6-30-2023

Lightweight and Effective Website Fingerprinting over Encrypted DNS

Yong Shao
Loyola University Chicago

Kenneth Hernandez
Loyola University Chicago

Kia Yang
Loyola University Chicago

Eric Chan-Tin
dchantin@luc.edu

Mohammed Abuhamad
Loyola University Chicago

Recommended Citation

Shao, Yong; Hernandez, Kenneth; Yang, Kia; Chan-Tin, Eric; and Abuhamad, Mohammed. Lightweight and Effective Website Fingerprinting over Encrypted DNS. 2023 Silicon Valley Cybersecurity Conference (SVCC), , : 1-8, 2023. Retrieved from Loyola eCommons, Computer Science: Faculty Publications and Other Works, http://dx.doi.org/10.1109/SVCC56964.2023.10165086

Lightweight and Effective Website Fingerprinting over Encrypted DNS

***Abstract*—The DNS over HTTPS (DoH) protocol is implemented to improve the original DNS protocol that uses unencrypted DNS queries and responses. With the DNS traffic, an eavesdropper can easily identify websites that a user is visiting. In order to address this concern of web privacy, encryption is used by performing a DNS lookup over HTTPS. In this paper, we studied whether the encrypted DoH traffic could be exploited to identify websites that a user has visited. This is a different type of website fingerprinting by analyzing encrypted DNS network traffic rather than the network traffic between the client and the web server. DNS typically uses fewer network packets than a website download. Our model and algorithm can accurately predict one out of $10,000$ websites with a 95% accuracy using the first 50 DoH packets. In the open-world environment with $100,000$ websites, our model achieves an F1-score of 93%.**

***Index Terms*—Webbrowsing, Website Fingerprinting, DoH, Privacy, DNS**

I. Introduction

When surfing the Internet via a web browser, an essential step is to translate a user-friendly domain name to an IP address. It is the responsibility of the DNS (Domain Name System) protocol. However, the DNS queries and responses are in plaintext, which has become an increasingly significant concern in web privacy. The DNS-over-HTTPS (DoH) protocol was suggested to encrypt the DNS traffic over the HTTPS layer. A DoH-enabled web browser sends an encrypted DoH request to a DoH server, which resolves the DNS query in the DoH request. Then the DoH server sends the DNS response back in an encrypted manner.

Traditional website fingerprinting (WF) [1] captures the HTTP traffic between a web browser and web server and extracts the network traffic metadata, such as network packet size, network packet direction, and timing information between network packets. It then applies some machine learning algorithm to retrieve any pattern from the metadata and predict the website a user has visited. Website fingerprinting has been shown to work in encrypted network traffic and anonymized network traffic using Tor [2].

The motivation for applying website fingerprinting attack techniques to DNS over HTTPS (DoH) network traffic is that DoH was deployed to protect the privacy of users on the web. Our goal is to show that DoH does not completely protect the privacy of users and the website that a user is visiting can still be obtained through the encrypted DNS traffic. Further work is needed to ensure that the DNS traffic is not unique for each website. Moreover, DNS traffic is shorter, thus performing website fingerprinting on DNS traffic can be more efficient using less resources.

This paper focuses on website fingerprinting using the DoH traffic only between the web browser and the DoH server. The DoH traffic is generally smaller than the web traffic required in the traditional website fingerprinting since it is only required while resolving the domain name of a web server and other domain names embedded in the website. We explore the task of website fingerprinting using the DoH traffic in two experiment settings covering closed- and open-world scenarios. In the closed-world experiment, the goal is to identify each website within a list of selected websites. In the open-world experiment, the goal is to classify whether a website belongs to a list of selected websites.

Although there have been some attempts at website fingerprinting using DoH traffic [3]–[5], this work provides some significant improvements, such as 1) we use a much bigger dataset with $100,000$ websites and 2) we are the first to show that using a closed-world model for training to then predict the open-world model for DoH traffic is feasible. DoH is also much more prevalent now since Google and Mozilla have enabled DoH by default in Chrome and Firefox in 2020.

Contributions. The contributions of this paper are as follows:

- Apply website fingerprinting techniques to the DoH traffic to predict the website. Our models use fewer input features (i.e., at most 50 features for each website) than previous website fingerprinting work while still achieving a high accuracy in both the closed-world and open-world environments.
- The first to conduct different open-world experiments, such as a large proportion (i.e., up to 90%) of testing websites are unseen websites during training, using the closed-world model to predict the open-world website for DoH traffic.

II. Background

A. Threat Model

Figure 1 shows our threat model. The adversary can see all encrypted DNS (DNS over HTTPS) traffic but might not necessarily see the web traffic. The adversary can only eavesdrop on the traffic and cannot modify, inject, or drop packets. The goal of the adversary is to predict the website that the user is visiting.

B. Related Work

Website fingerprinting (WF) attacks [1], [6]–[22] attempt to predict the website a user is visiting based only on the network traffic metadata such as the size of each network packet, the direction of each network packet, and the timing information

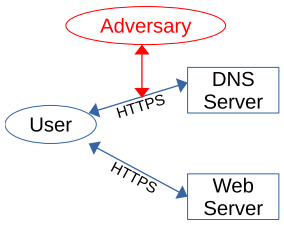

Fig. 1. Threat Model.

between each network packet. The network traffic is encrypted and usually anonymized, that is the real destination IP address is not visible.

