

It's Not Just the Site, It's the Contents: Intra-domain Fingerprinting Social Media Websites Through CDN Bursts

Kailong Wang*

National University of Singapore
dcswaka@nus.edu.sg

Junzhe Zhang*

Sichuan University
2016141052007@stu.scu.edu.cn

Guangdong Bai*

The University of Queensland
g.bai@uq.edu.au

Ryan Ko

The University of Queensland
ryan.ko@uq.edu.au

Jin Song Dong

National University of Singapore
dcsdjs@nus.edu.sg

ABSTRACT

The website fingerprinting (or *inter-domain WSF*), enhanced by various machine learning techniques, has shown its power to identify websites a user has visited. To our best knowledge, a finer-grained problem of *web page fingerprinting* (or *intra-domain WPF*) has not been systematically studied by our research community. The WPF attackers, such as government agencies who enforce Internet censorship, are keen to identify the particular web pages (e.g., a political dissident's social media page) the target user has visited.

In this work, we investigate the intra-domain WPF against social media websites. Our study involves the realistic on-path passive attack scenario. We reveal that delivering large-size data such as images and videos via Content Delivery Networks (CDNs), which is a common practice among social media websites, makes intra-domain WPF highly feasible. The occurring network traffic while the browser is rendering a social media page exhibits temporal and volumetric patterns that are sufficiently recognizable by machine learning algorithms. We characterize such patterns as *CDN bursts*, and use features extracted from them to empower classification algorithms to achieve a high classification accuracy (96%) and a low false positive rate (0.02%). To alleviate the threat of intra-domain WPF, we also propose and evaluate countermeasures such as deviating the packet interval time and inserting dummy requests.

CCS CONCEPTS

- Web fingerprinting → Intra-domain web fingerprinting; • Social media sites; • web browsing privacy;

ACM Reference Format:

Kailong Wang, Junzhe Zhang, Guangdong Bai, Ryan Ko, and Jin Song Dong. 2021. It's Not Just the Site, It's the Contents: Intra-domain Fingerprinting Social Media Websites Through CDN Bursts. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia*. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3442381.3450008>

*This work is partly done when the first two authors visited the University of Queensland. Guangdong Bai is the corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8312-7/21/04.

<https://doi.org/10.1145/3442381.3450008>

1 INTRODUCTION

Deploying encrypted channels has become a security principle for Internet communication. As reported by WatchGuard [52], more than 80% of the 100,000 top-visited websites have been under the protection of HTTPS. Despite providing sufficient confidentiality, the encrypted channels still inadvertently expose the metadata, such as endpoints' IP addresses, packet sizes and traffic direction, to the network attacker. These metadata have been fully or partially abused by fingerprinting attacks for identifying websites that a user has just visited [9, 17–20, 25, 33, 38, 50, 51, 54]. The attacks aim to distinguish the web domains, and thus we refer to them as *inter-domain website fingerprinting* (or *inter-domain WSF*). The state-of-the-art inter-domain WSF has achieved a significantly high accuracy (> 95%). Their capacity, however, may be restricted when fingerprinting pages out of the same social media website which contains millions of theme-wise similar (e.g., layout, style and HTML elements) but *content-wise* distinct pages.

Even though the contents shared on the social media pages are generally considered less privacy-sensitive, the accumulative behaviors of browsing them encompass indicative information regarding a user, such as his/her political orientations and religious beliefs [14]. Concerns on such risks have been on the rise due to the increasing emphasis on censorship and surveillance worldwide nowadays. As shown by [22], the global Internet censorship is undergoing a shift from the website filtering to the content-wise surveillance, particularly amid the increasingly stringent scrutiny on social media websites after their involvement in misinformation spreading related to a series of terrorist attacks in New Zealand [47], Sri Lanka [45] and Myanmar [12].

In this work, we conduct a systematic study on fingerprinting the web pages within social media websites. We refer to it as *intra-domain web page fingerprinting* (or *intra-domain WPF*). Our study assumes an on-path passive attacker that eavesdrops on the network, e.g., a malicious ISP or an honest-but-curious gateway. The attacker merely utilizes the metadata of the traffic¹ without breaking the encryption, and strives to find features to differentiate web pages that are from the same web domain. To make our study realistic, we consider a slightly different threat model from those in the inter-domain WSF literature—the intra-domain WPF is not designated to be against the anonymity networks such as VPN or Onion Router (Tor) that incorporate a proxy or a collection of relays to hide IP addresses and routes (and also part of metadata such as

¹The *traffic* in this paper refers to encrypted traffic unless stated otherwise.

the packet sizes in Tor). As is revealed by the latest statistics, only 7% of Internet users use VPNs frequently [15]; the Tor traffic (400 Gbit/s bandwidth [37]) accounts for merely 0.06% of the total global Internet traffic (77,800 GB/s actual traffic [1]). Hence, our applied threat model may retain sufficient fidelity for studies on the scale of nation-wide network or even Internet.

The intra-domain WPF is more challenging than it appears, although it may be considered as a trivial task due to the success of the inter-domain WSF and the availability of advanced learning techniques. In a typical social media website, almost all its web pages (such as Facebook user profile pages) are generated from the same template, such that similar traffic patterns are exhibited. Consequently, those previously distinguishable features that classification algorithms rely on are blurred out. Some “higher-definition” fingerprints capable of differentiating among such similar web pages are demanded. The used features should be highly individualized and robust in characterizing the subtle differences among pages.

To identify such fingerprints, we first examine the traffic of social media websites (detailed in Section 2.3). Characterized by their so-called *social-content-centric* feature, social media websites experience enormous traffic from the large-size contents such as photos and videos their users upload/download. For example, in Facebook, 350 million photos are uploaded and 8 billion videos are viewed each day [3]. To facilitate efficient delivery of these large-size contents, social media websites tend to host them separately, typically with CDN, from the web servers which host only small objects like HTML and CSS files. We find that when a browser renders a social web page, the traffic between the browser and the CDN may exhibit patterns that can uniquely characterize the page.

Based on this finding, we propose a fingerprint named *CDN bursts*, by adapting the notion of the *traffic bursts* that are extensively used in the inter-domain WSF literature [28, 49]. A CDN burst is defined as an aggregation of several temporally adjacent network packets originated from a certain CDN server (identified by its unique IP address); a sequence of bursts further characterize a web page. From the perspective of the application layer, the personalized social media contents diversify the Document Object Model (DOM) trees among pages. Since the browser’s rendering process in essence is to segment network stream into objects (e.g., images and videos), the DOM structure would then “shape” this semantics-aware segmentation, and implicitly influence how the browser fetches contents from CDN servers. As a result, the information conveyed by CDN bursts, such as the size of a retrieved content and the intervals of retrievals, may (partially) expose the DOM structure. This may be further strengthened by the dynamic or asynchronous loading features (e.g., Ajax) extensively used by social media sites.

To further demystify the CDN bursts, we conduct a series of in-depth studies. We design an algorithm to extract CDN bursts incorporating both temporal and volumetric factors from the network traces, and to derive a feature set for classification. Being aware of the importance to estimate information leaked by the features, thanks to a recent work by Li et al. [25], we also quantify the mutual information between the web pages and each CDN burst, which measures the amount of information contained in and (possibly) leaked by the CDN burst. Our experimental evaluation demonstrates that this amount of information is sufficient for the



Figure 1: The intra-domain WPF attacker model

intra-domain WPF to fingerprint social media pages. Finally, we propose countermeasures by obscuring the temporal and volumetric characteristics that the attacker utilizes to construct CDN bursts.

Contributions. Our study of fingerprinting the social media pages is the first systematic research of its kind. We show that intra-domain WPF, although previously considered as unachievable as finding a needle in a haystack [32], can achieve a performance at least as accurate as that of the state-of-the-art inter-domain WSF. This sheds light on an underresearched domain and may inspire more future research on the *general* intra-domain WPF. As a summary, the contributions of this paper are as follows.

- **A new type of web fingerprinting attack.** We propose the novel intra-domain WPF that aims to differentiate the web pages within the same social media website, assuming a realistic and representative attacker model.
- **A novel fingerprint and its countermeasures.** We propose CDN bursts as a new fingerprint to characterize social media pages. Through the formalization of CDN bursts and quantification of information leakage, we demonstrate their feasibility and effectiveness in intra-domain fingerprinting. We also propose and evaluate the possible countermeasures.
- **A comprehensive evaluation and practical results.** We analyze the performance of the proposed intra-domain WPF attack on four top-visited social media websites, including Facebook, YouTube, Twitter and Instagram. We follow the data collection approaches used by website fingerprinting literature [32, 49] to acquire a representative dataset, which consists of over 424,000 traces from over 12,000 unique web pages. Our dataset is released to facilitate future studies [2].

2 THE INTRA-DOMAIN WPF ATTACK

Similar to the inter-domain WSF, we formalize the intra-domain WPF as a classification problem, where the informative and distinguishable features play a crucial role. In this section, we study the CDN traffic and demonstrate that CDN bursts contain distinguishable features for classifying traffic of social media pages.

