Machine Learning-Assisted Website Fingerprinting Attacks with Side-Channel Information: A Comprehensive Analysis and Characterization

Han Wang[1], Hossein Sayadi[2], Avesta Sasan[3], Sai Manoj P D[3], Setareh Rafatirad[1], and Houman Homayoun[1]
[1]University of California, Davis, CA, USA
[2]California State University, Long Beach, CA, USA
[3]George Mason University, Fairfax, VA, USA
[1]{hjlwang, hhomayoun,srafatirad}@ucdavis.edu, [2]{hossein.sayadi}@csulb.edu,[3]{asasan,spudukot}@gmu.edu

2021 22nd International Symposium on Quality Electronic Design (ISQED) | 978-1-7281-7641-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISQED51717.2021.9424290

***Abstract*—The World Wide Web has become an essential part of modern society over the last decade, where people acquire new knowledge, conduct businesses, and their daily tasks. Such transformation makes the users' website browsing history contains more sensitive information, like health condition, political interests, financial situations, etc. Some defense mechanisms such as SSH tunnels and anonymity networks (e.g., Tor) have been proposed to cope with the potential website behaviors leakage and enhance browsing security. Nevertheless, some recent studies have demonstrated the possibility of fingerprinting websites based on side-channel information such as cache usage, memory utilization, CPU activity, and hardware performance counters. Such attacks observe the side-channel information and leverage effective machine learning techniques to infer which website a user is visiting via anonymity networks and encrypted proxies. Passive execution and extensive use of such side channels make the detection and mitigation of such side-channel information leakage-based fingerprinting attacks more challenging. This work presents a comprehensive analysis of state-of-the-art research on applying machine learning techniques on various side-channel features to develop effective website fingerprinting attacks.**

***Index Terms*—Side-channel, Machine learning, Website Security and Privacy,**

I. Introduction

The World Wide Web has become an essential part of modern society over the last decade, where people acquire new knowledge, conduct businesses, and their daily tasks. New concerns about system security and users' privacy arise from the transformation to the virtual world. Through stealing users' online behaviors and access patterns, attackers can further induce personal or even sensitive information, including sexual orientation or political beliefs and affiliations. A web browser fingerprint attack is referred to the extraction of browsing history from users. Many attacks developed [19], [22] observe packet sizes, packet timings, direction of communication, etc. to infer the website that the user is visiting with the usage of machine learning classification algorithms.

To protect users' online privacy and improve the system security, many researchers have also proposed to hide the network traffic of users [4], [17], [26]. Tor network [26] constructs an overlay network of collaborating servers, called relays. It encrypts the Internet traffic between users and web servers by transmitting the traffic between relays to prevent external observers from identifying the traffic of specific users. The Tor Browser based on the Firefox web browser further protects users by disabling features that may be used for tracking users.

Despite considerable progress in developing users' privacy protection mechanisms, there are still many recent privacy violation attacks that rely on the application of Machine Learning (ML) algorithms that are trained with computer systems' side-channel information collected when a website is visited. These attacks stem from existing side-channel vulnerabilities like systems power analysis [15], CPU activity [4], on-chip cache memories [23], memory footprints [9], storage [12], and hardware events [7]. Such attacks pose significant security threats to privacy preservation due to three main reasons as follows:

(1) Flexible Deployment: The attacks can collect side-channel information without physical access, enabling more diverse and flexible attack scenarios. They can further embed in a malicious website or malware and start monitoring once the malicious website is visited or malware is executed.

(2) Passive Execution. The website fingerprinting attacks [4], [7], [23] steal users' browsing history by observing and analyzing side-channel information without adding detectable performance overhead on the victim's process since this information is always collected by the system for debugging and optimization purposes.

(3) Side-Channel Existence. Though it is proven that side-channel information of visiting websites can be exploited by attackers which violate users' privacy, disabling the collection of this information is not practical, since these features are mostly used for performance/power optimization, software debugging, and scheduling purposes [10], [24], [29]–[31].

