Do You Hear What I Hear? Fingerprinting Smart Devices Through Embedded Acoustic Components

Anupam Das
University of Illinois at Urbana-Champaign
das17@illinois.edu

Nikita Borisov
University of Illinois at Urbana-Champaign
nikita@illinois.edu

Matthew Caesar
University of Illinois at Urbana-Champaign
caesar@illinois.edu

ABSTRACT

The widespread use of smart devices gives rise to privacy concerns. Fingerprinting smart devices can jeopardize privacy by allowing remote identification without user awareness. We study the feasibility of using microphones and speakers embedded in smartphones to uniquely fingerprint individual devices. During fabrication, subtle imperfections arise in device microphones and speakers, which induce anomalies in produced and received sounds. We exploit this observation to fingerprint smartphones through playback and recording of audio samples. We explore different acoustic features and analyze their ability to successfully fingerprint smartphones. Our experiments show that not only is it possible to fingerprint devices manufactured by different vendors but also devices that have the same maker and model; on average we were able to accurately attribute 98% of all recorded audio clips from 50 different Android smartphones. Our study also identifies the prominent acoustic features capable of fingerprinting smart devices with a high success rate, and examines the effect of background noise and other variables on fingerprinting accuracy.

Categories and Subject Descriptors

K.6.m [**Management of Computing and Information Systems**]: Miscellaneous — Security; H.5.1 [**Multimedia Information Systems**]: Audio input/output

Keywords

Fingerprinting; Privacy; Acoustic feature; Microphone; Speaker

1. INTRODUCTION

Mobile devices, including smartphones, PDAs, and tablets, are quickly becoming widespread in modern society. In 2012 a total of 1.94 billion mobile devices were shipped, of which 75% were smart and highly-featured phones [5, 8, 14]. Canalys predicted that the mobile device market will reach 2.6 billion units by 2016, with smartphones and tablets continuing to dominate shipments [14]. The rapid uptake of intelligent mobile devices is not surprising, due to the numerous advantages they provide consumers, from entertainment and social applications to business and advanced computing capabilities. However, smartphones, with all their interactive, location-centric, and connectivity-based features impose threatening concerns on user privacy [9, 34, 39, 51].

In this paper we thoroughly analyze a technique for fingerprinting the *hardware* of smartphones. The observation is that even if the software on mobile devices is strengthened [35,62,70], hardware-level idiosyncrasies in microphones and speakers can be used to fingerprint physical devices. During manufacturing, imperfections are introduced in the analog circuitry of these components, and as such, two microphones and speakers are never alike. Through an observational study, we find that these imperfections are substantial enough, and prevalent enough, that we can reliably distinguish between devices by passively recording audio streams, and conducting simple spectral analyses on the recorded audio streams. Our approach can substantially simplify the ability for an adversary to track and identify people in public locations which can threaten the privacy of mobile device users. Our approach requires small amounts of data — for example, we show that with our technique, an adversary could even use the short ringtones produces by mobile device speakers to reliably track users in public environments. Alternatively, a stealthy app (e.g., an online game) can access the microphone to uniquely distinguish all users running the app.

Our approach centers around the development of a fingerprinting mechanism, which aims to "pull out" imperfections in device circuitry. Our mechanism has two parts: a method to extract auditory fingerprints and a method to efficiently search for matching fingerprints from a database. To generate fingerprints of speakers we record audio clips played from smartphones onto an external device (i.e., laptop/PC) and vice versa for generating fingerprints of microphones. We also generate the combined fingerprint of speaker and microphone by playing and recording audio clips simultaneously on smartphones. We use two different classifiers to evaluate our fingerprinting approach. Moreover, we test our fingerprinting approach for different genre of audio clips at various frequencies. We also study various audio features that can be used to accurately fingerprint smartphones. Our studies reveals that mel-frequency cepstral coefficients (MFCCs) are the dominant features for fingerprinting smartphones. Lastly, we analyze the sensitivity of our fingerprinting approach against different factors like sampling frequency, distance between speaker and recorder, training set size and ambient background noise.

Contributions. Our contributions are summarized below:

- We analyze the feasibility of fingerprinting smart devices by leveraging the manufacturing idiosyncrasies of microphones and speakers embedded in smart devices.

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.

http://dx.doi.org/10.1145/2660267.2660325.

- We study a large spectrum of existing audio features and their ability to accurately fingerprint smartphones. We find that mel-frequency cepstral coefficients (MFCCs) perform particularly well in fingerprinting smartphones.
- We investigate two different classifiers to evaluate our fingerprinting approach. We conclude that Gaussian mixture models (GMMs) are more effective compared to k-NN classifiers in classifying recorded audio fingerprints.
- We perform experiments across several different genres of audio excerpts. We also analyze how different factors like sampling frequency, distance between speaker and recorder, training set size and ambient background noise impact our fingerprinting accuracy.

Roadmap. The remainder of this paper is organized as follows. We give an overview of our fingerprinting approach in Section 2. We discussed related work in Section 3. In Section 4, we discuss why microphones and speakers integrated in smartphones can be used to generate unique fingerprints. In Section 5, we describe the different audio features considered in our experiments, along with the classification algorithms used in our evaluation. We present our experimental results in Section 6. We also discuss some limitations of our approach in Section 7. Finally, we conclude in Section 8.

2. OVERVIEW

In this section we give an overview of our approach and present several viable attack scenarios. We also identify the key challenges that we address in this paper.

The key observation behind our work is that imperfections in smart device hardware induce unique signatures on the received and transmitted audio streams, and these unique signatures, if identified, can be used by an adversary to fingerprint the device. We consider three fingerprinting scenarios: speaker, microphone, and joint speaker-microphone fingerprinting. In the first case, an attacker in a public environment, such as a cafe or shopping mall, records audio generated by a smartphone speaker, such as a ringtone. The attacker can then use the recorded audio samples to track and identify users. Alternately, the attacker may obtain audio recorded by a smartphone microphone and use that to identify the user who made the recording; this can have forensic applications. A third way to track users is to convince them to install a malicious application (e.g., a free online game), which can play and record audio clips using the device's speaker and microphone. The app can then stealthily upload the recorded audio clips to the attacker (e.g., piggybacking it on log-in information or game state), who can then use the audio samples to uniquely distinguish each user. To do this, the application would require access to both the speaker and microphone, as well as network access, but such permissions are not unusual for applications and are unlikely to raise alarm, especially given that a significant portion of the users cannot comprehend the full consequences of smartphone permissions [36, 45].

Our approach consists of two main tasks. The first task is acquiring a set of audio samples for analysis in the first place. To do this, we have a *listener* module, responsible for receiving and recording device audio. The listener module could be deployed as an application on the smart device (many mobile OSes allow direct access to microphone inputs), or as a stand-alone service (e.g., the adversary has a microphone in a public setting to pick up device ringtones). The next task is to effectively identify device signatures from the received audio stream. To do this, we have an *analyzer* module, which leverages signal processing techniques to localize spectral anomalies, and constructs a 'fingerprint' of the auditory characteristics of the device. A critical part of this task involves determining what sort of acoustic features and audio analysis techniques are most effective in identifying unique signatures of device-hardware. There are a large number of audio properties which could be used (spectral entropy, zero crossings, etc.) as well as a broad spectrum of analysis algorithms that can be used to summarize these properties (principle component analysis, linear discriminant analysis, feature selection, etc.). We will study alternative properties to characterize hardware-induced auditory anomalies in Section 5.1 as well as algorithms for effectively clustering them in Section 5.2.

3. RELATED WORK

Fingerprints have long been used as one of the most common bio-metrics in identifying human beings [27, 61]. The same concept was extended to identifying and tracking unique mobile transmitters by the US government during 1960s [47]. Later on with the emergence of cellular networks people were able to uniquely identify transmitters by analyzing the externally observable characteristics of the emitted radio signals [60].

Physical devices are usually different at either the software or hardware level even if they are produced by the same vendor. In terms of software based fingerprinting, researchers have looked at fingerprinting techniques that differentiate between unique devices over a Wireless Local Area Network (WLAN) simply through a timing analysis of 802.11 probe request frames [30]. Others have looked at exploiting the difference in firmware and device driver running on IEEE 802.11 compliant devices [37]. 802.11 MAC headers have also been used to track unique devices [40]. Pang et al. [57] were able to exploit traffic patterns to carry out device fingerprinting. Open source toolkits like Nmap [50] and Xprobe [68] can remotely fingerprint an operating system by identifying unique responses from the TCP/IP networking stack.

Another angle to software based fingerprinting is to exploit applications like browsers to carry out device fingerprinting [33]. Yen et al. [69] were successful at tracking users with high precision by analyzing month-long logs of Bing and Hotmail. Researchers have also been able to exploit JavaScript and popular third-party plugins like Flash player to obtain the list of fonts installed in a device which then enabled them to uniquely track users [18]. Other researchers have proposed the use of performance benchmarks for differentiating between JavaScript engines [54]. Furthermore, browsing history can be exploited to fingerprint and track web users [56]. The downside of software based fingerprints is that such fingerprints are generated from the current configuration of the system which is not static, rather it is likely to change over time.

