Gollector: Measuring Domain Name Dark Matter from Different Vantage Points

Kaspar Hageman[1], René Rydhof Hansen[2], and Jens Myrup Pedersen[1]

[1] Department of Electronic Systems, Aalborg University, Denmark
{kh,jens}@es.aau.dk
[2] Department of Computer Science, Aalborg University, Denmark
rrh@cs.aau.dk

Abstract. This paper proposes *Gollector*, a novel tool for measuring the domain name space from different vantage points. Whereas such measurements have typically been conducted from a single (or few) vantage point, our proposed solution combines multiple measurements in a single system. *Gollector* allows us to express the relative difference in the covered domain name space, and the temporal characteristics, as domain name *dark matter*. We leverage a three week trace from four vantage points, by applying the tool to three security-related use cases: early domain registration detection, data leakage in a split horizon situation, and a proposed method for subdomain enumeration. We release the *Gollector* source code to the research community to support future research in this field.

Keywords: DNS · TLS · Domain names · Measurements

1 Introduction

The DNS has historically played, and continues to play, a vital role in many different areas of network security research, including examining Internet censorship [31], spam detection [38,26] and identifying botnet communication [20]. The highly distributed nature of DNS, with information scattered across millions of domain name servers, means there exists neither a method to reliably observe all interactions with the DNS, nor a method to reconstruct the full domain name space. Consequently, researchers (implicitly or explicitly) choose one or more *vantage points*, e.g., network locations or network datasets, from which to conduct DNS measurements.

The (DNS) vantage point has a major impact on the DNS-related data that can be collected and processed. This is the case both for *passive* measurements, e.g., traffic monitoring of DNS resolvers that is highly dependent on physical location, but also for *active* measurements, e.g., due to geographical split horizons [11] or censorship [31]. We will refer to the part(s) of the domain name space that cannot be observed from a given vantage point as *DNS dark matter* (with respect to that viewpoint).

Different (partial) solutions have been deployed by the research community, including simply increasing the number of vantage points covered as well as including more different *types* of vantage points. However, to the best of our knowledge, there have been no studies focusing specifically of the impact of choosing different vantage points and solutions discussed in the literature have been mostly ad-hoc. In this paper we introduce the *Gollector* tool and framework as a step towards a more systematic and structured treatment of vantage points specifically, and DNS data collection and analysis in general. In particular, we intend *Gollector* to become the "one-stop-shop" for DNS collection and analysis. Therefore, we open-source the tool for the research community to use, and the source code can be found at:

`https://anonymous.4open.science/r/gollector-663A`

We start the paper by providing the relevant background (Section 2) and place our work in the context of prior research (Section 3). In the remainder of the paper, we present the main contributions of the paper:

- We provide an overview of the existing vantage points, describing their conceptual advantages and drawbacks (Section 4).
- We present a novel tool and framework, *Gollector*, which combines data collected from different vantage points, allowing for the comparison of each vantage point (Section 5).
- We apply the tool to a sample dataset (Section 6) and leverage the data (in Section 7) for three previously unexplored use cases related to *DNS dark matter*, and show that combining vantage points has a positive impact on DNS studies: *(i)* early domain detection, *(ii)* data leakage with split horizons and *(iii)* subdomain enumeration.

2 Background

Systems on the Internet are commonly addressed within two namespaces: the IP address name space and the domain name space. IP addresses are used by routing devices to forward network traffic to the correct host, whereas domain names are more user-friendly and are therefore used by humans to address hosts. The Domain Name System (DNS) has been facilitating the translation between these two namespaces since the 1980s [30]. The domain name space forms a tree structure, where each node in the tree is a label, and a domain name is a composition of all labels in a path of the tree from the root to a leaf. The first layer of nodes are referred to as top level domains (TLDs) and for the right-most label in a domain name (e.g., `.com` is the TLD for the domain `www.example.com`). A domain – sometimes referred to as a Fully Qualified Domain Name (FQDN) – can further be decomposed in an apex part and a subdomain part; an apex domain is the domain under which the domain name is registered, with the subdomain being the remaining part of the domain name. The higher levels of the domain name space are highly-regulated and reserved, and newly created domains (i.e., the apex domains) can be created under so-called public suffixes

only [16]. A part of the domain name space that is managed by a particular organisation are referred to as a zone. For instance, registries - the operators of the TLDs – are authoritative of the zone which comprises all domains under that TLD, and maintains a zone file in which all mappings between the domain and IP name space are stored, or DNS records. The content of these zone files is served by *authoritative name servers* to DNS clients, answering queries with the appropriate DNS records if the name server is authoritative for the queried data, or with a reference to another name server if not. Clients commonly rely on intermediate DNS server, referred to as resolving name servers or *resolvers*, to resolve DNS queries on their behalf while caching data locally to increase the performance of the DNS ecosystem as a whole.

The functionality of the DNS forms a fundamental basis for the workings of many other protocols on the Internet, including the Hypertext Transfer Protcol (HTTP) and Transport Layer Security (TLS). HTTP and TLS form the secure communication channel used by web browsers and web servers to communicate. Documents are exchanged by requesting specific locations identified by (among others) the domain name of the server that hosts the resource. TLS, providing the encryption layer to this exchange, relies on digital certificates exchanged between the browser and server which are used by clients to verify the authenticity of the web server [33]. These digital certificates embed a set of domain names in them, which denote the domains for which the certificate is valid. A certificate is signed by one of hundreds of trusted third-parties, the Certificate Authority (CA), who is tasked to verify the identity of an owner behind a certificate request prior to issuing the certificate. Due to two major incidents in which a CA erroneously issued a certificate [32,1], the Certificate Transparency (CT) was developed to audit the issuance behaviour of each CA [29]. In this framework, CAs submit newly-issued certificates to publicly available, append-only logs, and as such any third-party can monitor these logs.

