Hidden Treasures – Recycling Large-Scale Internet Measurements to Study the Internet's Control Plane

Jan Rüth(✉), Torsten Zimmermann, and Oliver Hohlfeld

RWTH Aachen University, Aachen, Germany
{rueth,zimmermann,hohlfeld}@comsys.rwth-aachen.de
https://icmp.netray.io

Abstract. Internet-wide scans are a common active measurement approach to study the Internet, e.g., studying security properties or protocol adoption. They involve probing large address ranges (IPv4 or parts of IPv6) for specific ports or protocols. Besides their primary use for probing (e.g., studying protocol adoption), we show that—at the same time—they provide valuable insights into the Internet control plane informed by ICMP responses to these probes—a currently unexplored secondary use. We collect one week of ICMP responses (637.50M messages) to several Internet-wide ZMap scans covering multiple TCP and UDP ports as well as DNS-based scans covering >50% of the domain name space. This perspective enables us to study the Internet's control plane as a by-product of Internet measurements. We receive ICMP messages from ∼171M different IPs in roughly 53K different autonomous systems. Additionally, we uncover multiple control plane problems, e.g., we detect a plethora of outdated and misconfigured routers and uncover the presence of large-scale persistent routing loops in IPv4.

1 Introduction

Internet scans are a valuable and thus widely used approach to understand and track the evolution of the Internet as one of the most complex systems ever created by humans. They are widely applied in different fields, including networking and security research: e.g., to find vulnerable systems [9], to measure the liveness of IP addresses [3], or to measure the deployability of new protocols, features [11], or their evolution [33]. Advances in scanning methodologies enabled probing the entire IPv4 address space for a single port within minutes or hours, depending on the available bandwidth and configured scan rate (see tools such as ZMap [10] or MASSCAN [18]). Thereby, regular scans of the entire IPv4 address space have become feasible, e.g., providing an insightful perspective into protocol evolution (see e.g., QUIC [31]). This line of scan-based works has created a rich body of contributions with valuable insights into Internet structure and evolution. These works have in common that they focus on one particular feature or protocol as their objective to study (*primary use*).

D. Choffnes and M. Barcellos (Eds.): PAM 2019, LNCS 11419, pp. 51–67, 2019.
https://doi.org/10.1007/978-3-030-15986-3_4

Table 1. Weekly scan schedule fueling our dataset, DNS-based scans use our own resolver infrastructure. For IPv4-wide scans, we utilize ZMap.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Source	DNS						
Protocols & Ports	TCP/443, gQUIC/443						
Source	Alexa	1% IPv4	IPv4				
Protocols & Ports	TCP/80, TCP/443	TCP/80, TCP/443	TCP/80	iQUIC/443	gQUIC/443	TCP/443	

In this work, we argue that Internet-wide scans have a less explored *secondary use* to study the Internet control plane while scanning for their primary use, e.g., to detect routing loops while *primarily* probing for QUIC-capable servers. That is, we study Internet control plane responses sent via ICMP as response to non-ICMP probe packets (e.g., QUIC) and show that Internet-wide scans are a hidden treasure in that they produce a rich ICMP dataset that is currently neglected, e.g., to uncover network problems. The interesting aspect is that these ICMP-responses are a valuable secondary use that is generated as by-product of any Internet-wide scan. They thus enable to study the Internet control plane (e.g., to detect routing loops) without requiring dedicated scans (as performed a decade ago [20,36]) that would increase the scanning footprint.

Our observations on the Internet's control plane are fueled by regular ZMap scans of the IPv4 address space for multiple TCP and UDP ports as well as DNS-based scans of top lists and zone files for mainly TLS, HTTP/2, and QUIC. We evaluate one full week of ICMP responses to multi-protocol Internet-scans covering the entire IPv4 address space and >50% of the domain name space(base domains).

Our contributions are as follows:

- We propose to use Internet-wide scans to study the Internet control plane via ICMP response, e.g., to detect routing loops or misconfigurations.
- Within our one week observation period, we collect ~637.50M ICMP messages which we make available at [22].
- We shed light on how Internet-scans trigger ICMP responses across the Internet.
- Our data shows a plethora of misconfigured systems e.g., sending ICMP redirects across the Internet or producing deprecated source quench messages.
- We find many networks and hosts to be unreachable, our scans uncover large sets of unreachable address space due to routing loops.
- We provide a growing ICMP dataset at https://icmp.netray.io.

Structure. The next section (Sect. 2) starts by providing an overview of our ICMP dataset. Following this, we dive into our dataset and dissect it (Sect. 3). Driven by our findings, we inspect unreachable hosts due to routing loops and quantifies their presence in today's Internet (Sect. 4). Finally, we discuss related works (Sect. 5) and conclude the paper (Sect. 6).

2 Scan Infrastructure and Dataset

Infrastructure. Our scans are sourced by two different modes, on the one hand, we use the ZMap [10] port scanner on multiple machines to perform different scans over the course of a week, and on the other hand, we continuously probe >50% of the DNS space. Table 1 summarized our weekly scan schedule, we did not specifically create these scans and this schedule for this paper, it is the result of ongoing research efforts.

