# Old but Gold: Prospecting TCP to Engineer and Real-time Monitor DNS Anycast (extended)

**ISI Technical Report ISI-TR-739b**
**Update April 2021**

Giovane C. M. Moura (1) John Heidemann (2) Wes Hardaker (2)
Pithayuth Charnsethikul (2) Jeroen Bulten (3)
Joao Ceron (1) Cristian Hesselman (1,4)
1: SIDN Labs, 2: USC/ISI, 3: SIDN, 4:University of Twente

## ABSTRACT

DNS[1] latency is a concern for many service operators: CDNs exist to reduce service latency to end-users, but must rely on global DNS for reachability and load-balancing. Today, DNS latency is monitored by active probing from distributed platforms like RIPE Atlas or commercial services. While Atlas coverage is wide, its 10k sites see only a fraction of the Internet. In this paper we show that passive observation of TCP handshakes can measure *live DNS latency, continuously, providing good coverage of current service clients*. Estimating RTT from TCP is an old idea, but applying this approach to DNS has never scrutinized like this before. We show that there is sufficient TCP DNS traffic today to provide good operational coverage (particularly of IPv6), and very good temporal coverage (better than existing approaches), enabling near-real time evaluation of DNS latency. We also show that DNS servers can optionally solicit TCP to broaden coverage. We quantify coverage and show that estimates of DNS latency from TCP is consistent with UDP latency. Our approach finds previously unknown, real problems: *DNS polarization* is a new problem where a hypergiant sends global traffic to one anycast site rather than taking advantage of the global anycast deployment. Correcting polarization in Google DNS cut its latency from 100 ms to 10 ms; correcting polarization from Microsoft cut Azure latency from 90 ms to 20 ms. Finally, *real-time* use of our approach for a European country-level domain has helped detect and correct a BGP routing misconfiguration that detoured European traffic to Australia. We incorporated our approach into *ENTRADA*, our open source data warehouse for DNS. We release our monitoring tool (`Anteater`), which has been operational for the last 2 years on this country-level top-level domain.

[1]This report is an updated version of the June 2020 report. In this version, we extend §2.2, by using far larger datasets to compare DNS/UDP and DNS/TCP round-trip times. This new version also includes Appendix G, which includes screenshots of Anteater, our monitoring tool. Moreover, it updates Anteater and releases it as freely. It also includes our changes to Knot DNS to solict TCP queries from clients to increase coverage, and it adds TCP RTT support to `dnsanon v1.12`. Finally, it clarifies the relationship between our work and prior work at `.cz` [28, 29]

## 1 INTRODUCTION

Latency is a key performance indicator for many DNS operators. DNS latency is seen as a bottleneck in web access [60]. Content Delivery Networks (CDNs) are particularly sensitive to DNS latency because, although DNS uses caching extensively to avoid latency, many CDNs use very short DNS cache lifetimes to give frequent opportunities for DNS-based load balancing and replica selection [15]. As a result of operator attention to DNS latency, low latency is a selling point for many commercial DNS operators, many of whom deploy extensive distributed systems with tens, hundreds, or more than 1000 sites [9].

DNS deployments often use *IP anycast* [30, 42] to reduce latency for clients. A *DNS service* is typically provided by two or more *authoritative DNS servers* [20], each defined on DNS on a separate IP addresses (in the NS record set [32]). With IP anycast, the IP address assigned to the authoritative DNS server is announced from many physically distributed *sites*, and BGP selects which clients go to which site. DNS clients often select the lowest-latency authoritative server when they have a choice [36, 38]. We will show later (§5) that improving anycast latency shifts traffic loads between servers.

DNS latency has been extensively studied [10, 35, 55]. Previous studies have looked at both absolute latency [55] and how closely it approaches speed-of-light optimal [25, 59]. A number of papers measure DNS latency from measurement systems with distributed vantage points such as RIPE Atlas [48], sometimes to optimize latency [8, 31]. Recent work has shown how to measure anycast catchments with active probes with Verfploeter [13, 14], and there is ongoing work to support RTT measurements. However, both RIPE and Verfploeter approaches to measure latency provide mixed coverage: large hardware-based measurements like RIPE Atlas only have about 11k active vantage points and cover only 8670 /24 IPv4 network prefixes [46] (May 2020), and commercial services have fewer than that. Verfploeter provides much better coverage, reaching millions of networks, but it depends on a response from its targets and so cannot cover networks with commonly deployed ICMP-blocking firewalls. It is also difficult to apply to IPv6 since it requires a target list, and effective IPv6 hitlists are an open research problem [16]. Finally, with the cost of active probing, Verfploeter is typically run daily and is too expensive to detect hourly changes (and RTT measurement support will require twice as much probing).

The main contribution of this paper to show that the handshake latency during *TCP connection setup is effective at estimating DNS client latency*. The TCP handshake has been used to estimate RTT at endpoints since 1996 [19], and it is widely used in passive analysis of HTTP (for example, [53]). This paper describes the technique's application to DNS – an idea shared with us by Casey Deccio. In a non peer-reviewed work performed previously but independently from our own, .cz operators [28, 29] also employed DNS/TCP RTT to evaluate latency from their services. While both use the same idea (derive latencies from the TCP handshake), ours provides a comprehensive validation and a series of carefully documented routing reconfiguraiton.We show measured latency from UDP and our estimates from TCP are similar (§2.2). We show that DNS servers can chose to solicit TCP from selected clients to increase coverage, if they desire, and we have implemented this option in the Knot authoritative DNS server.

Our second contribution is to show that *TCP-handshakes provide effective estimation* of DNS latency. Although DNS most often uses UDP, leaving DNS-over-TCP (shortened to DNS/TCP) to be often overlooked, we show that there is enough DNS/TCP traffic to support good coverage of latency estimation. We show that if we prospect through it we can find the latency "gold". Unlike prior approaches, passive analysis of TCP provides more coverage as busy clients send more queries, some with TCP. For .nl, the top 100 ASes are responsible for more than 75% of queries (§2.1.1). By scaling coverage with actual traffic, continuous passive RTT estimation can *increase temporal coverage* beyond current active approaches. For .nl, we cover 20k ASes every hour (§2.1.3). Finally, passive anlysis is the only approach that provides good coverage for IPv6 networks, overcoming the problem of active probing with stateless IPv6 addresses [41].

Our final contribution is to show that *TCP-based latency estimation matters*—it detects latency problems in operational networks, improving *latency engineering* in anycast (§4). We identify *DNS polarization* a problem that occurs when an Internet "hypergiant" [43] sends traffic over their own backbone to one anycast location rather than taking advantage of an existing global anycast serice. We show the importance of detecting and correcting this problem, reducing latency inflation by 150 ms for many clients of Google and Microsoft as they access .nl ccTLD and two commercial DNS providers.

Deployment also shows our tool matters. We have instrumented our open source *ENTRADA* [57, 64] with DNS/TCP RTT analysis. We provide a new tool, `Anteater`, that analyzes DNS/TCP RTT continuosly to detect errors and failures in real-time (we released it freely at [33]). These tools have been operational for more than two years at SIDN, the the Netherlands .nl ccTLD operator. That tool detected several problems in that time. In one case, some users experienced large increases in RTT due to traffic from Europe going to an anycast site in Australia (§4.4). Finally, we have just added RTT exaction to open-source `dnsanon v1.12` [1], and have been observing RTTs at B-Root since March 2021.

Our Knot changes are freely available [11]. Because our data is from public TLDs and therefore has privacy concerns, our DNS data is not available.

## 2 DNS/TCP FOR RTT?

While UDP is the preferred transport layer for DNS, TCP support has always been required to handle large replies [7]. TCP has also always been used for zone transfers between servers, and now increasing numbers of clients are using TCP in response to DNSSEC [2], response-rate limiting [61], and recently DNS privacy [21].

The RTT between a TCP client and server can be measured passively during the TCP session establishment [19, 31] or during the connection teardown [53]. For passive TCP observations to support evaluation of anycast networks for DNS, (a) enough clients must send DNS over TCP so they can serve as *vantage points* (VPs) to measure RTT, and (b) the RTT for queries sent over TCP and UDP should be the same.

We next verify these two requirements, determining how many clients can serve as VPs with data from three production authoritative servers (§2.1) – two from the .nl zone, and B-root, one of the Root DNS servers [52]. We then compare the RTT of more than 8k VPs with both TCP and UDP to confirm they are similar (§2.2), towards two large anycast networks: K and L-Root, two of the 13 anycast services for the Root DNS zone.

### 2.1 Does TCP provide Enough Coverage?

To assess whether DNS/TCP has enough coverage in production authoritative servers, we look at production traffic of two DNS zones: .nl and the DNS Root. For each zone we measure: (a) the number of resolvers using the service; (b) the number of ASes sending traffic; (c) the fraction of TCP queries the servers receive; (d) the percentage of resolvers using both UDP and TCP; and (e) the RTT of the TCP packets.

Our goal is to get a good estimate of RTT latency that covers every client's network. If every query were TCP, we could determine the latency of each query and get 100% coverage. However, most DNS queries are sent over UDP instead of TCP. We, therefore, look for *representation*—if we have a measured query over TCP, is its RTT the same as the RTTs other queries that use UDP, or that are from other nearby resolvers? If network conditions are relatively stable, the TCP query's RTT can represent the RTT for earlier or later UDP queries from the same resolver. It will likely represent the RTT for other resolvers in the same /24 as well. It is even possible it may represent the RTT for other resolvers in the same AS. Each assumption gives us greater coverage but increases the chances that the TCP RTT differs from the RTT of the other queries.

*2.1.1 `.nl` authoritative servers.* `.nl` currently (Oct. 2019) has 4 Authoritative DNS services, all configured with IP anycast. We next examine data from two of four authoritative services, called here Anycast Services A and B. The anycast services consist of 6 and 18 sites distributed globally. Each is run by a third-party DNS operator, one headquartered in Europe and the other in North America. They have no joint commercial relationship and we believe they have disjoint service infrastructure.

We analyze one week of traffic (2019-10-15 to -22) for each services using *ENTRADA*. That week from each service handles about 10.9 billion queries from about 200k resolvers and 50k Autonomous Systems (ASes), as can be seen in Table 1. This week of data shows that TCP is used rarely, less than 7% of queries each anycast service. However, those queries can represent more than a fifth of resolvers and 44% of ASes.

While these 44% of ASes is lower than we would prefer, *they account for a majority of traffic*—the top 100 ASes account for 78% and 75% of all queries for Services A and B (Figure 1). (The top 10 ASes are responsible for about half of all queries.) We cover the majority of traffic because passive coverage is best in networks with the most traffic, so it inherently focuses on networks that are most operationally important, with heavy hitters and the most traffic. In addition, we describe below how we plan to induce coverage, potentially getting data for all ASes that send traffic.

It is often appropriate for one TCP to represent its AS. For ASes where all recursive resolvers are co-located, latency to one is the same as to the others, so this assumption is appropriate. Table 2 shows coverage using this assumption: the relatively few number of TCP queries can represent *95–98% of all traffic* under this assumption. As such, an operator does not need to know DNS/TCP latency to all IPv4 and IPv6 address space; queries from TCP resolvers covers most of its clients.

*Root DNS.* To confirm that DNS/TCP provides coverage beyond `.nl`, we also look at how many TCP queries are seen at most Root DNS servers [52] over the same period.

Table 3 shows RSSAC-002 statistics [22, 63] from 11 of the 13 Root DNS services reporting at this time. As can be seen, the ratio of TCP traffic varied for each service (known as "letters", from A to M) and IPv4 or IPv6, overall ranging from 2.8 (A Root over IPv4) up 18.9% (J Root over IPv6). This data suggests the root letters see similar DNS/TCP rates as `.nl`.

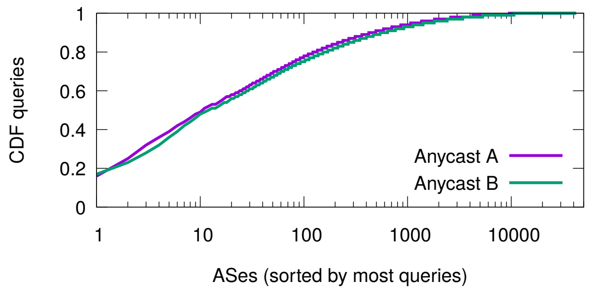


**Figure 1: `.nl`: queries distribution per AS.**

*2.1.2 Inducing Coverage.* While TCP coverage is not complete, we can get complete coverage by actively managing traffic to induce occasional TCP queries, as is often done in web systems (for example [54]). The DNS specification includes the TC bit to indicate a truncated reply that must be retried over TCP. DNS Receiver Rate Limiting [61] (RRL) uses this mechanism to force possible UDP-based address spoofers to resend their queries with TCP, allowing TCP cookies to source-address prevent spoofing.

A DNS server can use this same mechanism to solicit TCP queries from selected clients, allowing us to determine RTTs. We have implemented this capability in the Knot DNS server [12], building on Knot's existing RRL implementation. Our implementation tracks each block (/24 IPv4 prefix, or /56 IPv6 prefix). When a UDP request from that block arrives, if there are an insufficient number of TCP queries in the last hour, it returns an answer with the TC bit set with some probability. The probability of not setting the bit, and the required number of RTT observations per hour are both configurable.

*2.1.3 Temporal Coverage.* Next we investigate how much *temporal coverage* passive analysis of DNS/TCP provides. We require TCP connections to observe latency in each time period with confidence, so traffic rate per AS determines our temporal precision. We hope traffic allows temporal precision of 0.5 to 4 hours so passive analysis can support near-real-time monitoring over the day (§4.4).