2.1 Problem Statement and Attacker Model

The intra-domain WPF is a type of traffic analysis attack. Considering such a scenario: a social media website hosts a set of publicly accessible web pages \mathbb{W} , and the victim user may visit any page in a set \mathbb{V} ($\mathbb{V} \subset \mathbb{W}$). The attacker owns a classifier C trained from a set of pages \mathbb{A} ($\mathbb{A} \subset \mathbb{W}$). Given the traffic τ generated during loading a page $\omega \in \mathbb{V}$, the goal of the attack is to maximize the probability $\Pr(C(\tau) = \iota_\omega)$, where ι_ω is the label² of ω .

Figure 1 shows an overview of the intra-domain WPF attacker model. Similar to prior fingerprinting studies, we assume a *passive*

²We treat each page as a class, so the label means its class label. For simplicity, we abuse ω as the label of itself in the remaining of this paper. An $\omega \notin \mathbb{A}$ is labeled as \perp .

on-path attacker on the network or transport layer. The attacker can passively collect but not intercept, modify, delay or decrypt network packets. In reality, the possible attackers could range from compromised network access points, proxies and routers, local network administrators, Internet Service Providers (ISP), to Autonomous Systems (AS) between the victim user and the web servers.

The attacker model in the intra-domain WPF is slightly different from that in the inter-domain WSF. In the latter, the web domains that a victim has visited are considered sensitive, while in the intra-domain WPF, the attacker is more concerned with the content-centric privacy. In particular, the attacker's general interest is whether the target victim has visited a particular *page* (which includes a politician's tweets, for example), while the visited web domain (e.g., [twitter.com](#)) is out of interest. Therefore, the intra-domain WPF is not aimed at VPN and Onion Router (Tor) which are commonly used for hiding the specific services (i.e., domains). In addition, our attacker model also excludes the web attacker who can remotely execute code on the client side, e.g., JavaScript code injected into a benign web page.

In website fingerprinting studies, it is common to evaluate the attacks against both *closed-world* and *open-world* models [8, 49]. We define them using the relation between \mathbb{A} and \mathbb{V} . The case of $\mathbb{A} = \mathbb{V}$ corresponds to the closed-world model, and the case of $(\mathbb{A} \neq \mathbb{V}) \wedge (\mathbb{A} \cap \mathbb{V} \neq \emptyset)$ to the open-world model. These two models in essence represent the extent to which the attacker is able to train his classifier on the pages that are likely to be visited by the victim. The ideal scenario for the attacker is to know all pages the victim may visit, i.e., the closed-world model. This has been regarded unrealistic [33, 38] though. Nevertheless, the closed-world model is useful to benchmark classification and fingerprinting approaches. We thus keep both models in our evaluation.

2.2 CDN Bursts as Classification Features

Extracting a set of consistently informative features from the network traffic is the utmost important factor for a successful classification. In general, the size, temporal distribution and source/destination are the major characteristics of the packets. Therefore, we generalize a traffic generated while loading a social media page w as a sequence $T(w)$: $T(w) = \langle (p_1, t_1, ip_1), (p_2, t_2, ip_2), \dots, (p_N, t_N, ip_N) \rangle$, where (p_i, t_i, ip_i) is a packet originated from ip_i with timestamp t_i , and size p_i ($p_i > 0$). We call $T(w)$ a *loading trace* of web page w .

In the CDN traffic, the incoming packets are more informative than the outgoing ones (in terms of size variation, source, etc.). We therefore further reorganize the loading trace $T(w)$ to group the incoming CDN packets from ip into $T_{ip}^{in}(w)$:

$$T_{ip}^{in}(w) = \langle (p_1^{in}, t_1^{in}, ip), \dots, (p_M^{in}, t_M^{in}, ip) \rangle,$$

where $M \leq N$. Hereafter, we use *traces* to indicate incoming traces.

Given a CDN trace $T_{ip}^{in}(w)$, a CDN burst is a segment of the trace containing a sequence of consecutive and temporally adjacent packets. The k -th CDN burst is defined as B_k^{in} :

$$B_k^{in} = \langle (p_j^{in}, t_j^{in}, ip), (p_{j+1}^{in}, t_{j+1}^{in}, ip), \dots, (p_q^{in}, t_q^{in}, ip) \rangle,$$

where $0 < j \leq M$; $p_j^{in}, p_{j+1}^{in}, \dots, p_q^{in}$ are consecutive incoming packets; $t_{l+1} - t_l < \delta$ for all $l \in [j, q]$. The threshold δ which we name as *burst threshold* is a parameter whose optimal value can be fine-tuned (discussed in Section 5.3).

Based on the definitions above, we summarize the CDN burst feature engineering as Algorithm 1. It takes as input a CDN trace $T_{ip}^{in}(w)$ originated from IP address ip when loading page w , and outputs the CDN burst feature set. Intuitively, it first aggregates packets into bursts and then calculates the burst sizes which are the final features written into the feature set.

Algorithm 1: CDN Burst Based Feature Engineering

```

Input:  $T_{ip}^{in}(w)$ 
Output: Feature Set  $\mathbb{F}$ 
Function FeatureEngineering ( $T_{ip}^{in}(w)$ )
   $\mathbb{F} = \emptyset, k = 1, B_1^{in} = \langle \rangle$ 
  for  $i$  in range(1,  $M$ ) do
     $B_k^{in}.append((p_i, t_i, ip))$ 
    if  $t_{i+1}^{in} - t_i^{in} < \delta$  then
       $| B_k^{in}.append((p_{i+1}, t_{i+1}, ip))$ 
    else
       $| \mathbb{F}.append(total\ size\ of\ all\ packets\ in\ B_k^{in})$ 
       $| k \leftarrow k + 1$ 
    end
  end
  return  $\mathbb{F}$ 
end

```

The rationale for proposing CDN bursts is that the social media pages may be embedded with large-size objects (e.g., images and videos) which are typically hosted by CDN servers. Objects with various sizes may appear in different places among pages (based on the order that page owners upload them to the “timelines”, for example), such that the DOM structures of these pages may vary. This results in significant variation in the browser’s page downloading behaviors. When an object is being downloaded, it is split into multiple packets for transmission, and these packets are likely to appear in a batch. We observe that the time interval between such two consecutive packets is much shorter (0.1%) than that between the HTTPS requests to fetch two adjacent objects (~0.01ms v.s. ~10ms), because it takes time for the browser to parse and render the page before sending out more HTTPS requests. In this situation, the CDN bursts become distinguishable classification features since a burst is essentially a “reconstruction” of an application-level object.

The burst threshold δ is a subtle factor. In an ideal scenario where the browser sends sequential requests for the objects, any δ between 0.01ms and 10ms would enable the “object reconstruction”. However, considering that modern browsers send requests in parallel (e.g., the web worker in HTML5), it may be more robust to reconstruct a batch of objects into a burst. We take this strategy in our construction, and detail the selection of δ in Section 5.3.

2.3 CDN Bursts in Real-world Scenarios

To explore the feasibility of the intra-domain WPF using CDN bursts, we conduct an empirical study on two of the most followed Instagram profile pages, National Geographic (ranked 14th most followed, denoted as `Ins_page_1`) and Katy Perry (ranked 20th most followed, denoted as `Ins_page_2`). We simulate a scenario shown in Figure 2 where the victim user is browsing two pages `Ins_page_1` and `Ins_page_2`. The theme and script files which

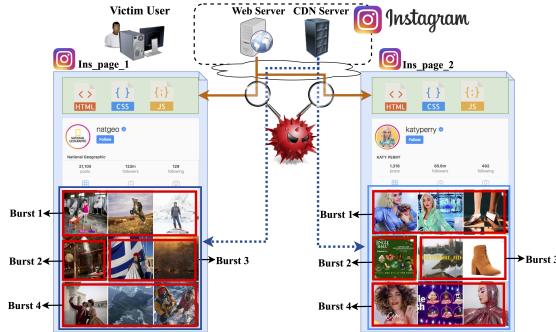


Figure 2: A running example

include the HTML, CSS and JavaScripts are fetched from the web server, and the large-size image/video contents are stored by the CDN sever (with IP address 157.240.25.63). We visit each Instagram profile for 50 times and record the generated network traffic.

First, we visualize the traces derived from the traffic in Figure 3 (a), without applying our feature engineering algorithm. We plot the cumulative incoming data size versus time to reflect the *temporal and volumetric features* commonly used by website fingerprinting techniques[9, 49, 50]. As is shown, the traces are almost indistinguishable, and significant parts are even intertwined. This situation, in practice, would be further worsened by the network fluctuations.

We also assess the effectiveness of the *cumulative volumetric features*, which are another extensively used type of features [17, 32]. We investigate the distribution of the cumulative transmitted data volume of 30,000 collected Instagram profile pages included in one of our datasets (further discussed in Section 3.2), and find that 95% of those pages have the cumulative transmitted data volume in range between 1.1 MB to 1.4 MB, as shown in Figure 3 (b). This suggests that the classification algorithms which heavily rely on the cumulative volumetric features might also be ineffective. Besides, considering the traces are from the same web server, other traditional statistical features [17, 49] which are stemmed from the variances among inter-domain web servers, such as maximum packet size and packet count per second, are likely to fail.

