Website Fingerprinting at Internet Scale

Andriy Panchenko*, Fabian Lanze*, Andreas Zinnen[†], Martin Henze[‡],
Jan Pennekamp*, Klaus Wehrle[‡], and Thomas Engel*
*University of Luxembourg (LU), [†]RheinMain University of Applied Sciences (DE), [‡]RWTH Aachen University (DE)
E-mail: *{firstname.lastname}@uni.lu, [†]andreas.zinnen@hs-rm.de, [‡]{lastname}@comsys.rwth-aachen.de

***Abstract*—The website fingerprinting attack aims to identify the content (i.e., a webpage accessed by a client) of encrypted and anonymized connections by observing patterns of data flows such as packet size and direction. This attack can be performed by a local passive eavesdropper – one of the weakest adversaries in the attacker model of anonymization networks such as Tor.**

In this paper, we present a novel website fingerprinting attack. Based on a simple and comprehensible idea, our approach outperforms all state-of-the-art methods in terms of classification accuracy while being computationally dramatically more efficient. In order to evaluate the severity of the website fingerprinting attack in reality, we collected the most representative dataset that has ever been built, where we avoid simplified assumptions made in the related work regarding selection and type of webpages and the size of the universe. Using this data, we explore the practical limits of website fingerprinting at Internet scale. Although our novel approach is by orders of magnitude computationally more efficient and superior in terms of detection accuracy, for the first time we show that no existing method – including our own – scales when applied in realistic settings. With our analysis, we explore neglected aspects of the attack and investigate the realistic probability of success for different strategies a real-world adversary may follow.

I. Introduction

Anonymous communication on the Internet is about hiding the relationship between communicating parties. For many people, in particular for those living in oppressive regimes, the use of anonymization techniques is the only way to exercise their right to freedom of expression and to freely access information, without fearing the consequences. Besides, these techniques are often used to bypass country-level censorship. Hence, the users of anonymization techniques strongly rely on the underlying protection as defined in their attacker model. For them, it is particularly important to know the level of protection provided against considered adversaries. Several methods for low-latency anonymous communication had been proposed by the research community but only a few systems have been actually deployed. The Tor network [8] – the most popular system nowadays that is used by millions of daily users – promises to hide the relationship between the sender of a message and its destination from a *local observer*. This is the entity that eavesdrops traffic between the sender and the first anonymization node. It can be, for example, a local system administrator, an ISP, or everyone in the sending range of a signal if the user is connected via a wireless link. An entity with such capabilities is one of the weakest adversaries in the attacker model of this and other anonymization techniques [8].

The website fingerprinting (WFP) attack is a special case of *traffic analysis*. Performed by a local eavesdropper, it aims to infer information about the content (i.e., the website visited) of encrypted and anonymized connections by observing patterns of data flows. Here, the attacker merely utilizes meta information, such as packet size and direction of traffic, without breaking the encryption. Before the end of 2011, Tor was considered to be secure against this threat [11], [21]. Since then, it has become an active field of research. Several related works showed the feasibility of the WFP attack in Tor, however, using relatively small datasets (compared to the size of the world wide web) and proposed voluminous countermeasures. A recent work [14] questions the practical realizability of the attack in the light of assumptions typically made and the impact of the base-rate fallacy on the classification results.

In this paper, we propose a novel WFP attack based on a subtle method to map network traces into a robust representation of a class (in machine learning, a class is a set or collection of abstracted objects that share a common characteristic; in our case these are traces recorded for the same webpage). We abstract the loading process of a webpage by generating a cumulative behavioral representation of its trace. From this, we extract features for our classifier. These implicitly cover characteristics of the traffic that other classifiers have to explicitly consider, e.g., packet ordering or burst behavior. By design, our classifier is robust against differences in bandwidth, congestion, and the timing of a webpage load. As we will show, this approach outperforms all existing state-of-the-art classifiers in terms of classification accuracy while being computationally tremendously more efficient.

To evaluate the severity of the WFP attack in reality, we constructed the most representative dataset that has ever been assembled in this domain. It consists of over 300,000 webpages (this is ten times larger than the biggest set used before, i.e., the one described in [14]) and is not subject to simplified assumptions made by the related work. For instance, most researchers consider only the index pages, i.e., those that web servers provide for a requested domain (e.g., [26], [9], [11]). Their objective is limited to differentiating index pages, i.e., to detect certain index pages within a set of other index pages. We clearly differentiate the classification of webpages and websites. Our datasets enable for the first time to study the detectability of both single webpages and complete websites within realistic Internet traffic (serving as

NDSS '16, 21-24 February 2016, San Diego, CA, USA
Copyright 2016 Internet Society, ISBN 1-891562-41-X
http://dx.doi.org/10.14722/ndss.2016.23477

background noise). Further, we do not limit our dataset to the most popular websites according to Alexa[1], since we argue that their index page cannot serve as realistic background traffic. For our datasets, we combined different sources for information such as links distributed via Twitter or traces of a Tor exit node in order to create a random and representative sample of webpages actually visited on the Internet (or, over Tor in particular) at the time of evaluation.

We use our superior attack together with our collected data to study the limits of webpage and website fingerprinting at Internet scale. We investigate the probability of success for different strategies a realistic adversary may follow. We show that with existing classifiers under realistic conditions, *webpage* fingerprinting for *any* webpage is similar to finding a needle in a haystack – in general it is doomed to failure. However, *website* fingerprinting, despite being a more realistic scenario, is also easier to handle for existing classifiers. Our evaluation reveals several tactics that increase the probability for a successful attack.

The contributions of this paper are as follows:

1) We propose a novel WFP attack on Tor based on the idea to sample features from a cumulative representation of a trace. We show that our approach outperforms all attacks existing in the literature on the state-of-the-art dataset in terms of classification accuracy while being computationally more efficient by orders of magnitude.
2) We provide the most comprehensive dataset to evaluate the WFP attack. Instead of being limited to index pages of popular websites, it contains various web pages actually retrieved on the Internet. We managed to assemble more than 300,000 of such pages.
3) Even allowing the attacker an optimal strategy, we show that web*page* fingerprinting at Internet scale is practically unfeasible on the one hand while web*site* fingerprinting has a chance to succeed on the other. We explore neglected aspects of the attack and investigate the realistic probability of success for different strategies a real-world attacker may follow.

II. Background

Tor (The Onion Router) is the most popular anonymization network to date with more than two million daily users[2]. It is designed particularly for low-latency applications such as web browsing. Tor routes connections through virtual tunnels, called *circuits*, which typically consist of three *onion routers*[3] *(OR)*. The traffic is encrypted in layers, i.e., the client establishes a symmetric encryption key with each OR on the circuit, encrypts the data with all keys consecutively, and each OR decrypts its layer on the path. This technique ensures that no relay on a path can know both the origin and destination of a transmission at the same time. The goal of Tor is to improve users' privacy by hiding routing information and communication content. However, Tor is not able to obscure the size, direction and timing of transmitted packets. Information leakage based on these metrics constitutes the foundation of the website fingerprinting attack. The objective is to match patterns of a website load trace to a previously-recorded trace in order to reveal which particular website a user is visiting over the anonymized and encrypted path. Typically, multiple traces of a single website are retrieved and analyzed. These are called *instances*.

Website fingerprinting is commonly evaluated in two scenarios: in the *closed-world* scenario, the number of websites a user may visit is limited to a fixed number. Obviously, this scenario is not realistic. However, it is suitable to compare and analyze the performance of classification approaches. In the more realistic *open-world* scenario, the adversary tries to identify whether a visited website belongs to a given set of monitored websites even though the user may also visit sites unknown to the adversary. Here, we call this set of sites, which are unknown, the *background set* and the set of monitored sites the *foreground set*, correspondingly. In the remainder of this paper we clearly distinguish between the terms “website” and “web page”. A website is a collection of web pages, which are typically served from a single web domain. The initial web page of a website is called the *index page*[4]. This page is served by the web server when a user queries the domain name of the corresponding website. In the related work, website fingerprinting is commonly applied only for such index pages. In our evaluation, we extend the universe to arbitrary web pages, and differentiate between the objectives of an adversary, e.g., to monitor all pages belonging to a particular website, or to monitor a single particular web page.

Attacker Model

We assume the attacker to be a passive observer. He does not modify transmissions and he is not able to decrypt packets. The attacker is able to monitor traffic between the user and the entry node of the Tor circuit. Hence, he either monitors the link itself or a compromised entry node. Further, we assume the attacker to possess sufficient computational resources to train the fingerprinting technique on large training datasets.

III. Related Work

As early as 1996, Wagner and Schneier discovered that traffic analysis can be used to draw conclusions about the content of encrypted SSL packets [24]. We categorize the related work in this research domain into traffic analysis on encrypted connections in general, website fingerprinting on anonymization networks in particular, and countermeasures that have been proposed against such attacks.

A. Traffic Analysis on Encrypted Connections

The first implementation of a website fingerprinting attack was described by Cheng and Avnur [7] in 1998. By looking at file sizes, the authors aimed to identify which specific file was accessed on a known server over an SSL-protected connection. Similarly, Hintz [12] targeted identifying individual websites when the server is not known, e.g., when using an anonymization proxy. In order to detect whether a website from a given blacklist had been visited over an SSL-protected connection,

[1] http://www.alexa.com

[2] According to https://metrics.torproject.org for August 2015.

[3] The onion routers are known as *entry-*, *middle-*, or *exit*-nodes, depending on their position in the circuit.

[4] The index page is often also called the *homepage* or *main page*.

