Criteria and Analysis for Human-Centered Browser Fingerprinting Countermeasures

Vafa Andalibi
Indiana University Bloomington
vafandal@indiana.edu

Erfan Sadeqi Azer
Indiana University Bloomington
esadeqia@alumni.iu.edu

L. Jean Camp
Indiana University Bloomington
ljcamp@indiana.edu

Abstract

Browser fingerprinting is a surveillance technique that uses browser and device attributes to track visitors across the web. Defeating fingerprinting requires blocking attribute information or spoofing attributes, which can result in loss of functionality. To address the challenge of escaping surveillance while obtaining functionality, we identify six design criteria for an ideal spoofing system. We present three fingerprint generation algorithms as well as a baseline algorithm that simply samples a dataset of fingerprints. For each algorithm, we identify trade-offs among the criteria: distinguishability from a non-spoofed fingerprint, uniqueness, size of the anonymity set, efficient generation, loss of web functionality, and whether or not the algorithm protects the confidentiality of the underlying dataset. We report on a series of experiments illustrating that the use of our Partially-Dependent *algorithm for spoofing fingerprints will avoid detection by machine learning approaches to surveillance.*

1. Introduction

In this work, we contribute three new algorithms for spoofing fingerprints as well as an empirical analysis for evaluation of their anonymity and efficiency as browser fingerprinting countermeasures. We map this evaluation to the motivations for fingerprinting spoofing taken from previous research on usability and acceptability of Privacy Enhancing Technologies (PETs). We illustrate how different instantiations of these algorithms vary in terms of how well they address user motivations for avoiding fingerprinting. We specifically examine the fingerprint generators in terms of the potential deterioration of site functionality. Using the distribution of results from the models and an existing dataset of fingerprints, we evaluate the implementations in terms of different threat models. We include a straightforward sampling from the fingerprint dataset as a baseline in our analysis. Specifically, our contributions are as follows:

Enumeration of the (sometimes implicit) requirements and threat models from different fingerprint spoofing approaches, and the design criteria resulting from those threat models.

Design of three new algorithms that generate spoofed fingerprints to address these criteria with four variations for each algorithm.

Definition of general comparison metrics based on the enumerated criteria.

Illustration of the usage of the introduced metrics in evaluating the new algorithms, and their general applicability by comparing our algorithms to a baseline sampling approach.

Evaluating the strength of the new algorithms in the context of multiple empirical experiments, specifically, generating fingerprints and distinguishing a spoofed fingerprint from genuine ones.

In addition to these contributions, we will also make the code repository publicly available for ease of reproducibility of our analysis.

2. Motivation

Web-based fingerprinting identifies the visitors by leveraging the variability of machines, configuration data, and browser parameters. Although there are benign applications of these passive tracking techniques [1], some web service providers use browser fingerprinting for the purpose of user tracking [2]. An analysis of the top million websites found fingerprinting was extremely common with more than 80,000 entities engaged in tracking [3].

The desire to avoid ubiquitous tracking and information sharing is indicated by the widespread adoption of ad-blocking and anti-tracking software [4]. This, in turn, has resulted in anti-ad-blocking activities [5] including the detection of ad-blockers for denial of access and changing domains to avoid ad-blocking rules [4]. In addition to direct denials of service, protecting against fingerprinting with some tools has a cost that

URI: https://hdl.handle.net/10125/80257
978-0-9981331-5-7
(CC BY-NC-ND 4.0)

includes loss of specific desired functionality.

It would be reasonable to expect anti-fingerprinting technology to have popularity similar to that of ad-blocking, but that has not yet been observed. Perhaps it is because current measures are not scalable and the very tools distributed to mitigate fingerprinting may create an additional threat by identifying an individual as one seeking privacy, thus risking that they are subjected to more targeted surveillance [6].

Previous research in usable security has found that participants choose not to use privacy-enhancing software in general because of the loss of functionality [7]. Users may want the benefits of anti-fingerprinting technologies, but the loss of functionality is too high a cost for the corresponding privacy benefits. In that case, preventing the functionality loss may broaden the use of privacy-enhancing software [8, 9]. Finally, the work by Juarez et al. [10] on identifying the vulnerability of fingerprinting by local adversaries enumerates the requirements for the availability of data for attackers.

3. Related Work

In the previous section, we identified primarily motivating research. In this section, we focus on the technical research that informs our approach to spoofing algorithms.

Panopticlick project was the first impactful research about browser fingerprinting [11, 12]. Panopticlick illustrated the scope and potential of the threat of fingerprinting. Following up on this, Nikiforakis et al. [13] examined the code from three popular browser fingerprinting code providers used for web-based device fingerprinting and showed how browsers are vulnerable to fingerprinting using novel browser identifying techniques. Kaur et al. [14] analyzed the evolution of browser fingerprinting and its effect on users' privacy.

Al-Fannah and Li [15] performed an analysis on the fingerprinting vulnerability of modern browsers. They illustrated that browsers installed in default modes provide very different levels of susceptibility to fingerprinting and identified the features underlying this difference. In the following year, Yan and Kaur [16] implemented an exhaustive analysis of features that can be used in the TCP/IP connection to fingerprint a user. The sets of features identified in these analyses informed our own feature selection.

