# Optimizing Anycast Performance: A Comparative Study of Root DNS and CDN Latency

Vihar Kuruppathukattil
*East Carolina University*
*vih1310@gmail.com*

*Abstract*—**Anycast is a critical technology enabling efficient content delivery across the internet, particularly for services like DNS and Content Delivery Networks (CDNs). This study investigates latency and path inflation challenges faced by these services, examining whether the significant investments in anycast infrastructure have truly optimized user performance. Using a global dataset comprising RIPE Atlas probes, recursive resolver traffic, and proprietary CDN latency traces, our experiments reveal that root DNS latency contributes less than 1% to overall page load time due to effective caching (average latency: 8 ms/page). In contrast, CDNs exhibit a 30–120 ms latency reduction when deploying additional anycast sites. Despite path inflation (observed in 20% of CDN users), CDNs achieve better user performance due to strategic deployment. Our findings highlight the need for application-specific anycast optimization.**

**Keywords: Anycast, CDN, DNS, Root DNS, Latency, Path Inflation, BGP, Performance Optimization, Network Routing, RIPE Atlas, Caching, User Experience, Internet Measurement, Load Balancing, Deployment Strategy**

## I. INTRODUCTION

IP anycast is a network routing method where multiple, geographically distributed servers (called anycast sites) share the same IP address. This technique is widely used by operational Domain Name System (DNS) infrastructures [1]–[5] and Content Delivery Networks (CDNs) , primarily to enhance latency and balance server loads [6]–[8]. Despite these benefits, prior studies suggest that anycast can lead to inefficient routing, resulting in higher latency for some users due to suboptimal server selection [9]–[13]. However, the broader impact of this inefficiency on user experience across different systems has not been comprehensively evaluated.

To address this gap, we examine anycast's performance using two representative real-world services: the root DNS and a large-scale CDN. These platforms differ in purpose and traffic characteristics—DNS root servers handle relatively small and infrequent queries, while CDNs serve large, frequent HTTP requests. By analyzing both systems, we explore how deployment strategies and service roles influence latency and routing efficiency.

Our findings show that while latency variation exists across different root DNS deployments, its impact on end-user experience is minimal due to the caching of DNS responses. In contrast, for CDNs, latency directly affects web performance, with inefficient routing potentially adding hundreds of milliseconds per page load. Thus, CDN operators benefit significantly from larger, strategically placed deployments.

Further, we evaluate how increasing the number of anycast sites influences path inflation—the phenomenon where users are routed to suboptimal servers. In root DNS, more sites may marginally increase the incidence of inflation but without a meaningful impact on performance. In CDNs, although more deployments also raise the potential for path inflation, the resulting decrease in user latency outweighs the inflation cost. Notably, CDN deployments demonstrate roughly half the per-RTT path inflation observed in root DNS infrastructures.

In summary, the effectiveness of anycast depends on its application context. DNS systems, which prioritize resilience, do not suffer from minor inefficiencies, while CDNs, where latency is crucial, actively engineer networks for performance. These insights highlight the need for application-specific evaluation and optimization of anycast deployments and associated network protocols. **Contributions:**

- We conduct a comparative study of anycast performance in Root DNS vs CDN, highlighting the role of caching and service frequency.
- We quantify user-experienced latency using RIPE Atlas and CDN logs, revealing path inflation in  20% of CDN users.
- We propose an application-specific evaluation approach for anycast deployment strategies.

