Identifying DNS Scanners

Master Thesis

SIDN LABS

WWU MÜNSTER

Topic

- ZDNS[1] and MassDNS[2] were published around 2016
  - Allow resolution of millions of domain names per minute
- OpenINTEL was not
  - “[…] if lots of researchers were to set up similar infrastructures this would have a significant and possibly disruptive impact on the Internet.” [3]
- Does availability of those tools really pose a risk to DNS infrastructure?

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

[2] https://github.com/blechschmidt/massdns

[3] R. van Rijswijk-Deij, M. Jonker, A. Sperotto and A. Pras, "A High-Performance, Scalable Infrastructure for Large-Scale Active DNS Measurements", p. 1886

Research Questions

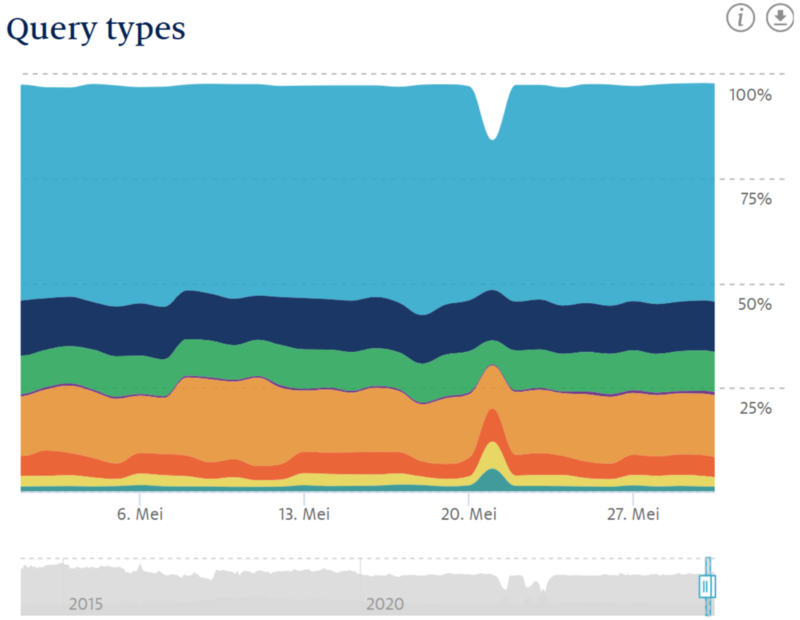
- Can scans be found in the requests to authoritative ccTLD name servers?
- How can they be distinguished from normal traffic?
- Who performs scans and for what purpose?
- What portion of traffic is due to scans?
- Does availability suffer?

Research Questions

- Can scans be found in the requests to authoritative ccTLD name servers? **Yes**
- How can they be distinguished from normal traffic? **Many patterns**
- Who performs scans and for what purpose? **Diverse groups, diverse purposes**
- What portion of traffic is due to scans? **Approx. 30 %**
- Does availability suffer? **No**

Relevance

https://stats.sidnlabs.nl/nl/dns.html#query%20types



- Little research/analysis is done on the composition of DNS traffic, especially at TLD level
  - IP/Port scanning, Root servers,
    Bogus traffic, Performing scans,
    Resolver classification (e.g. monitoring or public resolvers)
- Reasons for scanning: domaining, academics, protection of trademarks, monitoring, finding vulnerabilities, web scraping, bulk email sending, and more!
- Results could help…
  - understand incoming traffic
  - substantiate best practices and protection mechanisms, if necessary
  - sanitize data (DNS-based popularity)

Data

- Individual requests recorded by all .nl AuthNS Anycast sites
- Processed by SIDN's Entrada[1]
- Stored in Apache Hadoop for querying with the Spark query engine
- Around 5 billion queries from 1 million different IP addresses each day

```
root
 |-- id: integer (nullable = true)
 |-- time: long (nullable = true)
 |-- qname: string (nullable = true)
 |-- ttl: integer (nullable = true)
 |-- ipv: integer (nullable = true)
 |-- prot: integer (nullable = true)
 |-- src: string (nullable = true)
 |-- srcp: integer (nullable = true)
 |-- dst: string (nullable = true)
 |-- dstp: integer (nullable = true)
 |-- aa: boolean (nullable = true)
 |-- tc: boolean (nullable = true)
 |-- z: boolean (nullable = true)
 |-- rcode: integer (nullable = true)
 |-- qtype: integer (nullable = true)
 |-- country: string (nullable = true)
 |-- asn: string (nullable = true)
 |-- labels: integer (nullable = true)
 |-- proc_time: integer (nullable = true)
 |-- server_location: string (nullable = true)
 |-- pub_resolver: string (nullable = true)
…
(60 columns in total)
```