In this work, we first summarize various platforms and side-channel vulnerabilities leveraged by state-of-the-art website fingerprinting attacks. Then, we conduct a comprehensive analysis of machine learning classifiers adopted in such attacks in terms of various evaluation metrics such as accuracy, true positive rate, false positive rate, and F measure. Furthermore, we discuss the potential mitigation methods for defending

the computing systems against such side-channel oriented privacy violation attacks. The remainder of this paper is organized as follows. The background and relevant work are described in Section II. The design methodology are discussed in Section III. Section IV provides a comprehensive analysis of different ML-based SCA detectors across various metrics. Finally, Section V discusses the potential mitigation strategy.

II. Background and Related Works

A. Website Fingerprinting Attacks

Since website browsing history contains much sensitive information like medical status, political interest, etc., it is critical to prevent leakage, protect users' privacy, and enhance the system's security against potential cyber-attacks. However, prior research has demonstrated that fingerprinting attacks can be independent of operating systems and browsers and rely on side-channel information collected passively. Such attacks can be launched both remotely and locally, or via a peripheral device. The side-channel information exists across different computing abstracts, including hardware, system, and network. Once side-channel information is collected, machine learning-based classification is leveraged to infer which websites users visit.

Taking fingerprinting attacks that exploit hardware performance counters (HPCs) as an example, attacks can profile the HPCs data for each website offline and use them to build a machine learning-based classifier model. And then attackers activate the HPCs monitoring in users' computers and send them to remote classification models that can output the website users are visiting, like "google.com". The research in [1] demonstrated 86.3% classification accuracy based on HPCs information.

B. Related Work

Protecting browsing history has emerged recently as a crucial concept to ensure the preservation of users' privacy. For instance, the secure shell (SSH) protocol [33] is proposed to encrypt and authenticate messages in one session. Similarly, Tor project [26] is one of the most popular networks-based anomaly detection, where messages are not directly routed to the receiver but encrypted and forwarded according to ephemeral paths of an overlay network. Though great progress made by such works, there are still a number of attacks which could bypass such protection mechanisms and extract users' browsing history.

Some recent website fingerprint attacks exploit the clients' machine information when visiting different websites like memory footprint [9], storage [25], etc. Some attacks prepare a malicious website for users to visit or a local malware to be launched on the target host to obtain the information. For example, [23] launches a Prime+Probe attack to measure cache occupancy through a malicious website. Then, a deep learning method is leveraged to classify websites and recover users browsing history. Other works such as [9] samples the memory footprint of browsers through the procfs file system in Linux.

III. Website Fingerprinting Methodology

This section presents the details of three main website fingerprinting attack models and their exploited side-channel information. Then, the utilized website database and machine learning classifiers the recent works are characterized.