Hardware based fingerprinting approaches rely on some static source of idiosyncrasies. It has been shown that network devices tends to have constant clock skews [53] and researchers have been able to exploit these clock skews to distinguish devices through TCP and ICMP timestamps [46]. However, clock skew rate is highly dependent on the experimental environment [67]. Researchers have also extensively looked at fingerprinting the unique transient characteristics of radio transmitters (also known as RF fingerprinting). RF fingerprinting has been shown as a means of enhancing wireless authentication [49, 55]. It has also been used for location detection [58]. Manufacturing imperfections in network interface cards (NICs) have also been studied by analyzing analog signals transmitted from them [21, 38]. More recently Dey et al. have studied manufacturing idiosyncrasies inside smartphone accelerometer to distinguish devices [31]. However, their approach requires some form of external stimulation/vibration to successfully capture the manufacturing imperfection of the on-board accelerometer. Moreover, there are different contexts in which audio prints can be more useful, e.g., software that is not allowed to access the accelerome-

ter, as well as an external adversary who fingerprints nearby phones with a microphone.

Our work is inspired by the above work in hardware-based fingerprinting, but we focus on fingerprinting on-board acoustic components like speakers and microphones. In this setting, Clarkson's work [26] is perhaps the most closely related to ours. He showed that it is possible to distinguish loudspeakers by analyzing recorded audio samples emitting from them. However, his experiments used special audio clips that contained 65 different frequencies, whereas we are using common audio excerpts like ringtones. Moreover, his experiments ignored the subtlety introduced by microphones. In fact in one experiment, though statistically not meaningful as it tested only two similar microphones, he found no variation across microphones. We, on the other hand found that microphones can vary across different units. Finally, his study did not thoroughly analyze the different acoustic features that can be used to successfully carry out device fingerprinting. As a result, he was able to achieve only 81% accuracy in distinguishing heterogeneous loudspeakers.

Audio fingerprinting has a rich history of notable research [23]. There are studies that have looked at classifying audio excerpts based on their content [41, 65]. Others have looked at distinguishing human speakers from audio segments [20, 22]. There has also been work on exploring various acoustic features for audio classification [52]. One of the more popular applications of audio fingerprinting has been music genre and artist recognition [43, 48].

Our work takes advantage of the large set of acoustic features that have been explored by existing work in audio fingerprinting. However, instead of classifying the content of audio segments, we utilize acoustics features to capture the manufacturing imperfections of microphones and speakers embedded in smart devices.

4. SOURCE OF FINGERPRINTS

In this section we will take a closer look at the microphones and speakers embedded on today's smartphones. This will provide an understanding of how microphones and speakers can act as a potential source for unique fingerprints.

4.1 Closer Look at Microphones

Microphones in modern smartphones are based on Micro Electro Mechanical Systems (MEMS) [10, 12, 17]. To enhance active noise and echo canceling capabilities, most smartphones today have more than one MEMS microphone. For example, the iPhone 5 has a total of three embedded MEMS microphones [10]. According to the IHS-iSuppli report, Apple and Samsung were the top consumers of MEMS microphones in 2012, accounting for a combined 54% of all shipped MEMS microphones [17]. .

A MEMS microphone, sometimes called a microphone chip or silicon microphone, consists of a coil-less pressure-sensitive diaphragm directly etched into a silicon chip. It is comprised of a MEMS die and a complementary metal-oxide-semiconductor (CMOS) die combined in an acoustic housing [7, 11]. The CMOS often includes both a preamplifier as well as an analog-to-digital (AD) converter. Modern fabrication techniques enable highly compact deigns, making them well suited for integration in digital mobile devices. The internal architecture of a MEMS microphone is shown on Figure 1. From the figure we can see that the MEMS microphone's physical design is based on a *variable capacitor* consisting of a highly flexible diaphragm in close proximity to a perforated, rigid back-plate. The perforations permit the air between the diaphragm and back-plate to escape. When an acoustic signal reaches the diaphragm through the acoustic holes, the diaphragm is set in motion. This mechanical deformation causes capacitive change which in turn causes voltage change. In this way sound pressure is converted into an electrical signal for further processing. The back-chamber acts as a acoustic resonator and the ventilation hole allows the air compressed inside the back chamber to flow out, allowing the diaphragm to move back into its original place.

The sensitivity of the microphone depends on how well the diaphragm deflects to acoustic pressure; it also depends on the gap between the static back-plate and the flexible diaphragm. Unfortunately, even though the manufacturing process of these microphones has been streamlined, no two chips roll off the assembly line functioning in exactly the same way. Imperfections can arise for the following reasons: slight variations in the chemical composition of components from one batch to the next, wear in the manufacturing machines or changes in temperature and humidity. While subtle imperfections in the microphone chips may go unnoticed by human ears, computationally such discrepancies may be sufficient to discriminate them, as we later show.

4.2 Closer Look at Microspeakers

Micro-speakers are a scaled down version of a basic acoustic speaker. So lets first look at how speakers work before we discuss how microspeakers can be used to generate unique fingerprints. Figure 2(a) shows the basic components of a speaker. The diaphragm is usually made of paper, plastic or metal and its edges are connected to the suspension. The suspension is a rim of flexible material that allows the diaphragm to move. The narrow end of the diaphragm's cone is connected to the voice coil. The voice coil is attached to the basket by a spider (damper), which holds the coil in position, but allows it to move freely back and forth. A permanent magnet is positioned directly below the voice coil.

Sound waves are produced whenever electrical current flows through the voice coil, which acts as an electromagnet. Running varying electrical current through the voice coil induces a varying magnetic field around the coil, altering the magnetization of the metal it is wrapped around. When the electromagnet's polar orientation switches, so does the direction of repulsion and attraction. In this way, the magnetic force between the voice coil and the permanent magnet causes the voice coil to vibrate, which in turn vibrates the speaker diaphragm to generate sound waves.

Figure 2(b) shows a typical MEMS microspeaker chip and Figure 2(c) shows the components inside the microspeaker [24]. The components are similar to that of a basic speaker; the only difference is the size and fabrication process [25, 44, 63]. The amplitude and frequency of the sound wave produced by the speaker's diaphragm is dictated respectively by the distance and rate at which the voice coil moves. Each speaker component can introduce variations into the generated sound. For example, variations in the electromagnetic properties of the driver can cause differences in the rate and smoothness at which the diaphragm moves. Therefore, due to the inevitable variations and imperfections of the manufacturing process, no two speakers are going to be alike, resulting in subtle differences in the produced sound. In our work, we develop techniques to computationally localize and evaluate these differences.

5. FEATURES AND ALGORITHMS USED

In this section we briefly describe the acoustic features that we used in generating device fingerprints. We also discuss the classification algorithms used in identifying the devices from which the fingerprints originated.

5.1 Audio Features

Given our knowledge that imperfections exist in device audio hardware, we now need some way to detect them. To do this, our approach identifies *acoustic features* from an audio stream, and

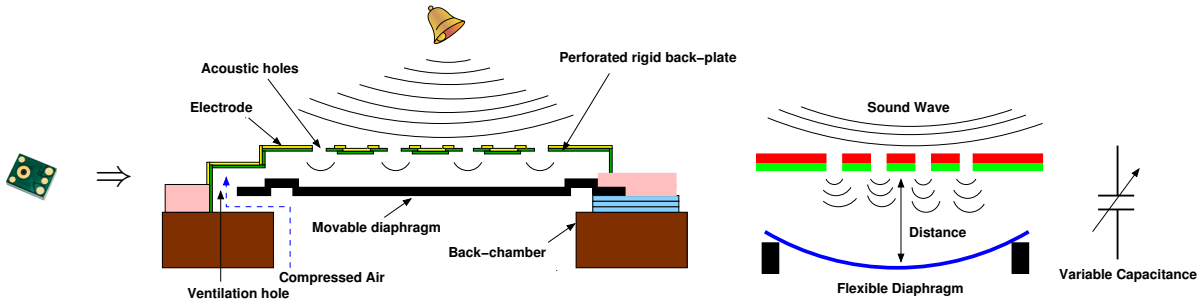
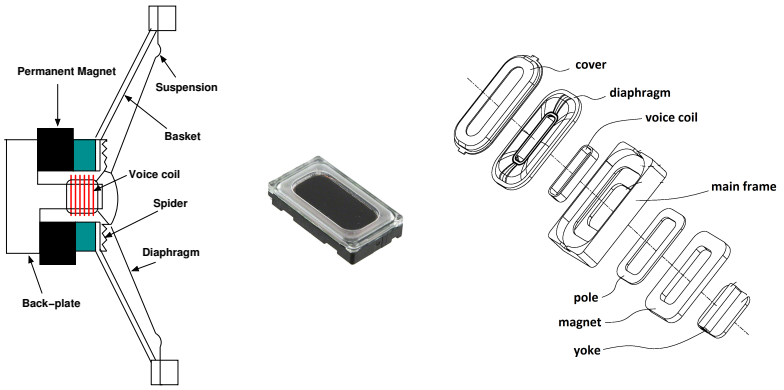


Figure 1: The internal architecture of MEMS microphone chip used in smartphones.