[1] https://entrada.sidnlabs.nl/

Methodology

1. Manual work:
    1. Understand and examine the data
    2. Find out how to classify a scan as a human
    3. Collect some ground-truth data
2. Machine Learning:
    1. Implement features describing resolver behavior
    2. Apply a clustering algorithm
    3. Evaluate
3. Use to answer research questions

Results

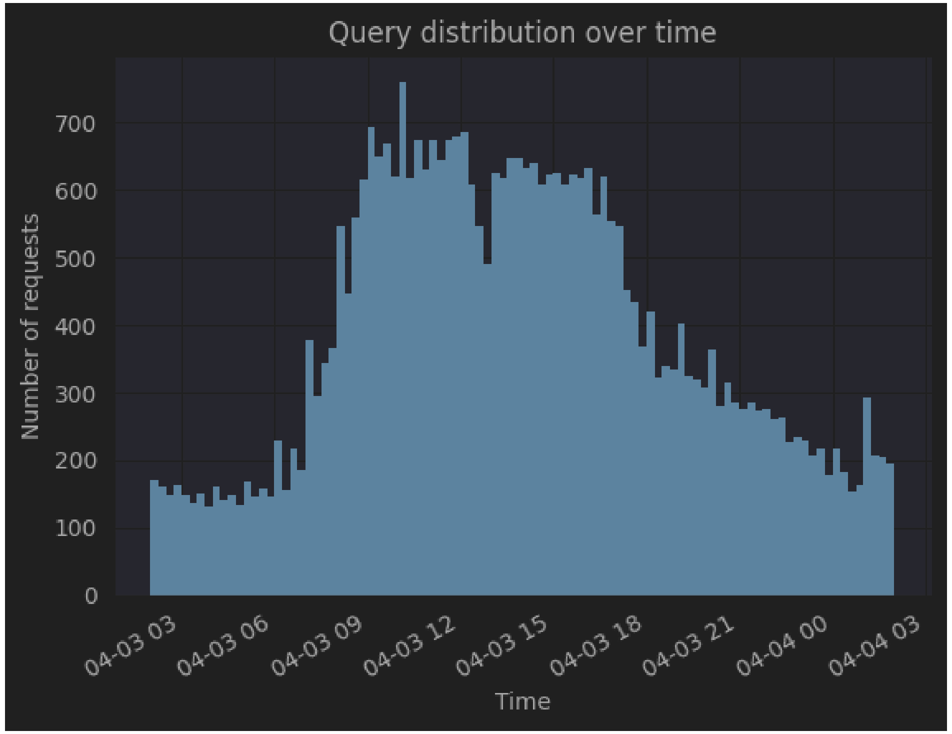
Results - 1

- More than 67 scanning operations identified and confirmed
  - 714 different sources
  - 437 million queries per day (10.3 % of all traffic)
  - Just part of the total scan traffic
- Scans show distinct behavior
  - Sometimes more obvious, sometimes less obvious
  - Most relevant:

    Query distribution over time, percentage of NXDOMAIN answers, alphabetical ordering, question types

