Identifying DNS Scanners from a TLD Perspective

Pascal Huppert
462 248

October 21, 2024

Research performed at SIDN Labs

Thesis presented for the degree of
Master of Science

Contents

Following the publication of this thesis, an article providing a shorter summary of this work will be published on sidnlabs.nl.

Acknowledgments

I have been privileged to be allowed to analyze traffic reaching the .nl authoritative name servers. Working with this data provides best circumstances for this research, as TLD name servers are arguably the best point of view to spot scanning campaigns.

I would like to thank everybody at SIDN for being so welcoming to yet another German (who could not even speak Dutch at the start of the project). Moritz Müller and Thymen Wabeke, who supervised my research at SIDN, supported me greatly even next to their own work and growing family. They listened to every idea and new insight I had and patiently helped me find answer to all of my questions and issues I could not solve by myself.

From the start, the whole organization was open to supporting my research and welcoming in taking me in and offering a pleasant working environment. Many others showed interest in my topic of research and were helping me to connect with the topic and understand all relevant bits and pieces of the Dutch DNS infrastructure. Often, I could tap into the practical knowledge of others about the topic, who have been working on the DNS for many years, some perhaps longer than I have been on this earth. Moreover, my colleagues at SIDN Labs showed my what a great experience research can be if performed in exchange with engaging people.

1 Introduction

With the immense growth of internet traffic, gathering data about it has become increasingly worthwhile for companies and academics alike. One valuable kind of data concerns domain names, administrated in a decentralized manner in the domain name system (DNS). It is already known that scans (for example active DNS measurements) are being performed for research [34][16][17] and commercial purposes alike, whether that is checking any known name or checking a number of names repeatedly to monitor proper functioning and uptime of a service.

Reasons for Scanning Huge scanning operations are often used to gather data on the structure of the internet at large. Name servers, being the phone books of the internet, are a particularly interesting target of scans and data scraping as they hold information about public web pages. Campaigns that try to extract information in bulk from DNS servers can have different goals, such as:

1. Gathering data for research [34],
2. finding out which domain names are available to be acquired in order to make a profit from buying and selling domains,
3. finding out about different services and vulnerabilities running in a domain,

and more.

Excess Traffic Just as spotted repeatedly in root servers before [8], undesired traffic might put an extra burden on TLD name servers as well. Many TLD operators, like SIDN, do not publish the full content of their zone file, making them private to some degree, and motivating scans. Even though DNS servers, as for the .nl TLD, are well

online.nl	inc.nl	services.nl	service.nl	international.nl	info.nl
shop.nl	consulting.nl	tech.nl	web.nl	net.nl	blog.nl
group.nl	usa.nl	china.nl	store.nl	media.nl	france.nl
design.nl	solutions.nl	art.nl	law.nl	music.nl	

Table 1.1: First names from one campaign querying most popular words from a list, querying more than 300 names per millisecond. These names were all queried within a single millisecond.

equipped to handle today's requests, unwarranted traffic can result in more energy consumption, higher latency, and expensive upgrades that would otherwise not have been necessary.

Thus, the impact of scans on DNS zones ought to be known.

While the software projects ZDNS [16] and MassDNS [5] have been shared openly, other projects, such as OpenINTEL [34], voice concerns about the impact of releasing their software to the public [34, p. 1886], due to concerns about burdening the name servers.

Furthermore, finding out about scanning operations could yield valuable insights: Analysis of requests to DNS servers has already been the topic of multiple research papers [9][8][29][34], dating back to at least 2008 [9].

Detection These previous analyses of DNS query datasets, however, have looked at the root zone DNS servers instead of top level domains [8][9][29]. For DNS scanning operations, these root servers are not of particular interest because their very limited number of records can easily be cached and only needs to be queried once, even if resolving multiple names. 3LDs and deeper are presumably less often targeted by scans. Thus, collecting data from TLD name servers is most suitable to see the requests that occur during a scanning operation.

This work takes a position on the receiving side of such operations to see whether scans can be identified from the incoming requests to an authoritative name server, thus using passive DNS measurement. The Stichting Internet Domeinregistratie Nederland (SIDN) is the organization managing the names under the .nl zone, including the DNS infrastructure and more. The Dutch TLD contains around 6 million domain names, which makes it one of the largest Country-Code Top Level Domain (ccTLD)s in the world.[1] SIDN is a foundation managing the Dutch ccTLD on all fronts, including registration, name server operation, fraud detection and continual research and development, offering other services as well. It is a medium-sized organization with around 100 employees, situated in Arnhem, the Netherlands. This work uses data from the .nl TLD name servers of SIDN.

Research Questions This research aims to determine whether coherent scanning operations can be identified from the request data using data from TLD name servers. The leading research questions are:

1. How much traffic from scans can be found in the traffic to SIDN's authoritative name servers?
2. Which features do these scans exhibit, and which features are relevant for identifying them?

[1] `https://dnib.com/articles/the-domain-name-industry-brief-q4-2023`

3. What is the impact of this traffic on the availability of SIDN's DNS servers?

Structure of this Thesis This work analyzes traffic of the .nl name servers and finds that such scans can be found in the data and originate from various sources. After explaining the background of the topic and providing an overview of the related work, the main part of this document features two chapters about conceptually distinct steps in the research:

- Chapter 2 will explain details about the DNS relevant to this work.
- Chapter 3 will go into detail about previous research on the topics of scannign, DNS abuse, DNS measurement, and resolver classification.
- Chapter 4 describes the two means used for the analysis:
  - Section 4.1 will explain in which ways analysis and patterns can help find scans and which values are relevant for identification and classification. It contains many examples of resolver behavior.
  - Section 4.2 will then apply those measures in an approach using feature engineering and clustering to gain a broader view and more detailed description of scanning behavior. It contains large-scale analyses of resolver behavior and statistics about the results, concerning the total traffic.
- Section 4.1.6 will lay out the results, including scans found, statistics about scan types and groups, and descriptions of their behavior.
- Lastly, Chapter 6 will summarize the results, lay out implications, and provide starting points for further research.

The contributions of this work are:

1. Providing an overview of the scans and automated traffic to SIDN's servers that can be found in one day's traffic.
2. Characterizing features of scans.
   - Determining features of the scans that occur, such as whether they are asking for existing or non-existent names, predominantly distinct or repeated names and scanning constantly or at regular intervals.
   - Provide a feature set that can be used to find outliers in resolvers querying authoritative name servers, especially those showing signs of scanning behavior.
3. Analyzing the potential impact of large scanning operations and making suggestions for how to mitigate it.

2 Background

This section will introduce all necessary concepts, so that even a reader unfamiliar with internet technicalities can understand the rest of this document. If familiar with the topics covered, any parts of these can be skipped. After a brief explanation of relevant networking details (ASes and IP), it will introduce all relevant details of the DNS and this specific point of research, namely DNS, scanning, SIDN's authoritative name servers and motivational and ethical considerations of the topic.

2.1 Internet: Structure

The internet is a decentralized network. An interested user might wonder, though, what exactly this means. After all, a household usually establishes connection to the internet via an Internet Service Provider (ISP), hierarchically organized. And presumably, there could exist something similar connecting multiple ISPs. While this is partially the case with Upstream Providers and IXPs, such a system would be less efficient and fault tolerant, as much traffic would have to travel through the top of the hierarchy.

Instead, the internet is made up of smaller networks that can work independently, with outages of one network not affecting the availability of another, in the best case. These internet components are called ASes, and usually owned and maintained by an organization. Routing of internet traffic takes, simply speaking, the shortest available path through other ASes. ASes are registered and each is assigned a unique number.

Users do not often see AS numbers, but may be more familiar with IP addresses. IP addresses are also numbers, but they describe single hosts. These must also be globally unique and are thus given out be the network authorities in the form of prefixes containing a number of IP addresses equal to a power of two. Each IP address belongs to one AS. In practice, this affiliation is established by an AS broadcasting a message signaling that a specific IP prefix can be reached by routing traffic to this AS.

Obtaining an IP prefix is similar to buying a stretch of land: It gives you the right to use it within fixed boundaries. There are two versions of the Internet Protocol, IPv4 and IPv6. The main difference is that IPv6 addresses are longer (128 bits vs. 32 bits), and thus abundantly available, whereas global IPv4 addresses are a rather valuable commodity. This is why devices on a local network might have their own, globally unique IPv6 address, but usually share a common IPv4 address.

For this research, the differences between IPv4 and IPv6 are not of particular importance. Both have addresses made from bits, and these are conceptually ordered from most to least important: The prefix, i.e. the first bits of an IP address, determine what organization's computer or network is being contacted, while the later bits might distinguish between machines of a network or even constitute different addresses belonging to the same machine. For different technical purposes, there are also a few IP prefixes reserved for special use cases. They cannot be registered and many of them are not valid for use on the internet.

2.2 DNS: Basics

The Domain Name System (DNS) is the part of the internet whose main purpose is the translation of domain names such as `uni-muenster.de` into Internet Protocol (IP) addresses [39, p. 1]. The former are human-readable, while the latter are necessary for communication between machines. Using these, the target server can then be contacted. In this, each valid host name is associated with one or more IP addresses.

Like the internet, DNS is a distributed system in more than one way. Most importantly, names are organized in a hierarchical order. The names resemble this order: `www.sidn.nl` is located inside of what is called a zone, together with `impact.sidn.nl` and all other valid names that end in `.sidn.nl`. This particular zone belongs to the SIDN, and together with `sidnlabs.nl`, `google.nl` and `example.nl`, it is located inside the `nl`-zone. Lastly, the `nl`-zone is, together with other zones such as `com`, `de` and `net`, located inside the root zone.

In this way, each zone except for the root has a parent, and for each zone there is a zone file that contains data about all child zones. Each zone is owned by a person or organization: SIDN manages which names are contained in the `nl`-zone, Google manages which names are contained in `google.nl` and where the name itself points to and so on. Thus, the hierarchical structure also resembles delegation over the authority of each zone.

Then, each zone also needs a so-called *authoritative name server* making information about the zone publicly available. The root zone has 13 authoritative name servers, and there are currently around 1500 Top-Level Domain (TLD)s, many of the most popular belonging to countries, such as `nl`, `ru`, `de`, `cn` and `tk`.

Each part of a domain name is called a *label*. Labels can be any string of arbitrary length, but are governed by rules which exist both for technical reasons and introduction through registries. Generally, most special characters are not allowed in labels. For the NL-zone, only letters and numbers are allowed, and hyphen can be placed between two symbols that are not hyphens.

Each label defines a zone that is nested in the parent zone. Parent zones then contain and delegate the child zones, which are linked to the parent zone in this way. Each

zone has an authoritative name server that knows every name and datum within that zone. All information within a zone is stored in the form of *records*, including text, child zones, IP addresses and cryptographic keys. What a record can contain is largely determined by the existing record types, although use of the TXT record type allows storing arbitrary data for any purpose. When querying information about a DNS zone, the record type is provided to determine which information is sought after.

A combination of multiple labels with dots in between, accurately describing a certain host, is usually called a Fully-Qualified Domain Name (FQDN). FQDNs are what users on the internet usually work with in the form of websites or mail servers, and they contain 2 or more labels, with some exceptions such as `localhost`. On the global internet, ICANN prohibits TLDs from containing A or AAAA records, and most software will not recognize domain names containing only one label. SIDN also disallows the most common 3LD names to be used as 2LDs such as `www.nl` or `mail.nl`.

In order to resolve a name, a client queries a DNS server, typically asking to find out which IP address to contact about a given domain name. The server can then respond with the requested data or signal the name does not exist, or that another error occurred, each with a different *response code*.

Resolving a FQDN requires multiple steps, since it contains more than one label. The servers authoritative for the domains have to be queried in order, and each one asked about their child domain. This is a tedious process and can be optimized. Instead of each client performing the whole iteration to find records for a domain name, clients normally send their requests to a proxy, which performs the work on behalf of them and returns the result records. Any actor resolving a domain name is called a *resolver*, and such ones resolving full domain names through an iterative process are called *recursive resolvers*. Recursive resolvers are necessary for the DNS to work, and they also perform caching to improve performance and decrease the overall traffic volume.

The topmost domain of the hierarchy is the *root zone*, which is the parent of all TLD and available around the world, having 13 different IP addresses. For a resolver to be able to make any contact to a name server, it needs to know one of these root server IP addresses.

2.3 DNS: Details

Similar to how every valid host name needs to resolve to an IP address, RFC 1912 also suggests that each host should have an associated name [4, Section 2.1], which is not considered best practice anymore.

The delegation in the DNS follows the same hierarchical structure as the zones: An owner of a domain can delegate child domains to other people or organizations by including the correct records in the zone file, or manage these zones themselves. The authoritative servers for the child zones can either be separate or the same machine, that is one

computer can be authoritative for an arbitrary number of domains, including parent and child zones or completely unrelated zones. It is possible that one server has a large number of zone files, and authority is given to it only by including it in the DNS hierarchy, in a record of the parent zone. Although it is not intended, rogue servers can answer queries about any names they are not authorized for, and this cannot be prevented without further security measures. Since recursive resolvers only query name servers that the specific name is delegated to, this is not an issue.

The recursive resolver performing the lookup on behalf of a client can in practice run 1. on-device, 2. on a local router for clients within the network or 3. on completely different infrastructure for use by any device on the internet [39, p. 16]. Using external resolvers (the second or third kind) is standard practice [39, p. 16] and can save energy and computational power on client devices. The local piece of software that only sends are request to another machine to perform the iterative lookup is called a *stub resolver* in this case [39]. For additional benefits, external recursive resolvers can also use caching to increase performance, responsiveness and availability for their users.

Some recursive resolvers are available to the broad public and can be used freely. These are called *public resolvers*, often operated by large companies such as Google or Cloudflare. Such resolvers offer their capacity for free, and can help in collecting data about website popularity, for example. Choosing a resolver is of course the choice of each user, although the default settings are rarely changed in practice.

This separation of user devices and recursive resolvers means that authoritative name servers such as the ones from SIDN are not generally queried directly by clients, even though there is little technical hurdle to perform the lookups locally. Also, RFCs mandate that servers are not meant to be both authoritative and recursive at the same time. This is because authoritative name servers are critical infrastructure without which a DNS zone would become unavailable, whereas recursive resolvers offer their computing or network capacity. Mixing both would pose a risk to the availability of the authoritative name servers, so these two concerns are supposed to be kept separate.

As a side note, a user (device) querying a recursive resolver and that recursive resolver then querying an authoritative name server are two conceptually very different actions that might as well use different protocols. Indeed, the DNS protocol distinguishes between them only through two bit-flags: *recursion desired* and *recursion available*, which sometimes may make the roles of the three involved parties unclear. When contacting a recursive resolver, the client sets the Recursion Desired (RD) bit and receives an answer with the Recursion Available (RA) bit set. Querying an authoritative name server with the RD bit set is invalid, even though the implementation may ignore it.

More measures exist to ensure availability of authoritative name servers. Like most large services, they feature redundancy: A zone can have multiple authoritative name servers with different IP addresses or names by featuring multiple Name Server (NS) records. Additionally, IP Anycast allows a single IP address to be shared by different sites. This is no rocket science, it only means that packets sent to the address in question are sent

to the closest instance, utilizing normal pathfinding. Should a site become completely unavailable, traffic is routed to a different site. For .nl, more than 50 sites exist for 3 different IP addresses / host names. This further increases reliability and latency of the DNS.

Recursive resolver optimize performance by caching records, and the longer a record can be cached, the better for performance and efficiency. On the other hand, zone files are not static, but may change at any moment. To enable this without deteriorating performance, DNS records have a Time to Live (TTL). Authoritative name servers may change their records, and the TTL determines how long old records may be used before the authoritative servers must be queried again by the resolvers, like a kind of shelf life. Resolvers may implement their own policies on how long to keep records before querying again, but this time may not exceed the TTL. This means that a caching resolver may not cache records for longer than the TTL allows, but there is no lower boundary.

DNS builds on top of other protocols, most importantly IP and User Datagram Protocol (UDP)/Transmission Control Protocol (TCP). As the transition from IPv4 to IPv6 is still going on, DNS (and SIDN's servers) support both versions, i.e. each server has both an IPv4 and an IPv6 address, 6 addresses in total. Regarding the communication protocol, DNS (including all major servers) support both UDP and TCP, that is connection-less and connection-based communication. Using UDP is recommended, since it lacks the overhead that TCP requires for establishing a connection. Because name resolution has to finish before a connection to a target can be established, it directly and inevitably adds to the latency a user experiences. Because of this, usual practice is to perform lookups via UDP. Should a packet be too large to transmit via a single datagram, the DNS server signals to the client to send the same request again via TCP (using the *truncation* bit (`tc`)). The transmitted UDP response is incomplete in this case.

Default UDP packet size for DNS is 512 Bytes, but EDNS, a series of protocol extensions for DNS, provides a mechanism to increase this. The client (resolver) can signal to the contacted name server that larger UDP packets can be sent by specifying a maximum length. This is relevant for modern extensions that add signing to the DNS [39, p. 21]

Use of UDP has opened the DNS to different kinds of vulnerabilities. Because it is connectionless, UDP in principle allows spoofing of the source IP address. The DNS server then sends responses unsolicited to another host, enabling DDoS reflection and amplification. To mitigate this, some servers do not answer large numbers of queries from the same source via UDP (for this reason and others). The `tc` bit can be used in this case as well, signaling that the client - should the queries be legitimate - should switch to TCP.

Response packets can be spoofed similarly. By sending a spoofed response to a DNS query, resolvers can be slipped an incorrect DNS record, potentially *poisoning* the cache. A first line of defense against this attack is the randomization of UDP ports and query IDs. The query ID is a number included in each request by the client that allows them

to match the received answer packets, and should be chosen randomly. These measures make cache poisoning more difficult, but do not completely prevent it (for more information, see `https://www.cloudflare.com/learning/dns/dns-cache-poisoning/`). Privacy is also a concern, as UDP and TCP are unencrypted protocols.

These vulnerabilities also explain why unprotected open recursive resolvers are considered a security risk, and open resolvers need to be well-maintained. In 2008, one study [10] already found around 17 million open DNS resolvers.

Additional security measures such as DNSSEC, DNS over TLS and DNS over HTTPS are meant to protect against the aforementioned attacks and others and improve privacy, but have been slow in adoption. Privacy was not a concern when the DNS was first developed, and the data contained in the DNS is of course regarded as public. But today, it is seen that privacy is one of the biggest shortcomings of DNS [39, p. 16]. Queries can contain sensitive personal data, and even zone files can contain personal names and similar data in the form of domain names. More details about the privacy side of the DNS can be found in [39]. Relevant for this research is DNS Security Extensions (DNSSEC), which cryptographically proves the authenticity of the chain of delegation for a domain name, simply by adding certificates to DNS transactions to prevent spoofing. In practice, this means there are resolvers that evaluate those signatures, and others that do not; and a few record types exist specifically for DNSSEC. One important part are Next Secure (NSEC) records, which provide authenticated proof of the non-existence of a domain name. An NSEC record certifies that no domain name is in existence between two other domain names that are adjacent alphabetically within the zone. In the simplest form, any NX domain response would include an NSEC record with the (lexicographically) next and previous *existing* domain names, which enables zone walking by querying these domain names in order. Next Secure 3 (NSEC3), a newer version, uses hashing of the domain names to mitigate zone walking, although it may be feasible to crawl all hashes and then crack them on a client machine. More about NSEC and NSEC3 can be found in [39, p. 24ff.].

The DNS is also often used for censorship and blocking of unwanted content [39, p. 16], both politically and against malicious or unsafe content. In this case, the service in question is not made completely unavailable, but name resolution is prevented, like when removing a person from a phone book.

Privacy was not a concern when the DNS was first developed, because the contained data is of course regarded public, but now it is seen that user privacy is one of the big shortcomings of the DNS [39, p. 16]. Another best practice called *qname minimization* [6][7] advises recursive resolvers to reduce the information contained in queries during iterative resolving as much as possible, which means not including 3LD labels in queries to TLD name servers and using a generic query type for all but the last query . The query type can be NS (originally dictated), A/AAAA (current advice) or any other type whose authority is delegated [7, p. 2.1]. While [7, p. 2.2] defines query name minimization to reduce the number of labels included in the request to the fewest necessary, some resolvers seem to use other means. Often, subdomains are seen to be replaced either

by underscores (`_.example.com`) or asterisks (`*.example.com`), which may be how some resolvers implement query name minimization[1], as neither symbol is valid in domain names. Relaxed implementations of query name minimization will, in case a name cannot be resolved successfully, fall back to querying with full name and original query type.[2]

Just like with these RFCs, DNS and its structure are built on rules, conventions, and conventions that have become rules. Asking a random person today, they will tell you that website addresses end in `.com` or `.nl`, but it is technically possible for a TLD to have Address (A) records, and therefore a website. A rule by Internet Company for Assigned Numbers and Names (ICANN) forbids it, as it would contradict the notion that many users and browsers have of what constitutes a domain name. That `co.uk` is a public suffix and not a website is simply trivia.

Looking at TLDs, different types exist. ccTLDs such as `.nl`, `.uk` or `.cn` and generic Top-Level Domain (gTLD)s such as `.com` and `.org` are the most popular ones. Some ccTLDs are often used for their letters instead of their affiliation (`.it`, `.tv`), and they vary wildly in reputation. Some, such as `.com` and `.tk`, are often used maliciously, whereas other registries such as SIDN put a lot of effort and research into detecting misuse of names and building a more reputable zone.

Zone files are not generally publicly available. Operators of gTLD are required by ICANN to share their zone's contents under certain circumstances, but zone files of ccTLD are often kept secret.

Non-standard Unicode symbols can be used in domain names using Punycode, which is a method of encoding those symbols within a domain name so that it contains only basic characters. This is done by the browser or client software, transparent to the user, and can be disallowed by a zone owner, often because it allows very similar looking, but distinct, names, for the confusion of the user.

For details about any part of the DNS, please see the article by van der Toorn et al. [39].

2.4 Scanning

Very commonly, scanning is done in IPv4 address space. Both malicious and good-willed actors scan for open ports and services on each address to find and exploit or notify about vulnerabilities. For IPv4, the total number of addresses is 2^{32}, which is small enough for a full scan to be performed by a single machine within an acceptable time frame, today.

[1] `https://isc.sans.edu/diary/Underscores+and+DNS+The+Privacy+Story/29002`

[2] see `https://labs.ripe.net/author/wouter_de_vries/making-the-dns-more-private-with-qname-minimisation/`

IPv6 is different, because the address space of size 2^{128} is much to vast to be scanned in full within any realistic time frame, using any arbitrary number of machines. That is to say the address space of IPv6 is many orders of magnitude larger than can be feasibly scanned in total. For this reason, any IPv6 scans must use a heuristic of which addresses to scan, for example a list of known addresses.

The DNS is similar: While we are not talking about individual hosts being scanned, for non-public zones, an authoritative server is essentially an oracle deciding, for each name, whether it exists or not. In both cases, the search space is too vast to be fully covered, and lists or other heuristics become necessary. Since names can contain many symbols, at least including letters and digits, the number of possible 10-letter domain names is already larger than $36^{10} \approx 3.7 \cdot 10^{15}$. In this way, domain names and IPv6 addresses aid in obfuscation in a way that IPv4 addresses do not. The necessity of lists counts for both DNS measurements [39, p. 14], as well as other scans, such as domaining.

For the purpose of this paper, scanning will be defined as transverse data collection, as opposed to data collection that is only longitudinal. *Transverse* in this context means the broad collection of data about the *current* state of the DNS, whereas similar phenomenons such as monitoring might collect longitudinal data by querying a small number of names very regularly. Thus, monitoring is a longitudinal method of data collection, but if it only targets a small number of names, it does not count as scanning for the purpose of this work, and it will not be the *primary concern* of this research.

Scanners (actors performing a scan) will require a name list to do so. Different methods of acquiring such a list exists, and some have been scientifically evaluated. Presumably popular ones include extracting domain names from public TLS certificate registrations using CT logs as well as using any kind of public list, for example from Common Crawl. CT logs are not directly related to the DNS ecosystem, but they contain data about TLS certificates that have been given out and therefore include domain names capable of using HTTPS. Nowadays, this is a very basic security practice.

Software for performing DNS zone scans is readily available, with two open-source projects being MassDNS and ZDNS. MassDNS comes with a list of public resolvers that can be used for enhancing performance, and has a more performance-oriented, straightforward query behavior as default settings, repeating queries with little waiting time. It support a few different record types ZDNS on the other hand supports all typical record types, and is also available for free use. The OpenINTEL project uses custom software, which is not available to the public.

2.5 Point of View

`.nl` is, of course, the ccTLD of the Netherlands, and the 5th largest ccTLD in the world. SIDN is the foundation entrusted with administrating this DNS zone since 1996, and

since then, the zone has grown to about 6 million 2LDs. According to SIDN, `.nl` is supposed to be one of the most secure and trusted TLDs.