(a) (b) (c)

Figure 2: (a) The basic components of a speaker, (b) A typical MEMS microspeaker, (c) The internal architecture of a microspeaker chip.

uses the features to construct a *fingerprint* of the device. Computing acoustic features from an audio stream has been a subject of much research [19, 23, 52, 65]. To gain an understanding of how a broad range of acoustic features are affected by device imperfections we investigate a total of 15 acoustics features (listed in Table 1), all of which have been well-documented by researchers. Due to space limitation we exclude detailed description of each acoustic feature, however, an elaborate description of the audio features is available in our technical report [28].

5.2 Classification Algorithms

Next, we need some way to leverage the set of features to perform device identification. To achieve this, we leverage a *classification algorithm*, which takes observations (features) from the observed device as input, and attempts to classify the device into one of several previously-observed sets.

To do this, our approach works as follows. First, we perform a training step, by collecting a number of observations from a set of devices. Each observation (data point) corresponds to a set of features observed from that device, represented as a tuple with one dimension per feature. As such, data points can be thought of as existing in a hyper-dimensional space, with each axis corresponding to the observed value of a corresponding feature. Our approach then applies a classification algorithm to build a representation of these data points, which can later be used to associate new observations with device types. When a new observation is collected, the classification algorithm returns the most likely device that caused the observation.

To do this effectively, we need an efficient classification algorithm. In our work, we compare the performance of two alternate approaches described below: *k-nearest neighbors* (associates an incoming data point with the device corresponding to the nearest "learned" data points), and *Gaussian mixture models* (computes a probability distribution for each device, and determines the maximal likely association).

k-NN: The k-nearest neighbors algorithm (k-NN) is a nonparametric lazy learning algorithm. The term "non-parametric" means that the k-NN algorithm does not make any assumptions about the underlying data distribution, which is useful in analyzing real-world data with complex underlying distribution. The term "lazy learning" means that the k-NN algorithm does not use the training data to make any generalization, rather all the training data are used in the testing phase making it computationally expensive (however, optimizations are possible). The k-NN algorithm works by first computing the distance from the input data point to all training data points and then classifies the input data point by taking a majority vote of the k closest training records in the feature space [32]. The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct. We will discuss more about the choice of k in Section 6.

GMM: A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. The unknown patterns and mixture weights are estimated from training samples using an *expectation–maximization* (EM) algorithm [29]. During the matching phase the fingerprint for an unknown recording is first compared with a database of pre-computed GMMs and then the class label of the GMM that gives the highest likelihood is returned as the expected class for the unknown fingerprint. GMMs are often used in biometric systems, most notably in human speaker recognition systems, due to their capability of representing a large class of sample distributions [59, 65].

Table 1: Explored acoustic features

#	Feature	Dimension	Description
1	RMS	1	Square root of the arithmetic mean of the squares of the signal strength at various frequencies
2	ZCR	1	The rate at which the signal changes sign from positive to negative or back
3	Low-Energy-Rate	1	The percentage of frames with RMS power less than the average RMS power for the whole audio signal
4	Spectral Centroid	1	Represents the center of mass of a spectral power distribution
5	Spectral Entropy	1	Captures the peaks of a spectrum and their locations
6	Spectral Irregularity	1	Measures the degree of variation of the successive peaks of a spectrum
7	Spectral Spread	1	Defines the dispersion of the spectrum around its centroid
8	Spectral Skewness	1	Represents the coefficient of skewness of a spectrum
9	Spectral Kurtosis	1	Measure of the flatness or spikiness of a distribution relative to a normal distribution
10	Spectral Rolloff	1	Defines the frequency below which 85% of the distribution magnitude is concentrated
11	Spectral Brightness	1	Computes the amount of spectral energy corresponding to frequencies higher than a given cut-off threshold
12	Spectral Flatness	1	Measures how energy is spread across the spectrum
13	MFCCs	13	Compactly represents spectrum amplitudes residing inside the mel-frequency range
14	Chromagram	12	Representation of the distribution of energy along the 12 distinct semitones or pitch classes
15	Tonal Centroid	6	Maps a chromagram onto a six-dimensional Hypertorus structure

6. EVALUATION

In this section we perform a series of experiments to evaluate how well we can fingerprint smartphones by exploiting the manufacturing idiosyncrasies of microphones and speakers embedded in them. We start by describing how we performed our experiments (Section 6.1). Next, we briefly discuss the setup for fingerprinting devices through speaker, microphone and a combination of both (Section 6.2). We then discuss a framework for determining the *dominant* (most-relevant) set of audio features that can be used in fingerprinting smartphones (Section 6.3). Then, we look at fingerprinting devices, first, of different maker-and-model (Section 6.4), followed by devices of same maker-and-model (Section 6.5) and finally, a combination of both with multiple units of different models (Section 6.6). The performance of our approach is affected by certain aspects of the operating environment, and we the study sensitivity to such factors in Section 6.7.

6.1 Methodology

To perform our experiments, we constructed a small testbed environment with real smartphone device hardware. In particular, our default environment consisted of a 266 square foot (14'x19') office room, with nine-foot dropped ceilings with polystyrene tile, comprising a graduate student office in a University-owned building. The room was filled with desks and chairs, and opens out on a public hall with foot traffic. The room also receives a minimal amount of ambient noise from air conditioning, desktop computers, and florescent lighting. We placed smartphones in various locations in the room. To emulate an attacker, we placed an ACER Aspire 5745 laptop in the room. To investigate performance with inexpensive hardware, we used the laptop's built-in microphone to collect audio samples. We investigate how varying this setup affects performance of the attack in Section 6.7.

Devices and tools: We tested our device fingerprinting on devices from five different manufacturers. Table 2 highlights the model and quantities of the different phones used in our experiments.

Table 2: Types of phones used

Maker	Model	Quantity
Apple	iPhone 5	1
Google	Nexus One	14
	Nexus S	8
Samsung	Galaxy S3	3
	Galaxy S4	10
Motorola	Droid A855	15
Sony Ericsson	W518	1
Total		52

We also investigate different genres of audio excerpts. Table 3 describes the different types of audio excerpts used in our experiments. Duration of the audio clips varies from 3 to 10 seconds. The default sampling frequency of all audio excerpts is 44.1kHz unless explicitly stated otherwise. All audio clips are stored in WAV format using 16-bit pulse-code-modulation (PCM) technique.

Table 3: Types of audio excerpts

Type	Description	Variations
Instrumental	Musical instruments playing together, e.g., ringtone	4
Human speech	Small segments of human speech	4
Song	Combination of human voice & instrumental sound	3

For analysis we leverage the following audio tools and analytic modules: *MIRtollbox* [13], *Netlab* [15], *Audacity* [3] and the Android app *Hertz* [6]. Both MIRtoolbox and Netlab are MATLAB modules providing a rich set of functions for analyzing and extracting audio features. Audacity and Hertz are mainly used for recording audio clips on computers and smartphones respectively.

For analyzing and matching fingerprints we use a desktop machine with the following configuration: Intel i7-2600 3.4GHz processor with 12GiB RAM. We found that the average time required to match a new fingerprint was around 5–10 ms for k-NN classifier and around 0.5–1 ms for GMM classifier.

Evaluation metrics: We use standard multi-class classification metrics—*precision*, *recall*, and *F1-score* [64]—in our evaluation. Assuming there are fingerprints from n classes, we first compute the true positive (TP) rate for each class, i.e., the number of traces from the class that are classified correctly. Similarly, we compute the false positive (FP) and false negative (FN), as the number of wrongly accepted and wrongly rejected traces, respectively, for each class i ($1 \leq i \leq n$). We then compute precision, recall, and the F1-score for each class using the following equations:

$$\text{Precision, } Pr_i = TP_i/(TP_i + FP_i) \quad (1)$$

$$\text{Recall, } Re_i = TP_i/(TP_i + FN_i) \quad (2)$$

$$\text{F1-Score, } F1_i = (2 \times Pr_i \times Re_i)/(Pr_i + Re_i) \quad (3)$$

The F1-score is the harmonic mean of precision and recall; it provides a good measure of overall classification performance, since precision and recall represent a trade-off: a more conservative classifier that rejects more instances will have higher precision but lower recall, and vice-versa. To obtain the overall performance of

the system we compute average values in the following way:

$$\text{Avg. Precision, } AvgPr = \frac{\sum_{i=1}^{n} Pr_i}{n} \quad (4)$$

$$\text{Avg. Recall, } AvgRe = \frac{\sum_{i=1}^{n} Re_i}{n} \quad (5)$$

$$\text{Avg. F1-Score, } AvgF1 = \frac{2 \times AvgPr \times AvgRe}{AvgPr + AvgRe} \quad (6)$$

Each audio excerpt is recorded/played 10 times, 50% of which is used for training and the remaining 50% is used for testing. We report the maximum evaluation obtained by varying the number of neighbors (k) from 1 to 5 for the k-NN classifier and considering 1 to 5 Gaussian distributions per class. Since GMM parameters are produced by the randomized EM algorithm, we perform 10 parameter-generation runs for each instance and report the average classification performance. We also compute the 95% confidence interval, but we found it to be less than 0.01 and therefore, do not report it in the rest of the paper.