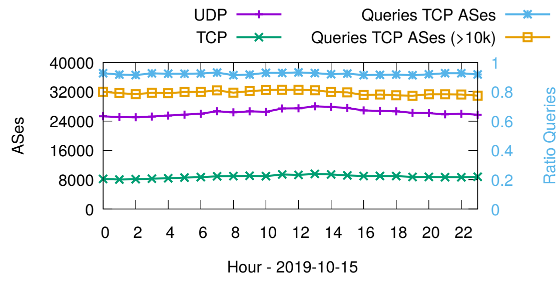
To evaluate the number of TCP queries per AS in a given time interval, we analyze `.nl` traffic from Anycast A and B. We single out one day of traffic (the first day of Table 1, 2019-10-15). On this day, Anycast A and Anycast B received UDP queries from ~37k ASes over IPv4, and from ~6.4k ASes over IPv6 (notice that numbers in Table 1 are higher given they take into account the full week).

| | Queries | | Resolvers | | ASes | |
|---|---|---|---|---|---|---|
| | **Anycast A** | **Anycast B** | **Anycast A** | **Anycast B** | **Anycast A** | **Anycast B** |
| Total | 5 237 454 456 | 5 679 361 857 | 2 015 915 | 2 005 855 | 42 253 | 42 181 |
| IPv4 | 4 005 046 701 | 4 245 504 907 | 1 815 519 | 1 806 863 | 41 957 | 41 891 |
| UDP | 3 813 642 861 | 4 128 517 823 | 1 812 741 | 1 804 405 | 41 947 | 41 882 |
| TCP | 191 403 840 | 116 987 084 | 392 434 | 364 050 | 18 784 | 18 252 |
| *ratio TCP* | 5.02% | 2.83% | 21.65% | 20.18% | 44.78% | 43.58% |
| IPv6 | 1 232 407 755 | 1 433 856 950 | 200 396 | 198 992 | 7 664 | 7 479 |
| UDP | 1 160 414 491 | 1 397 068 097 | 200 069 | 198 701 | 7 662 | 7 478 |
| TCP | 71 993 264 | 36 788 853 | 47 627 | 4 6190 | 3 391 | 3 354 |
| *ratio TCP* | 6.2% | 2.63% | 23.81% | 23.25% | 44.26% | 44.85% |

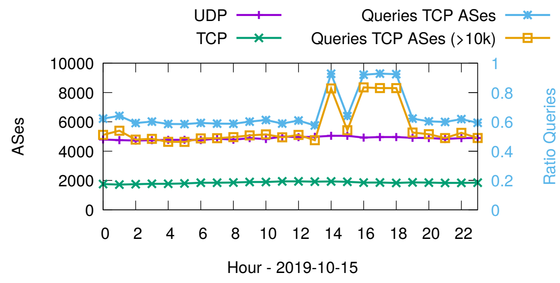
**Table 1: DNS usage for two authoritative services of `.nl` (Oct. 15–22, 2019).**

| | **Anycast A** | **Anycast B** |
|---|---|---|
| IPv4 | 4 005 046 701 | 4 245 504 907 |
| from TCP ASes | 3 926 025 752 | 4 036 328 314 |
| Ratio (%) | 98.02% | 95.07% |
| from TCP resolvers | 2 306 027 922 | 1 246 213 577 |
| Ratio (%) | 57.7% | 29.35% |
| IPv6 | 1 232 407 755 | 1 433 856 950 |
| from TCP ASes | 1 210 649 060 | 1 386 035 175 |
| Ratio (%) | 98.23% | 96.66% |
| from TCP resolvers | 533 519 527 | 518 144 495 |
| Ratio (%) | 43.29% | 36.13% |

**Table 2: Queries per Services for ASes and Resolvers that send TCP queries for `.nl` (Oct. 15–22, 2019).**



**(a) Anycast A: IPv4**



**(b) Anycast A: IPv6**

**Figure 2: `.nl` temporal coverage for Anycast A**

To evaluate how many ASes report enough data to estimate RTTs each hour, Figure 2 shows TCP queries per hour for Anycast A. As a baseline, IPv4 (Figure 2a) sees about 26.3k ASes that send UDP queries per hour (IPv4), and 4.8k for IPv6. Of these, about 8.8k also send TCP queries (1.8k for IPv6), allowing some IPv4 RTT information about 33% of ASes and for IPv6, 38% of ASes. However, these ASes that *also* send TCP queries are responsible for the majority of *all* queries (blue line in Figure 2): more than 90% of IPv4 queries, and more than 60% of all IPv6 queries. If we only consider ASes that send *at least* 10k TCP queries/hour, we still account for most of traffic (yellow line in Figure 2).

These fractions are consistent over the course of the day (as shown in this figure), with similar results at Anycast B (Appendix A). We conclude that a large number of ASes can be measured every hour with DNS/TPC. (We repeated the same analysis for 2019-10-21, and the same results hold – we include it in Appendix A).

**Summary:** We see that TCP data provide good operational coverage and great temporal coverage. More importantly, TCP provides the *only* insight into IPv6 latency, since current active methods do not generalize to IPv6.

## 2.2 DNS/UDP vs. DNS/TCP RTT

We expect round-trip-times measured with DNS/TCP and DNS/UDP to be similar. Next we investigate that assumption.

We can compare DNS/UDP and DNS/TCP RTTs by comparing *query response times* and accounting for TCP connection setup. DNS/UDP makes a direct request and gets a response, while in DNS/TCP we set up the TCP connection (with a SYN–SYN/ACK handshake), so a TCP DNS request should take two RTTs (assuming no connection reuse, TCP fast-open, or other optimizations). After accounting for TCP's handshake we should get similar RTT estimates.

| | A | B | C | D | F | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Total | 70601 | 40601 | 59033 | 88136 | 144635 | 31702 | 66582 | 115162 | 76761 | 105041 | 42702 |
| IPv4 | 58552 | 33925 | 47675 | 74565 | 125020 | 25706 | 55874 | 96727 | 61378 | 88046 | 33687 |
| UDP | 56921 | 32334 | 45568 | 70969 | 118738 | 25234 | 51208 | 87891 | 60312 | 84059 | 31925 |
| TCP | 1631 | 1591 | 2107 | 3596 | 6282 | 472 | 4665 | 8836 | 1065 | 3986 | 1762 |
| *Ratio (TCP)* | 2.87% | 4.92% | 4.62% | 5.07% | 5.29% | 1.87% | 9.11% | 10.05% | 1.77% | 4.74% | 5.52% |
| IPv6 | 12049 | 6675 | 11357 | 13571 | 19614 | 5995 | 1070 | 18435 | 15383 | 16994 | 9014 |
| UDP | 11659 | 6280 | 10966 | 13071 | 18919 | 5825 | 936 | 15511 | 15108 | 16576 | 8268 |
| TCP | 389 | 394 | 391 | 499 | 694 | 169 | 1342 | 2923 | 274 | 418 | 746 |
| *Ratio TCP* | 3.34% | 6.29% | 3.57% | 3.82% | 3.67% | 2.92% | 14.34% | 18.84% | 1.82% | 2.52% | 9.03% |

**Table 3: DNS queries (in millions) for Root DNS (E and G missing) – 2019-10-15 – 2019-10-22.**

| | K-Root | | L-Root | |
|---|---|---|---|---|
| | UDP | TCP | UDP | TCP |
| Date | Sept 4–5, 2020 | | Sept 5–6, 2020 | |
| Freq. | 4min | 8min | 4min | 8min |
| Probes | 10520 | 8676 | 10586 | 8989 |
| ∩ Probes | 8582 | | 8892 | |
| Queries | 3749892 | 1045605 | 3779763 | 1062557 |
| ∩ Queries | 3063836 | 1034233 | 3181098 | 1055888 |

**Table 4: DNS/UDP vs DNS/TCP Atlas measurements.**

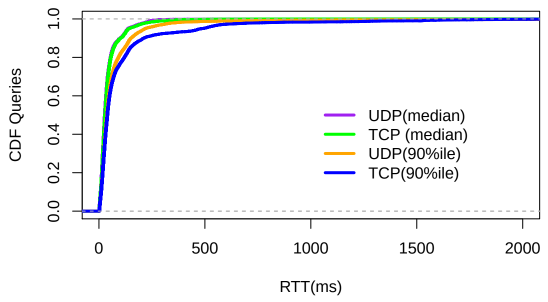


**Figure 3: `L-Root`: CDF of median and 90%ile RTT for DNS/UDP and DNS/TCP.**

To confirm this claim we measure DNS/UDP and DNS/TCP query response times using RIPE Atlas [47]. Atlas provides about 11k devices in different locations around the world, allowing us to test many network conditions. As *targets*, we evaluate two large, globally distributed, production and public DNS anycast networks: `L-Root`, with 167 anycast sites, and `K-Root`, with 79 sites, which are two of the thirteen authoritative servers for the Root DNS zone. To measure DNS/UDP latency from probes to these root letters, we leverage existing measurements that run continuously on Ripe Atlas, every 4 minutes (`[k-l]root-udp` in [45]).

We then created DNS/TCP measurements towards `L` and `K-Root` (`[k-l]root-tcp` in [45]), respecting our daily querying limits within the Atlas platform. We employed roughly 8.5k probes, which we ran for 24 hours, every 8 minutes (twice the interval of UDP measurements). Table 4 shows these datasets.

In these measurements, each Atlas queries directly the IPv4 address of the `K` and `L-Root`, bypassing their resolvers. For a fair comparision, we consider only probes that are present in both UDP and TCP measurements (∩ Probes): 8.5k and 8.9k for `K` and `L-Root`, respectively. In the same way, we only evaluate queries from these matching probes: ~3M UDP queries, and ~ 1M for TCP measurements, for each Root Letter (∩ Queries). In total, we analyze 8M queries for both letters. Then, for each Atlas probe, we compute its latency distribution.

Given Atlas' DNS/TCP response time includes two round-trip times (TCP hanshake plus query/answer time [49]), we divide the DNS/TCP latency values by two before generating, given we are interested in one RTT. Figure 3 shows the RTT CDF for both DNS/UDP and DNS/TCP for `L-Root`, from the 8.9k Atlas Probes. We see that the median latencies are essentially the same, for both UDP and TCP traffic. And roughly 70% of the 90 percentile RTT is also essentially the same for these 8k vantage points and `L-Root` (`K-Root` CDF shows similar results so we show it in Appendix E). This experiment proves that passively observed TCP RTTs can represent the RTTs that DNS/UDP will see.

## 3 PRIORITIZING ANALYSIS

We have shown that DNS/TCP can be mined to determine RTTs (§2). Operational DNS systems must serve the whole world, there are more than 42k active ASes sending DNS queries to authoritative servers. Both detection and resolution of networking problems in anycast systems is labor intensive: detection requires both identifying specific problems and their potential root causes. Problem resolution requires new site deployments or routing changes, both needing human-in-the-loop changes involving trouble tickets, new hardware, and new hosting contracts.

**Overview:** We use two strategies to *prioritize* analysis of problems that are most important: per-anycast site analysis and per client AS analysis, and rank each by median latency, interquartile range (IQR) of latency, and query volume.

Studying anycast sites focus on "our" side of the problem, highlighting locations in the anycast service we are responsible for that shows high latency toward sites, drawing our attention.

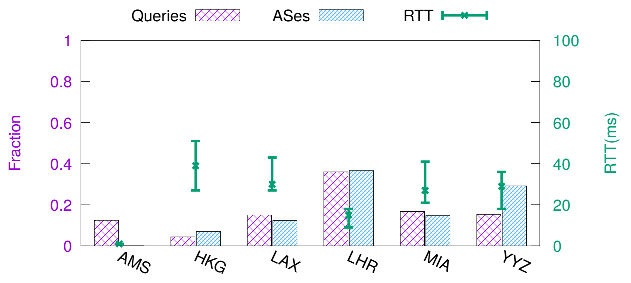
Clients ASes examine the *user* side of the problem (at recursive resolvers), since client latency is a goal in DNS service. While performance in client ASes can be difficult to improve because we do not have a direct relationship with those network operators, we show in §4 that we can address problems in some cases.

Finally, we consider median latency, interquartile range, and query volume to prioritize investigation. Median latency is a proxy for overall latency at the site. Interquartile range, the difference between 75%ile and 25%ile latencies, captures the *spread* of possible latencies at a given site or AS. Finally, query volume (or rate) identifies locations where improvements will affect more users. We sort by overall rate rather than the number of unique sources to prioritize large ASes that send many users through a few recursive resolvers (high rate, low number of recursive IPs).

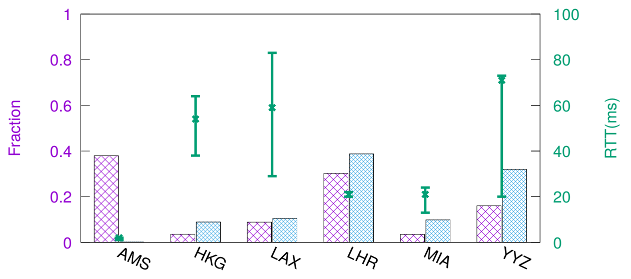
**Prioritization by Site:** Figure 4 shows per-site latency for `.nl`, broken out by protocol (IPv4 and IPv6) and by site, for two anycast services (A and B). For each site, we show two bars: the fraction of total queries and number of ASes (filled and hatched bar in each cluster). We overlay both with whiskers for latency (with median in the middle and 25%ile and 75%ile at whisker ends). In these graphs some sites (such as CDG for Anycast B in IPv6) stand out with high interquartile ranges, while others with lower interquartile range (for Anycast B, LAX-A and NRT in IPv4 and NRT and GRU in IPv6).We look at these cases in detail in §4.

