ZDNS vs MassDNS: A Comparison of DNS Measurement Tools

Jeremy Dix, Patrick Sattler*, Johannes Zirngibl*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: jeremy.dix@tum.de, sattler@net.in.tum.de, zirngibl@net.in.tum.de

***Abstract*—The DNS is an ever-growing essential part of today's Internet. Its ability to function well is crucial for the Internet's stability and maintainability. To ensure that, the DNS needs to incorporate scalable and high-performing technologies. Active DNS measurement on a large-scale basis plays a critical role in developing these technologies. ZDNS and MassDNS are two frameworks that provide toolsets for performing such DNS measurements while taking different approaches to accomplish this task. This paper describes and compares these frameworks and their features in detail and provides an overview of their respective resolution accuracy. Most notably, it shows that both tools incorporate features, e.g., for internal recursion and lookup chain exposure (ZDNS) or subdomain enumeration (MassDNS). It demonstrates that, while both frameworks have a similar accuracy under the same circumstances, MassDNS tends to experience more timeouts in general. The paper aims to help researchers and other interested parties choose the right tool for their use case by better understanding their specific capabilities.**

***Index Terms*—dns, networking, performance, benchmark, measurement, zdns, massdns**

1. Introduction

Nowadays, the World Wide Web is bigger than ever before. According to the Domain Name Industry Brief, it currently comprises about 359.3 million domain names [1] and is exponentially growing each year. The Domain Name System (DNS) is critical for accommodating this vast number of domains and is integral to the Internet's infrastructure. Its primary purpose is to convert human-readable domain names into IP addresses.

Due to its fast growth [1], the DNS requires highly performing, scalable, and secure technologies to keep up with the increasing demand. In particular, the field of active DNS measurement is crucial for conducting the necessary research to create these technologies. For example, a research team at the University of Twente published a paper that describes the importance of extensive active DNS measurement and its challenges, e.g., the burdens imposed on the DNS by daily scans [2].

For conducting research involving active DNS measurement, there is a need for high-performance tools that can perform domain name resolution on a large scale. Two such recently developed tools that are actively being used for DNS research are ZDNS and MassDNS.

ZDNS is a command line tool implemented in Go [3], created by Durumeric et al. at Stanford University as part of the ZMap Project [4]. In contrast, MassDNS is a single-threaded DNS stub resolver written in C, created by Birk Blechschmidt, a security engineer at Deutsche Telekom, and Quirin Scheitle, a former researcher at the Technical University of Munich [5].

Comparing these two tools is particularly interesting, as their feature set, purpose, and commitment to delivering high resolution speeds are very alike. MassDNS, for instance, promises a speed of up to 350.000 lookups per second [5]. However, while these toolsets have the same primary purpose, they still differ in functionality, usability, resolution accuracy, and scalability. Depending on the use case, this might lead to one tool performing better. Hence, this paper compares both tools w.r.t. the latter properties to determine their suitability for specific use cases.

2. Related Work

ZDNS and MassDNS mainly contribute to the field of active DNS measurement. Joint research projects like TIDE [6] or OpenINTEL [7] are significant contributors to this research area as well. OpenINTEL measures more than 252 million domains every day, thus precisely logging the state of the DNS over time [7]. The University of Twente, as well as research organizations like SIDN Labs [8], SURFnet [9], and NLnet Labs [10] are all major contributors to OpenINTEL, as well as other active DNS measurement research projects (e.g., [2], [11], [12]). Notably, another DNS measurement platform, Censys Search [13], uses ZDNS for its DNS measurements [3]. MassDNS is also used in various DNS measurement research projects, e.g., [14] or [15].

The ZDNS developers themselves published a paper in 2022 in which ZDNS is compared to other DNS tools [3]. The paper explains ZDNS' properties, highlights its performance in large-scale domain resolution, and indicates possible use cases.

Apart from ZDNS and MassDNS, several other tools exist that were created by research teams for internal measurements in the context of various research projects [2], [16]. Additionally, there are several Linux tools with similar purposes, e.g., domain information groper (`dig`) [17], or `nslookup` [18]. However, in contrast to the frameworks presented in this paper, these solutions/tools are generally not published, preventing other scientists from using them in their research projects, or they are not optimized for large-scale DNS measurements.

Lastly, tools like ZDNS and MassDNS typically rely on third-party recursive resolvers, like Unbound [19] or the Cloudflare public resolver (1.1.1.1) [20] to resolve

 doi: 10.2313/NET-2024-04-1_10

domain names. These resolvers provide the necessary infrastructure for communication with the name servers in the DNS.

