A Comparative Analysis of Network Data Distribution in Active DNS Measurements. ZDNS vs OpenINTEL.

THALIS STAVROPOULOS, University of Twente, The Netherlands

The Domain Name System (DNS) plays a critical role in the proper functionality of the Internet. Subsequently, a number of tools have been created to take measurements that can be used to understand better how it operates in practice. One aspect of the design of such tools is the query distribution among nameservers. While the designers of the tools do take into consideration and discuss this, there seems to be a lack of in-depth analysis of how the queries are distributed. In this paper, we make a comparison of two active DNS measuring tools, OpenINTEL and ZDNS, and how their recursive resolvers distribute queries. Despite their different objectives and design choices we try to make a fair comparison. We captured enough network traffic and analysed it. By scanning a million domain names we were able to capture enough network traffic to analyse. Based on the data collected we set a few key metrics to examine the extent to which the distribution of queries is fair, meaning more even. This study concludes that OpenINTEL has an overall more even distribution of queries which is indicated by the majority of metrics.

Additional Key Words and Phrases: Internet, DNS, active measurements

1 INTRODUCTION

The Domain Name System (DNS) is a crucial component of the Internet which translates domain names into Internet Protocol (IP) addresses. As the number of devices on the Internet increases over the years, reaching over a billion, so does the importance of a well-functioning DNS infrastructure [3]. Essentially, every device connected to the Internet becomes unreachable if there is a DNS outage since it will be practically impossible to determine a host's IP address.

For said reason, DNS can be a big target for malicious actors that might attempt to attack part of the infrastructure, examples of such attacks are DDoS, Fraud, Worms, Botnets, and Spam [11]. In fact, in January 2019 the Cybersecurity and Infrastructure Security Agency and the Department of Homeland Security issued a warning about a series of incidents in which attackers were able to tamper with the DNS infrastructure and redirect internet traffic to malicious websites [6].

To combat such malicious activity it is possible to find patterns in the activity of malicious actors by analysing DNS data. There have been a number of studies that, using DNS datasets, develop algorithms and Artificial Intelligence (AI) to detect and prevent attackers before they can take action [5] [10] [12] [15].

For the data to realistically reflect on how the DNS infrastructure operates, a high volume of it is needed. Such a volume of data requires the collaboration of experts and organizations since taking on such tasks can prove to be challenging regarding both the collection and storage of the data. A number of tools have been developed that are specifically designed to scan the DNS infrastructure on such a scale. OpenINTEL and ZDNS are two of these tools which will also be the focal point of this paper.

Typically, when a comparison between two software tools is made the main attribute evaluated is performance. However, the focus of this paper will be a comparison of the two in terms of query/data distribution. ZDNS and OpenINTEL were designed to scan a large part of the Internet. Therefore, we investigate how these queries are distributed among the various nameservers. To achieve this, active measurements will be taken using said tools and analysis will be performed to determine how evenly the data is distributed.

2 BACKGROUND

2.1 Terminology

Before continuing we establish a few keywords that will be used throughout the paper. The fairness of the distribution is meant to represent how evenly the data and queries are spread. The more "fair" is considered, the more even the distribution will be.

Authoritative name servers are servers which have the authority to serve certain information. This information can be the IP you are looking for or other information needed to find that server. In this paper, we refer to them as nameservers.

Later on, in the results, we split the data between Top Level Domains (TLDs) and domain name nameservers. The TLD nameservers are the nameservers directly below the root servers. Such nameservers are for the .com, .nl, .org and other TLDs. All other nameservers that are below the TLDs are referred to as domain name servers.

When taking measurements of the DNS there are two methods that can be used, these can be either passive or active.

Passive measurements involve observing the traffic at some point between the client and the nameservers. This can be at the nameserver, the recursive resolver or the client itself.

Active measurements on the other hand entail actively sending the queries and storing the results. In our case both ZDNS and OpenINTEL take active DNS measurements.

2.2 Motivation

Measuring the DNS infrastructure can prove to be an effective way to better understand it and improve its security. However, this has to be done responsibly since the tools developed can also be used for malicious practices.

In their paper Kountouras et al., creators of Active DNS Project, gathered both passive and active DNS measurements in their university's network [5]. They noted that with the active DNS measurements, they got a lot more addresses, an order of magnitude larger and argued that active DNS measurements are more in-depth. This is to be expected since passive DNS measurements are highly affected by the cyber-behaviour of their users. Thus, addresses that are not popular among the users of the network will not appear in

TScIT 38, Feb 3, 2023, Enschede, The Netherlands

the capture. On the other hand, less popular addresses and addresses that are not active yet are being captured by active measurements. This kind of measurement can be particularly useful in finding ways to detect domain names that are created by malicious actors that have not taken action yet.

