Towards Automated DNS Censorship Circumvention

Felix Lange
Paderborn University

Niklas Niere
Paderborn University

Juraj Somorovsky
Paderborn University

Abstract

Censorship is employed by many governments and ISPs worldwide, with an increasing trend in recent years. One of the most censored protocols is DNS: censors target unencrypted and encrypted DNS to prevent clients from resolving the domain name of unwanted websites. Despite much research on DNS censorship, only a few tools can circumvent it.To support users affected by DNS censorship, we present DPYProxy-DNS, a DNS resolver that automatically detects and employs a working DNS censorship circumvention. We demonstrate the effectiveness of DPYProxy-DNS by automatically circumventing DNS censorship in China and Iran and analyzing DNS censorship mechanisms in these countries. Our analyses reveal that DNS censorship in Iran is ineffective against encrypted DNS. In China, DPYProxy-DNS revealed two consistently working circumvention techniques for unencrypted DNS: TCP segmentation for DNS over TCP and ignoring DNS responses injected by the Great Firewall of China (GFW). Our findings reveal varying levels of DNS censorship across different countries, underscoring the importance of the automated circumvention approach we provide with DPYProxy-DNS.

Keywords

censorship, circumvention, DNS, automated circumvention tool

1 Introduction

Many countries restrict the Internet access of their residents [24, 26]. Russia [35, 46, 47], Iran [1, 4, 11, 23, 28], and China [3, 6, 13, 43–45] are arguably the countries that have been analyzed the most. Using so-called deep packet inspection (DPI), countries' censors inspect the packets of different protocols to prevent access to certain websites. For instance, censors inspect packets of the Transport Layer Security (TLS), HTTP, and DNS protocols. Accessibility of the DNS protocol is a bottleneck for other protocols[16, 29]: clients must resolve every domain to an IP address using the DNS protocol before accessing the domain with another protocol. Therefore, censors often target DNS, leading to extensive DNS censorship analyses in the past [2, 3, 5, 15, 24, 32, 33, 39]. This is especially easy for unencrypted DNS as censors can directly extract the queried domain and potentially censor it.

Encrypted DNS. To enhance privacy and to prevent censors from analyzing unencrypted DNS queries, five encrypted DNS protocols have been specified: DNS over HTTPS (DoH) [17], DNS over TLS (DoT) [18], DNS over QUIC (DoQ) [19], DNS over HTTPS/3 (DoH3) [17, 19], and DNS over DTLS (DoD) [37]. These protocols encrypt the DNS messages between the client and the resolver, preventing censors from extracting the queried hostname. Facing encrypted DNS protocols, censors must resort to IP blocking of DNS resolvers or to fingerprinting encrypted DNS traffic [38]. Previous research showed that many countries attempt to restrict the use of encrypted DNS [24] through these means.

Censorship Circumvention Tools. To circumvent countries' censorship efforts, various tools have been developed. Some of these tools tunnel users' traffic to a server behind the firewall, evading censorship through custom protocols such as VPNs [40, 41]. Because these tools typically tunnel all traffic, they also prevent censors from intercepting DNS requests. Another type of tool does not require an external server and operates solely on the client's machine, thereby evading censorship by manipulating client-originating traffic [14, 21, 22, 31, 42]. While such tools exist for protocols such as HTTP and TLS, clients have to find alternative ways to circumvent DNS censorship. Intra [12], an Android-based circumvention tool, offers DNS censorship circumvention through encrypted DNS. However, Intra currently only supports DoH and does not provide an automated way to find a successful circumvention technique. Other tools may rely on server-side support of a custom protocol [34] or a dedicated external server [9]; we focus on standard-compliant techniques deployable on the client side.

Research Gap. Despite the existence of numerous censorship circumvention tools that operate without an external server, none of these tools automatically and reliably circumvent DNS censorship. Browsers such as Chrome and Firefox and operating systems such as Linux and Windows allow users to configure an encrypted DNS server; however, they often only allow one type of encrypted DNS (e.g., DoH in Firefox), and encrypted DNS itself is being targeted by censors. We also hypothesize that DNS censorship is applied inconsistently across resolvers; therefore, we present an overview of censorship across different resolvers. Overall, we identify the need for a censorship circumvention tool that automatically determines a working DNS censorship circumvention method, and—complementing other locally operating circumvention tools—implements its circumvention in a local DNS resolver accessible to other tools and programs on the same machine.

Contributions. In our work, we address this research gap by exploring and automatically circumventing inconsistent DNS censorship in China and Iran. To this end, we implemented previously known DNS circumvention techniques and encrypted DNS protocols in an open-source circumvention tool DPYProxy-DNS—an extension of DPYProxy [21]. Using DPYProxy-DNS, we present an updated and comprehensive overview of current DNS censorship techniques in China and Iran, two of the most prominent global censors. Our findings emphasize the need for automated censorship circumvention approaches, facing unknown and inconsistent censorship behavior. We also detect that the DNS configurations of modern browsers are prone to censorship.

Free and Open Communications on the Internet 2026(1), 1–10

In summary, we contribute the following:

- We perform a systematic evaluation of current DNS support and censorship practices. Our results highlight the DNS support of public DNS resolvers, their censorship in Iran and China, and techniques for censorship circumvention. We reveal inconsistent DNS censorship in both countries—highlighting the need for automated circumvention.
- We identify two issues in browsers: browsers issue unencrypted DNS queries for their pre-defined encrypted DNS servers and always include the Server Name Indication (SNI) extension in their encrypted SNI queries—showing that encrypted DNS in browsers is no solution to DNS censorship.
- Motivated by inconsistencies in real-world DNS censorship and limited encrypted DNS settings in browsers, we implemented an automated DNS censorship circumvention tool that successfully circumvents censorship in Iran and China. We provide the source code of our tool in a public repository as an extension for DPYProxy.[1]

2 Background

In the following, we introduce the necessary background concepts for our work. We introduce unencrypted and encrypted DNS and explain how TLS encrypts DNS queries. Afterward, we detail censors' possibilities to analyze and block all forms of DNS. Lastly, we outline standard deployments of encrypted DNS in browsers.

2.1 DNS

The most common use case of DNS is to resolve a domain to an IP address before a client accesses a website. The client sends a type *A* query containing the domain to a DNS resolver, and the DNS resolver responds with the IP address associated with the domain. DNS was initially standardized to run unencrypted on port 53 over TCP and UDP [27].

Encrypted DNS. To enhance privacy and to prevent third parties from analyzing unencrypted DNS directly, several encrypted DNS protocols exist. In this work, we analyze five encrypted DNS protocols: DoT [18], DoH [17], DoH3 [17, 19], DoQ [19], and DoD [37]. DoT runs on port 853 over TCP, DoD on port 853 over UDP, DoH on port 443 over TCP, DoH3 on port 443 over UDP, and DoQ on port 853 over UDP. A more detailed overview of the different protocols can be seen in Appendix A. In general, one DNS resolver can support multiple encrypted DNS protocols while also supporting unencrypted DNS. All of these encrypted DNS resolvers have a corresponding domain, which is used for messages on the TLS and HTTP layer. Notably, DNS resolvers can be accessed via domain names. In that case, the domain of the DNS resolver has to be resolved over potentially unencrypted DNS queries first.

