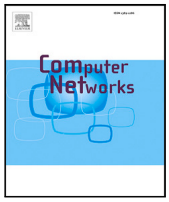
Computer Networks 212 (2022) 109046

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Measuring DoH with web ads

Patricia Callejo [a], Marcelo Bagnulo [a,*], Jaime González Ruiz [a], Andra Lutu [b], Alberto García-Martínez [a], Rubén Cuevas [a]

[a] Universidad Carlos III de Madrid, Spain
[b] Telefónica, Spain

ARTICLE INFO

Keywords:
DNS
Security
DoH
Measurements

ABSTRACT

In this paper we present a large measurement study of the impact on the performance of the adoption of HTTPS as a transport for the DNS protocol (DoH) with public resolvers compared to the existent approach of using non-encrypted transport of DNS queries with the resolver services locally provided by ISPs. Using on web-ads as the mean to execute our tests, we perform over 42 million measurements from more than 4 million vantage points distributed in 32 countries and served by over 2,500 ISPs. We find that, the median resolution time increased 17 ms when using DoH with Cloudflare, 41 ms when using DoH with Quad9, 68 ms when using DoH with Google and 170 ms when using DoH with DNS.SB, compared to using Do53 with the local resolver for a non-cached name. We find similar increases even when using caching. The results presented in the paper contribute to the ongoing discussion of the tradeoffs involved in the combined adoption of public resolvers and DoH.

1. Introduction

The Domain Name System (DNS) is part of the Internet's critical infrastructure. Most communications over the Internet are preceded by a domain name resolution through DNS. While it was originally conceived without any security (it was a different time and a different Internet), over the years a number of security mechanisms have been built into the DNS, notably, DNSSEC [1] to provide integrity protection for the content of the DNS queries. The latest addition to the DNS security measures is to leverage on the secure protocol HTTPS as a transport for DNS queries [2], called DoH (DNS over HTTPS) for short. This is primarily envisioned to protect the communication between the client and its resolver. By using DoH, the communication between the client and the resolver obtains a number of security features, including confidentiality and integrity protection for the information exchanged between both parties.

We make two (trivial) observations regarding DoH deployment: First, while DoH protects against third parties that wish to eavesdrop the contents for the communication, the resolver still has full visibility of the client's queries and the responses to those. Second, in order to enable DoH, both the client and the resolver must support DoH. These observations are particularly relevant in the context of the *deployment* of DoH.

In early 2020, Mozilla announced [3] that Firefox would use DoH with Cloudflare's resolver by default for its US-based users,[1] changing not only the DNS transport to DoH, but also the resolver used, overriding the resolver selection of the host/user. Probably many factors contributed to the design of this deployment strategy. Clearly, by using a DoH-enabled resolver such as the one Cloudflare provides, the adoption of DoH is expedited, since users do not need to wait for their local resolver to support DoH to enjoy its benefits. Other motivations may also include a wish to protect users against DNS information harvesting from their traditional local resolver providers (i.e., the ISPs [4]). Be this as it may, the result is that for Firefox users in the USA, the default behavior is not only to adopt DoH, but also to use an alternative resolver. If this DoH roll-out strategy proves successful, others may follow, resulting in different apps using different resolvers for the same clients, none of these honoring the local resolver selection made by the host/operating system.

Before changing the default behavior to adopt DoH with Cloudflare's resolver, Mozilla recruited 25,000 volunteers to experimentally measure the impact on performance of the combined change of DNS transport and resolver. The experiment measured over a billion of DoH transactions and resulted in a (rather succinct) blogpost [5], that reported that for most queries (about 80%) the adoption of Cloudflare/DoH adds a penalty of 6 ms when querying for a non-cached

* Corresponding author.
E-mail addresses: pcallejo@inst.uc3m.es (P. Callejo), marcelo@it.uc3m.es (M. Bagnulo), andra.lutu@telefonica.com (A. Lutu), alberto@it.uc3m.es (A. García-Martínez), rcuevas@it.uc3m.es (R. Cuevas).

[1] Users may opt-out by manually configuring the browser.

https://doi.org/10.1016/j.comnet.2022.109046
Received 10 December 2021; Received in revised form 8 April 2022; Accepted 11 May 2022
Available online 21 May 2022

query, while the remaining 20% queries were faster (sometimes hundreds of ms faster). Mozilla's report was not peer reviewed and the resulting dataset was not made public for the community to validate their findings. Reproducing Mozilla's measurements is challenging without their capability to rapidly roll-out the tests through a browser update, and reach to thousands of volunteers out of their user base. Thus, we argue that there is value in having a third party perform a large scale study of the subject and verify their results. This is one of the main motivations for the work we present in this paper. Moreover, extending the measurements to other public resolvers, other browsers/applications and other geographical regions (as we show that we have done) provides a broader understanding of the impact of the DoH roll-out strategy Mozilla adopted, and that can inform other stakeholders how to move forward regarding DoH.

In this paper, we perform a large-scale measurement study on the combined effect of adopting both DoH and a public resolver on the DNS performance end-users experience. More specifically, we compare the DNS resolution delay experienced by clients when querying four different public resolvers using DoH against the delay they undergo when they use their default resolver using non-encrypted UDP-based DNS transport (called *Do53* for short). We rely on web advertisements as a vehicle for our measurements in order to obtain a large number of measurement vantage points. By using this approach, we managed to recruit over 4 million end-users who executed our measurements, resulting in over 42 million DNS queries.

The main contributions of the paper are:

- We showcase the breadth of the methodology for measuring the combined effect of changing the resolver and the DNS transport from millions of vantage points.
- We contrast the results Mozilla reported from their pilot study [5]. Specifically, our vast measurement campaign yields a somehow larger penalty compared to theirs when using Cloudflare's public resolver through DoH.
- We extend the analysis to other public resolvers (e.g., Google, Quad9, DNS.SB) and other browsers (notably Chrome and Edge). We find that the impact on performance of the combined change of DNS transport (from Do53 to DoH) and resolver (from local ISP provider to public provider) greatly depends on the provider of the public resolver. While Quad9/DoH performs similarly to Cloudflare/DoH, the other two (Google/DoH and DNS.SB/DoH), significantly under-perform when compared to the local resolver/Do53 combination (both in terms of the median value and the 99th percentile value of the resolution delay).
- We find that the performance of local resolvers also varies greatly, and that there is a small set of local resolvers that exhibits poor performance. In other words, the vast majority of local resolvers – when accessed through Do53 – outperform the public resolver through DoH.

2. Measurement methodology

We use an online advertisement network [6] to perform our measurements from millions of vantage points [7,8]. We insert lightweight JavaScript code into advertisements (hereafter ads), such that when an impression of these custom ads prints in a client webpage, the inserted JavaScript code executes, and the configured tests are performed (see Section 3 for test details). With this technique we can run experiments from the end-user perspective, and the results are representative of the actual user experience. This methodology allows us to perform millions of measurements in a short period of time.

We program the ad to perform DNS queries using different resolvers and different transports, and we measure the DNS resolution delay for each query. Regarding transports, we make queries using both DoH and Do53, and, regarding resolvers, we use the default resolver configured in the client resolver (which in most cases is provided by the ISP to its subscribers) and four public resolvers, namely, CloudFlare, Google, Quad9 and DNS.SB. We chose these resolvers because they are public resolvers that support DoH and, more importantly, they support the JSON API for DoH[9]. Moreover, while the first three resolvers have a large global footprint, the latter is less spread. Thus, we expect to also observe the impact of the resolvers' global presence.

In each client, we query the default resolver using the default transport used by the client, and we also query the four public resolvers using DoH, in order to compare their performance. The names we use in the queries are under the *doh-serv.com* domain, for which we control the authoritative server. This allows us to capture the traffic that the authoritative server receives as result of the queries the ads generate. We also setup a Control Server that collects the results the clients viewing the ad report.

Through the ad, we are able to trigger a DNS query to the default resolver configured in the client by simply making an HTTPS request to a domain name. We use information about the Operating System and the User Agent (typically a browser) to infer whether the client performed the query using DoH, Do53 or else. By inspecting the source IP address of the query issued by the resolver to our authoritative server, we can determine if the default resolver is a public one or a local/private resolver. We further describe the details about our inference of resolver type (public resolver or otherwise) and the DNS transport in Section 4.1.

In terms of the DoH queries, we integrate specific code in the ad that performs queries to arbitrary name servers supporting DoH using their JSON API.[2] In this case, no inference is needed because we explicitly set the transport (DoH) and the resolver.[3]

We use the Performance Resource Timing API [10] available in most commonly used browsers and user agents to obtain detailed information about different metrics related to the DNS queries (e.g., the resolution delay).