Other than the security aspect, the DNS measurements can be used to further improve the functionality of the DNS infrastructure. By looking at the data we can identify which parts can be improved or are not operating as planned, either because of malicious intent or unintentionally, because of a lack of technical skills to solve the problem. Weaver et al. in their paper in 2001 took active measurements of the DNS infrastructure network and found problems, such as result wildcarding where recursive resolvers alter the responses that returned an error to direct users to third-party websites [14]. The paper concluded with a number of recommendations for DNS implementers and DNS API users.

This paper explores how ZDNS and OpenINTEL, two active DNS measurement tools, handle the scheduling of queries. Although their goals might differ to some degree, they both were designed to scan a very large number of domain names. It is therefore important to study their behaviour and the impact they can have on the DNS infrastructure.

2.3 Tools

Before defining the goals of this study it is important to provide some information on the two tools this paper will be based on.

2.3.1 OpenINTEL.

OpenINTEL is a project that started in 2015 [13]. At the start, the goal of the project was to collect DNS data of the top 3 largest TLDs and store them. The project has been going on until this day, actively collecting DNS measurements every 24 hours. There have been numerous additions to the list of TLDs covered. The data is stored and can be requested by researchers who need to use it to better understand DNS.

The authors of the paper set a number of goals which affected their decision on the architecture of the system. The system must be able to take measurements of every domain specified every day consistently. The reason is that for the data to be meaningful there should be no gaps between measurements. Furthermore, the system is designed with scalability in mind. The addition of more TLDs should also be feasible without major work required. They achieve this by having a cluster manager per TLD which divides the domain names into chunks and adds them to a pool in which workers can get a chunk and process it. Afterwards, the result will be returned to the cluster manager which will add some more data and store it. This approach allows for scalability. If a new TLD is to be added, a new cluster manager can be set up and a number of workers will be assigned to it. Additionally, if the number of domains under a TLD is increased, new workers can be assigned to the appropriate cluster manager. Most importantly, they address the challenge of query pacing and discuss how there was a trade-off between speed and the impact on the DNS infrastructure.

One thing to consider is that OpenINTEL uses Unbound[1], an open source resolver which acts as a local recursive resolver. This intends to take the burden off the recursive resolvers that would perform such a task.

[1]https://github.com/NLnetLabs/unbound

Finally, they address the issue of sending an excessive number of queries to the same nameservers. Large-scale providers will most likely hold records for a large number of domain names. Consequently, these servers will receive more queries than others. In their analysis regarding the workload the scanning poses on the nameserver, they concluded that it is not excessive. It is, however, a significant amount and they warn that it could pose a problem if other researchers start running similar measurements.

2.3.2 ZDNS.

ZDNS[2] is a command-line tool developed to perform DNS lookups [4]. The goal was to create a high-performance tool that is also easily extensible. The creators have evaluated the performance of ZDNS to be greater than other similar tools. In their paper, Izhikevich et al mention that they have implemented an internal recursive resolver that researchers could use. The decision was made so as to not burden ISP-based and public recursive resolvers.

The creators of ZDNS warn users that it should be used with caution. By default, ZDNS creates 1000 threads, and they recommend 1000-5000 threads for most systems, which might cause some DNS providers to be overwhelmed. For that reason, it is possible to configure ZDNS to use fewer threads. Nevertheless, they suggest first communicating with network administrators before running any large-scale scans.

ZDNS was designed to be publicly available since most other tools are closed-sourced and results in researchers having to implement their own tools from scratch[8]. Additionally, the authors of the paper mention that Unbound has over 30 features requested but have not been resolved for years. Therefore, ZDNS was designed as a base framework that researchers could use to develop tools that satisfy their needs.

3 RELATED WORK

There have been a number of papers that make use of ZDNS and OpenINTEL worth mentioning.

In 2017 Paul et al. created a system called Iris that focuses on the detection of DNS manipulation. The system makes use of ZDNS to perform PTR lookups of all the open DNS resolvers that were found using ZMAP [1]. Another example of the use of ZDNS is the paper by Dimova et al [2]. In their paper, they discuss anti-tracking evasion techniques used to bypass anti-tracking measures by making use of CNAME records. The measurements for retrieving CNAME and A records were done using ZDNS.

Ramin et al. made use of the active DNS measurements of OpenINTEL to detect suspicious domains that use Unicode homoglyph characters. The Internationalized Domain Name (IDN) allows the use of Unicode characters in the domain name. This can be exploited to trick people into visiting third-party websites by replacing characters in the domain name with ones that look identical. Sommese et al. have used the active DNS measurements provided by OpenINTEL to investigate the impact of DDoS attacks on the DNS infrastructure. Their findings show that although millions of domain names

[2]https://github.com/zmap/zdns

were the target of a DDoS attack, the DNS infrastructure was not impacted by it [10].

