Comparing DNS Resolvers in the Wild

Bernhard Ager
T-Labs/TU Berlin
bernhard@net.t-labs.tu-berlin.de

Wolfgang Mühlbauer
ETH Zurich
muehlbauer@tik.ee.ethz.ch

Georgios Smaragdakis
T-Labs/TU Berlin
georgios@net.t-labs.tu-berlin.de

Steve Uhlig
T-Labs/TU Berlin
steve@net.t-labs.tu-berlin.de

ABSTRACT

The Domain Name System (DNS) is a fundamental building block of the Internet. Today, the performance of more and more applications depend not only on the responsiveness of DNS, but also the exact answer returned by the queried DNS resolver, e. g., for Content Distribution Networks (CDN).

In this paper, we compare local DNS resolvers against GoogleDNS and OpenDNS for a large set of vantage points. Our end-host measurements inside 50 commercial ISPs reveal that two aspects have a significant impact on responsiveness: (1) the latency to the DNS resolver, (2) the content of the DNS cache when the query is issued. We also observe significant diversity, even at the AS-level, among the answers provided by the studied DNS resolvers. We attribute this diversity to the location-awareness of CDNs as well as to the location of DNS resolvers that breaks the assumption made by CDNs about the vicinity of the end-user and its DNS resolver. Our findings pinpoint limitations within the DNS deployment of some ISPs, as well as the way third-party DNS resolvers bias DNS replies.

Categories and Subject Descriptors

C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*Internet*

General Terms

Measurement, Performance

Keywords

DNS resolvers, measurement, performance analysis

1. INTRODUCTION

The Domain Name System (DNS) was originally intended to provide a naming service, i. e., one-to-one mappings between a domain name and an IP address. Since then, the popular applications have changed from static content hosting to distributed and dynamic content delivery. As a consequence, DNS is a highly scalable system that fulfills the needs of applications that often have very strong requirements in terms of responsiveness of DNS [9, 10, 12]. The scalability of the DNS system stems from the heavy use of caching by DNS resolvers [8].

Today, the DNS system has become a commodity infrastructure that allows applications to map individual users to specific content. This behavior clearly diverges from the original purpose of deploying DNS, and is sometimes considered as abusing it [17]. Given the importance of DNS for end-user experience and how much the DNS system has changed over the last decade, in this paper we study DNS deployment in commercial ISPs and compare it with widely used third-party DNS resolvers, GoogleDNS and OpenDNS.

Based on active measurements carried out across more than 50 commercial ISPs, spanning 5 continents and 28 countries around the world, we study the responsiveness and the returned IP addresses by the local DNS resolvers as well as GoogleDNS and OpenDNS. Our results show that a surprisingly high number of commercial ISPs suffer from poor latency to the local DNS resolver. In general, our results do not reveal drastic differences between the local DNS resolvers, GoogleDNS, and OpenDNS, in terms of responsiveness. Several ISPs show clear signs of DNS load balancing, that leads to a poor usage of DNS caching.

Our findings also reveal that third-party DNS resolvers do not manage to redirect the users towards content available within the ISP, contrary to the local DNS ones. We conjecture and partly validate that the reason for this behavior of third-party DNS resolvers has to do with their location, typically outside ISPs, in combination with current inability of DNS resolvers to indicate the original IP address of the end-host in the DNS requests [5]. The current advantage of local DNS resolvers is their ability to represent the end-user in terms of geographic location and its vicinity to content.

In spite of the importance of DNS, we are not aware of any work that has performed such an extensive study of the DNS system based on measurements from end-hosts, and compared local DNS resolvers against third-party DNS resolvers.

The remainder of this paper is structured as follows: we start with an overview of the DNS system (Section 2) and describe our data set and how it was collected in Section 3. Then Section 4 analyzes our results before we conclude in Section 5.

IMC'10, November 1–3, 2010, Melbourne, Australia.

2. DOMAIN NAME SYSTEM

In this section we give a brief overview of the design of the Domain Name System (DNS) and how the different DNS components interact. We also take a look at how DNS is being used today.

