Journal of Network and Computer Applications 176 (2021) 102941

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

On the reduction of authoritative DNS cache timeouts: Detection and implications for user privacy

Tomas Hernandez-Quintanilla [a], Eduardo Magaña [b,c,*], Daniel Morató [b,c], Mikel Izal [b,c]

[a] *Kent State University, Department of Computer Science, Kent, OH, 44242, USA*
[b] *Public University of Navarre, Department of Electrical, Electronic and Communications Engineering, Pamplona, Spain*
[c] *Institute of Smart Cities, Calle Tajonar 22, 31006, Pamplona, Spain*

A R T I C L E I N F O

Keywords:
DNS cache
User privacy
DNS manipulation

A B S T R A C T

The domain name system (DNS) is an Internet network service that is used by hosts to resolve IP addresses from symbolic names. This basic service has been attacked and abused many times, as it is one of the oldest and most vulnerable services on the Internet. Some DNS resolvers conduct DNS manipulation, in which authoritative DNS responses are modified. This DNS manipulation is sometimes used for legitimate reasons (e.g., parental control) and other times is used to support malicious activities, such as DNS poisoning or data collection. Between these DNS manipulation activities, some Internet service providers (ISPs) are changing the DNS cache timeout of the DNS responses with which their DNS resolvers responded to obtain additional data about their subscribers. These data can be a detailed web browsing profile of the user. This approach does not require a large investment and can yield huge benefits if the information is used or sold. Therefore, user privacy is disputed. We conducted a study in which we analyse how ISPs use this DNS manipulation, propose a method for identifying this DNS manipulation by the end-user and determine the amount of information an ISP can collect by using it. We also developed a public web tool, for which the source code is available, that can help Internet users determine whether their privacy is being compromised by their ISP via the exploitation of DNS cache timeouts. This service can facilitate the collection of data on how many people are victims of this abuse and which ISPs around the world are utilizing this technique.

1. Introduction

The domain name system (DNS) provides a fundamental service to Internet users, as it provides mappings from fully qualified domain names (FQDNs) to numerical IP addresses. These mappings are a necessary link that connects human users to the routing information of the Internet. Standards such as DNSsec and DNS over HTTPS (DoH) (Osterweil et al., 2007; Szalachowski and Perrig, 2017; Dickinson, 2020; Lian et al., 2013) have been developed for securing this protocol. However, DNSsec only provides trust in DNS information, and messages are not ciphered. With DoH, messages are ciphered, but its use remains anecdotal (Lu et al., 2019). Maintaining the DNS as it was 40 years ago renders it vulnerable to abuses and attacks (Klein and Pinkas, 2019; Zhao et al., 2007; Pearce et al., 2017) and provides internet service providers (ISPs) with requested DNS domains, which they can exploit to obtain data regarding the actions their clients perform using their Internet service. For example, an end-user who is browsing the Internet will request the DNS resolution of all the DNS domains of the visited web pages.

Most Internet traffic is ciphered (Torres et al., 2017) (for example, using secure web browsing with HTTPS). Therefore, deep packet inspection does not provide much information to ISPs. Some information can still be obtained from un-encrypted headers, for example the internet protocol address of the accessed servers, however, capturing and processing tens or hundreds of gigabits per second present in ISP trunks is highly costly or even unfeasible (Moreno et al., 2015). The DNS service becomes a weak point that can be exploited by ISPs to obtain as much information as possible about the browsing patterns of their subscribers. We have confirmed the existence of ISPs that utilize this type of practice. ISPs own the local DNS, which is configured by default for their subscribers by the DHCP (dynamic host configuration protocol). That local DNS is responsible for resolving all DNS requests sent by

* Corresponding author. Public University of Navarre, Department of Electrical, Electronic and Communications Engineering, Pamplona, Spain.
E-mail addresses: thernan8@kent.edu (T. Hernandez-Quintanilla), eduardo.magana@unavarra.es (E. Magaña), daniel.morato@unavarra.es (D. Morató), mikel.izal@unavarra.es (M. Izal).

https://doi.org/10.1016/j.jnca.2020.102941
Received 10 May 2020; Received in revised form 15 September 2020; Accepted 23 November 2020
Available online 3 December 2020

subscribers; therefore, it is the point at which the DNS responses can be easily modified by the ISP.

The TTL (time-to-live) field of DNS responses is used to realize scalability, as it enables the caching of FQDN information by an intermediate name server or user host, thereby preventing the host from needing to query constantly the authoritative name servers. This TTL controls the update frequency of the DNS records. Do not confuse this TTL field with the field of the same name in the IP header. Increasing TTL is being used by traffic control tools (Drako, 2013), and its use has been studied for additional applications in mobile environments (Wu et al., 2007), in order to reduce server load. Otherwise, decreasing TTL can be used by ISPs without the user's awareness to extract more information about browsing habits, as the user has accepted these conditions in the contract with his ISP. Reducing the TTL value of a DNS response does not substantially affect the Internet latency (Bhatti and Atkinson, 2011), and it does not significantly increase the load for DNS servers (Jung et al., 2002). This behaviour of ISPs has not been exposed in the literature before, and of course, ISPs do not expose publicly what they are doing and why. The only explanation for the modification of TTLs from ISPs that we have found is that it allows them to obtain many more samples of DNS requests from their subscribers. Changing the DNS settings to another DNS server in the client side is not a valid solution for preventing ISPs from collecting information by reducing the TTL of the DNS responses because ISPs can inspect DNS queries to third-party DNS servers and interfere with their responses (DNS transparent proxy). Therefore users are disabled from taking measures to prevent the ISP from compromising their privacy.

By reducing the TTL of DNS responses, ISPs are requiring the client to make more DNS requests, and these DNS request can be used by the ISP to identify, with higher temporal granularity, the sites that their subscribers are browsing. For example, for the domain booking.com (an important travel and reservations web site), the TTL is 86,400; hence, a DNS request would be conducted every 24 h. By modifying the TTL to 60 s, a DNS request will be conducted every minute, which enables ISPs to obtain a better web browsing profile of the user. It is well-know that ISPs are using DNS traffic to make this profiling and that they do not agree with migrating to secure DNS proposals that would block the activity (Ma et al., 2015). The manipulation of TTL can improve this profiling in a very significate way.

Although tools are available that can help advanced users identify DNS servers that manipulate DNS answers to prevent possible attacks (Trevisan et al., 2017), no practical and easy-to-use tool is available that enables regular Internet users to identify possible abuses that involve DNS manipulation. To satisfy this need, we developed a web tool, namely DNSPrivacyTester[1], that can identify possible abuses that involve DNS manipulation not only from ISP DNS resolvers but also from any other resolver that might be changing the TTL values of DNS responses to attack users and violate their privacy. The results demonstrate that it is possible to identify DNS manipulation using the user web browser with high accuracy in less than 3 min and that ISPs are using this TTL manipulation, especially for mobile networks.

The main contributions of this paper are the following:

- We have identified a technique used by ISPs to get more information about the behaviour of their subscribers. To the best of our knowledge, this was not documented previously in the literature. This technique is based in the manipulation of the TTL field in DNS responses and it compromises user privacy.
- We have presented a methodology to enable subscribers to identify when their ISPs are manipulating the DNS responses in such a manner that they could obtain high resolution browsing information. The procedure uses only response times from DNS queries.
- We have developed a public web tool which implements the aforementioned methodology and we made the source code available.
- We have carried out a measurement campaign to validate our proposal and analysed the impact on privacy of Internet users in terms of the amount of information an ISP can collect by using the manipulation of the TTL field in DNS responses.

The remainder of this paper is organized as follows: Section 2 describes the manipulation of DNS responses to compromise user privacy. In Section 3, we explain the system that is used to collect DNS resolution times from end users. Then, in Section 4, we discuss the identification of DNS manipulation using DNS resolution times. In Section 5, we describe our online tool that enables Internet users to check if their DNS has been manipulated and, therefore, if their privacy has been compromised. In Section 6, we discuss results that were obtained using our online tool. In Section 7, we briefly describe related work. Finally, in Section 8, we present the conclusions of this study.

2. Using DNS manipulation to compromise user privacy

2.1. DNS caches

When measuring the impact of changing the TTL of the DNS answers, several factors must be analysed. First, we must identify the types of DNS caches that store information about a domain when accessing an Internet service. An application will always try to retrieve the DNS information of a domain from a cache before making a DNS query to a server, as it is faster and requires fewer resources. Three types of DNS caches are commonly used by Internet applications: the browser or application cache, the operating system (OS) cache and the local DNS server cache (Fig. 1). These three caches store DNS information of domains to reduce request/response times and to reduce the load at the authoritative DNS servers.

As illustrated in Fig. 1, when a client application makes a DNS query for a new domain, the DNS information that is retrieved from the authoritative DNS server of the domain and served to the querying device by its local DNS server is stored in both the OS DNS cache of the device and the local DNS server that forwarded the answer. Both can store the information about the domain for the same amount of time, which is specified in the TTL field of the DNS answer, but this behaviour at the OS cache depends on the type of OS and the application. Once the answer has been cached by the OS, the client application can access the DNS information and also cache it for a duration that depends on the application that is being used.

In the path between the authoritative DNS server and the local DNS server, there can be intermediate DNS servers. The cache of the local DNS server that forwards the DNS response stores the DNS information of the domain. The client application or host does not have any way to control the local DNS server cache; hence, this server can change the TTL of the DNS answers without the client's consent. Any intermediate DNS server could change the TTL, but the effect would be the same for the

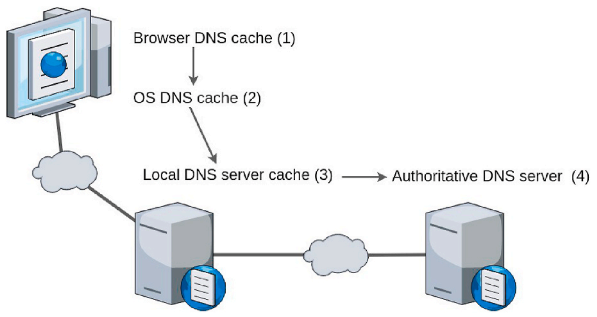


Fig. 1. Agents that are involved in the task of answering a DNS query for a web browser in the client.

[1] Available online at http://DNSPrivacyTester.tlm.unavarra.es.

end user. This local DNS server can store a DNS response for as long as the TTL indicates and modify this TTL in the responses that are delivered to the client host. The modified TTL is provided only to the client host; hence, subsequent requests from the client host can be resolved by the same local DNS server without increasing the load of the authoritative server of a domain. This cache plays an important role in our study since when we want to determine whether a local server is changing the TTL of a DNS answer, the DNS resolution times will depend on this cache.

Fig. 2 illustrates a possible scenario: a device is accessing continuously a specified Internet DNS domain with an authoritative TTL of 10 min. By default, in a normal scenario (Fig. 2a), the device would make a DNS request every 10 min, thereby honouring the authoritative DNS TTL. The device would cache the DNS information for these 10 min. However, in the same scenario but with the local DNS server changing the DNS TTL to 1 min, the device would honour the modified DNS TTL and, hence, would make a DNS request each minute (Fig. 2b). In this case, the local DNS server would cache the domain for the authoritative TTL, thereby reducing the number of DNS requests to one each 10 min to the authoritative DNS server. With this DNS manipulation, the number of DNS requests that are submitted from the device and received by the local DNS server that is provided by the ISP would increase from 1 to 10 requests each 10 min.

The operating system caches in typical operating systems (for example, Windows and macOS) will respect the DNS TTL values that are specified by the local DNS server; hence, they do not affect the normal behaviour of the protocol when querying a domain name. In Linux distributions, there is no OS-level DNS caching unless extra services, such as *NSCD* or *DNSmasq*, are enabled. The name server caching daemon (*NSCD*) has a default cache time of 15 min and ignores any TTL values that are specified. The DNSmasq service uses the TTL value to determine the cache expiration time. In most cases, these services are not active in Linux by default; therefore, the DNS TTLs that are specified by a DNS server are not honoured. Then, the DNS cache that is implemented at the user application determines the duration for which DNS records are stored. For browsers, this cache time is always fixed.