2.2 TLS

All of the analyzed encrypted DNS protocols use TLS to encrypt DNS queries. Before sending the encrypted DNS queries, TLS uses unencrypted messages to exchange secrets and agree upon connection parameters. The most important unencrypted message is the first message from the client to the server: the *ClientHello* message. This message usually contains multiple extensions—specifying further important information like the supported versions of the client. The SNI extension is optional and contains the domain that a client is attempting to access. The SNI extension is required when multiple domains share the same IP address; without it, the server can not select the correct certificate. DNS servers can often be accessed with and without an SNI extension. Their certificates often contain their hostnames; however, they can also include their IP addresses. Including the IP address directly allows clients to use encrypted DNS while omitting the SNI extension, verifying the server's identity via their IP. This prevents censors from analyzing the SNI extension. Theoretically, browsers could store the SNI value for certificate validation even without sending the SNI. In practice, browsers verify the certificate for the IP in case no SNI extension is sent.

2.3 Censorship

Censorship refers to the practice of restricting access to certain information. It is employed by many countries in the form of Internet censorship—restricting access to certain websites or services [7, 23, 46]. To determine whether a user accesses a website or service, censors analyze many different protocols, such as DNS [3], TLS [30], HTTP [13], or TCP [6], gathering as much information as possible. To block a connection, censors drop packets or inject packets that tear down a connection: for TCP-based connections, censors can inject TCP RST packets; for UDP-based connections, censors have to inject protocol-specific answers such as false DNS responses.

Unencrypted DNS Censorship. When a client is accessing a website and uses standard—unencrypted—DNS over UDP, a censor can read the name—for example, `wikipedia.org`—a client is trying to resolve and can then decide whether to censor the request. An in-path censor—analyzing packets before forwarding them—can, for example, drop a to-be-censored query and answer with its own response, containing an IP that leads to a block page or another harmless site. One example of such a censor is the one in Iran [4, 23]. An on-path censor—analyzing packets after forwarding them—can, for example, directly inject its own response while trying to be faster than the intended response from the contacted DNS resolver. By default, clients interpret the first observed answer to a sent query. One example of such a censor is the GFW in China [3, 15]. An example of an in-path and an on-path censor can be found in Appendix B.

Encrypted DNS Censorship. All of the analyzed encrypted DNS protocols use TLS internally to encrypt the following application messages, which makes it impossible for a censor to analyze them to censor only specific domains. Instead, a censor can decide to block all traffic to known public resolvers that support these encrypted protocols based on the observed IP addresses and used ports. To avoid overblocking of services running on the same IP or port, censors can instead analyze the domain transmitted within the SNI extension in the *ClientHello* message. If a connection that uses encrypted DNS should be censored, injecting manipulated DNS responses is not an option due to TLS's authentication. Instead, for example, an in-path censor can drop packets to create timeouts, and

[1] https://github.com/UPB-SysSec/DPYProxy

an on-path censor can inject TCP RST packets for all TCP-based protocols like DoT and DoH. Additionally, Siby et al. [38] showed that it is possible to fingerprint encrypted DNS traffic with respect to specific resolved domains. While possible in theory, we are not aware of any censor utilizing such techniques.

2.4 Encrypted DNS in Browsers

To enhance the privacy of their users—by encrypting DNS requests and hiding them from third parties—all major browsers support encrypted DNS. While we focus on desktop clients, mobile clients also support encrypted DNS. More specifically, browsers support DoH, which can usually be configured in the browser's settings. In light of this, when using encrypted DNS in a censored country, the most straightforward way is to configure the used browser directly to use DoH. A user can either directly enter the domain or IP of a DNS resolver that supports DoH or select one of the resolvers that are configured by default in a drop-down menu. This includes Cloudflare and NextDNS for Firefox, and for Chromium, this includes NextDNS, CleanBrowsing, Cloudflare, Google, and OpenDNS. Notably, when configuring one of the default options or a custom resolver domain, browsers first perform an unencrypted DNS query to determine the resolver's IP. This unencrypted DNS request can be omitted by configuring a custom resolver IP address instead. Initially, we planned to test Safari; however, Safari does not have any settings directly for encrypted DNS and instead uses the resolver configured in the operating system's settings.

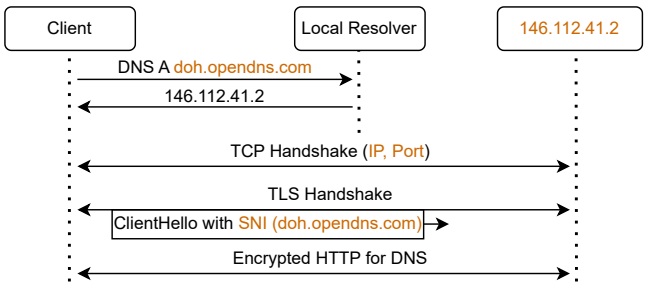

Figure 1: DoH usage example in Chromium with default OpenDNS resolver. A censor can use everything highlighted in orange to censor the connection at this point.

3 The Diverse Possibilities for DNS Traffic Censorship

Figure 1 visualizes the censorship possibilities against DoH in Chromium. All fields highlighted in orange inform a censor about the presence of DoH, allowing it to tear down the entire connection.

Resolving the Resolver's Domain. At first, Chromium sends an unencrypted DNS request over UDP for the domain doh.opendns.com to the system's default resolver. As this is done unencrypted, a censor can observe the requested domain and decide whether to censor this request. If the request is censored, it could, for example, be dropped or another IP could be injected, leading to an unintended site, block page, or possibly another DoH resolver that the censor controls.

TCP Handshake. After this initial request, if the request was not censored, a TCP handshake is performed with the actual server. At this point, a censor could observe the IP and/or port that the resolver uses. This can be used by a censor to either censor the resolver in its entirety—via IP—or censor the usage of specific protocols like DoH—via specific ports. The censor can interrupt the connection in this step by either dropping the packets or injecting TCP RST packets.

TLS Handshake. After the TCP connection has been established, the next step is the TLS handshake. The important part for a censor is the first message—the *ClientHello*—containing the SNI extension. This can further be used by a censor to identify a specific resolver. For OpenDNS, Chromium always sends the value doh.opendns.com. If the censor decides to censor the connection at this point, it can drop all packets or inject TCP RSTs.

Encrypted DNS Request. Finally, the encrypted DNS request is sent. Although this DNS request is encrypted, the message flow still enables the censor to perform censorship based on three main aspects: the initial unencrypted DNS request, the TCP IP and port information, and the transmitted SNI value. In this work, we analyze these ways to censor encrypted DNS traffic and provide a tool that can circumvent all mechanisms.

4 DPYProxy-DNS

To complement existing circumvention methods and tools with DNS censorship circumvention methods, we developed DPYProxy-DNS: an extension of the open-source censorship circumvention tool DPYProxy. DPYProxy-DNS provides uncensored DNS services to a local client. It acts as a regular DNS resolver and circumvents DNS censorship by automatically testing and determining a successful circumvention technique. We drew our circumvention techniques from existing standardization [17, 17–19, 19, 27], such as encrypted DNS—and previous research [6, 13, 15]—such as TCP segmentation. Notably, DPYProxy-DNS is developed in Python, and we tested it on Linux, Windows, and macOS. In the future, it would be a promising direction to add a mobile version. In the following, we list the eight circumvention modes supported by DPYProxy-DNS and detail how DPYProxy-DNS automatically determines a working censorship circumvention technique.