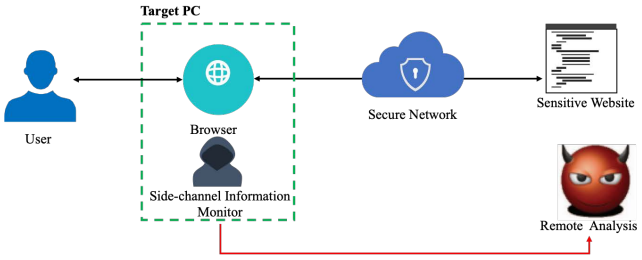

Fig. 1: Native attack model

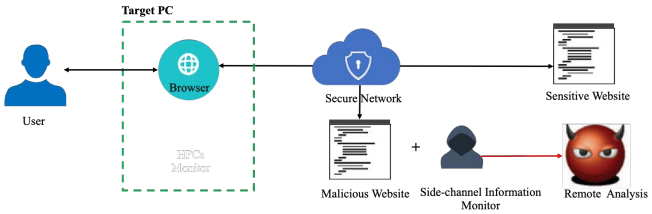

Fig. 2: Malicious website attack model

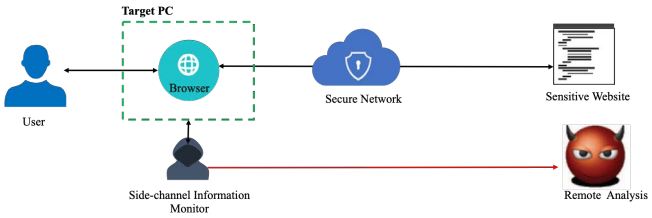

Fig. 3: Hardware attack model

A. Threat Model

As shown in Figure 1 to Figure 2, there are three popular threat models targeting stealing users' browsing history as summarized in Table I including native attack model [7], malicious website attack model [23], and hardware attack model [5].

1) Native Attack Model: In this attack model, the assumption is that the malicious code is resided in the host machine already, which can be done by inserting it into benign applications or downloading accidentally. With this model, attacks can be activated as long as the malicious codes are installed already and do not need users to open certain websites.

2) Malicious Website Attack Model: By comparison, this attack model assumes that users click on a malicious website link; thereby, malicious JavaScript codes are executed in local computers. Compared to the native model, this model is more flexible to update the attack and can be less visible since local malware scan cannot detect malicious codes. For both models, the malicious codes are only in charge of side-channel information monitoring without compromising systems.

TABLE I: Exploited Side-channel List

Side-channel	Browser	OS	Threat Model	Leakage Component	Sampling Rate
Hardware Performance Counter [7]	Chrome	Linux	Native	Performance Monitoring Unite	10K
Magnetometer [18]	Smart phones	Andriod	Native, Malicious Website	Sensor	10-100
Temporary Storage [12]	Chromium Linux, Chrome	Win, Android	Malicious Website	Quota management API	N/A
Shared event loops [27]	Chrome	Linux, Mac OS	Malicious Website	N/A	40k
Data usage statistics [25]	Chrome, Firefox, Tor	Android	Native	N/A	20-50
Cache Occupancy [23]	Chrome, Firefox,Safari,Tor	Win, Linux,MacOS	Malicious Website	Last-level cache	10-500
GPU memory [14]	Chromium,Firefox	Linux	Native	GPU memory	N/A
Power Usage [5]	Chrome	Mac OS, Windows, Linux	Hardware	AC Power consumption	7.8k-250k
Power Usage [32]	Chrome	smart phones	Hardware	Power consumption	200k

3) Hardware Attack model: As shown in Figure 3, this model collects hardware properties, mainly power consumption, when various websites are opened. Some of them infer the side-channel information based on hardware monitoring components such as USB and power meter. Though physical access is needed under this mode, [5] measures the power consumption and achieves 98% website classification accuracy, posing a significant security threat to computer systems and privacy.

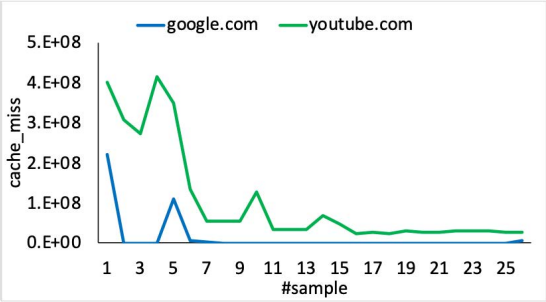

Fig. 4: Cache-misses temporal sequences of google.com and youtube.com

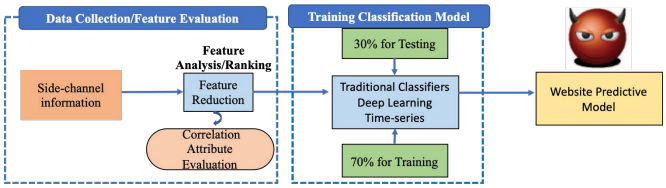

Fig. 5: Remote analysis model implementation

B. Website Database Description

Alexa Top Sites [2] is used as a website database for fingerprinting attacks based on side-channel information. There are two types of dataset used in prior attacks: close World and open World. It is assumed that classifiers have seen all websites for the close world while classifiers have not seen some websites for the open world dataset. The preparation for the two databases is described in detail following.

