MicPrint: Acoustic Sensor Fingerprinting for Spoof-Resistant Mobile Device Authentication

Yongwoo Lee
University of Wisconsin–Madison
yongwoo.lee@wisc.edu

Jingjie Li
University of Wisconsin–Madison
jingjie.li@wisc.edu

Younghyun Kim
University of Wisconsin–Madison
younghyun.kim@wisc.edu

ABSTRACT

Smartphones are the most commonly used computing platform for accessing sensitive and important information placed on the Internet. Authenticating the smartphone's identity in addition to the user's identity is a widely adopted security augmentation method since conventional user authentication methods, such as password entry, often fail to provide strong protection by itself.

In this paper, we propose a sensor-based device fingerprinting technique for identifying and authenticating individual mobile devices. Our technique, called MicPrint, exploits the unique characteristics of embedded microphones in mobile devices due to manufacturing variations in order to uniquely identify each device. Unlike conventional sensor-based device fingerprinting that are prone to spoofing attack via malware, MicPrint is fundamentally spoof-resistant since it uses acoustic features that are prominent only when the user blocks the microphone hole. This simple user intervention acts as implicit permission to fingerprint the sensor and can effectively prevent unauthorized fingerprinting using malware. We implement MicPrint on Google Pixel 1 and Samsung Nexus to evaluate the accuracy of device identification. We also evaluate its security against simple raw data attacks and sophisticated impersonation attacks. The results show that after several incremental training cycles under various environmental noises, MicPrint can achieve high accuracy and reliability for both smartphone models.

CCS CONCEPTS

• **Security and privacy → Multi-factor authentication**; • **Human-centered computing → Smartphones**;

KEYWORDS

Device fingerprinting, multi-factor authentication, sensor, microphone

ACM Reference Format:
Yongwoo Lee, Jingjie Li, and Younghyun Kim. 2019. MicPrint: Acoustic Sensor Fingerprinting for Spoof-Resistant Mobile Device Authentication . In *16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous), November 12–14, 2019, Houston, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3360774.3360801

MobiQuitous, November 12–14, 2019, Houston, TX, USA

ACM ISBN 978-1-4503-7283-1/19/11…$15.00
https://doi.org/10.1145/3360774.3360801

1 INTRODUCTION

The past decade has seen an unprecedentedly rapid growth of applications and services based on network-connected mobile computing platforms. The network connectivity has facilitated the development of various novel mobile applications and services but at the cost of acute security and privacy concerns due to sensitive and confidential data placed on the Internet. Researchers and security practitioners have sought strong security mechanisms to protect such data from malicious attacks, but current security methods have often failed to provide proper protection. It is not difficult to find incidents where security and privacy vulnerabilities in mobile systems have resulted in tangible damages—in a recent example, simple identity fraud led to hackers stealing 24 million dollars in cryptocurrencies from a crypto investor [12].

The key to protecting security- and privacy-sensitive information is controlling who accesses what information [33]. This is accomplished by proper authentication to verify the identity of the requesting user and/or device and its level of clearance to access. Traditional authentication methods that rely solely on password protection have failed to provide proper authentication. It is known that users often choose weak passwords or re-use passwords for different purposes [15]. A successful attack on Mozilla's Bugzilla bug tracker in 2014 that exposed the browser's 185 non-public vulnerabilities was due to that a privileged user had set the same password on another website that was breached [21]. Another study shows that about 98% of users have only three different passwords that they use frequently, and more than 50% of them never change the passwords unless required by the system [30], making the impact of password leaks broader.

As a countermeasure to this limitation, *multi-factor authentication* that requires authentication information in more than one modality has become popular in many applications where elevated security is demanded. For example, in addition to a password, an authenticator can verify another knowledge factor (i.e., "something the user knows") by asking additional security questions, such as mother's maiden name. However, this approach is still vulnerable to attacks by simple guessing or social engineering [5]. In addition, it often requires the user to enter a long answer to the additional security question, which tends to deteriorate the user experience. A promising alternative method is to verify a possession factor (i.e., "something the user has"), where the user must prove that he/she has something that is pre-registered with the authenticator, such as a one-time password (OTP) token and a smart card. However, carrying an additional device only for authentication purpose is not a convenient solution. Instead, using the unique identification of the device itself, i.e., the device's *fingerprint vector*, as a possession factor is gaining popularity for its convenience.

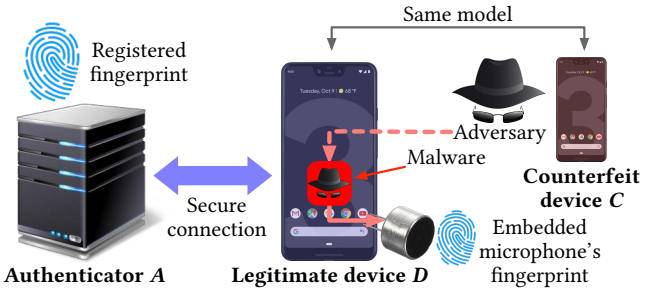


Figure 1: System model.

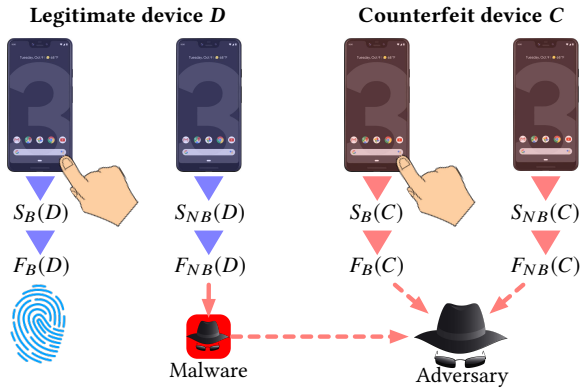


Figure 2: Acoustic fingerprints of D and C, with the microphone hole blocked or not blocked.

To accomplish reliable mobile device-based multi-factor authentication, it is crucial to generate a secure and reliable fingerprint vector to de-anonymize and identify each device. A fingerprint vector of a device can be generated from its software settings, network status, or hardware components. *Sensor-based fingerprinting* is a promising approach that falls into the category of hardware-based approaches, which exploits the fact that each sensor device has a unique characteristic that is unpredictable and unreproducible due to variations in hardware manufacturing process. A variety of device fingerprinting methods have been proposed for mobile devices using their rich sensors, such as accelerometers [4, 13], gyroscopes [10], and microphones [9].

The key challenge in realizing strong authentication based on mobile device fingerprinting is to prevent spoofing attacks, where the attacker obtains and alters the fingerprint vector to impersonate the victim. Unfortunately, conventional sensor-based fingerprint vector generation processes are prone to this type of attacks. Mobile applications can access sensors any time as long as they obtain necessary permissions when installed. An adversary can easily make malware that obtains legit-looking permission to access the sensor used for fingerprinting. Once such malware is installed, it is difficult (if not impossible) to distinguish the legitimate, functional use of the sensor from the malicious use. For example, a game app that uses an accelerometer will be given permission to access the accelerometer and can stealthily generate the fingerprint of it. To make it worse, most users tend to grant unnecessary permissions without understanding the implications and never revisit them later on [19]. As a result, sensor-based device fingerprinting has been proposed mainly for device tracking, rather than authentication, and only in limited scenarios has it been proposed to use for authentication, e.g., under the assumption of limited attacker capabilities or close physical proximity between the device and the authenticator.