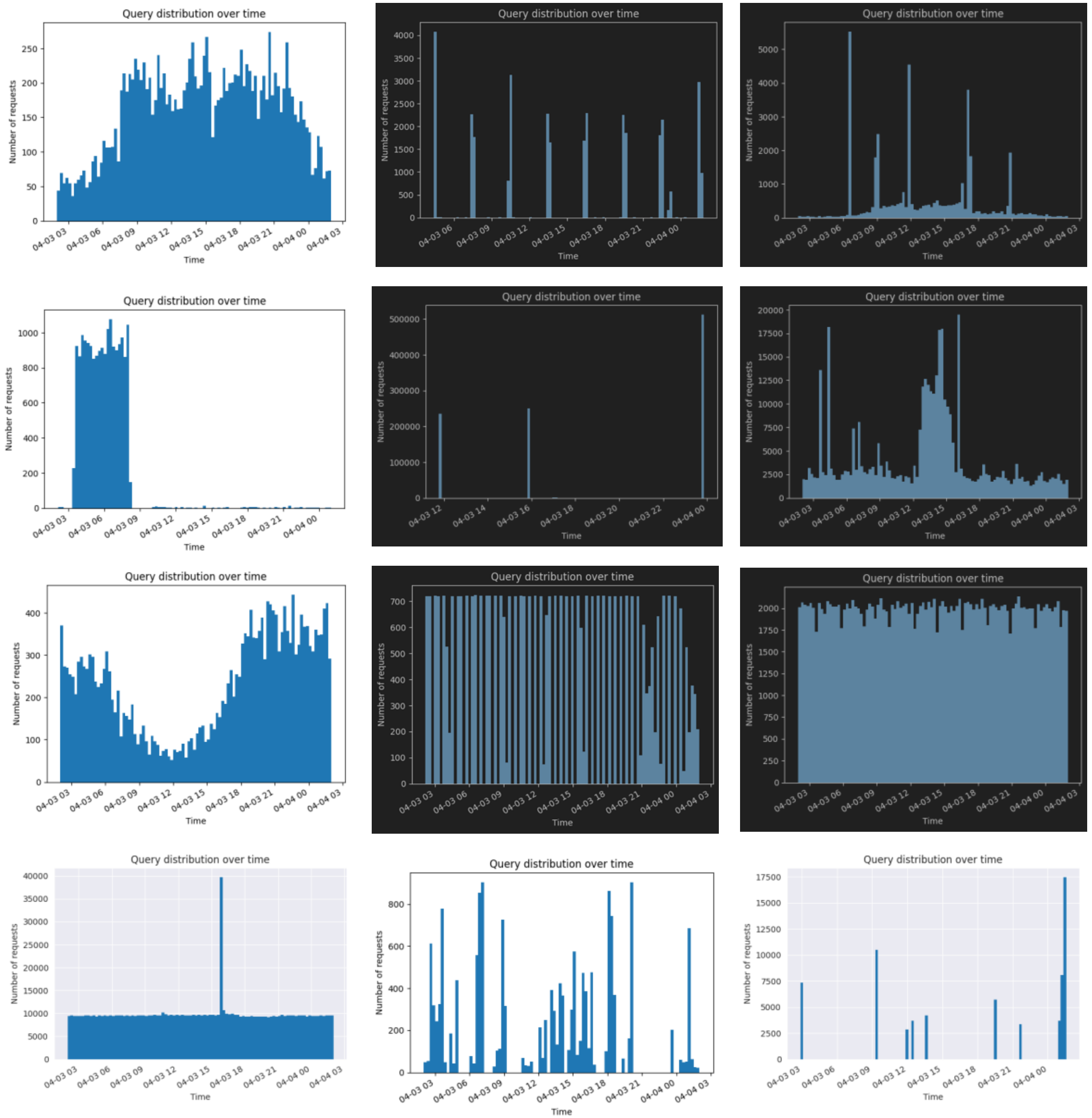
Results – 2

expected



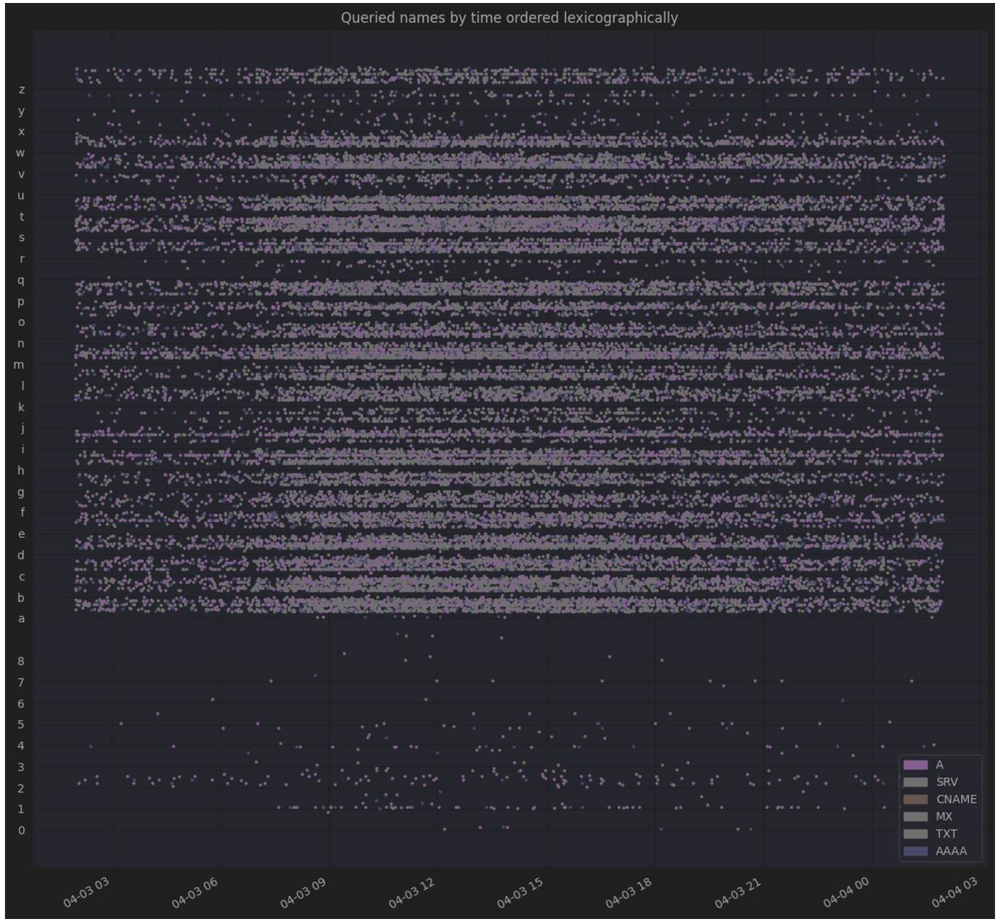
Constant traffic, more during the day

unusual