3 Related work

Given the ubiquitous nature of domain names on the Internet, the vast majority of applications interact with them in one way or another, including applications with malicious intent. For example, phishing websites are commonly hosted on typosquatted domains (i.e., a catch-all term for domain names that are similar-looking to benign domains) [37] and bots within a botnet rely the DNS as a communication channel to dynamically identity the location of their bot masters [36]. As such, the security community has relied on data collected through domain name measurements in order to better understand and mitigate such types of malicious behaviour. Relying on network traces, including DNS and TLS, to build extensive domain-name related datasets has been an ongoing process for many years. Passive DNS was proposed in 2004 as a method to support DNS data recovery, by collecting DNS query in the wild, thereby being able to replicate the state of zone files at a particular point in time [40]. The ISC implemented a version of these ideas in 2012 [25], which resulted in the commer-

cialization of the framework in 2013 under the company Farsight Security [21]. The ENTRADA project focuses on collecting passive DNS traffic from the perspective of authoritative name servers instead [41]. Alternatively, researchers relied on active measurements for creating longitudinal dataset. The OpenIntel project collects a fixed set of DNS records for all apex domains within a set of TLD zones on a daily basis [34]. In their paper, the authors have collected data from three general TLD zone files comprising 50% of the apex domain name space, but have since then expanded to more general TLDs and sixteen country-code TLDs[3] [10]. Hohlfeld [27] expands upon this approach, by both collecting more than only DNS records (e.g., TLS support and particular TCP settings) and by relying more domain name sources besides zone files (i.e., passive DNS and domains extracted from CT logs). Similarly, Project Sonar scans the IPv4 address space for (among others) TLS certificates, reverse DNS misconfigurations, and various TCP and UDP services [15].

Reconstructing the full domain name space is a complex task due tot he distributed nature of the DNS. For many TLDs, access to zone files is controlled through the Centralized Zone Data Service (CZDS) [13], whereas access to other TLDs is more restricted, available ad-hoc involving NDAs [27], making a full replication of the apex domain space difficult. Several techniques have been used to circumvent this, such as "zone-walking" for DNSSEC-enabled zones [35,18] or abusing misconfigurations of DNS name servers, that allowed for a full zone file disclosure through zone transfer request [23]. Alternatively, these zones can be partially reconstructed by relying on other data sources, including certificate transparency [39] and the aforementioned passive DNS [40,25]. Even though passive DNS collects FQDNs (in addition to apex domains), mapping out the full FQDN domain name space is even more complex than the apex domain name space. The penetration testing community has relied on domain enumeration as one method of "reconnaissance", or information gathering about a particular target. As a result, a number of tools exists that support subdomain enumeration or DNS-based reconnaissance [5,7,6,2,3]. Typically, these subdomains are identified by scraping third-party sources that have collected this information prior (e.g., search engines) [22] or by generating candidate FQDNs [5,7,3].

Gollector stands apart because it is intended to collect passive domain-name related data from more vantage points than any of these research or commercial initiatives. Furthermore, the tool is unique in the sense that it specifically emphasizes the *differences* between vantage points, allowing us to evaluate the relative dark matter between each vantage point. Lastly, in contrast to tools from the penetration testing community, *Gollector* collects traffic from a global perspective, rather than focusing on a small set of individual domain names.

4 Vantage points

When passively collecting traffic in the DNS (i.e., capturing traffic generated by a client population, rather than generating own DNS traffic) the measurement

[3] as of July 2021

vantage point is a determining factor for what fraction of the total DNS traffic one can observe, as illustrated by Figure 1. Similarly, TLS traffic can be passively collected to observe the certificate (and other parameters) exchanged during a TLS handshake, which suffers from the same vantage point limitation as passive DNS collection. Alternatively, domain name related information can be extracted from other data sources that are not related to passive traffic. Beside active measurement – the process of actively probing servers for their responses to acquire information – sources are available that provide insight in the management and operation of a domain name. TLD zone files act as a ground-truth of the all domains registered directly under a TLD, and can be used to infer registrations and domain expirations. The CT framework is an alternative source of TLS certificates, as it provides researchers access to publicly available, append-only logs of newly-issued certificates by CAs. The logs guarantee that new certificate are published within a certain time frame – the Maximum Merge Delay (MMD) – such that the logs remain up to date with the latest issued certificates.

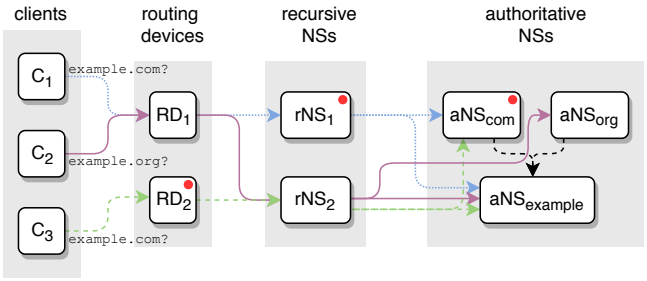


Fig. 1: The different vantage points (denoted by •) from which to conduct passive DNS measurements. Each arrow represents a DNS query send between two devices, where the colors indicate from which client the request originated. The various vantage points have different observations based on the querying behaviour of the three clients: RD_1 observes $\{C_1, C_2\}$, rNS_2 observes $\{C_1, C_3\}$ and $\mathrm{aNS}_{\mathrm{com}}$ observes $\{C_2, C_3\}$.

Even though these vantage points have their inherent differences, there is a commonality between them: the part of the domain name space they observe and the timing of those observations. Certain domain names or even full TLDs may be observed from one vantage point, but would never be observable from another. As such, a part of the domain name space can be considered dark matter for the latter. Out of our previously-discussed vantage points, we identify four vantage points that significantly differ from each other[4]: passive DNS from a resolver, passive DNS from an authoritative name server, CT logs and zone

[4] The difference from data collect from a routing device from a network operator and a DNS resolver may be insignificant if both vantage point are owned by the same party, in case of an ISP.

files. We can compare these vantage points according to the following properties. The WIDTH indicates across how many TLDs the vantage point is capable of capturing domains from. The DEPTH shows what part of the FQDN a vantage point is capable of collecting domain names from. Generally, it is either at an FQDN or an apex level. The TIME GRANULARITY represents the precision at which particular events are registered. A prime example of a vantage point with a large time granularity are zone files, which are published once a day, preventing the identification of precise additions to the zone file. A related dimension is the MAXIMUM TIME DELAY, which denotes how long it takes, worst case, for an event to be registered by a particular vantage point. Lastly, for those vantage points that passive collect data from a client population (i.e., passive DNS measurements), the POPULATION COVERAGE illustrates the type of population is being covered by the vantage point.

