Encrypted DNS $\implies$ Privacy? A Traffic Analysis Perspective

Sandra Siby
EPFL

Marc Juarez
imec-COSIC KU Leuven

Claudia Diaz
imec-COSIC KU Leuven

Narseo Vallina-Rodriguez
IMDEA Networks

Carmela Troncoso
EPFL

***Abstract*—Virtually every connection to an Internet service is preceded by a DNS lookup which is performed without any traffic-level protection, thus enabling manipulation, redirection, surveillance, and censorship. To address these issues, large organizations such as Google and Cloudflare are deploying recently standardized protocols that encrypt DNS traffic between end users and recursive resolvers such as DNS-over-TLS (DoT) and DNS-over-HTTPS (DoH). In this paper, we examine whether encrypting DNS traffic can protect users from traffic analysis-based monitoring and censoring. We propose a novel feature set to perform the attacks, as those used to attack HTTPS or Tor traffic are not suitable for DNS' characteristics. We show that traffic analysis enables the identification of domains with high accuracy in closed and open world settings, using 124 times less data than attacks on HTTPS flows. We find that factors such as location, resolver, platform, or client do mitigate the attacks performance but they are far from completely stopping them. Our results indicate that DNS-based censorship is still possible on encrypted DNS traffic. In fact, we demonstrate that the standardized padding schemes are not effective. Yet, Tor — which does not effectively mitigate traffic analysis attacks on web traffic— is a good defense against DoH traffic analysis.**

I. Introduction

Regular Domain Name System (DNS) requests have been mostly sent in the clear [1]. This situation enables entities, such as Internet Service Providers (ISPs), Autonomous Systems (ASes), or state-level agencies, to perform user tracking, mass surveillance [2, 3] and censorship [4, 5]. The risk of pervasive surveillance and its consequences has prompted Internet governance, industry actors, and standardization bodies to foster privacy protections [6, 7]. In particular, for DNS, these bodies have standardized two protocols: DNS-over-TLS (DoT) [8] and DNS-over-HTTPS (DoH) [9]. These protocols encrypt the communication between the client and the recursive resolver to prevent the inspection of domain names by network eavesdroppers. These standarization bodies also consider protection mechanisms to limit inference of private information from traffic metadata, such as the timing and size of network packets, of the encrypted DNS communication.

These mechanisms protect against traffic analysis by padding traffic [10], or by multiplexing the encrypted DNS traffic with other traffic, *e.g.*, when DoH and web HTTPS traffic share a single TLS tunnel (see §8.1 [9]).

During 2018, Google and Cloudflare launched public DoH resolvers [11, 12], while Mozilla added DoH support to Firefox [13], which will become the default DNS protocol version for US users in September 2019 [14]. These efforts aim to leverage DoH and DoT's capacity to enhance users' browser traffic security guarantees [15]. Yet, it is known that even when communications are encrypted, traffic features such as volume and timing can reveal information about their content [16, 17, 18, 19, 20, 21], and whether measures like padding are sufficient to protect users against eavesdroppers. As of today, existing evaluations of DoH implementations have focused on understanding the impact of encryption and transport protocols on performance [22, 23], and on cost [24, 25].

In this paper, we aim to fill this gap by studying the effectiveness of traffic analysis attacks in revealing users' browsing patterns from encrypted DNS. We focus our analysis on DoH, as its adoption by large industry actors (e.g., Google, Cloudflare, and Mozilla) makes it prevalent in the wild. For completeness, we include a comparison of the protection provided by Cloudflare's DoH and DoT resolvers. In our analysis, we consider a passive adversary, as described in RFC 7626 [6], who is placed between the client and the DNS resolver. The adversary's goal is to identify which web pages users visit, to either perform surveillance or censorship. As the RFC stresses, this adversary may be in "a different path than the communication between the initiator [*e.g.*, the client] and the recipient [*e.g.*, a web server]" [6], and thus can launch attacks even if they do not see all the traffic between endpoints.

We find that features traditionally used in atttacks on web traffic [16, 17, 18, 19, 26, 27] are not suitable for encrypted DNS traffic. As opposed to web traffic, DNS traffic —even if encrypted— is bursty, chatty, and is mostly composed of small packets. We engineer a *novel set of features* that, by focusing on local traffic features, enables successful attacks that identify requested websites on encrypted DNS. As encrypted DNS traces are much smaller that their web traffic counterparts, our techniques require *124 times less data than state-of-the-art traffic analysis on web traffic*, which is key to ensure scalability of attacks [28]. Furthermore, our new feature set on encrypted DNS traffic is *as effective or more so* than state-of-the-art attacks on web traffic in identifying web pages.

We also find that differences between the environment used by the adversary to train the attack (e.g., location, choice of client application, platform or recursive DNS resolver), and the environment where the attack is actually deployed, negatively affect the performance of the attack. Most prior work on traffic analysis assumes the adversary knows the environment where the attack will be deployed, but the adversary cannot trivially obtain that information a priori. Our features allow the adversary to infer that information and thus tailor the attacks accordingly, maintaining high attack performance for each specific environment.

Next, we evaluate existing traffic analysis defenses, including the standardized EDNS0 padding [10] —implemented by Cloudflare and Google in their solutions—, and the use of Tor [29] as transport, a feature available when using Cloudflare's resolver. We find that, unless EDNS0 padding overhead is large, current padding strategies cannot completely prevent our attack. Also, while Tor offers little protection against web page fingerprinting on web traffic [17, 18, 19, 30], Tor is an extremely effective defense against web page fingerprinting on encrypted DNS traffic.

Finally, we measure the potential of encryption to hinder DNS-based censorship practices. We show that under encryption, it is still possible to identify which packet carries the DNS lookup for the first domain. We quantify the collateral damage of blocking the response to this lookup, thereby preventing the user from seeing any content. We also show that, to minimize the effect of censorship on non-blacklisted websites, censors must wait to see, on average, 15% of the encrypted DNS traces.

Our main contributions are as follows:

- We show that the features for traffic analysis existing in the literature are not effective on encrypted DNS. We propose a new feature set that results in successful attacks on encrypted DNS and that outperforms existing attacks on HTTPS (Section V-A).
- We show that web page fingerprinting on DoH achieves the same accuracy as web page fingerprinting on encrypted web traffic, while requiring *124* times less volume of data. We show that factors such as end-user location, choice of DNS resolver, and client-side application or platform, have a negative impact on the effectiveness of the attacks but do not prevent them (Section V).
- We evaluate the defenses against traffic analysis proposed in the standard and show that they are not effective. We find that in the case of encrypted DNS, contrary to web traffic, routing over Tor deters web page identification on encrypted DNS traffic (Section VI).
- We evaluate the feasibility of DNS-based censorship when DNS lookups are encrypted. We show that the censor can identify the packet with the first domain lookup. We quantify the tradeoff between how much content from a blacklisted site the user can download, and how many non-blacklisted websites are censored as a side effect of traffic-analysis-based blocking (Section VII).
- We gather the first dataset of encrypted DNS traffic collected in a wide range of environments (Section IV).[1]

Impact Upon responsible disclosure of our attacks, Cloudflare changed their DoH resolver to include padding. This work was also invited to an IETF Privacy Enhancements and Assessments Research Group Meeting and will contribute to the next RFC for traffic analysis protection of encrypted DNS.

II. Background and Related Work

In this section, we provide background on the Domain Name System (DNS) and existing work on DNS privacy.

The Domain Name System (DNS) is primarily used for translating easy-to-read domain names to numerical IP addresses [2]. This translation is known as domain resolution. In order to resolve a domain, a client sends a DNS query to a *recursive resolver*, a server typically provided by the ISP with resolving and caching capabilities. If the domain resolution by a client is not cached by the recursive name server, it contacts a number of *authoritative name servers* which hold a distributed database of domain names to IP mappings. The recursive resolver traverses the hierarchy of authoritative name servers until it obtains an answer for the query, and sends it back to the client. The client can use the resolved IP address to connect to the destination host. Figure 1 illustrates this process.

Enhancing DNS Privacy. Security was not a major consideration in the first versions of DNS, and for years DNS traffic was sent in the clear over (untrusted) networks. Over the last few years, security and privacy concerns have fostered the appearance of solutions to make DNS traffic resistant to eavesdropping and tampering. Several studies have empirically demonstrated how the open nature of DNS traffic is being abused for performing censorship [33, 34] and surveillance [35, 2]. Early efforts include protocols such as DNSSEC [36] and DNSCrypt [37]. DNSSEC prevents manipulation of DNS data using digital signatures. It does not, however, provide confidentiality. DNSCrypt, an open-source effort, provides both confidentiality and authenticity. However, due to lack of standardization, it has not achieved wide adoption.

In 2016, the IETF approved DNS-over-TLS (DoT) [8] as a Standards Track protocol. The client establishes a TLS session with a recursive resolver (usually on port TCP:853 as standardized by IANA [8]) and exchanges DNS queries and responses over the encrypted connection. To amortize costs, the TLS session between the client and the recursive DNS resolver is usually kept alive and reused for multiple queries. Queries go through this channel in the same manner as in unencrypted DNS – chatty and in small volume.

In DoH, standardized in 2018, the local DNS resolver establishes an HTTPS connection to the recursive resolver and encodes the DNS queries in the body of HTTP requests. DoH considers the use of HTTP/2's Server Push mechanism. This enables the server to preemptively push DNS responses that are likely to follow a DNS lookup [38], thus reducing communication latency. As opposed to DoT, which uses a dedicated TCP port for DNS traffic and thus it is easy to monitor and block, DoH lookups can be sent along non-DNS traffic using existing HTTPS connections (yet potentially blockable at the IP level).

There are several available implementations of DoT and DoH. Since 2018, Cloudflare and Quad9 provide both DoH and DoT resolvers, Google supports DoH, and Android Pie has native support for DoT. DoH enjoys widespread support from browser vendors. Firefox provides the option of directing DNS traffic to a *trusted recursive resolver* such as a DoH resolver, falling back to plaintext DNS if the resolution over DoH fails. In September 2019, Google announced support for DoH in version 78 of Chrome [39]. Cloudflare also distributes a stand-alone DoH client and, in 2018, they released a hidden resolver that provides DNS over Tor, not only protecting lookups from eavesdroppers but also providing anonymity for clients towards the resolver. Other protocols, such as DNS-over-DTLS [40], an Experimental RFC proposed by Cisco in 2017, and DNS-over-

[1] Our dataset and code will be made public upon acceptance.

[2] Over time, other applications have been built on top of DNS [31, 32]

QUIC [41], proposed to the IETF in 2017 by industry actors, are not widely deployed so far.

Several academic works study privacy issues related to encrypted DNS. Shulman suggests that encryption alone may not be sufficient to protect users [42] but does not provide any experiments that validate this statement. Our results confirm her hypothesis that encrypted DNS response size variations can be a distinguishing feature. Herrmann et al. study the potential of DNS traces as identifiers to perform user tracking but do not consider encryption [43]. Finally, Imana et al. study privacy leaks on traffic between recursive and authoritative resolvers [44]. This is not protected by DoH and it is out of scope of our study.

III. Problem Statement

In this paper, we study if it is possible to infer which websites a user visits by observing encrypted DNS traffic. This information is of interest to multiple actors, *e.g.*, entities computing statistics on Internet usage [45, 46], entities looking to identify malicious activities [47, 48, 5], entities performing surveillance [35, 2], or entities conducting censorship [49, 34].

We consider an adversary that can collect encrypted DNS traffic between the user and the DNS recursive resolver (red dotted lines in Figure 1), and thus, can link lookups to a specific origin IP address. Such an adversary could be present in the users' local network, near the resolver, or anywhere along the path (e.g., an ISP or compromised network router). As noted in the RFC, this adversary may be "in a different path than the communication between the initiator and the recipient" [6].

This adversary model also includes adversaries that only see DoH traffic, e.g., adversaries located in an AS that lies between the user and the resolver but not between the user and the destination host. Measurements performed from our university network show that this is the case in a significant number of cases. Furthermore, we note that BGP hijacking attacks, which are becoming increasingly frequent [50], can be used to selectively intercept paths to DoH resolvers. In such cases, the adversary can only rely on DNS fingerprinting to learn which webpages are visited by a concrete user for monitoring, or censorship [35, 2]. If the adversary is actually in the path to the users' destination, she could perform traditional website fingerprinting. However, we show that web page fingerprinting on only DoH traffic achieves the same accuracy while requiring (on average) 124 times less data than attacking HTTPS traces that also include web traffic. This is critical to guarantee the scalability of attacks to a large number of targets [28].