2.6 Dataset

The dataset for this research is the passively collected data from all `.nl` authoritative name servers. It is important to mention that all Anycast sites are included, because clients can only send traffic to the nearest Anycast nodes as has been determined by the BGP routing protocol. Resolvers may choose to send their queries to any one of the three available IP addresses. Usually, this is done in similar extent to each server, or one is preferred over the others [30], which could skew the resulting data if it was not being collected from all sites.

The data can then be queried using a Spark instance on a Hadoop cluster. The data format includes all relevant fields of the DNS question and answer. Additionally, columns are added with some metadata about the source IP address: Which AS, country and organization it belongs to and whether it is a known public resolver from a large company.

In particular, the relevant fields of the database are:

- ID and timestamp
- Queried name (potentially including more than 2 labels)
- TTL of the IP packet
- IP version 4/6 and UDP/TCP
- Source IP address and port, destination IP address and port
- Bits from the query: recursion desired, recursion available, zero bit, DNSSEC checking disabled, DNSSEC ok
- Flags from the answer: authoritative answer, truncation, authenticated data
- Record counts: answer count, additional record count, name server count, question count
- Codes: operation code, question type, query class, response code
- Source metadata: country, AS number, AS organization, public resolver indication
- EDNS UDP packet size indication by the client
- Other EDNS fields
- Processing time
- Location of the server

- Request and result length

The large number of columns and sheer endless number of relationships possible within these columns will play an important role during the analysis in Chapter 4.

Both typical resolvers and the dataset as a whole show a clear distribution of queries among the hours of they day, with more queries being received around noon, and less being received at night. Most requests end successfully in response code 0 (OK), with around 10 % NX domains and a negligible amount of other response codes (including -1 for "no answer sent", mainly due to rate limiting).

In total, around 6 million 2LDs are registered currently, and on a normal day, around 5 billion queries reach the servers from more than 1 million different IP addresses. Only a small amount of those sources send a large number of queries. As can be guessed intuitively, a large portion of all queries as for a small number of most popular domain names.

Source port numbers are largely randomly chosen, with higher ports generally being used by more sources and thus seeing more traffic. IDs seem mostly randomly generated, with some exceptions, similar to source ports. As for protocols, IPv4 is still used more than IPv6, and UDP is used for almost all queries, as should be the case. Question types show a distinctive popularity distribution, as can be seen in Table 7.1, with many being in active use, but a few making up a large majority of queries.

The analysis tools developed during this research can be run on any machine with ENTRADA data in Apache Spark and a Jupyter installation. Therefore, other TLDs or DNS infrastructure providers can profit from the portability of the tools.

2.7 Infrastructure

Authoritative servers for `.nl` are available on the names `ns1/ns3/ns4.dns.nl`. There are more than 50 different Anycast sites distributed among several countries, maintained by SIDN and other organizations on behalf of SIDN, providing redundancy. Most of those servers are situated in western Europe and North America.

SIDN's zone file is not public, and NSEC3 is in use. Punycode is not allowed for 2LDs in the Dutch zone, but this rule cannot be enforced for deeper subdomains. Any queries containing Punycode for the 2LD are therefore unusual, and may be user errors in small quantities, or thoughtless scanning in larger quantities.

Once again, TLDs are the ideal vantage point for this research, because they usually constitute public suffixes, thus making bulk data about available names economically valuable. 2LDs can be registered by anyone, although the zone file is not usually public. For this reason, it is expected that many scans are performed on the nl zone, which is one of the most popular TLDs. Root-servers only hold a limited, well-known dataset on TLDs, which is easy to cache. 3LDs might also see some scans, as this work will also

uncover, but may be limited to specific sites and are of a different interest, with scans being performed for reasons such as security analysis. In general, 3LDs of non-public suffixes do not hold value in the same way as 2LDs.

2.8 Motivation

Research of this kind can help in many ways.

1. The extent and impact of DNS scanning has not been estimated before, even though it is widely known that such scans are commonplace in the ecosystem (during the establishment of ENTRADA, scans could already be noticed [14][p. 5]).
2. Characterization of scans allows to judge the resilience of SIDN's infrastructure, and draw up conclusions about current protection mechanisms such as rate limiting and automatic scaling, as well as recommendations about policies and implementations to keep any possible impact from scans small.
3. Software for DNS scanning is publicly available, and the impact of this has not been evaluated yet.
4. While SIDN does not see serious issues with the traffic volume from scans, this might be different for comparable organizations, for example companies offering open DNS resolvers. Research to this issue can shed light on the burden that open resolvers face and how they can be protected.
5. This work will also provide insight into resolver behavior, and which classes exist that can be distinguished.
6. Optimistically speaking, parts of this research could contribute to understanding the behavior of recursive resolvers better and provide tools and insights for use in future research.
7. While SIDN's infrastructure is not affected adversely by scans, this might well be the case for public resolvers, which need more capacity for each incoming request to perform recursion.

2.9 Ethics

The privacy framework regarding ENhanced Top-level domain Resilience through Advanced Data Analysis (ENTRADA) can be found described in [14].

The most relevant considerations for this kind of data analysis are the privacy of users. As described in [14], large and small resolvers are querying SIDN's name servers, with larger resolvers being less likely to reveal data about identifiable persons. Under typical

circumstances, a person's devices do not query Authoritative Name Server (AuthNS) directly, so that any traffic is mixed with that of many other users. Wherever possible, this research restricts itself to the analysis of sources (i.e. IP addresses or groups of such) that are large and cannot reasonably contain traffic from only a small number of users. Small sources are only included in such analyses that perform aggregation to a high level of abstraction, and therefore produce an output that cannot constitute personal data.

More privacy considerations regarding the processing of DNS data from authoritative servers can be found in [14].

3 Related Work

This chapter will provide an overview about the current state of the research on the topic of DNS scanning and DNS resolver classification.

The DNS has been a topic of active research. Related research falls into one of the following categories:

- Studies about *DNS traffic* ([1], [8], [9], [23], [29], [40]). These have often focused on one or more of the DNS root servers ([8], [9]), but others have already attempted classification of resolvers with different goals [1], [40], [32].
- Others have looked at methods of collecting data from the DNS, i.e. DNS measurement ([8], [12], [17]), for which different software products have been developed ([5], [16], [20], [34]). Those tools have further been compared in other works ([11], [38]).
- Comparable studies have studied scans in other domains, such as the IP address space [33].

3.1 DNS Abuse

The DNS plays an important role in many security-related problems on the internet, serving as a target or tool for many kinds of attacks [18]. A general theme is that the DNS is a critical piece of internet infrastructure that has been and is insufficiently secured against attacks, with many remedies having been added years later and not seen widespread adoption.

This makes it an attractive target for many attacks. Attacks on the DNS can target confidentiality (cache snooping, changing a user's DNS settings), integrity (cache poisoning, spoofing replies), and availability (DoS targeting DNS, DoS reflection and amplification using DNS).

One part of the ecosystem are recursive resolver, most of which should not be open to the public. However, it is known that millions are, and Kührer et al. have shown that some of them send incorrect responses [18, pp. 1f].

This work examines DNS scanning, which can be counted as a innocuous form of DNS abuse. Open recursive resolvers also play a role in enabling efficient scanning.

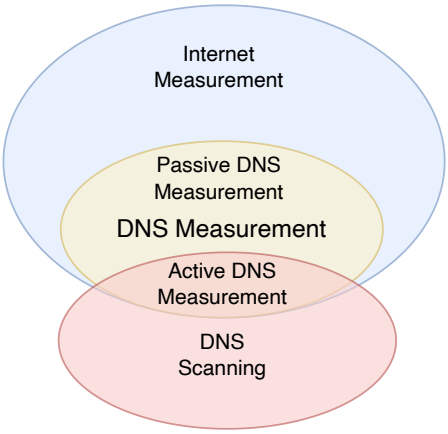


Figure 3.1: Visualization of the relationship between DNS measurement and DNS scanning

3.2 DNS Measurement

Measurements are an important tool of research in the DNS [39, p. 12]. They are performed for different use cases spanning from analysis of critical infrastructure changes [31] to that of specific use cases of the DNS (e.g. [21]). Distinction is made between active and passive DNS measurement, with active DNS measurements constituting scanning. The purpose for academic DNS scans stems from the important role that the DNS plays in the internet as a whole, but also in many illicit operations such as hiding command & control servers [17], and DNS servers also play a role in the amplification of DDoS attacks [8][29]. Previous studies on the DNS have found a lot of invalid or repeated queries, especially at the root zone. Previous studies have found a large number of bogus queries made to name servers [8], and queries from scans can also be considered to have a less justified purpose. The counterpart of bogus queries are inconsistencies in DNS data. [35] looked at inconsistencies between the NS records of parent and child zone AuthNS, sorting discrepancies into classes and providing statistics.

To facilitate research such as the aforementioned, multiple DNS measurement projects focus on automated data collection for arbitrary research purposes. Crucially, this enables longitudinal studies to also use past data. DNSDB is a passive DNS measurement project [39, p. 15], and there are several active measurement projects: OpenINTEL, Censys and even SIDN's own Dmap.

Different software has been developed for the systematic querying of large numbers of domain names. These fall into the field of active snapshot-DNS measurement, and can take both the perspective of a client (using a public recursive resolver) or query authoritative name servers directly. ZDNS [16] and MassDNS [5] have both been published around 2016, and they are freely available for everyone to use. ZDNS is part of the widely used ZMap project [13]. While ZDNS and MassDNS can take snapshot measure-

ments, OpenINTEL [34] collects longitudinal data. It also uses custom software, but has not made it public. The authors state that publishing their software could have a detrimental effect on authoritative name servers, and instead opt for sharing the results of their frequent scans with interested researchers. Also, the project tries to minimize the burden of their operations on name servers, by sending as few queries as possible and spacing them generously over each day. Two studies ([11] and [38]) have compared the tools against each other and found that MassDNS, of the three tools, generally puts the heaviest burden on the targeted name servers, sometimes diminishing the rate of received answers in the process.

[39] mentions a few active DNS measurement projects, which are a subgroup of scans. These include OpenINTEL, Active DNS Project and Netray.

MassDNS and ZDNS are mere software products, while OpenINTEL is an ongoing operation collecting data every day. It uses unbound as a local resolver, never contacting public resolvers, whereas MassDNS and ZDNS can be set to perform recursion through external resolvers, improving performance. OpenINTEL scans through all domain names once per day. About the impact of their work on the name servers, Verisign has stated that "they see the measurement and that while it is a non-trivial amount of traffic, it is not problematic" [34]

Other work has been done on the side of DNS infrastructure providers. From this vantage point, classifying and understanding traffic becomes a goal. While scans do not pose an issue for the infrastructure at SIDN, for example, the results of research into DNS traffic could allow for a better understanding of who is querying name servers and why. Towards this goal, attempts have been made to classify individual resolvers using features, both engineered [1], [32] and sometimes learned [40].

The latter project using learned features, called DNS2Vec, provided a proof-of-concept for a similar kind of classification, using only the Word2Vec model trained on sentences of resolvers that queried the same name. An explanation of the machine learning method can be found in Section 4.2.1. While DNS2Vec was shown to perform well in distinguishing public resolvers from different companies and different kinds of Google resolvers as well as classifying a resolver's country, it was not created for a specific classification purpose and will be shown to be largely unsuitable for discrimination of scans in this thesis.

Profiling Recursive Resolvers at Authoritative Name Servers Closer to this work are the ones by Açıkalın [1] and Qiao [32]. The work by M.A. Açıkalın uses a similar approach for feature engineering, but only uses rather simple features. It then explores supervised machine learning methods for classification of different predetermined resolver classes (cloud firms, hosting firms, ISPs, IT companies, research-related ASes and open resolvers) with high accuracy [1, p. 52]. For classification of scans, this nuanced distinction of different companies and their resolvers is irrelevant, and the only analysis

done in this part will concern openness of resolvers and the background/affiliation of networks which perform scans. A diverse feature set containing purpose-built calculations will be used to perform open-ended clustering to explore the dataset. [1] also provides explanations of possible causal relationships between resolver types and feature values. I attempt to provide such explanations as well, but concede that not all behaviors and abnormalities can be explained fully, such as some repetitions of queries and other patterns seen in open resolvers. Detecting abnormalities is easier than explaining them.

The work by J. Qiao also uses some engineered features for classification, but with a modest goal of distinguishing between legitimate resolvers and monitoring traffic. Furthermore, it employs 4 weeks of data while classifying source IP addresses, which is a rather large amount of data, and their filtering by sources with continual activity might exclude a large number of relevant IP addresses, e.g. because backend IP addresses of public resolvers and hosting providers might change regularly. Clustering is also used in that study to judge the usefulness of the feature set, with the ultimate goal of creating a classifier.

This work falls into the category of passive DNS measurement at the standpoint of authoritative name servers, and it fits the criteria of having many collection points (one at each Anycast site), large datasets and the use of big data analytics [39, p. 15]. It only employs (unsupervised) clustering to find patterns and groups, as well as putting a larger emphasis on engineering meaningful features. Therein, the goal is not to distinguish between resolvers and other kinds of clients, but to find and characterize scanning campaigns - whether that is IP addresses only sending scanning traffic, or public resolvers containing traces of scans. Furthermore, affiliation between different sources should be established to be able to group whole campaigns based on the patterns they exhibit, whereas classification is not the main objective of this work. The passive measurement data used is longitudinal in nature, but it goes back less than 2 years. Instead of using the full data, however, only data from two single days of traffic is processed. Data from the first day is used for analysis and manual classification of some sample sources, data from the second day is used to test the accuracy of my clustering approach on a second day using random samples. The focus is more on exploring the landscape of existing scans and traffic, rather than classification with a narrow goal, and no complex or black-box machine learning is used.

3.3 Other Scans

Scanning of the IPv4 address space occurs on a daily basis and is performed for malicious, academic and protective purposes alike. Unlike IPv4, IPv6 address space is vast and infeasible to cover within a short time. In this way, IPv6 scanning is comparable to the lookup of the domain name space: Actors need an idea about which names to query, and much of the behavior is non-canonical. Scanning of the IPv6 address space has been [33], and many of the ideas, such as aggregation by prefix and classification decisions can

be applied for DNS, similarly. One difference to IPv6 scanning is that for DNS, different means of data collection are used. Whereas IPv6 scanning detection works with large internet telescopes, DNS scanning detection only requires a single authoritative DNS server for data collection.

For finding scans in IP address space, simple definitions with thresholds can be used, e.g. classifying any source as a scan if more than 1000 different IP addresses are contacted within a day with no single one being contacted more than 10 times. This makes for a clear definition and reliable results. DNS scans, however, are more complex to find and can be very diverse in nature. Not just 2LDs can be queried, but also 3LDs, and scans can thus include asking for many different 2LDs or for one single 2LD repeatedly with different subdomains. Also, scans can ask for names repeatedly (e.g. of popular AuthNS, for different query types, web crawlers returning to the same domain, or due to misconfiguration). Much data is available, because each query contains several different fields and flags. Because traffic is restricted to the DNS protocol, relevant knowledge about the DNS will play an important role in the methodology of finding scans. Having a full picture of the nl traffic, this will also make some parts easier to explain, such as name order, breaches of query conventions, and so on.

To the best of my knowledge, this is the first work of this kind.

4 Methodology

The contributions of this research are:

- Defining and characterizing DNS scans
- A set of tools and visualizations to judge resolvers regarding scanning behavior
- Estimation of how much traffic stems from scans, and further details about the origin and type of this traffic, including both direct scans and indirect scans using open resolvers as an intermediate
- Features that allow detecting groups of resolvers with similar behavior, specifically aimed at distinguishing between scans and non-scans
- A clustering solution using those features with a high accuracy in distinction and grouping of campaigns
- An estimate of the impact of scans on authoritative name servers and of the effectiveness of current countermeasures (rate limiting)

Section 4.1 will delve into important considerations for resolver behavior, patterns, and how to manually classify sources. Section 4.2 will then build on these insights to generate features for each resolver and cluster them according to their behavior and chance of containing scan traffic.

4.1 Manual Classification

This chapter will further clarify what scanning behavior is, how it can be found, and which results were achieved by and what software was developed for manual classification of sources.

First in this section, definitions will be provided for otherwise unspecific terms such as *scanning operation*, and questions and hypotheses will be stated. Then, the following sections delve into relevant aspects of the data and explaining how scans can be found using different patterns.

4.1.1 Technical Environment

Data about the traffic to the Dutch name servers is recorded using a software called Entrada [20], which is in active development by SIDN.

4.1.2 Problem Statement

Before this project, no research has been done at SIDN and no papers published here or elsewhere attempting to find scans in DNS data. At SIDN, this has not been a priority, neither in the past nor now, since scans do not strain the availability of the name servers. Therefore, I will define what constitutes a scan and then provide hypotheses as a basis for the rest of this writing.

Definitions

A *scanning operation* shall be defined as an automated querying of a large number of domain names for the purpose of gathering data in bulk, where the purpose is directly connected to the DNS queries, seeing the DNS not as a service to enable connections on the internet, but as the object to collect data about. Explicitly not covered by this definition are:

- Individual queries triggered by a human being's everyday actions on the internet.
- Queries sent by an e-mail server or any other piece of software looking to connect to a limited number of other hosts for fulfilling their purpose.

Because these situations are considered normal usage of the DNS.

Different other kinds of automated analysis can also count as DNS scanning, such as:

- Web crawling
- Monitoring

These do not have the DNS as their main focus, but show largely similar patterns, fitting the definition of scanning. They are also automated, large-scale traffic that is not directly initiated by a human's actions and thus potentially unsolicited.

Examples

1. A resolver asking for NS records of domains names in alphabetical order, with constant query volume throughout the whole day while not repeating a single name once, classifies as being part of a scanning operation.
2. A resolver querying MX records of various different names, with many repetitions and no apparent order, does not classify as part of a scanning operation, but may be a mail server.

This definition does not provide a canonical way of identifying scanning operations among other traffic. Neither does it allow us to make a clear distinction between scans and non-scans. There will be various resolvers which cannot confidently be classified as one or the other. However, it allows making presumptions about which patterns a scanning operation might go along with, which will be the topic of the next section.

Normal Traffic *normal traffic* shall be any traffic that is not part of a scan. Note that normal traffic is usually concretely triggered by a user, whether that is typing a URL into an address bar of a browser, sending an email or receiving an email from a newsletter. In this, the intention of the action is connected to the queried name, which is not the case for scans.

Resolvers Any IP address sending DNS queries to an authoritative name server is, for the sake of this research, regarded a resolver. Thus, a more pragmatic definition is used instead of the technical one.

Resolver Behavior When writing about *Resolver behavior*, what is meant is the kind of queries a resolver sends, including all metadata, such as order, flags, names, and so on. The most trivial representation of resolver behavior is a database containing all queries and their metadata, such as what ENTRADA creates.

4.1.3 Analyses

To answer the research questions set in Paragraph 1, the following analyses will be presented:

1. Analysis of patterns in the total resolver population
2. Classification and analysis of individual resolvers's traffic
    a) Creating a small dataset of found scans and normal traffic
    b) Determining which features of traffic are affected by / different for scanning behavior

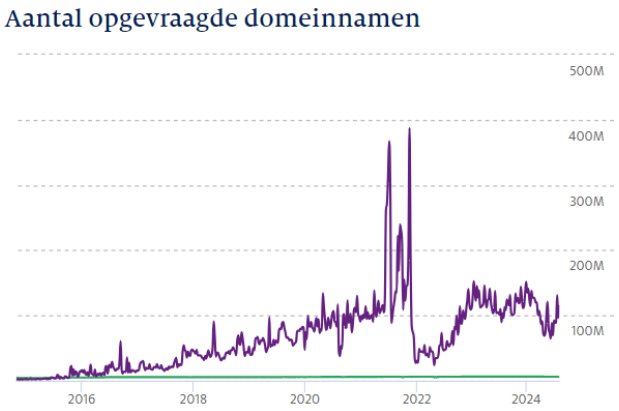


Figure 4.1: Graph of the count of queried domain names during each week from the SIDN labs statistics page, with spikes for no obvious reason.

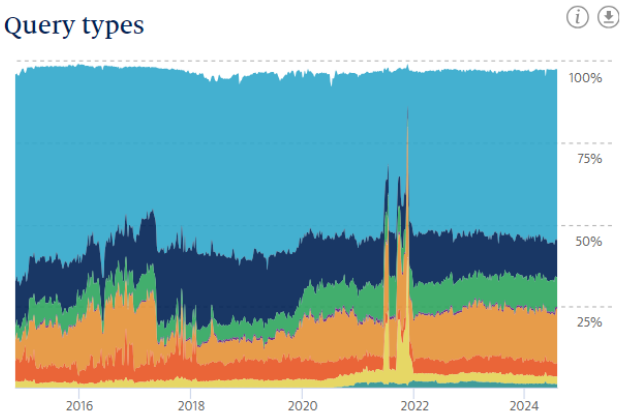


Figure 4.2: Graph of the share of query types from the SIDN labs statistics page

3. Feature creation from each resolver's traffic and
   a) Clustering on those features with the goal of distinguishing scanning resolvers
   b) Visualizations and demonstrations in feature space
4. Determining the impact of scans, including a case study of a large scan which occurred on May 21st

4.1.4 Anomalies

Rough statistics about DNS traffic can be found on the SIDN labs website describing different patterns and trends. One can see for example, that more traffic stems from

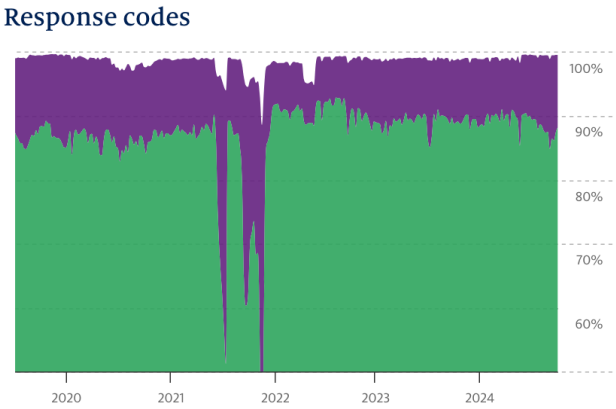


Figure 4.3: Graph of the share of response codes from the SIDN labs statistics page. Please note that the y-axis is shifted.

public cloud resolvers over time, or that the number of labels in requests is slowly declining due to implementation of But a lot of more curious anomalies can be found: Days and weeks in which unusually large amounts of traffic occur, or response codes differ wildly from the usual split (Figure 4.3), or different query types are used (Figure 4.2), or more labels are used in queries, for example. Before 2015, the number of valid (existing) domain names was larger than the number of NX domains being queried. Today, the number of NX domains encountered is larger than the number of existing domain names by a factor of about 20. While this goes hand in hand with growing traffic volume, the increase has been so severe that it probably cannot be explained by this aspect alone, and the diagram also shows great variance from day to day.

A hypothesis could be that at least some of the spikes in response codes correspond to scans (see Figures 4.1, 4.2, and 4.3). Other causes might be changes or bugs in software.

Several types of analysis and data exploration have been done before starting to classify individual resolvers. This chapter will describe some analysis and patterns in the data that are relevant to the later parts, as well as some considerations. These points each concern one or more columns of the data.

What underlies all patterns explained next is the idea that normal resolvers have typical ranges of behavior within which they operate, and that these ranges are different from what scanning resolvers exhibit. For example, scans might have a different fraction of distinct query names, or of response code 3 occurring. Whether particularly large or small values are indications of scans is not clear a priori.

Distinct Percentage Recursive resolvers are expected to cache results. Still, the TTL does not allow records to be cached for more than one hour, and names may be evicted

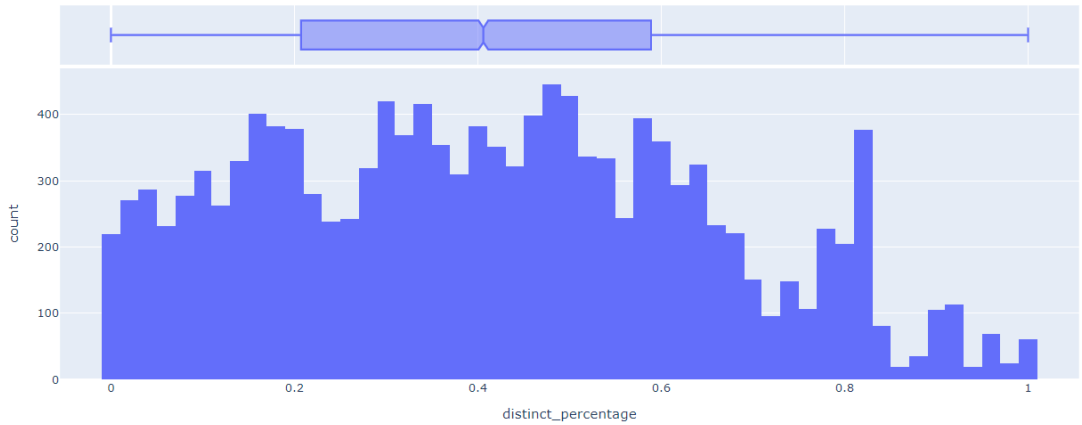


Figure 4.4: Histogram and box plot of each resolver's fraction of distinct 2LDs (resolvers with >10k queries on April 3rd). Exact formula: number of distinct 2LDs queried ÷ query count

from cache even before this time runs out, which means that any recursive resolver serving clients will repeat names eventually within a day. On the other hand, an unpredictable client base and effective caching typically mean that names are mostly not queried more than 10 times per resolver per day. This leads to a broad distribution (Figure 4.4) with a median of 40 % and a standard deviation of about 20 %, featuring some resolvers to either side that deviate from expected values. While the histogram is rather ragged, unusual peaks can be seen around 100 %, 80 % and 20 %.

