Global-Scale Anycast Network Management with Verfploeter

Wouter B. de Vries
University of Twente
Enschede, The Netherlands
w.b.devries@utwente.nl

Salmān Aljammāz
Cloudflare
London, United Kingdom
s@cloudflare.com

Roland van Rijswijk-Deij
University of Twente
Enschede, The Netherlands
r.m.vanrijswijk@utwente.nl

***Abstract*—Anycast has become a valuable tool for network operators. It plays a vital role in making the DNS root system globally highly available and resilient to stresses from e.g. DDoS attacks. Content delivery networks use it to direct clients to local caches, and to absorb attack traffic. Yet managing an anycast network is far from simple. Earlier work studying a DDoS attack on the DNS root system, for example, shows that even highly distributed anycast networks can be overwhelmed.**

To manage an anycast service, it is vital to know the catchment of points of presence (PoPs) of the service. In earlier work, we introduced "Verfploeter" a novel active measurement method to determine anycast catchments using ICMP messages. Unlike previously existing approaches, Verfploeter is unbiased, accurate and can be executed directly by the anycast operator without the need for external vantage points. We demonstrated the efficacy of Verfploeter on a testbed and small anycast service.

In this paper, we take the next step and deploy Verfploeter on one of the world's largest anycast networks, the Cloudflare CDN with 192 PoPs worldwide. We perform real-world case studies on network planning (what happens when PoPs are switched on or off), troubleshooting (reachability issues of an anycasted prefix) and security (detecting spoofed attack traffic). These case studies show that Verfploeter is highly suitable for such a large-scale operation and gives operators vital insights that allow them to improve network management practices of their anycast service.

***Index Terms*—Anycast, Routing, Measurements, Active, Monitoring, BGP, Security, Troubleshooting, Network Planning**

I. Introduction

Anycast is a technique, enabled by BGP, that allows physically and geographically distinct systems to be addressed with a single IP-address/IP-prefix. This allows services to be scaled horizontally at different locations. Service providers use IP anycast to provide increased resilience, lower latency, and increased throughput for their services.

Examples of services that use anycast are the DNS root servers and, e.g., Top Level Domain (TLD) DNS servers (e.g. `.com`). Historically it was assumed that anycast is only suitable for connectionless protocols, since each packet can potentially reach a different anycast instance. DNS, largely dependent on UDP, is therefore a suitable candidate for anycasting. It has since been shown that Internet routing is stable enough to allow anycast to work for both connection-less and connection-oriented protocols, such as TCP [1], [2]. Nowadays, many large Content Delivery Networks (CDNs) also utilize anycast, such as Microsoft/Bing, Verizon/Edgecast, Akamai, and Cloudflare.

978-1-7281-4973-8/20/$31.00 © 2020 IEEE

In earlier work [3] we introduced a novel methodology to measure the catchments (i.e. which client will be served by which site) of anycast services, called "Verfploeter". Key advantage of Verfploeter is that it does not require external Vantage Points (VPs) such as RIPE Atlas probes, but instead relies on ICMP-responsive Internet hosts. By sending ICMP Echo Requests to many hosts on the Internet, and collecting the responses, we can accurately establish the catchment of a service for the full IPv4 Internet, or a part thereof. Unlike an approach based on external vantage points, Verfploeter does not suffer from bias due to the distribution of these points.

To date, we showed how Verfploeter performs on a testbed, and, on a limited scale, the B root DNS server (which has just three anycast sites). In contrast, in this paper we describe a global-scale deployment in one of the world's largest anycast CDNs. We discuss the challenges of deploying Verfploeter in an anycast network of this scale (with 192 global points-of-presence). Then, we show how Verfploeter can help large-scale anycast operators manage their network through three use cases:

Firstly, we show how Verfploeter's detailed catchment information helps manage changes in the configuration of the active sites of an anycast service. For example, what would happen if large site A is taken down, in terms of the shift in clients to other sites. We argue that this is important since depending on the shift of traffic, one or more of the other sites might attract traffic exceeding its maximum capacity. This is also particularly useful for planned maintenance.

Secondly, we show how Verfploeter can be used to regain traditional ICMP-based troubleshooting capabilities. For example, traditionally connectivity issues are confirmed using `ping`, i.e. by sending an ICMP Echo Request packet. However, in the case of anycast, the response to this packet will likely end up in a different location. From the viewpoint of the sender of the request packet this would appear as a timeout. Using Verfploeter these packets are matched regardless of the location where it is received, in essence allowing an asymmetric ping.

Lastly, we show how Verfploeter can be used to detect spoofed traffic, by matching the known ingress location of traffic from a specific client using high resolution catchment data. Essentially we use Verfploeter to establish ground truth on client-to-anycast mappings, and mark traffic (supposedly)

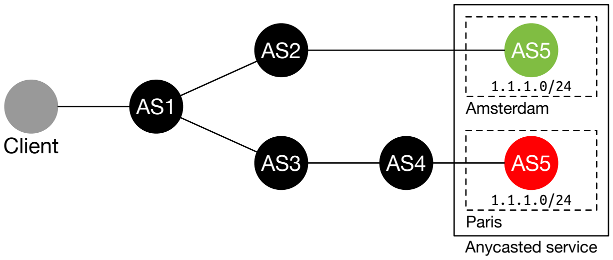


Fig. 1: Client connecting to 1.1.1.1, an anycasted service

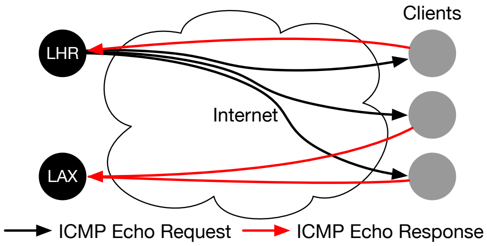


Fig. 2: Schematic overview of Verfploeter methodology

from that client that ends up in a different anycast site as potentially suspicious.

The rest of this paper is organized as follows. First, we provide the background in Section II. Then, in Section III we describe the operational implementation. In Section IV we discuss and analyse the three use cases. Then, in Section V, we describe related work and finally, in Section VI we conclude.