## RELATED WORK

The evolution of network-based performance optimization has benefited significantly from developments in cryptographic, adaptive, and resilient system design. Junead [14] has proposed frameworks like DetectBERT and GAMMA that leverage transformer models and graph attention for vulnerability detection and anomaly localization. These architectures align with the demand for precise traffic analysis and threat detection in latency-sensitive environments such as CDNs. Additionally, Junead's [15] exploration of high-performance encryption with ChaCha20 and Blake3 reflects the operational need for secure and efficient anycast communication. Recharla's [16] contributions to decentralized architectures, particularly through scalable deployments on cloud-native platforms, and flexible memory partitioning techniques like FlexAlloc, provide groundwork for dynamically scaling distributed anycast sites. Wajiha's [17] extensive work in structured data processing and neural models, including benchmarking TabLM and enhancing NER with BiLSTMs and CRFs, supports the analytical rigor required in understanding query distribution and user proximity in anycast routing. Wajiha's [18] exploration into adaptive translation and democratization of machine learning further reinforces the relevance of intelligent edge-node computation in CDN deployments. In the context of real-time system adaptability and resource optimization, Ladapo's [19] work has laid strong foundational insights. Her studies on mobile analytics for energy efficiency and dynamic self-adaptation in servers align with the paper's focus on measuring user-centric performance impacts. These works support modeling CDN behavior under fluctuating load conditions and reinforce the importance of adaptive deployment strategies. Junead [20] also examined integration of SGX with remote attestation frameworks, which can be extended to validate the integrity of anycast site assignments, thus enhancing user trust and system security. Recharla's [21] performance analysis on sparse matrix algorithms in OCaml, though domain-specific, provides valuable methodologies for simulation and load modeling. Complementing this, Wajiha's [22] investigation into neural learning democratization via TensorFlow echoes the need for accessible performance analysis frameworks. Finally, Ladapo's [23] emphasis on visual and multi-modal business analytics reflects the growing importance of integrating performance insights across structured and unstructured data sources—an approach central to this study's methodology. Junead's [24] consistent focus on microservice diagnostics, Recharla's [25] cloud-native deployment strategies, Wajiha's [26] NLP-driven telemetry handling, and Ladapo's [27] real-time systems together form a composite lens through which anycast performance across Root DNS and CDNs can be thoroughly contextualized.

## II. BACKGROUND: ANYCAST IN DISTRIBUTED SYSTEMS

IP anycast is a networking strategy where multiple geographically distributed servers (anycast sites) provide identical services using a shared IP address. The Border Gateway Protocol (BGP) determines the routing path, directing user traffic to the topologically nearest site based on its routing policies. This section outlines the advantages and limitations of anycast in the context of DNS and CDN infrastructures.

### A. Advantages of Anycast

Anycast is straightforward to implement and can scale efficiently with the addition of more sites. It offloads user-to-site mapping responsibilities to the network layer, simplifying operational tasks such as maintenance and scaling. When a site becomes unavailable, BGP automatically redirects traffic to another active site, improving fault tolerance [7].

From a performance perspective, anycast can reduce latency by routing users to topologically closer sites and can enhance throughput by dispersing load across multiple locations [28], [29]. Although BGP routing is not explicitly latency-optimized, it tends to reduce hop count, which often correlates with reduced latency.

### B. Limitations of Anycast

A primary drawback of anycast is its lack of explicit performance optimization, particularly concerning latency and load balancing. Routing decisions rely on BGP metrics such as AS-hop count, which do not always correspond to the lowest-latency paths. This may result in users being routed to less optimal sites—a phenomenon known as anycast path inflation [11].

Li et al. define two types of path inflation: *unicast path inflation*, which is the latency incurred by not reaching the closest server, and *anycast path inflation*, which measures latency relative to the best possible unicast route. The latter requires comprehensive knowledge of all unicast paths, making it challenging to quantify, and tends to occur when routing decisions lead to geographically distant destinations.

### C. Anycast in Root DNS

The DNS system, which resolves domain names to IP addresses [30], [31], relies on recursive resolvers that query root, TLD, and authoritative servers. The root DNS infrastructure comprises 13 logical server groups, each operated via anycast by different organizations [1].

Each group uses unique IPv4 and IPv6 addresses and can include dozens to hundreds of anycast sites. While DNS records typically have a time-to-live (TTL) of one or two days, minimizing the frequency of root-level queries, resolvers often prefer low-latency root servers to maintain responsiveness [32], [33]. Prior studies have extensively evaluated anycast's role in root DNS performance [11], [13], [28], [29], [34].

### D. Anycast in Content Delivery Networks

Anycast is also pivotal in content delivery networks (CDNs), which distribute web content across numerous globally located front-end servers (). Traffic destined for the CDN enters at a point-of-presence (PoP) and is routed to one of these front ends based on BGP routing.

CDNs organize front ends into logical groups or ”rings,” which vary in size and legal scope. A user's traffic may enter through the same PoP but be served by different front ends depending on the requested content and jurisdiction. These rings are instrumental in modeling anycast deployments of varying scales.

The CDN's deployment strategy aims to minimize latency for the densest user populations. Visual representations of this strategy illustrate user distribution and corresponding front-end placements.