We omit graphs for `B-root` by site due for anonymization.

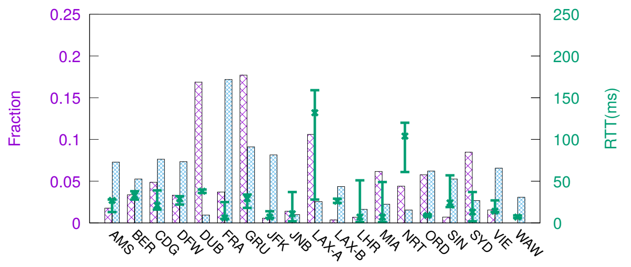
**Prioritization by Client AS:** Figure 5 and Figure 18 (Appendix C) show the distribution of latency for the top-ten ASes with largest query volume for Anycast A and B of `.nl`. (Due to space constrains, we show the complete list of AS names in Appendix B). While many ASes show good latency (low median and small interquartile range), we see the top two busiest ASes for Anycast A in IPv4 (Figure 5a) show a high median and large interquartile range (Figure 5b). These ASes experience anycast polarization, a problem we describe in §4.3.



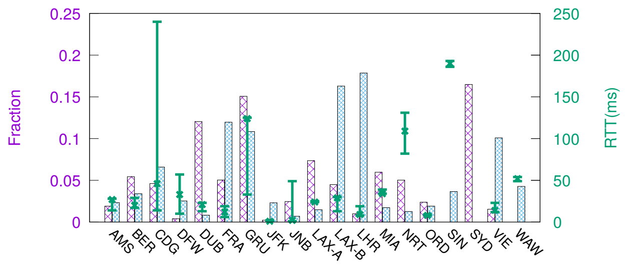
(a) Anycast A: IPv4



(b) Anycast A: IPv6

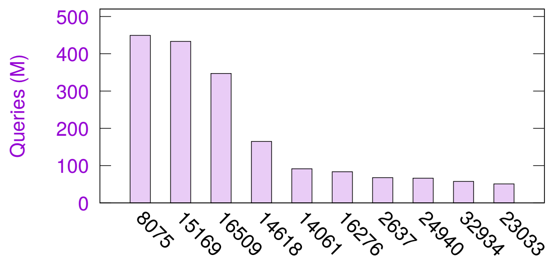


(c) Anycast B: IPv4

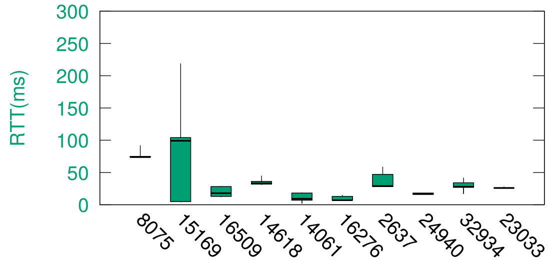


(d) Anycast B: IPv6

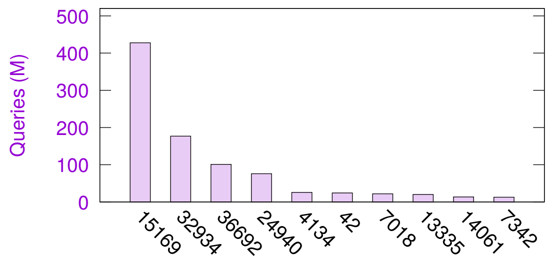
**Figure 4: `.nl` distribution of queries and ASes per site (pink bars) and latency (median, 25%ile, and 75%ile, (green lines), for each anycast site, for two services (Anycast A and B) and two protocols (IPv4 and IPv6). Data from 2020-10-15 to -22.**



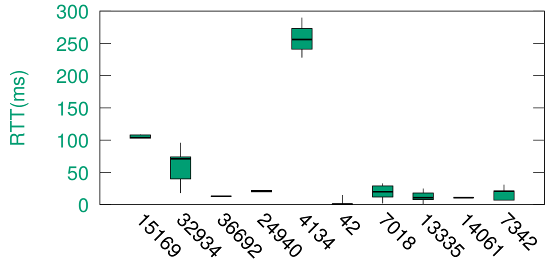
**(a) Anycast A – IPv4 – Queries**



**(b) Anycast A – IPv4 – RTT**

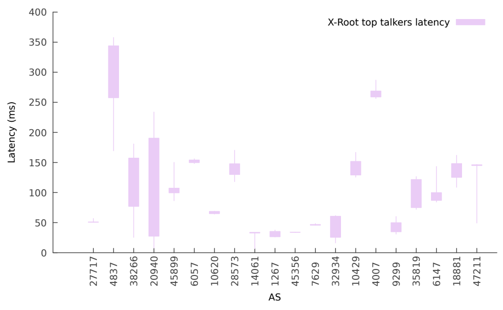


**(c) Anycast A – IPv6 – Queries**

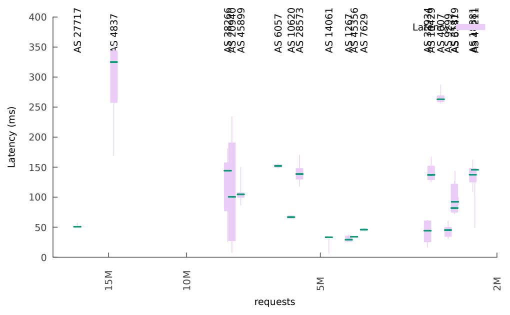


**(d) Anycast A – IPv6 – RTT**

**Figure 5: `.nl` Anycast A queries and RTT for the top 10 ASes ranked by most queries (bars left axis). Data; 2019-10-15 to -22. ASes list on Appendix B).**



**(a) `B-root`: latency of top talkers by rank**



**(b) `B-root` latency of top talkers by data size (log scale)**

**Figure 6: `B-root` latency by volume – AS list in Table 8.**

Figure 6 shows latencies for the top ASes for `B-root`. Here we show quartile ranges as boxes, and with the 10%ile and 90%ile values as whiskers. Rather than split by protocol, here we show both rankings (Figure 6a) and by query rate (Figure 6b) on the $x$-axis. While rank gives a strict priority, showing ASes by rate helps evaluate how important it is to look at more ASes (if the next AS to consider is much, much lower rate, addressing problems there will not make as large a difference to users).

We identify specific problems from these graphs next.

## 4 PROBLEMS AND SOLUTIONS

Given new information about both IPv4 and IPv6 latency from DNS/TCP (§2), and priorities (§3), we next examine anycast performance for two of the four anycast services operating for `.nl`, and for `B-root`. For each problem we describe how we found it, the root causes, and, when possible, solutions and outcomes.

### 4.1 Distant Lands

The first problem we describe is *distant lands:* when a country has no anycast server locally and has limited connectivity to the rest of the world. When trans-Pacific traffic was metered, these problems occurred for Australia and New Zealand. Today we see this problem with China. China has a huge

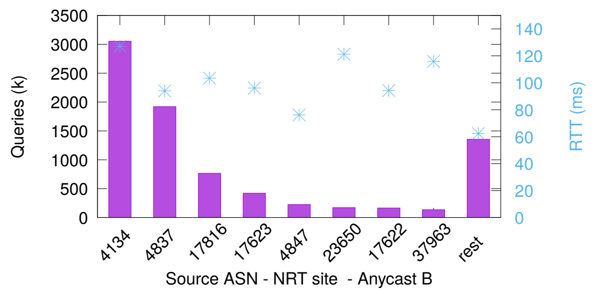


**Figure 7: Anycast B, Japan site (NRT): Top 8 querying ASes are Chinese, and responsible for 80% of queries.**

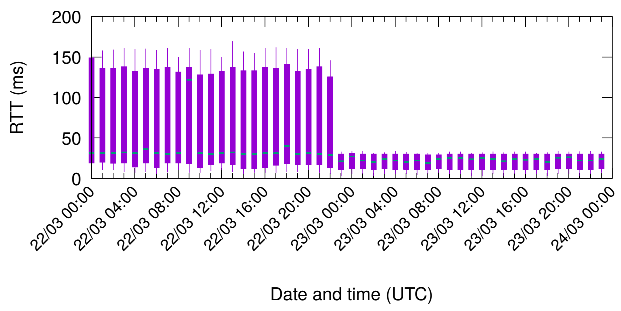


**Figure 8: Anycast B and Comcast: RTT before and after resolving IPv6 misconfiguration.**

population of Internet users, but its international network connections can exhibit congestion [65].

**Detection:** We discovered this problem by observing large interquartile latency for `.nl`'s Anycast B in v4 (Figure 4c) and v6 (Figure 4d) at Tokyo (NRT, both v4 and v6), Singapore (SIN, v6), and CDG (v6), all with 75%iles over 100 ms.

These wide ranges of latency prompted us to examine which recursive resolvers visiting these sites and showed high latency. Many queries come from ASes in Asia (Figure 7). NRT sees many queries (6.1% of total, more than it's "fair share" of 5.2%). Of the top 10 ASes sending queries to NRT, 9 are from China (see Figure 7).

We see a number of Chinese ISPs also send IPv6 traffic to Paris (CDG), resulting in it's wide spread of RTTs. Not only must traverse congested international links, but they then travel to a geographically distant anycast site, raising the 75%ile RTT at CDG over 100 ms (even though its median is under 22 ms).

**Resolution:** While we can diagnose this problem, the best resolution would be new anycast servers for Anycast B inside China. The operator is considering in having a premium transit to Chinese ISPs, as many of the operator's clients may not be confortable with co-location inside China (and only recently have foreign providers been allowed to operate locally there [65]).

## 4.2 Prefer-Customer to Another Continent

The second root-cause problem we found is when one AS prefers a distant anycast site, often on another continent, because that site is a customer of the AS. (Recall that a common BGP routing policy is *prefer customer:* if an AS can satisfy a route through one of its customers, it prefers that choice over an alternate route through a peer or transit provider. Presumably the customer is paying the AS for service, while sending the traffic to a peer or via transit is either cost-neutral or incurs additional cost.)

We have seen this problem in two situations, at `.nl` Anycast B's Brazil site, and with `B-root` for its site in South America.

**`.nl` Detection:** We detected this problem for `.nl` Service B by observing high IPv6 median latency (124 ms) for queries is in São Paulo, Brazil (GRU) in Figure 4d. Examination of the data shows many of the high-latency queries are from Comcast (AS7922), a large U.S.-based ISP. As with China and CDG, this case is an example of queries traveling out of the way to a distant anycast site, ignoring several anycast sites already in North America. We confirmed that North American clients of this AS were routing to the Brazil site by checking CHAOS TXT queries [4] from RIPE Atlas probes to Anycast B (data: ComcastV6 [45]).

**`.nl` Resolution:** We contacted `.nl` Anycast B's operator who identified that the issue was that one of their upstream providers. This provider had deployed BGP communities to limit the IPv4 route to South America. After our contact, they deployed the same community for IPv6 and the Comcast traffic remained in the US.

We first confirm the problem was resolved by analyzing traces from Anycast B, and by confirming that Comcast IPv6 clients were now answered by other North American sites. The solution reduced 75%ile latency by 100 ms: in Figure 8 before the change, IPv6 shows IQR of 120 ms for Anycast B. After this change on 2020-03-23t00:00, we see the IQR falls to 20 ms. Second, we also verified with Atlas probes hosted on Comcast's network (data: ComcastV6-afterReport in [45]), and the median RTT from Comcast Atlas was reduced from 139 ms to 28 ms.

**`B-root` Detection:** `B-root` has observed high latencies for traffic going to a South-American anycast site of `B-root`. As with `.nl` and GRU, we examined traffic and identified a primarily-North American ISP that was sending all of its traffic to the South American site, ignoring all other lower-latency sites. We then confirmed that an AS purchases transit from this ISP.

| | Queries | Queries Top Site | (% top site) |
|---|---|---|---|
| Google | 860 775 677 | 860 774 158 | 99.9998 |
| IPv4 | 433 145 168 | 433 145 119 | 99.9999 |
| IPv6 | 427 630 509 | 427 629 039 | 99.9997 |
| Microsoft | 449 460 715 | 449 455 487 | 99.9988 |
| IPv4 | 449 439 957 | 449 434 729 | 99.9988 |
| IPv6 | 20 758 | 20 758 | 100 |

**Table 5: Anycast A: Polarized ASes and query distribution (Oct 15-22,2019).**

**B-root Resolution:** We do not yet have a completely satisfactory resolution to this problem. Unfortunately the AS that purchases transit from the North American ISP does not directly peer with B-root, so we cannot control its peering. We currently poison the route to prevent latency problems, but that greatly reduces traffic arriving at this site.

## 4.3 Anycast Polarization with Google and Microsoft

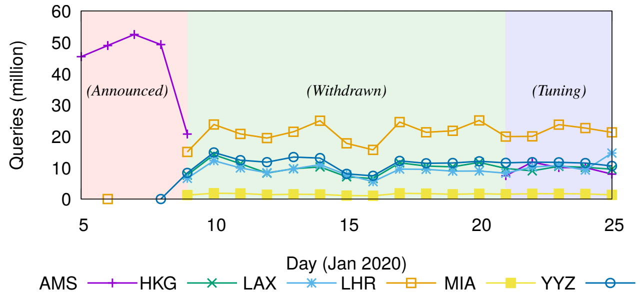
We next describe *anycast polarization*, a problem we believe has not been previously described. Like prefer-customer, it involves high latency that derives from traffic being needlessly sent to another continent. But it follows from BGP's limited knowledge of latency (AS path length is its only distance metric) and the flattening of the Internet [24].