6.2 Fingerprinting Acoustic Components

6.2.1 *Process of Fingerprinting Speaker*

An attacker can leverage our technique to passively observe audio signals (e.g., ringtones) emitting from device speakers in public environments. To investigate this, we first look at fingerprinting speakers integrated inside smartphones. For fingerprinting speakers we record audio clips played from smartphones onto a laptop and we then extract acoustic features from the recorded audio excerpts to generate fingerprints as shown in Figure 3. We look at devices manufactured by both the same vendor and different vendors.

Figure 3: Steps of fingerprinting speakers.

6.2.2 *Process of Fingerprinting Microphone*

Attackers may also attempt to fingerprint devices by observing imperfections in device microphones, for example by convincing the user to install an application on their phone, which can observe inputs from the device's microphone. To investigate the feasibility of this attack, we next look at fingerprinting microphones embedded in smartphones. To do this, we record audio clips played from a laptop onto smartphones as shown in Figure 4. Again we consider devices made by both the same vendor and different vendors.

Figure 4: Steps of fingerprinting microphones.

6.2.3 *Process of Fingerprinting both Speaker and Mic*

An attacker may attempt to fingerprint devices by observing imperfections in both device microphone and speaker, for example by convincing the user to install a game on their phone which requires access to device speaker and microphone to interact with the game (something like *Talking Tom Cat*). The attacker could potentially play a theme song at the start of the game and at the same time make a recording of the audio clip. To investigate the feasibility of this attack, we build an android app that plays and records audio clips simultaneously and uploads the data to a remote server. The recorded audio clips would then enable the attacker to characterize the imperfections of microphones and speakers embedded inside smartphones. Figure 5 summarizes the whole process.

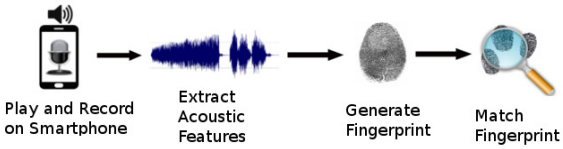


Figure 5: Steps of fingerprinting both microphones and speakers.

6.3 Feature Exploration

At first glance, it seems that we should use all features at our disposal to identify device types. However, including too many features can *worsen* performance in practice, due to their varying accuracies and potentially-conflicting signatures. Hence, in this section, we provide a framework to explore all the 15 audio features described in Section 5.1 and identify the *dominating subset* of all the features, i.e., which combination of features should be used. For this purpose we adopt a well known machine learning strategy known as *feature selection* [42, 66]. Feature selection is the process of reducing dimensionality of data by selecting only a subset of the relevant features for use in model construction. The main assumption in using feature selection technique is that the data may contain redundant features. Redundant features are those which provide no additional benefit than the currently selected features. Feature selection techniques are a subset of the more general field of feature extraction, however, in practice they are quite different from each other. Feature extraction creates new features as functions of the original features, whereas feature selection returns a subset of the features. Feature selection is preferable to feature extraction when the original units and meaning of features are important and the modeling goal is to identify an influential subset. When the features themselves have different dimensionality, and numerical transformations are inappropriate, feature selection becomes the primary means of dimension reduction.

Feature selection involves the maximization of an objective function as it searches through the possible candidate subsets. Since exhaustive evaluation of all possible subsets are often infeasible (2^N for a total of N features) different heuristics are employed. We use a greedy search strategy known as *sequential forward selection* (SFS) where we start off with an empty set and sequentially add the features that maximize our objective function. The pseudo code of our feature selection algorithm is described in Algorithm 1.

The algorithm works as follows. First, we compute the F1-score that can be achieved by each feature individually. Next, we sort the feature set based on the achieved F1-score in descending order. Then, we iteratively add features starting from the most dominant one and compute the F1-score of the combined feature subset. If adding a feature increases the F1-score seen so far we move on to the next feature, else we remove the feature under inspection. Having traversed through the entire set of features, we return the subset of features that maximizes our device classification task. Note that this is a greedy approach, therefore, the generated subset might not

Algorithm 1 Sequential Feature Selection

Input: Input feature set F
Output: Dominant feature subset D
$F1\_score \leftarrow []$
for $f \in F$ **do**
  $F1\_score[f] \leftarrow Classify(f)$
end for
$F' \leftarrow sort(F, F1\_score)$ #In descending order
$max\_score \leftarrow 0$
$D \leftarrow \varnothing$
for $f \in F'$ **do**
  $D \leftarrow D \cup f$
  $temp \leftarrow Classify(D)$
  if $temp > max\_score$ **then**
    $max\_score \leftarrow temp$
  else
    $D \leftarrow D - \{f\}$
  end if
end for
return D

always provide optimal F1-score. However, for our purpose, we found this approach to perform well, as we demonstrate in latter sections. We test our feature selection algorithm for all three types of audio excerpts listed in Table 3. We evaluate the F1-score using both k-NN and GMM classifiers.

Note that the audio excerpts used for feature exploration and the ones used for evaluating our fingerprinting approach in the following sections are not identical. We use different audio excerpts belonging to the three categories listed in Table 3, so as to *not bias* our evaluations.

6.4 Different Maker-and-Model Devices

In this section we look at fingerprinting smartphones manufactured by five different vendors. We take one representative smartphone from each row of Table 2 giving us a total of 7 different smartphones. We look at fingerprinting these devices first by using the microphone and speaker individually and next, by combining both microphone and speaker.

6.4.1 Feature Exploration

First, we look at exploring different acoustic features with the goal of obtaining the dominant subset of features. Table 4 highlights the maximum F1-score achieved by each acoustic feature for the three different types of audio excerpt. The maximum F1-score is obtained by varying k from 1 to 5 (for k-NN classifier) and also considering 1 to 5 gaussian distributions per class (for GMM classifier). Each type of audio is recorded 10 times giving us a total of 70 samples from the 7 representative handsets; 50% of which (i.e., 5 samples per handset) is used for training and the remaining 50% is used for testing. All the training samples are labeled with their corresponding handset identifier. Both classifiers return the class label for each audio clip in the test set and from that we compute F1-score. The table also highlights the subset of features selected by our sequential feature selection algorithm and their corresponding F1-score. We find that most of the time *MFCCs* are the dominant features for all categories of audio excerpt.

To get a better understanding of why *MFCCs* are the dominant acoustic features we plot the MFCCs of a given audio excerpt from three different handsets on Figure 6. All the coefficients are ranked in the same order for the three handsets. We can see that the magnitude of the coefficients vary across the handsets. For example, coefficient 3 and 5 vary significantly across the three handsets. Hence, MFCCs are highly suitable features for fingerprinting smartphones.

6.4.2 Fingerprinting using Speaker

We test our fingerprinting approach using three different types of audio excerpt. Each audio sample is recorded 10 times, 50% of which is used for training and the remaining 50% is used for testing. We repeat this procedure for the three different types of audio excerpt. Table 5 summarizes our findings (values are reported as percentages). We simply use the acoustic features obtained from our sequential feature selection algorithm as listed in Table 4. From Table 5 we see that we can successfully (with a maximum F1-score of 100%) identify which audio clip came from which smartphone. Thus, fingerprinting smartphones manufactured by different vendors seems very much feasible using only few acoustic features.

Table 5: Fingerprinting different smartphones using speaker output

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[1,7]	97.6	97.1	97.4	[13]	100	100	100
Human speech	[13]	95.2	94.3	94.8	[13]	100	100	100
Song	[15]	97.6	97.1	97.4	[13]	100	100	100

∗ Features taken from Table 4

6.4.3 Fingerprinting using Microphone

Similar to speakers, we find microphone properties differ quite substantially across vendors. We exploit this phenomenon to fingerprint smartphones through microphones. To test our hypothesis we test our fingerprinting approach using three different types of audio excerpt. Each audio sample is again recorded 10 times; we use 50% for training and the other 50% for testing. Table 6 summarizes our findings (values are reported as percentages). We use the same set of features obtained from our sequential feature selection algorithm as listed in Table 4. From Table 6 we see that we can achieve an F1-score of over 97%. These results suggest that smartphones can be successfully fingerprinted through microphones too.

Table 6: Fingerprinting different smartphones using mic

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[13,1]	95.2	94.3	94.8	[13,1,7]	100	100	100
Human speech	[15,9,1]	95.2	94.3	94.8	[13,15,11]	97.6	97.1	97.4
Song	[13,1,12]	97.6	97.1	97.4	[13,1,9]	100	100	100

∗ Features taken from Table 4

6.4.4 Fingerprinting using Microphone and Speaker

We now look at fingerprinting smartphones through both microphones and speakers. For this experiment we build an android app to collect data from different smartphones. Our app plays and records different audio clips simultaneously and uploads the data to a remote server. As we are using an android app for our data collection, we had to exclude iPhone5 and Sony Ericsson W518 handset from this experiment (reducing our pool of handsets to 5 devices). Again each audio sample is recorded 10 times, half of which is used for training and the other half for testing. We use the features obtained from Table 4. Table 7 summarizes our findings (values are reported as percentages). We see that we can achieve an F1-score of 100%. Thus, a malicious app having access to only speaker and microphone can successfully fingerprint smartphones.

