If a Path is Inflated, and Noone Uses It, Is It Inefficient?

Anonymized

ABSTRACT

Anycast is a means of distributing content that has been praised for its simplicity and performance, yet criticized for inflating user latencies in some cases. However, anycast CDNs and anycast DNS resolvers serve latency-sensitive content to millions presenting an apparent contradiction. To resolve this discrepancy in understanding, we analyze anycast in two important settings: the root DNS and a large anycast CDN, placing an emphasis on how anycast's performance translates to end user latency. First, we question whether latency matters at all, and find that while it does in the CDN setting, it makes little difference in the root DNS due to heavy caching near end users – root DNS resolution only accounts for at most 2 ms per day for most users. We then examine how anycast specifically impacts performance to end users, and again demonstrate users see negligible effects from anycast path inflation. Users of a large anycast CDN, conversely, can experience a few ms per page load of anycast path inflation, with the effect becoming more prevalent in larger anycast deployments. The effects of the inflation, however, are dwarfed by the performance benefits users see by, on average, visiting closer anycast sites. Hence we demonstrate that although anycast can inflate latencies, the degree to which this affects users largely depends on the setting, and the prevalence of this inflation does not tell the whole story – anycast still does a good job routing users to sites.

1 INTRODUCTION

IP anycast, an approach to routing in which geographically diverse servers known as anycast replicas all use the same IP address, is used by a number of operational DNS [1, 2, 4, 10, 15] and CDN [5, 8, 26] systems today, in part because of its support to improve latency to clients and decrease load on each anycast server [17, 21, 23]. However, studies ([19, 24]) have argued that anycast provides sub-optimal performance for some users, compared to the lowest latency one could achieve given deployed replicas. Despite results demonstrating theoretical routing inefficiency, how these inefficiencies impact user experience is not well understood. Anycast, therefore, presents a paradox providing benefits of increased capacity and decreased latency while also purportedly hurting performance.

To resolve this, we take a step back and ask pointed questions – why do distributed systems such as CDNs and operational DNS use anycast, how does anycast affect their performance, and how does this performance ultimately translate to end user experience? Towards answering these questions we analyze anycast from two different angles: the root DNS and a large anycast CDN, chosen for their overlapping, yet distinctive, goals. The root DNS servers feature in studies [11, 12, 19, 22**?**] involving anycast because it is relatively easy to gain access to root DNS data, because it is straightforward to gain access to information about their deployments, and because they are run by several organizations [1]. This last fact manifests itself in a diverse set of deployment strategies, for essentially the same service. Here, examining the root DNS and anycast CDNs is particularly interesting because this analysis illustrates how the setting in which we study anycast heavily influences the conclusions we can draw – while we find that mitigating anycast path inflation is quite important for anycast CDNs, the impact of latency *at all* in the root DNS setting is negligible.

Although we do wish to examine how anycast affects performance both at the roots and in anycast CDNs, we first take a step back and examine whether performance (that is, latency) matters *at all* (section 3). By leveraging global root DNS traces and user data from a large anycast CDN, we demonstrate that the effect of root DNS latency on user-perceived latency is negligible, accounting for perhaps a few milliseconds of wait time per day or fractions of a percent of a page load. This is due mostly to heavy caching of root DNS records; hence, regardless of how inflated paths to the root replicas are, this latency is amortized over large user populations. Conversely, we show that latency matters considerably for anycast CDNs, comprising tens of percents of page load time (section 5). Collectively these results suggest organizations managing the root servers have little incentive to improve latency and, consequently, it makes little sense to evaluate the latency achievable with anycast by looking at the root servers.

With these basic results about latency, we revisit frequently posed questions about how anycast specifically impacts these services. We find that, even though round-trip times differ significantly by root DNS anycast deployment size, these differences are negligible when looked at from a per-page load perspective, making at most an INSERT_NUMBER millisecond difference. Similarly, even though we find that increasing deployment size can lead to more inflation in the roots, this inflation negligibly factors into page load times (section 4). Conversely, we find that for an anycast CDN, although increasing deployment sizes does make anycast path inflation more prevalent, the latency per page load decreases by tens of milliseconds (section 6) with additional sites. Moreover, regardless of deployment size, the path inflation for the large anycast CDN is less than half that of the roots for more than 90% of users (of the large anycast CDN). Hence larger deployment sizes can provide tangible latency benefits to anycast CDNs, but probably provide little benefit in the root DNS setting, and the magnitude of these benefits are largely dependent on deployment details.