Various features [23], such as packet timing [24], cumulative sizes [25], and n-grams [5], have been used. SVM (Support Vector Machine), Random Forest, k-NN (k-nearest-neighbor), and deep learning [26] have all been used as the machine learning algorithm. There are two general models for website fingerprinting: closed-world and open-world. The closed-world model is a multi-class model where the adversary trains the model to predict the correct website out of N websites. The open-world model is a binary classification task where a small number of websites are in a monitored set and a larger number of websites are in a unmonitored set. The adversary aims to determine if a user is visiting a website in the monitored set or not. For the closed-world model, the accuracy has been in the 95+$\%$ showing that WF attacks are a threat to web privacy. For open-world model, previous work achieved over 90% True Positive Rate with low False Positive Rate.

DNS is unencrypted, allowing anybody to eavesdrop on the server that a user is visiting even though encryption of the actual network traffic is done. DNS over HTTPS provides encryption to hide the DNS name. Moreover, all the DNS lookups go to one DNS server (e.g., Cloudflare or Google), which does not leak any data. Previous works looking at the privacy of DNS [3], [5] have shown that padding is not enough to preserve the privacy of DNS over TLS (DoT) [27] and DNS over HTTPS (DoH) traffic. 66% of all traffic were correctly labeled. Hoang et al. [28] showed that IP addresses can still allow network-level adversaries to determine the website visited with an 84% success rate.

The idea of WF attacks have been applied to other applications such as social media [29], voice recognition [30], web searches [31], and DNS over HTTPS [4], showing that the idea behind using network traffic metadata such as packet sizes and number of packets can be applied to a broad range of applications to determine the content a victim is visiting.

The closest work to this paper is [5], which performed a similar attack on Cloudflare DoH servers. We used a similar algorithm and utilized the first 50 packets. Even though [5] found that there was not much improvement after the 15th packet, we found 50 packets to provide a good accuracy. Our experiments show that website fingerprinting is still possible now after two years from that paper. Our models use the size and direction features of the first 50 outgoing DoH packets. The algorithm we used is Random Forest. This paper makes **significant** improvements and additions from the related paper. First, we used a much larger dataset of $10,000$ websites in the closed world and $100,000$ websites in the open world. Even with the larger dataset, we achieved a 95% accuracy in closed-world environment with $10,000$ websites and a 93% F1-score in open-world experiment with $100,000$ websites. Our research also looked at different models for the open-world experiments —to the best of our knowledge, we are the *first to do so*, especially using the closed-world model to predict websites in the open-world environment. The high prediction accuracy of this model shows that an adversary only needs to train on a small closed-world model and can scale the attack to the size of the Internet.

III. Methods

A. Experimental Setup

Figure 2 shows the general setup of our experiments. Before starting the data collection procedure, Firefox safe mode was disabled to turn off the pop-up window asking the user to launch Firefox in safe mode when Firefox crashes. To collect one sample, we started Firefox for 10 seconds to ensure that the web browser completed the initial communication with the Cloudflare server. We started *tcpdump* in capture mode, set a filter with two IP addresses of Cloudflare server and destination port 443, and waited for 5 seconds to confirm that *tcpdump* was ready to go. Then we opened a tab in Firefox to connect to a website and waited for 30 seconds for the webpage to finish loading. After the web page was loaded, we stopped *tcpdump* and waited for 5 seconds to ensure that *tcpdump* saved the log file successfully. Finally, we stopped Firefox and waited for 10 seconds to confirm that Firefox was entirely stopped, and then cleaned up the web cache, browsing history, and stored session. It took 1 minute in total to capture one sample of a website. This process was followed for all the websites and samples.

B. Data Collection

The Alexa top 1 million websites list, downloaded on February 2021, was used for data collection. We collected the DoH traffic for the first 10,500 websites in the list and 20 samples for each website, that contributed to our closed-world dataset. The 20 samples of each website are collected from two virtual machines, and there are about 7 days gap between two samples of one website in each virtual machine, so the 20 samples of each website are across 10 weeks. Initially, we estimated that 5% of websites in that list would be unreachable due to various reasons, such as unresolved domain and HTTP server error, so we added 500 more websites to our closed-world data collection. The data collection process is similar to previous website fingerprinting work. The data collecting environment is Ubuntu 20.04 virtual machine, the web browser is Firefox 90.0 with DoH enabled by default, and the DoH resolver is set as Cloudflare server (IP address: 104.16.248.249 and 104.16.249.249). The data was collected from June 2021 to August 2021.