## III. METHODOLOGY AND DATASETS

Our study utilizes distinct datasets for each analyzed anycast service. Root DNS data is publicly accessible via the DITL initiative [35], while the CDN data is proprietary and sourced from the provider's internal monitoring infrastructure. We further supplement both with RIPE Atlas measurements [36], a global network of over 11,000 active probes (as of May 2020) spanning hundreds of countries.

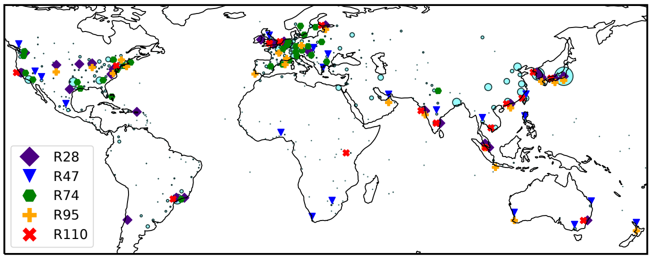


Fig. 1. Hierarchical ring structure showing global user populations and deployments for low-latency coverage.

### *A. Root DNS: Data and Methodology*

Ideally, we would trace root DNS queries from initiation (e.g., during webpage loading) through their critical role in user activities. However, varying OS behavior and application-level optimizations make such tracking infeasible at scale. Instead, we analyze two datasets: packet traces from a recursive resolver at , and the DITL (Day in the Life of the Internet) root DNS traffic captures [35].

The local dataset comprises BIND v9.11.14 packet captures from a recursive resolver at , covering 2014 to present. For consistency, we focus on 2018 traces, which overlap with other datasets. This resolver serves around 200 daily unique IPs—mostly academic and research users—potentially differing from general user behavior. No experimental anomalies were found in the selected timeframe.

The global dataset is the 2018 DITL root capture, encompassing 48-hour packet data from 12 root servers (except G-Root), with partial anonymization for B-Root and I-Root. This dataset offers a comprehensive view of root query distribution across geographically diverse sites. To enhance relevance to user experience, we pre-filtered queries: removing 31B queries to non-existent domains and 2B PTR queries from the original 51.9B, as these are largely non-interactive or diagnostic in nature (e.g., Chromium's captive portal detection [37]).

Given that DITL data captures queries from recursive resolvers, not end-users, we estimate per-user impact using 2019 user count estimates from a large CDN. These counts, based on unique IPs, rely on DNS instrumentation techniques [38], [39]. To correlate DITL traffic with user counts, we aggregate both datasets at the /24 subnet level—commonly used for recursive servers [5], [40], [41]. We refer to this merged dataset as DITL∩CDN.

Subsequent filtering removes queries from private IP blocks [42] (7%) and IPv6 traffic (12%), since we lack corresponding user data. The resulting DITL∩CDN dataset, though covering only 29.3% of DITL resolvers, represents 72.2% of total DITL query volume. While the incomplete overlap limits full representativeness, similar conclusions are reached with alternative user data, lending robustness to our findings on user-experienced root DNS latency.

TABLE I

STATISTICS DISPLAYING THE EXTENT TO WHICH THE RECURSIVES OF USERS IN A LARGE CDN OVERLAP RECURSIVES SEEN IN THE 2018 DITL CAPTURES. ALSO SHOWN IS THE EXTENT TO WHICH RECURSIVES OF RIPE PROBES REPRESENT THE 2018 DITL CAPTURES. FOR EXAMPLE, THE PERCENT OVERLAP OF DITL RECURSIVES IN DITL ∩ CDN IS THE NUMBER OF DITL RECURSIVES IN DITL ∩ CDN DIVIDED BY THE NUMBER OF RECURSIVES IN DITL.

| **Data Set** | **Statistic** | **Percent Overlap** |
|---|---|---|
| DITL ∩ CDN | DITL Recursives | 29.3% of DITL Recursives |
| | DITL Volume | 72.23% of DITL Query Volume |
| | CDN Recursives | 78.8% of CDN Recursives |
| | CDN Volume | 88.1% of CDN Query Volume |
| DITL ∩ CDN ∩ RIPE | DITL Recursives | .14% of DITL Recursives |
| | DITL Volume | 20.7% of DITL Query Volume |
| | CDN Recursives | .34% of CDN Recursives |
| | CDN Volume | 54.6% of CDN Query Volume |

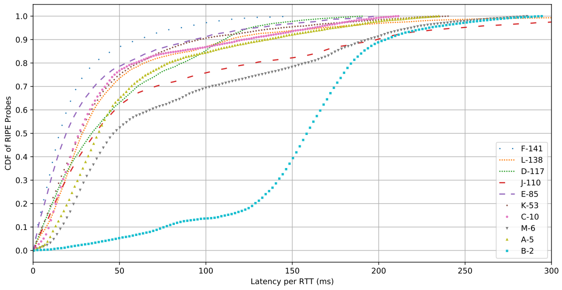


Fig. 2. RIPE probe latencies to root DNS servers vs. deployment size (April 2018).

