Fingerprinting Smart Devices Through Embedded Acoustic Components

Anupam Das, Nikita Borisov and Matthew Caesar
University of Illinois at Urbana-Champaign

Abstract

The widespread use of smart devices gives rise to both security and privacy concerns. Fingerprinting smart devices can assist in authenticating physical devices, but it can also jeopardize privacy by allowing remote identification without user awareness. We propose a novel fingerprinting approach that uses the microphones and speakers of smart phones to uniquely identify an individual device. During fabrication, subtle imperfections arise in device microphones and speakers which induce anomalies in produced and received sounds. We exploit this observation to fingerprint smart devices through playback and recording of audio samples. We use audio-metric tools to analyze and explore different acoustic features and analyze their ability to successfully fingerprint smart devices. Our experiments show that it is even possible to fingerprint devices that have the same vendor and model; we were able to accurately distinguish over 93% of all recorded audio clips from 15 different units of the same model. Our study identifies the prominent acoustic features capable of fingerprinting devices with high success rate and examines the effect of background noise and other variables on fingerprinting accuracy.

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 [6,9,15]. Canalys predicted that the mobile device market will reach 2.6 billion units by 2016, with smartphones and tablets continuing to dominate shipments [15]. 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 and information security. There has been a large body of research work highlighting and proposing solutions for privacy and security issues of smartphones [10,36,37,42,60,73,82]. All these works center around securing the software, including the operating system and network stack, of mobile devices, for example by instilling fine-grained access control policies, or restricting dataflow, containing private data, to a network sink.

In this paper we propose a novel technique for fingerprinting the hardware of smartphones. The observation is that even if the software on mobile devices is strengthened, hardware-level idiosyncrasies in microphones and speaker 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 observing audio, and conducting a simple spectral analysis on the recorded audio. Our approach can substantially simplify the ability for an adversary to track and identify people in public locations, identify callers, and produce other threats to the security and privacy of mobile device users. Our approach works well even with few samples— for example, we show that with our techniques, an adversary could even use the short ringtones produces by mobile device speakers to reliably track users in public environments.

Our approach centers around development of a novel 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 on an external device (i.e., laptop/PC) and vice versa for gen-
erating fingerprints of microphones. 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 elaborate study various audio features that can be used to accurately fingerprint smartphones. Our study reveals that mel-frequency cepstral coefficient (MFCC) is the dominant feature for fingerprinting smartphones. We also 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. We offer the following contributions:

- We propose a novel approach to fingerprinting smart devices. Our approach leverages the manufacturing idiosyncrasies of microphones and speakers embedded in smart devices.
- We study feasibility of a spectrum of existing audio features that can be used to accurately fingerprint smartphones. We find that the mel-frequency cepstral coefficient (MFCC) performs particularly well for fingerprinting smartphones.
- We investigate two different classifiers to evaluate our fingerprinting approach. We conclude that Gaussian Mixture Models (GMM) are more effective in classifying our 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 the accuracy of our fingerprinting.
- Finally, we discuss how our fingerprinting approach can be used as an additional factor for authentication.

Roadmap. The remainder of this paper is organized as follows. Section 2 gives an overview of our fingerprinting approach. We discuss why microphones and speakers integrated in smartphones can be used to generate unique fingerprints in Section 3. Section 4 describes the different audio features considered in our experiments, along with the classification algorithms used in our evaluation. Section 5 elaborately presents our experimental results. We discuss two diatomic applications of our device fingerprinting in Section 6. We describe some related works in Section 7. Section 8 discusses some limitations of our approach. Finally we conclude in Section 9.

2 Overview

In this section we give an overview of our approach, and identify the key challenges that we address in this paper.

The key insight behind our work is that imperfections in smart device hardware induce unique signatures on received/transmitted audio, and these unique signatures, if identified, can be used to fingerprint the device. Our approach consists of three key components. The first challenge we encounter 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 (e.g., the adversary has a microphone in a public setting to pick up device ringtones). The next challenge 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 construct a ‘fingerprint’ of the auditory characteristics of the device.