We also collected one (1) sample of the first 100,000 websites in the Alexa list, that contributed to our open-world

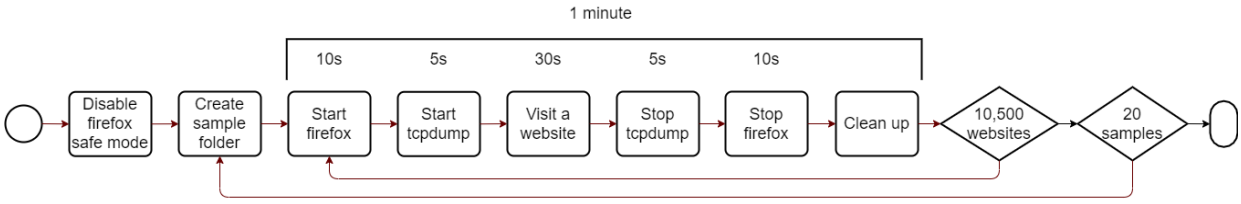

Fig. 2. Data collection procedure.

dataset. The first 10,500 websites from that list are the same as the 20 samples. In effect, we collected 21 samples from 10,500 websites and 1 sample from 89,500 websites.

C. Data Preprocessing

Irrelevant Websites Filtering. As part of our data collection process, we eliminate websites that were not reachable, accessible, or without any actual content, because they do not send/receive any content related to DNS traffic. We used the *cURL* tool to access all the 10,500 websites using the same HTTP header generated by the Firefox web browser. Then we analyzed the returned HTTP code and HTTP response body. Around 800 websites were removed from our dataset due to the following reasons: (1) websites with an unresolved domain name (610 websites); (2) websites with response body less than 20 bytes (39 websites) – this likely means blank websites; (3) websites blocked by a network administrator (110 websites); (4) websites with "404 Not Found" error (30 websites); (5) websites with "403 Forbidden" error (16 websites). We did not remove or filter similar websites, such as *google.com* and *google.co.uk*.

Further Preprocessing. The preprocessed JSON files consisted of the dataset used for our experiments. For the closed-world experiment, we further cleaned the dataset by only including websites with 20 samples. Thus, some websites with fewer than 20 samples, which could be due to the website being unavailable, are filtered.

D. Feature Selection

Each packet of DoH traffic includes one or multiple DNS queries or responses because multiple DNS queries or responses may be merged into one IP packet if they are transmitted close together. We extracted the outgoing packets (i.e., from web browser to DoH server) and incoming packets (i.e., from DoH server to web browser), including the size and direction of each packet. The sequences of outgoing or incoming packets are static for each website since they are mainly determined by sequential resources (e.g., CSS files, JavaScript files, image files) in the web page located in servers with different domain names or subdomains. However, other factors impacting the timing of these packets make the total network traffic dynamic, such as the client environment, network congestion, and DoH server load.

Example. The following are two samples of DoH traffic captured for the website *google.com* under the same client environment. The sizes of each packet are shown. Negative numbers indicate incoming packets, while positive numbers indicate outgoing packets.

Sample 1: [60, 78, 56, 78, -184, -31, -149, -31, 60, 82, 56, 82, -165, -31, -153, -31, 56, 81, 56, 81, -183, -31, -213, -31, 60, 83, 56, 83, -271, -31, -163, -31, 56, 83, 56, 83, -184, -31, -175, -31, 60, 88, 56, 88, **-159, -31, 56, 82, -171, -31, 56, 82,** -174, -31, -186, -31, 56, 95, 56, 95, -166, -31, -178, -31]

Sample 2: [60, 78, 56, 78, -184, -31, -149, -31, 60, 82, 56, 82, -153, -31, -165, -31, 60, 81, 56, 81, -187, -31, -199, -31, 60, 83, 56, 83, -288, -31, -166, -31, 56, 83, 56, 83, -175, -31, -184, -31, 60, 88, 56, 88, **56, 82, 56, 82, -171, -31, -158, -31,** -174, -31, -186, -31, 60, 95, 56, 95, -162, -31, -178, -31]

Both samples have almost the same sequence of numbers following a pattern of four outgoing packets and four incoming packets, but after the 44-th packet, the incoming packets [-159, -31, -171, -31] in the first sample are received earlier than the incoming packets [-171, -31, -158, -31] in the second sample.

Due to this packet "rearrangement", we do not aggregate the size of packets based on direction as is usually done in website fingerprinting. Instead, we treat the size of packets as a unique sequence for each website. Further, we do not mix the outgoing and incoming packets together into one sequence, and we instead create two separate sequences: one sequence of outgoing packets, and the other sequence of incoming packets. For each website, even in a different environment (e.g. a slow network), the sequence of outgoing packets and the sequence of incoming packets should be almost the same as long as the resources in that website have not been changed. As an example, for the website *google.com*, the two separate sequences are as follows. The sequence of outgoing packets is [60, 78, 56, 78, 60, 82, 56, 82, 56, 81, 56, 81, 60, 83, 56, 83, 56, 83, 56, 83, 60, 88, 56, 88, 56, 82, 56, 82, 56, 95, 56, 95]; and the sequence of incoming packets is [-184, -31, -149, -31, -165, -31, -153, -31, -183, -31, -213, -31, -271, -31, -163, -31, -184, -31, -175, -31, -159, -31, -171, -31, -174, -31, -186, -31, -166, -31, -178, -31].