These scans typically involve scanning TCP/80 for TCP initial window configurations [30] or TCP fast open support. Further, we investigate TCP/443 for HTTP/2-support [37] and TLS, additionally, we scan on UDP 443 for Google QUIC (gQUIC) [31] and IETF-QUIC (iQUIC). Our DNS-based scans are fueled by using our own resolvers to resolve various record types for domains listed in zone files of multiple TLDs (e.g., .com, .net, .org), which can be obtained from the registries, and we use A-records to investigate TLS, HTTP/2, and gQUIC. All of our scans including the DNS resolutions originate from a dedicated subnet. To collect all ICMP traffic that is directed towards these hosts, we install a mirror port at their uplink switch and filter it to only contain ICMP traffic that belongs to our measurement network. Since we perform no measurements that generate ICMP messages themselves, we exclude those sent from our host (only ping responses) leaving us with only incoming ICMP traffic.

Dataset. We base our observations on one full week in September 2018. In this week we received 169 GB resp. ~637.50M ICMPv4[1] messages (excluding those explicitly triggered in Sect. 4). ICMP messages follow the structure shown in Fig. 1, they are fundamentally made up of a type field and, to further specify a subtype, a code field, and depending on their value additional information may follow.

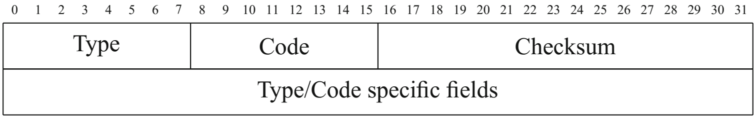


Fig. 1. ICMP header structure. Type and this type's sub type (code) determine message contents, e.g., often packets triggering the ICMP message are quoted.

[1] Please note that we do not have a fully IPv6-capable measurement infrastructure and thus focus on IPv4 only.

Table 2. ICMP types with their occurrence frequency in our dataset. Ordered by frequency.

Type	Count	Uniq. IP	Uniq. AS
Dest. Unreach.	476.68M	170.30M	52.92K
TimeExceeded	139.53M	455.13K	18.40K
Redirect	18.12M	243.25K	2.29K
EchoRequest	3.12M	10.64K	861
SourceQuench	46.18K	2.65K	364

Type	Count	Uniq. IP	Uniq. AS
EchoReply	6.08K	301	58
Other	1.48K	606	43
TimestampReq.	73	9	6
Param.Problem	20	16	9
Addr.MaskReq.	4	1	1

3 Study of ICMP Responses

To begin our investigations, we first summarize the ICMP responses to our scans by looking at the distribution of ICMP message types and their frequency of occurrence in Table 2. We observe 75 different ICMP type/code combinations during our observation period with significantly different occurrence frequencies. While we mostly receive standardized ICMP messages, we also receive some messages for which we could not find a standard, summarized as *Other* in Table 2, on which we do not further focus in this paper. The table lists the total count of these messages as well as the number of unique source IPs (router/end-host IPs) that generated the messages and number of ASes they are contained in. Over the course of the week, we run different scans. Notably, on Sundays and Mondays (see Table 1), no IPv4-wide ZMap scans are performed.

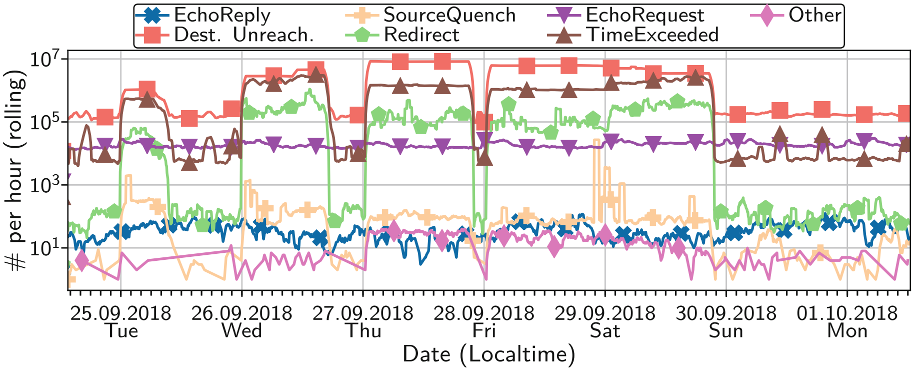


Fig. 2. Number of ICMP messages receiver per hour and type over the course of a week. Note the log scale and that we used a rolling sum over 1 h.

Figure 2 thus puts the data from Table 2 into a temporal context showing the rolling sum over 1 h intervals of the major ICMP types. We observe that the ICMP traffic varies over the course of the week, e.g., echo requests are rather static, other types like destination unreachable mainly follow our ZMap scan schedule.

Quoted IP Packet. Apart from the different ICMP types, many ICMP messages contain parts of the packet that caused the creation of the messages. We further inspect these quoted IPv4 packets within the ICMP messages. From all received ICMP messages, 99.5% are supposed to contain IP packets (according to the RFCs), of these only 0.07% cannot be decoded, e.g., because there is simply not enough data or these are no IPv4 packets. Of the decodable packets, we find 180.25M unique source IP/payload length combinations, 76% are longer than 40 bytes, i.e., enough to inspect IP and TCP headers when no options are used[2], 24% are exactly 28 byte long, so just enough to inspect the transport ports. Thus, when no options are used, the chances are high that ICMP messages received by an ICMP receiver can be demultiplexed to the respective application process. This extends the finding in [26] that showed a prevalence of 28 byte responses for TCP `traceroutes`. Next, we focus on the destination address field within the quoted IP header. These should correspond to addresses which are targeted by our scanners. Interestingly, from all ICMP messages, we find over 1.06M messages with destination IPs that are in reserved address space, i.e., unallocated or private addresses (e.g., 192.168.0.0/24). Since all our scanners explicitly blacklist these IP addresses, we believe that these messages are produced behind network address translations (NATs). We next use the contained source addresses to understand the relation to our measurements.