A key question that remains, which forms a major focus of this paper, is in construction of an effective fingerprinting scheme. Our goal is to determine a scheme that maximizes the ability to distinguish different devices. To do this, it helps to have some understanding of how devices differ at a physical level. Devices can vary at different layers of the manufacturing process. The most obvious way to distinguish devices manufactured by different vendors is to analyze the protocol stack installed in the devices. Usually different vendors have their own distinct features integrated inside the protocol stack. A close analysis of the protocol stack can help in distinguishing devices from different vendors. However, this approach is not helpful in distinguishing devices produced by the same vendor. To distinguish devices produced by the same vendor we need to look more deeply into the devices themselves because at the hardware level no two device are same. Hardware imperfections are likely to arise during the manufacturing process of sensors, radio transmitters and crystal oscillators suggesting the existence of unique fingerprints. This idiosyncrasies can be exploited to distinguish devices. Figure 1 illustrates the different device specific features that could be utilized to identify devices uniquely. We investigate properties of device hardware in more detail in Section 5.

A second aspect to this question is what sort of 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, pitch, 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 4.1 as well as algorithms for effectively clustering them in
3 Source of Fingerprints

In this section we will take a closer look at the microphones and speakers embedded on today’s smartphones. This will help understand how microphones and speakers can act as a potential source of unique fingerprints.

3.1 Closer Look at Microphones

Microphones in modern smartphones are based on Micro Electro Mechanical Systems (MEMS) [11,13,18]. 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 [11]. According to the iSuppli report, Apple and Samsung were the top consumers of MEMS microphones in 2012, accounting for a combined 54% of all shipped MEMS microphones [18].

A MEMS microphone, sometimes called a microphone chip or silicon microphone, consists of a coiled spring mechanism with 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 [8,12]. The CMOS often includes both a preamplifier as well as an analog-to-digital (AD) converter. Modern fabrication techniques enable highly compact designs, making them well suited for integration in digital mobile devices. The internal architecture of a MEMS microphone is shown on Figure 2.

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 allow 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 pressure-release mechanism where 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 [1]. 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.

3.2 Closer Look at Microspeakers

Micro-speakers are a scaled-down version of a basic acoustic speaker. So let’s first look at how speakers work before we discuss how microspeakers can be used to generate unique fingerprints. Figure 3(a) shows the basic components of a speaker. The diaphragm is usually made of paper, plastic or metal and its edges are connected to a voice coil. The voice coil is attached to the voice coil (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 3(b) shows a typical MEMS microspeaker chip and Figure 3(c) shows the components inside the micro-speaker [26,81]. The components are similar to that of a basic speaker; the only difference is the size and fabrication process [28,49,74]. 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. However, due to the inevitable variations and imperfections of the manufacturing process, no

---

1 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.
two speaker are going to be alike. Thus, subtle differences in sound generated by different speakers can arise. In our work, we develop techniques to computationally localize and evaluate these differences.

4 Audio Features and Classification Algorithms

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

4.1 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 uses the features to construct a fingerprint of the device. Computing acoustic features from an audio stream is a subject of much research \cite{20, 25, 61, 76}. 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 \ref{table:1}), all of which have been well-documented by researchers. A detailed description of each acoustic feature is available in Appendix\ref{appendix:a}.

4.2 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 classi-
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                                    |
| 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                |

In our work, we compare 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 maximally-likely association).

k-NN: The k-nearest neighbors algorithm (k-NN) is a non-parametric 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 [34]. 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 5.

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 [30]. 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 [70,76].

5 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 5.1). Next, we briefly discuss the setup for fingerprinting devices through speakers and microphones (Section 5.2 and 5.3). We then look at fingerprinting devices made by different vendors (Section 5.4) and later on focus on identifying devices manufactured by the same vendor (Section 5.5). We also perform an analysis of which features help most when identifying devices from the same vendor, by determining the dominant (most-relevant) set of audio features (Section 5.5.1). The performance of our approach is affected by certain aspects of the operating environment, and we study sensitivity to such factors in Section 5.6.

5.1 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 (used to house the computer science department). 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 fluorescent lighting. We placed smartphones in various locations in the room. To enu-
late 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 (an attacker willing to purchase a higher-quality microphone may attain better performance). We investigate how varying this setup affects performance of the attack in Section 5.6.