We assume that the adversary has access to *encrypted DNS traffic traces* that are generated when the user visits a website via HTTP/S using DoH to resolve the IPs of the resources. A encrypted DNS trace, which we also call DoH trace, comprises the resolution of the visited website's first-party domain, and the subsequent resolutions for the resources contained in the website, e.g., images and scripts. For instance, for visiting Reddit, after resolving `www.reddit.com`, the client would resolve domains such as `cdn.taboola.com`, `doubleclick.net` and `thumbs.redditmedia.com`, among others.

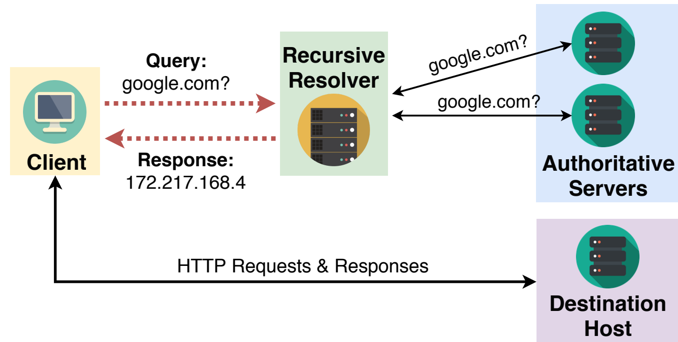

Figure 1: DNS resolution: To visit `www.google.com`, a user queries the recursive resolver for its IP. If the record is not cached, the recursive resolver queries an authoritative resolver and forwards the response to the client. The client uses the IP in the response to connect to the server via HTTP. We consider an adversary placed between the client and the resolver (i.e., observes the red dotted lines).

We consider two adversarial goals. First, *monitoring* the browsing behavior of users, which we study in Section V; and, second, *censoring* the web pages that users visit, which we address in Section VII. These two goals differ in their data collection needs. Monitoring adversaries can collect full traces to make their inferences as accurate as possible, as they do not take any action based on their observations. In contrast, censorship adversaries need to find out which domain is being requested as fast as possible so as to interrupt the communication. Thus, they act on partial traces.

IV. Data Collection

To collect data we set up Ubuntu 16.04 virtual machines with DoH clients that send DNS queries to a public DoH resolver. We use Selenium[3] (version 3.14.1) to automatically visit a webpage from our list, triggering DNS lookups for its resources. We restart the browser in between webpage visits to ensure that the cache and profile do not affect collection. We capture network traffic between the DoH client and the resolver using *tcpdump*, and we filter the traffic by destination port and IP to obtain the final DoH traffic trace.

We collect traces for the top, middle, and bottom 500 webpages in Alexa's top million websites list on 26 March 2018, 1,500 webpages in total. We note that even though 1,500 is still a small world compared to the size of the Web, the largest world considered in evaluations similar to ours for website fingerprinting on web traffic over Tor is of 800 websites [51, 30, 52, 18, 53, 19].

We visit each webpage in a round-robin fashion, obtaining up to 200 samples for every webpage. For our open world analysis, we collect traces of an additional 5,000 webpages from the top domains of the Alexa list. We collected data during two periods, from 26 August 2018 to 9 November 2018, and from 20 April 2019 to 14 May 2019. Data from these two periods is never mixed in the analysis.

We consider different scenarios varying DoH client and resolver, end user location and platform, and the use of DNS

[3]https://www.seleniumhq.org/

Table I: Overview of datasets.

Name	Identifier	# webpages	# samples
Desktop (Location 1)	LOC1	1,500	200
Desktop (Location 2)	LOC2	1,500	60
Desktop (Location 3)	LOC3	1,500	60
Raspberry Pi	RPI	700	60
Firefox with Google resolver	GOOGLE	700	60
Firefox with Cloudflare resolver	CLOUD	700	60
Firefox with Cloudflare client	CL-FF	700	60
Open World	OW	5,000	3
DoH and web traffic	WEB	700	60
DNS over Tor	TOR	700	60
Cloudflare's EDNS0 padding implementation	EDNS0-128	700	60
Recommended EDNS0 padding	EDNS0-468	700	60
EDNS0 padding with ad-blocker	EDNS0-128-adblock	700	60
DoT with Stubby client	DOT	700	60

traffic analysis defenses. Table I provides an overview of the collected datasets. In order to better understand the vulnerability of DNS encryption to traffic analysis, we designed heterogenous experiments that look into different aspects of the problem, resulting in multiple datasets of varied sizes and collected under different conditions – in many cases, several datasets for each experiment type. We also collected a dataset (using the Stubby client and CLoudflare's resolver) to study the effectiveness of defenses on DoT traffic as compared to DoH. Since Cloudflare had already implemented padding of responses for DoT traffic at the time of data collection, we were not able to collect a dataset of DoT without padding. In the following sections, we use the Identifier provided in the second column to refer to each of the datasets. Note that unless specified otherwise, we use Cloudflare's DoH client.

Data curation. We curate the datasets to ensure that our results are not biased. First, we aim at removing the effect of spurious errors in collection. We define those as changes in the websites for reasons other than those variations due to their organic evolution that do not represent the expected behavior of the page. For instance, pages that go down during the collection period.

Second, we try to eliminate website behaviors that are bound to generate classification errors unrelated to the characteristics of DNS traffic. These occur when different domains generating the same exact DNS traces, e.g., when webpages redirect to other pages or to the same resource, or when web servers return the same errors (e.g., 404 not found or 403 forbidden).

We use the Chrome over Selenium crawler to collect the HTTP request/responses, not the DNS queries responses, of all the pages in our list in LOC1. We conduct two checks. First, we look at the HTTP response status of the *top level domain*, i.e., the URL that is being requested by the client. We identify the webpages that do not have an HTTP OK status. These could be caused by a number of factors, such as pages not found (404), anti-bot mechanisms, forbidden responses due to geoblocking [54] (403), internal server errors (500), and so on. We mark these domains as conflicting. Second, we confirm that the top level domain is present in the list of requests and responses. This ensures that the page the client is requesting is not redirecting the browser to other URLs. This check triggers some false alarms. For example, a webpage might redirect to a country-specific version (`indeed.com` redirecting to `indeed.fr`, results in `indeed.com` not being present in the list of requests); or in domain redirections (`amazonaws.com` redirecting to `aws.amazon.com`). We do not consider these cases as anomalies. Other cases are full redirections. Examples are malware that redirect browser requests to `google.com`, webpages that redirect to GDPR country restriction notices, or webpages that redirect to domains that specify that the site is closed. We consider these cases as invalid webpages and add them to our list of conflicting domains.

We repeat these checks multiple times over our collection period. We find that 70 webpages that had invalid statuses at some point during our crawl, and 16 that showed some fluctuation in their status (from conflicting to non-conflicting or vice versa). We study the effects of keeping and removing these conflicting webpages in Section V-B.

V. DNS-BASED WEBSITE FINGERPRINTING

Website fingerprinting attacks enable a local eavesdropper to determine which web pages a user is accessing over an encrypted or anonyimized channel. It exploits the fact that the size, timing, and order of TLS packets are a reflection of a website's content. As resources are unique to each webpage, the traces identify the web even if traffic is encrypted. Website fingerprinting has been shown to be effective on HTTPS [27, 26, 55], OpenSSH tunnels [56, 57], encrypted web proxies [58, 59] and VPNs [60], and even on anonymous communications systems such as Tor [61, 30, 51, 52, 16, 17, 18, 19].

The patterns exploited by website fingerprinting are correlated with patterns in DNS traffic: which resources are loaded and their order determines the order of the corresponding DNS queries. Thus, we expect that website fingerprinting can also be done on DNS encrypted flows such as DNS-over-HTTPS (DoH) or DNS-over-TLS (DoT). In this paper, we call *DNS fingerprinting* the use of traffic analysis to identify the web page that generated an encrypted DNS trace, i.e., website fingerprinting on encrypted DNS traffic. In the following, whenever we do not explicitly specify whether the target of website fingerprinting is DNS or HTTPS traffic, we refer to traditional website fingerprinting on web traffic over HTTPS.

A. DNS fingerprinting

As in website fingerprinting, we treat DNS fingerprinting as a supervised learning problem: the adversary first collects a training dataset of encrypted DNS traces for a set of pages. The page (label) corresponding to a DNS trace is known. The adversary extracts features from these traces (e.g., lengths of network packets) and trains a classifier that, given a network trace, identifies the visited page. Under deployment the adversary collects traffic from a victim and feeds it to the classifier to determine which page the user is visiting.

Traffic variability. Environmental conditions introduce variance in the DNS traces sampled for the same website. Thus, the adversary must collect multiple DNS traces in order to obtain a robust representation of the page.

Some of this variability has similar origin to that of web traffic. For instance, changes on the website that result on varying DNS lookups associated with third-party embedded resources (e.g., ads shown to users); the platform where the client runs, the configuration of the DoH client, or the software using the client which may vary the DNS requests (e.g., mobile versions of websites, or browsers' use of pre-fetching); and the effects of content localization and personalization that determine which resources are served to the user.

Additionally, there are some variability factors specific to DNS traffic. For instance, the effect of the local resolver, which depending on the state of the cache may or may not launch requests to the authoritative server; or the DNS-level load-balancing and replica selection (e.g., CDNs) which may provide different IPs for a given domain or resource [62].

Feature engineering. Besides the extra traffic variability compared to web traffic, DNS responses are generally smaller and chatty than web resources [62, 63]. In fact, even when DNS lookups are wrapped in HTTP requests, DNS requests and responses fit in one single TLS record in most cases. These particularities hint that traditional website fingerprinting features, typically based on aggregate metrics of traffic traces such as the total number of packets, total bytes, and their statistics (e.g., average, standard deviation), are inadequate to characterize DoH traffic. We test this hypothesis in the following section, where we show that state-of-the-art web traffic attacks' performance drops in 20% when applied on DoH traces (see Table III).

To address this problem, we introduce a novel set of features, consisting of n-grams of TLS record lengths in a trace. Following the usual convention in website fingerprinting, we represent a traffic trace as a sequence of integers, where the absolute value is the size of the TLS record and the sign indicates direction: positive for packets from the client to the resolver (outgoing), and negative for the packets from the resolver to the client (incoming). An example of this represenation is the trace: $(-64, 88, 33, -33)$. Then, the uni-grams for this trace are $(-64), (88), (33), (-33)$, and the bi-grams are $(-64, 88), (88, 33), (33, -33)$. To create the features, we take tuples of n consecutive TLS record lengths in the DoH traffic traces and count the number of their occurrences in each trace. The intuition behind our choice is that n-grams capture patterns in request-response size pairs, as well as the local order of the packets sequence. To the best of our knowledge, n-grams have never been considered as features in the website fingerprinting literature.

We extend the n-gram representation to traffic bursts. Bursts are sequences of consecutive packets in the same direction (either incoming or outgoing). Bursts correlate with the number and order of resources embedded in the page. Additionally, they are more robust to small changes in order than individual sizes because they aggregate several records in the same direction. We represent n-grams of bursts by adding lengths on packets in a direction inside the tuple. In the previous example, the burst-length sequence of the trace above is $(-64, 121, -33)$ and the burst bi-grams are $(-64, 121), (121, -33)$.

We experimented with uni-, bi- and tri-grams for both features types. We observed a marginal improvement in the classifier on using tri-grams at a substantial cost on the memory requirements of the classifier. We also experimented with the timing of packets. As in website fingerprinting [30], we found that it does not provide reliable prediction power. This is because timing greatly varies depending on the state of the network and thus is not a stable feature to fingerprint web pages. In our experiments, we use the concatenation of uni-grams and bi-grams of both TLS record sizes and bursts as feature set.

Algorithm selection. After experimenting with different supervised classification algorithms, we selected Random Forests (RF) which are known to be very effective for traffic analysis tasks [18, 64].