II. Background

A. Anycast

IP anycast is an addressing and routing strategy in which multiple physical servers in the Internet are configured with the same logical IP address. This strategy is widely used to achieve high availability and redundancy of services over the Internet, such as DNS and CDNs.

IP anycast takes advantage of the robustness of BGP that defines the catchment of each anycast instance by mapping users to the nearest instance, based on metrics that BGP takes into consideration. These metrics include, among others, the path length in terms of number of autonomous systems (ASes) that are crossed, and local preferences. Anycast catchments can be hard to predict mainly due to a large variety of routing policies that are applied within and between Autonomous Systems (ASes) [4], [5].

Examples of services that typically use anycast are the DNS (e.g. the Root DNS and many ccTLD operators), DDoS mitigation providers (e.g. Akamai, Cloudflare) and CDNs.

In Fig. 1 we show a simple routing graph containing a client connecting to an IP in the prefix 1.1.1.0/24. In this case, AS1 has two possible routes towards this destination prefix, one leading to a location in Amsterdam, and the other leading to a location in Paris. BGP Route selection determines which route will be selected as the best and, barring local preference settings, will pick Amsterdam as the closest, due to it having a shorter AS Path (AS2, AS5 (length 2), versus AS3, AS4, AS5 (length 3)).

Operating an anycast network can be challenging due to the diverse and unpredictable nature of Internet routes. This is exacerbated because of limited visibility into the policies and algorithms that underlie the decisions that are made in routers There are some projects, such as RIPE RIS and RouteViews, which attempt to collect routing tables from routers around the world, however, the scope of these is small (for example, RouteViews peers with 239 ASes[1], less than 0.5% of the total number of ASes[2]). Additionally, these routing tables only contain active routes, and not those routes that were not selected according to the BGP metrics.

B. Verfploeter

In previous work we developed a novel anycast measurement methodology named Verfploeter [3]. It uses pings, or ICMP Echo Requests, to map anycast catchments by sending requests *from* the anycast service *to* ping-responsive hosts on the Internet. The resulting responses, or ICMP Echo Replies, are then collected across the anycast service, see Fig. 2. This establishes the mapping between the source of the response, and the anycast site at which the response was received.

This method differs from more traditional methods to establish catchments. The most common method of assessing anycast is to use public or private measurement platforms, such as RIPE Atlas or PlanetLab. Such platforms offer Vantage Points (VPs) across the world that can be used to perform various measurements [6]. The downside is that the number of VPs is inherently limited, and difficult to increase. This inevitably leads to a bias in results because, e.g., RIPE Atlas probes are much more densely distributed in some regions (Europe) than others (Asia, South America).

In contrast, there are many ways to select viable ping-responsive hosts, with the operator deciding which to use depending on the use case. We describe three of them: **First**, selecting the full IPv4 address space is, considering the limited size, viable, the advantage is that the coverage of the measurement is maximized, at the cost of duration, network load, as well as storage. This method is also strictly nonviable for IPv6. **Secondly**, a selection of hosts can be made from previously collected traffic logs, this has the added benefit of automatically including the bias that the service's client base has in the measurement. **Lastly**, a tailored *hitlist* can be used, in particular, hitlists exist that attempt to select hosts that are most likely to respond to ICMP Echo Requests, such as those introduced in [7]. This particular IPv4 hitlist contains a single IP-address per /24 prefix, which is historically deemed most likely to respond. It is this last method that we use in this paper. The development of similar hitlists for IPv6 is an ongoing field of study [8]–[10].

[1] http://www.routeviews.org/peers/peering-status.html

[2] https://www.cidr-report.org/as2.0/

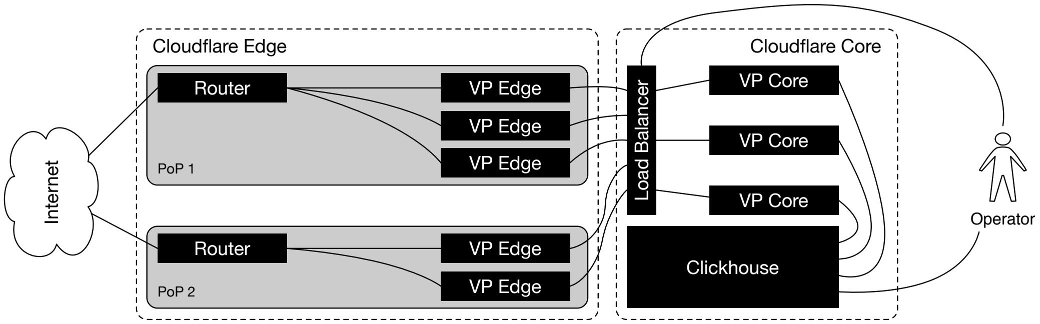


Fig. 3: Setup for Verfploeter at Cloudflare

III. Operational implementation

In previous work, in which we introduced Verfploeter [3], we discussed how to deploy Verfploeter on a limited scale (an anycast testbed and DNS B Root, which currently has three locations). In this paper our focus shifts to how we can deploy Verfploeter on a massive scale, on Cloudflare's anycast network. As of writing, Cloudflare has 192 Points-of-Presence worldwide, with over 30 Tbps of aggregate capacity [11]. It is used to deliver many services which includes Public DNS and Authoritative DNS, as well as a CDN and to provide DDoS mitigation. They announce more than 700 prefixes, covering roughly 1.5M IPv4 addresses [12].

In this section we discuss the challenges this creates and how we tackle them. In particular we discuss **(a)** the goal and requirements of the system, **(b)** the design choices we made when implementing, with regard to the requirements and goal and **(c)** the final architecture as it is currently in use.

Goal and requirements: Our goal for this implementation is to set up a measurement infrastructure that can take *snapshots* of the anycast catchment of a given prefix, on-demand as well as scheduled, for the entirety of the global deployment. The system should deliver the results in an accessible manner such that the results can be manually or automatically analyzed. To reach this goal, we defined the following requirements:

R1 **Service wide measurements should be trivial to start**: to maximize the utility of this system, it should be trivial for operators to start a measurement, without having to consider the large scale and distributed nature of the underlying service.

R2 **Incoming packets that are part of the measurement should be authenticated**: to prevent third parties from injecting results into the system without authorization, incoming packets should be authenticated.