2 BACKGROUND: ANYCAST IN DISTRIBUTED SYSTEMS

IP anycast is a system in which geographically diverse servers known as anycast replicas all use the same IP address, and therefore rely on BGP's notion of propagating the best route towards the destination. We discuss why distributed services on the Internet may wish to use Anycast, and ultimately leverage this knowledge in later sections to discuss why the root DNS and CDNs may use IP anycast. The benefits we are about to mention are *potential* – the realization of such benefits depends on the deployment details of the system using IP anycast. We also provide some background on the two services we investigate – the root DNS and a large anycast CDN, as this context will prove useful in later sections.

2.1 Potential Benefits of Anycast

[why distributed systems might use anycast, different performance benefits they might shoot for] IP anycast is first and foremost, simple and scalable. Network managers offload the responsibility of mapping users to sites to the network, and may add more sites by simply advertising the address from more locations. IP anycast can provide low latencies from users to destinations, since it relies on the network's notion of "best path"; practically, this is handled by the Border Gateway Protocol (BGP). Anycast also offers resiliency in that, should one site go offline, the network will handle routing users to different sites that are still running. A final potential benefit offered by anycast is load balancing. The idea is that users are spread out over the network, and so they will roughly be routed to corresponding anycast sites in different areas of the network [21]. Although recent research suggests anycast may not naturally balance load as was previously thought [19], what is more potentially useful is that anycast routes traffic predictably. Prior work suggests that only a very small fraction of anycast paths are unstable [**?**], and so network managers may provision in advance for expected (if unbalanced) load.

2.2 Root DNS anycast

[brief description of dns resolution process, caching potential] DNS is a fundamental service on the Internet that maps human readable hostnames to IP addresses [**? ?**]. To resolve a hostname, a user will send DNS requests in the form of UDP packets to one or more recursive resolvers (RRs) provided by their ISP[1]. The RR then requests the records from a root DNS server, top level domain (TLD) server and authoritative DNS (ADNS) server corresponding to the record the user requested. Since each request is a correspondence between the RR and a remote server, there can be several requests made by the RR for a single end-user request.

The root DNS servers [1] are grouped into thirteen letters, and are managed by twelve distinct organizations. Each letter consists of a certain number of anycast replicas, with actual numbers ranging from a few to a few hundred, and each letter is assigned a unique v4 and v6 address. Each replica serves DNS records for the TLD records, of which there are a little over 1,000. All but two of these records has a TTL of two days, with the exceptions having TTL's of one day. When a RR needs to query a root server, it may query whichever one it wishes (subject to network administration policy); however, recursive resolver software is known to query high performing (that is, low latency) letters more often. The performance of anycast in the root DNS has been extensively studied in a variety of contexts [11, 12, 19, 22**?**].

2.3 CDN Anycast

[description of microsoft's CDN, with a focus on intricacies of rings, and how we use them to simulate deployments] To investigate anycast, To study anycast in a setting that contrasts the root DNS, we leverage a large anycast CDN that serves millions of users from more than 100 front ends. Traffic destined for the large anycast CDN will enter the CDNs network at a peering point, and be routed to one of the anycast replicas serving the content. This large anycast CDN has a logical hierarchy of layers, called rings, that serve different types of content. Hence, traffic from a user prefix destined for the CDN may end up at two different front ends (depending on the service), but will ingress into the network at the same peering point.

In the following, we use these rings to simulate anycast deployments of different sizes, as a way of coarsely assessing anycast's behavior with increasing deployment size. That is, since various services use different sized rings, we are able to isolate performance metrics for each ring. Although

[1] The user can specify whatever RR they wish, but one can sensibly assume the typical user's RR is set by the ISP, broadcasted through DHCP.

the front ends serving content are anycasted, this is not a perfect analogy to actually deploying different sized anycast networks on the Internet, since traffic traversing the public Internet will take the same path, regardless of the ring in question. Hence we are, to some extent, only assessing anycast within the CDNs network.

3 DOES DNS ROOT LATENCY MATTER?

Before answering questions about anycast latency, we would first like to understand root DNS latency. Here we not only measure what root DNS latencies are, but also how this ground truth latency manifests itself in *user perceived* latency. We approach this from two main perspectives: local and global. The former allows us to estimate the fraction of page load time (PLT) during which a client is waiting for root DNS resolution and the latter allows us to estimate the total time per day a user is waiting for root DNS resolution.

[ISI results – root dns latency cdf and PLT implications Main idea is cache hit rate is high, and root requests are rarely generated.]