Gómez-Boix et al. [17] analyzed the uniqueness of fingerprints in their dataset of just over 2 million fingerprints gathered from 15 French websites, arguing that only 33.6% of the fingerprints are unique. Tanabe et al. [18] studied the effect of attribute combination on fingerprinting accuracy and found that the addition of features could increase attacker accuracy.

4. Goal and Contributions

Our goal is to design a fingerprinting defense that addresses some of the reasons users may have for rejecting or not adopting anti-fingerprinting tools. To build on previous research, our algorithms are designed to address the factors that the PET community has identified as underlying reasons for the lack of adoption of PETs. Following the terminology in Table 1, we detail our contributions as follows:

First, we clarify the problem statement by systematizing the requirements of an ideal fingerprint spoofing system and summarizing them as six main characteristics: undetectability, guaranteed non-uniqueness, increased k-anonymity, dataset privacy, efficiency, and least burden.

Second, we define a spectrum on which fingerprint spoofing models follow a trade-off between the aforementioned characteristics. We then create three probability models on that spectrum. On one end of the spectrum, we offer the Independent model which is detectable but efficient and respects the dataset privacy. On the other end, we create the Fully-Dependent model, which is undetectable and guarantees 2-anonymity (as defined in Table 1) but is not efficient and does not preserve the dataset privacy. The fingerprints generated by Fully-Dependent model are also *valid* and indistinguishable from the *genuine* fingerprints. We do this since there are jurisdictions where using PETs risks creating additional surveillance [19, 20], and detection in such jurisdictions can also lead to blocking of users or resources [21]. Additionally, we present a third model in the middle of the spectrum, namely Partially-Dependent, that is undetectable and preserves the dataset privacy while being sufficiently efficient.

Third, we examine four variations for each of these models, i.e., Greedy-Constrained, Greedy-Unconstrained, Random-Constrained, and Random-Unconstrained, that have different applications. A Greedy model generates fingerprints that are shared between multiple users. The approach of maximizing crowd size by simply setting a single shared fingerprint in a group of users to increase k-anonymity is the approach used by Tor [22]. A Constrained model sets an initial attribute to its correct values (e.g., language, location for maps, resolution for streaming) and constructs a spoofed fingerprint around that. Ideally, a model generating Constrained fingerprints reduces the cost of using this obfuscating technology by minimizing the disruption of services provided by the website. In the case of the Greedy variation (i.e. Greedy-Constrained),

Table 1: Definitions for the terminology used in this work

Term	Definition
Fingerprint	A combination of characteristics for a browser, used for identifying or tracking a user.
Paradoxical Fingerprint	A fingerprint that could not technologically exist, e.g., a fingerprint of an iPhone that claims a 36-inch screen.
Valid Fingerprint	A fingerprint that is not paradoxical.
Genuine Fingerprint	A valid fingerprint that is not tampered with or altered in any way.
Spoofed Fingerprint	A fingerprint that has been altered by either a user or a program.
Attribute-Value Pair	A key-value pair in which the key represents an attribute used in the fingerprint spoofing system and the value represents a value taken by that attribute, e.g., {"video":"1920x1080x24"}.
Spoofing System	A generative model that outputs fingerprints intended to obfuscate the user.
[Spoofing] Model	A mathematical representation of the probability distributions over the space of fingerprints.
Evaluator	A computational model that estimates the occurrence probability of a given fingerprint.
Generator	A computational routine that generates a fingerprint based on a model.
Dataset-Sampler	A generator that draws a uniform sample from a fingerprint dataset.
2-Anonymity	A generator is said to guarantee 2-anonymity if any fingerprint it generates is used by one or more other users, hence the anonymity set is an integer greater than 1 (i.e., 2).

users can also hide in the crowd by assigning the same spoofed fingerprint. The model may also generate a spoofed fingerprint using the already set attributes by selecting random values for other attributes (i.e., Random-Constrained). Alternatively, the model can output a stochastic spoofed fingerprint with no constraint (i.e., Random-Unconstrained) or a fingerprint with the largest possible crowd size (i.e, Greedy-Unconstrained) without consideration of functionality.

Fourth, based on the characteristics we defined for an ideal spoofing model, we propose comparison metrics and illustrate their use in evaluating our models in the context of the following experiments: generating spoofed fingerprints, distinguishing a valid fingerprint from a spoofed one in a pair, and detecting the spoofed fingerprints in a set of fingerprints.

An overall goal of our design and implementation is to move the frontier on the trade-off between privacy and functionality in fingerprint spoofing.

5. Problem Statement

In response to the prevalence of browser fingerprinting (Section 2) and using the definitions given in Table 1, we define a *fingerprint modeling system* called FPModeler as a generative model with optional evaluation elements. When generating, FPModeler can operate in two modes: 1) *Unconstrained*, in which the system generates a random but valid fingerprint, and 2) *Constrained*, where given an input in the form of an attribute and corresponding value as a constraint, the system outputs a fingerprint that is random, technically valid, and with selected attribute(s) set as desired. Next, we detail the evaluation metrics for FPModeler.

5.1. Design Criteria and Evaluation Metrics

Building on the motivation presented in Section 2, in this section we define the criteria we used to evaluate variations of FPModeler.