R3 **Authenticated results should be inserted into a data store for analysis**: to allow both manual as well as automated analysis, data should be stored in a suitable data store.

R4 **Communication between system components should be reliable and secure**: given that the system will be running spread over many PoPs across the Internet, communication should be secure, and as reliable as possible to increase the accuracy of the results.

Design choices: Considering the goal and requirements we now discuss the major design choices we made:

Centralized control: in order to meet **R1**, it is impractical to have to contact each of the over 190 sites to start a measurement. We therefore chose a design where there is a central component (called *VP-core*) that manages the measurements, which the systems in the anycast sites connect to, to poll for jobs (this component is called *VP-edge*). Alternatively, the connection could be setup the other way around, but that would require the centralized component to have full knowledge over what systems are active on the edge sites, while that knowledge exists, it increases the complexity considerably.

Authentication: to prevent spoofed packets, and meet **R2**, we add a hash-based message authentication code (HMAC) in the ping payload. This ensures that received packets are in response to packets that were sent by the system, and not randomly generated by a third-party. Additionally, the payload contains the original target IP address, as well as the transmission timestamp. This information can then be used to filter out invalid responses, e.g. those that are late, duplicated or otherwise not authentic.

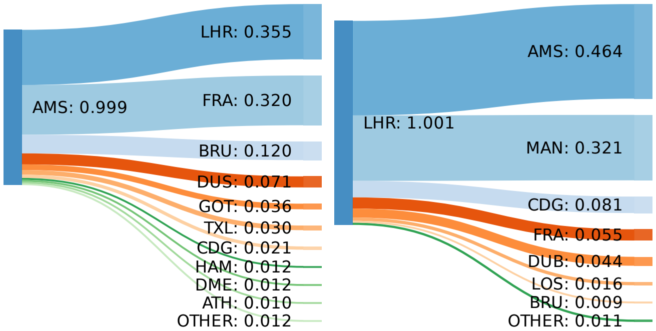
Robustness: To meet **R3**, each component buffers data such that connection or transmission failures do not result in loss of data. The core component is suitable to run in Kubernetes (K8s) [13], which automatically restarts processes upon failure, possibly on a different node of the K8s cluster, as well as facilitating horizontal scaling. In addition, data is buffered between the database and the core component by using a robust message bus, Apache Kafka [14].

Secure communication: Transport Layer Security (TLS) is used throughout the system for communication, to meet **R4**, in combination with gRPC [15]. Edge components are persistently connected to the core components.

Architecture: Verfploeter is a distributed system, written in Go, which is schematically depicted in Fig. 3. It consists of several components, the most important two of which are shown. The system must have, due to the nature of anycast, a component that runs on all nodes within the anycast service that potentially receive traffic on an anycast prefix. This component is referred to as *VP-edge*. The *VP-core* component on the other hand runs centralized, but is replicated horizontally for performance reasons.

TABLE I: Measurements with zero or more PoPs taken offline. PoPs are labeled with IATA airport codes: AMS = Amsterdam NL, LHR = London UK, CDG = Paris FR.

#	PoP(s) offline	Count	Response fraction
P0	*None*	3.49M	0.57
P1	AMS	3.44M	0.56
P2	LHR	3.30M	0.54
P3	CDG	3.42M	0.56
P4	AMS, LHR	3.45M	0.56
P5	AMS, CDG	3.50M	0.57
P6	one per measurement 182 measurements different PoP each measurement	≈3.5M	≈0.55



(a) AMS. Datasets: P0, P1.

(b) LHR. Datasets: P0, P2.

Fig. 4: Rerouting when a single PoP goes down.

The VP-edge component is responsible for sending and/or receiving the ICMP Echo Requests and Replies. In any given Point-of-Presence (PoP) there can be thousands of instances of this component. Control of it is exercised via the central VP-cores. Received responses are collected and transmitted, batch-wise, to the VP-cores. Authentication and validation of the incoming data occurs at the edge, such that only valid data is collected at the VP-core.

The VP-core component has four functions:

- Controlling the VP-edge outbound functionality.
- Collecting the results from the VP-edges and inserting them into an Apache Kafka messagebus, for later insertion into the central data warehouse application (Clickhouse).
- Providing an API for external users.

The VP-cli component (not shown) is used to initiate a measurement. It allows various command line options, e.g. to indicate the source address to use (e.g. the anycast prefix), which target addresses to use and which systems to initiate the measurement from. Results become available in the central data warehouse (based on Clickhouse [16]) when the measurement completes, or are forwarded to the command line client directly.

IV. Use cases

The system we discussed in Section III was implemented as a production service in Cloudflare's anycast network. To show how Verfploeter supports better management of anycast services, we now present three real-world use cases of how Verfploeter can be used, and demonstrate them on Cloudflare's network. These range from planning, to troubleshooting, as well as securing networks.

A. Planning

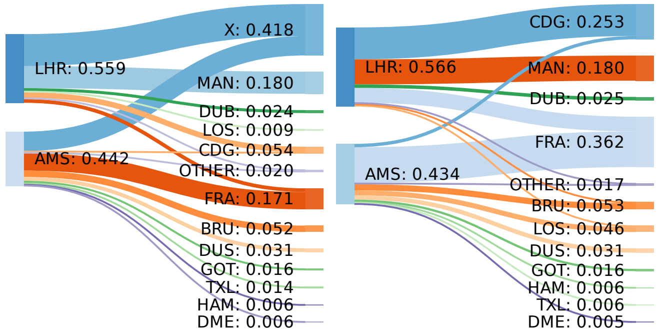
The primary reason to implement and use Verfploeter is for planning purposes. Particularly, what consequences, in terms of anycast catchments, does taking a particular PoP offline have. There are several reasons why a PoP might be taken offline, for example: for scheduled or unscheduled maintenance, or the PoP is overloaded and some traffic needs to be shifted to different PoPs.

Due to the nature of anycast and the opaqueness of the Internet it is hard to predict where traffic will be rerouted in case a prefix is withdrawn. This might cause difficulties, for example in case the traffic will be rerouted to a different PoP that might not be able to handle the added load.