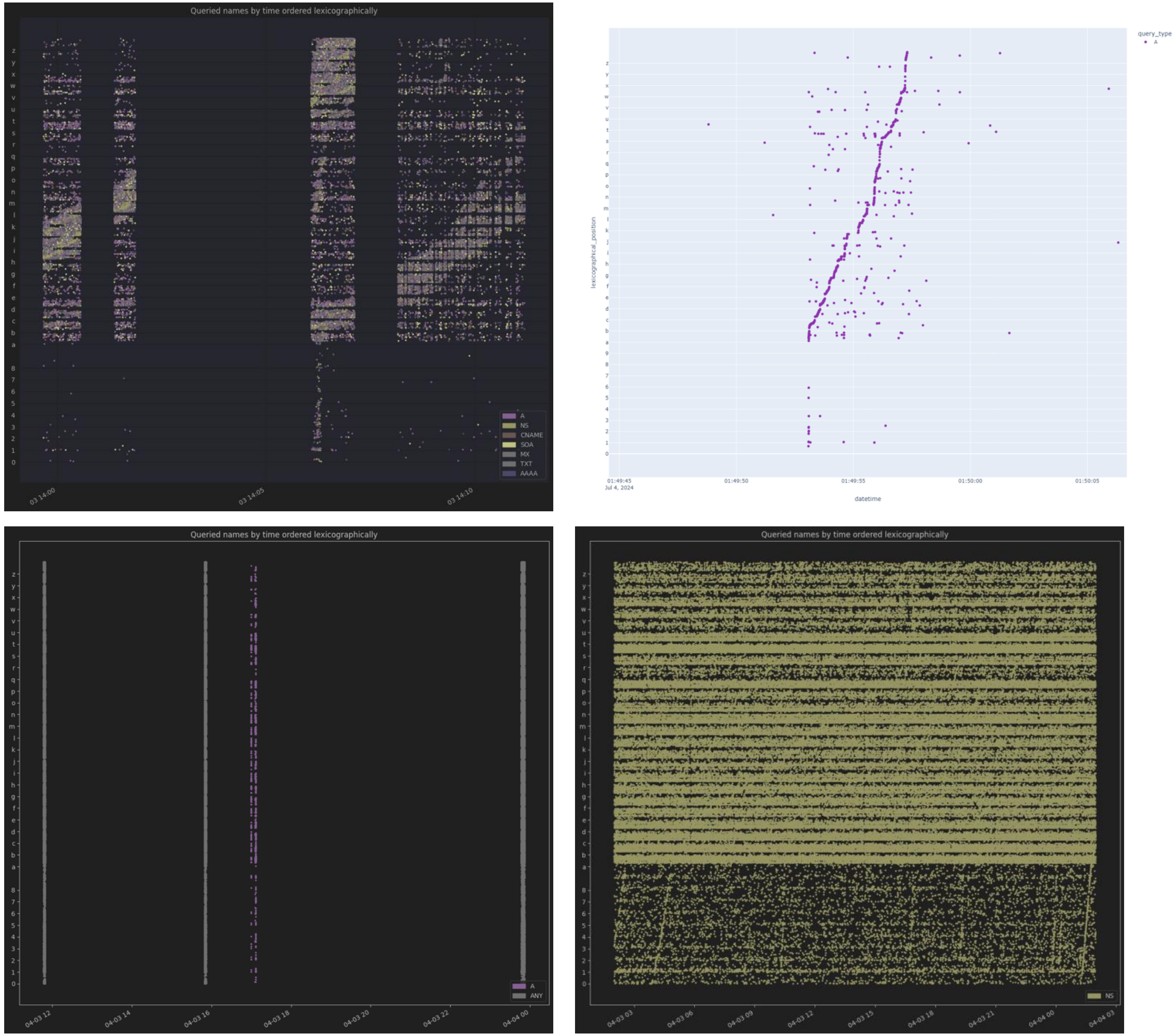
Results - 2

expected



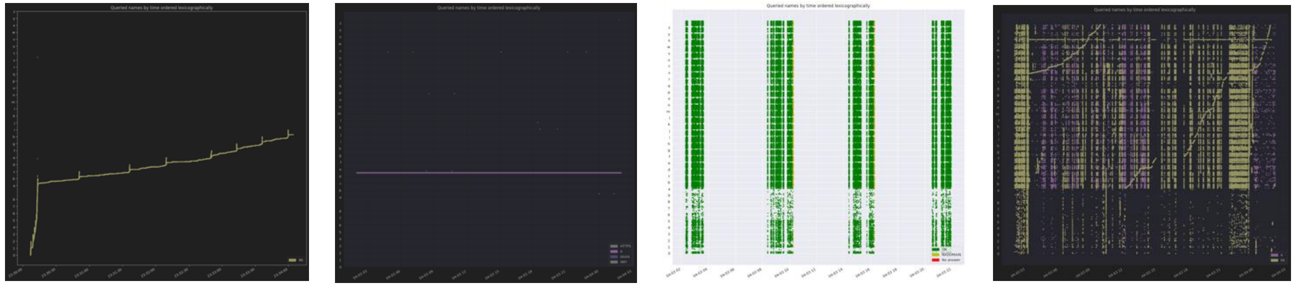
No relationship between time, domain name and query type