For mobile devices, the behaviour differs. In Android systems, OS-level DNS caching is not conducted (as it uses a Linux kernel), and applications cache the DNS records by default. Applications in Android typically cache the DNS records by honouring the authoritative TTL values (in contrast to applications in Linux), and if the TTL value is lower than 60 s, the application cache stores the DNS records for a minimum of 60 s. In iOS systems, there is no OS DNS cache or browser DNS cache; hence, these systems completely rely on the DNS servers and their caches.

For desktop devices (Windows, macOS, and Linux), the client's application DNS cache typically alters the normal behaviour of the DNS, and the duration for which it stores the DNS responses depends of the client application that is being used. We will consider web browsers as the main examples of client applications.

Table 1 presents the periods of time for which several of the most popular browsers (Statcounter, 2020) enforce cache DNS records by default, which depend on the operating system that is being used. Most of these browsers cache results for 60 s or more, and this time is fixed independently of the TTL in the DNS response. This table was constructed by analysing the DNS traffic of devices when various browsers

Table 1
Most popular browsers' DNS cache times.

Operating system	Browser	Application DNS cache (seconds)
Windows 10 1903	Google Chrome 76.0.3809.132	60
	Opera 63.0.3368.84	60
	Mozilla Firefox 68.0.1	60
	Microsoft IE 11.799.17134.0	1800
	Microsoft Edge 44.18362	1800
macOS 10.14.6	Chrome 76.0.3809.100	60
	Mozilla Firefox 68.0.1	60
	Safari 12.1.2	0
Android 9.0	Google Chrome 76.0.3809.132	original TTL, with a minimum of 60
	Firefox 68.0.1	original TTL, with a minimum of 60
	Samsung Internet 9.4	original TTL, with a minimum of 60
iOS 12.4	Chrome 76.0.3809.100	0
	Mozilla Firefox 68.0.1	0
	Mobile Safari 12.1.2	0
Linux Ubuntu 18.04	Chromium 76.0.3809.100	60
	Firefox 68.0.1	60

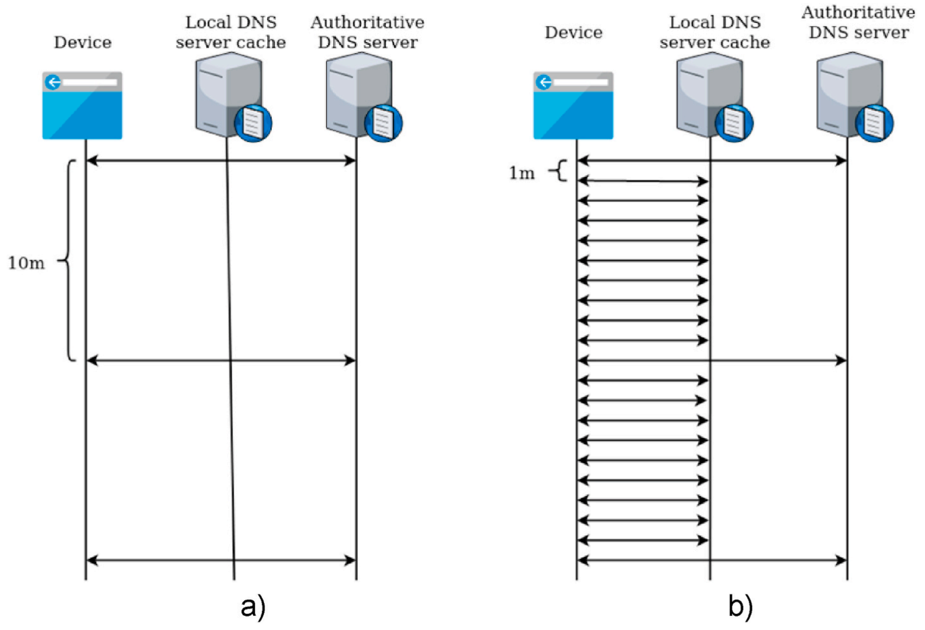


Fig. 2. DNS queries that are performed a) in a normal scenario with an authoritative TTL of 10 min and b) in a scenario with DNS manipulation, in which the TTL is changed to 1 min at the local DNS server.

were being used, and it is consistent with the data that were gathered in a previous study (Klein and Pinkas, 2019). To obtain these results, we configured a device with each of the operating systems and with each of the web browsers to use a local DNS server that was under our control using the DNS server software named Bind. In the configuration of Bind, we specified that it was a forwarding DNS server and that the TTL for every domain would be set to 0 in the DNS response that was sent to the clients. Then, we queried a specified web address each second for a whole day for each combination of operating system and web browser, and we checked how many queries were resolved by the browser's cache before querying the authoritative DNS server.

The use of a DNS cache at the application level (browser) has been reported several times (Cohen and Kaplan, 2001), and there is an ongoing debate regarding whether it is useful or not. The behaviour we observed during our research is that when the browser or other client application accesses a resource on the Internet, it conducts a query regarding the domain. Then, the information from the response is cached in the operating system cache, and the browser accesses it and stores this information for a minimal period of time, regardless of the TTL value that is specified in the DNS response. When the information that is stored in the browser cache expires, it accesses the operating system cache again until the DNS information that is stored in the operating system's cache expires.

This client application cache limits the effect of reducing the TTL value of the DNS responses because even though the ISP DNS servers can reduce it to as low as 0 s (to disable the DNS cache), the responses are cached in the client application cache for a minimum time regardless of the TTL. Therefore, new DNS requests will be conducted only when the browser's cache expires (in most cases, every minute, as specified in Table 1), thereby limiting the effectiveness of this technique.

Fig. 1 illustrates the sources of DNS information regarding a domain that an application may query to access an Internet service. Of the three caches it can access, the only cache that can have a fixed TTL value should be the browser's cache. Both the OS cache and the local DNS server cache should expire according to the TTL value that is specified by the authoritative server in the DNS response. The minimum time that a DNS response will be cached by a browser that made a DNS query will be determined by the browser's DNS cache.

2.2. Changing of the DNS TTL against the user's privacy

The information that an ISP can extract by analysing the DNS traffic is independent of the encryption that other protocols such as TLS or HTTPS provide at the application layer. The information that is provided by DNS queries is easy to analyse and does not require complex algorithms to be processed, as the domain name that is visited by the user is specified in the DNS request packets that are sent to the local DNS servers. The ISP typically provides this local DNS server to its subscribers, and the DNS logs can be collected and analysed easily. This information can be used to construct a user's Internet browsing profile, which can be used for lucrative purposes or even sold to third parties.

It is not technically possible to prevent ISPs from extracting data about their clients through the DNS, except by changing the local DNS server that is in use (e.g., to the Google public DNS or the Cloudflare public DNS). Nonetheless, the DNS resolver that is selected for resolving the queries could still collect information about their clients. However, changing the TTL value of the DNS responses and, therefore, altering the normal behaviour of the Internet is an active method for information collection that is conducted without the user's awareness. When using a technique of this type, the privacy of Internet users is compromised, and their data are exposed.

The simplicity and low cost of reducing the TTL value of DNS responses to obtain better information about the users of a network render this technique suitable for other environments, such as those with open Wi-Fi networks. These networks include locations such as malls, parks or other public places in which the technique, when applied, leaves the user's privacy exposed and enables DNS server administrators to collect information about any person that connects to the network. The collected information can depend on the location of the client; therefore, it can be used to provide more personal and targeted services to the client, such as showing ads or promoting services with special conditions that are selected according to the websites that the client has browsed.

The DNS TTL modification can be conducted by using transparent DNS proxies even if we do not control the local DNS server (Vavrusa and Grant, 2018; Kührer et al., 2015; Liu et al., 2018). These proxies are being used by several ISPs to intercept all the user's DNS lookup requests and to transparently proxy and cache the results. This forces the users to use the transparent DNS proxy, even if they have changed their DNS settings to use an 'open' DNS server or any other local DNS server. Hence, changing the DNS settings at the client side is not a solution for preventing ISPs from collecting information by reducing the TTL of the DNS answers.

As has been discussed previously, several strategies have been developed for gathering information about the user through the DNS (García-Dorado et al., 2018; Kim and Zhang, 2015; Chitpranee and Fukuda, 2013). These strategies obtain the chain of the main websites that have been visited by the user while ignoring the domains that host secondary content, such as ads, images, and social network widgets. They use the DNS requests that have been conducted over time to identify the browsing chain. Not all web requests generate DNS requests due to the use of the caches; hence, estimations are conducted by considering TTL of each requested domain. In addition, modern browsers use prefetch and preload techniques (Monrose and Krishnan, 2010), which generate ghost requests and their corresponding DNS requests, thereby complicating the identification. These strategies focus mostly on extracting information regarding the number of times a user visits a domain that can be clearly differentiated when accessed (because a new DNS request is generated each time). The same inference with popular domains is highly difficult because a DNS request can be conducted via the secondary content of another website (for example, a Google maps widget on a commercial web site). The accuracy of these systems can be improved significantly if the DNS TTL can be selected. For example, if all domains would have the same TTL, websites that have been visited the same number of times would generate the same number of DNS requests. The state-of-the-art identification strategies would be able to collect information about the number of times a user visits both popular domains and less popular domains, and all the websites would make DNS queries for their root domain records more frequently. This would enable a simpler identification system that would count a visit whenever it would receive a DNS query for the root domain of a webpage.

2.3. Most common TTLs in DNS domains

To analyse the possible impact of reducing the TTL value of DNS responses, we examined the TTL values of the most popular Internet websites. Fig. 3 displays the cumulative distribution function (CDF) of the TTL values for the 10,000 most popular websites on the Internet according to the Alexa Top 1 million sites on the Internet ranking from September 2019 (Alexa, 2020). We collected these data by directly querying the authoritative name servers of each domain. Approximately 75% of these domains have a TTL that exceeds 300 s. For these domains, one DNS request each 5 min would be intercepted by the ISP without TTL modification. Almost 90% of the domains have TTLs that exceed 60 s; hence, the technique of reducing the TTL of DNS answers is useful even in the presence of the application cache (typically 60 s).

Fig. 4 presents the CDF of TTL values of Internet domains that represent 44.26% of the Internet's web traffic, according to the SimilarWeb top-50 websites in the Internet ranking (SimilarWeb, 2020). This graph has been constructed by identifying the authoritative TTL values of these 50 domains, multiplying each of these values by the percentage of web traffic they cursed and dividing the result by the total

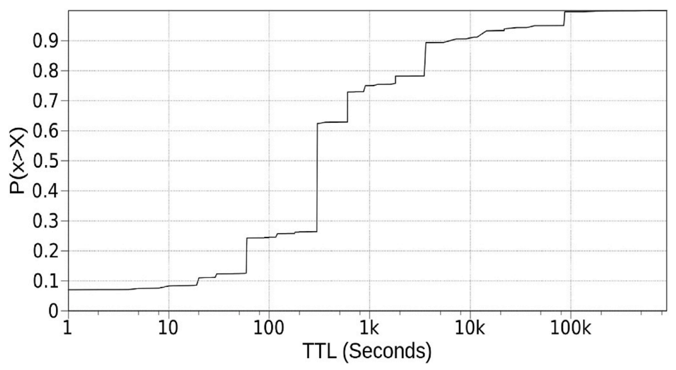


Fig. 3. CDF of the TTLs of DNS answers that were generated by querying the authoritative DNS servers of the 10,000 most popular websites on the Internet according to the Alexa ranking.