3. General Comparison

This chapter gives a comprehensive overview of both tools' features, compares their usability and implementation, and lists some of their approaches to fault mitigation. Table 1 summarizes both tools' most significant properties.

TABLE 1: Feature and property overview.

Feature	ZDNS	MassDNS
Number of fully supported record types	70	11
Supported L4 protocols	UDP, TCP	UDP
Concurrency method	Goroutines	epoll
Built-in resolver	✓	-
Iterative lookups	✓	✓
Lookup chain exposure	✓	-
Additionally queried record types	`A`, `CNAME`	`PTR`
Native subdomain enumeration support	-	✓
JSON output	✓	✓

3.1. Feature Overview

Both tools serve the purpose of resolving domain names by querying various resource record types. However, while ZDNS fully supports a total of 70 DNS record types [21], MassDNS only supports the 11 most commonly used types for its human-readable output formats: `A`, `AAAA`, `CAA`, `CNAME`, `DNAME`, `MX`, `NS`, `SRV`, `PTR`, `SOA`, and `TXT` [5].

Moving on, ZDNS features its own standalone caching recursive resolver library [22], which is built upon the DNS library by Miek Gieben [23]. It consists of all DNS resource record types, including the records from the DNS Security Extensions (DNSSEC) [22].

ZDNS' Built-in Resolver. While MassDNS exclusively relies on third-party recursive resolvers to perform DNS queries, ZDNS also has a built-in iterative resolver. It makes use of the aforementioned library for its operation. According to the developers of ZDNS, an advantage of this resolver is its ability to circumvent potential rate limits by public resolvers when querying with high concurrency. This resolver also performs local recursion, which exposes internal DNS procedures to the user, enabling them to conduct more detailed research. Moreover, the resolver includes a selective response cache. It minimizes the cache entry size by only buffering Name Server (`NS`) and glue records. By leveraging this technique, it prevents excessive disposal of cache entries while reducing the number of needed queries and helping the resolver with subsequent recursion. [3]

Still, when comparing the performance of ZDNS' iterative resolver to public resolvers like Cloudflare, it can be seen that public resolvers are better suited for large-scale lookups of billions of domains. This is due to bigger caches, and thus better resolution accuracy with bigger scale [3]. Therefore, for large-scale lookups, both tools depend on external recursive resolvers to reach their full potential.

ZDNS' Lookup Modules. ZDNS encompasses several *lookup modules*, namely `mxlookup` and `alookup` [21]. These modules can automatically perform additional queries when processing Mail Exchange and Canonical Name records in addition to looking up the records specified in the original query. The `mxlookup` module is able to automatically perform additional `A` record lookups when querying for `MX` records, while the `alookup` module can interpret `CNAME` records and query their contents.

MassDNS' Python Scripts. MassDNS has several distinctive features as well. A couple of Python scripts expand the use cases of MassDNS, particularly with regard to reconnaissance scanning. For instance, a script for automatically resolving previously queried PTR records named `ptr.py` enables the user to efficiently perform Reverse DNS [24].

In addition, the script `subbrute.py` allows for brute-force subdomain enumeration with MassDNS [5]. It works similarly to SubBrute, a high-performance DNS query spider, which can perform DNS record and subdomain enumeration [25]. MassDNS includes several more subdomain enumeration scripts, e.g., the script `ct.py` which uses the crt.sh Identity Search to scrape for Certificate Identity logs and extract subdomains from them [26]. These scripts present an advantage of MassDNS, as their functionality in enumerating subdomains is useful for reconnaissance scans and penetration tests, which are performed for detecting potential security risks or vulnerabilities of domains.

3.2. Usability Comparison

Both DNS frameworks provide the user with various CLI options for setting basic DNS lookup parameters, like the timeout time and query retry count.

ZDNS additionally has the option to use the built-in iterative resolver via setting the `--iterative` flag, but also an option to enable UDP socket recycling (see Section 3.3), as well as options for choosing specific transport layer protocols or *lookup modules*. [21]

In contrast, most notably, MassDNS has an option to unset the DNS Recursion Desired (RD) bit via setting the `--norecurse` flag [5]. This option enables the user to perform non-recursive lookups, and thus makes it easier to carry out DNS cache snooping [27] or subdomain enumeration (see Section 3.1).