Previous research on browsing illustrated that a badly designed obfuscating algorithm, which generated invalid data actually caused the machine to be highly identifiable, decreasing rather than increasing anonymity as the attributes were never similar to the genuine records in the dataset and always technically invalid [23, 24]. The result is a requirement for FPModeler to generate spoofed fingerprints that are individually **indistinguishable** from a genuine fingerprint. The ability of FPModeler to meet this requirement contributes to multiple criteria: The first is ensuring that the distribution of the generated fingerprints is similar to that of the genuine fingerprints so that the use of FPModeler is **undetectable**. Second, generated spoofed fingerprints should not be unique, i.e., **Non-Uniqueness**. The third criterion extends this, where we evaluate the size of the expected anonymity set, i.e. **K-anonymity**.

There is a potential positive secondary effect

on everyone's privacy that results from spoofed fingerprints being indistinguishable from genuine samples. In general, anonymity loves company [6]. In this specific case, consider a tracking service that continuously improves by observing browser fingerprints. The possibility of filtering out the spoofed fingerprints increases the accuracy of such a tracking system, making it more privacy-invasive. Conversely, contaminating the system's observed fingerprints with indistinguishable spoofed fingerprints could reduce the accuracy of such a tracking system [25].

A spoofing system should also be **efficient**, meaning that runtime requirements for processing power and delay are minimal. FPModeler should be sufficiently efficient to allow frequent generation of distinct unlinkable spoofed fingerprints.

A system should also require the user to accept the **least burden** possible. That is, when there is a core functionality of a website, the spoofed fingerprint should not interfere with that functionality.

For a single user to remain obfuscated, we require information about distributions of attributes, and that requires a large dataset of fingerprints. Such a dataset perversely generates its own inherent privacy concerns [26, 23]. The dataset itself should be protected to ensure the privacy of the users whose data it contains. The differences between the fingerprints in the underlying dataset and that of the visitors of the tracking websites enable the trackers to identify the users of FPModeler. Such identification would subvert FPModeler user anonymity by undermining indistinguishability. Therefore, another goal is to ensure **Dataset Privacy**.

In summary, the criteria for evaluation of FPModeler are as follows:

1. **Indistinguishability:**

   Undetectability: determines the similarity between the distribution of FPModeler's generated fingerprints and the distribution of genuine fingerprints.

   Guaranteed Non-Uniqueness: makes sure that a spoofed fingerprint is not unique and is taken by at least another user of the system.

   k-anonymity: is a stronger requirement than non-uniqueness when $k > 2$; it requires that each generated fingerprint is part of an anonymity set with a cardinality of at least k. Higher values of k allow the users to hide in a larger crowd. This is the approach taken by the Tor browser where the k-anonymity set is all Tor users.

2. **Efficiency**: specifies the efficiency of the FPModeler regarding runtime and space (memory) usage.

3. **Least burden**: identifies if the FPModeler settings disrupts the functionality of the website.

4. **Dataset Privacy**: indicates how well the FPModeler is concealing information about the dataset.

We compare our models according to these criteria and present our comparison in Section 6.2.

6. Methods

We present three models and a data sampler (used as a baseline), so that we cover the spectrum of models defined between the following two distinct approaches:

Oblivious: in which FPModeler chooses a value for each attribute without any prior information about the joint distribution of attribute-values. **Lossless**: in which the fingerprints *generated* by FPModeler are uniform samples from a pre-constructed dataset.

In the oblivious approach, the spoofing system is highly efficient and protects dataset privacy. The shortcoming of this approach is that the generated fingerprints can be paradoxical and, thus, easily detectable. A lossless approach guarantees the undetectability of the generated fingerprints (assuming that the given dataset accurately represents the distribution of genuine fingerprints). Yet, it is neither efficient nor does it preserve the dataset privacy.

We examine how these approaches offer trade-offs between desirable features. It is possible to define other models and variations but in this paper, we focus on those that are most representative of this spectrum. In the next section, we include three probability models on the spectrum.

6.1. Probability Models

Here we specify multiple probability models located on the spectrum between oblivious and lossless.

Notation. Let F denote a typical fingerprint with n attributes $a_1, \ldots, a_n$. Each attribute a_i can take values from a finite set $\mathcal{A}_i$. Based on this notation, we write $P(F)$ to denote the probability (likelihood) for the fingerprint attributes of an genuine browser taking the values in F. Based on these definitions we have

$$\sum_{v_1 \in \mathcal{A}_1} \cdots \sum_{v_n \in \mathcal{A}_n} P(a_1 = v_1, a_2 = v_2, ..., a_n = v_n) = 1$$

In the following, we present multiple models that provide an estimation for $P(a_1 = v_1, a_2 = v_2, ..., a_n = v_n)$ based on some assumptions and pre-tuned parameters.

Independent This model provides a probability for a fingerprint based on an assumption that the distribution of values for each attribute is independent of other attributes. This assumption simplifies the calculations and results in a low number of parameters. Hence, we have,

$$\begin{aligned} P(F) &= P(a_1 = v_1, a_2 = v_2, ..., a_n = v_n) \\ &= P(a_1 = v_1) \times P(a_2 = v_2) \times \cdots \times P(a_n = v_n) \end{aligned} \tag{1}$$

Notice that this model has $\Pi_{i=1}^{n} |\mathcal{A}_i|$ parameters. Each parameter represents the probability of one attribute-value. One can infer these parameters by processing a given dataset of fingerprints.

