Tunneling through DNS over TLS providers

Lukáš Melcher
CTU in Prague
Prague, Czech Republic
melchluk@fit.cvut.cz

Karel Hynek
CTU in Prague & CESNET a.l.e.
Prague, Czech Republic
hynekkar@fit.cvut.cz

Tomáš Čejka
CESNET a.l.e.
Prague, Czech Republic
cejkat@cesnet.cz

***Abstract*—DNS over TLS (DoT) is one of the approaches for private DNS resolution, which has already gained support by open resolvers. Moreover, DoT is used by default in Android operating systems. This study investigates the possibility of creating DNS covert channels using DoT, which is a security threat that benefits from the increased privacy of encrypted communication. We evaluated the performance and usability of DoT tunnels created via commonly used resolvers. Our results show that the performance characteristics of DoT tunnels differ vastly depending on the used DoT resolver; however, the creation of a DoT tunnel is possible, reaching speeds up to 232 Kbps. Moreover, we successfully transferred data via DoT servers claiming Anti-Virus protection and family-friendly content.**

***Index Terms*—DNS over TLS, exfiltration, tunnel, covert channel, measurement**

I. Introduction

Domain Name Resolution (DNS) protocol is used to translate human-friendly domain names and machine-usable IP addresses and vice versa. Since DNS is essential for comfortable internet usage, DNS requests carrying unencrypted domain names occur before almost every connection. These facts have already been exploited in large-scale DNS surveillance programs such as QUANTUMDNS and MORECOWBELL operated by governmental agencies [1], which triggered concerns over user privacy in the broad public.

The engineering community targeted the privacy concerns by proposals of encrypted DNS protocols such as DNS over TLS (DoT) and DNS over HTTPS (DoH). Both approaches are gaining popularity across service providers [2], [3] and consumer software. DoH is already supported by all major web browsers [4], and DoT is enabled in Android OS (from version 9.0 Pie) [5] by default. Moreover, DoT has the potential for securing the communication between recursive and authoritative resolvers since it is already deployed between Cloudflare DNS and Facebook [6].

However, with increased privacy comes the security concerns of its abuse and intentional hiding from network detectors. According to the summary published by Hynek et al. [4], there are already multiple malware samples leveraging DNS encryption to establish a private channel for exfiltration and command and control (C2) via DNS tunneling.

DNS tunneling exploits DNS protocol, where attackers can encode data into queried domain names and their responses, effectively creating a covert channel. DNS tunnels can be used for exfiltration, C2 communication, and other malicious purposes [7]. DNS tunneling can be directly used with encrypted DNS; however, it is much more challenging to detect it reliably. Even though there are already approaches for DoH tunnel detection [4], DoT security research is still nascent despite the fact it poses similar risks.

With the encrypted DNS approaches, the network traffic visibility has been transferred from local internet service providers or organizational firewalls onto global DNS providers. There already are DNS servers claiming anti-malware or family-friendly protection[1]. However, we are unaware of any research that tests their protection properties. Moreover, it could be expected that the service providers are already performing DNS tunnel detection to prevent DNS server overloading.

In this research, we have selected multiple well-known DoT providers and tested the possibility of DNS tunnel creation. We have created a testing setup and measured the tunnel performance properties (such as speed, stability, and latency) to evaluate their usability for malicious purposes. Through this testing, we are targeting the questions about the security impacts of DoT deployment. Is it possible to establish a communication tunnel using DoT? Do service providers deploy any tunnel protection?

II. Related Work

DoT has been primarily studied in terms of its adoption. Deccio et al. [8] performed a DoT capability scan across open DNS resolvers in 2019. According to their results, the DoT adoption was very poor. From around 1.2 million open resolvers, only 1,747 (0.15%) supported DoT. Following study performed by Doan et al. [3] in 2020 found 2,151 showing an increasing trend in DoT support. The DoT adoption across individual users was also studied by Garcia et al. [2], who concluded that DoT is still minor while it is increasing in the observed traffic.

Even though DoT was proposed in 2016 [9], encrypted DNS research mainly targets DoH [10], which was standardized two years later in 2018. However, the security concerns raised by Borgolte et al. [11] apply to all encrypted DNS approaches, including DoT. The concerns stemmed mainly

[1]https://kb.adguard.com/en/general/dns-providers

from the lack of visibility and inheritance of security issues from the traditional DNS. The primary scope of this study is the DNS exfiltration and tunneling, which has already been studied in unencrypted DNS.