4.1 Circumvention Modes

UDP. As a first mode, DPYProxy-DNS sends standard DNS messages over **UDP** port 53 to a specified DNS resolver. This might be successful for censorship systems that simply do not censor specific resolvers. It can also serve as a baseline to see if a resolver is censored at all.

Last Response. Motivated by the DNS censorship of the GFW [15], the second mode consists of sending DNS messages over UDP port 53 and then waiting for a specific config-supplied time. After waiting for this time—three seconds by default—DPYProxy-DNS takes the last received response in the **Last Response** mode as the answer to the sent message. An example of this mode is visualized in Appendix C. This mode is targeted specifically for censorship systems that are on-path and try to be faster than the resolver's original response.

TCP. As a third mode, DPYProxy-DNS sends DNS messages to a public resolver over **TCP** port 53. Censorship systems might fail to censor this mode due to solely focusing on the way most DNS queries are performed, which is over UDP.

TCP Segmentation. The method of splitting one message into multiple smaller TCP segments—called **TCP segmentation**—is a highly successful circumvention technique for other protocols [6, 13]. In light of this, DPYProxy-DNS sends DNS messages with applied TCP segmentation to TCP port 53 of public resolvers as its fourth mode. During TCP segmentation, DPYProxy-DNS sends the DNS message split across multiple fragments of equal size: the default is set to 20 bytes.

Encrypted DNS. The remaining four modes consist of sending DNS messages over encrypted protocols to a public DNS resolver. DPYProxy-DNS supports **DoT**, **DoH**, **DoH3**, and **DoQ**. For all of these four modes, it is configurable whether the corresponding hostname is sent in the SNI extension of TLS. If it is being sent, DPYProxy-DNS automatically verifies the server's certificate; otherwise, the user receives a warning that the certificate is not verified. Not verifying the certificate in TLS can result in a potential Man-in-the-Middle attack by a third party, leading to possibly manipulated DNS responses. While it is still possible in theory to save the correct hostname first and then use it to validate the certificate without sending it, it is out of the scope of our tool. Notably, we did not implement DoD as a mode since early tests have shown that no tested resolver supports it.

4.2 Tool Usage

Once started, DPYProxy-DNS operates as a local DNS resolver and accepts unencrypted DNS requests over UDP and TCP. Using one of the modes described in Section 4.1, DPYProxy-DNS applies a circumvention method to the request, forwards it to another resolver, and returns the result to the client. The selected circumvention mode and its application are agnostic to the DNS client: it suffices to configure DPYProxy-DNS as a DNS resolver to circumvent censorship.

Selection of Modes. DPYProxy-DNS can be configured to disable any of the eight specified modes and is also written to easily implement and add more modes in the future. Notably, to harden possible fingerprinting, we pre-generate a list of all mode–resolver combinations that we want to test and then randomize the execution order. A combination is labeled as working if—given a censored domain and possible IP ranges as parameters—the censored domain resolves to the correct IP and is not censored for a majority of five consecutive tries.

General Workflow. To determine DNS censorship, DPYProxy-DNS attempts to resolve www.google.dj against Google's IP ranges[2]. In regions where www.google.dj is not censored, users can supply the tool with a custom hostname and IPs. Once a mode–resolver combination is labeled as working, DPYProxy-DNS uses this combination to send all DNS messages. Furthermore, the working configuration is saved to a local file, which can be read in a later execution to minimize the time for startup—the read-in configuration is verified once to make sure it still works. Our tool is already implemented with modularization such that, in the future, circumvention strategies for other protocols can be used alongside DNS. For example, the already existing TLS module of DPYProxy can be used at the same time as our new DNS module. When both modules are running, the TLS module uses the DNS module for all of its DNS queries by default—circumventing DNS and TLS censorship at the same time. Additional resolvers such as UncensoredDNS [36] can be added flexibly to DPYProxy-DNS by extending a single Enumerator.

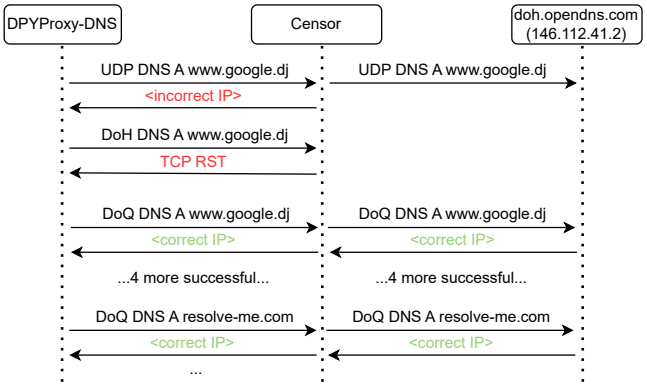

Figure 2: Example for the illustration of DPYProxy-DNS's auto mode with www.google.dj chosen as the censored domain. After several tries, it selects the combination of DoQ and doh.opendns.com as a successful circumvention.

Auto Mode. To automate the selection of a successful mode, DPYProxy-DNS has an additional auto mode, which is used by default. Figure 2 visualizes the auto mode with an example. After trying several modes, DoQ is finally selected as a working circumvention strategy for all further DNS requests.

5 Evaluation Methodology

With our evaluation, we determine what protocols DNS servers support and whether they work in censored countries. To this end, we first gathered a list of public DNS resolvers to analyze. Then, given this list, we determined which of DPYProxy-DNS's modes are supported by each resolver. Finally, given all supported resolver–mode combinations, we analyzed which of those combinations are censored from a server on which the scan is executed. In the following, we outline the methodology of our evaluation.

5.1 List of Public DNS Resolvers

As a starting point, we compiled a list of all public DNS resolvers that are configured by default on Firefox and Chromium (cf. Section 2.4). To this list, we added all resolvers from dnsprivacy[3] and wikipedia.[4] Some providers have multiple IPs, possibly enabling different filtering mechanisms like a family filter that filters out websites for adults only. We included all of these different IPs in our tests. In total, we gathered six providers from Adguard, six from

[2]https://www.gstatic.com/ipranges/goog.json, Accessed: 01.07.2025

[3]https://dnsprivacy.org/public_resolvers/, Accessed: 17.03.2025

[4]https://en.wikipedia.org/wiki/Public_recursive_name_server, Accessed: 17.03.2025

CleanBrowsing, six from Cloudflare, two from Google, two from Gcore, six from Mullvad, seven from Cisco, six from Quad9, one from Wikimedia, six from Yandex, and two from NextDNS. Notably, because OpenDNS was bought by Cisco in 2015,[5] we listed the configured OpenDNS resolver in Chromium as a seventh resolver for Cisco. To not reveal working resolvers to censors, we explicitly omit a list of IPs and hostnames from the paper.