In this paper, we introduce a *spoof-resistant sensor-based device fingerprinting method* that addresses this challenge for realizing strong mobile device authentication. More specifically, we exploit the unique fingerprint of a built-in acoustic sensor (i.e., microphone) that can be generated *only when the user explicitly consents* by blocking the microphone hole with his/her finger. We show that, when blocked with a finger, microphones exhibit different acoustic characteristics that cannot be reconstructed from acoustic characteristics captured without blocking it. With an action that is as simple as using a (biometric) fingerprint sensor, the proposed method prevents unauthorized sensor fingerprinting even if an app has permission to access the sensor. The proposed method can be applied to commodity smartphones without any hardware or kernel modification.

2 SYSTEM AND THREAT MODELS

In this section, we specify the system model of MicPrint for device authentication and describe three threat models that we consider within the system model.

2.1 System Model

The system model we assume in this paper consists of three entities, as shown in Figure 1: a legitimate mobile device (D) possessed by a legitimate user, a counterfeit mobile device (C) possessed by an adversary, and an authenticator (A). Both D and C are of the same model and have an embedded microphone. From the perspective of security, we assume the following: (i) D and A are connected via secure wired and wireless networks; (ii) A is protected by proper security mechanisms; (iii) D is vulnerable to malware through which the adversary can access the microphone anytime; and (iv) the microphone on D can be accessed anytime by any of its apps with proper permission, but only one app at a time can access the microphone.

As shown in Figure 2, we let S_B and F_B be an audio sample recorded with the microphone hole blocked and its acoustic feature vector, respectively. Similarly, S_{NB} and F_{NB} denote an audio sample recorded with the microphone hole open and its acoustic feature vector, respectively. As discussed above, the adversary can obtain $F_B(C)$ and $F_{NB}(C)$ directly from its own device C, and $F_{NB}(D)$ from the victim's device D via the malware. Ideally, A should accept an authentication request only when $F_B(D)$ is presented by D and reject all other requests.

2.2 Threat Model

The goal of the adversary is to impersonate the legitimate user to gain access to A. We assume that malware is already installed on D and has microphone access permission. We consider three

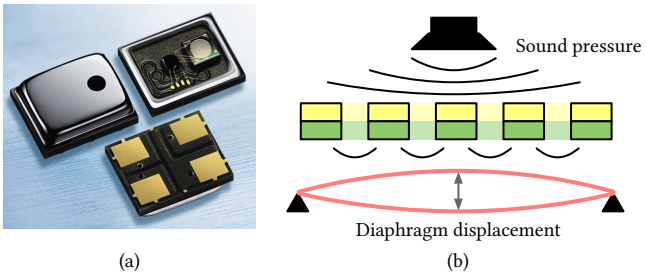


Figure 3: (a) Infineon silicon MEMS microphone [11]. (b) Internal structure of a MEMS microphone.

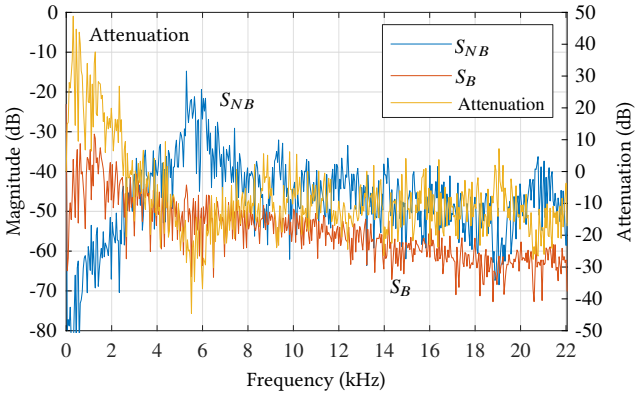


Figure 4: Spectrum of recorded audio samples with the microphone hole blocked (S_B) and with it open (S_{NB}).

hypothetical yet realistic threat models: passive raw audio attack, active raw audio attack, and synthesis attack.

Passive raw audio attack. The simplest attack method we consider is passive raw audio attack where the adversary uses $F_B(C)$ as a counterfeit fingerprint of D. This attack assumes that two devices of the same model will exhibit the same acoustic fingerprint, i.e., $F_B(D) = F_B(C)$. This simple attack does not even require the adversary to install malware on D. If the attack is successful, it is the most effective attack that any device of the same model as C can be a victim, even without installing malware.

Active raw audio attack. Active raw audio attack is another simple attack where the adversary captures raw audio samples of D through malware to stealthily obtain $F_{NB}(D)$. The adversary uses $F_{NB}(D)$ instead of $F_B(D)$ to be authenticated by A as D. The malware can use the microphone virtually anytime to capture audio samples in various environments, but not when the microphone hole is blocked because collecting acoustic samples with a blocked hole is done by the user only when he/she intends to generate the fingerprint using a legitimate app. Note that this attack model assumes the same attack capability as in the spoofing attack against conventional sensor-based device fingerprinting. This attack relies on the assumption that acoustic fingerprint will not be changed by blocking microphone hole, i.e., $F_B(D) = F_{NB}(D)$. If successful, the adversary can attack any device by just infecting victims with malware without having to physically acquire a counterfeit device of the same model.

Synthesis attack. The most sophisticated attack we consider is synthesis attack where the adversary synthesize $F_B(D)$ using $F_{NB}(D)$, $F_B(C)$, and $F_{NB}(C)$. This attack requires both malware installed on D and the possession of a physical device of the same model C. The adversary uses $F_{NB}(D)$ acquired from D via malware when ambient noise is minimal (e.g., during nighttime) to minimize the potential impact of the ambient noise. Next, the adversary acquires $F_B(C)$ and $F_{NB}(C)$ from C in the same environmental noise and obtains a linear transfer function f that converts $F_{NB}(C)$ to $F_B(C)$. Finally, the synthesized $F_B(D)$ is obtained by applying f to $F_{NB}(D)$. The assumption that this attack relies on is that f obtained from C is also applicable to D if the ambient noise is the same, i.e., $F_B(C) = f(F_{NB}(C)) \Rightarrow F_B(D) = f(F_{NB}(D))$.

3 ACOUSTIC FINGERPRINTING

In this section, we discuss the source of unique randomness utilized in acoustic fingerprinting.

3.1 Manufacturing Variability of Microphones

Micro-electro-mechanical systems (MEMS) microphones are fabricated on silicon wafers through the semiconductor production process and offer low cost, low power consumption, and very small packages, making them attractive for smartphones with stringent power and form-factor constraints. MEMS microphones measure the displacement of a diaphragm in response to sound pressure and subsequently converts it to a voltage signal measured by an analog-digital converter (ADC). Figures 3(a) and 3(b) show an example of a commodity MEMS microphone and the typical internal structure of a MEMS microphone, respectively.