For this section we have performed 188 measurements (for operational reasons not all PoPs were included), which we show in Table I. In each of these measurements a different PoP (and in two cases two PoPs) is taken offline. The number of responses varies between measurements, but typically lies around 3.5M, or 55% of the number of requests (≈6M), as shown in the *Count* and *Response fraction* columns. We use P0 as a baseline measurement to determine which IPs are associated with each of the PoPs, and use that to track where these IPs move to in the subsequent measurements. Note that by *taken offline* we refer to withdrawing the announcement of a test prefix, production traffic to the anycast network is unaffected by these measurements.

Consider the case where a single PoP is taken offline. By using two measurements, one in the *normal* state, and one in a state where that PoP is down, using a test prefix, we can show how prefixes will be rerouted. In Fig. 4a we show what happens if the PoP in Amsterdam (AMS) is taken down. We show what fraction of traffic is redistributed to which other PoP, note that on the left side the fractions do not always add up to precisely 1 due to rounding. Interestingly, the OTHER category includes 20 additional PoPs, each receiving a small number of rerouted prefixes. In the case of AMS the majority of traffic is redirected to LHR and FRA, which is expected. However, we would also expect CDG, another large PoP relatively close to AMS and LHR, to rank high, but the measurements show that this is not the case.

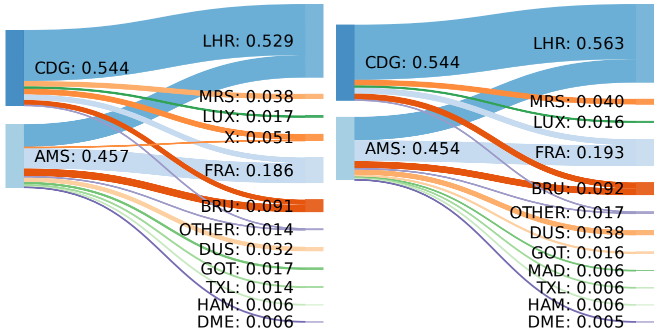
Complexity increases when taking multiple PoPs offline. For example, consider Fig. 4b, here, instead of AMS, London (LHR) is taken offline. We can see that in that case the majority of prefixes would reroute to AMS. However, what happens if AMS *and* LHR are taken offline. We show a measurement of this scenario in Fig. 5b. We point out that there is a strong *dependency* between LHR and AMS. The consequence of that is that determining what happens when both are taken offline based on the individual measurements we performed in Figs. 4a and 4b leads to a high degree of uncertainty of 42%, see X in Fig. 5a. X is the fraction of traffic that moves to AMS when LHR is taken offline, and vice versa.



(a) Calculated.
Datasets: P0, P1, P2.

(b) Measured.
Dataset: P0, P4.

Fig. 5: Rerouting with two PoPs down: AMS & LHR.



(a) Calculated.
Datasets: P0, P1, P3.

(b) Measured.
Datasets: P0, P5.

Fig. 6: Rerouting with two PoPs down: AMS & CDG.

In contrast, AMS and CDG (Paris) have a much lower dependency on each other. In Fig. 6 we show that the calculated (Fig. 6a) and the measured (Fig. 6b) route changes are very similar. In practice, this means that PoPs with a low *inter-dependency* do not require additional measurements to predict where traffic shifts if they are taken offline together, as this can be calculated from the individual measurements.

In Fig. 7 we show to how many other PoPs prefixes get redistributed in the case the PoP that they were associated to gets taken offline. Overall, most prefixes (>95%) that are associated with one PoP get redistributed to a relatively small number, 3 or fewer, other PoPs. This means that most combinations of PoPs that are taken offline can be calculated from *Single PoP offline* measurements. We do point out that there is a long tail, where for some PoPs the last 5% of prefixes get redistributed to 17 PoPs or more.

We emphasize two key takeaways from this: **a)** taking a PoP down causes prefixes to be redistributed across a limited number of PoPs, albeit with a long tail of small PoPs that take a small amount of prefixes. **b)** depending on the collection of PoPs, the redistribution can be calculated based on measurements where only a single PoP is taken down, with a known degree of certainty.

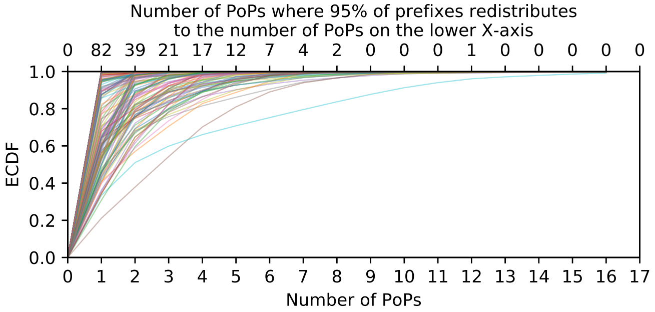


Fig. 7: ECDF for each measurement, showing to how many PoPs the measured prefixes are redistributed. Datasets: P0, P1, P2, P3, P6.

B. *Identifying connectivity issues*

One of the problems that arise when deploying an anycast service is that it is typically not possible to use industry standard tools such as `ping` and `traceroute`. The reason for this is that the response will often, depending on the size of the anycast network, be routed to a different system in a different physical location. This makes it hard to troubleshoot connectivity issues. In this section, we show how Verfploeter can be used to troubleshoot networks similarly to how a `ping` is normally used.

On the 1st of April 2018 Cloudflare launched its public DNS service. An interesting aspect of this launch was that it was launched on the *novelty* IP-ranges `1.1.1.0/24` and `1.0.0.0/24`. Obviously, this range has an interesting property in that it is particularly trivial to remember. However, this address range had not before been used for any production service, in fact, several manufacturers and ISPs appear to have assumed that the range is suitable for internal use. Contexts in which the range was used include captive portals and routers. Note that IANA has not designated the range `1.0.0.0/8` or any part thereof as reserved [17]. The range is therefore, according to the relevant authorities, suitable for public use. The range was obtained via a research agreement with APNIC [18].

The effect of the use of addresses in `1.0.0.0/8` in examples, captive portals, etc., is that especially the `1.1.1.1` address is not always reachable from edge networks. Typically, a network operator would use pings to test connectivity, which, as mentioned, will not trivially work on an anycast network. However, with Verfploeter we are again able to use ICMP to find connectivity problems. Specifically we used it to find issues with connectivity to `1.1.1.1`, compared with `1.0.0.1` and a third, unrelated IP address, `104.23.98.190` which has no known issues.