2.1 DNS Primer

DNS relies on a distributed database with a hierarchical structure. The root zone of the DNS system is centrally administered and serves its *zone* information via a collection of *root servers*. The root servers delegate responsibility for specific parts (zones) of the hierarchy to other *name servers*, which may in turn delegate the responsibility to other name servers. At the end, each site is responsible for its own *domain* and maintains its own database containing its information and operates an *authoritative* name server.

The whole database is usually queried by end-hosts using a local name server called *caching resolver*. If this name server receives a query for information that it does not have, it must fetch this information from another name server. If the server does not know how to contact the authoritative server for a zone, it will query a root server[1]. The root server will *refer* the resolver to another server that is authoritative for the domain that is immediately below the root and of which the zone is a part. The resolver will then query this server, and so forth, stepping down the tree from the root to the desired zone.

For efficiency reasons DNS relies heavily on caching [8]. All information that a name server delivers to a resolver is cached for a duration specified in the TTL field of the *resource records* (RR). Caching today is usually also performed on end-hosts by the operating system's *stub resolver*, as well as applications, e. g., web browsers.

2.2 DNS Today

When DNS was introduced in 1983, its sole purpose was to resolve host names into IP addresses in a more scalable fashion than the until then used `hosts` file. Since then a number of features and new uses have found their way into the omnipresent DNS. In addition to the increasing complexity within the DNS protocol itself [16], new and oftentimes unforeseen (ab)uses have been established. Paul Vixie gives an overview in [17]. The most important points of critique are as follows:

CDN load balancing: Content delivery networks set short TTLs on their DNS answers to allow for short reaction times to load shifts, thus crippling cacheability and scalability of the whole system. In addition, CDNs tailor their reply for the IP address of the requesting resolver using the often misguided assumption that the DNS resolver is close to the client originating the request.

NXDOMAIN catcher: Some ISPs and also OpenDNS mangle a negative reply with the NXDOMAIN status code into a positive one with the IP address of a search website under the control of the ISP. By hosting advertisements along the search results it is easily possible to increase the profit margin. While this may work to some degree for web browsing, applications relying on proper delivery of NXDOMAIN records, e. g., email, are inevitably hampered.

A third-party ecosystem around DNS has evolved over the last couple of years. Players like OpenDNS, AdvantageDNS, UltraDNS, and most recently Google offer open resolvers to anyone with different feature sets. OpenDNS Basic does NXDOMAIN catching but offers phishing and botnet protection for free. Furthermore, OpenDNS increases the service level for payment between 5 dollars a month up to several thousand dollars per year for business customers. When Google Public DNS entered the market, their highest-valued goals were to "speed up your browsing experience" and to "improve your security." To achieve both targets Google advertises an impressive list of optimizations and fine tuning [2], e. g., prefetching, load balancing with shared cache, validity checking, and nonce prepending. Google Public DNS also refrains from utilizing NXDOMAIN to make profit. From an implementation perspective, most if not all of the third-party resolvers host their DNS servers on multiple sites around the globe and use anycast to guide DNS clients to the nearest resolver.

In this open market space a user annoyed by his ISP's DNS can easily choose for cost-free third-party service. Tools like namebench [3] might help him in choosing a well-performing one. The irony however is that a user by choosing a different DNS than the one assigned by his ISP will most likely undermine the traffic matrix optimizations performed by CDNs and ISPs, and can potentially even lower his quality of experience due to longer download times.

3. MEASUREMENTS

Data collection on end-hosts is intrinsically difficult due to lack of willingness to participate or due to privacy concerns [7]. Yet, for this paper we are interested in the DNS performance as perceived by end-users and therefore make efforts to collect data directly from users' computers. This section describes our data and how it was collected.

To achieve our goal of comparing the responsiveness of various DNS resolvers, we wrote code that performs DNS queries for more than 10,000 hosts. Amongst other tasks, the code measures DNS response times and returned TTLs for all queried hosts, relying on different DNS resolvers. We asked friends to run our code in early April 2010, leading to traces from more than 60 vantage points, covering 28 countries and 5 continents. Overall, we have obtained traces for some 50 commercial ISPs. During our measurements, the following information was collected:

1. **Vantage point:** Our code initially determines the public IP address and operating system of the executing machine as well as a current time stamp and the IP address of the local DNS resolver.