Sun et al. [23] proposed Jaccard's coefficient as a metric for the similarity between observed and pre-collected traffic patterns, allowing websites with slightly varying sizes to be matched. These early works showed the general feasibility of the website fingerprinting attack by considering the total sizes of resources. However, they assumed that each request is associated with a separate TCP connection – a constraint that only holds for early versions of the HTTP protocol. Nowadays, HTTP makes use of persistent connections and pipelining[5] to improve performance. Hence, it is no longer possible to trivially distinguish between single web object requests. Bissias et al. [3] were the first to perform website fingerprinting based on IP packet sizes and inter-packet arrival times instead of web object sizes. This allows the attack to be generalized to VPN or SSH tunnels as well as WPA-protected wireless networks. To further improve the attack, Liberatore and Levine [16] compared the effectiveness of Jaccard's coefficient and the naïve Bayes classifier on SSH-protected channels. Lu et al. [18] showed that website fingerprinting can be improved by considering information about packet ordering. Several related works do not focus on website fingerprinting in particular, but rather on the detection of other distinct characteristics of network traffic, e.g., the language of a Voice-over-IP (VoIP) call [30], or spoken phrases in encrypted VoIP calls [29]. Gong et al. [10] even showed the feasibility of a remote traffic analysis (where the adversary does not directly observe the traffic pattern) by exploiting queuing side channel in routers.

B. WFP in Anonymization Networks

In 2009, Herrmann et al. [11] were the first to apply website fingerprinting to the anonymization networks JAP [2] and Tor [8] as well as on OpenSSH, OpenVPN, Stunnel, and Cisco IPsec-VPN. In addition to the classifiers used by Liberatore and Levine, the authors also evaluated a multinominal naïve Bayes classifier. Using this classifier and a dataset consisting of 775 index pages, they achieved recognition rates above 90% for single-hop systems, but only 20% for JAP, and as low as 2.95% for Tor. Therefore, Tor was considered to be secure against website fingerprinting until Panchenko et al. [21] increased the recognition rate for Tor to an alarming degree using an approach based on Support Vector Machines (SVM) in 2011: in the dataset provided by Herrman et al., they recognized more than 54% of the URLs correctly when accessed over Tor. Moreover, the authors were the first to evaluate website fingerprinting in an open-world scenario, i.e., they recognized a small number of (monitored) pages in a set of thousands of unknown, random pages that classifier has never seen before. Here, they achieved a recognition rate of up to 73%. These results spawned a significant amount of interest in the research community.

Dyer et al. [9] compared existing classifiers and additional features on datasets with 2, 128, and 775 websites. However, their proposed time, bandwidth, and variable n-gram classifiers did not improve the recognition rate compared to the approach of Panchenko et al. in any of the considered scenarios. In 2012, Cai et al. [5] presented an approach achieving a recognition rate of over 80% for a dataset with 100 URLs and over 70% for 800 URLs. Like Panchenko et al., they utilized an SVM, but their features are based on the optimal string alignment distance (OSAD) of communication traces. They were the first to study the recognition of different pages of a website and the effect of clicking on embedded links, i.e., browsing within the same website, using a Hidden Markov Model. Though such a user behavior turned out to be detectable with a high probability, their study was limited to two websites only. Wang et al. [26] improved the optimal string alignment distance approach of Cai et al. and enhanced the data preparation methods by statistically removing Tor management packets. With these improvements they obtained recognition rates of better than than 90% for both the closed-world (100 URLs) and the open-world (1,000 URLs) scenarios. Recently, the authors further improved the recognition rates in larger open-world scenarios ($>$ 5,000 URLs) using a novel k-Nearest Neighbor (k-NN) classifier, which also significantly reduces the time necessary for training compared to previous results [25].

The latest contributions to the field of website fingerprinting in Tor were made by Juarez [14], Cai [4], and Kwon et al. [15]. Juarez et al. [14] critically evaluate typical assumptions in WFP attacks. They showed that the accuracy of classification decreases by 40% in less than 10 days and further declines almost to zero after 90 days for Alexa Top 100 pages due to content change. Also, the accuracy drops dramatically if a user performs multitab browsing or if different generations of the Tor Browser Bundle (TBB) are used for training and testing. Unfortunately, their analysis did not consider an attacker that is able to use different versions/settings for training although a realistic adversary would have this capability. Moreover, the authors observe a similar impact on the attack, if the adversary is not able to train using exactly the same Internet connection as the user. The authors are the first to consider the base-rate fallacy in the scope of the WFP attack[6]. They show that, though the accuracy of classification is very high, due to a large universe size, in most of the cases the adversary would wrongly conclude that the user had accessed a monitored page. To improve the situation, they propose to refrain from positive classification if the probability difference between the two closest classification decisions is below a certain threshold.

Cai et al. [4] analyze WFP attacks and defenses from a theoretical perspective using a feature-based comparative methodology. The goal is to provide bounds on the effectiveness of proposed defenses, i.e., to which extent certain defenses hide which feature. Moreover, the authors propose a methodology to transfer closed-world results to an open-world setting. However, we argue that their theoretical definition of the corresponding open-world classifier cannot hold in practice. The key idea in the open-world scenario is to test a classifier on traces of websites it has never seen before. In practice, it cannot be guaranteed that an open-world classifier identifies a monitored page *if and only if* the corresponding closed-world classifier detects that particular page as defined in their work. Instead of deriving theoretical bounds, we perform a practical evaluation of the attack.

Recently, Kwon et al. [15] have applied the WFP attack in the scope of Tor hidden services. Their approach can

[5] HTTP pipelining is a method in which multiple requests are sent via a single TCP connection without waiting for the corresponding responses.

[6] Note, however, that this issue has already been publicly discussed in the Tor community before, see https://blog.torproject.org/blog/critique-website-traffic-fingerprinting-attacks

substantially distinguish a Tor hidden service connection from a regular Tor circuit – assuming the attacker controls the entry node – but has only moderate success in differentiating between hidden services.

C. Countermeasures against WFP

Several countermeasures have been proposed to protect against website fingerprinting attacks. *Padding* as a basic countermeasure was first studied by Liberatore and Levine [16]. Tor employs padding to generate cells of a fixed size, which are indistinguishable. While padding operates on a per-packet level, *traffic morphing* aims to adapt a complete packet trace such that it looks similar to another packet trace [31]. However, Dyer et al. [9] showed traffic morphing to be ineffective as a defense against WFP in practice.

A number of countermeasures aim to create a *continuous data flow*. Panchenko et al. [21] proposed creating background noise by loading a random website in parallel with the actually desired website thus obfuscating the real transmission. However, Wang et al. [25] stated that this approach is not powerful enough to prevent website fingerprinting if the traffic overhead is to be kept reasonable. Introducing BuFLO (Buffered Fixed-Length Obfuscation), Dyer et al. [9] reduced the amount of information exploitable by an adversary, by sending packets with a fixed size and at fixed intervals. Cai et al. pointed out several disadvantages of this approach [5]: besides a high overhead in bandwidth and time, BuFLO may reveal the total transmission size under certain conditions and further is not able to adapt for congestion. To overcome these flaws, they proposed Congestion-Sensitive BuFLO (CS-BuFLO). Cai et al. also proposed Tamaraw [4], a heavily-modified version of BuFLO, which improves performance primarily by treating incoming and outgoing packets differently. Glove [20] is an SSH-based defense that uses knowledge of website traces for traffic morphing. The idea is to cluster all web pages into large similarity groups and add only a small amount of cover traffic to make all the pages within a cluster indistinguishable. Hence, the attacker can only identify the cluster to which the web page belongs, but not the web page itself. Built upon Tamaraw, a similar idea – called Supersequence – is proposed by Wang et al. [25]. However, to be successful, this approach needs to have a-priori information about each page to be protected. To overcome this, Wang and Goldberg propose *Walkie Talkie* [28] – a general defense that enables the Tor Browser to transmit in half-duplex mode. The idea is to buffer packets in one direction and send them in bursts together with dummy traffic. This usually results in a lower bandwidth overhead compared to Tamaraw or Supersequence and allows for a variable packet rate to deal with congestion.

Finally, several *countermeasures at the application layer* have been proposed that do not introduce additional traffic. As a response to the evaluation of Panchenko et al. [21], the Tor project released an experimental patch of the Tor Browser Bundle, which randomizes the pipeline size (i.e., the quantity of requests processed in parallel) and the order of requests for embedded website objects. This technique is called *randomized pipelining* [22]. HTTPOS [19] (HTTP Obfuscation) follows a similar approach of altering packet sizes, web object sizes, and timing by modifying HTTP and TCP requests. This is achieved, e.g., by changing the HTTP accepted-range header field that is used to specify the byte range of a requested resource. This functionality is typically utilized to resume a download of a larger web object. The client can, for example, change the traffic pattern of requesting a large resource to the traffic pattern of multiple requests of small resources. However, both Cai et al. [5] and Wang et Goldberg [26] showed that these defenses are not as effective as assumed and, in the case of randomized pipelining, might even lead to increased recognition rates on small datasets. Since the severity of the WFP attack has not been comprehensively studied to date, none of these countermeasures is currently applied in Tor.

IV. Data Sets

This section describes the datasets used in our evaluation. The significance and plausibility of results strongly depends on the dataset used for training and testing a classifier. In general, a dataset should be a representative, independent, random sample of the considered universe, here, the world wide web. All prior works in this area limited their samples to index pages of the most popular websites or small sets of sites known to be blocked in certain countries. We argue that these datasets do not reflect a representative sample of the Internet. First, they do not contain any webpage of a site besides the index page while the majority of retrieved web pages are not main pages but articles, profiles or any other type of sub-page. Users surfing the web often follow links, e.g., communicated through social networks or they retrieve the index page of a website in order to manually select interesting links. This is particularly important when conducting an open-world analysis. Even if the attack focuses on fingerprinting certain index pages, it is important to use realistic traffic traces as background data to evaluate the success. This fact has been neglected in the related research, i.e., the problem has been simplified to classify index pages within a set of other index pages instead of real traffic. Second, none of the existing datasets allows an evaluation of fingerprinting for complete websites, though this constitutes an attack scenario to be expected in reality (a censor may rather be interested in blocking or monitoring access to Facebook entirely instead of blocking the Facebook login-page only). Third, the world wide web consists of billions of pages. Therefore, results obtained on small datasets do not allow generalization. Our novel datasets aim to avoid the limitations described above. Additionally, we tested our methods also on data provided by other researchers, particularly to compare the effectiveness of our attack. We now describe the compilation of our datasets in detail.