Both tools have contributed to the research of the DNS ecosystem.

4 RESEARCH GOALS

The goal of this research paper is to study how fair ZDNS and OpenINTEL are regarding the distribution of queries.

ZDNS and OpenINTEL are both active DNS measurement tools. However, they were designed with different objectives in mind. Thus, it will be interesting to investigate how fair is the distribution of queries.

A few Research Questions are defined to guide this research.

Research question 1: How much data/queries were sent and to how many nameservers?

Looking at the total data and queries sent to the nameservers will give us a good first impression of the load that was sent to the nameservers in total. It will also enable us to see any potential abnormalities in the amount of data sent. The number of nameservers could also be used as a first impression of the distribution of queries.

Research question 2: How evenly are the queries distributed among nameservers?

To answer this question a few metrics will be used to make a comparison between the two tools. Thereafter, we will be able to determine which of the two tools is overall fairer.

Research question 3: Which nameservers received the highest number of queries/data and how much did they receive?

After looking at the overall distribution, we will further investigate the group of nameservers that received the most data. The goal is to check whether there are a handful of nameservers that receive the majority of traffic. We want to further examine what the distribution between these nameservers is and which of the two tools is fairer.

5 METHODOLOGY

For the collection of data, the same method was used for both tools. The goal was to get the A records of the domain names. A sample of 1 million domain names was used as input. This number of domain names would be high enough for the results to reflect on how the scheduling is done but low enough to not have a significant impact on the DNS infrastructure. The list of the 1 million domain names was obtained by Tranco[3], a research-oriented domain name ranking list hardened against manipulation [7]. The list came as a csv file with each entry being the ranking and the domain name. For ZDNS the `--alexa` option was specified which indicates that the input format is an Alexa Top Million list. Moreover, the list was first randomized before being used as input. Because the top domain names were mostly .com, and then others followed, the list was first randomized to even out the frequency at which domain names in the .com zone were queried.

For a number of reasons, it was decided to run ZDNS on a more powerful machine than OpenINTEL (see appendix B). Firstly, because one of the goals of OpenINTEL is to not have a big impact on the DNS infrastructure, it was designed to send queries at a slow pace. Consequently, it does not require a powerful machine. On the other hand, ZDNS was designed to run on a powerful machine that could make use of its processing power for better performance. Even so, the machine used was not as powerful as the one Izhikevich et al used in their paper [4]. This might have been the reason for an issue that will be shortly mentioned. The only hint on how to choose the right configuration was that in the README file of ZDNS they suggest using 1000 to 5000 go threads[4]. It was therefore decided to keep the default configuration, see appendix B. It is of high importance to mention that OpenINTEL and ZDNS do not have the same concept of caching. Thus, matching the cache size would not necessarily provide a more fair comparison. The most important aspect which could affect the performance was that both can reuse sockets and are able to make use of large number of ports.

Both scans were scheduled for 01:00 UTC time(02:00 local time). Users are not very active during that time, thus making sure the measurements do not have a significant impact on the infrastructure.

While the scheduled measurements were ongoing the network traffic on port 53 was captured, with 53 being the port used for DNS resolution. Once the measurement was finished the information needed was filtered. We are only interested in the outgoing traffic since we want to see the data sent by the tools and do not have control over the data received. In addition, the NS and A records were filtered. Occasionally, we would observe other queries being sent such as DNSSEC queries, which acted as noise. Although it did not add up to a significant amount of data it was decided to filter them out.

Once the data was collected and filtered the analysis was performed to extract useful information regarding the distribution of data to the nameservers. Specifically, each unique nameserver was mapped to the number of queries and the amount of data it was sent to it.

Afterwards, when it was required the root zone file was used to make the distinction between TLD nameservers and non-TLD nameservers[5].

5.1 Challenges faced

While running ZDNS scan the following challenge was faced. When running the measurement for 1 million domain names about 5% of the domain names resulted in `ITERATIVE_TIMEOUT` error. This information was extracted from the metadata of the ZDNS measurement results. This would not be the case if the input was not as large, at about 10 thousands domain names. There was an attempt to lower the error rate by trying to change the configuration of ZDNS. However, the result was a considerably slower performance without consistently lowering the error rate. Thus, it was decided to use the previous configuration and accept the error rate. We speculate that the error might be the cause of the machine not being as powerful as required.

The first measurement taken with OpenINTEL revealed that the recursive resolver would favour nameservers with IPv6 addresses. This was an issue because ZDNS exclusively sends queries to nameservers with IPv4. While it seemed that OpenINTEL had a more even distribution, it could be the case that some of the pairs of IPv4 and

[3]https://tranco-list.eu/

[4]https://github.com/zmap/zdns/

[5]https://www.internic.net/domain/root.zone

IPv6 addresses could correspond to the same machine, also called "siblings". There are a number of techniques to identify IP sibling relationships [9], but implementing them would introduce some unwanted uncertainty. For that reason, it was decided to disable querying IPv6 nameservers for OpenINTEL.