It could be expected that higher query counts lead to higher distinct percentages, because more queries within the same time frame mean more different names queried and more effective caching, reducing the number of repeated names. But this is not the case. In Figure 4.5, it can be seen that resolvers with more queries generally ask for a smaller fraction of distinct names. This is also true for specific groups (notice for example resolvers from Google, represented by the red streaks with downward slope running from the top left corner to the bottom right). This could be because worse caching - repeating queries more often than necessary - leads to more queries being sent, or because larger numbers of queries result in more cache evictions if memory is not increased.

What becomes clear from both diagrams is that many outliers exist among resolvers - to the right and left of Figure 4.4 and to the top right of Figure 4.5. It can also easily be determined that the largest number of distinct names queried by a single resolver is around 3 or 4 million.

The reason for such a complex distribution in Figure 4.4 might be the many factors that influence distinct percentage: Number of clients, behavior of clients, caching policy, available space for caching, queried names, diversity of clients and more, specifically many implementation details.

Figure 4.5: Scatter plot of query count and distinct percentage of resolvers. Colors represent the organization of the IP's AS, but are not unique.

4.1.5 Patterns

This section features more patterns relevant for judging scanning behavior.

Label Count As for the count of labels in each query, the usual and recommended number of labels to include in a query to SIDN's AuthNS is 2 (see `https://stats.sidnlabs.nl/nl/dns.html#aantal%20labels%20in%20domeinnamen`). Query name minimization suggests that resolvers should not query our TLD with more than 2 labels, because additional labels are norrmally not necessary for answering the query and reveal more data about the intent or user than necessary. On average, a query contains 2.37 labels in April 2024. Looking at the fraction of requests with ¿ 2 labels, it becomes clear that some resolvers implement qname minimization very stringently, while others do not, and a small number even send a very large amount of queries providing a 3LD as well.

Usage of Public Resolvers in Scanning There is a trade-off between freshness/latency and performance in DNS scanning. Using public resolvers increases performance significantly for known software [11, p. 54], even increasing success rate for ZDNS [11, p. 56]. MassDNS even provides a list of open resolvers for use with the software.[1]

However, using open resolvers comes at the cost of freshness and latency, because external recursive resolvers add another party to the network traffic and may return cached results, whereas AuthNS per definition return fresh results.

[1] `https://github.com/blechschmidt/massdns/blob/master/lists/resolvers.txt`

Most Queried Names Normal 2LD entries for the NL zone have a TTL of 1 hour, implying that they should not be queried more than 24 times per day with perfect caching. However, records for popular name server names, such as `transip.nl`, may be queried more often for several reasons:

- They are often authoritative for many names
- They often have multiple subdomains such as ns1/ns2/...
- They can be authoritative for names outside of the nl zone

Therefore, it is typical to see these among the most queried 2LDs of resolvers, and some even query them extremely often due to faulty caching, perhaps.

Query Count A staggering number of IP addresses send less than 1000 requests each. Very few sources send very large numbers of requests, forming a long-tailed distribution. Resolvers sending more than a million queries in a day are uncommon, as most public resolvers use multiple backend IP addresses to send traffic from. The highest number of queries sent by a single resolver on April 3rd is 17.6 million queries, the rightmost point in Figure 4.5. All of the largest resolvers have a rather low distinct name fraction of 30 % or less. Resolvers exist with around 7 million queries and 47 % distinct names, and 2.7 million queries and 100 % distinct names, which is suspicious. These sources also receive response code OK for more than 98 % of queries, making them record holders for most NL zone coverage, that is largest number of distinct valid names at around 4.6 million out of 6.3 million registered names. The specific IP addresses all belong to large hosting companies such as Hetzner Online, BIT or Ledl.net.

NX domain percentage There seems to be a typical range for the percentage of NX domains requested by each resolver. In the total dataset, it is around 10 % on a normal day. For resolvers, it can vary a little, but there are clear outliers in both directions. Both a small (about 1 %) and large (more than 30 %) fraction of NX domains is unusual. Even though 10 % of queries contain NX domains, 95 % of all names queried are non-existent, since they constitute a much larger number of names.

Distinct name percentage When comparing the number of distinct domain names queried by a source to the number of total queries, usually there is a relatively high number less than 100 %, presumably dependent on the caching mechanisms of each resolver. For large numbers of queries, this percentage necessarily decreases, as the Dutch zone only contains around 6 million names. Here, too, there is a typical range and outliers can be found to both sides.

Peaks in Query Volume Even in the overall traffic, unusual peaks can be observed.

Question Types Question types are used with wildly differing frequency, with many of them not serving much use.

Bogus A number of queries are unanswerable due to their parameters, such as (visibly) wrong source IP addresses, invalid domain names, qtypes, and more. They amount to short of 1 % of the total traffic, although the full list of conditions is incomplete. Previous research focusing on bogus has defined it as queries that are invalid or unanswerable, particularly by not following current standards, and needlessly repeated queries. This work has taken a simpler approach and looked at only the invalid queries. Relevant for this are RFCs regarding DNS query format, as well as RFC 1918 about reserved address space.

Repeated Queries Repeated queries are more difficult to analyze, especially because they need a good definition of what counts as an unnecessary repetition. However, it can be seen that many repetitions within a short time frame occur and this kind of extra traffic could make up a large percentage of all traffic.

Subdomains Many queries to the NL name servers contain more than 2 labels (see the traffic statistics on the SIDN labs website), although this is discouraged through [7]. Adding to this, apart from common subdomains such as `www` and technical ones such as `_dmarc`.

www is sometimes regarded as more likely in traffic coming from humans [1, p. 36], however there are also resolvers using this subdomain in a suspiciously large fraction of their requests, presumably because it is relevant for scanners, too. Therefore, this is not a linear relationship. Some resolvers ask in large number for subdomains like `*.xyz.nl`, which is not valid. Tens of millions of such queries come in each day, and since 3LDs do not concern our name servers, these queries are answered. But a typical name server at the 2LD level will not answer them because `*` is not a valid label. Asterisks are used in wildcards, that is when queries for every possible label should be answered with a particular record regardless of the label. One might speculate that those asterisks are trying to ask for any subdomain. Only in less than 0.1 % of queries including asterisks does one appear in a different position than as the first label.

Invalid Symbols Many queries, even from public resolvers, contain invalid symbols in the qname. Most often, this is probably an @-symbol, stemming from email addresses that are mistakenly sent to the resolver.

Source Ports Source port numbers increase somewhat as expected, as different technical implementations will either use ports starting from 1024, 32768 or others. This means that in overall traffic, higher port numbers are used more. Some queries are also sent from ports ¡ 1024, and some port numbers see too high numbers of queries and are outliers, such as 32768 or 14.

Query IDs A similar pattern emerges with query IDs, which should be completely random, but also contain visible outliers.

Wrong TLDs Some queries also come in that have a different TLD than `nl`.

Qclasses For basically all traffic, qclass internet should be used. Only a few queries come in with a different qclass.

Truncation and TCP Queries that receive truncated responses are supposed to be repeated via TCP. In practice, this is not the case for all truncated answers.

Unsanswered Queries Queries usually only remain unanswered if the source runs into our servers' rate limiting. This is the case for about 1 % of traffic.

Label Count While most queries contain exactly 2 labels as should be the case except for rare occasions, there are also queries with 1, 0, or many more labels.

UDP and TCP UDP usage is generally very high as expected, and there are sources that do not use TCP at all, which can happen if the maximum UDP packet size is large enough. There are also sources that use TCP for a significant fraction of their queries, which is unusual.

Coverage 99.8 % of all names from the `NL` zone occur in the traffic of one day. While no single source covers all names (the most being around 70 %), large ASes such as Google reach near 100 % coverage of the NL zone each day through their public resolver traffic.

All these points should corroborate the following conclusions:

- There are many ways in which a single query can be unusual ore even invalid, but
- there are even more ways in which the behavior of a resolver can be unusual.

Some of these patterns will be relevant for the following chapters, while others did not show relevance for the topic of scanners.

Patterns

Before considering any data, one might already suppose some behavioral patterns that can be reasonably expected to co-occur with scanning operations.

Because the purpose of a scan is to collect information, a *high number of queries* should be expected. Since the subjects of the DNS are domain names, and the NL zone contains a great many of them, scans will be querying a a*large number of different domain name*, and might *avoid repeat queries*. On the other hand, there might be reasons for interest in a subgroup containing a *smaller number of domain names*, and *very regular querying* to notice differences in a timely manner.

While normal day-to-day traffic caused by regular users should be organic, traffic stemming from scans might exhibit more mechanical behavior, such as:

- sudden peaks or drops in query volume at times, rather than a smooth curve
- a very high, very low or very regular percentage of NX domains
- an unusual choice in target name servers, such as not addressing all three name server IP addresses
- querying names in a specific order, such as alphabetically
- querying only a certain kind of domain names (e.g. not containing numbers or only containing a single word)
- unusually straightforward patterns in qtype, number of labels, protocol usage, IDs or query times
- a large number of queries

To sum up, it is expected that traffic stemming from scans shows behavior that is unusual, and is therefore distinguishable from normal, purpose-driven DNS traffic. In particular, there are many features of normal traffic that are difficult to mimic when performing scans. For each feature, it is easy to come up with a scan that does seem normal in this way, but it is assumed to be difficult to hide scans fully, that is to make it seem normal in every way. As a first step, lists of most extreme outliers were used to find potential scanning sources. The different types of lists are:

1. Highest percentage of distinct domain names queried
2. Highest and lowest NX domain percentage
3. Highest number of queries in total
4. Highest intersection with domain name lists from CT logs and Common Crawl
5. Most queries sent within 60 seconds

Aspects used

The crucial goal of this research is the ability to tell apart scanning operations and normal traffic. This distinction is non-trivial, even for humans, because scans can display various kinds of behavior, often looking normal in many regards. Finding scans needs adequate measures, analysis tools and visualizations, even before taking further steps such as grouping or machine learning. From a simple list or table of incoming queries, it is difficult to tell the motivation behind the traffic in most cases, even for a human.

Thus, different kinds of visualizations and an elaborate script have been developed as part of this work to make it as easy as possible for a human to find patterns in traffic that could hint at a scan. This script takes a part of the dataset as input, usually filtered to only contain queries by a single source IP address, and prints various statistics, excerpts from the traffic, graphs and tables to enable easier assessment of the traffic.

The following aspects are relevant and indicate scanning behavior to varying degrees. Only some of them allow certain classification by themselves.

- Total query count, number of distinct 2LDs queried, and fraction of queries asking for a new name.
  - Tiny (<5 %) fractions asking for new names indicate unusual repetition of names, which is often caused by monitoring or scanning 3LDs. They often go hand-in-hand with a small fraction of NX domains, because querying NX domains many times in a row does not gain any knowledge. 3LD scanning should not necessarily be visible at the TLD level because the results can be cached, but in practice many such scans are visible, even within the traffic of popular public resolvers. Small distinct name fractions are a strong indication of scans, but need to be confirmed further.
  - Large (>90 %) fractions of new names indicate most names being queried just once. It is unusual to see values close to 100 %, because queries for the authoritative name servers of subdomains are often part of a scan's traffic, and are repeated regularly. However, scans do exist which query purely 2LDs, without querying any further authoritative name servers. Presumably, they might only be interested in whether a delegation exists in the TLD name servers, without caring about the actual records on the authoritative server for the name. Large fractions of new names are a strong indication of scans, especially when query count is also high, but need to be confirmed by another attribute.
  - Particularly large numbers of queries on the order of millions usually also indicate a scan, because normal resolvers only send tens or hundreds of thousands of queries. Indeed, out of 18 sources (out of 95 total) checked with more than 2 million queries, all were sending scanning traffic. However, this can

only be taken as a hint, because there can be other reasons for large traffic volumes, and each scan has to be confirmed through other attributes.

- A small excerpt from the query traffic shortly after midnight can already yield different insights:
  - Alphabetical order is unusual. There is no reason why organic traffic from clients of a resolver should follow any alphabetical order, and any querying of a substantial number of names (at least 20 to be discernible from patterns of random origin) can only be caused by some sort of list or pattern being queried, which is regarded as a scan per definition. Any such patterns must thus be scan traffic.
  - Multiple queries within a millisecond are unusual. If the same name is repeated, but only 1 or 2 times, potentially even with different query types such as A/AAAA or A/DS, or MX/DS, this is often normal behavior. Querying different names or one name repeatedly in such quick succession is often a sign of scanning.
  - Particularly, if *each* name is repeated with a fixed set of unchanging query types (such as A/AAAA/TXT/DS), then this is a very strong indication of scanning traffic. Normal resolvers query for those record types specified by clients, plus possible A/AAAA records in combination or DS/DNSKEY records for DNSSEC, or NS in the case of qname minimization. Unusual groups of more than 2 query types are a strong indication that the traffic is generated with the goal of gathering data.
- Query IDs are examined. The number of different IDs used in the traffic is compared to the number of IDs expected if generated randomly. A histogram is created to ensure the distribution is even.

  IDs should be randomly generated, and this is the case for the vast majority of both normal resolvers and scans. Two cases were observed for this not being true:
  - Query IDs were reused for repeated queries within a very short time, which was not deemed a sign of scanning.
  - Query IDs were not correctly generated, for example being chosen round-robin or only one ID being used, which is a sign of very rudimentary implementation and was only found in otherwise clearly distinguishable scans (e.g. Paragraph 5.3.1).
- Distribution of queries onto source ports should be random among non-reserved ports, but often shows patterns. Some scans use simple software that does not correctly randomize ports, for example only using a single port for all queries like *unanimous* (Paragraph 5.3.1). Both IDs and source ports can also be used to encode information as done in [18], and anything other than a random port

distribution is unusual, but any relationship to scanning is unclear. Different software technicalities might play into this aspect.

- Query counts for each time of day are plotted in a histogram like Figure 4.15. A typical distribution is even, but shows more traffic during the day, without extreme spikes. Here, many anomalies can show:
  - Small spikes or raggedness in traffic are to be expected. More, the less queries are sent. Extreme raggedness as seen in Figure 7.8 is highly unusual and strongly indicates abnormal traffic, often scans.
  - Large spikes in traffic volume, as in Figure 4.16, mostly come from automated traffic.
  - Extraordinarily even traffic distribution without variances is similarly unusual and very probably automatic in nature, being a strong indication of a scan.
  - Gaps in traffic do not occur often for normal resolvers and can be a sign of scanning, but no causal relationship can be assumed if no other clear patterns emerge, because even open resolvers can sometimes have gaps in their service.

  All these patterns can vary in severity and still need to be confirmed in plausibility by other values, such as query types or response codes during the specific times.
- Alphabetical distribution. Each language and zone has a characteristic distribution of starting letters. The exact proportions are not relevant, but some domain names start with digits, with anything larger than 2 being extremely rare.
  - Any alphabetic distribution with a significant amount of digits queried is a strong hint at generated domain names. This goes hand-in-hand with names of a specific length occuring too often. A significant number of generated domain names always indicate a scan.

    This can also manifest as rare letters in general appearing more often than usual, with general proportions of the symbols occurring being correct. This can be seen in Figure 4.7 and 4.6.
  - If one or a few letters are much more common than others, this can be the result of one of two things:
    * Excessive repetition of names, such as for monitoring or subdomain scanning
    * Clustering of names within a particular range of the alphabetic, such as when scanning in alphabetical order incompletely or when scanning similar names to an example, that is names differing only in some letters

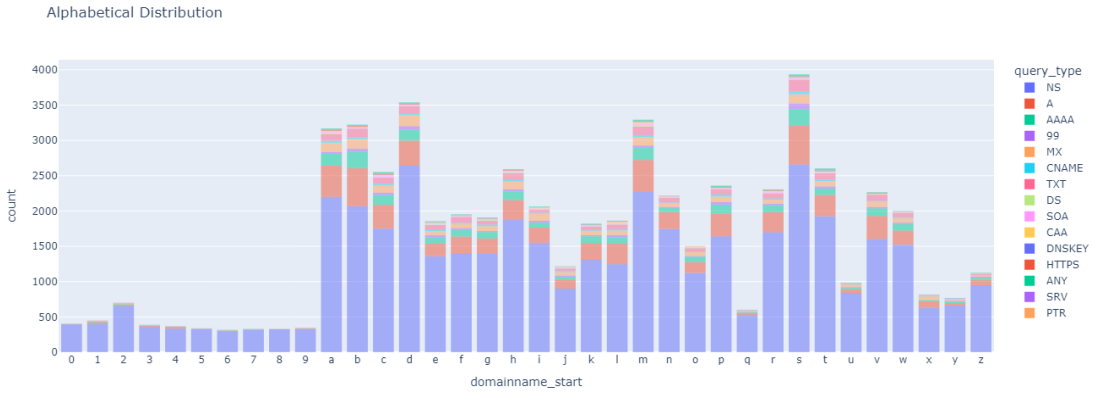


Figure 4.6: Starting letters from a public resolver affected by *sleeve*, a large scanning operation with randomly generated names. Counts are slightly evened out by many A queries about random names (bottom-most part of each bar in blue).

As the distribution of starting letters is highly characteristic, any discrepancies are signs of unusual traffic. While name servers often make up a significant part of traffic and the distribution might look slightly different for mail traffic, for example, large variations only occur due to extraordinary behavior, given a large enough sample size.

- Country, AS, organization and public resolver status (whether it is a known public resolver). Sources from hosting companies are often homogeneous in traffic, while public/open resolvers usually only contain scans mixed with other traffic, making them more difficult to spot. Multiple scans can be contained, and they are often visible either as spikes in time distribution or patterns in the query scatterplot. The organization can be a strong indication of sources being affiliated, i.e. part of the same scanning campaign. All of these attributes are not meant to aid in classification and only aid in knowing where to search for signs of scanning.

Manual classification was used to confirm labels of each label mentioned unless specified otherwise.

Sources from the aforementioned lists were examined and classified. Then, with a list of scanning sources, those were assigned to the same group that showed commonalities beyond simple coincidence, identifying IP addresses belonging to the same operation. For this purpose, DNS2Vec [40] was also used to find similarities in queried names, which cannot easily be seen otherwise.

This showed that examining sources was a useful and pragmatic simplification and suitable for finding many scans. The manual classification also showed that at least 3 % of all traffic stems from scans, with probably a 10-times higher number in actuality.

It is hoped that the visualizations will provide valuable tooling for future analysis of resolver behavior. In addition, different extensions on the database columns have been developed to provide a more easy-to-use and powerful experience in working with this data.

A difficulty of this and any other resolver-based approach is that IP addresses do not perfectly describe individual sources. Even ignoring the heterogeneity of large resolvers, even individual resolvers cannot be seen to be represented by exactly one IP address. Not only can the backend of a single resolver be implemented in such a way that it uses different IP addresses, but even MassDNS has the feature to use IPv6 prefixes for its requests, distributing traffic between thousands of different IP addresses and hiding from analysis. Of course, the sizes of used prefixes are not known, and other IP ranges will contain completely different resolvers in a prefix of the same size. Thus, some kind of intelligent handling of aggregation size is necessary for a full analysis. The three possible groups for aggregation are IP address, IP prefix (of varying size), and AS. We will keep this problem in mind and ignore it at first for the following research.

In the absence of a better solution, aggregation by IP address is least error-prone, as prefix sizes are difficult to determine and AS assignments might change over time and are missing for parts of the data. Entrada automatically adds columns containing the AS number and organization behind the AS. Using ASes for aggregation could lead to incomplete results, and non-round prefix lengths are computationally intensive to compute in SQL due to the IPs being stored in string format. At first, abstraction from IP addresses will only be done where multiple addresses are found to have similar behavior.

Even on just one day of data, plenty of scans could be identified. Out of 89 IP addresses that were evaluated manually, 54 were clearly performing scans of some kind, 6 were showing no signs of scanning and 29 could not be clearly classified because they either contained scans in parts of their traffic or were unclear even to the trained eye. These sources were not picked randomly and thus do not make for a representative sample of the traffic. Of 4 randomly picked and classified IP addresses, none could be reasonably classified as scans. One reason for this is that most sources only send a very small number of queries in a day (the median is 21 for the evaluated day).

Using the data obtained through this work, further analysis can go in two different directions: Forward, analyzing traffic to find scans, and backward, analyzing differences about scans and non-scans to find further patterns and evaluate effectiveness.