1) Close World Dataset: The closed world dataset means that each website is sensitive and exists in the training dataset. The proposed attack model considers distinguishing a relatively small list of websites, ranging from 30 to 100, and the classifiers are required to conduct multi-label classification and output result like "google.com", "yahoo.com", or "facebook.com", etc.

2) Open World Dataset: Compared to the close world dataset, the open wold dataset only has two values for the label: "sensitive" and "non-sensitive". Usually, the sensitive traces are from the close world, and the number of non sensitive traces are much larger than sensitive traces. Hence, the open world dataset also contains a large set of non-sensitive websites up to 5,000, all of which attacker is expected to generally label as "non-sensitive" [23]. And each of the additional non-sensitive traces represents the behavior of a single unique website.

C. Side-Channel Information

Table I lists the most common exploited side-channel information in existing website fingerprinting attacks. Such vulnerabilities include hardware performance counters, magnetometer, temporary storage, shared event loops, data usage, etc. It is also noticeable that both desktops and smartphones are vulnerable to fingerprinting attacks, and most commercial browsers are not able to effectively protect users' privacy. Based on the table, it is safe to conclude that website fingerprinting exists widely across various hardware and software settings. As an example, in Figure 4, we analyzed hardware performance counter behavior in the website fingerprinting process. In particular, the figure presents HPC-based temporal sequences ($cache-misses$ event) of $google.com$ and $youtube.com$. As observed, the collected HPC traces of two sample websites have shown a significantly different trend line providing an ample attack surface to identify the visited website information by monitoring low-level HPCs information and applying effective machine learning techniques.

D. Usage of Machine Learning Techniques

Machine learning has drawn lots of attention from industry and academia in the past two decades. It has made significant progress in recognition, detection, and automation across different data formats, including images, voice, text, video, etc. For the side-channel based website fingerprinting attacks, all the datasets are labeled during the training phase, and hence supervised classifiers are employed. A brief introduction of classifiers used in prior attacks is given in the following.

1) Histograms-based Classification: This approach transfers feature values into a histogram with k ranges representing the number of features falling into each k range. Then, the Euclidean distance is used to measure the similarity between the k range distribution. This approach is suitable for dataset that has more irregular patterns but similar value distribution.

2) k-th Nearest Neighbor (kNN): The main methodology for kNN is to find the closest training sample to a test sample and gives the label of the training sample to the test sample. Specifically, kNN uses the Euclidean distance to measure the distance between training samples and testing samples and finds k closest training samples.

3) Decision Tree (DT): Decision Tree has a number of branches to split input data into different classes. The main step is to find each splitting point threshold for the features and achieve the highest accuracy classification accuracy. It has been applied to many classification tasks, like malicious application, detection [28], [31].

4) Support Vector Machine (SVM): The strategy is to find a boundary between classes and increase the distance between them, reducing the over-fitting problem. To achieve this, SVM transfers the input dataset into a multi-dimensional representation with a mapping function. Afterward, hyperplanes are created to classify the data. In SVM based learning, input data is converted to a multi-dimensional model by using mapping functions.

5) Convolutional Neural Network (CNN): Convolutional Neural Network is widely used for image classification, temporal sequence identification and also numerous applications. A convolutional layer in a deep neural network learns patterns of local structure in the input signal and can learn feature representations over a sequence of input data [13]. To enable the features' importance automatically, it creates bridges between higher and lower dimensional mappings for the input data [7].

6) Long Short-Term Memory (LSTM): For the LSTM [11], each temporal trace includes a time-ordered sequence of side-channel information on which an LSTM network detects temporal patterns that are important for discriminating different websites. One and two layers of LSTM neurons with various numbers of neurons per layer were explored. Each node learns a different sequence pattern and the collection of sequence pattern detectors from all the nodes connected to the output layer are used to classify each temporal sequence.