2. **Resolver:** Periodically, we determine the RTT[2] towards the local, Google, and OpenDNS resolver and perform traceroutes towards these three resolvers. This reveals potential route changes and the proximity of DNS resolvers to our vantage points.

[1]The first query can go to some authoritative server below the root if there exists cached information.

[2]We rely on the response time reported by `dig` when querying for the root zone NS records, rather than using `ping` or `traceroute`.

3. **Host:** For each of our approximately 10,000 target host names we perform, using the `dig` program, two consecutive DNS queries and measure the response times. Comparing response times between the first and second query gives insights into caching and load balancing, see Section 4.2. Besides response times, we record the returned TTL values and the IP addresses of the DNS responses.

Presumably, the majority of Internet users rely on DNS services that are provided by their ISP. This local DNS resolver is automatically configured during the dial-in handshake or via DHCP (*Local DNS*). Yet, alternative DNS providers claim to speed up the browsing experience (GoogleDNS) [2], and some users think that DNS queries can be processed much more quickly by employing a large cache of domain names (OpenDNS) [4]. To check for potential differences in performance, our code sends the same DNS queries to multiple DNS servers: the locally configured DNS server, to 8.8.8.8, and to 208.67.222.222, whereby the latter two are the DNS IP addresses used by Google and OpenDNS,[3] respectively.

In order to improve its efficiency, DNS heavily relies on caching, i. e., storing DNS query results for a period of time in DNS cache servers provided by ISPs or in home routers that implement DNS caches. As we seek to investigate potential bias in DNS response times for different types of queried hosts (e. g., popular vs. rarely queried hosts), we download from Alexa the list of top 1,000,000 sites [1], and select more than 10,000 hosts to be queried by our code as follows:

top5000: These are the 5,000 most popular hosts according to the Alexa ranking. The answer to many of these DNS queries may already be stored at close-by cache servers. We point out that the top5000 hosts are selected based on a global ranking, and hence are not necessarily the most popular hosts if ranked by country, region, etc.

tail2000: These are 2,000 hosts from the tail of the Alexa ranking and are less likely to be cached in close-by DNS servers.

embedded: Many web-pages include embedded content (e. g., AVI, Flash, or Java) that the browser may have to retrieve separately from different domains. We take the top 1,000 hosts according to the Alexa ranking, download with `wget` the content of all these hosts and compile a list of domains from which such embedded content is retrieved. By doing so, we obtain some 3,500 host domains.

Restricting to 10,000 hosts allows our measurements to finish within a couple of hours, which turned out to be acceptable to our end-users. Resolving the names of our 10,000 hosts reveals that a significant number of them (709) rely on DNS redirection. The set **redirected** contains all host names for which we see a CNAME record to an external domain (such as a CDN). The set **akamaized** is a subset of **redirected**, containing the 434 hosts that are redirected to Akamai. Information about redirection will be used for the CDN study in Section 4.3.

In principle, there can be interactions between two different vantage points in our experiments if the script is run close in time and based on the same list of hosts: for example, OpenDNS or GoogleDNS can cache the answer when vantage point A sends a query. When another vantage point B sends the same query, the response time will be significantly shorter if the reply is already in the cache. However, by inspecting timestamps in our traces and the DNS servers' approximate locations as revealed by traceroute and the RTT, we can infer whether interactions may have happened. The traces presented here are carefully selected and do not show any degree of interaction.

4. EVALUATION OF DNS RESOLVERS

In this section we rely on the measurements explained in the previous section, and analyze the behavior of the different DNS resolvers, with respect to responsiveness (Section 4.1), DNS deployment (Section 4.2) and the returned answers (Section 4.3).

4.1 Responsiveness

Google claims on its website that Google Public DNS speeds up browsing performance [2]. One primary goal of this paper is to understand the impact of the selected DNS resolver on the observed DNS response time. Is it really true that alternative DNS resolvers such as GoogleDNS or OpenDNS offer better performance than the local resolver?