Examples

Many interesting examples of traffic patterns from manually classified sources can be seen in figures Figure 4.7 and following. All of the visualizations in this section describe the traffic of one source only, that is IP, AS or IP prefix. More examples can be found in Section 7

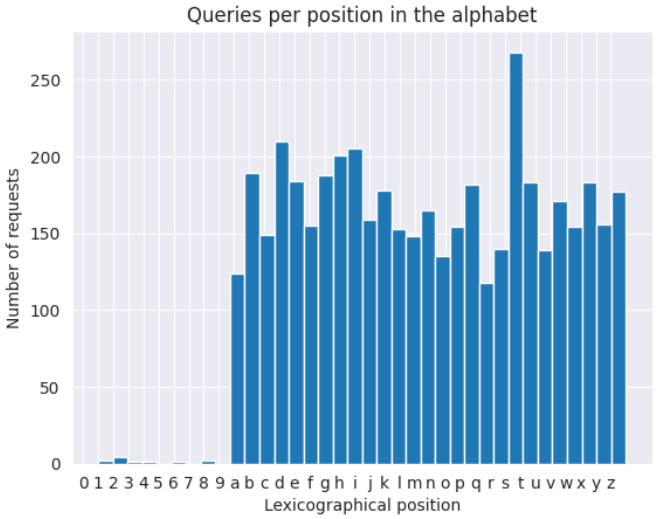


Figure 4.7: An alphabetic distribution of domain names that is unusually even

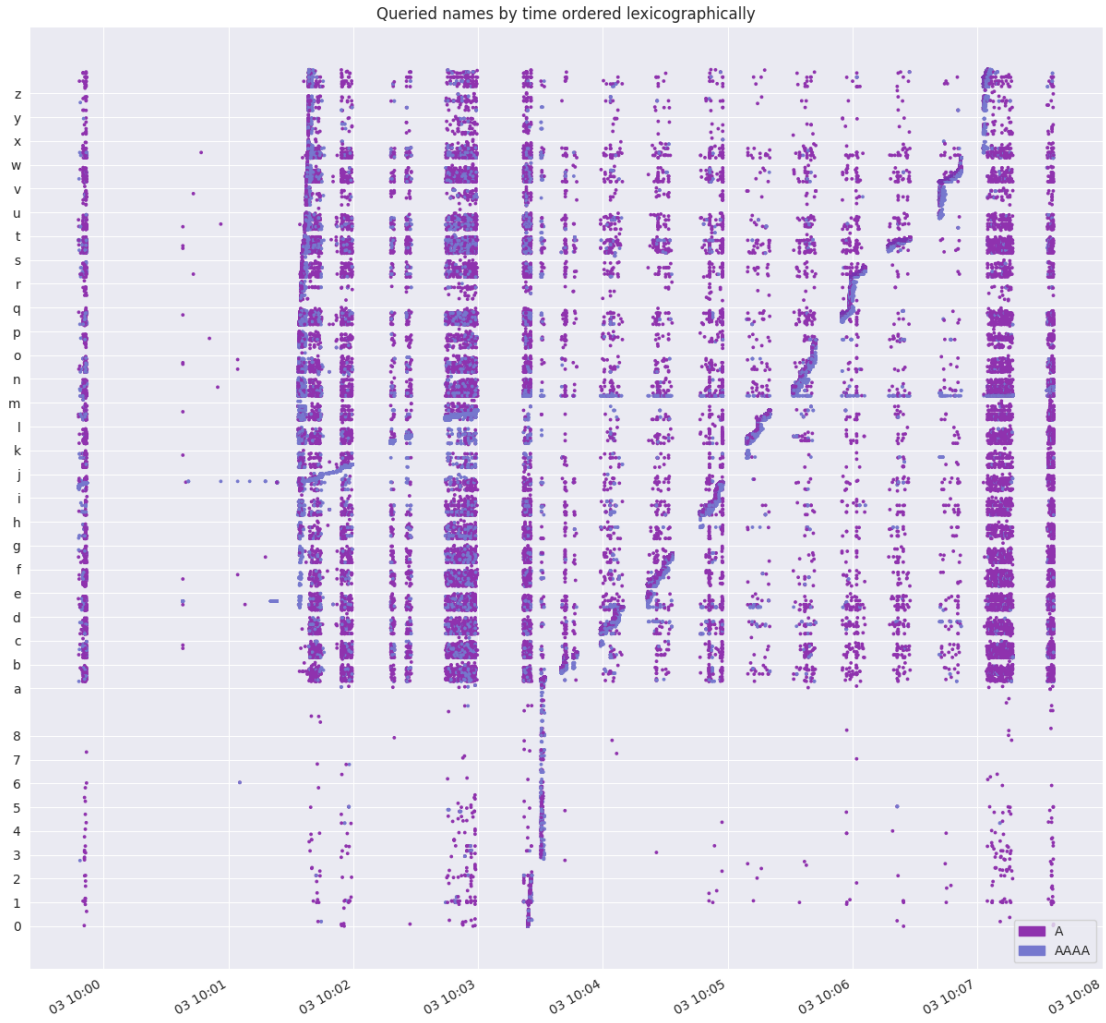


Figure 4.8: Queries containing a scanning operation with alphabetical order among other traffic, visible as a diagonal line. Time frame ≈ 10 minutes

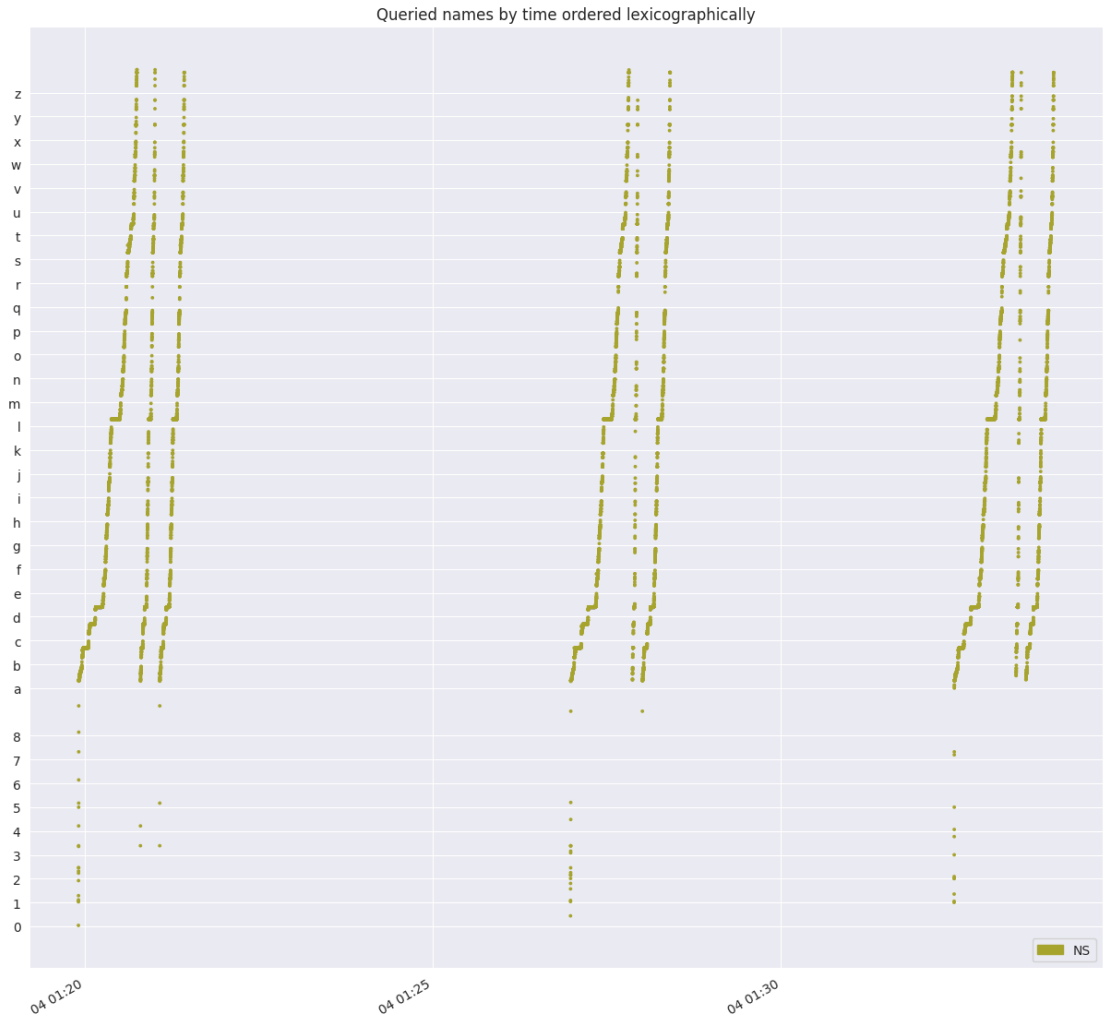


Figure 4.9: Queries performed in short bursts that are each in alphabetical order (time frame of about 15 minutes)

Figure 4.10: Traffic with a clear, mechanical-looking pattern in time (time frame = 1 day). Traffic like this could stem from monitoring of some kind, but is certainly machine-generated in origin.

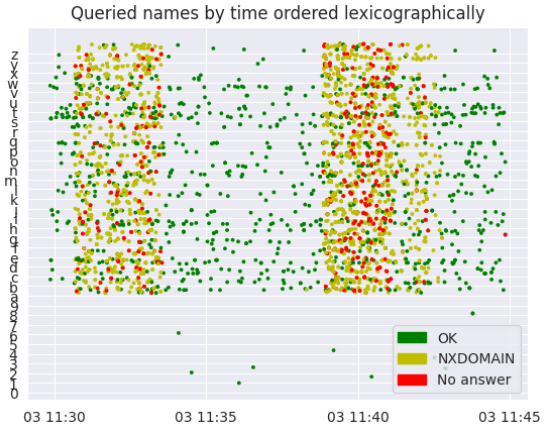


Figure 4.11: Unusual peaks in traffic whose nature becomes clearer when looking at response codes (time frame = 15 minutes)

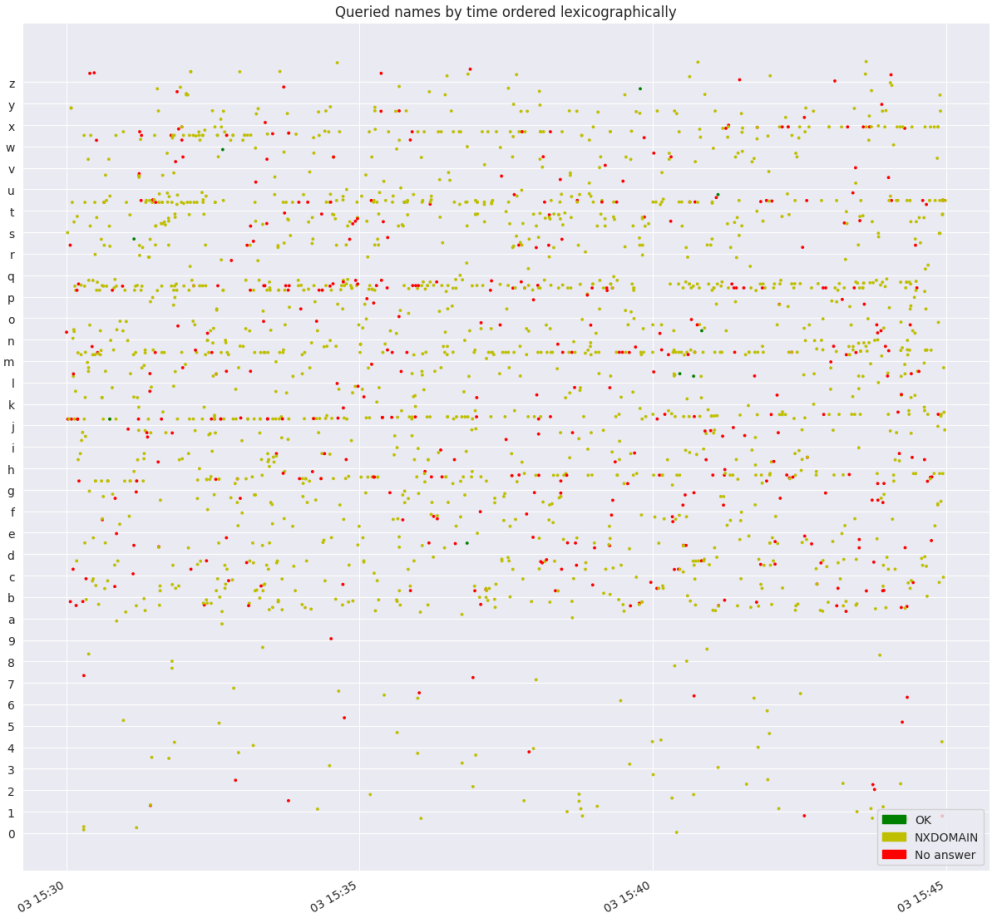


Figure 4.12: Traffic barely containing registered names (time frame = 15 minutes)

Figure 4.13: A scanning operation asking for NS records first, then for A records in a second phase (time frame = about 25 minutes)

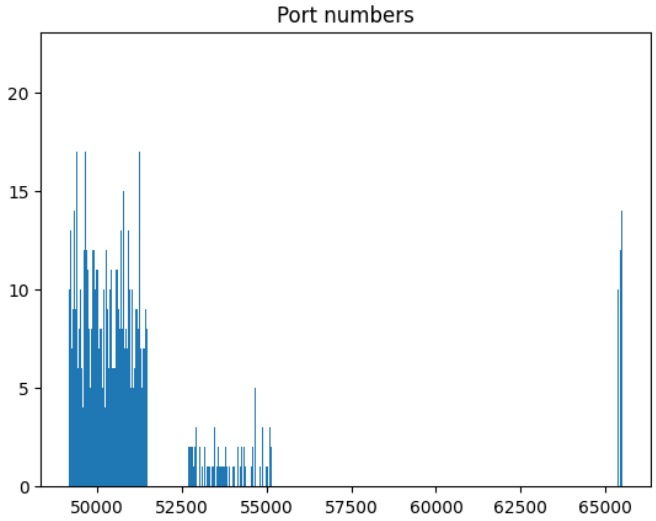


Figure 4.14: Only some specific port ranges are used with different frequency

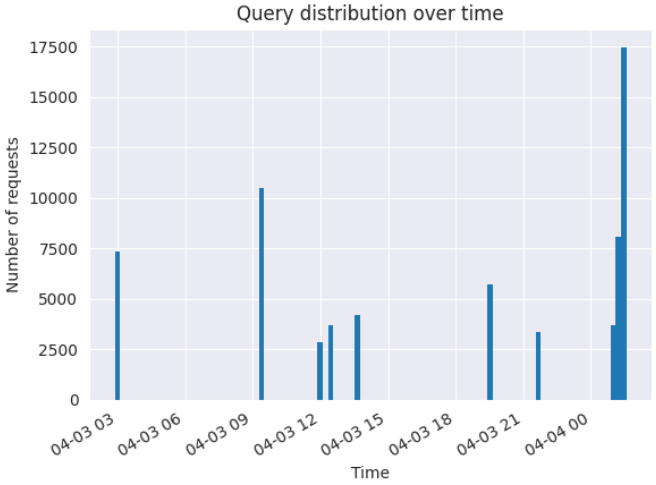


Figure 4.15: Highly unusual distribution of traffic over one day, with irregular gaps

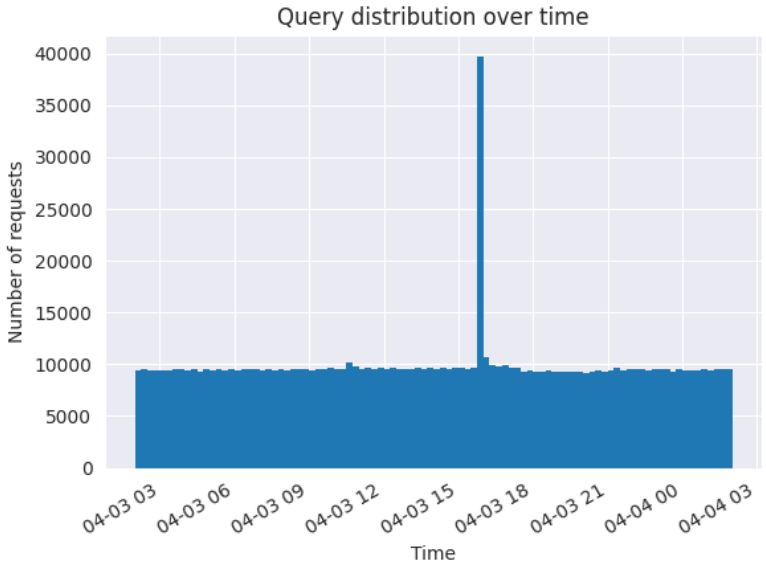


Figure 4.16: Unusually steady time distribution with one peak, from operation *cream*

Groups and Operations

For this work, we define each IP address as representing a resolver. This aligns with the definitions of previous work, but does not agree precisely with the technical definition. Public resolvers mostly use different backends for one user-facing recursive resolver, and presumably, scans might also use entire IPv6 prefixes to hide their behavior from an IP-address-based analysis. We ignore the technical definition of what a resolver is, and presume, for the sake of simplicity, that every IP address querying our servers constitutes a resolver.

During manual analysis, it quickly becomes clear that some resolvers show similarities that cannot be incidental, and sometimes are so similar that they can be assumed to form a coherent group sending traffic identical in nature (if not in names). All instances of this similarity occur with resolvers of similar origin, so no public resolvers showed identical behavior to ones from the networks of hosting providers, and so on. Extreme similarity between resolvers is assumed to mean affiliation, because the number of differences that can occur are vast, and query patterns as seen in the colorful scatterplots, as well as time distribution, are highly characteristic.

4.1.6 Results

The classification process of scans vs. non-scans is non-straighforward. The most obvious feature scans can exhibit is an alphabetic querying order and a large percentage of distinct names, each name being queried only once. While this is true for some scans, many do not exhibit those patterns. Many other patterns are similar in that they help find some scans, but are surely not applicable to others. A larger number of dimensions for analyses also yields many sources that deviate in some dimensions, but not others, and cannot be classified easily.

Examples of graphs with patterns can be seen in this chapter. What becomes clear is that scans are very diverse, perhaps more diverse than normal traffic, but usually unusual in one or multiple ways. Features of scans come in two kinds:

1. Those that emerge from the content of queries sent, such as a low NX domain count, alphabetic order, or a large distinct name fraction. These also mostly give insight into the motivation of the scan, such as monitoring few domain names, or domain scraping for a large number of names.
2. Implementations patterns, which follow conventions or RFCs, such as randomly generated IDs, qname minimization, or usage of TCP and EDNS UDP. While these do not provide any insight into a resolver's purpose, they help distinguish implementations and configurations, which can be less sophisticated for some scans.

Many campaigns do not try to hide, blatantly using single IP addresses to query millions of names, or using resolver configurations that make them easy to see (e.g. leaving the

recursion desired bit set to 1). As is often true on the internet, there is nothing that cannot be found here.

Calculating correlation scores between different traits and whether a source is a scan did not yield any strong indicators for features, which might have many reasons:

- Scan can be very diverse, with no single feature discriminating all of them well.
- The ground-truth dataset is small, being created by hand.
- Scans might exhibit unusual behavior, but mostly not just in one extreme. Take NX domain percentage, for example. It is clear that most domainers would like to minimize the number of queried NX domains, because they would like to have exact coverage of the NL zone, and no more. And it is true that many scans have a low NX domain percentage, lower than traffic from other resolvers. However, other scans also use inaccurate lists and are therefore asking for 90 % or even more NX domains.

Using just one feature is insufficient for finding all scans, and a multi-dimensional method is necessary. The manual classification approach is rather convoluted, requiring domain knowledge and examination of traffic in various dimensions. Besides, classifying sources takes time. The clustering approach described next is supposed to alleviate these issues while giving a clear picture of total traffic composition.

4.2 Clustering

4.2.1 Features / Similarity Measures

Gaining further insights into the data requires it to be machine-readable, i.e. describing patterns using quantitative measures instead of graphs. This research is taking the approach of defining a measure of similarity on the DNS data. Intuitively, this can be done at two different levels, with both approaches helping to analyze traffic with the goal of finding scans. A discussion of clustering individual queries vs. clustering resolvers can be found on page 104.

This work employs clustering on resolvers. The approach is easily extensible, with the possibility of adding new features in the future. Particularly, projects like DNS2Vec, intersection with various lists, and new ideas for features can be added easily. An overview of different applications possible with this approach can be found in Section 4.2.1.

Procedure

In the following, features are used to describe the behavior of each resolver sending requests to our name servers, only excluding those that send merely a small amount of traffic (<10k queries per day). This means that many small resolvers are eliminated from the dataset which do not send an amount of traffic that is sufficient for a robust calculation of features, heavily reducing the number of resolvers (to about 5 %) while retaining most of the traffic for analysis (about 84 %). Even manual classification is difficult for sets of less than 10k queries, and any scans this small are insignificant. Figure 4.1.5 mentions that it is possible for scans to use vast IPv6 prefixes to distribute queries among many addresses, but this topic is left for future research.

Features

The features used try to describe well the aspects that proved indicative of scanning behavior in the manual analysis. Particularly, they are distinct name percentage, query count, average domain name length, query types, average number of queries per domain name and repetition statistics.

Additional to straightforward features, such as the percentage of queries with a specific flag set or using a particular query type or TCP, a variety of more sophisticated features have been added that are specifically relevant for the distinction of scanners. In contrast to [1], this work uses a large number of features engineered for the specific purpose and performs clustering on it. The purpose of finding scans is different than in [32], and unsupervised learning is used to explore the data and avoid training a classifier on a dataset that is too small.

In total, 8 groups of features are combined to obtain the feature set:

1. A number of features obtained by one aggregation by source,
2. statistics about the number of queries sent per domain name,
3. metadata about the source (one-hot-encoding of the resolver country),
4. statistics about the steadiness of the number of queries being sent,
5. statistics about the repetition of queries,
6. statistics about the time between consecutive queries,
7. percentages of names that can be found in typical name lists and
8. DNS2Vec embeddings (described next in 4.2.1).

None of the features mix data from different resolvers, so all following explanations can be read as including "for each resolver separately".

Obtained Through Straightforward Aggregation This group contains 147 features and is calculated by performing a single aggregation on all queries.

Count Count: Total number of queries

Distinct Names Distinct Count: Number of distinct names (2LDs) queried, Distinct Percentage: Distinct Count ÷ Count

Time Time Average: average timestamp among all queries, Time Deviation: standard deviation of timestamps, Time Minimum, Time Maximum, Time Frame: maximum minus minimum

Labels Label Count x Percentage: Number of queries with exactly x labels ÷ Count (for $x \in [0, \ldots, 9]$), Label Count More Than 9 Percentage

Bogus Bogus Count: absolute number of bogus requests sent, Bogus Percentage: Bogus Count ÷ Count

ID ID Minimum, ID Average, ID Deviation

TCP TCP Percentage

Flags Zero Bit Percentage, Authoritative Answer (AA) Percentage, Truncation (TC) Percentage, RD Percentage, RA Percentage, Authenticated Data (AD) Percentage, Checking Disabled (CD) Percentage

Source Port Source Port Minimum, Source Port Average, Source Port Deviation

Response Codes Response Code 0 Percentage, Response Code 3 Percentage, No Answer Percentage, Response Code Other Percentage

EDNS UDP EDNS UDP Average, Minimum, Maximum, Variability: Maximum - Minimum

Domain name length Domain Name Length Average, Deviation

TTL TTL Average

IP version IPv6 Percentage

First characters Name Starts With x Percentage (for each letter and digit x)

Targets Destination x Percentage (for each server $x \in$ [ns1, ns3, ns4])

Operation Codes Operation Code x Percentage (for each operation code $x \in$ [query, inverse query, status, notify, update, stateful operations])

Query Types Query Type x Percentage (for each valid query type x)

Query Class Query Class x ∈[internet, chaos, hesiod, none, any])

Punycode Punycode Percentage: fraction of names starting with `xn--`

Query Counts Per Name This group contains 6 features and is evaluated by first calculating the query count for each domain name, and then processing these counts.

Per Name Query Count Per Name Query Count Average: average number of queries per 2LD, Deviation: standard deviation of the number of queries per 2LD

Per Name Query Percentage Per Name Query Percentage Average: Per Name Query Count Average ÷ Count, Min, Max, Deviation: analogous

Source Metadata This group contains the resolver's AS's country, encoded as a one-hot-vector using 249 binary-valued features.

Country Country is x (for every country or dependency x, taken from the list of all 249 ISO 3166-1 A-2 country codes as used in ENTRADA, taken from Wikipedia)

Burst Query Frequency Statistics This group contains 11 features describing the variability of query counts over time.

Count over 60 seconds Count Over Short Time Frame Max: maximum number of queries sent within a single minute, Count Over Short Time Frame Min, Count Over Short Time Frame Different: Max - Min, Highest Count Time: start of the time frame with maximum query count, Lowest Count Time, Count Over Short Time Frame Quotient: Min ÷ Max, Count Over Short Time Frame Deviation: standard deviation over the counts for each bucket,

Percentage over 60 seconds Percentage Over Short Time Frame Max, Min, Deviation, Difference: analogous to Count

Repetition Statistics This group contains 3 features describing repetitions within the queries. A repetition is a query by the same source carrying the same query type and domain name (2LD).

Repetition Repetition Percentage: fraction of queries which are repetitions (have already occurred once before, thus the first query is not a repetition), Repetition Time Average: average time difference between a repeated query and the identical query before (Repetition Time), Repeat Time Deviation: deviation of Repetition Times

Query Delay Statistics This group contains 2 features describing the time delay between consecutive queries (Inter-query Times).

Inter-query Time Inter Query Time Average: average Inter-query Time, Inter Query Time Deviation: standard deviation of Inter-query Times

Name List Intersection This group contains 6 features describing the overlap of queried 2LDs with 3 different name lists. The name lists in question are 1. A full list of registered NL domain names 2. A list of NL domain names extracted from Common Crawl 3. A list of NL domain names extracted from Certificate Transparency logs item 1 is only relevant because these lists have a historical dimension: They are data from June of 2023, and thus allow analyzing whether traffic might contain only names that were registered longer than a year ago. It would be preferable to use up-to-date lists, but we will see later that these features did not add sufficient meaning to the descriptions to justify extra effort.

For each list l, the following features are generated:

l *Intersection* l Query Intersection Percentage: fraction of queries with 2LDs included in the list, l Name Intersection Percentage: fraction of distinct names (i.e. duplicates removed) with 2LDs included in the list

DNS2Vec Embeddings This group contains the 30 DNS2Vec embeddings as further features. DNS2Vec is explained in detail in Section 4.2.1.

DNS2Vec Features DNS2Vec i: embedding dimension i of DNS2Vec ($i \in [1, \ldots, 30]$)

Data sanitization consists only of removing outgoing traffic. One hope is that this feature set will see more use and development in the future. It is meant to be widely usable, not just for clustering IP addresses, but also to find differences in behavior over multiple days, and for any possible use case that requires traffic description, although originally being created for detection of scanners.

DNS2Vec

DNS2Vec [40] was a project at SIDN applying Word2Vec to DNS data. Word2Vec is a machine learning model used to generate embeddings of words, which are non-trivial to describe using features. It utilizes text consisting of sentences as training data. The main assumption enabling Word2Vec's learning is that words that appear in the same context in a sentence are also related in meaning. For example, the sentence "I am sleeping on a ..." could end with either one of the words "couch" or "bed". Thus, we assume couches and beds to be similar, and this is true: Both are generally large, soft pieces of furniture. On the other hand, the sentence "There is a large ... in the living room." could include "couch" or "TV", but not "bed", and so the embedding of the word "couch" will reflect that it also fits more into the scheme of leisure. Using full training sentences, embeddings are optimized so that the probability of a word occurring in a given context is estimated accurately, so as not to include "bed" in the second sentence, for example. The assumption behind this way of learning is "Words that appear in the same context must be similar in meaning". One can easily see how this can lead to accurate, nuances descriptions of words.