A. Data sets provided by Wang et al.

To compare the performance of different approaches in terms of classification accuracy and computation time, it is essential to evaluate the classifiers on the same datasets. Wang et al. provide two datasets[7]; one, which we refer to as `WANG13`, had been used to evaluate the outdated OSAD classifier in [26], and the other, which we call `WANG14`, had been used to evaluate and compare the k-NN approach in [25]. The `WANG13` dataset contains traces of 100 websites with 40 instances each. The websites are based on Alexa's top sites, where the authors manually removed different localizations of the same site (e.g., google.com and google.de). Obviously, this dataset is only

[7] https://cs.uwaterloo.ca/~t55wang/wf.html

suitable for a closed-world analysis due to its limited size. The WANG14 dataset was built for an open-world analysis. It contains 100 websites with 90 instances each that can be used as foreground class, i.e., as the set of sites monitored by an adversary, or for a closed-world evaluation. This subset was compiled from a list of blocked websites in China, the United Kingdom, and Saudi Arabia. Further, WANG14 includes traces of 9,000 websites drawn from Alexa's Top 10,000 with one instance each to serve as background data. Note that both datasets include only the index page of each website.

Both datasets include information about the direction of each cell and certain management cells (SENDME) were removed using a probabilistic method described in [26]. While WANG13 provides only cell direction and order, WANG14 also includes a timestamp for each cell. This timestamp is necessary for extracting the required characteristics used by the k-NN classifier. In our evaluation we also investigate different layers to extract the information used to generate trace representations (see Section VII-A). Since the required information of these layers is not available in the WANG13 set, we re-recorded the websites used with 100 instances each using our own approach. This allows us to extract all layers of data representation (i.e., also those where SENDME cells are included). Additionally, we can transform our own format to the input format of the k-NN classifier and compare our results. We refer to this dataset as ALEXA100.

B. RND-WWW: An Unbiased Random Sample of the World Wide Web

Obtaining a representative sample of web pages visited by typical users is a challenging task. Logs of central intermediaries, when they are available, e.g., for Internet Service Providers, are generally not publicly available due to privacy concerns. If we were to monitor the web surfing behavior of users, the test results could be biased by the selection of users (e.g., students), and the monitoring process itself, e.g., by the Hawthorne effect[8]. To avoid these issues, we combined several sources, each of which covers a different aspect of anticipated user behavior. We call this dataset RND-WWW. In detail, it is composed of web pages gathered using the following methods:

1) **Twitter** is a popular social network with more than 300 million average monthly active users that offers micro-blogging services, i.e., users are allowed to post messages with a maximum length of 140 characters, called *tweets*. Many people use this service to distribute links to web pages they are currently interested in, and it is to be assumed that many users follow these links. Therefore, Twitter serves as a source of URLs of recent actual interest. Twitter provides an API which enables live access to a stream of randomly-chosen tweets that are currently being posted. From this stream we extracted all HTTP links over a period of three days and resolved the original URL, since Twitter applies a URL shortening service. From this source we were able to gather about 70,000 unique URLs of web pages.
2) **Alexa-one-click**: As described above, it is uncommon only to visit the index page of a popular website. Instead, users typically also click on links on these pages, or they follow external links that directly lead to subpages of a site, e.g., a link to a particular article on a news website. To simulate this behavior, we loaded the index page of each website in the Alexa Top list of the 20,000 most popular sites and followed a randomly chosen link on this page. We then included the resulting page in our dataset.
3) **Googling the trends**: Google, by far the most popular search engine, publishes the keywords that were queried most frequently in past years per country as *trends*[9]. We used 4,000 trends from Australia, Canada, Germany, Hong Kong, India, Israel, Japan, Singapore, Taiwan, Russia, the United Kingdom, and the USA and queried the corresponding country-specific Google website for these keywords. We then randomly selected a link with a probability of 0.5 from the first, 0.25 from the second, 0.125 from the third, and 0.0625 from the fourth and fifth result pages and included the selected target web page in our dataset.
4) **Googling at random**: We selected 20,000 English terms at random from the Beolingus German-English dictionary[10] and entered them into Google Search. From the results, we selected web pages with the same method as described for Google trends.
5) **Censored in China**: We added a list of 2,000 websites blocked in China according to http://greatfire.org.

After removing duplicate entries, we were able to combine more than 120,000 unique web pages in total from these sources. In the related research, the set of websites that is considered to be monitored by an adversary (i.e., the foreground class) is commonly selected from a set of URLs that are known to be actually blocked. In our evaluation, however, we are more interested in investigating whether it is feasible for an adversary to monitor *any* possible web page. Therefore, we randomly selected a sample of 1% of the pages included in RND-WWW as the foreground class and downloaded 40 instances of each with the methods described in Section V-A. Accordingly, the remaining 99% served as background traffic and were downloaded in one instance each.

Finally, the foreground set of RND-WWW consists of 1,125 retrievable web pages, combined from 712 different websites. The sites with the highest frequency are http://facebook.com (88 pages) and http://instagram.com (86 pages). The background set is composed of 118,884 unique and accessible web pages distributed among 34,580 different websites. Besides the four websites http://facebook.com, http://youtube.com, http://instagram.com, and http://tumblr.com, no website is represented by more than 2,000 pages. Moreover, 28,644 websites occur only once in RND-WWW, i.e., they are represented by a single web page.

[8] This effect refers to the psychological phenomenon that individuals modify their behavior as a response to the awareness of being monitored.

[9] http://www.google.com/trends/

[10] Beolingus contains almost 1,000,000 terms including sayings, aphorisms, and citations. The database is available for offline use at http://ftp.tu-chemnitz.de/pub/Local/urz/ding/de-en/de-en.txt.

C. Monitoring Real Tor Traffic

For the reasons mentioned in Section II, the website fingerprinting attack is typically evaluated against Tor. Therefore, an intuitively representative sample of web pages to be considered for our evaluation are those pages which are *actually accessed* through the Tor network. Thus, Tor itself serves as source of URLs used for our second dataset, called `TOR-Exit`.

To get access to this data, we operated a public Tor exit node. We ensured that the *fast* and *stable* flags were assigned to our node and that its information was thoroughly propagated by the Tor directory service. Hence, it was fully integrated into the operational service of Tor and used in the circuits of real users. We captured HTTP requests from this exit node over a duration of one week. We deliberately limited our selection to plain HTTP traffic as we did not want to interfere with encrypted connections. In general, it is not possible to infer from a HTTP GET request which web page has actually been retrieved, because there is a separate request for each object embedded into a page. Since we are primarily interested in web pages actually visited by Tor users, we extracted URLs in the following manner: HTTP requests typically include a header element called *HTTP referer*[11], which provides the URL of the web page that linked to the resource being requested. Hence, in the case of an embedded object, the referer points to the page containing this object, and if the user followed a link, the referer points to the page containing this link. Thus, the value of the referer serves as suitable source of URLs for web pages that users actually visited.

From all HTTP requests that contain an HTTP referer[12], we include the web page pointed to by the referer in our dataset. From those requests without a referer, we extracted the domain name from which the object is requested and added the website (or, more precisely, its index page) accessible through this domain to our dataset. In both cases, we discarded duplicate entries but deliberately included different web pages belonging to the same website if available. Additionally, we removed query parameters such as session identifiers, as these would render a subsequent retrieval impossible. Further, we removed all URLs linking to pure advertisement services, as these are obviously not the target web pages sought by users. With the method described above, we obtained a list of 211,148 unique web pages. The set contains 65,409 unique web domains of which 45,675 occur only once. Each website is represented by fewer than 2,000 web pages.

Ethical considerations: The collection of URLs for the `TOR-Exit` dataset does not strictly follow the guidelines for ethical Tor research that were published after we finished our experiments[13]. Still, we believe that it is important to know the degree of protection offered by the real Tor network. There can be no better evaluation than using those pages that are actually retrieved via Tor. While running our exit nodes, we made every effort to minimize any potential harm to the users and tried to examine all the risks that may exist. Our scripts extracted and stored only the URLs – without timestamps, traces or any other data. From these we automatically removed all identifying information such as a session identifier. Hence, with the information we stored there is only a minimal risk to harm the anonymity of Tor users. This complies with the recommendations for statistical analyses in the Tor network (except that we did not publicly discuss our algorithms before conducting the experiments) [17]. Furthermore, we will not make this dataset or parts of it publicly available before consulting the Tor research community and an appropriate ethics feedback panel such as the Ethics Feedback Panel for Networking and Security[14].

D. Website-Collections

We compiled an additional dataset, called `WEBSITES`, with the aim to investigate whether it is possible to fingerprint a complete website, given that the adversary is only able to use a subset of its pages for training. We assume this to be one of the most realistic attack scenarios. To do this, we selected 20 popular websites that cover different categories (e.g., news sites, social networks, online shops), different layouts, and contents from different regions in the world. Within each of these websites we selected a set of 50 different accessible pages by following links from the index page applying the same method as described for 'Googling the trends' in Section IV-B. We then recorded 90 instances of the index page and 15 instances for each of the 50 subpages for all sites. The complete list of websites used in this dataset is available in the appendix.

V. Experimental Setup

In practice, without loss of generality, we assume that an attacker retrieves a certain amount of relevant web pages by himself as training data for fingerprinting, using the anonymization network that he assumes his victim uses as well. He records the transferred packets with a traffic analyzing tool which provides information about IP layer packets, i.e., the length of the packet, the time the packet was sent or received, the order in which the packets were sent and received, etc.

The attacker can make use of various information contained in the dumps to create a profile of each web page, called *fingerprint*. Later, wiretapping on the victim's traffic, the attacker tries to match the collected test data to a known fingerprint. Usually, a difference between patterns in training and test data is to be expected due to a number of reasons, e.g., indeterministic packet fragmentation, updates in web pages, varying performance of Tor circuits, etc. Hence, the attacker needs to apply statistical methods to compare the recorded information to the fingerprints and to probabilistically match it to a certain web page.

A. Data Collection

We accessed the Tor network using the Tor Browser Bundle. This self-contained software package combines a preconfigured Tor client and a stand-alone web browser based on Mozilla Firefox. We used version 3.6.1, which includes patches against website fingerprinting by applying randomized pipelining. The intention of TBB is to provide an easily-deployable solution that leaks the minimum amount of information due to identical browser configurations.