*4.3.1 Detecting the Problem.* We discovered this problem by examining DNS/TCP-derived latency from the top two ASes sending queries to .nl Anycast A. As seen in Figure 5b and Figure 5d, AS8075 (Microsoft) and AS15169 (Google) show very high IPv4 median latency (74 ms and 99 ms), and Google shows a very high IQR (99 ms) Google also shows a high IPv6 median latency (104 ms).

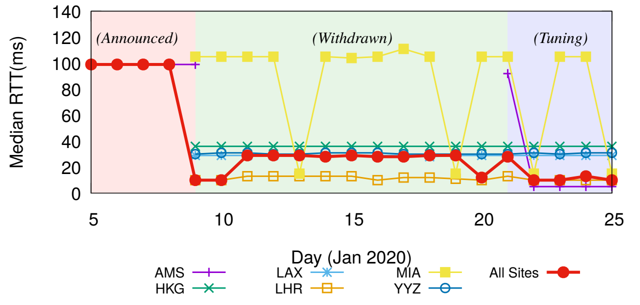
Both Google and Microsoft are "hypergiants", with datacenters on multiple continents (for .nl, ~85% of Google's traffic is from its Public Resolvers [34, 58]). Both also operate their own international backbones and peer with the Internet in dozens of locations. These very high latencies suggest much of their DNS traffic is traveling between continents and not taking advantage of .nl's global anycast infrastructure.

*Confirming the problem:* .nl Anycast A has six sites, so we first examine how many queries go to each site. Table 5 show the results—all or very nearly all (four or five "nines") go to the a single anycast site due to routing preferences. For Google, this site is in Amsterdam, and for Microsoft, Miami.

While a preferred site is not a problem for a small ISP in one location, it is the root cause of very high latency for these hypergiants. They are routing their global traffic over their own backbones to one physical location. Even if it is the best destination for some of their traffic, one location



**(a) IPv4 - Queries**



**(b) IPv4 - RTT (ms)**

**Figure 9: .nl Anycast A: queries and median RTT per site from Google (AS15169) – January 2020.**

can never be the lowest latency for all globally distributed datacenters. They defeat any advantages anycast has for reducing latency [38, 55].

*4.3.2 Depolarizing Google to* .nl *Anycast A. Root-cause:* We first investigated Google's preference for AMS. .nl directly operates the AMS site (the other 5 sites are operated by a North American DNS provider). We determined (working with both the AMS and Google operators) that Google has a direct BGP peering with the site at AMS. BGP prefers routes with the shortest AS-PATH, and in addition, ASes often prefer Private Network Interconnect (PNIs) over equal length paths through IXPs, so it is not surprising it prefers this path. (The general problem of BGP policy interfering with lowest latency is well documented [5, 6, 8, 25, 31, 55]. We believe we are the first to document this problem with hypergiants and anycast through PNI.)

We next describe how we worked with the AMS operators and Google to resolve this problem. We document this case as one typical resolution to show the need for continuous observation of DNS latency through DNS/TCP not find the problem and confirm the fix.

| Op. | Day | Time | Prepend | Community | AMS(%) |
|---|---|---|---|---|---|
| 1 | 21 | 15:00 | 2x | – | >0 |
| 2 | 22 | 9:53 | 2x | NE | >0 |
| 3 | 22 | 9:59 | 1x | – | 100 |
| 4 | 22 | 10:21 | 1x | NE | 100 |
| 5 | 22 | 10:37 | 1x | NE,15169:13000 | 100 |
| 6 | 22 | 11:00 | 2x | NE | >0 |

**Table 6: BGP manipulations on AMS site of Anycast A – IPv4 and IPv6 prefixes to Google (AS15169) on Jan 21, 2020 (Time in UTC). NE: No Export**

Figure 9 show the effects of our traffic engineering on anycast use and query latency for IPv4 (IPv6 figures in Appendix F). Each graph shows traffic or median client latency for each of the 6 `.nl` Anycast A sites. (Query latency is determined by DNS/TCP traffic over each day.) The graphs show behavior over January 2020, January 5th to 9th (the left, pink area) before any changes, the 9th to the 21st (the middle, green area) when the AMS route was withdrawn, and finally after the 21st (the right, blue region) when AMS was restored, but with different kinds of policy routing.

These graphs confirm that AMS received all traffic from Google initially, causing Anycast A to be experienced by Google as an *unicast* service, defeating the whole purpose of anycast. We see that the median latency for Google about 100 ms, a large value made worse given Google sends most queries to this service (Figure 5a). Withdrawing the AMS peering with Google corrects the problem with queries now sent to most sites, and, as such, enabling Google to take advantage of the anycast configuration of the service. We see median latency dropping to 10 to 40 ms, although still around 100 ms at YYZ in Toronto, Canada, for IPv4. LHR is now the busiest site, and although it is in Europe (although not in the European Union),

Use of the North American sites greatly lowers median latency. We show in Figure 10 the depolarization results for all sites combined, for IPv4 and IPv6. For both IPv4 and IPv6, we see median latency for all sites combined reducing 90 ms, from 100 to 10,ms. The IQR was reduced from 95 to 10 ms for IPv4. For IPv6, we observed few queries over TCP between Jan. 1 and 9, so they are not representative. After depolarizing, we see more queries over TCP.

Although overall latency improves, omitting the AMS site misses the opportunity to provide better latency to their datacenters in the Netherlands and Denmark. We therefore resumed peering over the BGP session, experimenting with several policy routing choices shown in Table 6. We experimented with 1x and 2x AS-PATH prepending, no-export, and a Google-specific "try-not-to-use this path" community string [17]. We found that no-export and the community string had no effect, perhaps because of the BGP session, and

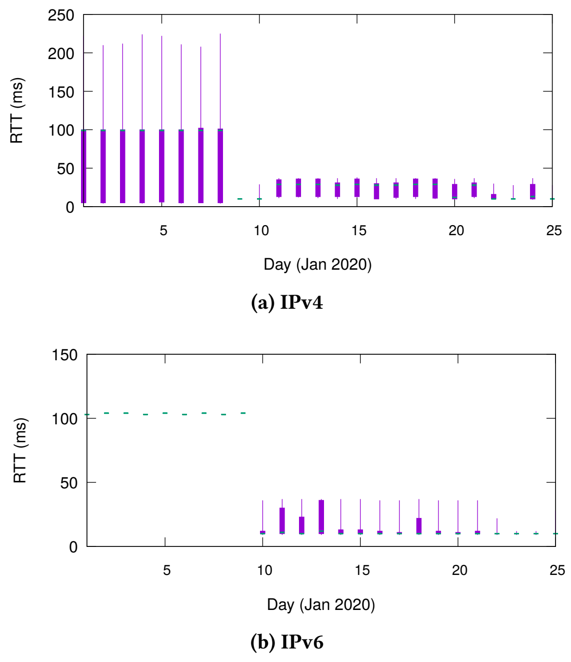


**Figure 10: Google depolarization results and RTT.**

neither did single prepending. However double AS-PATH prepending left AMS with about 10% of the total traffic load. Full details of our experiments are in an appendix (§H.1).

*4.3.3 Depolarizing Microsoft to `.nl` Anycast A.* **Detection:** We discovered Microsoft anycast polarization through analysis of DNS/TCP across ASes ( Figure 5b and Figure 5d) AS8075 (Microsoft) and AS15169 (Google) Microsoft's preferred site for `.nl` Anycast A is Miami (MIA), a different preference than Google's, but the outcome was the same: huge latency (median 80 ms) because global traffic goes to one place.

**Resolution:** Again, we worked with the operators at `.nl` Anycast A MIA and Microsoft to diagnose and resolve the problem. We confirm that Anycast had a peering session with Microsoft in MIA, and not at any other sites. Again, the result was a short AS-PATH and a preference for all Microsoft datacenters to use the Microsoft WAN to this site rather than other `.nl` Anycast A anycast sites (having different upstream providers per anycast site may cause such traffic distributions [31]).

Options that could mitigate this polarization include depeering with Microsoft in MIA, peering with Microsoft at the remaining sites, or possibly BGP-based traffic engineering. Because our ability to experiment with BGP was more limited at this site, and we could not start new peerings at other sites, the operator at MIA de-peered with Microsoft at our recommendation.

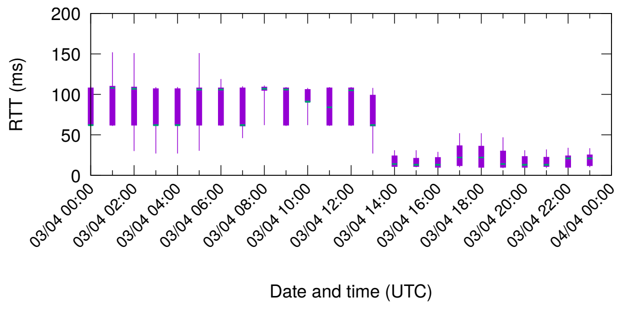


**Figure 11: .nl Anycast A and Microsoft (IPv4): RTT before and after depolarization.**

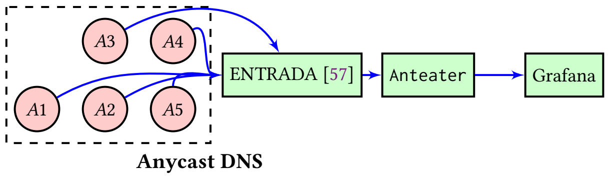


**Figure 12: DNS/TCP RTT near real-time monitoring at `.nl`**

Figure 11 shows latency for this AS before and after our solution. Removing the direct peering addressed the problem, and Microsoft traffic is now distributed across all `.nl` Anycast A sites. As a result, the IQR falls from about 80 ms to 13 ms. The median latency also falls by 70 ms, from 90 ms to 20 ms. Our technique identifies problems with polarization, and shows the dramatic improvement that results.

## 4.4 Detecting BGP Misconfiguration in Near Real-Time

Because it poses no additional cost on the network, passive measurement of anycast latency with DNS/TCP is an ideal method for *continuous, on-the-fly* detection of BGP misconfiguration.

To this end, we have developed and deployed `Anteater` within `.nl`, which is a *real-time* monitoring system that retrieves DNS/TCP RTT continuously.

We show the `Anteater` architecture in Figure 12. First, traffic is collected at authoritative DNS servers of `.nl`, which are then exported to ENTRADA [57, 64], an open source DNS traffic streaming warehouse that employs Hadoop [56] and continuously ingests `pcap` files from these servers. (we also use ENTRADA in `.nl` not only for `Anteater`, but also for several other applications [37, 62]). It is *ENTRADA* that extracts RTT for incoming TCP packets, from its handshake sessions and makes it available to be queried using Impala [23], an open-source SQL engine for Hadoop.

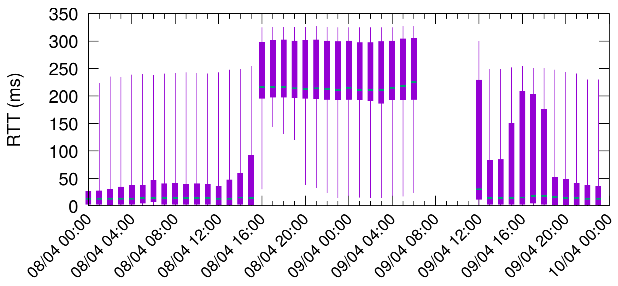


**Figure 13: Anycast B SYD site: Latency for IPv4.**

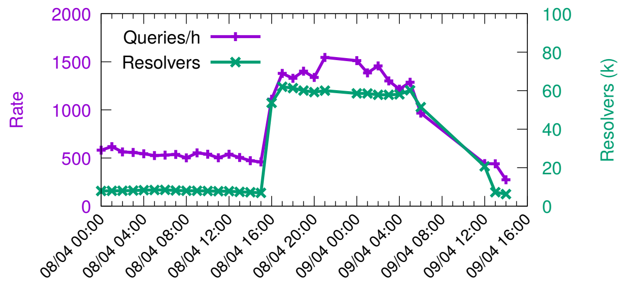


**Figure 14: Anycast B SYD site: queries rate and resolvers.**

`Anteater` then retrieves DNS/TCP RTT data for each anycast server, anycast site, and various ASes, on an hourly basis (given that `.nl` authoritative servers have good temporal coverage in 1h frames §2.1.3). It stores the information on a relational database (PostgreSQL). We then use Grafana [18], a data visualization dashboard and alert system. We then configure RTT thresholds that triggers Grafana to send `.nl` operators email alerts. `Anteater` has been used in `.nl` for the past 9 months and its proven helpful in detecting BGP misconfigurations. We include a series of `Anteater` screen captures in Appendix G. We release `Anteater` at [33].

Next we illustrate this use-case with one example from that deployment.

*EU traffic winding up in Australia:* On 2020-04-08, `.nl` operators received an alert from `Anteater` that detected a jump in median DNS RTT for Anycast B, from 55 ms to more than 200 ms (see Figure 13) but only for IPv4 traffic, and not for IPv6.

To investigate this change, we evaluated the number of ASes (Figure 13), resolvers, and query rates (Figure 14) using `Anteater`'s Grafana dashboard. We see that all grew when latency fell: with many more ASes and about 3× more queries and resolvers. To rule out DDoS attacks or a sudden burst in popularity for our domain, we confirmed that these ASes and resolvers have migrated from other sites (mostly Germany, site FRA) and went to SYD. Since many of these clients are in Europe, this nearly antipodal detour explains the latency increase.

We reached out to the operator of `.nl` Anycast B SYD. They confirmed and were already aware of the routing change. They informed us that a set of their SYD prefixes had accidentally to propagated through a large, Tier-1 transit provider. Since this provider peered with many other ASes in many places around the globe, their propagation of the Anycast B anycast prefix provided a shorter AS-Path and sent traffic to SYD.