Each microphone exhibits subtle but consistent differences in how the diaphragm response to the same sound pressure [9]. The variability is induced by semiconductor fabrication imperfections that result in the mechanical deformation of the diaphragm and gap between rigid back-plate and the diaphragm, and so on. The mechanical variability manifests as variability in the sound recorded by the microphone, and, in turn, the acoustic features of the recorded sound.

3.2 Acoustic Fingerprint in MicPrint

In this section, we first discuss the background of the proposed acoustic fingerprinting method. The source of uniqueness we exploit in MicPrint is the embedded microphone's acoustic fingerprint that is prominent only when the microphone hole is blocked by a finger, but not when it is open. As discussed above, each microphone has a unique fingerprint that is distinguishable from others when the hole is not blocked, but relying on F_{NB} makes the fingerprint vulnerable to spoofing via malware. To implement a secure fingerprinting and authentication method, a new fingerprint F_B, captured with the hole blocked, should meet the following requirements: (i) F_B of each device should be unique, and (ii) one should not be able to easily produce F_B of a device from F_{NB} of the same

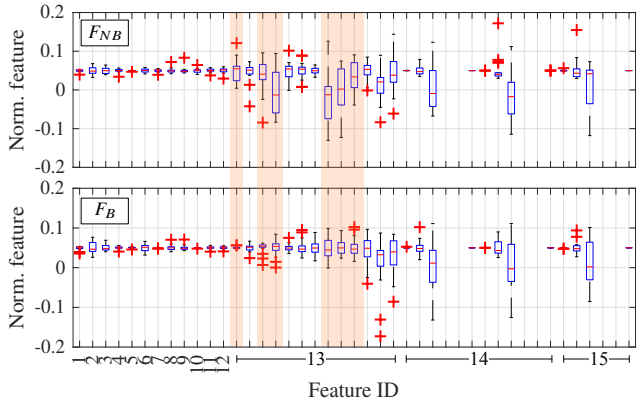


Figure 5: Distributions of 15 acoustic features of F_{NB} and F_B of the same device. Discriminating features are highlighted.

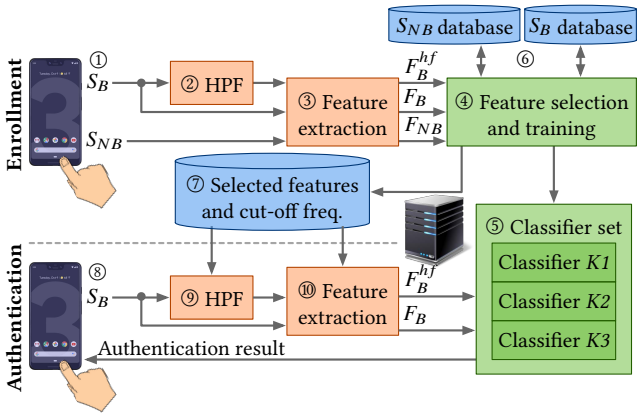


Figure 6: MicPrint design.

device. Since the uniqueness of microphones originates from fundamental hardware imperfection, we can assume the uniqueness of F_B regardless of the blockage of the hole.

The second requirement is explored by a simple preliminary study. We record an audio sample in an environmental noise (noise source "Lounge" from SoundJay [3]) for 1 s with the microphone hole blocked (S_B) and with it open (S_{NB}), using the same smartphone. Figure 4 shows their sound spectrums and the attenuation by the blockage. We can see that attenuation from S_{NB} to S_B is not linear across the frequency range. Moreover, we can observe additional sound added by the finger in the low-frequency range. In Figure 5, the distribution of 15 acoustic features of 20 samples of F_B and F_{NB} is presented[1]. Each sample is 1-s long, and each feature vector is normalized by its 1-norm. We can see some features, such as the highlighted ones, show clear discrimination between F_B and F_{NB}. Therefore, an adversary cannot correctly produce F_B by simple manipulation of F_{NB} obtained via malware. Later in this paper, we also show that even a sophisticated synthesis attack that uses a linear transfer function cannot successfully produce correct F_B.

4 MICPRINT DESIGN

In this section, we describe the design of MicPrint for secure device fingerprinting and authentication.

4.1 Overall Design

Device authentication by MicPrint consists of two phases, as shown in Figure 6: an enrollment phase and an authentication phase. The *enrollment phase* is a process in which a legitimate user registers a legitimate device D to the authenticator A. First, the authentication app on D captures two short audio samples, $S_B(D)$ and $S_{NB}(D)$, and sends them to A. Then, A extracts acoustic features $F_B(D)$ and $F_{NB}(D)$ from the audio samples and trains a set of authentication classifiers that will accept only $F_B(D)$ and reject everything else. The *authentication phase* is a process in which A verifies the authenticity of D using the authentication classifier set. In this phase, D records only $S_B(D)$ and sends it to A, and A extracts its feature vector $F_B(D)$. Finally, A authenticates D if it is recognized by the classifier set, based on that $F_B(D)$ can be generated only when the legitimate user explicitly indents authentication. In case that MicPrint fails to authenticate due to environmental noise, A requests a more reliable but less convenient authentication factor, such as a security question. Once authenticated, the classifier set is retrained with additional $S_B(D)$ and $S_{NB}(D)$ to improve recognition rate. The rest of this section describes each process in more detail.

[1]Description of the acoustic features is presented later in Table 1.

4.2 Audio Sampling and Feature Extraction

Both the enrollment and authentication phases begin with generating an acoustic fingerprint vector from recorded audio samples. Since it is a compute-intensive process, the required computation is done by A that has more computing performance than D. We use 15 acoustic features that are widely used for acoustic fingerprinting [9], which capture both temporal and spectral characteristics of a signal as shown in Table 1.

For enrollment, s samples of S_B and s samples of S_{NB} of duration T are captured on D and sent to A (Step ①), where s is the number of audio samples. A high-frequency filter (HPF) is applied to S_B to remove low-frequency acoustic artifacts added by finger's blocking (Step ②), as discussed in Section 3. Next, we extract 15 features each from S_B, high-pass-filtered S_B, and S_{NB}, to generate F_B, F_B^{hf}, and F_{NB}, respectively (Step ③).

For authentication, s samples of S_B of duration T each are captured on D and sent to A (Step ⑧). S_{NB} is not required. Similar to enrollment, A applies an HPF (Step ⑨) and feature extraction (Step ⑩) to generate F_B^{hf} and F_B.

4.3 Feature and Cut-off Frequency Selection

To improve the accuracy of classification, we select a subset of acoustic features that best discriminates D from other devices, rather than using all the features [34, 36] (Step ④). Since there are 15 features, selecting the optimal subset of n features in a brute force manner requires evaluations of ${}_{15}C_n$ combinations. We find that $n = 2$ provides an acceptable accuracy at a reasonable training

Table 1: ID, dimension, and description of acoustic features used [9].