Table 1 shows an overview of the four vantage points and their dimensions. Both a DNS resolver and CT logs are capable of observing across a variety of TLDs, although it depends on the DNS client population and certificate issuers, respectively, which TLDs are actually observed. The domains covered by zonefile of a TLD are registered at an apex domain level, and thereby this vantage point does not cover FQDNs as opposed to the others. Furthermore, the zone files provided by the CZDS are updated on a daily basis [12], and therefore have a one-day granularity, whereas the other vantage points have a highly precise (i.e., sub-second) granularity. Every CT log operator defines a MMD, or maximum merge delay, which denotes the amount of time the operator will take as a maximum to publish newly-issued certificates to the log, which tends to be 24 hours. The one day granularity of zone files indicates that it can take up to a day for a newly-registered domain to appear in the zone file. Lastly, for the two passive DNS vantage points, there is difference in the DNS population coverage; an authoritative name servers receives global traffic for domains within their zone, whereas a resolver serves only a local, smaller population.

Table 1: Summary of vantage points

	Dimension				
Vantage points	WIDTH	DEPTH	TIME GRANULARITY	MAXIMUM TIME DELAY	POPULATION COVERAGE
CT logs	All TLDs	FQDN	Precise	MMD	–
Resolver	All TLDs	FQDN	Precise	–	Local
Authoritative NS	One TLD	FQDN	Precise	–	Global
TLD Zone file	One TLD	Apex	Daily	One day	–

5 Design

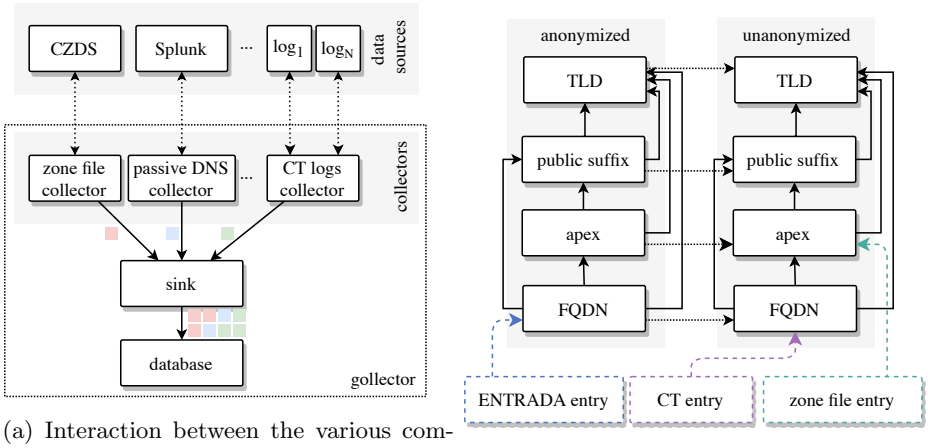
The current state of the art tooling lacks the possibility of conducting analyses between vantage points at a full domain name space scale. Based on this, we derived of a set of design goals that shaped the design and implementation of *Gollector*. We first present these design goals, followed by an overview of the architecture of *Gollector* describing how each of the individual goals is met.

Firstly, the main purpose of *Gollector* is to allow for the data collection of DNS and domain name related information from different vantage points (G_1). The design of the tool must allow the collection of from new vantage points to be added in a future point in time (G_2). Given the large number of existing domain names, and the volume of DNS and TLS data that is generated on the Internet, the tool should handle data collection at a large scale and remain highly performant (G_3). The data structures in which the data is stored must allow for post-collection analysis (G_4). The collection of DNS traces may contain sensitive information, which third-party data sources may be hesitant to share with researchers, and may only be willing to do so in an anonymized form. However, the anonymization of data may make post-collection analysis more difficult, and less detailed, and as such we would like to preserve the relationship between unanonymized data and anonymized data in *Gollector* (G_5).

5.1 Architecture

To meet G_1, the architecture of *Gollector* consists of modular components: (1) a set of data collectors, (2) a data sink and (3) a database for persistent data storage (see Figure 2a). Each individual collector is a small component dedicated to collecting domain name related data from one vantage point, and sending the resulting data to the sink. So far, we implemented four collectors (see 5.2). The collectors and the sink communicate securely using gRPC, allowing the collectors and sink to operate on different machines and thereby collectors to operate in different network environment (i.e., collect data from different vantage points). The decoupling of collectors from the other components of *Gollector* allows new collector modules to be developer in the future, thereby meeting G_2. The sink is designed to accept messages from the various collectors, extract database models from the messages and insert these models in the underlying database. As of now, we use PostgreSQL as the underlying database, as our database models naturally fit in a relational database model, and for future implementations we can switch to database intended for big-data analytics. The sink inserts new models in the database in batches rather than individual queries, resulting in a high-performance insertion rate (meeting G_3). Domain names are stored in the database as a collection of database models (see Figure 2b how the models relate to each other). Each collected FQDN is segmented in its parts, according to the domain name hierarchy (i.e., top level domain, its public suffix, the apex domain and the FQDN). Each segment is inserted as its separate row in its own table, and has a foreign key to all parts higher in the hierarchy. The data in this database is enriched by adding more tables with pointers (i.e., foreign

keys) to these domain-related tables; a timestamped certificate may point to an FQDN, whereas a timestamped zone file entry may point to an apex domain instead. This makes it easy to answer questions such as *"How many unique apex domains are observed under each TLD?"* or *"For a given apex domain, how many certificates have we seen?"*, and thereby fulfills G_4. Lastly, the segmented storage of domain names also applies to anonymized domain names. Instead of the storing the domain name directly, we store an anonymized version of each segment by hashing[5] the segment after appending a seed to the segment. The database maintains a mirrored set of tables for anonymized, including the foreign keys from segments lower in the hierarchy to upper segments. In order to analyze data collected in both its anonymized and unanonymized form, we link the two sets of tables by adding a reference from the anonymized table to the unanonymized table (meeting G_5). This link will only exist if a particular segment has been seen in both an anonymized dataset and an un-anonymized dataset, and thus will not apply to all observed domain names. This method does not provide full anonymity of domains, as the link between an anonymized segment and its un-anonymized form leaks the full segment, and the method is also prone to frequency analysis. However, in the tradeoff between anonymity and the potential analyses to conduct, we consider this acceptable.