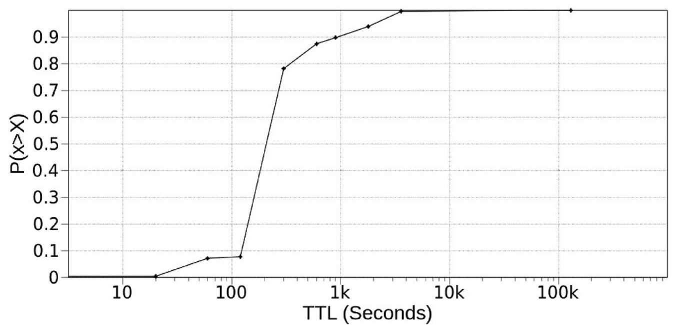


Fig. 4. CDF of the TTLs of the top-50 domains, which represent 44.26% of Internet traffic, in terms of traffic contribution, according to the SimilarWeb ranking.

amount of traffic they all shared. Table 2 presents the twenty most representative domains by traffic from the SimilarWeb classification and their TTLs from September 2019.

According to Fig. 4 and Table 2, approximately 90% of the domains with more traffic have TTL values of 300 s or longer. Therefore, reducing the TTL values of the DNS responses that contain the DNS records of these domains can increase by 5 times the number of DNS requests. This increase can provide 5 times more information about a user's navigation profile, and it can increase the accuracy of the systems that are used to analyse user traffic.

Table 2
Traffic shares and TTL values of the 20 most visited domains of the Internet according to SimilarWeb.

Domain ranking position by traffic	Domain	Traffic share	TTL (s)
1	google.com	18.18%	300
2	youtube.com	7.06%	300
3	facebook.com	5.80%	300
4	baidu.com	1.46%	600
5	wikipedia.org	1.19%	600
6	yahoo.com	0.96%	1800
7	twitter.com	0.92%	1800
8	pornhub.com	0.79%	3600
9	instagram.com	0.73%	60
10	yandex.ru	0.72%	300
11	xvideos.com	0.69%	300
12	xnxx.com	0.59%	300
13	ampproject.org	0.58%	300
14	amazon.com	0.57%	60
15	live.com	0.57%	3600
16	vk.com	0.51%	900
17	netflix.com	0.47%	60
18	qq.com	0.40%	600
19	mail.ru	0.40%	60
20	whatsapp.com	0.39%	300

A previous study of a dataset of DNS queries that were made by 85 clients (Callahan et al., 2013) yielded similar results: more than 70% of the dataset's Internet traffic was conducted by connections to domains with TTL values that exceeded 100 s. The diversity of the TTL values among the domains in the Internet reduce the ISPs' performance in characterizing their clients' traffic. Therefore, having a unique, low TTL value that is for all the domains that their servers resolve is enormously helpful for characterising their clients' traffic.

2.4. Performance problems that are due to TTL reduction

The reduction of DNS TTL results in related performance problems in the client side: increases in the amount of DNS traffic and in the delay in client applications. This is because DNS requests/responses are small packets and they are sent by or to a local DNS server over a relatively short distance; hence, the RTT is typically on the order of milliseconds. However, these problems are not significant.

In the server side, the performance problems will be related to the increase in the number of received DNS requests. These requests can be more bursty due to DNS request synchronization. If the TTL is reduced by a factor of 10, the number of DNS requests and, therefore, the load in the local DNS server are increased by a factor of 10. Typically, local DNS servers attend to relatively few clients; hence, this scalability is possible. If this increase would affect to root DNS servers or authoritative DNS servers, the scalability problems would be more severe, but this does not occur because the TTL modification is conducted at the local DNS server or in an intermediate DNS transparent proxy. Those use cache systems, and they respond to the increasing rate of DNS requests while not affecting root DNS servers or authoritative DNS servers.

For a bursty load, in a scenario with a TTL modification to 10 s for all domains, DNS cached domains would expire at the same moment for almost all domains, thereby requiring the local DNS server to attend to many queries in a short period of time and, thus, to require more resources than usual to perform its tasks. As most ISPs have no interest in increasing the loads at local DNS servers, they typically change the TTL of the DNS responses at an intermediate local DNS server (a local DNS server or a transparent DNS proxy) so that the ISP can scale more easily the intermediate server.

This technique also prevents a domain owner from detecting the use of this technique, as it cannot determine whether a local DNS server is changing the TTL of the DNS answers it serves. This renders our study more difficult as we can only demonstrate that there are ISPs that are reducing the TTL of the DNS answers by collecting data from the client side. Low TTL values for a domain were already being used by domain owners to facilitate domain migrations, as the DNS records would expire faster and, therefore, the user would obtain the new mapping from the FQDN to the new IP address faster. The use of this technique could explain some cases in which the TTL values of domains are sporadically lower. In contrast, in the case of DNS manipulation, we observed that TTL changes occurred in some ISPs for all DNS responses indiscriminately and over time.

2.5. Threat model

To analyse threats in a typical residential access provided by an ISP, we employ a widely used systematic approach to threat modeling, called STRIDE (acronym for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privilege) which was created by Loren Kohnfelder and Praerit Garg (Kohnfelder and Garg, 1999; Hernan et al., 2006).

It is assumed that the ISP is not trying to impersonate/spoof the user, tamper user service data, infiltrate in user systems or disrupt the service in any way that would make the user just use another ISP. The ISP is in a perfect man-in-the-middle position and could use this advantage to attack the users but these attacks are out of the scope of this work.

The DNS model usually puts the ISP as a proxy for external DNS servers, as shown in Fig. 5. The single head arrows describe the behaviour that the DNS service is meant to follow, a request is sent by the client, if needed the ISP's DNS server queries the Global DNS service for the DNS answer, and then the DNS server sends the response to the client. As DNS information stays rather stable, this interaction is refreshed regularly, governed by the expiration of timers whose values are decided by global DNS servers.

In this scenario, the relay of DNS queries poses a threat of user information disclosure as it provides the ISP with the domain names the client is requesting to be resolved. This may be used to perform client profiling for commercial purposes.

Moreover the ISP can tamper with DNS data coming from global servers before relaying it to the users. This may be stopped by the users by configuring strong authentication from global DNS servers (DNSSEC) or by using cyphered communication to its own trusted DNS servers (for example using DoH) instead of using the ISP provided DNS servers. However most residential users don't worry enough about privacy or lack technical expertise to know the threat and to carry out any of these remedies.

By tampering DNS data, the ISP can achieve denial of service to clients by providing wrong responses to some queries. This would cause service disruption to the users, which is not in the interest of the ISP, whose payed job is precisely providing correct network connectivity. Thus, it is used only in extreme cases like when asking for recognised malware sites or for accomplishing legally-requested mandatory blocking of piracy sites.

A more useful tampering for ISPs would be to interact with DNS expiration dates. It is well known that the DNS service is being used by ISPs to profile their users (Yan and Lee, 2020). Reducing the TTL value of DNS records in the responses results in more frequent requests from the users and therefore more information that could be used to improve a profile about clients, increasing the already existing threat for information disclosure. It is not a new type of information that is gained but the frequency of this information that is increased. Gathering more frequent information allows a better profiling, more invasive of user privacy. In a non-compromised scenario, maybe the only information that the ISP can obtain is that the user accessed a web site at least once in a day; however, decreasing the DNS TTL to 5 min it could know how many times he visited this web site and when in the day it did so, with 5 min resolution.

DNS response modifications could be observed by the client using access link monitoring. However, it does not affect general perceived performance, thus a typical residential internet user with no technical background is unlikely to notice it.

3. Online tool for obtaining DNS resolution times

We are interested in detecting when the privacy of an end user has been compromised via modification of the TTL in DNS responses. The easiest way for a user to check for DNS-TTL manipulation would be to use the typical utilities for DNS resolution in the command line interface

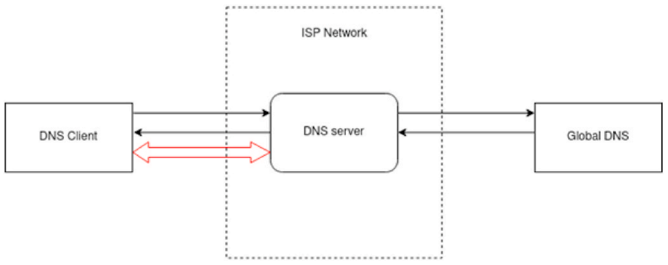


Fig. 5. Components of the system being analysed for the threat model.

(such as the *dig/nslookup* command tools) to request a specified domain from the client side and to examine the TTL value of the DNS answer that is delivered by the local DNS server. This value can be compared with the authoritative TTL (which is obtained without using DNS requests that could be also modified). If they do not match (considering the remaining time in the cache), it is assumed that a local DNS server is altering the DNS service. An executable or mobile app could be developed for collecting this information, but this is not practical as most people are reluctant to download a tool with a single function that is executed in a short period of time. We require a tool for collecting the DNS resolution times of end-users across the Internet as easily as possible.

A more convenient strategy for collecting massive DNS resolution times from end-users was to develop a web service that would not require to the user to download files and that could be used from any device. This web service is extended in next sections to detect whether the privacy of a user is being compromised, and it can report the results in a short period of time.

The first problem we faced in developing this tool was that is not possible to conduct DNS queries from a web page or to access the TTL information of DNS responses from a web browser. Commercial web tools are available that request resources for which records are allocated to a DNS server that they control and retrieve for users the information that was collected by their DNS servers. However, our objective was to prove that Internet users' privacy was being compromised by DNS servers that ISPs control. Most local DNS servers that are controlled by ISPs are private; hence, directly querying them was not a viable strategy for determining whether they were altering the DNS service. ISP subscribers are the only entities that can provide data on whether the ISPs are altering the DNS service or not. Therefore, a system that collects data from the subscribers had to be developed.

The end-user (ISP subscriber) will access a webpage with his web browser. Then, the web browser of the end-user must request additional web resources periodically over a period of time and capture the DNS resolution times that they required for loading. To realize this, we created a JavaScript code that was embedded in the webpage that requests through AJAX a URL (uniform resource locator, which characterizes a full webpage path) with a polling interval (typically 10 s). To bypass the browser's cache we used different URLs in every request. Via this approach, for each request, a different TCP connection to the server will be used (with no persistent connections), along with a new DNS query, if necessary.

The DNS resolution times of the requested resources are displayed in the developer tools network menu that most browsers feature; however, we must automate the recollection of these data. Through a JavaScript API, namely, PerformanceResourceTiming (Grigorik, 2020), we can access these resolution times using the values of the *domainLookupStart* and *domainLookupEnd* fields that are returned by the API. This API is still in a development phase and behaves differently among browsers. In some browsers, such as Mozilla Firefox, the DNS resolution times that are specified by the API are not exactly the same as those that are shown in the developer tools network tab. Other browsers, such as Safari, do not fully support the API, and others have such high DNS caching times (e.g., Internet Explorer or Edge) that if we want the service to deliver the result of the test quickly, their use is not a viable option. The browser that procured the best results was Google Chrome. Its caching time is low (only 1 min), it is available for all OSs and platforms, and the resolution times that are specified by the API are more accurate than those that are specified by other browsers (instead of delivering the DNS resolution times in milliseconds, it delivers them in microseconds). This provides us with more reliable information about the DNS resolution times and, therefore, about which cache source is queried to obtain the DNS record of a domain at each moment.

The web service was spread among Internet users, mainly in Spain. More than ten thousand tests were conducted by more than one thousand users during August and September of 2019. The tool requests a random URL within the test domain every 10 s for up to several hours, depending on the time for which the user keeps the webpage open. The test domain has an authoritative TTL of 180 s. Table 3 presents the total number of Internet users that used our tool and the number of samples that were collected from them. A sample consists of each DNS resolution time that was observed while a user was conducting the test. As we allowed the users conduct the test an unlimited number of times, many users conducted it for several minutes; hence, the number of provided samples differed among users in the research. Desktop users conducted longer tests than mobile users, and for example the number of samples from macOS is larger than that from Android, although the number of users of the former is smaller.