Output. Further considering the output, there are several similarities and differences in its format and verbosity. MassDNS incorporates a total of 5 different output formats: domain list output, simple text output, full text output, binary output, and Newline Delimited JSON (NDJSON) output [5], [28]. The simple text output can be controlled by a total of 10 advanced options for high customization [5]. It is important to note that out of these output options, only the binary output conserves the whole DNS response data consisting of all received records [5]. All other formats only preserve the record types mentioned in Section 3.1. Most notable, however, is the JSON output format. It outputs all response packets as JSON objects in the output style of `dig`, which are easily programmatically interpretable, thus enabling researchers to parse their results efficiently.

ZDNS, on the other hand, only features output in JSON, whereby different output verbosities exist: `short`, `normal`, `long`, and `trace`. These verbosities can be further customized through the `--include-fields` option. [21]

The ZDNS' `trace` feature is particularly interesting, as it can display the whole lookup chain/trace of every request when performing internal recursion. It lists the responses of all name servers of the trace, starting from the root name server [3]. The `dig` tool has a similar feature, which can be used via the `+trace` argument [17]. MassDNS, on the other hand, lacks an analogous feature.

In conclusion, although MassDNS supports more output options than ZDNS, the most significant output variant for further processing, namely the JSON output method, is supported by both tools. However, in ZDNS, this output format is customizable, whereas in MassDNS, it is not.

3.3. Implementation Differences

The performance and reliability of each tool highly rely on its implementation. For this reason, key implementation aspects of both tools are compared below.

ZDNS and MassDNS both rely on different forms of concurrency when executing queries. ZDNS uses Goroutines, i.e., lightweight threads, for looking up names in parallel. Each Goroutine is assigned a UDP socket, which sends and receives DNS packets. By default, ZDNS reuses its UDP sockets, meaning they are used repeatedly for different lookups until program termination. This improves ZDNS performance and scalability, as the creation of new sockets for each new query is circumvented. [3]

MassDNS, on the other hand, does not use threads in its C implementation. It instead utilizes the `epoll` I/O event notification facility together with a self-written hashmap implementation to carry out concurrent queries [29], [5]. MassDNS creates one UDP socket per `--bind-to` entry and then assigns all sockets to the `epoll` instance [5]. The hashmap respectively stores the domain name queries that are supposed to be processed next, so that the `epoll` instance can take the next batch of domain names and use an available socket to execute the lookups. Additionally, MassDNS can be run utilizing multiple processes. However, it then operates using a shared-nothing architecture [5], meaning that queries are not synced between processes, and the outputs of each process are stored in different files.

As mentioned above, both tools utilize UDP sockets for their lookups, but ZDNS also allows for the use of TCP sockets. A disadvantage of using the latter is that it negatively affects performance, as TCP sockets do not allow for reusability. [21]

3.4. Fault Mitigation Approaches

ZDNS and MassDNS share one big issue; they both can overwhelm DNS servers and experience rate limiting by public resolvers when run with high concurrency. To resolve this issue, both tools provide an option to use multiple resolvers for querying.

ZDNS offers this feature via its `--name-servers` option [21]. However, as described in Section 3.1, the latter issue can be avoided entirely by using ZNDS' built-in resolver.

MassDNS offers a similar feature to use multiple resolvers via its `--resolvers` option [5]. Besides that, for dodging IPv6 resolver rate-limiting, MassDNS provides the `--rand-src-ipv6 <your_ipv6_prefix>` option [5]. This option allows MassDNS to pick from a specified subnet of source IPv6 addresses for each lookup, thus effectively evading rate limits by some IPv6 resolvers. Other than that, there is no other option to minimize the load on DNS servers when using MassDNS except to manually decrease the hashmap size or lower the retry count from the default 50 [5].

4. Performance Evaluation

The performance of each tool in successfully resolving domain names is crucial for assessing the capabilities of both frameworks. It is evaluated by comparing each tool's resolution accuracy and fallibility. Furthermore, the Top Level Domains (TLDs) of the domain names resolved exclusively by one of the tools, but not the other, are compared to demonstrate either tool's effectiveness when querying for domain names with specific TLDs.

4.1. Resolution Accuracy Comparison

The resolution accuracy of ZDNS and MassDNS was compared by querying for `A` and `AAAA` records of a custom dataset containing all domain names from the Cloudflare (CF) Radar Top 1 Million (1 009 126 domain names on 11-30-2023) [30] and the Chrome User Experience Report (CrUX) (995 207 distinct domain names on 11-30-2023) [31]. These domain lists were chosen because they are both publicly available and contain the most visited, i.e., most resolved domains of the Internet.

