UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

JOSÉ CESAR CHAGASTELLES PINTO

Analysis and Comparison of Encrypted DNS and Classic DNS

Work presented in partial fulfillment
of the requirements for the degree of
Bachelor in Computer Science

Advisor: Prof. Dr. Lisandro Zambenedetti Granville
Coadvisor: Dr. Eder John Scheid

Porto Alegre
February 2024

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Carlos André Bulhões Mendes
Vice-Reitora: Profª. Patricia Helena Lucas Pranke
Pró-Reitor de Graduação: Prof. Cíntia Inês Boll
Diretora do Instituto de Informática: Profª. Carla Maria Dal Sasso Freitas
Coordenador do Curso de Ciência de Computação: Prof. Marcelo Walter
Bibliotecária-chefe do Instituto de Informática: Alexsander Borges Ribeiro

ABSTRACT

The Domain Name System (DNS) is one of the main components of the Internet, performing the translation of domain names (*e.g.*, google.com) to IP addresses (*e.g.*, 142.251.133.196). However, this system was not created with security as its main characteristic, resulting in various attacks that can originate from DNS servers, such as flooding and spoofing. In addition, the confidentiality of DNS queries is affected (and consequently the privacy of users), as all communication by default occurs using plain text. Therefore, attackers and unauthorized agents can access these queries, carrying out malicious activities or censoring users' connections. To solve this particular security problem, new solutions have been proposed, such as DNS over TLS (DoT) and DNS over HTTPS (DoH). Despite these solutions addressing security and privacy issues, they introduce additional layers and processes, which can lead to performance issues in domain resolution, thus sacrificing performance for security. Therefore, this work analyzes and compares the performance of encrypted DNS protocols against classic DNS, showing that the performance impact exists but is a reasonable trade-off against encryption benefits.
Keywords: DNS. Internet Access. Communication Protocols. Measurement.

Análise e Comparação entre DNS Criptografado e DNS Clássico

RESUMO

O Domain Name System (DNS) é um dos principais componentes da Internet, realizando a tradução de nomes de domínio (por exemplo, google.com) para endereços IP (por exemplo, 142.251.133.196). Entretanto, esse sistema não foi criado com segurança como principal característica, resultando em diversos ataques que podem originar de servidores DNS, como flooding e spoofing. Além disso, a confidencialidade das consultas DNS é afetada (e consequentemente a privacidade dos usuários), pois toda a comunicação por padrão acontece utilizando texto puro. Logo, atacantes e agentes não autorizados podem ter acesso a estas consultas, realizando atividades mal-intencionadas ou censurando a conexão de usuários. Para resolver este problema de segurança em particular, novas soluções foram propostas, como DNS over TLS (DoT) e DNS over HTTPS (DoH). Apesar destas soluções atacarem o problema de segurança, elas introduzem camadas e processos adicionais, os quais podem levar a problemas de desempenho na resolução de domínios, trocando assim desempenho por segurança. Portanto, esse trabalho analisa e compara a performance de protocolos DNS encriptados contra a de DNS clássico, mostrando que o impacto na performance existe porém é uma troca razoável pelos benefícios da comunicação encriptada.

Palavras-chave: DNS, Acesso à Internet, Protocolos de Comunicação, Medição.

LIST OF FIGURES

Figure 2.1 DNS Hierarchy 11
Figure 2.2 DNS Query 13
Figure 2.3 TLS Handshake 16

Figure 4.1 Measurement Tool Design 20
Figure 4.2 Extraction of *dig*'s Reported Query Time using *awk* 24
Figure 4.3 Output CSV file example: headers and data 24

Figure 5.1 Do53 Lookup Tool Performance 29
Figure 5.2 DoH Lookup Tool Performance 30
Figure 5.3 DoT Lookup Tool Performance 31
Figure 5.4 Results of Protocol Performance using Different Tools 32
Figure 5.5 Protocol Performance: aggregated 32

LIST OF TABLES

Table 3.1 Review of Literature on DNS Resolvers Performance Research 19

Table 5.1 Lookup Success Rate based on Resolver and Tool using Do53 33
Table 5.2 Lookup Success Rate based on Resolver and Tool using DoH 33
Table 5.3 Lookup Success Rate based on Resolver and Tool using DoT 34
Table 5.4 Resolver Latency: ICMP ping RTT 35

LIST OF ABBREVIATIONS AND ACRONYMS

DNS	Domain Name System
Do53	DNS-over-Port 53
DoH	DNS-over-HTTPS
DoT	DNS-over-TLS
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
TLS	Transport Layer Security
IETF	Internet Engineering Task Force
NIC	Network Information Center
ISP	Internet Service Provider
TLD	Top Level Domain
RR	Resource Records
IoT	Internet of Things
ICMP	Internet Control Message Protocol
OS	Operating System

CONTENTS

1 INTRODUCTION **9**
2 BACKGROUND **11**
2.1 DNS Functioning **12**
2.2 Encrypted DNS **14**
2.2.1 DNS over HTTPS (DoH) 14
2.2.2 DNS over TLS (DoT) 15
2.2.3 Overhead 15
3 RELATED WORK **17**
4 ANALYZING DNS PERFORMANCE AND OVERHEAD **20**
4.1 Methodology and Design **20**
4.2 Implementation **21**
4.2.1 Lookup Tools 22
4.2.2 Measurements 23
4.2.3 Analysis 24
4.2.4 Extending the Tool 26
5 EVALUATION **28**
5.1 Experiment Setup **28**
5.2 Lookup Tool Performance **28**
5.2.1 DNS over Port 53 (Do53) 29
5.2.2 DNS over HTTPS (DoH) 30
5.2.3 DNS over TLS (DoT) 30
5.3 DNS Protocol Performance **31**
5.4 Lookup Success Rates **33**
5.5 Resolver Latency **34**
5.6 Discussion and Limitations **35**
6 SUMMARY, CONCLUSION AND FUTURE WORK **37**
REFERENCES **39**
APPENDIX A — PUBLISHED PAPER – ERRC 2023 ..... **42**
APPENDIX B — SUBMITTED PAPER – ISCC 2024 ..... **49**

1 INTRODUCTION

Established in 1983, the Domain Name System (DNS) emerged as a critical component of the Internet (MOCKAPETRIS; DUNLAP, 1988). Its primary function is to translate user-friendly hostnames (*e.g.*, *google.com*) into their corresponding Internet Protocol (IP) addresses, effectively serving as the "phone book" of the Internet (KUROSE; ROSS, 2016). Nearly all Internet communication starts with a DNS lookup, and complex websites which require content from multiple third parties might perform hundreds of DNS requests before loading a single page (BUTKIEWICZ; MADHYASTHA; SEKAR, 2011). Thus, DNS performance is of concern as it directly impacts performance in most Internet-based communications (BOZKURT et al., 2017).

However, this system was not created with security as a main feature, resulting in several attacks that can originate from DNS servers, such as flooding and spoofing (SCHMID, 2021). Furthermore, the confidentiality of DNS queries is affected (and consequently the privacy of users), as all communication, by default, occurs using plain text through the User Datagram Protocol (UDP), so attackers and unauthorized agents can gain access to these queries, carrying out malicious activities or censoring user connections. To solve this particular security problem, new solutions have been proposed, including encrypted DNS protocols such as DNS-over-TLS (DoT) (HU et al., 2016) and DNS-over-HTTPS (DoH) (HOFFMAN; MCMANUS, 2018). Although these solutions attack the security and privacy problems, they introduce additional layers and processes (*e.g.*, the exchange of keys in DoH), which can lead to performance issues when resolving domains, thus trading performance for security.

In this sense, past work has measured DNS performance extensively and under different conditions. For example, (AGER et al., 2010) thoroughly analyzed the performance of DNS with distributed measurements across more than 50 different Internet Service Providers (ISPs), in over 28 countries, comparing local and public DNS resolvers. (BöTTGER et al., 2019) focused on comparing performance between DNS and its encrypted versions, DoT and DoH, and their impact in webpage loading times. (AFFINITO; BOTTA; VENTRE, 2022) measured DoH performance overhead as well as malicious domain protection. Although there is work comparing encrypted DNS protocol performance and overhead, they do not discuss the selection of tools used in the experiments or provide their code to foster reproducibility.

Thus, based on the literature research and identified gaps, this work analyzes and

compares the response time of resolving different domains using classic DNS, DoH, and DoT to quantify the overhead when relying on these protocols. To achieve that, we design and implement a solution that automates measurements by querying and storing the data gathered. It utilizes different domains and resolvers defined in lists through input files. We then analyze the results and compare protocol performance. Further, we also share how the different DNS lookup tools behave and their impact on results and provide our solution as open-source for further research.

In summary, we present three main contributions:

1. A selection and comparison of DNS Lookup Tools;
2. A performance comparison of DNS and encrypted DNS protocols; and
3. A customizable DNS performance benchmarking tool.

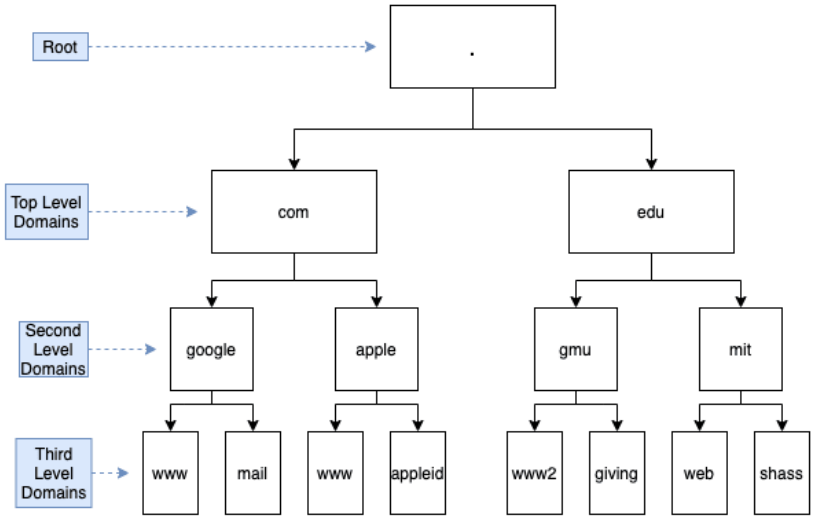
The remainder of this work is structured as follows. Section 2 presents an overview of the DNS, its design and functionality. Section 3 compares related work on analyzing DNS resolver performance. Section 4 details the methodology used in the measurements, the tool selection for the experiments and the solution implementation. Section 5 presents the setup used for the experiment and discusses the results obtained. Lastly, Section 6 summarizes key findings and suggests future work.

2 BACKGROUND

Before the Internet, in the days of its predecessor ARPAnet, the name-to-address mapping for every host connected to the network was managed in a single file called `hosts.txt`, maintained by an organization called Network Information Center (NIC). Changes were emailed to NIC, and fetching of the updated `hosts.txt` file was done through File Transfer Protocol (FTP) (LIU; ALBITZ, 2001). As the network grew and moved to TCP/IP, scaling problems emerged, and a new system was needed. Paul Mockapetris then introduced the first design of the Domain Name System (DNS) in 1983 (MOCKAPETRIS, 1983).

The DNS is comprised of two main components: a distributed database, structured as a hierarchy of DNS servers, and an application-layer protocol that allows end-hosts (*i.e.*, clients) to query the database by using local nameservers called *Resolvers* (KUROSE; ROSS, 2016). To reach a global scale, the DNS database is comprised by a large number of servers distributed around the world, and no single DNS server has the information of all hosts in the Internet. In this sense, the structure of the database is similar to that of the Unix file system, as an inverted tree with the root at the top, and is indexed by domain names. Figure 2.1 illustrates such an inverted tree structure. The data associated with each domain name is stored in Resource Records (RR) (LIU; ALBITZ, 2001), which include information such as A records (mapping domain names to IPv4 addresses), AAAA records (mapping domain names to IPv6 addresses), and MX records (specifying the mail servers responsible for receiving emails for a domain).

Figure 2.1 – DNS Hierarchy



Source: (KHAN, 2020)

Nameservers are servers that store and manage the DNS records for specific domains. On the other hand, zones refer to a specific subset of the DNS namespace controlled by a nameserver, and contain DNS records for the resources within that zone. An important part of DNS scalability is delegation, which allows the parent domain owner to distribute the responsibility of resolving queries for the subdomain to a different set of authoritative name servers. By delegating DNS, the parent domain can effectively manage and organize its subdomains, ensuring better performance and scalability. DNS delegation is commonly used when different entities or organizations are responsible for managing various parts of a domain and helps to streamline the DNS resolution process (LIU; ALBITZ, 2001).

There are three main classes of DNS servers: root, Top-Level Domain (TLD), and Authoritative Server (AS). As shown in Figure 2.1, root nameservers stand at the top of the hierarchy, providing the starting point for any DNS lookup. There are 13 logical root nameservers worldwide, responsible for directing requests to the appropriate TLD nameservers. These TLD nameservers manage specific top-level domains (*e.g.*, `.br`, `.com`, `.org`, `.net`), directing further inquiries to the relevant authoritative nameservers. The authoritative nameservers hold the information for specific domains, and serve as the final source of truth, returning the data for requesting resolvers (LIU; ALBITZ, 2001).

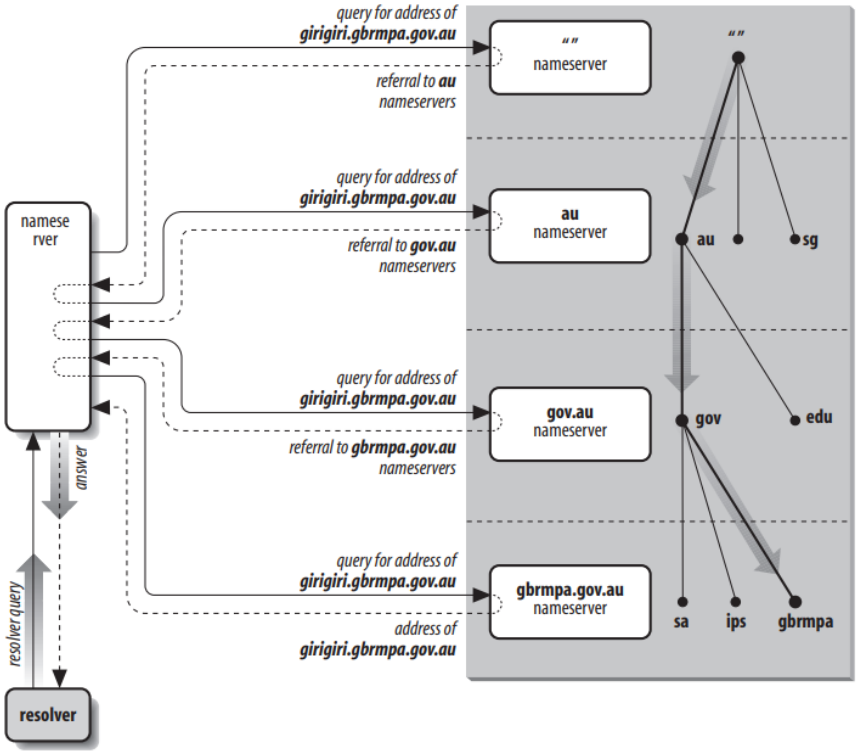
2.1 DNS Functioning

When a software requires information from the domain namespace, such as a Web browser that needs to translate the domain *www.inf.ufrgs.br* to its IP address, it invokes the client side of DNS resolver, initiating the queries that compose the so-called resolution process. Because of the inverted tree structure of the domain namespace, any domain can be reached by starting the search at the root nameservers (LIU; ALBITZ, 2001).

There are two types of DNS queries, iterative and recursive. In iterative queries, the DNS resolver contacts multiple DNS servers until it obtains the desired information. The resolver sends a query to a DNS server and expects a response containing either the requested information or a referral to another DNS server that may have the required data. The resolver then sends subsequent queries to the referred servers until it receives a response with the necessary information. This process continues until the resolver receives a complete answer or reaches a timeout. Recursive queries, on the other hand, involve the DNS resolver outsourcing the entire querying process to other DNS servers. In a re-

cursive query, the resolver sends a query to a DNS server, which checks its cache for the requested data. If the data is not found, the server acts on behalf of the resolver and contacts other DNS servers to recursively obtain the information required. The server follows this recursive process until it eventually obtains the answer and returns it to the resolver. Once the resolver receives the answer, it caches the information for future reference.