ID	Dim.	Feature name: Description
1	1	Root mean square (RMS): RMS of amplitude
2	1	Zero crossing rate (ZCR): Rate at which the sign of the signal changes
3	1	Low energy rate: Frame rate less than average energy (RMS)
4	1	Spectral centroid: Center of mass of the spectrum
5	1	Spectral entropy: Predominant peaks of the spectrum and their location
6	1	Spectral irregularity: Degree of variation of the successive peaks of the spectrum
7	1	Spectral spread: Standard deviation of the spectrum
8	1	Spectral skewness: Ratio of skewness to standard deviation
9	1	Spectral kurtosis: Spikiness or flatness of distribution compared to normal distribution
10	1	Spectral rolloff: Frequency corresponding to 85% of total energy
11	1	Spectral brightness: Amount of energy above cut-off frequency
12	1	Spectral flatness: Smoothness or spikiness of distribution
13	13	MFCCs: Mel-frequency cepstral coefficients
14	12	Chromagram: Distribution of energy along the pitches or pitch classes
15	6	Tonal centroid: 6-dimensional tonal centroid vector from chromagram

time. For each classifier, we evaluate the accuracy with all 15 features and ${}_{15}C_2 = 105$ combinations of two features and select the best-performing feature set. We also evaluate two different HPF cut-off frequencies, 5 kHz and 10 kHz, and select one that exhibits higher accuracy. The selected feature IDs and cut-off frequency is stored in A (Step ⑦) and used later during authentication to perform high-pass filtering (Step ⑨) and feature extraction (Step ⑩) on $S_B(D)$.

4.4 Cascaded Classification

The authenticity of the feature vectors is verified by a set of three cascaded binary classifiers, $K1$ and $K2$ trained for all legitimate devices, and $K3$, trained for each individual device (Step ⑤). The three classifiers collectively verify that the audio sample submitted by the device is genuinely recorded by D with its microphone hole blocked. If the audio sample is recorded by D but with the microphone hole open, if it is a synthesized one, or if it is a recorded using a different device, the classifier set rejects the authentication request. All three classifiers are implemented using a binary multilayer perceptron (MLP) network.

The first classifier $K1$, used for all legitimate devices, identifies whether an input audio sample is recorded with the microphone hole blocked or open. If the input audio sample is recorded with the microphone hole open, A determines that it is an active raw audio attack and rejects the authentication request without running $K2$ and $K3$. Since the difference between S_B and S_{NB} is most prominent in the low-frequency range, F_B without high-pass filtering is used as input. To train $K1$, $s\ F_B(D)$ samples are used as true samples and $s\ F_{NB}(D)$ samples are used as false samples. Also, other devices' audio samples in the databases of S_B and S_{NB} (Step ⑥) are used to train $K1$. The databases are incrementally updated when a new environmental noise source is introduced, especially when MicPrint fails to authenticate a legitimate device because of it.

The second classifier $K2$, used for all legitimate devices, identifies whether or not F_B is a synthesized acoustic feature vector. This classifier is trained by potential synthesized acoustic feature vectors generated using F_B and F_{NB} of other devices of the same model. More specifically, a transfer function $f : F_{NB} \rightarrow F_B$ is generated for each pair of F_B and F_{NB} of the other devices, and then f is used to generate false (synthesized) samples of $F_B(D)$ by applying it to $F_{NB}(D)$. The databases of S_B and S_{NB} are used for training $K2$.

Finally, the last classifier $K3$ distinguishes whether or not F_B^{hf} of D is as claimed by the requester. Unlike $K1$ and $K2$ that are system-wide classifiers, $K3$ is a device-specific classifier. $K3$ trained for D thwarts passive raw audio attack against D by rejecting other devices' S_B and accepting only $S_B(D)$. This classifier is trained using F_B^{hf} of D as true samples and F_B^{hf} of other devices of the same model as false samples. The database of S_B is also used for training $K3$.

Since the classifiers in MicPrint are used for authentication purpose, false negatives (i.e., inconvenience) are more tolerable than false positives (i.e., insecurity). As mentioned in 4.1, if the authenticity of the device cannot be verified by MicPrint, i.e., negative classification is made by either $K1$, $K2$, or $K3$, the user is requested to use a different authentication method that is more reliable (but potentially less convenient), such as entering a password. Once the authenticity has been verified by the secondary method, the databases of S_B and S_{NB} are updated with new audio samples and the classifiers are retrained to reduce false negative rate. In Section 5.2, we demonstrate the improvement of classification accuracy as the classifiers get retrained with more environmental noise sources.

5 EVALUATION

In this section, we verify MicPrint on commercial off-the-shelf (COTS) smartphones and evaluate its fingerprinting accuracy as well as security implications.

5.1 Experiment Setup

We implement and install the MicPrint app on twelve Android smartphones of two models: eight Google Pixel 1 and four Samsung Nexus. We assume identifying the device model is straightforward, so we focus on identifying individual devices within the same model. Audio recording function is implemented using standard Android application programming interface (API) without kernel modification. We capture 20 samples of S_B and 20 samples of S_{NB} per device and per ambient noise, where each sample is 1-s long ($T = 1\ s$).

Figure 7: Experimental setup: (a) Experimental setup including smartphones used. (b) Blocking microphone holes of Google Pixel 1 and Samsung Nexus using a finger.

In order to simulate favorable attack environment where the microphone hole is accidentally blocked by an object other than a finger, we covered the hole with a soft cloth while recording S_{NB}. All audio samples are captured at 44.1 kHz as 16-bit pulse code modulation (PCM) in WAV format. MIRtoolbox [26] and Weka [25] are used to extract and classify audio features in Matlab.

Ambient noises. To evaluate the impact of ambient noises, nine ambient noise sources in Table 2 are considered. The noise samples are played using an external speaker at a distance of 5 cm from the microphone. The maximum and minimum sound pressure levels (SPL) measured at the location of the microphone vary among noise samples; in a quiet environment ("Quiet room"), it is approximately 44 dB, and with an environmental noise ("Kids playing"), it can be up to 88 dB.

Evaluation metrics. As a device authentication method, the usability and security of MicPrint can be quantified using *true positive rate* (*TPR*) and *false positive rate* (*FPR*). *TPR* is defined as

$$TPR = \frac{TP}{TP + FN}, \quad (1)$$

where TP is the number of true positive classifications, and FN is the number of false negative classifications. It indicates the probability that a legitimate device is successfully authenticated and thus is a measure of *usability*. On the other hand, *FPR* is defined as

$$FPR = \frac{FP}{FP + TN}, \quad (2)$$

Table 2: SPL of ambient noises. Noise samples 1 through 8 are from [3], and noise source 9 is an in-house noise sample.

Noise ID	Environments	Min. SPL (dB)	Max. SPL (dB)
1	Airport gate	57.0	73.9
2	Bus interior 1	64.1	67.7
3	Crowd talking 1	69.3	81.1
4	Food court	55.5	72.3
5	Kids playing	64.3	88.5
6	Lounge	54.3	66.7
7	Metro interior 1	64.4	81.5
8	Restaurant	51.0	77.7
9	Quiet room	43.2	44.1

where FP is the number of false positive classifications, and TN is the number of true negative classifications. It indicates the probability that the authenticator fails to reject a counterfeit device and thus is a measure of *security*. Throughout this section, we use per-model average *TPR* and *FPR*.