We further break down the traffic based on its origins. We extract the CDN traffic out of the overall network traffic (by filtering packets from IP address 157.240.25.63) and plot it in Figure 3 (c). As is shown, the CDN traffic exhibits consistent and distinctive differences in burst sizes between the two pages, despite indistinguishable cumulative CDN traffic sizes (~0.6 MB for both pages). We then apply our feature engineering on all CDN traces of these two pages. For the purpose of intuitive visualization and comparison, we plot the value ranges of the derived feature set in Figure 3 (d). As can be seen, each burst size varies within a relatively narrow band, suggesting the features are robust and consistent. There is almost no overlapping region between bursts (except a marginal one for burst 1), indicating that the CDN bursts are distinguishable representations, or *fingerprint*s, of their traces.

3 DATA COLLECTION

The prerequisite for machine learning algorithms is a dataset with abundant representative data. However, since our work is the first

one conducting the intra-domain WPF, there is not dataset that could be directly used. Therefore, we resort to build a new setup for data collection. During the process of data collection (presented soon in Section 3.1), we take account of attacker's capabilities (defined in Section 2.1) to retain representativeness. We assume the attacker can use a network sniffing tool to record metadata of the network-layer packets, including the size of the packets, the timestamp of sending/receiving a packet, the addresses (port and IP), etc. Our collected datasets have been made available online [2] after anonymization to facilitate future research.

3.1 Setup

Without losing generality, the machine configurations and locations are intentionally diversified in our data collection process. We distribute our setup along the campuses of National University of Singapore and the University of Queensland (denoted by A and B respectively for simplicity), located in two continents. On Campus A, we utilize 4 *OpenStack* [31] Ubuntu 16.04 virtual machines hosted by a server with 20 CPUs and 50GB of RAM in our department, and 2 Windows 10 desktops with 8 CPU Cores and 16GB of RAM located in a research lab. On Campus B, we utilize 3 Ubuntu 16.04 virtual machines with 1 CPU and 2 GB of RAM located in an office. For loading the web pages, we use Firefox version 66.0.3 on Linux machines and Chrome version 70.0.3538.77 on Windows machines.

3.2 Data Collection Methodology

Considering that social media pages contain many dynamically-loaded contents, our data collection process has to simulate the actual browsing activities of the users, rather than straightforwardly sending HTTP requests without rendering pages. To this end, we use a web testing automation tool *Selenium* [40] to automatically generate page visits. Similar to other fingerprinting studies ([17, 42, 50]), the browser caches are disabled so that the captured traces are complete; the page visits are in the sequential manner (i.e., *single-tab*) so that the collected traces are not interwoven. We acknowledge these two settings may give the attacker advantage, but in reality, the caches of the main-stream browsers are usually too small (< 80 MB) to host many large-size objects, and the dwell time on a social media page is relatively long that interwoven traces could be rarely generated.

We use the *tcpdump* [44] to record the network traffic. We capture the HAR (HTTP Archive) files to record the browser activities during web page loading. To automatically log the HAR files, we invoke browsers' built-in developer modes by setting the preference `devtools.netmonitor.har.enableAutoExportToFile` in Firefox and using the *automated chrome profiling* [35] library in Chrome. Note that the HAR files are only required in the training process to identify the IP addresses of the CDN servers (detailed in Section 3.4), but not required when conducting the actual attack.

We select the top four social media websites from the Alexa Top Sites for our evaluation, including Facebook, YouTube, Twitter and Instagram. For each site, up to 30,000 (to be detailed soon) most liked or followed user profiles listed on Trackalytics [46] are collected. On each profile page, the collection process has to let the browsers stay sufficiently long for the page to be fully loaded. This time is set to be 14 seconds since the average web page loading

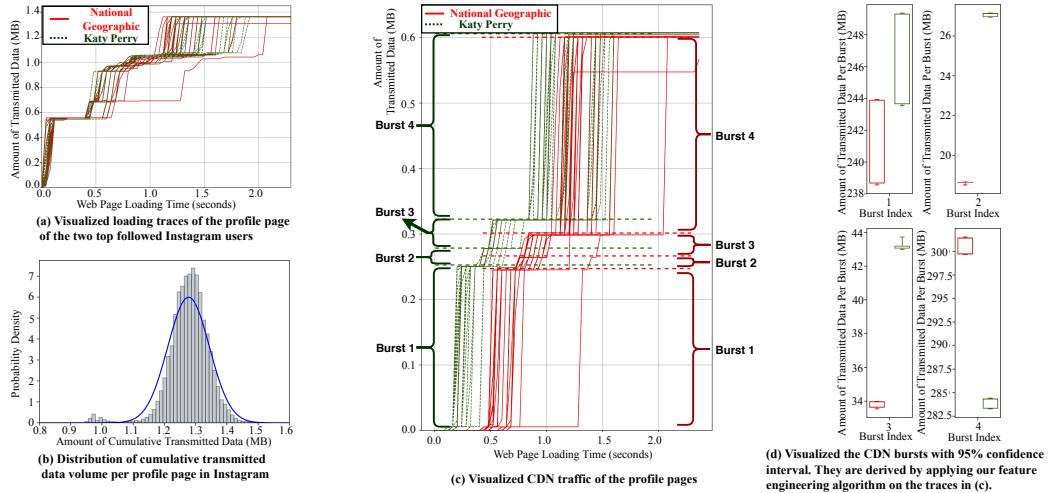


Figure 3: Visualization of network traffic, CDN traffic and CDN bursts of two example Instagram profile pages

time in most countries is around 8 to 10 seconds [6]. Between two pages, a break of at least 8 seconds is set to comply with the privacy policy of the websites and to avoid being blocked.

Closed-world Datasets. Datasets \mathbb{D}_1 and \mathbb{D}_2 are collected from Campus A and Campus B respectively. Dataset \mathbb{D}_1 is collected within two weeks. It contains the traces of the top 1,000 most liked/followed user profiles pages from each website. Due to its diversity (in terms of machines, software stacks and time span), we mainly use \mathbb{D}_1 to evaluate the performance of the intra-domain WPF in the closed-world model. Dataset \mathbb{D}_2 contains the traces of the top 300 pages from each site. It is collected within a relatively stable environment and within a relatively short time (mostly 2 days, but 1 week for Facebook due to its visit frequency constraint) so that it is “clean” for benchmarking our work. In both datasets, each page is recorded for 55 traces.

Open-world Datasets. For evaluation against open-world model, we have collected the dataset \mathbb{D}_3 from Campus A. \mathbb{D}_3 includes the traces of 300 *monitored* pages (denoted by \mathbb{M}) and up to 30,000 *unmonitored* pages (denoted by \mathbb{U}) for each site. Each page in \mathbb{M} is recorded for 55 traces. For each page in \mathbb{U} , three traces are recorded for redundancy. We cross validate these three traces by their sizes to remove the faulty ones, and keep only one trace for training and testing (detailed in Section 3.4).

3.3 Ethical Considerations

Our data collection generates overall 424K visits to the four social media websites. These visits are distributed over four weeks, and much fewer than their billions of daily visits. The impact to their servers is thus negligible. We only store the IP packets, in which all application-layer data are encrypted. Before releasing the datasets to facilitate future research, we anonymize any field that may reveal identities, including IP addresses, port numbers, labels, etc.

3.4 Data Pre-processing

Outlier Removal. Given the collected datasets, we first remove the faulty traces where no packets or only partial packets are received.

To further eliminate the possible noisy training data, we use a *interquartile range* (IQR) based approach [23] to remove the outliers, a standard way to detect noise used by literature [32]. Intuitively, the traces whose total transmitted data sizes deviate significantly from median value are considered as outliers and therefore are removed. For pages in \mathbb{D}_1 , \mathbb{D}_2 and the 300 monitored pages in \mathbb{D}_3 , we calculate the total transmitted data size for each trace in the training dataset, and eliminate those whose total size is out of the interquartile range $[Q_1 - 1.5IQR, Q_3 + 1.5IQR]$, where Q_1 and Q_3 represent first and third quartiles, and $IQR = Q_3 - Q_1$. In this way, an average of 5% traces in the training dataset are removed. Note that outlier removal is not applied on the testing traces. This ensures an unbiased evaluation of the classifier, and simulates a realistic scenario where the attacker has little control over the testing data.

In the remaining of this paper, when we refer to the three datasets, we actually mean the datasets after pre-processing.

CDN Traffic Identification. In reality, a CDN URL may be mapped to a list of IP addresses for load balancing. Therefore, we conservatively assume the attacker has to construct such a map before starting the attack. Because the number of IP addresses is quite small (at most 14 according to our study), it is feasible for the attacker to obtain a full list by gathering the URLs from profile pages and querying DNS servers multiple times prior to the attack. To simulate this, we query through parameters such as *requestId* and *ResourceReceiveResponse* in the collected HAR files, and find the URLs corresponding to image and video files. Then we resolve them into IP addresses by querying public DNS servers including Google DNS, OpenDNS Home, Alternate DNS and CleanBrowsing.