Figure 2.2 – DNS Query



Source: (LIU; ALBITZ, 2001)

Figure 2.2 illustrates the resolution process of the address *girigiri.gbrmpa.gov.au*, initiating at the resolver with a recursive query to the local nameserver, and traversing the domain namespace tree from the root nameserver all the way to the *gbrmpa.gov.au* nameserver through iterative queries, until it finds the final address to answer the initial query from the resolver (LIU; ALBITZ, 2001).

Without additional information, queries start at the root nameservers, making them essential to the DNS. However, if every query had to pass through these servers and traverse the whole hierarchy tree, the DNS would not scale so well. To offload some of that heavy traffic, the DNS implements caching mechanisms at multiple levels. Locally, every device (*e.g.*, computer, phone, and router) maintains its own DNS cache. Before issuing queries, the device's software first checks its local cache for the answer. Nameservers also employ caching to optimize resolution for multiple users. When a nameserver receives

a request, it first checks its cache for the corresponding record. If the record is present and still valid, the nameserver can respond immediately without contacting any other servers. This reduces the load on upstream nameservers and improves overall network performance.

2.2 Encrypted DNS

When DNS was created, it was not expected that it would become the backbone of a giant digital economy, and thus a very valuable target for cybercrime. DNS was designed and implemented with speed, reliability, and scalability as the main objectives, instead of privacy and security (SCHMID, 2021). Hence, traditional DNS communication transmits data unencrypted, making it vulnerable to surveillance and manipulation attacks.

To address these security concerns, encrypted DNS protocols have emerged. These protocols, primarily DNS over TLS (DoT) and DNS over HTTPS (DoH), encrypt DNS queries and responses, giving users greater control over their privacy and security while using the internet. They provide an encrypted tunnel between a user's device and a DNS resolver, preventing ISPs, governments, or other entities from intercepting and analyzing DNS traffic to track or manipulate online activities.

Despite the numerous benefits, deploying encrypted DNS protocols may face certain challenges. Compatibility issues with legacy systems, potentially slower performance due to increased encryption overhead, and the need for widespread adoption represent some of the obstacles. However, efforts are being made to overcome these hurdles and eventually make encrypted DNS the default standard.

2.2.1 DNS over HTTPS (DoH)

The DoH protocol was defined in October 2018 by the Internet Engineering Task Force (IETF) (HOFFMAN; MCMANUS, 2018). The amount of available DoH servers and DoH traffic have been growing since (LU et al., 2019).

DoH encapsulates DNS traffic within an HTTPS connection, over port 443, the same port used for secure web traffic. The client encodes the DNS query using either the HTTP GET or POST method, and the default URL path is /dns-query (*e.g.*, `https://dns.google/dns-query`). By utilizing the existing infrastructure of HTTPS,

DoH enables better compatibility with existing web-based security mechanisms, while also blending DNS requests seamlessly into web traffic. It avoids interference from firewalls and other network security measures.

2.2.2 DNS over TLS (DoT)

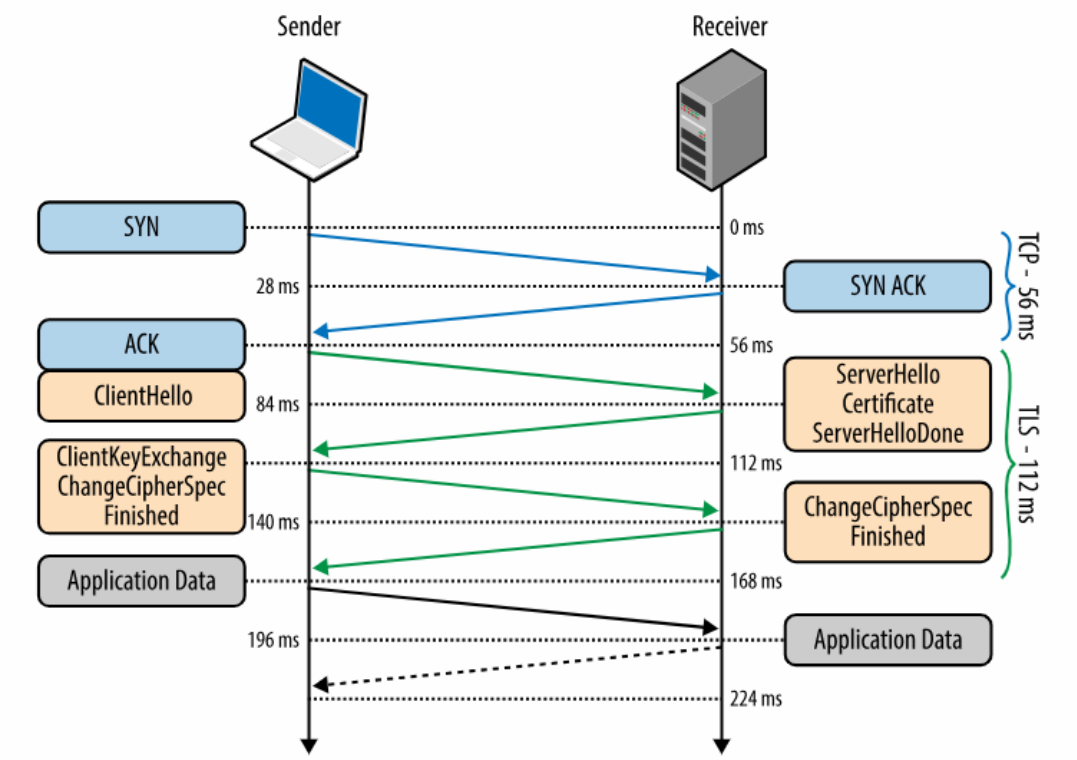
The DoT protocol was defined earlier than DoH, in May 2016 (HU et al., 2016). At this time, there were solutions for other DNS security aspects, such as verifying response integrity, but none that protected user privacy. Despite this head start, nowadays it has fewer servers and traffic than DoH, showing slower adoption (LU et al., 2019).

Similar to DoH, DoT encrypts DNS traffic using Transport Layer Security (TLS). However, instead of sending DNS queries over the standard HTTPS port, it uses port 853, making it ideal for certain environments where DoH may be subject to blocking or monitoring.

2.2.3 Overhead

Both DoH and DoT rely on the TLS cryptographic protocol to provide secure communication between client and server. To establish a connection, a process called TLS Handshake must occur, where both parties agree on the encryption algorithm to be used and exchange keys. Figure 2.3 shows a simplified illustration of a TCP connection using TLS. As can be seen, instead of a single round trip to establish connection, there are two additional ones necessary before the application data can be transmitted. The extra latency is the main overhead incurred when using encrypted DNS protocols. This overhead can be mitigated to an extend by re-using connections and caching.

Figure 2.3 – TLS Handshake



Source: (GRIGORIK, 2013)

3 RELATED WORK

We reviewed the existing literature on DNS performance analysis, focusing mainly on approaches that featured encrypted DNS protocols. By studying all the different methodologies utilized, we were able to gather valuable insights for developing our own.

(AFFINITO; BOTTA; VENTRE, 2022) compares the performance, employing the *pydig* tool, and security aspects of DNS resolvers provided by major Italian ISPs with public resolvers from Google and Cisco (*i.e.*, OpenDNS). Although local resolvers exhibit faster response times, the research finds that their security level matches the level of public resolvers, which indicates that users do not need to compromise their security for improved performance of public DNS resolvers.

In a similar study, (AGER et al., 2010) examines the impact of several DNS resolver responsiveness and cache content on applications such as Content Distribution Networks (CDN). With the use of comprehensive measurements across ISPs, relying on the *dig* Linux tool, the study reveals significant disparities in responses due to CDN location awareness and DNS resolver proximity, uncovering limitations in ISPs' DNS deployments and biases in third-party DNS replies.

(BöTTGER et al., 2019) investigates two encrypted DNS protocols, DoH and DoT. Using the *dnspython* lookup tool, the authors evaluate the DoH landscape and compare it with the DoT landscape. Furthermore, the study quantifies the impact of DoH on Web page load times, indicating that the protocol provides enhanced security with minimal impact on loading time performance.

In (SHARMA; FEAMSTER; HOUNSEL, 2022), encrypted DNS resolvers that support DoH are evaluated to address privacy concerns. The research shows that while some non-mainstream resolvers have higher response times, there are exceptions, indicating the possibility for users to utilize a broader range of encrypted DNS resolvers than those currently available in popular browser configurations.

(HOUNSEL et al., 2021) investigates the performance of encrypted DNS protocols and conventional DNS in home networks from the United States of America (USA). The research, conducted using a proprietary tool called *SamKnows*, revealed that privacy-focused DNS protocols, such as DoT, could outperform conventional DNS in terms of response times for certain resolvers, even with increased latency. The study underscores the need for DNS clients (*e.g.*, browsers) to evaluate latency and response times, suggesting that no single DNS protocol or resolver universally outperforms others for all clients.

Similarly, (DOAN; TSAREVA; BAJPAI, 2021) analyzes DoT adoption and performance, leveraging 3200 Réseaux IP Européens Network Coordination Center (RIPE) Atlas probes in home networks. That research reveals a 23.1% increase in DoT support among open resolvers and a low adoption of local resolvers at 0.4%. Although DoT exhibits higher failure rates and response times, local resolvers achieve response times comparable to public ones despite higher failure rates. Thus, it highlights the complexities and regional disparities in DoT implementation.

(HOUNSEL et al., 2020) explores the impact of the Do53, DoT, and DoH DNS protocols on query response times and page load times from global perspectives using the same tool as (SHARMA; FEAMSTER; HOUNSEL, 2022). Although DoH and DoT exhibit slightly higher response times than Do53, they can outperform Do53 in terms of page load times. However, in the conditions of reduced throughput and increased latency, Do53 becomes the fastest option for web page loading. Furthermore, Do53 and DoT show higher success rates in loading web pages compared to DoH. The research suggests strategies for enhancing DNS performance, including opportunistic partial responses and wire format caching, to address varying conditions and improve user experience.

Still in DoH, (BORGOLTE et al., 2019) discusses the policy implications of DoH. The authors systematically analyze DoH DNS resolvers, measure DoH's performance effects on Web page loading times using the Firefox Web browser, examine the competitive landscape of such an area, and explore the impact on stakeholders, such as ISPs and consumers. The work sheds light on the potential regulatory and policy implications of widespread DoH deployments.

(CHHABRA et al., 2021) investigates the performance, using the BrightData network, of DoH using a comprehensive dataset from 22 052 clients across 224 countries and territories. That research reveals mixed impacts on the performance of DoH-enabled DNS resolvers and highlights geographic disparities in DoH and Do53 resolution times, with clients from countries with low investment in Internet infrastructure being more prone to slowdowns when switching to DoH.

Lastly, (LU et al., 2019) is a large-scale study that collects data from Internet scanning, user-end measurement and passive monitoring logs to offer insights into encrypted DNS adoption and usage, as well as performance overhead, accessibility and latency. It is also a comparative study on different encrypted DNS protocols.

Table 3.1 – Review of Literature on DNS Resolvers Performance Research

Reference	Protocol	Lookup Tool	Analyzed Resolvers	Source Code	Domain Dataset	Distributed Measurements
(AFFINITO; BOTTA; VENTRE, 2022)	Do53, DoH	pydig	Google, OpenDNS	No	Yes	No
(BöTTGER et al., 2019)	Do53, DoH, DoT	dnspython	Several Resolvers	No	Yes	No
(AGER et al., 2010)	Do53	dig	Google, OpenDNS	No	Yes	Yes
(HOUNSEL et al., 2021)	Do53, DoH, DoT	SamKnows	Anonymized Public Resolvers	No	Yes	Yes
(HOUNSEL et al., 2020)	Do53, DoH, DoT	dns-measurement	Google, Cloudflare, Quad9	Yes	Yes	Yes
(BORGOLTE et al., 2019)	Do53, DoH	Firefox	Google, Cloudflare, Quad9	No	No	No
(SHARMA; FEAMSTER; HOUNSEL, 2022)	Do53, DoH	dns-measurement	Several Resolvers	Yes	Yes	Yes
(CHHABRA et al., 2021)	Do53, DoH	BrightData	Google, Cloudflare, Quad9, NextDNS	No	No	Yes
(DOAN; TSAREVA; BAJPAI, 2021)	Do53, DoT	RIPE Atlas	Google, Cloudflare, Quad9, CleanBrowsing, UncensoredDNS	Yes	Yes	Yes
(LU et al., 2019)	Do53, DoT, DoH, DoQ	RIPE Atlas	Google, Cloudflare, Quad9	Yes	Yes	Yes

DNS-over-Port 53 (Do53)

Table 3.1 presents a detailed review of these efforts and the tools used. In summary, (AFFINITO; BOTTA; VENTRE, 2022) and (AGER et al., 2010) focused on comparing local and public resolver performance. (BöTTGER et al., 2019; SHARMA; FEAMSTER; HOUNSEL, 2022; DOAN; TSAREVA; BAJPAI, 2021; CHHABRA et al., 2021; HOUNSEL et al., 2021) investigated the performance impact of using encrypted DNS through HTTPS or TLS protocols, while (HOUNSEL et al., 2020) and (BORGOLTE et al., 2019) do so with additional attention to Web page loading times. (LU et al., 2019) mainly evaluates adoption and popularity of encrypted DNS. Therefore, it can be stated that the research on evaluating the overhead and performance of different DNS protocols is relevant as there are recent efforts. However, there is room to address other aspects, such as the impact of the selected lookup tools and the availability of the source code of solutions to foster reproducibility.

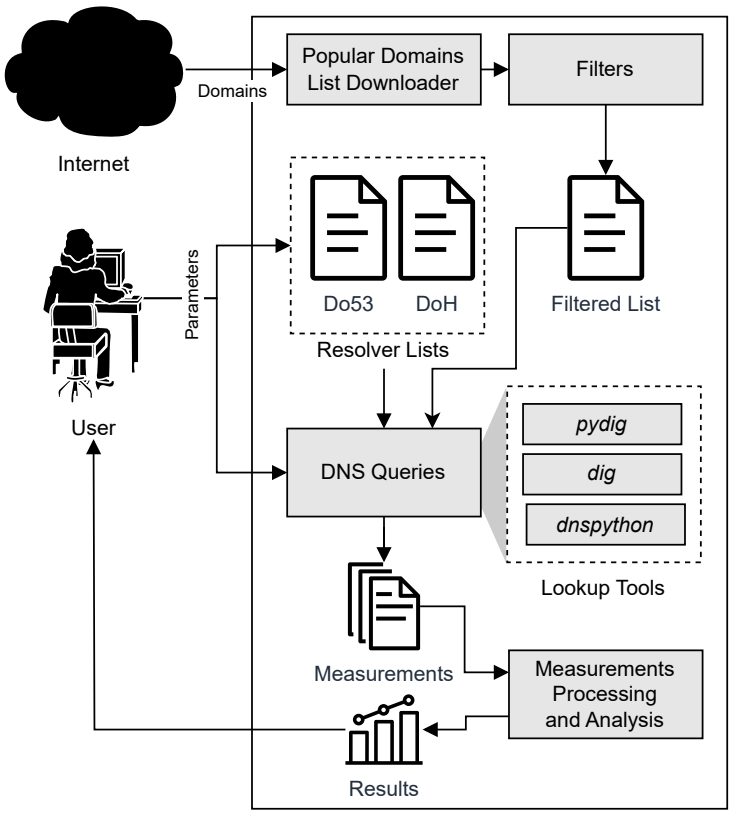
4 ANALYZING DNS PERFORMANCE AND OVERHEAD

This chapter details the methodology, describes the design and implementation of the solution created to perform the data collection and analysis, and the reasoning behind the selection of the different tools used. All source code, domain and resolver datasets, as well as the measurement results presented and discussed in the next section (*i.e.*, Section 5), are available at (PINTO, 2023) to promote the reproducibility of our experiments.

4.1 Methodology and Design

The performance metric most relevant for this study is the RT (Response Time) of a DNS lookup, which consists of the time elapsed between issuing the query and receiving a response from the resolver. By analyzing the mean RTs and query success rates we are able to compare the performance of different encrypted DNS protocols or lookup tools, as well as resolver availability.

Figure 4.1 – Measurement Tool Design



Source: (The Author, 2024)

To issue DNS queries (either Do53, DoH or DoT), we designed and implemented a Python measurement tool that executes and collects the results of each combination of tools (*e.g.*, *dig*, *pydig*, and *dnspython*), resolvers (*e.g.*, Google, Quad9, and Adguard) and domains (*e.g.*, *google.com* and *wikipedia.org*). Figure 4.1 depicts the design of our tool, including its main components and actors. The tool automatically retrieves the latest list of popular domains curated by the Tranco list project (Le Pochat et al., 2019), after that, filters (*e.g.*, selection of *n* top domains) are applied in the list to reduce or increase the number of domains to be analyzed. Further, the tool allows the user to input all the parameters, including resolver and domain lists and the number of DNS requests to be made for each resolver, domain and tool.