5.2 Support Evaluation

For the support evaluation, we executed DPYProxy-DNS on a VM that is located within the DFN[6]. For this evaluation, we tested all public DNS resolvers within our compiled list with all eight implemented modes. We performed one scan with and one scan without SNI present to additionally test whether the providers need this extension to be present. Additionally, as we were unaware of any resolvers supporting DoD, we ran a simple test to see if any of the analyzed resolvers perform a DTLS handshake.

5.3 Censorship Evaluation

For all supported mode–resolver combinations, we then evaluated whether a combination works as a successful DNS censorship circumvention strategy from a vantage point within a specific country. For each country, we performed two scans: one with the SNI extension present and one without. A mode–resolver combination is labeled as successful if out of a minimum of three tries, 2/3 successfully circumvent DNS censorship. If this is not the case, DPYProxy-DNS iteratively performs more tries in an attempt to reach this threshold until a maximum of 20. This is similar to the methodology applied by Niere et al. [30] when circumventing TLS censorship in China and Iran.

We evaluated DNS censorship in China and Iran, which are known for their extensive DNS censorship [3, 4, 15, 23]. For the evaluation in China, we selected www.google.dj as a censored domain, and the corresponding Google IP ranges as target IP ranges. We chose this domain as it triggers all three DNS injectors in China at the same time [3]—we confirmed this behavior. In Iran, we selected twitter.com as the censored domain and 10.10.34.34–10.10.34.36 as the block page IPs to avoid [23]. The specifications of the rented vantage points in Iran and China are present in Appendix D. Additionally, we rented a server in Russia to analyze; however, we had to exclude the results from our evaluation as we could not find any DNS censorship present at our vantage point.

As the default configurations for DoH on browsers require a DNS query to resolve the corresponding domain first, we check those domains for DNS censorship as a last part of this censorship evaluation. For this, we compiled a list of all domains that the analyzed resolvers use for DoH, such as security.cloudflare-dns.com for Cloudflare's public resolver with the security filter enabled.

5.4 Auto Mode Evaluation

Finally, we evaluated the time required for the auto mode to find a successful combination. We consider this evaluation important as it reflects the time a user has to wait when starting the tool and detecting a usable circumvention method for the first time. In light of this, we started our tool with the auto mode enabled 100 times in each analyzed country: Iran and China. For each run, we recorded the time for subsequent statistical analysis.

[5] https://newsroom.cisco.com/c/r/newsroom/en/us/a/y2015/m08/cisco-completes-acquisition-of-opendns.html, Accessed: 21.10.2025

[6] German National Research and Education Network, https://www.dfn.de/

6 Findings

We performed our support evaluation at the beginning of March 2025. Then, we evaluated censorship from our vantage points in China and Iran in the middle of March 2025. Due to initially missing the third DNS injector in China, we performed a reevaluation of the censorship in China in June. In the following, we detail our findings.

Table 1: DNS Resolver support. Most resolvers have a certificate for the IP directly, and all support connections without the SNI. No resolver requires an SNI extension in encrypted mode.

	Unencrypted				Encrypted				
	UDP	Last Resp.	TCP	TCP Seg.	DoT	DoH	DoH3	DoQ	DoD
Adguard (6)	●	●	●	●	●IP	●IP	●IP	●IP	–
Cisco (6)	●	●	●	●	●IP	●IP	–	–	–
Cisco Chromium (1)	●	●	●	●	–	●IP	–	–	–
CleanBrowsing (5)	●	●	●	●	●	●	–	–	–
CleanBrowsing (1)	●	●	●	●	●	–	–	–	–
Cloudflare (6)	●	●	●	●	●IP	●IP	●IP	–	–
Gcore (2)	●	●	●	●	–	–	–	–	–
Google (2)	●	●	●	●	●IP	●IP	●IP	–	–
Mullvad (6)	–	–	–	–	●	●	–	–	–
NextDNS (2)	–	–	–	–	●	●	●	●	–
Quad9 (6)	●	●	●	●	●IP	●IP	–	–	–
Wikimedia (1)	–	–	–	–	●	●	–	–	–
Yandex (6)	●	●	●	●	●IP	●IP	–	–	–
# Supporting	41	41	41	41	47	47	16	8	0

● = Supported mode.
– = Unsupported mode.
IP = The resolver's certificate contains its IP address, allowing certificate validation without the SNI. Unmarked resolvers only possess domain certificates.

6.1 Support Evaluation

Table 1 shows DNS resolvers' support for the techniques introduced in 4.1. All resolvers except Mullvad, Wikimedia, and NextDNS support the four unencrypted circumvention modes: UDP, Last Response, TCP, and TCP Segmentation. Mullvad, Wikimedia, and NextDNS explicitly advertise merely providing encrypted DNS. The general support for these four modes is unsurprising, as all are standard-conforming. The Last Response mode just adds additional waiting time for the client, and TCP segmentation is a standard feature of TCP. Notably, for these four modes, the SNI extension does not play a role as TLS is not present.

For encrypted DNS, the Adguard and NextDNS resolvers are the only ones that support DoQ. DoH3 is supported by all Adguard, Cloudflare, Google, and NextDNS resolvers. All resolvers except those from Gcore—Gcore only advertises unencrypted DNS—and the default Chromium resolver from Cisco support DoT. The latter may be because Chromium uses this resolver only for DoH. DoH is surprisingly not supported by one of the CleanBrowsing resolvers. All other public resolvers that generally advertise any form of encrypted DNS support DoH. We contacted CleanBrowsing support

to learn more about this potential issue and were informed that it is intentional. Notably, all resolvers support encrypted DNS resolution without the SNI extension being present in an encrypted mode (DoT, DoH, DoH3, DoQ). DoD is not supported by any resolver, which is why we did not implement this mode within DPYProxy-DNS.

IP Certificates. We detected that most resolvers present certificates containing a hostname and their IP address. Still, CleanBrowsing, Mullvad, Wikimedia, and NextDNS present certificates containing only the hostname: these can not be validated against the resolver's IP address. When supplying the hostname in the SNI value in the TLS handshake, certificate validation is performed implicitly by the underlying TLS library for both types of certificates. However, not sending the SNI value is desirable by circumvention tools as it hides information from the censor. For certificates that only contain the resolver's hostname, this introduces a challenge: the hostname is needed as an input for the certificate validation without sending it in the SNI extension.

While DPYProxy-DNS sends the SNI extension by default, the SNI extension can be disabled by the user. If no SNI is sent and the resolver does not have a certificate that is valid for the IP, we do not validate the certificate. In such cases, we print a warning to the user, as not validating the certificate of the server is dangerous due to possible Man-in-the-Middle attacks on the client that manipulate answers. In the future, DPYProxy-DNS could be extended to take a hostname as a parameter for its certificate validation regardless of whether it was sent in the SNI extension.

6.2 Censorship Circumvention Evaluation

Table 2 shows the results of our censorship evaluation. One key takeaway is that the Last Response and TCP Segmentation modes are very effective in censorship circumvention for unencrypted DNS in China. Additionally, we determine that the censorship of encrypted DNS is inconsistent in both countries, showing the need for an automated tool to discover such missed circumvention opportunities. In the following, we detail our findings.