6 RESULTS

While both the number of queries and data were analysed, it was decided to present only the figures for the number of queries. This decision was made to improve the readability of the paper. Fig. [1] and Fig.[2] represent the number of queries and the amount of data sent respectively. In appearance, the two graphs have no significant difference. The only difference is the order of magnitude. For the data, the order of magnitude is 10^6 whereas for the queries it is 10^5. This was the case for all graphs concerning the relation between data and queries for both OpenINTEL and ZDNS.

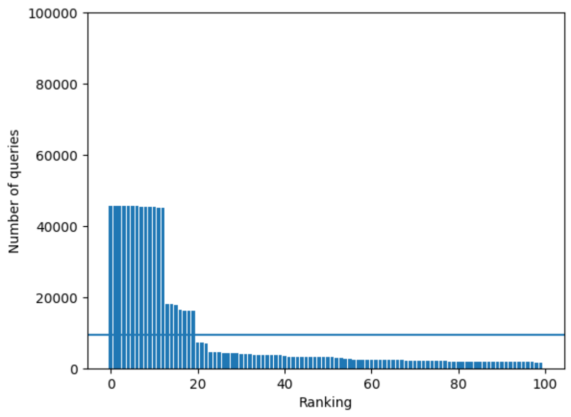

Fig. 1. OpenINTEL Query Distribution

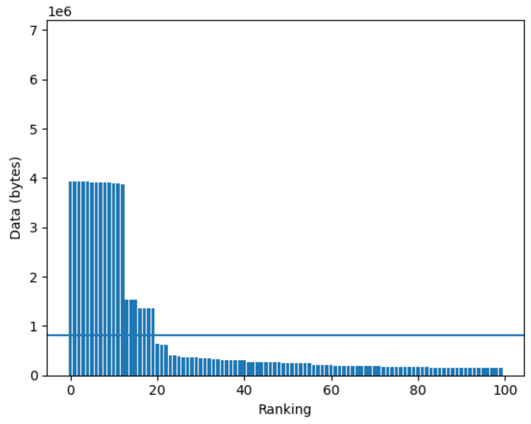

Fig. 2. OpenINTEL Data Distribution

6.1 Overall Data and nameservers

ZDNS and OpenINTEL sent comparatively the same amount of data, with OpenINTEL sending 5% more than ZDNS[1]. It is interesting to point out that despite OpenINTEL sending more data it sends fewer packets overall, making the mean length of packets for OpenINTEL 86 bytes and 75 bytes for ZDNS. Thus, answering Research Question 1.

	total nameservers	total data sent	total queries sent
ZDNS	141,446	213.05MB	2.85M
OpenINTEL	163,494	224.02MB	2.62M

Table 1. Data table

The excess data is due to the optimization settings of OpenINTEL, sending some additional information with every query which results in sending more data in total. However, it needs to be confirmed that this is indeed the reason for the excess data, perhaps in future work. Despite that, it has an insignificant impact on the results since the result for queries and data are almost identical showing.

In total, ZDNS sent queries to 141k unique nameservers and OpenINTEL to 163k. Since the domain names that were used as input to the two tools were not equally distributed to different TLDs it would make sense that some received more traffic than others. For instance, around 50% of the domain names were part of the .com TLD.

Further investigating the nameservers of the .com TLD showed that they received the majority of queries out of all the others. Looking at the root zone file revealed that the nameservers for .com are also responsible for the .net TLD. Thus it was not possible to distinguish between the two by only looking at the destination IP. For OpenINTEL the .com/.net nameservers received 22.7% of data and 22.6% of queries. For ZDNS the .com/.net nameservers received 23.6% of the data and 23.4% of queries. This further justifies that there were no anomalies.

Distinguishing the .com/.net nameservers in the top 100 (Fig.[3] and Fig.[4]) distribution makes it clear that these nameservers received most of the traffic. Furthermore, isolating the .com/.net nameservers also makes it apparent that OpenINTEL has a more even distribution, sending nearly the same number of queries to all 13 nameservers [5]. ZDNS on the other hand looks to prefer a specific nameserver over the other 12.

With this, we have a first view of the data. OpenINTEL and ZDNS sent comparatively the same amount of data with no anomalies detected. OpenINTEL also sent data to more nameservers in total which could hint to OpenINTEL being fairer.

Coming back to Research Question 1, we see how much data and the number queries sent overall by OpenINTEL and ZDNS comparatively the same with OpenINTEL sending a bit more. However, looking at the destination of the data, we notice that OpenINTEL sends to more unique nameservers in total. The results are summarized in table [1].