(a) Interaction between the various components

(b) The database models.

Fig. 2: Design of *Gollector*

5.2 Collectors

Each collector registers *events* related to a domain name. These events range from individual DNS queries to domain registrations. Depending on the collector,

[5] using SHA256

a collector may generate only one or a few events per domain, or may generate many events over the course of a measurement. The current implementation of *Gollector* consists of the collectors described below.

Zone file collection This collector can fetch zone files from the Centralized Zone Data Service (CZDS) [13] and zone files over HTTP from servers that provide access. The former is an API provided by ICANN that allows for a standardized way to access zone files of over a thousand gTLDs including `.com` and `.net`, whereas the latter is used to fetch the Danish `.dk` TLD. In both cases, the authentication is handled by the collector, and is configured through a configuration file during startup. The collector automatically requests access for zone files on a daily basis when the granted access expires, ensuring that data collection continues during long measurements. All available zone file are then collected on a daily basis, and any changes between zone files of two consecutive days are tracked. Domain names that appear and disappear in the most recent zone file are considered new registrations and expirations/removals respectively. Furthermore, we collect all domain names observed on the first day, but do not considered these to be registrations or expirations. *Gollector* stores the zone files both raw on disk as well as in a processed form in the underlying database, so it allows researchers to work with the raw zone file if needed.

Passive resolver DNS *Gollector* itself does not perform any passive DNS measurements itself, but rather relies prior-collected datasets instead. The supported format for parsing passive DNS data is in Splunk Stream [9] export data, which consists of a condensed form of individual DNS request-response pairs in a JSON format. From the logs, *Gollector* extracts the queried domain name and timestamp of the resolution, omitting any DNS-specific information, such as query types or resolved IP addresses.

CT logs Each individual CT logs provides an HTTP API that can be used to fetch CT log entries. Such entries contain a full certificate chain of the newly-signed certificate, including the timestamp it was added to the log. *Gollector* traverses each log (as recognized by Google[6]), fetching all entries per log. This collector parses each entry, extracts the embedded domain names from the newly-signed certificate and stores them with a timestamp when the certificate was submitted to the CT logs. Furthermore, the certificates are stored in their raw format in the underlying database, allowing for further, more in-depth, investigation when necessary.

ENTRADA This collector interfaces with the dataset generated by ENTRADA, used to collect DNS resolutions at a authoritative name server level. This dataset comprises DNS-specific attributes of each resolution, such as the query type, the specific resolved IP address(es) and the IP addresses from which the query

[6] `https://github.com/google/certificate-transparency-community-site/blob/master/docs/google/known-logs.md`

originates. We summarize this information by collecting some basic statistics related to an individual domain name that has been queried; the first time and the last time the domain was queried. As such, the information in *Gollector* is far smaller in size than the source dataset, at the cost of losing details.

6 Dataset

We applied *Gollector* to four types of data sources, for which we implemented the aforementioned four collectors (Section 5.2). We collected the data over a time period of three weeks[7]. For our experiments, we collected the passive DNS traffic from our Danish university network (with 10s of thousands of users) and the ENTRADA data from the Danish `.dk` TLD. We collected our certificate data from all recognized CT logs and attempted to retrieve all TLDs available from CZDS. Figure 2 illustrates the unique number of events, FQDNs, apexes, public suffixes and TLDs observed per vantage point. Note that for the zone files, we only collected domains registrations and expirations, rather than all entries in the zone files. Since domains are registered at an apex domain level at a DNS registry, the collected zone files do not contain any FQDNs. Whereas our ENTRADA collector found the most unique FQDNs (161.5M), these FQDNs tend to be centralized under a relatively small set of TLDs (272) compared to the the other vantage points. Conversely, our passive DNS collector identified the smallest number of FQDNs (6.0M), which is unsurprising given the relatively small number and homogeneity of clients the university network serves (i.e., primarily Danish students and academic staff). The CT log data spans most TLDs (1,087), which comprises 72.6% of all recognized TLDs [14].

Table 2: Overview of the collected data, denoting the unique number of events, FQDNs, apexes, public suffixes and TLDs observed per vantage point.

	Events	FQDNs	Apexes	Public suffixes	TLDs
Zone files	8,371,731	–	7,920,217	572	607
CT logs	114,182,670[†]	89,989,143	27,807,193	4,222	1,087
Passive DNS	200,146,260	6,046,480	1,213,405	1,125	580
ENTRADA	161,497,905	161,497,905	124,318,163	328	272

[†]We identify a CT event as a uniquely observed certificate.

We hypothesize that the ENTRADA and the passive DNS traffic is highly biased towards Danish traffic. In order to test this hypothesis, we analyze the distribution of unique apexes found per TLD for each of the vantage points. Table 3 shows for each vantage points, the percentage of apexes identified by that

[7] Between February 1st, 2021 and February 21st, 2021

vantage point under a particular TLD, showing the top 10 TLDs per vantage point. The results show our hypothesis to be confirmed for ENTRADA, with nearly all apexes falling under the `.dk` TLD, whereas this is not the cases for the passive DNS traffic. The passive DNS traffic contains a large number of apexes under reserved TLDs for internal use (i.e., `mynet`, `my_net`, `home`, `lan` [24]), and the `.com` TLD is more popular than the Danish TLD. The CT and zone file datasets are more in line with the general size of the TLDs; `.com` and `.net` are the largest TLDs.

Table 3: The top ten TLDs per vantage point in terms of unique number of apex domains identified under the TLDs.