[11] Note that HTTP referers may be disabled on the client side. However, they are enabled by default in the Tor Browser Bundle.

[12] From the captured HTTP requests, 96.3% contained a HTTP referer.

[13] https://blog.torproject.org/blog/ethical-tor-research-guidelines

[14] https://www.ethicalresearch.org/efp/netsec/

We recorded the traces of web pages using *tcpdump*. We automated the recording using the plugins Chickenfoot[15], iMacros, and Scriptish[16] and controlled the functionality of Tor with Stem, a Python implementation of the Tor Control Protocol (please note that our Tor controller does not interfere with the regular circuit creation, assignment and fetching of websites). With these methods we were able to automatically retrieve hundreds of thousands of web pages selected from a list of the corresponding URLs. Typically, multiple instances of a single web page are retrieved and analyzed. This is a requirement for training the classifier and for the cross-validation we use in our evaluation. We ensured that we never downloaded more than one instance of a single page through the same circuit, as this could distort the evaluation results[17].

B. Data Extraction and Processing

The features we use for fingerprinting are based on packet size, direction, and ordering. It is possible to extract this information at *different layers*: cell, TLS, and TCP. At the application layer, Tor embeds the encrypted data in fixed-size packets, called *cells*, with a length of 512 bytes, and cells are further embedded into TLS records. Note that multiple cells may be grouped into a single TLS record. Finally, in the transport layer, TLS records are typically fragmented into multiple TCP packets whose size is bounded by the maximum transmission unit (MTU). Alternatively, several TLS records can be within a single TCP packet. In Section VII-A we provide an evaluation of the different layers of extraction and discuss their implication on classification.

From our recorded data, we removed faulty traces that are identifiable either by being empty or by an HTTP status code indicating a load error. Further for pages with multiple instances, we removed outliers identified by the interquartile range, a standard measure to detect outliers in noisy measurement data [1]. To apply this method, we compute I as the sum of incoming packet sizes for each instance of a web page and the corresponding quartiles. We then remove those instances that do not satisfy the following inequality:

$$Q_1 - 1.5(Q_3 - Q_1) < I < Q_3 + 1.5(Q_3 - Q_1).$$

On average, 5% of the traces were discarded as outliers by this method.

VI. A NOVEL WEBSITE FINGERPRINTING ATTACK

The classifier with the highest accuracy known to date is that proposed by Wang et al. [25]. It is based on a k-Nearest-Neighbor machine learning approach. The algorithm calculates the distance between data points (here, packet sequences) and classifies a test point based on the class of the k closest training points. The major component of a k-NN classifier is the distance function. Wang et al. use the sum over weighted feature differences as distance metric. Based on prior knowledge, they manually selected a large set of features, including characteristics such as unique packet lengths, concentration of outgoing packets, or bursts. The weights are learned using an iterative process, where the weights are initialized with random values and then adjusted over several thousands of iterations in order to optimize them. The feature set used for their evaluation consists of almost 4,000 features.

We follow a contrasting approach. Instead of manually identifying characteristics that may contain significant information about the load behavior, we aim rather to derive our features from an abstract representation that implicitly covers all relevant characteristics. As identified in [25] and [21], there are four basic features that already contribute significant distinctive information: $\mathcal{N}_{\text{in}}$, the number of incoming packets, $\mathcal{N}_{\text{out}}$, the number of outgoing packets, $\mathcal{S}_{\text{in}}$, the sum of incoming packet sizes, and $\mathcal{S}_{\text{out}}$, the sum of outgoing packet sizes. Therefore, we include these four features in our feature set. To characterize the progress of the page load we propose using the cumulated sum of packet sizes as an abstract representation and to sample a fixed number n of additional features from this representation.

When we apply the methods described in Section V-B on our recorded data, we obtain a sequence of packet sizes, where a packet may refer to a raw TCP packet, a TLS record, or a Tor cell, depending on the layer used for data extraction. Given such a trace of packet sizes $T = (p_1, \ldots, p_N)$, where $p_i > 0$ indicates an incoming packet and $p_i < 0$ an outgoing packet, the cumulative representation of this trace is calculated as

$$C(T) = ((0, 0), (a_1, c_1), \ldots, (a_N, c_N)),$$

where $c_1 = p_1$, $a_1 = |p_1|$, and $c_i = c_{i-1} + p_i$, $a_i = a_{i-1} + |p_i|$ for $i = 2, \ldots, N$. From this representation, we derive n additional features $\mathcal{C}_1, \ldots, \mathcal{C}_n$ by sampling the piecewise linear interpolant of C at n equidistant points. This feature extraction process is illustrated in Figure 1. As a simplified example we show the cumulative representation of two traces T_1 and T_2 consisting of $|T_1| = 18$ and $|T_2| = 14$ packets (each of size ± 568 bytes) and the corresponding features $\mathcal{C}_i$ for $n = 10$. With this method, we are able to extract a fixed number of identifying characteristics from traces with varying length. Note that typically $N \gg n$, i.e., the trace of a website consists of significantly more packets than the number of features that we sample. In Section VII-B we show that $n = 100$ yields the best trade-off between classification accuracy and computational efficiency. In the following, we refer to this fingerprinting approach as CUMUL.

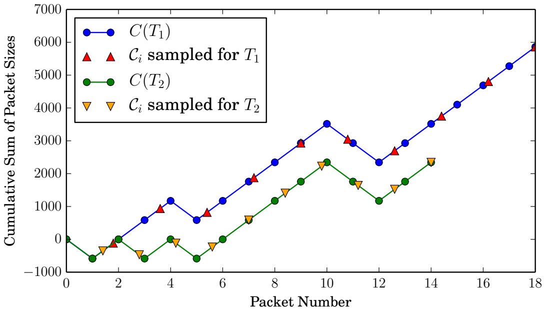


Fig. 1: Feature extraction from the cumulative representation of two traces

[15] http://groups.csail.mit.edu/uid/chickenfoot/

[16] http://imacros.net/ and http://scriptish.org/

[17] Note that an adversary is generally not able to download training instances over exactly the same circuit that the observed client uses. As path selection in Tor is randomized, this information is not available to the considered attacker (except the address of the entry node)

As a beneficial side-effect of our feature set, fingerprints can be intuitively visualized and compared. In Figure 2 we visualize sample fingerprints derived with our method from the recordings of two popular websites: about.com and google.de. For both websites we recorded 40 instances. As we can see, the load behavior of google.de is rather consistent and mainly characterized by a sequence of incoming packets, which is interrupted by a few outgoing packets at certain distinct points in the progress. The fingerprints derived for about.com, a website that publishes articles and videos on various topics, show a greater variation. This site contains several embedded dynamic objects (e.g., images) and their size and position in the trace may vary. Nevertheless, the two websites are characterized by clearly distinctive load behaviors. Our subtle method to represent this load behavior based on the cumulated packet sizes enables the differentiation of fingerprints of these two pages even by the human eye. Obviously, this is not always possible. Therefore, based on our feature set, we collect a set of valid fingerprints and apply a machine learning technique to differentiate them. We use a Support Vector Machine. Since the fingerprints have by definition a fixed length, we can directly use them as input to train the SVM classifier.

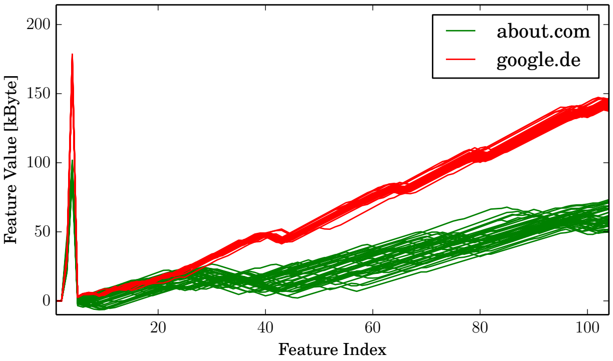


Fig. 2: Visualized fingerprints of two websites

To evaluate our approach, we used LibSVM [6] with a radial basis function (RBF) kernel, which is parametrized with parameters c and γ. LibSVM includes a tool to optimize these parameters using cross-validation. It applies a grid search, i.e., various combinations are tested and the one with the best cross-validation accuracy is selected. In its standard implementation the values are chosen from exponentially growing sequences $c = 2^{-5}, 2^{-3}, \dots, 2^{15}$ and $\gamma = 2^{-15}, 2^{-13}, \dots, 2^{3}$. We adjusted these sequences to $c = 2^{11}, \dots, 2^{17}$ and $\gamma = 2^{-3}, \dots, 2^{3}$, since parameters chosen from these ranges yielded the highest accuracy for our data while reducing the computation time. Before applying the SVM we scale each feature linearly to the range $[-1, 1]$. This prevents features in greater numeric ranges dominating those in smaller numeric ranges [13]. For all following evaluations in this paper where we do not explicitly mention a different methodology, we always apply our classifier as described in this section using 10-fold cross-validation.

VII. Evaluation and Discussion

In this section we evaluate our novel website fingerprinting attack in Tor. We first identify the optimal layer of data extraction, then we optimize the parametrization of our method. In Section VII-C we thoroughly compare our approach to the state-of-the-art attack, the k-NN classifier proposed by Wang et al. and show that our method is superior in terms of classification accuracy both in the closed- and open-world setting as well as regarding computational efficiency and scalability. Based on this, we evaluate our approach in different attack scenarios. We show that monitoring a single web page while considering realistic background traffic is doomed to failure. However, in Section IV-D we also provide insights into the scenario, where the attacker aims to monitor complete websites and show that this scenario is still ambitious but more feasible in practice, particularly when considering our improved strategies for training the classifier.

A. Layers of Data Extraction

As explained in Section V-B, there are three possible layers for data extraction: Tor cell, TLS record, and TCP packet. From the raw TCP data we can extract all three layers. The question we will address now is, which layer provides the most information content with respect to website fingerprinting. Our first intuition was that the most information is contained in the TLS layer because only at this layer the dependency of cells that belong together is included. If a record R is still being transmitted in one direction and the transmission of a second record R' in the opposite direction starts before the TCP packet containing the end of record R is sent, then R' *cannot* contain data that is sent in response to the data contained in R. We illustrate this situation in Figure 3.