6.2.1 China.

Unencrypted DNS. As a first observation, the GFW analyzes and censors all standard UDP and TCP DNS queries. This was expected as it conforms with previous research [3, 15]. The Last Response mode was very successful as a circumvention technique for UDP queries: it works on nearly all resolvers that support this mode. The exception is one Cloudflare resolver (1.1.1.1), which was fully IP-blocked via packet dropping. Due to how the GFW censors DNS for the other resolvers, it is enough to wait for three seconds and then take the last received response—which is the one from the queried resolver. This slows down the name resolution process for the user; however, once a name has been resolved, it can be cached and reused so that the overall overhead becomes minimal. This shows that utilizing information about the censor's infrastructure can be very beneficial when designing circumvention strategies. Another successful circumvention mode that worked for nearly all resolvers—except for the IP-blocked Cloudflare resolver—is TCP segmentation. This success was also observed for other protocols by past research [6, 13].

Encrypted DNS. DoQ and DoH3 works for all NextDNS resolvers, even with the SNI extension present. DoT also works with SNI present for all Mullvad, Wikimedia, and NextDNS resolvers. The GFW also does not censor DNS requests over DoH to all CleanBrowsing, Mullvad, Wikimedia, and NextDNS resolvers, even with the SNI extension present. On the other hand, all Yandex resolvers are only censored if the SNI extension is present. If it is omitted, these resolvers can also be used to resolve DNS requests within China. Finally, in China, only the domains for encrypted DNS of all Mullvad resolvers are censored when performing standard DNS requests at our vantage point.

6.2.2 Iran.

Unencrypted DNS. Consistent with the results from China, all resolvers are censored over standard UDP and TCP. The Last Response mode is not successful in Iran because the censor operates in-path, and the original request from the client never reaches the DNS resolver. The TCP Segmentation mode is partly successful in Iran as it works inconsistently across all resolvers that support it. During some repeated scan runs, it fully works in one run, while fully failing for all resolvers on others. We suspect that the Iranian censor can reassemble TCP fragments, depending on the current workload of the censorship system. Notably, TCP segmentation can theoretically still be used as a circumvention technique in Iran when enough requests are made—one request will work eventually.

Encrypted DNS. In Iran, DoQ is not censored at all and also works with an SNI extension present. DoH3 works for all six Cloudflare and NextDNS resolvers. Interestingly, one Cloudflare resolver (1.1.1.1) replies with the status code *Forbidden* at the HTTP layer when supplying an SNI value in Iran. During multiple executions, this happened consistently for Iran and never for China or in our support evaluation. We suspect this may occur because Cloudflare enforces certain export regulations on queries originating from IP addresses it determines to be located in Iran. We are uncertain why this occurs only with one of Cloudflare's resolvers.

DoH works for all Yandex and Cisco resolvers with the SNI present. All Quad9, Google, Adguard, CleanBrowsing, and NextDNS resolvers that support this protocol work only when the SNI extension is not present. DoT works for one Yandex, four Cisco, and all Wikimedia resolvers with the SNI extension present. Additionally, it works for two Quad9, three CleanBrowsing, all Adguard, all Google, and all NextDNS resolvers when the SNI extension is omitted. This shows that the observed censorship at our vantage point in Iran is more reliant on the SNI extension being present to perform censorship when compared to the censorship that we observed in China. Finally, only the non-Chromium Cisco, Wikimedia, and Yandex resolvers' domains were resolvable at our local DNS resolver at our vantage point.

6.2.3 The Effectiveness of DPYProxy-DNS.

DPYProxy-DNS's auto mode successfully circumvents DNS censorship at our Iranian and Chinese vantage points. The auto mode works as long as there is one successful resolver—mode combination from the machine on which the tool is executed. As our evaluation showed, many possible combinations work in Iran and China. Therefore, we strongly believe that DPYProxy-DNS can successfully circumvent DNS censorship in other countries as well.

Table 2: DNS Resolver support in China and Iran. Our results show that two unencrypted modes—Last Response and TCP Segmentation—are successful in China. Also, many encrypted DNS resolvers are reachable with some of the encrypted DNS protocols—some of them even with the SNI present.

	Unencrypted								Encrypted								Domain Resolvable?	
	UDP		Last Resp.		TCP		TCP Seg.		DoT		DoH		DoH3		DoQ			
Adguard (6)	○	○	●	○	○	○	●	◐	○	●-SNI	○	●-SNI	○	○	○	●	●	○
Cisco (4)	○	○	●	○	○	○	●	◐	○	●	○	●	–	–	–	–	●	●
Cisco (2)	○	○	●	○	○	○	●	◐	○	○	○	●	–	–	–	–	●	●
Cisco Chromium	○	○	●	○	○	○	●	◐	–	–	●-SNI	●-SNI	–	–	–	–	●	○
CleanBrowsing (3)	○	○	●	○	○	○	●	◐	○	○	●	●-SNI	–	–	–	–	●	○
CleanBrowsing (2)	○	○	●	○	○	○	●	◐	○	●-SNI	●	●-SNI	–	–	–	–	●	○
CleanBrowsing (1)	○	○	●	○	○	○	●	◐	○	●-SNI	–	–	–	–	–	–	●	○
Cloudflare (5)	○	○	●	○	○	○	●	◐	○	○	○	○	○	●[2]	–	–	●	○
Cloudflare (1)	○[1]	○	○[1]	○	○[1]	○	○[1]	◐	○	○	○	○	○	●[2]	–	–	●	○
Gcore (2)	○	○	●	○	○	○	●	◐	–	–	–	–	–	–	–	–	–	–
Google (2)	○	○	●	○	○	○	●	◐	○	●-SNI	○	●-SNI	○	○	–	–	●	○
Mullvad (6)	–	–	–	–	–	–	–	–	●	○	●	○	–	–	–	–	○	○
NextDNS (2)	–	–	–	–	–	–	–	–	●	●-SNI	●	●-SNI	●	●	●	●	●	○
Quad9 (4)	○	○	●	○	○	○	●	◐	○	○	○	●-SNI	–	–	–	–	●	○
Quad9 (2)	○	○	●	○	○	○	●	◐	○	●-SNI	○	●-SNI	–	–	–	–	●	○
Wikimedia	–	–	–	–	–	–	–	–	●	●	●	○	–	–	–	–	●	●
Yandex (5)	○	○	●	○	○	○	●	◐	○	○	●-SNI	●	–	–	–	–	●	●
Yandex (1)	○	○	●	○	○	○	●	◐	○	●	●-SNI	●	–	–	–	–	●	●
# Working	0	0	40	0	0	0	40	–	9	21	21	34	2	8	2	8	42	13

– = Unsupported, ○ = Supported but censored, ◐ = Works inconsistently, ● = Works.
-SNI = Works only when omitting the SNI extension.
[1] Not working due to complete IP blocking via packet dropping.
[2] With SNI, it is not censored by the censor; instead, Cloudflare replies with a Forbidden status code at the HTTP layer.