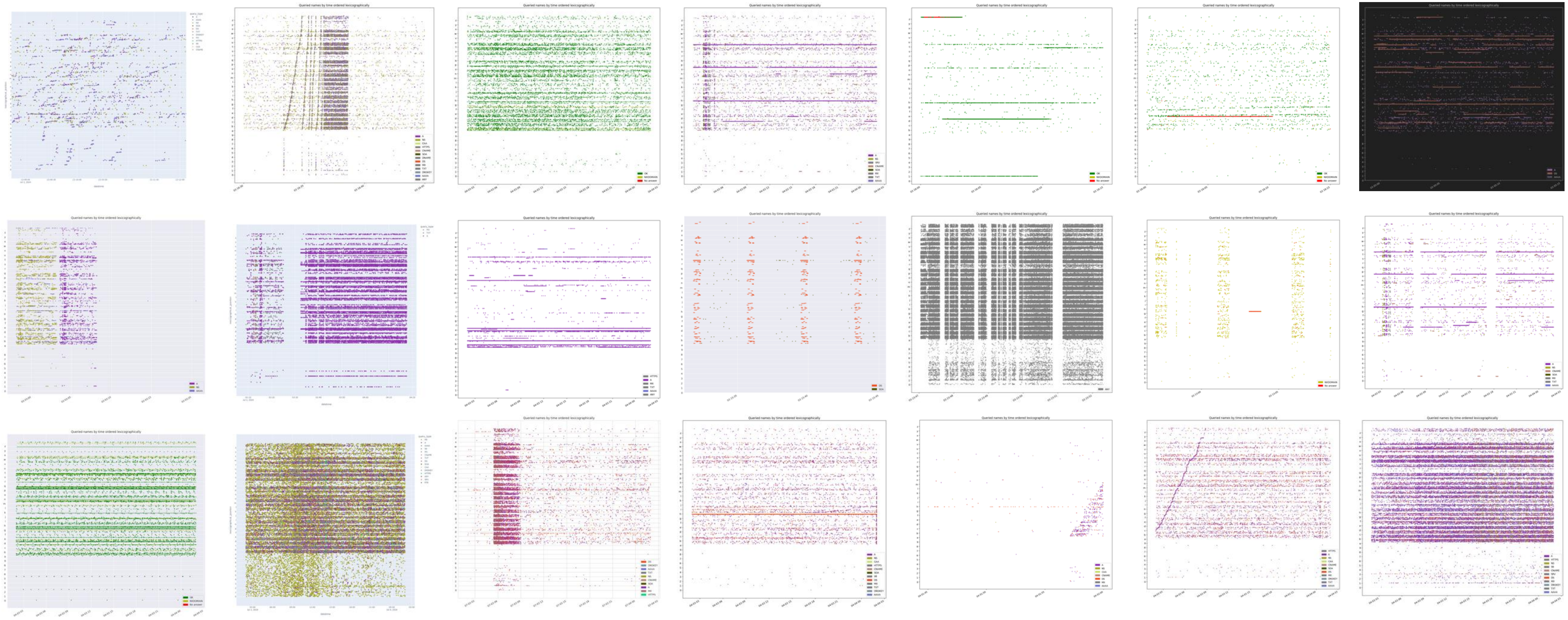
unusual



Results - 2

unusual





- How can we make sure we have found everything?
- How can we choose and group resolvers more systematically?

- How can we make sure we have found everything?
- How can we choose and group resolvers more systematically?

1. Create features
2. Calculate for each resolver
3. Clustering

→ Similarity measure, grouping

Features