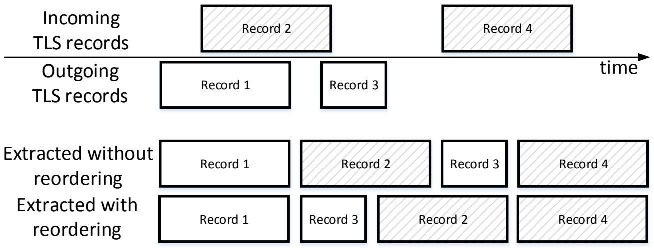


Fig. 3: Example of TLS record reordering

The figure shows four TLS records, their transmission time, and direction. If the records were ordered according to the beginning of each transmission, the resulting sequence would be $1, 2, 3, 4$. However, the data contained in record 3 cannot be a response to record 2, since its transmission has started before that of record 2 had ended. As we require an abstract representation of the loading behavior of the website – independent of noise introduced by the transmission – the sequence must be reordered to $1, 3, 2, 4$. Hence, we reordered the TLS records extracted from all our traces accordingly.

The best classifier known to date, proposed by Wang et al. [26], [25], uses the cell layer. Tor issues a special cell type called SENDME to ensure flow control. These cells are irrelevant for the load behavior of a website and, thus, are a source of noise in the measurement. Wang et al. use a probabilistic algorithm to identify and remove SENDME cells, sometimes leading to a slight improvement in accuracy. This method is also applicable when data is extracted as sequence of TLS records: if a cell is assumed to be a SENDME by the probabilistic algorithm, we can reduce the size of the record containing this cell by 512 bytes. This leads to five different layers of data extraction, which we evaluate: TCP, TLS, TLSNoSENDME, Cells, and CellsNoSENDME. These layers are illustrated in Figure 4 where it is assumed that Cell 3 is a SENDME. Accordingly, this cell would be removed for the CellsNoSENDME format and TLS record 1 would be shrunk by 512 bytes (indicated by the asterisk) for TLSNoSENDME.

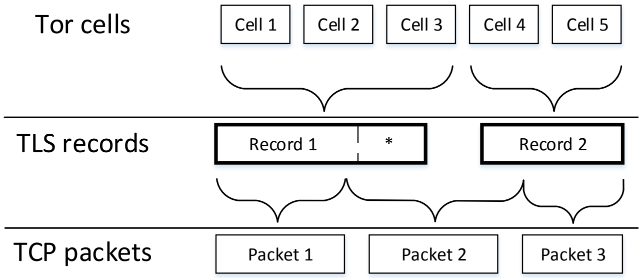


Fig. 4: Layers of data transport used for information extraction

To identify the best layer of data extraction we performed several experiments in different settings. Table I exemplarily shows the classification accuracy in a closed-world setting for the ALEXA100 dataset using 40, 60, and 90 instances for both classifiers, k-NN and our novel CUMUL. As we can see, the layer used for data extraction does not have a significant influence on classification accuracy. Surprisingly, throughout our experiments the TLS format did not yield the best results (regardless whether we reorder TLS records as described above or not). The best classification accuracy is achieved by extracting data on the Cells layer. However, the differences are small and similar results can be obtained using even the most basic layer of data representation that does not require any post-processing, i.e., TCP. The effect of removing SENDME cells in the TLS and Cells format with the probabilistic method did not achieve consistent results: it either minimally decreases or minimally improves the accuracy. Hence, the effect is negligible and not consistently beneficial. Obviously, this method is not able to reliably detect and remove these management cells. Thus, we omit removing SENDME cells in the following evaluations.

In the definition of the cumulative flow representation $C(T)$ used to derive the features of our classifier (see Section VI), we defined a_i to be incremented by $|p_i|$, i.e., the absolute packet size for each packet. However, intuitively, it may be beneficial to increment a_i by one instead of the absolute packet size when considering a layer of data extraction with varying packet sizes. Otherwise, it is not possible to differentiate whether data has been received in form of one larger chunk or several smaller chunks (e.g., TCP packets). We first assumed that a large cohesive data block is a distinctive characteristic of a webpage, e.g., an embedded object such as an image. Neglecting this information while processing data may negatively influence the classification accuracy. However, as our experiments indicated, this assumption turned out to be incorrect. Throughout all our experiments with different layers of extraction, incrementing a_i by absolute packet sizes yielded significantly better results than incrementing a_i by one (except for the Cells format, where both operations generate identical features due to equal chunk sizes). It appears that larger data chunks are not a characteristic of the page load behavior (that our attack strives to fingerprint) but rather of the Tor circuit or other network properties (and, hence, is to be treated as noise).

TABLE I: Accuracy of both classifiers for the ALEXA100 dataset (all values in %).

	90 Instances	**60 Instances**	**40 Instances**
k-NN (3736 features)			
Cells	91.60	91.95	88.89
CellsNoSENDME	91.97	91.76	89.50
CUMUL (104 features)			
TCP	92.52	91.58	90.43
TLS	91.18	90.06	89.22
TLSNoSENDME	92.02	91.97	90.28
Cells	92.22	91.99	90.53
CellsNoSENDME	91.72	91.33	90.03

B. *Optimizing Feature Sampling*

An important design choice for our novel classifier is the number n of features, i.e., the sampling frequency for the features $\mathcal{C}_1, \ldots, \mathcal{C}_n$. On the one hand, the more fine-grained we sample, the lower is the information loss caused by the sampling. On the other hand, a high number of features negatively influences the performance in terms of computation time and, thus, scalability for the SVM, because the number of features has a linear influence on the computational complexity of the underlying optimization problem[18]. To identify the optimal trade-off between classification accuracy and computation time, we varied the sampling frequency n between 10 and 200. The results are shown in Figure 5. As we can observe, there is no significant increase in classification accuracy for more than about 100 features for all the three layers of data extraction. Hence, we consider $n = 100$ as a good choice and use this number of features for the remaining evaluation.

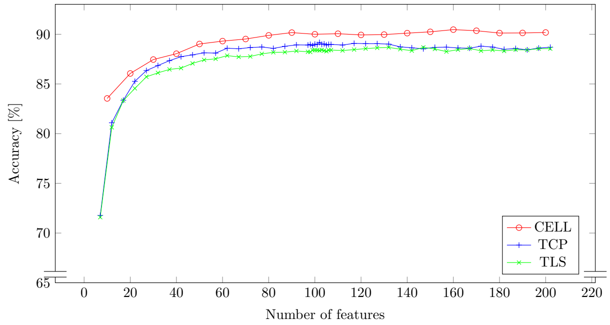


Fig. 5: Accuracy as a function of the number of sampled features

[18] The quadratic programming solver used in libsvm scales between $\mathcal{O}(fn^2)$ and $\mathcal{O}(fn^3)$, where n is the number of samples and f is the number of features, see http://scikit-learn.org/stable/modules/svm.html.

C. Comparison with State of the Art

In this section, we compare the performance of our novel technique to the state-of-the-art approach, i.e., the k-NN classifier proposed by Wang et al. [25], that has been shown to outperform prior approaches.

1) Closed World: We first compare the performance in the closed-world scenario using the ALEXA100 dataset and the 100 websites provided as foreground in the WANG14 dataset. Although the closed-world scenario is generally not realistic, it is suitable for comparing the classification performance of different classifiers. In this context, the accuracy, which is defined as the fraction of correct classifications (positive and negative) among the total number of cases examined, is a suitable metric. Note that in the case of unbalanced datasets that we have to face in the open-world setting, the accuracy is practically meaningless and, thus, other metrics are to be evaluated.

TABLE II: Accuracy of both classifiers for the WANG14 dataset (all values in %).

	90 Instances	**40 Instances**
k-NN (3736 features)	90.84	89.19
CUMUL (104 features)	91.38	92.03

The results for the ALEXA100 dataset are shown in Table I, where we also differentiate between all evaluated layers of data extraction. In Table II we show the results for the WANG14 dataset. Here, for reasons of clarity we only show the results for that data format for which each classifier performed best. As we can see, our CUMUL classifier, which is based on only 104 intuitive features – compared to 3736 synthetically generated features for k-NN – generally achieves a greater accuracy than k-NN on both datasets. However, the improvements are marginal, at about only 1 to 2 percentage points. Note that the accuracy obtained using both approaches is already remarkably high. Therefore, it is questionable whether further efforts to improve the accuracy on such datasets are reasonable. This could finally lead to the problem that the features consider unique characteristics of the particular websites contained in these sets and, hence, results might no longer be generalizable. This problem is related to *overfitting*, i.e., the effect of a model describing observed noise rather than the underlying relationship. Additionally, we identified a difference in the implementation of the cross-validation methodology of both classifiers that makes a fair comparison difficult: while the k-NN uses 60 instances of each foreground page for weight learning and 30 instances for testing, the SVM performs an internal cross-validation to optimize kernel parameters, which uses all 90 instances of each foreground page. Besides this basic separation difference, the selection of testing instances might also differ. To make the results more comparable, we modified the implementation of both approaches accordingly for the open-world scenario described in the next section.

2) Open World: For the comparison in the more realistic open-world scenario, we used the complete WANG14 dataset, consisting of 100 foreground pages (90 instances each) and 9,000 background pages (1 instance each). To avoid the cross-validation issue described above, we implemented an additional, enclosing 10-fold cross-validation. This step ensures that the data used for training and testing in each fold is exactly the same for both classifiers while all 90×100 foreground and 9000×1 background instances are used exactly once for testing. Thus, within each fold, we select 90% of the data (100×81 foreground and 8100×1 background instances) for training, i.e., weight learning of the distance function and calculating differences to the testing point in the case of k-NN and optimizing the kernel parameters in the case of the SVM. Correspondingly, we obtain 1800 testing predictions for each fold and classifier. For the comparison we consider two scenarios: *multi-class* and *two-class*. The basic difference is whether each foreground page is treated as a different class (multi-class) or the whole set of monitored pages forms a single class (two-class). In the two-class scenario, the chances for false positive classifications are lower, because confusion within the foreground (i.e., a particular monitored page is recognized as being a different monitored page) is irrelevant. Note that the difference in both scenarios is not a matter of optimizing a different classifier (as this would falsely imply that monitored pages have a similarity that distinguishes them from the background, which is not true in practice), but rather a matter of counting, i.e., whether to count confusion between two foreground pages as false positive or not.