Random forests (RF) are ensembles of simpler classifiers called decision trees. Decision trees use a tree data structure to represent splits of the data: nodes represent a condition on one of the data features and branches represent decisions based on the evaluation of that condition. In decision trees, feature importance in classification is measured with respect to how well they split samples with respect to the target classes. The more skewed the distribution of samples into classes is, the better the feature discriminates. Thus, a common metric for importance is the Shannon's entropy of this distribution. Decision trees, however, do not generalize well and tend to overfit the training data. RFs mitigate this issue by randomizing the data and features over a large amount of trees, using different subsets of features and data in each tree. The final decision of the RF is an aggregate function on the individual decisions of its trees. In our experiments, we use 100 trees and a majority vote to aggregate them.

Validation. We evaluate the effectiveness of our classifier measuring the *Precision*, *Recall* and *F1-Score*. Considering positives as websites correctly identified, and negatives websites classified to an incorrect label: Precision is the ratio of true positives to the total number of samples that were classified as positive (true positives and false positives); Recall is the ratio of true positives to the total number of positives (true positives and false negatives); and the F1-score is the harmonic mean of Precision and Recall.

We evaluate the effectiveness of DNS fingerprinting in two scenarios typically used in the web traffic analysis literature. A *closed world*, in which the adversary knows the set of all possible webpages that users may visit; and an *open-world*, in which the adversary only has access to a set of *monitored* sites, and the user may visit webpages outside of this set.

We use 10-fold cross-validation, a standard methodology in machine learning, to eliminate biases due to overfitting in our experiments. In cross-validation, the samples of each class are divided in ten disjoint sets. The classifier is then trained on nine of the sets and tested in the remaining one, proving ten samples of the classifier performance on a set of samples on which it has not been trained on. This estimates the performance of the classifier to unseen examples.

B. Evaluating n-grams features

We now evaluate the effectiveness of n-grams features to launch DNS fingerprinting attacks. We also compare these features with traditional website fingerprinting features in both DoH traffic and web traffic over HTTPS.

Evaluation in closed and open worlds. We first evaluate the attack in a closed world using the LOC1 dataset. We try three settings: an adversary that attacks the full dataset of 1,500 websites, an adversary attacking the curated dataset of 1,414 websites after we eliminate spurious errors, and an adversary attacking the full dataset but that considers regional versions of given pages to be equivalent. For example, classifying `google.es` as `google.co.uk`, a common error in our

Table II: Classifier performance for LOC1 dataset (mean and standard deviation for 10-fold cross validation).

Scenario	Precision	Recall	F1-score
Curated traces	0.914 ± 0.002	0.909 ± 0.002	0.908 ± 0.002
Full dataset	0.904 ± 0.003	0.899 ± 0.003	0.898 ± 0.003
Combined labels	0.940 ± 0.003	0.935 ± 0.003	0.934 ± 0.003

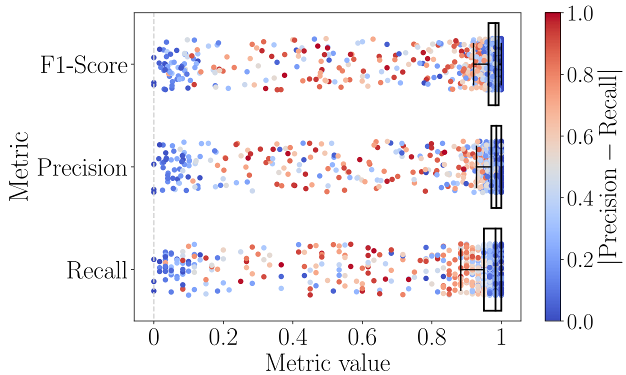

Figure 2: Performance per class in LOC1. Each dot represents a class and its color the absolute difference between Precision and Recall (blue low, red high).

classifier, is considered a true positive. We see in Table II that testing on the clean dataset offers just a 1% performance increase, and that considering regional versions as equivalent results provides an additional 3% increase. As in prior work on website fingerprinting [65], we have used confusion graphs. Confusion graphs are an intuitive representation of classification errors that allows to visualize collisions between pages. See Figures 13 and 16 in the Appendix for the confusion graphs of this evaluation. Given the minimal differences, in the reminder of the experiments we use the full dataset.

In the context of website fingerprinting, Overdorf *et al.* [65] showed that it is likely that the classifier's performance varies significantly between different individual classes. Thus, looking only at average metrics, as in Table II, may give an incomplete and biased view of the classification results. To check if this variance holds on DNS traces we study the performance of the classifier for individual websites. The result is shown in Figure 2. In this scatterplot each dot is a website and its color represents the absolute difference between Precision and Recall: blue indicates 0 difference and red indicates maximum difference (i.e., $|Precision - Recall| = 1$). We see that some websites (red dots on the right of the Precision scatterplot) have high Recall – they are often identified by the classifier, but low Precision – other websites are also identified as the this website. Thus, these websites have good privacy since the false positives provide the users with plausible deniability. For other pages (red dots on the right of the Recall scatterplot), the classifier obtains low Recall – it almost never identifies them, but high Precision – if they are identified, the adversary is absolutely sure her guess is correct. The latter case is very relevant for privacy, and in particular censorship, as it enables the censor to block without fear of collateral damage.

Open world. In the previous experiments, the adversary knew that the webpage visited by the victim was within the training

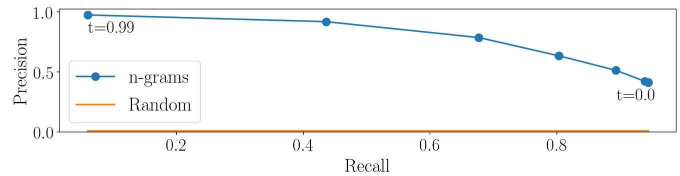

Figure 3: Precision-Recall ROC curve for open world classification, for the monitored class. The threshold, t, is varied from 0.0 to 0.99 in steps of 0.1 (standard deviation less than 1%).

dataset. We now evaluate the adversary's capability to distinguish those webpages from other unseen traffic. Following prior work [64, 66] we consider two sets of webpages, one *monitored* and one *unmonitored*. The adversary's goal is to determine whether a test trace belongs to a page within the monitored set.

We train a classifier with both monitored and unmonitored samples. Since it is not realistic to assume that an adversary can have access to all unmonitored classes, we create unmonitored samples using 5,000 webpages traces formed by a mix of the OW and LOC1 datasets. We divide the classes such that 1% of all classes are in the monitored set and 10% of all classes are used for training. We ensure that the training dataset is balanced, i.e., it contains equal number of monitored and unmonitored samples; and the test set contains an equal number of samples from classes used in training and classes unseen by the classifier. When performing cross validation, in every fold we consider a different combination of the monitored and unmonitored classes for training and testing so that we do not overfit to a particular case.

To decide whether a target trace is monitored or unmonitored, we use a method proposed by Stolerman et al. [67]. We assign the target trace to the monitored class if and only if the classifier predicts this class with probability larger than a threshold t, and to unmonitored otherwise. We show in Figure 3, the average Precision-Recall ROC curve for the monitored class over 10 iterations varying the discrimination threshold, t, from 0 to 0.99 in steps of 0.1. We also show the random classifier, which indicates the probability of selecting the positive class uniformly at random, and acts as a baseline. We see that when $t = 0.8$, the classifier has an F1-score of $\approx$ 0.7. This result suggests that traffic analysis is a true threat to DNS privacy.

Comparison to Web traffic fingerprinting. To understand the gain DNS fingerprinting provides to an adversary, we compare its effectiveness to that of Web traffic fingerprinting. We also evaluate the suitability of n-grams and traditional website fingerprinting features to both fingerprinting problems. We compare it to state-of-the-art attacks that use different features: the k-Fingerprinting attack, by Hayes and Danezis [18], that considers a comprehensive set of features used in the website fingerprinting literature; CUMUL, by Panchenko *et al.* [16] which focuses on packets' lengths and order through cumulative features; and Deep Fingerprinting (DF), an attack based on deep convolutional neural networks [19]. In this comparison, we consider a closed-world of 700 websites (WEB dataset) and use a random forest with the same parameters as classification algorithm. We evaluate the performance of the classifiers on only DoH traffic (DoH-only), only HTTPS traffic corresponding to web content traffic (Web-only), and a mix of

Table III: F1-Score of the n-grams, k-Fingerprinting, CUMUL and DF features for different subsets of traffic: only DoH traffic (DoH-only), only HTTPS traffic corresponding to web traffic (Web-only) and mixed (DoH+Web).

	DoH-only	Web-only	DoH + Web
n-grams	0.87	0.99	0.88
k-Fingerprinting [18]	0.74	0.95	0.79
CUMUL [16]	0.75	0.92	0.77
DF [19][4]	0.51	0.94	0.75

the two in order to verify the claim in the RFC that DoH [9] holds great potential to thwart traffic analysis. Table III shows the results.

First, we find that for DNS traffic, due to its chatty characteristics, n-grams provide more than 10% performance improvement with respect to traditional features. We also see that the most effective attacks are those made on web traffic. This is not surprising, as the variability of resources' sizes in web traffic contains more information than the small DNS packets. What is surprising is that the n-grams features *outperform* the traditional features *also* for website traffic. Finally, as predicted by the standard, if DoH and HTTPS are sent on the same TLS tunnel and cannot be separated, both set of features see a decrease in performance with n-grams offering 10% improvement.

In summary, the best choice for an adversary with access to the isolated HTTPS flow is to analyse that trace with our novel n-grams features. However, if the adversary is in 'a different path than the communication between the initiator and the recipient' [6] where she has access to DNS, or is limited in resources (see below), the DNS encrypted flow provides comparable results.

Adversary's effort. An important aspect to judge the severity of traffic analysis attacks is the effort needed regarding data collection to train the classifier [28]. We study this effort from two perspectives: amount of samples required – which relates to the time needed to prepare the attack, and volume of data – which relates to the storage and processing requiremets for the adversary.

We first look at how many samples are required to train a well-performing classifier. We see in Table IV that there is a small increase between 10 and 20 samples, and that after 20 samples, there are diminishing returns in increasing the number of samples per domain. This indicates that, in terms of number of samples, the collection effort to perform website identification on DNS traces is *much smaller* than that of previous work on web traffic analysis: Website fingerprinting studies in Tor report more than 10% increase between 10 and 20 samples [30] and between 2% and 10% between 100 and 200 samples [53, 19].

We believe the reason why fingerprinting DoH requires fewer samples per domain is DoH's lower intra-class variance with respect to encrypted web traffic. This is because sources of large variance in web traffic, such as the presence of advertisements which change accross visits thereby varying the sizes of the resources, does not show in DNS traffic for which the same ad-network domains are resolved [68].

[4]We evaluate DF's accuracy following the original paper, i.e., using validation and test sets, instead of 10-fold cross-validation.

Table IV: Classifier performance for different number of samples in the LOC1 dataset averaged over 10-fold cross validation (standard deviations less than 1%).

Number of samples	**Precision**	**Recall**	**F1-score**
10	0.873	0.866	0.887
20	0.897	0.904	0.901
40	0.908	0.914	0.909
100	0.912	0.916	0.913

In terms of volume of data required to launch the attacks, targeting DoH flows also brings great advantage. In the WEB dataset, we observe that web traces have a length of 1.842 MB ± 2.620 MB, while their corresponding DoH counterpart only require 0.012 MB ± 0.012 MB. While this may not seem a significant difference, when we look at the whole dataset instead of individual traces, the HTTPS traces require 73GB while the DoH-only dataset fits in less than 1GB (0.6GB). This is because DNS responses are mostly small, while web traffic request and responses might be very large and diverse (*e.g.*, different type of resources, or different encodings).

In our experiments, to balance data collection effort and performance, we collected 60 samples per domain for all our datasets. For the unmonitored websites in the open world, for which we do not need good models as we do not identify them, we collected three samples per domain.

C. DNS Fingerprinting Robustness

In practice, the capability of the adversary to distinguish websites is very dependent on differences between the environmental conditions at attack time and the setup for training data collection [69]. We present experiments exploring three environmental dimensions: time, space, and infrastructure.