7) Dynamic Time Warping (DTW): Time-series classification methods deal with such temporal datasets which is representing them in time-domain format and then calculate the distance as difference between time series [16]. DTW was introduced into classification problems and time-series mining by Berndt and Clifford [3], where dynamic programming was used to align sequences with different lengths. The main idea of DTW is to find an optimal match between two sequences by allowing a nonlinear mapping of one sequence to the other sequence and minimizing the distance between two sequences. Considering two HPC temporal sequences A and B as follows: A(a1, a2,..., an) and B(b1, b2,..., bm), DTW finds an optimal warping path between A and B by using dynamic programming to calculate the minimum cumulative distance.

E. Building Machine Learning Classifiers

Since the website fingerprinting attacks collect side-channel information per website with labels like "google.com" for close world dataset or "sensitive" for open world dataset, supervised learning is used to build a machine learning classifier for the remote analysis in Figure 1, Figure 2 and Figure 3. As shown, for building an effective ML-based classifier, there are two main steps as shown in Figure 5: including data collection/feature evaluation and training/testing classifiers.

Multiple traces (50 100) from each website are collected and labeled in the data collection step. Since one side-channel information might contain multiple features, like hardware performance counters, feature evaluation is conducted and the most prominent features can be chosen after this. While this is not necessary, cache occupancy based fingerprinting attack [23] only uses one feature and does not feature evaluation step.

For the training part, a wide range of classification algorithms are chosen as shown in Section III-D due to the accuracy can vary from on classifier to another, including classical models (kNN, DT, RandomForest, SVM), deep learning models (CNN, LSTM), time-series models (DTW) techniques. 70% dataset is used for training and cross validation while the rest is for testing. From the classification results of testing, the optimal classifier is chosen as the final website predictive model. This model can output the prediction result after receiving unlabeled traces in which each of the trace corresponds to a user's website visit.

IV. Evaluation of Website Fingerprinting Attacks

In this Section, we summarize the adopted approaches and metrics used to evaluate the effectiveness of machine learning-assisted website fingerprinting attacks.

A. Evaluation Approach

To comprehensively compare the various ML classifiers, both cross validation and percentage validation are employed in prior fingerprinting attacks.

1) Cross Validation: Cross validation splits the dataset into K (1, ..., n) folds and selects one of them as a testing dataset while the rest folds are used for the training dataset. The number of iteration times is decided by the accuracy increases over successive iterations and stops when the validation set's accuracy does not increase.

2) Percentage Validation: Another validation method is percentage based, meaning the collected database is split into two parts based on percentage setting. One is for training, and the other one is for testing. Since the trained machine learning classifiers have never seen the testing dataset, this method can imitate the testing result in the real-world and indicate how accurate the classifiers can be for on-line prediction.

B. Evaluation Metrics

To evaluate machine learning classifiers' performance, several metrics are employed, including accuracy, false positive rate, true positive rate, and F measure. The metrics are then introduced in detail and are further listed in Table III.

1) Accuracy: Accuracy measures the rate of samples classified correctly, indicating the attack success rate. Specifically, the accuracy is calculated based on the division of samples classified correctly and all samples.

TABLE II: Exploited Side-channel List

Side-channel	Investigated Classifier Models	Optimal Model	Close World Acc	Open World Acc
Hardware Performance counter [7]	kNN, DT, SVM, CNN	CNN	86.3%	N/A
Magnetometer [18]	Random Forest	Random Forest	90%	81%
Temporary Storage [12]	N/A	N/A	90%	N/A
Shared event loops [27]	Event Delay Histograms, DTW	DTW	91.1 %	N/A
Data usage statistics [25]	Jaccard index	Jaccard index	97%	N/A
Cache Occupancy [23]	CNN, LSTM	LSTM	91.4%	86.4%
GPU memory [14]	N/A	N/A	95.4%	N/A
Power Usage [5]	SVM	SVM	98%	N/A
Power Usage [32]	SVM, Random Forest, DTW	RandomForest	90%	N/A