After setting up all the parameters, the tool starts the process of issuing all the queries sequentially and collecting the relevant data from the responses. It then stores the data (*e.g.*, timestamp, return status code, RT, answer IPs) as Comma-Separated Values (CSV) files for post-processing and analysis. Because the data we are working with consists of relatively small CSV files, there was no need to include a database in the project design. The data processing and analysis is performed in another component, which filters and derives statistics for each studied aspect (*e.g.*, protocol and tool performance comparisons). The different benchmarks are presented as charts and saved as PDF files for the user. As the tool was designed to be extensible, other lookup tools can be added by creating new wrapper scripts for each tool and including them in the measurement component.

4.2 Implementation

The following sections details the description of different implementation aspects of the solution. First, we describe the different lookup tools used alongside usage examples. Then, we take a look at how the DNS query measurements were performed with a code snippet. Next, we detail the analysis part of the project where all the gathered data is processed and the results are displayed as chart visualizations. Lastly, we propose different ideas for possible improvements and extensions to the solution.

4.2.1 Lookup Tools

Regarding DNS lookup tools, we could observe different approaches based on Table 3.1, including the use of proprietary monitoring software such as SamKnows (Cisco, 2023) and BrightData (Bright Data Ltd., 2023) but also a non-commercial distributed monitoring tool, called RIPE Atlas (Réseaux IP Européens Network Coordination Centre RIPE NCC, 2023). For this work, we selected open source and accessible tools, which are the Python libraries *pydig* and *dnspython*, as well as the native *dig* Linux command.

pydig (SMITH, 2021) is a Python wrapper library for the *dig* command-line tool. Therefore, it relies on the native *dig* Linux tool, provided by the *bind* package (Internet Systems Consortium, Inc., 2023), which allows users to gather detailed information about DNS records, server response times, and domain configurations. Our project uses version 0.4.0 of *pydig*.

```
1 resolver = pydig.Resolver(
2   executable='/usr/bin/dig',
3   nameservers=[
4     '1.1.1.1',
5     additional_args=[
6       '+tries=1',
7       '+timeout=3'
8   )
9  result = resolver.query('inf.ufrgs.br', 'A')
```

Listing 4.1 – Do53 query in Python using *pydig*

Listing 4.1 shows an example of a Do53 query using *pydig*. In Line 1 a `resolver` object is created with arguments pointing to the *dig* executable path in Line 2, the nameserver address in Line 4, as well as the additional *dig* parameters for maximum retry attempts in Line 6 and time to connection timeout in Line 7. In Line 9 the query is executed in the *resolver* object's `query` method and its return value stored in the `result` variable.

dnspython (Dnspython Contributors, 2020) is a Python library that implements a full-fledged DNS toolkit from scratch. The toolkit can be used for several DNS-related actions, such as queries, zone transfers, and nameserver testing. The toolkit implements its communication using sockets to perform queries to DNS resolvers and interact with DNS servers. Our project uses version 2.4.2 of *dnspython*.

```
1 query = dns.message.make_query('inf.ufrgs.br', dns.rdatatype.NS)
2 result = dns.query.udp(query, '1.1.1.1', timeout=3)
```

Listing 4.2 – Do53 query in Python using *dnspython*

Listing 4.2 shows an example of a Do53 query using *dnspython*, which is called just `dns` in the example's code. In Line 1 a `query` object is created with arguments providing the domain (*inf.ufrgs.br*) to be resolved. In Line 2 the `udp` method is called which takes as argument the `query` object, the resolver address (*1.1.1.1*) and a `timeout` value of 3 seconds.

4.2.2 Measurements

The main measurement process occurs in the Python module called *encrypted-dns-measurement*, which executes queries for all combinations of resolvers and domains from the input lists using each protocol and lookup tool. This process is repeated for the sample size defined by the user.

```
1 for resolver in do53_resolvers:
2     results = []
3     for domain in domains:
4         q = pydig_wrapper.query('do53',domain,resolver)
5         results.append(q)
6     export_results('pydig','do53',resolver,results)
```

Listing 4.3 – Measurements of Do53 queries using *pydig*

Listing 4.3 exemplifies part of this process. For every `resolver` and `domain`, the `query` method of the tool wrapper for *pydig* is called and the return values appended to the `results` list. In Line 6 the `export_results` method is executed which writes the data to the CSV output files.

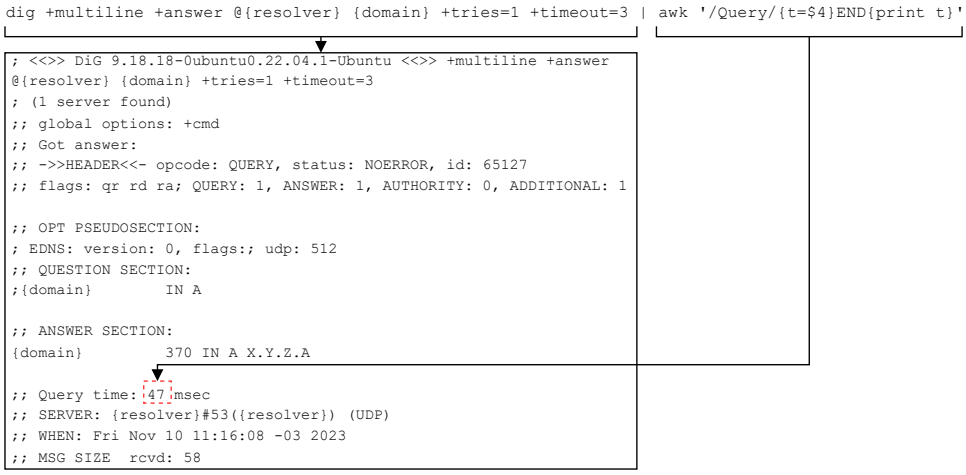
To obtain accurate RTs, we measure them using Python's `time` (Python Software Foundation, 2023). Listing 4.4 illustrates the methodology for measuring the RT of a Do53 query using the `time` module. In Line 3, the query is constructed; in Line 4, the start time is recorded to fetch the NS; in Line 5, the query is sent; in Line 6, the end time of the query is captured, and finally, in Line 7, the temporal difference between the end and start times is computed.

```
1 mq = dns.message.make_query(domain, dns.rdatatype.NS)
2 start_time = time.time()
3 q = dns.query.udp(mq, resolver, timeout=3)
4 end_time = time.time()
5 res_time = end_time*1000 - start_time*1000
```

Listing 4.4 – RT Measurement of a Request using *dnspython*

To serve as a baseline and to compare with the tool RTs measured in Python's `time` (Python Software Foundation, 2023), we also used *awk* (Free Software Foundation, Inc, 2009), a program that implements the AWK domain-specific language, to extract *dig*'s reported query time as depicted in Figure 4.2. The figure illustrates the command used to retrieve the *dig* query time in the

Figure 4.2 – Extraction of *dig*'s Reported Query Time using *awk*

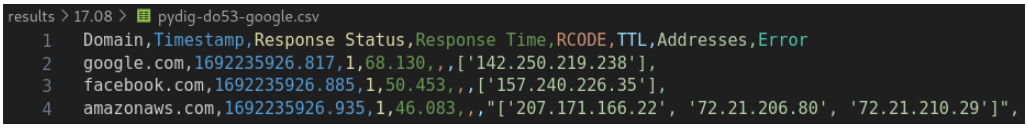


Source: (The Author, 2024)

top of the figure, and, in the solid-line square, the result of a DNS query of a `{domain}` to a `{resolver}` using *dig*. The query time of that specific query is highlighted in the red-dashed square, which is extracted using the *awk* command depicted at the top.

The finished measurement process output is a `/results/` folder of CSV files with the naming convention being *{tool}-{protocol}-{resolver}.csv*, which are then fed into the analysis component of the project.

Figure 4.3 – Output CSV file example: headers and data



Source: (The Author, 2024)

Figure 4.3 shows the content of an example CSV file `pydig-do53-google.csv`. The file name indicates that this data corresponds to queries made using *pydig* tool and Do53 protocol. In Line 1 we can observe the header names that represent each column.

4.2.3 Analysis

The data analysis and visualization component of the project occurs in the Python module called *encrypted-dns-benchmark*, after the measurements and results collection is done.

To manipulate and query our data we use the open source data analysis Python library *pandas* (Pandas Contributors, 2009) in its version 2.1.0.

```
1 dfs = list()
2 for t in tools:
```

```
3     for p in protocols:
4         for r in resolvers:
5             csv_file = f'./results/{t}-{p}-{r}.csv'
6             data = pandas.read_csv(csv_file)
7             data['Tool'] = t
8             data['Protocol'] = p
9             data['Resolver'] = r
10            data['Timestamp'] = pandas.to_datetime(data['Timestamp'])
11            dfs.append(data)
12 df = pandas.concat(dfs, ignore_index=True)
```

Listing 4.5 – Reading CSV files' data to *pandas*

Listing 4.5 shows how the component gathers and concatenates all the data from the `/results/` folder of CSV files. In Line 1 the `dfs` list is created which will hold all the `pandas.DataFrame` objects, each representing the data from a single CSV file. For each combination of lookup tools, protocols and resolvers, the CSV file is read into a `DataFrame` object in Line 6 with the additional data columns `Tool`, `Protocol` and `Resolver` added for the identification of each row in the final unified `DataFrame`. In Line 12 the `concat` method is called to join all objects.

With this unified `DataFrame` of all data we are able to filter and query it using *pandas* as if it were a database.

```
1 for p in protocols:
2     df2 = df.query(f'Protocol == "{p}" & {query_success_conditions}')
      [["Tool", "Response Time", "Resolver"]]
3     final_df = pd.pivot_table(
4         df2,
5         values="Response Time",
6         index="Resolver",
7         columns="Tool",
8         aggfunc=('count','mean','std')
9     )
```

Listing 4.6 – Querying and shaping data with *pandas*

Listing 4.6 is a snippet from the tool performance comparison method. For each protocol, it generates the `DataFrame` needed to plot the chart that compares mean RTs of each tool and resolver combination. In Line 2 the `query` method is used on our unified dataframe to filter all queries for the selected protocol, excluding unsuccessful queries. Then in Line 3 we aggregate and reshape the `DataFrame` so we can have statistics such as total rows, mean RTs and RT standard deviations available for each tool and resolver combination. This new `DataFrame` is ready to be

plotted so we can visualize and compare lookup tool performance.

For data visualization of our results we chose the *matplotlib* (Matplotlib Contributors, 2012) *Python* library and used version 3.7.3.

```
1 colors=['darkgray','gray','dimgray','lightgray']
2 ax = df_pivot['mean'].plot(kind="bar", color=colors, yerr=df_pivot['
    ci95_range'], capsize=4)
3 f = ax.get_figure()
4 ax.set_xlabel("Resolvers")
5 ax.set_ylabel("Response Time (ms)")
6 plt.subplots_adjust(bottom=0.3)
7 plt.show()
8 f.savefig(f'results/tool_benchmark_{protocol}.pdf', bbox_inches='
    tight', dpi=300)
```

Listing 4.7 – Plotting results with *matplotlib*

Listing 4.7 shows how the final processed data is stylized and turned into human-friendly visualizations, saved to Portable Document Format (PDF) files. We chose bar charts in gray scale tones for a clean look, set the Y and X axis labels, and adjust the size and positioning. Finally, in Line 8 each chart plot is saved in PDF with the naming convention `tool_benchmark_{protocol}`.

4.2.4 Extending the Tool

It is possible to include different lookup tools to the project by creating Python wrapper files for each. For the wrapper to be compatible, it has to implement the functions that issue queries for each protocol supported and return the results in the format specified.

```
def get_query_result_dict():
    return {'Domain': None,
            'Timestamp': None,
            'Response Status': None,
            'Response Time': None,
            'RCODE': None,
            'TTL': None,
            'Addresses': [],
            'Error': None
        }
def query(protocol, domain, resolver, endpoint=''):
    if protocol == 'do53':
        return do53_query(domain, resolver)
    elif protocol == 'doh':
```

```
15          return doh_query(domain, resolver, endpoint)
16      elif protocol == 'dot':
17          return dot_query(domain, resolver)
18  def do53_query(domain, resolver):
19      result = get_query_result_dict()
20      # QUERY CODE
21      return result
22  def doh_query(domain, resolver, endpoint):
23      result = get_query_result_dict()
24      # QUERY CODE
25      return result
26  def dot_query(domain, resolver):
27      result = get_query_result_dict()
28      # QUERY CODE
29      return result
```

Listing 4.8 – Tool Wrapper Example

Listing 4.8 shows the organization of the wrapper file. For each query function (Line 18, Line 22 and Line 26) the developer must implement the necessary code to execute queries in the new lookup tool, format the results and return them according to the model in the dictionary stored in `result`. If the tool is not a Python package, it is recommended the use of the `subprocess` module to invoke the tool process, as is done for the *dig* tool wrapper.

There is room for improvements of the solution. Firstly, better test coverage and error handling could be integrated. The packaging of the solution for virtualization or containerization would allow for straightforward distributed measurements, possibly in cloud services such as Amazon Web Services (AWS) (AMAZON, 2023). Lastly, additional lookup tools and protocols could be supported, as well as longer default lists of DNS resolvers and domains to perform the measurements.

5 EVALUATION

This section describes the setup of the experiments in Section 5.1, presents the results of lookup tool and protocol performance comparisons in Section 5.2 and Section 5.3, discusses the lookup success rates in Section 5.4, explores the latency to resolvers in Section 5.5, and adds further discussion about the results in Section 5.6, while also outlining the limitations of our work.

5.1 Experiment Setup

The setup of the experiment was an 2.3 GHz Intel® Core™ i5-6200U machine running Debian 11 Linux operating system, with 16 GB of RAM. To have a stable network connection, the experiments were performed using an 100 Mbps Ethernet cable connected to the Internet with the vantage point being the Institute of Informatics (INF) network of the Federal University Rio Grande do Sul (UFRGS) located in Porto Alegre, in the south of Brazil, which has a 10 Gbps dedicated link but no special Quality-of-Service (QoS) treatment for DNS-related traffic.

It should be mentioned that the local cache was disabled in the operating system (*i.e.*, flag `cache` set to `no` in the `/etc/systemd/resolved.conf` file) of the machine used in the measurements so that this cache does not affect the results.

Our resolver data set consists of five widely used public resolvers that we also derived from literature research (*cf.* Table 3.1), being Google, Cloudflare, Quad9, CleanBrowsing, and Adguard. For our domain dataset, we selected the most popular domain on the Tranco list (Le Pochat et al., 2019), retrieved from September 13, 2023, which was *www.google.com*. However, the lists of resolvers and domains can be adjusted by the user or given as input files at program startup.

The measurements are taken by issuing queries for each combination of DNS resolver, DNS protocol, and lookup tool. This action is then repeated 500 times in a loop so that a significant sample size of results is obtained with a narrow 95% confidence interval during the analysis process.

5.2 Lookup Tool Performance

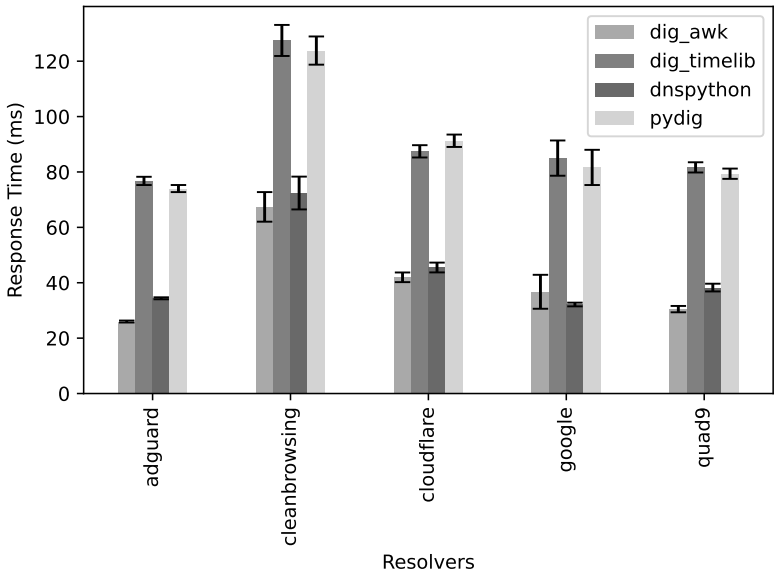
The first analysis conducted consisted of comparing the performance of different lookup tools to verify whether the choice of tool meaningfully impacts DNS query results. We chose to plot three charts, one for each DNS protocol examined (Do53, DoH, DoT), comparing the mean RTs of each tool for the queries to a given resolver.