1) Robustness over time: DNS traces vary due to the dynamism of webpage content and variations in DNS responses (e.g., service IP changes because of load-balancing). To understand the impact of this variation on the classifier, we consider collect data LOC1 for 10 weeks from the end of September to the beginning of November 2018. We divide this period into five intervals, each containing two consecutive weeks, and report in Table V the F1-score of the classifier when we train the classifier on data from a single interval and use the other intervals as test data (0 weeks old denotes data collected in November). In most cases, the F1-score does not significantly decrease within a period of 4 weeks. Longer periods result in a significant drops – more than 10% drop in F1-score when the training and testing are separated by 8 weeks.

This indicates that, to obtain best performance, an adversary with the resources of a university research group would need to collect data at least once a month. However, it is unlikely that DNS traces change drastically. To account for gradual changes, the adversary can perform continuous collection and mix data across weeks. In our dataset, if we combine two- and three-week-old samples for training; we observe a very

Table V: F1-score when training on the interval indicated by the row and testing on the interval in the column (standard deviations less than 1%). We use 20 samples per webpage (the maximum number of samples collected in all intervals).

F1-score	0 weeks old	2 weeks old	4 weeks old	6 weeks old	8 weeks old
0 weeks old	0.880	0.827	0.816	0.795	0.745
2 weeks old	0.886	0.921	0.903	0.869	0.805
4 weeks old	0.868	0.898	0.910	0.882	0.817
6 weeks old	0.775	0.796	0.815	0.876	0.844
8 weeks old	0.770	0.784	0.801	0.893	0.906

small decrease in performance. Thus, a continuous collection strategy can suffice to maintain the adversary's performance without requiring periodic heavy collection efforts.

2) Robustness across locations: DNS traces may vary across locations due to several reasons. First, DNS lookups vary when websites adapt their content to specific geographic regions. Second, popular resources cached by resolvers vary across regions. Finally, resolvers and CDNs use geo-location methods for load-balancing requests, *e.g.*, using anycast and EDNS [70, 71].

We collect data in three locations, two countries in Europe (LOC1 and LOC2) and a third in Asia (LOC3). Table VI (leftmost) shows the classifier performance when crossing these datasets for training and testing. When trained and tested on the same location unsurprisingly the classifier yields results similar to the ones obtained in the base experiment. When we train and test on different locations, the F1-score decreases between a 16% and a 27%, the greatest drop happening for the farthest location, LOC3, in Asia.

Interestingly, even though LOC2 yield similar F1-Scores when cross-classified with LOC1 and LOC3, the similarity does not hold when looking at Precision and Recall individually. For example, training on LOC2 and testing on LOC1 results on around 77% Precision and Recall, but training on LOC1 and testing on LOC2 yields 84% Precision and 65% Recall. Aiming at understanding the reasons behind this asymmetry, we build a classifier trained to separate websites that obtain high recall (top 25% quartile) and low recall (bottom 25% quartile) when training with LOC1 and LOC3 and testing in LOC2. A feature importance analysis on this classifier that LOC2's low-recall top features have a significantly lower importance in LOC1 and LOC2. Furthermore, we observe that the intersection between LOC1 and LOC3's relevant feature sets is slightly larger than their respective intersections with LOC2. While it is clear that the asymmetry is caused by the configuration of the network in LOC2, its exact cause remains an open question.

3) Robustness across infrastructure: In this section, we study how the DoH resolver and client, and the user platform affect influence the attack's performance.

Influence of DoH Resolver. We study two commercial DoH resolvers, Cloudflare's and Google's. Contrary to Cloudflare, Google does not provide a stand-alone DoH client. To keep the comparison fair, we instrument a new collection setting using Firefox in its *trusted recursive resolver* configuration with both DoH resolvers.

Table VI (center-left) shows the result of the comparison. As expected, training and testing on the same resolver yields the

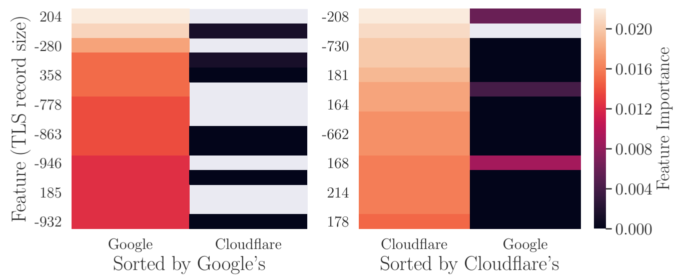

Figure 4: Top 15 most important features in Google's and Cloudflare's datasets. On the left, features are sorted by the results on Google's dataset and, on the right, by Cloudflare's.

best results. In particular, we note that even though Google hosts other services behind its resolver's IP and thus DoH traffic may be mixed with the visited website's traffic (e.g., if a web embeds Google third-party) the classifier performs equally for both resolvers.

As in the location setting, we observe an asymmetric decrease in one of the directions: training on GOOGLE dataset and attacking CLOUD results in 13% F1-score, while attacking GOOGLE with a classifier trained on CLOUD yields similar results as training on GOOGLE itself.

To investigate this asymmetry we rank the features according their importance for the classifiers. For simplicity, we only report the result on length unigrams, but we verified that our conclusions hold when considering all features together. Figure 4 shows the top-15 most important features for a classifier trained on Google's resolver (left) and Cloudflare's (right). The rightmost diagram of each column shows the importance of these features on the other classifier. Red tones indicate high importance, and dark colors represent irrelevant features. Grey indicates that the feature is not present.

We see that the most important features in Google are either not important or missing in Cloudflare (the right column in left-side heatmap is almost gray). As the missing features are very important, they induce erroneous splits early in the trees, and for a larger fraction of the data, causing the performance drop. However, only one top feature in the classifier trained on Cloudflare is missing in Google, and the others are also important (right column in right-side heatmap). Google does miss important features in Cloudflare, but they are of little importance and their effect on performance is negligible.

Influence of end user's platform. We collect traces for the 700 top Alexa webpages on a Raspberry Pi (RPI dataset) and an Ubuntu desktop (DESKTOP dataset), both from LOC1. We see in Table VI (center-right) that, as expected, the classifier has good performance when the training and testing data come from the same platform. However, it drops to almost zero when crossing the datasets.

When taking a closer look at the TLS record sizes from both platforms we found that TLS records in the DESKTOP dataset are on average 7.8 bytes longer than those in RPI (see Figure 12 in Appendix C). We repeated the cross classification after adding 8 bytes to all RPI TLS record sizes. Even though the classifiers do not reach the base experiment's performance, we see a significant improvement in cross-classification F1-score

Table VI: Performance variation changes in location and infrastructure (F1-score, standard deviations less than 2%).

Location	LOC1	LOC2	LOC3
LOC1	0.906	0.712	0.663
LOC2	0.748	0.908	0.646
LOC3	0.680	0.626	0.917

Resolver	GOOGLE	CLOUD
GOOGLE	0.880	0.129
CLOUD	0.862	0.885

Platform	DESKTOP	RPI
DESKTOP	0.8802	0.0003
RPI	0.0002	0.8940

Client	CLOUD	CL-FF	LOC2
CLOUD	0.885	0.349	0.000
CL-FF	0.109	0.892	0.069
LOC2	0.001	0.062	0.908

to 0.614 when training on DESKTOP and testing on RPI, and 0.535 when training on RPI and testing on DESKTOP.

Influence of DNS client. Finally, we consider different client setups: Firefox's trusted recursive resolver or TRR (CLOUD), Cloudlflare's DoH client with Firefox (CL-FF) and Cloudflare's DoH client with Chrome (LOC2). We collected these datasets in location LOC2 using Cloudflare's resolver.

Table VI (rightmost) shows that the classifier performs as expected when trained and tested on the same client setup. When the setup changes, the performance of the classifier drops dramatically, reaching zero when we use different browsers. We hypothesize that the decrease between CL-FF and LOC2 is due to differences in the implementation of the Firefox's built-in and Cloudflare's standalone DoH clients.

Regarding the difference when changing browser, we found that Firefox' traces are on average 4 times longer than Chrome's. We looked into the unencrypted traffic to understand this difference. We used a proxy to man-in-the-middle the DoH connection between the client and the resolver[5], obtaining the OpenSSL TLS session keys with Lekensteyn's scripts[6]. We use this proxy to decrypt DoH captures for Firefox configured to use Cloudflare's resolver, but we could not do the same for Google. Instead, we man-in-the-middle a curl-doh client[7], which also has traces substantially shorter than Firefox. We find that Firefox, besides resolving domains related to the URL we visit, also issues resolutions related to OSCP servers, captive portal detection, user's profile/account, web extensions, and other Mozilla servers. As a consequence, traces in CL-FF and CLOUD datasets are substantially larger and contain contain different TLS record sizes than any of our other datasets. We conjecture that Chrome performs similar requests, but since traces are shorter we believe the amount of checks seems to be smaller than Firefox's.

4) Robustness Analysis Takeaways: Our robustness study shows that to obtain best results across different configurations the adversary needs i) to train a classifier for each targeted setting, and ii) to be able to identify her victim's configuration. Kotzias et al. demonstrated that identifying client or resolver is possible, for instance examining the IP (if the IP is dedicated to the resolver), or fields in the ClientHello of the TLS connection (such as the the Server Name Indication (SNI), cipher suites ordering, etc.) [72]. Even if in the future these features are not available, we found that the characteristics of the traffic itself are enough to identify a resolver. We built classifiers to distinguish resolver and client based on the TLS record length. We can identify resolvers with 95% accuracy, and we get no errors (100% accuracy) when identifying the client.

[5]https://github.com/facebookexperimental/doh-proxy
[6]https://git.lekensteyn.nl/peter/wireshark-notes
[7]https://github.com/curl/doh

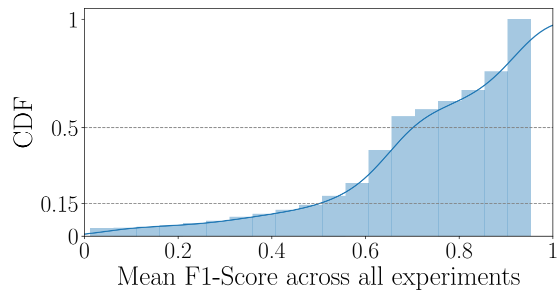

Figure 5: Cumulative Distribution Function (CDF) of the per-class mean F1-Score.

Regarding users' platform, we see little difference between desktops, laptops, and servers in Amazon Web Services. When the devices are as different as a desktop and a constrained device such a Raspberry Pi, the classifier's accuracy drops.

Finally, our longitudinal analysis reveals that, keeping up with the changes in DNS traces can be done at low cost by continuously collecting samples and incorporating them to the training set.

Survivors and Easy Preys. Finally, we study whether there are websites that are particularly good or bad at evading fingerprinting under all the configurations. We compute the mean F1-Score across all configurations as an aggregate measure of the attack's overall performance. We plot the CDF of the distribution of mean F1-scores over the websites in Figure 5. This distribution is heavily skewed: there are up to 15% of websites that had an F1-Score equal or lower than 0.5 and more than 50% of the websites have a mean F1-Score equal or lower than 0.7.

We then rank sites by lowest mean F1-Score and lowest standard deviation. At the top of this ranking, there are sites that *survived* the attack in all configurations. Among these survivors, we found Google, as well as sites giving errors, that get consistently misclassified as each other. We could not find any pattern in the website structure or the resource loads that explains why other sites with low F1 survive. We leave a more in-depth analysis of the survivors for future work. At the bottom of the ranking, we find sites with long domain names, with few resources and that upon visual inspection seem to present low variability. In Tables X, XI, XII in the Appendix, we list the top-10 sites in the tails of the distribution.

VI. DNS Defenses against Fingerprinting

In this section, we evaluate the effectiveness of existing techniques to protect encrypted DNS against traffic analysis. Table VII summarizes the results. We consider the following defenses:

EDNS(0) Padding. EDNS (Extension mechanisms for DNS) is a specification to increase the functionality of the DNS protocol [73]. It specifies how to add *padding* [10], both on DNS clients and resolvers, to prevent size-correlation attacks on encrypted DNS. The recommended padding policy is for clients to pad DNS requests to the nearest multiple of 128 bytes, and for resolvers to pad DNS responses to the nearest multiple of 468 bytes [74].

Cloudflare's DoH client provides functionality to set EDNS(0) padding to DNS queries, leaving the specifics of the padding policy to the user. We modify the client source code to follow the padding strategy above. Google's specification also mentions EDNS padding. However, we could not find any option to activate this feature, thus we cannot analyze it.