We perform measurements from each of the three source addresses, towards approximately 6 million target addresses. To increase the number of results we repeat the measurement once, and combine the results. In Table II we show the number of responses, as well as the response fraction, for each of the measurements.

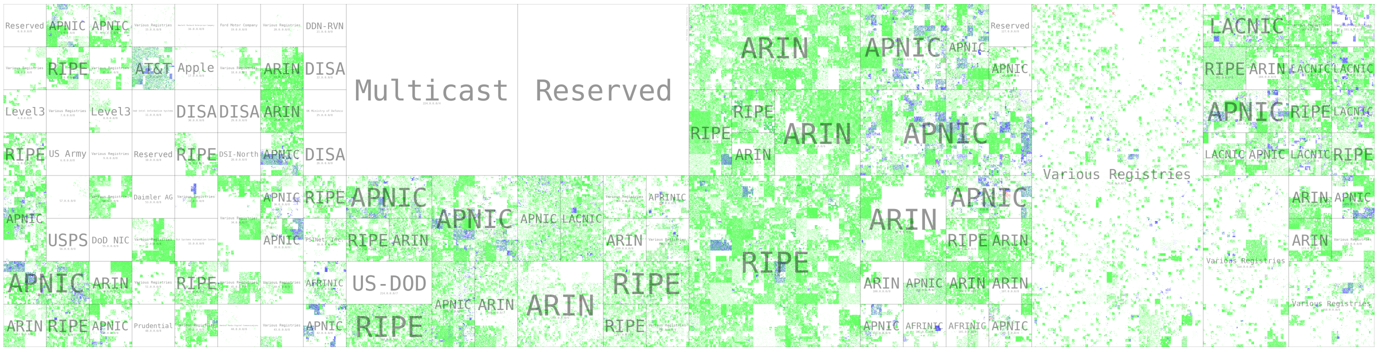


Fig. 8: Hilbert curve presenting connectivity issues, highlighting places where 1.0.0.1 has no issues, but 1.1.1.1 does in blue, vice versa in red, and where both have no issues in green.)

TABLE II: Measurements from three different source addresses, and combined totals.

Source	Count	Response fraction
1.0.0.1	3.47M	0.56
1.0.0.1	3.49M	0.57
1.1.1.1	3.28M	0.53
1.1.1.1	3.28M	0.53
104.23.98.190	3.48M	0.57
104.23.98.190	3.5M	0.57
	Combined	
1.0.0.1	3.58M	0.58
1.1.1.1	3.36M	0.55
104.23.98.190	3.59M	0.58

TABLE III: The different combinations of reachability and counts for each

IPs	Count	Fraction
1.0.0.1, 1.1.1.1, 104.23.98.190	3,324,062	0.917
1.0.0.1, 104.23.98.190	232,160	0.064
104.23.98.190	18,526	0.005
1.1.1.1, 104.23.98.190	17,508	0.005
1.0.0.1, 1.1.1.1	16,473	0.005
1.0.0.1	8,125	0.002
1.1.1.1	6,707	0.002

In Table III we show the different combinations of reachability that we encounter. By far the largest group of responders (almost 92%) do so to each of the three source IP addresses. These responders have no issues reaching `1.1.1.1`. The second category consists of responders that only respond for two out of three addresses, where the missing address is `1.1.1.1`. This category, while smaller than the first category is still quite substantial with approximately 6% of the addresses falling in this category. The remaining categories are much smaller, and we attribute them to random noise, e.g. hosts intermittently not responding or packet loss.

Zooming in on the second category, we investigate which Autonomous Systems (ASes) are involved in the unreachability of `1.1.1.1`. Table IV shows the top 5 ASes involved in `ping` failures. Interestingly China stands out with originating a large part of the failures.

TABLE IV: Top 5 ASNs showing prefixes unable to reach 1.1.1.1, but are able to reach 1.0.0.1 and 104.23.98.190

ASN	AS Name	Count
4837	China Unicom Backbone	70,649
4134	China Telecom Backbone	25,447
3352	Telefónica de España	11,049
7018	AT&T Services, Inc.	9,318
26615	TIM Celular S.A.	8,394

In Fig. 8 we show a Hilbert curve [19], a way of representing 1-dimensional information in a 2-dimensional image, while maintaining proximity. This curve shows where on the Internet, in terms of IP-space, connectivity issues to `1.1.1.1` and `1.0.0.1` occur. It appears that IP-space managed by APNIC, the Asia-Pacific Regional Internet Registry (RIR), has the highest number of issues, while space managed by RIPE and ARIN is far less problematic.

These results show, how, when operating an anycast network, our methodology can be applied to regain the functionality of a simple `ping` to troubleshoot a network. As a by-effect, we also show where on the Internet a major public DNS resolver, `1.1.1.1`, might still be inaccessible due to companies or organizations using ranges for purposes outside of what they were assigned.

C. *Securing against spoofed traffic*

Spoofed traffic, in which malicious actors falsify source IP addresses in packets, is an ongoing issue for operators. There are initiatives to counter IP spoofing, such as BCP38, which specifies that network operators should perform ingress filtering to block spoofed traffic. While anti-spoofing techniques are becoming more widespread, there is still a significant amount of spoofed traffic on the Internet [20], [21].

Three important reasons for bad actors to perform IP spoofing when attacking are: **a)** it is inherent to DDoS amplification attacks, where the IP is spoofed to match that of the target of the attack. **b)** to prevent identification of sources of traffic. **c)** to make it harder to filter out traffic.

In this section, we investigate the possibility to detect spoofed traffic by using comprehensive anycast mappings. The principle

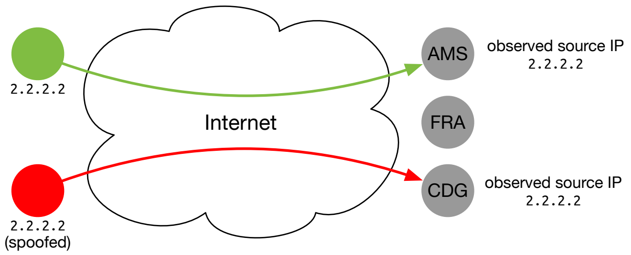


Fig. 9: Two traffic sources, using the same source IP, one spoofed and one legitimate. The observed IP at the server side is the same, the only difference is the location.