TABLE III: Results for the open-world scenario of both classifiers using the WANG14 dataset (all values in %).

	multi-class		**two-class**	
	k-NN	CUMUL	k-NN	CUMUL
TPR	89.61	96.64	90.59	96.92
FPR	10.63	9.61	2.24	1.98

The results are shown in Table III. For the multi-class scenario, we can see that our method clearly outperforms the k-NN with a TPR, which is 7 percentage points higher while achieving a lower FPR. Further we observe that the TPR and FPR of the modified k-NN implementation perfectly match the ROC curve shown in [25]. Thus, we conclude that our modification did not influence the classification accuracy. It would be interesting to compare complete ROC curves of both classifiers instead of single values. However, we have to leave this for future work due to the enormous computation time required.

For a realistic adversary such as a state-level censor, confusion within the monitored pages is not a problem. Therefore, the results in the two-class scenario are applicable in practice. Here, we see the same relation in the results, i.e., our approach achieves a clearly higher TPR for a lower FPR. Interestingly, we observe a significant decrease of false positives when considering only two classes. This means that most of the false positives in the multi-class scenario have been caused by intra-foreground confusion, although it is reasonable to expect significantly more background pages to be mistakenly classified as monitored. We identified the reason for this observation in the compilation of websites used in the WANG14 dataset. This set contains multiple pairs of sites that are very similar, for instance, two different localizations of Yahoo[19]. Obviously, such similarities confuse any classification approach and it is a subject for debate to label such sites as one class.

[19] https://cs.uwaterloo.ca/~t55wang/knnsitelist.txt

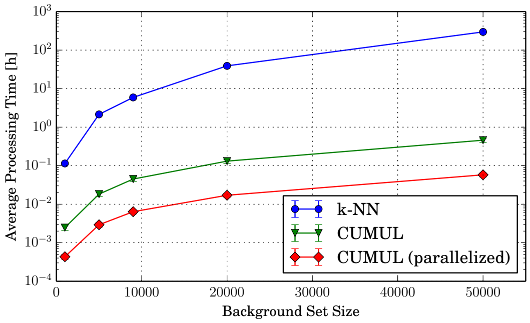


Fig. 6: Comparison of runtimes for the different approaches (y-axis scaled logarithmically)

Besides the detection accuracy we also compared the computational performance in terms of runtimes for our novel approach and the k-NN. To do this, we selected 100 foreground pages at random from the `RND-WWW` dataset and performed an open-world evaluation using both approaches under identical settings (same machine, same cross-validation methodology). Figure 6 shows the average computation times required by each method for varying background set sizes. Note that libSVM has a buit-in parallelization functionality while the implementation of Wang et al. does not support parallelization. Therefore, we evaluated our approach with and without this option enabled. The results are unambiguous: our approach is faster by several orders of magnitude (note that the y-axis is scaled logarithmically) and scales significantly better for increasing dataset sizes. Moreover, the performance of our approach can additionally be increased by enabling parallelization (particularly for optimizing the kernel parameters) and the runtime of our method could be further improved without significant loss of accuracy, by reducing the feature sampling frequency, see Section VII-B. We assume that the computational costs for the k-NN classification (in particular, the method used to adjust the weights of the distance function) increase faster with a growing number of instances. Due to its immense computational costs we have to omit further evaluations on large-scale datasets using this classifier.

Taking all results into account, we conclude that our novel website fingerprinting attack outperforms all previously proposed methods both in the closed- and the open-world scenario, while obtaining such results with significantly less computational effort and, thus, much faster. Hence, we are equipped with the best website fingerprinting technique known to date that we now use to investigate how severe the WFP attack is in reality.

D. Webpage Fingerprinting at Internet Scale

We now address the fundamental question whether the website fingerprinting attack scales when applied in a realistic setting, i.e., whether it is possible to detect a single webpage in real-world Internet traffic in general and in Tor traffic in particular. To do this, we apply and optimize a separate classifier for each page in the foreground class, i.e., the set of pages to be monitored. We argue, and our experiments confirm this claim, that this is the dominant strategy for an attacker in order to increase the probability for success. We later discuss the implications of this strategy for the scenario where the adversary aims to monitor multiple web pages. We concentrated on the open-world scenario, where the universe of web pages a user may visit is not artificially restricted to a small set. Therefore, this evaluation allows us to investigate whether our novel attack allows an attacker to fingerprint the load behavior of a single page within realistic Internet noise.

We selected all 1,125 available foreground pages in `RND-WWW`. To evaluate whether the attack scales, we investigated increasing sizes of the background class. Concretely, we evaluate the attack for a varying size b of background pages, with $b \in \{1000, 5000, 9000, 20000, 50000, \text{MAX}\}$, where MAX corresponds to the maximum available background set size. Further, we evaluated two scenarios. Assume the classifier is trained on a particular web page w of our foreground set, where w is not contained in the background class. We then considered two types of background traffic:

- **unfiltered**: the background class remains unchanged. The question we address here is whether it is possible to monitor an arbitrary webpage in general.
- **filtered**: all other web pages w', which belong to the same web site as w, are removed from the background set. This is to get an intuition for the upper bound of the detection efficacy.

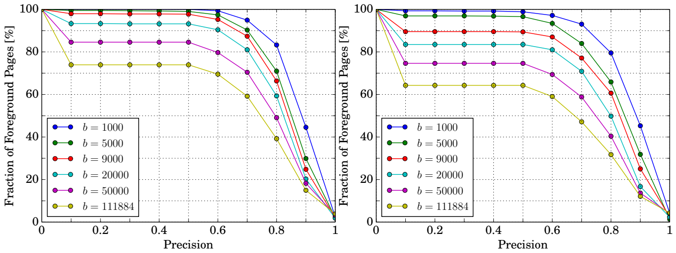
The difference in these scenarios is whether other web pages that belong to the same website are treated as false positives or not. The filtered scenario provides an upper bound to the detection efficacy, because it assumes that users do not visit other pages of the monitored site. The unfiltered scenario shows whether a particular page of a site can be monitored while users may visit other (unmonitored) pages of that site. The unfiltered scenario is more difficult in practice as it is to be expected that different pages of the same site exhibit a similar load pattern and, thus, confuse the fingerprinting method. It follows that in the filtered scenario we have to compile a different background set for each considered page of the foreground class, since different instances must be removed. Therefore, the size MAX may vary slightly throughout our evaluation.

The accuracy, i.e., the probability of a true result (either true positive or true negative), cannot serve as indicator of the adversary's success in practice, since the sizes of the foreground and the background class are heavily unbalanced. We illustrate this with an intuitive example. Assume we train the classifier for one foreground page (i.e., using 40 instances) and use a background class consisting of 1,000 pages (with one instance each). In this case, it is trivial to define a classifier that achieves an accuracy above 96%: a classifier that rejects *any* instance, i.e., which classifies any given instance as background, classifies 1,000 out of 1,040 cases correctly, i.e., 96.15%. This effect becomes even more pronounced when the size of the background class is increased, e.g., to 99.9% for 50,000 background instances. Therefore, we use two metrics that are commonly applied in similar domains: *precision* and *recall*. The recall[20] corresponds to the probability that access to a monitored page is detected. In the related work, the quality of a website fingerprinting technique had mostly been evaluated

[20] The recall is mathematically equivalent to the True Positive Rate.

using only the True Postive Rate and False Positive Rate. These metrics are at first glance intuitive as they express both the fraction of accesses to monitored pages that were detected (TPR) and the probability of false alarms (FPR). However, a low FPR leads to incorrect interpretations if the *prior*, i.e., the fraction of monitored pages within the total number of visited pages, is not taken into account. This effect is known as *base rate fallacy*. Recent works [4], [14] started to consider this fact.

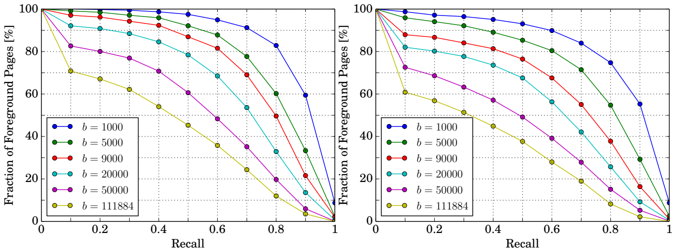
The precision is defined as the number of true positives divided by the number of positive test outcomes. This metric takes account of the prior and the actual size of the universe. It corresponds to the probability that a classifier is actually correct in its decision when it claims to have detected a monitored page. Which metric is more important depends on the objective of the adversary. If the adversary wants to uniquely identify the user that has visited a particular monitored web page, then the precision is more important, because otherwise many innocent users may be suspected. If the primary goal is to restrict the set of users to those that *may* have visited monitored web pages, then the recall is more important, because the probability that accesses to monitored pages are detected at all is more important than the number of false alarms. Ideally, from the adversary's perspective, precision and recall both should be equal or close to one. In this case, he ensures that all users visiting monitored pages are detected and the detection is practically always correct. We calculated these two metrics for both scenarios (filtered and unfiltered) and each web page considered as being monitored, i.e., each page contained in the foreground class. We repeated this calculation for different values of b as described above and used 10-fold cross-validation in each run. The results are shown in Figures 7 and 8, visualized using complementary cumulative distribution functions[21] (CCDFs). We can see that both metrics, precision and recall, clearly decrease for increasing background set sizes. In the filtered scenario, more than 80% of the foreground pages are detectable with a precision greater than 0.8 if the considered universe is small, i.e., $b = 1,000$. However, only 40% achieve at least this precision when the background class is extended to the maximum number available. Moreover, for fewer than 20% of the pages does the classifier achieve a precision greater than 0.9.