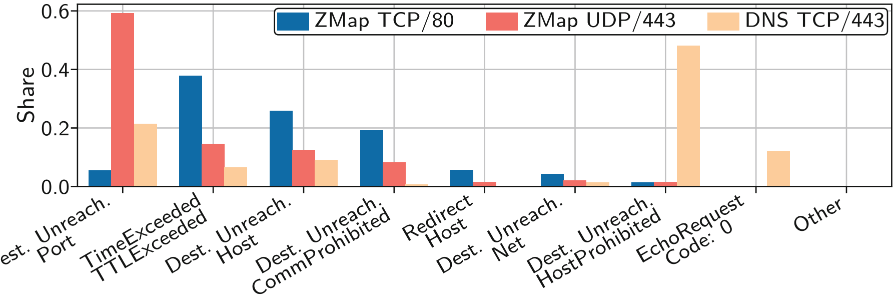


Fig. 3. ICMP messages triggered by ZMap and DNS-based scans.

Takeaway. *ICMP traffic shows a temporal correlation to measurement traffic, most messages indicate unreachability. In our collected dataset, quoted IP packets typically contain enough information to inspect everything up to the end of the TCP header. Further, a substantial number of messages seems to be generated behind NATs allowing to peek into private address spaces.*

3.1 Responses to Individual Measurements

Since we perform a variety of different measurements independent of this study, our first investigation is how different measurements affect the generation of

[2] To reduce the capture size, our packet capture caps packets at 98 byte allowing no further investigation, we find 67% having the maximum capture size.

ICMP traffic. To this end, we compare two ZMap scans and a purely DNS-based scan. For the ZMap scans, we focus on one that enumerates reachable TCP port 80 (HTTP) and UDP port 443 (QUIC) hosts, for DNS, we use a scan that probes for HTTP/2 support via TCP port 443. We are able to clearly tie the ICMP messages to the different scans via IPs and ports either from the quoted IP message or from IP itself.

Figure 3 shows the distribution of ICMP types *and* codes (top 8) that we receive for the respective scans. As already indicated by Table 2, we receive a large amount of destination unreachable messages. However, depending on the scan, their amount and share greatly vary, especially when looking at the respective code. For example, unreachable ports are very common for our UDP-based ZMap scan, however, in comparison, the TCP-based ZMap scan shows only a small fraction of unreachable ports. This is no surprise as TCP should reply with a RST-packet if a port is unreachable and does typically not generate ICMP messages. In contrast, there is no such mechanism in UDP, even through something comparable to TCP's RST exists in QUIC. However, QUIC is implemented in user-space, thus when the kernel cannot demultiplex a packet to a socket it must resort to issuing an ICMP unreachable message. Looking at our DNS-based scan, we still find that more than 20% of the ICMP messages signal unreachability through ICMP in contrast to TCP RSTs, something that, e.g., the default ZMap TCP-SYN scan module simply ignores in contrast to its UDP counterpart. Since in all major operating systems TCP handles signaling closed ports, we believe that these hosts issuing ICMP replies are actively configured either in their own firewalls (e.g., iptables) or in a dedicated firewall to do so. We find only 16.49K IPs issuing *all* 1.13M ICMP port unreachable messages, supporting our assumption that dedicated machines filter this traffic.

Looking at the other types/codes, we find that a non-negligible share of ICMP messages indicate that hosts are not reachable via the Internet either due to TTLs expiring or because their host or network cannot be reached. Apart from this, we observe that TCP port 443 is often firewalled (HostProhibited).

Takeaway. *Depending on the protocol and port, we get different feedback from the Internet's control plane. Our findings indicate that, e.g., ICMP port unreachable messages should not be ignored for TCP-based scans as is currently the case.*

3.2 ICMP Echos

ICMP echo requests (Type: 8) are the typical `ping` to which an echo reply is sent. RFC792 defines only a single code point, i.e., code = 0 which represents "no code", still we observe some non-standard code points. Some security scanners use non-standard code points for operating system fingerprinting, e.g., a standard Linux will echo the requested code point in its reply. Still, pings to our measurement infrastructure seem quite common, for code = 0, we find 10.57K unique IPs out of 840 autonomous systems (ASes). It seems that our scanning activities trigger systems to perform ping measurements towards us, yet, we do not know their actual purpose. We suspect that this could be caused by intrusion detection systems (IDSs) that monitor the liveness of our hosts.

Echo Replies. Since our hosts do not perform echo requests, we were surprised to find echo replies in our dataset. We observe different code points with different frequencies but overall we find over a couple of thousand of these replies. To investigate what causes these seemingly orphaned messages, we inspect their destinations. Since our measurements are identifiable either by IP or additionally by weekday, we associate messages to measurements. We find most echo replies are with code = 3 (except for 5 messages), all 5.75K of these echos are destined to our DNS resolvers and originate from only 86 IP addresses in 2 Chinese ASes. While many ICMP packets contain IP quotations, echo replies typically do not, they usually mirror data contained in the echo request. Yet, we still find IP packets together with DNS query *responses* that are destined to our resolver. Thus, it seems that the packets are generated on the reverse path, however, they are not sent back to the source (DNS server) but they are forwarded to the destination (us). Inspecting the source IP within the IP fragments, we find IP addresses from the same two ASes, as it turns out the 88 ICMP source IPs all respond to DNS queries which hints at their use as a DNS server cluster. Yet, we were unable to manually trigger these ICMP reply packets when trying to send DNS requests to these IPs, we only observed that DNS requests were always answered by two separate packets from the same IP, however, with different DNS answers. Further, the packets seem to stem from different IP stacks (significantly different TTLs, use of IP ID or not, don't fragment bit set or not). While the different stack fingerprints could be the result of middleboxes altering the IP headers, the general pattern that we observe hints at DNS spoofing.