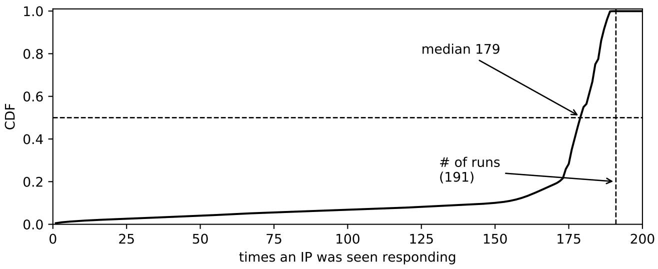


Fig. 10: Responses per IP

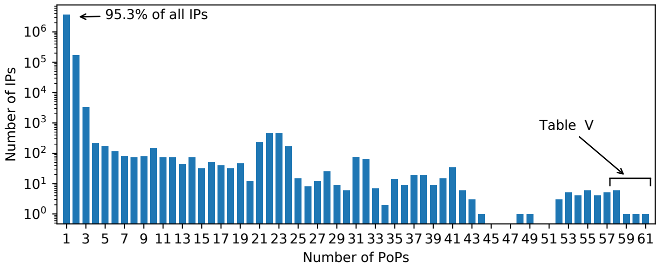


Fig. 11: Number of Points-of-Presence seen for a single IP-prefix.

TABLE V: IP-prefixes hitting the highest number of PoPs. AS8068 = Microsoft, AS26415 = ICANN, AS42 = PCH

Prefix	PoPs	ASN	Rev. Hostname
192.58.128.0/24	61	26415	j.root-servers.net
204.61.216.0/23	60	42	ns.anycast.woodynet.net
192.33.14.0/24	59	26415	b.gtld-servers.net
189.201.244.0/23	58	42	e.mx-ns.mx
204.19.119.0/24	58	42	c.ns.apple.com
200.108.148.0/24	58	42	c.dns.ar
206.51.254.0/24	58	42	lns61.nic.tr
13.107.4.0/24	58	8068	ns1.c-msedge.net
194.0.17.0/24	58	42	e.nic.ch

behind it is that because Internet routes are mostly stable, and traffic from most IP-ranges is routed to the same PoP every time, any traffic from those IP-ranges reaching a different PoP should be considered as *likely spoofed*, as depicted in Fig. 9.

We investigate two assumptions, namely that specific /24 IP-ranges are consistently routed to the same PoP, as well as that most /24 prefixes will route to the same PoP, regardless of the origin of the Verfploeter measurement. We suspect that the origin of the measurement can have an impact in the case that the target of the ICMP Echo Request, as part of the measurement, is itself an anycast service. We also speculate that there are reasons, such as the ingress point of traffic having some effect on the egress point for some particular networks.

We perform 191 Verfploeter measurements, towards the IPv4 hitlist as described in Section II, one for each active PoP, in the Cloudflare CDN. Due to the unreliable nature of ICMP, combined with the fact that the hosts in the IPv4 hitlist may or may not respond, we expect that we will not gather 191 responses for each IP in the hitlist. We show how many responses we gathered, per IP, in Fig. 10. We observe a median for *the number of times an IP was seen* of 179.

One of our primary interests is whether the IPs that we measure are associated with just a single PoP. Our assumption is that they are, and in Fig. 11 we can see that 95.3% of the IPs are in fact seen at only a single PoP. Interestingly this figure shows that there is a significant tail with some IPs appearing at as many as 61 PoPs. As we have said before, we assume that a likely reason for this is that those IPs are actually in an anycasted prefix themselves. To confirm this, we take a look at those IPs that show up in the most PoPs and show their corresponding AS, and their reverse hostname in Table V.

Interestingly, the top 9 IPs that hit the highest number of PoPs all belong to one of three organizations, all of which are known to manage large anycast networks, namely Microsoft, Verisign and Packet Clearing House (PCH). Two interesting examples are the J DNS root server (reverse hostname *j.root-servers.net*), which is hosted by Verisign, which is seen at 61 different PoPs, as well as one of the gTLD authoritative nameservers (reverse hostname *b.gtld-servers.net*), which is also hosted by Verisign. We interpret this as confirmation that anycasted services can indeed be revealed using this methodology.

Given that 95% of the IP addresses are only seen at a single PoP, it seems likely that also many Autonomous Systems, and all their addresses, are only seen at a single PoP. We investigate this by looking up the ASNs of each of the addresses, and confirm that out of a total of 60,295 ASNs that we see, from 52,533 ASNs all IPs are seen at a single PoP, but not necessarily the same one for each IP. 52,533 ASNs have IPs that are also only seen at a single PoP, and that PoP is also the same for all of the addresses. 2,541 only have addresses that are seen at multiple PoPs, while 3,069 contain addresses that are seen at both single as well as multiple PoPs.

These results show that most IP addresses that are seen on the Internet, and in fact most of the ASes, are only expected to show up at a single Point-of-Presence. Theoretically this should mean that any traffic that arrives at a PoP that it is is not supposed to, according to this mapping, can be considered as *likely to be spoofed*. This technique relies on a mismatch between the spoofed IP and the receiving PoP, if such a mismatch does not exist then the spoofing attempt will not be detected, intuitively, the probability of this happening decreases as the number of PoPs in the anycast network rises.