To answer this question, it is crucial to rely on measurements that are carried out directly from end-hosts connected to commercial ISPs, see Section 3. If the DNS deployment within the local ISP is properly done, we would expect very small latencies to the resolvers maintained by the local DNS. Yet, we find cases where GoogleDNS and OpenDNS outperform the local DNS resolver in terms of observed response times.

We select two vantage points that are representative for many other traces. The first is located in Germany and we qualify it as "good ISP". The second is based in the US and we call it "bad ISP". It is not our goal to assess individual ISPs. Rather these terms reflect that the "good ISP" shows better DNS performance in terms of response times, load balancing, caching, etc., compared to the "bad ISP", see Section 4.2. Figure 1 and Figure 2 display the CCDF of the response times, in milliseconds, observed at these two vantage points. The leftmost part of the curves on these two figures shows the minimal latency that has been achieved by the DNS resolvers across all 10,000 queries. This minimal latency can be seen as a metric to characterize the proximity of a DNS resolver to the actual end-host. Figure 1 shows a case where the smallest latency differs significantly between the local ISP, OpenDNS, and GoogleDNS at 11ms, 24ms and 44ms. Although both GoogleDNS and OpenDNS maintain a large set of strategically placed resolvers and rely on anycast to route DNS queries, their latencies are far higher than those of the local resolver. The local resolver appears to be close to the end-host. This underlines the importance of placing a resolver in the proximity of end-hosts.

Surprisingly, there are cases where we observe that GoogleDNS or OpenDNS perform as well if not better than the local ISP resolver, see Figure 2. For our "bad ISP" the network distance towards OpenDNS appears to be especially

[3]Other DNS IP addresses such as 8.8.4.4 for GoogleDNS are generally configured as secondary DNS server.

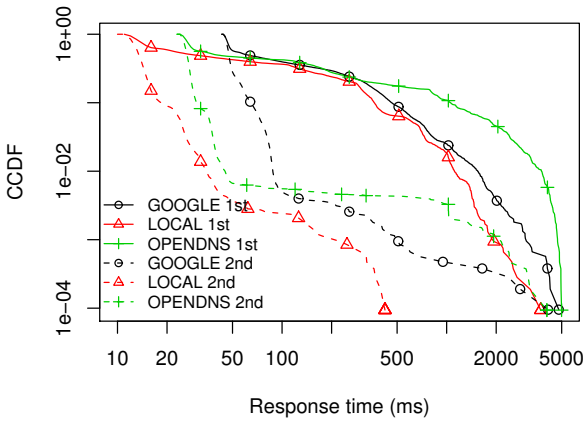

Figure 1: CCDF of response times for "good ISP". The local resolver has significantly lower RTTs than both GoogleDNS and OpenDNS.

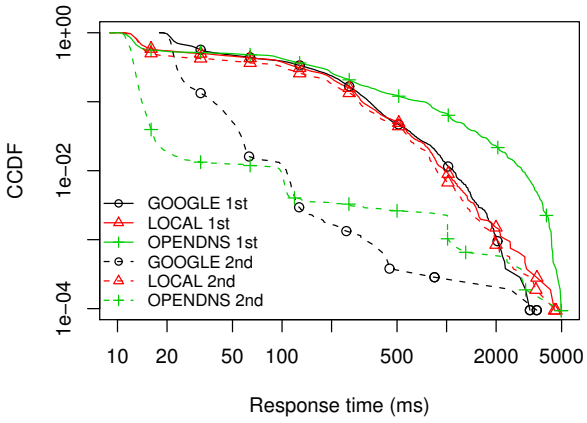

Figure 2: CCDF of response times for "bad ISP". The time for the second query does not improve much.

small. Indeed, the RTT towards OpenDNS is only 10ms while it is 11ms towards the local DNS. In total, we observe 17 vantage points where either GoogleDNS or OpenDNS have in the worst case the same latency as the local DNS. On 21 vantage points, the local DNS is at least 25ms faster than the other two third-party resolvers; for the remaining 29 vantage points the local DNS only marginally outperforms GoogleDNS and OpenDNS.