3.3 Source Quench

ICMP Source Quench (SQ) messages (Type: 4, Code: 0) were a precursor of today's ECN mechanism, used to signal congestion at end-hosts and routers. The original idea (RFC792 [28]) was that a router should signal congestion by sending SQ messages to the sources that cause the congestion. In turn, these hosts should react, e.g., by reducing their packet rate. However, research [12] found that SQ is ineffective in e.g., establishing fairness and IETF has deprecated SQ-generation in 1995 [2] and SQ-processing in 2012 in general [17]. Major operating systems ignore SQ-messages for TCP at least since 2005 to counter blind throughput-reduction attacks [16]. Further, [13] claims that SQ is rarely used because it consumes bandwidth in times of congestion.

In our traces, we observe 2.65K unique IPs located in 364 ASes issuing SQ messages, despite the deprecation. Out of these IPs, 34.42% are located in only 5 ASes. Moreover, 609 IPs that generate SQ messages were directly contacted by our measurement infrastructure, i.e., are the original destination of the request causing this SQ message (according to the IPv4 header contained within the ICMP message). Among the remaining SQ messages, we find a few messages where the original destination and the source of the SQ messages are located in ASes of different operators, i.e., possible transit networks. Exemplarily, we observe that IPs located in AS1668 (AOL Transit Data Network) and AS7018 (AT&T) issued SQ messages when IPs located in AS8452 (Telecom Egypt) were contacted. As a final step, we see that 53 destination IPs in our measurements trigger the generation of SQ messages and are also contained in A-records of our DNS data that we collect. Out of these 53 IPs, 22 IPs generated the SQ messages themselves, i.e., no on-path intermediary caused the creation of this message.

In addition, we checked how vendors implement or handle this feature. Cisco removed the SQ feature from their IOS system after Version 12 in the early 2000s [5]. Hewlett Packard's cluster management system (Serviceguard) generated SQ messages due to a software bug in a read queue, which was fixed by a patch in 2010 [21]. In their router configuration manual (September 2017), Nokia also marks SQ messages as deprecated [27]. Although we cannot identify devices and their operating system version in our measurements, we assume that some devices are not updated to a current version or are following a configuration that enables them to generate SQ messages. This is not forbidden per se but given that ICMP SQ creation was deprecated over 20 years ago, our findings highlight that removing features from the Internet is a long term endeavor.

3.4 Redirect

ICMP redirect messages (Type: 5), are sent by gateways/routers to signal routes to hosts. While [15] finds networks which require redirect messages to be architected sub-optimally in the first place, RFC1812 [2] states that a router *must not* generate redirect messages unless three properties are fulfilled: *(i)* The packet is being forwarded out the same physical interface that it was received from, *(ii)*, the IP source address in the packet is on the same logical IP (sub)network as the next-hop IP address, and *(iii)*, the packet does not contain an IP source route option. Similar checks [4] are used by receiving hosts to check the validity of the message (e.g., redirected gateway and issuing router must be on the same network).

Since none of the 18.12M redirect messages originate from our network, the routers generating them either violate rule *(ii)* or some obscure address translation is in place on their networks. In our data, we even find roughly 2.7K unique redirects to private address space. Within our dataset, we observed 105.78K network redirects and 18.01M host redirects. Network redirects are problematic since no netmask is specified and it is up to the receiving router to interpret this correctly. For this reason, RFC1812 [2] demands that routers *must not* send this type. We find that the network redirects originate from 238 different ASes affecting nearly 19k different destinations of which less than 20 are mapped in any of our DNS data. Yet, all these ASes thus contain questionable router configurations that are outdated at least since 1995. Similarly, we find that the much larger fraction of host redirects originate from 2.20K ASes that affected over 400k destinations of which we find roughly 900 mapped in our DNS data. This suggests that a substantial number of end-systems are connected via sub-optimally architected or misconfigured networks.

Table 3. ICMP messages received indicating some form of unreachability with known type and code ordered by frequency.

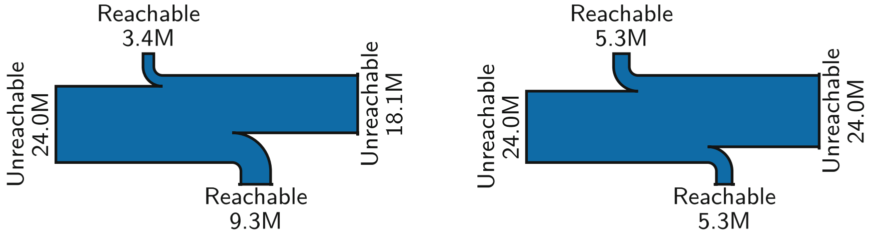
Type	Code	Count
Dest. Unreach.	Port	256.72M
TimeExceeded	TTLExceeded	139.52M
Dest. Unreach.	Host	107.15M
	CommProhibited	71.70M
	HostProhibited	23.07M
	Net	17.94M
	Protocol	51.04K
Dest. Unreach.	Frag.Needed	26.66K
	NetProhibited	26.28K
TimeExceeded	Frag.Reassembly	7.31K
Dest. Unreach.	HostUnknown	336
	NetTOS	25
	NetUnknown	6
	SourceIsolated	2