3.1 Data Sources

The recursive resolver of interest (running BIND 9) saves all traffic traversing over port 53 to file, and has done so for 5 years, providing us with a rich source of data. This corpus of users is relatively small, and consists of university traffic, so the specifics of the analysis we conduct may not extend to other RR's. For example, over 2018, we saw roughly 900 unique IP addresses, with about 100 unique IP addresses each day. This might be a smaller corpus of users than is seen at the RR of, say, an ISP in a large metro area. Nevertheless, we wish to observe some high-level features of the data, and their implications. Figure **??** shows the latencies of all queries seen at ISI over one year. We can see that the latencies are divided into (roughly) 3 regions: sub-millisecond latency, low latency, and high latency. The first region corresponds to cached queries. The second region probably corresponds to DNS resolutions for which the resolving server was close. Finally, the third region probably corresponds to queries that had to travel to distance servers, or required a few rounds of recursion to fully resolve the domain. The results, strikingly similar to those presented in [9], suggest that ISI is no more or less "connected" than the typical recursive resolver.

3.2 How Often Does a User Interact with the Roots?

From packet traces of all of 2018, we would like to quantify the effect caching root DNS records has on users, and how this differs from the ideal behavior users can experience. Specifically, we are interested in the number of queries to the root server as a fraction of user requests to the recursive resolver. We refer to this metric as the cache miss rate, as it approximates how often a TLD record is not found in the cache of the RR in the event of a user query. We say approximately since, for example, the recursive resolver may have sent multiple root requests per user query, or root requests without any user query triggering them.

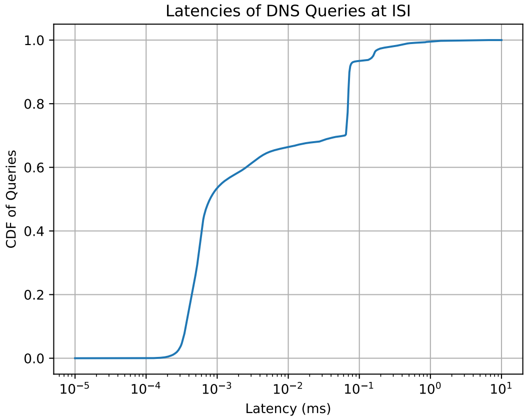

Figure 1: CDF of user DNS query latencies seen at a recursive resolve at ISI, over the course of one year.

Assumptions	Web Page Load Time (ms)	3,000
	Root DNS Latency (ms)	500
	Number of DNS Look-Ups Per Web Page	3
Statistics	Number of User Queries (millions)	14.9
	Number of Root Transactions	73,200
Implications	Percent of user Queries Resulting in a Root Transaction	.49
	Expected Speed-up in PLT with No Root Latency (ms)	8
	Resulting PLT Speedup (percent)	.25

Table 1: Root querying statistics gathered from the ISI RR for a representative month of 2018, and associated implications of how root latency impacts users of ISI.

Relevant statistics and their implications on user-perceived latency are presented in Table **??**. We find that daily cache miss rates of the resolver range from .1% to 4.5%, with a median value of .5%. This is quite a wide range of cache miss rates, and is potentially skewed by the traffic generated by the internet measurement lab (ISI). However, assessing the latency implications for cache miss rates higher than the median is simply a multiplicative factor and the qualitative conclusions are the same.

To calculate the impact of root latency on user experience, assume that a web page load takes N seconds, that

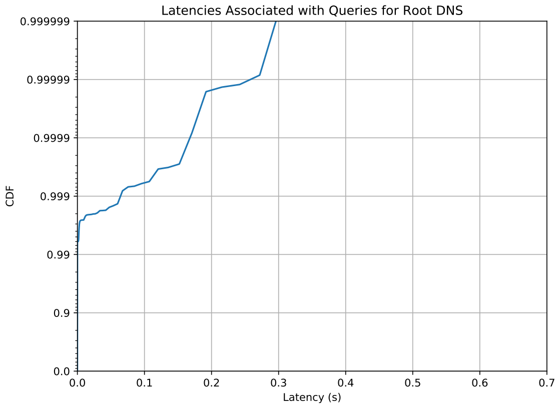

Figure 2: Root DNS latency for queries made by users of the ISI recursive resolver during 2018. user queries that did not generate a query to a root server were given a latency of 0.

there are M serial DNS requests per page, that the root latency is k seconds, and that the cache miss rate is p. Then, the resulting (average) latency due to root DNS resolution per page load is given by pMk, which, as a fraction of PLT is $\frac{pMk}{N}$.