4 LEARNING ALGORITHM AND FEATURE INFORMATIVENESS

In this section, we discuss the web fingerprinting as a classification problem in the context of intra-domain WPF. We also examine the informativeness of our classification features derived from CDN bursts, inspired by [25] which urges for the quantification of the information leakage from the classification features.

4.1 Technique and Evaluation Metrics

Classification Technique. In our datasets, each web page in the monitored set (i.e., the \mathbb{A} in closed world and the \mathbb{M} in open world) is a class, and thus $|\mathbb{A}|$ or $|\mathbb{M}| + 1$ classes are included (one extra indicates the unmonitored class). To suit the multi-class classification, we select the random forests in our work. Random forests are a classification technique [7] based on the votes made by the majority of decision tree ensemble included in the forest. Random forests have been employed to classify the encrypted network traffic by recent studies on website fingerprinting [17, 54]. They require shorter training time and fewer training instances than traditional machine learning (such as SVM and KNN) [27, 32, 49, 50] and deep learning algorithms (such as autoencoder or CNN) [24, 25, 30, 42]. For example, fewer than 100 labeled training data for each class is sufficient for random forests, while CNN may require at least 500 each [53] such that it demands much longer collection period than the former. The efficiency of random forests is crucial for fingerprinting social media pages whose contents change so frequently that timely updating of the datasets and retraining of the classifiers are necessary. In addition, although random forests rely on manually crafted feature sets, the analysts would be able to interpret the features better (e.g., to analyze the feature-based information leakage and their relative importance) compared to those derived from deep neural networks.

Each tree in the random forests is trained from the fixed-sized labeled feature sets. Despite the bootstrap sampling process used for tree construction, cross validation is still necessary in an unbiased evaluation for our classifier. Cross validation is also important to determine a robust burst threshold δ . Therefore, we apply a 10-fold stratified cross validation. The training part (9 folds) is used to generate a classifier from the “fingerprint” of each monitored page. The testing part (1 fold) is used to validate whether a trace matches any of the fingerprints. We pad/truncate the feature vector to a fixed length of 50 which is larger than most vectors’ dimensions.

Classification Performance Measurement. To determine the classification performance, we adopt the following four metrics³ commonly used in the web fingerprinting literature [17, 39].

- **Accuracy** (*accuracy*, $\frac{TP+TN}{total}$) is the ratio of the correctly classified web pages to the total number of input web pages.
- **True positive rate** (*TPR*, $\frac{TP}{TP+FN}$) is the probability of correctly classifying the web pages.
- **False positive rate** (*FPR*, $\frac{FP}{FP+TN}$) is the probability of incorrectly classifying the web pages.
- **Bayesian detection rate** (*BDR*) is the probability that a web page is correctly identified by the classifier as a labelled monitored web page. For simplicity, we assume the visiting of web pages by users follow the uniform distribution. Using Axelsson’s definition [4], we formulate *BDR* as

$$BDR = \frac{TPR \cdot f_m}{TPR \cdot f_m + FPR \cdot f_u},$$

where f_m is the fraction of traces from monitored pages in the total web traces, and f_u is that for unmonitored pages.

³To facilitate the metrics definitions, we use the following denotations: TP (true positive), FP (false positive), TN (true negative) and FN (false negative).

Among the four metrics, *accuracy* is used for evaluating the classification performance in the closed-world setting, since the class sizes (a class size refers to the number of traces for each page) are balanced in \mathbb{D}_1 and \mathbb{D}_2 (i.e., each data class takes a fixed proportion of the total sampled data). In contrast, *TPR*, *FPR* and *BDR* are used in the open-world setting since the sizes of the monitored and unmonitored classes are significantly imbalanced. *TPR* and *FPR* check whether the correct classification rate and false alarm rate for monitored pages are balanced, while *BDR* measures the feasibility of the attack in reality by taking into consideration the fraction of the monitored pages in the sampled pages. The reason we use *BDR* as the evaluation metric in open-world setting is because a low *FPR* does not necessarily imply a high probability of successful attacks when the fraction of the monitored web pages are extremely low, also known as the *base rate fallacy paradox* [5].

4.2 Feature Information Leakage Measurement

Fundamentally, classifiers can distinguish among classes since the classification features contain some amount of information. These features, on the other hand, can leak information to the attacker. To quantify such leakage and also to figure out the feature importance, we formalize the information contained in each feature as the *mutual information* in a similar way as presented by Li et al. [25]. It quantifies the amount of information about a random variable (i.e., the identity of a web page) derived from another variable (i.e., the features extracted from the network traffic).

Definition 1. The amount of information about a web page contained in the fingerprint of the trace is $I(F; W)$:

$$I(F; W) = H(W) - H(W|F), \quad (1)$$

where F is a random variable corresponding to a single or a set of features in the fingerprint of a trace; W is a random variable about the web page identity; $I(F; W)$ denotes the mutual information between variables F and W ; $H(W)$ denotes the information entropy of variable W ; $H(W|F)$ denotes the conditional entropy of the variable W given the variable F .

Below we calculate $H(W)$ and $H(W|F)$. Assuming that the page the user is browsing is $w \in \mathbb{V}$, $H(W)$ can be calculated as follows.

$$\begin{aligned} H(W) &= - \sum_{w \in \mathbb{V}} Pr(w) \log_2 Pr(w) \\ &= - \frac{C|\mathbb{V}|}{C|\mathbb{A}| + |\mathbb{V} - \mathbb{A}|} \log_2 \frac{C}{C|\mathbb{A}| + |\mathbb{V} - \mathbb{A}|} \end{aligned} \quad (2)$$

where C is the number of traces for each monitored page; $Pr(w)$ is the probability that the identified page is w .

Given a variable F_j representing the j -th feature whose value range is denoted by \mathbb{F}_j , the conditional entropy of the variable W , i.e., $H(W|F_j)$ is, as follows.

$$H(W|F_j) = - \sum_{f_j \in \mathbb{F}} Pr(f_j) \sum_{w \in \mathbb{V}} Pr(w|f_j) \log_2 Pr(w|f_j), \quad (3)$$

where $Pr(f_j)$ is the probability that feature F_j has a value f_j , and $Pr(w|f_j)$ is the probability that the identified page is w given that the value of feature F_j is f_j .

According to Equation 3, the key for calculating the final information leakage is to derive the probability density function (PDF)

of feature F_j and the conditional probability $Pr(w|f_j)$. We estimate the former using a histogram-based approach commonly used for entropy estimation [34]. In particular, we partition the range \mathbb{F}_j of feature F_j into n equal intervals (l_j^i, u_j^i) where $i \in \{1, 2, \dots, n\}$. $Pr(f_j)$ is represented by the fraction of traces whose feature F_j evaluates to $f_j \in (l_j^i, u_j^i)$, against the total number of traces in the whole dataset. $Pr(w|f_j)$ is represented by the fraction of traces whose labels are w and feature F_j evaluates to $f_j \in (l_j^i, u_j^i)$, against the total number of traces whose feature F_j evaluates to $f_j \in (l_j^i, u_j^i)$ in the whole dataset. Therefore, we have

$$Pr(f_j) = \frac{K_j^w}{C|\mathbb{V}|} \text{ and } Pr(w|f_j) = \frac{N_j^w}{K_j^w}, \quad (4)$$

where K_j^w is the number of traces in \mathbb{D}_3 whose feature F_j evaluates to $f_j \in (l_j^i, u_j^i)$; N_j^w is the number of traces that are from web page w and feature F_j evaluates to $f_j \in (l_j^i, u_j^i)$.

All in all, the information leakage of the j -th feature can be quantified as follows.

$$\begin{aligned} I(F_j; W) &= H(W) - H(W|F_j) \\ &= \sum_{f_i \in \mathbb{F}} \frac{K_j^w}{C|\mathbb{V}|} \sum_{w_i \in \mathbb{V}} \frac{N_j^w}{K_j^w} \log_2 \frac{N_j^w}{K_j^w} \\ &\quad - \frac{C|\mathbb{V}|}{C|\mathbb{A}| + |\mathbb{V} - \mathbb{A}|} \log_2 \frac{C}{C|\mathbb{A}| + |\mathbb{V} - \mathbb{A}|} \end{aligned} \quad (5)$$

This is used to quantify information leakage by CDN bursts in Section 5.1.

5 EVALUATION

Our evaluation focuses on the performance of the intra-domain WPF and the effectiveness of CDN bursts as classification features. We explore the three main research questions.

- **RQ1.** Can CDN bursts be used as a distinguishable “fingerprint” for the intra-domain WPF?
- **RQ2.** Is intra-domain WPF feasible in reality (i.e., the open-world scenario) and to what extend can it scale?
- **RQ3.** What factors may affect the performance of CDN bursts in the intra-domain WPF?