2.1. Measurement process

We run the experiments inserting our custom JavaScript code in an ad platform. Once its loaded, we configure the experiments in the ad platform (see Section 2.3 for further details), which delivers the ad to the end-users. Then the measurement process starts. We depict the overall measurement process in Fig. 1, and we summarize its main steps as follows:

(1) When a client (*UserA* and *UserB* in the figure) opens a website/app showing the ad, it executes the embedded tests upon printing.
(2) The tests trigger a number of DNS queries to the DNS resolver, which in turn issue queries to the authoritative DNS server. Both the ad running in the client and our DNS authoritative server collect the results of the tests.
(3) The ad in the client reports the results back to our control server, using WebSockets.

[2] We cannot use the GET/POST method for DoH queries defined in [2], because it results in a Cross-Origin Resource Sharing (CORS) error when executed from within the ad, due to the lack of the necessary CORS headers.,doh-json.

[3] Using this approach, we make the client to use DoH to query the selected resolvers, but we have no control over the transport used by the resolver to communicate with the authoritative server and/or other intermediate elements.

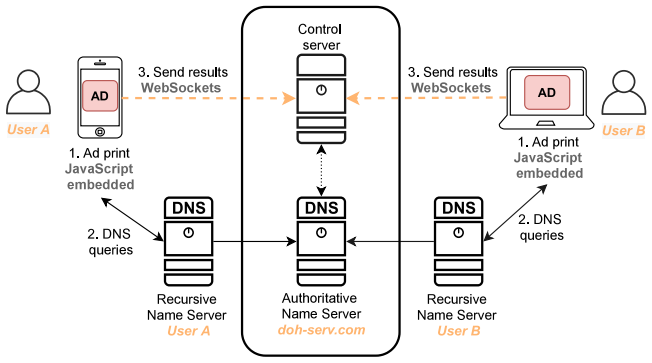


Fig. 1. Methodology overview.

2.2. Validation of measurement approach

In order to validate the proposed measurement approach and quantify the accuracy of the obtained measurements, we performed a number of tests in the lab. Specifically, we simulated the ad using the Puppeteer API, which is a headless Chromium browser and performed a 100 DNS queries. For each query, we obtained the DNS resolution delays using both the Performance API and from the packet traces (using Wireshark). We then compared the delay obtained from both sources. We find that the difference between the delays measured using the two methods are always smaller than 0.5 ms. We also observe that the values obtained from the Performance API is always larger than the correspondent value obtained using the packet traces, meaning than when we compare the delays between different resolvers, the errors will partially cancel each other.

We also measure how the load in the CPU affects the measurement results. We applied both a 3x and 5x CPU throttling to the experimental setup during the control test. We find that even with a throttling of 5x, the difference between the results obtained from both sources is smaller than 0.9 ms.

2.3. Experimental design considerations

We use an ad provider [6] to distribute the ads and print them in clients' browsers. The ad provider's platform offers a set of knobs that we (as any advertiser) can use to tune the measurements and select the target clients. The ad provider we chose allows us to select clients based on the operating system they use and also by geographic location, at country granularity. The geographic coverage of the measurements is also limited by the footprint of the ad provider. In the case of the ad provider we use, its footprint covers America, Europe and a few countries of Africa.

Though using ads allows us to perform measurements from millions of vantage points, we must take into account several practical and ethical considerations when designing the measurements.

The duration of the measurements done in each client is limited by the ad exposure time. This time is unknown beforehand (as it depends on the user browsing behavior), and it varies for each ad impression. According to previous similar studies [7], the mean ad lifetime varies around approximately 30 s; thus, we design our measurements accordingly. In particular, measurements run concurrently whenever it is possible. We validate that concurrency does not affect measurements results by doing local tests, simulating different loads in the measurement client.

Moreover, since the ad exposure time is variable, we expect that even if the time required to do the measurements is below the mean ad lifetime, in some cases, the measurements will fail to complete. When processing the results, we filter out the partial results that are unusable.

Table 1
Queries the ad executes to capture the DNS resolution delay for non-cached domain names. `idAd` is a string that is unique for each ad impression (i.e., one execution of all the queries).

DNS Query
idAd.doh-serv.com (to the default resolver)
x+idAd.doh-serv.com (to the default resolver)
dns.google/resolve?name = g+idAd.doh-serv.com&type = A
dns.google/resolve?name = gg+idAd.doh-serv.com&type = A
cloudflare-dns.com/dns-query?name = c+idAd.doh-serv.com&type = A
cloudflare-dns.com/dns-query?name = cc+idAd.doh-serv.com&type = A
dns.quad9.net:5053/dns-query?name = q+idAd.doh-serv.com&type = A
dns.quad9.net:5053/dns-query?name = qq+idAd.doh-serv.com&type = A
doh.dns.sb/dns-query?name = d+idAd.doh-serv.com&type = A
doh.dns.sb/dns-query?name = dd+idAd.doh-serv.com&type = A

In the design of the experiments, we also took into account the impact for the client of running the tests, both in terms of resource consumption and in terms of privacy. We describe these considerations in depth in Section 9.

Performing the measurements using the proposed methodology has an economic cost, since we need to pay for the ad impressions. However, the costs is fairly low. The total costs for all the measurements presented in this paper is below $200.

3. Description of tests

In this section, we provide a description of the experiments we perform using our ads-based measurements methodology. We design specific tests to collect insights regarding the performance of different resolvers and DNS transports (namely, Do53 and DoH). The source code for all the tests is freely available at: https://github.com/Jaimegruiz/TFMDOH.

3.1. Resolution delay for a non-cached domain name

Our goal with this experiment is to measure the delay to perform the resolution of a name that is not present in the resolver cache, using the different resolvers and the different transports we selected. This is relevant because non-cached DNS queries significantly affect web performance, accounting for up to 13% of the critical path delay for page load times [11].

We measure the resolution time using the four selected public resolvers using DoH. We also measure the DNS resolution time using the default resolver configured in the client and the default transport the client uses with this resolver. We then process the obtained results to only preserve the measurements regarding queries performed to local resolvers using Do53 (see Section 4.1 for details).

We resolve domain names under a domain in our control (the `DoH-serv.com` domain) that are specifically crafted for this experiments. To use non-cached names in our queries, the domain names we query for have the format `prefix.DoH-serv.com`, with "prefix" being unique for each different query. This approach ensures that the full name being queried is not present in any cache. However, it is still possible that the records for the TLD and/or for the `DoH-serv.com` domain are present in some of the caches and not in others, which may affect our measurements. However, the effect of this would be negligible given the sheer size of our measurement campaign (we perform millions of queries for a few thousands resolvers), as only the first query for each resolver is affected at most.

The use of unique names also allow us to match the queries received at the authoritative DNS server under our control and the results reported by the clients. We list in Table 1 the different queries we run for this experiment. The authoritative server for this domain is located in Madrid, Spain.

For the DoH experiments, we measure the resolution time for the first query per resolver (which includes the delay introduced by the TLS handshake) and the resolution time for a subsequent query, which does not include the handshake delay.[4] We argue that, while both metrics are relevant, the second one is closer to daily operations, since it is likely that in most cases the client will have a persistent TLS connection with its public resolver.

As we show in Table 1, for each resolver, we perform two unique queries.[5] We perform all the first queries to the different resolvers concurrently, and we perform the second query to each resolver upon the reception of the response to the first query. We verified that making several concurrent queries does not impact the measured resolution time.

For each test, we collect measurement data records at the authoritative server, and from the client directly, which reports to our control server through the ad. We are able to map the records obtained from the ad and the ones collected at the authoritative server thanks to the unique identifier we embed in the domain name the client resolves upon printing the ad. For each ad impression, the client report includes the identifier assigned to this ad impression (*idAd* in Table 1), the resolution delays for each query performed, the user agent (including the OS version), and the access technology (if reported by the client). We retrieve the IP prefix of the client from the source address of the IP packet. From each query the authoritative server receives, we collect the IP address of the resolver and the EDNS0 Client Subnet (ECS) flag.[6]

3.2. Measuring the impact of caching in the resolution delay

We design a set of tests to capture the impact of caching in the resolution delay. There are several levels of caches involved in the DNS [13]. There is a DNS cache in the resolver, and there is also a local DNS cache in the end-host itself. If the queried domain name is present in the local cache (at the end-host), the resolution delay is not affected by the selection of transport nor the resolver used.[7] This case is not interesting for our study. Our goal is to measure the effect of the resolver cache in the resolution delay. The difficulty is, then, to find a name that is in the resolver cache, and not in the local cache.[8] Querying for a name in the top positions of the list of popular domain names would most likely result in names that are frequently present in both caches, not serving our purpose. Leveraging the large number of measurement vantage points that we can reach thanks to our methodology, we propose to rely on the measurement clients themselves to populate the resolver caches with the domain name we specifically craft for these experiments (namely, *test.DoH-serv.com*, under our control). Thus, even if the first query for our (unpopular) domain name will be non-cached, the clients that follow and run the same query will find the name stored in the resolver cache (but not in the local cache).