Although N and k depend on a number of factors difficult to measure, p can be measured for each recursive resolver (as shown in Table **??**), and M is relatively constant. To demonstrate this last point, we performed page loads for 500 popular domains using the Selenium web-browser and calculated the number of blocking DNS requests per page load using methods in [25]. We find 95% of requests resulted in M ≤ 3 and so use this in what follows. As exaggerated upper bounds on the impact of root DNS latency, we choose N = 3 and k = .5 [2]. Hence, supposing root latency is completely removed from the equation, a user saves, on average, 8 ms per page load. As a percentage of the total page load time, this is .25%, which is measurable, but not significant. Note that for a conservative estimate of the cache miss rate of 4.5%, this would translate to approximately 2.3% of a single page load.

As a visualization of these results, Figure **??** shows a CDF of root DNS latency experienced for queries over 2018. Requests that do not generate a query to a root server are counted as having a root latency of 0.

Figure **??** suggests it is particularly rare for a user query to generate a query to a root DNS server and Table **??** demonstrates that the latency implications are small when amortizing latencies over large user populations. Collectively these demonstrate that users of ISI don't experience much latency due to root DNS resolution. However, the impact on user performance is not as small as one would expect. For example, a particular day during January 2018 sees as many as 900 queries to the root server for the COM NS record. Given the 2 day TTL of this record, this query frequency is absurdly large. This suggests that heuristic arguments that users rarely experience root latency because cached TLD records have long TTL's are not sufficient.

[DITL – root DNS per day latency; Main idea is that amortizing requests seen to the roots over large user populations makes latency implications small]

[Embedded in the above is a comparison between high latency and low latency roots with implications to user latency, so No, latency doesn't matter]

4 ANYCAST PERFORMANCE IN THE ROOT DNS

[anycast path inflation per RTT & per page load; Main idea is path inflation does become more prevalent with increasing deployment, but makes no difference to users][bar graph showing which sites are hit, corroborates the idea that inefficiency grows with deployment size, usually]

[use these results to suggest sites are added to root deployments for resilience, since there is no difference from a PPL perspective]

5 DOES ANYCAST CDN LATENCY MATTER?

[latency per RTT and per page load at various ring sizes; Main idea is RTT latency (and perceived latency) is significant from a PPL perspective, and ?depends on the application?][compared to root DNS per page load; idea is CDN latency is orders of magnitude more significant PPL, and latency PPL 10th percentile vs 90th percentile is large]

6 ANYCAST PERFORMANCE IN CDNS

[path inflation per RTT/page load, by ring; Main idea is that inflation becomes more prevalent, but latency PPL still goes down][inefficiency by ring; shows fewer users go to the closest site, but still latency PPL goes down] [geographic path inflation per RTT/page load, compared to the roots; CDN inflation < root inflation, argue that CDN works to control it via peering[case studies of intermetro variability, or unexpectedly poor performance, highlighting the intricacies (time permitting)]

7 RELATED WORK

IP anycast performance is usually studied in the context of two applications: the root DNS servers, and CDNs. In

[2] According to [http archive], RIPE Atlas, and the results from ISI itself, these are quite conservative estimates.

addition to these topics, we discuss studies of popular recursive resolvers, and user-centric measurements of web performance.

7.1 Root DNS Anycast Performance

The performance of anycast in the context of root DNS is generally gauged by anycast's ability to balance load among server replicas or provide low latency to users. Generally, all studies conclude that anycast successfully balances load, while latency performance depends on the specific deployment configuration. [22] looks at a DDoS attack on the root name server infrastructure, and generally shows that anycast is a good defense mechanism against such attacks. An earlier study, [24] confirms that anycast protects the root DNS infrastructure against such attacks and, furthermore, that anycast routes users to an optimal location in most cases. [12] looks at user latency to C, F, K, and L-root and attributes better performance to good geographic location and peering strategies. These findings coincide with an earlier study, [7], who conclude the performance of anycast is intrinsically linked to deployment strategy. Additionally [12] finds that as few as 12 sites can provide "good" latency to users. [19], [11], [12], and [20] are all examples of studies who quantify latencies to various root servers, and note how these compare to the (optimal) latency of the closest unicast alternative for the user who issued the query.