Research works like Dietrich et al. [12] describe the malware C2 communication via a stealthy DNS channel. Merlo et al. [13] focused on DNS-based tunnel communication and its performance. According to his measurement, DNS tunnels can achieve 500 Kbps of stable throughput, which is sufficient for large data transfer. Nevertheless, there are also effective approaches for DNS tunnel detection with accuracy reaching 100% [14], when the tunnel is not created stealthily. However, when using encrypted DNS, even a blatant abuse is challenging to detect [4], [14].

Security research has so far focused on encrypted DNS abuse only via DoH. MontazoriShatori [15] created a DoH tunnel detector using a Machine Learning algorithm, achieving an F1 score of 0.999. Another outcome of their research is the DoHBrw dataset, which was then used in other following studies listed in [4] with similar accuracy. Even though these studies prove the feasibility of encrypted DNS tunnel detection, they still suffer from limitations, mainly leading from the methodology of creation DoHBrw dataset. It contains only lab-created traffic, and the encrypted DNS tunnels could be distinguished primarily by volumetric statistics of transferred bytes and packets [4].

We are not aware of any study considering DNS tunneling via DoT. As the first step in our research of DoT tunnels, we explore the feasibility of their creation via the most common DNS providers. The motivation of our research rises from the advance in traditional DNS tunnel detection, which has become very accurate in the last years [14] and also from the security concerns about the mass deployment of encrypted DNS raised by the community. The DNS service providers have the opportunity to mitigate these threats and perform payload-based and signature-based detection of DNS abuse, which are, according to Wang et al. [14], the most accurate approaches.

III. Measuring Methodology

The proper methodology used for the experiments is crucial in obtaining relevant results. Therefore, this section provides a detailed description of our experiments and the setup.

A. Selection of DoT resolvers

The first step was to identify relevant DoT-capable resolvers. We decided to focus our experiments on well-known and established DNS resolvers since they are more important from a security perspective. We argue that an attacker can deploy its own unprotected resolver; therefore, the DoT connection to small and unknown resolvers can always be considered suspicious and filtered by the firewall while not disrupting DNS service for other users. The main threat arises from abusing well-known services such as Google DNS, which is used by millions of benign users. Moreover, we wanted to select resolvers with global reach, making the results applicable worldwide and representing global security risks related to encrypted DNS.

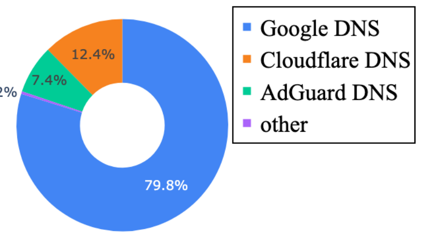

Fig. 1. Share of DoT resolvers in CESNET network. The share was calculated from the number of DoT connections.

To select the most popular DoT resolvers, we worked with a large Czech National Research and Education Network Operator, CESNET, which provides internet service to more than half a million users.

The anonymized one-month traffic from CESNET captured in January 2022 was analyzed to obtain the most popular DoT resolvers. DoT traffic was selected via port filtering since it uses port 853/TCP [9], resulting in 10 million DoT connections. Consequently, domain names of used resolvers were then extracted from Server Name Indication extension, which is transmitted during the TLS handshake. The share of individual resolvers is depicted in Fig. 1. Resolvers marked as "other" were mainly local, operated either by CESNET itself or by universities. Since these resolvers are not used globally and are not dominant even in CESNET (only 0.4% of all DoT connections), we decided not to include them in our analysis.

The real-world analysis of DoT traffic showed only three well-established providers that represent 99.6% of all DoT traffic on CESNET. Since the analysis was only limited to traffic from Czechia, we also used a list of well-known DoT

TABLE I
Evaluated public DoT resolvers, marked ones allowed creation of DoT tunnel