4. Identification of DNS TTL manipulation

4.1. DNS behaviours per operating system

Let's consider a scenario in which the domain that is being used while gathering data has an authoritative TTL of 180 s. Suppose the client is using a desktop application (for example, a browser) that has an application cache timeout of 60 s (e.g., the Google Chrome browser, as shown in Table 1). The application requests the resolution of the domain every 10 s, and it continues to send the requests for 180 s (the authoritative TTL). We will analyse a **normal scenario** (without TTL modification) and a **privacy-compromised scenario** (with TTL modification), along with their variants, according to the peculiarities of each OS.

In a normal scenario and an OS DNS cache (Windows or macOS), we should obtain:

- 15 queries to the DNS browser cache: using a 60-s browser cache, 5 out of 6 requests are resolved by this cache.
- 2 queries to the OS DNS cache: with a cache timer value at this level equivalent to the authoritative TTL, from the 3 requests that are not resolved in the browser cache (one every 60 s), 2 are resolved by the OS DNS cache.
- 1 query to the authoritative DNS server through the local DNS server: only 1 request every 180 s is not solved by previous caches.

This behaviour is represented in Fig. 6. In total, approximately 83.33% of the DNS resolution times should correspond to the browser's DNS cache, 11.1% of the resolution times should correspond to the OS DNS cache and 5.56% of the resolution times should correspond to DNS queries that were made to the domain authoritative servers.

In a normal scenario but without an OS DNS cache and with an application DNS cache that is equivalent to the original TTL (Android devices), we should obtain:

- 17 queries to the DNS browser cache: as Android application DNS caches honour the TTL of DNS answers, in a 180-s interval, this cache is accessed for 170 s.
- 1 query to the authoritative DNS server through the local DNS server: only 1 request every 180 s is not responded to by the application's cache.

Those queries are represented in Fig. 7. In total, approximately 94.44% of the DNS resolution times should correspond to the browser's DNS cache, and 5.56% of the resolution times should correspond to DNS queries that were made to the authoritative servers of the domain.

Table 3
Numbers of users and collected samples per operating system.

Operating system	Number of users	Number of collected samples
Windows	1131	206,495
macOS	215	61,589
Android	960	24,023
Linux	266	60,817
iOS	77	1104

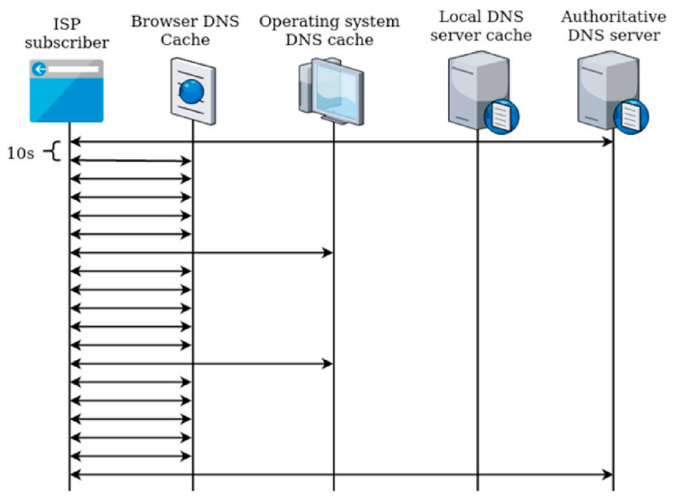


Fig. 6. DNS sources that are queried by Windows and macOS devices in a normal scenario.

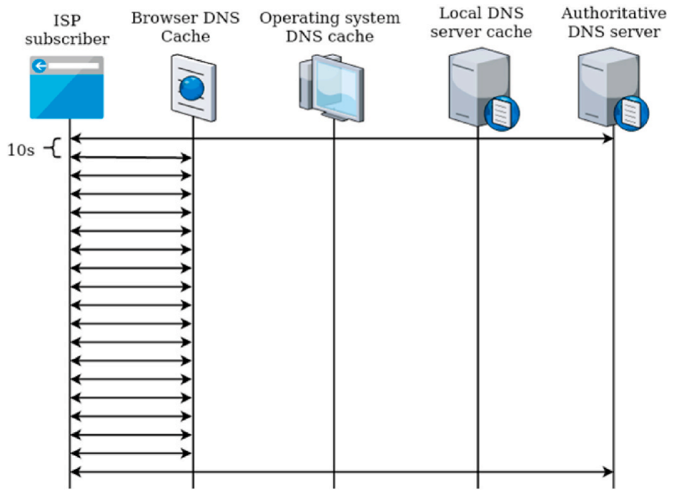


Fig. 7. DNS sources that are queried by Android devices in a normal scenario.

In a normal scenario without an OS DNS cache and with a browser DNS cache of 1 min (the Linux case), we should obtain:

- 15 queries to the DNS browser cache: with a 60-s browser cache, 5 of each 6 requests are resolved in this cache.
- 2 queries to the local DNS server cache: with a cache timer value at this level equivalent to the authoritative TTL, from the 3 requests that are not resolved in the browser cache (one every 60 s), 2 are resolved by the local DNS cache.
- 1 query to the authoritative DNS server through the local DNS server: only 1 request every 180 s is not solved by previous caches.

Those queries are represented in Fig. 8. In total, approximately 83.33% of the DNS resolution times should correspond to the browser's DNS cache and 11.1% to the local DNS server cache, and 5.56% of the resolution times should correspond to DNS queries that are made to the authoritative servers of the domain.

In a normal scenario without an OS DNS cache or a browser DNS cache (the iOS case), none of the DNS queries are solved by the device, and we should obtain:

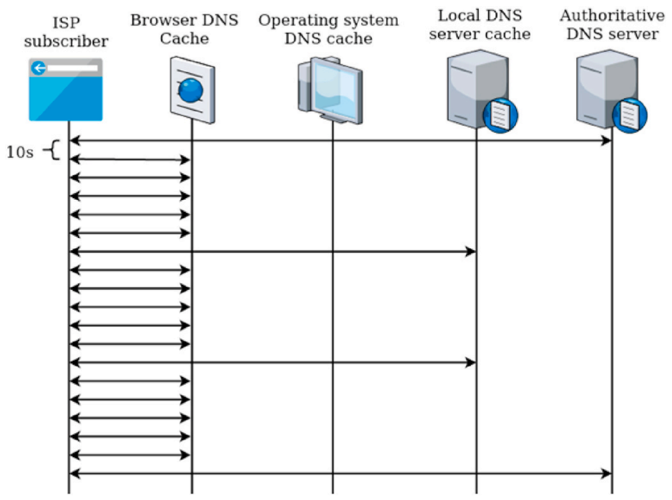


Fig. 8. DNS sources that are queried by Linux devices in a normal scenario.

- 17 queries to the local DNS server cache: the queries that are made every 10 s would only be solved by this DNS record source.
- 1 query to the authoritative DNS server through the local DNS server: only 1 request every 180 s is not solved by previous caches.

In this scenario, which is illustrated in Fig. 9, 94.44% of the DNS resolution times should correspond to the local DNS server cache, and 5.56% of the resolution times should correspond to DNS queries that are made to the authoritative servers of the domain.

We can generalize to any authoritative TTL (TTL_{auth}), any polling interval (t_{poll}), any browser cache time ($t_{browser}$) and the use of an OS DNS cache (OS = 1 if used and OS = 0 otherwise). For iOS, the lack of a browser cache is represented by $t_{browser} = 0$.

We define t_{local} as the maximum value between the browser cache time and the polling interval, that will represent the time interval between DNS requests from the client to the local DNS server. We define A' as the maximum time during which there will be no requests to the local DNS server when abiding by the authoritative TTL, rounded to polling intervals.

$$t_{local} = max\left(t_{browser}, t_{poll}\right) \tag{1}$$

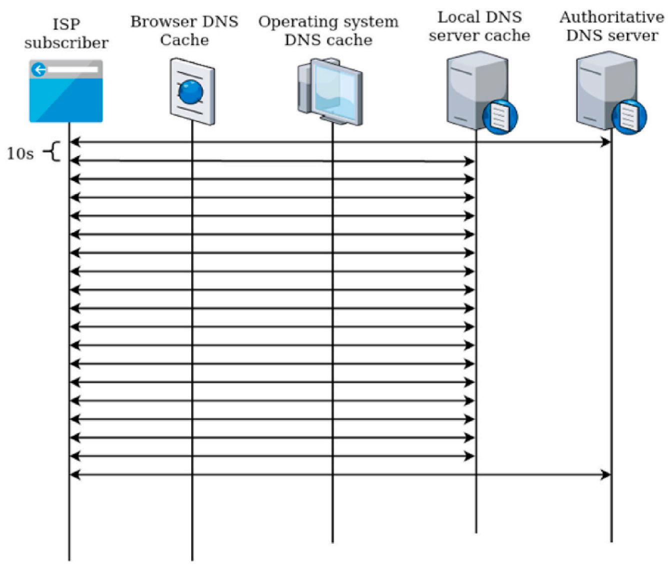


Fig. 9. DNS sources that are queried by iOS devices in a normal scenario.

$$TTL'_{auth} = \left\lfloor \frac{TTL_{auth}}{t_{poll}} \right\rfloor t_{poll} \tag{2}$$

Ratio of queries to the DNS browser cache

$$R_{browser} = \frac{\left\lfloor \left(TTL'_{auth}/t_{local}\right) \left(t_{local}/t_{poll} - 1\right) \right\rfloor}{TTL'_{auth}/t_{poll}} \tag{3}$$

Ratio of queries to the OS DNS cache

$$R_{os} = OS \frac{\left(\left\lceil TTL'_{auth}/t_{local} \right\rceil - 1\right)}{TTL'_{auth}/t_{poll}} \tag{4}$$

Ratio of queries to the local DNS cache

$$R_{local} = (1 - OS) \frac{\left(\left(\left\lceil TTL'_{auth}/t_{local} \right\rceil - 1\right)\right.}{TTL'_{auth}/t_{poll}} \tag{5}$$

Ratio of queries to the authoritative DNS cache

$$R_{auth} = \frac{t_{poll}}{TTL'_{auth}} \tag{6}$$

A normal scenario will be characterized by the number of occurrences of each type of request as described in equations (3)–(6).

In a privacy-compromised scenario (in which the ISP changes the TTL), the sources of DNS responses change according to the modified TTL. For a TTL that has been changed to less than the default browser cache time (1 min typically), the OS cache is not used as a DNS information source as it always expires before it receives a new request. For example, if the modified TTL is 10 s, we should obtain:

- 15 queries to the DNS browser cache: with a 60-s browser cache, 5 of each 6 requests are resolved in this cache.
- 2 queries to the local DNS server cache: since the modified TTL is too low, the OS DNS cache expires before a new request is received from the browser, and the device queries the local DNS server again.
- 1 query to the authoritative DNS server through the local DNS server: only 1 request every 180 s is not solved by previous caches.

Hence, approximately 83.33% of the DNS resolution times should correspond to the browser DNS cache, 11.1% of the resolution times should correspond to queries that are resolved by the local DNS server cache, and 5.56% of the resolution times should correspond to DNS queries that are made to the authoritative servers for the domain information. Fig. 10 illustrates the sources that a device would query in the

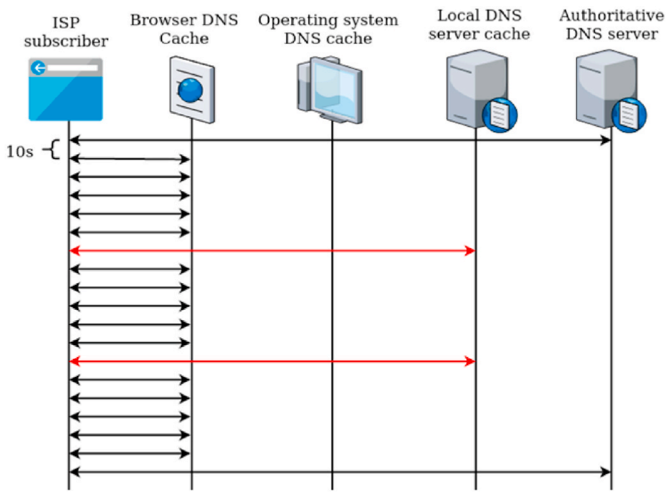


Fig. 10. DNS sources that are queried by a device in a privacy-compromised scenario.

privacy-compromised scenario. Comparing Figs. 6–9 and Fig. 10, we can identify the differences between the scenarios. For a privacy-compromised scenario with a modified TTL that exceeds the default browser cache, for example, 80 s, intermediate values that are between those of the previous scenarios are obtained.