7.2 CDN Anycast Performance

Some CDNs (e.g. Cloudflare, Edgecast, Fastly) use IP anycast to augment their serving infrastructure. When deploying an Anycast CDN (ACDN), delivering content to users with low latency becomes a high priority, as there is a large financial incentive to do so. The simplicity of IP anycast comes at the cost of having coarse grained control over where user queries land. Shifting user load between nodes during peak hours, for example, is a challenging problem. As a potential solution, [3] and [14] use DNS redirects at ADNS servers to shift load among anycast nodes, albeit in slightly different ways. [8] analyzes what latency users are achieving, compared to optimal, when being routed to anycast nodes and finds that 10% of users experience a latency inflation of at least 100 ms.

7.3 Recursive Resolvers and the Benefits of Caching

Similar to the RR analysis conducted here, [16] looks at DNS traffic on a small network and notably finds that 16% of queries resulted in queries to the root, most of which were for invalid domains. As this study is quite old, it is no surprise that this rate has decreased (recall we observed .5% of queries resulted in queries to the root) since browser designers and network engineers understand the importance of caching. [9] also looks at a RR and analyzes statistics of DNS exchanges occurring over it including DNS transaction latencies. Both [27] and [18] look at certain pathological behaviors of popular recursive resolvers, and the implications these behaviors have on root DNS load.

7.4 Web Performance

Although we were unable to find any specific study that looked at how web performance and root DNS latency were related, there are certainly studies characterizing web performance. [25] characterizes web performance bottlenecks in (at the time) new broadband networks, and finds that latency is the main bottleneck for PLT when the user's bandwidth exceeds 16 Mbps. However, the study does not realistically emulate a page load and, in particular, can not analyze the effect of having multiple DNS resolutions per page. Similarly, [6] analyzes how each step of a page load contributes to the aggregate PLT using a tool designed in-house. However, unlike [25], they did not conduct a large measurement campaign and do not include information about multiple DNS lookups per page. A more recent study, [13] provides a brief survey of web performance measurement studies and explains why it is difficult (with current practices) to compare two different studies in web performance.

8 CONCLUSION

ideally we would control the linebreak in the latex so it occurs after "it," I'm not sure it is a contradiction. Aren't the people criticizing it well aware that it serves many users? I was trying to get across that it is odd for (for example) Microsoft to use anycast to serve latency sensitive content if it is known Anycast inflates latencies. Yes, I think your instinct of wanting to convey that is good, I was just suggesting thinking a bit more about the phrasing. I mean to change this to a macro, switching between "a large anycast CDN" and "Microsoft's anycast CDN".

Do you think I need to talk more about what sort of data I use to draw these conclusions in the abstract? You might consider a brief phrase like "Analyzing global traces gathered from a large anycast CDN and from all root DNS servers worldwide". Not necessary in the abstract, but can help. In the intro, you definitely want to talk a little about your data. It isn't clear what this is referring to, closer than what, etc Since these results and what we want to say about them are in flux, I will wait before clarifying. Will we also have an argument that it is less prevalent for CDNs? The analysis is in flux, and the data isn't perfect. This seems important but is buried in the middle of a paragraph I recommend not putting any placeholders in without markup. We should use markup like or that makes it colored, bold, and perhaps renders as something like [[[TODO: INSERT NUMBER]]]you use this phrase a lot in the abstract and intro, but I'm not sure it's clear what you mean +mcalder@microsoft.com Is this a valid description of Microsoft's anycast network, in the context that we leverage it? Please let me know if you think any major details are missing. perhaps a quick set of speed checker measurements this analysis is in flux might be an unfair comparison since the root DNS has many more nodes than Microsoft and more active debugging of bad routes, probably interesting that in 10 minutes they observe so many queries for COM TLD yet don't see any issue with that Might be an interesting tool to use Mark shared in an email – shows time between DNS queries & TCP connection starts can be big, which suggests DNS is not blocking

REFERENCES

[1] [n.d.]. root-servers.org

[2] [n.d.]. akamai.com/us/en/multimedia/documents/white-paper/akamai-designing-dns-for-availability-and-resilience-against-ddos-attacks.pdf

[3] Hussein A Alzoubi, Seungjoon Lee, Michael Rabinovich, Oliver Spatscheck, and Jacobus Van Der Merwe. 2011. A practical architecture for an anycast CDN. *ACM Transactions on the Web (TWEB)* 5, 4 (2011), 17.