- Different flags (truncation, recursion desired, ...)
- Basic statistics (query count, distinct percentage, country)
- Percentage of queries with each

  response code, operation code, query type, query class, number of labels, UDP/TCP, target server, starting symbol for the domain name, IPv4/IPv6, punycode
- Statistics about

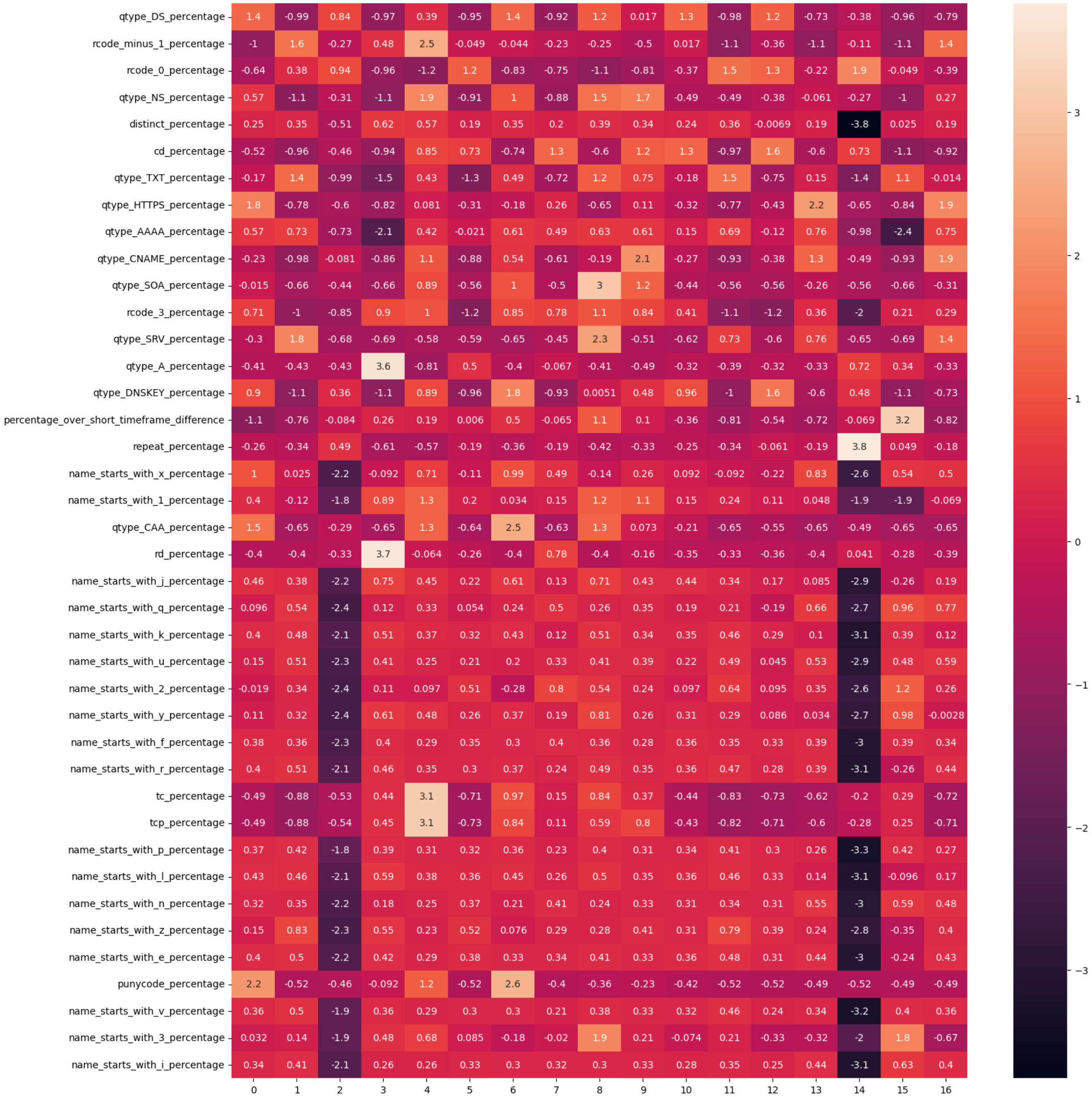
  query timestamps, domain name length, IDs, source port, EDNS UDP, packet TTL