The results of the experiments are shown in Figures 5.1, 5.2 and 5.3, respectively. For each resolver in the *x* axis, different bars are plotted for each tool, and the *y* axis represents the mean RT measured in milliseconds. The *dig_awk* bar represents the measurements using the *dig* command to get the DNS query time, *dig_timelib* represents the measurements using the Python's `time` library (Python Software Foundation, 2023) and calling *dig* from the Python's `subprocess` module (ASTRAND, 2003), the *dnspython* bar represents the measurements performed using the *dnspython* library and, lastly, the *pydig* bar represents the measurements with the *pydig* library.

5.2.1 DNS over Port 53 (Do53)

Figure 5.1 shows that the measured RTs of the Do53 queries vary for both different tools and different resolvers. For the same resolver, we can observe RT differences as high as 153% between different tools, as can be seen in the case of *dnspython* (32 ms) and *pydig* (81 ms) mean RTs for the Google resolver. On average, we can see that the best performance comes from the RTs reported of native *dig*, closely followed by *dnspython*. The performance of *Pydig* was consistently worse, and similar to that of our implemented *dig* tool measurements using the Python time module.

Figure 5.1 – Do53 Lookup Tool Performance

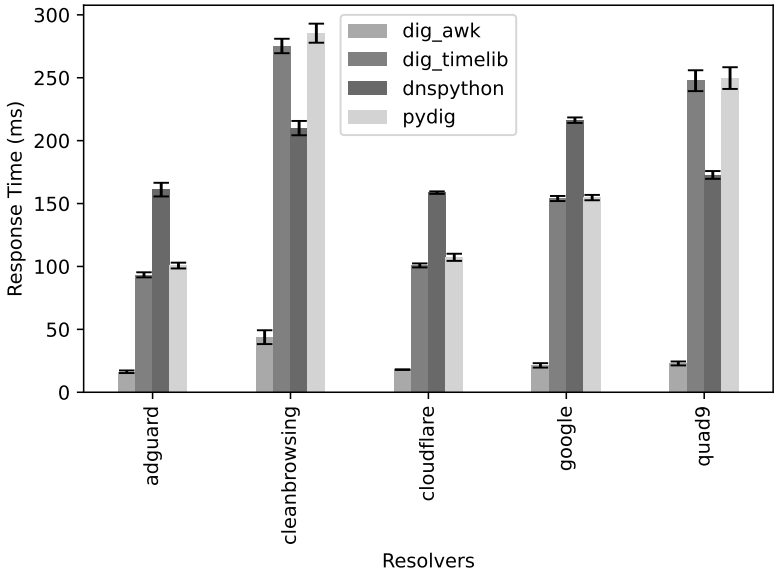


Source: (The Author, 2024)

5.2.2 DNS over HTTPS (DoH)

When employing the DoH protocol to perform queries using the selected tools, a different behavior can be observed, as shown in Figure 5.2. For example, there was a difference of 1012% between the result of the *dig* reported query time (21 ms) and *dnspython* reported RT (216 ms) for the Google resolver. This behavior is different from the behavior observed using the Do53 protocol, where *dig* and *dnspython* presented similar results. Such a behavior can be explained by *dnspython*'s implementation of the DoH query, which includes the time required to create all the sessions and exchange the keys using HTTPS; whereas the *dig* reported RT only measures the query time.

Figure 5.2 – DoH Lookup Tool Performance

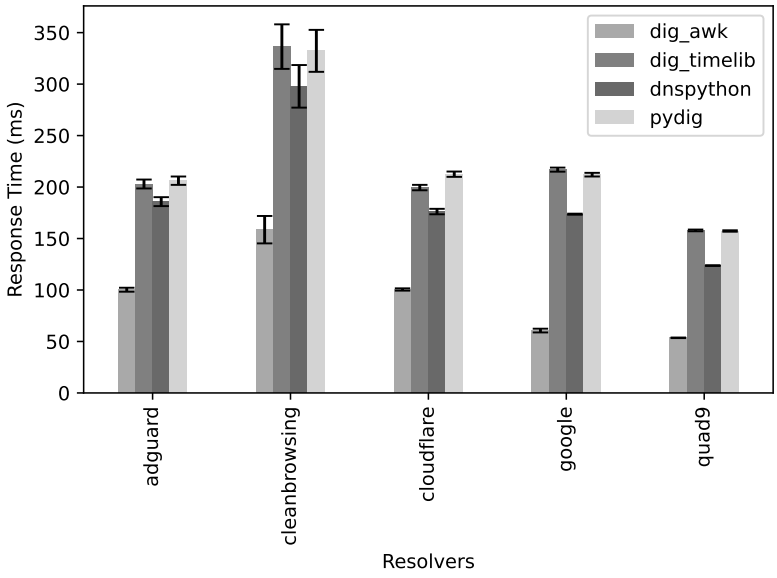


Source: (The Author, 2024)

5.2.3 DNS over TLS (DoT)

Finally, the DoT query results are presented in Figure 5.3. The best performance remains the native *dig*. The gap between native *dig* and *dnspython* is still higher on average than the Do53 results, which further corroborates the theory that the TLS handshake overhead is not measured by *dig*, only *dnspython*. The performances of *Pydig* and *dig* tool measurements using the Python time module are slower on average.

Figure 5.3 – DoT Lookup Tool Performance



Source: (The Author, 2024)

5.3 DNS Protocol Performance

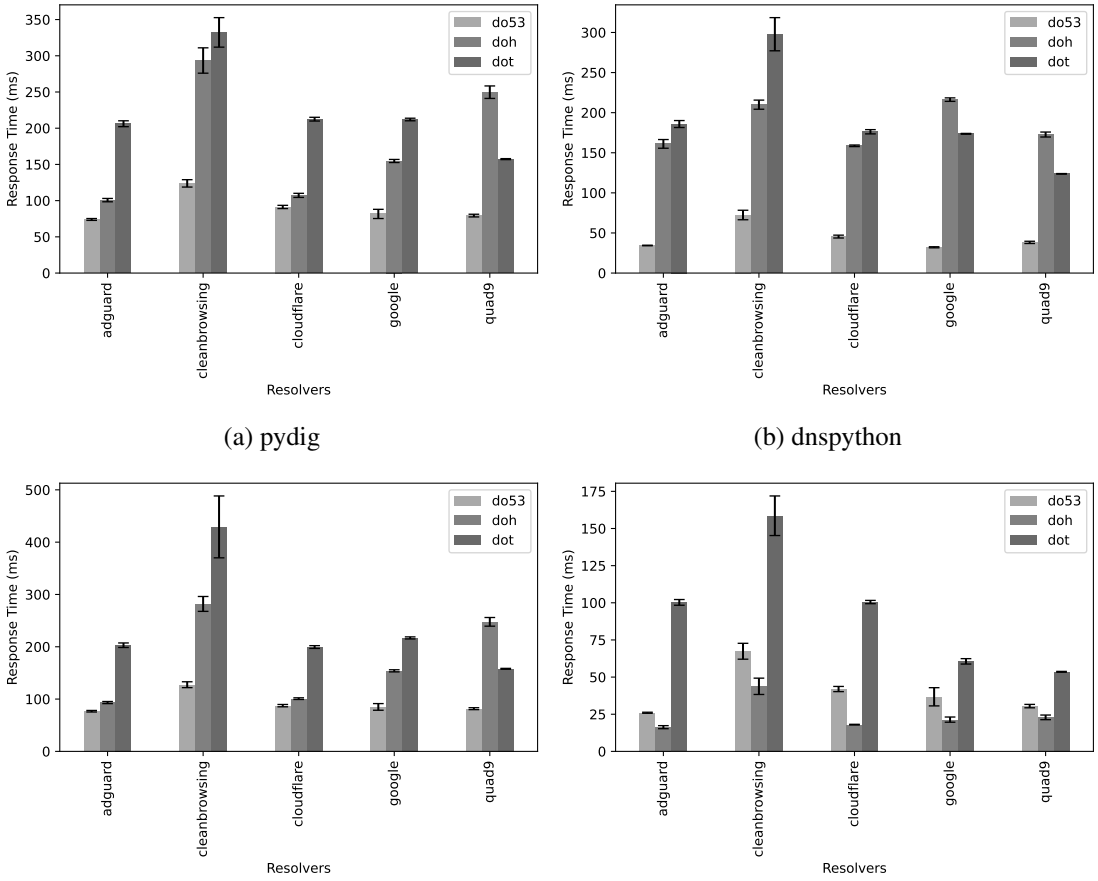
The main question of our work remains whether encrypted DNS protocols introduce significant overhead when compared to Do53. Similarly to the lookup tool analysis, the main metric to analyze is the mean RT of the executed DNS queries. For our analysis, we plot a chart for each lookup tool used that compares the mean RTs of each protocol, as well as one with the aggregated performances of every tool.

The results are displayed in Figure 5.4a, Figure 5.4b, Figure 5.4c, Figure 5.4d and, the aggregated results in Figure 5.5. For each resolver in the x-axis, different bars are plotted for each protocol, and the y-axis represents the mean RT measured in milliseconds.

Figure 5.4d is the only odd-looking chart, where DoH performed better than unencrypted Do53 on average. This can likely be explained by the method of extraction for the query RT metric. While the time elapsed for all the queries using other tools was measured using Python's `time` (Python Software Foundation, 2023), we extract *dig*'s reported query time using *awk* (Free Software Foundation, Inc, 2009). This means that for other tools, the RT metric for DoH and DoT includes the time for connection setup and TLS handshake, while *dig* only reports the actual query time, after it already has a connection to the resolver server.

As can be seen in the aggregated mean RT results in Figure 5.5, Do53 had the best average performance (66.7 ms), followed by DoH (141.2 ms), and lastly DoT (163.23 ms). On average, DoH performed 111.69% slower than its unencrypted counterpart, and DoT performed 144.72% slower. Interestingly, only one resolver showed superior encrypted DNS performance using DoT instead of DoH (Quad9), where DoT queries on average performed 50.14 ms or 28.93% faster.

Figure 5.4 – Results of Protocol Performance using Different Tools



(a) pydig

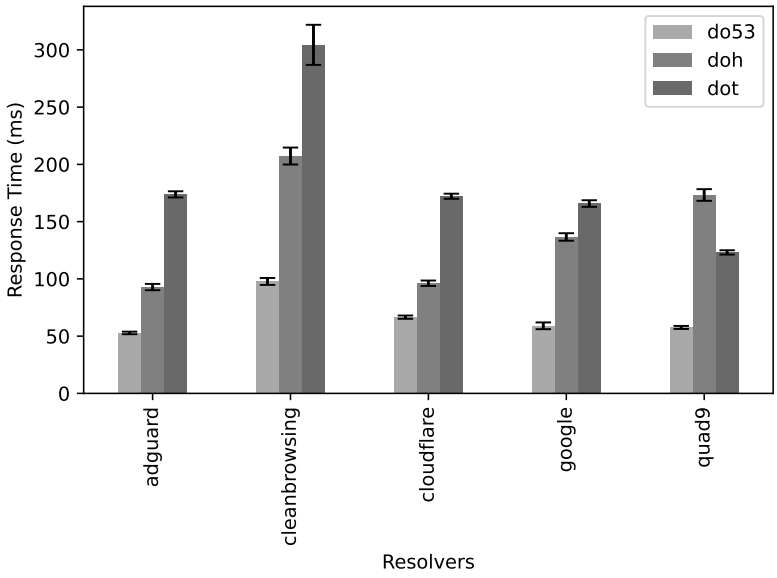
(b) dnspython

(c) dig (timelib)

(d) dig (awk)

Source: (The Author, 2024)

Figure 5.5 – Protocol Performance: aggregated



Source: (The Author, 2024)

5.4 Lookup Success Rates

In addition to analyzing the RT of the queries, we analyzed the lookup success rate of all the queries issued. This analysis on the lookup success rate provides a in-depth understanding of the reliability and effectiveness of the selected tools and selected resolvers in successfully resolving DNS queries. Thus, providing another key contribution of our research and adding another aspect to be considered when selecting the DNS lookup tool.

Table 5.1 – Lookup Success Rate based on Resolver and Tool using Do53

		Lookup Tool			
Resolver	**IP Address**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	94.140.14.14	99.8%	99.8%	99.8%	100%
Cleanbrowsing	185.228.168.168	93.0%	93.0%	91.6%	91.6%
Cloudflare	1.1.1.1	99.0%	99.0%	98.8%	99.2%
Google	8.8.8.8	92.2%	92.2%	90.2%	92.2%
Quad9	9.9.9.9	92.4%	92.4%	92.0%	94.6%
	Mean Rate	95.3%	95.3%	94.5%	95.5%

Table 5.1 summarizes the success rate of the 500 DNS lookup tasks performed for the *www.google.com* domain using Do53 and each of the selected tools on 5 different DNS resolvers (AdGuard, Cleanbrowsing, Cloudflare, Google, and Quad9). From the mean rate results (last line of Table 5.1), it can be seen that there is no significant difference or correlation between the lookup tool and the success rate. However, the success rate is closely related to the DNS resolver that performs the lookup; the rows highlighted in light gray represent the two best DNS resolvers (*i.e.*, the resolvers with the highest lookup success rate). Adguard using the *pydig* tool was the combination able to resolve all 500 lookups without any error (*e.g.*, timeout); Cloudflare was also consistent, in terms of the lookup success rate, for each lookup tool with a 99% mean success rate between tools.

Table 5.2 – Lookup Success Rate based on Resolver and Tool using DoH

		Lookup Tool			
Resolver	**HTTP Endpoint**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	*https://dns.adguard.com/dns-query*	100%	100%	100%	100%
Cleanbrowsing	*https://doh.cleanbrowsing.org/doh/adult-filter*	100%	99.8%	99.8%	99.8%
Cloudflare	*https://cloudflare-dns.com/dns-query*	100%	100%	100%	100%
Google	*https://dns.google/dns-query*	100%	100%	100%	100%
Quad9	*https://dns.quad9.net/dns-query*	100%	100%	100%	100%
	Mean Rate	100%	99.9%	99.9%	99.9%

The lookup success rate of the same queries to *www.google.com* using the same tools and resolvers was collected using the DoH protocol. Table 5.2 presents the results of the success rate, which was 100% in all cases, except for the Cleanbrowsing resolver (row highlighted in gray),

which failed once during all queries, resulting in a success rate of 99.8%. Compared to Do53, it can be stated that DoH queries are more stable and will return the result of the query successfully given their use of HTTPS using TCP instead of UDP, which has no delivery guarantee.

Table 5.3 – Lookup Success Rate based on Resolver and Tool using DoT

		Lookup Tool			
Resolver	**IP Address**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	*94.140.14.14*	100%	100%	100%	100%
Cleanbrowsing	*185.228.168.168*	100%	97.6%	95.8%	97.8%
Cloudflare	*1.1.1.1*	100%	100%	100%	100%
Google	*8.8.8.8*	100%	100%	100%	100%
Quad9	*9.9.9.9*	100%	100%	100%	100%
	Mean Rate	100%	99.5%	99.2%	99.6%

Finally, the same analysis is repeated for queries that used the DoT protocol. Table 5.3 presents the results of the success rate, which was 100% in all cases, except for the Cleanbrowsing resolver (row highlighted in gray), which failed, due to timeouts, 12 to 21 times out of 500 depending on the tool. Similarly to DoH, we can observe higher success rates than when using the Do53 protocol, likely due to TCP features that try to guarantee delivery, as opposed to UDP which has none.

5.5 Resolver Latency

Another variable which could possibly skew DNS response time results is the latency to resolvers, especially considering the experiments were run from a single vantage point, which could have a bad connection or route to some of the public resolvers. To test whether the latency is impactful, we analyzed the Round Trip Time (RTT) of Internet Control Message Protocol (ICMP) echo requests using the *ping* utility. For each resolver, we transmitted 1000 packets and collected the minimum, average and maximum RTTs, as well as the packet loss amount.