Let's denote the modified TTL as ($TTLm_{auth}$). We define $TTLm^{auth}$ as the maximum value between the modified TTL and the polling interval that will represent the time interval between DNS requests to the authoritative DNS server (equation (7)).

$$TTLm'_{auth} = max\left(TTLm_{auth}, t_{poll}\right) \tag{7}$$

We can generalize to any modified authoritative TTL ($TTLm_{auth}$), any polling interval (t_{poll}), any browser cache time ($t_{browser}$) and the use of an OS DNS cache (OS = 1 if used and OS = 0 otherwise). For iOS, the absence of a browser cache is again represented by $t_{browser} = 0$. A compromised scenario will be characterized by the number of occurrences of each type of request computed by equations (8)–(11).

Ratio of queries to the DNS browser cache	$Rm_{browser} = R_{browser}$	(8)
Ratio of queries to the OS DNS cache	$Rm_{os} = OS\ \frac{\lceil\left(TTLm'_{auth}/t'_{browser}\right) - 1\rceil\ \left(\lceil TTL'_{auth}/t_{local}\rceil - 1\right)}{\left(TTL'_{auth}/t_{poll}\right)\ \left(\lfloor TTLm'_{auth}/t_{local}\rfloor + 1\right)}$	(9)
Ratio of queries to the local DNS cache	$Rm_{local} = \frac{\lceil TTL'_{auth}/t_{local}\rceil - 1}{TTL'_{auth}/t_{poll}} - OS\ \frac{\lfloor TTLm'_{auth}/t'_{browser}\rfloor\ \left(\lceil TTL'_{auth}/t_{local}\rceil - 1\right)}{\left(TTL'_{auth}/t_{poll}\right)\ \left(\lfloor TTLm'_{auth}/t_{local}\rfloor + 1\right)}$	(10)
Ratio of queries to the authoritative DNS cache	$Rm_{auth} = R_{auth}$	(11)

This characterization will be used to identify when a user is in a normal or a privacy-compromised scenario. For operating systems with an OS DNS cache, DNS requests to the local DNS server (Fig. 10) will be essential for identifying a privacy-compromised scenario.

Using these formulas, we extracted the percentages of queries to each DNS resolution source for TTLs of 180, 300 and 3600 s. For simplicity, we established a polling interval of 10 s for all cases, and we set the value of the modified TTL to 10 s. The obtained results for the normal and privacy-compromised scenarios are presented in Table 4.

We collected data from over a thousand users who used our privacy tester tool (Section 3), some of whose DNS responses were being manipulated and others whose privacy was not being compromised. The tool probes every 10 s during an interval of up to several hours, which depends on the user. The test domain had an authoritative TTL of 180 s. Fig. 11 presents the complementary cumulative distribution functions (CCDFs) of the DNS resolution times of five random users whose privacy is not being compromised (because they follow patterns such as that in Figs. 6–9), each of whom is using a different operating system (with or without an OS DNS cache). The distributions show drops down in various time instants for each user that are related to the 4 DNS times that we discussed previously (browser, OS, local DNS and authoritative DNS).

In Fig. 11 a, for Windows, 3 drops down are observed, which correspond to 3 sources of DNS resolution: (from left to right) the browser DNS cache (at 20us), the OS DNS cache (at 0.8 ms) and the authoritative DNS server through the local DNS server (at 70 ms). The percentage of DNS queries that are resolved by the sources of DNS information for all samples that were collected by our online tool are presented in Table 5. This table compares the experimental results with the theoretical results that are obtained by applying expressions (1–11). The percentage of samples for each source matches the theoretical value that was calculated previously, thereby supporting our hypothesis.

The largest difference between the theoretical and experimental results is observed for macOS. This is attributed to the macOS DNS cache being so efficient that the difference between the DNS resolution times when accessing it versus the browser's DNS cache is small; hence, we might have counted some of the resolutions that corresponded to the browser's cache as OS resolutions. However, the sum of the two corresponds to 94.35% of the resolutions, which is very close to the corresponding theoretical result, namely, 94.45%.

The first two drops down in the CCDF are not exactly at the same timestamps for the users because their computers differ in terms of CPU power, operating system and browser. However, the time scales are highly similar: the browser DNS cache is on the order of tens of microseconds and the OS DNS cache is on the order of milliseconds. The resolution times for the authoritative DNS server are similar for the users, and the values are on the order of tens of milliseconds.

For Android and macOS users, only two drops down can be observed. In Android, the browsers rely completely on their own DNS cache. In macOS, the DNS resolution times when accessing the operating system cache are so low that they cannot be distinguished from accesses to the browser cache. According to Fig. 11, neither iOS nor Linux systems have by default an OS DNS cache that honours a domain TTL value, as we demonstrated previously. As both systems will query local DNS servers whenever the browser DNS cache expires, we cannot determine whether a client's ISP is modifying the TTL of DNS answers when using one of these operating systems.

Fig. 12 presents the CCDF of the DNS resolution times of an example user whose privacy is being compromised (the ISP is changing the TTL to 10 s). For Windows and macOS users, the privacy-compromised profiles are almost equal because there is no OS DNS cache affecting Windows, according to Table 4. The "desktop user" curve of Fig. 12 can be applied to both. In this case, there are also 3 drops down that correspond to 3 sources of DNS resolution: (from left to right) the browser DNS cache, the local DNS cache and the authoritative DNS server. The percentage of samples for each source matches the theoretical values that were calculated previously. For the mobile user of Fig. 12, there are 3 drops down that correspond to the same sources of DNS records as in Fig. 11, whereas in a normal scenario, there should only be 2. Fig. 12 also shows

Table 4
Ratios of queries to each DNS information source according to the values of the domain TTL, the browser cache and the presence of an OS DNS cache.

			Browser	OS		Local		Authoritative
TTL_{auth}	$t_{browser}$	OS	Normal & Compromised	Normal	Compromised	Normal	Compromised	Normal & Compromised
180	60	1	83.33%	11.11%	0.00%	0.00%	11.11%	5.56%
	60	0	83.33%	0.00%	0.00%	11.11%	11.11%	5.56%
	0	1	0.00%	94.44%	0.00%	0.00%	94.44%	5.56%
	0	0	0.00%	0.00%	0.00%	94.44%	94.44%	5.56%
300	60	1	83.33%	13.33%	0.00%	0.00%	13.33%	3.33%
	60	0	83.33%	0.00%	0.00%	13.33%	13.33%	3.33%
	0	1	0.00%	96.67%	0.00%	0.00%	96.67%	3.33%
	0	0	0.00%	0.00%	0.00%	96.67%	96.67%	3.33%
3600	60	1	83.33%	16.39%	0.00%	0.00%	16.39%	0.28%
	60	0	83.33%	0.00%	0.00%	16.39%	16.39%	0.28%
	0	1	0.00%	99.72%	0.00%	0.00%	99.72%	0.28%
	0	0	0.00%	0.00%	0.00%	99.72%	99.72%	0.28%

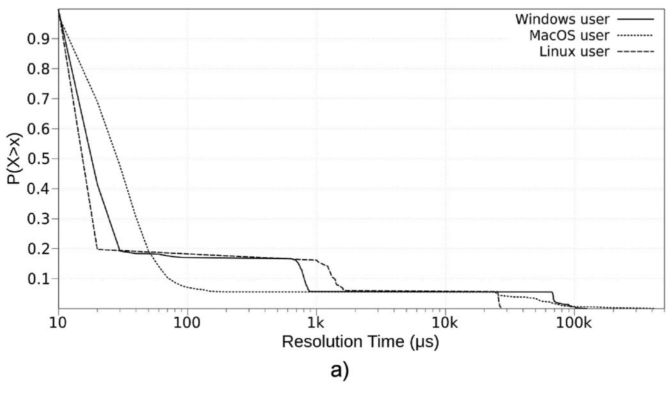


Fig. 11. CCDFs of the DNS resolution times for five example users in the normal scenario (no TTL modification), which are differentiated by operating system and environment: a) desktop environments and b) mobile environments.

the theoretical curves that were obtained via expressions (1–11) for each case. The theoretical curves were obtained by creating as many cache accesses (browser cache, local DNS server cache and authoritative DNS server) as there should be in a period, according to Table 4. The values of the resolution times of each type of source are determined by the probability of obtaining a time value for each source cache.

The profiles in Figs. 11 and 12 show that it is possible to distinguish between normal and privacy-compromised scenarios. In a privacy-compromised scenario, the presence of high intermediate resolution times that are close to the authoritative resolution times that correspond to local DNS server resolutions will indicate that the TTL field is being modified. This characteristic will be used by our tool to offer an automated system for the identification of privacy-compromised scenarios.

4.2. DNS resolution times by cache type

Four types of sources are used by a browser (or any other client application) when it wants to resolve a domain: the browser cache, the OS DNS cache, the local DNS server cache and the authoritative DNS server through the local DNS server. These requests/responses will differ in terms of resolution time. If we identify which source is answering our

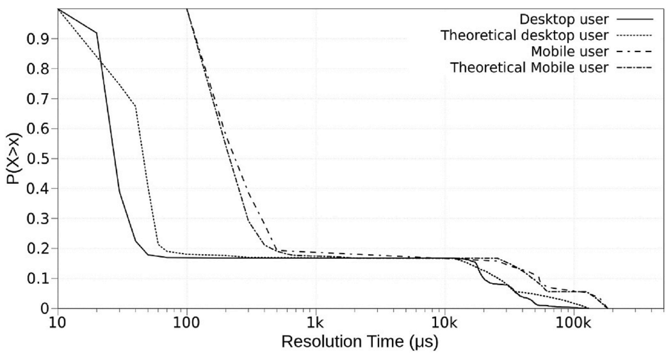


Fig. 12. CCDFs of the DNS resolution times for two example users in the privacy-compromised scenario (ISP changes TTL), which are differentiated by platform: desktop (Windows/macOS) and mobile (Android).

Table 5
Ratios of the DNS queries that were resolved by each DNS information source according to the experimental and theoretical results.