**Devices and tools:** We tested our device fingerprinting on devices from five different manufacturers. Table 2 highlights the model and quantities of the different phone sets used in our experiments. As we emphasized earlier we look at phones produced by both different and same manufacturer; hence the difference in quantities in Table 2.

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.

For analysis we leverage the following audio tools and analytic modules: MIRtoolbox [14], Netlab [16], Audacity [3], and the Android app Hertz [7]. 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 [75]—in our evaluation. Assuming there are fingerprints from n classes (i.e., n distinct phones), 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 ≤ i ≤ n). We then compute precision, recall, and the F1-score for each class using the following equations:

\[
\text{Precision, } Pr_i = \frac{TP_i}{TP_i + FP_i} \quad (1) \\
\text{Recall, } Re_i = \frac{TP_i}{TP_i + FN_i} \quad (2) \\
\text{F1-Score, } F1_i = \frac{2 \times Pr_i \times Re_i}{Pr_i + Re_i} \quad (3)
\]

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 tradeoff: 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 using the following equations:

\[
\text{Avg. Precision, } AvgPr = \frac{\sum_{i=1}^{n} Pr_i}{n} \quad (4) \\
\text{Avg. Recall, } AvgRe = \frac{\sum_{i=1}^{n} Re_i}{n} \quad (5) \\
\text{Avg. F1-Score, } AvgF1 = \frac{2 \times AvgPr \times AvgRe}{AvgPr + AvgRe} \quad (6)
\]

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 evaluations 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.

5.2 Process of Fingerprinting Speakers

An attacker can leverage our algorithms to passively observe audio emitted from device speakers (e.g., ringtones), 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 as shown in Figure 4. We look at both devices manufactured by different vendors and the same vendor.

---

Table 2: Types of phones used

| Maker       | Model       | Quantity |
|-------------|-------------|----------|
| Apple       | iPhone 5    | 1        |
| Google      | Nexus 4G    | 1        |
| Samsung     | Galaxy Note 2 | 1     |
| Motorola    | Droid A855  | 15       |
| Sony Ericsson | W518     | 1        |

Table 3: Types of audio excerpts

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

---

\[\text{Precision, } Pr_i = \frac{TP_i}{TP_i + FP_i} \]

\[\text{Recall, } Re_i = \frac{TP_i}{TP_i + FN_i} \]

\[\text{F1-Score, } F1_i = \frac{2 \times Pr_i \times Re_i}{Pr_i + Re_i} \]

\[\text{Avg. Precision, } AvgPr = \frac{\sum_{i=1}^{n} Pr_i}{n} \]

\[\text{Avg. Recall, } AvgRe = \frac{\sum_{i=1}^{n} Re_i}{n} \]

\[\text{Avg. F1-Score, } AvgF1 = \frac{2 \times AvgPr \times AvgRe}{AvgPr + AvgRe} \]

---

\[\text{AvgPr} \quad (4) \]

\[\text{AvgRe} \quad (5) \]

\[\text{AvgF1} \quad (6) \]

---

We also computed the 95% confidence interval, but we found it to be less than 0.01.
5.3 Process of Fingerprinting Microphones

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 feasibility of this attack, we will 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 5. Again we look at both devices manufactured by different vendors and the same vendor.

5.4 Fingerprinting Devices From Different Vendors

In this section we look at fingerprinting smartphones manufactured by five different vendor. We look at fingerprinting the devices through both microphone and speaker.

5.4.1 Fingerprinting Speaker

We found fingerprinting smartphones manufactured by different vendors is relatively simpler compared to fingerprinting devices manufactured by the same vendor. The main reason behind this is that the sensitivity of the speaker volume of different smartphones were quite different making it easier to track them. Figure 6(a) shows an audio sample played from five different smartphones. As we see the signal strength of the audio signals are quite different from each other. Hence, simple acoustic features like RMS value and spectral entropy are good enough to obtain good clusters of data points. Figure 6(b) shows a plot of spectral entropy vs. RMS value for 50 samples of an audio excerpt (10 samples from each handset). We see that acoustic features like spectral entropy and RMS value generate good clusters for each type of smartphone.