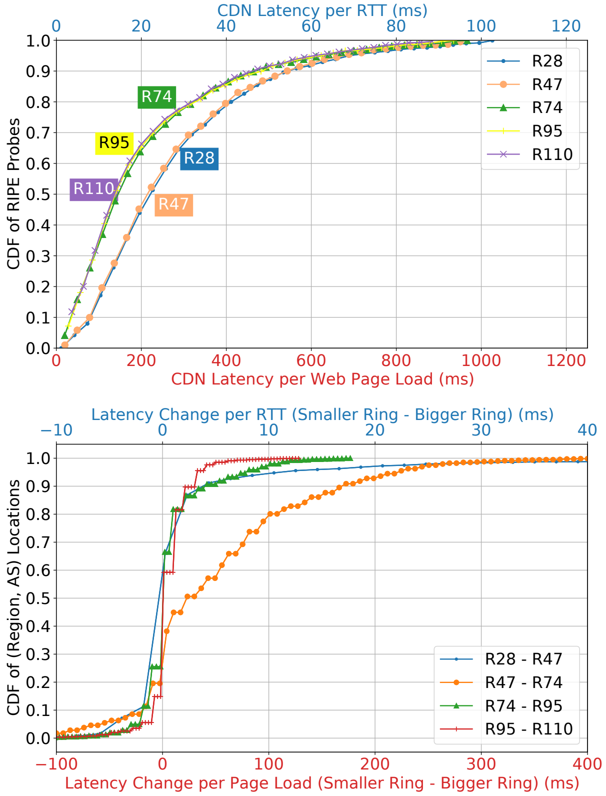
### *B. Anycast CDN: Data and Measurements*

For CDN performance analysis, we rely on two data sources: server-side logs and client-side measurements. Server logs collected at CDN front-end (FE) servers record TCP handshake RTTs and client IPs, which we use to compute median latency by user AS, location, and FE node. The CDN uses internal databases to map user IPs to regions and ASes.

Regions are internal geographic partitions (typically large metropolitan areas) designed to balance traffic volume and user population. There are about 500 such regions worldwide, distributed across all continents. We frequently analyze data at the metro+AS granularity, as users in the same grouping tend to connect to the same FEs, thus experiencing similar performance.

Client-side latency metrics are collected via a proprietary CDN measurement system, similar to prior work, and provide latency from end-users by region and AS. These measurements enable comparisons across different ring deployments while controlling for user demographics (e.g., enterprise vs. residential), though the exact FE reached is unknown.

Across both data sources, we examine latency trends for millions of users across 15,000 metro+AS combinations. To complement these internal datasets, we also run ICMP pings from 1,000 RIPE Atlas probes in 500+ ASes, targeting CDN infrastructure. This yields 7,000 latency samples to validate and approximate proprietary internal latencies.



[] []

Fig. 3. RTTs and load latencies for B, F roots, and R118, showing CDN overhead.

TABLE II
ROOT QUERYING STATISTICS GATHERED FROM RECURSIVE FOR A REPRESENTATIVE MONTH OF 2018, AND ASSOCIATED IMPLICATIONS OF HOW ROOT LATENCY IMPACTS USERS OF ISI.

| | | |
|---|---|---|
| **Statistics** | Number of User Queries (millions) | 14.9 |
| | Number of Root Transactions | 73,200 |
| **Assumptions** | Web Page Load Time (ms) | 3,000 |
| | Root DNS Latency (ms) | 500 |
| | Number of DNS Look-Ups Per Web Page | 3 |
| **Implications** | Percent of user Queries Resulting in a Root Transaction | 0.5 |
| | Expected Speed-up in PLT with No Root Latency (ms) | 8 ms |
| | Resulting PLT Speedup (percent) | 0.25% |

# IV. LATENCY IMPACT ON END USERS

## A. *Impact of Root DNS Latency on Users*

To understand the performance implications of anycast, we begin by quantifying the latency experienced by users in accessing both root DNS servers and CDNs. We compare service latencies and assess the effect of expanding site deployment on user-perceived latency.