	Browser cache		Operating system cache		Local server cache		Authoritative server cache	
	Theoretical	Experimental	Theoretical	Experimental	Theoretical	Experimental	Theoretical	Experimental
Windows	83.33%	83.36%	11.11%	11.10%	0.00%	0.00%	5.56%	5.55%
macOS	83.33%	80.43%	11.11%	13.93%	0.00%	0.00%	5.56%	5.65%
Linux	83.33%	83.87%	0.00%	0.00%	11.11%	10.48%	5.56%	5.65%
Android	94.44%	94.17%	0.00%	0.00%	0.00%	0.00%	5.56%	5.83%
iOS	0.00%	0.00%	0.00%	0.00%	94.44%	94.74%	5.56%	5.26%

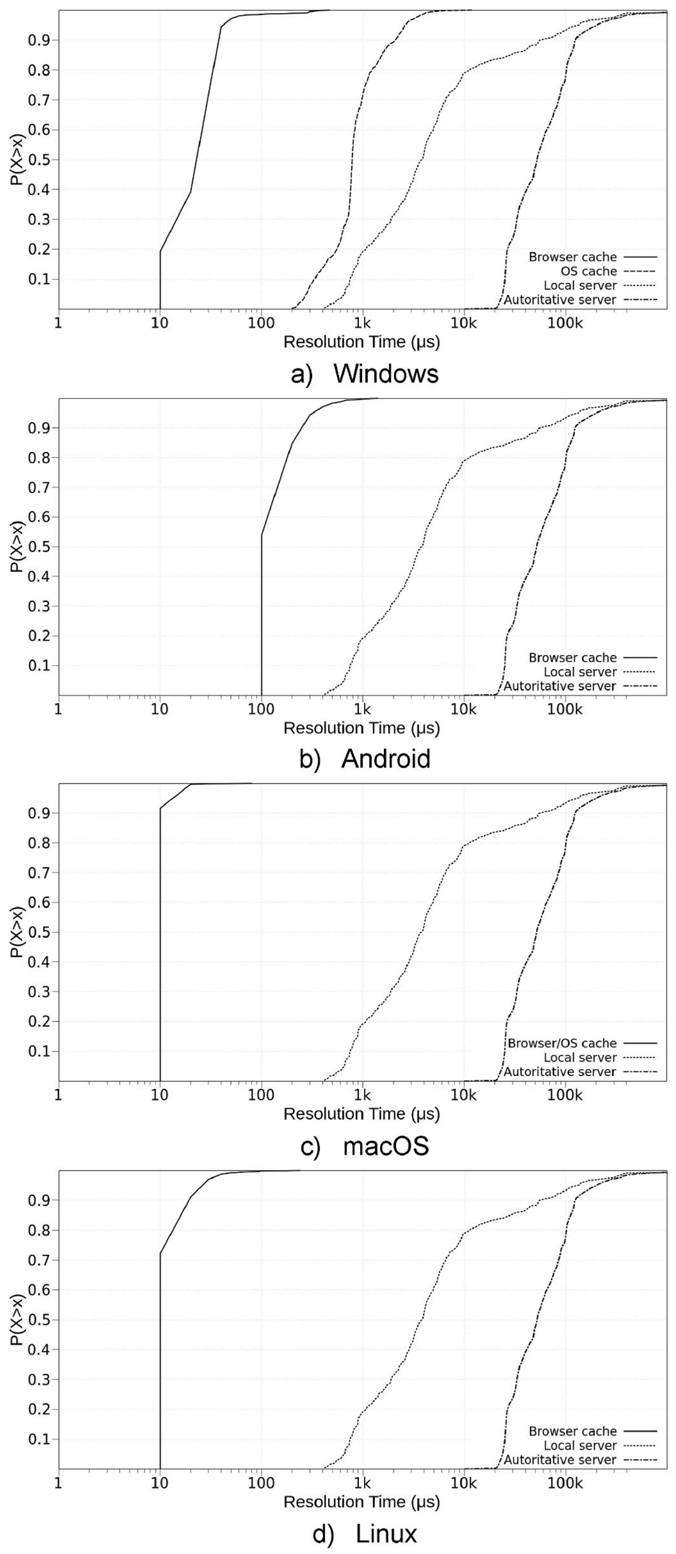


Fig. 13. CDF of the DNS resolution times of each DNS source that an operating system can access. a) Windows, b) Android, c) macOS, and d) Linux.

Table 6
DNS resolution times (in milliseconds) of the DNS sources for various platforms.

	Windows	Android	macOS	iOS	Linux
Browser DNS cache	0.01–0.3	0.1–2	0.01–0.1	–	0.001–0.25
Operating system DNS cache	0.2–6	–	0.01–0.1	–	–
Local DNS server cache	>0.4	>0.4	>0.4	>0.4	>0.4
Authoritative DNS server query	>1	>1	>1	>1	>1

request, we will be able to determine if an ISP is manipulating the TTL field of the DNS responses.

To characterize these resolution times, we have used the collected information from our online tool (Section 3). We divided the DNS resolution times into 4 groups according to the source of the response: the browser cache, the OS cache, the local DNS server cache and the authoritative DNS server (through the local DNS server). The times are associated with the caches according to the sequence in which they appear in successive responses, as expressed in expressions (1–11). The DNS resolution time for each source will vary according to the device (operating system) that is being used. We distinguish among the main operating systems in desktop environments (Windows, Linux, and macOS) and mobile environments (Android, iOS). Fig. 13 presents the CDFs of the DNS resolution times that are observed for each of the DNS information sources that the operating systems may access for resolving DNS queries.

In the figure, the timings of the browser and OS caches can differ among the operating systems. Linux and Android systems do not have an operating system cache by default; hence, their browsers rely completely on their own application cache. For the iOS platform (Apple iPhone and iPad devices), no figure is provided because they do not have a browser cache or an OS cache. For each platform, the differences in the distributions of the DNS resolution times among the 4 caches are used to determine whether a DNS request has been answered by each cache by focusing on the DNS resolution time. These differences are reported in Table 6. Clients using the online tool had heterogeneous OS versions. Windows users varied from Windows 7 to Windows 10. Android users were in a broad range from Android 4.4.2 (KitKat) to Android 10 (Android Q). macOS versions were as old as OS X Lion (10.7.1) or as recent as macOS Catalina (10.15.1). Finally, Linux users reported Kernels version 3 and version 4.

According to Fig. 13 and Table 6, the range of DNS response times from different sources sometimes overlap. For example, a resolution time of 0.25 ms can correspond to an access to either the OS cache or the browser cache in Windows OS. However, the sequence of expected responses from expressions (1–11) can facilitate the identification of the cache source. The local DNS server response times always exceed the OS cache response times and are always lower than the times when querying an authoritative DNS. These DNS response times can be characterized to determine which DNS source the client application is using and to deduce the occurrence of DNS TTL manipulation. In platforms such as Android or Linux, there is no OS DNS cache. In Android, the DNS response times that correspond to accesses to the browser DNS cache can be much longer and can exceed 1 ms in some cases. In the case of macOS, the time that is needed to resolve DNS queries by accessing the operating system cache is so short that it is not possible to differentiate when the browser is accessing its own DNS cache or the operating system DNS cache; hence, it is the most reliable operating system for determining whether an ISP is manipulating the DNS service.

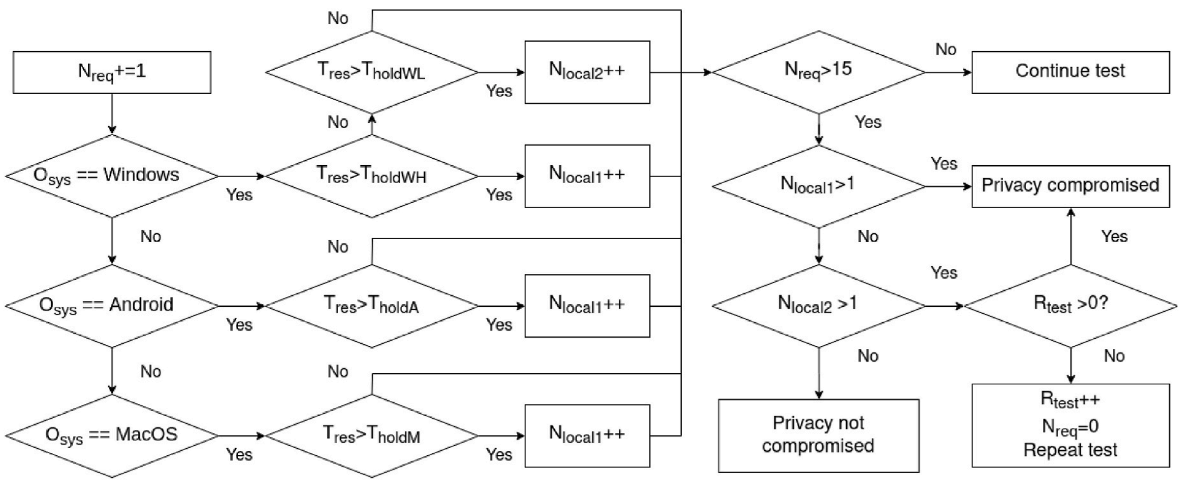


Fig. 14. Flowchart of the algorithm for checking DNS behaviour.

5. DNSPrivacyTester tool for identifying TTL-DNS manipulation

5.1. Selection of operational parameters

The online tool that is described in Section 3 for obtaining DNS resolution times has been extended to identify the source cache of each DNS resolution for the identification of a privacy-compromised scenario. As explained previously, our service determines whether an Internet user's privacy is being compromised by requesting periodically a URL resource. To ensure that the DNS records of this resource have not been previously requested by another client of the same local DNS server, we assign each client a random subdomain that is composed of 21 random characters. To do this, we configured our domain as a Wildcard DNS record (Lewis, 2006), which allows clients to access the resources of the server using various subdomains of the main domain that is assigned to its IP address. These subdomains have the same authoritative TTL value as the main domain, and we can use as many as we need. We also had to enable the cross-origin resource sharing (CORS) (Lekies et al., 2011) of a PHP file at the server that will be requested using the various subdomains to use only one physical server to allocate the web service for various URLs.

To determine whether a DNS resolution time corresponds to a DNS query to a local DNS server or not, we must define thresholds that depend on the operating systems and the resolution times that are listed at Tables 4 and 6. We will also consider that a DNS query that is resolved by a local DNS server can be as little as 0.4 ms, as we have observed from users using our tool.

For macOS, the resolution times of queries that are solved by the operating system's cache and the browser's cache are always much shorter than the resolution times of queries that are solved by local DNS servers (Fig. 13c); hence, any DNS answer with a response time that exceeds $T_{holdM} = 0.4$ ms is regarded as solved by a local DNS server, namely, there is no possibility of obtaining false-positive or false-negative results.

When using Android systems, the access to the browser's DNS cache takes longer than in macOS systems. Accessing the browser's DNS cache can take between 0.1 ms and 1.4 ms (Fig. 13b). From Figs. 12b and 0.6% of the queries that were solved by the browser's cache had a resolution time that exceeded $T_{holdA} = 0.7$ ms, and we never obtained two consecutive times that exceeded T_{holdA} in the same test. This means that our approach will be based in getting two resolution times greater than T_{holdA} (samples 6 and 12 from Fig. 10) to identify that both requests are answered by a local server and therefore that the TTL is being manipulated. Since 7.31% of the local DNS resolution times are below T_{holdA}(Fig. 13b), some of the cases in which ISPs are reducing the TTL of DNS responses could be undetected (false negatives), however, we have considered more important to be certain of any positive detected. $T_{holdA} = 0.7$ ms is selected to obtain zero false positives and a low rate of false negatives. For the statistics we present in section 6, even if the test from a user reports a false negative, it can be corrected by another user completing the test from the same ISP.

For Windows, the response time values from the operating system DNS cache overlap substantially with the local DNS server response times; hence, detection in Windows devices is more difficult than for Android or macOS devices. We define two thresholds when using Window devices, $T_{holdWL} = 1$ ms and T_$T_{holdWH} = 3.7$ ms. Both thresholds are selected to minimize the number of false positives at the expense of getting some false negatives. Resolution times from the OS cache never exceed T_{holdWH}, and they only exceed T_{holdWL} in 27.5% of the cases. This 27.5% can be reduced to 8.05% if we require two consecutive samples to exceed the threshold. Resolution times from the local DNS server below T_{holdWL} are 19.21% of the cases and they get reduced significantively if we require two consecutive samples to not exceed this threshold. Following this analysis, the proposed algorithm is the following: a DNS response time that exceeds T_{holdWH} is considered to correspond to a query that is solved by a local DNS server. A DNS response time that is between T_{holdWL} and T_{holdWH} is not conclusive, and more samples are needed. Finally, a response time that is below T_{holdWL} is considered to correspond to a query that is solved by the OS cache and if erroneous (false negative) it can be improved with more samples. The algorithm is described in more detail using s flowchart in the next section.