(a) filtered

(b) unfiltered

Fig. 7: `RND-WWW`: CCDF of precision for increasing background set sizes



(a) filtered

(b) unfiltered

Fig. 8: `RND-WWW`: CCDF of recall for increasing background set sizes

For the recall we make similar, yet even more remarkable observations. If we assume an attacker whose primary objective is to cover as many visits to monitored pages as possible, hence, who is interested in a high recall, and let the threshold of the recall, which identifies a foreground page as being 'detectable' (ignoring false alarms) be 0.5, i.e., a page is assumed to be detectable if the classifier detects access in at least half the cases. Such an adversary is able to detect almost each web page if b =1,000 but only less than 50% of the pages for b =111,884. If we further assume a page to be 'reliably detectable' if the recall is greater than 0.9, then the attacker is still able to reliably detect 60% of the pages for b =1,000. However, if b is increased to the maximum value in our evaluation, the rate of reliably detectable pages drops below 5%. What is more, recall that this scenario is simplified to the benefit of the attacker and is to provide an intuition for the upper bound, as it assumes the background to be filtered, which is not possible in practice.

In general, our assumptions regarding the more realistic unfiltered scenario are confirmed. When other pages of the website that the considered foreground page belongs to, are to be classified as true negatives, precision and recall decrease further. Obviously, in this case the classifier mistakenly confuses more monitored pages with noise and vice versa.

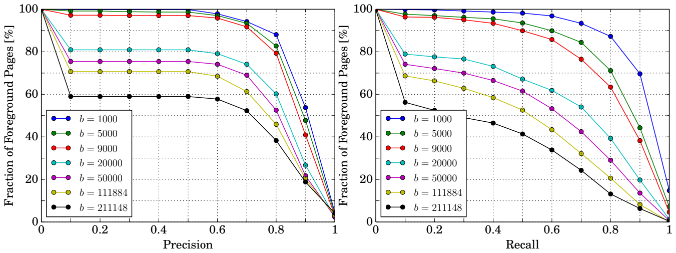


Fig. 9: `TOR-Exit`: CCDF of precision and recall for increasing background set sizes

We performed the open-world evaluation also using the `TOR-Exit` dataset as background noise in order to find out whether there is a fundamental difference in the results when the sets of background pages are compiled from pages that are actually visited using Tor instead of being randomly chosen or particularly popular. The results are shown in Figure 9. This dataset is considerably larger, leading to MAX >

[21] In a CCDF, a point (x, y) expresses that a fraction of y of observations was found to have a value greater than or equal to x.

200,000. Consequently, the required computation time for each foreground page and background set size is increased. To keep the overall computation feasible, we limited the number of foreground pages to 850. Thus, the fractions calculated for the CCDFs are not directly comparable to those in Figure 7. However, the general trend in the results remains unchanged. The fraction of foreground pages for a fixed value of precision and recall steadily decreases with increasing background sizes.

In summary, we see that for each step of increasing the number of background pages, i.e., the size of the considered universe, both precision and recall decline. Hence, it is to be assumed that this trend continues for further increases in universe size. Taking into account that MAX is still vanishingly low compared to the number of pages in the world wide web, we can conclude that the described attack *does not scale*. Recall that this evaluation is still overestimating the success probability of a real-world adversary, because due to using 10-fold cross-validation, we conceded him to train the classifier using 90% of the entire universe. In practice, this is not possible, e.g., for the world wide web and, thus, only a small subset can be selected for training. To investigate this, we experimentally fixed a subset of 5,000 pages (a number that is assumed to be closely sufficient in the related work [27]) for training and only increased the size of the set used for testing.

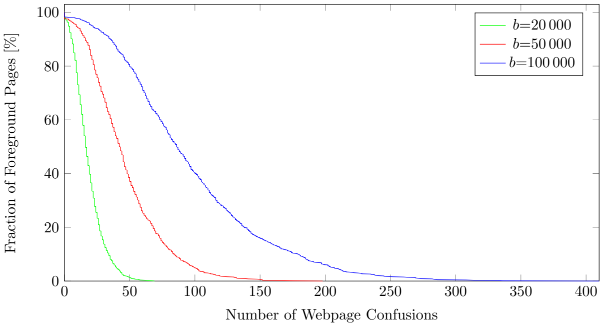


Fig. 10: CCDF of the minimum number of background pages that are mistakenly confused with the considered foreground page for different background set sizes. The number of training samples is fixed to 5,000.

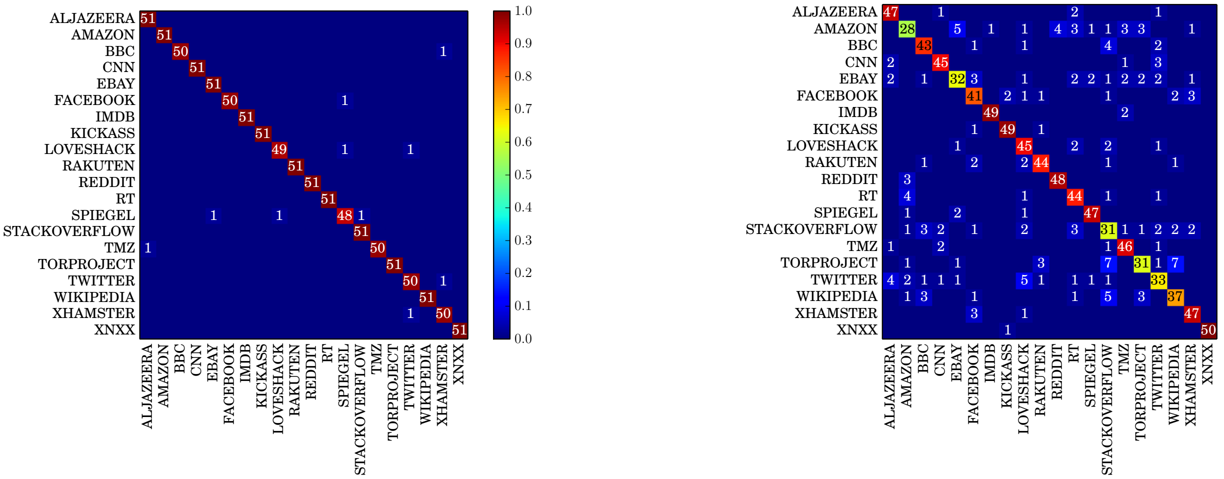
Obviously, this reduces the classification accuracy. To give an impression of the practical implications, in Figure 10 we show the absolute minimum number of background pages that are confused with the considered foreground page. Here, we used 1,000 pages from RND-WWW as foreground and the plot shows the CCDF for that fold of a 4-fold cross-validation, that yielded the minimum number of confusions (i.e., the best result from the adversary's perspective). 2% of this foreground set, i.e. 20 web pages, for $b = 20,000$ (i.e., in dimensions of universe size considered in the related work), do not have a single confusion. But if b is increased to 100,000, *each* of the 1,000 considered foreground pages is mixed up with at least 8 pages in the background. This demystifies the assumption, that there may be web pages that are particularly easy to fingerprint: as we show there is not a single page for which no confusingly similar page exists in a realistic universe.

E. Detection of Websites

We have shown in the previous section that fingerprinting a single web page is not feasible. We now investigate another, more realistic attack scenario where the adversary aims to monitor a complete website. In this context the strategy described to be optimal in the webpage scenario above would be disadvantageous, since training a separate classifier for each page of a site dramatically increases the number of false positives to be expected (because the classification consists of multiple decisions while a single false decision is enough to confuse two websites). In general a web page may only be altered due to dynamical content. It has already been shown by Juarez et al. [14] that page updates have a profound impact on the classification accuracy. Websites, however, can additionally change due to adding new pages or removing existing pages. Besides, for many websites it is practically infeasible to fingerprint each page contained due to their enormous amount, e.g., facebook.com. Hence, the objective to create a website classifier is more challenging. Therefore, we now analyze which attack strategies an adversary may follow and analyze their efficacy. To be realistic in this regard, we only concede the attacker to use a subset of available pages for training of a website.

First, we investigate the most simple scenario to get an intuition about the complexity of the problem in relation to a webpage classifier. To do this, we performed the following experiment. In two disjoint closed-world settings, we aimed to differentiate between 20 websites in our WEBSITES dataset. In case (a), a website is only represented by multiple instances of its index page as it is typically evaluated in the related work. This corresponds to the problem of classifying web pages. In case (b), a site is given by a subset of its webpages. In both cases we used 51 instances per class (website), i.e., all available instances of the index page in case (a) and 51 different other non-index pages in one instance each in case (b). Figure 11 shows the confusion matrices for both cases using heatmaps. As we can see, websites can be "perfectly" separated based on their index page (accuracy: 99%). Contrary, the classification based on a subset of webpages is much less accurate even in such a tiny closed-world setting (accuracy: 82%).

We now consider different attack strategies an attacker may apply to improve the website classification accuracy. The first strategy assumes that the index page is particularly characteristic for the whole website. Therefore, we included it (in 20 instances) in the training set of the classifier in case (b). However, the assumption turned out to be false: the presence or absence of the index page during the training does not have impact on the resulting accuracy. Second, having only one instance per non-index webpage may not be enough and thus could deteriorate the resulting accuracy. However, even using 15 instances for each of the 50 webpages per site did not improve the classification significantly (accuracy: 85.99%). The third strategy does not only consider one class per website, but instead classifies each page within a site seperately to take accout of their diversity. Then, confusion within the classes representing different webpages of the same website is ignored and counted as true positive. The classifier built according to this strategy yielded even a slightly worse result (accuracy: 82.01% vs. 85.99%) than the one with one class per website. We assume that this happens because of overfitting,



(a) Only index pages per site

(b) Different webpages per site

Fig. 11: Closed-world website classifier: confusion matrices for different scenarios

the classifier becomes trimmed to detect single webpages it has already seen and not to generalize characteristics of the website.