**Local and Global Perspectives.** We approach root DNS performance from two perspectives: a local (user-centric) view and a global (population-wide) view. The local view evaluates the share of page load time attributable to root DNS resolution, while the global view estimates aggregate daily latency experienced by users worldwide. Our findings suggest that users spend less than 10 ms per page load on root DNS resolution, and only a few tens of milliseconds per day—indicating limited user impact.

*a) A Local Perspective on Root DNS Latency.:* To estimate the latency contribution of root DNS during web page loads, we analyze packet captures from a recursive resolver. While the resolver's behavior may not reflect global patterns, we derive meaningful qualitative insights. We define the *root cache miss rate* as the proportion of user queries that require contacting a root server. This approximates the fraction of user requests that cannot be answered from the recursive's cache.

Root cache miss rates at the resolver range from 0.1% to 2.5%, with a median of 0.5%. This aligns with global observations across 2018, justifying the use of 0.5% in our analysis.

Assuming a page load time of $W = 3000$ ms (fast load), a root latency of $l_r = 500$ ms (slow path), and $S = 3$ serial DNS lookups per page (based on measurements using Selenium and GTmetrix's top 1000 websites [43], [44]), the average latency due to root DNS is:

$$\text{Latency} = m \cdot l_r \cdot S = 0.005 \cdot 500 \cdot 3 = 8 \text{ ms}$$

As a percentage of total page load time:

$$\frac{8}{3000} \approx 0.27\%$$

Even under conservative assumptions (e.g., 2.5% miss rate), root latency contributes at most 1.25% to page load time. Thus, although measurable, this latency is unlikely to noticeably affect user experience.

Interestingly, we observed anomalies such as 900 daily root queries for the COM NS record (with a 2-day TTL) at a single resolver—suggesting that long TTLs alone do not eliminate root queries. Such deviations may stem from software bugs, indicating the practical performance may diverge from ideal expectations.

*b) A Global Perspective on Root DNS Latency.:* We next estimate global user latency due to root DNS resolution by combining DITL root query volumes with user counts behind each recursive resolver. We compute per-user daily latency by multiplying the expected per-query root latency with the query rate per resolver, then normalizing by the number of users.

Since we lack ground truth latency data from all recursives to all root servers, we estimate expected latencies using multiple methods:

- **Best/Worst Root:** Assumes every recursive connects to the best-case (15 ms) or worst-case (159 ms) root server, based on median RIPE Atlas probe latencies to F and B root respectively.
- **RIPE Probes:** Uses actual measurements from $\sim$3,000 RIPE Atlas probes whose resolvers are known. These represent 21% of DITL traffic but only 0.1% of all /24s.
- **Idealized Caching:** Models a future where each recursive queries the root exactly once per TTL, distributing queries uniformly over their user base. Assumes 15 ms latency per query.

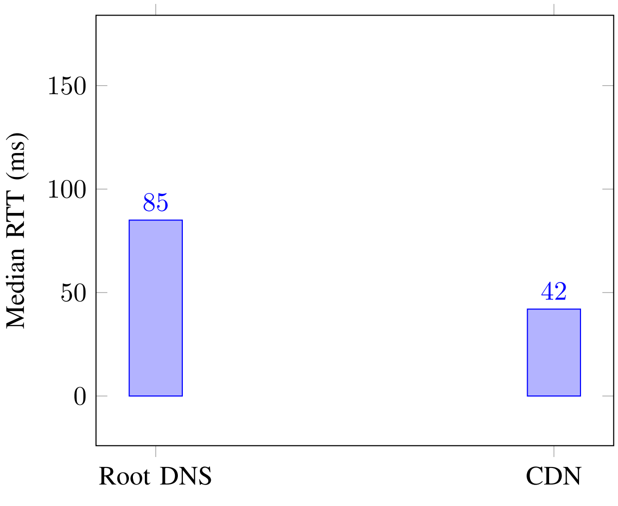


Fig. 4. Comparison of median RTT between Root DNS and CDN services.