We also confirmed these routing changes on the RIPE RIS database of routing changes [50]. (Details are in §H.2.)

While catchment changes are not bad, route leaks that mis-route Europe to Australia are not an improvement. The lightweight nature of DNS/TCP observations of latency support 24x7 monitoring and allowed us to detect this problem, which is why we developed `Anteater` to monitor the `.nl` operations.

## 5 ANYCAST LATENCY EFFECTS ON TRAFFIC

While DNS/TCP can be used to discover anycast latency, does latency matter? DNS caching means *users* are largely insulated from latency. However, we next confirm that latency does influence *traffic* to services when users have the choice of several. This effect was previously shown from clients [39], but not its impact on services.

Prior work has considered recursive resolver preference for lower latency [39]. Here we turn that analysis around and explore how changing anycast infrastructure shifts a client's preferences towards authoritative name servers. We confirm that lower latency results in increased traffic from recursive resolvers that have a choice between multiple anycast service addresses providing the same zone. (This question differs from studies that examine the optimality of a specific anycast service with multiple sites [25, 26].)

To examine this question we use public RSSAC-002 statistics for the root server system [52]. From this we use the "traffic-volume" statistic, which reports queries per day for each root anycast service. (Recall that the Root DNS is provided by 13 different anycast service addresses per IP version, each using a different anycast infrastructure.) We show 6 months of data here (2019-11-01 to 2020-05-31), but we noticed similar trends since 2016. This analysis omits G- and I-Root, which did not provide data during this period.

Figure 15 shows the fraction of traffic that goes to each anycast service in the root server system for one year. Two root letters deployed additional sites over this period: B-Root originally had 2 sites but added 3 sites in 2020-02-01, then optimized routing around 2020-04-01. H-Root originally had

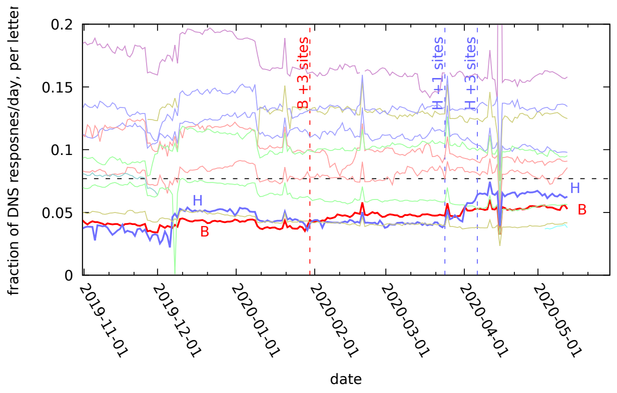


**Figure 15: Fraction of traffic going to each root anycast service, per day, from RSSAC-002 data. B- and H-Root are bold lines.**

2 sites but deployed 4 additional sites on 2020-02-11 and 3 additional sites on 2020-04-06. While other letters also added sites, B and H's changes were the largest improvements relative to their prior size. We see that, B and H's share rises from about 4% in 2019-11 to about 6% in 2020-05.

This data confirms that when new sites are created at a root letter, they offer some clients lower latency for that letter. Lower latency causes some clients to shift more of their traffic to this letter, so its share of traffic relative to the others grows.

## 6 RELATED WORK

*Passive TCP evaluation*: Janey Hoe was the first to extract RTT from the TCP handshake [19], and it has been used by several groups since then (for example, in Facebook HTTP traffic [53]). We use this old idea, but we apply it to DNS RTT estimation and to use to engineer and monitor in near real time Anycast DNS services. In a non peer-reviewed work performed previously but independently from our own, `.cz` operators [28, 29] also employed DNS/TCP RTT to evaluate latency from their services. While both use the same idea (derive latencies from the TCP handshake), ours provides a comprehensive validation (§2) and we act on the results, by carefully manipulating BGP to solve the identifed problems, and reduce latency in up to 90% (§4). Besides, our work includes freely three tools: `dnsanon`, `Anteater`, and a modified version of KnotDNS. Linux `ss` and `ip` utilities – both part of `iproute2` [27] can be also used to retrieve TCP sockets information, including RTT. However, they only provide *averages* estimatives, which are prone to application processing delays.

*Anycast performance and DNS:* Anycast has been an active research topic over the last years. Ballani *et al.* [5] have proposed using a single upstream provider to avoid routing unexpected behavior. Schmidt *et al.* [55] have investigate the impact of *number of sites* and performance of anycast services, and pointed that sometimes, more sites may even lead to performance degradation. Moura *et al.* [35] have investigated how anycast react when DDoS attacks take place, by analyzing the 2015 attacks against the Root DNS servers [51]. They show how catchment affects how sites experience the query load distribution, with some sites becoming unavailable and others remaining active.

There is one approach to measure anycast latency today: active measurements. RIPE Atlas [47] measures latency from about 11k physical devices distributed around the world. Commercial services are known to have fewer vantage points. Our approach instead uses passive analysis of TCP traffic from real clients. It provides far better coverage than RIPE Atlas (§2.1). We expect that Verfploeter will soon support RTT measurements. Even when it does support RTT measurements, our approach provides coverage for most of the networks that currently generate traffic. In addition, since our analysis is passive, it places no additional strain on other networks and can run 24x7.

Li *et al.* [25] have proposed using new BGP communities to improve the site catchment, which, in turn, would requires protocol changes. Contrary to their approach, ours relies only on passive TCP traffic and does not involve protocol changes.

*Anycast optimization for large CDNs with multiple providers*: Going beyond how many sites and where to place them, McQuistin *et al.* [31] have investigated anycast networks with multiple upstream providers – which is typical for large CDNs. Given that each site may have its own set of peers/uspstreams, that may create inconsistencies, like in the case of Google and Anycast A (§4.3.2). They devise a methodology aims at daily measure the catchments of the network using active measurements, and compare where and how it changes for different configuration scenarios in terms of peers. Our work relates to them in the sense that we change peering for two sites of Anycast A in §4.3.2 and §4.3.3, in order to fix the polarization issue. Schlinker *et al.* [53] analyzes the change of catchments and use TCP RTT on Facebook's CDN, which most traffic is HTTP, and whose primary clients are real users. Authoritative DNS traffic, on the other hand, typically relates to the between resolver and authoritative server, and it's mostly UDP.

*Performance-aware routing*: Todd *et al.* [3] compare data from proposals for performance-aware routing from three content/cloud providers (Google, Facebook, and Microsoft) and show that BGP fares quite well for most cases. Others proposed to perform traffic engineering based on packet loss, latency and jitter [40, 44].

## 7 CONCLUSIONS

We have shown that DNS TCP connections are a useful source of latency information about anycast services for DNS. Although TCP is not (today) the dominant transport protocol for DNS, we showed that there is enough DNS/TCP to provide good coverage for latency estimation. We also showed how we prioritize use of this information to identify problems in operational anycast networks. We have used this approach to study three operational anycast services: two anycast servers of `.nl`, and one root DNS server (`B-root`). We documented one new class of latency problems: anycast polarization, an interaction where hypergiants get pessimal latency (100–200 ms) because of an poor interaction between their corporate backbones and global anycast services. We showed how we addressed this problem for `.nl`'s Anycast A with both Google and Microsoft. We also documented several other problems for anycast latency discovered through our analysis of DNS/TCP and showed that it enables continuous monitoring. Last, we release freely two tools (`dnsanon` and `Anteater`), and a modified version of KnotDNS. We believe this approach will be of use to other DNS operators.

## ACKNOWLEDGMENTS

We thank the operators of `.nl` Anycast A and B and `B-root` for their time and collaboration. We also thank Casey Deccio for first proposing using TCP hanshake to measure DNS latency. We also thanks Klaus Darilion for his comments on paper drafts.

John Heidemann's research in this paper is supported in part by the DHS HSARPA Cyber Security Division via contract number HSHQDC-17-R-B0004-TTA.02-0006-I (PAADDOS), and by NWO.

Giovane C. M. Moura, Joao Ceron, Jeroen Bulten, and Cristian Hesselman research in this paper is supported by the Conconrdia Project, an European Union's Horizon 2020 Research and Innovation program under Grant Agreement No 830927.

## REFERENCES

[1] ANT ISI. 2021. Dnsanon: extract DNS traffic from pcap to text with optionally anonymization. https://ant.isi.edu/software/dnsanon/index.html.

[2] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. 2005. *DNS Security Introduction and Requirements.* RFC 4033. Internet Request For Comments. ftp://ftp.rfc-editor.org/in-notes/rfc4033.txt

[3] Todd Arnold, Matt Calder, Italo Cunha, Arpit Gupta, Harsha V. Madhyastha, Michael Schapira, and Ethan Katz-Bassett. 2019. Beating BGP is Harder than We Thought. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks* (Princeton, NJ, USA) *(HotNets '19)*. Association for Computing Machinery, New York, NY, USA, 9–16. https://doi.org/10.1145/3365609.3365865

[4] R. Austein. 2007. *DNS Name Server Identifier (NSID) Option.* RFC 5001. Internet Request For Comments. https://www.rfc-editor.org/rfc/rfc

5001.txt

[5] Hitesh Ballani and Paul Francis. 2005. Towards a Global IP Anycast Service. In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (Philadelphia, Pennsylvania, USA) *(SIGCOMM '05)*. Association for Computing Machinery, New York, NY, USA, 301–312. https://doi.org/10.1145/1080091.1080127

[6] Hitesh Ballani, Paul Francis, and Sylvia Ratnasamy. 2006. A Measurement-based Deployment Proposal for IP Anycast. In *Proceedings of the 2006 ACM Conference on Internet Measurement Conference (IMC)*. ACM, 231–244.

[7] R. Bellis. 2010. *DNS Transport over TCP—Implementation Requirements*. RFC 5966. Internet Request For Comments. https://www.rfc-editor.org/rfc/rfc5966.txt

[8] Ray Bellis. 2015. Researching F-root Anycast Placement Using RIPE Atlas. ripe blog https://labs.ripe.net/Members/ray_bellis/researching-f-root-anycast-placement-using-ripe-atlas. https://labs.ripe.net/Members/ray_bellis/researching-f-root-anycast-placement-using-ripe-atlas

[9] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. 2013. Mapping the Expansion of Google's Serving Infrastructure. In *Proceedings of the ACM Internet Measurement Conference* (johnh: pafile). ACM, Barcelona, Spain, 313–326. https://doi.org/10.1145/2504730.2504754

[10] Sebastian Castro, Duane Wessels, Marina Fomenkov, and Kimberly Claffy. 2008. A Day at the Root of the Internet. *ACM Computer Communication Review* 38, 5 (April 2008), 41–46.

[11] Pithayuth Charnsethikul. 2021. Knot Extensions to support TCP Soliciation. https://ant.isi.edu/software/dnsrtt/.

[12] CZ.NIC. 2011. Knot DNS. https://www.knot-dns.cz/.

[13] Wouter B. de Vries, Salman Aljammaz, and Roland van Rijswijk-Deij. 2020. Global Scale Anycast Network Management with Verfploeter. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium* (johnh: pafile). IEEE, Budapest, Hungary.

[14] Wouter B. de Vries, Ricardo de O. Schmidt, Wes Haraker, John Heidemann, Pieter-Tjerk de Boer, and Aiko Pras. 2017. Verfploeter: Broad and Load-Aware Anycast Mapping. In *Proceedings of the ACM Internet Measurement Conference*. London, UK. https://doi.org/10.1145/3131365.3131371

[15] John Dilley, Bruce Maggs, Jay Parikh, Harald Prokop, Ramesh Sitaraman, and Bill Weihl. 2002. Globally Distributed Content Delivery. *IEEE Internet Computing* 6, 5 (Sept. 2002), 50–58. https://doi.org/10.1109/MIC.2002.1036038

[16] Oliver Gasser, Quirin Scheitle, Pawel Foremski, Qasim Lone, Maciej Korczyński, Stephen D. Strowes, Luuk Hendriks, and Georg Carle. 2018. Clusters in the Expanse: Understanding and Unbiasing IPv6 Hitlists. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. Association for Computing Machinery, New York, NY, USA, 364–378. https://doi.org/10.1145/3278532.3278564

[17] Google. 2020. Google BGP communities. https://support.google.com/interconnect/answer/9664829?hl=en. (Jan. 2020).

[18] Grafana Labs. 2020. Grafana documentation. https://grafana.com/.

[19] Janey C. Hoe. 1996. Improving the Start-up Behavior of a Congestion Control Scheme for TCP. In *Proceedings of the ACM SIGCOMM Conference* (johnh: pafiles). ACM, Stanford, CA, 270–280.

[20] P. Hoffman, A. Sullivan, and K. Fujiwara. 2018. *DNS Terminology*. RFC 8499. IETF. http://tools.ietf.org/rfc/rfc8499.txt

[21] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. 2016. *Specification for DNS over Transport Layer Security (TLS)*. RFC 7858. Internet Request For Comments. https://doi.org/10.17487/RFC7858

[22] ICANN. 2014. RSSAC002: RSSAC Advisory on Measurements of the Root Server System. https://www.icann.org/en/system/files/files/rssac-002-measurements-root-20nov14-en.pdf.

[23] Marcel Kornacker, Alexander Behm, Victor Bittorf, Taras Bobrovytsky, Casey Ching, Alan Choi, Justin Erickson, Martin Grund, Daniel Hecht, Matthew Jacobs, et al. 2015. Impala: A Modern, Open-Source SQL Engine for Hadoop.. In *Cidr*, Vol. 1. 9.