Table 7: Fingerprinting different smartphones using mic & speaker

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[10]	96.7	96	96.3	[13]	100	100	100
Human speech	[12]	96.7	96	96.3	[13]	100	100	100
Song	[10]	96.7	96	96.3	[13]	100	100	100

∗ Features taken from Table 4

Table 4: Feature exploration using sequential forward selection technique for different maker-and-model smartphones

#	Feature	Fingerprinting Speakers Maximum F1-Score (%)						Fingerprinting Microphones Maximum F1-Score (%)						Fingerprinting Speakers and Microphones Maximum F1-Score (%)					
		Instrumental		Human Speech		Song		Instrumental		Human Speech		Song		Instrumental		Human Speech		Song	
		k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM
1	RMS	**97.4**	98	80.8	78.3	88.8	86.2	87.2	84.4	62.7	70.3	80.2	82.8	93.1	92.7	89	80.8	93.1	96.3
2	ZCR	57.6	52.1	63.7	48.6	77	77	76.4	74.3	75.7	76.8	72.4	71.3	83.3	83.3	84.6	78.8	93.1	96.3
3	Low-Energy-Rate	69.2	51.8	52.8	38.2	59.6	58.8	47	42.2	26	19.5	38.8	36.6	93.1	90	81.7	77.7	96.3	96.3
4	Spectral Centroid	91.5	88.4	59.4	55.8	88.1	87.7	73.7	69.7	70	77.9	73.7	76	82.3	85.1	40.5	42.7	74.2	76.1
5	Spectral Entropy	84	80.5	59.4	46.5	91.5	91	52.1	56.5	68	59.8	68.8	58.8	78.3	72.4	81	67.9	96.3	96.3
6	Spectral Irregularity	31.3	51.4	37.5	46.8	43.2	50.3	49.3	55	52.6	49.7	53	48.7	74.6	72.7	55.4	53.6	90.1	81.1
7	Spectral Spread	90.2	89.2	56.7	56.7	91	85.7	90.4	86.2	50.5	54.8	81.6	76.5	86.5	86.5	92.7	92.4	93.1	93.1
8	Spectral Skewness	68.3	82.1	69	62	82.9	79.9	71.4	63.4	65.6	63.6	61.8	49.7	92.3	89.7	84.6	86.2	96.3	96.3
9	Spectral Kurtosis	76.7	79.5	68.1	60.2	88.6	86.9	63.9	61.2	65.2	65.1	85.4	58	89.5	86.3	60.7	59.5	81.6	85.3
10	Spectral Rolloff	86.6	86.4	85.8	66.7	74.8	76.1	37.8	42.7	82.9	83.8	67.7	73	**100**	96.3	92.7	92.7	**100**	96
11	Spectral Brightness	87.5	85.2	70.9	87.7	85.5	77.1	66.4	64.8	60.3	60.2	63.7	67.4	80.4	79.4	81.6	67.6	87.4	90.2
12	Spectral Flatness	84.5	84	61.6	61.3	95.1	97.4	92.5	92.5	66.7	68.9	74.8	74.4	85.1	85.1	**96.3**	92.7	**100**	96.3
13	MFCCs	**97.4**	**100**	**100**	**100**	94.8	**100**	**94.8**	**94.8**	79.9	**92.1**	**90.4**	**95.4**	93.1	**100**	**96.3**	**96.3**	92.7	**100**
14	Chromagram	84.3	79.4	81.1	**100**	97.4	**100**	88.4	77.7	88.7	86	78.4	87.7	96.3	93.1	88.6	**96.3**	86.5	**100**
15	Tonal Centroid	86.4	88.4	80.3	98.2	**100**	**100**	91.9	92.7	**92.1**	86.4	89.6	92.5	96.3	**100**	**96.3**	**96.3**	**100**	**100**
Sequential Feature Selection		[1,7]	[13]	[13]	[13]	[15]	[13]	[13,1]	[13,1,7]	[15,9,1]	[13,15,11]	[13,1,12]	[13,1,9]	[10]	[13]	[12]	[13]	[10]	[13]
Max F1-Score		100	100	100	100	100	100	97.4	100	92.1	93	92.5	97.4	100	100	96.3	96.3	100	100

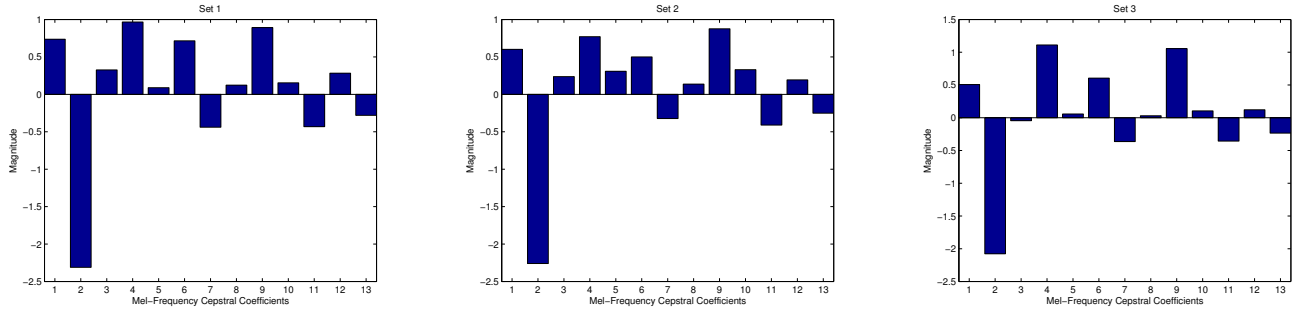


Figure 6: MFCCs of the same audio sample taken from three different handsets manufactured by the same vendor. We can see that some of the coefficients vary significantly, thus enabling us to exploit this feature to fingerprint smartphones.

6.5 Same Maker-and-Model Devices

In this section, we look at fingerprinting smartphones manufactured by the same vendor and are of the same model. From Table 2 we see that we have 15 Motorola Droid A855 handsets which is the largest number among all the other different types of smartphones in our collection. We therefore use these 15 devices for all the experiments in this section. We found that fingerprinting smartphones of the same maker-and-model was relatively a tougher problem. We again look at fingerprinting these devices, first, through the microphone and speaker individually and then by combining both the microphone and speaker.

6.5.1 Feature Exploration

First, we determine the dominating subset of acoustic features that can be used for fingerprinting smartphones of the same model. To obtain the fingerprinting data we record audio clips played from 15 Motorola Droid A855 handsets. Each type of audio is recorded 10 times giving us a total of 150 samples from the 15 handsets; 50% of which is used for training and the remaining 50% is used for testing. Table 8 shows the maximum F1-score achieved by each acoustic feature for the three different types of audio excerpt. The table also highlights the dominating subset of features selected by our sequential feature selection algorithm. We again find that *MFCCs* are the dominant features for all categories of audio excerpt.

6.5.2 Fingerprinting using Speaker

We now look at fingerprinting the 15 Motorola Droid A855 handsets. Table 9 highlights our findings. We test our fingerprinting approach against three different forms of audio excerpt. We use the acoustic features obtained from our sequential feature selection algorithm as listed in Table 8. From Table 9, we see that we can achieve an F1-score of over 94% in identifying which audio clip originated from which handset. Thus fingerprinting smartphones through speaker seems to be a viable option.

Table 9: Fingerprinting similar smartphones using speaker output

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[13,14]	96.7	96	96.3	[13,14]	98.4	98.1	98.3
Human speech	[13]	98.9	98.7	98.8	[13,14]	98.9	98.7	98.8
Song	[13,7]	93.2	92	92.6	[13,14]	95.6	93.3	94.5

∗ Feature numbers taken from Table 8

6.5.3 Fingerprinting using Microphone

We now investigate fingerprinting similar smartphones manufactured by the same vendor through microphone-sourced input. We again use 15 Motorola Droid A855 handsets for these experiments. We use the features obtained through Algorithm 1 which are listed in Table 8. Table 10 shows our findings. We see similar results compared to fingerprinting speakers. We were able to achieve an F1-score of 95% in identifying the handset from which the audio excerpt originated. Thus fingerprinting smartphones through microphones also appears to be a feasible option.