5.2 Classification Accuracy Evaluation

We first evaluate the classification accuracy of the individual classifiers. Two factors that affect the *TPR* and *FPR* are considered: (i) the number of authentication attempts made, c, and (ii) the number of audio samples, s. As c increases, i.e., the increasing number of authentication attempts made, more acoustic fingerprints under various ambient noises are added to the databases of S_B and S_{NB}, making MicPrint more robust and reliable. For the same reason, as s increases, i.e., as more number of audio samples are captured each time, the accuracy of MicPrint increases, but at the cost of slightly reduced usability due to a longer authentication time. In our experiments, we vary s among 1, 3, or 5, and c ranges from 1 to 9. We also assume that, when a new device D enrolls, the databases of S_B and S_{NB} do not have samples from D, but they have a sufficient number of samples captured from other devices and that they are stored securely.

First, we validate how accurately classifier $K1$ thwarts active raw audio attack by accepting F_B and rejecting F_{NB}. In each enrollment of D, s samples of $S_B(D)$ and the same number of samples of $S_{NB}(D)$ are added to the databases. Then, $K1$ is retrained. The first row of Figure 8 shows *TPR* and *FPR* of Google Pixel 1 and Samsung Nexus. The results show that $K1$ achieves high *TPR* above 90% and low *FRP* below 5% across all c and s, except for Google Pixel 1 when $s = 1$. More specifically, $K1$ for Google Pixel 1 achieves 98% *TPR* and 3% *FPR* after $c = 1$ (i.e., from the first use of MicPrint) when $s = 3$ (i.e., the user needs to block the microphone hole only for three seconds). Similarly, $K1$ for Samsung Nexus achieves 94% *TPR* and 1% *FPR* after $c = 3$ when $s = 3$. The high accuracy in distinguishing S_B versus S_{NB} is mainly due to the fact that the distinct difference between their acoustic features.

Next, we evaluate classifier $K2$. This classifier rejects synthesized acoustic fingerprints but accepts unmodified acoustic fingerprints. In each enrollment of D, s samples of $S_B(D)$ are added to the database to retrain $K2$. The second row in Figure 8 shows *TPR* and

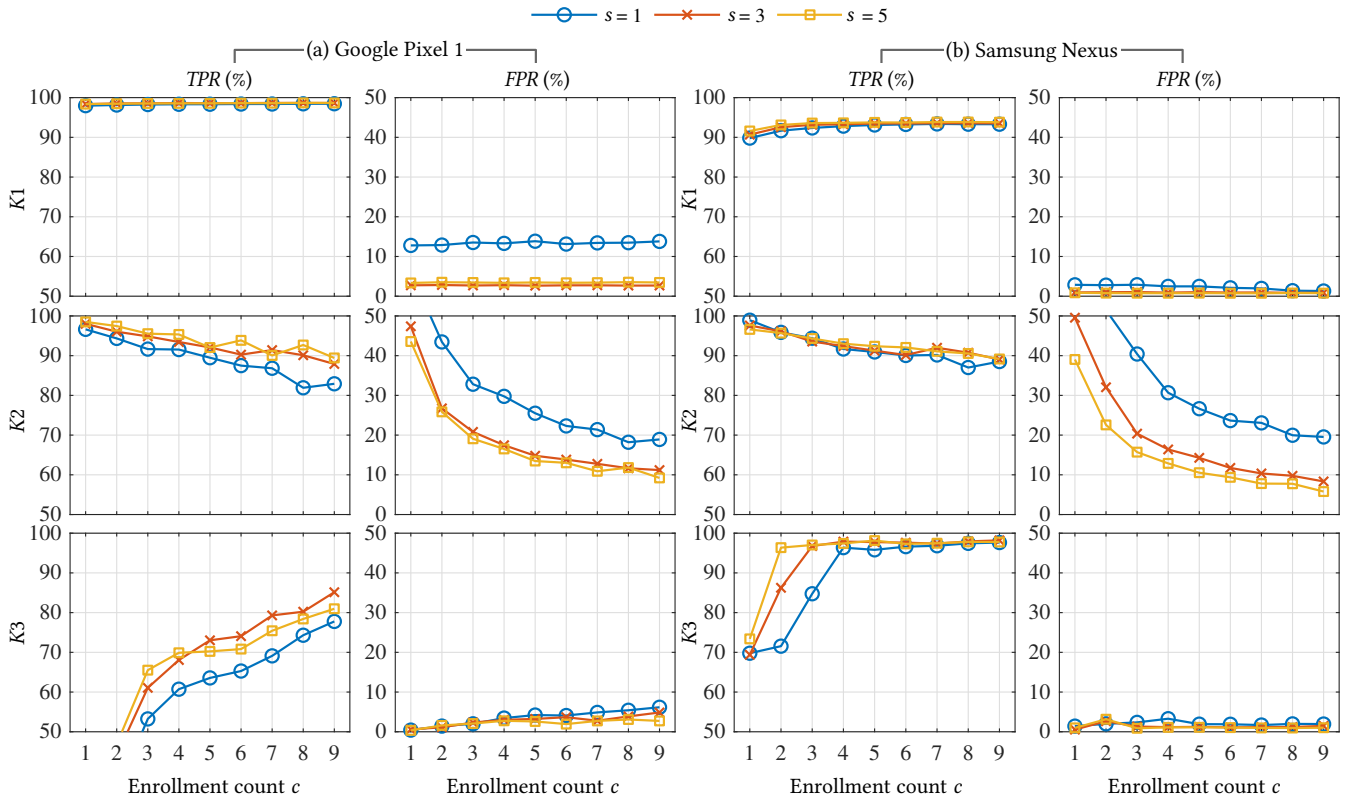


Figure 8: *TPR* and *FPR* of individual classifiers for (a) Google Pixel 1 and (b) Samsung Nexus.

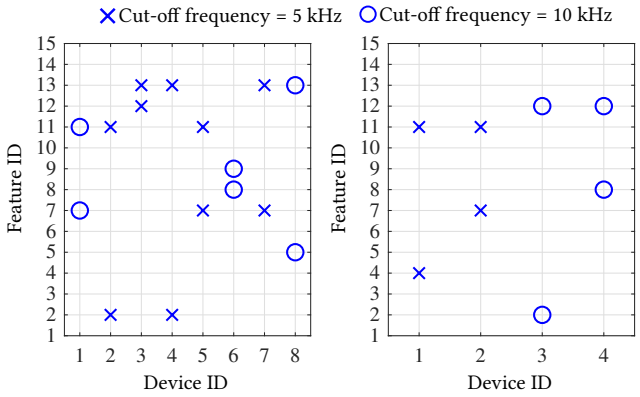
FPR of $K2$. We can see that as $K2$ is retained with more samples (i.e., as c increases) its *FPR* rapidly decreases while *TPR* slightly decreases. Initially, when there are no sufficient samples of $S_B(D)$ under various environmental noises, $K2$ is too permissive that *FPR* is over 50%. However, after a few more rounds of enrollment, *FPR* dramatically decreases, eventually down to around 10% when $c = 9$ and $s = 3$ or 5. In the meantime, *TPR* of $K2$ is maintained around 90%.