[24] Craig Labovitz, Scott Iekel-Johnson, Danny McPherson, Jon Oberheide, and Farnam Jahanian. 2010. Internet Inter-Domain Traffic. In *Proceedings of the ACM SIGCOMM Conference* (johnh: pafile). ACM, New Delhi, India, 75–86. https://doi.org/10.1145/1851182.1851194

[25] Zhihao Li, Dave Levin, Neil Spring, and Bobby Bhattacharjee. 2018. Internet Anycast: Performance, Problems, & Potential. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication* (Budapest, Hungary) *(SIGCOMM '18)*. Association for Computing Machinery, New York, NY, USA, 59–73. https://doi.org/10.1145/3230543.3230547

[26] Jinjin Liang, Jian Jiang, Haixin Duan, Kang Li, and Jianping Wu. 2013. Measuring Query Latency of Top Level DNS Servers. In *Proceedings of the International conference on Passive and Active Measurements (PAM)*. 145–154.

[27] Linux Foundation. 2021. networking:iproute2 [Wiki]. https://wiki.linuxfoundation.org/networking/iproute2.

[28] Maciej Andzinski . 2019. Passive analysis of DNS server reachability. https://centr.org/library/library/centr-event/rd14-andzinski-passive-analysis-of-dns-server-reachability-20190529.html.

[29] Maciej Andzinski . 2019. Passive analysis of DNS server reachability. https://www.nic.cz/files/nic/IT_19/prezentace/12_andzinski.pdf.

[30] D. McPherson, D. Oran, D. Thaler, and E. Osterweil. 2014. *Architectural Considerations of IP Anycast*. RFC 7094. IETF. http://tools.ietf.org/rfc/rfc7094.txt

[31] Stephen McQuistin, Sree Priyanka Uppu, and Marcel Flores. 2019. Taming Anycast in the Wild Internet. In *Proceedings of the Internet Measurement Conference* (Amsterdam, Netherlands) *(IMC '19)*. Association for Computing Machinery, New York, NY, USA, 165–178. https://doi.org/10.1145/3355369.3355573

[32] P.V. Mockapetris. 1987. *Domain names - implementation and specification*. RFC 1035. IETF. http://tools.ietf.org/rfc/rfc1035.txt

[33] Giovane C. M. Moura. 2021. Anteater. https://github.com/SIDN/anteater

[34] Giovane C. M. Moura, Sebastian Castro, Wes Hardaker, and Cristian Hesselman. 2020. Clouding up the Internet: how centralized is DNS traffic becoming?. In *Proceedings of the ACM Internet Measurement Conference*. ACM, Virtual Conference, *(to appear)*.

[35] Giovane C. M. Moura, Ricardo de O. Schmidt, John Heidemann, Wouter B. de Vries, Moritz Müller, Lan Wei, and Christian Hesselman. 2016. Anycast vs. DDoS: Evaluating the November 2015 Root DNS Event. In *Proceedings of the ACM Internet Measurement Conference* (johnh: pafile). ACM, Santa Monica, California, USA, 255–270. https://doi.org/10.1145/2987443.2987446

[36] Giovane C. M. Moura, John Heidemann, Ricardo de O. Schmidt, and Wes Hardaker. 2019. Cache Me If You Can: Effects of DNS Time-to-Live (extended). In *Proceedings of the ACM Internet Measurement Conference* (johnh: pafile). ACM, Amsterdam, the Netherlands, to appear. https://doi.org/10.1145/3355369.3355568

[37] G. C. M. Moura, M. Müller, M. Wullink, and C. Hesselman. 2016. nDEWS: A new domains early warning system for TLDs. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*. 1061–1066.

[38] Moritz Müller, Giovane C. M. Moura, Ricardo de O. Schmidt, and John Heidemann. 2017. Recursives in the Wild: Engineering Authoritative DNS Servers. In *Proceedings of the ACM Internet Measurement Conference* (johnh: pafile). ACM, London, UK, 489–495. https://doi.org/10.1145/3131365.3131366

[39] Moritz Müller, Giovane C. M. Moura, Ricardo de O. Schmidt, and John Heidemann. 2017. *Recursives in the Wild: Engineering Authoritative DNS Servers*. Technical Report ISI-TR-720. USC/Information Sciences Institute.

[40] Priyadarsi Nanda and AJ Simmonds. 2009. A scalable architecture supporting QoS guarantees using traffic engineering and policy based routing in the internet. *International Journal of Communications, Network and System Sciences* (2009).

[41] T. Narten, R. Draves, and S. Krishnan. 2007. *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. RFC 4941. Internet Request For Comments. ftp://ftp.rfc-editor.org/in-notes/rfc4941.txt

[42] C. Partridge, T. Mendez, and W. Milliken. 1993. *Host Anycasting Service*. RFC 1546. IETF. http://tools.ietf.org/rfc/rfc1546.txt

[43] Enric Pujol, Ingmar Poese, Johannes Zerwas, Georgios Smaragdakis, and Anja Feldmann. 2019. Steering hyper-giants’ traffic at scale. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*. 82–95.

[44] Bruno Quoitin, Cristel Pelsser, Olivier Bonaventure, and Steve Uhlig. 2005. A performance evaluation of BGP-based traffic engineering. *International journal of network management* 15, 3 (2005), 177–191.

[45] RIPE NCC. 2019. RIPE Atlas Measurement IDS. https://atlas.ripe.net/measurements/ID. , where ID is the experiment ID: kroot-udp: 10311, kroot-tcp: 26995926, lroot-udp: 10308, lroot-tcp:26995949, GoDNS-21: 23859473 , GoTrace-21: 23859475 GoDNS-22: 23863904, GoTrace-22: 23863901, ComcastV6: 24269572, ComcastV6-afterReport: 24867517, ChinaNetV6: 24257938, DNS/TCP:22034303, DNS/UDP: 22034324,.

[46] RIPE NCC. 2020. RIPE Atlas Probes. https://ftp.ripe.net/ripe/atlas/probes/archive/2020/05/.

[47] RIPE NCC Staff. 2015. RIPE Atlas: A Global Internet Measurement Network. *Internet Protocol Journal (IPJ)* 18, 3 (Sep 2015), 2–26.

[48] RIPE Network Coordination Centre. 2015. RIPE Atlas. https://atlas.ripe.net.

[49] RIPE Network Coordination Centre. 2018. RIPE Atlas - Raw data structure documentations,https://atlas.ripe.net/docs/data_struct/.

[50] RIPE Network Coordination Centre. 2020. RIPE - Routing Information Service (RIS). https://https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris.

[51] Root Server Operators. 2015. Events of 2015-11-30. http://root-servers.org/news/events-of-20151130.txt.

[52] Root Server Operators. 2019. Root DNS. http://root-servers.org/.

[53] Brandon Schlinker, Italo Cunha, Yi-Ching Chiu, Srikanth Sundaresan, and Ethan Katz-Bassett. 2019. Internet Performance from Facebook’s Edge. In *Proceedings of the Internet Measurement Conference* (Amsterdam, Netherlands) *(IMC ’19)*. ACM, New York, NY, USA, 179–194. https://doi.org/10.1145/3355369.3355567

[54] Brandon Schlinker, Hyojeong Kim, Timothy Cui, Ethan Katz-Bassett, Harsha V. Madhyastha, Italo Cunha, James Quinn, Saif Hasan, Petr Lapukhov, and Hongyi Zeng. 2017. Engineering Egress with Edge Fabric: Steering Oceans of Content to the World. In *Proceedings of the ACM SIGCOMM Conference* (johnh: pafile). ACM, Los Angeles, CA, USA, 418–431. https://doi.org/10.1145/3098822.3098853

[55] Ricardo de O. Schmidt, John Heidemann, and Jan Harm Kuipers. 2017. Anycast Latency: How Many Sites Are Enough?. In *Proceedings of the Passive and Active Measurement Workshop* (johnh: pafile). Springer, Sydney, Australia, 188–200. http://www.isi.edu/%7ejohnh/PAPERS/Schmidt17a.html

[56] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. 2010. The hadoop distributed file system. In *2010 IEEE 26th symposium on mass storage systems and technologies (MSST)*. Ieee, 1–10.

[57] SIDN Labs. 2020. ENTRADA - DNS Big Data Analytics. https://entrada.sidnlabs.nl/.

[58] SIDN Labs. 2020. .nl stats and data. http://stats.sidnlabs.nl.

[59] Ankit Singla, Balakrishnan Chandrasekaran, P. Brighten Godfrey, and Bruce Maggs. 2014. The Internet at the Speed of Light. In *Proceedings of the* (johnh: pafile). ACM, Los Angeles, CA, USA. https://doi.org/10.1145/2670518.2673876

[60] Steve Souders. 2008. High-Performance Web Sites. *Commun. ACM* 51, 12 (Dec. 2008), 36–41. https://doi.org/10.1145/1409360.1409374

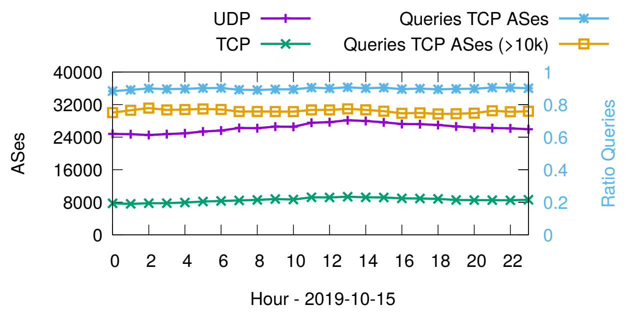
[61] Paul Vixie. 2012. Response Rate Limiting in the Domain Name System (DNS RRL). blog post http://www.redbarn.org/dns/ratelimits. http://www.redbarn.org/dns/ratelimits

[62] Thymen Wabeke, Giovane CM Moura, Nanneke Franken, and Cristian Hesselman. 2020. Counterfighting Counterfeit: detecting and taking down fraudulent webshops at a ccTLD. In *International Conference on Passive and Active Network Measurement (PAM2020)*. Springer, 158–174.

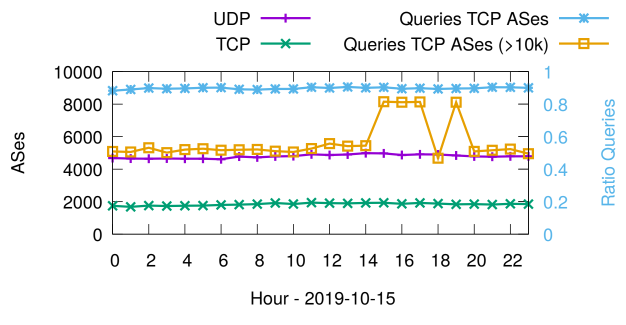
[63] Duane Wessels. 2020. RSSAC002-data. https://github.com/rssac-caucus/RSSAC002-data/.

[64] Maarten Wullink, Giovane CM Moura, Moritz Müller, and Cristian Hesselman. 2016. ENTRADA: A high-performance network traffic data streaming warehouse. In *Network Operations and Management Symposium (NOMS), 2016 IEEE/IFIP*. IEEE, 913–918.

[65] Pengxiong Zhu, Keyu Man, Zhongjie Wang, Zhiyun Qian, Roya Ensafi, J. Alex Halderman, and Haixin Duan. 2020. Characterizing Transnational Internet Performance and the Great Bottleneck of China. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 1, Article 13 (May 2020), 23 pages. https://doi.org/10.1145/3379479

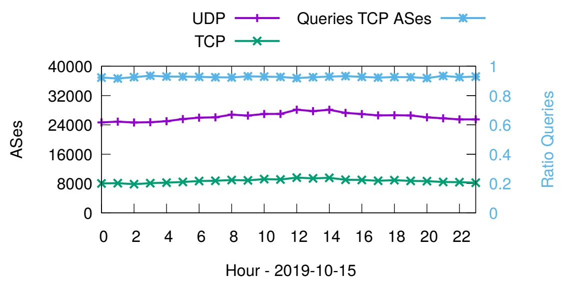


(a) Anycast B: IPv4

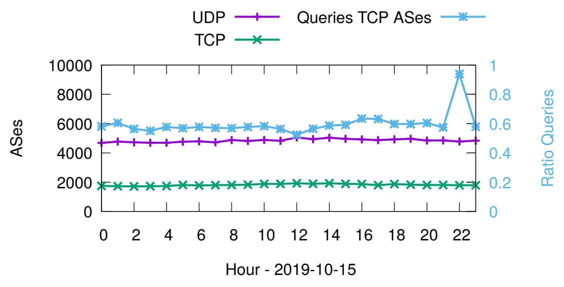


(b) Anycast B: IPv6

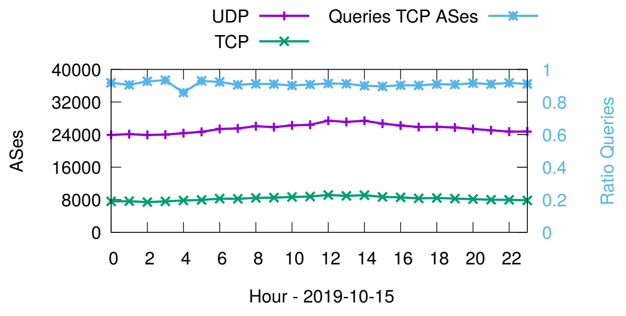
**Figure 16: `.nl` temporal coverage for Anycast B**



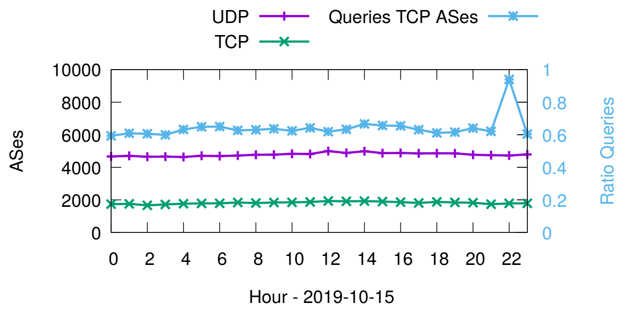
(a) Anycast A: IPv4



(b) Anycast A: IPv6



(c) Anycast B: IPv4



(d) Anycast B: IPv6

**Figure 17: `.nl` temporal coverage for Anycast A and B on 2019-10-21**

## A EXTRA GRAPHS ON TEMPORAL COVERAGE

Figure 16 show the temporal coverage of Anycast B for `.nl`.