TABLE III: Evaluation metrics for performance of ML-based classification

Evaluation Metric	**Description**
True Positive (TP)	Correct positive prediction
False Positive (FP)	Incorrect positive prediction
True Negative (TN)	Correct negative prediction
False Negative (FN)	Incorrect negative prediction
Specificity: True Negative Rate	$TNR = TN/(TN + FP)$
False Positive Rate	$FPR = FP/(FP + TN)$
Accuracy	$ACC = (TP + TN)/(TP + FP + TN + FN)$
Area Under the Curve	$AUC = \int_0^1 TPR(x)dx = \int_0^1 P(A > \tau(x))dx$
F measure (F score)	$Fmeasure = 2 \times (P \times R)/(P + R)$

2) True Positive Rate (TRP): As mentioned in Section III-B, attacks conduct binary classification for open world data, identifying whether the visited website is sensitive or non-sensitive. Classifiers often consider sensitive websites as non-sensitive ones. As a result, the true positive rate metric helps represent the proportion of correctly identified sensitive websites. The positive samples are basically the correctly classified sensitive samples.

3) False Positive Rate (FPR): As a companion of true positive rate, false positive rate identifies the rate of non-sensitive websites (i.e., negative samples) wrongly classified as sensitive websites (i.e., positive samples).

4) Receiver Operating Characteristic (ROC) Curve: Receiver Operating Characteristics (ROC) Curve is produced by plotting the fraction of true positives rate versus the fraction of false positives for a binary classifier. The best possible classifier would thus yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 0% false positives and 100% true positives. The Area under the ROC Curve (AUC) metric corresponds to the probability of correctly identifying "sensitive" and "non-sensitive". Robustness is referred to how well the classifier distinguishes between the two classes for all possible threshold values. Higher AUC indicates better robustness for ML classifiers.

5) F Measure: F-measure is interpreted as a weighted average of the precision (p) and recall (r), which is formulated as $\frac{2\times(p\times r)}{p+r}$. The precision is the proportion of the sum of true positives versus the sum of positive samples, and the recall is the proportion of samples that are predicted positive of all the samples that are positive. F-measure is a more comprehensive evaluation metric over accuracy (percentage of correctly classified samples) since it takes both the precision and the recall into consideration.

C. Attacking Results Comparison

As shown in Table II, the classification accuracy across various fingerprinting attacks is presented for both close world and open world dataset. Additionally, the investigated classifiers and the optimal classifier model for each attack are summarized. It can be seen that the website classification accuracy ranges from 86.3% to 98% across various side-channel based website fingerprinting attacks for close world dataset. Only two recent works [18], [23] consider open world dataset and achieve 81% and 86.4% accuracy respectively. Another interesting fact is that Random Forest, SVM, DTW, CNN, LSTM are the most commonly used classification techniques while the deep learning model is more effective than others [7].

V. Mitigation Approach

Browsing security and privacy preservation have become increasingly important due to the wide use of the Internet across various areas. To response, [21], [33] are proposed to leverage SSH based protection methods; [33] encrypts and authenticates messages in one session, and [21] adds cover traffic conservatively while maintaining high levels of security. Similarly, Tor project [26] is one of the most popular networks-based anomaly traffic transmission approaches, where messages are not directly routed to the receiver but encrypted and forwarded according to ephemeral paths of an overlay network. Although such countermeasures are effective, they can only combat attacks that exploit network-based side-channel information.