6.2 TLD and domain name nameservers

It was interesting to investigate the difference in data between the domain name servers and the TLD nameservers since .com/.net proved to have received about a quarter of all data and queries sent.

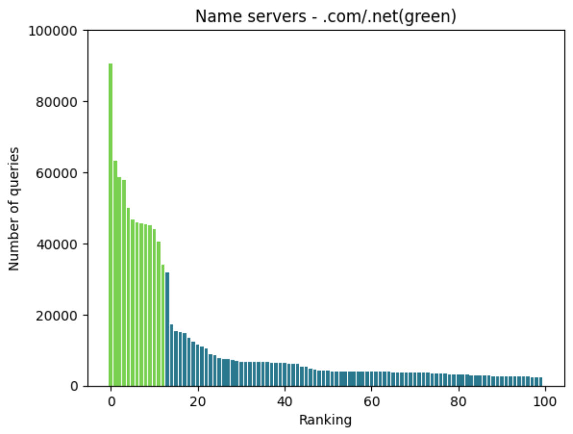

Fig. 3. ZDNS Query Distribution with .com/.net nameservers marked

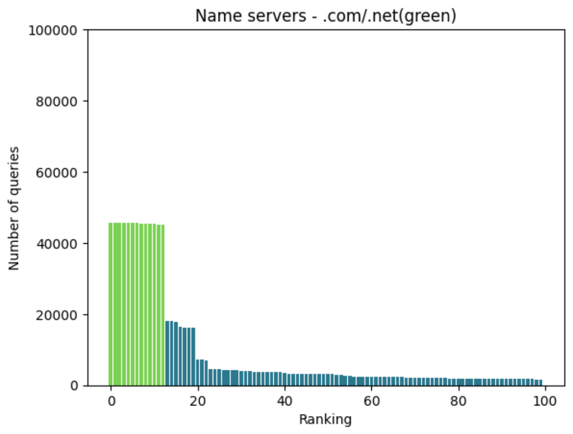

Fig. 4. OpenINTEL Query Distribution with .com/.net nameservers marked

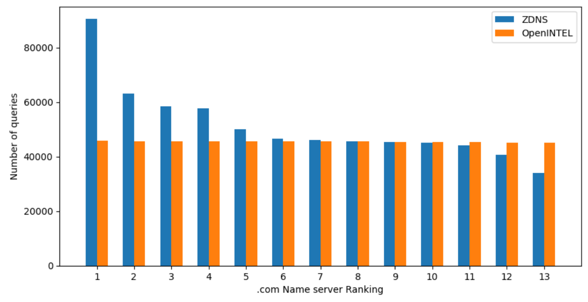

Fig. 5. OpenINTEL Query Distribution with .com/.net nameservers marked

Overall, ZDNS sent queries to 141k nameservers of which 2,779 were TLD nameservers. Similarly, OpenINTEL sent queries to a total of 163,494 unique nameservers of which 3,086 were TLD nameservers. Although the TLD nameservers amounted to only 1.96% they were responsible for about 48% of the total data sent to nameservers. This was the case for both OpenINTEL and ZDNS without any significant difference. Looking at the number of unique TLD and domain name nameservers ranked by the amount of data sent also confirms this trend (see appendix A).

Further looking into the TLD nameservers that received most data we find that .com/.net nameserver received the majority. For OpenINTEL the .com/.net nameservers received 22.6% of the queries and for ZDNS then 23.4%.

Looking at Research Question 2 we can say the the TLD nameservers received more data than the domain name servers. Furthermore, the .com/.net TLD nameservers received the majority out of all other nameservers.

6.3 Cumulative distribution function and Gini coefficient

Another aspect to look at is the cumulative distribution function of the queries sent over the number of nameservers. Moreover, it is possible to determine the percentage of nameservers that contribute to the majority of the queries sent. Fig.[6] shows the cumulative distribution for OpenINTEL and ZDNS. A point of interest would be the 90% point, where the number of nameservers that received 90% of the total queries is pointed out. That 90% is distributed to 27k for OpenINTEL and 10k for ZDNS.