CT		Zones		Passive		ENTRADA	
TLD	%	TLD	%	TLD	%	TLD	%
com	44.3%	com	59.2%	mynet	23.9%	dk	100%
tk	4.5%	icu	10.7%	my_net	22.0%	arpa	0.0%
de	4.4%	net	4.4%	home	19.1%	com	0.0%
net	3.9%	xyz	3.9%	lan	16.5%	org	0.0%
org	2.9%	org	3.1%	com	4.9%	net	0.0%
uk	2.3%	wang	2.4%	dk	2.0%	se	0.0%
ru	1.6%	page	2.2%	localdomain	1.2%	0	0.0%
nl	1.6%	site	1.8%	net	1.1%	0	0.0%
br	1.5%	bar	1.0%	dlinkrouter	0.7%	0	0.0%
it	1.4%	club	0.8%	org	0.6%	0	0.0%

7 Use cases

We demonstrate the utility of *Gollector* by diving deeper into three use cases. Firstly, we evaluate the impact of the temporal differences of the four vantage points by analyzing how effective they are in recognizing newly-registered domains. Additionally, we leverage the relative dark matter differences between passive DNS measurements from a resolver and an authoritative name server perspective to investigate the split-horizon setup of our local university network. These two use cases showcase the benefits of multiple vantage points. Lastly, we leverage the full set of FQDNs for a domain name generation algorithm, as an alternative to brute-force subdomain enumeration techniques employed by penetration testing tools.

7.1 Early detection of domain names

Various malicious actors rely on domain registrations for their operations, such as botnet operators (for domain fluxing) and phishers (for typo-squatting and hosting phishing sites in general). Prior work has demonstrated that the involved

domains tend to be abused within a few days after their registration, after which they have already served their (malicious) purpose [42]. From a defense perspective, identifying such domains in the early part of their lifecycle is therefore of critical importance. A domain registration can be detected at different points of time depending on the vantage point. Zone files are a logical choice, as they originate from the party that registers new domains (i.e., registries) but have as a limitation that they are created with a one-day granularity[8]. We investigate if other vantage points provide a more accurate – and thereby earlier – time of registration, especially focusing on CT logs, as they cover all TLDs rather than just one (which is the case for ENTRADA and to a lesser extent the passive DNS from university network).

We identified a registration of 4,438,966 domains over the course of 20 days[9] for an average of 221,948 domains per day. For each of these registrations, we identify if the other three vantage (i.e., CT logs, passive DNS and ENTRADA) observed the domains as well. Firstly, we identify if these vantage points observed the domain registrations *at all* in the full timeline. This serves as an indication to what extent domain registration remain undetected and the coverage of the domain name space the vantage points have compared to zone files. We follow this up by identifying which these domains were detected *before* the zone files registered these domains. For those domains, the one-day granularity of zone files is surpassed by the granularity of the other vantage point. Lastly, we identify the domains that were detected *within seven days* after the zone file identified the domain being registered. Wullink *et al* [42] showed that phishing domains tend to be most active within the first seven days of their registration (based on the DNS traffic the domain receives). Therefore, identifying such domain registrations within seven days is highly beneficial for mitigation these attacks.

Table 4 shows the results, both in absolute numbers and the percentage of the total number of zone file registrations. Since the ENTRADA dataset operates at a single TLD's name server, we differentiate between the full dataset and the dataset for the `.dk` domains only. Across all TLDs, the CT logs has a relatively high coverage, with almost one in four domain registrations being detected across the whole dataset. The passive DNS and ENTRADA dataset have a much smaller coverage, with only 0.01% and 1.13% of domain registrations being detected. Notably, CT logs provide an earlier detection of domains compared to newly zone files for 13.79% of domains. When looking at the `.dk` domain only, all vantage points detect more registrations than the full dataset, with ENTRADA detecting nearly all registrations. Furthermore, of the domain registrations detected by the passive DNS dataset, almost all of them were `.dk` domains (with only 133 non-Danish registrations detected). In none of the cases, the percentage of identified registrations was significantly improved by including the first seven days after registration. For identifying new domain registrations,

[8] Registries will have access to more accurate registration data than just the zone files, so this is a limitations for researcher who only have access to the zone files

[9] Since a registration is detected by computing the difference of the zone files of two subsequent days, we are missing the registrations of the first day of our measurement.

zone files are still primarily but this can be supported by CT logs and ENTRADA for individual TLDs.

Table 4: Detection of newly registered domain names for non-zone files vantage points. The results for both the full set of TLDs and the `.dk` zone only are shown.

	All TLDs					
	Absolute			Percentual		
	CT	Passive	ENTRADA	CT	Passive	ENTRADA
Overall	971,318	533	46,628	23.6%	0.01%	1.1%
Before	568,436	216	25,713	13.8%	0.01%	0.62%
Within 7 days	325,277	169	4688	7.9%	0.00%	0.11%
	`.dk` only					
Overall	16,476	63	46,495	34.9%	0.13%	98.5%
Before	0	0	25,673	0.00%	0.00%	54.4%
Within 7 days	639	3	4,601	1.35%	0.01%	9.74%

7.2 Split horizon and data leakage

Large organisations commonly operate a *split-horizon* DNS infrastructure, where DNS resolutions receive different responses depending on the location of the requester. Use cases include load balancing, or protecting sensitive information that should only be accessible from within a corporate network [19]. Furthermore, the exposure of the existence of a particular hostname can already provide insight in an organisations inner workings and should potentially be protected against. We leverage our passive DNS data and ENTRADA dataset – both relatively biased towards the `dk` TLD – to investigate potential data leakage in our dataset. We identify which domain names are likely to be only used internally and what data is leaked outside the network through DNS queries. The split-horizon set up should result in particular domain names only being queried within the university network.