Table 5.4 shows the results obtained. No packet loss was recorded in this sample. Google and Quad9 delivered the best performances, with Adguard and Cloudflare only slightly behind. Cleanbrowsing showed the worst results, being at least 400% slower on average compared to the other resolvers. This difference in performance can be observed in the results of lookup tool and protocol performance comparisons, where the mean RTs of queries issued to the Cleanbrowsing resolver performed consistently worse in every analysis.

Table 5.4 – Resolver Latency: ICMP ping RTT

			RTT (ms)		
Resolver	**IP Address**	*loss*	*min*	*max*	*avg*
Adguard	94.140.14.14	0%	26.58	31.58	27.21
Cleanbrowsing	185.228.168.168	0%	135.27	151.85	136.68
Cloudflare	1.1.1.1	0%	23.06	60.73	29.95
Google	8.8.8.8	0%	23.57	28.85	24.77
Quad9	9.9.9.9	0%	23.79	39.85	24.56
	Average	0%	46.45	62.57	48.63

5.6 Discussion and Limitations

The observed mean RTs across different lookup tools are indicative of lookup tool selection having a significant impact on DNS query performance measurements. We were able to identify performance differences of up to 1012% for mean RTs of different tools using the same DNS protocol and public resolver. Our analysis of the lookup success rates shows that queries issued to encrypted DNS protocols fail, on average, less often than queries using plain text Do53. This is likely due to TCP and HTTPS features that try to guarantee delivery. Additionally, the resolver latency tests strongly suggest that the RTT of packets sent to resolvers directly impacts query RTs. This is because the resolver with the highest latency (Cleanbrowsing) performed consistently worse across all different lookup tool and protocols.

Based on the results found, it can be concluded that encrypted DNS protocols do introduce an amount of overhead in DNS queries. This much was expected, given the additional steps and features of TCP, TLS and HTTPS protocols when compared to plain UDP traffic. It can be argued, however, whether this overhead is significant or tolerable for end users, especially those browsing the web. With the increasing complexity of websites, the loading time of webpages is nowadays measured in the order of seconds (KELTON et al., 2017). Considering that the average performance impact measured going from Do53 to DoH and DoT were 74.5 ms and 96.53 ms respectively, we posit that the extra time taken to encrypt DNS queries is both tolerable to the average user and very worthwhile when considering the other benefits of encryption such as privacy and security.

A limitation of our work is the fact that the measurements were conducted from a single vantage point, which means the results obtained are susceptible to be skewed by other factors such as local network conditions or DNS resolver availability in the area. However, the focus of this work is to analyze the difference in RT between tools and protocols, not between DNS resolvers. Therefore, the use of a single vantage point does not invalidate the results presented in this section and the considerations made in this work.

A takeaway from working with different DNS tools is that *dig* is a great choice for simple use cases due to its speed and simplicity, while *dnspython* implements a more complete DNS toolkit which can suit many different needs.

6 SUMMARY, CONCLUSION AND FUTURE WORK

In this work, we analyzed the performance impact that encrypted DNS protocols introduce, as well as other potentially relevant aspects of DNS such as lookup tool performance and resolver latency and availability. The literature on DNS performance measurement was researched to investigate efforts this topic and select the most used lookup tools, DNS resolvers, and domains in the approaches. On the basis of that, three tools (*i.e.*, *dig*, *pydig*, and *dnspython*) and five public DNS resolvers (*i.e.*, Adguard, Cleanbrowsing, Cloudflare, Google, and Quad9) were selected for our analysis. To perform the experiments in a reproducible manner, we designed and implemented an open-source tool that performs DNS lookups using the selected tools and resolvers, repeating the process several times for a significant sample size. Such a tool can be customized and extended, with small modification, to accommodate additional lookup tools, DNS protocols, resolvers, and domains.

Our analysis of lookup success rates showed that DoH and DoT are consistently more stable, with higher success rates of queries when compared to those issued using Do53, likely because of TCP advantages over UDP such as retransmissions. Regarding resolver latency and availability tests, we conclude that the choice of resolver directly impacts DNS query RTs and success rates, with latency to the resolver being an important factor.

On the topic of lookup tool selection, our findings indicate it has a significant effect on DNS performance measurements. This difference in results can be explained due to the tool implementations, which in our case varies from using the Operating System (OS) to call an external DNS lookup tool (*e.g.*, *pydig* and *dig*), to using native Python sockets to create DNS requests (*e.g.*, *dnspython*). Thus, it is suggested that researchers carefully select the tool when designing future experiments and take into account that the results might be affected not because of network conditions or the implementation and deployment of the DNS server but because of the tool they are relying on. Additionally, there was no correlation between the lookup tool and the DNS lookup success rate, the rate being only related to the DNS resolver used.

In regards to the main research question of this work, that is whether encrypted DNS protocols introduce significant overhead or not, our results suggest that they do not. The small performance impact of adding encryption stems primarily from the TLS protocol handshake RTTs, and is negligible when considering the average Internet user. Furthermore, the importance of the privacy and security gained by encrypting the DNS protocol cannot be understated, in a world where the Internet is ubiquitous. Thus, it is suggested that users prefer the use of encrypted DNS protocols (*e.g.*, DoH and DoT) in favor of Do53.

Future work on the present research includes, but is not limited to, *(i)* add diversity of vantage points (*e.g.*, in different countries) for distributed measurements, *(ii)* simulate different

processing and network conditions that could impact DNS measurements, such as different cellular networks (*e.g.*, *3G*, *4G*, and *5G*) and resource-constrained environments such as Internet of Things (IoT) devices, *(iii)* analyze DNS network packets in-depth to gather more granular temporal information, *(iv)* increase the selection of tools being compared and include tools implemented in different languages to assess the impact of the language, and *(v)* increase support for additional DNS protocols, such as DNS-over-QUIC (DoQ) (HUITEMA; DICKINSON; MANKIN, 2022) and Oblivious DNS over HTTPS (ODoH) (KINNEAR et al., 2022).

REFERENCES

AFFINITO, A.; BOTTA, A.; VENTRE, G. Local and Public DNS Resolvers: Do You Trade Off Performance Against Security? In: **IFIP Networking Conference (IFIP Networking 2022)**. Catania, Italy: [s.n.], 2022. p. 1–9.

AGER, B. et al. Comparing DNS Resolvers in the Wild. In: . New York, NY, USA: Association for Computing Machinery, 2010. (IMC '10), p. 15–21. ISBN 9781450304832. Available from Internet: <https://doi.org/10.1145/1879141.1879144>.

AMAZON. **Amazon Web Services**. 2023. <https://aws.amazon.com/>.

ASTRAND, P. **PEP 324 – subprocess - New Process Module**. 2003. <https://peps.python.org/pep-0324/>.

BORGOLTE, K. et al. How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem. **SSRN Electronic Journal**, 01 2019.

BöTTGER, T. et al. An Empirical Study of the Cost of DNS-over-HTTPS. In: **ACM Internet Measurement Conference (IMC 2019)**. Amsterdam, Netherlands: [s.n.], 2019. p. 15—21.

BOZKURT, I. et al. Why Is the Internet so Slow?! In: . [S.l.: s.n.], 2017. p. 173–187. ISBN 978-3-319-54327-7.

Bright Data Ltd. **Bright Data - The World's #1 Web Data Platform**. 2023. <https://brightdata.com/>.

BUTKIEWICZ, M.; MADHYASTHA, H. V.; SEKAR, V. Understanding Website Complexity: Measurements, Metrics, and Implications. In: **ACM Conference on Internet Measurement Conference (IMC 2011)**. Berlin, Germany: [s.n.], 2011. p. 313–328. ISBN 9781450310130.

CHHABRA, R. et al. Measuring DNS-over-HTTPS Performance around the World. In: **ACM Internet Measurement Conference (IMC 2021)**. Virtual Event: [s.n.], 2021. p. 351–365. ISBN 9781450391290. Available from Internet: <https://doi.org/10.1145/3487552.3487849>.

Cisco. **SamKnows - Internet Performance Monitoring**. 2023. <https://www.samknows.com/>.

Dnspython Contributors. **dnspython Python Library**. 2020. <https://www.dnspython.org/>.

DOAN, T. V.; TSAREVA, I.; BAJPAI, V. Measuring dns over tls from the edge: Adoption, reliability, and response times. In: HOHLFELD, O.; LUTU, A.; LEVIN, D. (Ed.). **Passive and Active Measurement**. Cham: Springer International Publishing, 2021. p. 192–209. ISBN 978-3-030-72582-2.

Free Software Foundation, Inc. **gawk - Pattern Scanning and Processing Language** . 2009. <https://linux.die.net/man/1/awk>.

GRIGORIK, I. **TLS Handshake**. 2013. <https://hpbn.co/transport-layer-security-tls/>.

HOFFMAN, P. E.; MCMANUS, P. **RFC 8484: DNS Queries over HTTPS (DoH)**. 2018. <https://www.rfc-editor.org/rfc/rfc8484.txt> Acessado em 4 de Maio de 2023.

HOUNSEL, A. et al. Comparing the Effects of DNS, DoT, and DoH on Web Performance. In: **Proceedings of The Web Conference 2020**. New York, NY, USA: Association for Computing Machinery, 2020. (WWW '20), p. 562–572. ISBN 9781450370233. Available from Internet: <https://doi.org/10.1145/3366423.3380139>.

HOUNSEL, A. et al. Can Encrypted DNS Be Fast? In: **Passive and Active Measurement**. Cham: Springer International Publishing, 2021. p. 444–459. ISBN 978-3-030-72582-2.

HU, Z. et al. **RFC 7858: Specification for DNS over Transport Layer Security (TLS)**. 2016. <https://rfc-editor.org/rfc/rfc7858.txt> Acessado em 4 de Maio de 2023.

HUITEMA, C.; DICKINSON, S.; MANKIN, A. **RFC 9250: Oblivious DNS over HTTPS**. 2022. <https://www.rfc-editor.org/rfc/rfc9250.html> Acessado em 4 de Maio de 2023.

Internet Systems Consortium, Inc. **Bind 9 - Versatile, Classic, Complete Name Server Software**. 2023. <https://www.isc.org/bind/>.

KELTON, C. et al. Improving user perceived page load times using gaze. In: **14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)**. Boston, MA: USENIX Association, 2017. p. 545–559. ISBN 978-1-931971-37-9. Available from Internet: <https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/kelton>.

KHAN, B. **DNS Explained. Hierarchy and Architecture**. 2020. <https://dev.to/blake/dns-explained-hierarchy-and-architecture-18pj>.

KINNEAR, E. et al. **RFC 9230: Oblivious DNS over HTTPS**. 2022. <https://datatracker.ietf.org/doc/html/rfc9230> Acessado em 4 de Maio de 2023.

KUROSE, J. F.; ROSS, K. W. **Computer Networking: A Top-Down Approach**. 7. ed. [S.l.]: Pearson, 2016. ISBN 978-0-13-359414-0.

Le Pochat, V. et al. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In: **Network and Distributed System Security Symposium (NDSS 2019)**. San Diego, USA: [s.n.], 2019.

LIU, C.; ALBITZ, P. **DNS and BIND**. 4. ed. [S.l.]: O'Reilly & Associates, 2001. ISBN 0-596-00158-4.

LU, C. et al. An end-to-end, large-scale measurement of dns-over-encryption: How far have we come? In: **Proceedings of the Internet Measurement Conference**. New York, NY, USA: Association for Computing Machinery, 2019. (IMC '19), p. 22–35. ISBN 9781450369480. Available from Internet: <https://doi.org/10.1145/3355369.3355580>.

Matplotlib Contributors. **matplotlib Python Library**. 2012. <https://matplotlib.org/>.

MOCKAPETRIS, P. **Domain names: Concepts and facilities**. RFC Editor, 1983. RFC 882. (Request for Comments, 882). Available from Internet: <https://www.rfc-editor.org/info/rfc882>.

MOCKAPETRIS, P.; DUNLAP, K. J. Development of the Domain Name System. In: **Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM 1988)**. Stanford, California, USA: [s.n.], 1988. p. 123–133.

Pandas Contributors. **pandas Python Library**. 2009. <https://github.com/pandas-dev/pandas>.

PINTO, J. C. C. **Encrypted DNS Benchmark**. 2023. <https://github.com/jchagastelles/encrypted-dns-benchmark>.

Python Software Foundation. **`time` — Time Access and Conversions**. 2023. <https://docs.python.org/3/library/time.html>.

Réseaux IP Européens Network Coordination Centre RIPE NCC. **RIPE Atlas**. 2023. <https://atlas.ripe.net/>.

SCHMID, G. Thirty Years of DNS Insecurity: Current Issues and Perspectives. **IEEE Communications Surveys & Tutorials**, v. 23, n. 4, p. 2429–2459, August 2021.

SHARMA, R.; FEAMSTER, N.; HOUNSEL, A. **Measuring the Availability and Response Times of Public Encrypted DNS Resolvers**. 2022. ArXiv 2208.04999, cs.CR.

SMITH, L. **pydig - Github Repository**. 2021. <https://github.com/leonsmith/pydig>.

APPENDIX A — PUBLISHED PAPER – ERRC 2023

PINTO, José C. C.; SCHEID, Eder J.; FRANCO, Muriel F.; GRANVILLE, Lisandro Z.. Analyzing and Comparing DNS Lookup Tools in Python. In: ESCOLA REGIONAL DE REDES DE COMPUTADORES (ERRC), 20. , 2023, Porto Alegre/RS. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2023 . p. 49-54. DOI: https://doi.org/10.5753/errc.2023.913.

- **Title:** *Analyzing and Comparing DNS Lookup Tools in Python*
- **Contribution:** A comparison of different DNS lookup tools in Python.
- **Abstract:** The performance of Domain Name System (DNS) resolvers is crucial, as the majority of the communication in the Internet starts with a DNS lookup to resolve a domain an IP address to reach the desired content. In this sense, the academia has been devoted to measure and analyze the performance of DNS resolvers using different tools. However, such tools might present different results due to their implementation and affect the measurements. Hence, this paper provides an analysis and comparison of there different DNS lookup tools employed in the literature and discuss the impact of the tool selection.
- **Status:** Published
- **Qualis:** -
- **Conference:** 20ª Escola Regional de Redes de Computadores (ERRC 2023)
- **Date:** October 23 - October 25, 2023
- **Local:** Porto Alegre, RS, Brasil
- **URL:** <https://sol.sbc.org.br/index.php/errc/article/view/26004>
- **Digital Object Identifier (DOI):** <https://doi.org/10.5753/errc.2023.913>

Analyzing and Comparing DNS Lookup Tools in Python

José C. C. Pinto, Eder J. Scheid, Muriel F. Franco, Lisandro Z. Granville

Informatics Institute (INF) – Federal University of Rio Grande do Sul (UFRGS)
Porto Alegre – RS – Brazil

{jccpinto,ejscheid,mffranco,granville}@inf.ufrgs.br

Abstract. *The performance of Domain Name System (DNS) resolvers is crucial, as the majority of the communication in the Internet starts with a DNS lookup to resolve a domain an IP address to reach the desired content. In this sense, the academia has been devoted to measure and analyze the performance of DNS resolvers using different tools. However, such tools might present different results due to their implementation and affect the measurements. Hence, this paper provides an analysis and comparison of there different DNS lookup tools employed in the literature and discuss the impact of the tool selection.*

1. Introduction

Established in 1983, the Domain Name System (DNS) emerged as a critical component of the Internet [Mockapetris and Dunlap 1988]. Its primary function is to translate user-friendly hostnames (*e.g.*, *google.com*) into their corresponding Internet Protocol (IP) addresses, effectively serving as the "phone book" of the Internet [Kurose and Ross 2016]. Nearly all Internet communication starts with a DNS lookup, and complex websites which require content from multiple third parties might perform hundreds of DNS requests before loading a single page [Butkiewicz et al. 2011]. Thus, DNS performance is of concern as it directly impacts performance in most Internet-based communications [Bozkurt et al. 2017].

Past work has measured DNS performance extensively and under different conditions. For example, [Ager et al. 2010] thoroughly analyzed the performance of DNS with distributed measurements across more than 50 different ISPs, in over 28 countries, comparing local and public DNS resolvers. [Böttger et al. 2019] focused on comparing performance between DNS and its encrypted versions, DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH), and their impact in webpage loading times. [Affinito et al. 2022] measured DoH performance overhead as well as malicious domain protection. However, they all use different DNS lookup tools (*e.g.*, dig, dnspython, and pydig). When evaluating DNS performance, the lookup tool used might introduce additional overhead and skew results, underscoring the necessity for careful tool selection when designing experiments.