In addition to the EDNS(0) padding, we wanted to see whether simple user-side measures that alter the pattern of requests, such as the use of an ad-blocker, could be effective countermeasures. We conduct an experiment where we use Selenium with a Chrome instance with the AdBlock Plus extension installed. We do this with DoH requests and responses padded to multiples of 128 bytes.

Upon responsible disclosure of this work, Cloudflare's added padding to the responses of their DoH resolver. However, when analyzing the collected data we discover that they do *not* follow the recommended policy. Instead of padding to multiples of 468 bytes, Cloudflare's resolver pads responses to multiples of 128 bytes, as recommended for DoH clients. In order to also evaluate the recommended policy, we set up an HTTPS proxy (*mitmproxy*) between the DoH client and the Cloudflare resolver. The proxy intercepts responses from Cloudflare's DoH resolver, strips the existing padding, and pads responses to the nearest multiple of 468 bytes.

As we show below, none of these padding strategies completely stops traffic analysis. To understand the limits of protection of padding, we simulate a setting in which padding is perfect, *i.e.*, *all* records have the same length and the classifier cannot exploit the TLS record size information. To simulate this setting, we artificially set the length of all packets in the dataset to 825, the maximum size observed in the dataset.

DNS over Tor. Tor is an anonymous communication network. To protect the privacy of its users, Cloudflare set up a DNS resolver that can be accessed using Tor. This enables users to not reveal their IP to the resolver when doing lookups.

To protect users' privacy, Tor re-routes packets through so-called onion routers to avoid communication tracing based on IP addresses; and it packages content into constant-size cells to prevent size-based analysis. These countermeasures have so far not been effective to protect web traffic [16, 17, 18, 19]. We study whether they can protect DNS traffic.

Results. Our first observation is that EDNS0 padding is not as effective as expected. Adding more padding, as recommended in the specification, does provide better protection, but still yields an F1-score of 0.45, six orders of magnitude greater than random guessing. Interestingly, removing ads helps as much as increasing the padding, as shown by the EDNS0-128-adblock experiment. As shown below, Perfect padding would actually deter the attack, but at a high communication cost.

As opposed to web traffic, where website fingerprinting obtains remarkable performance [61, 18, 19], Tor is very effective in hiding the websites originating a DNS trace. The reason is that DNS lookups and responses are fairly small. They fit in one, at most two, Tor cells which in turn materialize in few observed TLS record sizes. As a result, it is hard to find features unique to a page. Also, DNS traces are shorter than normal web traffic, and present less variance. Thus, length-related features, which have been proven to be very important in website fingerprinting, only provide a weak 1% performance improvement.

Table VII: Classification results for countermeasures.

Method	Precision	Recall	F1-score
EDNS0-128	0.710 ± 0.005	0.700 ± 0.004	0.691 ± 0.004
EDNS0-128-adblock	0.341 ± 0.013	0.352 ± 0.011	0.325 ± 0.011
EDNS0-468	0.452 ± 0.007	0.448 ± 0.006	0.430 ± 0.007
Perfect padding	0.070 ± 0.003	0.080 ± 0.002	0.066 ± 0.002
DNS over Tor	0.035 ± 0.004	0.037 ± 0.003	0.033 ± 0.003
DNS over TLS	0.419 ± 0.008	0.421 ± 0.007	0.395 ± 0.007

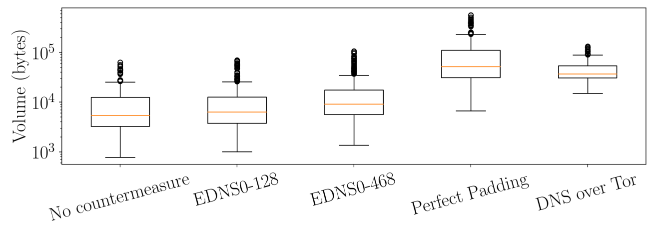

Figure 6: Total volume of traffic with and without countermeasures.

Even though Perfect padding and DNS over Tor offer similar performance, when we look closely at the misclassified webpages, we see that their behavior is quite different. For Tor, we observe misclassifications cluster around six different groups, and in Perfect padding they cluster around 12 (different) groups (Figures 14 and 15 in the Appendix). For both cases, we tested that it is possible to build a classifier that identifies the cluster a website belongs to with reasonable accuracy. This means that despite the large gain in protection with respect to EDNS(0), the effective anonymity set for a webpage is much smaller than the total number of webpages in the dataset.

Finally, we evaluate defenses' communication overhead. For each countermeasure, we collect 10 samples of 50 webpages, with and without countermeasures, and measure the difference in total volume of data exchanged between client and resolver. We see in Figure 6 that, as expected, EDNS0 padding (both 128 and 468) incur the least overhead, but they also offer the least protection. DNS over Tor, in turn, offers lower overhead than Perfect padding.

Comparison with DNS over TLS (DoT). Finally, we compare the protection provided by DNS over HTTPS and over TLS, using the DOT dataset. As in DoH, Cloudflare's DoT resolver implements EDNS0 padding of DNS responses to a multiple of 128 bytes. The cloudflared DoT client, however, does not support DoT traffic. Thus, we use the Stubby client to query Cloudflare's DoT padded to a multiple of 128 bytes.

Our results (shown in the last row of Table VII) indicate that DoT offers much better protection than DoH – about 0.3 reduction in F1-score. The reason is that DoT traffic presents much less variance than DoH, and thus are less unique. We show this in Figure 7, where we represent a histogram of sizes of the sent and received TLS records for both DoH and DoT for 100 webpages. As both flows are encrypted, we cannot investigate the underlying reasons for this difference.

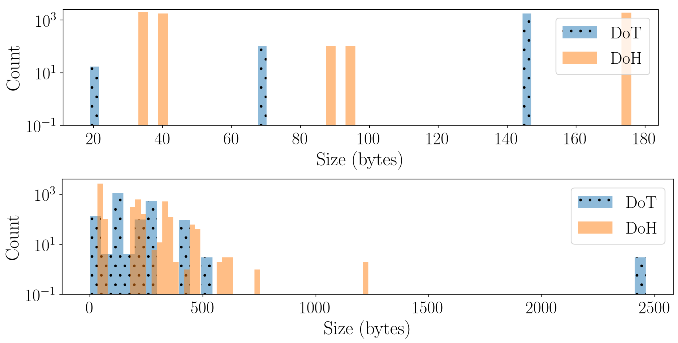

Figure 7: Histogram of sent (top) and received (bottom) TLS record sizes for DoT and DoH.

VII. Censorship on Encrypted DNS

DNS-based blocking is a wide-spread method of censoring access to web content. Censors inspect DNS lookups and, when they detect a blacklisted domain, they either reset the connection or inject their own DNS response [75]. The use of encryption in DoH precludes content-based lookup identification. The only option left to the censor is to block the resolver's IP. While this would be very effective, it is a very aggressive strategy, as it would prevent users from browsing any web. Furthermore, some DoH resolvers, such as Google's, do not necessarily have a dedicated IP. Thus, blocking their IP may also affect other services.

An alternative for the censor is to use traffic analysis to identify the domain being looked up. In this section, we study whether such an approach is feasible. We note that to block access to a site, a censor not only needs to identify the lookup from the encrypted traffic, but also needs to do this as soon as possible to prevent the user from downloading any content.

A. Uniqueness of DoH traces

In order for the censor to be able to uniquely identify domains given DoH traffic, the DoH traces need to be unique. In particular, to enable early blocking, *the first packets* of the trace need to be unique.

To study the uniqueness of DoH traffic, let us model the set of webpages in the world as a random variable W with sample space Ω_W; and the set of possible network traces generated by those websites as a random variable S with sample space Ω_S. A website's trace w is a sequence of non-zero integers: $(s_i)_{i=1}^n, s_i \in \mathbb{Z} \setminus \{0\}, n \in \mathbb{N}$, where s_i represents the size (in bytes) of the i-th TLS record in the traffic trace. Recall that its sign represents the direction – negative for incoming (DNS to client) and positive otherwise. We denote partial traces, *i.e.*, only the l first TLS records, as S_l.

We measure *uniqueness* of partial traces using the conditional entropy $H(W \mid S_l)$, defined as:

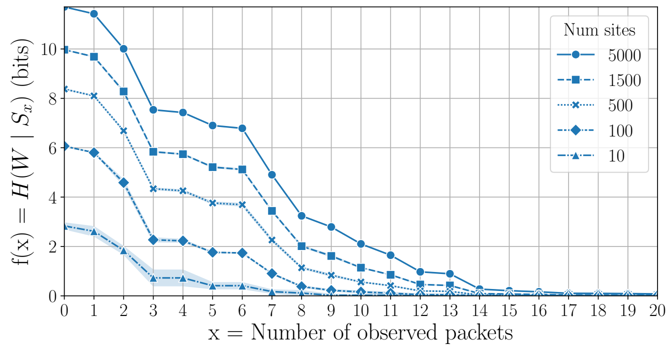

Figure 8: Conditional entropy $H(W \mid S_l)$ given partial observations of DoH traces for different world sizes ($|\Omega_W| = \{10, 100, 500, 1500, 5000\}$). Each data point is averaged over 3 samples.

$$H(W \mid S_l) = \sum_{\forall o \in \Omega_{S_l}} \Pr[S_l = o] H(W \mid S_l = o) \,.$$

Here, $H(W \mid S_l = o)$ is the Shannon entropy of the probability distribution $\Pr[W \mid S_l = o]$. This probability describes the likelihood that the adversary guesses websites in W given the observation o. The conditional entropy $H(W \mid S_l)$ measures how distinguishable websites in Ω_W are when the adversary has only observed l TLS records. When this entropy is zero, sites are perfectly distinct. For instance, if the first packet of every DoH trace had a different size, then the entropy $H(W \mid S_1)$ would be 0.

We compute the conditional entropy for different world sizes $|\Omega_W|$ and partial lengths l, using traces from the OW dataset. We show the result in Figure 8. Every point is an average over 3 samples of $|\Omega_W|$ webs. These webs are selected uniformly at random from the dataset. The shades represent the standard deviation across the 3 samples.

Unsurprisingly, as the adversary observes more packets, the traces become more distinguishable and the entropy decreases. For all cases, we observe a drop of up to 4 bits within the first four packets, and a drop below 0.1 bits after 20 packets. As the world size increases, the likelihood of having two or more websites with identical traces increases, and thus we observe a slower decay in entropy. We also observe that, as the world size increases the standard deviation becomes negligible. This is because the OW dataset contains 5,000 websites. Thus, as the size of the world increases, the samples of Ω_W contain more common websites.

Even when considering 5,000 pages, the conditional entropy drops below 1 bit after 15 packets. This means that after 15 packets have been observed, there is one domain whose probability of having generated the trace is larger than 0.5. In our dataset 15 packets is, on average, just 15% of the whole trace. Therefore, on average, the adversary only needs to observe the initial 15% of a DoH connection to determine a domain with more confidence than taking a random guess between two domains. We observe that as the number of pages increase, the curves are closer to each other indicating that convergence on uniqueness after the 15-th packet is likely to hold in larger datasets.

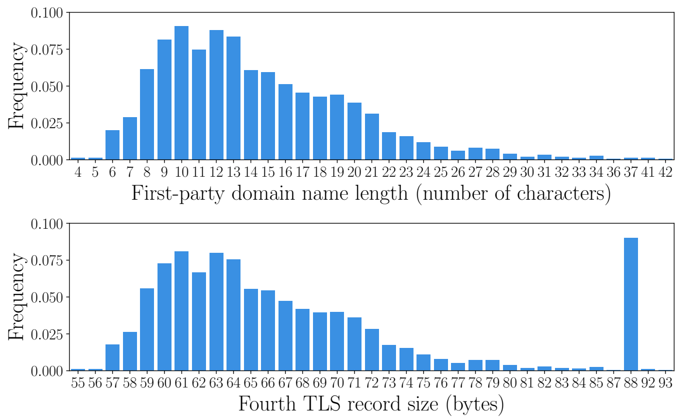

Figure 9: Histograms for domain name length (top) and fourth TLS record length (bottom) in the LOC1 dataset, for 10 samples (normalized over the total sum of counts).