Table 10: Fingerprinting similar smartphones using microphone

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[13,8,12]	95.9	94.7	95.3	[13,8,12]	96	94.7	95.3
Human speech	[13]	98.9	98.7	98.8	[13,14]	100	100	100
Song	[13,14,10]	96.4	96	96.2	[13,14]	96.5	95.7	96.1

∗ Feature numbers taken from Table 8

6.5.4 Fingerprinting using Microphone and Speaker

We now look at the effect of combining microphone and speaker in fingerprinting similar smartphones. We use our android app to

Table 8: Feature exploration using sequential forward selection technique for same model smartphones

#	Feature	Fingerprinting Speakers Maximum F1-Score (%)						Fingerprinting Microphones Maximum F1-Score (%)						Fingerprinting Speakers and Microphones Maximum F1-Score (%)					
		Instrumental		Human Speech		Song		Instrumental		Human Speech		Song		Instrumental		Human Speech		Song	
		k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM	k-NN	GMM
1	RMS	34.9	33.8	16.6	12.3	20	25.7	40.2	36.9	19.6	20.1	23.5	28.6	87.2	83	92.1	93	89.5	89.2
2	ZCR	29.7	26.5	12.2	14.4	13	7.1	22.7	29.6	26.2	22.6	44.5	41.9	59.8	60.5	56.8	58.4	67.6	75.2
3	Low-Energy-Rate	12.5	14.8	15	5.7	21.8	18.7	22.6	24.8	5.2	7.4	10.7	13.4	67.4	70.9	30.1	36.7	69.7	64.5
4	Spectral Centroid	28	30.5	12.2	19	39.9	40.3	17.3	24.8	16.6	12.9	33.7	35.7	30.1	27.5	25.7	30.1	35.1	32.7
5	Spectral Entropy	20.9	19.8	14.2	16.6	33.9	26.3	29.1	22.2	15.2	15.1	40.3	36	78.5	70.3	52.9	54.8	81.8	81.8
6	Spectral Irregularity	14.5	11.7	7.4	14.7	11.8	17.5	12.6	16.3	13.2	17.9	15.8	18.6	63.9	54.6	32.5	33.6	62.5	67.1
7	Spectral Spread	36.4	43.7	11.3	14.3	35.2	38.4	17.2	22.6	16.4	14.9	36.2	34.8	84.6	81.3	67.6	62.8	87	87.4
8	Spectral Skewness	33.9	29.1	13.3	15.5	31.5	40.3	31.8	28.1	20.8	13.7	38	43.1	85.7	88.3	58.7	54.4	70.9	68.9
9	Spectral Kurtosis	30.5	29.1	11.6	16	31.1	36.8	28.5	26.1	20.3	14	45.8	39.2	80.3	80.6	51.9	49.9	82.2	76.2
10	Spectral Rolloff	40.4	39	14.9	14.3	38.7	41.1	30	32.8	15.1	11.6	46.1	44	79.4	73.4	46.9	51.5	77.2	71.8
11	Spectral Brightness	32.1	31.6	18.9	21.8	18.5	17.9	22.5	20.3	12.6	16	33.1	27.4	86	88	75.2	69.2	87.5	79.5
12	Spectral Flatness	34.9	31	19.8	13.3	32.4	30	24.6	23.8	17.2	12.2	39.2	35.5	79.8	79	45.5	45.4	86.4	87.2
13	MFCCs	**90.4**	**96.5**	**91.3**	**97.5**	**90**	**91.4**	**89**	**93.5**	**98.8**	**96.2**	**94.1**	**97.5**	**100**	**100**	**98.7**	**100**	**100**	**100**
14	Chromagram	79.1	70.6	72.9	66	80.6	80	71.5	55.3	75	88.7	87.3	85.3	98.8	95.8	97.6	**100**	**100**	96.5
15	Tonal Centroid	77	60	65.4	53.4	63.6	53.8	67.8	51.3	70	70.8	83.1	79.4	98.8	94.8	95.2	92.7	**100**	98.8
Sequential Feature Selection		[13,14]	[13,14]	[13]	[13,14]	[13,7]	[13,14]	[13,8,12]	[13,8,12]	[13]	[13,14,2]	[13,14,10]	[13,14]	[13]	[13]	[13]	[13]	[13]	[13]
Max F1-Score		97.5	97.7	93.7	98.2	91.5	92.9	93	96.7	98.8	97.5	96.3	97.9	100	100	98.7	100	100	100

collect 10 samples per audio clip, half of which is used for training and the remaining half for testing. Table 11 highlights our findings. We see that we were able to fingerprint all test samples accurately. Thus combining the idiosyncrasies of both the speaker and microphone seems to be the best option to distinguish smartphones of same maker and model. So, if a malicious app can get access to the speaker (which does not require explicit permission) and microphone (which may require explicit permission, but many games nowadays require access to microphone anyway) it can successfully track individual devices.

Table 11: Fingerprinting similar smartphones using mic & speaker

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[13]	100	100	100	[13]	100	100	100
Human speech	[13]	100	100	100	[13]	100	100	100
Song	[13]	100	100	100	[13]	100	100	100

∗ Features taken from Table 8

6.6 All Combination of Devices

In this section we look at fingerprinting all the devices in our collection (i.e., 50 *android* smartphones after excluding iphone5 and Sony Ericsson W518). We combine microphone and speaker to generate the auditory fingerprint of smartphones. We do so because in the previous sections we found that combining speaker and microphone yielded the highest accuracy. First, we perform acoustic feature exploration to determine the dominant features. Table 12 highlights our findings. We see that again *MFCCs* are the dominant features for all categories of audio excerpt. This is expected as we saw similar outcomes in Table 4 and Table 8.

Next, we evaluate how effectively we can fingerprint the 50 android smartphones. The setting is similar to all the previous experiments where each audio clip is recorded 10 times, 50% of which is used for training and the remaining 50% for testing. We use our android app to collect all the audio samples. Table 13 shows our obtained fingerprinting results. We see that we can obtain an F1-score of over 98% in fingerprinting all the 50 smartphones. This result suggests that a malicious app having access to microphone and speaker can easily fingerprint smartphones.

6.7 Sensitivity Analysis

In this section we investigate how different factors such as audio sampling rate, training set size, the distance from audio source to recorder, and background noise impact our fingerprinting performance. Such investigations will help us determine the conditions under which our fingerprinting approach will be feasible, specially if the attacker is tracking devices in public locations. For the following set of experiments we only focus on fingerprinting similar-model smartphones from the same vendor (as this has been shown to be a tougher problem in the previous section) and consider only fingerprinting speakers as this is applicable to the scenario where the attacker is tracking devices in public locations. We also consider recording only ringtones (i.e., audio clips belonging to our defined 'Instrumental' category in Table 3) for the following experiments. Since we are recording ringtones we use the features highlighted in Table 8 under the 'Instrumental' category.

Table 12: Feature exploration using sequential forward selection technique for all smartphones

#	Feature	Maximum F1-Score (%)					
		Instrumental		Human Speech		Song	
		k-NN	GMM	k-NN	GMM	k-NN	GMM
1	RMS	82.7	80	87.3	84	78.7	76.8
2	ZCR	51.3	48.2	50.3	45.9	48.5	45.9
3	Low-Energy-Rate	45.2	40.6	19.4	15.4	31.9	33.8
4	Spectral Centroid	35.6	34.7	23.7	25.8	25.7	30.1
5	Spectral Entropy	56.2	60.8	46.3	48.1	67.7	67.7
6	Spectral Irregularity	46.1	47	25.9	23.6	26.9	35.3
7	Spectral Spread	57.4	57	54.2	49.7	70.9	74.1
8	Spectral Skewness	50.3	53.9	34.5	32.5	52.7	59.9
9	Spectral Kurtosis	45	47.7	37.1	38.6	51.5	54.2
10	Spectral Rolloff	49.5	53.5	48.4	45.9	59.1	62.8
11	Spectral Brightness	52.1	54.5	38.1	35.3	59.2	61.7
12	Spectral Flatness	61	60.1	61.6	63.4	67.3	68.3
13	MFCCs	**100**	**100**	**100**	**99.6**	**100**	**99.6**
14	Chromagram	96.2	93.4	98.9	95.8	99.6	98.2
15	Tonal Centroid	96	89	95.5	91.8	98.5	98.5
Sequential Feature Selection		[13]	[13]	[13]	[13]	[13]	[13]
Max F1-Score		100	100	100	99.6	100	99.6

Table 13: Fingerprinting all smartphones using mic & speaker

Audio Type	k-NN				GMM			
	Features*	$AvgPr$	$AvgRe$	$AvgF1$	Features*	$AvgPr$	$AvgRe$	$AvgF1$
Instrumental	[13]	99.3	98.8	99	[13]	98.6	98.1	98.3
Human speech	[13]	99.7	99.6	99.6	[13]	99.4	99.2	99.3
Song	[13]	99.7	99.6	99.6	[13]	100	100	100

∗ Features taken from Table 12

6.7.1 *Impact of Sampling Rate*

First, we investigate how the sampling rate of audio signals impacts our fingerprinting precision. To do this, we record a ringtone at the following three frequencies: 8kHz, 22.05kHz and 44.1kHz. Each sample is recorded 10 times with half of them being used for training and the other half for testing. Figure 7 shows the average precision and recall obtained under different sampling rates. As we can see from the figure, as sampling frequency decreases, the precision/recall also goes down. This is understandable, because the higher the sampling frequency the more fine-tuned information we have about the audio sample. However, the default sampling fre-

quency on most hand-held devices today is 44.1kHz [4], with some of the latest models adopting even higher sampling rates [1]. We, therefore, believe sampling rate will not impose an obstacle to our fingerprinting approach, and in future we will be able to capture more fine grained variations with the use of higher sampling rates.

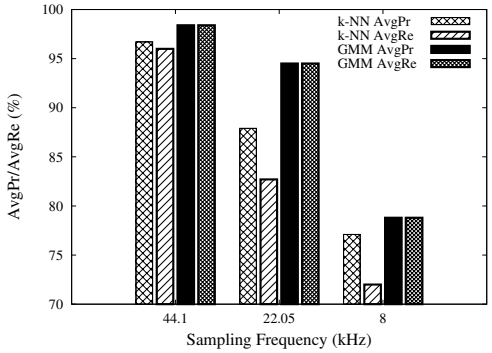


Figure 7: Impact of sampling frequency on precision/recall.