E. Algorithm Selection

Random Forest (RF) is an ensemble machine learning algorithm that contains multiple decision trees. It combines output from all the decision trees to make more stable predictions. The decision tree model makes decisions on data by creating branches from a tree root, which essentially presents the conditions in the data and provides an output on a tree leaf. The decision tree model may overfit the training data (high variance and low bias), leading to an unstable prediction. Random Forest develops individual decision trees by focusing on both observations and variables of the training data. It builds two types of randomness into the decision trees: each

tree is built on different samples randomly selected from the original training samples; each tree is built on a different subset of features randomly selected from original training features. Then the Random Forest takes majority voting from all the decision trees for classification problems or averages the total output from all the decision trees for regression problems.

RF Implementation. We used two approaches for model training: batch mode and non-batch mode. We trained two sets of 50 decision trees in the batch mode by splitting the training data into two groups. In the non-batch mode, we trained one set of 70 decision trees with all the training data. The RF models trained over both approaches can achieve the same generalization performance; the benefit of the batch mode approach is that the model can be trained in a smaller memory environment than the non-batch mode approach, as we trained each set of 50 trees with half of the training data. However, the training time of the batch mode approach will be more than the training time using the non-batch mode approach.

Given a sequence of numbers, which is the length of packets in a time series, we obtained n-gram features from that sequence. For example, for the sequence of outgoing packets captured for *google.com* [60, 78, 56, 78, 60, ...], the bi-gram features are: (60, 78), (78, 56), (56, 78), (78, 60), etc.; and the one-gram features are: (60), (78), (56), etc. We counted the occurrences of n-gram features in each sample and then fit these values with a label into the Random Forest classifier for model training. Due to memory and time constraints, we applied both one-gram and bi-gram features to the RF model. The next section shows that our prediction accuracy was still good using only these two features.

IV. Results and Analysis

A. Closed-world Experiment

In the closed-world experiment, the goal is to build a model to predict one website that a user has visited from a predefined set of websites. Recall that we collected 20 samples of DoH traffic for each of the $10,500$ websites. After removing unreachable or inaccessible websites, we were left with around $10,000$ websites in our dataset. Thus, our model is expected to generalize one class out of the 10,000 classes. We used 90% of the closed-world dataset for model training and 10% of the closed-world dataset for model testing. The cross-validation accuracy was 94%, and the test accuracy was 95%.

A high memory computer engine from Google Cloud Platform (GCP) was used to train the Random Forest model – it contains 80 CPU cores and 640 GB of memory. It takes around 20 minutes to train the RF model.

For this multi-class and single-label problem with a balanced data set, where every class has the same number of samples, the accuracy is chosen as the evaluation metric, and stratified 5-fold cross-validation is used to evaluate the models. We tuned the models based on the average accuracy for all folds. The accuracy in one-fold is defined by the number of correctly predicted samples over the total number of samples in the validation set of each fold.

TABLE I
Cross-validation accuracy for using a different number of incoming or outgoing packets.

Direction	# Packets	Accuracy
Incoming	25	57.37%
	50	64.52%
	75	63.63%
Outgoing	25	85.73%
	50	94.77%
	75	**95.09%**
Both	25	68.33%
	50	87.54%
	75	93.20%

First, we trained the model with a different number of DoH packets and considered either incoming packets only, outgoing packets only, or both direction packets. Table I shows the results. We considered the first (initial) 25 packets in the DoH network traffic, the first 50 packets, and the first 75 packets. It can be seen from the table that considering only outgoing packets has higher accuracy than considering either both direction packets or only incoming packets; considering the first 50 outgoing packets have a similar accuracy compared to the accuracy with the first 75 outgoing packets. Therefore, we used the first 50 outgoing packets for model training and prediction in the rest of this paper.

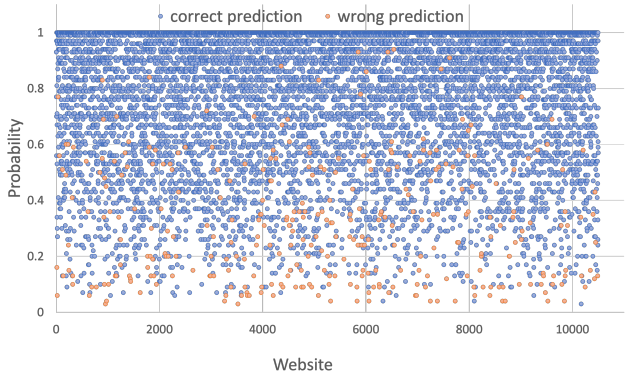

Fig. 3. Closed-world experiment: probability of prediction for the 10,000 websites.

About 5% of websites were classified as a different domain name compared with the expected domain name in the test dataset. Figure 3 shows the probability of prediction for each website. The graph shows the prediction for the valid 10,000 websites out of the total 10,500 websites. Some websites have no data because they are unreachable. Each dot represents one website. A blue dot indicates a correct prediction, and a red dot indicates an incorrect prediction. As can be seen, the majority of websites were correctly predicted. Most of the incorrect predictions have a low probability. We analyzed the 326 wrong predictions (misclassified websites), and they can be categorized into four categories:

1) Same websites with different domain names (e.g., *google.com.hk* – *google.com.br* and *salesforce.com* – *cloudforce.com*.
2) Websites that require user interaction (e.g., a login page).
3) Websites with potential security risks, e.g., the certificate presented by the website is not valid.
4) Websites with a similar or identical sequence of DoH traffic, e.g., 14 websites have the same traffic [60, 77, -171, -31].