Name	Domain Name	IP address
Google DNS	dns.google.com	8.8.8.8
CleanBrowsing	family-filter-dns.cleanbrowsing.org	185.228.168.168
AliDNS	dns.alidns.com	223.5.5.5
BlahDNS	dot-de.blahdns.com	78.46.244.143
Bitdefender	ore-dns.bitdefender.net	35.247.80.47
Cloudflare	one.one.one.one	1.1.1.1
Dismail	fdns2.dismail.de	159.69.114.157
Quad9	dns9.quad9.net	9.9.9.9
AppliedPrivacy	dot1.applied-privacy.net	146.255.56.98
NextDNS	dns.nextdns.io	178.255.154.59
Adguard	dns.adguard.com	94.140.14.14
Adguard-F	dns-family.adguard.com	94.140.15.16
Bitdefender	fra-dns.bitdefender.net	35.242.226.78
Digitalcourage	dns3.digitalcourage.de	5.9.164.112
Bitdefender	ore-dns.bitdefender.net	35.247.80.47
dns.sb	dns.sb	185.222.222.222

resolvers maintained by DNS privacy project[2]. Together, we evaluated 16 DoT resolvers listed in Tab. I. All of them are operated by large global organizations, and thus we assumed they have an extensive user base. Therefore, the observed DoT communication with them usually does not raise suspicion. We also purposely selected family-filtered versions of DNS resolvers (when available), which we considered more protective. Moreover, the CleanBrowsing family filter explicitly claims security protection [16].

B. Experimental environment

Contrary to DoH, we are not aware of any malware or exfiltration tool, that would natively support DoT. However, it does not mean, that DoT cannot be misused. There are DNS to DoT translation proxies, which are transparent for connected devices, leaving them unaware of encryption. Any software (including malware) which does not support DoT natively can then take advantage of encryption when the proxy is deployed (e.g., on a router).

Our testing setup is depicted in Fig. 2 and follows the scenario with DoT proxy deployed on the edge router of small LAN. There are three main entities: 1) Router with DoT Proxy, 2) DNS Tunnel Target, 3) Rogue User performing DNS tunneling, and 4) Benign Users.

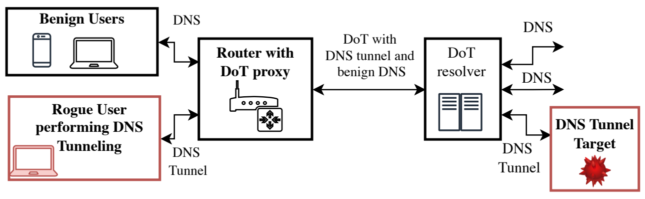

Fig. 2. DoT tunnel measurement setup

1) Router with DoT proxy: We used router with OpenWrt operating system[3]. Since OpenWrt is Linux-based, we could install a third-party DoT proxy. Moreover, we set the router as a primary DNS resolver for connected clients.

We have installed Stubby[4] DoT proxy into the router and used it throughout all our experiments. It was operated with default configuration settings. We provided only the domain name of the used resolver as the configuration entry.

2) DNS Tunnel Target: Represents the server-side of DNS tunnel. We registered the domain name using freenom.com free domain name provider. We rented Virtual Private Server (VPS) with a 1 Gbps connection and set it as an authoritative DNS server for the registered domain. On the VPS, we executed the server side of the DNS tunnel. During our experiments, we used Iodine[5] and DNS2TCP[6] for tunneling.

[2]https://dnsprivacy.org/public_resolvers/
[3]We used TP-Link Archer AC1750 with https://openwrt.org
[4]https://dnsprivacy.org/dns_privacy_daemon_-_stubby/
[5]https://github.com/yarrick/iodine
[6]https://github.com/alex-sector/dns2tcp

a) Iodine: It is a well-known DNS tunneling tool. The tunnel is created on the data-link layer. Thus IP headers are also transmitted. During its start-up, Iodine creates a specialized network interface that any application can use.

b) DNS2TCP: It performs tunneling on the TCP layer. Thus IP headers are not transmitted. The developers claim that contrary to IP-over-DNS approaches (such as Iodine), the lack of IP headers increases throughput. However, it cannot tunnel arbitrary traffic, and the "resource" (application listening on the tunnels server-side) needs to be specified. We have used SSH as a resource for all of our experiments. Unlike other supported resources such as SMTP or POP3, SSH allowed us to tunnel traffic easily with various speeds and characteristics.

3) Rogue User performing DNS tunneling: This entity represents the client-side of the DNS tunnel. The DNS tunneling tools were executed on a Linux-based machine connected via WiFi to the *Router with DoT proxy*. Moreover, tunneling tool was set to use a Stubby DNS proxy running on the router.