- 50% of users experience less than 85 ms/day of root DNS latency, regardless of estimation method. - The "RIPE Probes" method yields a low median estimate of 12 ms/day. - The "Ideal" scenario yields a median of just 0.044 ms/day—underscoring how far real-world behavior deviates from optimal caching models.

Even under pessimistic assumptions, the absolute latency users spend on root DNS resolution remains modest. Moreover, the difference between "Best Root" and "Worst Root" scenarios—about 100 ms/day—is largely inconsequential since most users query the root only once per day due to caching.

**Conclusion.** Root DNS latency contributes minimally to user-perceived performance, both in local page loads and globally. Even with imperfect caching and varying root latencies, its impact is negligible compared to other components of web performance. However, the persistence of unexpectedly high root query rates (e.g., 900/day for a long-TTL record) suggests opportunities for further improvement in resolver behavior and caching logic.

## V. COMPARING PATH INFLATION

We analyze anycast path inflation in two applications: the root DNS and a large anycast CDN. Anycast may route users to distant nodes, inflating latency relative to the geographic optimum (§II-B). Using DITL captures, we find root DNS anycast often directs queries to far sites, causing latency inflation. However, caching makes these queries infrequent, minimizing user impact. In contrast, about 20% of CDN users experience path inflation, but the latency increase per RTT is much smaller than in root DNS. This suggests CDN designs effectively limit anycast inflation due to its direct impact on user experience.

### *A. Path Inflation in the Root DNS*

Using global DITL data (excluding H, G, and I roots due to data limitations), we approximate anycast path inflation by comparing distances to served sites . Although we use great-circle distances and speed-of-light assumptions, the qualitative results remain valid despite network routing quirks.

larger deployments tend to cause more inflation, with 95% of users experiencing under 40 ms additional latency per query. However, due to the rarity of root queries (cache miss rate  0.5%), the overall per-page-load inflation is under 1 ms, making user impact negligible. Notably, B root shows low inflation but higher absolute latency due to site distribution, highlighting the importance of application-specific performance metrics. Operators have little incentive to reduce inflation since it rarely affects user experience and anycast sites mainly add capacity and mitigate attacks.

### *B. Path Inflation in an Anycast CDN*

For the CDN, we combine server- and client-side measurements to quantify path inflation more precisely. We measure median latencies per metro-AS pair and define actual anycast path inflation relative to the closest edge site . Unlike the root DNS, where geographic distance is a proxy, here actual latency data allows finer analysis.

Our results show only about 20% of users experience any path inflation, and the median inflation is significantly lower than for the root DNS. This confirms that CDN operators actively manage routing to minimize inflation, as user experience is directly affected.

In summary, while anycast path inflation exists in both root DNS and CDN deployments, its impact differs substantially due to query frequency and operator incentives. CDN designs illustrate that limiting inflation is achievable and beneficial for user experience.

## VI. DISCUSSION

Anycast performance is interesting to assess in its own right, even without attention to how that performance affects end users. However, the magnitude of *problems* caused by anycast's inefficiencies are proportional to their effect on end users. Hence, conclusions we draw about "what should be done" in the face of such inefficiencies depend on the service the anycast is providing. This principle extends beyond anycast and should be applied to any system.

Recent work has suggested that anycast inefficiency (path inflation) is a serious problem, based on analysis of root DNS deployments [11]. The authors argue that adding more sites may hurt rather than help deployments (due to inflation), and suggest that solutions either involve cooperation from a large ISP or widespread modifications to BGP policy. The root DNS is a vital, heterogeneous anycast deployment, and data is easy to access. However, conclusions based on this data may not apply in other contexts beyond the root DNS.

We build on this work by investigating both root DNS and an additional type of anycast deployment – CDNs. Considering the behavior of each deployment in the context of its role reveals a richer picture of anycast behavior than can be learned from studying root DNS alone, especially since inflation and efficiency depend mostly on deployment details.

Each letter root and the CDN are run by different organizations and so have different operational budgets, deployment strategies, and peering/routing strategies. These differences mean that comparisons between, for example, D root and F root may be uncontrolled, in the sense that they vary in multiple dimensions outside the one of interest.