We test our fingerprinting approach using three different types of audio excerpts. Each audio sample is recorded 10 times giving us a total of 50 samples from the five handsets. 50% of the samples are used for training and the remaining 50% are used for testing, and we repeat this procedure for the three different types of audio excerpt. Table 4 summarizes our findings (values are reported as percentages). We simply use signal RMS value and spectral entropy as input features for the \( k \)-NN classifier, while for the GMM classifier we added MFCCs as an additional feature because doing so increased the GMM classifier’s success rate. From Table 4 we see that we can successfully (with a precision rate of 100%) identify which audio clip came from which smartphone. Thus fingerprinting smartphones manufactured by different vendors seems very much feasible using only 2 to 3 acoustic features.

Figure 4: Steps of fingerprinting speakers.

Figure 5: Steps of fingerprinting microphones.

Figure 6: a) Audio sample taken from five different handsets. b) Plotting audio samples taken from five different handsets using acoustic features — signal RMS value and spectral entropy.
5.5 Fingerprinting Devices of the Same Model

In this section we look at fingerprinting smartphones manufactured by the same vendor. We found that this was relatively a tougher problem and as such we first explore all the 15 acoustic features listed in Table 1 to determine the dominating subset of features. Next, we carry out our fingerprinting task using the dominant subset of acoustic features. We again fingerprint devices through all the 15 audio features described in Section 4.1 and the remaining 50% are used for testing. Table 5 summarizes our findings (values are reported as percentages).

Table 5: Fingerprinting different smartphones using microphone

| Audio Type   | k-NN                  | GMM                  |
|--------------|-----------------------|----------------------|
|              | AvgPr | AvgRe | AvgF1 | AvgPr | AvgRe | AvgF1 |
| Instrumental | 96.7  | 96.3  | 96.3  | 96.7  | 96.3  | 96.3  |
| Human speech | 93.3  | 92.6  | 92.6  | 96.7  | 96.7  | 96.3  |
| Song         | 96.7  | 96.3  | 100   | 100   | 100   | 100   |

* Feature numbers taken from Table 1.

5.5.1 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 explore all the 15 audio features described in Section 4.1 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 [46, 78]. 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 is 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
Algorithm 1  Sequential Feature Selection

Input: Input feature set $F$
Output: Dominant feature subset $D$

$F_1\_score \leftarrow []$

for $f \in F$ do
    $F_1\_score[f] \leftarrow Classify(f)$
end for

$F' \leftarrow sort(F, F_1\_score)$ # In descending order

$max\_score \leftarrow 0$

$D \leftarrow \emptyset$

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 $D$

not 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. Table 6 highlights the maximum F1-score 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). 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 (i.e., 5 samples per class) are used for training and the remaining 50% are 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. Table 6 shows the maximum F1-score achieved by each acoustic feature for the three different types of audio excerpt. We also provide the time required to extract each feature. The table highlights the subset of features selected by our sequential feature selection algorithm and their corresponding F1-score. We find that MFCCs are the dominant feature for all category of audio excerpt. Chromagram also generates high F1-score.

To give 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 7. 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 three handsets. For example coefficients 3 and 5 vary significantly across the three handsets. This makes MFCCs a prevalent choice for fingerprinting smartphones.

5.5.2 Fingerprinting Speakers

We now look at fingerprinting smartphones manufactured by the same vendor. For these set of experiments we use 15 Motorola Droid A855 handsets. Table 7 highlights our findings. We again 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 6. From Table 7 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.
5.5.3 Fingerprinting Microphone

We now investigate fingerprinting smartphones made by the same vendor through microphone-sourced input. We use 15 Motorola Droid A855 handsets for these experiments. We use the features obtained through Algorithm 1, which are listed in Table 6. Table 8 summarizes our findings. We see similar results compared to fingerprinting speakers. We were able to achieve an F1-score of 93% in identifying the handset from which the audio excerpt originated. Thus fingerprinting smartphones through microphones also appears to be feasible.