3.5 Unreachable Hosts

Reachability is a fundamental requirement to establish any means of communication. Given that Table 2 lists 476.68M destination unreachable messages this looks troublesome at first. Yet, not all unreachability is bad, e.g., firewalls actively protect infrastructure from unpermitted access, i.e., when iptables *rejects* a packet (in contrast to simply dropping it) it generates an ICMP response. By default, a port unreachable message (Type: 3, Code: 3) is produced but other types can be manually specified by the network operator. Our scans in themselves certainly trigger a certain amount of firewalls or some IDSs. In contrast, when a path is too long and the IP TTL reaches zero, routers typically generate an ICMP TTL exceeded message indicating that the destination is not reachable but this time due to the network's structure. Similarly, ICMP destination unreachable messages for host unreachable (Type: 3, Code: 1) should indicate that there is currently simply no path to a host, e.g., because it is not connected or the link is down. Table 3 summarizes the unreachability that we observe in our dataset.

As already indicated in Sect. 3.1, our UDP-based ZMap scans have the highest share of port unreachable messages putting them at the top. We inspect the origin of the messages and the actual destination that our scans targeted to see if the end-hosts generate the messages or an intermediate firewall. It seems that 96% of the messages are indeed generated by end-hosts or machines that can answer on their behalf (NATs).

Host and Network. Unreachable hosts and networks codes are used to give hints that currently no path is available and the RFCs explicitly note that this may be due to a transient state and that such a message is not proof of unreachability. To check for transient states, we compare the unreachable hosts on Thursday with those on Friday in our ZMap (both UDP 443) scan and additionally with the same scan (Thursday) one week later (captured separately from our initial dataset) and investigate if hosts become reachable that were unreachable before or vice versa.



(a) Thursday to Friday. **(b)** Thursday to Thursday one week later.

Fig. 4. Different scans (left to right of each plot) trigger different amount of host unreachable messages. (a) Compares the changes within one day. (b) Within one week.

Figure 4 visualizes the change between these two days (a) and within one week (b) for host unreachable messages. We can see that within two days, the majority of hosts remain unreachable, a small number of hosts that were previously *reachable*[3] become unreachable, similarly, previously unreachable hosts become reachable. Looking at the changes within a full week, we observe that the total amount of unreachable hosts stays the same, however, roughly the same amount of previously reachable host become unreachable and vice versa. To dig into these once unreachable and then reachable hosts, we inspect to which AS they belong finding that 82% of all hosts are from the same ASes. A possible explanation might be that while our observations seem to indicate a change, the ICMP message generation is subject to rate-limiting [19]. Thus there might be routers that generated unreachable messages on Thursday for a certain host, however, this router could be subject to rate-limiting on Friday for the same host or the week after leading to a false impression of reachability and continuity, still, a substantial number of hosts remain unreachable. Another possibility is that some hosts are only up at certain times of the day leading to differences in the reachability.

Time Exceeded. Similar to host unreachability, Time Exceeded messages (Type:11) indicate unreachability but due to network issues. Either the Fragment Reassembly (Code: 1) time was exceeded, i.e., the time that IP datagrams are buffered until they can be reassembled when IP fragmentation happens, or the TTL runs out (Code: 0), i.e., the path length exceeds the sender-defined limit. For the former, we find some thousand messages but they stem from only 30 ASes, since many of our scans use small packets, fragmentation is unlikely in

[3] With reachable we actually mean *not unreachable*, i.e., we do not get ICMP unreachable messages, which must not mean that this host was reached by the scan.

the first place. Yet, for example, the UDP ZMap scans use roughly 1300 byte per packet which is in the range of typical [7] MTUs when fragmentation could occur. Since the default ZMap functions to create IP packets (which we use) do not set the *don't fragment bit*, only some of our measurements trigger the 26.66K *fragmentation needed and DF set* ICMP messages (see Table 3). However, over time, these ICMP messages could give valuable insights into path MTU in the Internet.

TTL Exceeded messages have the second largest occurrence (139.52M) within our dataset. They were produced in 18.40K different ASes covering 35.5M different destinations that our scans tried to reach of which ∼32K are again present in A-records of our DNS data and are thus unreachable. We inspect the TTL field of the quoted IP packets that triggered the ICMP messages to see if the TTL was actually zero when the message was generated. To do so, we first generate all unique pairs of router IP and TTL values and then count the different TTLs observed. Out of these, 97% of the TTLs show a value of one, followed by ∼2.4% with a zero, we expect these two, since a router should drop a TTL = 0 or, depending on the internal pipeline, also TTL = 1, when the packet is to be forwarded. Nevertheless, we also find larger TTLs, 2, 3, 4, 5, and 6 directly follow in frequency, yet, we also find some instances of over 200 or even 255. The very large TTLs could hint at middleboxes or routers rewriting the TTL when they generate the message to hide their actual hop count. The lower numbers could be indicators for MPLS networks. By default, e.g., Cisco [6] and Juniper [24] routers copy the IP TTL to the MPLS TTL on ingress and also decrement the IP TTL within the MPLS network. It is possible to separate IP TTL and MPLS TTL and there are heated discussions whether one should hide the MPLS network from traceroutes or not which has also been subject of investigations [8]. Thus packets expiring within an MPLS network will still trigger an ICMP TTL exceeded, however, the quoted IP packet will have the TTL value they had at the MPLS ingress router, thus, if the IP TTL is still copied at ingress a traceroute could still reason about an MPLS network.