As we present next in Section 4, in the campaign for measuring the delay for the resolution of a non-cached domain name, we recruited over 2 million clients that were connected to the Internet through over 2,500 ISPs. Thus, in our study we can indeed reach many clients per ISP, as expected. The proposed approach is to configure all clients to query for the same domain name that we have exclusively created for the purpose of this experiment. This domain name will not be present in any cache when we first launch the measurements (i.e., for the first client running the measurement tests). The first client performing the measurement will install the domain name in the resolver cache. Since we control the authoritative server for the domain name we use here, we configure the Time To Live (TTL) of the DNS record to a large value (24 h), enabling resolvers which honor the TTL to maintain the records in their caches throughout the duration of the experiment. Thus, all subsequent measurement clients served by the same resolver will find the domain name present in the resolver cache. Resolvers, specially public resolvers and local resolvers from large ISP have complex setups involving multiple instances both for the frontend and the backend [14] and consistency of the cached information cannot be assumed throughout the multiple instances of the same resolver. However, because we are making millions of queries to the public resolvers, even though they have multiple instances, each of them is likely to receive multiple queries for our measurement campaign, allowing us to appreciate the effect of caching. Keeping only the results from IPSs with more than 100 clients allow us to apply a similar argument applies for complex local resolvers.

The proposed approach would then result in both queries for non-cached names (the first query received by the resolver) and queries for cached names (the subsequent queries to the same resolver). The difficulty is that, because we are always querying for the same name, we are unable to tell them apart when we are processing the results.[9]

We are thus able to capture a mix of delays for queries for non-cached names and names present in the resolver cache, with prevalence for the latter. In order to reduce the impact of the non-cached names in the results, we require that our measurements come from those ISPs for which we have over 100 clients. This does not eliminate the contribution of the non-cached queries though (especially because some (large) ISPs have several resolvers), each of which have their own cache. Other factors, such as the use of `EDNS0 Client Subnet` option, may also result in querying to the authoritative server, even if the queried name is already in the resolver cache. Nevertheless, all these factors influence the resolution delay experienced by the users in real life, so our measurements do reflect the impact of caching in the resolution delay as perceived by users.

We collect the same data as in the previous experiment, namely, the client report including the measured delays and related information, and we also collect the information related to the queries received at the authoritative server. In order to force the default resolver to send a query to the authoritative server (even if the queried name *test.DoH-serv.com* is present in the resolver's cache), in this experiment we also program the ad to query for a non-cached name (which includes the unique ad identifier) using its default resolver. This is needed to be able to discover which resolver the client is using as default.

4. Dataset

For our experiments, we create two different ads: (i) Ad1 for measuring the resolution delay for a non-cached name (Section 3.1) and (ii) Ad2 for measuring the resolution delay when caching is involved (Section 3.2). With these, we then perform two measurements campaigns (one for each ad). The data about each of these campaigns if presented in Table 2. Given that each ad runs 10 queries each, our dataset includes a total of approximately 42 million DNS queries across both campaigns.

As mentioned in Section 2.3, the ad provider allow us to target specific geographical regions (within the coverage of the provider) and specific operating systems (OS). In terms of geographical coverage, we

[4] We validated in the lab that the different OSes and browsers reuse the connection for the subsequent query.

[5] There is nothing distinctive between these queries, other than they are each unique, and one is performed after the other (i.e., we do not launch both queries to the same resolver at the same time).

[6] The ECS is an option that allows the recursive resolver to specify the IP subnetwork of the client that is issuing the query, so the authoritative server can take it into account when replying, see [12].

[7] The resolution delay in this case is zero.

[8] Note that we do not control the local cache in the client, so we cannot clear it.

[9] In the test for measuring the resolution delay for a non-cached name, we were able to uniquely identify each query because it was for a unique name, but this is not the case in this test.

Table 2
Data about the ad campaigns.

Name	Start date	End date	Impressions
Ad1	26th Jan 2021	1st Feb 2021	1,677,438
Ad2	8th Jan 2021	12th Jan 2021	2,485,832

Table 3
Countries per geographical region where the targeted clients are located.

Europe	Latin America	USA
Spain	Peru	USA
France	Colombia	
UK	Mexico	
Germany	Argentina	
Portugal	Ecuador	
Netherlands	Chile	
Ireland	Guatemala	
Belgium	Honduras	
	Paraguay	
	Costa Rica	
	Panama	
	El Salvador	
	Uruguay	
	Nicaragua	

targeted clients located in 24 countries located in 3 regions, as detailed in Table 3.[10] In terms of the OS at the end-user, we targeted clients using Android, Windows and Linux. We obtained a large majority of Android clients (two thirds) and Windows (one third). We avoided printing the ad in clients running on MacOS and iOS because these operating systems clear the fields of the Performance Resource Timing API that we use in our analysis.

While we can target specific OSes, we cannot target the user agent of the clients. In our campaigns, we obtained that about 80% of the clients used Chrome browsers (including Chrome, Chrome Mobile and Chrome Mobile WebView), Firefox and Edge having about 7% of the clients each; the remaining includes other browsers and apps, such as Facebook, Samsung Internet, Instagram, Opera, etc.

In terms of the end-user access technology, we program the ad to report the type of interface used for accessing the Internet (i.e., Ethernet, Wifi or cellular). We find that approximately one third of the clients did not report back this information. Out of the ones who did report it, approximately one third of the clients were connected to the Internet through cellular access and the rest was though a fixed access (these include those using WiFi).

One of the potential concerns of the proposed approach is that the ad is displayed for a limited and unknown amount of time in the client, and it may not be enough to perform all the measurements. In our dataset, we find that 87% of ads that reported results back performed all the queries, while the rest reported only a subset of the results. Additionally, we found that the clients displaying the ads connect to more than 2,500 different ISPs in total.

4.1. Data cleaning

Each of the measurement campaigns for Ad1 and Ad2 generate two data sets: the primary dataset, which is the one of the control server and the auxiliary data set which is the one obtained at our authoritative DNS server. The primary dataset is composed by records that store the DNS resolution delay for each of the resolvers and additional information about the client, including the OS, the UA, the access technology, the IP prefix and the ad ID. The auxiliary data set contains information of the IP address of the resolver used in each ad, identified by the Ad ID. Using the Ad ID we can link the records from the primary and auxiliary databases. We combine and pre-process the records from the primary and auxiliary databases following the process depicted in Fig. 2.

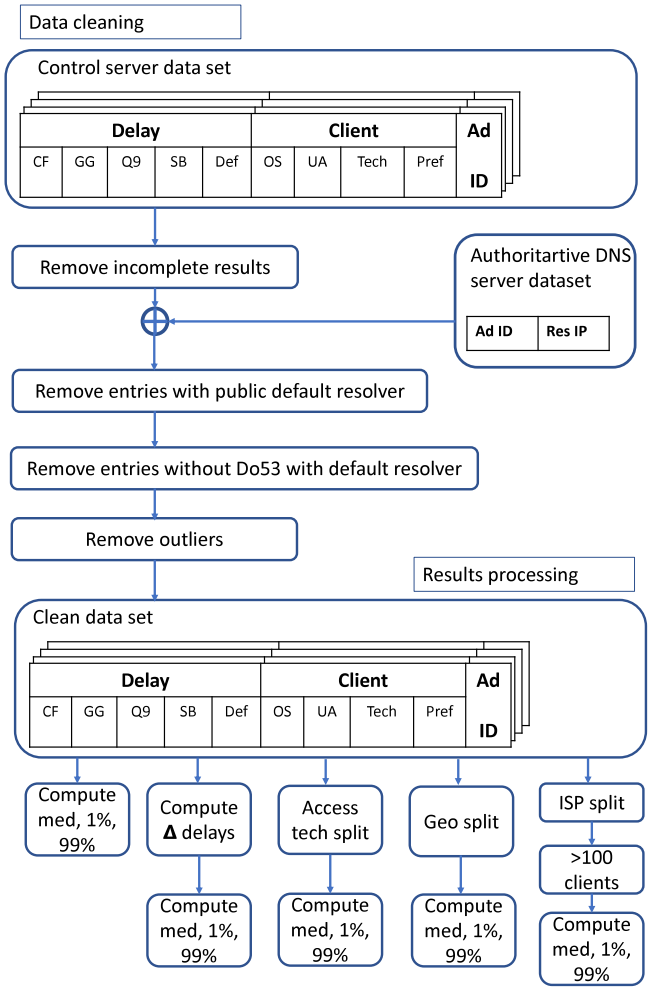


Fig. 2. Method for data pre-processing and processing.

Due to the nature of our measurement methodology (which relies on ads being displayed in different clients), there is about 13% of the clients that did not report back all the expected results. In order to avoid biasing our results when we compare the DNS resolvers or the transports, we compute the values for various metrics (e.g., median delay) using only the results from those clients that received replies for all the resolvers/transports we test (first step in the data cleaning depicted in Fig. 2.)