In this paper, we compare the performance of different DNS lookup tool libraries in Python. For that, we review the literature to gather the most common used tools and select three of them as focus of our analysis. We collect a dataset of measurements by performing several DNS queries to different public resolvers using the selected tools. Our analysis focuses on Response Time (RT), which is the time elapsed between issuing the DNS request and receiving a response. Our objective is to determine the impact that tool selection could have on DNS performance measurements.

The rest of this paper is structured as follows. Section 2 resents an overview of the DNS and its design. Section 3 describes the tool selection for the experiments. Section 4 details the methodology employed in the measurements and its implementation. Section 5 presents the experiment setup and discusses the results obtained. Lastly, Section 6 summarizes key findings and suggests future work.

2. Domain Name System (DNS)

The DNS is comprised of two main components: a distributed database, structured as a hierarchy of DNS servers, and an application-layer protocol that allows end-hosts (*i.e.*, clients) to query the database, by using local name servers called Resolvers [Kurose and Ross 2016]. To reach a global scale, the DNS database is comprised by a large number of servers distributed around the world, and no single DNS server has the information of all hosts in the Internet. In this sense, the structure of the database is similar to that of the Unix file system, as an inverted tree with the root at the top, and is indexed by domain names. The information, such as IP address, associated with each domain name is stored in Resource Records (RR) [Liu and Albitz 2001].

When a software requires information from the domain namespace, such as a Web browser that needs to translate the domain *www.inf.ufrgs.br* to its IP address, it invokes the client side of DNS resolver, initiating the queries that compose the so-called resolution process. Because of the inverted tree structure of the domain namespace, any domain can be reached by starting the search at the root nameservers. The resolver queries the root servers, which queries the Top-level Domain (TLD) servers (*e.g.*, `.br`) and down the name space tree of servers until an authorative server for the organization (*e.g.*, `.inf`) can return the IP address for *www.inf.ufrgs.br*. Thus, completing the resolution process.

Without additional information, queries start at the root name servers, making them essential to the DNS. However, to offload some of that heavy traffic, caching is very important, as it prevents name servers from querying root nameservers each time it receives a request for an answer it does not have locally. Additionally, caching increases the speed of name resolution as in any part of a query chain a DNS server might have cached the required answer or the address of the authoritative nameserver for the zone, therefore, reducing the number of required queries for resolution [Liu and Albitz 2001].

3. Selection of Lookup Tools

To select the tools to be analyzed in this work, we reviewed the literature on research approaches that focused on analyzing the performance of DNS resolvers, as we were unable to find work on comparing lookup tools directly. Table 1 presents such a review. While all of the works focused measuring DNS performance, they varied in objective and scope. [Affinito et al. 2022] and [Ager et al. 2010] focused on comparing local and public resolver performance. [Böttger et al. 2019], [Sharma et al. 2022], and [Doan et al. 2021] investigated the performance impact of using encrypted DNS through HTTPS or TLS protocols, while [Hounsel et al. 2020] and [Borgolte et al. 2019] do so with additional attention to Web page loading times.

Regarding DNS lookup tools, we could observe different approaches, including the use of proprietary monitoring software such as SamKnows and BrightData but also

a non-commercial distributed monitoring tool, called RIPE Atlas. For this work, we selected open-source and accessible tools, which are the Python libraries *pydig 0.4.0* and *dnspython 2.4.2*, as well as the native *dig* Linux command. Both are available through *PyPI* and *Github* and present good enough documentation, and were relatively simple to setup and experiment with.

Table 1. Review of Literature on DNS Resolvers Performance Research

Reference	Protocol	Lookup Tool	List of Analyzed Resolvers
[Affinito et al. 2022]	Do53, DoH	pydig	Google, OpenDNS
[Böttger et al. 2019]	Do53, DoH, DoT	dnspython	Several Resolvers
[Ager et al. 2010]	Do53	dig	Google, OpenDNS
[Hounsel et al. 2021]	Do53, DoH, DoT	SamKnows	Anonymized Public Resolvers
[Hounsel et al. 2020]	Do53, DoH, DoT	dns-measurement	Google, Cloudflare, Quad9
[Borgolte et al. 2019]	Do53, DoH	Firefox	Google, Cloudflare, Quad9
[Sharma et al. 2022]	Do53, DoH	dns-measurement	Several Resolvers
[Chhabra et al. 2021]	Do53, DoH	BrightData	Google, Cloudflare, Quad9, NextDNS
[Doan et al. 2021]	Do53, DoT	RIPE Atlas	Google, Cloudflare, Quad9, CleanBrowsing, UncensoredDNS

DNS-over-Port 53 (Do53)

4. Methodology and Implementation

Our resolver dataset consists of 6 widely used public resolvers which we also derived from literature research (*cf.* Table 1), being Google, Cloudflare, Quad9, Cisco, CleanBrowsing, and Adguard. For our domain dataset, we selected the most popular domain of the Tranco list [Le Pochat et al. 2019], retrieved from September 13, 2023, which was *www.google.com*. For each one of the DNS resolvers, the lookup to the selected measurements was performed in a loop 500 times so that a significant sample size and confidence interval could be gathered during analysis.

To issue DNS queries from all the tools selected, we implemented a measurement tool that executes and collects the results of each combination of tools (*e.g.*, dig, pydig, and dnspython), resolvers (*e.g.*, Google, Quad9, and Adguard) and domains (*e.g.*, *google.com* and *ufrgs.br*). The tool stores the results as CSV files for posterior analysis. The performance metric relevant for this study is the Response Time (RT) of a lookup, which consists in the time elapsed between issuing the query and receiving a response from the resolver. To obtain accurate RTs we measure them using Python's `time` module for each of the tools selected.

5. Results and Discussion

The setup of the experiment was an 2.3 GHz Intel® Core™ i5-6200U machine running on a Linux Debian 11 operating system, with 16 GB of RAM. The experiments were performed in a single home connection using an Ethernet cable connected to the Internet. All source code, domain and resolver datasets, as well as measurement results are available at `https://github.com/jchagastelles/encrypted-dns-benchmark`.

The experiment results are depicted in Figure 1. For each resolver in the *x*-axis, different bars are depicted, and the *y*-axis represents the response time, in milliseconds, measured by each tool. The *dig_awk* bar represents the measurements using the *dig* command to get the DNS query time, *dig_timelib* represents the measurements using the Python's `time` library [Python Software Foundation 2023] and calling *dig* from the

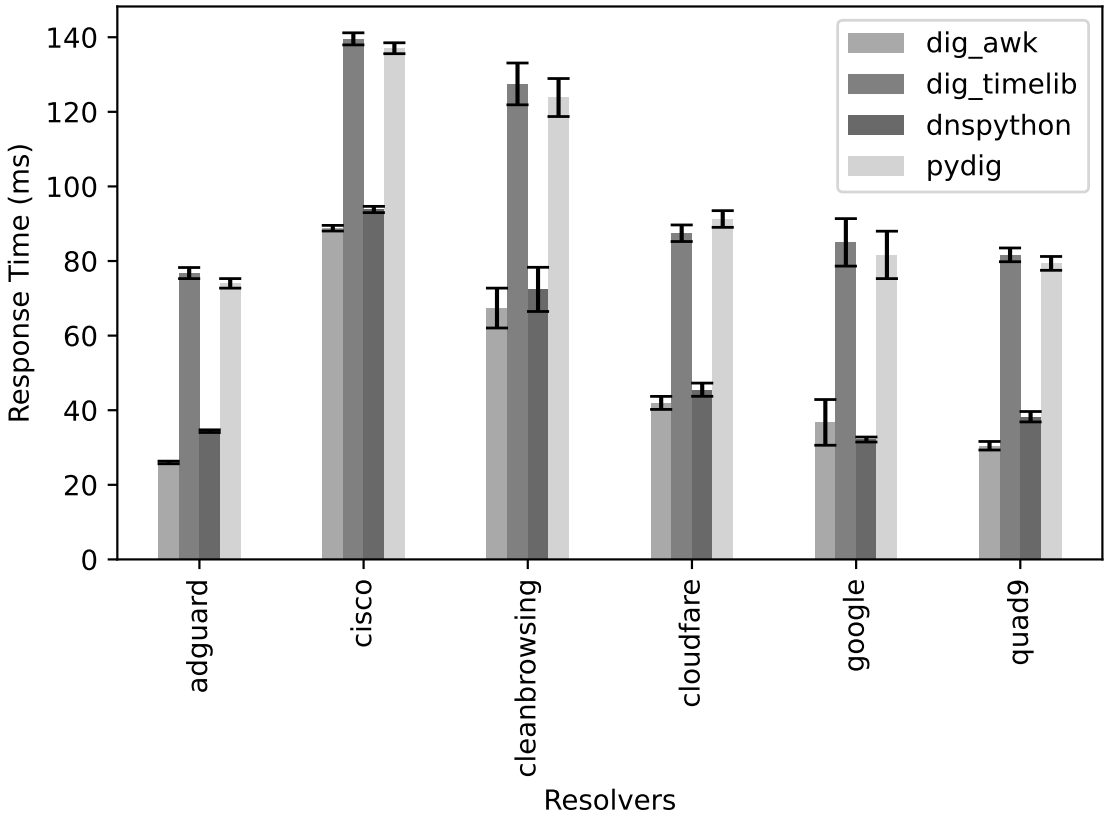


Figure 1. Results of the Experiments using Different DNS Lookup Tools

Python's `subprocess` module [Astrand 2003], the *dnspython* bar represents the measurements performed using the *dnspython* library and, lastly, the *pydig* bar represents the measurements with the *pydig* library.

It can be seen that the measured RTs varied both for different tools and for different resolvers. For the same resolver, we can observe RT differences as high as 153% between different tools, as can be seen in the case of *dnspython* (32ms) and *pydig* (81ms) mean RTs for the Google resolver. On average, we can see that the best performance is from the native *dig*'s reported query RTs, closely followed by *dnspython*. *Pydig* performance was consistently worse, and similar to that of our implemented *dig* tool measurements using the Python time module.

One reason that might explain the performance differences between *dnspython* and *pydig* is the fact that *pydig* acts as a wrapper to *dig*, using the `subprocess` module; thus, it requires system calls (*e.g.*, opening a process and reading from process descriptors) from Python to the Operating System (OS). In contrast, *dnspython* performs the queries directly in native Python code by relying on native UDP and TCP sockets, which results in a faster communication with the resolver compared to *pydig*. Thus, it shows that *dnspython* is the closest one to the native *dig* command (*i.e.*, *dig_awk* in Figure 1).

6. Conclusion and Future Work

In this paper, we analyzed the performance impact of using different DNS lookup tools in DNS performance measurements. The literature on DNS performance measurement

was researched to investigate and select which were the most employed tools and DNS resolvers in the approaches.Based on that, three tools and six DNS resolvers were selected for our analysis.

To perform the experiments in a reproducible manner, we designed and implemented a tool that performs DNS lookups using the selected tools to the six resolvers several times. From the experiments's results, we found significant variation in lookup RTs across different tools and resolvers, with performance impacts as high as 153%.

Based on our findings, we conclude that the tool selection directly impacts results when analyzing DNS performance. This difference in results can be explained due to the tool's implementation, which varies from using the OS to call an external DNS lookup tool (*e.g.*, *dig*) or using native Python sockets to create the DNS requests (*e.g.*, *dnspython*). Thus, it is suggested that researchers carefully select the tool when designing future experiments and take into consideration that results might be impacted.

Future work on the topic includes, *(i)* exploring the tool performance impact on encrypted DoH and DoT protocols, *(ii)* increasing the selection of tools being compared, and *(iii)* adding diversity of vantage points (*e.g.*, in different countries) and network conditions (*e.g.*, mobile networks) of the measurements.

References

Affinito, A., Botta, A., and Ventre, G. (2022). Local and Public DNS Resolvers: Do You Trade Off Performance Against Security? In *IFIP Networking Conference (IFIP Networking 2022)*, pages 1–9, Catania, Italy.

Ager, B., Mühlbauer, W., Smaragdakis, G., and Uhlig, S. (2010). Comparing DNS Resolvers in the Wild. IMC '10, page 15–21, New York, NY, USA. Association for Computing Machinery.

Astrand, P. (2003). PEP 324 – subprocess - New Process Module. `https://peps.python.org/pep-0324/`.

Borgolte, K., Chattopadhyay, T., Feamster, N., Kshirsagar, M., Holland, J., Hounsel, A., and Schmitt, P. (2019). How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem. *SSRN Electronic Journal*.

Böttger, T., Cuadrado, F., Antichi, G., Fernandes, E. L. a., Tyson, G., Castro, I., and Uhlig, S. (2019). An Empirical Study of the Cost of DNS-over-HTTPS. In *Proceedings of the Internet Measurement Conference (IMC 2019)*, pages 15—21, Amsterdam, Netherlands.

Bozkurt, I., Aguirre, A., Chandrasekaran, B., Godfrey, P., Laughlin, G., Maggs, B., and Singla, A. (2017). Why Is the Internet so Slow?! pages 173–187.

Butkiewicz, M., Madhyastha, H. V., and Sekar, V. (2011). Understanding Website Complexity: Measurements, Metrics, and Implications. In *ACM Conference on Internet Measurement Conference (IMC 2011)*, page 313–328, Berlin, Germany.

Chhabra, R., Murley, P., Kumar, D., Bailey, M., and Wang, G. (2021). Measuring DNS-over-HTTPS Performance around the World. In *ACM Internet Measurement Conference (IMC 2021)*, page 351–365, Virtual Event.

Doan, T. V., Tsareva, I., and Bajpai, V. (2021). Measuring dns over tls from the edge: Adoption, reliability, and response times. In Hohlfeld, O., Lutu, A., and Levin, D., editors, *Passive and Active Measurement*, pages 192–209, Cham. Springer International Publishing.

Hounsel, A., Borgolte, K., Schmitt, P., Holland, J., and Feamster, N. (2020). Comparing the effects of dns, dot, and doh on web performance. In *Proceedings of The Web Conference 2020*, WWW '20, page 562–572, New York, NY, USA. Association for Computing Machinery.

Hounsel, A., Schmitt, P., Borgolte, K., and Feamster, N. (2021). Can Encrypted DNS Be Fast? In *Passive and Active Measurement*, pages 444–459, Cham. Springer International Publishing.

Kurose, J. F. and Ross, K. W. (2016). *Computer Networking: A Top-Down Approach*. Pearson, 7 edition.

Le Pochat, V., Van Goethem, T., Tajalizadehkhoob, S., Korczyński, M., and Joosen, W. (2019). Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Network and Distributed System Security Symposium (NDSS 2019)*, San Diego, USA.

Liu, C. and Albitz, P. (2001). *DNS and BIND*. O'Reilly & Associates, 4 edition.

Mockapetris, P. and Dunlap, K. J. (1988). Development of the Domain Name System. In *Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM 1988)*, page 123–133, Stanford, California, USA.

Python Software Foundation (2023). `time` — Time Access and Conversions. `https://docs.python.org/3/library/time.html`.

Sharma, R., Feamster, N., and Hounsel, A. (2022). Measuring the Availability and Response Times of Public Encrypted DNS Resolvers. arXiv 2208.04999, cs.CR.