5.1 RQ1: Intra-domain WPF in Closed-world

Experiment Setup. To answer RQ1, we choose accuracy (cf. Section 4.1) as the evaluation metric of the intra-domain WPF in the closed-world model. In the literature, the closed-world model is extensively used as a baseline to benchmark the fingerprinting algorithm [8, 17, 32, 38, 49]. To avoid the influence on accuracy due to the unbalanced class distribution in the datasets, we use our datasets \mathbb{D}_1 and \mathbb{D}_2 in the experiments. We apply 10-fold stratified cross validation on the datasets. According to our study on the burst threshold δ , which is detailed soon in Section 5.3, we set δ to be 0.05s when we apply our feature engineering algorithm (Algorithm 1) in all experiments.

Performance of Intra-domain WPF with Varying Number of Pages. We use varying numbers of web pages (i.e., $|\mathbb{A}|$ or $|\mathbb{V}|$) as in the closed-world model, $\mathbb{A} = \mathbb{V}$ ranging from 200 to 1,000 from \mathbb{D}_1 for the evaluation. As shown in Figure 4, we achieve accuracy

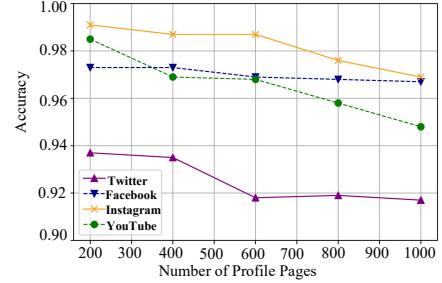


Figure 4: Intra-domain WPF with varying number of pages

Table 1: Comparison of fingerprinting approaches on \mathbb{D}_2

Attacks	Accuracy			
	Instagram	Facebook	Twitter	YouTube
k-fp	0.181	0.833	0.340	0.367
CUMUL	0.829	0.964	0.909	0.894
WPF	0.953	0.932	0.903	0.948

Table 2: Time and memory consumption

Attacks	Training Time (min) / Memory (GByte)			
	Instagram	Facebook	Twitter	YouTube
k-fp	6.2 / 5.5	5.0 / 5.4	5.0 / 5.6	4.5 / 4.7
CUMUL	> 3.5 hours / ~5.0 in each site			
WPF	1.5 / 3.7	3.0 / 4.9	2.5 / 4.7	2.0 / 4.1

The data is collected on a Surface Pro 6 with Intel i5 4-Core CPU@1.6GHz and 8G RAM. The training time includes the cross-validation time. The time consumed by feature engineering is not shown, as each approach takes similar time (~10min).

in the range of 0.92 to 0.99. It is also noticeable that the accuracy has a minor drop for Twitter. This may be because in Twitter, the CSS and JavaScript files are also located in the same CDN servers as images and videos, which may slightly disturb the CDN bursts.

Performance of Inter-domain Features for Intra-domain WPF.

As the inter-domain WPF has been well researched, we also explore the applicability of its techniques to the domain of intra-domain WPF. We apply two state-of-the-art inter-domain WPF approaches, k-fingerprinting (denoted by k-fp) proposed by Hayes et al. [17] and CUMUL proposed by Panchenko et al. [32], to our datasets. K-fp is based on random forests and CUMUL is on SVM. For k-fp, we adapt our data format for it and use its released code in our experiment; for CUMUL, due to incompatibility issues, we re-implement it by closely following the same feature extraction and classification techniques detailed in its paper. To avoid bias, we use \mathbb{D}_2 in our experiments due to its cleanliness (cf. Section 3.2). The results are listed in Table 1, and the computation time and memory consumption are listed in Table 2.

Although it may be unfair to directly compare approaches targeting different attacker models, our approach generally achieves higher accuracy and lower overhead (shorter training time and

Table 3: Information leakage by the features of CDN bursts (bit) measured in the closed-world model

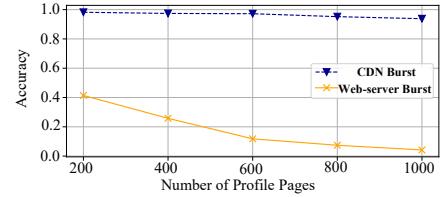
Burst Index	1	2	3	4	5	6	7	8
Info Leak	3.87	3.22	2.64	2.08	1.83	1.71	1.51	0.52

less memory consumption) than the other two techniques. Our approach slightly outperforms CUMUL in general, but requires much shorter training time and less resource. There are modest improvements in accuracy, ranging from 5 to 10 percentage points, compared with CUMUL in Instagram and YouTube. This improvement is likely because CDN bursts capture the subtle differences of the objects better than the cumulative features that CUMUL relies on. Our approach marginally underperforms CUMUL in accuracy by 0.1 to 3 percentage points in Facebook and Twitter, plausibly due to the fact that content personalization results in greater variations in the cumulative sizes of pages, which could be better captured by inter-domain fingerprinting techniques like CUMUL. The accuracy of k-fp is relatively low since the features used by it mostly focus on the packet counts rather than the packet lengths. For intra-domain web pages, the packet count features have lower distinguishing capacity since web pages generated from the same template and sourced from the same web server would exhibit similar packet-wise patterns.

Apart from the existing inter-domain WSF approaches, we also explore the capacity of deep learning in the intra-domain WPF using a larger dataset we have collected. For each website, we select the most liked or followed 100 pages and each page is recorded with 500 traces. We conduct our experiments with a recent deep learning based WSF approach named DF, which is proposed by Sirinam et al [42]. We have observed that the size of the training samples has a significant impact on the classification performance. In particular, when DF is evaluated using the same setting as [42], its accuracy (ranging from 0.5 to 0.85) is significantly lower than our approach (> 0.9); when evaluated using a larger dataset (> 300 traces per page), its accuracy becomes comparable with ours. This implies that our approach may be more realistic in practice than the deep learning based counterparts which require extended data collection period and higher computation capacity.

Information Leakage by CDN Bursts. We use dataset \mathbb{D}_2 to quantitatively evaluate the information included in CDN bursts which is utilized in classification (cf. Equation 5). We measure the average information leakage for both individual CDN bursts and the cumulative CDN burst. The results of the most informative 8 bursts are shown in Table 3. The CDN bursts are indexed according to their sizes. Larger bursts are observed to leak more information (up to 3.87 bits) than the smaller ones (down to 0.5 bit).

We note that the information leakage is calculated on the basis of each feature as an indication of its effectiveness. There may be correlation but not causation between the information leakage of individual features and the final classification accuracy since the classifier also utilizes additional information pertaining to groups of features. This type of information leakage will be investigated as our future work.

**Figure 5: Performance comparison of web-server bursts and CDN bursts for intra-domain WPF in Instagram pages.****Table 4: Training and Testing Datasets**

Training Set	9 folds of the traces in \mathbb{M}_{tr} and a set of randomly selected traces $\mathbb{U}_{tr} \subset \mathbb{U}$
Testing Set	The remaining 1 fold the traces in \mathbb{M}_{tr} and the remaining traces in the set $(\mathbb{U} - \mathbb{U}_{tr})$

Intra-domain WPF Using Web-server Bursts. We also apply our feature engineering algorithm to web server traffic to explore whether the web-server bursts can be used as fingerprints. We measure the classification accuracy for Instagram traces in our dataset \mathbb{D}_1 as an illustration, as shown in Figure 5. It is observed that the accuracy using web-server bursts is significantly lower than that using CDN bursts. On the other hand, the accuracy is not extremely low due to the marginal differences among web pages. However, this difference becomes trivial as the number of pages increases, and thus the accuracy declines at a faster pace (23.3% from 200 pages to 1,000 pages) than that based on CDN bursts (2.8% from 200 pages to 1,000 pages). This indicates the web-server bursts may not be robust features.

5.2 RQ2: Intra-domain WPF in Open-world

Experiment Setup. To answer RQ2, we conduct the intra-domain WPF in the more realistic open-world model. Due to significant size imbalance in the monitored (i.e., \mathbb{M}) and unmonitored set (i.e., \mathbb{U}), we use TPR, FPR and BDR as our evaluation metrics. TPR and FPR determine whether the detection rate and false alarm rate are balanced, i.e., the classification *correctness*. For *feasibility* evaluation, we use the metric BDR, which takes into consideration not only TPR and FPR, but also the ratio of monitored and unmonitored set size, such that the rates on the unmonitored set (much larger than monitored set) would not dominate the metrics.