Since we were surprised to see this many TTL exceeded messages across all scanner types (see Sect. 3.1), we checked our scanners to see which TTL they were actually using to see if our setup simply has too small values. All our ZMap-based scanners initialize the TTL field with its maximum of 255 possible hops, all scanners building on top of the transport layer interfaces, in contrast, use the current Linux default of 64 hops as also recommended in RFC1700 [29]. Given that we are at least on the recommended hop count, this leaves us with three possibilities, *(i)* the current recommendation of 64 is too low to reach these hosts, *(ii)* there are middleboxes modifying the TTL to a much lower value, or, *(iii)* there are routing loops on the path to these hosts. After shortly summarizing our findings, we continue by exploring the latter.

3.6 Summary

As the previous sections have shown, our Internet-wide scans produce an insightful *secondary* dataset of ICMP responses. Driven by these messages, we identified a potential DNS spoofer, found that long deprecated source quench messages are still generated in today's Internet and that ICMP redirects are sent across different administrative domains pointing to several outdated and misconfigured networks. Without crafting a dedicated dataset, our scans enable us to study Internet reachability and we believe that longitudinal studies offer a way to deal with the challenge of ICMP rate-limiting.

4 Routing Loops

Routing loops are an undesirable control plane misconfiguration that render destination networks unreachable and that challenge a link's load [35]. In essence, IP's TTL protects the Internet from indefinitely looping packets and thus ICMP TTL messages inform the sender that a router dropped a packet after exceeding the allowed number of router hops (TTL). While the potential for routing loops is known, only a few studies investigated their presence a decade ago [20,36], current information on the presence and prevalence is missing. Therefore, we study routing loops on the basis of ICMP TTL exceeded messages triggered by our scans. We further argue that routing loops can be frequently investigated as a by-product of Internet-wide scans that are regularly conducted for different purposes.

4.1 Methodology: Detecting Loops

ICMP TTL exceeded messages are not necessarily caused by loops, also overly large paths or middleboxes could trigger these messages. To investigate whether or not an actual loop is present, we perform `traceroutes` for the original destinations (in the quoted IP) of the ICMP TTL exceeded messages. Since our traceroutes are subject to ICMP rate-limiting, especially when packets start to loop, we customize traceroute. Our traceroute slows down its sending rate when detecting an already seen IP address (loop indicator), otherwise, it follows the design of *Paris traceroute* [1] reusing flow identifiers for each hop to trigger the same forwarding behavior in ECMP-like load balancers.

Since the traceroutes can still be noisy due to hosts that do not generate ICMP at all or are still subject to rate-limiting, especially when also other traffic flows into a loop, we put strong demands on our loop. For each hop on the path that does not generate a reply, we assign a new unique label, all others are simply labeled by the answering IP. From this list of labels, we create a directed graph connecting each label-induced node to its successor and on this path we compute all elementary cycles using [23]. On an elementary cycle, no node appears twice except that the first and last node are the same. Then, on each of these possible cycles we inspect the node with the highest degree, and if this node's degree is larger than 5[4], we mark this traceroute as having a loop. This will yield

[4] This is basically a precaution against bad load balancers traded against the required TTL.

loops as long as at least one router in the loop generated ICMP TTL exceeded messages, which we found to work reasonably well when traceroute pauses the packet generation for at least 500 ms when observing an already seen IP address. Thus in a loop of two routers, we will send each router a packet roughly every second.

4.2 Routing Loops in the Wild

We seed our traceroutes by ICMP TTL exceeded messages generated from our Internet-wide scans[5]. Since we get way too many TTL exceeded messages to traceroute them all without generating substantial rate-limiting, we restrict us to a single traceroute for each unique /24 subnet within 30-min intervals. Thus for two TTL exceeded messages for a destination from the same /24 subnet, we only perform a single traceroute if the messages arrive within 30 min.

For our assessment of routing loops, we investigate TTL exceeded messages in the last week of August 2018. To avoid rate-limiting we also limit our traceroutes that we perform in parallel; generating all traceroutes for this single week took us until the end of September 2018. While this skews our data, it enables us to reason about the persistence of these loops since every 30 min the same /24 could be scheduled for a rescan. In total, we performed ∼27M traceroutes to ∼612K different /24 subnets from 28K ASes, of these, 439K subnets from 19.8K ASes are unreachable due to a loop. We further inspect how many loops are present and if loops are only within a single AS or whether loops cross AS borders and are thus potentially on a peering link. To do so, we count the number of distinct loops and ASes involved in the loops and find 167K different loops in 13.9K ASes. Of these loops, 136K have IPs for all routers involved in the loop, thus allowing an in-depth inspection. Looking at the ASes involved, we find that 13% (17.7K) already cover all different ASes paths involved (i.e., we replaced each IP by the respective AS), of these 4.8K cross AS boundaries. The top three ASes involved in the loops are AS171 (Cogent) a Tier-1, AS9498 (BHARTI Airtel Ltd.), an Indian ISP, and AS3549 (Level 3), again a Tier-1.