All links visited on 25/09/2023

APPENDIX B — SUBMITTED PAPER – ISCC 2024

PINTO, José C. C.; SCHEID, Eder J.; FRANCO, Muriel F.; GRANVILLE, Lisandro Z.. Eeny, Meeny, Miny, Moe: Analyzing and Comparing the Selection of DNS Lookup Tools. In: IEEE Symposium on Computers and Communications (ISCC 2024), 2024, Paris, France. pp. 1-6. Submitted. Under review

- **Title:** *Eeny, Meeny, Miny, Moe: Analyzing and Comparing the Selection of DNS Lookup Tools*
- **Contribution:** A comparison of different DNS lookup tools in Python using the classic DNS protocols and encrypted ones.
- **Abstract:** The performance of Domain Name System (DNS) resolvers is crucial, as most of the communication on the Internet starts with a DNS lookup to resolve a domain of an IP address to reach the desired content. In this sense, academia has been devoted to measuring and analyzing the performance of DNS resolvers using different tools, either tailored for each work or generic. However, such tools might present different results due to their implementation and affect the measurements. Therefore, this paper reviews the literature on DNS performance research to gather the tools and DNS resolvers most used and, based on this, provides an analysis and comparison of the different DNS lookup tools employed in the literature and discusses the impact of tool selection on measurement results. Research showed that tool selection has an impact on results but not on the lookup success rate.
- **Status:** Submitted. Under review
- **Qualis:** A2
- **Conference:** 29th IEEE Symposium on Computers and Communications (ISCC 2024)
- **Date:** June 26 - June 29, 2024
- **Local:** Paris, France
- **URL:** -
- **Digital Object Identifier (DOI):** -

Eeny, Meeny, Miny, Moe: Analyzing and Comparing the Selection of DNS Lookup Tools

Jose C. C. Pinto, Eder J. Scheid, Muriel F. Franco, Lisandro Z. Granville

Institute of Informatics (INF), Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
{jccpinto, ejscheid, mffranco, granville}@inf.ufrgs.br

***Abstract*—The performance of Domain Name System (DNS) resolvers is crucial, as most of the communication on the Internet starts with a DNS lookup to resolve a domain of an IP address to reach the desired content. In this sense, academia has been devoted to measuring and analyzing the performance of DNS resolvers using different tools, either tailored for each work or generic. However, such tools might present different results due to their implementation and affect the measurements. Therefore, this paper reviews the literature on DNS performance research to gather the tools and DNS resolvers most used and, based on this, provides an analysis and comparison of the different DNS lookup tools employed in the literature and discusses the impact of tool selection on measurement results. Research showed that tool selection has an impact on results but not on the lookup success rate.**

***Index Terms*—DNS, Services and Protocols, Measurement**

I. Introduction

Established in 1983, the Domain Name System (DNS) emerged as a critical component of the Internet [1]. Its primary function is to translate user-friendly hostnames (*e.g.*, *wikipedia.org*) into their corresponding Internet Protocol (IP) addresses, effectively serving as the "phone book" of the Internet [2]. Almost all Internet communication starts with a DNS lookup, and complex websites which require content from multiple third parties might perform hundreds of DNS requests before loading a single page [3]. Throughout the history of DNS design and implementation, efforts have been made to minimize latency, such as providing recommendations on how DNS operators can optimize a DNS service to minimize latency for several clients [4]. Thus, DNS performance is of concern, as it directly impacts performance in most Internet-based communications [5].

Past work has extensively measured DNS performance under different conditions. For example, [6] thoroughly analyzed DNS performance with distributed measurements across more than 50 different Internet Service Providers (ISP), in more than 28 countries, comparing local and public DNS resolvers. [7] focused on comparing the performance between DNS and its encrypted versions, DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH), and their impact on web page loading times. [8] measured DoH performance overhead, as well as malicious domain protection. However, they all use different DNS lookup tools (*e.g.*, *dig*, *dnspython*, and *pydig*). When evaluating DNS performance, the lookup tool used to perform the measurements might introduce additional overhead and skew the results, underscoring the necessity for careful tool selection when designing experiments.

In this paper, we compare the performance of different Python DNS lookup tool libraries. For that, we review the literature to gather the most widely used tools and select three of them as the focus of our analysis. We collect a dataset of measurements by performing several DNS queries to different public resolvers using the selected tools using two different protocols, the classic DNS-over-Port 53 (Do53), the DoH, and DoT (both DoH and DoT extending previous work [9]). Our analysis focuses on Response Time (RT), which is the time elapsed between issuing the DNS request and receiving a response. Our objective is to determine the impact that tool selection could have on DNS performance measurements.

The remainder of this paper is structured as follows. Section II compares related work. Section III describes the selection of the tools for the experiments and details the methodology used in the measurements. Section IV presents the setup of the experiment and discusses the results. Lastly, Section V summarizes key findings and suggests future work.

II. Related Work

To select the tools (*cf.* Section III-A) to be analyzed in this work, we reviewed the literature on research approaches focused on analyzing the performance of DNS resolvers. Although all efforts focused on measuring DNS performance, they varied in objective and scope.

[8] compares the performance, employing the *pydig* tool, and security aspects of DNS resolvers provided by major Italian ISPs with public resolvers from Google and Cisco (*i.e.*, OpenDNS). Although local resolvers exhibit faster response times, the research finds that their security level matches the level of public resolvers, which indicates that users do not need to compromise their security for improved performance of public DNS resolvers.

In a similar study, [6] examines the impact of several DNS resolver responsiveness and cache content on applications such as Content Distribution Networks (CDN). With the use of comprehensive measurements across ISPs, relying on the *dig* Linux tool, the study reveals significant disparities in responses due to CDN location awareness and DNS resolver proximity, uncovering limitations in ISPs' DNS deployments and biases in third-party DNS replies.

Reference	Protocol(s)	Lookup Tool	List of Analyzed Resolvers
[8]	Do53, DoH	pydig	Google, OpenDNS
[6]	Do53	dig	Google, OpenDNS
[7]	Do53, DoH, DoT	dnspython	Google, Cloudflare, Quad9, CleanBrowsing, PowerDNS, BlahDNS, SecureDNS, Rubyfish, Commons Host
[10]	Do53, DoH	dns-measurement	*https://github.com/dnscrypt/dnscrypt-resolvers*
[11]	Do53, DoH, DoT	SamKnows	Anonymized Public Resolvers
[12]	Do53, DoT	RIPE Atlas	Google, Cloudflare, Quad9, CleanBrowsing, UncensoredDNS
[13]	Do53, DoH, DoT	dns-measurement	Google, Cloudflare, Quad9
[14]	Do53, DoH	Firefox	Google, Cloudflare, Quad9
[15]	Do53, DoH	BrightData	Google, Cloudflare, Quad9, NextDNS

[7] investigates two encrypted DNS protocols, DoH and DoT. Using the *dnspython* lookup tool, the authors evaluate the DoH landscape and compare it with the DoT landscape. Furthermore, the study quantifies the impact of DoH on Web page load times, indicating that the protocol provides enhanced security with minimal impact on loading time performance.

In [10], encrypted DNS resolvers that support DoH are evaluated to address privacy concerns. The research shows that while some non-mainstream resolvers have higher response times, there are exceptions, indicating the possibility for users to utilize a broader range of encrypted DNS resolvers than those currently available in popular browser configurations.

[11] investigates the performance of encrypted DNS protocols and conventional DNS in home networks from the United States of America (USA). The research, conducted using a proprietary tool called *SamKnows*, revealed that privacy-focused DNS protocols, such as DoT, could outperform conventional DNS in terms of response times for certain resolvers, even with increased latency. The study underscores the need for DNS clients (*e.g.*, browsers) to periodically evaluate latency and response times, suggesting that no single DNS protocol or resolver universally outperforms others for all clients.

Similarly, [12] analyzes DoT adoption and performance, leveraging 3200 Réseaux IP Européens Network Coordination Center (RIPE) Atlas probes in home networks. That research reveals a 23.1% increase in DoT support among open resolvers and a low adoption of local resolvers at 0.4%. Although DoT exhibits higher failure rates and response times, local resolvers achieve response times comparable to public ones despite higher failure rates. Thus, it highlights the complexities and regional disparities in DoT implementation.

[13] explores the impact of the Do53, DoT, and DoH DNS protocols on query response times and page load times from global perspectives using the same tool as [10]. Although DoH and DoT exhibit slightly higher response times than Do53, they can outperform Do53 in terms of page load times. However, in the conditions of reduced throughput and increased latency, Do53 becomes the fastest option for web page loading. Furthermore, Do53 and DoT show higher success rates in loading web pages compared to DoH. The research suggests strategies for enhancing DNS performance, including opportunistic partial responses and wire format caching, to address varying conditions and improve user experience.

Still in DoH, [14] discusses the policy implications of DoH. The authors systematically analyze DoH DNS resolvers, measure DoH's performance effects on Web page loading times using the Firefox Web browser, examine the competitive landscape of such an area, and explore the impact on stakeholders, such as ISPs and consumers. The work sheds light on the potential regulatory and policy implications of widespread DoH deployments.

Lastly, [15] investigates the performance, using the BrightData network, of DoH using a comprehensive dataset from 22 052 clients across 224 countries and territories. That research reveals mixed impacts on the performance of DoH-enabled DNS resolvers and highlights geographic disparities in DoH and Do53 resolution times, with clients from countries with low investment in Internet infrastructure being more prone to slowdowns when switching to DoH.

In summary, [8] and [6] focused on comparing local and public resolver performance. [7], [10]–[12], [15] investigated the performance impact of using encrypted DNS through HTTPS or TLS protocols, while [13] and [14] do so with additional attention to Web page loading times. Table I presents a detailed review of these efforts and the tools used.

III. Analyzing and Comparing DNS Lookup Tools

This section presents the reasoning behind the selection of the tools based on the literature research conducted, details the methodology employed to compare the tools, and describes the implementation of the approach to perform the queries.

A. DNS Lookup Tools Selection

Regarding DNS lookup tools, we could observe different approaches based on Table I, including the use of proprietary monitoring software such as SamKnows [16] and BrightData [17] but also a non-commercial distributed monitoring tool, called RIPE Atlas [18]. For this work, we selected open source and accessible tools, which are the Python libraries *pydig* and *dnspython*, as well as the native *dig* Linux command. These tools are described in the following paragraphs.

pydig [19] is a Python wrapper library for the *dig* command-line tool. Therefore, it relies on the native *dig* Linux tool, provided by the *bind* package [20], which allows users to gather detailed information about DNS records, server response times, and domain configurations. In contrast, *dnspython* [21] is a Python library that implements a full-fledged DNS toolkit from scratch. The toolkit can be used for several DNS-related actions, such as queries and nameserver testing. The toolkit implements its communication using sockets to perform queries to DNS resolvers and interact with DNS servers.

B. Methodology and Implementation

Our resolver data set consists of five widely used public resolvers that we also derived from literature research (*cf.* Table I), being Google, Cloudflare, Quad9, CleanBrowsing, and Adguard. For our domain dataset, we selected the most popular domain on the Tranco list [22], retrieved from September 13, 2023, which was *www.google.com*. For each resolver, the lookup of the selected measurements was performed in a loop 500 times so that a significant sample size and a confidence interval of 95% could be collected during the analysis.

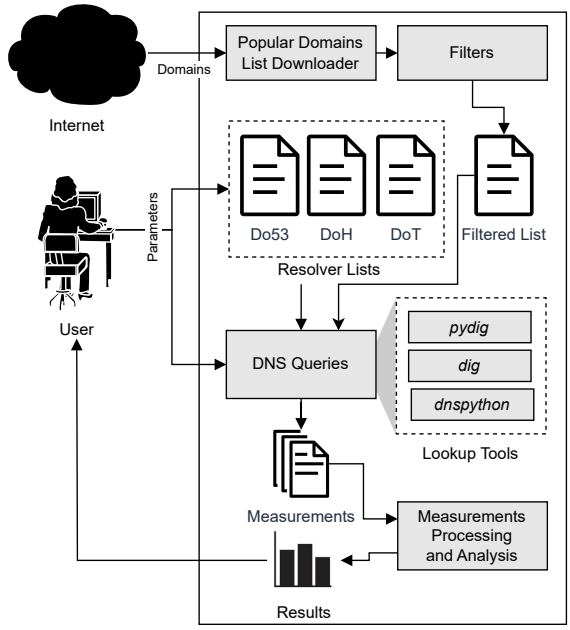


Fig. 1: Measurement Tool Design

To issue DNS queries from all the tools selected, we designed and implemented a Python measurement tool that executes and collects the results of each combination of tools (*e.g.*, *dig*, *pydig*, and *dnspython*), resolvers (*e.g.*, Google, Quad9, and Adguard) and domains (*e.g.*, *google.com* and *wikipedia.org*). Figure 1 depicts the design of proposed tool including its main components and actors. The tool automatically retrieves the latest list of popular domains curated by the Tranco list project [22], after that, filters (*e.g.*, selection of *n* top domains) are applied in the list to reduce or increase the number of domains to be analyzed. Further, the tool allows the user to input all the parameters, including the resolver lists and the number of DNS requests to be made for each resolver, domain and tool. The tool stores the output of the queries (*e.g.*, return codes, RT, and timestamp) as Comma-Separated Values (CSV) files for post-processing and analysis. This processing is performed in a dedicated component and the results are presented for the user. As the tool was designed to be extensible, other lookup tools can be added by creating a new wrapper to such a tool and including it in DNS queries component.

The performance metric relevant for this study is the RT of a lookup, which consists of the time elapsed between issuing the query and receiving a response from the resolver. To obtain accurate RTs, we measure them using Python's `time` module for each of the tools selected. Listing 1 illustrates the methodology for measuring the RT of a Do53 query using the `time` module. In Line 3, the query is constructed; in Line 4, the start time is recorded to fetch the NS; in Line 5, the query is sent; in Line 6, the end time of the query is captured, and finally, in Line 7, the temporal difference between the end and start times is computed.

```
1 mq = dns.message.make_query(domain, dns.rdatatype.
      NS)
2 start_time = time.time()
3 q = dns.query.udp(mq, resolver, timeout=3)
4 end_time = time.time()
5 res_time = end_time*1000 - start_time*1000
```

Listing 1: RT Measurement of a Request using *dnspython*

To serve as a baseline and to compare with the tool RTs, we also used *awk* [23], a program that implements the AWK domain-specific language, to extract *dig*'s reported query time as depicted in Figure 2. The figure illustrates the command used to retrieve the *dig* query time in the top of the figure, and, in the solid-line square, the result of a DNS query of a {domain} to a {resolver} using *dig*. The query time of that specific query is highlighted in the red-dashed square, which is extracted using the *awk* command depicted at the top.

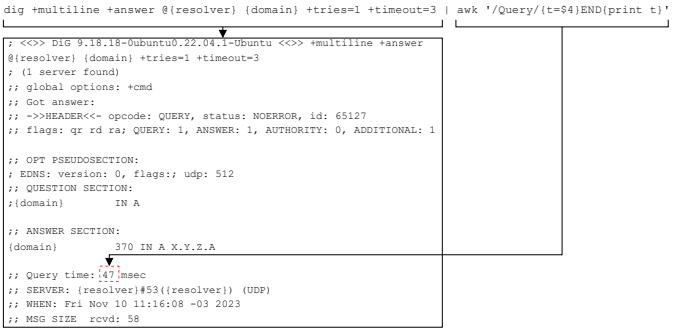


Fig. 2: Extraction of *dig*'s Reported Query Time using *awk*

All source code, domain, and resolver data sets, as well as the measurement results presented and discussed in the next section (*i.e.*, Section IV-A), are available at [24] to promote the reproducibility of the results of the experiments.

IV. Results and Discussion

This section describes the setup of the experiment (*i.e.*, hardware and vantage point) in Section IV-A, presents the results of the Do53 and DoH measurements in Section IV-B and the lookup success rate of both protocols in Section IV-C, and discusses the results and outlines the limitations of our work in in Section IV-D.

A. Experiment Setup

The setup of the experiment was an 2.3 GHz Intel® Core™ i5-6200U machine running Debian 11 Linux operating system, with 16 GB of RAM. To have a stable network connection, the experiments were performed using an 100 Mbps Ethernet cable connected to the Internet with the vantage point being an academic institution located in Porto Alegre (south of Brazil) which has a 10 Gbps dedicated link but no special treatment for DNS-related traffic. Local cache was disabled (*i.e.*, flag `cache` set to `no` in the `/etc/systemd/resolved.conf` file) of the machine used in the measurements to not affect the results.

B. Results

Three different experiments were conducted. The first experiment used the conventional method of sending DNS queries using UDP and port 53 (*i.e.*, Do53), the standard protocol for DNS. In this experiment, DNS queries were sent without encryption, which potentially leaves the transmitted data vulnerable to interception or tampering. In contrast, the second amd third experiments used the DoH and DoT protocol, two secure alternatives to Do53 which uses queries encapsulated within HTTPS or TLS connections, ensuring that the data exchanged between the client and the DNS server is encrypted and secure from interception or tampering by unauthorized parties. By comparing the results of these experiments, it is possible to evaluate the behavior of DNS lookup tools and DNS queries with different protocols, providing insights to analysis using such protocols.

The results of the Do53, DoH, and DoT experiments are shown in Figures 3, 4, and 5, respectively. For each resolver in the *x* axis, different bars are represented, and the *y* axis represents the response time, in milliseconds, measured by each tool. The *dig_awk* bar represents the measurements using the *dig* command to get the DNS query time, *dig_timelib* represents the measurements using the Python's `time` library and calling *dig* from the Python's `subprocess` module, the *dnspython* bar represents the measurements performed using the *dnspython* library and, lastly, the *pydig* bar represents the measurements with the *pydig* library.