5.2. Flow diagram of the tool

The proposed web tool is organized into two phases: In the first phase, a validation is conducted in which the service confirms that the user is using macOS, Windows or Android and is using Google Chrome browser. Once this has been validated, the user is able to start the test. If the user starts the test, the tool stores in a database information about its IP address, ISP, HTTP-user agent, assigned subdomain and timestamp of

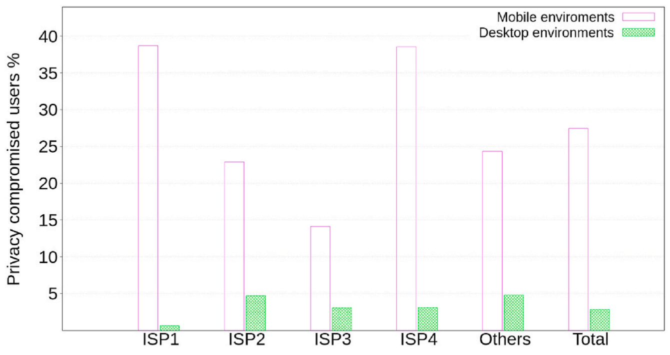


Fig. 15. Percentages of users whose privacy is being compromised by an ISP, which are classified by the environment (mobile or desktop).

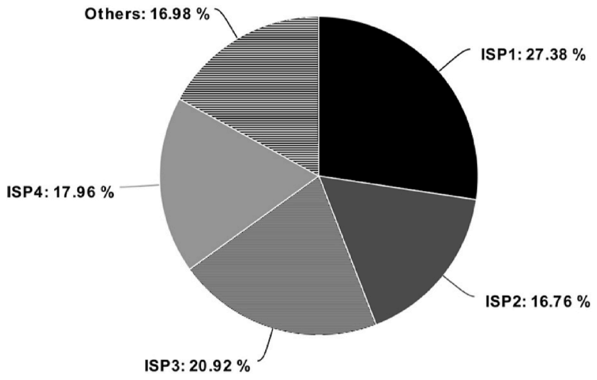


Fig. 16. Percentages of Internet users who conducted the privacy test, divided by ISPs.

the first time a resource is requested. This information will be used for classifying the results, but it is not necessary for the tool to operate. The IP address of each client enables the identification of the autonomous system using the WHOIS service and, therefore, of the ISP of the user, in addition to the country, region name and city name. The User-Agent is an option in the HTTP header and provides information on the type of web browser and the operating system of the client.

In the second phase, a target random subdomain is polled every polling interval to obtain a profile of the DNS response times that are obtained by the client and to deduce the DNS behaviour. A polling interval of 10 s is selected because it enables several samples to be obtained before the cache of typical browsers expires (1 min), and the duration of the test is equivalent to the authoritative TTL value of the target subdomain. An authoritative TTL of 180 s enables 18 samples to be obtained and enables an identification that is similar to that proposed in Section 4.1. Due to miscalibrations in the JavaScript timeouts, the exact samples where the local DNS server responses occur may be shifted. To correct this error, we analyse the samples according to their sequence numbers, namely, from 6 to 10 and from 11 to 15. This will not alter the results as the thresholds always exceed the values of the browser DNS cache resolution times.

Fig. 14 illustrates the algorithm that is used to analyse each of the DNS resolution times that we obtain during the test. In this diagram, the operating system is represented as the variable O_{sys}, the obtained DNS response time as t_{res}, and the four thresholds as T_{holdM} for macOS systems, T_{holdA} for Android systems and T_{holdWL} and T_{holdWH} for Windows systems (lower and upper thresholds respectively). The values of these thresholds were described in the previous section. The number of the request that is being analysed is stored in a variable, namely, N_{req}. The number of request with response times between T_{holdWL} and T_{holdWH} is stored in variable N_{local2}, and the number of requests that exceed any of the other thresholds are stored in variable N_{local1}. The variable R_{test} indicates the number of times the test has already been completed in the same session, and it is used in windows devices when obtained samples are not reliable enough to make a decision. In that case, R_{test} is incremented by one and the test is repeated again. The variables N_{req}, N_{local1}, N_{local2} and R_{test} have an initial value of 0.

According to Fig. 14, the test requires 16 samples of DNS response times to determine whether the user's ISP is modifying the TTL of DNS answers; hence, the test is 160 s long. Once the result has been extracted, the user is notified whether its ISP is modifying the TTL of DNS answers or not. If the result indicates that the test should be repeated, a new test will start, and once it has been completed, the result will be communicated to the user.

With a polling interval of 10 s and a total measuring time of 180 s, we capture 18 DNS response times. With a typical browser cache of 60 s and an OS level cache, we would obtain 15 DNS resolution times that are related to the browser DNS cache and 1 DNS resolution time that is related to the authoritative DNS server. The 2 remaining DNS response times correspond to the OS DNS cache in the normal scenario (Figs. 6–9) or to the local DNS server in the privacy-compromised scenario (if the TTL is changed to a value that is less than the browser cache; see Fig. 10). If only 1 DNS response time is obtained for each later source (which is a very rare scenario that is mainly caused by problems in the client PC), the test is conducted for an additional 1 or 2 min with a new random domain. Depending on the new sample, the final decision is made. It is necessary to obtain 2 consecutive samples with times that are related to the OS DNS cache or the local DNS server cache. The test can finish after obtaining 2 DNS resolution times that differ from the browser DNS cache source. In approximately 160 s, it is possible to identify the scenario in most cases.

For clients without an OS DNS cache, such as Android, the browser cache follows the value of the TTL that was obtained from the local DNS server. In a normal scenario, the local DNS server maintains the authoritative DNS TTL; therefore, the tool will obtain 17 DNS response times that are related to the browser DNS cache and 1 that is related to the authoritative DNS server. However, in a privacy-compromised scenario, if the authoritative DNS TTL is changed to a value that is less than the browser cache time, there will be 15 DNS response times related to the browser DNS cache, 2 that are related to the local DNS server and 1 that is related to the authoritative DNS server.

In summary, with a polling interval of 10 s and a total measuring time of 160 s, we capture 16 DNS response times. In the privacy-compromised scenario (with or without an OS cache), there must be 2 DNS response times that correspond to the local DNS server cache as the source. This is the final decision that the tool must make.

In the graphic interface of the tool, during the test, users can view two charts that display the DNS resolution times they are obtaining in real time, along with a log with the DNS response times that are obtained and the time at which the request for the loaded resource was completed. The users can also compare their results with the results that are obtained by other users from the same or different ISPs. The webpage has also a section with information about how the test is conducted and its objective so that non-expert Internet users can understand and interpret the results. The tool[2] is publicly available, and over a thousand unique users have used the tool.

6. Detected cases of ISPs manipulating the TTL in DNS responses

Fig. 15 illustrates the percentage of users of each ISP whose privacy

[2] The source code is available at https://github.com/grsst-upna/DNSPrivacyTester.

Table 7
Results that were obtained, which are separated by environment and into positive (privacy-compromised), false-positive, false-negative and negative (privacy not compromised) results.

	Environment		Total
	Mobile	Desktop	
Positive	25.00%	3.11%	10.89%
False Positive	0%	0%	0%
False Negative	0.54%	9.48%	6.30%
Negative	74.46%	87.41%	82.81%

we have detected, using DNSPrivacyTester tool, to be compromised. They are classified by the type of environment (mobile/desktop). Large differences have been observed between mobile (Android) and desktop (Windows and macOS); hence, the data are presented separately for each environment. The four most important Spanish ISPs are represented in the Figure as ISP1, ISP2, ISP3 and ISP4, and all the other ISPs, about which we collected data, are represented together as Others. The represented data are obtained from the analysis of more than ten thousand tests by more than one thousand users that were conducted during August and September of 2019.

As shown in Fig. 15, all ISPs have a percentage of users whose privacy is being compromised; hence, currently, the TTL modification is not used in the whole network of each provider. ISPs seem to apply the technique of reducing the TTL of DNS responses in mobile environments more often than in desktop environments, but we should also consider that it is easier to detect a user whose privacy is being compromised when the test is conducted by a mobile device (lower percentage of false negatives).

Fig. 16 presents the percentage of the performed tests that corresponds to each of the ISPs that were studied. Most of the tests were conducted by Internet users of ISP1, ISP2, ISP3 and ISP4, while other ISPs (over 80 ISPs) only cover 16.98% of the Internet users that conducted the tests.

According to Section 4.2, there were cases in which we could not determine whether an ISP was modifying the TTL or not, due to the large values that were obtained for the DNS resolution times when accessing the operating system cache in Windows and the browser cache in Android systems. Table 7 presents the results that we obtained for each platform. This table was constructed by applying the algorithm that we designed to the samples that were obtained during the tests. According to Table 6, the percentage of false positives that were obtained is 0%, as our system is designed to prevent false-positive results. The ground truth is obtained by considering all samples from each user and not only the first 160 s. The number of false-negative results is only 6.30%. The explanation is that to reduce the number of false-positive results, the number of false negative results must be increased via the use of higher thresholds if the number of tests that are performed is not increased.

In total, according to the percentage of positive results that were obtained, a total of 10.89% of users are having their privacy compromised by their ISPs through the DNS service. It is expected that in the near future, this percentage will increase because this technique is one of the inexpensive options to characterize user behaviour when the percentage of encrypted traffic increases.

7. Related work

Many systems that use active probing to infer the popularity of a domain have been proposed (Wills et al., 2003; Rajab et al., 2008; Ma et al., 2015). These methods require the recursive query of DNS servers regarding domains. Through commands such as dig or nslookup, a program can collect information on the last time someone accessed a domain. These methods do not affect the DNS records that are stored in the server's cache but retrieve accurate information about the resources its users are querying. However, the performance of these methods in estimating the popularity of a domain is limited by the authoritative TTL that is set for the DNS record by its owner (Wills et al., 2003). If ISPs set a low and constant TTL for all resolved domains, the accuracy of these methods would be improved, as clients would query their DNS servers more frequently and more uniformly, thereby facilitating the collection and processing of DNS information. Systems that rely on the passive analysis of packets that are captured in DNS resolvers, such as those described in (García-Dorado et al., 2018; Kim and Zhang, 2015), would also benefit from a reduction of the TTL of DNS answers. These studies focus on inferring the user behaviour and the popularity of web pages, but they do not test the DNS conditions of Internet users or examine how their privacy is compromised by ISPs, although they assume that ISPs do not change TTL values.

The system that is described in (García-Dorado et al., 2018) detects which websites a user has accessed. When a user enters a webpage, several DNS queries with various TTLs are triggered, and it weighs the importance of each one based on the TTL and popularity. A low and uniform TTL for all DNS records would render the system more accurate and simpler as the importance of a DNS response could be established based only on the popularity of the domain. In (Kim and Zhang, 2015), the system detects the most visited domains by a user by constructing sets of the most queried DNS records by the user. A low and uniform TTL would not only increase the amount of information they collect but also put all domains under equal conditions for inclusion in the set of the most visited domains, thereby improving substantially the accuracy of the system.

In the case of a tracking system such as that described in (Herrmann et al., 2013), the analysis is based on examining DNS requests that are resolved by DNS servers and tracking Internet users based on their behaviours. Increasing the number of queries that are made by each client would facilitate the tracking of these users and render the obtained results more accurate, thereby reducing their anonymity.

It has been reported that there are malicious open DNS servers that change the DNS responses to redirect users to custom destinations (Trevisan et al., 2017) and that some open DNS resolvers increase the TTL value of DNS responses to reduce servers load (Kührer et al., 2015). However, we have not been able to find any study that reported and effectively demonstrated the existence of ISPs that manipulate DNS responses to increase the collection of information about their clients. Another study (Liu et al., 2018) suggests a method for identifying DNS resolvers that are intercepting DNS queries that are made by Internet clients, but it does not provide information about DNS manipulations that are conducted by legitimate DNS resolvers. Reference (Pearce et al., 2017) describes a method for measuring DNS manipulation at a global scale, but it does not provide a method that could be used by regular Internet users to check whether their ISP is manipulating the DNS service or not. Our tool, namely, DNSPrivacyTester, requires direct but simple action from Internet users and estimates DNS manipulation from a client-side perspective, which enables us to collect information about manipulations by ISP DNS servers that could not be collected by any other tool.