The most misclassified websites belong to the first three categories. Our model currently has no way to distinguish them; however, for category 1, they are technically the same website. Category 2 websites can be excluded from our data collection as they require further user interaction. As future work, we could consider using an automated tool like Selenium to interact with each website —this is also future work for the general website fingerprinting attack. Category 3 websites can be discarded due to security risks. The fourth category have only few websites, and technically it is impossible for our models (or any model) to correctly predict the domain name because they have the same sequence of DoH traffic.

B. Open-world Experiment

Due to the high accuracy and less training time of the Random Forest algorithm, we only focused on using Random Forest in the open-world experiment instead of both Random Forest and RNN.

There are only two classes in the open-world experiment: "monitored" and "unmonitored". The model's goal is to predict whether a website that a user has visited is in the "monitored" class or the "unmonitored" class, based on the captured DoH traffic between the web browser and DoH server. This experiment effectively becomes a binary classification problem. More specifically, a DoH fingerprinting attacker would be more interested in whether a user visits a website in the monitored set. Usually, the monitored set/class contains fewer websites than the unmonitored set. For this binary classification problem with unbalanced data, where two classes have uneven distribution, the precision, recall, and F1-score are chosen as the evaluation metrics. Stratified 5-fold cross-validation was then used to evaluate the models. The F1-score of the monitored class was chosen as the validation score. The model was evaluated based on the average validation scores in all folds. The F1-score is a measure of taking both precision and recall into account, and it is defined as the harmonic mean of precision and recall. F1-score $= (2 * \text{Precision} * \text{Recall}) \div (\text{Precision} + \text{Recall})$. The precision for the monitored class is the ratio of the number of samples that are correctly predicted as the monitored class (true positives) divided by the number of samples that are predicted (either correctly or incorrectly) as the monitored class (the sum of true positives and false positives). The recall for the monitored class is the ratio of the number of samples that are correctly predicted as the monitored class (true positives) divided by the number of samples that are in the "monitored" class (the sum of true positives and false negatives).

TABLE II
CROSS-VALIDATION PERFORMANCE IN OPEN-WORLD EXPERIMENT WITH 1,000 MONITORED WEBSITES.

F1-score	Precision	Recall	Threshold
92.4%	93.1%	91.7%	0.20

TABLE III
FINAL MODEL PERFORMANCE WITH VARYING PROPORTION OF UNSEEN WEBSITES IN OPEN-WORLD EXPERIMENT 1.

# Websites	Unseen	F1-score	Precision	Recall	Threshold
10,000	0	92.9%	92.5%	93.2%	0.31
20,000	50%	90.0%	88.5%	91.6%	0.36
50,000	80%	86.1%	85.7%	86.6%	0.44
100,000	90%	80.5%	78.5%	82.7%	0.50

We performed three different open-world experiments with a different number of websites. The first was building a model to monitor 1,000 websites within 10,000 websites; the second was to monitor 10,000 websites within 100,000 websites; the last was using a pre-trained closed-world model to predict open-world websites.

Experiment 1: 1,000 monitored websites. For the first open-world experiment, as shown in Figure 4, we used the collected samples of the closed-world for training, randomly selected 1,000 websites as monitored websites, and treated the other 9,000 websites as unmonitored websites. The model was trained with the Random Forest algorithm and evaluated with stratified 5-fold cross-validation. In each fold, we have 16 samples for each website from "monitored" and "unmonitored" classes for training; we have 4 samples for each website from "monitored" and "unmonitored" classes for validation. The default threshold (that maps probabilities to class labels) always results in poor performance for the classification problem with a severe class imbalance, so we tuned the threshold based on the Precision-Recall curve to improve the performance. Table!II shows the average F1-score for all folds was 92% when the threshold was tuned to 0.20.

Figure 4 also shows where we used the testing dataset. The test dataset is one sample of $100,000$ websites. The training dataset consists of 20 samples of $10,000$ websites. The size of the testing dataset varies from $10,000$ to $100,000$ – this changes the percentage of unseen websites from 0% to 90%. Unseen websites are websites that are not in the training dataset. The F1-score is shown on the left of the figure. When the size of the testing dataset was $10,000$, the F1-score was 93%. When the size of the testing dataset was increased to $100,000$, the F1-score decreases to 80%. Table III shows the F1-score, precision, recall, and threshold used when varying the number of unseen websites. As the number of unseen websites increases, the F1-score, precision, and recall all decrease as expected since there are more websites (up to 90%) that have not been previously seen.