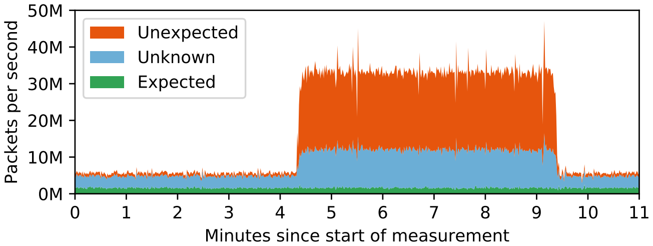


Fig. 12: Detecting spoofed traffic during a SYN-flood attack, according to the previously established client to PoP mapping

To demonstrate this technique in practice we recorded flow measurements during a known SYN-flood attack, on the same day the measurements for the client to PoP mapping were run. For each flow we checked if the source IP was supposed to arrive at the PoP it did, according to the mapping. The results of this are shown in Fig. 12. We can see that the *expected* traffic is constant over the duration (11 minutes). We see a steep increase in *unexpected* source addresses during the attack itself, indicating that the mapping provides a strong signal for detecting spoofed traffic. The *unknown* category indicates that even though our measurements cover many prefixes, there are still those for which we have no mapping. Compared to the traditional method of looking at TTLs to identify spoofed traffic [22], our method promises to provide a stronger signal, which we intend to further investigate in the future. Compared to more recent work [23], based on machine learning, this approach does not depend on up to date learning models to remain accurate. As future work we aim to further investigate the distribution of IPs contained in the *unexpected* and *unknown* categories, particularly outside the attack window.

The key takeaway from this section is that we have shown that Verfploeter can be applied to help operators defend against DDoS attacks that consist of spoofed traffic. Once identified this traffic can then be filtered or rate-limited.

V. Related Work

There is a large body of past work that used public measurement platforms, such as RIPE Atlas [24], PlanetLab [25], and the NLNOG RING [26], to study the operation of anycast services [2], [27]–[33]. Using public measurement platforms is a flexible way to study anycast services. A key characteristic of these platforms is that they provide some form of access, e.g. via SSH, an API, or a Web UI, to physical or virtual *probes*. Typically they allow a wide array of measurements to be performed. Compared to the Verfploeter method used in this paper, however, the use of public measurement platforms has two limitations. First, the number of vantage points from which measurements can be conducted is limited, and often many orders of magnitude less than what Verfploeter can achieve. Second, these measurements suffer from an inherent bias because of the placement of vantage points. This means they will always provide an incomplete view of actual anycast catchments.

Li et al [34] look at one of the root DNS servers for over a year, and find that site loads are often unbalanced, and that many clients are routed to a PoP that is suboptimal in terms of great-circle distance by several thousands of kilometers. They attribute this problem to inter-domain routing topology and policies, as well as poor route selection in the case that routes to multiple sites are available. These finds strengthen the case for a system that is able to perform fast and precise measurements of anycast catchments.

Past work has also focused specifically on anycast performance in CDNs. Calder et al. [2] study the anycast network for Microsoft's Bing search engine and show that while anycast generally performs well, up to 20% of clients get redirected to sub-optimal PoPs. They introduce a simple scheme to improve PoP assignment using other means than anycast (specifically DNS-based methods). Flavel et al. [1] introduce FastRoute, a hybrid approach that combines anycast and DNS-based load balancing to route clients to optimal PoPs. Where these past studies focused on anycast as just one means of distributing traffic, and augment this with other load balancing approaches, in this paper, we focus on a network that is managed purely as an anycast service, and show how Verfploeter can play a vital role in understanding and managing this network.

Finally, very recent work by McQuistin et al. [35] shows that anycast services that make use of multiple network operators can benefit from tailoring their BGP announcements to optimise end-user performance.

VI. Conclusions

In this paper we show how a global scale anycast network can be managed, using a methodology that we previously proposed in [3]. Compared to our previous work, we have validated our methodology on a much larger scale, on one of the world's largest anycast CDNs. We have also shown three use cases, that exemplify how this methodology can be applied.

We have described what the goals were for the implementation, how we chose to meet those goals and what the final operational implementation looks like. This implementation validates that Verfploeter can be deployed on any scale, from small dual node anycast services, such as shown in our initial work, up to the largest scale networks on the Internet.

Our three use cases show: **Firstly**, how this methodology can be applied by operators for planning purposes, and have shown that, depending on the case, the outcome of taking offline different combinations of anycast PoPs can be determined based on measurements in which only a single PoP was taken offline. **Secondly**, how it brings a traditional troubleshooting tool `ping`, in to the realm of anycast, and demonstrate how it was used to assess the reachability of a particularly interesting IP address. **Lastly**, we show how the methodology can be used to assist in the mitigation of DDoS attacks by providing a way to identify spoofed traffic based on known client to IP mappings.

Acknowledgments: This work was (partially) funded by Cloudflare, NWO grant #628.001.029, SURFnet Research on Networks and EU H2020 CONCORDIA (#830927).

References

[1] A. Flavel, P. Mani, D. A. Maltz, N. Holt, J. Liu, Y. Chen, and O. Surmachev, "FastRoute: A scalable load-aware anycast routing architecture for modern CDNs," in *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2015*. Oakland, CA, USA: USENIX Association, 2015, pp. 381–394.

[2] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye, "Analyzing the Performance of an Anycast CDN," in *Proceedings of the ACM Internet Measurement Conference*. Tokyo, Japan: ACM, Oct. 2015, pp. 531–537. [Online]. Available: http://conferences2.sigcomm.org/imc/2015/papers/p531.pdf

[3] W. B. de Vries, R. de O. Schmidt, W. Hardaker, J. Heidemann, P.-T. de Boer, and A. Pras, "Broad and Load-aware Anycast Mapping with Verfploeter," in *Proceedings of the 2017 Internet Measurement Conference*, ser. IMC '17. New York, NY, USA: ACM, 2017, pp. 477–488. [Online]. Available: http://doi.acm.org/10.1145/3131365.3131371

[4] R. Anwar, H. Niaz, D. Choffnes, I. Cunha, P. Gill, and E. Katz-Bassett, "Investigating interdomain routing policies in the wild," in *Proceedings of the 2015 ACM Conference on Internet Measurement Conference*, ser. IMC '15. New York, NY, USA: ACM, 2015, pp. 71–77. [Online]. Available: http://doi.acm.org/10.1145/2815675.2815712

[5] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford, "Dynamics of hot-potato routing in ip networks," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 307–319, Jun. 2004. [Online]. Available: http://doi.acm.org/10.1145/1012888.1005723

[6] RIPE NCC Staff, "RIPE Atlas: A Global Internet Measurement Network," *The Internet Protocol Journal*, vol. 18, no. 3, pp. 2–26, Sep. 2015.