As we describe in Section 3, we measure the *local* resolver by sending queries to the *default* resolver. By *local resolver* we typically expect a private resolver service, typically offered by the access network (i.e., the ISP) to its customers. However, in some cases, the default resolver is not a local resolver, but a public one. We remove these users from our analysis, since their setup does not match our experimental goal (second step in Fig. 2). We reiterate that the purpose of our study is to understand the differences of performance between a local resolver through Do53 and a public resolver though DoH from the point of view of real users. Thus, correctly separating the measurements that represent these two setups is paramount.

We separate public resolvers from local resolvers (which we define as above) by separating the queries from known global/public

[10] USA is both a country and a region in our analysis

resolvers,[11] and assuming that the remaining ones are the ones corresponding to the local resolvers. We identify the public resolvers using the source IP address contained in the query received by our authoritative server.[12] As previously reported [15], few public resolvers concentrate most of the users. In our dataset, about 75% of clients that used a public resolver as default resolver used Google, 14% used Cloudflare, 10% used OpenDNS, and the remaining 1% used other public resolvers (including Quad9, FreeDNS, etc.). We note several notable absences in our list (e.g., public resolvers popular in China), which is an artifact of the geographical distribution we requested of the clients that printed our ads.

In addition to the filtering we describe above, we further need to ensure that the queries to the local resolver we separate actually use Do53, as per our experimental design, while the ones to global resolvers use DoH. To this end, we use the resolver (retrieved using the source IP address of the queries arriving to our authoritative server) and the User Agent to determine if the client is using Do53 or DoH with its default resolver. To achieve this, for our measurement campaigns we separate those users whose operating systems do not support DoH, in particular (i) any version of Android older than 9[13] [16]; (ii) any version of Windows older than 10 [17].

However, even if the OS does not support DoH, some browsers have enabled default DoH support directly at the application level. Note that we cannot target in our measurement campaigns specific browser versions, thus we need to filter our dataset *ex-post*, and only select the queries for which we can establish with high confidence the transport type. For this, we know that the following browsers will use/not use DoH by default:

- Chrome: For versions newer than 79[14] [18,19] will use DoH by default with any of the following resolvers: Cleanbrowsing, Cloudflare, Comcast, DNS.SB, Google, IIJ, OpenDNS, Quad9, OVDR . When using other resolvers, Chrome will not enable DoH by default and will use the DNS transport supported by the OS.
- Firefox: versions newer than 68 enable DoH by default for US users [20].[15]
- Edge: for versions newer than 86 [21], Edge will use DoH against the same set of resolvers listed for Chrome.

We assume that most clients do not manually enable DoH support and that the default behavior prevails. Given that we are measuring a large number of clients, we believe that the assumption will hold for most of them (and the few that modify the default behavior will not significantly affect our results).

As depicted in the third step in Fig. 2, we remove then form our dataset all the results corresponding to clients that not used Do53 to communicate with their default resolver. Also, there are cases in which the combination does not determine the transports; we also exclude these data points from further analysis.

Finally, we discarded from our dataset (across both Ad1 and Ad2 campaigns) all the queries that returned resolution delay values close to zero (i.e., below 3 ms), since most likely these queries were resolved by the local cache (e.g. the ad may print twice for the same user.[16]) We also observed some very large values for the resolution delays (e.g., up to 1 min in some cases). In order to prevent that such outliers distort our results, we discard the top 1% measurements values for each resolver we test, as depicted in the fourth step in Fig. 2.

Once we have clean the dataset, we can now compute the different metrics relative the DNS performance related to the distributions of the resolution delay for different resolvers. In Fig. 2 we present the different metrics computed during the processing of the results.

Table 4
Resolution delay for a non-cached name using the different public resolvers through DoH vs. using the local resolver through Do53, expressed in milliseconds. The Δ column contains the difference of the median resolution delay of the resolver in that row and the one of the local using Do53.

Transport	Resolver	Med	Δ	1st%	99th%	Stds
Do53	Local	153.9	0	14.6	1,136.8	243.5
DoH (Initial Query)	Google	403.1	249.2	77.7	2,746.0	508.5
	Cloudfl	394.6	240.7	59.4	2,841.6	538.2
	Quad9	493.0	339.1	66.0	3,386.3	624.6
	DNS.SB	671.9	518	124.3	3,716.8	682.3
DoH (Re-use)	Google	221.8	67.9	49.0	1,019.4	196.0
	Cloudfl	171.1	17.2	25.2	822.1	154.9
	Quad9	195.4	41.5	26.9	913.9	166.3
	DNS.SB	323.1	169.2	74.8	1,169.9	255.2

5. Resolution delay for a non-cached name

We compare the resolution delay for queries for a non-cached name performed to the local resolver using Do53 and the delay to the four different global resolvers using DoH (Section 3.1). In Table 4 we present different metrics associated to the resolution delay for a non-cached name for the four public resolvers using DoH and the local resolvers using Do53. For each of the resolvers, we performed roughly 2 million queries.

We observe that the delay for initial query for the public resolvers using DoH is significantly larger than the delay for a query to the local resolver using Do53 (blue highlighted line in Table 4). This is expected, since the first query using DoH requires the establishment of the TLS connection with the server. This is however not critical, since it is expected that most DoH implementations would reuse the TLS connection for subsequent queries, so this delay will only be incurred for the first one. In the case that client connects to the DoH public server using its domain name, the resolution delay for the initial query also includes the resolution delay for the resolver's name using the default resolver. For the rest of the paper, we focus on the queries re-using the TLS connection, as it is the most common and relevant case.

Regarding subsequent queries done to the public resolvers through DoH reusing the existent TLS connection (marked "Re-use" in Table 4), we observe that the median delay for the local resolver/Do53 is always smaller than the one for the public resolvers/DoH. However, the increments vary greatly, from as small as 17.2 ms for in the case of Cloudflare, to 41.5 ms for Quad9, 67.9 ms for Google and

[11] https://www.publicdns.xyz.

[12] Because we are using the queries received by the authoritative server to identify queries form public resolvers, queries issued by clients to DNS forwarders pointing to public resolvers are also classified as coming from a public resolver.

[13] Newer versions of Android will try to use DoT instead of Do53, if supported.

[14] In version 79, DoH was enabled for 1% of users and it became available for all users in version 83.

[15] The travel restrictions to tackle the COVID-19 pandemic which were in effect at the time of our study make it unlikely that a device originally from the US is roaming elsewhere.

[16] This filter removes data-points only for Ad2. For Ad2, the results' distribution is multimodal with one peak in 0 ms both for public and local resolvers. For the datapoints close to zero, about 70% had the same IP address than other query. This led us to conclude that the results close to 0 ms were repeated queries for the same client, which results were locally cached. This is possible, because, even though we configured the ad platform to avoid printing the ad multiple times in the same client, the techniques used to avoid repetition are error prone.

even as much as 171.1 ms for DNS.SB. On the other hand, all public resolvers/DoH except DNS.SB exhibit a smaller 99th percentile, with Cloudflare achieving the higher reduction of 300 ms. While for both Cloudflare and Quad9 the 99th percentile of the delay is significantly reduced, for the other two resolvers, there is little or no benefit for longer queries. We note that our measurements results for Cloudflare show somehow higher median resolution delay than Mozilla previously reported in their pilot study [5] (6 ms) and similar results regarding the 99th percentile. While the performance of DNS.SB is somehow expected, given their smaller global footprint (compared to the other ones tested), Google's resolver poor performance compared to other global public resolvers is unexpected, given the large footprint of Google's DNS service.

Takeaway: There are large differences in performance among public resolvers (even between large resolvers), and the selection of the public resolver heavily determines the impact on the performance experienced by the user when switching to a public resolver through DoH from its local resolver though Do53. The penalty results in an increase of the median delay ranging from 14 ms in the best performing public resolver up to 170 ms to the worst one.

In Fig. 3 we plot the CDF of the difference of the resolution delay of a non-cached name between using the local resolver with Do53 and using different public resolvers with DoH, reusing the TLS connection. We calculate the deltas per client, from the values of the delay measurements we collect from each client. These results align to our previous observations. In the case of DNS.SB, only 10% of clients experience a smaller delay when using the public resolver/DoH compared to the local resolver/Do53. This number increases to 20% in the case of Google/DoH, and further more to 35% in the case of Cloudflare/DoH and Quad9/DoH.

In terms of additional delay, compared to using the local resolver/Do53, half of the clients using the public resolvers/DoH experienced an additional delay of at least 14 ms when using Cloudflare/DoH, 17 ms when using Quad9/DoH, 63 ms when using google DoH and 135 ms when using DNS.SB/DoH. We also observe that there is a set of clients that experienced a significantly larger (over 100 ms more) delay using the public resolvers/DoH than when using a local resolver/Do53 (the right part of the graph). This set is considerably smaller than the one that experienced a larger increase on the other direction (on the left side of the graph) but it is non-negligible. This result was not reported in Mozilla's pilot study [5].