Figure 17 shows temporal coverage for 2019-10-21.

## B ANYCAST A AND B TOP ASES

Table 7 shows ASes names and countries for top ASes observed for Anycast A and Anycast B.

## C ANYCAST B EXTRA DATA

## D EXTRA B-ROOT DATA

Table 8 shows all top talkers by organizations, and Table 9 by IPv4, and Table 10 by IPv6.

Figure 20 shows additional latency information.

## E `K-ROOT` DATA

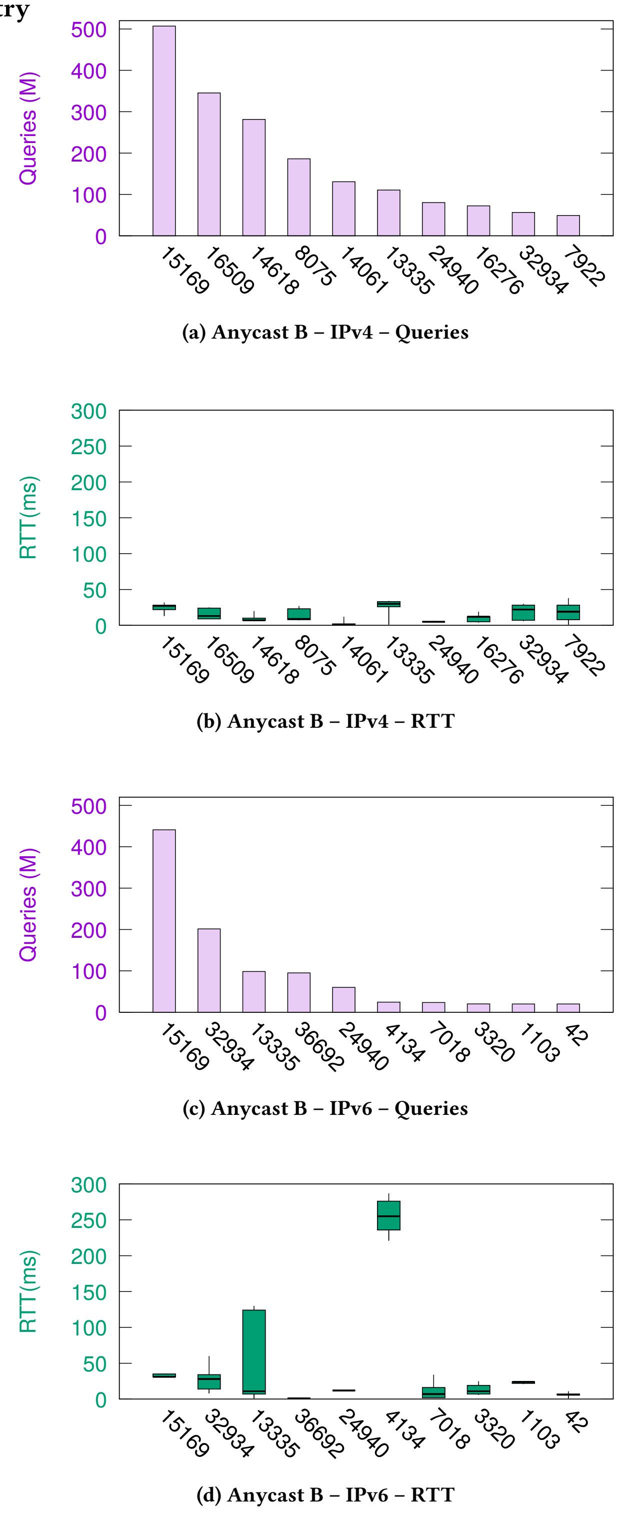
Figure 21 shows the CDF of the RTT for `K-Root` from Atlas probes (Table 4) . Similarly to `L-Root` data (Figure 3), it shows that most probes have similar RTT (median and 90%ile) towards this anycast service.

| AS Number | AS Name | Country |
|---|---|---|
| 42 | WoodyNet (PCH) | US |
| 1103 | SURFNet BV | NL |
| 2637 | Georgia Institute of Technology | US |
| 3320 | Deutsche Telekom AG | DE |
| 4134 | China Telecom Backbone | CN |
| 7018 | AT&T Services, Inc. | US |
| 7342 | Verisign infrastructure and Operations | US |
| 7922 | Comcast | US |
| 8075 | Microsoft Corporation | US |
| 13335 | Cloudflare | US |
| 14061 | DigitalOcean LLC | US |
| 14618 | Amazon.com Inc. | US |
| 15169 | Google | US |
| 16276 | OVH SAS | FR |
| 16509 | Amazon.com Inc. | US |
| 23033 | Wowrack.com | US |
| 24940 | Hetzner Online GmbH | DE |
| 32934 | Facebook, Inc. | US |
| 36692 | Cisco OpenDNS, LLC | US |

**Table 7: ASes in the Top 10 Lists of Anycast A and B**

| ASN | CC | Owner |
|---|---|---|
| 27717 | VE | Corporacion Digitel C.A. |
| 4837 | CN | CHINA169-BACKBONE CNCGROUP China169 Backbone |
| 38266 | IN | VODAFONE-IN Vodafone India Ltd. |
| 20940 | EU | AKAMAI-ASN1 |
| 45899 | VN | VNPT-AS-VN VNPT Corp |
| 6057 | UY | Administracion Nacional de Telecomunicaciones |
| 10620 | CO | Telmex Colombia S.A. |
| 28573 | BR | CLARO S.A. |
| 14061 | US | DIGITALOCEAN-ASN - DigitalOcean, LLC |
| 1267 | EU | ASN-WIND IUNET |
| 45356 | LK | MOBITEL-LK IS Group, No:108, W A D Ramanayake Mawatha |
| 7629 | PH | EPLDT-AS-AP 5F L.V. Locsin Bldg |
| 32934 | US | FACEBOOK - Facebook, Inc. |
| 10429 | BR | Telefonica Data S.A. |
| 4007 | NP | SUBISU-CABLENET-AS-AP Subisu Cablenet (Pvt) Ltd, Baluwatar, Kathmandu, Nepal |
| 9299 | PH | IPG-AS-AP Philippine Long Distance Telephone Company |
| 35819 | SA | MOBILY-AS Etihad Etisalat Company (Mobily) |
| 6147 | PE | Telefonica del Peru S.A.A. |
| 18881 | BR | TELEFÔNICA BRASIL S.A |
| 47211 | RU | KOLPINONET-AS |

**Table 8: Top 20 talker organizations to `B-root`– see Figure 6.**



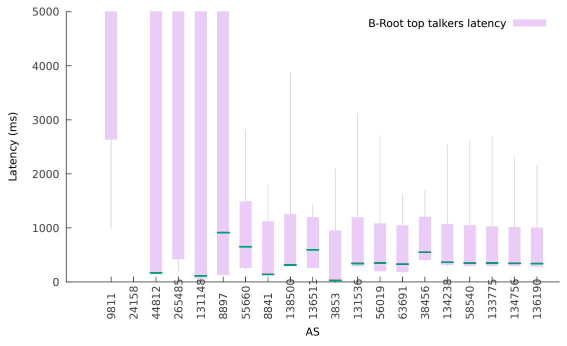
**(a) Anycast B – IPv4 – Queries**

**(b) Anycast B – IPv4 – RTT**

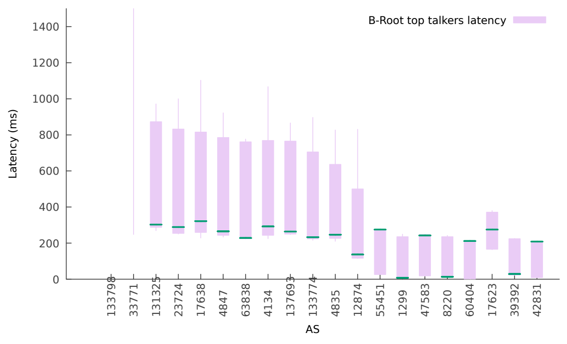
**(c) Anycast B – IPv6 – Queries**

**(d) Anycast B – IPv6 – RTT**

**Figure 18: `.nl` Anycast B query RTT for the top 10 ASes ranked by most queries (bars left axis). Data: 2019-10-15 to -22.**

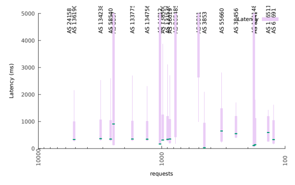


(a) `B-root` AS latency diversity for IPv4 by rank

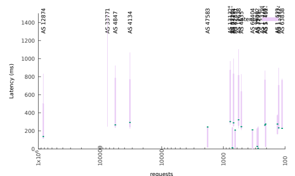


(b) `B-root` latency diversity for IPv6 by rank

**Figure 19: Latency analysis to `B-root` by AS latency diversity.**



(a) `B-root` AS latency diversity for IPv4 by data size



(b) `B-root` AS latency diversity for IPv6 by datasize

**Figure 20: Latency analysis to `B-root` by AS latency diversity.**

| ASN | CC | Owner |
|---|---|---|
| 9811 | CN | BJGY srit corp.,beijing. |
| 24158 | TW | TAIWANMOBILE-AS Taiwan Mobile Co., Ltd. |
| 44812 | RU | IPSERVER-RU-NET Fiord |
| 265485 | BR | UP NET TELECOM |
| 131148 | TW | BOT-AS-TW Bank Of Taiwan |
| 8897 | GB | KCOM-SPN (Service-Provider Network) (ex-Mistral) |
| 55660 | ID | MWN-AS-ID PT Master Web Network |
| 138500 | NP | AS138500 |
| 136511 | PH | BEC-AS-AP Broadband Everywhere Corporation |
| 3853 | US | WHIDBEY - Whidbey Telephone Company |
| 134756 | CN | CHINANET-NANJING-IDC CHINANET Nanjing IDC network |
| 58540 | CN | CHINATELECOM-HUNAN-ZHUZHOU-MAN Zhuzhou |
| 134238 | CN | CT-JIANGXI-IDC CHINANET Jiangx province IDC network |
| 38456 | AU | SPEEDCAST-AU SPEEDCAST AUSTRALIA PTY LIMITED |
| 133775 | CN | CHINATELECOM-FUJIAN-XIAMEN-IDC1 Xiamen |
| 4913 | US | NET-CPRK - Harris CapRock Communications, Inc. |
| 5377 | NO | MARLINK-EMEA |
| 55722 | NR | CENPAC-AS-AP Cenpac Net Inc |
| 134771 | CN | CHINATELECOM-ZHEJIANG-WENZHOU-IDC WENZHOU, ZHEJIANG Province, P.R.China. |
| 136190 | CN | CHINATELECOM-YUNNAN-DALI-MAN DaLi |

**Table 9: Top 20 IPv4 ASNs with large handshake latency distributions.**

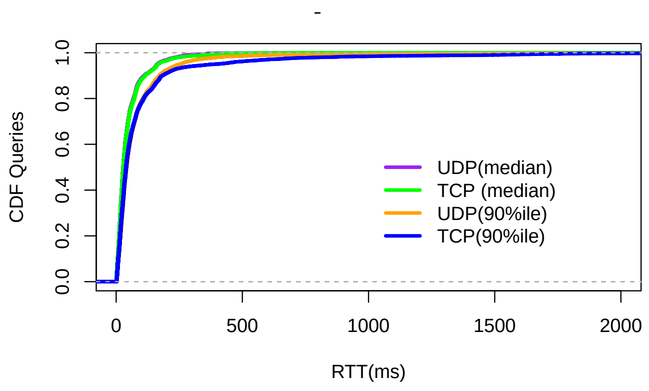


**Figure 21: `k-Root`: CDF of median and 90%file RTT for DNS/UDP and DNS/TCP.**

| ASN | CC | Owner |
|---|---|---|
| 133798 | ID | SMARTFREN-AS-ID PT. Smartfren Telecom, Tbk |
| 33771 | KE | SAFARICOM-LIMITED |
| 23724 | CN | CHINANET-IDC-BJ-AP IDC, China Telecommunications Corporation |
| 63838 | CN | CT-HUNAN-HENGYANG-IDC Hengyang |
| 12874 | IT | FASTWEB |
| 55451 | TH | AS55451 |
| 47583 | LT | AS-HOSTINGER |
| 60404 | NL | LITESERVER |
| 17623 | CN | CNCGROUP-SZ China Unicom Shenzen network |
| 42831 | GB | UKSERVERS-AS UK Dedicated Servers, Hosting and Co-Location |
| 4809 | CN | CHINATELECOM-CORE-WAN-CN2 China Telecom Next Generation Carrier Network |
| 17754 | IN | EXCELL-AS Excellmedia |
| 202196 | NL | BOOKING-BV |
| 12989 | NL | HWNG |
| 264496 | BR | IR TECNOLOGIA LTDA ME |
| 37009 | NA | MTCASN |
| 7633 | IN | SOFTNET-AS-AP Software Technology Parks of India - Bangalore |
| 45899 | VN | VNPT-AS-VN VNPT Corp |
| 27435 | US | OPSOURCE-INC - Dimension Data Cloud Solutions, Inc. |
| 45570 | AU | NETPRES-AS-AP Network Presence |

**Table 10: Top 20 IPv6 ASNs with large handshake latency distributions.**

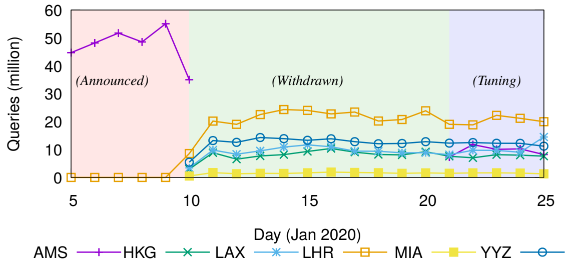
## F DEPOLARIZING GOOGLE: IPV6 GRAPHS

Figure 22 shows the IPv6 depolarization graphs for Anycast A and Google.