Both tools were tested once using the Unbound recursive resolver, while ZDNS was also tested using the CF public resolver. The CF resolver was additionally chosen for ZDNS, as, according to the ZDNS developers' scans, ZDNS overall works best when paired with CF [3]. The built-in ZDNS iterative resolver was not tested, as MassDNS does not feature a native resolver, thus rendering a comparison inequitable.

Inherently, MassDNS uses 10 000 concurrent lookups. However, due to ZDNS only using 1000 Goroutines by default [21], MassDNS was tested with a hashmap size of both 10 000 and 1000 to provide a more balanced comparison. To minimize resolver overload, ZDNS was tested using its default retry count of 1, while MassDNS' retry count was lowered from 50 to 3.

Success Rates. The success rates discussed below were achieved by ZDNS and MassDNS using the Unbound resolver and a concurrency level of 1000 lookups. Figure 1 depicts the encountered error codes while querying for `A` records using the aforenamed parameters.

First, considering the CF Radar query results, it became apparent that, regardless of whether `A` or `AAAA` records were queried, MassDNS had a slightly higher resolution success rate than ZDNS. For instance, MassDNS successfully resolved 96.54% of `A` record queries, whereas ZDNS resolved 96.51% of queries with success. In the CrUX scan results, while ZDNS excelled in resolution accuracy this time, the success rates of both tools were also quite similar, but they both encountered a higher

 doi: 10.2313/NET-2024-04-1_10

number of errors. In the case of A records being queried, MassDNS had a slightly lower success rate of 95.39%, while ZDNS marginally outperformed MassDNS with a success rate of 95.49%.

In conclusion, the results show that, under equivalent circumstances, both tools have a similar accuracy. Which tool gets a slight advantage depends on the queried domain list. However, as seen in Section 4.2, there is a major disparity in accuracy when tweaking the tools' parameters.

Error Comparison. Examining the status/error codes in Figure 1 reveals some more interesting aspects. When scanning the CrUX list, both tools encountered about twice as many SERVFAILS and NXDOMAIN errors as they did while querying the CF Radar list.

Concerning TIMEOUT encounters, each time, MassDNS experienced more timeouts than ZDNS. For instance, while resolving the CF Radar A records, MassDNS experienced 988 more timeouts than ZDNS. The timeout difference is even more significant when looking at the CrUX A record lookup results, where it amounts to 1420. Considering that MassDNS only timed out 4973 times in this scan, this represents a timeout increase of about 28.55% when compared to ZDNS.

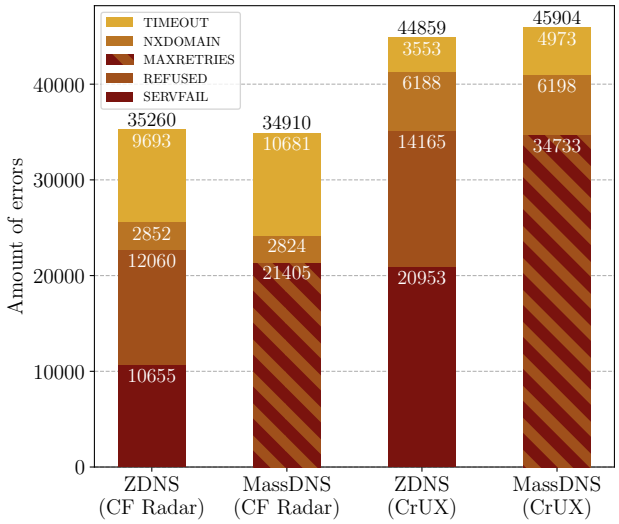

Figure 1: Comparison of A record query errors. The MassDNS specific MAXRETRIES status code encompasses SERVFAIL and REFUSED errors.

Resolved Domain Name Comparison. The domain names that both tools resolved successfully were mostly similar. For both domain lists, the similarity percentage when querying A records amounted to about 99.76%.

However, when analyzing the domain names each tool resolved exclusively, some noteworthy aspects became visible. Table 2 depicts the TLDs where ZDNS and MassDNS exhibited the most notable disparities in successful resolutions. The table shows that ZDNS performed slightly better when resolving .com domain names. The tool resolved 853 domains which MassDNS could not successfully query. In fact, ZDNS excelled when resolving domain names for all TLDs listed in Table 2 except when querying .ru domains. In this case, MassDNS outperformed ZDNS by uniquely resolving an additional 569 domain names.