DNS has been reported to be the most vulnerable Internet service nowadays, and ISPs have little interest in implementing measurements that enhance their users' privacy (Lee, 2019; Hunter, 2019; Brodkin, 2019). There are many solutions that Internet users can implement to secure their DNS information, such as DNS over TLS (DoT), DNS over

HTTPS (DoH) and DNSSEC; however, their use is mostly anecdotal (Osterweil et al., 2007; Szalachowski and Perrig, 2017; Lu et al., 2019; Lian et al., 2013), although most OSs and applications allow the implementation of DNS encryption (Dickinson, 2020). The problem is that regular Internet users are unaware of the importance of their privacy, and in most cases, they do not know how to implement methods to protect it.

8. Conclusions

We have presented a strategy for determining whether the DNS service is being manipulated by ISPs to collect browsing data on their subscribers. The study focused on the effectiveness of reducing the TTL of the DNS responses and the factors that can affect the amount and quality of the data that are collected via this technique. We have analysed the DNS record sources that an application may access when it wants to retrieve DNS information about a domain and how they affect the amount of data that the ISP may obtain.

We have analysed the impact that the use of this technique of reducing the TTL values of DNS responses can have in the current Internet scenario. We have addressed the consequences of the use of this technique, with an emphasis on the privacy concerns it raises. Finally, we have developed a public tool that can help all Internet users (regardless of their knowledge about Internet protocols) check if their ISPs are compromising their privacy via this technique. The results reach 0% of false positives for privacy-compromised scenario, and 6.30% of false negatives that can be reduced with a longer measurement time.

According to the results that were obtained from users, the technique of reducing the TTL value of DNS answers is currently being used at various levels by large ISPs, thereby supporting the privacy concerns of Internet users.

Author contribution

Tomas Hernandez-Quintanilla: Software, Data curation, Investigation, Writing – original draft, Validation. Eduardo Magaña: Conceptualization, Methodology, Investigation, Writing – original draft, Supervision. Daniel Morató; : Conceptualization, Writing – review & editing. Mikel Izal: Writing – review & editing.

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.

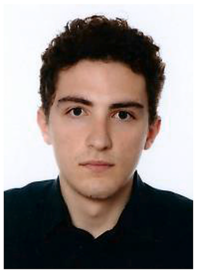
Acknowledgements

This work was supported by the Spanish State Research Agency with project PID2019-104451RB-C22/AEI/10.13039/501100011033.

References

Brodkin, J., 2019. ISPs lied to Congress to spread confusion about encrypted DNS, Mozilla says. https://arstechnica.com/tech-policy/2019/11/isps-lied-to-congress-to-spread-confusion-about-encrypted-dns-mozilla-says/.
Bhatti, S.N., Atkinson, R., 2011. Reducing DNS caching. In: IEEE Conference on Computer Communications Workshops, pp. 792–797. Shanghai.
Callahan, T., Allman, M., Rabinovich, M., 2013. On modern DNS behavior and properties. Comput. Commun. Rev. 43 (3), 7–15.
Chitpranee, R., Fukuda, K., 2013. Towards passive DNS software fingerprinting. In: Proceedings of the 9th ACM Asian Internet Engineering Conference (AINTEC '13), pp. 9–16. New York, NY, USA.
Cohen, E., Kaplan, H., 2001. Proactive Caching of DNS Records: Addressing a Performance Bottleneck. Computer Networks, pp. 707–726.
Drako, D., 2013. Policy-managed DNS server for to control network traffic. U.S. Patent No. 8, 447, 856. Washington, DC.
Alexa, 2020. Alexa top sites on the web accessed april 2020. https://www.alexa.com/topsites.
García-Dorado, J.L., Ramos, J., Rodríguez, M., Aracil, J., 2018. DNS weighted footprints for web browsing analytics. J. Netw. Comput. Appl. 35–48.
Grigorik, I., 2020. Performance resource API for javascript. Online, accessed april 2020. https://w3c.github.io/navigation-timing/#PerformanceResourceTiming.
Hunter, M., 2019. Encrypted DNS Could Help Close the Biggest Privacy Gap on the Internet. Why Are Some Groups Fighting Against It? Online, accessed april 2020. https://www.eff.org/deeplinks/2019/09/encrypted-dns-could-help-close-biggest-privacy-gap-internet-why-are-some-groups.
Hernan, S., Lambert, S., Ostwald, T., Shostack, A., 2006. Threat modeling: uncover security design flaws using the STRIDE approach. MSDN Mag. 6 (11), 68–75.
Herrmann, D., Banse, C., Federrath, H., 2013. Behavior-based tracking: exploiting characteristic patterns in DNS traffic. Comput. Secur. 39, 17–33.
Jung, J., Sit, E., Balakrishnan, H., Morris, R., 2002. DNS performance and the effectiveness of caching. IEEE/ACM Trans. Netw. 10 (5), 589–603.
Kim, D.W., Zhang, J., 2015. You Are How You Query: Deriving Behavioral Fingerprints from DNS Traffic", *Security And Privacy In Communication Networks*, vol. 164. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, pp. 348–366.
Klein, A., Pinkas, B., 2019. DNS Cache-Based User Tracking. Network and Distributed System Security Symposium.
Kohnfelder, L., Garg, P., 1999. The Threats to Our Products. Microsoft Interface, Microsoft Corporation, p. 33.
Kührer, M., Hupperich, T., Bushart, J., Rossow, C., Holz, T., 2015. Going wild: large-scale classification of open DNS resolvers. In: Proceedings of the ACM Conference on Internet Measurement Conference, pp. 355–368. New York, USA.
Lee, T., 2019. Why big ISPs aren't happy about Google's plans for encrypted DNS. Online, accessed april 2020. https://arstechnica.com/tech-policy/2019/09/isps-worry-a-new-chrome-feature-will-stop-them-from-spying-on-you/.
Lekies, S., Johns, M., Tighzert, W., 2011. The state of the cross-domain nation. In: Proceedings of the 5th Workshop on Web 2.0 Security and Privacy.
Lewis, E., 2006. The Role of Wildcards in the Domain Name System. RFC4592.
Lian, W., Rescorla, E., Shacham, H., Savage, S., 2013. Measuring the practical impact of DNSSEC deployment. In: Proceedings of the 22nd USENIX Conference on Security, pp. 573–588. Washington, D.C.
Liu, B., Lu, C., Duan, H., Liu, Y., Li, Z., Hao, S., Yang, M., 2018. Who is answering my queries: understanding and characterizing interception of the DNS resolution path". Proceedings of the Applied Networking Research Workshop 15–16. New York, NY, USA.
Lu, C., Liu, B., Li, Z., Hao, S., Duan, H., Zhang, M., Leng, C., Liu, Y., Zhang, Z., Wu, J., 2019. An end-to-end, large-scale measurement of DNS-over-encryption: how far have we come?. In: Proceedings of the Internet Measurement Conference Amsterdam, Netherlands.
Ma, Xiaobo, Zhang, Junjie, Li, Zhenhua, Li, Jianfeng, Jing, Tao, Guan, Xiaohong, John, C., Lui, S., Don, Towsley, 2015. Accurate DNS query characteristics estimation via active probing. J. Netw. Comput. Appl. 47, 72–84. ISSN 1084-8045.
Monrose, F., Krishnan, S., 2010. DNS prefetching and its privacy implications: when good things go bad. In: Proceedings of the 3rd USENIX Conference on Large-Scale Exploits and Emergent Threats: Botnets, Spyware, Worms, and More, 10-10.
Moreno, V., Ramos, J., Santiago del Río, P., García-Dorado, J., Gomez-Arribas, F., Aracil, J., 2015. Commodity packet capture engines: tutorial, cookbook and applicability. IEEE Communications Surveys & Tutorials 17 (3), 1364–1390.
Osterweil, E., Massey, D., Zhang, L., 2007. Observations from the DNSSEC Deployment. 3rd IEEE Workshop on Secure Network Protocols, Beijing, pp. 1–6.
Pearce, P., Jones, B., Li, F., Ensafi, R., Feamster, N., Weaver, N., Paxson, V., 2017. Global measurement of DNS manipulation. In: 26th USENIX Security Symposium, pp. 307–323. Vancouver, BC, Canada.
Rajab, M.A., Monrose, F., Terzis, A., Provos, N., 2008. "Peeking through the cloud: DNS-based estimation and its applications", *applied Cryptography and network security*. Lect. Notes Comput. Sci. 5037.
SimilarWeb, 2020. SimilarWeb top websites. Online, accessed april 2020. https://www.similarweb.com/top-websites.
Statcounter browser market share worldwide. Online, accessed april 2020. https://gs.statcounter.com/browser-market-share.
Szalachowski, P., Perrig, A., 2017. Short paper: on deployment of DNS-based security enhancements. Financial Cryptography and Data Security. Lecture Notes in Computer Science 10322.
Torres, L.M., Magana, E., Morato, D., Garcia-Jimenez, S., Izal, M., TBDClust, ", 2017. Time-based density clustering to enable free browsing of sites in pay-per-use mobile internet providers". J. Netw. Comput. Appl. 99, 17–27.
Trevisan, M., Drago, I., Mellia, M., Munafò, M.M., 2017. Automatic detection of DNS manipulations. In: IEEE International Conference on Big Data (Big Data), pp. 4010–4015. Boston, MA.
Vavrusa, M., Grant, D., 2018. Managing domain name system (DNS) queries using a proxy DNS server. U.S. Patent Application 10/033, 692.
Wills, C.E., Mikhailov, M., Shang, H., 2003. Inferring relative popularity of internet applications by actively querying DNS caches. In: Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, pp. 78–90. New York, NY, USA.

Wu, Y., Tuononen, J., Latvala, M., 2007. Performance analysis of DNS with TTL value 0 as location repository in mobile internet. In: IEEE Wireless Communications and Networking Conference. Kowloon, pp. 3250–3255.
Yan, Z., Lee, J., . The road to DNS privacy". Future Generat. Comput. Syst. 112, 604–611.
Zhao, F., Hori, Y., Sakurai, K., 2007. Analysis of privacy disclosure in DNS query. In: International Conference on Multimedia and Ubiquitous Engineering. MUE'07, Seoul, pp. 952–957.
Dickinson, S., 2020. DNS Privacy Implementation Status. Online, accessed april 2020. https://dnsprivacy.org/wiki/display/DP/DNS+Privacy+Implementation+Status.



Tomas Hernandez-Quintanilla is a visiting student at Kent State University, USA. He received his degree in Telecommunications Engineering at Public University of Navarra, Spain, in 2020. In 2019, he held a scholarship at Naudit High Performance Computing and Networking. Since then, he is collaborating with the Telecommunications, Networks and Services Research Group at Public University of Navarra, Spain.

Eduardo Magaña received his M.Sc. and Ph.D. degrees in Telecommunications Engineering from Public University of Navarra, Pamplona, Spain, in 1998 and 2001, respectively. Since 2005, he is associate professor at Public University of Navarra. During 2002 he was a postdoctoral visiting research fellow at the Department of Electrical Engineering and Computer Science, University of California, Berkeley. His main research interests are network monitoring, traffic analysis and performance evaluation of communication networks.

Daniel Morató received the M.Sc. degree in telecommunication engineering and the Ph.D. degree from the Public University of Navarre, Spain. During 2002 he was a visiting postdoctoral fellow at the Electrical Engineering and Computer Sciences Department, University of California, Berkeley. Since 2006 he has been working at the Department of Automatics and Computer Sciences, Public University of Navarre, as an associate professor. His research interests include high-speed networks, performance and traffic analysis of Internet services and network monitoring.

Mikel Izal received his M.Sc. and Ph.D. degrees in telecommunication engineering in 1997 and 2002 respectively. In 2003 he worked as a scientific visitant at Institute Eurecom, Sophia-Antipolis, France, performing measures in network tomography and peer-to-peer systems. Since then, he has been with the Department of Automatics and Computer Sciences of the Universidad Pública de Navarra where he is an Associate Professor. His research interests include traffic analysis, network tomography, high speed next generation networks and peer to peer systems.