Partially-Dependent This model takes the dependence between the pairs of attribute-values into account. In this case, $P(a_1 = v_1 | a_2 = v_2)$ denotes the *conditional* probability of attribute a_1 taking the value v_1 given that a_2 has taken the value v_2. For this model we make the simplifying assumption that this dependence is only for one level, i.e., conditioned on the event $a_1 = v_1$, the events $a_2 = v_2$ and $a_3 = v_3$ are independent. (For example, screen size depends on the device, while operating system and location are independent.) Based on this assumption we have

$$\begin{aligned} P(F) &= P(a_1 = v_1 \mid a_2 = v_2, \ldots, a_n = v_n) \\ &= P(a_1 = v_1) \times P(a_2 = v_2 \mid a_1 = v_1) \\ &\times P(a_3 = v_3 \mid a_2 = v_2) \times \ldots \\ &\times P(a_n = v_n \mid a_{n-1} = v_{n-1}) \end{aligned} \tag{2}$$

Notice that this model contains the parameters corresponding to the attribute-value pairs, e.g., $P(a_i = v_i | a_j = v_j), \forall\, 1 \leq i \neq j \leq n$, as well as the parameters in Independent. One can represent these parameters as a weighted directed graph with attribute-values as nodes and the conditional probabilities as the weights of edges. The weights of nodes also correspond to the $P(a_i = v_i), \forall\, 1 \leq i \leq n$.

When FPModeler assumes limited independence, the result of the calculation does not depend on the order in which the conditional probabilities are multiplied in Equation 2. When this condition is not satisfied, the estimation of $P(F)$ depends on the order of attributes. To achieve a consistent and well-defined estimation function, we choose an arbitrary (but fixed) order and follow it whenever this model is used.

Fully-Dependent This model preserves dependence information between attribute-values and store all non-zero probabilities of fingerprints. Here, we describe a tree data structure realizing the requirement. Fix an arbitrary order on the attributes, say $a_1, a_2, \ldots, a_n$. We label the nodes at i^{th} level with a tuple of attribute-values for the first i attributes. Also, we store the number of fingerprints matching the first i attributes to the tuple indicated as the label. Each node in the i^{th} level have, at most, $|\mathcal{A}_{i+1}|$ children in $i+1^{\text{st}}$ level, iterating over all possible values for attribute a_{i+1}. This data structure makes it easy to answer the conditional probability queries like $P(a_i = v_i | a_1 = v_1, a_2 = v_2, \ldots, a_{i-1} = v_{i-1})$ as well as full fingerprint probability queries like $P(a_1 = v_1, a_2 = v_2, \ldots, a_n = v_n)$.

One of the applications of the models presented in this section is the evaluation of the validity of fingerprints. The equations defined for each model to calculate the fingerprint probability $P(F)$ are used as a measure for distinguishing valid and paradoxical fingerprints. This can be achieved by defining a threshold with fingerprints having a probability below that threshold being considered invalid, and at or above it as valid.

6.2. Using the Models in the Generation Tasks

We implemented the models to generate fingerprints with the variations described below. The accuracy and efficiency of the main generative approaches are detailed in Section 8. On the offensive side, after using FPModeler to detect that the value of an attribute has been spoofed, the service provider has multiple options. These options include *(i)* denial of service to the user (as responses to ad-blockers and Tor illustrate); *(ii)* attempting to track by matching the partial fingerprint to a previous fingerprint of the user; or *(iii)* removing the value of the suspicious attribute to impute a new value and then use the updated fingerprint for tracking.

6.3. Variations

For each model, we have considered the following boolean characteristics:

Random or Greedy: A Random generator model selects any non-zero probability so that a large set of fingerprints will have the same distribution of attributes as a large set of fingerprints in the wild. The benefit of this is that the set as a whole is indistinguishable from genuine fingerprints. The disadvantage is that there is a risk of small anonymity sets. A Greedy generator model always selects the most likely attribute. The benefit of this is that there will be the largest anonymity set. The disadvantage is that, over time or based on the underlying data, the generated fingerprints will reach a predictable equilibrium so that any entity using the specific Greedy result could be identified with some probability as using a false fingerprint. Note that

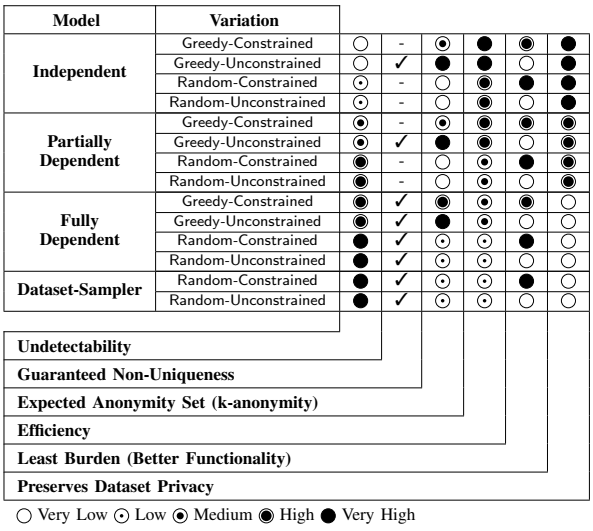


Figure 1: The differences in the metrics across all four possible modes of operation of each of the models presented in this paper.

Dataset-Sampler (as defined in Table 1) cannot have a Greedy variation since it is not a probability model.

Constrained or Unconstrained: A Constrained fingerprint will always have a particular value for an attribute. This is useful when an attribute is inextricably linked with the functionality of a website, e.g., time zone for a calendar website. On the other hand, an Unconstrained generator just randomly generates a fingerprint without focusing on any of the attributes.