Table 8: Fingerprinting similar smartphones using microphone output

| Audio Type | Features | 1-NN AvgPr | 1-NN AvgRe | GMM AvgPr | GMM AvgRe |
|------------|----------|------------|------------|-----------|-----------|
| Instrumental | [13,14] | 95.9 | 95.3 | 95.6 | 95.6 |
| Human speech | [13,14] | 95.9 | 95.3 | 95.6 | 95.6 |
| Song | [13,14] | 95.9 | 95.3 | 95.6 | 95.6 |

5.6 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. For the following set of experiments we will only focus on fingerprinting smartphones from the same vendor and we only consider fingerprinting speakers as we saw almost identical outcomes for fingerprinting microphones. We also consider recording only ringtones (i.e., an audio clip belonging to our defined Instrumental category) for the following experiments. Since we are recording ringtones we only use the features highlighted in Table 6 under ‘Instrumental’ category.

5.6.1 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 8 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 frequency on most handheld devices today is 44.1kHz, with some of the latest models adopting even higher sampling rates. 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 8: Impact of sampling frequency on precision/recall.
5.6.2 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 9 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 9: Impact of varying training size

| Training samples per class | k-NN Features [13,14] | GMM Features [13,14] |
|----------------------------|-----------------------|----------------------|
|                            | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF |
| 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 | 98.3  | 98    | 98.1 |

* Feature numbers taken from Table 6

5.6.3 Varying Distance between Audio Source 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/recall. For this experiment we use a separate external microphone as the signal capturing capacity of microphones embedded inside laptops 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. Figure 9 shows the experimental setup and Table 10 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 as the distance between the smartphone and microphone increases, 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 10: Impact of varying distance

| Distance (meters) | k-NN Features [13,14] | GMM Features [13,14] |
|-------------------|-----------------------|----------------------|
|                   | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF |
| 0.1               | 96.7  | 96    | 96.3 | 98.3  | 98    | 98.1 |
| 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    | 61   | 76.8  | 76    | 76.4 |
| 5                 | 64.5  | 62.7  | 63.6 | 77    | 73.3  | 75.1 |

* Feature numbers taken from Table 6

5.6.4 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 scenario types were 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 11 highlights the four different scenarios that we are considering. To capture audio signals under such scenarios – external speakers (2 pieces), placed between the smartphone and microphone, were constantly replaying the respective ambient noise in the background while recording of ringtones played from different handsets were taking place. We consider a distance of two meters from the audio source to recorder. The ambient background sounds were obtained from PacDV [2] and SoundJay [17]. Table 11 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 11: Impact of ambient background noise

| Environments | k-NN Features [13,14] | GMM Features [13,14] |
|--------------|-----------------------|----------------------|
|              | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF | AvgPr | AvgRe | AvgF |
| Shopping Mall| 88.8  | 85.5  | 87   | 95.1  | 93.3  | 94.2 |
| Restaurant/Cafe| 90.5  | 89.7  | 90.1 | 92.5  | 90.7  | 91.6 |
| City Park    | 91.7  | 90    | 90.8 | 95.2  | 94.1  | 94.6 |
| Airport Gate | 91.3  | 89.5  | 90.4 | 94.5  | 93.3  | 93.9 |

* Feature numbers taken from Table 6
6 Applications

Fingerprinting smart devices can be thought of as a double-edged sword when it comes to device security. On one hand, it can jeopardize privacy, as it allows remote identification without user awareness. On the other hand, it could potentially be used to enhance authentication of physical devices. We discuss these potential applications below.

6.1 Multi-factor Authentication

Conventional computing systems authenticate users by verifying some static factors such as user generated passwords (which may be coupled with additional security questions like pin code or phone number). A password consists of a string of characters, remembered by the human user, which can be provided as a proof of identity. However, passwords are vulnerable to guessing algorithms. Moreover, if passwords ever leak they potentially open an opportunity for an unauthenticated user to access the system. Often systems do not incorporate mechanisms to verify whether the authenticated user is using an authorized device. Modern highly-secure organizations (e.g., military and department of defense) are therefore, moving towards using various forms of active authentication for their employees [5].