We plotted the Precision-Recall curve for the monitored class in the open-world experiment in Figure 5. Each curve

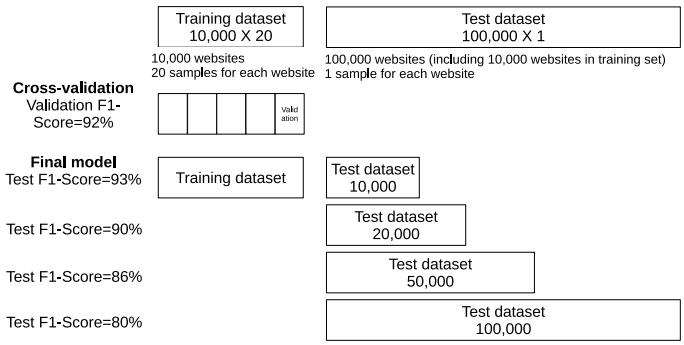

Fig. 4. Open-world Experiment with 10,000 monitored sites, part of the training dataset. The test dataset size varies as shown. F1-score shown on the left.

is for one open-world experimental setup with a different proportion of unseen websites. From this curve, we can see how the precision and recall vary when moving the threshold, and the black dot on each curve indicates the best F1-score metric combining precision and recall under a specific threshold value enumerated in Table III. The F1-score dropped by about 12% when increasing the number of unseen websites by one order of magnitude. We show in "Experiment 3" that our algorithm scales much better.

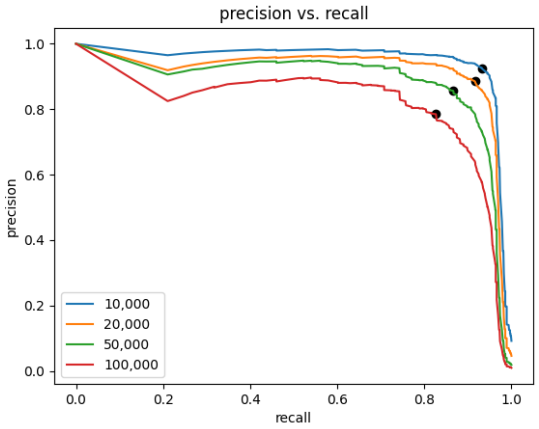

Fig. 5. Precision-Recall curve for the monitored class in open-world experiment 1.

Experiment 2: 10,000 monitored websites. The second open-world experiment considers $10,000$ monitored websites. The difference between the first open-world experiment and the second open-world experiment is that the whole dataset, including both closed-world (20 samples of 10,000 websites) and open-world (1 sample of 100,000 websites), was considered. Figure 6 illustrates this experiment. We chose the closed-world websites as monitored websites (10,000) and the open-world websites as unmonitored websites (90,000). The model was trained with the Random Forest algorithm and evaluated with stratified 10-fold cross-validation. In each fold, we had 90% samples for each website from the "monitored" class, and 1 sample for 90% websites (81,000) from the "unmonitored" class for training; we had 10% samples for each website from the "monitored" class and 1 sample for 10% websites (9,000) from the "unmonitored" class for validation. In the validation dataset of each fold, the 9,000 websites from the "unmonitored" class were never-seen websites during training. Table IV shows the average F1-score for all folds was 93% when the threshold was tuned to 0.63.

TABLE IV
CROSS-VALIDATION PERFORMANCE IN OPEN-WORLD EXPERIMENT 2 WITH 10,000 MONITORED WEBSITES.

F1-score	Precision	Recall	Threshold
93.2%	93.0%	93.4%	0.63

Experiment 3: From Closed-world Model to Open-world. In the last experiment, we used the closed-world model to predict the website from the open-world dataset. This approach differs from previous work where they retrained the model utilizing only the open-world dataset. The training is using the 20 samples for each website from the closed-world dataset. From Figure 7, all the websites outside of the closed-world websites are misclassified. This is expected since these websites do not belong to any classes that the closed-world model was trained on, and the model still generalizes a class from the closed-world websites with a corresponding probability. For websites in the closed-world, the model intends to output a class with a higher probability; for websites out of the closed-world, the model intends to output a class with a lower probability. Therefore, we want to find a cutoff probability that indicates whether the model is confident in its prediction or not, and then we can apply this probability to judge whether one website belongs to the closed-world range (monitored websites) or the open-world range (unmonitored websites).

As the number of monitored websites and the number of unmonitored websites are unbalanced, we used the F1-score of the monitored class to measure the performance of this experiment. We tried different cutoff probabilities and achieved the best F1-score when the cutoff probability is 0.49. As a result, the precision of the monitored class is 67%, the recall of the monitored class is 84%, and the F1-score of the monitored class is 75%.

As shown in Table V,

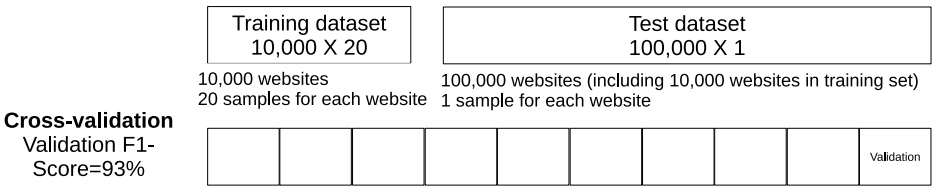

Fig. 6. Open-world experiment with 10,000 monitored websites.