To sum up, website classification in a closed-world scenario is significantly more difficult compared to index page classification as it is typically performed in the related work. In reality, it is to be expected that for certain websites the adversary is not able to train on all sub-pages due to their number, similar to the case that he cannot train a webpage classifier on the whole universe. We experimented reducing the number of pages available for training to 20 and tested against the remaining 30 pages. As expected, the accuracy degraded to 69.65%. None of our evaluated strategies improved the probability of success. However, the results do not indicate that website fingerprinting is generally infeasible (since several websites, e.g., KICKASS or XNXX are reliably detectable in this setting, see Figure 11).

Moreover, the transition of results obtained in closed-world to the realistic open-world setting is typically not trivial. We evaluated website fingerprinting in the open world, using the WEBSITES dataset as the foreground and RND-WWW as the background. Figure 12 shows the average precision and recall (together with 95% confidence intervals) for increasing background set sizes. For comparison, we derived the same visualization also for the (unfiltered) webpage fingerprinting scenario (i.e., from the data shown in Figures 7b and 8b). As the results indicate, website fingerprinting scales better than web page fingerprinting, although also for this objective precision and recall decline for increasing background set sizes. However, it appears that the precision stabilizes on a high level in the case of website classification. Although the results obtained in closed-world settings suggested that website classification is less promising than webpage classification (here represented by index-page classification), the results in the open world reveal the contrary. This also substantiates our assumption that closed-world results cannot be trivially generalized. In summary, to optimize his probability to succeed in website fingerprinting, the attacker should crawl many different pages of a site in favor of crawling many instances per page or of overestimating the importance of the index page.

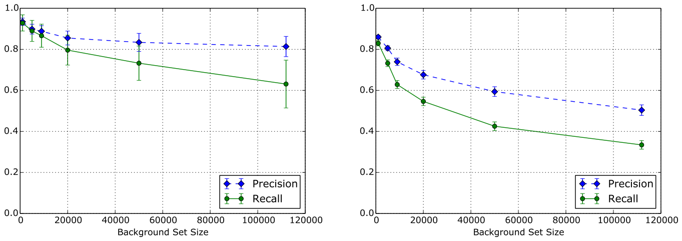


Fig. 12: RND-WWW: precision and recall for increasing background set sizes, classification of websites (left-hand) vs. webpages (right-hand)

VIII. Conclusion and Future Work

In this paper we proposed a novel website fingerprinting approach, which we showed to be superior both in terms of detection accuracy and computational efficiency. For a comprehensive evaluation of attacks in this domain we compiled the most representative large-scale datasets of webpages that are actually retrieved on the Internet known to date. This allows for the first time the evaluation of the WFP attack against Tor using realistic background noise. By publishing our datasets and tools we want to advance the ongoing research and discussion in this important field. We are the first to evaluate the WFP attack at Internet scale by avoiding the simplification made in the related work that particularly background traffic only consists of the transmission of index pages of popular websites. As we showed, webpage fingerprinting does not scale for any considered page in our datasets and any state-of-the-art classifier. Hence, the attack cannot be reliably used to convict users, but it may be used to limit the set of possible suspects.

The attack scenario to fingerprint websites, i.e., a collection of webpages served under the same domain, is not only more realistic but also significantly more effective using our attack method. We investigated several strategies to improve the success probability and emphasized the most promising tactics. Using our realistic datasets, a fact to be considered in future work is that users often remain on a website, i.e., they retrieve multiple pages of that site consecutively, e.g., by following links. We assume, that exploiting this information can further increase the adversary's confidence.

Acknowledgements

The authors would like to thank Norbert Landa and Robert Echelmeyer for their support while performing some of the experiments for this paper. Parts of this work have been funded by the EU H2020 Project "Privacy Flag", the Luxembourg National Research Fund (FNR), and the Excellence Initiative of the German federal and state governments.

References

[1] M. Baron, *Probability and Statistics for Computer Scientists*, Chapman and Hall, Eds. CRC Press, 2007.

[2] O. Berthold, H. Federrath, and S. Köpsell, "Web mixes: A system for anonymous and unobservable internet access," in *Proceedings of Designing PETs: Workshop on Design Issues in Anonymity and Unobservability*, July 2000.

[3] G. D. Bissias, M. Liberatore, D. Jensen, and B. N. Levine, "Privacy vulnerabilities in encrypted http streams," in *Proceedings of Workshop on PETs*, Dubrovnik, Croatia, May 2005.

[4] X. Cai, R. Nithyanand, T. Wang, R. Johnson, and I. Goldberg, "A systematic approach to developing and evaluating website fingerprinting defenses," in *Proceedings of ACM CCS*, Scottsdale, AZ, USA, 2014.

[5] X. Cai, X. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proceedings of ACM CCS*, Raleigh, NC, USA, October 2012.

[6] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, 2011, available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[7] H. Cheng and R. Avnur, *Traffic Analysis of SSL Encrypted Web Browsing*, Project paper, University of Berkeley, 1998. [Online]. Available: http://www.cs.berkeley.edu/~daw/teaching/cs261-f98/projects/final-reports/ronathan-heyning.ps

[8] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of USENIX Security*. San Diego, CA, USA: USENIX Association, 2004.

[9] K. P. Dyer, S. Coull, T. Ristenpart, and T. Shrimpton, "Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail," in *Proceedings of IEEE S&P*, San Francisco, CA, USA, 2012.

[10] X. Gong, N. Borisov, N. Kiyavash, and N. Schear, "Website detection using remote traffic analysis," in *Proceedings of PETS*. Vigo, Spain: Springer, July 2012.

[11] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: Attacking popular privacy enhancing technologies with the multinomial naïve-bayes classifier," in *Proceedings of the ACM CCS Workshop on Cloud Computing Security*. Chicago, IL, USA: ACM Press, 2009.

[12] A. Hintz, "Fingerprinting websites using traffic analysis," in *Proceedings of PETS*, 2002.

[13] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, *A Practical Guide to Support Vector Classification*, National Taiwan University, 2010. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

[14] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A critical evaluation of website fingerprinting attacks," in *Proceedings of ACM CCS*. Scottsdale, Arizona, USA: ACM Press, 2014.

[15] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: Passive deanonymization of tor hidden services," in *Proceedings of USENIX Security*, Washington, D.C., 2015.

[16] M. Liberatore and B. N. Levine, "Inferring the source of encrypted http connections," in *ACM CCS*, Alexandria, VA, USA, October 2006.

[17] K. Loesing, S. J. Murdoch, and R. Dingledine, "A case study on measuring statistical data in the Tor anonymity network," in *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, ser. LNCS. Springer, January 2010.

[18] L. Lu, E.-C. Chang, and M. Chan, "Website fingerprinting and identification using ordered feature sequences," in *Proceedings of ESORICS*, Athens, Greece, September 2010.

[19] X. Luo, P. Zhou, E. W. W. Chan, W. Lee, R. K. C. Chang, and R. Perdisci, "Httpos: Sealing information leaks with browser-side obfuscation of encrypted flows," in *Proceedings of NDSS*, San Diego, CA, USA, February 2011.

[20] R. Nithyanand, X. Cai, and R. Johnson, "Glove: A bespoke website fingerprinting defense," in *Proceedings of ACM WPES*. Scottsdale, Arizona, USA: ACM Press, 2014.

[21] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proceedings of ACM WPES*. Chicago, IL, USA: ACM Press, October 2011.

[22] M. Perry, "Experimental Defense for Website Traffic Fingerprinting," https://blog.torproject.org/blog/experimental-defense-website-traffic-fingerprinting, 2011.

[23] Q. Sun, D. Simon, Y.-M. Wang, W. Russell, V. Padmanabhan, and L. Qiu, "Statistical identification of encrypted web browsing traffic," in *Proceedings of IEEE S&P*. Oakland, CA, USA: IEEE, May 2002.

[24] D. Wagner and B. Schneier, "Analysis of the SSL 3.0 protocol," in *Proceedings of the 2nd USENIX Workshop on Electronic Commerce (EC-96)*. USENIX Association, November 1996.

[25] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg, "Effective attacks and provable defenses for website fingerprinting," in *Proceedings of USENIX Security*, San Diego, CA, USA, August 2014.

[26] T. Wang and I. Goldberg, "Improved website fingerprinting on tor," in *Proceedings of ACM WPES*, Berlin, Germany, November 2013.

[27] ——, "On realistically attacking tor with website fingerprinting," University of Waterloo, Tech. Rep., 2015.

[28] ——, "Walkie-talkie: An effective and efficient defense against website fingerprinting," University of Waterloo, Tech. Rep., 2015.

[29] C. V. Wright, L. Ballard, S. Coull, F. Monrose, and G. Masson, "Spot me if you can: Uncovering spoken phrases in encrypted voip conversations," in *Proceedings of IEEE S&P*, Oakland, California, USA, May 2008.

[30] C. V. Wright, L. Ballard, F. Monrose, and G. M. Masson, "Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob?" in *Proceedings of USENIX Security*, Boston, MA, USA, August 2007.

[31] C. V. Wright, S. E. Coull, and F. Monrose, "Traffic morphing: An efficient defense against statistical traffic analysis," in *Proceedings of NDSS*, San Diego, CA, USA, 2009.

Appendix

```
http://www.aljazeera.net/       Arabic news site
http://www.amazon.com/          Retailer
http://www.bbc.co.uk/           British news site
http://cnn.com/                 American news site
http://www.ebay.com/            Online auction platform
http://www.facebook.com/        Social website
http://www.imdb.com/            Online database
http://kickass.to/              Torrents
http://www.loveshack.org/       Dating board
http://www.rakuten.co.jp/       Japanese retailer
http://www.reddit.com/          Entertainment, social news
http://rt.com/                  Russian news site
http://www.spiegel.de/          German news site
http://stackoverflow.com/       Knowledge market
http://www.tmz.com/             Celebrity news
http://www.torproject.org/      Online Anonymity
http://twitter.com/             Microblogging
http://en.wikipedia.org/        Internet encyclopedia
http://xhamster.com/            Adult content
http://xnxx.com/                Adult content
```

Listing 1: Sites included in the `WEBSITES` dataset