Device fingerprinting can be used to provide a multi-factor authentication framework that will enable a system administrator to validate whether authenticated users are using their allocated devices to log in into the system. This scenario of course is applicable to high-security conscious organizations where tracking authenticated users is not against any privacy violation. This can be done by leveraging our techniques, for example by instructing the user’s device to record an audio sample broadcast over the PA system, or transmit an audio session over the phone. Alternatively, the device may be able to “fingerprint itself”, by playing a received small audio clip out its speaker, simultaneously recording via its microphone, and then transmitting the result over the network to the authentication server for verification. In this way we can tie both user and device identity together to form a multi-factor authentication framework. As a side note this only provides additional assurance, rather than a foolproof authentication method. However, we believe our approach is more robust than existing software based two-factor authentication systems (e.g., for systems where you need to submit a token along with a password, if the attacker gets hold off the secret key then he/she can generate the desired token) as it is harder to mimic hardware level imperfections.

6.2 Device Tracking

By the same token, an attacker can violate user privacy by via a similar approach, or installing a malicious application on the user’s device, or recording broadcasted audio in public environments. For example, a malicious application (e.g., a game) can play small audio segments, record them using the device’s microphone, and upload recorded clips to the attacker. To do this, the application would require access to both microphone and network access permission, but this might not be a big assumption to make: most users are unaware of the security risks associated with mobile apps and a significant portion of the users cannot fully comprehend the full extension of all the permissions [18].

Alternatively, the attacker may sit in public environments (cafe, shopping mall), and record broadcasted audio (speakerphone conversations, ringtones) with the intent to track and identify users.

7 Related Work

Fingerprints have long been used as one of the most common biometrics in identifying human beings [29, 72]. The same concept was extended to identifying unique mobile transmitters by the US government during 1960s [55]. 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 [71].

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 differentiates between unique devices over a Wireless Local Area Network (WLAN) simply through a timing analysis of 802.11 probe request frames [31]. Others have looked at exploiting the difference in firmware and device driver running on IEEE 802.11 compliant devices [39]. 802.11 MAC headers have also been used to track unique devices [44]. Pang et al. [67] were able to exploit traffic patterns to carry out device fingerprinting. Open source toolkits like Nmap [59] and Xprobe [79] 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 [35]. Yen et al. [80] 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 [19]. Other researchers have proposed the use
of performance benchmarks for differentiating between JavaScript engines \[64\]. Furthermore, browsing history can be exploited to fingerprint and track web users \[66\]. 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 \[65\] and researchers have been able to exploit these clock skews to distinguish devices through TCP and ICMP timestamps \[54\]. However, clock skew rate is highly dependent on the experimental environment. Researchers have also extensively looked at fingerprinting the unique transient characteristics of radio transmitters (also known as Radio Frequency (RF) fingerprinting). RF fingerprinting has been shown as a means of enhancing wireless authentication \[22\] \[37\] \[65\] \[77\]. It has also been used for location detection \[68\]. Manufacturing imperfections in network interface cards (NICs) have also been studied by analyzing analog signals transmitted from them \[23\] \[41\]. More recently Dey et al. have studied manufacturing idiosyncrasies inside smartphone accelerometer to distinguish devices \[32\]. 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 accelerometer, as well as an external adversary who fingerprints nearby phones with a microphone.

Our work is inspired by the aforementioned hardware based fingerprinting works, but instead of focusing on wireless transmitters or on-board sensors that require external stimulation, we focus on fingerprinting on-broad acoustic components like microphones and speakers.

Audio fingerprinting has a rich history of notable research works \[25\]. There are studies that have looked at classifying audio excerpts based on their content \[45\] \[76\]. Others have looked at distinguishing human speakers from audio segments \[21\] \[24\]. There has also been work on exploring various acoustic features for audio classification \[61\]. One of the more popular applications of audio fingerprinting has been music genre and artist recognition \[47\] \[56\].

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

8 Discussion and Limitations

Our approach has several limitations. First, we experimented with 15 devices manufactured by the same vendor; it is possible that a larger target device pool would result in lower accuracy. That said, distinctions across different device types are more clear; additionally, audio fingerprints may be used in tandem with other techniques, such as accelerometer fingerprinting \[32\], to better discriminate between devices. Secondly, we evaluated our fingerprinting precision/recall under only two types of classifiers (GMM and k-NN). Other forms of classification such as ensemble based approaches could possibly achieve better results, as ensemble based methods use multiple models to obtain better predictive performance than any single classifier \[33\]. However, as a first step we were able to achieve over 93\% precision using simple k-NN and GMM classifiers, and our results may point to the concern that relatively simple techniques have a high success rate. 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.