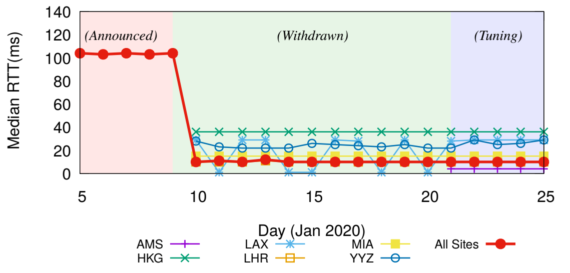
## G ANTEATER SCREENSHOTS

We use Grafana as a visualization dashboard for `Anteater`. Figure 23 shows a series of screenshots. In Figure 23a, we show the average RTT for each anycast service and IP version. The data is shown in hourly bins, which is the frequency `Anteater` retrieves data. For this graph, we have configure an alert on 60ms (horizontal lines). Value above that trigger alerts. Note that for this graph, we aggregate data across every single anycast site of each authoritative server.

The next graph (Figure 23b) shows the average DNS/TCP RTT for each *anycast site* of Anycast B (we remove each site name for anonymity). Similarly to Figure 23a, we have configure alerts for each anycast site, given sites may have a wide variation in RTT depending on where its clients come from.



**(a) IPv4 - RTT (ms)**



**(b) IPv6 - RTT(ms)**

**Figure 22: `.nl` Anycast A: queries and median RTT per site from Google (AS15169) – January 2020.**

Figure 24 shows the DNS/TCP RTT per AS number.

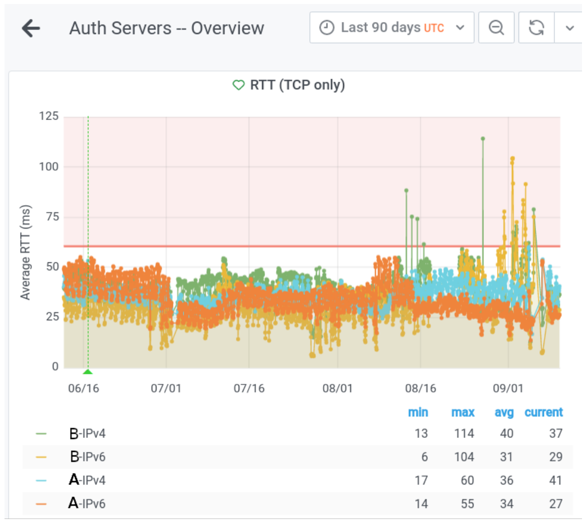
## H CASE STUDY DETAILS

### H.1 Details of the `.nl` Anycast A AMS site Peering with Google

*Bringing back AMS.* We first start by announcing our prefixes with 2x prepending towards Google in AMS. That causes *part* of the traffic to AMS (Figure 25), as we intended. Adding No-export option to the announcement, on the 22nd did not have any effect (private peering) We experimented further (operations 3–4) with prepeding our announcement only 1 time. That, in turn, caused Google to become polarized again for Anycast A

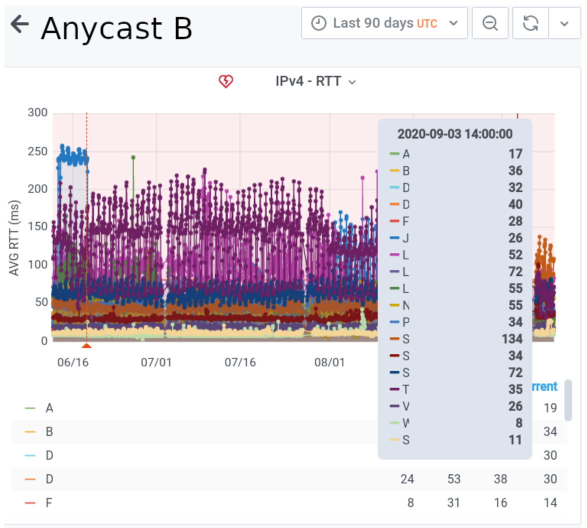
We also carried measurements from 20 Atlas probes located at Google' s networks during these changes, as shown in Table 11. Figure 26 show these results for Altas.

So we determined empirically that 2x prepend solved the issue, and 1x did not. We wondered if we could reduce the polarization with BGP communities while keeping prepending to 1x. We tested at Op. 5 the the community `15169:13000`, which is provided by Google places lowest priority on BGP choices ("try to not serve traffic here") [17]. We see in Figure 27 that this operation (10:37) caused not catchment changes,



| | min | max | avg | current |
|---|---|---|---|---|
| B-IPv4 | 13 | 114 | 40 | 37 |
| B-IPv6 | 6 | 104 | 31 | 29 |
| A-IPv4 | 17 | 60 | 36 | 41 |
| A-IPv6 | 14 | 55 | 34 | 27 |

(a) DNS/TCP RTT per Anycast server and IP version – 3 months of hourly data. Values above horizontal line trigger an email alert.



2020-09-03 14:00:00

| | |
|---|---|
| A | 17 |
| B | 36 |
| D | 32 |
| D | 40 |
| F | 28 |
| J | 26 |
| L | 52 |
| L | 72 |
| L | 55 |
| N | 55 |
| P | 34 |
| S | 134 |
| S | 34 |
| S | 72 |
| T | 35 |
| V | 26 |
| W | 8 |
| S | 11 |

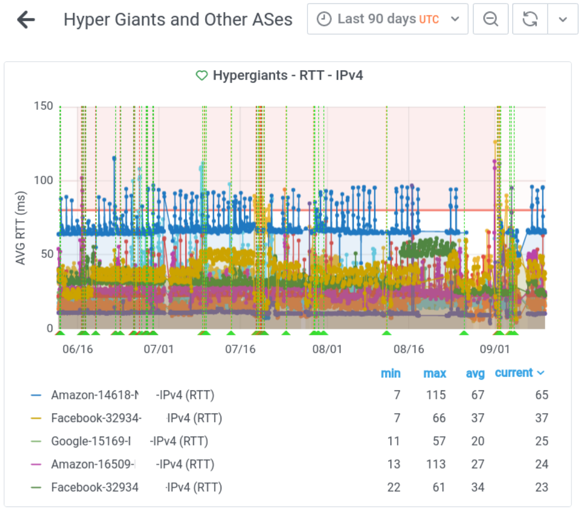
| | | | | |
|---|---|---|---|---|
| A | | | | 19 |
| B | | | | 34 |
| D | | | | 30 |
| D | 24 | 53 | 38 | 30 |
| F | 8 | 31 | 16 | 14 |

(b) DNS/TCP RTT per Anycast site of Anycast B (site name removed): 3 months of hourly data – overlay shows values at specified time.

Figure 23: `Anteater` screenshots on production in `.nl`.

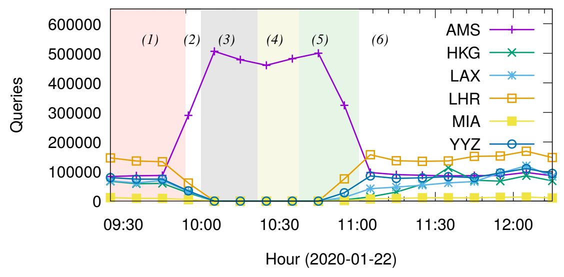
| ID | Type | Frequency |
|---|---|---|
| GoDNS-21 | DNS `hostname.bind` | 300s |
| GoTrace-21 | Traceroute | 300s |
| GoDNS-22 | DNS `hostname.bind` | 300s |
| GoTrace-22 | Traceroute | 900s |

Table 11: Ripe Atlas measurements used in the AS15169 tuning experiments on (2020-01-21 and 22) [45].

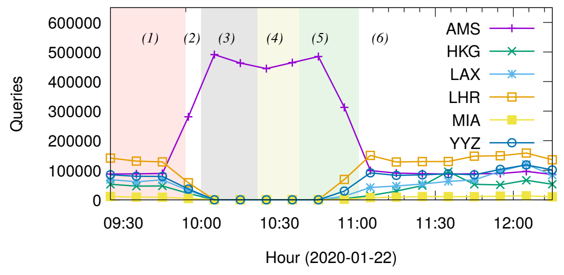


| | min | max | avg | current |
|---|---|---|---|---|
| Amazon-14618-N -IPv4 (RTT) | 7 | 115 | 67 | 65 |
| Facebook-32934- -IPv4 (RTT) | 7 | 66 | 37 | 37 |
| Google-15169-I -IPv4 (RTT) | 11 | 57 | 20 | 25 |
| Amazon-16509- -IPv4 (RTT) | 13 | 113 | 27 | 24 |
| Facebook-32934 -IPv4 (RTT) | 22 | 61 | 34 | 23 |

Figure 24: DNS/TCP RTT per Cloud Provider AS number (server name removed)



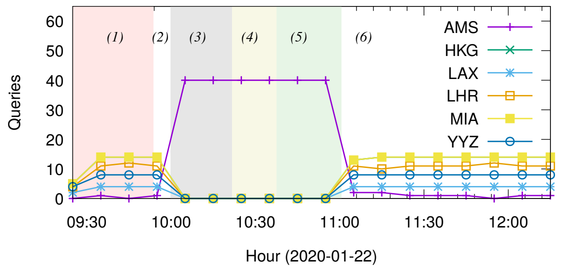
(a) IPv4 - Anycast A



(b) IPv6 - Anycast A

Figure 25: Catchment changes according to BGP manipulations (from Anycast A passive data).

and, as such, the communities did not influence traffic selection. Even though Google supports BGP communities, the ultimate decision on how to use it its own their side (and they are clear about it [17]). This particular example showcase when communities do not work as one hoped for.



(a) IPv4 - Ripe Atlas 20 VPs from Google AS15169

**Figure 26: Catchment changes according to BGP manipulations from Ripe Atlas.**

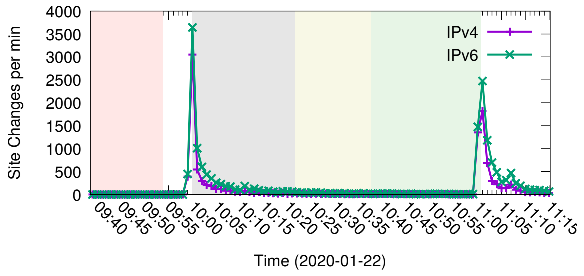


**Figure 27: Timeseries (per minute) of Google' s resolver changing sites on Anycast A.**

Last, we prepend the announcement 2x, and traffic resume going to AMS. As can be seen in Figure 9, the median RTT for Google and Anycast A improved significantly. Overall, BGP operations do not always work as one would expect.

## H.2 Details about Detections of BGP Misconfiguration

By investigating routing events associated with Anycast B on RIPE RIS [50] we could confirm changes in the route visibility of the results reported in §4.4.. RIPE RIS is a public database that collects and store Internet routing data in several points distributed globally, typically located in IXPs. Internet data routing data consist of BGP messages observed by collectors, including prefix announcement, prefix withdraw and update messages. For instance, it is possible to have a historical view of the paths to a specific destination.

Figure 28 shows the number of paths to Anycast B seen by Route RIS collectors. On 2020-04-08 at 15:59:50 we observed several BGP update messages that have triggered an increase of the number of paths to Anycast B, from 340 to 362. The visibility of Sever B became quite steady till the next day, on 2020-04-08 at 06:37, when, after a few BGP withdraw messages, it has returned again to 340.

The routing events coincide with the observation of our DNS/TCP RTT measurement as depicted on Figure 13. Routing updates force BGP neighbors to reprocess the routing table and update the best route selection. Thus, after new routing paths provided by the first set of BGP update messages (2020-04-08 at 15:59:50) the traffic was shifted to new routes and ended up in our site in SYD. The situation changed back to the previous situation when the new routing path was removed on 2020-04-08 at 06:37.

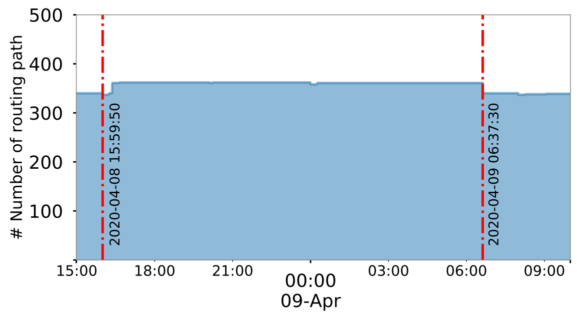


**Figure 28: Number of paths to Sever B seen by Route RIS collectors.**