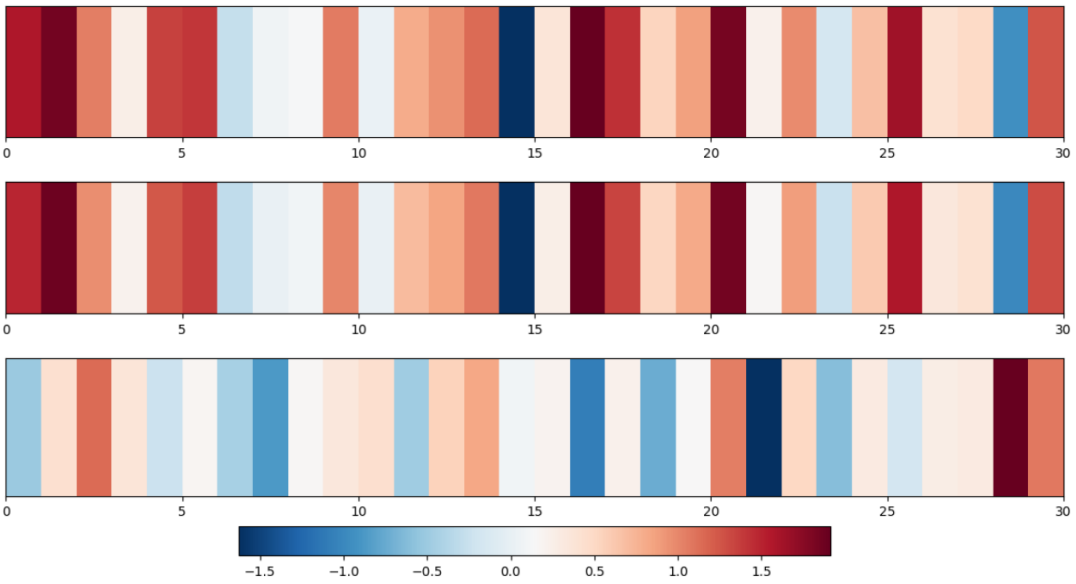


Figure 4.17: Embeddings of two similar and one unrelated resolver as embedded by DNS2Vec. Figure created using the model trained by Thymen Wabeke

DNS2Vec uses the Word2Vec model to generate embeddings for resolver IP addresses and domain names. Now, IP addresses should be described instead of words, so the sentences for learning consist of source IP addresses. Each sentence contains only IP addresses that queried a specific name, and thus have a commonality. This way, the underlying assumption indirectly translates to "Resolvers querying the same names are similar". In particular, this means that DNS2Vec only learns from domain names and IP addresses, but not from other data in a DNS query. The generated embeddings numerically represent the types of names a resolver queries, providing a similarity measure between each. This is otherwise difficult to achieve. In particular, because analyzing domain names is a task with no clear starting point. Because of this, DNS2Vec adds a representation of the specific names queried to the features of this work.

DNS2Vec also has a second part, switching the roles of resolvers and domain names to describe domain names using features.

DNS2Vec was trained only on sources with at least 15 queries over the given time frame, which should limit its use in the features. However, it only describes IP addresses, not using any other groupings, see Figure 4.1.5. Conceptually, this should be easy to change if necessary.

Expansion

Feature engineering draws its meaning from knowledge about the topic. In addition, it requires good ideas about which aspects to include and how to describe them numerically. Time constraints also limited the number of ideas that could be realized. While I hope that the feature set from this work adds useful ideas to describe resolver behavior, there are still many possibilities for additional features, even very promising ones, and implementation of them would be a worthwhile cause for future research:

Name Ordering Simply calculating the number of times that a queried domain name lies lexicographically after the last queried name. This should be true in 50 % of cases, or less if there are repetitions. But "lexicographically after" and "lexicographically before" should be similarly often if there is no order. If there is alphabetical order, all queries will satisfy only one of these conditions, so any discrepancy indicates ordering within the names.

Name Lists While 2 different name lists are included in the features, they are outdated. Other potential sources of names can be added in, including name lists from other TLDs.

Use Cases

These features bring many helpful new use cases:

1. Distance between two given resolvers can be determined in feature space. Comparing distances of resolvers from different scan operations, duplicates were successfully eliminated, that is resolvers with conclusively matching behavior but categorized as distinct campaigns.
2. Most similar sources can be found by providing an example or all currently known resolvers. This was used to find additional sources for known scans and to extend classifications to groups of resolvers. Extending groups in this way was not fully completed due to time constraints, and so it must be noted that campaigns as presented in the results may still miss some sources, therefore being larger than presented.
3. It allows determining examples of resolvers that are very different from all previously classified ones, and therefore display interesting new traits.
4. Excentricity of each resolver can be calculated by using percentiles in each feature and calculating the product. This was attempted and it was visible that scans had, generally, larger values than normal traffic, but no attempts at using this for classification have been made yet.
5. Clustering
    - Evaluating a smaller, representative set of resolvers

- Estimating proliferation of scan types
- Prototypical classification: Using already labeled data, each cluster can be given a label to achieve a very simple method of classification. Alternatively, k-NN or more complex classifiers can be used. One idea is using prototypical values for different classes of behavior, and determining whether a resolver matches the class by using cosine distance. This is similar to a Nearest-Neighbor classifier with a focus on deviation from the average.
- Clustering smaller subsets of features. While this does not facilitate classification, it might give insights into technical and behavioral aspects of resolvers in an even more explainable way. For example, clustering on features related to truncation (TC Percentage, EDNS UDP, TCP, Response Code 3) might group clusters by their handling of truncation, probably yielding a group of resolvers that does not handle truncation correctly and refuses to switch to TCP. Similarly, clustering on technical features such (ID statistics, source ports, operation codes, destination choice, repeat time statistics, ...) could give insights into what software is being used. Even smaller feature sets, combined with simple methods of classification, can be used to judge resolvers on smaller aspects of their traffic, for example correct randomization of IDs or distribution over time, to gain more abstract descriptions and find patterns/groups in those aspects. Questions that could be answered in this way include:
  - How are the query types distributed?
  - What times are requests sent?
  - What kind of names are they querying?
  - What might be their goal?

  Another example for a use case is operating system fingerprinting using IP packet's TTL values like in [1, p. 35].
- Scans can be subdivided further using the same features. In doing so, it might be possible to approximately answer the following questions using clustering:
  - What kind of scan is it?
  - Which individual campaigns can be found?
  - What might be the motivation behind the scan?
  - What software/tool is used?
  - What kind of name list is used?

  Particularly the first and second questions could already be answered in part using the clustering, but it may be possible to gain more insights on the other questions, as well, which was not done in this work.

Any way of clustering allows for grouping all traffic, without leaving anything out. In Section 4.1.5, only top-scoring resolvers by different metrics were classified, but this way, resolvers with new patterns can be found as well. Backwards analysis can then be applied to describe how well the known, manually classified data was categorized into distinct groups.

4.2.2 Clustering

Clustering was employed on the resolver features to find complete groups and classify behavior with a look at the big picture of all traffic. The manual classification work has only been done on select samples that showed suspicious behavior regarding a single feature, but clustering takes into account all features and should therefore be more robust than any method using single features.

Distinguishing scans and normal traffic is only one goal. Since resolvers from the same scanning campaign often show similar patterns in behavior, the hope is to find outliers and groups that could not be identified without, and new patterns that did not come to light before. There are multiple goals that can be optimized for, such as distinguishing between scans and non-scans or grouping campaigns.

Measures of Success

What constitutes success depends on the use case of the clustering. There are scores that can be applied to describe the fit of a clustering without regard to a particular use case or ground-truth, such as the silhouette score, the Davies-Bouldin-index and the Calinski-Harabasz-Score. These measures are therefore goal-agnostic.

On the other hand, it can be evaluated how well the clustering represents a given ground-truth grouping, creating goal-specific measures. These measures use categories, which can be:

- Scan/Non-scan
- Organzations
- Scan groups

For all three, the adjusted rand index was implemented, which uses random sampling of pairs to determine the percentage of pairs that are clustered correctly (same category $\Leftrightarrow$ same cluster), with a modification for the scan/non-scan dimension. As laid out previously in Section 4.1.5, scans are unlikely to be separable by a linear boundary from non-scans due to them exhibiting different behavior that is often extreme to both sides of a distribution. Because of this, it is not expected that the clustering can distinguish between just one cluster of scans, and one cluster containing other traffic. Instead, both groups may be distributed among different clusters, and a more modest goal is pursued:

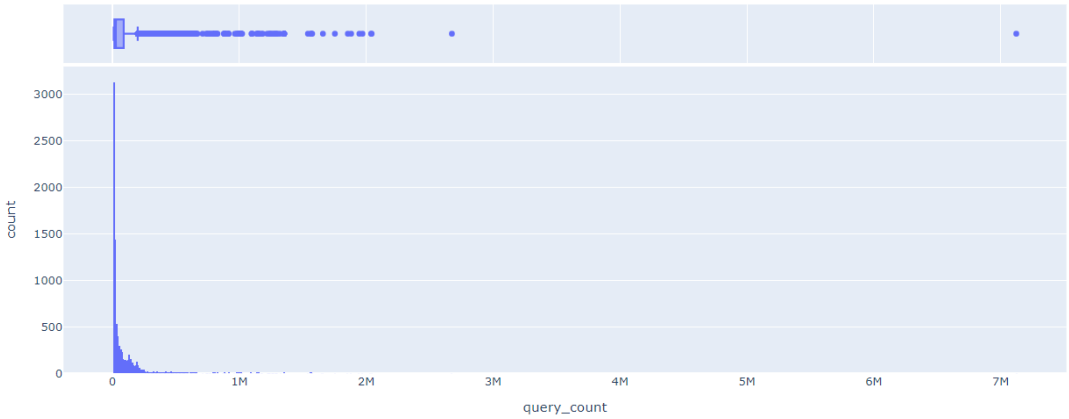


Figure 4.18: Histogram and box plot (top) of the Query Count feature before processing (random sample, 25 % of the full dataset).

No scans and non-scans should appear in the same cluster. Therefore, the rand index for scans is modified so that only those pairs are regarded as relevant where one example is a scan and the other a non-scan. It is thus not required that any two scans or two non-scans are contained in the same cluster. All rand scores are adjusted, so that a random clustering has an expected score of 0 and a perfect clustering has a guaranteed score of 1. Negative scores are also possible.

Preprocessing

The calculated features have very different domains, and some of them, like the total query count, can be arbitrarily large and are difficult to work with without scaling. The goal-agnostic measures for clustering quality suggest that cluster coherence is poor when using raw features. Taking Query Count as an example, it is easy to see why this is:

1. Query Count is on a scale of thousands and millions, while fractions, which make up many of the features, have a scale of tenths and hundredths.
2. The most active resolvers queries 17.6 million times, which is many orders of magnitude larger than the smallest sources that are still considered at 10 thousand queries. Mean, average and standard deviation are also on the order of tens of thousands of queries, making those sources extreme outliers. This can be seen in Figure 4.18

While item 1 suggests normalizing the features, item 2 shows that this would not be enough to obtain a suitable distribution. Therefore, two kinds of transformations are applied to different kinds of features:

- For features such as Query Count that can grow arbitrarily large for some sources, the logarithm is applied to achieve a more useful description of the resolver activity. Otherwise, large numbers will dominate the feature distribution. This transformation is applied to the columns Query Count, Distinct Count, Bogus Count, Per Name Query Count Average/Deviation and Count Over Short Time Frame Max/Min/Difference.
- For percentages, the value distances ought not to be regarded as uniform. For example, there is, logically, little difference between two sources that ask for 45 % and 50 % distinct domain names respectively, while the difference between 5 % and 0 % means huge differences is behavior. For this reason, percentages are stretched by applying the tangent, so that values close to 0 or 1 lead to larger differences.

Logarithm and tangent are undefined for extreme values, so tiny changes are used to deal with extreme spikes. This affects value 0 for the logarithm and percentages 0 and 1 for the tangent. The exact calculations are:

$$\begin{aligned} c' &= \ln(c + \varepsilon) && \text{for counts } c \\ p' &= \tan((p - 0.5) \cdot \pi \cdot (1 - \varepsilon)) && \text{for percentages } p \end{aligned}$$

With $\varepsilon = \frac{1}{100}$.

Afterwards, all columns are normalized, except for the percentages and binary columns. Percentages and binary columns are only centered.

These columns have a limited domain, and some of them have immensely tiny standard deviation, such as rare countries or query types that see only a dozen queries per day. Dividing those features by their standard deviation would exaggerate their relevance in the features and decreases clustering performance in each measure, even when using a lower limit for the standard deviation to divide by. I hypothesize this is because the standard deviation of these features already implies their relevance and is best kept intact. It follows that Query Type A Percentage is a more relevant feature than Query Type SSHFP, which is very rare. This suits the clustering's goals nicely.

One might argue that usage of rare query types is a sign of a resolver being a large public resolver and therefore relevant for classification. An it is true that rare query types are often seen in large open resolvers, because they handle organic queries that may ask for any type of record, but:

- Asking for 0% SSHFP query type, for example, is non-informative, meaning such a resolver could still be part of a scan or not.
- Clustering open resolvers is not the goal. In fact, many public resolvers contain scans, and many even send little traffic except for scans. When trying to classify open resolvers however, a feature encompassing multiple rare query types seems like a promising idea.

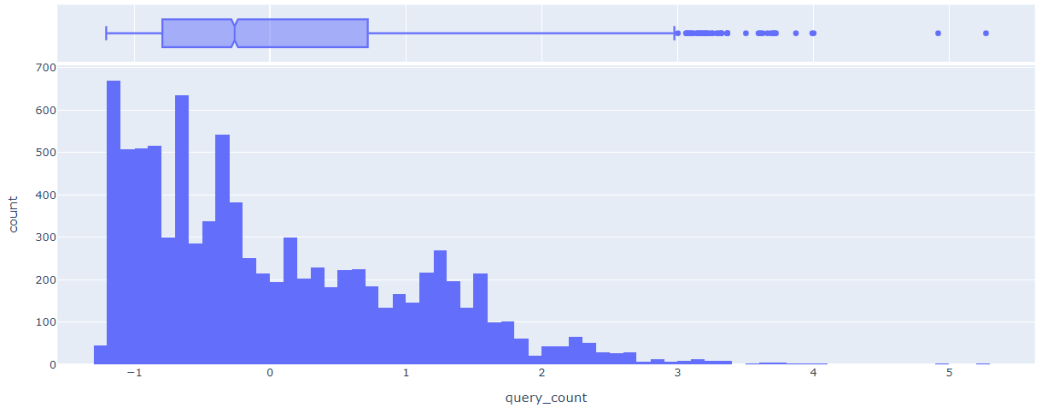


Figure 4.19: Histogram and box plot of Query Count after preprocessing, featuring a more even distribution with fewer outliers.

Algorithm

Multiple algorithms were tested: 1. k-Means for its capabilities in anomaly detection and finding clusters even without obvious spaced in between 2. DBSCAN for its ability to follow clusters with arbitrary shapes density-based and extract outliers 3. Mean-Shift for the ability to estimate the number of clusters without manual input 4. agglomerative Clustering for its hierarchical clustering, which fits the structure of the problem well, which has rougher scan types and smaller campaign within each type.

Although all approaches seem appropriate for the use case, the well-established k-Means algorithm also proved best for this use case, producing already better-than-random results without further processing with only 16 clusters, with very good grouping when using $k = 128$. For each algorithm, and for k-Means in particular, samples that are further apart in any dimension are less likely to grouped together in the same cluster, which will be relevant for the next step.

Weights

Further preprocessing includes weighting each feature by multiplying its values with a constant. This is done because 1. some features are more relevant than others, and 2. features may contain interdependencies.

Relevance For example, if two different sources have distinct percentages 0 and 1, respectively, they are certainly completely different types of scans. On the other hand, RD fractions of 0 and 1 would just hint at differences (errors) in software or configuration, and could very well be clustered together. Nevertheless, both attributes are relevant for

the cause. Intuitively, sources that exhibit an unusual distinct percentage should be more likely to be put into a separate cluster than such ones with an unusual RD percentage.

Interdependencies Some features are also interdependent: For example some query types like A and AAAA cause more large answers and thus more truncation [26]. Other features are interdependent in more intricate ways, for example No Response Percentage and Truncation Percentage should generally increase TCP Percentage, and Punycode Percentage goes along with increased Response Code 3 Percentage because Punycode is not allowed in the NL zone. This implies that there are redundancies within the features, essentially doubling the influence of some. These interdependencies are too complex be removed manually, and using Principal Component Analysis or other forms of dimensionality reduction would reduce explainability of the results, because distances are then calculated on transformed data instead of original features.

Giving each feature a relevance is therefore necessary to obtain a good clustering result. Feature weights can also be used to set different focuses and are goal-oriented, and these ones were created to distinguish between scan sources and normal sources with a high accuracy. Distinguishing between scan groups or scan campaigns can presumably be done with similar weights, but is a slightly different task.

The feature weights used are shown in Table 4.1. They were obtained by manually estimating each feature's relevance while optimizing for the adjusted rand index of scans/non-scans. For each feature, it was checked that moving it up or down in relevance would not significantly improve the score, and no changes were accepted that went against common sense.

The results indicate that bogus queries are not an indication of scanning, although first assumed to be.

Feature Distribution Analyzing the distribution of these features on the previously collected ground-truth dataset, differences show between scans and non-scans. Figure 4.20 shows that found scans have a much higher variance in their Per Name Query Count Deviation than the base population, and that the wide population has a larger variance than normal traffic. This can easily be explained, as the query counts per name will follow a long-tailed distribution for normal traffic. Scans often query each name just once or a fixed number of times, or query a small set of names or just one name very often, which leads to small or large values in this feature.

Of course, the groups were not created from random samples and might be biased towards extreme values themselves, but similar patterns can be found in most features, including those that were not used for sampling during manual classification. This even includes very simple features, such as the starting letters of queried names, or domain name length.

Most relevant features ($w = 16$)
Distinct Percentage

Highly relevant features ($w = 8$)
Response Code 0 Percentage
No Response Percentage
Repeat Percentage
Domain Name Length Average

Very relevant features ($w = 4$)
Query Type x Percentage (for all x)
Pct. Over Short Time Frame Min.
Pct. Over Short Time Frame Diff.
Name Starts With x Pct. (for all x)
Per Name Query Count Average
Per Name Query Count Deviation
Recursion Desired Pct.
Response Code 3 Pct
Domain Name Length Deviation
Pct. Over Short Time Frame Dev.

Relevant features ($w = 2$)
ID Average
TC Percentage
CD Percentage
Source Port Minimum
Punycode Percentage
Time Frame
TCP Percentage
Repeat Time Average
Pct. over Short Time Frame Maximum
Per Name Query Percentage Average
Count Over Short Time Frame Quotient

Less relevant features ($w = 1$)
Repeat Time Deviation
Destination x Percentage
Time Deviation
Time Min
Time Max
Bogus Percentage

Less relevant features (continuation)
ID Minimum
AA Percentage
Source Port Average
Source Port Deviation
Response Code Other Percentage
Operation Code x Percentage
Inter Query Time Deviation
Time Average
EDNS UDP Variability

Irrelevant features ($w = 0$)
AD Percentage
Distinct Count
Bogus Count
Z Percentage
EDNS UDP Average
EDNS UDP Minimum
EDNS UDP Maximum
Query Class x Percentage
ID Deviation
Query Intersection x Percentage
Name Intersection x Percentage
Highest Count Time
Lowest Count Time
Label Count x Percentage
Labels More Than 9 Percentage
Per Name Query Percentage Deviation
Per Name Query Percentage Minimum
Per Name Query Percentage Maximum
Query Count
Country Is x
Count Over Short Time Frame Dev.
DNS2Vec Dimension x
Count Over Short Time Frame Max
Count Over Short Time Fram Min
Count Over Short Time Frame Difference
TTL Average
IPv6 Percentage
RA Percentage

Table 4.1: Weights used for each feature

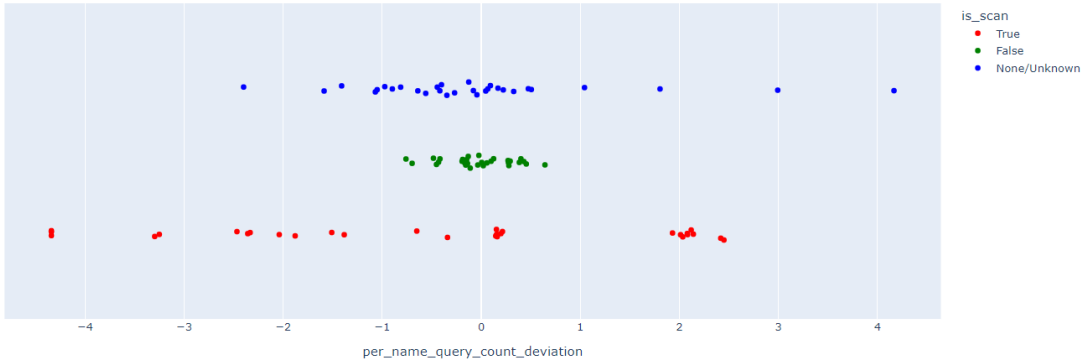


Figure 4.20: Strip plot of feature Per Name Query Count Deviation for each resolver, grouped by scan/non-scan/unlabeled, with 40 samples per group.

4.2.3 Evaluation

To gain an unbiased evaluation of the clustering approach, it was applied on feature vectors from the 3rd of July using the same parameters described above. The algorithm was run 10 times, overwriting the result whenever the alternative is better in all goal-agnostic scores. The resulting grouping was then sampled to gain 10 resolvers per group. These samples were manually classified by whether a scan was visible in the traffic with some being inconclusive. The fraction of traffic from scans was estimated for each sample labeled "contains scan", again with some samples being impossible to estimate reliably.

4.2.4 Results

The goal-agnostic measures deem the quality of the clustering to be rather poor, meaning that many examples exist that are rather far from cluster centers and clusters are close to each other, not yielding nice, empty cluster borders. Little can be done about this apart from proper preprocessing, as this aspect lies in the nature of the dataset. These scores were used to determine hyperparameters, that is the number of clusters to use, and showed that around 16 and 128 clusters provide a good fit for the dataset. Other than this, they are not very actionable.

128 clusters were used at first, and improved weights then made it possible to achieve a good accuracy with only 17 clusters, as well.

Even with difficult-to-obtain features such as DNS2Vec embeddings excluded and a significantly reduced feature count, a modified rand index of 0.97 can be achieved for the scan-label on the gathered data, implying a classification accuracy of 98% and an F1 score of 99% on the collected data. On top, the same configuration achieves a rand index of 0.60% on scan campaign labels and 0.30 on organizations, even though these have large counts that cannot possibly be fit perfectly in 16 clusters.

Cluster	Sources	Queries	Org. in sample	Scan Fraction Avg.	SF Dev.
0	116	5,647,404	10	1	0.00
1	4,709	263,395,848	9	0,02	0.03
2	2,356	250,275,650	9	0,81	0.45
3	6,657	536,579,248	4	0,13	0.12
4	1,295	90,764,917	4	0,99	0.02
5	3,482	133,521,306	2	0,29	0.44
6	1,641	108,307,386	2	0,42	0.33
7	5,126	653,470,176	8	0,3	0.40
8	6,077	161,719,690	5	0,17	0.26
9	2,479	157,053,701	8	0,84	0.37
10	3,15	113,962,102	5	1	0.00
11	361	20,316,434	1	1	0.00
12	4,327	226,155,323	7	0,2	0.18
13	415	33,309,089	9	0,93	0.08
14	1,639	49,406,304	3	0,73	0.28
15	2,384	235,991,782	4	0,31	0.22
16	1,493	56,439,095	1	0,37	0.23

Table 4.2: Results of manual classification of clustering of July 3rd

While the measures describing clustering fit are still rather poor, the cause of this might be the large dataset (around 50.000 resolvers) and the diverse behavior, which could better be described with a large number of clusters, but this would be infeasible to manually classify. Even without weighting the features, good clustering can be achieved, which divides scans much better than random. This proves the importance of the preprocessing of the features. Increasing the number of clusters leads to even better results, with rand indexes for scans and groups increasing further to give an almost perfect distinction of scans and non-scans in the labeled data, and the measures for clustering fit improving as well, and 128 clusters being a more precise fit, if not infeasibly large.

One particular strength of the features is finding similar resolvers. Using the features not for clustering, but instead finding the most similar resolvers to a given example, yields very good results. In this way, other sources from the same scan campaign can easily be identified due to their similar features. The closest resolvers often form a group that can easily be checked to have consistent behavior.

Evaluation

170 sources, 10 random sources from each cluster, were manually classified. Table 4.2 shows statistics of each cluster and the labeled samples from it. The clusters have a varying amount of diversity, with some samples containing resolvers from one single company (11, 16) and others being very diverse (0, 1, 2, 13). More importantly, while

resolvers from some clusters were very clear to classify (0, 10, 11), other clusters show large variety in the contained resolvers and the amount of scanning within them (2, 5, 9).

This goes to show that while some scans can be easily found using the clustering, others are not yet reliably distinguished from other traffic.