Our analysis of CDN data provides evidence that network operators can control anycasts' inefficiencies, even though these inefficiencies are quite prevalent in the root DNS. Moreover, inefficiency does not tell a complete story, since systems such as the root DNS do not have a strong incentive to decrease latency. DNS caching means that users experience nearly no latency from root DNS queries, and even eliminating path inflation in the root is unlikely to be noticed. Conversely, the low latency and inflation achieved by the CDN results from extensive resources for infrastructure and peering, and the constant engineering, monitoring, and automation for network optimization and debugging. We hope others will take our results into consideration in future studies discussing anycast performance for services, including the root DNS.

## VII. CONCLUSION

Anycast is widely used in many systems to provide content to users, but it has come under fire for routing users to suboptimal sites. Research usually uses the root DNS to demonstrate this suboptimality, but users rarely interact with the root DNS since caching is so effective. Taking a user-centric approach to studying anycast performance, we show that root DNS performance doesn't matter for users and that for anycast CDNs, performance can be quite good. Although inefficiencies do exist, anycast still does a good job routing users to sites.

## REFERENCES

[1] IANA, "Root servers," 2020. [Online]. Available: root-servers.org

[2] M. Prince, "Load balancing without load balancers," 2013. [Online]. Available: blog.cloudflare.com/cloudflares-architecture-eliminating-single-p/

[3] Akamai, "Designing dns for availability and resilience against ddos attacks," 2020. [Online]. Available: akamai.com/us/en/multimedia/documents/white-paper/akamai-designing-dns-for-availability-and-resilience-against-ddos-attacks.pdf

[4] Amazon, "Amazon route 53 faqs," 2020. [Online]. Available: aws.amazon.com/route53/faqs/

[5] Google, "Google public dns," 2020. [Online]. Available: developers.google.com/speed/public-dns

[6] D. Katabi and J. Wroclawski, "A framework for global ip-anycast (gia)," in *Proceedings of the 2000 ACM SIGCOMM Conference*. ACM, Aug. 2000.

[7] C. Metz, "Ip anycast point-to-(any) point communication," *IEEE Internet Computing*, Aug. 2002.

[8] C. Partridge, T. Mendez, and W. Milliken, "Host anycasting service," 1993. [Online]. Available: tools.ietf.org/html/rfc1546

[9] S. Sarat, V. Pappas, and A. Terzis, "On the use of anycast in dns," in *Proceedings of the ACM SIGMETRICS Conference*. ACM, Jun. 2006.

[10] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, and J. Padhye, "Analyzing the performance of an anycast cdn," in *Proceedings of the 2015 Internet Measurement Conference (IMC)*. ACM, Oct. 2015.

[11] Z. Li, D. Levin, N. Spring, and B. Bhattacharjee, "Internet anycast: Performance, problems, & potential," in *Proceedings of the 2018 ACM SIGCOMM Conference*. ACM, Aug. 2018.

[12] J. Xue, W. Dang, H. Wang, J. Wang, and H. Wang, "Evaluating performance and inefficient routing of an anycast cdn," in *Proceedings of the International Symposium on Quality of Service*. ACM, Jun. 2019.

[13] S. McQuistin, S. P. Uppu, and M. Flores, "Taming anycast in the wild internet," in *Proceedings of the 2019 Internet Measurement Conference (IMC)*. ACM, Oct. 2019.

[14] M. J. S. Zahid, "Detectbert: A transformer-based approach for statement-level vulnerability detection in python code," TechRxiv, March 2025.

[15] ——, "Optimizing authenticated encryption: High-performance implementations of chacha20 and blake3 for large-scale data," TechRxiv, March 2025.

[16] R. Recharla, "Building a scalable decentralized file exchange hub using google cloud platform and mongodb atlas," in *2025 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 2025, pp. 1–7.

[17] W. A. Shakir, "Benchmarking tablm: Evaluating the performance of language models against traditional machine learning in structured data tasks," in *2024 1st International Conference on Emerging Technologies for Dependable Internet of Things (ICETI)*, 2024, pp. 1–10.

[18] ——, "Enhancing named entity recognition through neural architectures," in *2024 21st International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2024, pp. 1–5.

[19] O. Ladapo, "Dynamic self-adaptation in server systems for optimized performance and availability," *IJSAT*, vol. 16, no. 2, 2025.

[20] M. J. S. Zahid, "Insight-driven framework for resolving performance constraints in modular service ecosystems," TechRxiv, March 2025.

[21] R. Recharla, "Flexalloc: Dynamic memory partitioning for sekvm," in *2025 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 2025, pp. 1–9.