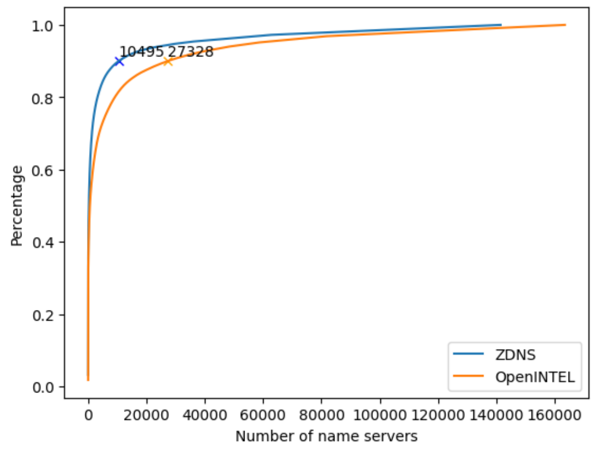

Fig. 6. OpenINTEL and ZDNS Cumulative distribution function

	top 13	top 100	90% of data
ZDNS	0.002	0.574	0.799
OpenINTEL	0.128	0.627	0.787

Table 2. Gini Coefficient

An additional way to calculate the difference in distribution between OpenINTEL and ZDNS is the Gini coefficient. The Gini coefficient measures the fairness in distribution mostly used to measure the fairness of wealth distribution in countries. In this paper it will be used to calculate the inequality between the queries and data sent to the nameservers. The lower the value the fairest the distribution.

A coefficient of 0 indicates a perfectly even distribution whereas a coefficient of 1 total inequality.

As seen in the cumulative distribution function the upper 10% of the queries are spread over a large number of nameservers. For that reason, the Gini coefficient was calculated for the top nameservers that received 90% of the data and queries. The coefficient was also calculated for the top 13 nameservers and the top 100 [2].

For the nameservers that received 90% of the queries both ZDNS and OpenINTEL have a large coefficient. This can be justified by the uneven number of domain names used as input. The TLD of the domain names that were queried were unequal, with 50% of all domain names being in the .com TLD. Consequently, it is not possible to have a perfectly even distribution between the nameservers.

For the top 13 nameservers, which are all .com/.net nameservers, OpenINTEL has a significantly more even distribution with a Gini coefficient of 0.002 which is very close to 0. ZDNS also has a relatively low coefficient of 0.128.

For the top 100 ZDNS appears to have a more even distribution with a coefficient of 0.574 against OpenINTEL with 0.627.

Lastly, the numerical finding of this study, namely the Gini coefficient and the cumulative function, allow us to answer Research Question 3.

7 DISCUSSION

Most metrics indicate that OpenINTEL is overall fairer than ZDNS.

Both the Gini coefficient and the Cumulative distribution function show us that OpenINTEL distributes the queries more evenly. The cumulative distribution in Fig.[6] shows how the distribution for OpenINTEL is more smoothed out with a less sharp corner than ZDNS. Furthermore, OpenINTEL sends 90% of its queries to more than double the number of nameservers that ZDNS does, which indicates that the distribution of queries is fairer.

It becomes even more clear when looking at the query distribution for the top 13 nameservers. These nameservers are all of the .com/.net TLD. Fig. [5] shows OpenINTEL having a more even distribution, sending almost the same number of queries to all the nameservers. ZDNS seems to prefer one of the servers more than the others, with the nameserver ranked 13th having been sent less than half the queries the nameserver ranked 1st was sent.

Unlike the other metrics, the Gini coefficient for the top 100, ZDNS appears to have a more even distribution with a coefficient of 0.574 against OpenINTEL with 0.627. The reason is that OpenINTEL has a few big jumps in queries sent because of the difference in TLD. As previously mentioned .com/.net nameservers receive the most queries compared to all other nameservers. Therefore, the 14th nameserver which is not .com/.net receives considerably less data. Because OpenINTEL has a more even distribution between nameservers of the same TLD the difference between the 13th and 14th nameserver is greater for OpenINTEL than ZDNS. This is in turn interpreted as a less even distribution by the Gini coefficient. Perhaps it would be more appropriate to group the nameservers and calculate the Gini coefficient among the same TLD nameservers, but due to the time limit, it was not possible.

This is not to say that ZDNS does not handle the distribution well. The primary goal of ZDNS is to allow researchers to have a fast tool that they can easily extend to fit their needs. In that case, it might be more beneficial to sacrifice some fairness for performance.

As the OpenINTEL authors explained in their paper there was a trade-off between performance and the pacing of queries. Having to run the scan every day could pose a significant workload to the nameservers. Therefore, performance was sacrificed to have less of an impact on the DNS infrastructure.

7.1 Limitations

Although the results showed that OpenINTEL has a more even distribution there are a few things to consider. This is important to think of since there could have been some pitfalls that affected the result.

Firstly and most importantly, the data that was gathered was the result of a single scan. It is therefore possible that the data gathered in the study does not reflect how the tools would operate and more samples are required.

As discussed, a number of queries resulted in an error is unclear how this could have affected the results. Furthermore, the hardware limitation along with the configuration of both tools could have also affected the overall result. That is to say that a combination of better hardware and different configuration settings could change the results of the fairness of ZDNS. The error codes we received could have been the result of not having enough power from the hardware. To overcome this a more powerful machine would be required that matched the specs of the machine Izhikevich et al. used in their paper.