Finally, we evaluate the accuracy of device-specific classifier $K3$, which accepts only D's acoustic fingerprints F_B^{hf} and rejects other devices' F_B^{hf}. In each enrollment of D, new $S_B(D)$ is added to the database, and $K3$ is retrained. As discussed in Section 4.3, during the retraining of $K3$, the optimal subset of acoustic features and the cut-off frequency for each device are selected. Figure 9 is an example of selected features and the cut-off frequency of Google Pixel 1 and Samsung Nexus after the 9th enrollment ($c = 9$) and when $s = 5$. We can see that the optimal feature subset is different from device to device. Figure 10 shows the selection frequency of each acoustic feature as a result of the retraining. We can see that different models show different most effective features: mel-frequency cepstral coefficients (feature ID 13) and spectral spread (feature ID 7) are most frequently used for Google Pixel 1, while spectral brightness (feature ID 11) and zero crossing rate (feature ID 2) are most frequently used for Samsung Nexus. Therefore, we find the optimal set of acoustic features for each device during the enrollment phase and use it for reliable authentication. The plots in the last row of Figure 8 show *TPR* and *FPR* of $K3$. As $K3$ is retained with more samples (i.e., as c increases) its *TPR* rapidly increases while *FPR* slightly increases. For Google Pixel 1, $K3$ is initially too restrictive that *TPR* is less than 50%. However, after a few more rounds of enrollment, *TPR* increases dramatically, eventually up to around 80% when $c = 9$ for $s = 3$ or 5. In the meantime, *TPR* of $K3$ is maintained below 10%. The results for Samsung Nexus show even better accuracy. When $s = 3$, *TPR* of $K3$ for Samsung Nexus exceeds 95% after three enrollments ($c = 3$), and *TPR* is maintained below 5%.

Overall, we can conclude that the incremental training of the classifiers effectively improves *TPR* and *FPR* to a usable level within less than ten enrollments under different environmental noises. The selection of $s = 5$ seems reasonable considering the short authentication time required to capture audio samples and reach a satisfactory accuracy regarding different classifiers and models, while $s = 3$ also shows comparable accuracy.

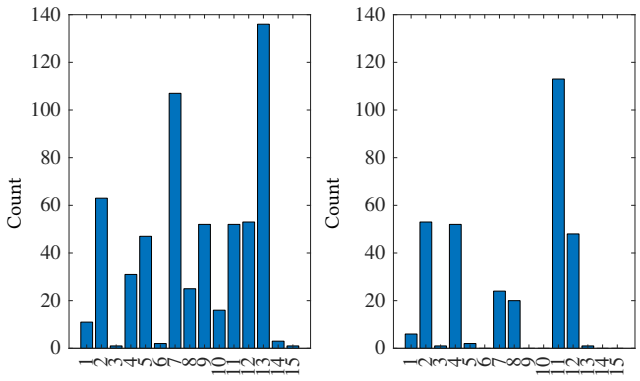
5.3 Authentication Accuracy Evaluation

Finally, to validate MicPrint as a device authentication method, the accuracy of classification of all three classifiers combined is evaluated. We use the classifiers that are trained with nine different



(a) Google Pixel 1 (b) Samsung Nexus

Figure 9: An example of an optimal subset of acoustic features after 9-th enrollment ($c = 9$) when five audio samples are captured in the enrollment ($s = 5$).

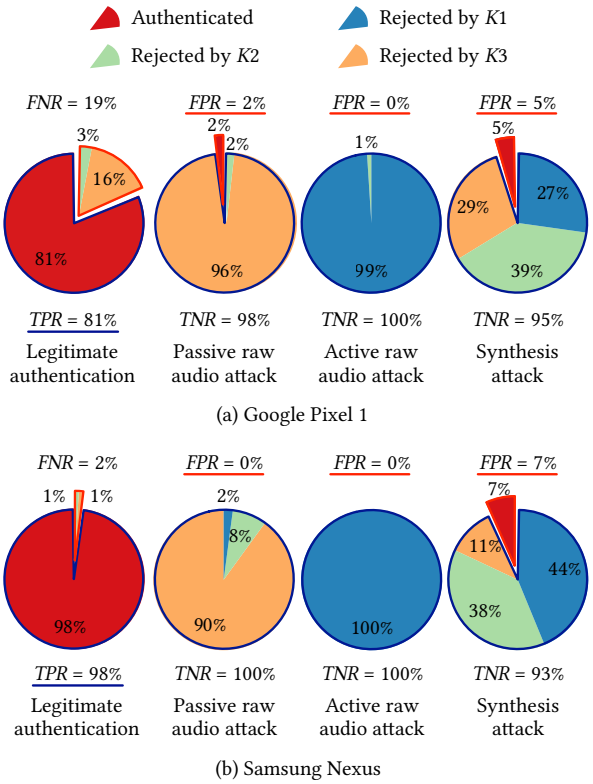


(a) Google Pixel 1 (b) Samsung Nexus

Figure 10: Selection frequency of acoustic features for (a) Google Pixel 1 and (b) Samsung Nexus in all cases.

environmental noises and five audio samples, and in each authentication attempt, we record the same number of audio samples ($s = 5$). We evaluate average TPR and FPR of all devices for legitimate audio samples S_B as well as counterfeit audio samples under three attack scenarios discussed in Section 2.2. We also evaluate FNR and TNR, where FNR is false negative rate (i.e., $FNR = 1 - TPR$) and TNR is true negative rate (i.e., $TNR = 1 - FPR$). For each device and each authentication scenario, 36 authentication and attack attempts are made. As discussed in Section 4.4, each audio sample is sequentially classified by $K1$, $K2$, and $K3$. Only when at least three out of five samples per authentication request are classified to be legitimate by all three classifiers, A will authenticate the requester, otherwise, A will reject the requester.

The first column of Figures 11(a) and 11(b) shows the average TPR of authentication of legitimate devices. The overall of TPR of MicPrint is 81% for Google Pixel 1 and 98% for Samsung Nexus. The



(a) Google Pixel 1

(b) Samsung Nexus

Figure 11: Authentication accuracy of MicPrint for Google Pixel 1 and Samsung Nexus.

results show a good agreement with TPR of individual classifiers discussed in Section 5.2. The second, third, and fourth column in Figures 11(a) and 11(b) show the overall FPR of passive raw audio attack, active raw audio attack, and synthesis attack, respectively. Against passive raw audio attack, MicPrint achieves 2% FPR for Google Pixel 1 and 0% FPR for Samsung Nexus, as shown in the second column of Figures 11(a) and 11(b). This attack bypasses $K1$ and $K2$, but $K3$ successfully thwarts it. Next, against active raw audio attack, MicPrint achieves 0% FPR for both Google Pixel 1 and Samsung Nexus. This attack is mostly thwarted by $K1$ which rejects S_{NB}, but $K2$ also contributes to the zero FPR by rejecting any counterfeit attempts that $K1$ missed. This result shows that MicPrint is almost immune to passive raw audio attack, which is the main difference between MicPrint and existing sensor-based device authentication techniques. Finally, MicPrint achieves 5% FPR for Google Pixel 1 and 7% FPR for Samsung Nexus against synthesis attack, as shown in the fourth column of Figures 11(a) to 11(b). As shown in the figure, all classifiers, $K1$, $K2$ and $K3$ contribute to rejecting this attack. Although $K3$ is designed to distinguish individual devices' acoustic fingerprint from each other, it can also discriminate synthesized ones from legitimate ones. Overall, for any of the three attack scenarios that we consider in this paper, MicPrint exhibits high TPR and low FPR.