[4] Amazon. [n.d.]. aws.amazon.com/route53/faqs/

[5] Amazon. [n.d.]. aws.amazon.com/cloudfront/

[6] Alemnew Sheferaw Asrese, Pasi Sarolahti, Magnus Boye, and Jorg Ott. 2016. WePR: a tool for automated web performance measurement. In *2016 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 1–6.

[7] Hitesh Ballani, Paul Francis, and Sylvia Ratnasamy. 2006. A measurement-based deployment proposal for IP anycast. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM, 231–244.

[8] Matt Calder, Ashley Flavel, Ethan Katz-Bassett, Ratul Mahajan, and Jitendra Padhye. 2015. Analyzing the Performance of an Anycast CDN. In *Proceedings of the 2015 Internet Measurement Conference*. ACM, 531–537.

[9] Thomas Callahan, Mark Allman, and Michael Rabinovich. 2013. On modern DNS behavior and properties. *ACM SIGCOMM Computer Communication Review* 43, 3 (2013), 7–15.

[10] Cloudflare. [n.d.]. Load Balancing without Load Balancers. blog.cloudflare.com/cloudflares-architecture-eliminating-single-p/

[11] Lorenzo Colitti, Erik Romijn, Henk Uijterwaal, and Andrei Robachevsky. 2006. Evaluating the effects of anycast on DNS root name servers. *RIPE document RIPE-393* 6 (2006).

[12] Ricardo de Oliveira Schmidt, John Heidemann, and Jan Harm Kuipers. 2017. Anycast Latency: How Many Sites Are Enough?. In *International Conference on Passive and Active Network Measurement*. Springer, 188–200.

[13] Theresa Enghardt, Thomas Zinner, and Anja Feldmann. 2019. Web performance pitfalls. In *International Conference on Passive and Active Network Measurement*. Springer, 286–303.

[14] Ashley Flavel, Pradeepkumar Mani, David Maltz, Nick Holt, Jie Liu, Yingying Chen, and Oleg Surmachev. 2015. Fastroute: A scalable load-aware anycast routing architecture for modern CDNs. In *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*. 381–394.

[15] Google. [n.d.]. developers.google.com/speed/public-dns

[16] Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. 2002. DNS performance and the effectiveness of caching. *IEEE/ACM Transactions on networking* 10, 5 (2002), 589–603.

[17] Dina Katabi and John Wroclawski. 2000. A framework for scalable global IP-anycast (GIA). *ACM SIGCOMM Computer Communication Review* 30, 4 (2000), 3–15.

[18] Matthew Lentz, Dave Levin, Jason Castonguay, Neil Spring, and Bobby Bhattacharjee. 2013. D-mystifying the D-root Address Change. In *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 57–62.

[19] Zhihao Li, Dave Levin, Neil Spring, and Bobby Bhattacharjee. 2018. Internet anycast: performance, problems, & potential.. In *SIGCOMM*. 59–73.

[20] Jinjin Liang, Jian Jiang, Haixin Duan, Kang Li, and Jianping Wu. 2013. Measuring query latency of top level DNS servers. In *International Conference on Passive and Active Network Measurement*. Springer, 145–154.

[21] Christopher Metz. 2002. IP anycast point-to-(any) point communication. *IEEE Internet Computing* 6, 2 (2002), 94–98.

[22] Giovane Moura, Ricardo de O Schmidt, John Heidemann, Wouter B de Vries, Moritz Muller, Lan Wei, and Cristian Hesselman. 2016. Anycast vs. DDoS: Evaluating the November 2015 root DNS event. In *Proceedings of the 2016 Internet Measurement Conference*. ACM, 255–270.

[23] Craig Partridge, Trevor Mendez, and Walter Milliken. [n.d.]. Host Anycasting Service. tools.ietf.org/html/rfc1546

[24] Sandeep Sarat, Vasileios Pappas, and Andreas Terzis. 2006. On the use of anycast in DNS. In *Proceedings of 15th International Conference on Computer Communications and Networks*. IEEE, 71–78.

[25] Srikanth Sundaresan, Nazanin Magharei, Nick Feamster, Renata Teixeira, and Sam Crawford. 2013. Web performance bottlenecks in broadband access networks. *ACM SIGMETRICS Performance Evaluation Review* 41, 1 (2013), 383–384.

[26] Verizon. [n.d.]. verizondigitalmedia.com/media-platform/delivery/network/

[27] Yingdi Yu, Duane Wessels, Matt Larson, and Lixia Zhang. 2012. Authority server selection in DNS caching resolvers. *ACM SIGCOMM Computer Communication Review* 42, 2 (2012), 80–86.