In addition to deploying resolvers in the proximity of end-hosts, another key aspect to achieve good DNS performance is efficient caching. With respect to caching, Google aims to increase the number of cache hits through load balancing DNS resolvers that avoid cache fragmentation or by actively prefetching names [2]. When the curves for the first query are close to vertical, this shows that the caches are primed. Based on our plots, the three resolvers do not seem to be so well primed. While GoogleDNS performs significantly better on the tail of the curve than OpenDNS for the traces shown, this does not hold in general. Based on our measurements, we can neither confirm nor refute any gains obtained from techniques such as name prefetching or load balancing for shared caching as Google or OpenDNS may use.

To study caching behavior, our measurements always perform two consecutive DNS queries for the same name. Comparing the curves in Figure 1 for the first and second DNS query, we observe considerably faster response times for the second query due to caching of the DNS answers by the resolvers. The differences in the latencies to the resolvers become then even more obvious. Typically over 95% of the second queries are being answered within 100ms.

Overall, the barrier to achieve lower DNS response times seems to be the distance to the local resolver, once the DNS resolver cache is properly populated. GoogleDNS for instance does not seem to achieve their 50ms objective [10] for most of the queries in the case of the "good ISP" (Figure 1). There are still less than 1% of the cases for which all resolvers take more than 100ms to answer, due to non-cacheable records, one-time errors, and measurement artifacts.[4]

4.2 DNS Deployment

The observation from the previous section, that most of the second queries can be answered from the cache, does not hold for the local DNS of the "bad ISP" (Figure 2). In this section, we dig further into the results from Section 4.1, by showing the results from the first query against those of the second query on a scatter plot (see Figure 3 and 4). The x-axis of Figure 3 and 4 show the response time in milliseconds for the first query, while the y-axis the response time in milliseconds for the second query.

In Figure 3, we observe one horizontal line per DNS resolver for the "good ISP", meaning that the response times for the second queries show only small variation and are consistently better than those for the first query. An ISP that has a properly deployed DNS infrastructure should show this kind of pattern. However, several of our vantage points display a behavior like the "bad ISP" (Figure 4), where points are scattered along a horizontal and vertical line, as well as the diagonal. We explain this behavior by a load balancing setup without a shared cache. Sections with a sharp decline in the CCDF for the second query (Figure 1) correspond to the horizontal patterns in Figure 3: the first query primed the cache and the second query could be served from that cache. The diagonal in Figure 4 stems from hostnames for which both queries needed an iterative resolving because the second query was redirected to a different resolver. Finally, the vertical line springs from host names for which the first query could be served from the cache, while the second query was directed to a different resolver where the cache was not primed. Our conjecture is supported by consistent observations in which a significant proportion of the TTLs for the first and the second query differ considerably.

Several ISPs for which we have multiple traces display this behavior in a consistent way. Furthermore, we see this behavior for both OpenDNS and GoogleDNS in several traces. We conjecture that OpenDNS and GoogleDNS also use load balancing for highly loaded sites.

For some ISPs, we observe high RTTs towards the local DNS *and* load balancing. We conjecture that in these cases the DNS infrastructure is centralized and requires load bal-

[4]Typically, the second latency is considerably higher than the first one for at most 10 out of 10,000 hosts. Potential reasons may include fluctuations in routing, links resets, server reconfiguration, etc.