We also investigate whether the different network access technologies impact the resolution delay. Essentially, we observe that the overall behavior is similar in the two access technologies (in terms of which resolvers perform better), only that an extra delay of roughly 70 ms is introduced in the case of the cellular compared to the case of fixed access. We also observe an increase in the standard deviation of roughly 40 ms for all the public resolvers in the case of cellular compared to fixed, which is also reasonable since wireless conditions tend to exhibit wider variations in capacity. We also look into the first percentile and we see that there is an increase of roughly 20 ms for all resolvers when comparing cellular versus fixed.

We do not observe any significant differences when separating the results per operating system nor per User Agent.

5.1. Geographical region impact

We next separate our dataset based on the main geographical regions which our dataset covers. We divide our data into Europe, the U.S. and Latin America. For each resolver, we collected approximately 420,000 datapoints for LATAM, 520,000 for Europe and 300,000 datapoints for USA. We summarize our results in Table 5.

We first note that the distance from the end-user to the authoritative server dominates the delay values. Since the name is not in the resolver's cache, the latter must retrieve it from the authoritative server.

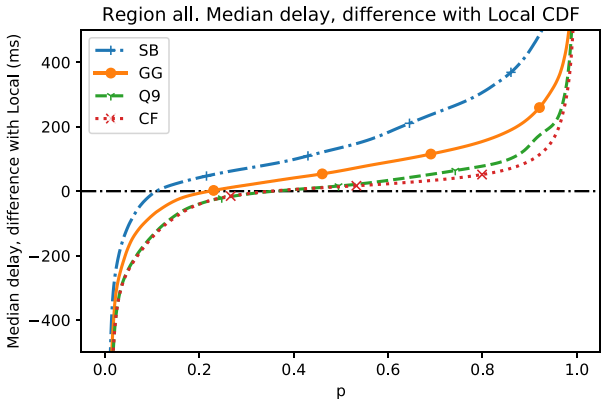


Fig. 3. CDF of the difference of the resolution delay of a non-cached name between the delay of the local resolver using Do53 and the different public resolvers using DoH, reusing the TLS connection. Expressed in ms.

Table 5
Resolution delay for a non-cached name using the different public resolvers through DoH and the local resolver though Do53, separated by geographical region, expressed in ms.

Region	Resolver	Med	1st%	99th%	Std
LATAM	Local (Do53)	270.3	145.4	2,184.1	324.2
	Google (DoH)	374.3	188.2	1,342.2	205.2
	Cloudfl (DoH)	276.5	171.2	1,107.4	161.5
	Quad9 (DoH)	277.6	176.5	1,250.5	182.0
	DNS.SB (DoH)	524.3	220.0	1,457.0	267.6
Europe	Local (Do53)	58.0	10.9	458.6	85.7
	Google (DoH)	132.8	43.8	691.7	120.3
	Cloudfl (DoH)	76.8	20.7	528.2	89.2
	Quad9 (DoH)	92.2	21.2	586.1	105.0
	DNS.SB (DoH)	166.7	66.3	821.0	149.7
USA	Local (Do53)	158.0	92.1	834.6	147.9
	Google (DoH)	200.2	111.4	897.0	144.0
	Cloudfl (DoH)	170.3	109.4	780.0	113.7
	Quad9 (DoH)	182.7	98.9	889.1	134.9
	DNS.SB (DoH)	383.1	142.5	1,056.0	168.4

Since the authoritative server is located in Europe, we observe that the Europe clients experience a significantly lower delay than the clients located in other regions, followed by the clients in the USA and finally the clients located in LATAM.

Takeaway: Overall, the median delay for the local resolver/Do53 is smaller than the one for public resolvers/DoH, but the differences vary depending on the public resolver and the region. For instance, in LATAM and in the USA, both Cloudflare and Quad9 perform similar than the local resolver in terms of the median (less than 4% added delay in LATAM and less than 15% in USA). In Europe, the delay added by the use of public resolvers/DoH is at least an additional 30% higher, increasing up to 130% for Google/DoH, or 186% for DNS.SB.

We find surprising the poor performance of Google/DoH's resolver, for clients located in Europe, the median delay is twice the one observed for the local resolvers/Do53. In the USA, Google resolver/DoH performs much closer to the local resolver/Do53 than in other regions. DNS.SB/DoH underperforms compared to the local resolver/Do53 in all regions, as expected from a small public resolver.

5.2. Analysis per ISP

We have observed significant differences in the performance of the different public resolvers. We hint that this may also apply for the different local resolvers available. We next discriminate the data based

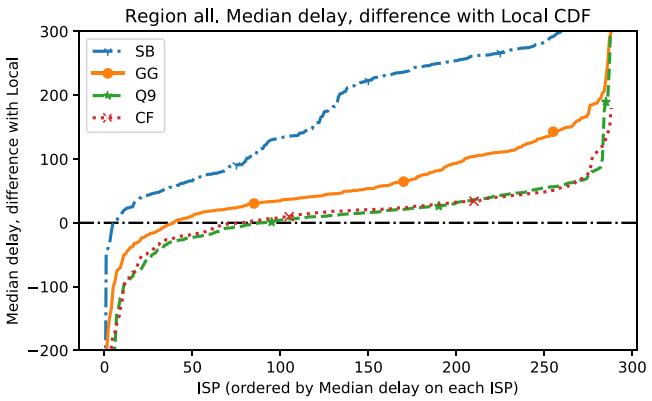


Fig. 4. CDF of the difference of the medians of the delay for the resolution of a non-cached name using the different public resolvers/DoH and the local resolver/Do53 per ISP. Expressed in ms.

on the ISP. We classify the queries received by our authoritative server according to the ISPs using the source address contained in the query. We look into the ISPs that have at least 100 impressions for each resolver. These are 289 ISPs that account for 70% of the impressions. For each of these ISPs, we compute the difference between the median of the resolution time obtained using each of the public resolvers/DoH and the median of the resolution time obtained using the local resolver/Do53. In Fig. 4, we plot the CDF of the differences computed for the different ISPs. We find that, for a reduced number of ISPs, their local resolvers/Do53 perform worst than the public resolvers/DoH — while for the large majority of ISPs, it is the opposite. More specifically, for DNS.SB there is only one ISP (out of 290) whose median value of the resolution delay using the local resolver/Do53 is at least 50 ms larger than the median of the DNS.SB resolver, while this number increases to 12 (4%) in the case of Google, 23 (8%) for Cloudflare and 28 for Quad9 (10% of ISPs). We can see that the set of ISPs whose local resolver underperforms more than 50 ms with respect to Quad9 is roughly a superset of those underperforming with respect to Cloudflare, which in turn is a superset include those of Google which in turn includes the one for DNS.SB. In other words, there is a small set of local resolvers that perform poorly.

We further find that the percentage of ISPs whose local resolver over Do53 under-performs compared to the public resolver over DoH changed in the following way: 2% for DNS.SB, 14% for Google, 27% for Cloudflare and 31% for Quad9. However, we find that the local resolver over Do53 out-performs the public resolver over DoH by at least 50 ms in median value for 12% of the ISPs for Cloudflare, 15% of the ISPs for Quad9, 51% of the ISPs for Google and 89% of the ISPs for DNS.SB.

Takeaway: There is a small set of ISPs whose local resolver/Do53 perform poorly. In these cases, the use of public resolvers/DoH can significantly improve the performance. However, for the vast majority of ISPs, the local resolver/Do53 combination performs better.

6. EDNS0 client subnet support

We next analyze the support for EDNS0 Client Subnet (ECS) in the different resolvers. This is relevant because the use of ECS affects the caching behavior, so having insights on how widespread ECS is supported will also help us to interpret the results in the measurements involving caching in the next section.

Table 6 provides the details of ECS support in the different resolvers and transports. We find that Cloudflare, Quad9 and DNS.SB do not support ECS. Google supports ECS caching,[17] so only a small percentage of queries contain the ECS (we assume that the initial queries after the cached information times out). We observe a similar behavior in the Google resolver to DoH and Do53 queries.

Regarding the local resolvers, only 0.13% of the queries included ECS. In terms of different local resolvers, we observe that 4,495 resolvers that generated queries, only 438 of them included the ECS (we distinguish resolvers using the source IP address of the query).

[17] See https://en.wikipedia.org/wiki/Google_Public_DNS.

Table 6
Percentage of the total impressions that included the ECS option, per resolver and per transport.

Resolver (Transport)	ECS supported	Total prints
Local resolver (Do53)	0.13 %	3,772,947
Google (DoH)	5.1%	4,527,265
Cloudflare (DoH)	0.0%	4,471,866
Quad9 (DoH)	0.0%	4,471,063
DNS.SB (DoH)	0%	4,465,818

Table 7
Resolution delay using the different public resolvers through DoH and the local resolver though Do53 with caching, expressed in [ms]. The last column includes the ratio between the total queries users made via the ads and the ones the authoritative server received.