7.2 Future work

For future work, it would be appropriate to take more measurements using the tools to confirm our findings. As mentioned the data used was collected from running only a single scan session. Lastly, it would be interesting to investigate the distribution between nameservers that belong to the same TLD. The top 13 nameserver which were all of the .com/.net TLD clearly showed that OpenINTEL had a more fair distribution.

8 CONCLUSION

Looking at the results and the metrics used in this study, it is fair to say OpenINTEL has a more fair distribution than ZDNS. This is evident looking at the distribution of the top 13 nameservers, which are of the same TLD. It clearly indicating OpenINTEL having a more even distribution with each nameserver receiving almost the same number of queries. However, it is important to keep in mind that they have different objectives which have influenced their design choices. Further studies will have to be carried out to confirm the results.

REFERENCES

[1] [n. d.]. Global Measurement of DNS Manipulation | USENIX. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/pearce

[2] Yana Dimova, Gunes Acar, Lukasz Olejnik, Wouter Joosen, and Tom Van Goethem. [n. d.]. The CNAME of the Game: Large-scale Analysis of DNS-based Tracking Evasion. ([n. d.]), 2021. https://sub.example.com/

[3] ICANN Security and Stability Advisory Committe. 2019. SAC105 The DNS and the Internet of Things: Opportunities, Risks, and Challenges. (2019).

[4] Liz Izhikevich, Gautam Akiwate, Briana Berger, Spencer Drakontaidis, Anna Ascheman, Paul Pearce, David Adrian, and Zakir Durumeric. [n. d.]. ZDNS: A

Fast DNS Toolkit for Internet Measurement. ([n. d.]), 11. https://doi.org/10.1145/3517745.3561434

[5] Athanasios Kountouras, Panagiotis Kintis, Chaz Lever, Yizheng Chen, Yacin Nadji, David Dagon, Manos Antonakakis, and Rodney Joffe. 2016. Enabling network security through active DNS datasets. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9854 LNCS (2016), 188–208. https://doi.org/10.1007/978-3-319-45719-2{_}9

[6] Christopher C Krebs and Russell T Vought. 2019. CISA CYBER+INFRASTRUCTIJRE. (2019). https://www.us-cert.gov/ncas/current-activity/201

[7] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczy, Wouter Joosen, and Ku Leuven. [n. d.]. TRANCO: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. ([n. d.]). https://doi.org/10.14722/ndss.2019.23386

[8] Jiarun Mao, Michael Rabinovich, and Kyle Schomp. 2022. Assessing Support for DNS-over-TCP in the Wild. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 13210 LNCS (2022), 487–517. https://doi.org/10.1007/978-3-030-98785-5{_}22/TABLES/4

[9] Quirin Scheitle, Oliver Gasser, Minoo Rouhi, and Georg Carle. 2017. *Large-Scale Classification of IPv6-IPv4 Siblings with Variable Clock Skew.* Technical Report.

[10] Raffaele Sommese, K. C. Claffy, Roland van Rijswijk-Deij, Arnab Chattopadhyay, Alberto Dainotti, Anna Sperotto, and Mattijs Jonker. 2022. Investigating the impact of DDoS attacks on DNS infrastructure. *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC* (10 2022), 51–64. https://doi.org/10.1145/3517745.3561458

[11] Olivier Van Der Toorn, Moritz Müller, Sara Dickinson, Cristian Hesselman, Anna Sperotto, and Roland Van Rijswijk-Deij. 2022. Addressing the challenges of modern DNS a comprehensive tutorial. *Computer Science Review* 45 (2022), 100469. https://doi.org/10.1016/j.cosrev.2022.100469

[12] Olivier van der Toorn and Anna Sperotto. 2018. Threat Identification Using Active DNS Measurements. *12th International Conference on Autonomous Infrastructure, Management and Security, AIMS 2018 - Proceedings* (2018), 1–5.

[13] Roland Van Rijswijk-Deij, Mattijs Jonker, Anna Sperotto, and Aiko Pras. 2016. A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements. (2016). https://doi.org/10.1109/JSAC.2016.2558918

[14] Nicholas Weaver, Christian Kreibich, Boris Nechaev, and Vern Paxson. [n. d.]. Implications of Netalyzr's DNS Measurements. ([n. d.]). http://netalyzr.icsi.berkeley.edu.