Even though Iodine is highly configurable, it also supports autodetection and selects the most suitable configuration based on the used resolver. During our measurement, we deployed Iodine in a default configuration; thus, the optimal parameters were selected automatically.

DNS2TCP does not support autodetection, nor is it as configurable as Iodine. It allows configuration of only resolver (we used *Router with DoT proxy*), "resource" application (we used SSH), and timeout interval. The timeout interval (a maximum server's answer delay) was set to 3 seconds since it is the default value; thus, we assumed it is commonly used.

4) Benign Users: This entity represents actual devices using the internet and creating DNS requests to make background noise. These devices were two laptops, desktops, and four Smart Phones actively used by the users. Moreover, background traffic made the experiments more realistic, considering our setup with the DoT proxy deployed on the router.

C. Definition of measured performance characteristics

We selected four performance characteristics: 1) Tunnel stability, 2) Packet Loss, 3) Packet Delay, and 4) Throughput.

1) Tunnel stability: It represents the time interval for which the tunnel stayed connected and ready for use. During the measurement, we did not transfer any large volumes of data in the tunnel; instead, we only used the tunnel for C2-like communication sending short packets with only 4 B of data. The client sent a message every ten seconds, which was then immediately followed by the server's response. This measurement verified the feasibility of C2 communication. When the connection remained active for more than 360 minutes, we stopped the experiment.

2) Packet Loss: It represents the number of packets lost in the tunnel. Since Iodine creates a tunnel on the IP layer, we could use the Ping program to measure packet loss. The packet loss was not measured for DNS2TCP since it creates a reliable transport layer.

3) Packet Delay: It represents the Round Trip Time (RTT) of a packet transmitted via the tunnel. Similarly, as in Packet Loss, we used the Ping program to obtain these characteristics for the Iodine-based tunnel. For DNS2TCP we used TCP version of Ping. Since DNS2TCP was using SSH as a resource, we used port tunneling inside SSH for this measurement.

4) Throughput: It represents the achieved throughput of the tunnel. We measured it by sending a large file ($\sim$1 GB) via SSH using the `scp` program. Since some tunnels did not perform well and we could not transmit the whole file in a reasonable time, we always left the transmission active for at least 30 minutes. The resulting throughput was then calculated from the successfully transmitted amount of data. This measurement verified the feasibility of data exfiltration.

D. Experiments execution

Altogether, the experiments took place between January 2022 and March 2022. Each resolver was tested at least three times during a day. All experiments were conducted from single location in Prague, Czech Republic. The router was connected to the network via 30 Mbps connection, which is according to report [17] an average connection speed in the country. Nevertheless, we assume, that location and connection speed has negligible impact on the results, since the tested providers have multiple servers located around the world to improve their performance.

The experiment procedure consisted from following steps: 1) Startup of DNS tunnel server, 2) Configuration of Stubby to use evaluated resolver, 3) Startup of DNS tunnel client on separate Machine. 4) Execution of performance measurement described in Sec. III-C, and finally 4) gathering the results and their interpretation.

The workflow remained unchanged for all selected DoT resolvers to maintain comparability of results.

IV. Results

We could not create a DoT tunnel through most of the selected well-known resolvers listed in Tab. I. The tunnel was successfully created only via seven out of 16 evaluated DoT resolvers, which are written with the measured performance characteristics in Tab. II.

Generally, Iodine was more successful in connections and outperformed DNS2TCP in all measured characteristics. Dismail offered the best throughput from all measured resolvers while being very stable. The C2-like communication was uninterrupted for the whole 360 minutes till we ended the stability experiment. Even though we achieved the highest throughput with Iodine, the DNS2TCP tunnel was not established.

Google DNS also performed very well, with high throughput and high stability. The average RTT was more than 30 ms smaller than Dismal. Moreover, Google DNS is the only resolver where we successfully created a DNS2TCP tunnel.

The family-friendly and malware protection CleanBrowsing was also very stable when used with C2-like communication. However, the achieved throughput of only 7.2 Kbps limits the possibility for exfiltration. Similar performance was also measured via AliDNS or BlahDNS.

The tunnel created via AppliedPrivacy was very unstable. We could barely measure the RTT characteristics and packet loss with the Iodine. The throughput could not be measured at all because the tunnel collapsed when we attempted to send a large file. With Quad9, we could only measure the tunnel stability with C2-like communication. The tunnel immediately collapsed when we tried to perform RTT measurements with the ping program.

