Donnet. 2019. Hic sunt proxies: Unveiling proxy phenomena in mobile networks. In *2019 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 227–232.

A ICMP AND ICMPV6 QUOTING

Table 5 details ICMP quoting behavior in terms of replying hop IPs as well as quote sizes at all 9 of our VPs. We see more full quotes whereas there are more partial quoters. Receiving more full quote packets while observing more unique partial quoters, means that our probes take more paths containing more full quoters than partial quoters. As mentioned in Section 5.4, there is a non-negligible increase in full quoters when compared with previous studies. We find the increase in full quote routers encouraging, even though the rate of change is rather slow. We also study the quoting behavior of ICMPv6 Time Exceeded messages. We find no observations of partial quoting behavior.

B IPV6 STATE ENCODING

Yarrpbox uses different header fields to encode values needed for its stateless operation (see Section 3 for details). Figure 9 shows the state encoding of the IPv6 probe packet as sent by Yarrpbox. In addition to the encoded values highlighted with colors, the probe packet contains the following values in the payload: The Yarrp ID field is a 4-byte string that is used to identify if the reply is in response to Yarrpbox probing. The Instance field is used to distinguish different instances of Yarrpbox running on the same machine. The Fudge field is a 2-byte field added to the payload to ensure that the TCP checksum can be set to the same fixed value across all probes.

C LIST OF MIDDLEBOX INTERFERENCES

Tables 6 and 7 show the list of interferences for IPv4 and IPv6, respectively. It contains both header fields for which Yarrpbox can pinpoint interferences as well as header fields for which interferences can not be pinpointed. The latter either carry integrity over other fields or have some sort of state encoded into them. This category does not exist for the results of our IPv6 measurement. Section 3.1.1 discusses this in detail. Note that we include DiffServ and TC modifications in the table for comparison reasons only. We do not consider these modifications to be middlebox behavior.

D ALIAS RESOLUTION RESULTS

Table 10 shows the detailed alias resolution results for the Midar, Speedtrap, and SNMPv3 datasets.

E OVERVIEW OF DISTRIBUTED MEASUREMENTS

Tables 8 and 9 show an overview of our geographically distributed measurements for IPv4 and IPv6, respectively. Please see Section 5 for more details.

	Replies			Hop IPs		
VP	792	1812	Other	792	1812	Both
India	45.4%	55.4%	0.06%	61.3%	37.9%	1.3%
Germany	57%	42.9%	0.1%	61.6%	37.8%	1.1%
Brazil	56.9%	43%	0.09%	60.1%	39.1%	1.1%
US West	44.8%	55.1%	0.08%	62.9%	36.3%	1.2%
South Africa	34.8%	65.1%	0.05%	60.6%	38.9%	1.2%
Australia	42.9%	56.9%	0.2%	62.1%	37.1%	1.3%
Sweden	39.2%	60.7%	0.06%	60.4%	38.8%	1.2%
US East	60.5%	39.4%	0.07%	61.4%	37.9%	1.1%
University	28.6%	71.4%	0.1%	61%	38%	0.9%

Table 5. Replies (left) and replying hop IPs (right) with 28-byte (RFC 792), full quote (RFC 1812), and other/both sized replies for IPv4 SYN scans.

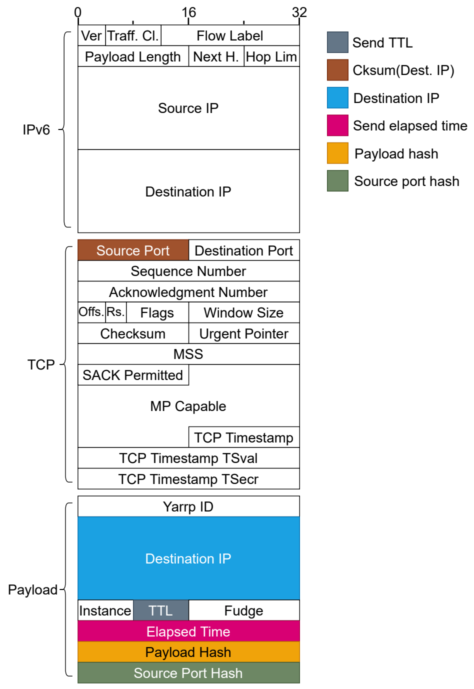

Fig. 9. Fields for state and hash encoding in IPv6.

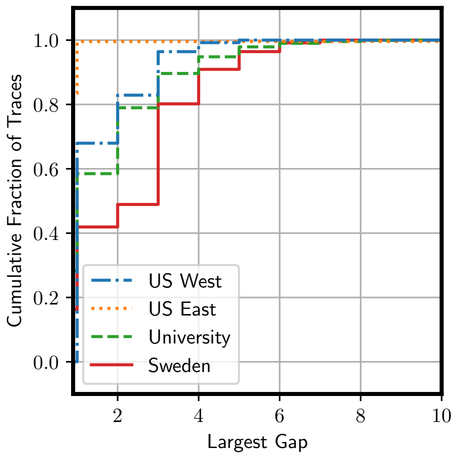

Fig. 10. Largest IPv6 trace gaps from different VPs.