The importance of the fourth packet. We hypothesized that the large drop by the fourth packet might be caused by one of these records containing the DNS lookup. As these traces do not contain padding, the domain length would be directly observable in the trace. We verify this hypothesis by comparing the frequency of appearance of the domain's and outgoing fourth record's length (incoming records cannot contain a lookup). We discard TLS record sizes corresponding to HTTP2 control messages, e.g., the size "33" which corresponds to HTTP2 acknowledgements and outliers that occurred 5% or less times. However, We kept size "88" even though it appears too often, as it could be caused by queries containing 37-characters-long domain names.

We represent the frequency distributions in Figure 9. We see that the histogram of the sizes of the fourth TLS record in our dataset is almost identical to the histogram of domain name lengths, being the constant difference of 51 bytes between the two histograms the size of the HTTPS header. This confirms that the fourth packet often contains the first-party DoH query. In some traces the DoH query is sent earlier, explaining the entropy decrease starting after the second packet.

B. *Censor DNS-blocking strategy*

We study the collateral damage, *i.e.*, how many sites are affected when a censor blocks one site after a traffic-analysis based inference. We assume that upon decision, the censor uses standard techniques to block the connection [76].

High-confidence blocking. To minimize the likelihood of collateral damage, the adversary could wait to see enough packets for the conditional entropy to be lower than one bit. This requires waiting for, on average, 15% of the TLS records in the DoH connection before blocking. As those packets include the resolution to the first-party domain, the client can download the content served from this domain. Yet, the censor can still disrupt access to subsequent queried domains (subdomains and third-parties). This is a strategy already used in the wild as a stealthy form of censorship [76].

To avoid this strategy, the clients could create one connection per domain lookup, thereby mixing connections belonging to the censored page and others. At the cost of generating more traffic for users and resolvers, this would force the censor to drop all DoH connections originating from a user's IP or throttle their DoH traffic, causing more collateral damage.

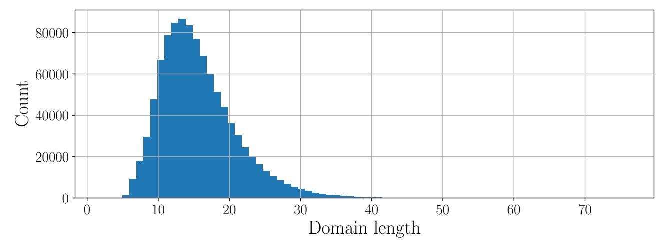

Figure 10: Distribution of domain length in the Alexa top 1M.

Table VIII: Number of domains in each censorship test list and their presence in the Alexa top 1M.

	Turkmenistan	**Iran**	**S. Arabia**	**Vietnam**	**China**
Censored domains	344	877	284	274	191
Not in Alexa-1M	246	600	219	218	94

Block on first DoH query. A more aggressive strategy is to drop the DoH connection before the first DoH response arrives. While this guarantees that the client cannot access any content, not even `index.html`, it also results in all domains with same name length being censored.

In order to understand the collateral damage incurred by domain length based blocking relying on the fourth packet, we compare the distribution of domain name lengths in the Alexa top 1M ranking (see Fig. 10) with the distribution of domain names likely to be censored in different countries. For the former we take the global ranking as per-country rankings only provide 500 top websites which are not enough for the purpose of our experiments and, for some countries, the lists are not even available. We take the test lists provided by Citizen Labs [77]. These lists contain domains that regional experts identify as likely to be censored. While appearance in these lists does not guarantee that the domains are actually censored, we believe that they capture potential censor behavior.

We take five censorship-prone countries as examples: Turkmenistan, Iran, Saudi Arabia, Vietnam, and China. Since the Alexa list contains only domains, we extract the domains from the URLs appearing in Citizen Labs' test lists. Table VIII shows the total number of domains in each list and the number of those domains that are not present in the Alexa-1M list. We observe that at most 51% (China) of the domains appear in the ranking. For the other countries the ratio is between 21% and 32%. This indicates that the potentially censored domains themselves are mostly not popular.

While in themselves these domains are not popular, as the censor can only use their length to identify related DoH lookups, there are two effects. First, there will be some collateral damage: allowed websites with the same domain length will also be blocked. Second, there will be a gain for the censor: censored websites with the same domain length are blocked at the same time.

First, we study the trade-off between collateral damage and censor gain. The minimum collateral damage is attained when the censor blocks domains of length 5 (such as `ft.com`), resulting on 1,318 affected websites. The maximum dam-

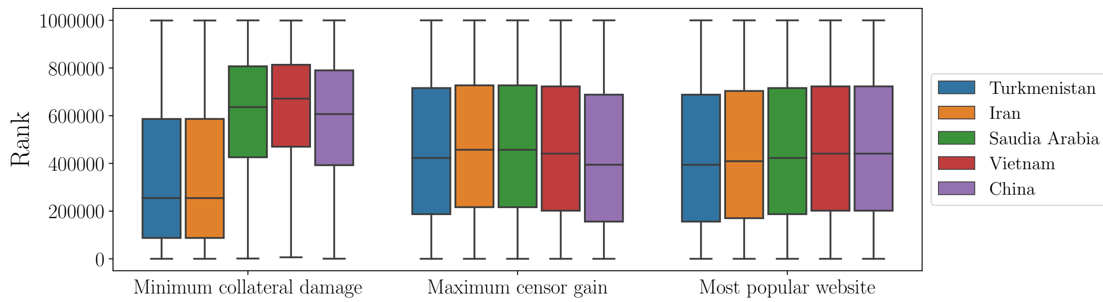

Figure 11: Collateral damage in terms of ranking for three blocking strategies: minimum collateral damage set size, maximum censor gain, most popular website.

age happens when censoring domains of size 10 (such as `google.com`) which affects other 66,923 websites, domains of size 11 (such as `youtube.com`) which affects other 78,603 websites, domains of size 12 (such as `facebook.com`) which affects other 84,471 websites, domains of size 13 (such as `wikipedia.org`) which affects other 86,597 websites, and domains of size 14 (such as `torproject.org`, which affects other 83,493 websites. This means that, in the worst case, collateral damage is at most 8.6% of the top 1M list.

To understand the censor gain, we consider Iran as an example. The maximum gain is 97, for domain length of 13, *i.e.*, when the censor blocks domains of length 13, it blocks 97 domains in the list. At the same time, it results in large collateral damage (86,557 websites). The minimum gain is obtained for domain length 5, which only blocks one website, but causes small collateral damage (1,317 domains). If Iran blocks the most popular domain in the list (google.com), this results in a collateral damage of 66,887 websites.

The volume of affected websites is of course representative of damage, but it is not the only factor to take into account. Blocking many non-popular domains that are in the bottom of the Alexa list is not the same as blocking a few in the top-100. We show in Figure 11 boxplots representing the distribution of Alexa ranks for three blocking strategies: minimum collateral damage, maximum censor gain, and blocking the most popular web. For each country, these strategies require blocking different domain lengths. Thus, in terms of volume, the damage is different but in the order of the volumes reported above. We observe that the minimum collateral damage strategy results in different impact depending on the country. Although for all of them the volume is much lower than for the other strategies, in Turkmenistan and Iran the median rank of the blocked websites is much lower than those in the other strategies, indicating that this strategy may have potentially higher impact than blocking more popular or high-gain domains. On the positive side, minimum collateral damage in Saudi Arabia, Vietnam, and China, mostly affects high-ranking websites. Thus, this strategy may be quite affordable in this country. The other two strategies, maximum gain and blocking the most popular website, result on a larger number of websites blocked, but their median ranking is high (above 500,000) and thus can also be considered affordable.

In conclusion, our study shows that while block on first DoH query is an aggressive strategy, it can be affordable in terms of collateral damage. More research is needed to fine-tune our analysis, e.g., with access to large per-country rankings, or exact lists of blacklisted domains.

VIII. Looking Ahead

We have shown that, although it is a great step for privacy, encrypting DNS does not completely prevent monitoring or censorship. Current padding strategies have great potential to prevent censorship, but our analysis shows that they fall short when it comes to stopping resourceful adversaries from monitoring users' activity on the web.

Our countermeasures analysis hints that the path towards full protection is to eliminate size information. In fact, the repacketization in constant-size cells offered by Tor provides the best practical protection. Tor, however, induces a large overhead both in the bandwidth required to support onion encryption, as well as in download time due to the rerouting of packets through the Tor network.

We believe that the most promising approach to protect DoH is to have clients mimicking the repacketization strategies of Tor, without replicating the encryption scheme or re-routing. This has the potential to improve the trade-off between overhead and traffic analysis resistance. A complementary strategy to ease the achievement of constant-size flows, is to rethink the format of DNS queries and its headers. Reducing the number of bits required for the headers would make it easier to fit queries and responses in one constant-size packet with small overhead.

Besides protection against third party observers, it is important that the community also considers protection from the resolvers. The current deployment of DoH, both on the side of resolvers and browsers, concentrates all lookups in a small number of actors that can observe the behavior of users. More research in the direction of Oblivious DNS [78] is needed to ensure that no parties can become main surveillance actors.

References

[1] Stephane Bortzmeyer. DNS privacy considerations. 2015.

[2] Christian Grothoff, Matthias Wachs, Monika Ermert, and Jacob Appelbaum. NSA's MORECOWBELL: Knell for DNS. Unpublished technical report, 2017.

[3] The NSA files decoded. https://www.theguardian.com/us-news/the-nsa-files. Accessed: 2019-05-13.

[4] Earl Zmijewski. Turkish Internet Censorship Takes a New Turn. https://dyn.com/blog/turkish-internet-censorship, 2014. Accessed: 2018-12-06.

[5] Nicholas Weaver, Christian Kreibich, and Vern Paxson. Redirecting dns for ads and profit. In *FOCI*, 2011.

[6] S. Bortzmeyer. DNS Privacy Considerations. RFC 7626, 2015.

[7] A.Cooper, H. Tschofenig, B. Aboba, J. Peterson, J. Morris, M. Hansen, and R. Smith. Privacy Considerations for Internet Protocols. RFC 6973, 2015.

[8] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman. Specification for DNS over Transport Layer Security (TLS). RFC 7858, RFC Editor, May 2016.

[9] P. Hoffman and P. McManus. DNS Queries over HTTPS (DoH). RFC 8484, RFC Editor, October 2018.

[10] A. Mayrhofer. The edns(0) padding option. RFC 7830, RFC Editor, May 2016.

[11] Google Public DNS over HTTPS (DoH) supports RFC 8484 standard. https://security.googleblog.com/2019/06/google-public-dns-over-https-doh.html. Accessed: 2019-04-03.

[12] Cloudflare DNS over HTTPS. https://developers.cloudflare.com/1.1.1.1/dns-over-https/. Accessed: 2018-05-07.

[13] Selena Deckelmann. dns over https (doh) testing on beta. https://blog.mozilla.org/futurereleases/2018/09/13/dns-over-https-doh-testing-on-beta, 2018. accessed: 2018-12-30.

[14] Whats next in making Encrypted DNS-over-HTTPS the Default. https://blog.mozilla.org/futurereleases/2019/09/06/whats-next-in-making-dns-over-https-the-default/. Accessed: 2019-09-07.

[15] Geoff Huston. DOH! DNS over HTTPS explained. https://labs.ripe.net/Members/gih/doh-dns-over-https-explained, 2018. Accessed: 2018-12-27.

[16] Andriy Panchenko, Fabian Lanze, Andreas Zinnen, Martin Henze, Jan Pennekamp, Klaus Wehrle, and Thomas Engel. Website fingerprinting at internet scale. In *Network & Distributed System Security Symposium (NDSS)*, pages 1–15. IEEE Computer Society, 2016.

[17] Tao Wang and Ian Goldberg. On realistically attacking tor with website fingerprinting. In *Privacy Enhancing Technologies Symposium (PETS)*, pages 21–36. De Gruyter Open, 2016.

[18] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *USENIX Security Symposium*, pages 1–17. USENIX Association, 2016.

[19] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *ACM Conference on Computer and Communications Security (CCS)*, pages 1928–1943. ACM, 2018.

[20] Andrew M. White, Austin R. Matthews, Kevin Z. Snow, and Fabian Monrose. Phonotactic reconstruction of encrypted voip conversations: Hookt on fon-iks. In *IEEE Symposium on Security and Privacy (S&P 2011)*, 2011.