We could not connect to most of the evaluated resolvers; thus, we can assume they have deployed protection against DNS tunneling. However, seven out of 16 tested resolvers can be misused for DNS tunneling, even when performed without stealthiness using Iodine. The DNS2TCP success in tunnel creation was much lower. We are not sure why the Iodine outperformed DNS2TCP in all measured characteristics. We assume that Iodine's success is caused by its autodetection feature, which tailors the configuration settings (such as maximal request size and type of request) for each resolver. However, this hypothesis needs to be further investigated, which is out of the scope of this work.

Compared to other works measuring tunnel performance over traditional DNS, DoT tunnels perform much worse. The highest observed value (232 Kbps via BlahDNS) achieved around 50% of throughput measured by Merlo et al. [13]. We assume that the performance drop is caused by the overhead created by the TCP connection since the DNS exfiltration tools are not designed for DNS via the reliable channel.

Apart from Dismail and Google DNS, most of the other resolvers performed DNS traffic throughput throttling making the tunnel slower (around 8 Kbps); thus less usable for sending large volumes of data. However, for five resolvers, the tunnel was very stable, and it could be used for low-throughput traffic such as C2 communication, including the CleanBrowsing, which claims anti-malware protection.

Unfortunately, the tunnel created via Google DNS, the most used resolver on the CESNET network (see Sec. III-A), showed very good performance and could be misused for malicious purposes such as exfiltration or long-lasting C2 channels. Even though other popular DoT resolvers (such as AdGuard or Cloudflare) perform DNS tunnel protection, just from the market share of Google DNS ($\sim$80% on CESNET network), we can conclude that DoT tunneling must be considered a serious problem. Our results show that even well-known and established providers allow threat actors to exploit nascent research in private-resolution technologies and bypass DNS tunneling network detection systems.

V. Response from DoT providers

To validate our results, we have contacted the tested providers via email, informing them about the results and asking them if they perform any DNS tunnel prevention. The emails were sent more than one month before the submission,

TABLE II
DoT tunnels performance results for resolvers, for which the tunnel was successfully established. The abbrevation in the column titles stands for: Estab. — Tunnel was successfully established, RTT-min — minimal Round Trip Time, RTT-avg — average Round Trip Time, RTT-max — maximal Round Trip Time, RTT-std — Standard Deviation of Round Trip Times

	Tool	**Estab.**	**Stability**	**RTT-min**	**RTT-avg**	**RTT-max**	**RTT-std**	**Loss**	**Througput**
Google DNS	iodine	**Yes**	>360 min	27.9 ms	51.4 ms	3593.1 ms	147.4 ms	2.50%	176 Kbps
	dns2tcp	**Yes**	120 min	28.7 ms	213.4 ms	17 365.2 ms	986.4 ms	—	148 Kbps
CleanBrowsing	iodine	**Yes**	>360 min	50.6 ms	1103.8 ms	10 821.6 ms	1489.8 ms	16.80%	7.2 Kbps
	dns2tcp	**No**	—	—	—	—	—	—	—
AliDNS	iodine	**Yes**	>360 min	161 ms	211 ms	8637.6 ms	1192 ms	57.60%	0.8 Kbps
	dns2tcp	**No**	—	—	—	—	—	—	—
BlahDNS	iodine	**Yes**	50 min	70.7 ms	618 ms	5126.6 ms	1006.5 ms	23.60%	8 Kbps
	dns2tcp	**No**	—	—	—	—	—	—	—
Dismail	iodine	**Yes**	>360 min	37.4 ms	82 ms	3345 ms	258.6 ms	2.60%	232 Kbps
	dns2tcp	**No**	—	—	—	—	—	—	—
AppliedPrivacy	iodine	**Yes**	4 min	531 ms	1241.6 ms	7161.4 ms	1198.4 ms	9.20%	—
	dns2tcp	**No**	—	—	—	—	—	—	—
Quad9	iodine	**Yes**	2 min	—	—	—	—	—	—
	dns2tcp	**No**	—	—	—	—	—	—	—

and we got a reply only from Applied Privacy, CleanBrowsing, Quad9, and Google — through all of them, we were able to create a DoT tunnel. Unfortunately, till the submission, we did not receive any reply from other service providers; thus, we could not check our results completely.