1) DNS-over-Port 53 (Do53): Figure 3 shows that the measured RTs of the Do53 queries vary for both different tools and different resolvers. For the same resolver, we can observe RT differences as high as 153% between different tools, as can be seen in the case of *dnspython* (32 ms) and *pydig* (81 ms) mean RTs for the Google resolver. On average, we can see that the best performance comes from the RTs reported of native *dig*, closely followed by *dnspython*. The performance of *Pydig* was consistently worse, and similar to that of our implemented *dig* tool measurements using the Python time module.

2) DNS-over-HTTPS (DoH): When employing the DoH protocol to perform queries using the selected tools, a different behavior can be observed, as shown in Figure 4. For example, there was a difference of 1012% between the result of the *dig* reported query time (21 ms) and *dnspython* reported RT (216 ms) for the Google resolver. This behavior is different

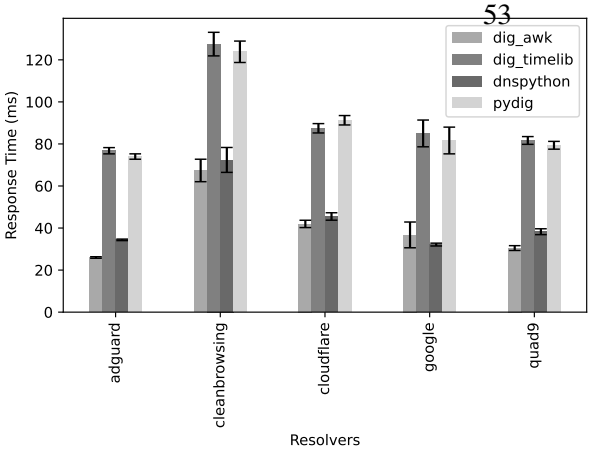


Fig. 3: Results of the Experiments using Do53

from the behavior observed using the Do53 protocol, where *dig* and *dnspython* presented similar results. Such a behavior can be explained by *dnspython*'s implementation of the DoH query, which includes the time required to create all the sessions and exchange the keys using HTTPS; whereas the *dig* reported RT only measures the query time.

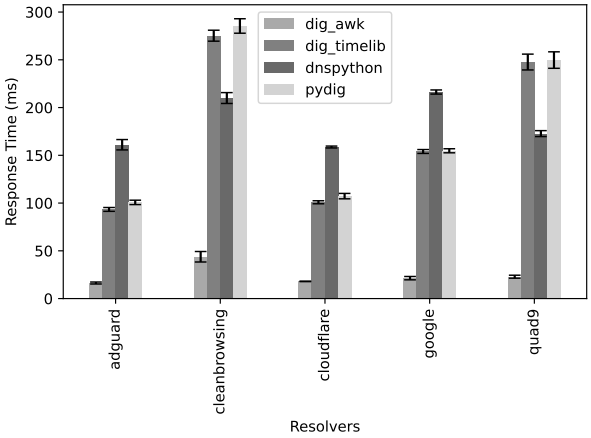


Fig. 4: Results of the Experiments using DoH

3) DNS-over-TLS (DoT): The DoT query results are illustrated in Figure 5. As depicted in the figure, the best performance remains the native *dig* tool. Moreover, the difference between the native *dig* and *dnspython* is still higher on average than the Do53 results (*cf.* Figure 3, which further corroborates the theory that the TLS handshake overhead is not measured by *dig*'s reported RT, only in *dnspython*. Lastly, the performances of *Pydig* and *dig* tool measurements using the Python time module are slower on average.

C. Lookup Success Rate

In addition to analyzing the RT of the queries, we analyzed the lookup success rate of all the queries issued for Do53, DoH and DoT. This analysis on the lookup success rate provides a in-depth understanding of the reliability and effectiveness of

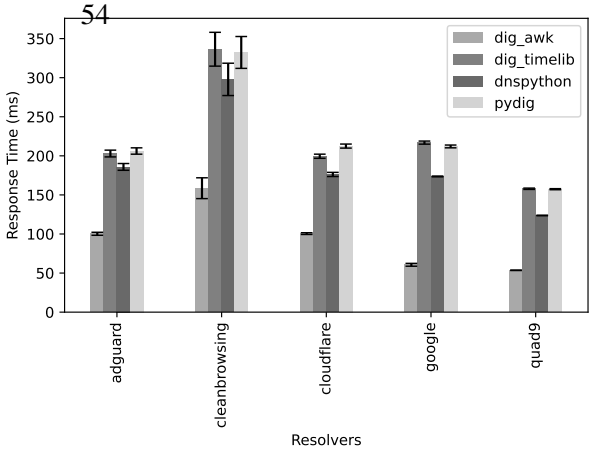


Fig. 5: Results of the Experiments using DoT

the selected tools and selected resolvers in successfully resolving DNS queries. Thus, providing another key contribution of our research and adding another aspect to be considered when selecting the DNS lookup tool.

Table II summarizes the success rate of the 500 DNS lookup tasks performed for the *www.google.com* domain using Do53 and each of the selected tools on different DNS resolvers. From the mean rate results, it can be seen that there is no significant difference or correlation between the lookup tool and the success rate. However, the success rate is closely related to the DNS resolver that performs the lookup; the rows highlighted in light gray represent the two best DNS resolvers (*i.e.*, the resolvers with the highest lookup success rate). Adguard using the *pydig* tool was the combination able to resolve all lookups without any error (*e.g.*, timeout); Cloudflare was also consistent, in terms of the lookup success rate, for each lookup tool with a 99% mean success rate.

TABLE II: Lookup Success Rate using Do53

		Lookup Tool			
Resolver	**IP Address**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	94.140.14.14	99.8%	99.8%	99.8%	100%
Cleanbrowsing	185.228.168.168	93.0%	93.0%	91.6%	91.6%
Cloudflare	1.1.1.1	99.0%	99.0%	98.8%	99.2%
Google	8.8.8.8	92.2%	92.2%	90.2%	92.2%
Quad9	9.9.9.9	92.4%	92.4%	92.0%	94.6%
	Mean Rate	95.3%	95.3%	94.5%	95.5%

The lookup success rate of the same queries to *www.google.com* using the same tools and resolvers was collected using the DoH protocol. Table III presents the results of the success rate, which was 100% in all cases, except for the Cleanbrowsing resolver (row highlighted in gray), which failed once during all queries, resulting in a success rate of 99. 8%. Compared to Do53, it can be stated that DoH queries are more stable and will return the result of the query successfully given their use of HTTPS using TCP instead of UDP, which has no delivery guarantee.

Finally, the same analysis is repeated for queries using the DoT protocol. Table IV presents the results of the success rate,

TABLE III: Lookup Success Rate using DoH

		Lookup Tool			
Resolver	**HTTP Endpoint**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	*https://dns.adguard.com/dns-query*	100%	100%	100%	100%
Cleanbrowsing	*https://doh.cleanbrowsing.org/doh/adult-filter*	100%	99.8%	99.8%	99.8%
Cloudflare	*https://cloudflare-dns.com/dns-query*	100%	100%	100%	100%
Google	*https://dns.google/dns-query*	100%	100%	100%	100%
Quad9	*https://dns.quad9.net/dns-query*	100%	100%	100%	94.6%
	Mean Rate	100%	99.9%	99.9%	99.9%

which was 100% in all cases, except for the Cleanbrowsing resolver (row highlighted in gray), which failed due to timeouts (12 to 21 times out of 500, depending on the tool). Similarly to DoH, we can observe higher success rates than when using the Do53 protocol. This behaviour is likely due to TCP feature of DoH and DoT that guarantee delivery, as opposed to UDP which does not offer such a guarantee.

TABLE IV: Lookup Success Rate using DoT

		Lookup Tool			
Resolver	**IP Address**	*dig_awk*	*dig_timelib*	*dnspython*	*pydig*
Adguard	*94.140.14.14*	100%	100%	100%	100%
Cleanbrowsing	*185.228.168.168*	100%	97.6%	95.8%	97.8%
Cloudflare	*1.1.1.1*	100%	100%	100%	100%
Google	*8.8.8.8*	100%	100%	100%	100%
Quad9	*9.9.9.9*	100%	100%	100%	100%
	Mean Rate	100%	99.5%	99.2%	99.6%

D. Discussion and Limitations

One reason that could explain the performance differences between *dnspython* and *pydig* is the fact that *pydig* acts as a wrapper of the *dig* command, using the `subprocess` module; thus, it requires system calls (*e.g.*, opening a process and reading process descriptors) from Python to the Operating System (OS). In contrast, *dnspython* performs the queries directly in native Python code by relying on native UDP and TCP sockets, which results in faster communication with the resolver compared to *pydig*. Moreover, the *dig_awk* command retrieves the RT of the query directly from the response, not adding the OS overhead in the RT. Thus, it shows that *dnspython* is the closest one to the native *dig* command (*i.e.*, *dig_awk* in Figure 3).

However, this behavior can only be assumed for DNS queries using the Do53 protocol. When performing queries using the DoH and DoT protocol, the *dnspython* tool and *dig* vary considerably, with *dig* presenting the lower RTs of all the investigated tools. Furthermore, *dig_timelib* and *pydig* showed similar results in both Do53 and DoH, which shows that there is no difference when using either. However, *dnspython* presented higher RTs in the majority of resolvers. Thus, the implementation of the DoH protocol in *dnspython* presents an overhead that must be considered for this case. Based on results found, it can be concluded that not all DNS resolvers provide a consistent DNS resolver service for users, which might affect user's browsing and overall Internet usage experience. In addition, such inconsistency was also found in the tools used in the literature to measure the performance of the DNS resolver, leading to skewed results.

A limitation of the work presented herein is the fact that the measurements were conducted from a single vantage point. In this sense, RTs can be affected by network conditions and routing decisions. However, the focus of this work is to analyze the difference in RT between tools and not between DNS resolvers. Therefore, the use of a single vantage point does not invalidate the results presented in this section and the considerations made in this work. In addition, such a limitation is planned to be addressed in future work.

V. Conclusion and Future Work

In this paper, we analyzed the performance impact of using different DNS lookup tools in DNS performance measurements of DNS-over-Port 53 (Do53), DNS-over-HTTPS (DoH), and DNS-over-TLS (DoT) queries. The literature on DNS performance measurement was researched to investigate efforts on such a topic and select which were the most used lookup tools and DNS resolvers in the approaches. On the basis of that, three tools (*i.e.*, *dig*, *pydig*, and *dnspython*) and five public DNS resolvers (*i.e.*, Adguard, Cleanbrowsing, Cloudflare, Google, and Quad9) were selected for our analysis.

From the results of the experiment, we found a significant variation in RT lookups across different tools and resolvers, with performance impacts as high as 153% of Do53 and 1012% for DoH queries. Additionally, there was no correlation between the lookup tool and the DNS lookup success rate, the rate being only related to the DNS resolver used. Further, DoH showed to be consistently more stable in the success rate of the queries in all of the selected tools and resolvers compared to the success rate of queries issued using Do53.

However, based on our findings, we conclude that tool selection directly impacts results when analyzing DNS performance. This difference in results can be explained due to the tool's implementation, which varies from using the Operating System (OS) to call an external DNS lookup tool (*e.g.*, *dig*) or using native Python sockets to create DNS requests (*e.g.*, *dnspython*). Thus, it is suggested that researchers carefully select the tool when designing future experiments and take into account that the results might be affected not because of network conditions or the implementation or deployment of the DNS server but because of the tool they are relying on. Further, this aspect also impacts on the data quality that is used as input for analysis and statistics.

Future work on the present research includes, but is not limited to, *(i)* increase the selection of tools being compared, *(ii)* include tools implemented in different languages to assess the impact of the language, *(iii)* analyze DNS network packets in-depth to gather more granular temporal information, and *(iv)* add diversity of vantage points and network conditions (*e.g.*, mobile networks) of the measurements.

References

[1] P. Mockapetris and K. J. Dunlap, "Development of the Domain Name System," in *Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM 1988)*, Stanford, California, USA, August 1988, p. 123–133.

[2] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 7th ed. Pearson, 2016.

[3] M. Butkiewicz, H. V. Madhyastha, and V. Sekar, "Understanding Website Complexity: Measurements, Metrics, and Implications," in *ACM Conference on Internet Measurement Conference (IMC 2011)*, Berlin, Germany, 2011, p. 313–328.

[4] M. Müller, G. C. M. Moura, R. de O. Schmidt, and J. Heidemann, "Recursives in the Wild: Engineering Authoritative DNS Servers," in *ACM Internet Measurement Conference (IMC 2017)*, London, United Kingdom, 2017, pp. 489—-495.

[5] I. Bozkurt, A. Aguirre, B. Chandrasekaran, P. Godfrey, G. Laughlin, B. Maggs, and A. Singla, "Why Is the Internet so Slow?!" in *International Conference on Passive and Active Network Measurement (PAM 2017)*, Sydney, Australia, 02 2017, pp. 173–187.

[6] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig, "Comparing DNS Resolvers in the Wild," in *ACM SIGCOMM Conference on Internet Measurement (IMC 2010)*, Melbourne, Australia, 2010, p. 15–21.

[7] T. Böttger, F. Cuadrado, G. Antichi, E. L. a. Fernandes, G. Tyson, I. Castro, and S. Uhlig, "An Empirical Study of the Cost of DNS-over-HTTPS," in *ACM Internet Measurement Conference (IMC 2019)*, Amsterdam, Netherlands, October 2019, pp. 15—21.

[8] A. Affinito, A. Botta, and G. Ventre, "Local and Public DNS Resolvers: Do You Trade Off Performance Against Security?" in *IFIP Networking Conference (IFIP Networking 2022)*, Catania, Italy, June 2022, pp. 1–9.

[9] J. C. C. Pinto, E. J. Scheid, M. F. Franco, and L. Z. Granville, "Analyzing and Comparing DNS Lookup Tools in Python," in *XX Escola Regional de Redes de Computadores (ERRC 2023)*, Porto Alegre, RS, Brazil, Ocotber 2023, pp. 49–54.

[10] R. Sharma, N. Feamster, and A. Hounsel, "Measuring the Availability and Response Times of Public Encrypted DNS Resolvers," 2022, arXiv 2208.04999, cs.CR.

[11] A. Hounsel, P. Schmitt, K. Borgolte, and N. Feamster, "Can Encrypted DNS Be Fast?" in *Passive and Active Measurement*. Cham: Springer International Publishing, 2021, pp. 444–459.

[12] T. V. Doan, I. Tsareva, and V. Bajpai, "Measuring DNS over TLS from the Edge: Adoption, Reliability, and Response Times," in *Passive and Active Measurement*, O. Hohlfeld, A. Lutu, and D. Levin, Eds. Cham: Springer International Publishing, 2021, pp. 192–209.

[13] A. Hounsel, K. Borgolte, P. Schmitt, J. Holland, and N. Feamster, "Comparing the Effects of DNS, DoT, and DoH on Web Performance," in *Proceedings of The Web Conference 2020*, ser. WWW '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 562–572. [Online]. Available: https://doi.org/10.1145/3366423.3380139

[14] K. Borgolte, T. Chattopadhyay, N. Feamster, M. Kshirsagar, J. Holland, A. Hounsel, and P. Schmitt, "How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem," *SSRN Electronic Journal*, 01 2019.

[15] R. Chhabra, P. Murley, D. Kumar, M. Bailey, and G. Wang, "Measuring DNS-over-HTTPS Performance around the World," in *ACM Internet Measurement Conference (IMC 2021)*, Virtual Event, 2021, p. 351–365. [Online]. Available: https://doi.org/10.1145/3487552.3487849

[16] Cisco, "SamKnows - Internet Performance Monitoring," 2023, https://www.samknows.com/.

[17] Bright Data Ltd., "Bright Data - The World's #1 Web Data Platform," 2023, https://brightdata.com/.

[18] Réseaux IP Européens Network Coordination Centre RIPE NCC, "RIPE Atlas," 2023, https://atlas.ripe.net/.

[19] L. Smith, "pydig - Github Repository," 2021, https://github.com/leonsmith/pydig.

[20] Internet Systems Consortium, Inc., "Bind 9 - Versatile, Classic, Complete Name Server Software," 2023, https://www.isc.org/bind/.

[21] Dnspython Contributors, "dnspython Python Library," 2020, https://www.dnspython.org/.

[22] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation," in *Network and Distributed System Security Symposium (NDSS 2019)*, San Diego, USA, Feb. 2019.

[23] Free Software Foundation, Inc, "gawk - Pattern Scanning and Processing Language ," 2009, https://linux.die.net/man/1/awk.

[24] J. C. C. Pinto, "Encrypted DNS Benchmark," October 2023, https://github.com/jchagastelles/encrypted-dns-benchmark.

All links were visited in February 2024.