Custom Features

- Distribution of queries among all names queried
- Histogram of queries in time
- Max/Min number of queries sent within 60 seconds
- Statistics about:
  - Repeated queries
  - Time between consecutive queries
- Intersection of names with name lists
  - Common Crawl
  - Certificate Transparency
  - Registered names from 1 year ago
- DNS2Vec features (learned from domain names)

Results - Clustering

- High accuracy when distinguishing scans and non-scans on manually curated dataset (97 %)
- Easily able to find typical domaining or similarly obvious scans
- Is explainable



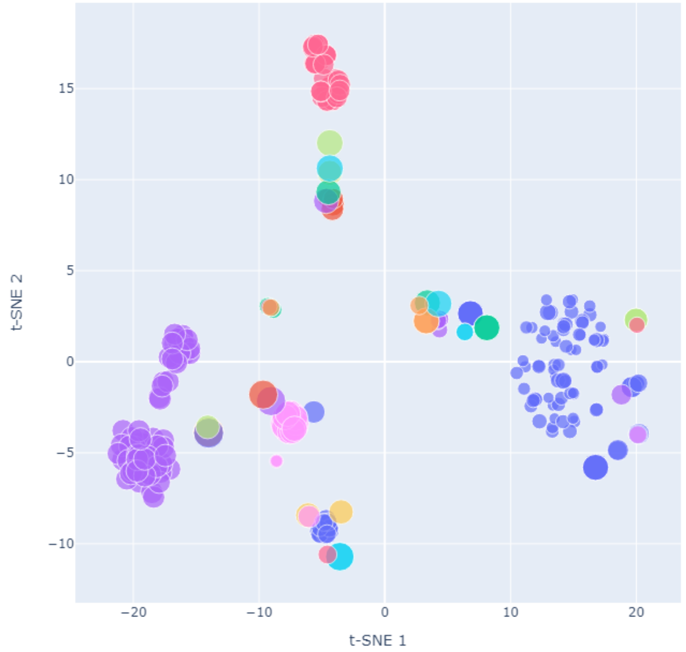
Findings

- Broad definition necessary, including bulk mail sending and monitoring
- Much of traffic unclear (shows signs of scan, but not definite)
  - Some scans are also *very* obvious
- About 30 % of traffic from scans (± 10 %)

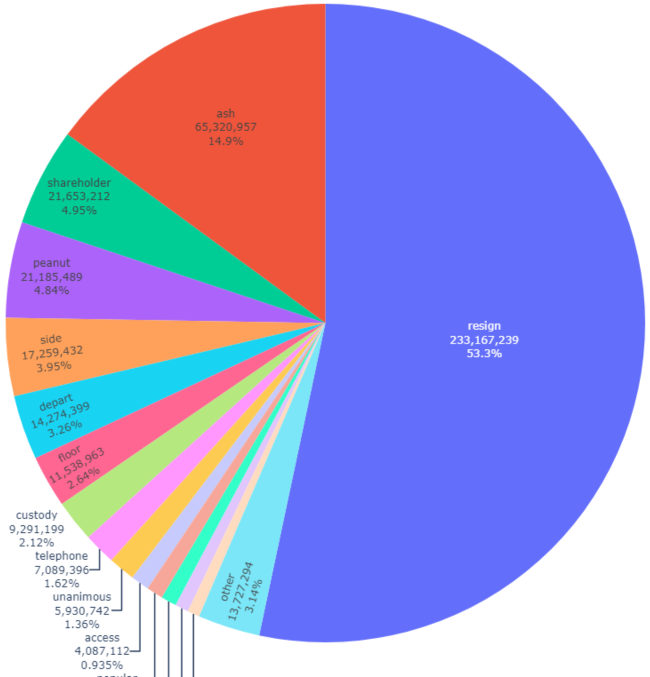
- AuthNS manage just fine, not affected by extremely large scans, even