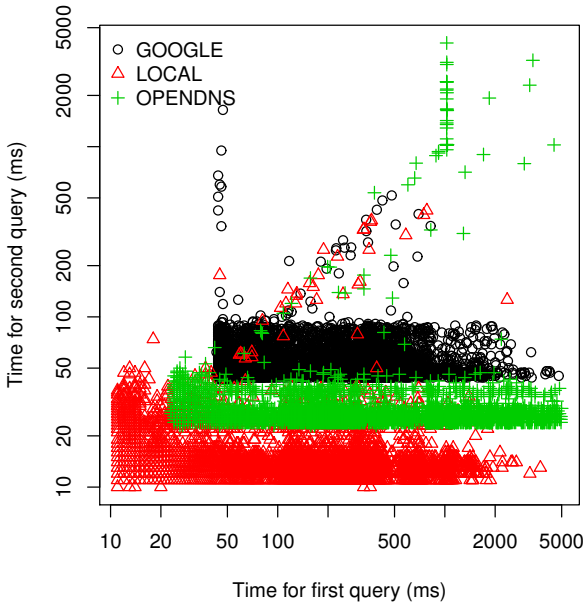

Figure 3: "Good ISP": as expected most of the second queries can be answered significantly faster than the first query due to caching.

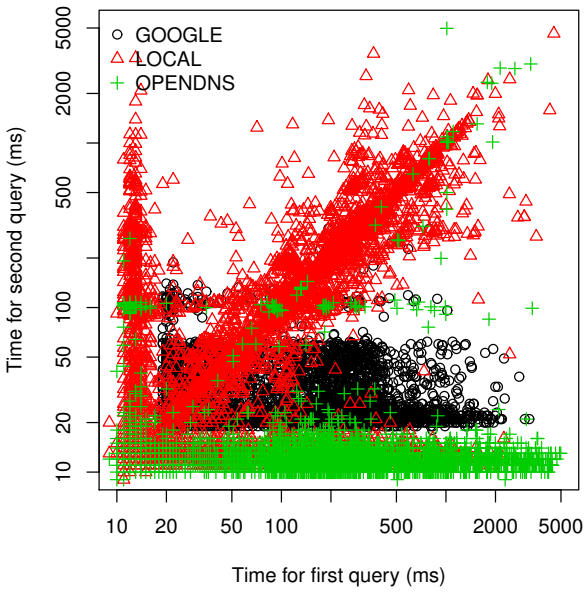

Figure 4: "Bad ISP": the ISP balances load over different DNS resolvers, so the second query cannot always be answered from the cache. A strong diagonal and a vertical line emerge.

ancing to compensate for the high number of queries arriving at a single location.

Although there are valid reasons to rely on DNS load balancing, the way some ISPs are implementing it prevents caching from being properly utilized. A hierarchical DNS infrastructure could improve hit rates while preserving distribution of load to different machines.

4.3 DNS Answers

For good end-user experience, fast DNS response times are important. While this aspect has already been studied in Section 4.1, we now investigate the exact answers that DNS returns, i. e., the resolved IP addresses. Frequently, there is more than one available option from where content can be retrieved. Possible reasons include both the use of load balancing and content distribution networks (CDNs) [9, 15]. The latter replicate content and provide copies near the client to optimize network performance [12]. The goal of this section is to study the observed diversity in resolved DNS names (i. e., IP addresses) across different vantage points and different DNS resolvers. In particular, we examine potential interferences between the choice of the DNS resolver and measurements done by CDNs. After all, CDNs such as Akamai determine which IP to return in the DNS response to the client based on measurements done against the DNS resolver, not against the client. Choosing a DNS resolver that is far from the end-host might vitiate the performance optimizations made by CDNs.

As expected, there is indeed diversity in the IP addresses returned by DNS. While we perform DNS queries for 10,000 unique host names in our experiments, the overall number of unique resolved IP addresses across all traces is 36,000. One reason is that we always perform two DNS queries for the same host name and then even repeat this for three different DNS resolvers. Apart from load balancing, this can be due to CDN content. When repeating queries or when changing between DNS resolvers, the resolved DNS names may be different depending on the mechanisms CDNs use to optimize network performance for the clients.

In Section 4.1 we find that the local DNS resolvers generally provide lower latencies due to their proximity to the end-hosts. Hence, one may speculate that they better represent the location of end-hosts than other resolvers. If possible, it might be more desirable for performance or economic reasons that CDNs be queried by a DNS resolver that is located within the local ISP of the end-host. Figure 5 shows for each vantage point of our study (x-axis), how many IP addresses that belong to the same ISP as the host of the vantage point, were returned by each DNS resolver, across all queried content.