[21] Se Eun Oh, Shuai Li, and Nicholas Hopper. Fingerprinting keywords in search queries over tor. *PoPETs*, 2017.

[22] The DNS Privacy Project. Initial Performance Measurements (Q1 2018). https://dnsprivacy.org/wiki/pages/viewpage.action?pageId=14025132, 2018. Accessed: 2018-12-27.

[23] The DNS Privacy Project. Initial Performance Measurements (Q4 2018). https://dnsprivacy.org/wiki/pages/viewpage.action?pageId=17629326, 2018. Accessed: 2018-12-27.

[24] Austin Hounsel, Kevin Borgolte, Paul Schmitt, Jordan Holland, and Nick Feamster. Analyzing the costs (and benefits) of DNS, DoT, and DoH for the modern web. *CoRR*, abs/1907.08089, 2019.

[25] Timm Bottger, Felix Cuadrado, Gianni Antichi, Eder Leao Fernandes, Gareth Tyson, Ignacio Castro, and Steve Uhlig. An empirical study of the cost of DNS-over-HTTPs. 2019.

[26] Brad Miller, Ling Huang, Anthony D Joseph, and J Doug Tygar. I know why you went to the clinic: Risks and realization of HTTPS traffic analysis. In *Privacy Enhancing Technologies Symposium (PETS)*, pages 143–163. Springer, 2014.

[27] Xiapu Luo, Peng Zhou, Edmond W. W. Chan, Wenke Lee, Rocky K. C. Chang, and Roberto Perdisci. HTTPOS: Sealing information leaks with browser-side obfuscation of encrypted flows. In *Network & Distributed System Security Symposium (NDSS)*. IEEE Computer Society, 2011.

[28] Milad Nasr, Amir Houmansadr, and Arya Mazumdar. Compressive traffic analysis: A new paradigm for scalable traffic analysis. In *ACM SIGSAC Conference on Computer and Communications Security, CCS*, 2017.

[29] DNS over Tor. https://developers.cloudflare.com/1.1.1.1/fun-stuff/dns-over-tor/. Accessed: 2018-12-09.

[30] Tao Wang and Ian Goldberg. Improved Website Fingerprinting on Tor. In *ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 201–212. ACM, 2013.

[31] iodine. https://code.kryo.se/iodine/. Accessed: 2019-05-13.

[32] Spamhaus. https://www.spamhaus.org/zen/. Accessed: 2019-05-13.

[33] Anonymous. Towards a comprehensive picture of the great firewall's dns censorship. In *USENIX Security Symposium*. USENIX Association, 2014.

[34] Paul Pearce, Ben Jones, Frank Li, Roya Ensafi, Nick Feamster, Nick Weaver, and Vern Paxson. Global measurement of dns manipulation. In *USENIX Security Symposium. USENIX*, page 22, 2017.

[35] Saikat Guha and Paul Francis. Identity trail: Covert surveillance using DNS. In *Privacy Enhancing Technologies Symposium (PETS)*, 2007.

[36] DNSSEC: DNS Security Extensions. https://www.dnssec.net/. Accessed: 2018-12-09.

[37] DNSCrypt. https://dnscrypt.info/. Accessed: 2018-12-09.

[38] G. Huston. DOH! DNS over HTTPS explained. https://blog.apnic.net/2018/10/12/doh-dns-over-https-explained/. Accessed: 2018-12-26.

[39] Kenji Baheux. Experimenting with same-provider DNS-over-HTTPS upgrade. https://blog.mozilla.org/futurereleases/2018/09/13/dns-over-https-doh-testing-on-beta, 2019. accessed: 2019-09-13.

[40] T. Reddy, D. Wing, and P. Patil. Dns over datagram transport layer security (dtls). RFC 8094, RFC Editor, February 2017.

[41] Specification of DNS over Dedicated QUIC Connections. https://www.ietf.org/id/draft-huitema-quic-dnsoquic-05.txt. Accessed: 2018-12-09.

[42] Haya Shulman. Pretty bad privacy: Pitfalls of DNS encryption. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society*. ACM, 2014.

[43] Dominik Herrmann, Christian Banse, and Hannes Federrath. Behavior-based tracking: Exploiting characteristic patterns in DNS traffic. *Computers & Security*, 2013.

[44] Basileal Imana, Aleksandra Korolova, and John S. Heidemann. Enumerating Privacy Leaks in DNS Data Collected above the Recursive. 2017.

[45] DNS-OARC: Domain Name System Operations Analysis and Research Center. https://www.dns-oarc.net/tools/dsc. Accessed: 2018-11-26.

[46] DNS-STATS: ICANN's IMRS DNS Statistics. https://www.dns.icann.org/imrs/stats. Accessed: 2018-12-06.

[47] Use DNS data to identify malware patient zero. https://docs.splunk.com/Documentation/ES/5.2.0/Usecases/PatientZero. Accessed: 2018-12-06.

[48] DNS Analytics. https://constellix.com/dns/dns-analytics/. Accessed: 2018-12-06.

[49] Anonymous. The collateral damage of internet censorship by dns injection. *SIGCOMM Comput. Commun. Rev.*, 42(3):21–27, 2012.

[50] Alerts about BGP hijacks, leaks, and outages. https://bgpstream.com/. Accessed: 2019-05-13.

[51] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *ACM Conference on Computer and Communications Security (CCS)*, pages 605–616. ACM, 2012.

[52] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *USENIX Security Symposium*, pages 143–157. USENIX Association, 2014.

[53] Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom Van Goethem, and Wouter Joosen. Automated website fingerprinting through deep learning. In *Network & Distributed System Security Symposium (NDSS)*. Internet Society, 2018.

[54] Allison McDonald, Matthew Bernhard, Luke Valenta, Benjamin VanderSloot, Will Scott, Nick Sullivan, J Alex Halderman, and Roya Ensafi. 403 forbidden: A global view of cdn geoblocking. In *Proceedings of the Internet Measurement Conference 2018*, pages 218–230. ACM, 2018.

[55] Roberto Gonzalez, Claudio Soriente, and Nikolaos Laoutaris. User profiling in the time of https. In *Proceedings of the 2016 Internet Measurement Conference*, pages 373–379. ACM, 2016.

[56] Marc Liberatore and Brian Neil Levine. "Inferring the source of encrypted HTTP connections". In *ACM Conference on Computer and Communications Security (CCS)*, pages 255–263. ACM, 2006.

[57] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-Boo, I still see you: Why efficient traffic analysis countermeasures fail. In *IEEE Symposium on Security and Privacy (S&P)*, pages 332–346. IEEE, 2012.

[58] Qixiang Sun, Daniel R Simon, Yi-Min Wang, Wilf Russel, Venkata N. Padmanabhan, and Lili Qiu. Statistical identification of encrypted web browsing traffic. In *IEEE Symposium on Security and Privacy (S&P)*, pages 19–30. IEEE, 2002.

[59] Andrew Hintz. Fingerprinting websites using traffic analysis. In *Privacy Enhancing Technologies Symposium (PETS)*, pages 171–178. Springer, 2003.

[60] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial Naïve-Bayes classifier. In *ACM Workshop on Cloud Computing Security*, pages 31–42. ACM, 2009.

[61] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In *ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 103–114. ACM, 2011.

[62] Mario Almeida, Alessandro Finamore, Diego Perino, Narseo Vallina-Rodriguez, and Matteo Varvello. Dissecting dns stakeholders in mobile networks. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, pages 28–34. ACM, 2017.

[63] David Plonka and Paul Barford. Context-aware clustering of dns query traffic. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, pages 217–230. ACM, 2008.

[64] Rob Jansen, Marc Juarez, Rafael Galvez, Tariq Elahi, and Claudia Diaz. Inside job: Applying traffic analysis to measure tor from within. In *Network & Distributed System Security Symposium (NDSS)*. Internet Society, 2018.

[65] Rebekah Overdorf, Marc Juarez, Gunes Acar, Rachel Greenstadt, and Claudia Diaz. How unique is your onion? an analysis of the fingerprintability of tor onion services. In *ACM Conference on Computer and Communications Security (CCS)*, pages 2021–2036. ACM, 2017.

[66] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *European Symposium on Research in Computer Security (ESORICS)*, pages 27–46. Springer, 2016.

[67] Ariel Stolerman, Rebekah Overdorf, Sadia Afroz, and Rachel Greenstadt. Breaking the closed-world assumption in stylometric authorship attribution. In *IFIP Int. Conf. Digital Forensics*, 2014.

[68] Muhammad Ahmad Bashir and Christo Wilson. Diffusion of user tracking data in the online advertising ecosystem. 2018.

[69] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. A critical evaluation of website fingerprinting attacks. In *ACM Conference on Computer and Communications Security (CCS)*, pages 263–274. ACM, 2014.

[70] John S Otto, Mario A Sánchez, John P Rula, and Fabián E Bustamante. Content delivery and the natural evolution of DNS: remote dns trends, performance issues and alternative solutions. In *Proceedings of the 2012 Internet Measurement Conference*, pages 523–536. ACM, 2012.

[71] John P Rula and Fabian E Bustamante. Behind the curtain: Cellular dns and content replica selection. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 59–72. ACM, 2014.

[72] Platon Kotzias, Abbas Razaghpanah, Johanna Amann, Kenneth G Paterson, Narseo Vallina-Rodriguez, and Juan Caballero. Coming of age: A longitudinal study of tls deployment. In *Proceedings of the Internet Measurement Conference*, pages 415–428. ACM, 2018.
[73] J. Damas, M. Graff, and P. Vixie. Extension mechanisms for dns (edns(0)). RFC 6891, RFC Editor, April 2013.
[74] Padding Policies for Extension Mechanisms for DNS (EDNS(0)). https://tools.ietf.org/html/rfc8467. Accessed: 2019-05-10.
[75] Michael Carl Tschantz, Sadia Afroz, Anonymous, and Vern Paxson. Sok: Towards grounding censorship circumvention in empiricism. In *IEEE Symposium on Security and Privacy (S&P)*, pages 914–933. IEEE, 2016.
[76] Sheharbano Khattak, Tariq Elahi, Laurent Simon, Colleen M Swanson, Steven J Murdoch, and Ian Goldberg. SoK: Making sense of censorship resistance systems. *Privacy Enhancing Technologies Symposium (PETS)*, 2016(4):37–61, 2016.
[77] URL testing lists intended for discovering website censorship. https://github.com/citizenlab/test-lists. Accessed: 2019-09-11.
[78] Paul Schmitt, Anne Edmundson, Allison Mankin, and Nick Feamster. Oblivious dns: practical privacy for dns queries. *Proceedings on Privacy Enhancing Technologies*, 2019(2):228–244, 2019.
[79] Davis Yoshida and Jordan Boyd-Graber. Using confusion graphs to understand classifier error. In *Proceedings of the Workshop on Human-Computer Question Answering*, pages 48–52, 2016.
[80] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

APPENDIX

A. Performance metrics.

In this section, we provide an overview of our classifer evaluation metrics. We use standard metrics to evaluate the performance of our classifier: *Precision*, *Recall* and *F1-Score*. We compute these metrics per class, where each class represents a webpage. We compute these metrics on a class as if it was a "one vs. all" binary classification: we call "positives" the samples that belong to that class and "negatives" the samples that belong to the rest of classes. Precision is the ratio of true positives to the total number of samples that were classified as positive (true positives and false positives). Recall is the ratio of true positives to the total number of positives (true positives and false negatives). The F1-score is the harmonic mean of precision and recall.

B. Estimation of Probabilities

In this section, we explain how we estimated the probabilities for the entropy analysis in Section VII.

We define the *anonymity set* of a trace s as a multiset:

$$A(s) := \{w^{\mathrm{m}_s(w)}\},$$

where $\mathrm{m}_s(w)$ is the multiplicity of a website w in $A(s)$. The multiplicity is a function defined as the number of times that trace s occurs in w.

The probability $\Pr[W = w \mid S_l = o]$ can be worked out using Bayes. For instance, for website w,

$$\Pr[W = w \mid S_l = o] = \frac{\Pr[W = w]\Pr[S_l = o \mid W = w]}{\sum_{i=1}^{m} \Pr[W = w_i]\Pr[S_l = o \mid W = w_i]} \tag{1}$$

We assume the distribution of priors is uniform, i.e., the probability of observing a website is the same for all websites: $\Pr[w_i] = \Pr[w_j] \quad \forall i, j$.