Persistence. To investigate the persistence, we restrict our view to traceroutes that were performed two weeks after the initial TTL exceeded message was triggered by our Internet-wide scans. In contrast to our previous observation, loops from roughly 150 ASes disappear, yet, we still find 4.6K loops crossing AS borders, in total still rendering 404K subnets unreachable. Thus, most loops seem to persist and are not resolved.

Loops at our Upstream ISP. Within our data, we also found loops in the AS of our upstream ISP. We contacted the ISP about our findings which they were able to confirm. Since many of the loops are outside of their administrative domain even though they manage the address space, they were still able to give

[5] Our dataset excludes TTL exceeded messages generated by these traceroutes.

us more details on a loop that they were able to fix. For one loop, they found that the first router had a static route for our tested destination towards its next hop, yet, the next hop had no specific forwarding information for this destination and thus used its default gateway, which however was the previous router with the static route thus causing the loop.

Takeaway. *Routing loops seem to persist in large parts of the Internet, challenging the question if the address space cut off by the loops is in use after all or if there are other routers that would be taken from different vantage points. We believe routing loops have a huge potential for causing congestion when exploited and thus a persistent monitoring seeded by large-scale Internet measurements that informs operators could be a long-term attempt to reduce routing loops.*

5 Related Work

Our work relates to approaches analyzing ICMP traffic and its generation in general, as well as approaches that focus on particular studies built upon ICMP, e.g., path/topology discovery and routing loops. In the following, we discuss similarities and differences to our work but we remark that the body of works building on top of ICMP is far larger but conceptually differ in that they do not analyze ICMP as a by-product.

Bano et al. [3] also use ZMap and capture *all* (cross-layer) responses to probe traffic to infer IP liveness but run specific measurements to generate this traffic, we believe that our dataset could be used to perform a similar analysis. Malone et al. [26] analyze the correctness of ICMP quotations. They base their analysis on a dataset obtained via `tcptraceroute` in 2005, targeting around 84K web servers. While most of the reported messages are of type ICMP time exceeded, they also find around 100 source quench messages, which were already deprecated then. As we have shown, by looking at the ICMP responses to Internet-wide scans, we are able to update their findings on a regular basis without having to craft a dedicated dataset. Guo et al. [19] present FADER, an approach to detect the presence of ICMP rate-limiting in measurement traces. While we did not focus on rate-limiting, we found indicators for rate-limiting. We believe that longitudinal studies seeded by Internet-wide scans can, in the long run, help to overcome limited visibility due to rate-limiting.

In 2002, Hengartner et al. [20] have characterized and analyzed the presence of routing loops in a Tier-1 ISP backbone trace. Xia et al. [35,36] have further tracerouted over 9M IP addresses to find routing loops in 2005. Transient routing loops have also been subject to investigation [34] and they are well studied [14,32]. Lone et al. [25] investigate routing loops in CAIDA data to study source address validation but do not focus on their prevalence in the Internet, further, in contrast to using the CAIDA dataset that actively runs traceroutes against all /24, we utilize indications from ongoing measurement data to investigate loops. While these works show that routing loops are a known problematic misconfiguration, their presence in the Internet has not been analyzed for over

10 years. By recycling Internet-wide scans, we can seed such investigations and enable persistent monitoring of this phenomenon showing that routing loops are still a problem today.

6 Conclusion

In this paper, we used ICMP responses triggered by large-scale Internet measurements to study how the Internet's control plane reacts to these measurements. Thereby, we found that these responses are hidden treasures that are typically neglected but offer great insights into the configuration of Internet-connected systems. Our analyses of different ICMP responses led us to many misconfigured routers, e.g., sending ICMP redirects across the Internet, or outdated systems, e.g., generating long-deprecated source quench messages. Further, our analysis showed a large and nuanced degree of unreachability in the Internet. More specifically, our scans hint at the existence of routing loops, which we found to persist in large parts of the Internet. We hope that these ICMP by-products are analyzed by more researchers when performing large-scale measurements and that the regular nature of these scans will enable persistent monitoring of the Internet's control plane and that, especially when brought to the attention of operators, misconfigurations can be fixed. To this end, we make our dataset publicly available at [22].

Acknowledgments. Funded by the Excellence Initiative of the German federal and state governments, as well as by the German Research Foundation (DFG) as part of project B1 within the Collaborative Research Center (CRC) 1053—MAKI. We would like to thank the network operators at RWTH Aachen University, especially Jens Hektor and Bernd Kohler as well as RWTH's research data management team.

References

1. Augustin, B., et al.: Avoiding traceroute anomalies with Paris traceroute. In: ACM IMC (2006)
2. Baker, F.: Requirements for IP Version 4 Routers. RFC 1812, RFC Editor (1995)
3. Bano, S., et al.: Scanning the internet for liveness. SIGCOMM CCR **48**(2), 2–9 (2018)
4. Braden, R.: Requirements for Internet Hosts - Communication Layers. RFC 1122, RFC Editor (1989)
5. Cisco: IP Routing Frequently Asked Questions. https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/28745-44.html#qa5
6. Cisco Systems, Inc.: Cisco IOS XR MPLS: mpls ip-ttl-propagate (2014). https://www.cisco.com/c/en/us/td/docs/routers/xr12000/software/xr12k_r4-1/mpls/command/reference/b_mpls_cr41xr12k/b_mpls_cr41xr12k_chapter_010.html#wp2864846713
7. Custura, A., Fairhurst, G., Learmonth, I.: Exploring usable Path MTU in the Internet. In: IFIP Network Traffic Measurement and Analysis Conference (2018)
8. Donnet, B., Luckie, M., Mérindol, P., Pansiot, J.-J.: Revealing MPLS Tunnels obscured from traceroute. SIGCOMM CCR **42**(2), 87–93 (2012)