[15] Ramin Yazdani, Olivier Van Der Toorn, and Anna Sperotto. 2020. A Case of Identity: Detection of Suspicious IDN Homograph Domains Using Active DNS Measurements; A Case of Identity: Detection of Suspicious IDN Homograph Domains Using Active DNS Measurements. (2020). https://doi.org/10.1109/EuroSPW51379.2020.00082

Appendix A

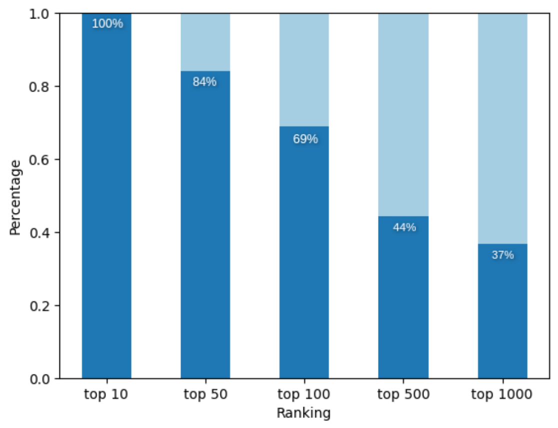

Fig. 7. ZDNS TLD(dark blue) - other domain name(light blue) nameservers ratio

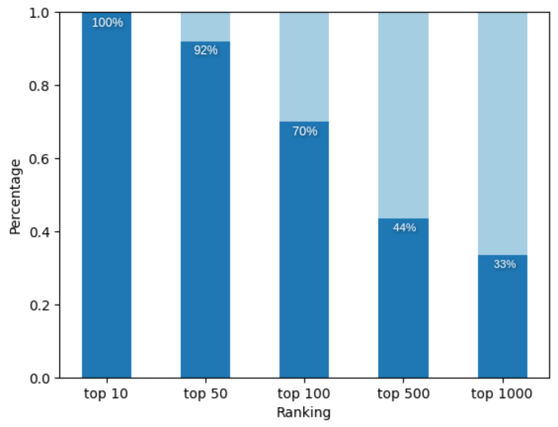

Fig. 8. OpenINTEL TLD(dark blue) - other domain name(light blue) nameservers ratio

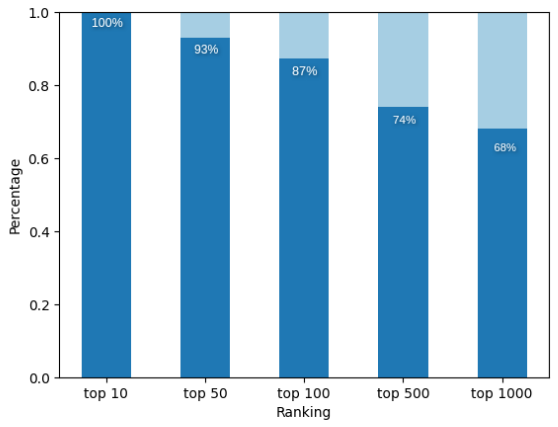

Fig. 9. ZDNS TLD(dark blue) - other domain name(light blue) data ratio

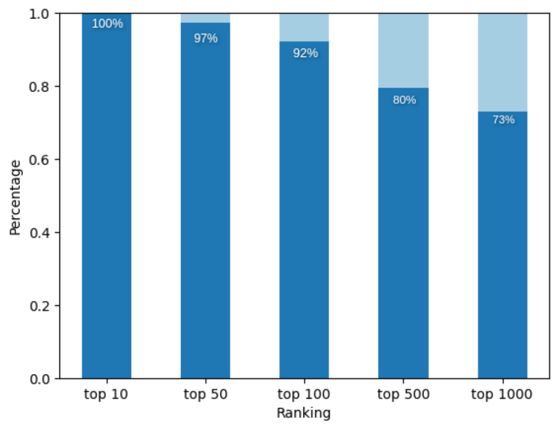

Fig. 10. OpenINTEL TLD(dark blue) - other domain name(light blue) data ratio

Appendix B

Option	Selected	Description
cache-size	10000 (default)	how many items can be stored in internal recursive cache
go-processes	8 (default GOMAXPROCS)	number of OS processes with GOMAXPROCS being all available cores
iteration-timeout	4 (default)	timeout for resolving a single iteration in an iterative query
iterative	true	Perform own iteration instead of relying on recursive resolver
max-depth	10 (default)	how deep should we recurse when performing iterative lookups
mx-cache-size	1000 (default)	number of records to store in MX -> A/AAAA cache
recycle-sockets	true (default)	Create long-lived unbound UDP socket for each thread at launch and reuse for all (UDP) queries
retries	1 (default)	how many times should zdns retry query if timeout or temporary failure
threads	1000 (default)	number of lightweight go threads
timeout	15 (default)	timeout for resolving an individual name

Table 3. ZDNS configuration settings

	OS	RAM	CPU	# of cores
ZDNS	Ubuntu 20.04	16GB	2.4GHz	8
OpenINTEL	Ubuntu 18.04	2GB	2.3GHz	1

Table 4. Hardware specs