Type of Interference	Number	Percentage
DiffServ	5,641,659	88.1%
IP ID/TSval/RW+UP	584,217	9.12%
TCP Tmsp. TSval	50,343	0.80%
NOP Addition	41,068	0.64%
MP CAP. Removal	38,348	0.60%
TSecr/RW+UP	16,826	0.30%
TCP UP+RW	15,147	0.20%
IP ID	5,186	-
TCP Seq. Number	3,866	-
IP Total Length	3,818	-
TCP Tmsp. Removal	1,533	-
SACK Perm. Removal	894	-
MSS Data	767	-
IP ID/TSval/TSecr/RW+UP	549	-
TCP Data Offset	464	-
TCP Flags	64	-
MSS Removal	62	-
Window Scale Addition	10	-
TCP Tmsp. TSecr	4	-

Table 6. Type of interference, the number, and the percentage of total interferences for the IPv4 SYN scan.

Type of Interference	Number	Percentage
Traffic Class	7,245,295	97.34%
TCP Checksum	58,071	0.78%
NOP Addition	36,958	0.50%
MP CAP. Removal	27,535	0.37%
TCP Seq. Number	23,735	0.31%
TCP Tmsp. Removal	21,782	0.29%
Sack Perm. Removal	21,583	0.29%
IP Flow Label	3,348	-
MSS Data	2,295	-
IP Payload Length	2,202	-
TCP Flags	293	-
TCP Tmsp. TSval	123	-

Table 7. Type of interference, the number, and the percentage of total interferences for the IPv6 SYN scan.

F TRACE GAPS IN DISTRIBUTED IPV6 MEASUREMENTS

Figure 10 shows the IPv6 trace gap distribution for different vantage points. Section 5.1 provides more information on trace gaps.

VP	Replies	Traces	Hop IPs	Total Interf.	Interf. (excl. DS)	Top Interf.	[%]	HC MB IPs
India	156.7M	11.9M	489.7K	10.26M	380.8k	IP ID/TSval/RW+UP	72.7%	3.4k
Germany	114.2M	11.9M	708.3k	17.60M	242.5k	NOP Addition	29.5%	3.6k
Brazil	123.9M	11.9M	669.2k	17.81M	222.5k	NOP Addition	31%	3.3k
US West	132.3M	11.9M	547.2k	17.30M	313.8k	IP ID/TSval/RW+UP	50.3%	3.8k
South Africa	172.1M	11.9M	461.3K	9.14M	467.1k	IP ID/TSval/RW+UP	76.3%	3.5k
Australia	140.6M	11.9M	469k	38.91M	251.3k	IP ID/TSval/RW+UP	57.6%	3.2k
Sweden	149.6M	11.9M	524.8K	34.61M	287.8k	IP ID/TSval/RW+UP	48.9%	3.6k
US East	98.2M	11.9M	414.7k	5.48M	371.9k	IP ID/TSval/RW+UP	68.6%	3.7k
University	92.8M	11.9M	698.2k	7.52M	253.7k	NOP Addition	27.8%	3.5k

Table 8. Overview of distributed IPv4 measurements.

VP	Replies	Traces	Hop IPs	Total Interf.	Interf. (excl. TC)	Top Interf.	[%]	HC MB IPs
India	146M	14.7M	208.1k	3.13M	71.4k	TCP Checksum	27.8%	1.8k
Germany	138.5M	14.7M	323.7K	12.26M	185.8k	TCP Checksum	26%	2.4k
Brazil	136.9M	14.7M	246.1k	8.82M	140.1k	TCP Checksum	26.3%	1.8k
US West	141.5M	14.7M	143.2k	7.77M	57.3k	TCP Checksum	32.5%	1.6k
South Africa	148.9M	14.7M	176.1k	6.68M	77.1k	TCP Checksum	30.2%	1.6k
Australia	135.6M	14.7M	125.6k	11.8M	37.8k	TCP Checksum	30.2%	1.1k
Sweden	146.4M	14.7M	204.1k	9.50M	106.4k	TCP Checksum	25.5%	1.5k
US East	103.1M	14.7M	78.3k	1M	44.8k	TCP Checksum	35.9%	1.8k
University	99.2M	14.4M	348.7k	7.18M	205.5k	TCP Checksum	26.7%	2.1k

Table 9. Overview of distributed IPv6 measurements.

Dataset	IPs	NS IPs	NS MBs	IPs per device
Midar	7.4k (86.1%)	2.2k (25.8%)	223	2.4/4.3
Speedtrap	9.2k (75.2%)	533 (4.3%)	30	2.2/3.3
SNMPv3	2.4k (11.4%)	2.1k (10.3%)	132	2.4/92.2

Table 10. Number of mapped IPs (IPs), non-singleton mapped IPs (NS IPs), non-singleton middleboxes (NS MBs), and average number of matched/total IPs for non-singleton middleboxes (IPs per device).

Received November 2022; revised April 2023; accepted April 2023