Thus, for each of our three models, there are four variation of generators: Greedy-Constrained, Greedy-Unconstrained, Random-Constrained, and Random-Unconstrained. Additionally, we have Random-Constrained and Random-Unconstrained variations of Dataset-Sampler, summing up to 14 generators in total: 12 generative models plus 2 generators based on Dataset-Sampler.

There are differences both among the models and across the variations of each model, as summarized in Fig. 1. All variations of the Fully-Dependent model and Dataset-Sampler guarantee non-uniqueness because their generated fingerprints are in the dataset and, hence, previously used at least once. Among the variations of the Independent and Partially-Dependent models, only the Greedy-Unconstrained guarantees non-uniqueness. This is because the constraint in the Greedy-Constrained variation may be unique to the user. The random variations of each model do not offer guarantees of non-uniqueness.

7. Dataset

We used a large dataset of browser fingerprints created by the Electronic Frontier Foundation (EFF). We particularly focused on a subset of attributes because they are easy to capture. However, our mathematical models are independent of these features: `cookie_enabled` indicating whether a cookie is enabled in the browser; `user_agent`, which is the User-Agent request header; `http_accept`, which contains a list of content types that the browser can accept; `timezone` indicates the timezone of the browser; `video` provides information about screen resolution of the browser; `super_cookies`, which contains information about the super cookies; `dnt_enabled` that indicates whether or not Do Not Track flag is set; `language`, which shows the supported languages of the browser; `platform` indicating the operating system of on which the browser is running; and finally `touch_support`, which contains information about whether or not the device has a touch screen. Two of these attributes, i.e., `user_agent` and `http_accept`, are given as raw strings that hold discrete pieces of information. For example, `user_agent` may have the browser version and OS version embedded in its value. Therefore, we parse the string values and extract some *sub-attributes* as follows: `browser_family`, `browser_version`, `os_family`, `os_version`, and `device_family` from `user_agent` (similarly, `mime_type` and `q_factor` from `http_accept`).

Any recorded dataset of fingerprints is expected to contain some spoofed fingerprints. For our purposes, we need to filter spoofed entries and keep the authentic fingerprints. Our results in Section 8 assume genuine, or at least representative, fingerprints. One of the challenges here is the fact that the unique valid fingerprints (perhaps without even a *similar* fingerprint) in the dataset which may conventionally be considered as outliers should not be filtered. An ideal approach filters out those spoofed fingerprints that are likely paradoxical.

Our first step was to filter out obviously spoofed fingerprints. For example, we deleted all fingerprints corresponding to the Tor browser [22].

For the second step, we used two well-known Machine Learning pipelines based on DBSCAN clustering (the first one with t-SNE dimension reduction and the second one without). These methods use a distance measure between all pairs of fingerprints in the dataset. We considered two

approaches for implementing the distance function. First, we implemented a high-accuracy, fine-grained, computationally expensive distance function which calculates the distance of each attribute separately based on the characteristics of that attribute. Second, we used a simpler approach based on string edit distance algorithms (Levenshtein distance). Edit distance is a method of quantifying the dissimilarity between two strings based on the minimum number of operations needed to transform one into another [27]. The distance between two fingerprints is calculated as the square root of the sum of the normalized Levenshtein distance between each attribute. Distances for an attribute pair were normalized using the maximum distance for that attribute across all evaluated fingerprints. After preparing the distance matrix for the dataset, we used DBSCAN to detect the outliers in the dataset in two ways: *(i)* DBSCAN was applied directly on the distance matrix *(ii)* two t-SNE features were extracted given the distance matrix and then DBSCAN was applied to the fingerprints on the embedded space. Outliers identified using either the fine-grain and dimension-reduced measures of distance were excluded from our models. The identified outliers were few in number (< 10).

8. Results

In the following, we detail the experiments used to evaluate the models and describe the results. These experiments help us assess the strength of suggested generators against the threat of identification of a fingerprint as spoofed or exposure of information from the underlying database.

To perform the experiments in this section, we created two non-intersecting subsets of fingerprints by dividing our dataset into two parts, each comprised of 1 million fingerprints. We refer to these datasets as training and test datasets accordingly. We performed part of our analysis using Receiver Operating Characteristic (ROC) curves. As a gentle reminder, the X and Y axes of the ROC curve correspond to *true positive rate* (i.e., trust positives divided by all positives) and *false-positive rate* (i.e., false positives divided by all positives).

Distinguishing Valid Fingerprint from Spoofed Ones: To compare the performance of the evaluators, we initially selected a sample set of 10k fingerprint pairs (20k in total)[1] where each pair has one valid fingerprint randomly picked from the dataset as well as one spoofed fingerprint generated by the Independent model. The evaluation of a model on a pair is defined as a "success" if the model assigned a higher probability to the valid fingerprint. We report the accuracy as the number of successes over the entire sample size.