As a first step, we identify what apex domain names are likely to be owned by the university network. We assume that internal apex domains are heavily used for various services within the network, thereby having many different FQDNs in use. As such, we collect the unique number of FQDNs observed under each apex domain in our dataset. Table 5 shows this count for the 10 most prevalent apex domains, and furthermore also shows percentage of total FQDNs observed in the passive DNS dataset it encompasses. Clearly, FQDNs under `aau.dk` are seen most often (more than 63% of observed FQDNs fall under this apex), suggesting that this domain is used for internal systems within the network. Indeed, this domains is owned by the university, whereas the other domains in the table are related to background services such as advertisement/analytics (e.g.,

`googlesyndication.com`, `cedexis-radar.net`), or network management (e.g., `bbsyd.net`, `emnet.dk`), and are not associated with the university.

From our ENTRADA dataset, we found 18,499 FQDNs under the `aau.dk` apex domain, a much lower number than the 3,8M seen in the passive DNS dataset. Not all of these domain names are necessarily sensitive information, as some of the domains used by the university are likely used to host public websites. Therefore, we turn to the domains that have both been seen by the passive DNS and the ENTRADA dataset: this set of domains comprises 2,813 FQDNs, or 15.2% of the `aau.dk` FQDNs seen in the ENTRADA dataset. Since we have no ground-truth of what is a sensitive domain and what is not, we compare this list of domains to the most common subdomains instead [8]. We found 435 (or 15.5%) of these domains is in the public list, leaving more than 2,300 subdomains potentially leaked.

Table 5: The 10 apex domains with the most observed unique FQDNs in the passive DNS dataset collected from the university network.

Apex domain	Unique FQDN count	%
`aau.dk`	3,829,837	63%
`googlesyndication.com`	344,058	6%
`technicolor.net`	61,151	1.01%
`cedexis-radar`.net	44,771	0.74%
`sophosxl.net`	39,297	0.65%
`bbsyd.net`	36,758	0.61%
`office.com`	30,215	0.50%
`emnet.dk`	23,540	0.39%
`obelnet.dk`	22,909	0.38%
`webspeed.dk`	21,569	0.36%

7.3 Subdomain enumeration

Part of the reconnaissance phase in penetration testing is subdomain enumeration, or the process of identifying all subdomains under a given apex domain. Typically, these subdomains are identified by scraping third-party sources that have collected this information prior (e.g., CT logs, search engines) [22] or by generating candidate FQDNs [5,7,3]. *Gollector* can support the former, as its database model allows to easily query all FQDNs observed under a particular apex domain. We present a method to support the latter as well. As opposed to the existing brute-forcing techniques, we infer a relationship between sets of subdomains, based on the co-occurence of these subdomains under a shared set of apex domains. As a result, our proposed method generates accurate candidate FQDNs and identifies relationships between subdomains that otherwise would

not be found. This inference is motivated by particular use case in which subdomains are likely to co-occur under the same apex domain; `cPanel` defines a set of *Service Subdomains*, or subdomains exposed by cPanel to provide interfaces to external components [17]. Therefore, the existence of a `cPanel` subdomain may indicate that the other *Service Subdomains* are also "in use" under the particular apex domain, even if a DNS dataset has not identified the existence. Our proposed method consists of the following steps: (1) we convert our dataset of subdomains and apex domains in a graph, (2) we compute a clique cover of this graph, (3) we prune these cliques according to the weights in each individual clique to filter out nodes that are not relevant to the clique, and (4) we generate a set of candidate FQDNs, based on the pruned cliques.

As a first step, we split up our set of FQDNs into their subdomain and apex parts, and subsequently create a graph in which the subdomains are modeled as nodes. Edges between two subdomains express the measure of overlap of the sets of apex domains under which both subdomains have been seen. The edge weight is computed as the Jaccard index [28] of the set of apex domains under which the first subdomain has been seen and the set of apex domains under which the second subdomain has been. We prune the edges that have a weight of zero (i.e., between subdomains that are never seen under the same apex domain), and remove any nodes that are without edges (i.e., subdomains never seen under the same apex domain as another subdomain).

We split up the nodes in our graph into a *clique cover*. Cliques are induced subgraphs such that each node is adjacent to all other nodes in the subgraph. This implies that every subdomain within a clique has been observed under the same apex domain with all other subdomains at least once. By assigning each node to a clique we reach a clique cover, which we achieve by relying on the algorithm defined in Appendix A[10].

A clique cover ensures every subdomains falls in a clique, but this does not guarantee there is a strong connection between the nodes within the clique. Therefore, we prune each clique to remove nodes that are not considered relevant. We scale the edge weights in each clique such that the highest weight equals 1, and then prune the nodes whose maximum edge weight falls under a given threshold. In our experiments, we used a threshold value of 0.6, which we found through thorough experimentation.

For each clique, we can now generate a set of candidate FQDNs. We maintain a set for each subdomain, denoting the apex domains under which the subdomain has been observed, based on all FQDNs in our dataset. For each clique, we define the set of apex domains that *any* of the subdomains in the clique has been observed under. The Cartesian product of the apex domains and subdomains then forms the tuples of apex domains and subdomains representing the candidate FQDNs. The FQDNs already seen in data are left out from this set, forming the final set of candidate FQDNs.

[10] There are potentially many clique covers, and our purpose is not to achieve a minimal clique cover

Results We applied this methodology to a dataset of 2 million randomly sampled FQDNs from our dataset, In total, we identified 8,410 cliques comprising 22,519 subdomains. Appendix B shows several examples of subdomains that form a clique. These subdomains were previously seen under a set of 1,021,175 apex domains. Given our cliques, we generated 2,349,911 FQDN candidates, resulting in an average of only 2.3 FQDNs per apex domain. Out of these candidates, we could successfully resolve 1,396,129 FQDNs, or 59% of candidates. Additionally, we also manually investigated some of the cliques in order to understand what the nature of these cliques are. This manual investigation was far from exhaustive, but we found cliques related to the software that runs on these domains (such as the `cPanel` example that drove this research) and cliques pointing to a specific organisations. An example of the former clique type are cliques for subdomains used by Magento, a highly-popular open-source eCommerce platform [4]. We identified 470 cliques related to this platform with subdomains containing the keyword `magento`, often having `shop` or `store` as another keyword being embedded in one of the subdomains. The latter type include a clique formed by subdomains under the apex domains `fbcdn.net` and `whatsapp.net`, containing 118 subdomains, indicating the relationship between Whatsapp and Facebook.