We use the open-world dataset \mathbb{D}_3 in our experiments. Recall that \mathbb{D}_3 includes $55 \text{ traces/page} \times 300 \text{ monitored pages}$ (i.e., \mathbb{M}), and one trace/page \times up to 30,000 unmonitored pages (i.e., \mathbb{U}) for each of the four sites. Two variables are investigated for their influence on the performance of the intra-domain WPF using 10-fold stratified cross validation. For each round, we construct the training and testing datasets as is shown in Table 4. We first select a subset \mathbb{M}_{tr} from \mathbb{M} , and use nine folds of the traces from each page for training and the remaining one fold for testing. Then, we select a subset \mathbb{U}_{tr} from \mathbb{U} for training and the remaining traces for testing. In our setting, \mathbb{M}_{tr} is to simulate attacker's target set; \mathbb{U}_{tr} is to simulate that the attacker includes extra web pages outside \mathbb{M} into

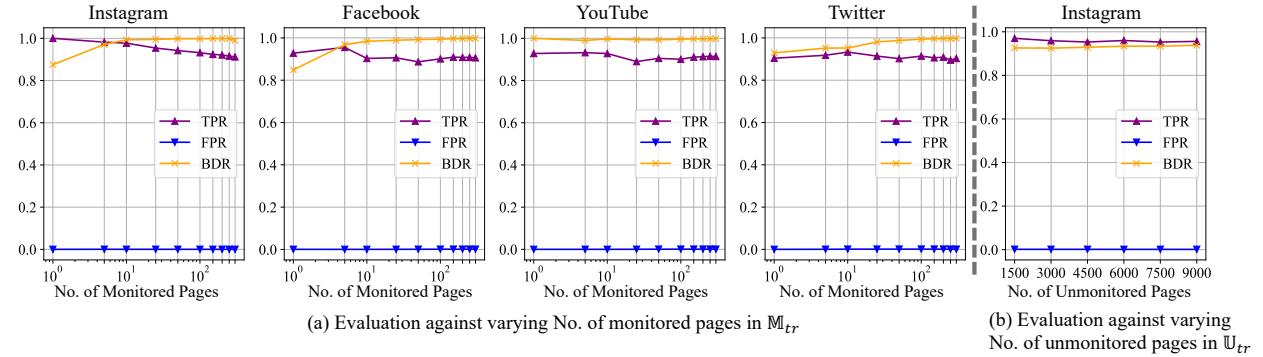


Figure 6: Performance of intra-domain WPF in the open-world model

his training dataset, indicating the attacker’s capability of profiling some web pages outside his targeted page set; U_{te} is to simulate the background noise when the user is browsing the intra-domain web pages but outside the attacker’s training dataset. We analyze the feasibility of intra-domain WPF by varying the size of M_{tr} and U_{tr} separately in this section.

Performance of Intra-domain WPF with Varying Number of Monitored Pages in Training Set. In this experiment, the size of M_{tr} varies while we maintain 1,000 pages in U_{tr} and 6,000 pages in U_{te} . We let the size of M_{tr} vary from 1 to 300, and determine the TPR, FPR and BDR. Our results are in Figure 6 (a). It is observed that intra-domain WPF achieves a stable performance against an increasing number of monitored pages in the training set. TPR exhibits a slight downward trend which stabilizes at around 0.9, while FPR shows a slight upward trend which stabilizes at around 0.001. The BDR quickly converges to nearly 1.0 at 50 monitored pages from around 0.9 at 1 monitored page.

Performance of Intra-domain WPF with Varying Number of Unmonitored Pages in Training Set. In this experiment, the size of U_{tr} varies while we maintain unchanged 30 pages in M_{tr} and 20,000 pages in U_{te} . We use Instagram, with 29,000 collected unmonitored pages, as a case study to evaluate the influence of the size of U_{tr} . Intuitively, the more unmonitored pages the attacker includes in the training set, the higher precision and fewer mistakes he would make in picking out the monitored pages. The reason we choose a monitored set of 30 pages against an unmonitored set of 20,000 pages is to simulate a realistic scenario in which the monitored pages take up merely a tiny portion (~2%) of the total pages possibly visited by the victim. We include varying sizes of U_{tr} from 1,500 to 9,000, and determine the TPR, FPR and BDR, as shown in Figure 6 (b).

It is observed that our approach achieves a consistent performance against an increasing number of unmonitored pages in the training set, with marginally decreasing TPRs at around 0.96 (from 0.97 down to 0.96), decreasing FPRs at around 0.0015 (from 0.0017 down to 0.0014) and increasing BDR at around 0.93 (from 0.92 up to 0.94). The consistently high BDR indicates the feasibility and high successful rate of intra-domain WPF to identify a small amount of pages from a significantly larger set. This result shows that intra-domain WPF is feasible without requiring much knowledge of pages outside attacker’s target set.

5.3 RQ3: Influencing Factors of CDN Bursts

In this section, we study the influence of the burst threshold δ and network latency on fingerprinting performance.

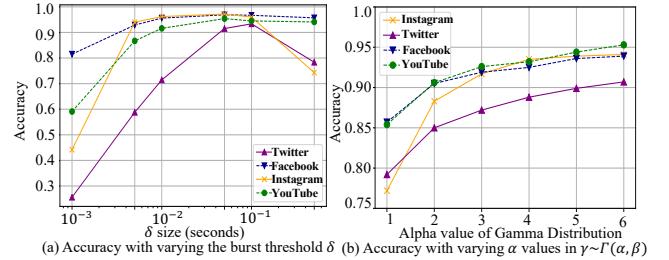


Figure 7: Evaluation on Influencing Factors

Burst Threshold δ . According to the feature engineering method introduced in Section 2.2, we group consecutive packets into a CDN burst when the inter-packet time gap is less than a value referred to as the burst threshold δ . The choice of δ would impact the total number of CDN bursts and the size of each burst. To determine a suitable δ for the real-world application of the intra-domain WPF, we measure the classification accuracy against various δ values ranging across magnitudes (from 1ms to 500ms) using dataset D_1 . When calculating the accuracy, we also apply 10-fold stratified cross validation to ensure the robustness and generalization in the results and to eliminate impacts of random noise and errors. The result is shown in Figure 7 (a). It is observed that the accuracy peak when δ is around 50ms to 100ms. The accuracy is significantly lower when δ value is very small (around 1ms) since many small and varying-sized CDN bursts are derived, resulting in inconsistent feature set. This confirms our estimation made in Section 2.2 that due to the concurrent requests and network latency, it is more robust to reconstruct a batch of objects, rather than an individual object, into a burst. Nonetheless, the accuracy starts dropping when δ increases beyond the optimal value. This is because multiple original bursts are grouped into fewer CDN bursts, reducing the amount of information.

Variation in Network Latency. The variation in the network latency, possibly due to unstable network condition, could lead to the fluctuation on the Packet Interval Time (PIT). This may “break”

the CDN bursts since the burst threshold δ is a temporal parameter. To investigate this influence, we construct a simulated dataset based on \mathbb{D}_2 by introducing extra latency to the timestamps of the original traces. The added latency γ follows a Gamma distribution ($\gamma \sim \Gamma(\alpha, \beta)$) with a shape parameter α and a rate parameter β which is commonly used for modeling the network latency [21]. We adopt 6 Gamma distributions with different spread levels (i.e., α). The higher the α , the less spread out the gamma distribution will be.) and a fixed mean value (i.e., α/β) of 48ms which is the average latency in the major CDN service providers [11]. We construct the CDN burst feature set from the simulated dataset and measure the classification performance using the same approach in Section 5.1. Our results are shown in Figure 7 (b).

As could be expected, the accuracy of intra-domain WPF increases as the latency becomes more concentrated. When the network latency distribution is concentrated within a small range (indicating that the network condition is relatively stable), there are only tiny shifts induced in the PITs. These shifts can be tolerated by a sufficiently large burst threshold δ , causing less loss to the accuracy. In contrast, when the latency spreads out, a larger latency can break through the burst threshold, generating “noisy” bursts.

6 LIMITATIONS AND COUNTERMEASURES

In this section, we discuss the limitations of our intra-domain WPF. We also propose and evaluate countermeasures against such attacks.

6.1 Limitations of Intra-domain WPF

Dependency on IP addresses. Our intra-domain WPF relies on the server IP addresses to extract the CDN packets out of the network traffic. This becomes infeasible when the network communication is protected by anonymizers such as VPN and Tor, in which only the proxy or the entry node is trackable. The anonymizers mix up CDN packets and web server packets, such that the boundary to group packets into bursts is blurred out. As a result, the distinguishability of the bursts may drop, as shown by our experiments on Instagram (Figure 4), web server bursts (Figure 5) and Tor browser (Section 6.2). However, the intra-domain WPF remains a prominent threat since only a small proportion (~7%) of Internet users frequently use VPNs and merely one third of them use it for social websites [15], and the Tor population is much smaller.

Content Drift in Social Media Websites. Website fingerprinting has been criticized for being impractical due to the changes in the web page contents over time, also known as the content drift [20]. Web pages in social media websites get updated particularly frequently, with millions of people posting on their social media daily [55]. The updated content in a page would result in the shift in the distribution of the objects, potentially causing changes in the patterns of CDN bursts and undermining the trained classifier. This requires updating the dataset and retraining the model timely to maintain a high accuracy. In this regard, our approach endures less impact from content drift than those requiring large numbers of training instances (i.e., deep learning based approaches).