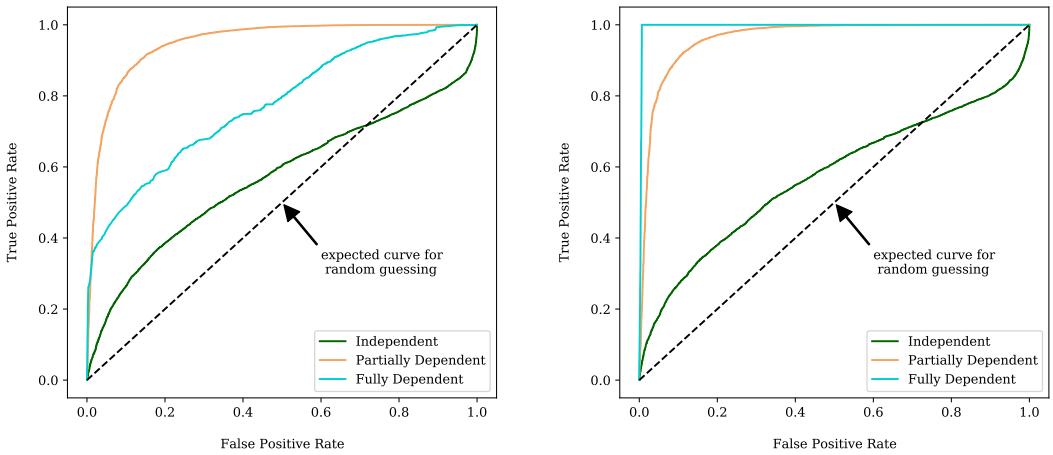
This experiment is done in two scenarios: 1) the training dataset is used in both training the evaluators and constructing the sample pairs; 2) the evaluators are trained on the training dataset, then the test dataset is used for constructing the sample pairs. Our results, as shown in Table 2, confirm that the accuracy of all models is higher in the first scenario compared to the second scenario. This is expected because the valid fingerprints have been shown to the models beforehand. The figure also shows that the Independent model has poor performance with a low accuracy of 56-57% in both scenarios. We also confirm that the Fully-Dependent model has the highest precision (of 99.6%) when the training set is used, i.e., this model is most effective for training. However, its precision drops to 77% when the test dataset is used. This observation reflects that the Fully-Dependent model stores details that are specific to the training dataset but are not effective in distinguishing valid from invalid information with a larger dataset; This phenomenon is referred to as overfitting in machine learning. On the other hand, Partially-Dependent has slightly worse but still good performance on the training dataset, with a much smaller decrease in accuracy ($< 2\%$) when encountering the test dataset.

Classification of Fingerprints: We use the same sample set from the previous experiment here. Recall that the sample set contains 20k fingerprints divided evenly between genuine and spoofed fingerprints. Each model assigns a probability for each fingerprint in the sample set. Using this distribution we set a threshold to classify the fingerprints as genuine or spoofed. We compare the results with the ground truth and count how many fingerprints from each class are classified correctly. We compare the results across different models using ROC curves. Here, "True Positive" denotes the number of genuine fingerprints detected as

Table 2: Accuracy of each evaluator for distinguishing valid fingerprints from the spoofed ones among 10k pairs using test dataset and training dataset.

Dataset	Evaluator		
	Independent	Partially Dependent	Fully Dependent
Test	56%	94%	77%
Training	57%	96%	99%

[1]Given the quadratic runtime of the approach described in this section, utilizing the whole dataset was not feasible. However, the qualitative comparisons resulting from the experiments are expected to hold regardless of the dataset size.



(a) Test Dataset (b) Training Dataset

Figure 2: ROC curves for the performance of each model in classifying a sample set of 20k fingerprints of genuine vs. spoofed fingerprints. Panel (a) and (b) samples are drawn from the test datasets and the training, respectively. The points in this plot correspond to different levels of the probability threshold for the output of each model.

genuine (or detected as spoofed for "False Negative") by the model, and "True Negative" refers to the number of spoofed fingerprints generated using the Independent model detected as spoofed fingerprints (or detected as genuine for "False Positive") by the model. The results are presented in Fig. 2. Again we ran the experiment twice, once reusing the test data (panel a) and once with the discrete training data (panel b).

The results confirm the findings from the previous experiment. Specifically, the Independent model performance is poor and closer to the random guessing. The Fully-Dependent model suffers from over-fitting as it performs flawlessly on the training dataset but drops precipitously on the test dataset. The Partially-Dependent model's performance is remarkably consistent on both datasets which makes it a better fit for use in practice.

Probability Distributions In the last experiment in this section, we present more detailed information about the performance of the evaluation models. In this experiment, each of the three previously described generator models was used to generate 10k fingerprints. In addition, the Dataset-Sampler was used to sample 10k fingerprints directly from the test dataset, resulting in a total of four batches of fingerprints. Then the evaluation models were used to calculate the probability distribution for each of these fingerprint batches. The histogram of these fingerprints is presented in Fig. 3. This histogram shows the negative log of the probabilities, meaning that the probability is higher when the values are closer to zero. Thus a flatter curve suggests more instances of low-probability fingerprints, which therefore indicates that the evaluation model is detecting an increasing number of paradoxical fingerprints from the generator model.

Note the Partially-Dependent model in the second column of Fig. 3. We can see that the model can more reliably distinguish the fingerprints generated by the Independent generator model when compared with the valid fingerprints of Dataset-Sampler as the former has flatter curve. Again the Independent model has a poor evaluation performance since the probabilities that it calculates for valid fingerprints are generally lower than the probabilities of fingerprints generated with the Independent generator model. Although the Fully-Dependent model presents a flatter histogram for the Independent generator compared to the valid fingerprints of Dataset-Sampler, its evaluation of Dataset-Sampler is inferior relative to the Partially-Dependent evaluation model.