6.3 Auto Mode Evaluation

For this evaluation, we computed timing statistics of 100 auto mode runs in Iran and China. The average time in China (13.78s) and in Iran (12.32s) is nearly the same, with the median in Iran (8.28s) being lower than the one in China (12.90s). The minimum values of 0.32s and 0.47s are due to a randomly chosen first combination that directly worked—like a Cisco resolver with DoH in Iran or a Google resolver with TCP segmentation in China. The high maximum values of 30.72s in China and 58.16s in Iran can be explained by the Last Response mode: while effective in China, it is the mode with the longest runtime due to the constant waiting time. In Iran, the maximum time is so high because the Last Response mode is not successful at all, leading to longer waiting times if such combinations are randomly chosen. We could have excluded the Last Response mode from our measurements; however, we wanted to examine the runtime of the auto mode without any prior knowledge and additional configuration. In practice, a user in Iran could disable the Last Response mode and significantly decrease the average runtime. Notably, this initial runtime is only necessary once: when a successful combination is found, it is saved in a configuration file for later executions. Furthermore, once a working combination has been determined, it can be used for all further DNS queries directly.

7 Discussion

Can Censors Block Encrypted DNS Altogether? Censors can block encrypted DNS via IP, port, and the domain in the SNI extension. DoT and DoQ are detectable as they use port 853; DoH and DoH3 use port 443, hiding the encrypted DNS traffic along other common protocols on this port: TLS and QUIC. This makes it challenging for a censor to block all DoH and DoH3 resolvers; this is especially the case when there is a website running on the same IP or when a proxy is used. In such a scenario, a censor would have to overblock or rely on SNI parsing. However, we showed that all resolvers also support connections without an SNI value. This allows one to establish an encrypted connection to DNS resolvers without an SNI extension. If SNI values must be used, one could use TLS censorship circumvention techniques [30]. Encrypted `ClientHello` (ECH) might also be an option although it is sparsely supported [16] and is itself limited by DNS censorship [29].

Browser Usage. When selecting one of the configured encrypted DNS resolvers in Firefox and Chromium, the browser first performs an *unencrypted* DNS request to resolve the domain for that resolver. This initial DNS request can only be circumvented when configuring a custom resolver while directly supplying the correct IP. This is a concerning observation in light of the censorship that we found in Iran. Nearly all of the tested domains are directly censored in Iran via DNS, returning a block page IP. This shows that a censor can effectively censor encrypted DNS by just censoring the first unencrypted DNS query. Furthermore, browsers do not have settings for including the SNI extension in encrypted DNS requests. This is the case even though no resolver requires it (cf. Table 1). We suggest browsers enable SNI omission for encrypted DNS to increase its censorship resilience.

Interplay of Different Protocols. The interplay of different protocols plays a crucial role in DNS censorship circumvention. All encrypted DNS protocols use multiple protocols internally that are

analyzed by censors such as TCP, UDP, TLS, and QUIC. While this provides a target for censors, it also allows more techniques for successful censorship circumvention. A promising future direction would be to combine known working circumvention techniques for multiple protocols simultaneously. Furthermore, the interplay of protocols is also important for DoH usage within browsers, as they perform an unencrypted DNS query first. This shows that even if censorship does not affect encrypted DNS directly in a country—or if it can be circumvented—it is enough to censor this one initial unencrypted DNS query to effectively make encrypted DNS unusable.

The Need for Automated Circumvention. Our censorship analysis in Iran and China (cf. Table 2) showed that there are encrypted DNS resolvers, where only a subset of the encrypted DNS protocols are censored. We suspect that some of them have been accidentally missed by the censors. Even though encrypted DNS in general is a known circumvention technique, this shows the need for automated circumvention tools that find such missed opportunities. Furthermore, the modularity of DPYProxy-DNS makes it easy to add more circumvention techniques for DNS censorship in the future. We strongly believe that our provided DPYProxy-DNS helps affected people to circumvent encrypted DNS censorship.

8 Related Work

Censorship Circumvention Tools. Circumventing DNS censorship can aid widely used circumvention tools that focus on TCP-based protocols such as TLS. For this reason, we implemented DPYProxy-DNS as an extension to the existing censorship circumvention tool DPYProxy. Other noteworthy censorship circumvention tools can potentially benefit from DPYProxy-DNS [8, 14, 22, 25, 42, 48]. Circumvention tools that tunnel users' DNS traffic do not require additional DNS censorship circumvention. Tor Snowflake [40], V2Ray [41], and VPNs are examples.

Worldwide View on the Reachability of Encrypted DNS. The paper by Li et al. [24] in 2024 is closely related to our work. They examined the reachability of DoT, DoH, DoH3, and DoQ on a global scale by first compiling a list of 1,302 encrypted DNS resolvers and then scanning them from 5,000 vantage points. They found that, in general, encrypted DNS is widely blocked in censored regions. We checked whether circumvention techniques like TCP segmentation are successful while focusing on publicly known resolvers. This focus allowed us to provide more in-depth results and to argue whether browsers allow effective encrypted DNS usage.

Manipulations in Encrypted DNS Responses. In 2021, Jin et al. [20] performed 7.4 million DNS queries on 3,813 DoT and 75 DoH resolvers and found that 1.66% of DoT and 1.42% of DoH responses have DNS manipulations present. This is why we chose well-known resolvers for our evaluation and why we still validate the returned IP address for our encrypted DNS modes.

Manipulations on the DNS Layer. In 2022, Harrity et al. [13] found several manipulations directly on the DNS layer to circumvent censorship in China. For example, they successfully circumvented DNS censorship by tampering with the count of questions specified in the message or activating DNS compression. DPYProxy-DNS does not manipulate the fields of the DNS; instead, it circumvents censorship by using different DNS protocols and enabling lower-layer features like TCP segmentation. With these standard-compliant circumventions, we hope to be compatible with as many resolvers as possible—none of the successful strategies from Harrity et al. worked with all tested resolvers.

9 Ethical Considerations

Our methodology has a minimal impact on third-party servers and people living in affected countries. For the support and censorship scan of public resolvers, we performed a maximum of 40 DNS requests for each resolver per mode: 20 with and 20 without SNI. We consider this amount of requests negligible for any public DNS resolver that is used to handle large amounts of traffic. For the scan to determine whether an encrypted DNS resolver domain is censored, we sent one DNS request for each analyzed public resolver to our local resolver at the corresponding vantage point. This number of requests is negligible for any resolver. Notably, we did not involve any humans in our study.

While renting our used vantage points, we ensured no involved entity is present on the sanctions list that applies to us [10]. We also comply with the export regulations, as the tool we used for the scans does not aid any form of oppression in the analyzed countries. Our institutional export control officer confirmed our procedure.

10 Conclusions

We implemented DPYProxy-DNS, a tool that automatically circumvents DNS censorship. We validated the effectiveness of our tool by automatically circumventing DNS censorship in China and Iran. Successful DNS censorship circumvention methods in both countries differed and were inconsistent: we judge DPYProxy-DNS's ability to automatically detect a working circumvention technique crucial for successful DNS censorship circumvention. During our evaluations, we found that the implementation of DoH in browsers and especially its default settings are insufficient to bypass censorship successfully. Browsers could aid censorship circumvention efforts by omitting the SNI extension in the TLS handshake with the DNS resolver and pre-configuring its default DNS resolver with an IP address instead of a domain name: this eliminates an initial unencrypted DNS request. We hope that our findings motivate browser vendors to consider censorship circumvention in their encrypted DNS configuration. Similarly, we hope to assist affected people and encourage further research on censorship.