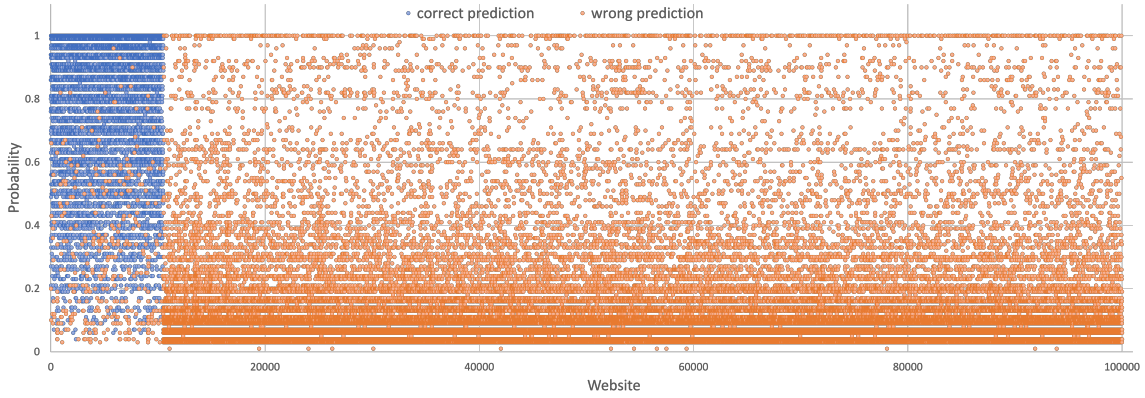

Fig. 7. Open-world experiment: probability of prediction for the 100,000 websites.

TABLE V
OPEN-WORLD EXPERIMENT CONFUSION MATRIX BASED ON THE PROBABILITY GENERALIZED BY THE CLOSED-WORLD MODEL.

	True	False	
Positive	8,022	3,913	
Negative	1,583	85,582	
Total	9,605	89,495	99,100

- The true positive value (8,022) is the number of closed-world websites with probability greater than or equal to the threshold.
- The false positive value (3,913) is the number of open-world websites with probability greater than or equal to the threshold.
- The true negative value (1,583) is the number of closed-world websites with probability less than the threshold.
- The false negative value (85,582) is the number of open-world websites with probability less than the threshold.

The total number of websites in this experiment is $100,000$; after eliminating websites that are not reachable or accessible, the actual number is $99,100$ ($9,605$ in closed-world and $89,495$ in open-world).

We want to emphasize that this is an important step. Regardless of how big the size of the open world dataset is, our model will still be accurate using only the closed world dataset of monitored sites. All the predictions for the unmonitored websites were "correct" in that the website prediction was wrong. Thus none of the unmonitored websites were correctly classified – this shows that the scale of unmonitored website could increase without any impact on the accuracy.

V. DISCUSSION

From the closed-world experiment analysis, we were able to categorize the 5% misclassified websites into four types. For websites in the first category, we can group these websites with different domain names into one single label while still yielding the correct prediction – this will improve the accuracy. For websites in the second category, we could try other web crawling tools like Selenium, which can automatically interact with the website during the data collection process. A 95% accuracy is high for website fingerprinting attacks.

Our training and testing dataset were collected over a 10 week period. The content of a website can change, so our models should also be trained with newly captured traffic. However, the DNS traffic is unlikely to change as often as the actual content; we leave for future work the examination of how effective DoH fingerprinting is over time.

Only the home pages of websites were visited. Unlike traditional website fingerprinting, the DNS traffic should still be similar across web pages within the same website.

We cleared the web browser cache during data collection and made sure that there was no background traffic. If the web browser cache is not cleared, the web browser may not download some of the resources for rendering the HTML page due to the cached resources in the local environment, resulting in less DoH traffic for some websites. Background web traffic may also initiate other DoH traffic, interfering with the intended DoH traffic data collection. Finally, we only considered CloudFlare's DoH server. Future work should

study other DoH servers, and look into DoH fingerprinting with background traffic, web browser cache, and different web browser software.

Although we used the Alexa top websites list, we do not expect the results would be any different if we had used the Tranco [32] list – website fingerprinting through DoH traffic would still be possible.

Ethical Principles. Only lab machines were utilized and the website fingerprinting attack only attempted to identify the websites that our lab machines visited. Although extra traffic was generated for the websites in our dataset, each website received a maximum of 21 visits over a period of 10 weeks which should not have caused too much overhead.

We will make our data collection code, data analysis code, and processed dataset in JSON format publicly available.

VI. Conclusion

In this paper, we have demonstrated that the encrypted DNS protocol (DNS over HTTPS DoH) cannot fully solve the web privacy issue introduced by the unencrypted DNS protocol. Encrypted DNS protects users' private information based on the secure and private HTTPS layer, but an eavesdropper can still identify websites that a user is visiting from the captured DoH traffic, requiring much less data than the current website fingerprinting attacks. Only the initial 50 outgoing packets from web browser to DoH server are needed to identify the website's domain name with a 95% accuracy.

References

[1] A. Hintz, "Fingerprinting websites using traffic analysis," in *Proceedings of the 2Nd International Conference on Privacy Enhancing Technologies*, ser. PET'02. Springer-Verlag, 2003, pp. 171–178.

[2] Tor, https://www.torproject.org/, 2023.

[3] J. Bushart and C. Rossow, "Padding ain't enough: Assessing the privacy guarantees of encrypted {DNS}," in *10th {USENIX} Workshop on Free and Open Communications on the Internet ({FOCI} 20)*, 2020.