Geo	Resolver	Med	1st%	99th%	Std	#	Ratio
Global	Local (Do53)	44.0	5.7	666.4	213.9	1.7M	87.1%
	Google (DoH)	118.4	4.2	837.9	164.5	1.7M	98.1%
	Cloudfl (DoH)	61.7	13.4	655.3	123.9	1.7M	99.9%
	Quad9 (DoH)	71.2	4.2	700.2	135.6	1.7M	99.8%
	DNS.SB (DoH)	203.3	4.3	978.2	219.8	1.7M	99.9%
LATAM	Local (Do53)	47.5	7.7	1,540.2	314.5	600K	95.8%
	Google (DoH)	163.7	9.2	971.8	188.6	600K	98.1%
	Cloudfl (DoH)	80.1	20.1	760.3	143.3	600K	99.9%
	Quad9 (DoH)	114.2	9.0	873.5	160.7	600K	99.8%
	DNS.SB (DoH)	342.9	9.6	1,199.6	230.8	600K	99.9%
Europe	Local (Do53)	34.6	4.9	347.8	115.6	800K	88.4%
	Google (DoH)	88.3	3.8	705.0	131.9	800K	99.0%
	Cloudfl (DoH)	49.8	11.9	582.8	102.8	800K	99.9%
	Quad9 (DoH)	49.6	3.8	575.0	102.9	800K	99.9%
	DNS.SB (DoH)	91.1	3.9	630.6	110.8	800K	99.9%
USA	Local (Do53)	74.6	7.5	640.2	138.9	300K	67.4%
	Google (DoH)	122.4	4.4	836.2	165.9	300K	96.3%
	Cloudfl (DoH)	63.0	14.1	616.0	129.1	300K	99.7%
	Quad9 (DoH)	69.3	4.2	586.0	126.5	300K	99.7%
	DNS.SB (DoH)	338.4	4.8	857.0	180.6	300K	99.9%

7. Impact of caching in the resolution delay

In this section, we compare the resolution delay results when using a public resolver/DoH and a local resolver/Do53 when caching is involved (Section 3.2). To this end, we measure the resolution delay using different resolvers/transports while querying repeatedly for the same domain name under our control. In order to increase the ratio of queries responded using the cached information, we keep only the results for those ISPs that have more than 100 measurement clients. After this filtering, we keep the results for 307 ISPs. These account for over 97% of the measurements (i.e. because of this we discard less than 3% of the impressions). Thus, for each public resolver and for the set of local resolvers, we performed about 1.7 million queries, for the same domain name. We present our results in Table 7. We only include delays for DoH re-using the TLS connection.

At first glance, we observe that the medians are much lower than the ones measured for the non-cached experiments, so our experiment is successful in incorporating the effect of caching in the resolution delay. However, we do observe that the observed medians are larger for the

LATAM and USA than for Europe, meaning that the non-cached queries still impact the results (as expected). In the last column of Table 7 we include the ratio of queries that were received by the authoritative server and the total queries made. While not perfectly accurate,[18] this ratio reflects the success ratio of the caching (i.e. all queries that were not received by the authoritative server were responded using the information available in a cache). We observe that the cache success ratio is much larger in the public resolvers than in the local resolvers. This is expected, as there is at least one local resolver per ISP while clients from several ISPs can use the same public resolver instance. We observe that the caching success ratio varies significantly per region, being the lowest in the USA. Looking closely, we observed that larger ISPs in the USA deploy a fairly large number of resolvers instances (we identified them by counting the source IP addresses contained in the queries received at the authoritative server). For an unpopular domain name, like the one we are using in our measurements, a large number of resolver instances probably reduces the delay towards the resolver, but also decreases the cache efficiency. When looking at the public resolvers, we observe that the Google public resolver has a lower cache success ratio than the other. While several factors may contribute to this observed behavior [14,22], this can be affected by the use of EDNS0 Client Subnet. As presented in 6, DNS.SB, Cloudflare and Quad9 never use it and Google, caches the support for ECS in the authoritative servers.

If we take a look at the global results in the upper part of Table 7, we observe that the differences in the medians of the public resolvers/DoH with respect to the local resolvers/Do53 are similar to the case of the non-cached name, namely 17 ms for Cloudflare, 30 ms for Quad9, 70 ms for Google and 160 ms for DNS.SB. However, because the delays are significantly lower, this implies a much larger increase in proportional terms, resulting in an increase of 40% for Cloudflare, 62% for Quad9, 170% for Google and 360% for DNS.SB. this is particularly relevant form the user's experience perspective, since over 80% of DNS queries are solved through the caches [23].

Regarding the 99th percentile, we do not observe a reduction as we did for the non-cached names when using the public resolvers/DoH. On the contrary, the best performing public resolver/DoH (Cloudflare) has a similar 99th percentile than the local resolvers/Do53 while the other show an increase of at least 100 ms.

Takeaway: When caching is involved, the increase of the median delays when using a public resolver/DoH compared to the local resolver/Do53 is at least 40% and up to 360%. About the 99th percentile, while we do not observe differences between the best performing public resolvers/DoH and the local resolvers/Do53, there is a notable increase for the remaining public resolvers/DoH.

Fig. 5 shows the CDF of the differences on the median delays for the ISPs with more than 100 clients. Similarly to the non-cached case, there is a small number of local resolvers/Do53 that perform poorly but for the remaining vast majority of ISPs, the local resolvers outperform the public ones/DoH. The performance penalty for the different ISP is consistently lower for Cloudflare, closely followed by Quad9, then Google and then DNS.SB. We observe that Quad9 has a sharp increase for larger value of the delay, surpassing Google. Looking more closely, this is because Quad9 perform worse than google in LATAM.

8. Discussion

In the last two sections, we measured the combined effect of changing both the resolver and DNS transport in the resolution delay. When comparing with the median resolution time of the local resolver/Do53, we observed an increase of 17 ms when using Cloudflare/DoH, 40 ms for Quad9, 70 ms for Google/DoH and 170 ms for DNS.SB (we obtained similar differences for non-cached queries and when caching is involved). What does this means from the user's perspective?

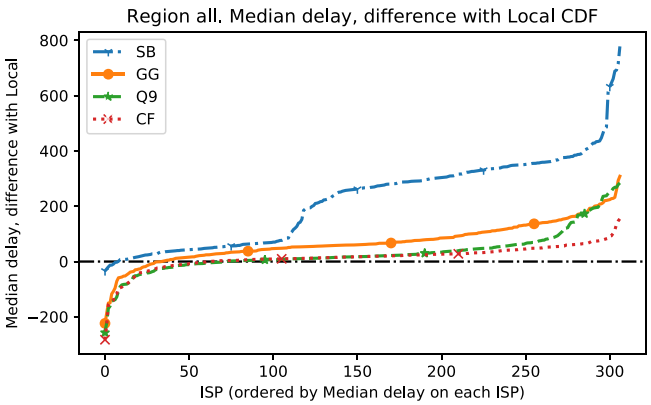


Fig. 5. CDF of the difference of the medians of the resolution delay using the different public resolvers/DoH and the local resolver/Do53 per ISP when using caching. Expressed in ms.

Some data-points allow to put this data in context.

- According to a Google survey [24], increasing the loading time of the search results for 500 ms increases the bounce rate (users that left the page without clicking) in 20%
- 100 ms loading time improvement led to an increase of revenues of a 0.6% for Bing [25], a 0.7% for Zalando [26] and 1% for Amazon [25]
- According to Bing, "an engineer that improves server performance by 10 ms (that is 1/30 of the speed that our eyes blink) more than pays for his fully-loaded annual costs".[25]

From this data, we conclude that migrating to resolver setups that increase the resolution delay in 100 ms or more certainly has an impact on the user's experience. For those combinations that result in a penalty of 10 ms, the situation is less clear. It is certainly one additional contribution and if there are several sequential DNS resolutions needed to load the page, they can add up to an amount that can affect user experience.

9. Ethical considerations

The experiments and data collection practices described in this paper have been supervised and approved by our institutional Data Protection Officer (DPO) and also by the Ethics Review Board (ERB) of UC3M. To the best of the authors' knowledge, the conducted experiments are compliant with the Terms of Service of our ad provider and any applicable law.

The two main potential ethical concerns associated with the experiments relate to: (i) the public IP address of the end-user, which is considered Personally identifiable information by the EU legislation; and (ii) the consumption of data and energy in end user devices to run the measurements.

Regarding the first one, we do not analyze nor store the full source IP address of the clients. Upon the reception of the report from the clients running the Ad to the control server, we extract the /24 prefix from the source address and we discard the full source address. The prefix is used to determine the client's ISP. No personal or sensitive data is collected. We have not done any specific targeting set-up in our ad campaigns that might cause any privacy concern to end-users.

[18] One query by the client may result in several queries to the authoritative server see [13].

Regarding the second issue, we first observe that if our Ad was not displayed in a particular client, another Ad would be. So, we design our Ad to consume a similar amount of resources than and alternative ''typical'' ads shown in online advertising. If our Ad would consume more resources, then our experiment would be imposing an overhead on users resources.