TABLE 2: Number of TLDs either tool resolved exclusively across all A record queries.

TLD	ZDNS	MassDNS
.com	853	352
.ru	223	792
.jp	589	153
.net	199	61
.info	130	55
.club	128	39
Total	2622	1929

4.2. Additional Findings

ZDNS paired with CF. Scanning the aforementioned domain lists (see Section 4.1) with ZDNS in combination with the CF public resolver revealed that ZDNS performs more effectively with it than with the Unbound resolver. For instance, the success rate rose from 96.54% to a near-perfect 99.26% when querying the CF Radar list for A records. This result underlines that ZDNS is more performant when paired with Cloudflare [3].

Timeouts in MassDNS. Running the previously presented scans with MassDNS using its default hashmap size of 10K revealed an interesting aspect regarding the times MassDNS experienced timeouts. For example, when scanning the CF Radar list for A records, MassDNS encountered 72 148 timeouts when using 10K concurrent lookups, which are 61 467 more timeouts than in the scan presented in Figure 1. In other words, MassDNS encountered around six times as many timeouts when performing 10K concurrent queries compared to only 1000 parallel lookups. A similar outcome was observed when scanning the CrUX list.

This behavior implies that MassDNS overwhelms name servers when querying with its default concurrency, which leads to timeouts that stem from rate limits. This notably adversely affects MassDNS' success rate in every scan. For instance, when scanning the CF Radar for A records, the success rate decreased from 96.51% to 90.66%. According to Durumeric et al., MassDNS' success rate drops even further to about 65% when using 45K concurrent lookups [3].

5. Conclusion and Outlook

ZDNS and MassDNS are both capable tools for pure large-scale domain name resolution.

ZDNS is highly preferable when reliability and extensibility are the priority. Its main advantage is that it offers a well-rounded, feature-rich package. Different properties, such as support for numerous record types, advanced DNS lookup modules, and a built-in resolver for, e.g., analyzing lookup chains, make it a versatile instrument suitable for intricate DNS analysis.

MassDNS is a more fitting choice if runtime takes priority over reliability. Its exceptionally high default concurrency rate renders it highly suitable for fast bulk queries. However, MassDNS' extensive parallelism comes at the expense of reduced reliability as the timeout rate increases. Nevertheless, MassDNS offers a variety of use cases, especially in the fields of reconnaissance scanning,

penetration testing, and Reverse DNS. The Python scripts that MassDNS features are highly beneficial for applications in these areas.

Both tools plan to implement new features and combat their shortcomings in the future. While ZDNS plans to extend its functionality by adding support for DNS over HTTPS and DNS over TLS [3], MassDNS aims to implement more adaptable concurrency mechanisms to prevent overwhelming resolvers, as well as more reconnaissance features, e.g., wildcard record detection [5]. These improvements will make both tools even more suitable for their respective use cases.

References

[1] DNIB, "The domain name industry brief quarterly report," https://dnib.com/articles/the-domain-name-industry-brief-q3-2023, Tech. Rep., 2023, [Online; accessed 1-March-2024].

[2] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras, "A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 6, pp. 1877–1888, 2016.

[3] L. Izhikevich, G. Akiwate, B. Berger, S. Drakontaidis, A. Ascheman, P. Pearce, D. Adrian, and Z. Durumeric, "ZDNS: A Fast DNS Toolkit for Internet Measurement," in *Proceedings of the 22nd ACM Internet Measurement Conference*, ser. IMC '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 33–43. [Online]. Available: https://doi.org/10.1145/3517745.3561434

[4] The ZMap Team, "The ZMap Project," https://zmap.io/, 2024, [Online; accessed 1-March-2024].

[5] B. Blechschmidt and Q. Scheitle, *MassDNS*, https://github.com/blechschmidt/massdns, [Online; accessed 1-March-2024].

[6] S. Meng, L. Liu, and V. Soundararajan, "Tide: Achieving Self-Scaling in Virtualized Datacenter Management Middleware," in *Proceedings of the 11th International Middleware Conference Industrial Track*, ser. Middleware Industrial Track '10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 17–22. [Online]. Available: https://doi.org/10.1145/1891719.1891722

[7] OpenINTEL, "Openintel," https://openintel.nl/, 2024, [Online; accessed 1-March-2024].