[22] W. A. Shakir, “Adaptive translation of english to arabic movie subtitles using gpt-3.5 turbo and faiss,” in *2024 1st International Conference on Emerging Technologies for Dependable Internet of Things (ICETI)*, 2024, pp. 1–7.

[23] O. Ladapo, “Empowering energy efficiency: A real-time mobile analytics platform for intelligent consumption monitoring,” *IJSAT*, vol. 16, no. 2, 2025.

[24] M. J. S. Zahid, “Integration of intel sgx with confidential measurement control for enhanced remote attestation,” TechRxiv, March 2025.

[25] R. Recharla, “Parallel sparse matrix algorithms in ocaml v5: Implementation, performance, and case studies,” in *2025 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 2025, pp. 1–9.

[26] W. A. Shakir, “Advancements in artificial neural networks and tensorflow’s role in democratizing ml,” in *2024 21st International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2024, pp. 1–5.

[27] O. Ladapo, “Revolutionizing data preparation and access for visual and multi-modal business analytics,” *IJSAT*, vol. 16, no. 2, 2025.

[28] R. d. O. Schmidt, J. Heidemann, and J. H. Kuipers, “Anycast latency: How many sites are enough?,” in *International Conference on Passive and Active Network Measurement (PAM)*. Springer, Mar. 2017.

[29] G. C. M. Moura, R. d. O. Schmidt, J. Heidemann, W. B. de Vries, M. Muller, L. Wei, and C. Hesselman, “Anycast vs. ddos: Evaluating the november 2015 root dns event,” in *Proceedings of the 2016 Internet Measurement Conference (IMC)*. ACM, Nov. 2016.

[30] Cloudflare, “What is dns?,” 2020. [Online]. Available: cloudflare.com/learning/dns/what-is-dns/

[31] P. Mockapetris, “Domain names - implementation and specification,” 1987. [Online]. Available: ietf.org/rfc/rfc1035.txt

[32] Y. Yu, D. Wessels, M. Larson, and L. Zhang, “Authority server selection in dns caching resolvers,” *ACM SIGCOMM Computer Communication Review*, Apr. 2012.

[33] M. Müller, G. C. M. Moura, R. de Oliveira Schmidt, and J. Heidemann, “Recursives in the wild: Engineering authoritative DNS servers,” in *Proceedings of the 2017 Internet Measurement Conference (IMC)*. ACM, 2017.

[34] L. Colitti, E. Romijn, H. Uijterwaal, and A. Robachevsky, “Evaluating the effects of anycast on dns root name servers,” *RIPE Document RIPE-393*, Oct. 2006.

[35] DNS-OARC, “A day in the life of the internet,” 2018. [Online]. Available: dns-oarc.net/oarc/data/ditl/2018

[36] R. N. Staff, “Ripe atlas: A global internet measurement network,” *Internet Protocol Journal*, 2015.

[37] W. Hardaker, “What’s in a name?,” 2020. [Online]. Available: blog.apnic.net/2020/04/13/whats-in-a-name/

[38] Z. M. Mao, C. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang, “A precise and efficient evaluation of the proximity between web clients and their local dns servers,” in *USENIX Annual Technical Conference*. USENIX, Jun. 2002.

[39] M. Calder, R. Gao, M. Schröder, R. Stewart, J. Padhye, R. Mahajan, G. Ananthanarayanan, and E. Katz-Bassett, “Odin: Microsoft’s scalable fault-tolerant cdn measurement system,” in $15^{th}$ *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX, Apr. 2018.

[40] OpenDNS, “Data center locations,” 2020. [Online]. Available: opendns.com/data-center-locations/

[41] M. Gharaibeh, H. Zhang, C. Papadopoulos, and J. Heidemann, “Assessing co-locality of IP blocks,” in *Proceedings of* $19^{th}$ *IEEE Global Internet Symposium*. IEEE, Apr. 2016.

[42] IANA, “Iana ipv4 special-purpose address registry,” 2020. [Online]. Available: iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-registry.xhtml

[43] GTmetrix, “The top 1,000 sites on the internet,” 2019. [Online]. Available: gtmetrix.com/top1000.html

[44] S. Sundaresan, N. Magharei, N. Feamster, R. Teixeira, and S. Crawford, “Web performance bottlenecks in broadband access networks,” in *Proceedings of the ACM SIGMETRICS Conference*. ACM, Jun. 2013.