6 DISCUSSION AND FUTURE WORK

User experience. Major factors that affect the user experience include the time required to complete authentication and the number of classifier retraining required to achieve acceptable *TPR* and *FPR*. The time required to complete authentication is $T \times s$ seconds. Both longer T and more s positively contribute to the faster improvement authentication accuracy, but they negatively impact user experience because the user is required to block the microphone hole for a longer time. As demonstrated in Section 5.2, when $T = 1$ and $s = 3$, each authentication attempt takes 3 seconds, and MicPrint achieves acceptable authentication accuracy after several authentication attempts.

Environmental noise. Strong environmental noise can negatively affect the accuracy of MicPrint. Also, environmental noise that is significantly different from previously observed ones can increase *FPR*. Therefore, it is important to retrain the classifiers with different environmental noises, especially $K2$ and $K3$. As demonstrated in Section 5.2, less than ten different environmental noises are sufficient to improve the robustness of the classifiers against noise. This means that the user may fail to be authenticated by MicPrint and may have to use an alternative authentication method up to ten times. Although this can pose some inconvenience, we argue that additional enrollment and retraining are required only when a new environmental noise is detected, and the only additional overhead of this is the use of the alternative authentication method. To further minimize this inconvenience, we could generate artificial environmental noises when we train the classifier for the first time. For example, all smartphones have a loud-speaker that can be used to play recorded environmental noises. By using different environmental noises for the initial training, the initial accuracy of classification can be greatly improved and a fewer number of additional enrollments will be required. The overhead of this approach will be longer initial training, but we can let the user make the decision.

7 RELATED WORK

Device fingerprinting methods can be divided into three categories based on the source of unique fingerprint vectors: software-based methods, network-based methods, and hardware-based methods.

Software-based methods utilize the patterns and traits of the installed software, such as browser settings [14, 22, 27], installed font list [28], and cookies [1]. These methods are relatively easy to deploy since they do not require hardware modification. *Network-based methods* rely on device IP address, network protocol fingerprints [6], and DNS resolver [7]. These methods are often client-passive that the authenticator does not require any explicit cooperation of the device and hence is even more convenient to deploy than software-based methods. However, both software- and network-based methods are inherently less stable than hardware-based methods because software and network configurations change over time due to software updates or device mobility. A significant change in software or network configurations will require the authenticator to temporarily use a less convenient but more reliable method to identify the device [2]. In addition, these methods are less resistant to spoofing since these configurations can be easily guessed or intercepted by attackers.

On the other hand, *hardware-based methods* rely on the random characteristics of a hardware component, sometimes called a physically unclonable function (PUF), that is due to inherent manufacturing process variations. Depending on the type of hardware component used for fingerprinting, hardware-based methods can be further categorized into those that rely on an extrinsic PUF and those that rely on an intrinsic PUF. Extrinsic PUFs are hardware components dedicated solely for device identification purpose, which can be implemented by exploiting the variability of path delay in an arbiter chain [16, 23] or frequency of a ring oscillator [24, 29, 31]. Extrinsic PUFs can generate a reliable fingerprint vector, but the addition of a new hardware component is costly and inapplicable to existing mobile devices.

A more cost-effective approach that can be generally applicable to existing mobile devices is to use an intrinsic component that already exists in the device but not for identification purposes, such as memory and sensors. Memory-based PUFs exploit random cell-to-cell variations, such as reset state variation or data retention time variation [32, 37]. Sensor-based PUFs are based on embedded sensors in mobile devices. For example, each motion sensor exhibits subtle differences in the response that can be identified using software [17, 18]. Image sensor-based methods exploit a spatially-random but temporally-invariant noise pattern in captured images, such as photo-response non-uniformity (PRNU) noise [35] or dark signal non-uniformity (DSNU) noise [20], to generate a unique fingerprinting vector. The key advantage of intrinsic PUF-based fingerprinting techniques is that it does not require extra hardware addition or modification. Since the source of randomness is an existing hardware component, any app can read the fingerprint from the hardware component using standard mobile API as long as the app has proper access permission.

Unfortunately, as discussed in Section 1, an adversary can also easily obtain the sensor fingerprints using the API if malware with necessary access permission can be installed. Therefore, these methods are proposed for device identification, rather than authentication, or for authentication in restricted scenarios with limited adversary capabilities. For example, in [8], the device and the authenticator should be located in close proximity so acoustic sound generated by the device's speaker can be captured by the authenticator's microphone. In [20, 35], it is assumed that the adversary cannot obtain raw (uncompressed) images from the victim, but if malware has access permission to the image sensor, raw images can easily be exploited by the adversary.

8 CONCLUSION

We introduced MicPrint, a secure and usable device authentication method based on acoustic fingerprint of microphones in mobile devices. It exploits the acoustic fingerprint of recorded audio samples obtained when a user intentionally blocks a microphone with a finger. MicPrint is universally applicable since microphones are ubiquitously embedded in virtually every mobile device. We implemented MicPrint on real smartphones without any hardware or software kernel modification, and demonstrated its performance against environmental variations. We also demonstrated

that MicPrint is secure against various impersonation attacks using audio samples from a device of the same model as the target device, even if the adversary can synthesize a large number of similar audio samples. In our experiments, MicPrint was able to accept legitimate devices' authentication requests and reject counterfeit devices' requests with high accuracy. We envision that MicPrint will be a promising mobile device authentication technique that is immediately applicable since it works on existing mobile devices with only a simple software implementation.

ACKNOWLEDGEMENTS

This work was supported by the Wisconsin Alumni Research Foundation and NSF under grants CNS-1719336 and CNS-1845469.

REFERENCES

[1] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. 674–689. https://doi.org/10.1145/2660267.2660347

[2] Furkan Alaca and P. C. van Oorschot. 2016. Device fingerprinting for augmenting web authentication: Classification and analysis of methods. In *Proceedings of the ACM Annual Conference on Computer Security Applications (ACSAC)*. 289–301. https://doi.org/10.1145/2991079.2991091

[3] SOUNDJAY ambient sound effects. Accessed 05/25/19. http://www.soundjay.com/ambient-sounds.html

[4] Hristo Bojinov, Yan Michalevsky, Gabi Nakibly, and Dan Boneh. 2014. Mobile device identification via sensor fingerprinting. *CoRR* abs/1408.1416 (2014). http://arxiv.org/abs/1408.1416

[5] Joseph Bonneau, Elie Bursztein, Ilan Caron, Rob Jackson, and Mike Williamson. 2015. Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at Google. In *Proceedings of the ACM International Conference on World Wide Web (WWW)*. 141–150. https://doi.org/10.1145/2736277.2741691

[6] Lee Brotherston. 2016. Stealthier attacks and smarter defending with TLS fingerprinting. https://github.com/LeeBrotherston/tls-fingerprinting.