Typical ads embed pixels from several players for different purposes: measuring KPIs associated to ad campaigns (viewability, clicks, etc.), checking the ad space properties, identifying fraudulent activity, etc. Moreover, most ads trigger in general at least one (but in many cases several) cookies. Hence, overall the resources consumed by a regular ad are expected to be larger in most cases compared to our ad. Based on our lab experiments consume at most 200 KBytes. According to [27], the average web page size is over 1.5 MB for desktop clients and approximately 0.9 MB for the mobile counterparts, so the load increase because of our ad is less than 1%. Compared to regular Ads, according to [28] 'the cost of downloading ads and tracking data sums as more than 30% of data volume' of the web page. Hence, the load imposed by our Ad is negligible compared to the overall Ad load. Other resources like CPU or memory used in our ad are rather limited, since we just force the browser to generate messages. Instead, a typical ad running pixels for checking online ads KPIs would be much more resource consuming in terms of CPU and memory.

Finally, regarding explicit consent, we do not obtain explicit consent from the users, as there is no way of doing so (i.e. users do not give explicit consent regarding Ads displayed when they visit web pages). We argue that there is an implicit consent involved. In order to run the experiments we are actually just taking advantage of the ecosystem as it exists now: namely, when a user receives an ad in the webpage she is viewing she is not giving explicit consent for whatever code might be executing together with that ad, but the consent is implicit by agreeing to continue viewing the content of that webpage.

10. Related work

In 2018, Mozilla conducted a measurement study [5] of DoH query response times with Firefox Nightly users. They observed that DoH queries were 6 ms slower than Do53 queries (in mean), and that DoH actually has faster response times than Do53 for the slowest queries. This experiment recruited 25,000 clients and performed a billion queries. They only used Cloudflare's DoH resolver and Firefox browser.

T. Böttger et al. [29] was one of the earliest papers analyzing DoH performance. The measured the resolution delays using different transports (DoH, Do53) and using a local resolver and both Google and Cloudflare resolvers for the top 1,000 Alexa domain names. They find that the local resolver using Do53 was faster, followed by Cloudflare using DoH and lastly Google using DoH. Our results are consistent with that. Our contribution is that we measured millions of vantage points, with thousands of real local resolvers.

C. Lu et al. [30] performs a large scale measurement study of both DoT and DoH. As vantage points, they use three SOCKS proxy networks, one of them being global and the other two located in China. They measure both reachability and performance to 3 global resolvers namely Cloudflare, Google and Quad9, using DoH, DoT and Do53. As a caveat, they use Do53 over TCP, since they rely on a SOCKS proxy network. Also, since they do not control the endpoint, to compute the resolution delay they need to estimate, and discount the delay between the measurement point and the vantage point. In any case, they only compare the different transports towards public resolvers. They can only compute the delays in the case of connection reuse. They observe that using DoH introduced an extra delay compared to using Do53 to the same public resolvers. In particular, they found that Cloudflare takes 6 ms longer (median). They do not provide data for Google nor Quad9. They do report that the added latency varies from country to country and in some countries they do observe that DoH is faster than Do53. In any case, they do not compare the performance with respect to local resolvers.

A. Hounsel et al. [31] experimentally compares the DNS query resolution delay and the page load time for different DNS transports and different resolvers. They perform the measurements for 5 vantage points in EC2 instances in US, Germany, Australia and North Korea. They tested 3 public resolvers, Google, Cloudflare and Qaud9 over DoH, DoT and Do53. They used a local resolver provided by EC2 using Do53. They emulate changing network conditions by using traffic shapers. Regarding the resolution delay, they measure the resolution delay for popular domain names, likely to be cached, and they observe that Do53 is a few ms faster than DoH in average (e.g. 6 ms for Cloudflare) but that the standard deviation is three times higher in Do53 than in DoH. The differences with our study are notable. First, we test millions of vantage points, with thousands of providers, covering different countries with a wide range of income per inhabitant. Another important difference is that they only tested the local resolver provided by EC2, while we are testing the real default resolvers used by clients.

In a follow paper [32], A. Hounsel et al. experimentally compares the performance of 3 public resolvers and the local resolver using DoH, DoT and Do53. They use 2,693 hardware probes directly attached to the router/modem in user's home network. The observed that the median latency for the resolution delay of a cached name is 0,85 ms for the local resolver using Do53 while the median delay for the 3 tested public resolvers through DoH ranged between 20 and 33 ms. The differences with our study include the size (thousands versus millions of vantage points), the geographical scope (USA versus USA/LATAM/EU), the type of probe (hardware probe connected to home router versus the actual application used by the user running in the user's device) and the access technology (fixed, versus fixed and cellular).

11. Conclusions

In this study we find that the impact on performance of using a public resolver through DoH (compared to the use of local resolver through Do53) heavily depends on the actual public and local resolver used. We observed that there are significant differences between resolvers, both local and public. Regarding the local resolvers, we observe that there is a reduced number of local resolvers that perform poorly compared to the public resolvers. Overall, the large majority of local resolvers outperform all the public ones. This is true in terms of the median delay with and without caching. Regarding public resolvers, we find that Cloudflare performs closer to local ones, Quad9 introduces a minor additional penalty in the majority of the cases, while Google and DNS.SB drastically increase the resolution delay (and decrease performance). The datapoint regarding Google's performance is particularly relevant as it serves for 75% of public resolver's clients [15]. While the absolute increase in the median delay is somehow constant when using and not using caching, respectively, the relative impact increases greatly, doubling and even tripling the median experienced delay when caching is involved.

In addition, through this study we have been able to reproduce and verify Mozilla's study [5]. We find similar results when comparing both the median and the 99th percentile resolution delay of a non-cached query for Cloudflare/DoH and the local resolver/Do53 (i.e., a minor increase in the median and an improvement in the 99th percentile,

which is consistent with Mozilla's report). When looking into the impact of caching, we observe that the increase in the median becomes significant in relative terms, and that the gain in the 99th percentile is no more. We extended the study to other resolvers and other browsers. We do not observe differences in terms of browsers.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been partially funded by the Internet Society (ISOC), the EU through the 5G-VINNI project (GA- 815279) and the Madrid Government (Comunidad de Madrid-Spain) under the Multiannual Agreement with UC3M in the line of Excellence of University Professors (EPUC3M21), and in the context of the V PRICIT (Regional Programme of Research and Technological Innovation). Funding for APC: Universidad Carlos III de Madrid (Read & Publish Agreement CRUE-CSIC 2022). Approval of the version of the manuscript to be published.

References

[1] Scott Rose, Matt Larson, Dan Massey, Rob Austein, Roy Arends, DNS Security introduction and requirements, in: Request for Comments, (4033) RFC Editor, 2005, http://dx.doi.org/10.17487/RFC4033, URL https://rfc-editor.org/rfc/rfc4033.txt.

[2] Paul E. Hoffman, Patrick McManus, DNS Queries over HTTPS (DoH), in: Request for Comments, (8484) RFC Editor, 2018, http://dx.doi.org/10.17487/RFC8484, URL https://rfc-editor.org/rfc/rfc8484.txt.

[3] Selena Deckelmann, Firefox continues push to bring DNS over HTTPS by default for US users, in: The Mozilla Blog, 2020.

[4] Nilay Patel, ATT And verizon both want to run massive ad-tracking networks to rival facebook, in: The Verge, 2018, Last accessed in 3/8/2021.

[5] Patrick McManus, Firefox nightly secure DNS experimental results, in: Firefox Nightly News, 2018, https://blog.nightly.mozilla.org/2018/08/28/firefox-nightly-secure-dns-experimental-results/, last accessed in 3/8/2021.

[6] TAPTAP Digital, TAPTAP. Location Intelligence for marketing, 2020, https://www.taptapdigital.com/.

[7] Patricia Callejo, Conor Kelton, Narseo Vallina-Rodriguez, Rubén Cuevas, Oliver Gasser, Christian Kreibich, Florian Wohlfart, Ángel Cuevas, Opportunities and challenges of ad-based measurements from the edge of the network, in: Proceedings of the 16th ACM Workshop on Hot Topics in Networks, 2017.

[8] G. Huston, APNIC Labs IPv6 measurement system, 2013, https://labs.apnic.net/?p=348.

[9] Google, JSON API For DNS over HTTPS (DoH), in: Google Public DNS Guide, 2021, Last accessed in 3/8/2021.

[10] MDN contributors, Performanceresourcetiming, in: MDN Web Docs, 2020, Last accessed in 3/8/2021.

[11] Xiao Wang, Aruna Balasubramanian, Arvind Krishnamurthy, David Wetherall, Demystifying page load performance with WProf, in: Proceedings of the USENIX Conference, 2013, pp. 473–486.

[12] Carlo Contavalli, Wilmer van der Gaast, David C Lawrence, Warren "Ace" Kumari, Client subnet in DNS queries, in: Request for Comments, (7871) RFC Editor, 2016, http://dx.doi.org/10.17487/RFC7871, RFC 7871.