6.2 Possible Countermeasures

Given that the intra-domain WPF utilizes the temporal and volumetric features of the CDN traffic, we propose defenses to mitigate

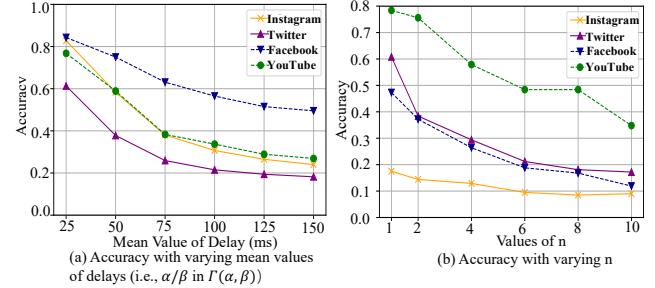


Figure 8: Simulated Evaluation on Defenses

these aspects in the traffic. To investigate their effectiveness, we use a small (for efficiency of experiments) dataset (denoted by \mathbb{D}_4) that includes 50 traces of 50 randomly selected Instagram profile pages from \mathbb{D}_2 . The experiments conducted in this section are all on \mathbb{D}_4 , unless stated otherwise.

Deviating Packet Interval Time in CDN Servers. One countermeasure is to break the expected packet arrival periodicity by deviating the so-called Inter Packet Arrival Time (IPAT). This obscures the boundary among the bursts such that the feature engineering algorithm may fail to produce robust features. To do this, the CDN server should add a marginal and randomized delay before it sends out a packet. This countermeasure is inspired by our finding from the experiment on the network latency in Section 5.3 that a spread-out network latency can significantly lower the accuracy of the intra-domain WPF. To explore its effectiveness, we simulate it by adding a random value that follows a Gamma distribution ($\Gamma(\alpha, \beta)$) to the arrival timestamp of each packets in our dataset. We adopt 6 Gamma distributions with a fixed spread level at $\alpha=0.25$ and varying average delays α/β ranging from 25ms to 150ms.

As shown in Figure 8 (a), the intra-domain WPF accuracy drops significantly with increasing mean values of random delays, given a fixed spread level. Intuitively, the patterns of CDN burst would be disrupted to a greater extent due to a larger random delay, indicating a better defense performance. In general, this defense would not incur an intolerable delay, compared with the average network latency (50ms, cf., Section 5.3), and it does not increase bandwidth overhead. Nonetheless, this countermeasure requires involvement of the CDN service providers due to the server-side changes.

Loading Dummy Packets from Web Browsers. This countermeasure forces the browser to send dummy requests to the same CDN server in parallel with the genuine requests, creating dummy CDN traffic to reshape the original CDN burst patterns. To simulate this countermeasure, we insert dummy packets into the traces in \mathbb{D}_4 . The inserted packets are extracted from Instagram pages which are not included in \mathbb{D}_4 . We parameterize the portion of packets to be inserted as $\frac{n}{10}$ ($0 \leq n \leq 10$) of the original traces. Intuitively, n dummy packets are inserted into every 10 packets.

Our results are shown in Figure 8 (b). It suggests that this countermeasure can perform well even with a small proportion of dummy packets. For example, the attack accuracy for Instagram drops to 0.175, down from 0.97, with only 10% of dummy packets. For Twitter and Facebook, the attack accuracy drops to below 40% with only 20% of the inserted dummy packets. The defense performs better

with even larger proportion of dummy packets. This countermeasure introduces limited bandwidth overhead and does not incur much latency. It requires no changes to the CDN server since it is deployed at browser side.

Anonymity Networks. Another possible defense is to browse the web pages through Tor network which conceals the IP information and incurs random latency due to the different routes selected. Since only entry node IP address is available, the bursts can be extracted only from the overall network trace. To explore the effectiveness, we constructed a small test dataset \mathbb{D}_5 including 50 traces for each of the 50 Instagram profiles loaded through Tor browser. We compare the classification accuracy on \mathbb{D}_5 with that on \mathbb{D}_4 , and find the accuracy drops to 0.292, compared to the that of 0.97 on \mathbb{D}_4 .

7 RELATED WORK

The intra-domain WPF is intrinsically a web fingerprinting technique. The related work in this area has been focusing on inter-domain WSF. This section summarizes both attacks and defenses.

7.1 Inter-domain Website Fingerprinting

The inter-domain website fingerprinting was firstly demonstrated feasible by a series of early studies [10, 18]. They have used the total volume of the data flow to differentiate encrypted network traffic. More recently, the inter-domain WSF techniques [9, 17, 19, 20, 25, 33, 38, 50, 51, 54] are still mainly based on the encrypted traffic characteristics including traffic patterns and traffic statistics, but they are typically evaluated under more stringent criteria. For instance, the performance is analyzed against both traditional encrypted channels and anonymous channels. The datasets used for evaluation are also reasonably large, containing at least thousands of unique websites.

Cai et al. [9] proposed an attack on Tor based on the optimal string alignment distance (OSAD) feature, achieving 80% accuracy in 100 websites. Wang et al. [49, 50] further improved the performance of inter-domain WSF attack on Tor and achieved accuracy over 90%. Wang et al. [51] explored the feasibility of practically deploying the inter-domain WSF system. Recently, Wang [48] proposed a new metric for evaluating the feasibility to realistically detect the sensitive web pages, and used three optimizers to boost the classifier performance. Gu et al. [16] evaluated the performance of inter-domain WSF attacks under multi-tab browsing setting. Hayes et al [17] proposed a novel inter-domain WSF technique based on a variant of random forest, which achieved a TPR of 85% and an FPR of 0.02% when monitoring 30 web pages out of over 100,000 unique web pages. Panchenko et al. [32] presented an approach based on an SVM classifier, utilizing the cumulative behavior representation of the web page loading trace.

In the most recent contributions, Deep Neural Networks (DNN) have been more adopted besides traditional machine learning algorithms. Rimmer et al. [38] proposed a CNN-based inter-domain WSF that automates feature engineering. The attack trained on 2,500 traces per site and achieved 96.3% accuracy. Oh et al. [30] utilized unsupervised DNN to extract low dimension informative features and achieved an accuracy of 94%. Sirinam et al. [42] presented a CNN-based classifier which achieved high accuracy on both undefended and defended Tor traffic. Sirinam et al. [43] further

proposed an attack based on N-shot learning which enables feature extraction using a pre-trained classifier. Their approach improves the attack robustness and shorten the classifier preparation time.

Compared to inter-domain WSF, intra-domain WPF attack focuses on identifying the exact web page browsed by the targeted user among the other similar web pages within the same web domain, utilizing a different set of web page-specific and volumetric-based features. To the best of our knowledge, intra-domain WPF in social media websites is an under-researched area. Miller et al. [28] utilized similar packet burst features extracted from the traces of a sequence of linked web pages within a website. Compared to our approach, this work requires constructing an interconnected structure of the targeted web pages accessible within a website. This restricts its applicability under our problem setting since web pages are generally independent of each other in social media sites. Schuster et al. [39] demonstrated the feasibility of identifying different videos based on the network traffic. They proposed a CNN-based attack which obtains an accuracy of near 99% among over 3000 YouTube videos. Shen et al. [41] proposed an attack to distinguish similar web pages using the cumulative packet length feature. Compared to our work, this work assumes an active attack model that is able to intercept network traffic between victim user and web servers.

7.2 Inter-domain Fingerprinting Defenses

A number of defenses have been proposed to counteract the fingerprinting. These mechanisms can be categorized into *traffic flow level* and *packet level*, and both aim to produce indistinguishable traffic. Panchenko et al. [33] presented a countermeasure based on generating background noise through loading a web page at random. Dyer et al. [13] proposed Buffered Fixed Length Obfuscation (BuFLO) to send packets with a fixed size at constant traffic rate. Cai et al. [9] proposed an improved version of BuFLO (CS-BuFLO), by reducing the bandwidth and time overhead and adapting for congestion. Furthermore, Cai et al. [8] presented a new defense Tamaraw with a higher performance compared to CS-BuFLO. Nithyanand et al. [29] designed the defense named Glove that groups web pages with similar traces into clusters and adds covert traffic, such that the web pages are indistinguishable within their own cluster.

There are also a few defenses in the application level. Tor introduced randomized pipelining [36] that shuffles pipeline size (i.e., number of paralleled requests) as well as the request orders for web objects. Luo et al. [26] constructed a defense called HTTPOS which is essentially a proxy that modifies the TCP and HTTP requests before sending them out.

8 CONCLUSION

In this paper, we propose the novel intra-domain WPF, which aims to determine which web page a user has just browsed. The proposed fingerprinting is based on a specially engineered feature set named CDN bursts. We show that the intra-domain WPF is feasible in social media websites, by investigating the informativeness of the classification features derived from CDN bursts. We comprehensively evaluate the intra-domain WPF with datasets collected from four top social media websites, which contain traces of over 10k unique pages and 400k page. Our evaluation demonstrate its high fingerprinting accuracy (an average accuracy of up to 96% and a

false positive rate as low as 0.02%) and computational efficiency (< 5mins training time). The datasets have been released to facilitate research in this area. To the best of our knowledge, this work is the first intra-domain WPF that is based on the patterns in CDN traffic. Our work should raise an alert on this previously neglected threat. Furthermore, we hope it will inspire more future research on the general intra-domain WPF.

ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for improving this manuscript. This research has been partially supported by the following funds. The National Research Foundation, Prime Ministers Office, Singapore, under its Corporate Laboratory@University Scheme, National University of Singapore and under its National Cybersecurity R&D Program, Singapore Telecommunications Ltd., the National Research Foundation Singapore under its NSoE Programme (award no. NSOE-TSS2019-05), the University of Queensland under the NSRSG grant 4018264-617225, and Oracle Labs Australia under the CR grant 4018264-024423.

REFERENCES

- [1] Visited in Feb 2021. Internet Live Stats. <https://www.internetlivestats.com>.
- [2] Visited in Feb 2021. Intra-domain webpage fingerprinting dataset. <https://github.com/ResearchWPF/Intra-domain-WPF.git>.
- [3] Salman Aslam. Visited in Oct 2020. Facebook by the Numbers: Stats, Demographics and Fun Facts. <https://www.omnicoreagency.com/facebook-statistics>.
- [4] Stefan Axelsson. 2000. The Base-rate Fallacy and the Difficulty of Intrusion Detection. *ACM Transactions on Information and System Security* (2000).
- [5] Maya Bar-Hillel. 1980. The base-rate fallacy in probability judgments. *Acta Psychologica* (1980), 211–233.
- [6] MachMetrics Speed Blog. Visited in Feb 2021. Average Page Load Times for 2018—How does yours compare? <https://www.machmetrics.com/speed-blog/average-page-load-times-websites-2018/>.
- [7] Leo Breiman. 2001. Random Forests. *Machine Learning* (2001), 5–32.
- [8] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. 2014. A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses. In *CCS*. 227–238.
- [9] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. 2012. Touching from a Distance: Website Fingerprinting Attacks and Defenses. In *CCS*. 605–616.
- [10] Heyning Cheng and Ron Avnur. 1998. Traffic Analysis of SSL Encrypted Web Browsing.
- [11] CloudMatch. Visited in Feb 2021. CDN Network Test. <https://cloudharmony.com/speedtest-downlink-uplink-latency-dns-for-cdn>.
- [12] Alan Davis. Visited in Oct 2020. How Social Media Spurred Myanmar’s Latest Violence. <https://iprnet/global-voices/how-social-media-spurred-myanmars-latest>.
- [13] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. 2012. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *IEEE S&P*. 332–346.
- [14] David Garcia. 2017. Leaking privacy and shadow profiles in online social networks. *Science Advances* (2017).
- [15] Ran Greenberg. Visited in Feb 2021. VPN Use and Data Privacy Stats for 2021. <https://www.vpnmentor.com/blog/vpn-use-data-privacy-stats/>.
- [16] Xiaodan Gu, Ming Yang, and Junzhou Luo. 2015. A novel Website Fingerprinting attack against multi-tab browsing behavior. In *IEEE Computer Supported Cooperative Work in Design*. 234–239.
- [17] Jamie Hayes and George Danezis. 2016. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *USENIX Security*. 1187–1203.
- [18] Andrew Hintz. 2003. Fingerprinting Websites Using Traffic Analysis. In *PETS*. 171–178.
- [19] Rob Jansen, Matthew Traudt, and Nicholas Hopper. 2018. Privacy-Preserving Dynamic Learning of Tor Network Traffic. In *CCS*. 1944–1961.
- [20] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. 2014. A Critical Evaluation of Website Fingerprinting Attacks. In *CCS*. 263–274.
- [21] Mehmet Karakas. 2003. *Determination of network delay distribution over the internet*. Ph.D. Dissertation. Citeseer.
- [22] Jonathan L. Zittrain, Robert Faris, Helmi Noman, Justin Clark, Casey Tilton, and Ryan Morrison-Westphal. 2017. The Shifting Landscape of Global Internet Censorship. *SSRN Electronic Journal* (01 2017).
- [23] Eric Langford. 2006. Quartiles in Elementary Statistics. *Journal of Statistics Education* 14, 3 (2006).
- [24] Qian Li, Yong Qi, Qingyuan Hu, Saiyu Qi, Yun Lin, and Jin Song Dong. 2020. Adversarial Adaptive Neighborhood with Feature Importance-Aware Convex Interpolation. *IEEE TIFS* (2020).
- [25] Shuai Li, Huajun Guo, and Nicholas Hopper. 2018. Measuring Information Leakage in Website Fingerprinting Attacks and Defenses. In *CCS*. 1977–1992.
- [26] Xiapu Luo, Peng Zhou, Edmond W. W. Chan, Wenke Lee, Rocky K. C. Chang, and Roberto Perdisci. 2011. HTTPS: Sealing information leaks with browser-side obfuscation of encrypted flows. In *NDSS*.
- [27] Siqi Ma, Ferdinand Thung, David Lo, Cong Sun, and Robert H Deng. 2017. Vurle: Automatic vulnerability detection and repair by learning from examples. In *European Symposium on Research in Computer Security*. Springer, 229–246.
- [28] Brad Miller, Ling Huang, A. D. Joseph, and J. D. Tygar. 2014. I Know Why You Went to the Clinic: Risks and Realization of HTTPS Traffic Analysis. In *PETS*. 143–163.
- [29] Rishab Nithyanand, Xiang Cai, and Rob Johnson. 2014. Glove: A Bespoke Website Fingerprinting Defense. In *WPES*. 131–134.
- [30] Se Eun Oh, Saikrishna Sunkam, and Nicholas Hopper. 2019. p1-FP: Extraction, Classification, and Prediction of Website Fingerprints with Deep Learning. *PoPETs* (2019), 191–209.
- [31] OpenStack. Visited in Feb 2021. <https://www.openstack.org>.
- [32] Andriy Panchenko, Fabian Lanze, Andreas Zinnen, Martin Henze, Jan Pennekamp, Klaus Wehrle, and Thomas Engel. 2016. Website Fingerprinting at Internet Scale. In *NDSS*.
- [33] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. 2011. Website Fingerprinting in Onion Routing Based Anonymization Networks. In *WPES*. 103–114.
- [34] Liam Paninski. 2003. Estimation of Entropy and Mutual Information. *Neural Computation* (2003), 1191–1253.
- [35] Paulirish. Visited in Feb 2021. Automated Chrome Profiling. <https://github.com/paulirish/automated-chrome-profiling>.
- [36] Mike Perry. Visited in Feb 2021. Experimental Defense for Website Traffic Fingerprinting. <https://blog.torproject.org/experimental-defense-website-traffic-fingerprinting>.
- [37] The Tor Project. Visited in Feb 2021. Tor Metrics. <https://metrics.torproject.org/bandwidth.html>.
- [38] Vera Rimmer, Davy Preuveneers, Marc Juarez, Tom van Goethem, and Wouter Joosen. 2018. Automated Website Fingerprinting through Deep Learning. In *NDSS*.
- [39] Roei Schuster, Vitaly Shmatikov, and Eran Tromer. 2017. Beauty and Burst: Remote Identification of Encrypted Video Streams. In *USENIX Security*. 1357–1374.
- [40] Selenium. Visited in Feb 2021. <http://www.seleniumhq.org>.
- [41] Meng Shen, Yiting Liu, Siqi Chen, Liehuang Zhu, and Yuchao Zhang. 2019. Webpage Fingerprinting using Only Packet Length Information. In *ICC*. 1–6.
- [42] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. 2018. Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning. In *CCS*. 1928–1943.
- [43] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew Wright. 2019. Triplet Fingerprinting: More Practical and Portable Website Fingerprinting with N-Shot Learning. In *CCS*. 1131–1148.
- [44] TCPdump. Visited in Feb 2021. <https://www.tcpdump.org>.
- [45] New York Times. Visited in Oct 2020. Sri Lanka Blocks Social Media, Fearing More Violence. <https://www.nytimes.com/2019/04/21/world/asia/sri-lanka-social-media.html>.
- [46] Trackalytics. Visited in Feb 2021. FREE SOCIAL MEDIA AND WEBSITE STATISTICS/ANALYTICS. <https://www.trackalytics.com/>.
- [47] Jane Wakefield. Visited in Oct 2020. Christchurch shootings: Social media races to stop attack footage. <https://www.bbc.com/news/technology-47583393>.
- [48] Tao Wang. 2020. High Precision Open-World Website Fingerprinting. In *IEEE S&P*. 231–246.
- [49] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. 2014. Effective Attacks and Provable Defenses for Website Fingerprinting. In *USENIX Security*. 143–157.
- [50] Tao Wang and Ian Goldberg. 2013. Improved Website Fingerprinting on Tor. In *WEPS*. 201–212.
- [51] Tao Wang and Ian Goldberg. 2016. On Realistically Attacking Tor with Website Fingerprinting. *PoPETs* (2016), 21–36.
- [52] WatchGuard. Visited in Feb 2021. Internet Security Report. <https://www.watchguard.com/wgrd-resource-center/security-report-q3-2018>.
- [53] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. 2018. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging* (2018), 611–629.
- [54] Junhua Yan and Jasleen Kaur. 2018. Feature Selection for Website Fingerprinting. *PoPETs* (2018), 200–219.
- [55] Jacqueline Zote. Visited in Feb 2021. Sprout Social. <https://sproutsocial.com/insights/social-media-statistics/>.