[7] Web browser security. Accessed 05/25/19. https://browserleaks.com

[8] Dajiang Chen, Xufei Mao, Zhen Qin, Weiyi Wang, Xiang-Yang Li, and Zhiguang Qin. 2015. Wireless device authentication using acoustic hardware fingerprints. In *Proceedings of the International Conference on Big Data Computing and Communications (BIGCOM)*, Vol. 9196. 193–204. https://doi.org/10.1007/978-3-319-22047-5_16

[9] Anupam Das, Nikita Borisov, and Matthew Caesar. 2014. Do you hear what I hear?: Fingerprinting smart devices through embedded acoustic components. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. 441–452. https://doi.org/10.1145/2660267.2660325

[10] Anupam Das, Nikita Borisov, and Matthew Caesar. 2016. Tracking mobile web users through motion sensors: Attacks and defenses. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. https://doi.org/10.14722/ndss.2016.23390

[11] Alfons Dehé, Martin Wurzer, Marc Füldner, and Ulrich Krumbein. 2013. The Infineon silicon MEMS microphone. In *Proceedings of the AMA Conferences Sensor*. AMA, 94–99. https://doi.org/10.5162/sensor2013/A4.3

[12] Ian Demartino. 2018. A complete explanation of the Michael Terpin, AT&T Lawsuit. CoinJournal, https://coinjournal.net/a-complete-explanation-of-the-michael-terpin-att-lawsuit/.

[13] Sanorita Dey, Nirupam Roy, Wenyuan Xu, Romit Roy Choudhury, and Srihari Nelakuditi. 2014. AccelPrint: Imperfections of accelerometers make smartphones trackable. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. https://doi.org/10.14722/ndss.2014.23059

[14] Peter Eckersley. 2010. How unique is your web browser?. In *Proceedings of the International Conference on Privacy Enhancing Technologies (PET)*. 1–18. http://dl.acm.org/citation.cfm?id=1881151.1881152

[15] Dinei Florencio and Cormac Herley. 2007. A large-scale study of web password habits. In *Proceedings of the ACM International Conference on World Wide Web (WWW)*. 657–666. https://doi.org/10.1145/1242572.1242661

[16] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. 2002. Silicon physical random functions. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*. 148–160. https://doi.org/10.1145/586110.586132

[17] Tom Goethem, Wout Scheepers, Davy Preuveneers, and Wouter Joosen. 2016. Accelerometer-based device fingerprinting for multi-factor mobile authentication. In *Proceedings of the International Symposium on Engineering Secure Software and Systems (ESSOS)*. 106–121. https://doi.org/10.1007/978-3-319-30806-7_7

[18] Thomas Hupperich, Henry Hosseini, and Thorsten Holz. 2016. Leveraging sensor fingerprinting for mobile device authentication. In *Proceedings of the International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*. 377–396. https://doi.org/10.1007/978-3-319-40667-1_19

[19] Patrick Gage Kelley, Sunny Consolvo, Lorrie Faith Cranor, Jaeyeon Jung, Norman Sadeh, and David Wetherall. 2012. A conundrum of permissions: Installing applications on an android smartphone. In *Proceedings of the International Conference on Financial Cryptography and Data Security (FC)*. 68–79. https://doi.org/10.1007/978-3-642-34638-5_6

[20] Younghyun Kim and Yongwoo Lee. 2018. CamPUF: Physically unclonable function based on CMOS image sensor fixed pattern noise. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*. 66:1–66:6. https://doi.org/10.1145/3195970.3196005

[21] Eduard Kovacs. 2015. Unpatched firefox flaws exposed in bugzilla breach. https://www.securityweek.com/unpatched-firefox-flaws-exposed-bugzilla-breach.

[22] Pierre Laperdrix, Walter Rudametkin, and Benoit Baudry. 2016. Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. 878–894. https://doi.org/10.1109/SP.2016.57

[23] Lang Lin, Dan Holcomb, Dilip Kumar Krishnappa, Prasad Shabadi, and Wayne Burleson. 2010. Low-power sub-threshold design of secure physical unclonable functions. In *Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED)*. 43–48. https://doi.org/10.1145/1840845.1840855

[24] Abhranil Maiti, Jeff Casarona, Luke McHale, and Patrick Schaumont. 2010. A large scale characterization of RO-PUF. In *Proceedings of the IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*. 94–99. https://doi.org/10.1109/HST.2010.5513108

[25] Weka 3: Data mining software in Java. Accessed 05/25/19. https://www.cs.waikato.ac.nz/ml/weka

[26] MIRtoolbox. Accessed 05/25/19. https://www.jyu.fi/hytk/fi/laitokset/mutku/en/research/materials/mirtoolbox

[27] Keaton Mowery, Dillon Bogenreif, Scott Yilek, and Hovav Shacham. 2011. Fingerprinting information in JavaScript implementations. In *Proceedings of the IEEE Web 2.0 Security and Privacy (W2SP)*.

[28] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2013. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. 541–555. https://doi.org/10.1109/SP.2013.43

[29] Tauhidur Rahman, Domenic Forte, Jim Fahrny, and Mohammad Tehranipoor. 2014. ARO-PUF: An aging-resistant ring oscillator PUF design. In *Proceedings of the EDAA Design, Automation Test in Europe Conference Exhibition (DATE)*. 69:1–69:6. http://dl.acm.org/citation.cfm?id=2616606.2616692

[30] Shannon Riley. 2006. Password security: What users know and what they actually do. *Usability News* 8, 1 (2006), 2833–2836.

[31] G. Edward Suh and Srinivas Devadas. 2007. Physical unclonable functions for device authentication and secret key generation. In *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*. 9–14. https://doi.org/10.1145/1278480.1278484

[32] Fatemeh Tehranipoor, Nima Karimian, Wei Yan, and John A. Chandy. 2017. DRAM-based intrinsic physically unclonable functions for system-level security and authentication. *IEEE Transactions on Very Large Scale Integration Systems* 25, 3 (March 2017), 1085–1097. https://doi.org/10.1109/TVLSI.2016.2606658

[33] Tim Thornburgh. 2004. Social engineering: The "dark art". In *Proceedings of the ACM Conference on Information Security Curriculum Development (INFOSECCD)*. 133–135. https://doi.org/10.1145/1059524.1059554

[34] George Tzanetakis and Perry Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* 10, 5 (July 2002), 293–302. https://doi.org/10.1109/TSA.2002.800560

[35] Diego Valsesia, Giulio Coluccia, Tiziano Bianchi, and Enrico Magli. 2017. User authentication via PRNU-based physical unclonable functions. *IEEE Transactions on Information Forensics and Security* 12, 8 (August 2017), 1941–1956. https://doi.org/10.1109/TIFS.2017.2697402

[36] Michel Vidal-Naquet and Shimon Ullman. 2003. Object recognition with informative features and linear classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 281–288 vol.1. https://doi.org/10.1109/ICCV.2003.1238356

[37] Wenjie Xiong, Andre Schaller, Nikolaos Athanasios Anagnostopoulos, Muhammad Umair Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer. 2016. Run-time accessible DRAM PUFs in commodity devices. In *Proceedings of the Cryptographic Hardware and Embedded Systems (CHES)*, Vol. 9813. 432–453. https://doi.org/10.1007/978-3-662-53140-2_21