[13] Giovane C.M. Moura, John Heidemann, Moritz Muller, Ricardo de O. Schmidt, Marco Davids, When the dike breaks: Dissecting DNS defenses during DDoS, in: Proceedings of the Internet Measurement Conference 2018, Association for Computing Machinery, New York, NY, USA, 2018, http://dx.doi.org/10.1145/3278532.3278534.

[14] Audrey Randall, Enze Liu, Gautam Akiwate, Ramakrishna Padmanabhan, Geoffrey M. Voelker, Stefan Savage, Aaron Schulman, Trufflehunter: Cache snooping rare domains at large public DNS resolvers, in: Proceedings of the ACM Internet Measurement Conference, Association for Computing Machinery, New York, NY, USA, 2020, http://dx.doi.org/10.1145/3419394.3423640.

[15] Geoff Huston, DNS Resolver centrality, in: APNIC Labs, 2019, Last accessed in 3/8/2021.

[16] Dave Burke, Introducing android 9 pie, 2018, https://android-developers.googleblog.com/2018/08/introducing-android-9-pie.html.

[17] Brandon LeBlanc, Announcing windows 10 insider preview build 19628, 2020, https://blogs.windows.com/windows-insider/2020/05/13/announcing-windows-10-insider-preview-build-19628/.

[18] Kenji Baheux, Addressing some misconceptions about our plans for improving the security of DNS, 2019, https://blog.chromium.org/2019/10/addressing-some-misconceptions-about.html.

[19] Kenji Baheux, A safer and more private browsing experience with secure DNS, 2019, https://blog.chromium.org/2020/05/a-safer-and-more-private-browsing-DoH.html.

[20] Staged rollout of DoH to US users, 2019, https://bugzilla.mozilla.org/show_bug.cgi?id=1573840.

[21] Mauro Huc, How to enable DNS over HTTPS on microsoft edge, 2020, https://pureinfotech.com/enable-dns-over-https-microsoft-edge/.

[22] Wouter B. de Vries, Roland van Rijswijk-Deij, Pieter-Tjerk de Boer, Aiko Pras, Passive observations of a large DNS service: 2.5 years in the life of google, IEEE Trans. Netw. Serv. Manag. 17 (1) (2020) 190–200, http://dx.doi.org/10.1109/TNSM.2019.2936031.

[23] Jaeyeon Jung, E. Sit, H. Balakrishnan, R. Morris, DNS performance and the effectiveness of caching, IEEE/ACM Trans. Netw. 10 (5) (2002) 589–603, http://dx.doi.org/10.1109/TNET.2002.803905.

[24] Tiffany Poss, How does load speed affect conversion rate? in: Modern Marketing Blog, 2016, Last accessed in 3/8/2021.

[25] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, Nils Pohlmann, Online controlled experiments at large scale, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: KDD '13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 1168–1176, http://dx.doi.org/10.1145/2487575.2488217.

[26] Christoph Luetke Schelhowe, Shuhei Kagawa, Thorbjoern Gruda, Jeff Cybulski, David Martin Jones, Loading time matters, in: Zalando Engineering Blog, 2018, Last accessed in 3/8/2021.

[27] Troy Johnson, Patrick Seeling, Desktop and mobile web page comparison: characteristics, trends, and implications, IEEE Commun. Mag. 52 (9) (2014) 144–151, http://dx.doi.org/10.1109/MCOM.2014.6894465.

[28] Stefano Traverso, Martino Trevisan, Leonardo Giannantoni, Marco Mellia, Hassan Metwalley, Benchmark and comparison of tracker-blockers: Should you trust them? in: 2017 Network Traffic Measurement and Analysis Conference, TMA, 2017, pp. 1–9, http://dx.doi.org/10.23919/TMA.2017.8002898.

[29] Timm Böttger, Felix Cuadrado, Gianni Antichi, Eder Leão Fernandes, Gareth Tyson, Ignacio Castro, Steve Uhlig, An empirical study of the cost of DNS-over-HTTPS, in: Proceedings of the Internet Measurement Conference, Association for Computing Machinery, New York, NY, USA, 2019, http://dx.doi.org/10.1145/3355369.3355575.

[30] Chaoyi Lu, Baojun Liu, Zhou Li, Shuang Hao, Haixin Duan, Mingming Zhang, Chunying Leng, Ying Liu, Zaifeng Zhang, Jianping Wu, An end-to-end, large-scale measurement of DNS-over-encryption: How far have we come? in: Proceedings of the Internet Measurement Conference, in: IMC '19, Association for Computing Machinery, New York, NY, USA, 2019, http://dx.doi.org/10.1145/3355369.3355580.

[31] Austin Hounsel, Kevin Borgolte, Paul Schmitt, Jordan Holland, Nick Feamster, Comparing the effects of DNS, DoT, and DoH on web performance, in: Proceedings of the Web Conference 2020, in: WWW '20, Association for Computing Machinery, New York, NY, USA, 2020, http://dx.doi.org/10.1145/3366423.3380139.

[32] Austin Hounsel, Paul Schmitt, Kevin Borgolte, Nick Feamster, Can encrypted DNS be fast? in: Proceedings of the Passive and Active Measurement (PAM) Conference, 2021.

Patricia Callejo is a post-doc researcher at UC3M-Santander Big Data Institute. She obtained her M.Sc. (2016) and Ph.D. (2020) at University III of Madrid in the field of Telematics Engineering. She was granted by RIPE Academic Cooperation Initiative (RACI) on RIPE 76 that took place in Marseille, France in 2018. The same year, she did an internship in the International Computer Science Institute (ICSI) at UC Berkeley (USA), as part of her Ph.D. She is author of conference papers such as ACM HotNets, ACM CoNEXT, and WWW. She has participated in EU H2020 projects. Her areas of interest include Internet measurements, online advertising, and web transparency.

Marcelo Bagnulo received the Electrical Engineering degree from the University of Uruguay and the Ph.D. degree in telecommunications from the Universidad Carlos III de Madrid (UC3M), Spain. Since 2008, he has been a tenured Associate Professor at UC3M. He has published more than 80 articles in the field of advanced communications in journals and congresses, including IEEE INFOCOM, ACM SIGCOMM, ACM Mobicom, ACM IMC, and IEEE/ACM TRANSACTIONS ON NETWORKING. He is the author of 21 RFCs in the Internet Engineering Task Force (IETF), including the Shim6 protocol for IPv6 multihoming and the NAT64/DNS64 tools suite for IPv6 transition. He has 26 H-index and 4258 total citations. His research interests include Internet architecture and protocols, interdomain routing, and security. From 2009 to 2011, he was a member of the Internet Architecture Board.

Jaime González Ruiz received a degree in Telecommunications from the University of Seville (Spain) in 2019 and a double Master degree both in Telecommunications and Cyber Security from University Carlos III in 2021. He now works as a Cyber Security Engineer at Iberdrola.

Andra Lutu is a Senior Researcher at Telefonica Research, in Madrid, Spain. Her main research interests lie in the areas of network measurements, interdomain routing and mobile networks. As part of Telefonica Research, Andra has been the recipient of an H2020 MSCA Individual Fellowship grant funding her work on Dynamic Interconnections for the Cellular Ecosystem (DICE).

Alberto García-Martínez received the degree in telecommunication engineering, in 1995, and the Ph.D. degree in telecommunications, in 1999. In 1998, he joined the Universidad Carlos III de Madrid (UC3M), where he has been an Associate Professor, since 2001. He has published more than 50 articles in technical journals (IEEE/ACM Transactions on Networking, Computer Networks), magazines (IEEE Wireless Communications, IEEE Communications Magazine), and conferences. He has coauthored three RFCs. His main interests include interdomain routing, transport protocols, network security, and blockchain technologies.

Rubén Cuevas is Associate Professor in the Telematics Engineering Department at Universidad Carlos III of Madrid (UC3M). He is also the Deputy Director of the UC3M-Santander Big Data Institute (IBiDAT). Between January and December 2012 he was Courtesy Assistant Professor in the Computer and Information Science department at University of Oregon. He obtained his Ph.D. and M.Sc. in Telematics Engineering and M.Sc. in Telecommunications Engineering at University Carlos III of Madrid in 2010, 2007 and 2005 respectively. Furthermore, he received his M.Sc. in Network Planning and Management at Aalborg University (Denmark) in 2006. Ruben is co-author of more than 70 papers in prestigious international journals and conferences such as ACM CoNEXT, WWW, Usenix Security, ACM HotNets, IEEE Infocom, ACM CHI, IEEE/ACM TON, IEEE TPDS, CACM, PNAS, Nature Scientific Reports, PlosONE or Communications of the ACM. He has been the PI of 13 research projects funded by the EU H2020 and FP7 programs, the National Government of Spain and private companies and overall has participated in 27 research projects. Ruben's main research interests include Online Advertising, Web Transparency, Personalization and Privacy, Internet Measurements and its application to solve longstanding problems in other disciplines such as economy or sociology.