[8] SIDN Labs, "About SIDN Labs," https://www.sidnlabs.nl/en/about-sidnlabs, 2024, [Online; accessed 1-March-2024].

[9] SURF, "About SURF," https://www.surf.nl/en/about, 2024, [Online; accessed 1-March-2024].

[10] NLnet Labs, "About," https://nlnetlabs.nl/about/, 2024, [Online; accessed 1-March-2024].

[11] R. Sommese, M. Jonker, J. van der Ham, and G. C. M. Moura, "Assessing e-Government DNS Resilience," in *2022 18th International Conference on Network and Service Management (CNSM)*, 2022, pp. 118–126.

[12] R. Sommese, G. C. M. Moura, M. Jonker, R. van Rijswijk-Deij, A. Dainotti, K. C. Claffy, and A. Sperotto, "When Parents and Children Disagree: Diving into DNS Delegation Inconsistency," in *Passive and Active Measurement*, A. Sperotto, A. Dainotti, and B. Stiller, Eds. Cham: Springer International Publishing, 2020, pp. 175–189.

[13] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A Search Engine Backed by Internet-Wide Scanning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 542–553. [Online]. Available: https://doi.org/10.1145/2810103.2813703

[14] J. Zirngibl, P. Sattler, and G. Carle, "A First Look at SVCB and HTTPS DNS Resource Records in the Wild," in *2023 IEEE European Symposium on Security and Privacy Workshops*, Jul. 2023.

[15] L. Degani, F. Bergadano, S. Mirheidari, F. Martinelli, and B. Crispo, "Generative adversarial networks for subdomain enumeration," 04 2022, pp. 1636–1645.

[16] J. Mao, M. Rabinovich, and K. Schomp, "Assessing Support for DNS-over-TCP in the Wild," in *Passive and Active Measurement: 23rd International Conference, PAM 2022, Virtual Event, March 28-30, 2022, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2022, pp. 487–517. [Online]. Available: https://doi.org/10.1007/978-3-030-98785-5_22

[17] Internet Systems Consortium, Inc., *dig - Linux man page*, https://linux.die.net/man/1/dig, [Online; accessed 1-March-2024].

[18] A. Cherenson, *nslookup - Linux man page*, https://linux.die.net/man/1/nslookup, [Online; accessed 1-March-2024].

[19] NLnetLabs, *Unbound*, https://github.com/NLnetLabs/unbound, [Online; accessed 1-March-2024].

[20] Cloudflare, "What is 1.1.1.1?" https://www.cloudflare.com/learning/dns/what-is-1.1.1.1/, 2024, [Online; accessed 1-March-2024].

[21] Z. Durumeric, P. Pearce, D. Adrian, S. Kalscheuer, and B. VanderSloot, *ZDNS*, https://github.com/zmap/zdns, [Online; accessed 1-March-2024].

[22] B. Blechschmidt and Q. Scheitle, *Alternative (more granular) approach to a DNS library (fork)*, https://github.com/zmap/dns, [Online; accessed 1-March-2024].

[23] M. Gieben, *Alternative (more granular) approach to a DNS library*, https://github.com/miekg/dns, [Online; accessed 1-March-2024].

[24] Cloudflare, "What is reverse DNS?" https://www.cloudflare.com/learning/dns/glossary/reverse-dns/, 2024, [Online; accessed 1-March-2024].

[25] The Rook, *subdomain-bruteforcer (SubBrute)*, https://github.com/TheRook/subbrute, [Online; accessed 1-March-2024].

[26] R. Stradling, "crt.sh Certificate Search," https://crt.sh/, [Online; accessed 1-March-2024].

[27] O. Farnan, J. Wright, and A. Darer, "Analysing Censorship Circumvention with VPNs Via DNS Cache Snooping," in *2019 IEEE Security and Privacy Workshops (SPW)*, 2019, pp. 205–211.

[28] T. Hoeger, C. Dew, F. Pauls, and J. Wilson, *NDJSON - Newline delimited JSON*, https://github.com/ndjson/ndjson-spec, [Online; accessed 1-March-2024].

[29] *epoll - Linux man page*, https://linux.die.net/man/7/epoll, [Online; accessed 1-March-2024].

[30] Cloudflare Radar, "Top 1000000 Domains," https://radar.cloudflare.com/domains, 2023, [Online; accessed 30-November-2023].

[31] Google, "Overview of CrUX," https://developer.chrome.com/docs/crux/, 2023, [Online; accessed 30-November-2023].