6.7.2 *Varying Training Size*

Next, we consider performance of the classifiers in the presence of limited training data. For this experiment we vary the training set size from 10% to 50% (i.e., from 1 to 5 samples per class) of all available samples. Table 14 shows the evolution of the F1-score as training set size is increased (values are reported as percentages). We see that as the training set size increases the F1-score also rises which is expected. However, we see that with only three samples per class we can achieve an F1-score of over 90%. This suggests that we do not need too many training samples to construct a good predictive model.

Table 14: Impact of varying training size

Training samples per class	k-NN Features [13,14]*			GMM Features [13,14]*		
	$AvgPr$	$AvgRe$	$AvgF1$	$AvgPr$	$AvgRe$	$AvgF1$
1	42	49.3	45.3	50	53.3	51.6
2	79.2	80	79.6	80.4	80	80.2
3	91.3	89.3	90.2	91.7	89.3	90.5
4	95.3	94.7	95	95.6	94.7	95.1
5	96.7	96	96.3	98.4	98.1	98.3

∗ Feature numbers taken from Table 8

6.7.3 *Varying Distance between Speaker and Recorder*

Next, we inspect the impact of distance between the audio source (i.e., smartphone) and recorder (i.e., laptop/PC) on fingerprinting precision. For this experiment we use a separate external microphone as the signal capturing capacity of the microphone embedded inside a laptop degrades drastically as distance increases. We use the relatively inexpensive ($44.79) Audio-Technica ATR-6550 shotgun microphone for this experiment and vary the distance between the external microphone and smartphone from 0.1 meter to 5 meters. Table 15 summarizes the F1-scores obtained as the distance between the smartphone and microphone varies. We see that as distance increases, F1-score decreases. This is expected, because the longer the distance between the smartphone and microphone, the harder it becomes to capture the minuscule deviations between audio samples. However, we see that even up to two meters distance we can achieve an F1-score of 93%. This suggests that our device fingerprinting approach works only up to a certain distance using any commercial microphones. However, using specialized microphones, such as parabolic microphones (usually used in capturing animal sounds from a far distance) could help in increasing the fingerprinting precision at even longer distances.

Table 15: Impact of varying distance

Distance (in meters)	k-NN Features [13,14]*			GMM Features [13,14]*		
	$AvgPr$	$AvgRe$	$AvgF1$	$AvgPr$	$AvgRe$	$AvgF1$
0.1	96.7	96	96.3	98.4	98.1	98.3
1	92.7	91.5	92	95.2	94.7	94.9
2	88.2	87.6	87.9	94.5	92	93.2
3	76.7	76	76.3	78.9	84	81.4
4	70.2	64	67	76.8	76	76.4
5	64.5	62.7	63.6	77	73.3	75.1

∗ Feature numbers taken from Table 8

6.7.4 *Impact of Ambient Background Noise*

In this section we investigate how ambient background noise impacts the performance of our fingerprinting technique. For this experiment we consider scenarios where there is a crowd of people using their smart devices and we are trying to fingerprint those devices by capturing audio signals (in this case ringtones) from the surrounding environment. Table 16 highlights the four different scenarios that we are considering. To emulate such environment, external speakers (2 pieces) are placed between the smartphone and microphone while recording is taking place. The external speakers are constantly replaying the respective ambient noise in the background. We consider a distance of two meters from the audio source to recorder. The ambient background sounds were obtained from PacDV [2] and SoundJay [16]. We also compute the signal-to-noise (SNR) ratio between the original ringtone and the different ambient background noises. The RMS (root-mean-square) value of the different background noises varied from approximately 13% (17.77 dB) to 18% (14.92 dB) of the RMS value of the ringtone under consideration. Table 16 shows our findings (values are reported as percentages). We can see that even in the presence of various background noise we can achieve an F1-score of over 91%.

Table 16: Impact of ambient background noise

Environments	SNR (dB)	k-NN Features [13,14]*			GMM Features [13,14]*		
		$AvgPr$	$AvgRe$	$AvgF1$	$AvgPr$	$AvgRe$	$AvgF1$
Shopping Mall	15.85	88.8	85.3	87	95.1	93.3	94.2
Restaurant/Cafe	17.77	90.5	89.7	90.1	92.5	90.7	91.6
City Park	15.43	91.7	90	90.8	95.2	94.1	94.6
Airport Gate	14.92	91.3	89.5	90.4	94.5	93.3	93.9

∗ Feature numbers taken from Table 8

7. DISCUSSION AND LIMITATIONS

Our approach has a few limitations. First, we experimented with 52 devices manufactured by different vendors; it is possible that a larger target device pool would lower accuracy. That said, distinctions across different device types are more clear; additionally, audio fingerprints may be used in tandem with other techniques, like accelerometer fingerprinting [31], to better discriminate between devices. Secondly, most of the experiments took place in a lab setting. However, we studied the impact of ambient background noise and still found our approach to be applicable. Lastly, all the phones used in our experiments were not in mint condition and some of the idiosyncrasies of individual microphones and speakers may have been the result of uneven wear and tear on each device; we believe, however, that this is likely to occur in the real world as well.

8. CONCLUSION

In this paper we show that it is feasible to fingerprint smart devices through on-board acoustic components like microphones and

speakers. As microphones and speakers are one of the most standard components present in almost all smart devices available today, this creates a key privacy concern for users. To demonstrate the feasibility of this approach, we collect fingerprints from 52 different smartphones covering a total of five different brands of smartphones. Our studies show that it is possible to successfully fingerprint smartphones through microphones and speakers, not only under controlled environments, but also in the presence of ambient noise. We believe our findings are important steps towards understanding the full consequences of fingerprinting smart devices through acoustic channels.

Acknowledgment

We would like to thank Thomas S. Benjamin for his valuable input during the initial phase of the project and all the anonymous reviewers for their valuable feedback. We would specially like to thank Romit Roy Choudhury and his group at UIUC for providing us with the bulk of smartphones used in our experiments. On the same note we would like to extend our gratitude to the Computer Science department at UIUC for providing us with the Motorola Droid phones. This paper reports on work that was supported in part by NSF CNS 0953655.

9. REFERENCES

[1] 5 of the best DACs. http://www.stuff.tv/music/5-best-dacs-how-make-your-digital-music-sound-amazing/feature. Accessed 05/15/2014.

[2] Ambient Sound Effects. http://www.pacdv.com/sounds/ambience_sounds.html. Accessed 05/15/2014.

[3] Audacity is free, open source, cross-platform software for recording and editing sounds. http://audacity.sourceforge.net/. Accessed 05/15/2014.

[4] Audio 4 Smartphones – Wolfson Microelectronics. http://www.wolfsonmicro.com/documents/uploads/misc/en/Audio4Smartphones.pdf. Accessed 05/15/2014.

[5] Global mobile statistics 2013. http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/a. Accessed 05/15/2014.

[6] Hertz, the WAV recorder. https://play.google.com/store/apps/details?id=uk.ac.cam.cl.dtg.android.audionetworking.hertz. Accessed 05/15/2014.

[7] How MEMS Microphones Fucntion. http://www.eeherald.com/section/design-guide/mems-microphone.html. Accessed 05/15/2014.

[8] IPhone and Android Apps Breach Privacy. http://www.gartner.com/newsroom/id/2335616. Accessed 05/15/2014.

[9] IPhone and Android Apps Breach Privacy. http://online.wsj.com/article/SB10001424052748704694004576020083703574602.html. Accessed 05/15/2014.

[10] MEMS microphone market. http://www.digikey.com/supply-chain-hq/us/en/articles/semiconductors/mems-microphone-market-revenues-soar-42-in-2012/1497. Accessed 05/15/2014.

[11] MEMS Microphone Model. http://www.comsol.com/blogs/mems-microphone-model-presented-asa-166-san-francisco/. Accessed 05/15/2014.

[12] MEMS microphone shipments to climb 30 percentage in 2013. http://electroiq.com/blog/2013/02/mems-microphone-shipments-to-climb-30-percent-this-year/. Accessed 05/15/2014.

[13] MIRtoolbox. https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox. Accessed 05/15/2014.

[14] Mobile device market to reach 2.6 billion units by 2016. http://www.canalys.com/newsroom/mobile-device-market-reach-26-billion-units-2016. Accessed 05/15/2014.

[15] Netlab: Algorithms for Pattern Recognition. http://www1.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/book/. Accessed 05/15/2014.

[16] SOUNDJAY-Ambient Sound Effects. http://www.soundjay.com/ambient-sounds.html. Accessed 05/15/2014.

[17] Top MEMS Microphone Suppliers. http://www.isuppli.com/MEMS-and-Sensors/MarketWatch/pages/Top-MEMS-Microphone-Suppliers-All-CanCount-on-Apple-for-Clear-and-Resounding-Success.aspx. Accessed 05/15/2014.