[4] D. Vekshin, K. Hynek, and T. Cejka, "Doh insight: Detecting dns over https by machine learning," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, 2020, pp. 1–8.

[5] S. Siby, M. Juarez, C. Diaz, N. Vallina-Rodriguez, and C. Troncoso, "Encrypted DNS ==> Privacy? A Traffic Analysis Perspective," in *NDSS*, 2020.

[6] L. Lu, E.-C. Chang, and M. C. Chan, *Website Fingerprinting and Identification Using Ordered Feature Sequences*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 199–214.

[7] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: Attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier," in *Proceedings of the 2009 ACM Workshop on Cloud Computing Security*, ser. CCSW '09. New York, NY, USA: ACM, 2009, pp. 31–42.

[8] M. Perry, "Experimental defense for website traffic fingerprinting," https://blog.torproject.org/blog/experimental-defense-website-traffic-fingerprinting, 2011.

[9] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, ser. WPES '11. New York, NY, USA: ACM, 2011, pp. 103–114.

[10] T. Wang and I. Goldberg, "Improved website fingerprinting on tor," in *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, ser. WPES '13. ACM, 2013, pp. 201–212.

[11] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 605–616.

[12] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg, "Effective attacks and provable defenses for website fingerprinting," in *Proceedings of the 23rd USENIX Conference on Security Symposium*, ser. SEC'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 143–157.

[13] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '14. New York, NY, USA: ACM, 2014, pp. 263–274.

[14] T. Wang and I. Goldberg, "On realistically attacking tor with website fingerprinting," in *Privacy Enhancing Technologies Symposium (PETS)*, 2016.

[15] J. Hayes and G. Danezis, "k-fingerprinting: A robust scalable website fingerprinting technique," in *25th USENIX Security Symposium (USENIX Security 16)*, Aug. 2016, pp. 1187–1203.

[16] K. Kohls, D. Rupprecht, T. Holz, and C. Pöpper, "Lost traffic encryption: Fingerprinting lte/4g traffic on layer two," in *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec '19. New York, NY, USA: ACM, 2019, pp. 249–260.

[17] C. Wang, J. Dani, X. Li, X. Jia, and B. Wang, "Adaptive fingerprinting: Website fingerprinting over few encrypted traffic," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, 2021, pp. 149–160.

[18] J.-P. Smith, P. Mittal, and A. Perrig, "Website fingerprinting in the age of quic." *Proc. Priv. Enhancing Technol.*, 2021.

[19] T. Wang, "High precision open-world website fingerprinting," in *2020 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society, may 2020, pp. 231–246.

[20] W. Cui, T. Chen, and E. Chan-Tin, "More realistic website fingerprinting using deep learning," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2020, pp. 333–343.

[21] P. Sirinam, N. Mathews, M. S. Rahman, and M. Wright, "Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS, 2019, p. 1131–1148.

[22] W. Cui, T. Chen, C. Fields, J. Chen, A. Sierra, and E. Chan-Tin, "Revisiting assumptions for website fingerprinting attacks," in *ACM Asia Conference on Computer and Communications Security*, ser. AsiaCCS '19. ACM, 2019.

[23] J. Yan and J. Kaur, "Feature selection for website fingerprinting," *PoPETs*, vol. 2018, no. 4, 2018.

[24] M. S. Rahman, P. Sirinam, N. Mathews, K. G. Gangadhara, and M. Wright, "Tik-tok: The utility of packet timing in website fingerprinting attacks," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 3, pp. 5–24, 2020.

[25] A. Panchenko, F. Lanze, A. Zinnen, M. Henze, J. Pennekamp, K. Wehrle, and T. Engel, "Website fingerprinting at internet scale," in *Proceedings of the 23rd Internet Society (ISOC) Network and Distributed System Security Symposium (NDSS 2016)*, 2016.

[26] P. Sirinam, M. Imani, M. Juarez, and M. Wright, "Deep fingerprinting: Undermining website fingerprinting defenses with deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. ACM, 2018, pp. 1928–1943.

[27] R. Houser, Z. Li, C. Cotton, and H. Wang, "An investigation on information leakage of dns over tls," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, ser. CoNEXT '19, New York, NY, USA, 2019, p. 123–137.

[28] N. P. Hoang, A. A. Niaki, P. Gill, and M. Polychronakis, "Domain name encryption is not enough: Privacy leakage via ip-based website fingerprinting," *Proc. Priv. Enhancing Technol.*, vol. 2021, no. 2, 2021.

[29] M. Di Martino, P. Quax, and W. Lamotte, "Realistically fingerprinting social media webpages in https traffic," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, ser. ARES '19, New York, NY, USA, 2019.

[30] J. Hyland, C. Schneggenburger, N. Lim, J. Ruud, N. Mathews, and M. Wright, "What a shame: Smart assistant voice command fingerprinting utilizing deep learning," in *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, ser. WPES, 2021.

[31] S. E. Oh, S. Li, and N. Hopper, "Fingerprinting keywords in search queries over tor," *PoPETs*, vol. 2017, 2017.

[32] V. L. Pochat, T. van Goethem, S. Tajalizadehkhoob, M. Korczynski, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *26th Annual Network and Distributed System Security Symposium, NDSS*. The Internet Society, 2019.