Google asked us for patience until they obtain the technical department's answer, but they did not send it even after one month of waiting. The reaction from other providers was always almost the same that they do not perform any DNS tunnel protection. Instead, they confirmed that DNS throttling is deployed to prevent overloading of their services, resulting in reduced performance. However, our results show that even the throttled DNS tunnel could be leveraged for C2 communication.

VI. Conclusion

The increased adoption of encrypted DNS also brings security concerns about the lost visibility into the traffic by network security and protection tools. The DNS service resolvers still have access to unencrypted DNS requests, which raises the question of whether they perform any protection against DNS abuse in the encrypted channel. In this study, we investigated the possibility of DNS tunnel creation via encrypted DoT channels over well-known and established DNS providers. According to our findings, most of the evaluated resolvers perform filtering, and we could not establish the tunnel. However, the tunnel created via five resolvers was very stable and usable for reliable C2 communication. Moreover, the tunnel created via Google DNS, which is by far the most popular resolver, was functional even for large data transfers achieving average throughput of 176 Kbps. Our experiments showed that DoT tunneling could be performed even over mass-used resolvers and not only by anonymous proxies, which could be filtered in the firewalls. DNS tunnel via DoT poses a tempting approach for threat actors; thus, we call for further research in DoT tunnel detection.

Acknowledgement

This research was funded by the Ministry of theInterior of the Czech Republic, grant No. VJ02010024: Flow-Based Encrypted Traffic Analysis, and by the Grant Agency of the CTU in Prague, grant No. SGS20/210/OHK3/3T/18 funded by the MEYS of the Czech Republic

References

[1] C. Grothoff, M. Wachs, M. Ermert, and J. Inria, "Towards secure name resolution on the internet," 02 2017.

[2] S. García et al., "Large scale measurement on the adoption of encrypted DNS," 2021. [Online]. Available: https://arxiv.org/abs/2107.04436

[3] Doan et al., "Measuring dns over tls from the edge: Adoption, reliability, and response times," in *PAM*. Springer, 2021.

[4] Hynek et al., "Summary of dns over https abuse," *IEEE Access*, vol. 10, pp. 54 668–54 680, 2022.

[5] E. Kline and B. Schwartz, "Dns over tls support in android p developer preview," Apr 2018. [Online]. Available: https://android-developers.googleblog.com/2018/04/dns-over-tls-support-in-android-p.html

[6] M. Bretelle, "Dns over tls: Encrypting dns end-to-end," 2020. [Online]. Available: https://engineering.fb.com/2018/12/21/security/dns-over-tls/

[7] T. A. Peña, "A deep learning approach to detecting covert channels in the domain name system," Ph.D. dissertation, 2020.

[8] C. Deccio and J. Davis, "DNS Privacy in Practice and Preparation," ser. CoNEXT '19. New York, NY, USA: ACM, 2019.

[9] Z. Hu et al., "Specification for DNS over TLS," RFC 7858, May 2016.

[10] Hoffman et al., "DNS Queries over HTTPS (DoH)," RFC 8484, 2018.

[11] Borgolte et al., "How DNS over HTTPS is Reshaping Privacy, Performance, and Policy in the Internet Ecosystem," *PPIE*, 2019.

[12] C. J. Dietrich, C. Rossow, F. C. Freiling, H. Bos, M. v. Steen, and N. Pohlmann, "On botnets that use dns for command and control," in *2011 Seventh EC2ND*, 2011.

[13] Merlo et al., "A comparative performance evaluation of DNS tunneling tools," in *CISIS*, 2011. [Online]. Available: https://doi.org/10.1007/978-3-642-21323-6_11

[14] Y. Wang et al., "A comprehensive survey on dns tunnel detection," *ComNet*, vol. 197, 2021.

[15] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. Habibi Lashkari, "Detection of doh tunnels using time-series classification of encrypted traffic," in *2020 IEEE DASC/PiCom/CBDCom/CyberSciTech*, 2020.

[16] CleanBrowsing, "Free dns content filtering," 2022. [Online]. Available: https://cleanbrowsing.org/filters/

[17] Cable.co, "Worldwide broadband speed league 2021." [Online]. Available: https://www.cable.co.uk/broadband/speed/worldwide-speed-league/#regions