Acknowledgments

We thank the reviewers for their insightful comments and constructive feedback. Niklas Niere and Felix Lange were supported by the German Federal Ministry of Research, Technology and Space (BMFTR) through the project KoTeBi (16KlS1556K). Niklas Niere was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—555828767.

References

[1] Collin Anderson. 2013. Dimming the Internet: Detecting Throttling as a Mechanism of Censorship in Iran. https://doi.org/10.48550/arXiv.1306.4361 arXiv:1306.4361 [cs].

[2] Anonymous. 2014. Towards a Comprehensive Picture of the Great Firewall's DNS Censorship. In *4th USENIX Workshop on Free and Open Communications on the Internet (FOCI 14)*. USENIX Association, San Diego, CA. https://www.usenix.org/conference/foci14/workshop-program/presentation/anonymous

[3] Anonymous, Arian Akhavan Niaki, Nguyen Phong Hoang, Phillipa Gill, and Amir Houmansadr. 2020. Triplet censors: Demystifying great Firewall's DNS censorship behavior. In *10th USENIX workshop on free and open communications on the internet (FOCI 20)*. USENIX Association. https://www.usenix.org/conference/foci20/presentation/anonymous

[4] Simurgh Aryan, Homa Aryan, and J. Alex Halderman. 2013. Internet Censorship in Iran: A First Look. In *3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/foci13/workshop-program/presentation/aryan

[5] Abhishek Bhaskar and Paul Pearce. 2022. Many roads lead to rome: How packet headers influence DNS censorship measurement. In *31st USENIX security symposium (USENIX security 22)*. USENIX Association, Boston, MA, 449–464. https://www.usenix.org/conference/usenixsecurity22/presentation/bhaskar

[6] Kevin Bock, George Hughey, Xiao Qiang, and Dave Levin. 2019. Geneva: Evolving Censorship Evasion Strategies. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*. Association for Computing Machinery, New York, NY, USA, 2199–2214. https://doi.org/10.1145/3319535.3363189

[7] Kevin Bock, iyouport, Anonymous, Louis-Henri Merino, David Fifield, Amir Houmansadr, and Dave Levin. 2020. Exposing and Circumventing China's Censorship of ESNI. https://gfw.report/blog/gfw_esni_blocking/en/

[8] bol van. 2024. zapret. https://github.com/bol-van/zapret

[9] chantra. 2025. chantra/doh-proxy. https://github.com/chantra/doh-proxy original-date: 2018-01-11T04:44:20Z.

[10] European Union. 2025. *EU Sanctions Map*. https://www.sanctionsmap.eu/

[11] Phillipa Gill. 2016. A Case Study in Iran. https://iclab.gitlab.io/post/iran_case_study_2016/

[12] Google Jigsaw. 2024. Jigsaw-Code/Intra. https://github.com/Jigsaw-Code/Intra

[13] Michael Harrity, Kevin Bock, Frederick Sell, and Dave Levin. 2022. GET /out: Automated Discovery of Application-Layer Censorship Evasion Strategies. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 465–483. https://www.usenix.org/conference/usenixsecurity22/presentation/harrity

[14] Sadegh Hayeri. 2024. GreenTunnel. https://github.com/SadeghHayeri/GreenTunnel

[15] Nguyen Phong Hoang, Arian Akhavan Niaki, Jakub Dalek, Jeffrey Knockel, Pellaeon Lin, Bill Marczak, Masashi Crete-Nishihata, Phillipa Gill, and Michalis Polychronakis. 2021. How Great is the Great Firewall? Measuring China's DNS Censorship. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 3381–3398. https://www.usenix.org/conference/usenixsecurity21/presentation/hoang

[16] Nguyen Phong Hoang, Michalis Polychronakis, and Phillipa Gill. 2022. Measuring the Accessibility of Domain Name Encryption and Its Impact on Internet Filtering. In *Passive and Active Measurement*, Oliver Hohlfeld, Giovane Moura, and Cristel Pelsser (Eds.). Springer International Publishing, Cham, 518–536. https://doi.org/10.1007/978-3-030-98785-5_23

[17] P. Hoffman and P. McManus. 2018. *DNS Queries over HTTPS (DoH)*. RFC 8484. IETF. http://tools.ietf.org/rfc/rfc8484.txt

[18] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. 2016. *Specification for DNS over Transport Layer Security (TLS)*. RFC 7858. IETF. http://tools.ietf.org/rfc/rfc7858.txt

[19] C. Huitema, S. Dickinson, and A. Mankin. 2022. *DNS over Dedicated QUIC Connections*. RFC 9250. IETF. http://tools.ietf.org/rfc/rfc9250.txt

[20] Lin Jin, Shuai Hao, Haining Wang, and Chase Cotton. 2021. Understanding the Impact of Encrypted DNS on Internet Censorship. In *Proceedings of the Web Conference 2021*. ACM, Ljubljana Slovenia, 484–495. https://doi.org/10.1145/3442381.3450084

[21] JonSnowWhite. 2024. DPYProxy. https://github.com/UPB-SysSec/DPYProxy

[22] krlvm. 2024. PowerTunnel. https://github.com/krlvm/PowerTunnel

[23] Felix Lange, Niklas Niere, Jonathan von Niessen, Dennis Suermann, Nico Heitmann, and Juraj Somorovsky. 2025. I(ra)nconsistencies: Novel Insights into Iran's Censorship. *Free and Open Communications on the Internet* (2025). https://www.petsymposium.org/foci/2025/foci-2025-0002.php

[24] Ruixuan Li, Baojun Liu, Chaoyi Lu, Haixin Duan, and Jun Shao. 2024. A Worldwide View on the Reachability of Encrypted DNS Services. In *Proceedings of the ACM Web Conference 2024*. ACM, Singapore Singapore, 1193–1202. https://doi.org/10.1145/3589334.3645539

[25] macronut. 2024. ghostcp. https://github.com/macronut/ghostcp

[26] Alexander Master and Christina Garman. 2023. A Worldwide View of Nation-state Internet Censorship. *Free and Open Communications on the Internet* (2023). https://petsymposium.org/foci/2023/foci-2023-0008.php

[27] P.V. Mockapetris. 1987. *Domain names - implementation and specification*. RFC 1035. IETF. http://tools.ietf.org/rfc/rfc1035.txt

[28] Nima Nazeri and Collin Anderson. 2013. Citation Filtered: Iran's Censorship of Wikipedia. https://www.semanticscholar.org/paper/Citation-Filtered%3A-Iran%E2%80%99s-Censorship-of-Wikipedia-Nazeri-Anderson/15f13eb5c7fa7128cd6551d0fe1c285a7763c0ea

[29] Niklas Niere, Felix Lange, Nico Heitmann, and Juraj Somorovsky. 2025. Encrypted Client Hello (ECH) in Censorship Circumvention. *Free and Open Communications on the Internet* (2025). https://www.petsymposium.org/foci/2025/foci-2025-0016.php

[30] Niklas Niere, Felix Lange, Robert Merget, and Juraj Somorovsky. 2025. Transport Layer Obscurity: Circumventing SNI Censorship on the TLS-Layer. In *2025 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, San Francisco, CA, USA, 1344–1362. https://doi.org/10.1109/SP61157.2025.00151