[18] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens, and B. Preneel. FPDetective: dusting the web for fingerprinters. In *Proceedings of the 2013 ACM conference on Computer and Communications Security*, CCS '13, pages 1129–1140, 2013.

[19] M. A. Bartsch and G. H. Wakefield. Audio Thumbnailing of Popular Music Using Chroma-based Representations. *IEEE Transactions on Multimedia*, 7(1):96–104, Feb 2005.

[20] F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-Garcia, D. Petrovska-Delacretaz, and D. A. Reynolds. A Tutorial on Text-Independent Speaker Verification. *EURASIP Journal on Advances in Signal Processing*, 4:430–451, 2004.

[21] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless Device Identification with Radiometric Signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, pages 116–127, 2008.

[22] J. Campbell, J.P. Speaker recognition: a tutorial. *Proceedings of the IEEE*, 85(9):1437–1462, Sep 1997.

[23] P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A Review of Audio Fingerprinting. *J. VLSI Signal Process. Syst.*, 41(3):271–284, Nov 2005.

[24] J. Chang and Y. Peng. Speaker, yoke thereof and method for manufacturing yoke, Jan 2012. US Patent 8,094,867.http://www.google.com/patents/US8094867.

[25] M. Cheng, W. Huang, and S. R. Huang. A silicon microspeaker for hearing instruments. *J. of Micromechanics and Microengineering*, 14(7):859–866, Jul 2004.

[26] W. B. Clarkson. *Breaking Assumptions: Distinguishing Between Seemingly Identical Items Using Cheap Sensors*. PhD thesis, Princeton, NJ, USA, 2012.

[27] S. COLE and S. Cole. *Suspect Identities: A History of Fingerprinting and Criminal Identification*. Harvard University Press, 2009.

[28] A. Das, N. Borisov, and M. Caesar. Fingerprinting smart devices through embedded acoustic components. *CoRR*, abs/1403.3366, 2014. http://arxiv.org/abs/1403.3366.

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. of the Royal Statistical Society. Series B*, 39(1):1–38, 1977.

[30] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee. Identifying Unique Devices Through Wireless Fingerprinting. In *Proceedings of the 1st ACM Conference on Wireless Network Security*, WiSec '08, pages 46–55, 2008.

[31] S. Dey, N. Roy, W. Xu, R. R. Choudhury, and S. Nelakuditi. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable. In *Proceedings of the 20th Annual Network and Distributed System Security Symposium*, NDSS'14, Feb 2014.

[32] R. Duda, P. Hart, and D. Stork. *Pattern classification*. Wiley, 2001.

[33] P. Eckersley. How Unique is Your Web Browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10, pages 1–18, 2010.

[34] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting Privacy Leaks in iOS Applications. In *Proceedings of the 17th Annual Network and Distributed System Security Symposium*, NDSS '11, 2011.

[35] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI'10, pages 1–6, 2010.

[36] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner. Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the 8th Symposium on Usable Privacy and Security*, SOUPS '12, pages 3:1–3:14, 2012.

[37] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. Van Randwyk, and D. Sicker. Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting. In *Proceedings of the 15th USENIX Security Symposium*, 2006.

[38] R. M. Gerdes, T. E. Daniels, M. Mina, and S. F. Russell. Device identification via analog signal fingerprinting: A matched filter approach. In *Proceedings of the 13th Annual Network and Distributed System Security Symposium*, 2006.

[39] C. Gibler, J. Crussell, J. Erickson, and H. Chen. AndroidLeaks: Automatically Detecting Potential Privacy Leaks in Android Applications on a Large Scale. In *Proceedings of the 5th International Conference on Trust and Trustworthy Computing*, TRUST'12, pages 291–307, 2012.

[40] F. Guo and T. cker Chiueh. Sequence Number-Based MAC Address Spoof Detection. In *Proceedings of 8th International Symposium on Recent Advances in Intrusion Detection*, RAID '05, 2005.

[41] G. Guo and S. Li. Content-based audio classification and retrieval by support vector machines. *IEEE Transactions on Neural Networks*, 14(1):209–215, Jan 2003.

[42] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3(26):1157–1182, Mar 2003.

[43] J. Haitsma and T. Kalker. A Highly Robust Audio Fingerprinting System. In *Proceedings of the 2002 International Symposium on Music Information Retrieval*, pages 107–115, 2002.

[44] S.-S. Je, F. Rivas, R. Diaz, J. Kwon, J. Kim, B. Bakkaloglu, S. Kiaei, and J. Chae. A Compact and Low-Cost MEMS Loudspeaker for Digital Hearing Aids. *IEEE Transactions on Biomedical Circuits and Systems*, 3(5):348–358, 2009.

[45] P. Kelley, S. Consolvo, L. Cranor, J. Jung, N. Sadeh, and D. Wetherall. A Conundrum of Permissions: Installing Applications on an Android Smartphone. In *Financial Cryptography and Data Security*, pages 68–79. 2012.

[46] T. Kohno, A. Broido, and K. C. Claffy. Remote Physical Device Fingerprinting. *IEEE Trans. Dependable Secur. Comput.*, 2(2):93–108, Apr 2005.

[47] L. Langley. Specific emitter identification (SEI) and classical parameter fusion technology. In *WESCON '93*, pages 377–381, Sep 1993.

[48] T. Li, M. Ogihara, and Q. Li. A Comparative Study on Content-based Music Genre Classification. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 282–289, 2003.

[49] Z. Li, W. Xu, R. Miller, and W. Trappe. Securing Wireless Systems via Lower Layer Enforcements. In *Proceedings of the 5th ACM Workshop on Wireless Security*, WiSe '06, pages 33–42, 2006.

[50] G. Lyon. Nmap: a free network mapping and security scanning tool. http://nmap.org/. Accessed 05/15/2014.

[51] K. Mahaffey and J. Hering. App Attack: Surviving the Explosive Growth of Mobile Apps. 2010. https://media.blackhat.com/bh-us-10/presentations/Mahaffey_Hering/Blackhat-USA-2010-Mahaffey-Hering-Lookout-App-Genome-slides.pdf.

[52] M. Mckinney and J. Breebaart. Features for Audio and Music Classification. In *Proceedings of the 2003 International Symposium on Music Information Retrieval*, pages 151–158, 2003.

[53] S. Moon, P. Skelly, and D. Towsley. Estimation and removal of clock skew from network delay measurements. In *Proceedings of the 18th Annual IEEE International Conference on Computer Communications*, INFOCOM '99, pages 227–234, 1999.

[54] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham. Fingerprinting Information in JavaScript Implementations. In *Proceedings of W2SP 2011*, May 2011.

[55] N. T. Nguyen, G. Zheng, Z. Han, and R. Zheng. Device fingerprinting to enhance wireless security using nonparametric Bayesian method. In *Proceedings IEEE INFOCOM*, pages 1404–1412, April 2011.

[56] L. Olejnik, C. Castelluccia, and A. Janc. Why Johnny Can't Browse in Peace: On the Uniqueness of Web Browsing History Patterns. In *Proceedings of the 5th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2012.

[57] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall. 802.11 User Fingerprinting. In *Proceedings of the 13th Annual International Conference on Mobile Computing and Networking*, pages 99–110, 2007.

[58] N. Patwari and S. K. Kasera. Robust Location Distinction Using Temporal Link Signatures. In *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, MobiCom '07, pages 111–122, 2007.

[59] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1-3):19–41, 2000.

[60] M. Riezenman. Cellular security: better, but foes still lurk. *IEEE Spectrum*, 37(6):39–42, Jun 2000.

[61] A. Ross and A. Jain. Information fusion in biometrics. *Pattern Recognition Letters*, 24(13):2115 – 2125, 2003.

[62] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev, and C. Glezer. Google Android: A Comprehensive Security Assessment. *IEEE Security and Privacy*, 8(2):35–44, 2010.

[63] I. Shahosseini, E. Lefeuvre, M. Woytasik, J. Moulin, X. Leroux, S. Edmond, E. Dufour-Gergam, A. Bosseboeuf, G. Lemarquand, and V. Lemarquand. Towards high fidelity high efficiency MEMS microspeakers. In *IEEE Sensors*, pages 2426–2430, 2010.

[64] M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45(4):427–437, 2009.

[65] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[66] Y. Yang and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, 1997.

[67] Z. Yang, L. Cai, Y. Liu, and J. Pan. Environment–aware clock skew estimation and synchronization for wireless sensor networks. In *Proceedings of the 31st Annual IEEE International Conference on Computer Communications*, INFOCOM '12, pages 1017–1025, 2012.

[68] F. Yarochkin, M. Kydyraliev, and O. Arkin. Xprobe project. http://ofirarkin.wordpress.com/xprobe/.

[69] T.-F. Yen, Y. Xie, F. Yu, R. P. Yu, and M. Abadi. Host Fingerprinting and Tracking on the Web:Privacy and Security Implications. In *Proceddings of the 19th Annual Network and Distributed System Security Symposium*, NDSS'12, 2012.

[70] Y. Zhou, X. Zhang, X. Jiang, and V. Freeh. Taming Information-Stealing Smartphone Applications (on Android). In *Proceedings of the 4th International Conference on Trust and Trustworthy Computing*, pages 93–107. 2011.