We find that the majority of DNS answers point to content outside the vantage point's network. GoogleDNS and OpenDNS even return IP addresses from different networks for all our traces. One of the reasons is that these resolvers are usually not located inside the ISP. Yet, for approximately 30 vantage points we observe that content is downloaded from at least 100 hosts located within the ISP's network when the local DNS resolver is queried. There even exist vantage points where local access occurs for 926 of our 10,000 host names.

Although this may not appear much, it is significant if we consider that this locally available content completely covers the `akamaized` set[5] (see Section 3). We harvest IP addresses of Akamai servers by sending DNS queries to Akamai content from different Planetlab servers. Interestingly, we find based on manual inspection that the vantage points with local content generally have an Akamai server deployed within the same network.

[5]Additionally, there is strong overlap with `top5000` and `embedded`.

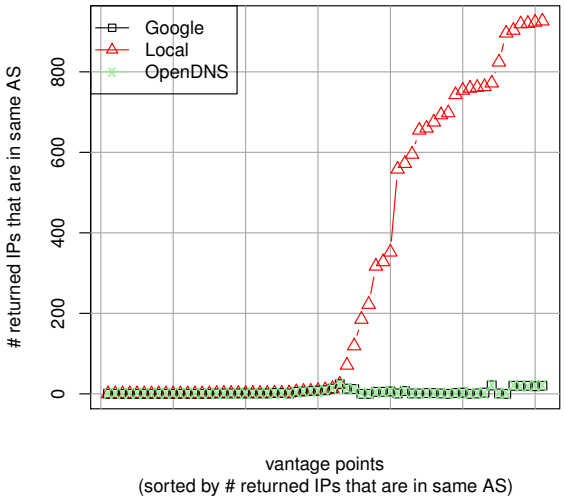

Figure 5: How often do DNS answers point to locally available content?

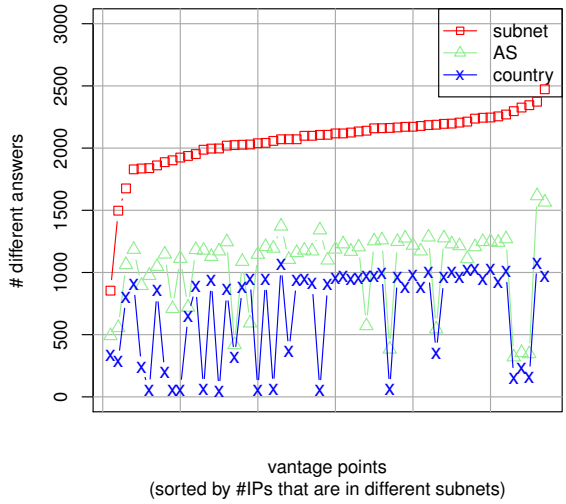

Figure 6: How often do the DNS answers of local resolver and GoogleDNS differ?

From Figure 5 we can infer that only local DNS resolvers direct end-users to content that is locally available in the network of the vantage point. Instead of matching the results of the resolvers against the network of the vantage point, we now directly compare our three DNS resolvers pairwise against each other. More precisely, we count how often the results of the two resolvers are different with respect to the subnet, autonomous system (AS), and country[6] of the DNS answer. Figure 6 presents for each vantage point of our study (x-axis) the number of differences for the comparison local DNS vs. GoogleDNS.

Figure 6 reveals that the answers to the DNS resolvers differ in terms of subnets for approximately 2,000 out of our 10,000 host names. In half of these cases, the returned IP addresses even belong to different ASs and countries. Since the local DNS resolver points to content inside the ISP's network for a significant number of host names (Figure 5), we claim that GoogleDNS and OpenDNS unnecessarily direct end-users to content servers in different ASs or even subnets. Due to space limitations we do not present the plots for the comparisons between Local vs. OpenDNS and Google vs. OpenDNS. Both plots are very similar to Figure 5. Apparently, whenever we change the DNS resolver, there will be different answers from DNS for at least some of the host names. This observation justifies recent activities of the IETF in the direction of standardizing a way to include the IP address of the original end-host in DNS requests [5].