We acknowledge that this is an unrealisitc assumption but we provide the mathematical model to incorporate the priors in case future work has the data to estimate them.

Assuming uniform priors allows us to simplify the Bayes rule formula since we can factor out $\Pr[W = w_i]$ in Equation 1.

Regarding the likelihoods of observing the traces given a website, we can use the traffic trace samples in our dataset as observations to estimate them:

$$\Pr[S_l = o \mid W = w_i] \approx \frac{\mathrm{m}_s(w_i)}{k_i}$$

Since we have a large number of samples for all the sites, we can fix the same sample size for all sites: $k_i = k_j \quad \forall i, j$. A fixed sample size allows us to factor out k_i in our likelihood estimates and, thus, the posterior can be estimated simply as:

$$\Pr[W = w \mid S_l = o] \approx \frac{\mathrm{m}_s(w)}{\sum_{i=1}^{m} \mathrm{m}_s(w_i)} = \frac{\mathrm{m}_s(w)}{|A(s)|}$$

.

That is the multiplicity of website w divided by the size of the s's anonymity set, which can be computed efficiently for all w and s using vectorial operations.

C. Extra results on attack robustness

In this section, we provide additional information on our experiment measuring the influence of end user's platform (Section V-C3). Figure 12 shows the difference in TLS record sizes for both platforms – the sizes follow a similar distribution, with a shift. Table IX shows the improvement in performance we get when removing this shift.

Table IX: Improvement in cross platform performance when removing the shift (standard deviation less than 1%).

Train	Test	Precision	Recall	F-score
DESKTOP	RPI	0.630	0.654	0.614
RPI	DESKTOP	0.552	0.574	0.535

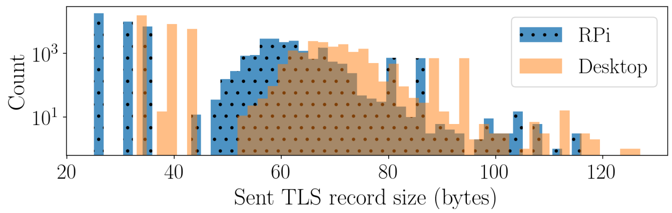

Figure 12: Distribution of user's sent TLS record sizes in platform experiment.

D. Survivors and Easy Preys

In this section, we show results from our analysis of survivors and easy preys, as discussed in Section V-C. We show the top-10 webpages with highest-mean and lowest-variance (Table X), lowest-mean and lowest-variance (Table XI), and highest-variance F1-score (Table XII).

Table X: Top-10 with highest-mean and lowest-variance F1-Score

Alexa Rank	Mean F1-Score	Stdev F1-Score	Domain name
777	0.95	0.08	militaryfamilygiftmarket.com
985	0.95	0.08	myffpc.com
874	0.95	0.08	montrealhealthygirl.com
712	0.95	0.08	mersea.restaurant
1496	0.95	0.08	samantha-wilson.com
1325	0.95	0.08	nadskofija-ljubljana.si
736	0.95	0.08	michaelnewnham.com
852	0.95	0.08	mollysatthemarket.net
758	0.95	0.08	midwestdiesel.com
1469	0.95	0.08	reclaimedbricktiles.blogspot.si

Table XI: Top-10 sites with lowest-mean and lowest-variance F1-Score

Alexa Rank	Mean F1-Score	Stdev F1-Score	Domain name
822	0.11	0.10	mjtraders.com
1464	0.11	0.08	ravenfamily.org
853	0.14	0.09	moloneyhousedoolin.ie
978	0.14	0.17	mydeliverydoctor.com
999	0.17	0.10	myofascialrelease.com
826	0.17	0.11	mm-bbs.org
1128	0.17	0.10	inetgiant.com
889	0.18	0.14	motorize.com
791	0.18	0.15	mindshatter.com
1193	0.20	0.14	knjiznica-velenje.si

Table XII: Top-10 sites with highest-variance F1-Score

Alexa Rank	Mean F1-Score	Stdev F1-Score	Domain name
1136	0.43	0.53	intothemysticseasons.tumblr.com
782	0.43	0.53	milliesdiner.com
766	0.43	0.53	mikaelson-imagines.tumblr.com
1151	0.43	0.53	japanese-porn-guidecom.tumblr.com
891	0.42	0.52	motorstylegarage.tumblr.com
909	0.42	0.52	mr-kyles-sluts.tumblr.com
918	0.44	0.52	mrsnatasharomanov.tumblr.com
1267	0.52	0.49	meander-the-world.com
238	0.48	0.49	caijing.com.cn
186	0.48	0.48	etsy.com

E. Confusion Graphs

We have used *confusion graphs* to understand the errors of the classifier. Confusion graphs are the graph representation of confusion matrices. They allow to easily visualize large confusion matrices by representing misclassifications as directed graphs. Confusion graphs have been used in website fingerprinting [65] and other classification tasks to understand classifier error [79].

Figures 13, 14 and 16 show the classification errors in the form of confusion graphs for some of the experiments presented in Sections V and VI. The graphs were drawn using Gephi, a software for graph manipulation and visualization. Nodes in the graph are domains and edges represent misclassifications between domains. The edge source is the true label of the sample and the destination is the domain that the classifier confused it with. The direction of the edge is encoded clockwise in the curvature of the edge. Node size is proportional to the node's degree and nodes are colored according to the community they belong to, which is determined by the Lovain community detection algorithm [80].

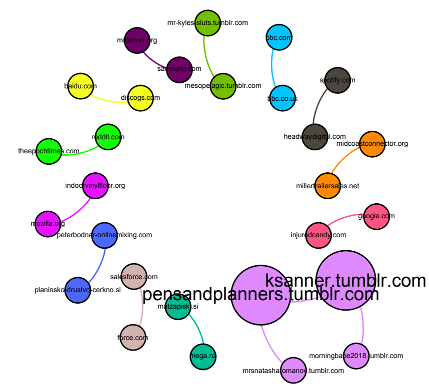

Figure 13: Confusion graph for the misclassifications in LOC1 that happen in more than one fold of the cross-validation and have different domain name length. We observe domains that belong to the same CDN (e.g., tumblr) or belong to the same entity (e.g., BBC, Salesforce). For others, however, the cause of the misclassification remains an open question.

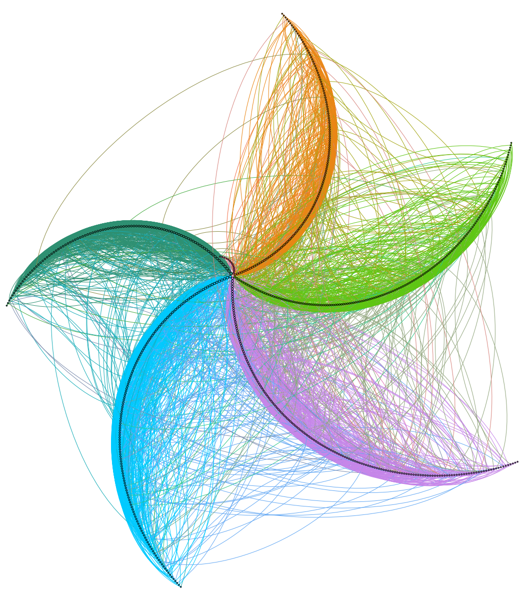

Figure 14: Confusion graph for all Tor misclassifications. We did not plot the labels to remove clutter. We observe that domains in one “petal” of the graph tend to classify between each other.

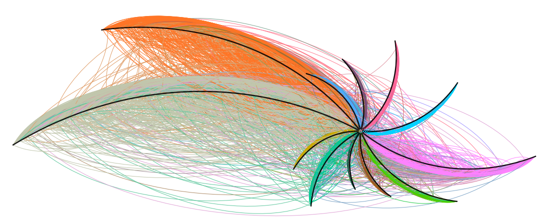

Figure 15: Confusion graph for all perfect padding misclassifications. We observe a larger number of petals as compared to DNS-over-Tor. Domains in one petal of the graph tend to classify between each other.

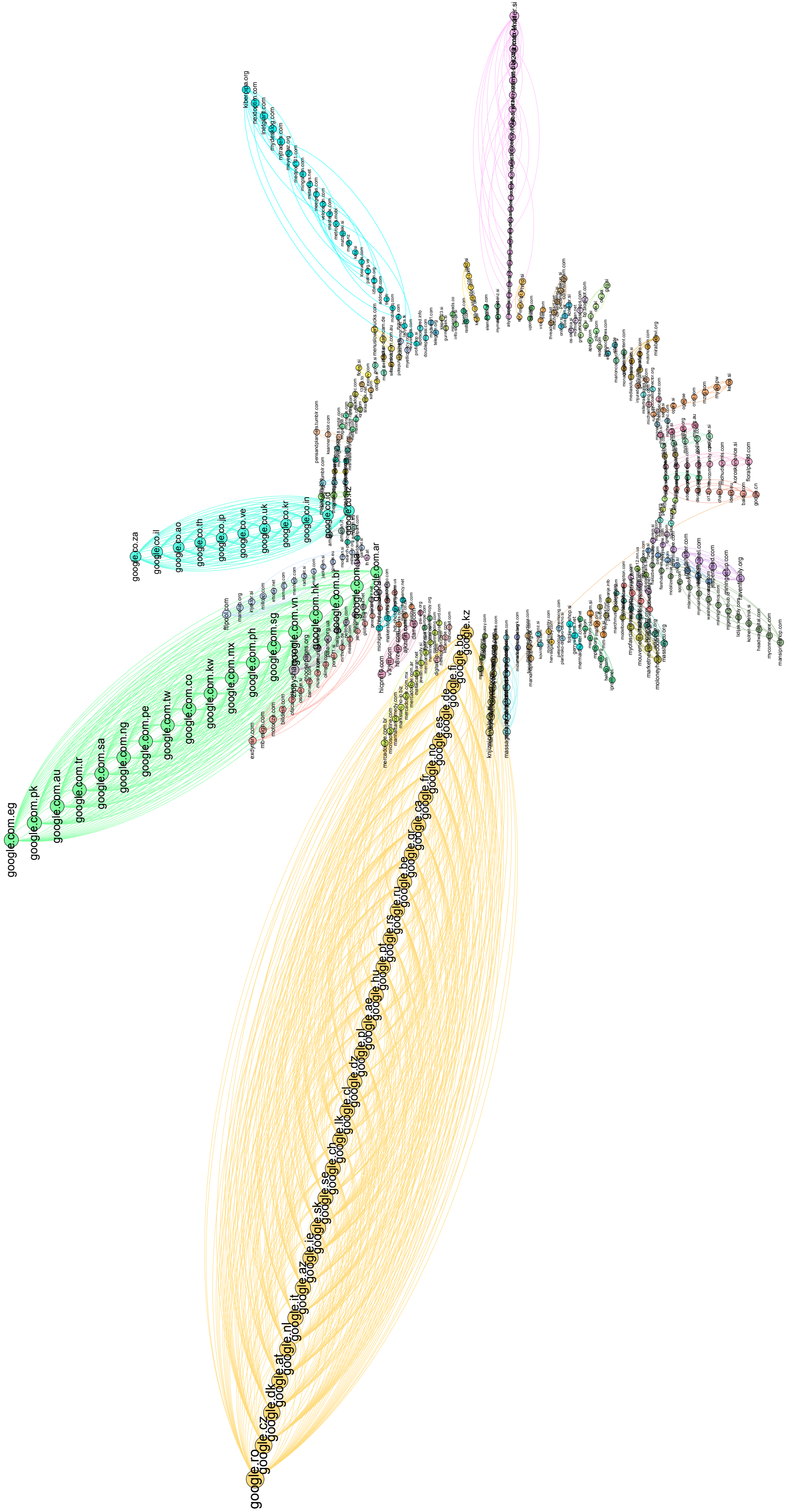

Figure 16: Confusion graph for all misclassifications in LOC1. We observe clusters of domains such as Google and clusters of domains that have the same name length. Interestingly, the only inter-cluster edge we observe is between one of the Google clusters and a cluster that mostly contains Chinese domains.