9. Durumeric, Z., et al.: The matter of heartbleed. In: ACM IMC (2014)
10. Durumeric, Z., Wustrow, E., Halderman, J.A.: ZMap: fast internet-wide scanning and its security applications. In: USENIX Security (2013)
11. Edeline, K., Kühlewind, M., Trammell, B., Donnet, B.: copycat: Testing differential treatment of new transport protocols in the wild. In: Proceedings of the Applied Networking Research Workshop (ANRW) (2017)
12. Finn, G.G.: A connectionless congestion control algorithm. SIGCOMM CCR **19**(5), 12–31 (1989)
13. Floyd, S.: TCP and explicit congestion notification. SIGCOMM CCR **24**(5), 8–23 (1994)
14. Francois, P., Bonaventure, O.: Avoiding transient loops during the convergence of link-state routing protocols. IEEE/ACM Trans. Netw. **15**, 1280–1292 (2007)
15. Gill, S.: ICMP redirects are ba'ad, mkay? Technical report, Team Cymru Inc. (2002)
16. Gont, F.: ICMP Attacks Against TCP. RFC 5927, RFC Editor (2010)
17. Gont, F.: Deprecation of ICMP Source Quench Messages. RFC 6633, RFC Editor (2012)
18. Graham, R.: MASSCAN: Mass IP Port Scanner (2018). https://github.com/robertdavidgraham/masscan
19. Guo, H., Heidemann, J.: Detecting ICMP rate limiting in the internet. In: Beverly, R., Smaragdakis, G., Feldmann, A. (eds.) PAM 2018. LNCS, vol. 10771, pp. 3–17. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76481-8_1
20. Hengartner, U., Moon, S., Mortier, R., Diot, C.: Detection and analysis of routing loops in packet traces. In: ACM SIGCOMM Workshop on Internet Measurement (2002)
21. Hewlett Packard: HP-UX - Serviceguard A.11.19 on HP-UX 11.31: Source Quench Seen for Every IPMON Ping. https://support.hpe.com/hpsc/doc/public/display?docId=emr_na-c02190964
22. Rüth, J., Zimmermann, T., Hohlfeld, O.: ICMP Dataset and Tools (2018). https://icmp.netray.io
23. Johnson, D.: Finding all the elementary circuits of a directed graph. SIAM J. Comput. **4**(1), 77–84 (1975)
24. Juniper Networks, Inc.: no-propagate-ttl - TechLibrary - Juniper Networks (2017). https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/no-propagate-ttl-edit-protocols-mpls.html
25. Lone, Q., Luckie, M., Korczyński, M., van Eeten, M.: Using loops observed in traceroute to infer the ability to spoof. In: Kaafar, M.A., Uhlig, S., Amann, J. (eds.) PAM 2017. LNCS, vol. 10176, pp. 229–241. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54328-4_17
26. Malone, D., Luckie, M.: Analysis of ICMP quotations. In: Uhlig, S., Papagiannaki, K., Bonaventure, O. (eds.) PAM 2007. LNCS, vol. 4427, pp. 228–232. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71617-4_24
27. Nokia: Router Configuration Guide Release 15.0.R5. https://infoproducts.alcatel-lucent.com/cgi-bin/dbaccessfilename.cgi/3HE11976AAACTQZZA01_V1_7450%20ESS%207750%20SR%207950%20XRS%20and%20VSR%20Router%20Configuration%20Guide%20R15.0.R5.pdf
28. Postel, J.: Internet Control Message Protocol. RFC 792, RFC Editor (1981)
29. Reynolds, J., Postel, J.: Assigned Numbers. RFC 1700, RFC Editor (1994)
30. Rüth, J., Bormann, C., Hohlfeld, O.: Large-scale scanning of TCP's initial window. In: ACM IMC (2017)

31. Rüth, J., Poese, I., Dietzel, C., Hohlfeld, O.: A first look at QUIC in the wild. In: Beverly, R., Smaragdakis, G., Feldmann, A. (eds.) PAM 2018. LNCS, vol. 10771, pp. 255–268. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76481-8_19
32. Sridharan, A., Moon, S., Diot, C.: On the correlation between route dynamics and routing loops. In: ACM IMC (2003)
33. Varvello, M., Schomp, K., Naylor, D., Blackburn, J., Finamore, A., Papagiannaki, K.: Is the web HTTP/2 yet? In: Karagiannis, T., Dimitropoulos, X. (eds.) PAM 2016. LNCS, vol. 9631, pp. 218–232. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30505-9_17
34. Wang, F., Qiu, J., Gao, L., Wang, J.: On understanding transient interdomain routing failures (2009)
35. Xia, J., Gao, L., Fei, T.: Flooding attacks by exploiting persistent forwarding loops. In: ACM IMC (2005)
36. Xia, J., Gao, L., Fei, T.: A measurement study of persistent forwarding loops on the internet. Comput. Netw. **51**, 4780–4796 (2007)
37. Zimmermann, T., Rüth, J., Wolters, B., Hohlfeld, O.: How HTTP/2 pushes the web: an empirical study of HTTP/2 server push. In: IFIP Networking Conference (2017)