Comparing Runtime & Memory Costs Recall that one of the criteria for evaluation is performance. We compared the runtime of the models relative to both model creation (training) and generating new fingerprints. The training time of all three models is asymptotically linear relative to the input size.

The results of our performance evaluation in terms of system requirements were that the highest performing model Partially-Dependent required ∼12MB to be stored on disk when trained on ∼1M fingerprints. Fully-Dependent models were ∼15x larger in size.

Generation of fingerprints using any of the three

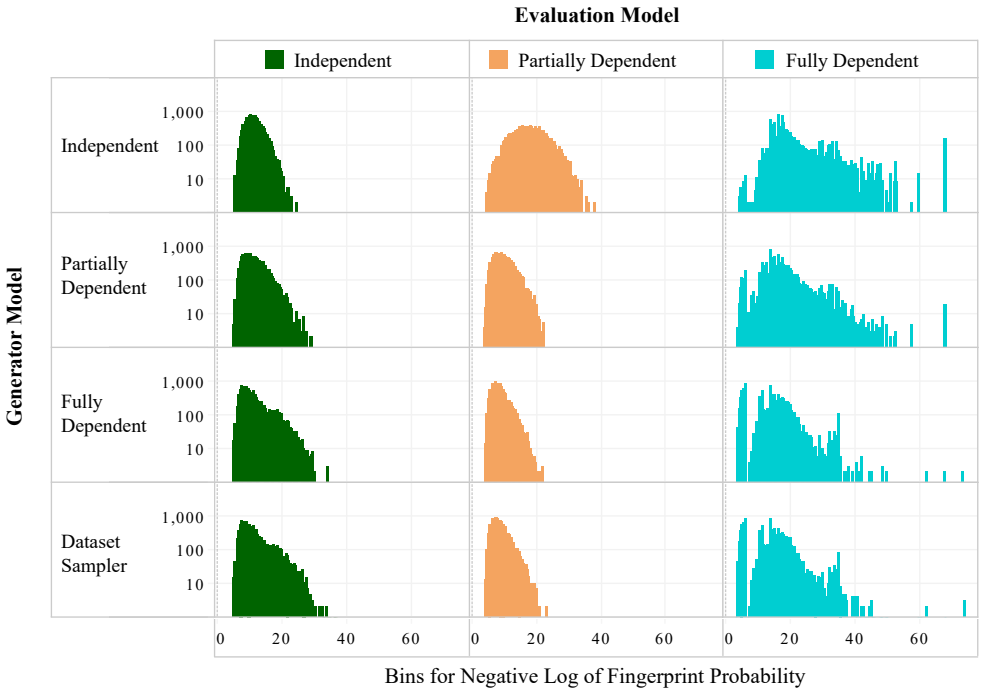


Figure 3: Distribution for the likelihood of fingerprints calculated by different models over 10k sample fingerprints generated by each model (the probability is higher when the values are closer to zero). The Generative models are built from the training dataset, and the evaluation models are built from the test dataset.

models has asymptotic runtime linear in terms of the number of attributes and thus is not expected to be an issue for users. Since the runtime of generating a single fingerprint is short and can be sensitive to unrelated factors, we measured the runtime of generating 1 million fingerprints for each model. We concluded that the runtimes of the models are very close despite their significant differences in space usages.

9. Conclusions

We constructed three algorithms for creating spoofed fingerprints and evaluated them according to criteria developed from previous research on why people use (or don't use) PETs. Each of these three approaches captures different levels of dependencies between attributes. All pairs of attributes are independent in the first model, only pairwise dependence is considered in the second model, and the full chain of possible dependencies is considered in the third model.

Based on that criteria we sought to efficiently provide anonymized fingerprints while maintaining the functionality of websites and remaining undetected by trackers. To do so we built a fingerprint model generator and evaluator called FPModeler and used a dataset of more than 2 million fingerprints as a starting point to generate our models. Maintaining the statistical privacy of the entries in this dataset was an additional requirement, based on protecting the privacy of the people who contributed fingerprints to the dataset.

We identified the trade-offs of different models in terms of the relationships between indistinguishability (i.e., detectability, guaranteed non-uniqueness, and k-anonymity), as well as efficiency, website functionality while spoofing, and the privacy of the participants whose data underlie the creation of FPModeler. We provided the detail necessary to repeat our evaluations for generating fingerprints, and repeated detection of a genuine fingerprint in pairs of sampled and generated entries.

Our results illustrate that it is feasible to use the Partially-Dependent model to provide undetectable and efficient fingerprint spoofing with high levels of anonymity; not only for the users leveraging the prior likelihoods to spoof convincingly but also for the participants who have contributed fingerprints to the dataset. In future work, we hope to evaluate the usability and acceptability of our models in browsers.

Acknowledgements This research was supported in part by the National Science Foundation awards CNS 1565375 and CNS 1814518, grant number #H8230-19-1-0310, Cisco Research Support, Google Research, and the Comcast Innovation Fund. Any opinions, findings, and conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Government, the National Science Foundation, Cisco, Comcast, Google, nor Indiana University.

References

[1] T. Unger, M. Mulazzani, D. Frühwirt, M. Huber, S. Schrittwieser, and E. Weippl, "Shpf: Enhancing Http (s) Session Security With Browser Fingerprinting," in *International Conference on Availability, Reliability and Security*, pp. 255–261, IEEE, 2013.