5 Results

5.1 Statistics

Figure 5.1 presents the statistics of all labeled and unlabeled traffic from large sources on April 3rd. More than 10 % of the day's traffic was labeled by the end of this study. Of course, proportions of scan and non-scan traffic from this chart are not representative. Still, about 10 % of traffic was confirmed to be generated by scanning.

Evaluation on data from 3rd of July shows that clustering is well able to find scans, with some very heterogeneous clusters, still. Picking examples from each cluster can help with finding outliers among resolvers, and presents various scans condensed into few samples. Taking the average scan fraction seen in samples for each cluster and extrapolating this to the full traffic of each cluster yields that around 30 % of queries are from some kind of scanning on July 3rd, with the true value very probably lying between 20 and 40 %.

Around 45 different operations could be found by grouping analyzed sources according to similar behavior. The smallest operations found scan using just a single IP address, performing 32783 queries within a day. The largest operations include a subdomain scanning operation using open resolvers, and a 2LD scanning operation running in the network of a German hosting provider.

In general, many scans are being performed from the networks of hosting providers. This accounts for at least 1/3 of the source IP addresses. While some, presumably amateur ones, scan from just a single address within a network, there are also very large operations.

5.2 Clustering

The same clustering method was applied on data from the 3rd of July, and 10 samples were taken from each of the 17 clusters and evaluated to ensure that accuracy was close to the 97 % determined on the potentially biased data from the 3rd of April. This also allows comparison between the two days.

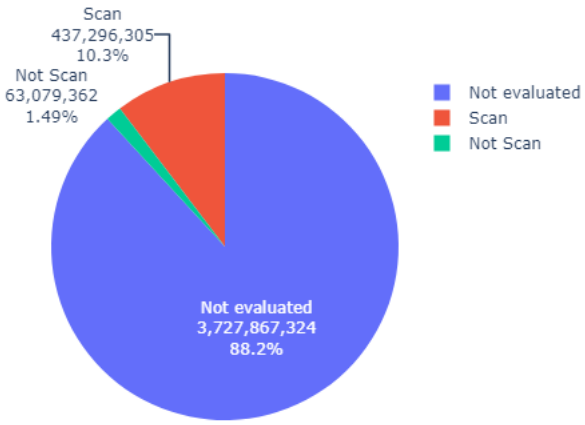
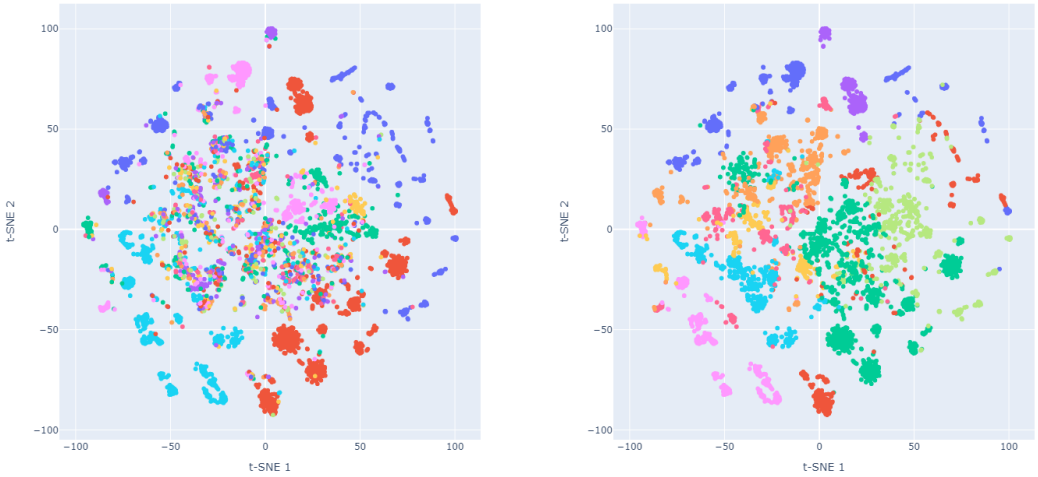


Figure 5.1: Proportions of traffic labeled (excluding traffic from sources with <10k queries).



(a) color is AS organization

(b) color is cluster

Figure 5.2: Visualizations of feature space for the 3rd of July using t-SNE with reduction to 2 dimensions (10k samples)

5.3 Scans

This section includes descriptions of some of the most prominent scans found during this research, plus descriptions of scans that are well-known and could not be found in the traffic. Scans can be distinguished by their type or goal, for example whether they focus on 2LDs or subdomains. This chapter is structured to distinguish between scans that use dedicated IP addresses versus such ones that use presumably open resolvers, i.e. that are mixed with non-scan traffic. This has implications for finding and describing them, and makes some aspects of them escape analysis.

In total, at least 62 different scanning operations could be found and confirmed manually spanning at least 498 resolvers. Each operation not having a canonical name such as OpenINTEL or Censys was assigned a random English word as a code name to facilitate description and referencing.

5.3.1 Dedicated Sources

2LDs

Unanimous One large dedicated operation comes from Sweden, from around 210 IP addresses from three different hosting providers. They all exhibit very similar behavior and can be found by their 100 % distinct domain names, up to 48 % unanswered queries per IP (due to rate limiting), sheer count of distinct domain names.

But their similarity goes much further, making these sources very easy to distinguish from others:

- Queries are distributed evenly over the whole day
- NX percentage is remarkably high at 99 %
- Too many names of length 35 are being queried, which are randomly generated strings
- Punycode is used significantly often
- NS3 is ignored, NS1 and NS4 are queried evenly
- Many names contain English words, rather than Dutch
- Only one single ID and one source port are used for all queries, and those are the same for all IP addresses
- All queries contain exactly 2 labels

Figure 5.3: Only scanning sources visualized in feature space using t-SNE. The more queries sent, the larger the dot.

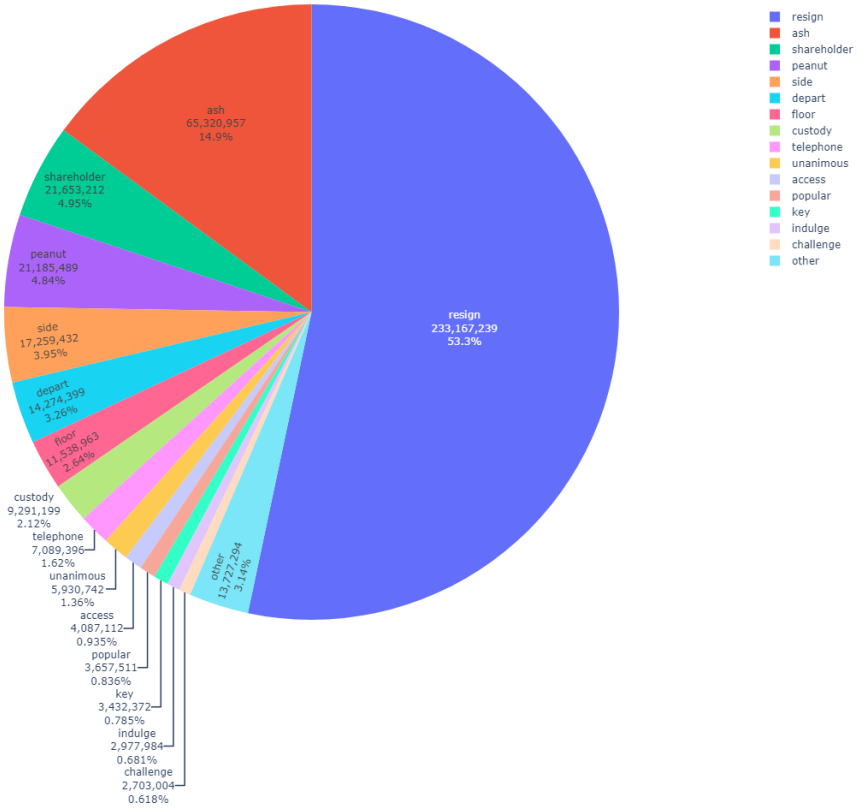


Figure 5.4: Traffic volume of each scanning operation among all scan traffic.

Because it is scanning only 2LDs without their corresponding name servers, this operation might be typical *domaining*, that is gathering information about available 2LDs. The prevalence of English in the names and high fraction of NX domain names indicate that the list of names being scanned might come from other TLD's zone files instead of the NL zone.

This scan sends around 190k queries per source, or around 40 million queries in total, making it one of the largest operations. 99 % of all queries receive response code NX domain, so the total number of identified existing domain names is rather small at less than 1 million, showing how inefficient this method of scanning is. The other attributes also support the hypothesis that not much effort has been put into optimizing the scan, with traffic constantly running into rate limiting and resolver behavior being as distinct as it can get. On the 3rd of July, this campaign could not be found, meaning it has probably stopped running.

Ash Another large operation was found originating from four IP addresses of a large German hosting provider. These are characterized by a high query count of 17 million per IP per day, around 75 % coverage of the NL zone, and a high query rate of almost 1 million queries within less than an hour, per source.

Other abnormalities include:

- Different qtypes are queried during different times
- Name query order is alphabetical at times
- *. subdomains are queried
- More than 2 labels are queried regularly

Telephone (OpenINTEL) As mentioned on their website `https://openintel.nl/problems/`, OpenINTEL uses fixed IP prefixes. Given one IP address as a starting point, the sources most similar in feature vectors mainly fall into one IPv4 /24 prefix and one IPv6 /116 prefix. Both prefixes send around 8 million queries each and cover around 100 % of all NL names. This is to be expected, because the NL zone file is shared with the project.

Abnormalities are:

- Querying seems to happen in two operations that overlap for a few hours during the day
- TLSA, AAAA, A and SOA records are queried, SOA is only queried in the first operation, TLSA only in the second, A/AAAA during both
- The TLSA scan contains more NX domains than the SOA one
- Average length of domain names is longer than for normal traffic

- Many queries contain more than 2 labels, often up to 5
- Only names of name servers are queried more than once per query type
- Around 3 queries per 2LD can be seen

Even with OpenINTEL, there is still potential for optimization. Records could be cached so that TLD name servers do not have to be queried again for each query type. The fact that only 3 queries reach our server per name, instead of the 11/12 mentioned by the website, shows that some caching is already done. Traffic is spread out well over the day, ending at some point in the evening, but nearly double during the time where both operations overlap, which is during the day, when querying is at its busiest from other sides as well.

5.3.2 Shared Resolvers

Subdomains

Resign The most significant scan using open resolvers concerns subdomains of a few selected 2LDs. It affects a large fraction of all open resolvers. Mentionably, out of 675 resolvers checked in total (with an unknown number being open resolvers), 90 were presumably open resolvers showing signs of this scan. It is most noticable in at least 71 resolvers from Cloudflare, whose traffic consists mainly of this scan. Because Cloudflare resolvers diligently minimize qnames, the traffic from this scan can seem as DS or A queries for a single name being repeated relentlessly, multiple times each millisecond at times. Other resolvers affected show that the names queried all contain more than 2 labels, with 3LDs and further apparently randomly chosen from a list of common subdomains like www, ftp, mail and so on.

These queries lead to a low NX domain fraction, a low distinct domain name fraction, and a heavily skewed distribution of starting letters, query types, and label counts (if qnames are not minimized).

The exact purpose of the scan is unknown. Subdomain scanning can be a part of penetration testing, which could also explain the limited number of names. Some of the 2LDs involved are using wildcard records, which could imply that software was used for subdomain scanning which cannot recognize wildcard records and kept querying for this reason. The traffic sums up to around 250 million queries, meaning this scan makes up 4 % of all traffic. One of the names appears in the top 50k domain names on the Tranco list of top sites for this day, and the other queried names also appear, even though most do not offer a website and are not popular enough to appear on the Tranco list. This goes to show that DNS scanning can also manipulate top site rankings.

This scan could be found on data from the 3rd of April. Subdomain scanning is still prevalent on the 3rd of July, and multiple names can be seen queried over a thousand times by seemingly open resolvers too. Some of the names are even the same as on April

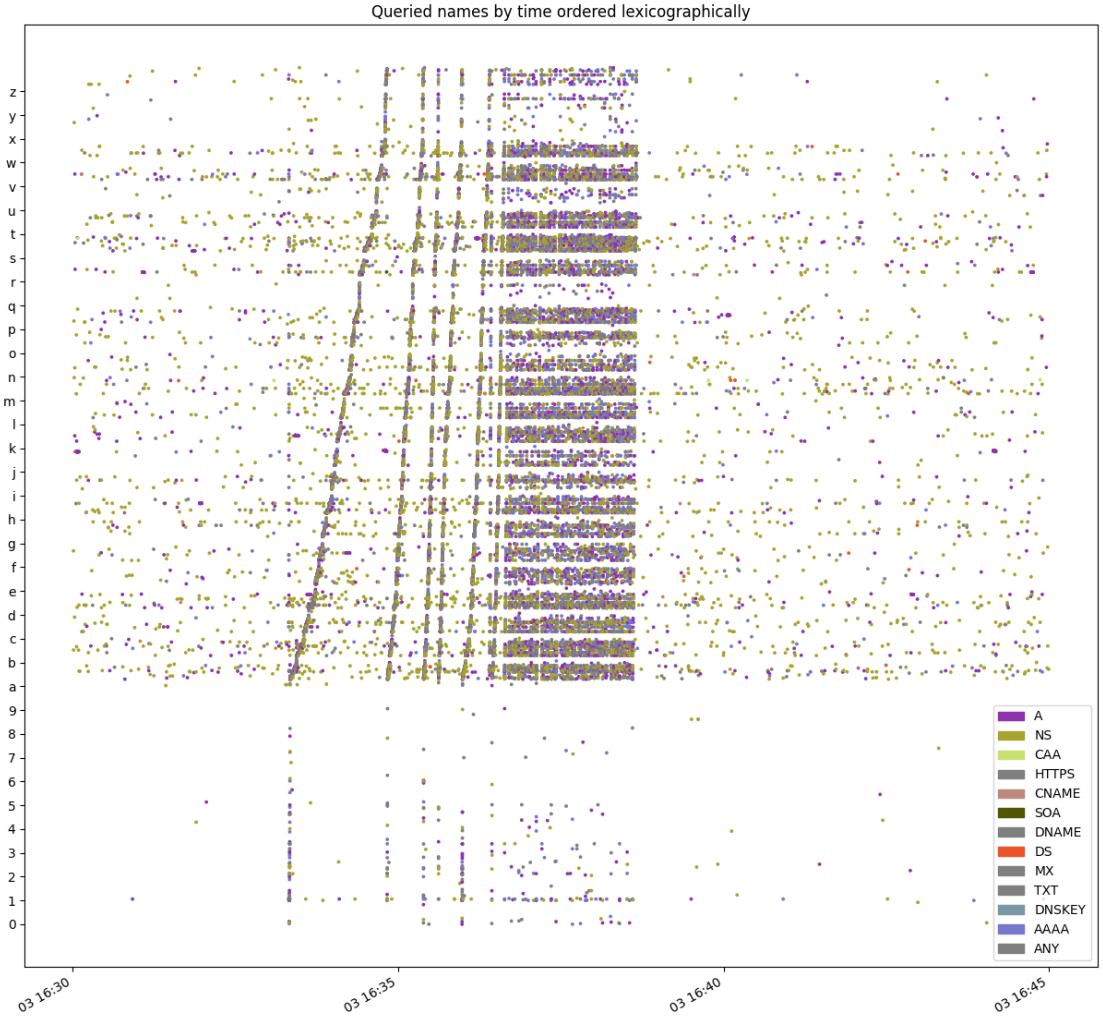


Figure 5.5: Alphabetic positions of names queried in operation *cream*.

3rd, but with far fewer queries. No names seem to be queried to the same extent as on April 3rd, reducing traffic from subdomain scanning significantly. This is probably part of the reason for the smaller query count on July 3rd in comparison to April 3rd.

2LDs

Cream Another scanning operation affects many public resolvers during a short 5-minute time frame on April 3rd. It is visible in multiple public resolvers from Google, causing a large spike in traffic as seen in Figure 4.16.

Different record types are queried for names in alphabetical order (Figure 5.5). These are mainly existing names, and the query volume causes some be left unanswered.

This operation could most easily be found through the large number of distinct domain names queried within a short time frame and the large discrepancy in query volume, increasing 15-fold for some resolvers at the time of the scan.

These scans highlight the issue that the existence of millions of open recursive DNS resolvers poses [18].

5.3.3 Name Similarity

Another unexpected type of scan could be found both in open resolvers and also from dedicated sources. A subtype of 2LD scanning, a few operations scan for names that are similar to a predefined name, often by changing or adding just one symbol. These scans often exhibit punycode, high non-existent (NX) domain fractions and heavily skewed alphabetic distributions, making them easy to find.

One source of such a scan belonged to the network of a trademark protection company. Similar methods may be used to find suitable phishing domain names.

5.3.4 Not Found

Censys All prefixes mentioned by Censys's website were examined, but none could be found to send a significant amount of traffic during the 3rd of April, or the week after.

Web Crawling Even though some resolvers could clearly be recognized as mail servers, no clear examples of web crawling could be found. This is probably due to web crawling not being easily classifiable - it might show signs of normal behavior and of scanning.

5.4 Case Studies

This section will explain some analyses performed that go beyond the common theme and the standardized methodology described previously. Two operations were looked at in more detail to better understand their origins and effects.

5.4.1 Subdomain Scanning in Public Resolvers

Scanning campaign *resign* was already described in Paragraph 5.3.2. Clearly, a list of open resolvers was used that goes beyond the largest public resolvers, and a list of plausible subdomains. Both of these are provided by MassDNS, and the MassDNS-list was found to contain IP addresses that send queries of this scan, although it cannot be assumed that front end IP addresses correspond to back end IP addresses in all cases. Abusing public resolvers for large scans such as this one is bad practice and

wastes resources of all parties involved, especially using potentially broken software. On the other hand, it is also unclear why the recursive resolvers queried the .nl name servers for each query. Any records returned by our servers should be cached and not be queried again, because the authoritative name servers for the 2LDs will not change in between each query. Other public resolvers, such as Google's, also contain traffic from this scan, but are not as swamped by it as others. They might implement rate limiting or blacklisting for IP addresses sending too many queries in a short time. As a final note, there are many reasons why these queries are redundant. It is unclear why SIDN's name servers should receive all of this traffic, and different implementation in one of many places could have prevented a significant amount of traffic.

The name servers exhibiting this behavior are spread all over the world, but many of the public resolvers are in Europe. This is a useful hint, because for CloudFlare's resolvers, the geographic position of the resolver is determined by IP Anycast, and cannot be set by the user. Digging deeper, small resolvers can be found that provide EDNS client subnet information for this traffic. Ironically, it is a small resolver from Germany that leaks the IP range and company that is the source of the traffic. This way, it could be confirmed that subdomains of a list of 33 domain names were being scanned for from the network of a Turkish hosting provider, with high likelihood. It can be assumed that almost any traffic about these names is part of the scan, because none of them are popular names, with many of them not having a website.

5.4.2 21st of May

On the 21st of May, the AuthNSes faced an unusually large amount of traffic, leading to distortions in almost all statistics on https://stats.sidnlabs.nl/nl/dns.html (query types, response codes, public resolvers, IPv4, TCP, 2 labels). The number of different NX domain names queried reached 322 million, in comparison to 77 and 78 million on the day before and after, respectively; and only 57 % of all queries were responded to with response code `NOERROR`, compared to 91 and 92 %.

Searching for traffic fitting exactly the unusual statistics (response code NX, no public resolver, IPv4, TCP, 2 labels) yields only 8.6 million queries out of 4.9 billion queries on this day, which is too few to cause the visible disturbances. However, comparing activity of *all* sources on the 21st of May and the day before and after yields 25 IP addresses of resolvers with significant activity on the 21st, but little activity on the days before and after. All of these resolvers send between 10 and 164 million queries on the 21st, but no more than 300 thousand (some 0) on the 20th. Also, all stem from the same hosting/cloud provider, and exist within a few IP prefixes.

Characteristics Analyzing the traffic closer with the tools developed for Section 4.1.5 additionally yields the following patterns:

- A suspiciously even distribution of queries over time

- Mostly NX queries
- 8 different query types, each used for exactly one eighth of the traffic, of which 3 are rare (SVCB, HTTPS, SRV)
- a slightly unusual alphabetic distribution, with higher digits being used relatively often and some letters being used too often (a more than b, t more than s), unlike other traffic
- rather long name lengths, with the peak of the distribution at 14 and no names shorter than 5 characters (excluding .nl)
- 24 % unanswered queries due to rate limiting
- usage of TCP for around 20 % of queries, which indicates a correct reaction to the truncation bit being set due to rate limiting
- 2,639,599,129 queries, which is 54 % of the day's traffic
- Few source ports being used for multiple millions of queries, thus not being randomized correctly
- 76 % NX domain queries, with only one out of 4000 queries asking for an existing name
- A few packets with inexplicable size and field values, which are too large to be correct DNS packets. It is thought that these were incorrectly parsed by ENTRADA and might either be multiple packets or different protocols. The nature of these packets is uncertain.
- Correct randomization of IDs, almost all queries containing exactly 2 labels, EDNS UDP of 1232 for every query, no alphabetic order or visible patterns in names or query types

This is undoubtably a large one-time scanning operation of huge proportions. Given its unusual size, it can also be used as a real-world benchmark of the AuthNS infrastructure. After explaining one possible motivation for the scan, processing times as also saved by ENTRADA will be examined to determine the impact.

Motivation Even with 2.6 billion queries, only about 2.5 % of .nl names could be guessed. We believe there might be a different goal to this scan, such as NSEC walking, which requires guessing names with specific hashes and would allow cracking of the acquired hashes locally thereafter. The hardware available from the hosting provider would presumably allow efficient cracking of hashes. This does not require multiple query types, however, and it is only a guess, with no evidence to confirm it. This hypothesis could be checked in the future by calculating the actual hashes of the domain names to compare them against NSEC records. For now, the intention of this scan remains unknown. Given that it originated from cloud infrastructure, costs for running

it might have been high, further substantiating the assumption that there must have been a worthwhile goal.

Impact The 2.6 billion queries were routed to only 3 Anycast sites, given that all sources were geographically close to each other (with a negligible amount of 3.4 thousand queries going to a fourth site due to changes in routing or other technical reasons). NS1 received the most queries at 1.2 billion, and 0.98 billion and 0.47 billion queries were received by the other sites, respectively.

Response times on this day saw changes due to the scan. For non-scan traffic, the standard deviation of the three average response times on 20th, 21st and 22nd of May was no more than 0.2 milliseconds except for all except for 3 smaller sites (less than 50 million queries each during all three days in total). For scan traffic on the affected sites, the deviation is 0.7, 0.33 and 6.52 milliseconds, each, with response times much higher than the average of other days, rising to over 11 milliseconds. The goal should be, however, to answer non-scan traffic reliably. Looking at average response times for traffic not stemming from the scan for each site, this sharp rise does not show. No site had a rise in average response time larger than 130 %, or 0.13 milliseconds. This should barely be noticeable by clients, as network latency is expected to be 1 or 2 orders of magnitude larger. One Anycast site responded to normal traffic (traffic except for the scan) twice as fast an on other days, with an average of 0.07 milliseconds instead of 0.14 on the 20th and 22nd. Neither I nor any colleague at SIDN had an idea as to why this might have happened, it completely escapes explanation.

What is certain is that the rise in response time on the 21st of May, if there even was one for a particular site, was unnoticeable in any case. This is even though Anycast sites experienced 3 to 9 times the typical daily traffic volume, traffic was only distributed among name server IP addresses, but not Anycast sites. This goes to show that even extraordinarily large scanning operations - as large as 40 % of normal daily traffic volume - do not pose a risk to the availability of the authoritative name servers, as measured increases in response time are insignificant.

Redundancy This being said, this operation also exhibited some of the signs which increase traffic unnecessarily:

- Even NX domain were queried multiple times with different query types
- Some queries were even sent multiple times using the same query type
- Rate limiting was faced throughout the whole time, so that a significant amount of queries were sent that were left unanswered

Using more effective caching, especially negative caching, NX domains would only need to be queried once to confirm that no record exists. Sending multiple identical queries causes more traffic for little advantage, and running into rate limiting so significantly

diminishes effectiveness greatly. On top, almost all names queried were non-existent. Summarizing all of this, much traffic was generated for little information gain, and the same data could have been collected much more efficiently.

6 Discussion

Through this work, it has become clear that most scans show clear abnormalities when compared against normal traffic and are identifiable via analysis of the query behavior. Using different measures ensures that more campaigns can be found. Sophisticated features are not necessary, since even with an easily calculable feature set, resolvers can be grouped well and accurately, achieving good scores for cluster purity. To confirm scans as such, many kinds of visualizations can help make otherwise verbose patterns understandable to the human eye.