Recent website fingerprinting attacks exploit the system level side-channel information, e.g., memory footprint [9], storage [25], hardware performance counters [1], to steal the browsing history. System level side-channel information is unusually obtained by attackers through a malicious website for users to visit, or a local malware to be launched on the target host. Afterwords, a deep learning method is leveraged to classify websites based on the obtained side-channel information and recover users' browsing history. To defend against such attacks, [8] proposes ML-based syntactic-semantic approach that detects browser fingerprinting attacks' behaviors by incorporating both static and dynamic JavaScript analysis. [6] proposes to monitor the running Web objects on user's browser and collect fingerprinting related data. Then it analyzes them and searches for patterns of fingerprinting attempts. Though effective, they only work for attacks deployed through malicious websites. Furthermore, advanced

attacks [7] can be deployed in native code and bypass such detection systems. [20] randomizes properties, like offsetHeight and plugins, to the JavaScript environment, which generates different fingerprints even for the same website and increases non-determinism for attackers. However, the randomization is complex and can change the visual appearance of websites.

VI. Conclusion

In the past several decades, the development of Internet and World Wide Web transfers many social activities to virtual world, highlighting the importance of protecting the system security and users' privacy. However, when a user visits a website in a web browser, there are a number of distinct patterns of run-time side-channel information exposed which can be used to infer sensitive information such users' browsing history. Such privacy violation also known as website fingerprinting attacks have received considerable attention recently due to the proliferation of modern computing systems and widespread application of machine learning in systems cybersecurity. In this work, we present a comprehensive analysis of state-of-the-art research on applying machine learning techniques on various side-channel features to develop effective website fingerprinting attacks.

Acknowledgment

This research was supported in part by NSF program under the award number 1936836.

References

[1] Perf. In *https://perf.wiki.kernel.org/index.php/Main Page*.

[2] Analitic, A. The top 500 sites on the web https://www.alexa.com/topsites.

[3] Berndt, D. J., and Clifford, J. Using dynamic time warping to find patterns in time series. In *KDD workshop* (1994), vol. 10, Seattle, WA, pp. 359–370.

[4] Booth, J. *Not so incognito: Exploiting resource-based side channels in JavaScript engines*. PhD thesis, 2015.

[5] Clark, S. S., Mustafa, H., Ransford, B., Sorber, J., Fu, K., and Xu, W. Current events: Identifying webpages by tapping the electrical outlet. In *European Symposium on Research in Computer Security* (2013), Springer, pp. 700–717.

[6] FaizKhademi, A., Zulkernine, M., and Weldemariam, K. Fpguard: Detection and prevention of browser fingerprinting. In *IFIP Annual Conference on Data and Applications Security and Privacy* (2015), Springer, pp. 293–308.

[7] Gulmezoglu, B., Zankl, A., Eisenbarth, T., and Sunar, B. Perfweb: How to violate web privacy with hardware performance events. In *European Symposium on Research in Computer Security* (2017).

[8] Iqbal, U., Englehardt, S., and Shafiq, Z. Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors. *arXiv preprint arXiv:2008.04480* (2020).

[9] Jana, S., and Shmatikov, V. Memento: Learning secrets from process footprints. In *2012 IEEE Symposium on Security and Privacy* (2012), IEEE, pp. 143–157.

[10] Kandalintsev, A., Cigno, R. L., Kliazovich, D., and Bouvry, P. Profiling cloud applications with hardware performance counters. In *The International Conference on Information Networking 2014 (ICOIN2014)* (2014), IEEE, pp. 52–57.

[11] Karim, F., Majumdar, S., Darabi, H., and Chen, S. Lstm fully convolutional networks for time series classification. *IEEE access 6* (2017), 1662–1669.

[12] Kim, H., Lee, S., and Kim, J. Inferring browser activity and status through remote monitoring of storage usage. In *Proceedings of the 32nd Annual Conference on Computer Security Applications* (2016), pp. 410–421.

[13] LeCun, Y., Bengio, Y., et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks 3361*, 10 (1995), 1995.

[14] Lee, S., Kim, Y., Kim, J., and Kim, J. Stealing webpages rendered on your browser by exploiting gpu vulnerabilities. In *2014 IEEE Symposium on Security and Privacy* (2014), IEEE, pp. 19–33.