[2] K. Boda, Á. M. Földes, G. G. Gulyás, and S. Imre, "User Tracking on the Web via Cross-Browser Fingerprinting," in *Nordic Conference on Secure IT Systems*, pp. 31–46, Springer, 2011.

[3] S. Englehardt and A. Narayanan, "Online Tracking: A 1-Million-Site Measurement and Analysis," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1388–1401, 2016.

[4] P. Snyder, A. Vastel, and B. Livshits, "Who Filters the Filters: Understanding the Growth, Usefulness and Efficiency of Crowdsourced Ad Blocking," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 2, pp. 1–24, 2020.

[5] R. Nithyanand, S. Khattak, M. Javed, N. Vallina-Rodriguez, M. Falahrastegar, J. E. Powles, E. De Cristofaro, H. Haddadi, and S. J. Murdoch, "Adblocking and Counter Blocking: A Slice of the Arms Race," in *6th USENIX Workshop on Free and Open Communications on the Internet (FOCI 16)*, 2016.

[6] R. Dingledine and N. Mathewson, "Anonymity Loves Company: Usability and the Network Effect," in *WEIS*, 2006.

[7] H. Assal, S. Hurtado, A. Imran, and S. Chiasson, "What's the Deal with Privacy Apps? A Comprehensive Exploration of User Perception and Usability," in *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*, 2015.

[8] A. Acquisti, C. Taylor, and L. Wagman, "The economics of privacy," *Journal of economic Literature*, vol. 54, no. 2, pp. 442–92, 2016.

[9] V. Garg and J. Camp, "Heuristics and Biases: Implications for Security Design," *IEEE Technology and Society Magazine*, vol. 32, no. 1, pp. 73–79, 2013.

[10] M. Juarez, S. Afroz, G. Acar, C. Diaz, and R. Greenstadt, "A Critical Evaluation of Website Fingerprinting Attacks," in *ACM SIGSAC*, 2014.

[11] E. F. Foundation, "Panopticlick." [Accessed on Aug 24 2021], 2020. `https://coveryourtracks.eff.org`.

[12] P. Eckersley, "How Unique is Your Web Browser?," in *International Symposium on Privacy Enhancing Technologies Symposium*, pp. 1–18, Springer, 2010.

[13] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting," in *2013 IEEE Symposium on Security and Privacy*, pp. 541–555, 2013.

[14] N. Kaur, S. Azam, K. Kannoorpatti, K. C. Yeo, and B. Shanmugam, "Browser Fingerprinting as User Tracking Technology," in *11th International Conference on Intelligent Systems and Control*, pp. 103–111, 2017.

[15] N. M. Al-Fannah and W. Li, "Not All Browsers are Created Equal: Comparing Web Browser Fingerprintability," in *International Workshop on Security*, pp. 105–120, 2017.

[16] J. Yan and J. Kaur, "Feature Selection for Website Fingerprinting," *Privacy Enhancing Technologies*, vol. 2018, no. 4, pp. 200–219, 2018.

[17] A. Gómez-Boix, P. Laperdrix, and B. Baudry, "Hiding in The Crowd: an Analysis of the Effectiveness of Browser Fingerprinting at Large Scale," in *Proceedings of the World Wide Web Conference*, pp. 309–318, 2018.

[18] K. Tanabe, R. Hosoya, and T. Saito, "Combining Features in Browser Fingerprinting," in *International Conference on Broadband and Wireless Computing, Communication and Applications*, pp. 671–681, 2018.

[19] G.-F. He, M. Yang, J.-Z. Luo, and L. Zhang, "Online Identification of Tor Anonymous Communication Traffic," *Ruanjian Xuebao/Journal of Software*, vol. 24, no. 3, pp. 540–556, 2013.

[20] A. Chaabane, P. Manils, and M. A. Kaafar, "Digging into Anonymous Traffic: A Deep Analysis of the Tor Anonymizing Network," in *Fourth International Conference on Network and System Security*, pp. 167–174, 2010.

[21] A. Dunna, C. O'Brien, and P. Gill, "Analyzing China's Blocking of Unpublished Tor Bridges," in *8th USENIX Workshop on Free and Open Communications on the Internet (FOCI 18)*, 2018.

[22] P. Laperdrix, "Browser Fingerprinting: An Introduction and the Challenges Ahead." [Online blog accessed on Aug 24 2021] Available at: `https://blog.torproject.org/browser-fingerprinting-introduction-and-challenges-ahead`.

[23] J. F. Duncan and L. J. Camp, "Conducting an Ethical Study of Web Traffic.," in *5th Workshop on Cyber Security Experimentation and Test (CSET)*, 2012.

[24] V. Andalibi, F. Christophe, and T. Mikkonen, "Analysis of Paradoxes in Fingerprint Countermeasures," in *21st Conference of Open Innovations Association FRUCT*, 2017.

[25] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas, "Adnostic: Privacy Preserving Targeted Advertising," in *Network and Distributed System Symposium (NDSS)*, 2010.

[26] M. Bailey, D. Dittrich, E. Kenneally, and D. Maughan, "The Menlo Report," *IEEE Security & Privacy*, vol. 10, no. 2, pp. 71–75, 2012.

[27] G. Navarro, "A Guided Tour to Approximate String Matching," *ACM computing surveys (CSUR)*, vol. 33, no. 1, pp. 31–88, 2001.