Our proposed can be integrated in existing penetration testing tools as an alternative to wordlist-based domain generators. On top of that, our cliques can be used to identify shared domain name ownership, and to assist security researchers in identifying domains hosting the same services.

8 Conclusions

In this paper, we introduced *Gollector* as a novel platform for collecting domain name and DNS-related information. Through a thorough overview of the DNS and TLS ecosystem, we present a set of vantage points from which this information can be retrieved. Through three uses cases, we leverage the differences between these vantage points. Firstly, we show that that CT logs and passive DNS traffic collected at an authoritative name server can serve as a source for early domain registration detection. Zone files are outperformed by the CT logs outperform in 13.8% of domains under all TLDs, and by the passive authoritative traffic in 54.4% of domains under the `.dk` TLD. Secondly, we compare passive DNS measurements from a university network with authoritative name server measurements to shed light on potential data leakage of subdomains under the main domain name in use by the university. Lastly, we present a method to generate potentially-existing FQDNs, which infers these FQDNs based on the association of subdomains and apex domains.

Acknowledgements

This research was carried out under the SecDNS project, funded by Innovation Fund Denmark. We would like to express our gratitude to Finn Büttner and Erwin Lansing for their assistance in collecting our passive DNS datasets.

Appendix A Clique cover algorithm

Algorithm 1 denotes the algorithm used to compute a clique cover for graph G. The intuition behind the algorithm is that two nodes – connected through an edge with a largest weight – have the largest priority to form a clique. The algorithm iterates of all edges in the graph assigns a clique to each node in the graph based on the interactions that are observed through the edges. Depending on whether the source and destination nodes of the edge are already in a clique, the algorithm creates new cliques, adds nodes to existing cliques or merges cliques together. The output of the algorithm is a hashmap of the clique assigned to each node in the graph. The implementation of the algorithm includes several optimizations to reduce the edges to evaluate.

Appendix B Examples of cliques

Table 6 contains several examples cliques. The table shows a general description of the what the subdomains may be intended for, the number of subdomains in the clique, the number of apexes associated with these subdomains, and the list of subdomains comprised by the clique.

References

1. Comodo SSL affiliate the recent RA compromise. https://blog.comodo.com/other/the-recent-ra-compromise/, accessed: 2021-07-23
2. DNSdumpster. https://dnsdumpster.com/, accessed: 2021-07-10
3. DSNRecon. https://github.com/darkoperator/dnsrecon, accessed: 2021-07-10
4. Magento. https://magento.com/, accessed: 2021-07-27
5. OWASP/Amass. https://github.com/OWASP/Amass, accessed: 2021-07-10
6. Subfinder. https://github.com/projectdiscovery/subfinder, accessed: 2021-07-10
7. Sublist3r. https://github.com/aboul3la/Sublist3r, accessed: 2021-07-10
8. The most popular subdomains on the internet. https://bitquark.co.uk/blog/2016/02/29/the_most_popular_subdomains_on_the_internet (2016), accessed: 2021-07-27
9. About Splunk stream. https://docs.splunk.com/Documentation/StreamApp/7.3.0/DeployStreamApp/AboutSplunkStream (2020), accessed: 2021-07-10
10. Openintel - current coverage. https://openintel.nl/coverage/ (2020), accessed: 2021-07-10
11. Using GeoIP with BIND 9. https://kb.isc.org/docs/aa-01149 (2020), accessed: 2021-07-10

Algorithm 1: Clique cover algorithm

```
1  Function cliqueCover (G);
   Input  : graph G of subdomain nodes
   Output: set of subdomain lists
2  edges = edgeListFrom(G);
3  edges = sortByWeight(edges);
4  cliques = {};
5  for edge in edges do
6  |  src, dst = nodes in edge;
7  |  cliqueSrc = cliques[src];
8  |  cliqueDst = cliques[dst];
9  |  if src not in clique and dst not in clique then
   |  |  /* both are without clique, create a new one                    */
10 |  |  c = newClique(src, dst);
11 |  |  cliques[src] = c;
12 |  |  cliques[dst] = c;
13 |  else if src in same clique as dst then
   |  |  /* src and dst are already in the same clique                   */
14 |  else if src not in clique and dst in clique then
   |  |  /* try to add dst to cliqueSrc                                  */
15 |  |  if cliqueSrc.formsCliqueWith(dst) then
16 |  |  |  cliques[src].add(dst);
17 |  |  end
18 |  else if src in clique and dst not in clique then
   |  |  /* try to add src to cliqueDst                                  */
19 |  |  if cliqueDst.formsCliqueWith(src) then
20 |  |  |  cliques[dst].add(src);
21 |  |  end
22 |  else if src and dst in different cliques then
   |  |  /* try to merge the two cliques                                */
23 |  |  if cliqueSrc.formsCliqueWith(cliqueDst) then
24 |  |  |  c = mergeCliques(cliqueSrc, cliqueDst);
25 |  |  |  cliques[src] = c;
26 |  |  |  cliques[dst] = c;
27 |  |  end
28 end
29 return cliques;
```

Table 6: Examples of cliques

Description	Subdomain count	Apex count	Subdomains
High-entropy subdomains	237	2	`adfqjkxr`, `aeovrpvk`, `anhpfctcxzcp`, `asqzcggxiy`, `bdzvxofezaejku`, ...
Email servers	5	34,249	`imap`, `xwa`, `xas`, `pop`, `smtp`
Western language-related subdomains	7	26,730	`en`, `es`, `fr`, `pt`, `it`, `ru`, `de`
More language-related subdomains	6	3,764	`ko`, `zh`, `cs`, `nl`, `ar`, `ja`
Content deliver network	9	5,197	`cdn-1`, `cdn-3`, `cdn-2`, `cdn-5`, `cdn-7`, ...