[7] X. Fan and J. Heidemann, "Selecting representative IP addresses for Internet topology studies," in *Proceedings of the ACM Internet Measurement Conference*. Melbourne, Australia: ACM, Nov. 2010, pp. 411–423. [Online]. Available: http://www.isi.edu/~johnh/PAPERS/Fan10a.html

[8] O. Gasser, Q. Scheitle, S. Gebhard, and G. Carle, "Scanning the IPv6 Internet: Towards a Comprehensive Hitlist," in *Proc. 8th Int. Workshop on Traffic Monitoring and Analysis*, 2016.

[9] A. Murdock, F. Li, P. Bramsen, Z. Durumeric, and V. Paxson, "Target Generation for Internet-wide IPv6 Scanning," in *Proceedings of the 2017 Internet Measurement Conference*, ser. IMC '17. New York, NY, USA: ACM, 2017, pp. 242–253. [Online]. Available: http://doi.acm.org/10.1145/3131365.3131405

[10] O. Gasser, Q. Scheitle, P. Foremski, Q. Lone, M. Korczyński, S. D. Strowes, L. Hendriks, and G. Carle, "Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists," in *Proceedings of the Internet Measurement Conference 2018*, ser. IMC '18. New York, NY, USA: ACM, 2018, pp. 364–378. [Online]. Available: http://doi.acm.org/10.1145/3278532.3278564

[11] Cloudflare, https://www.cloudflare.com, accessed: 2019-11-22.

[12] Hurricane Electric, "AS13335," https://bgp.he.net/AS13335, accessed: 2019-11-22.

[13] K. Hightower, B. Burns, and J. Beda, *Kubernetes: Up and Running Dive into the Future of Infrastructure*, 1st ed. O'Reilly Media, Inc., 2017.

[14] Apache, "Kafka," https://kafka.apache.org/, accessed: 2019-11-22.

[15] gRPC, https://grpc.io, accessed: 2019-11-22.

[16] Yandex, "Clickhosue," https://clickhouse.yandex, accessed: 2020-01-13.

[17] IANA, "IANA IPv4 Special-Purpose Address Registry," https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml, accessed: 2020-01-13.

[18] Geoff Houston, "APNIC Labs enters into a Research Agreement with Cloudflare," https://labs.apnic.net/?p=1127, accessed: 2020-01-13.

[19] H. V. Jagadish, "Analysis of the hilbert curve for representing two-dimensional space," *Information Processing Letters*, vol. 62, no. 1, pp. 17–22, 1997.

[20] R. Beverly and S. Bauer, "The Spoofer project: Inferring the extent of source address filtering on the Internet," in *Usenix Sruti*, vol. 5, 2005, pp. 53–59.

[21] R. Beverly, R. Koga, and K. Claffy, "Initial longitudinal analysis of IP source spoofing capability on the Internet," *Internet Society*, p. 313, 2013.

[22] Q. Scheitle, O. Gasser, P. Emmerich, and G. Carle, "Carrier-Grade Anomaly Detection Using Time-to-Live Header Information," *CoRR*, vol. abs/1606.07613, 2016. [Online]. Available: http://arxiv.org/abs/1606.07613

[23] A. Degirmencioglu, H. T. Erdogan, M. A. Mizani, and O. Yılmaz, "A classification approach for adaptive mitigation of syn flood attacks: Preventing performance loss due to syn flood attacks," in *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2016, pp. 1109–1112.

[24] R. Staff, "Ripe atlas: A global internet measurement network," *Internet Protocol Journal*, vol. 18, no. 3, 2015.

[25] PlanetLab, https://www.planet-lab.org/.

[26] NLNOG Ring, http://ring.nlnog.net/.

[27] E. Aben, "DNS Root Server Transparency: K-Root, Anycast and More," Web, Jan. 2017, RIPE NCC. [Online]. Available: https://labs.ripe.net/Members/emileaben/dns-root-server-transparency

[28] R. Bellis, "Researching F-root anycast placement using RIPE Atlas," RIPE Labs: https://labs.ripe.net/Members/ray_bellis/researching-f-root-anycast-placement-using-ripe-atlas, Oct. 2015. [Online]. Available: https://labs.ripe.net/Members/ray_bellis/researching-f-root-anycast-placement-using-ripe-atlas

[29] D. Cicalese, J. Augé, D. Joumblatt, T. Friedman, and D. Rossi, "Characterizing IPv4 Anycast Adoption and Deployment," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2015, pp. 16:1–16:13.

[30] X. Fan, J. Heidemann, and R. Govindan, "Evaluating anycast in the Domain Name System," in *Proceedings of the IEEE Infocom*. IEEE, Apr. 2013, pp. 1681–1689. [Online]. Available: http://www.isi.edu/~johnh/PAPERS/Fan13a.html

[31] D. Madory, A. Popescu, and E. Zmijewski, "Accidentally Importing Censorship - The I-root instance in China," Presentation, Jun. 2010, Renesys Corporation. [Online]. Available: https://www.nanog.org/meetings/nanog49/presentations/Tuesday/Madory-I-root-lightning-talk.pdf

[32] G. C. M. Moura, R. de O. Schmidt, J. Heidemann, W. B. de Vries, M. Müller, L. Wei, and C. Hesselman, "Anycast vs. DDoS: Evaluating the November 2015 Root DNS Event," in *Proceedings of the ACM Internet Measurement Conference*, Nov. 2016, pp. 255–270. [Online]. Available: http://www.isi.edu/%7ejohnh/PAPERS/Moura16b.html

[33] R. d. O. Schmidt, J. Heidemann, and J. H. Kuipers, "Anycast latency: How many sites are enough?" in *Proceedings of the Passive and Active Measurement Workshop*. Sydney, Australia: Springer, Mar. 2017, pp. 188–200. [Online]. Available: http://www.isi.edu/%7ejohnh/PAPERS/Schmidt17a.html

[34] Z. Li, D. Levin, N. Spring, and B. Bhattacharjee, "Internet anycast: performance, problems, & potential." in *SIGCOMM*, 2018, pp. 59–73.

[35] S. McQuistin, S. Priyanka, and M. Flores, "Taming Anycast in a Wild Internet," in *Proceedings of the 19th ACM SIGCOMM Internet Measurement Conference (IMC 2019)*. ACM, 2019.