Groups

- 10 % of traffic *definitely* scanning
- Majority on one day from single subdomain scanning operation

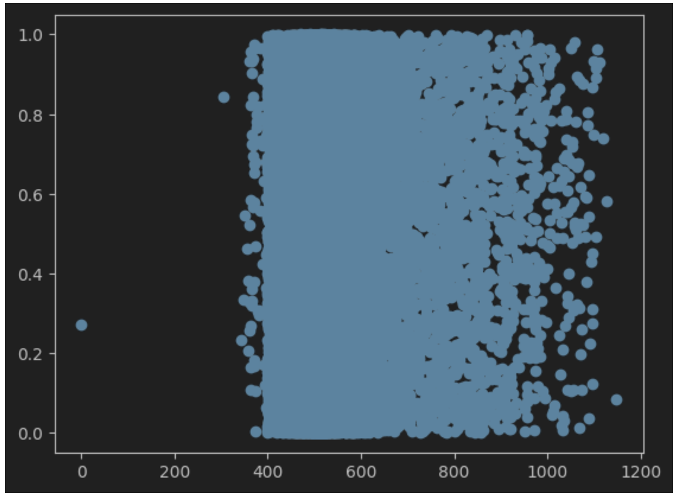


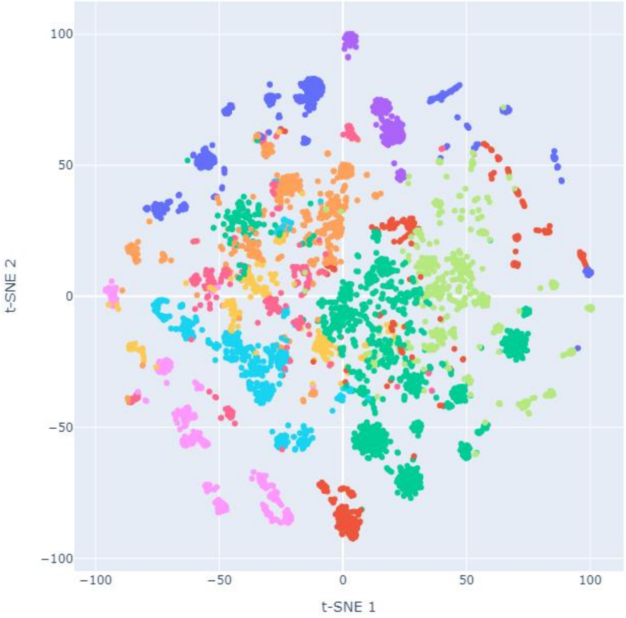
- Most scans from networks of hosting providers
- Public resolvers: Most contain subdomain scanning or small scans



Interesting Findings

- Some resolvers have little similarity to others
    - Others have a clear group of similar ones
- A significant number of resolvers simply ignore NS3 or NS4
- Clustering well-able to find public resolvers with scans
- Few features necessary
    - Can be calculated from just traffic





Other patterns

- Duplicate queries (why?)
- Repetitions within a short time (poor caching?)
- Querying name servers far too often
- Insufficient caching, or no negative caching for NX 2LDs
- Often either NS3 or NS4 completely ignored (out of 3 servers)

Lessons

- Scanning is highly prevalent
- Subdomain scanning can be seen in TLD traffic
    - Mostly done through public resolvers

Limitations and Challenges

- Only considering the NL zone, results expected to generalize
- Broad definition of scanning: Includes domaining, monitoring, and basically everything causing many contiguous DNS requests
- Analysis done on two days of data (5 billion queries each)
- Wide range of behavior that is not easy to explain/grasp/classify
- Scans might not be visible when looking at individual IP addresses
  - Scans distributed among hundreds, even different organizations, do exist!
    - Particularly large numbers of scans probably do not
- Public resolvers difficult to analyze
- Small chance of scans escaping detection (false negatives) due to feature count