12. About zone file access. https://www.icann.org/resources/pages/zfa-2013-06-28-en (2021), accessed 30-08-2021
13. Centralized zone data service. https://czds.icann.org/ (2021), accessed 30-08-2021
14. List of top-level domains. https://www.icann.org/resources/pages/tlds-2012-02-25-en (2021), accessed 30-08-2021
15. Project sonar. https://opendata.rapid7.com/about/ (2021), accessed: 2021-07-10
16. Public suffix list. https://publicsuffix.org/ (2021), accessed: 2021-07-10
17. value. https://documentation.cpanel.net/display/CKB/Service+Subdomains+Explanation (2021), accessed 30-08-2021
18. van Adrichem, N.L.M., Blenn, N., Lúa, A.R., Wang, X., Wasif, M., Fatturrahman, F., Kuipers, F.A.: A measurement study of DNSSEC misconfigurations. Security Informatics **4**(1) (oct 2015). https://doi.org/10.1186/s13388-015-0023-y, https://doi.org/10.1186%2Fs13388-015-0023-y
19. Aitchison, R.: DNS Techniques, pp. 163–207. Apress, Berkeley, CA (2011). https://doi.org/10.1007/978-1-4302-3049-6_8, https://doi.org/10.1007/978-1-4302-3049-6_8
20. Alieyan, K., Almomani, A., Manasrah, A., Kadhum, M.M.: A survey of botnet detection based on DNS. Neural Computing and Applications **28**(7), 1541–1558 (2017)
21. Behjat, A.: ISC spins off its security business unit. https://www.isc.org/blogs/isc-spins-off-its-security-business-unit/ (2013)
22. Bharath: A penetration tester's guide to sub-domain enumeration. https://blog.appsecco.com/a-penetration-testers-guide-to-sub-domain-enumeration-7d842d5570f6 (2018), accessed: 2021-07-24
23. Borges, E.: Wrong Bind configuration exposes the complete list of russian TLD's to the Internet. https://securitytrails.com/blog/russian-tlds (March 2018), accessed 30-08-2021
24. Eastlake, D., Panitz, A.: Reserved top level DNS names. BCP 32, RFC Editor (June 1999)
25. Edmonds, R.: ISC passive DNS architecture. https://mirror.yongbok.net/isc/kb-files/passive-dns-architecture.pdf (2012)
26. Hao, S., Kantchelian, A., Miller, B., Paxson, V., Feamster, N.: PREDATOR: Proactive recognition and elimination of domain abuse at time-of-registration. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 1568–1579. CCS '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2976749.2978317, https://doi.org/10.1145/2976749.2978317
27. Hohlfeld, O.: Operating a DNS-based active internet observatory. In: Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos. pp. 60–62. SIGCOMM '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3234200.3234239, https://doi.org/10.1145/3234200.3234239
28. Jaccard, P.: Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. Bull Soc Vaudoise Sci Nat **37**, 241–272 (1901)
29. Laurie, B., Langley, A., Kasper, E.: Certificate transparency. RFC 6962, RFC Editor (June 2013)

30. Mockapetris, P.: Domain names - implementation and specification. STD 13, RFC Editor (November 1987), http://www.rfc-editor.org/rfc/rfc1035.txt, http://www.rfc-editor.org/rfc/rfc1035.txt
31. Pearce, P., Jones, B., Li, F., Ensafi, R., Feamster, N., Weaver, N., Paxson, V.: Global measurement of DNS manipulation. In: 26th USENIX Security Symposium (USENIX Security 17). pp. 307–323. USENIX Association, Vancouver, BC (Aug 2017), https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/pearce
32. Prins, J.: DigiNotar certificate authority breach "operation black tulip". https://media.threatpost.com/wp-content/uploads/sites/103/2011/09/07061400/rapport-fox-it-operation-black-tulip-v1-0.pdf (2011), accessed: 2021-07-23
33. Rescorla, E.: The transport layer security (TLS) protocol version 1.3. RFC 8446, RFC Editor (August 2018)
34. van Rijswijk-Deij, R., Jonker, M., Sperotto, A., Pras, A.: A high-performance, scalable infrastructure for large-scale active DNS measurements. IEEE Journal on Selected Areas in Communications **34**(6), 1877–1888 (2016). https://doi.org/10.1109/JSAC.2016.2558918
35. Schlyter, J.: DNS security (DNSSEC) NextSECure (NSEC) RDATA format. RFC 3845, RFC Editor (August 2004)
36. Singh, M., Singh, M., Kaur, S.: Issues and challenges in DNS based botnet detection: A survey. Computers & Security **86**, 28–52 (2019). https://doi.org/https://doi.org/10.1016/j.cose.2019.05.019, https://www.sciencedirect.com/science/article/pii/S0167404819301117
37. Szurdi, J., Kocso, B., Cseh, G., Spring, J., Felegyhazi, M., Kanich, C.: The long "taile" of typosquatting domain names. In: 23rd USENIX Security Symposium (USENIX Security 14). pp. 191–206. USENIX Association, San Diego, CA (Aug 2014), https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/szurdi
38. van der Toorn, O., van Rijswijk-Deij, R., Geesink, B., Sperotto, A.: Melting the snow: Using active DNS measurements to detect snowshoe spam domains. In: NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium. pp. 1– 9 (2018). https://doi.org/10.1109/NOMS.2018.8406222
39. VanderSloot, B., Amann, J., Bernhard, M., Durumeric, Z., Bailey, M., Halderman, J.A.: Towards a complete view of the certificate ecosystem. In: Proceedings of the 2016 Internet Measurement Conference. pp. 543–549. IMC '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2987443.2987462, https://doi.org/10.1145/2987443.2987462
40. Weimer, F.: Passive DNS replication. In: FIRST Conference on Computer Security Incident (2005)
41. Wullink, M., Moura, G.C.M., Müller, M., Hesselman, C.: Entrada: A high-performance network traffic data streaming warehouse. In: NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium. pp. 913–918 (2016). https://doi.org/10.1109/NOMS.2016.7502925
42. Wullink, M., Muller, M., Davids, M., Moura, G.C.M., Hesselman, C.: ENTRADA: enabling DNS big data applications. In: 2016 APWG Symposium on Electronic Crime Research (eCrime). pp. 1–11 (2016). https://doi.org/10.1109/ECRIME.2016.7487939