[15] Lifshits, P., Forte, R., Hoshen, Y., Halpern, M., Philipose, M., Tiwari, M., and Silberstein, M. Power to peep-all: Inference attacks by malicious batteries on mobile devices. *Proceedings on Privacy Enhancing Technologies 2018*, 4 (2018), 141–158.

[16] Lin, J., et al. Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery* (2007).

[17] Luo, X., Zhou, P., Chan, E. W., Lee, W., Chang, R. K., and Perdisci, R. Httpos: Sealing information leaks with browser-side obfuscation of encrypted flows. In *NDSS* (2011), vol. 11.

[18] Matyunin, N., Wang, Y., Arul, T., Kullmann, K., Szefer, J., and Katzenbeisser, S. Magneticspy: Exploiting magnetometer in mobile devices for website and application fingerprinting. In *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society* (2019), pp. 135–149.

[19] Narayanan, A., Paskov, H., Gong, N. Z., Bethencourt, J., Stefanov, E., Shin, E. C. R., and Song, D. On the feasibility of internet-scale author identification. In *2012 IEEE Symposium on Security and Privacy* (2012), IEEE, pp. 300–314.

[20] Nikiforakis, N., Joosen, W., and Livshits, B. Privaricator: Deceiving fingerprinters with little white lies. In *Proceedings of the 24th International Conference on World Wide Web* (2015), pp. 820–830.

[21] Nithyanand, R., Cai, X., and Johnson, R. Glove: A bespoke website fingerprinting defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society* (2014), pp. 131–134.

[22] Rimmer, V., Preuveneers, D., Juarez, M., Van Goethem, T., and Joosen, W. Automated website fingerprinting through deep learning. *arXiv preprint arXiv:1708.06376* (2017).

[23] Shusterman, A., and et.al. Website fingerprinting through the cache occupancy channel and its real world practicality. *IEEE TDSC* (2020).

[24] Singh, K., Bhadauria, M., and McKee, S. A. Real time power estimation and thread scheduling via performance counters. *ACM SIGARCH Computer Architecture News 37*, 2 (2009), 46–55.

[25] Spreitzer, R., Griesmayr, S., Korak, T., and Mangard, S. Exploiting data-usage statistics for website fingerprinting attacks on android. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks* (2016), pp. 49–60.

[26] The Tor Project, I. The tor browser. *https://www.torproject.org/projects/torbrowser.html.en.*.

[27] Vila, P., and Köpf, B. Loophole: Timing attacks on shared event loops in chrome. In *26th {USENIX} Security Symposium ({USENIX} Security 17)* (2017), pp. 849–864.

[28] Wang, H., et al. Scarf: Detecting side-channel attacks at real-time using low-level hardware features. In *IOLTS* (2020).

[29] Wang, H., Rafatirad, S., and Homayoun, H. A+ tuning: Architecture+ application auto-tuning for in-memory data-processing frameworks. In *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)* (2019), IEEE, pp. 163–166.

[30] Wang, H., Sayadi, H., Kolhe, G., Sasan, A., Rafatirad, S., and Homayoun, H. Phased-guard: Multi-phase machine learning framework for detection and identification of zero-day microarchitectural side-channel attacks. In *2020 IEEE 38th International Conference on Computer Design (ICCD)* (2020), IEEE, pp. 648–655.

[31] Wang, H., Sayadi, H., Sasan, A., Rafatirad, S., Mohsenin, T., and Homayoun, H. Comprehensive evaluation of machine learning countermeasures for detecting microarchitectural side-channel attacks. In *Proceedings of the 2020 on Great Lakes Symposium on VLSI* (2020), pp. 181–186.

[32] Yang, Q., Gasti, P., Zhou, G., Farajidavar, A., and Balagani, K. S. On inferring browsing activity on smartphones via usb power analysis side-channel. *IEEE Transactions on Information Forensics and Security 12*, 5 (2016), 1056–1066.

[33] Ylonen, T., Lonvick, C., et al. The secure shell (ssh) protocol architecture, 2006.