[31] nomoresat. 2024. DPITunnel-android. https://github.com/nomoresat/DPITunnel-android

[32] Sadia Nourin, Van Tran, Xi Jiang, Kevin Bock, Nick Feamster, Nguyen Phong Hoang, and Dave Levin. 2023. Measuring and Evading Turkmenistan's Internet Censorship: A Case Study in Large-Scale Measurements of a Low-Penetration Country. In *Proceedings of the ACM Web Conference 2023*. ACM, Austin TX USA, 1969–1979. https://doi.org/10.1145/3543507.3583189

[33] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. 2017. Global Measurement of DNS Manipulation. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, Canada, 307–323. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/pearce

[34] DNSCrypt Project. 2013. DNSCrypt version 2 - Official Project Home Page. https://dnscrypt.info

[35] Reethika Ramesh, Ram Sundara Raman, Apurva Virkud, Alexandra Dirksen, Armin Huremagic, David Fifield, Dirk Rodenburg, Rod Hynes, Doug Madory, and Roya Ensafi. 2023. Network Responses to Russia's Invasion of Ukraine in 2022: A Cautionary Tale for Internet Freedom. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, USA, 2581–2598. https://www.usenix.org/conference/usenixsecurity23/presentation/ramesh-network-responses

[36] Rasmussen. 2009. UncensoredDNS - uncensoreddns blog. https://blog.uncensoreddns.org/

[37] T. Reddy, D. Wing, and P. Patil. 2017. *DNS over Datagram Transport Layer Security (DTLS)*. RFC 8094. IETF. http://tools.ietf.org/rfc/rfc8094.txt

[38] Sandra Siby, Marc Juarez, Claudia Diaz, Narseo Vallina-Rodriguez, and Carmela Troncoso. 2020. Encrypted DNS –> Privacy? A Traffic Analysis Perspective. In *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society, San Diego, CA. https://doi.org/10.14722/ndss.2020.24301

[39] Kushagra Singh, Gurshabad Grover, and Varun Bansal. 2020. How India Censors the Web. In *Proceedings of the 12th ACM Conference on Web Science (WebSci '20)*. Association for Computing Machinery, New York, NY, USA, 21–28. https://doi.org/10.1145/3394231.3397891

[40] The Tor Project. 2024. Snowflake. https://snowflake.torproject.org/

[41] v2fly. 2024. v2ray: A platform for building proxies to bypass network restrictions. https://github.com/v2fly/v2ray-core

[42] ValdikSS. 2024. GoodbyeDPI. https://github.com/ValdikSS/GoodbyeDPI

[43] Philipp Winter and Stefan Lindskog. 2012. How the Great Firewall of China is Blocking Tor. In *2nd USENIX Workshop on Free and Open Communications on the Internet (FOCI 12)*. USENIX Association, Bellevue, WA. https://www.usenix.org/conference/foci12/workshop-program/presentation/Winter

[44] Mingshi Wu, Jackson Sippe, Danesh Sivakumar, Jack Burg, Peter Anderson, Xiaokang Wang, Kevin Bock, Amir Houmansadr, Dave Levin, and Eric Wustrow. 2023. How the Great Firewall of China Detects and Blocks Fully Encrypted Traffic. 2653–2670. https://www.usenix.org/conference/usenixsecurity23/presentation/wu-mingshi

[45] Xueyang Xu, Z. Morley Mao, and J. Alex Halderman. 2011. Internet Censorship in China: Where Does the Filtering Occur?. In *Passive and Active Measurement*, Neil Spring and George F. Riley (Eds.). Springer, Berlin, Heidelberg, 133–142. https://doi.org/10.1007/978-3-642-19260-9_14

[46] Diwen Xue, Benjamin Mixon-Baca, ValdikSS, Anna Ablove, Beau Kujath, Jedidiah R. Crandall, and Roya Ensafi. 2022. TSPU: Russia's decentralized censorship system. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22)*. Association for Computing Machinery, New York, NY, USA, 179–194. https://doi.org/10.1145/3517745.3561461

[47] Diwen Xue, Reethika Ramesh, Valdik S S, Leonid Evdokimov, Andrey Viktorov, Arham Jain, Eric Wustrow, Simone Basso, and Roya Ensafi. 2021. Throttling Twitter: an emerging censorship technique in Russia. In *Proceedings of the 21st ACM Internet Measurement Conference (IMC '21)*. Association for Computing Machinery, New York, NY, USA, 435–443. https://doi.org/10.1145/3487552.3487858

[48] xvzc. 2024. SpoofDPI. https://github.com/xvzc/SpoofDPI

A Layer Overview of Analyzed Encrypted DNS Protocols

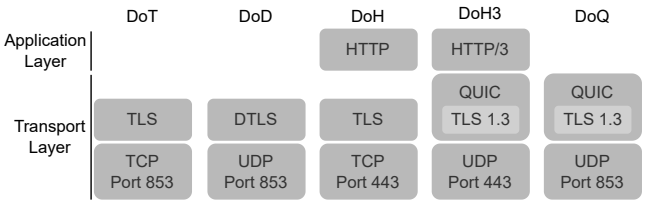

Figure 3: Layer overview of DoT, DoD, DoH, DoH3, and DoQ with additional port information.

B In-Path and On-Path Example

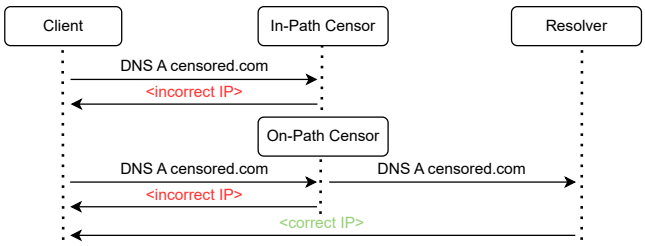

Figure 4: Example of an in-path and an on-path censor. The in-path censor can directly block the packet and inject its own response, while the on-path censor injects a response with the requirement to be faster than the intended response.

C Last Response Mode Example

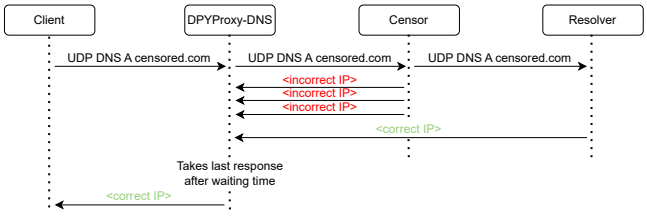

Figure 5: Example of DPYProxy-DNS's Last Response mode. The tool waits for a fixed waiting time and then chooses the last received message, which, in this example, is the correct IP.

D Server Specifications

Table 3: Specification of our Vantage Point in Iran.

Country:	Mashhad, Iran
Autonomous System Number:	201295
Vendor:	Avanetco
URL:	https://www.avanetco.com/
Internet Service Provider:	Shabakeh Ertebatat Artak Towseeh PJSC (private)

Table 4: Specification of our Vantage Point in China.

Country:	Zhengzhou, China
Autonomous System Number:	4837
Vendor:	China VPS Hosting
URL:	https://chinavpshosting.com/
Internet Service Provider:	CHINA UNICOM (state-owned)