The clearest giveaway of scans to the human eye is alphabetical query behavior, although many different patterns give insight into the nature of traffic. Different kinds of visualization help tremendously with manual classification, but cannot be applied to the whole dataset. Instead, sources have to be classified individually.

The most relevant patterns concern features that have a clear domain for normal traffic, such as response codes, query types and distinct percentage. Behavior from scans is less organic and deviates from the norm to one of both directions. For most scans, this is the case for multiple different features, making them more excentric than normal traffic.

Scanners like SIDN's Dmap, ones from the TU Munich could be found easily, and a more complete picture emerged through clustering. Some operations, like Censys' DNS scans, could not be confirmed. This might be due to the scanning period: OpenINTEL could be found as it performs measurements daily, and Dmap could also be found because it scans in the beginning of each month for multiple days. For others, this might not be the case.

The scans do not pose a risk to the availability of SIDN's name server infrastructure. Even though more than 20 scans have been found sending more than a million queries in a day, and even though the total traffic from these scans is at least 3 %, and most likely an order of magnitude larger, it can also be seen that most of them have queries that remain unanswered due to rate-limiting. For very blunt query behavior, the percentage of unanswered queries can grow arbitrarily large, while regular resolvers do not usually find any queries to be unanswered. Even the large amount of excess traffic on May 21st did not lead to an unacceptable increase in the processing time of queries for the 3 affected sites. Therefore, I conclude that the current infrastructure is well capable of handling the traffic and more, and that rate limiting should be effective at mitigation of unsolicited traffic, should need every arise.

The largest identified group of domain scanners achieved at least 71 % coverage of the NL zone's 2LDs. Large public DNS companies such as Google (responsible for about 10 % of the total traffic) are well capable of creating a nearly complete list, as their traffic includes very close to 100 % of all registered names.

Through clustering, it can be assumed that no very obvious examples of scans have been left unseen, but a small number of less obvious examples with mostly normal behavior might be hiding in the clusters. Of course, the features were chosen based on the patterns seen during manual classification, and many were added to see whether they can help. It is possible that other features could help with finding specific sorts of scans that completely escaped this work's attention.

The results and methods of this work can be expected to generalize well to other TLDs, and the analysis methods are easily expansible and insights-oriented, allowing detection of patterns that have not occurred in the dataset and providing elaborate output.

The topic of DNS scanning highlights the openness of the internet: It is possible for anyone to set up a website, a DNS server, a resolver or a scanning operation. Any single device could send enough queries in a day to be prominent in the data.

6.1 Ethical Considerations

The original motivation for this research is judging the impact of scans on DNS infrastructure. Although a large portion of traffic originates from scans, there are no signs of any performance deterioration due to it. The descriptions of resolver behavior using features and visualizations try to abstract from any sensitive information as much as possible. However, they describe resolver behavior more broadly than for example [32], and DNS data is privacy-sensitive. The features might as well allow distinguishing between mail servers and user-facing resolvers, just for example.

Any use of the tools from this work must be for a solely academic purpose or for improving or protecting DNS infrastructure, and it may never be used to discriminate against any individuals or limit any sort of traffic.

6.2 Actionable Advice

6.2.1 For Authoritative Name Server Operators

From the traffic analyzed, it becomes clear that unanswered queries are a clear sign of scanning operations occurring and that normal traffic does not run into rate limiting under normal circumstances. Visualizations like Figure 4.11 and Figure 7.3 show clearly that even among mixed traffic from a single source, rate limiting is precise at eliminating unwanted scanning traffic, while handling other traffic from the same source correctly.

6.2.2 For Open Resolver Operators

Repeated queries were seen from resolvers that were affected by 3LD enumeration. Clearly, most of these were unnecessary and could be mitigated by proper caching, or by rate limiting clients.

6.2.3 For Scanners (and those interested)

Scanning traffic is seemingly not easy to hide, especially when using dedicated IP addresses. An important step is limiting the number of queries that are sent within a short time frame and spreading traffic over multiple hours in order not to unnecessarily burden name servers. This also goes hand-in-hand with avoiding rate limiting. Also, basic conventions for valid query format should be followed, including not asking authoritative name servers for recursion and not asking for entries of the root zone at TLD servers.

6.2.4 For Hosters

Since most of the found scanning operations stem from the networks of hosters or cloud providers, their infrastructure plays a large role in enabling scanning. If scans are deemed undesired, rate limiting outgoing DNS queries might prove effective in reducing both scans and other malicious traffic.

6.3 Limitations

During optimization of the clustering, it was apparent that monitoring, with a very regular query schedule, showed similarity to some scans, making a distinction between them difficult. It is difficult to come up with an actionable definition of what constitutes a scan, and a top-down approach of looking at outliers from normal traffic seems more fruitful.

Additionally, while various operations and different strategies were found, there is no guarantee for completeness.

Similarly, scans that use public resolvers and such ones that are limited in time are more difficult to find and describe.

6.4 Future Work and Open Questions

This research leaves many directions of expansion to be covered by future work.

- While many aspects of resolver behavior have been examined, there are still countless dimension left unaddressed. Especially the feature vectors only cover most patterns in a basic way (e.g. average and variance of source ports). Behavior of resolvers can differ in lots of ways, and this work only covered a few that were most appropriate given the given data format and computational capabilities. It should be easy to examine the data in new ways, and to add more features to the calculation for consideration by the clustering algorithm, or to try out different algorithms. This includes other kinds of clustering algorithms, but also classifiers such as neural networks or decision trees.
- This work covered the NL zone. While it is reasonable to expect other non-public ccTLDs to see similar behavior from scanners, no analysis has been done to confirm this. Depending on similarities in the infrastructure such as whether they also use ENTRADA and Spark, the analysis can be run on data from other organizations with ease and little adaptation. Results can be compared against other TLDs to see whether scanner behavior is different between them (using different lists or words from different languages, for example).
- The time frame of the analysis is limited. While feature vectors and clustering can be used to compare resolvers not just against each other, but also against their own behavior at a later time, this has not been done yet. A longitudinal study could examine whether scans repeat, and whether scan traffic is generally increasing or decreasing over time.
- Similarly, the extent of whether scans are repeated or ongoing is not fully determined. The motivation behind most operations and the sheer number occurring on a given day imply that most scans are repeated regularly, and clustering on both April 3rd and July 3rd indicates many campaigns are ongoing, but this point should see more research.
- The clustering approach can be used for many other use cases. By including prefixes and ASes, it can be checked whether behavior of resolvers within one IP prefix or AS is consistent, or whether there are outliers. Feature vectors can also be calculated for different times to compare sources longitudinally.
- Combining this approach with passive measurement projects is possible and might allow insight into the sources of scans found in traffic of public resolvers.

7 Summary

This work looked at DNS scanning from the perspective of an AuthNS operator. In a first step, traffic from specific sources was looked at manually, and tools developed for visualizing patterns in traffic. In the second step, feature engineering and clustering was used to find more relevant examples, verify found scans and groups and help classify sources.

The definition chosen was rather broad, because bulk email sending might, given rigorous query name minimization, be indistinguishable from domaining. Scans can target 2LDs or 3LDs, and they can be executed from dedicated IP addresses or by using public resolvers for better performance.

The most relevant attributes for classification of scans are distinct name percentage, response codes, query types, and order of queried names. Label counts, countries, DNS2Vec embeddings, and intersection statistics with name lists (admittedly outdated ones) did not prove relevant for distinguishing scans. Scans exhibit unusual values in one or more dimensions due to their nature, and have a wider range of values for their features, making machine learning possible.

It was possible to classify 11.79 % of significant traffic of April 3rd manually, identifying more than 50 different operations including hundreds of IP addresses. More than 10 percent of traffic was identified to be due to scanning, and interpolating from random samples per cluster on July 3rd yields an estimate of 30 % of traffic being due to scanning on an ordinary day.

Knowledge of the DNS is important for classification, and even more so for feature engineering. Accuracy of clustering on the manually classified data is excellent (98 %), but evaluation on another day still shows inaccuracy in many clusters. While straightforward scans can easily be found by the clustering, other clusters are very diverse. Many sources are difficult to classify even manually.

Most scanning campaigns targeted at 2LDs are run from networks of hosting or cloud providers, but subdomain scanning is most commonly found in open resolvers. It is both very common and visible to the .nl name servers. A single subdomain scanning campaign on April 3rd caused 4 % of requests seen, swamping many open resolvers. On May 21st, 2.6 billion queries were sent by a single scan, which is 54 % of that day's traffic and enough to skew any longitudinal statistics. All queries were distributed among just 3 Anycast sites, but affected processing time only slightly. Rate limiting seems effective against scanning, and there does not need to be concern about AuthNS availability.

Bibliography

[1] M.A. Açikalin. *Profiling Recursive Resolvers at Authoritative Name Servers*. Aug. 2019. URL: http://essay.utwente.nl/79267/.

[2] Metin Açıkalın and Moritz Müller. *Wie klopt daar? Het classificeren van recursieve resolvers bij autoritatieve nameservers*. Dutch. English version available: https://www.sidnlabs.nl/en/news-and-blogs/who's-knocking-profiling-recursive-resolvers-on-authoritative-name-servers. URL: https://www.sidnlabs.nl/nieuws-en-blogs/wie-klopt-daar-het-classificeren-van-recursieve-resolvers-bij-autoritatieve-nameservers (visited on 06/06/2024).

[3] M. Tariq Banday. "Recent Developments in the Domain Name System". In: *International Journal of Computer Applications* 31 (Oct. 2011). DOI: 10.5120/3796-5227.

[4] David Barr. *Common DNS Operational and Configuration Errors*. RFC 1912. Feb. 1996. DOI: 10.17487/RFC1912.

[5] Birk Blechschmidt and Quirin Scheitle. *GitHub Repository: MassDNS - A high-performance DNS stub resolver*. 2020. URL: https://github.com/blechschmidt/massdns (visited on 04/26/2024).

[6] Stéphane Bortzmeyer. *DNS Query Name Minimisation to Improve Privacy*. RFC 7816. Mar. 2016. DOI: 10.17487/RFC7816.

[7] Stéphane Bortzmeyer, Ralph Dolmans, and Paul E. Hoffman. *DNS Query Name Minimisation to Improve Privacy*. RFC 9156. Nov. 2021. DOI: 10.17487/RFC9156.

[8] Nevil Brownlee, K.C. Claffy, and Evi Nemeth. "DNS measurements at a root server". In: *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No.01CH37270)*. Vol. 3. 2001, pp. 1672–1676. DOI: 10.1109/GLOCOM.2001.965864. URL: https://www.caida.org/catalog/papers/2001_dnsmeasroot/dmr.pdf.

[9] Sebastian Castro et al. "A day at the root of the internet". In: *SIGCOMM Comput. Commun. Rev.* 38.5 (Sept. 2008), pp. 41–46. ISSN: 0146-4833. DOI: 10.1145/1452335.1452341. URL: https://www.caida.org/catalog/papers/2008_root_internet/root_internet.pdf.

[10] David Dagon et al. "Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority". In: *Proc. 15th Network and Distributed System Security Symposium (NDSS)*. San Diego, CA, 2008. URL: http://www.citi.umich.edu/u/provos/papers/ndss08_dns.pdf.

[11] Jeremy Dix, Patrick Sattler, and Johannes Zirngibl. "ZDNS vs. MassDNS: A Comparison of DNS Measurement Tools". In: (2023). DOI: 10.2313/NET-2024-04-1_10.

[12] Zakir Durumeric et al. "A Search Engine Backed by Internet-Wide Scanning". In: *22nd ACM Conference on Computer and Communications Security*. Oct. 2015.

[13] Zakir Durumeric et al. *Ten Years of ZMap*. 2024. arXiv: 2406.15585 [cs.CR]. URL: https://arxiv.org/abs/2406.15585.

[14] C.E.W. Hesselman et al. *Een privacyraamwerk voor 'DNS big data'-toepassingen*. Tech. rep. SIDN, Dec. 2014. URL: https://scholar.google.nl/scholar?oi=bibs&cluster=4602678162007378891&btnI=1&hl=en.

[15] ICANN. *ICANN DNS Magnitude statistics page*. URL: https://magnitude.research.icann.org/ (visited on 06/06/2024).

[16] Liz Izhikevich et al. "ZDNS: a fast DNS toolkit for internet measurement". In: *Proceedings of the 22nd ACM Internet Measurement Conference*. IMC '22. Nice, France: Association for Computing Machinery, 2022, pp. 33–43. ISBN: 978-1-45-039259-4. DOI: 10.1145/3517745.3561434.

[17] Athanasios Kountouras et al. "Enabling Network Security Through Active DNS Datasets". In: *Research in Attacks, Intrusions, and Defenses*. Ed. by Fabian Monrose et al. Cham: Springer International Publishing, 2016, pp. 188–208. ISBN: 978-3-319-45719-2. URL: https://coeus-center.com/articles/activedns.pdf.

[18] Marc Kührer et al. "Going Wild: Large-Scale Classification of Open DNS Resolvers". In: *Proceedings of the 2015 Internet Measurement Conference*. IMC '15. Tokyo, Japan: Association for Computing Machinery, 2015, pp. 355–368. ISBN: 978-1-450-33848-6. DOI: 10.1145/2815675.2815683. URL: https://doi.org/10.1145/2815675.2815683.

[19] SIDN Labs. *DNS-verkeer*. URL: https://stats.sidnlabs.nl/nl/dns.htm (visited on 04/26/2024).

[20] SIDN Labs. *Entrada - DNS Big Data Analytics*. 2019. URL: https://entrada.sidnlabs.nl/ (visited on 04/26/2024).

[21] Hyeonmin Lee et al. "Under the Hood of DANE Mismanagement in SMTP". In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1–16. ISBN: 978-1-939133-31-1. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/lee.

[22] Lorenz Lehle, Patrick Sattler, and Johannes Zirngibl. "Structure and Origin of CT Based Domain Lists". In: (2023). DOI: 10.2313/NET-2023-11-1_15. URL: https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2023-11-1/NET-2023-11-1_15.pdf.

[23] Ziqian Liu et al. “Two Days in the Life of the DNS Anycast Root Servers”. In: *Passive and Active Network Measurement*. Ed. by Steve Uhlig, Konstantina Papagiannaki, and Olivier Bonaventure. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 125–134. ISBN: 978-3-540-71617-4. URL: https://www.caida.org/catalog/papers/2007_dns_anycast/dns_anycast.pdf.

[24] Alexander Mayrhofer, Michael Braun¨oder, and Aaron Kaplan. “DNS Magnitude - A Popularity Figure for Domain Names, and its Application to L-root Traffic”. In: 2020. URL: https://www.icann.org/en/system/files/files/dns-magnitude-05aug20-en.pdf.

[25] P. Mockapetris and K. J. Dunlap. “Development of the domain name system”. In: *Symposium Proceedings on Communications Architectures and Protocols*. SIGCOMM ’88. Stanford, California, USA: Association for Computing Machinery, 1988, pp. 123–133. ISBN: 978-0-89791-279-2. DOI: 10.1145/52324.52338.

[26] Giovane Moura et al. “Fragmentation, truncation, and timeouts: are large DNS messages falling to bits?” In: (2021). URL: https://www.sidnlabs.nl/en/news-and-blogs/fragmentation-truncation-and-timeouts-are-large-dns-messages-falling-to-bits (visited on 10/05/2024).

[27] Giovane C. M. Moura et al. “Clouding up the Internet: how centralized is DNS traffic becoming?” In: *Proceedings of the ACM Internet Measurement Conference*. Virtual Conference: ACM, 2020. DOI: 10.1145/3419394.3423625.

[28] Giovane C. M. Moura et al. “Domain names abuse and TLDs: from monetization towards mitigation”. In: *3rd IEEE/IFIP Workshop on Security for Emerging Distributed Network Technologies (DISSECT 2017), co-located with IFIP/IEEE International Symposium on Integrated Network Management (IM 2017)*. May 2017.

[29] Giovane C.M. Moura et al. *Anycast vs. DDoS: Evaluating the November 2015 Root DNS Event*. Information Sciences Institute technical report ISI-TR-2016-709b. United States: University of Southern California, May 2016. URL: https://ris.utwente.nl/ws/portalfiles/portal/5122813/ISI-TR-2016-709.pdf.

[30] Moritz Müller et al. *Recursives in the Wild: Engineering Authoritative DNS Servers*. Tech. rep. ISI-TR-720. USC/Information Sciences Institute, June 2017. URL: http://www.isi.edu/%7ejohnh/PAPERS/Mueller17a.html.

[31] Moritz Müller et al. “Roll, Roll, Roll your Root: A Comprehensive Analysis of the First Ever DNSSEC Root KSK Rollover”. In: *Proceedings of the Internet Measurement Conference*. IMC ’19. Amsterdam, Netherlands: Association for Computing Machinery, 2019, pp. 1–14. ISBN: 9781450369480. DOI: 10.1145/3355369.3355570. URL: https://par.nsf.gov/servlets/purl/10170347.

[32] Jing Qiao. *Detecting resolvers at .nz*. URL: https://web.archive.org/web/20220122132201/https://blog.nzrs.net.nz/detecting-resolvers-at-nz/ (visited on 01/22/2022).

[33] Philipp Richter, Oliver Gasser, and Arthur Berger. "Illuminating large-scale IPv6 scanning in the internet". In: *Proceedings of the 22nd ACM Internet Measurement Conference*. IMC '22. Nice, France: Association for Computing Machinery, 2022, pp. 410–418. ISBN: 978-1-45-039259-4. DOI: 10.1145/3517745.3561452.

[34] Roland van Rijswijk-Deij et al. "A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements". In: *IEEE Journal on Selected Areas in Communications* 34.6 (June 2016), pp. 1877–1888. ISSN: 1558-0008. DOI: 10.1109/JSAC.2016.2558918. URL: https://visitationcs.utwente.nl/pdf/A_High-Performance_Scalable_Infrastructure_for_Large-Scale_Active_DNS_Measurements.pdf.

[35] R Sommese et al. "When parents and children disagree: Diving into DNS delegation inconsistency". In: *Passive and Active Measurement Conference (PAM)* (). DOI: 10.1007/978-3-030-44081-7_11. URL: https://par.nsf.gov/servlets/purl/10186683.

[36] Raffaele Sommese and Mattijs Jonker. "Poster: Through the ccTLD Looking Glass: Mining CT Logs for Fun, Profit and Domain Names". In: *Proceedings of the 2023 ACM Conference on Internet Measurement*. IMC '23. Montreal QC, Canada: Association for Computing Machinery, 2023, pp. 714–715. ISBN: 979-8-40-070382-9. DOI: 10.1145/3618257.3624994.

[37] Raffaele Sommese, Roland Rijswijk-Deij, and Mattijs Jonker. *This Is a Local Domain: On Amassing Country-Code Top-Level Domains from Public Data*. Sept. 2023. URL: https://arxiv.org/pdf/2309.01441.

[38] Thalis Stavropoulos. *A comparative analysis of network data distribution in active DNS measurements. ZDNS vs OpenINTEL*. Jan. 2023. URL: http://essay.utwente.nl/94355/.

[39] Olivier van der Toorn et al. "Addressing the challenges of modern DNS a comprehensive tutorial". In: *Computer Science Review* 45 (2022), p. 100469. ISSN: 1574-0137. DOI: 10.1016/j.cosrev.2022.100469.

[40] Thymen Wabeke, Thijs van den Hout, and Moritz Müller. "DNS2Vec: representation learning toepassen op DNS-data". Dutch. In: (2024). English translation available at https://www.sidnlabs.nl/en/news-and-blogs/dns2vec-applying-representation-learning-to-dns-data. URL: https://www.sidnlabs.nl/nieuws-en-blogs/dns2vec-representation-learning-toepassen-op-dns-data (visited on 04/26/2024).

[41] Maarten Wullink, Giovane C. M. Moura, and Cristian Hesselman. "Dmap: Automating Domain Name Ecosystem Measurements and Applications". In: *IFIP/IEEE Network Traffic Measurement and Analysis Conference (TMA 2018)*. Vienna, Austria, June 2018. URL: https://tma.ifip.org/2018/wp-content/uploads/sites/3/2018/06/tma2018_paper10.pdf.

[42] Maarten Wullink et al. “ENTRADA: Enabling DNS Big Data Applications”. In: *APWG Symposium on Electronic Crime Research (eCRIME 2016), Toronto, ON, Canada. June 1, 2, and 3, 2016*. June 2016.

Declaration of Academic Integrity

I hereby confirm that this thesis, entitled *Identifying DNS Scanners from a TLD Perspective*, is solely my own work and that I have used no sources or aids other than the ones stated. All passages in my thesis for which other sources, including electronic media, have been used, be it direct quotes or content references, have been acknowledged as such an the sources cited. I am aware that plagiarism is considered an act of deception which can result in sanction in accordance with the examination regulations.

(date, signature of student)

I consent to having my thesis cross-checked with other texts to identify possible similarities and to having it stored in a database for this purpose.

I confirm that I have not submitted this thesis in part or whole as an examination paper before.

(date, signature of student)

Appendix

List of Abbreviations

A Address

AAAA IPv6 Address

AA Authoritative Answer

AD Authenticated Data

AS Autonomous System

ASN Autonomous System Number

AuthNS Authoritative Name Server

ccTLD Country-Code Top Level Domain

CD Checking Disabled

DNS Domain Name System

DNSSEC DNS Security Extensions

DS Delegation Signer

ENTRADA ENhanced Top-level domain Resilience through Advanced Data Analysis

FQDN Fully-Qualified Domain Name

gTLD generic Top-Level Domain

HDFS Hadoop File System

ICANN Internet Company for Assigned Numbers and Names

IP Internet Protocol

ISP Internet Service Provider

MX Mail Exchange

NAT Network Address Translator

NS Name Server

NSEC Next Secure

NSEC3 Next Secure 3

NX non-existent

RA Recursion Available

rcode Response Code

RD Recursion Desired

RFC Request for Comments

RR Resource Record

SIDN Stichting Internet Domeinregistratie Nederland

SOA Start of Authority

SQL Structured Query Language

TCP Transmission Control Protocol

TC Truncation

TLD Top-Level Domain

TTL Time to Live

UDP User Datagram Protocol

Overview of Relevant DNS Record/Query Types

Type	Code	Occurrence	Usage
A	1	50.74 %	The IPv4 address for a given hostname.
NS	2	14.41 %	The name server authoritative for a given name or zone.
AAAA	28	11.97 %	The IPv6 address for a given hostname.
DS	43	10.83 %	Contains the hash of the key signing key of a child zone, necessary to check the authenticity of the delegation.
TXT	16	2.53 %	Contains human-readable text, used for various purposes.
HTTPS	65	1.20 %	Contains information enabling faster connection to HTTPS hosts.
CNAME	5	0.95 %	Provides a canonical name, thus making the record-carrying zone an alias.
SOA	6	0.87 %	Contains data about the DNS zone itself, such as a serial number of the current zone file and TTL for negative caching.
ANY	255	0.52 %	Pseudo-type used for requesting any records available, which is handled differently by various implementations.
DNSKEY	48	0.43 %	Public part of a key.

Table 7.1: DNS record types and percentage of queries asking for each type on April 3rd. The top query types can also be found in the statistics at stats.sidnlabs.nl.

Type	Code	Occurrence	Usage
SRV	33	0.24 %	Data about a specific service available, e.g. port number.
TLSA	52	0.15 %	Associates a TLS certificate with the domain name, relating to DANE.
CAA	257	0.08 %	Specifies which certificate authority is authorized to issue certificates for this domain, to prevent rogue authorities from issuing false certificates.
PTR	12	0.01 %	Used for reverse IP-to-name lookup, reverse of an A/AAAA record
DNAME	39	0.01 %	Like CNAME, alias including all subdomains.
HINFO	13	0.00 %	
NAPTR	35	0.00 %	
NSEC	47	0.00 %	
SVCB	64	0.00 %	
RRSIG	46	0.00 %	
CDS	59	0.00 %	
NSEC3	50	0.00 %	
AFSDB	18	0.00 %	
CDNSKEY	60	0.00 %	
CERT	37	0.00 %	
NSEC3PARAM	51	0.00 %	
URI	256	0.00 %	
LOC	29	0.00 %	
SSHFP	44	0.00 %	
RP	17	0.00 %	
ZONEMD	63	0.00 %	
APL	42	0.00 %	
DHCID	49	0.00 %	
CSYNC	62	0.00 %	
DLV	32769	0.00 %	
EUI64	109	0.00 %	
HIP	55	0.00 %	
IPSECKEY	45	0.00 %	
EUI48	108	0.00 %	
KX	36	0.00 %	
KEY	25	0.00 %	
SIG	24	0.00 %	
SMIMEA	53	0.00 %	
OPENPGPKEY	61	0.00 %	
TA	32768	0.00 %	
TSIG	250	0.00 %	

Table 7.2: Continuation of Table 7.1

Additional Examples of Resolver Behavior

More scatterplots and histograms found to describe unusual resolver behavior.