5. CONCLUSION

Based on active measurements from inside more than 50 commercial ISPs, we have studied DNS performance by comparing the ISPs' DNS deployment against widely used third-party DNS resolvers, namely GoogleDNS and OpenDNS.

Typically, end-hosts experience very small latencies to the resolvers maintained by the local ISP, though there exist cases where GoogleDNS and OpenDNS outperform the local DNS resolvers in terms of the observed response times. Moreover, our findings suggest that several ISPs and OpenDNS rely on a load balancing setup without a shared cache, resulting in poor caching efficiency. Even Google Public DNS, despite their claim [2] exhibits the same behavior for a few vantage points. Moreover, we observe that third-party DNS resolvers do not manage to redirect the users towards content available within the ISP, contrary to the local DNS ones. This observation holds for all akamaized content.

Given the increasing share of CDN traffic [11, 13] we aim in the future to fully understand to what degree the choice of DNS resolvers does vitiate the performance optimizations made by CDNs. In this regard, we plan to rerun our experiments based on an enlarged set of vantage points and with an enhanced version of our script, e. g., to scrutinize caching and investigate the role of anycast in DNS performance.

[6]We used geolocation data to map IP addresses to countries [6].

Acknowledgments

The authors wish to thank all volunteers who ran our script on their systems and submitted the results. Moreover, we would like to thank our shepherd, Craig Partridge, and the anonymous reviewers for their constructive comments.

6. REFERENCES

[1] Alexa top sites. http:///www.alexa.com/topsites.
[2] Google Public DNS. http://code.google.com/intl/de-DE/speed/public-dns/.
[3] Namebench—open-source dns benchmark utility. http://code.google.com/p/namebench/.
[4] OpenDNS: What's Your Take? http://www.neowin.net/news/opendns-whats-your-take.
[5] C. Contavalli, W. van der Gaast, S. Leach, and D. Rodden. Client IP information in DNS requests. IETF draft, work in progress, draft-vandergaast-edns-client-ip-00.txt, Jan 2010.

[6] Hexasoft Development Sdn. Bhd. IP Address Geolocation to Identify Website Visitor's Geographical Location. `http://www.ip2location.com`.

[7] D. Joumblatt, R. Teixeira, J. Chandrashekar, and N. Taft. Perspectives on Tracing End-hosts: A Survey Summary. *Computer Communication Review*, 40(2):51–55, 2010.

[8] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. *IEEE/ACM Trans. Netw.*, 10(5):589–603, 2002.

[9] B. Krishnamurthy, C. Wills, and Y. Zhang. On the Use and Performance of Content Distribution Networks. In *Proc. of ACM IMW '01*, San Francisco, CA, USA.

[10] R. Krishnan, H. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. Moving Beyond End-to-end Path Information to Optimize CDN Performance. In *Proc. of ACM IMC '09*, Chicago, IL, USA.

[11] C. Labovitz, D. McPherson, and S. Iekel-Johnson. Internet observatory report. http://www.nanog.org/meetings/nanog47/.

[12] T. Leighton. Improving Performance on the Internet. *Commun. ACM*, 52(2):44–51, 2009.

[13] Ingmar Poese, Benjamin Frank, Bernhard Ager, Georgios Smaragdakis, and Anja Feldmann. Improving Content Delivery using Provider-aided Distance Information. In *Proc. of ACM IMC '10*, Melbourne, Australia.

[14] A. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante. Drafting Behind Akamai: Inferring Network Conditions Based on CDN Redirections. *IEEE/ACM Trans. Netw.*, 17(6):1752–1765, 2009.

[15] S. Triukose, Z. Al-Qudah, and M. Rabinovich. Content Delivery Networks: Protection or Threat? In *Proc. of ESORICS '09*, Saint-Malo, France.

[16] P. Vixie. DNS Complexity. *ACM Queue*, 5(3):24–29, 2007.

[17] P. Vixie. What DNS is Not. *Commun. ACM*, 52(12):43–47, 2009.