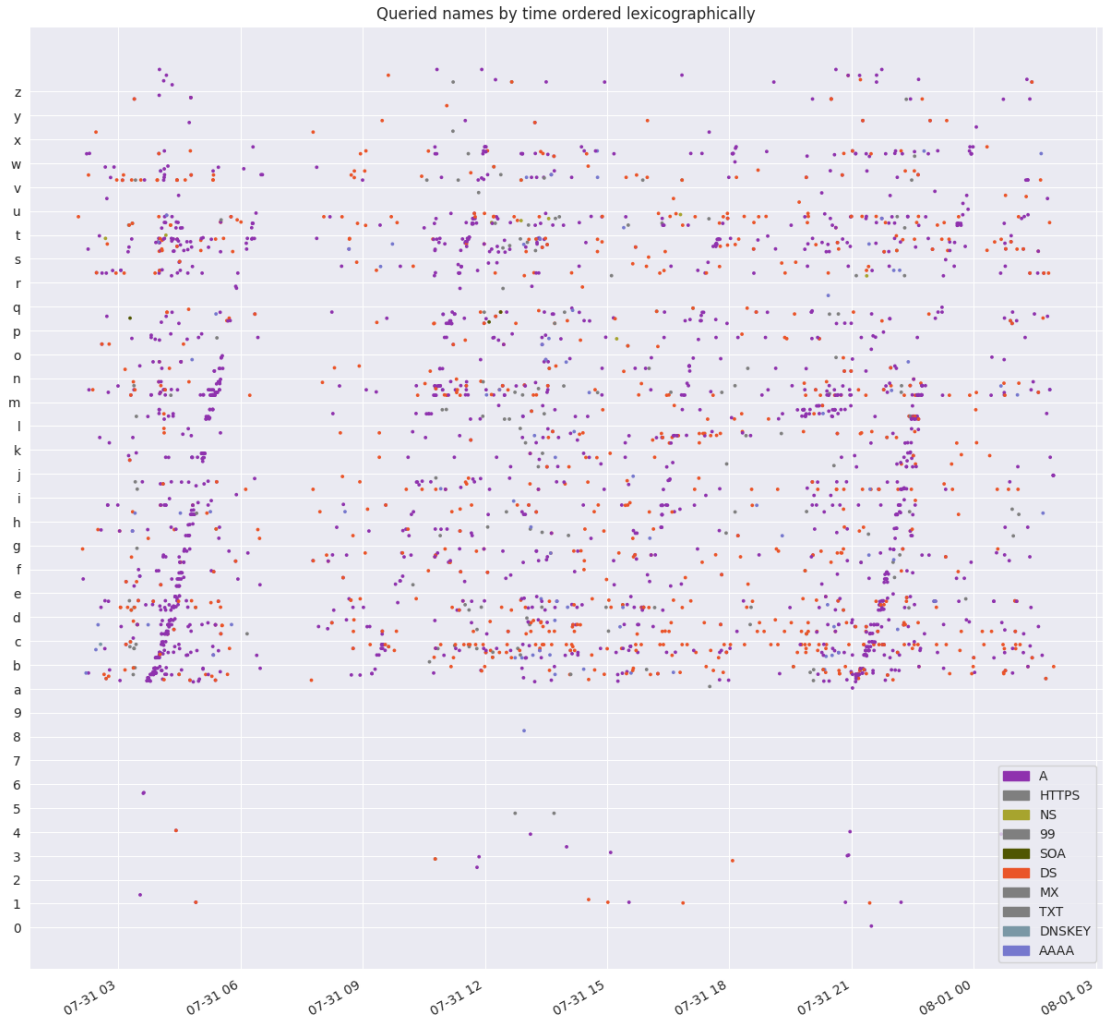


Figure 7.1: Traffic from a public resolver containing traces of a scanning operation with alphabetical ordering four times per day (time frame = 1 day)

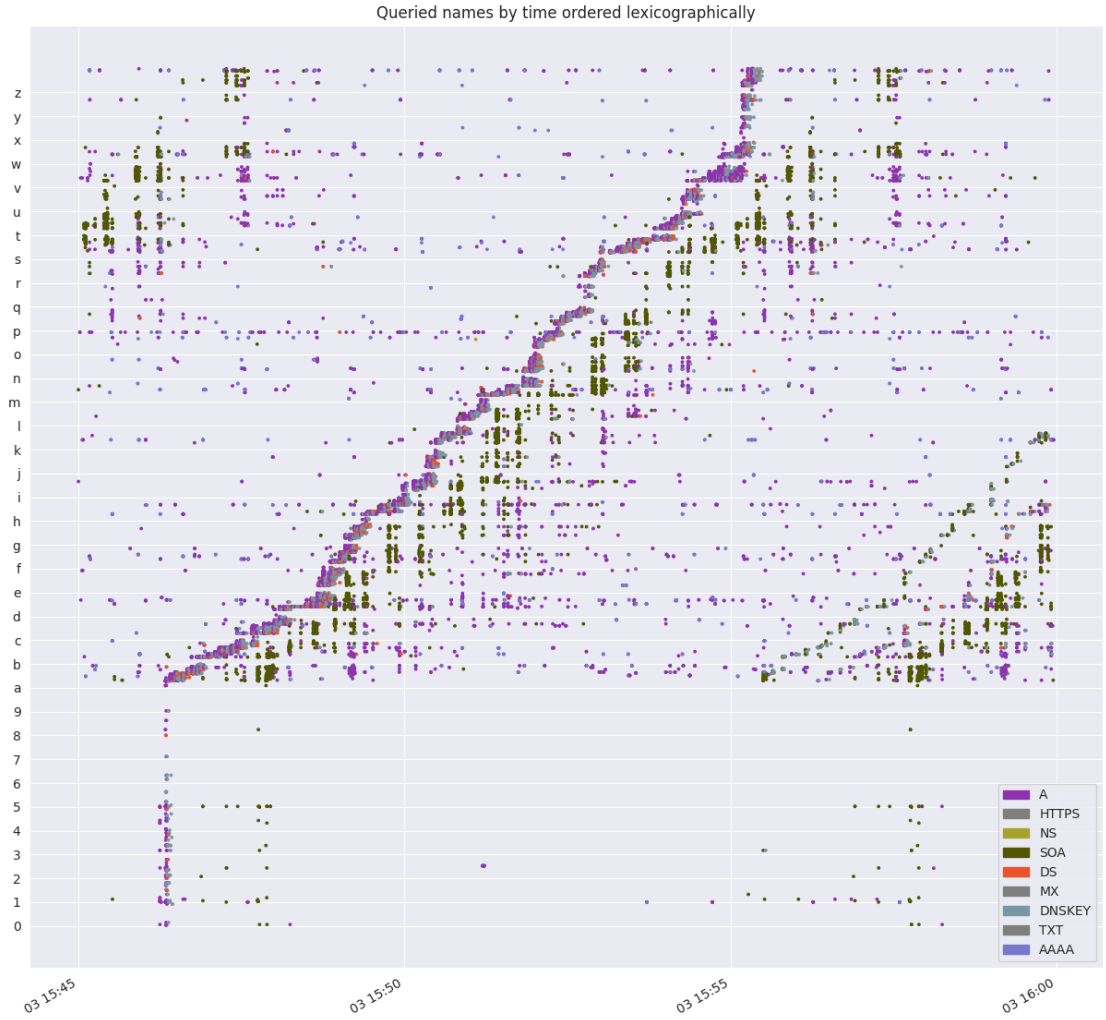


Figure 7.2: Traffic containing a scanning operation with alphabetical ordering and successive record types (time frame = 15 minutes). Different record types (colors) are clearly queried at different times. SOA records seem to be queried in batches (green vertical lines), while A and DS records are queried constantly (no vertical gaps)

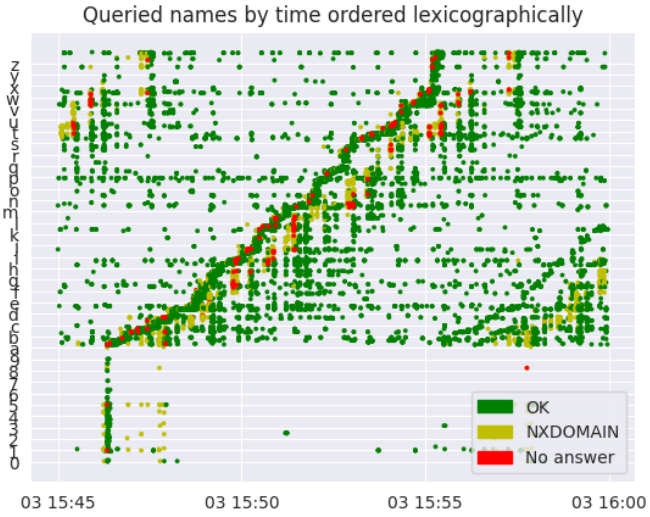


Figure 7.3: Response codes show many SOA queries from Figure 7.2 are NX domains, while the other queries are not. Only queries from the scan seem to be left unanswered (red), while any background traffic is answered reliably (green or yellow).

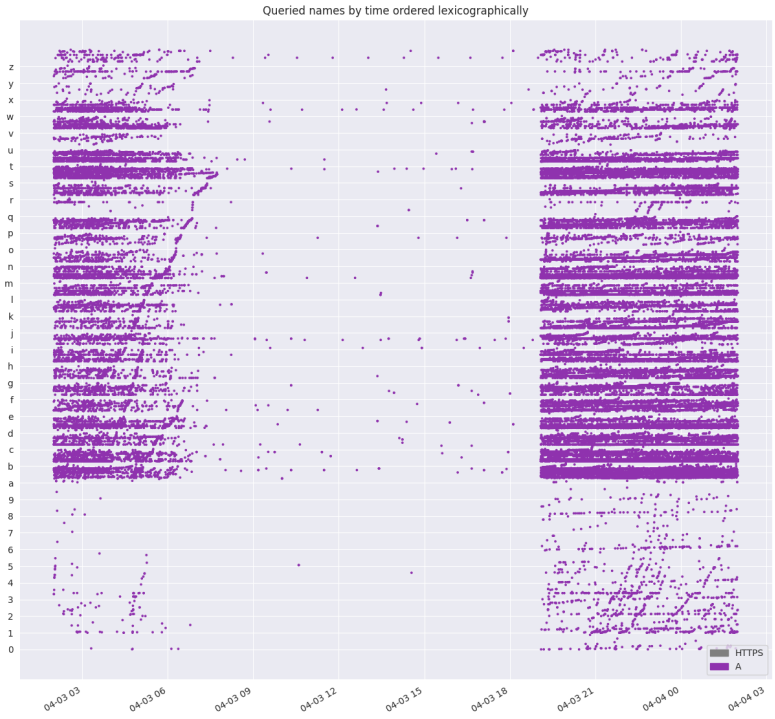


Figure 7.4: A scanning operation with alphabetical ordering visible in many parts (time frame = 1 day). The vast majority of traffic is sent between 7 PM and 7 AM, basically only one query type is used

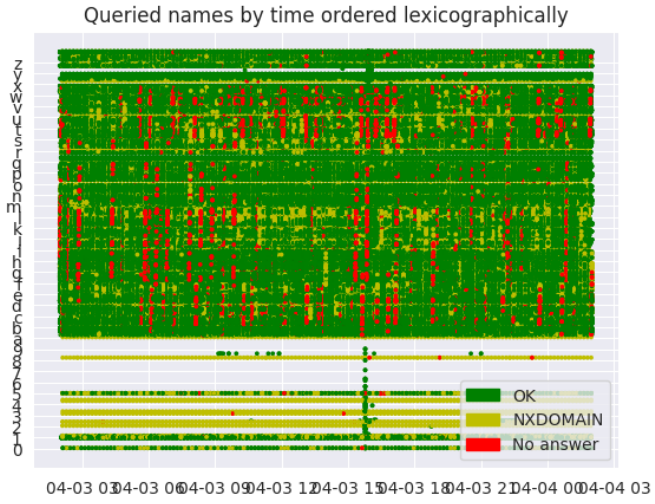


Figure 7.5: Dense traffic displaying a large number of unanswered queries and NX domains (time frame = 1 day)

Figure 7.6: The full-day visualization of Figure 4.13

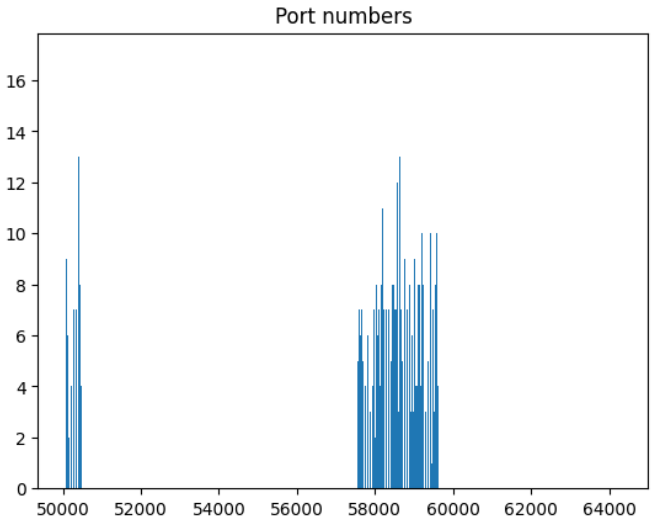


Figure 7.7: Only some specific port ranges are used

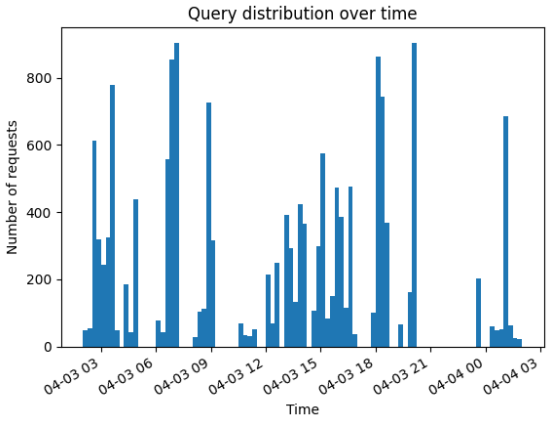


Figure 7.8: Unusually ragged time distribution with gaps

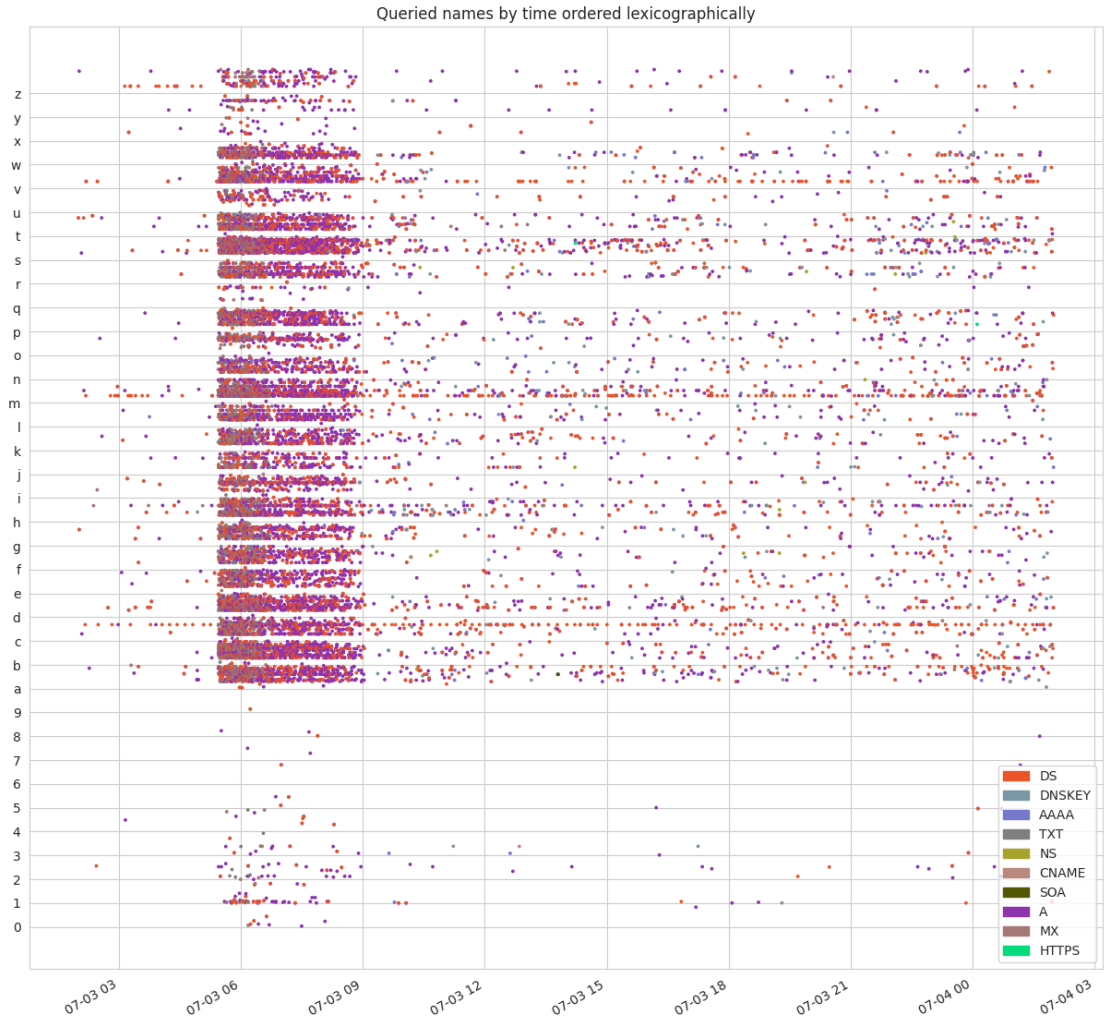


Figure 7.9: One peak in traffic from a public resolver due to scan *beginning*, probably relating to mail

Figure 7.10: Traffic from another resolver affected by *beginning*, showing more NX domains queried, too

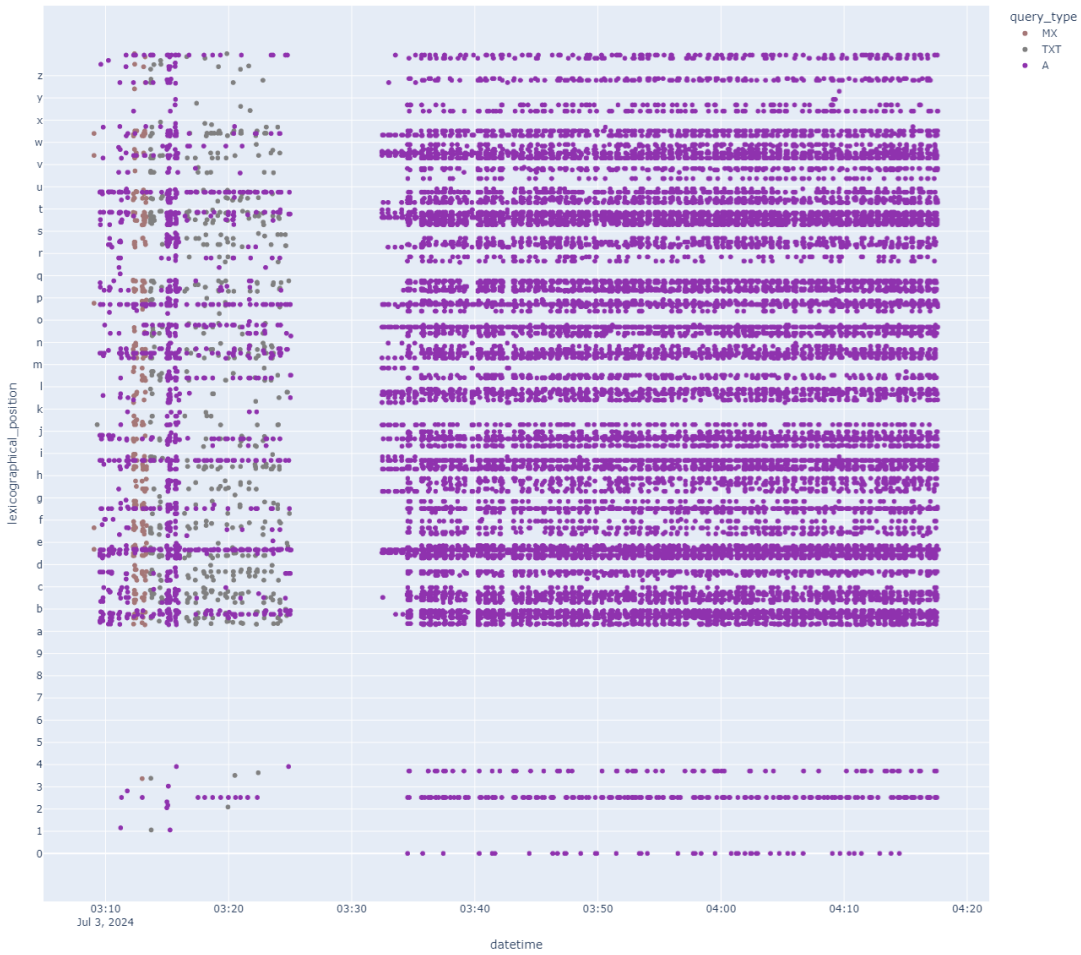


Figure 7.11: Traffic from scan *headline*, exhibiting waves in time and changing query types

Clustering Resolvers versus Queries

This section contains a discussion on the advantages and disadvantages of clustering resolvers or individual queries.

When applying clustering on DNs traffic, two options are available:

Queries Single queries (rows in the database) can be analyzed in their similarity to each other. This requires defining a similarity measure on queries, using queried names, different flags, source IP address and more. Then, similar queries can be clustered, finding strands of similar query behavior, potentially including many different sources, for example all querying the same name in a similar timeframe several times.

An upside of this approach is that it is source agnostic, meaning it could group traffic from many different sources and therefore correctly identify scanning campaigns showing in the traffic of public resolvers.

On the other hand, the lack of complex features means queried names would need to be considered carefully, for example by calculating the editing distance between names or defining sophisticated embeddings. Taking into account the large number of datapoints, this is computationally expensive. Additionally, in Section 4.1.5, it became clear that most signs of scanning behavior will not show when looking at individual queries, but can only be described when aggregating multiple datapoints. Because of this, even clear outliers found through the clustering may be difficult to classify.

Resolvers Creating feature vectors for resolvers allows comparing different sources of traffic to each other. Using many datapoints for each feature increases robustness of the descriptions. In particular, it makes the identification of outliers easy. As described previously, these have a higher likelihood of being scanning operations, and thus this method could lead to scans more directly.

However, this approach only allows for analysis of discrete chunks of traffic (that is, each source defining one chunk), even though open resolvers will have traffic coming from different sides and show different behavior depending on the kind of traffic that reaches them. Similarly, different public resolvers may show the same behavior. On top of that, a sensible definition is required as for what constitutes a single source. It is unclear whether to use IP addresses as a definition for sources, or prefixes, as described in Figure 4.1.5. Then, features have to be chosen which can be calculated efficiently from the data, which is not trivial. The features need to describe each resolver's behavior quantitatively and be appropriate for the use case of finding scans. This requires domain knowledge. However, it can build on top of some of the previous work on resolver classification done by New Zealand's [32], by M. Açıkalın [1], and SIDN [40]. Particularly DNS2Vec provides useful features in that they describe query behavior regarding the queried names, which is difficult to describe numerically with less sophisticated means. On top of that, the

insights from the previous section 4.1.5 can be used, including the implementations done.

Unlike clustering queries, the computationally heavy part of this approach lies in the calculation of the features instead of the clustering itself.

It could already be seen during the manual classification that many sources send only scanning traffic or only normal traffic, but a significant amount of public resolvers also contain traffic generated by scans, making it difficult to label the traffic when looking at sources as a whole. This approach, in its simplest form, assumes any source to either be wholly scan or not-scan, whereas this is not the case in practice for scans that use external recursive resolvers.

Considerations Analyzing on a single-query-basis can also generalize better over time, since time is just another dimension and there is no need to limit the time frame artificially, allowing for queries from different days to be clustered in one operation. Algorithms such as DBSCAN can generate long strands of queries, continuing for a long time. On the other hand, the sheer number of points to process might limit clustering to highly efficient algorithms such as k-Means or DBSCAN, and classification of single queries is probably impossible due to the small amount of information contained.

To summarize: While it does require more feature engineering, describing resolvers seems like the more promising option. It does assume a consistency of traffic within a single source, which is not necessarily given. Often, scans show stark peaks in activity or run for specific hours, and might be mixed with normal traffic in open resolvers. In this case, assuming that sources can be classified as a whole is incorrect. However, open resolvers showing similar behavior due to signs of the same scan will still have similar values in features, and they can also be compared over time and in desired degree of granularity. Detecting scanning campaigns in public resolvers should thus be possible with this method, as well.