9 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. By the same token, efficient fingerprinting may also serve to enhance authentication. To demonstrate feasibility of this approach, we collect fingerprints from five different brands of smartphones, as well as from 15 smartphones manufactured by the same vendor. 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.

Acknowledgement

We would like to thank Thomas S. Benjamin for his valuable input during the initial phase of the project. We would also like to thank the Computer Science department at UIUC and Google for providing us with the Motorola Droid phones.
References

[1] 5 of the best DACs. http://www.stuff.tv/music/5-best-dacs-how-make-your-digital-music-sound-amazing/feature. Accessed 02/27/2014.

[2] Ambient Sound Effects. http://www.pacdv.com/sounds/ambience_sounds.html. Accessed 02/27/2014.

[3] Audacity is free, open source, cross-platform software for recording and editing sounds. http://audacity.sourceforge.net/. Accessed 02/27/2014.

[4] Audio 4 Smartphones – Wolfson Microelectronics. http://www.wolfsonmicro.com/documents/uploads/misc/en/Audio4Smartphones.pdf. Accessed 02/27/2014.

[5] DARPA active authentication project. http://www.darpa.mil/Our_Work/I2O/Programs/Active_Authentication.aspx. Accessed 02/27/2014.

[6] Global mobile statistics 2013. http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/. Accessed 02/27/2014.

[7] How MEMS Microphones Function. http://www.eeherald.com/sound/2014/02/27/how-mems-microphones-function. Accessed 02/27/2014.

[8] IPhone and Android Apps Breach Privacy. http://www.gartner.com/newsroom/id/2335616. Accessed 02/27/2014.

[9] IPhone and Android Apps Breach Privacy. http://online.wsj.com/article/SB10001424052748704690400574602.html. Accessed 02/27/2014.

[10] MEMS microphone market. http://www.digikey.com/supply-chain-lgus/en/articles/sensors/mems-microphones/mems-microphone-market-revenues-soar-42-in-2012/1497. Accessed 02/27/2014.

[11] MEMS Microphone Model. http://www.comsol.com/blogs/mems-microphone-model-presented-as-a-166-san-francisco. Accessed 02/27/2014.

[12] MEMS microphone shipments to climb 30 percent in 2013. http://electroiq.com/blog/2013/02/mems-microphone-shipments-to-climb-30-percent-this-year/. Accessed 02/27/2014.

[13] MIRToolbox. https://www1.jyu.fi/hum/faitokset/musikkit/o/research/coe/materials/mirtoolbox/. Accessed 02/27/2014.

[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 02/27/2014.

[15] Netlab: Algorithms for Pattern Recognition. http://www1.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/book/. Accessed 02/27/2014.

[16] SOUNDJAY-Ambient Sound Effects. http://www.soundjay.com/ambient-sounds.html. Accessed 02/27/2014.

[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 02/27/2014.

[18] ACAR, G., JUAREZ, M., NIKIFORAKIS, N., DIAZ, C., GURSES, S., PIETSENS, F., AND PRENEEL, B. FPDetective: dusting the web for fingerprinters. In Proceedings of the 2013 ACM SIGSAC conference on Computer and Communications Security (2013), CCS ’13, pp. 1129–1140.

[19] BARTSCH, M. A., AND WAKEFIELD, G. H. Audio Thumbnailing of Popular Music Using Chroma-based Representations. IEEE TRANSACTIONS ON MULTIMEDIA 7, 1 (Feb 2005), 96–104.

[20] BIMBOT, F., BONASTRE, J.-F., FREDOUILLE, C., GRAVIER, G., MAGRIN-CHAGNONNEAU, I., MEIGNIER, S., MERLIN, T., ORTEGA-GARCÍA, J., PETROVSKA-DELECRETAZ, D., AND REYNOLDS, D. A Tutorial on Text-Independent Speaker Verification. EURASIP Journal on Advances in Signal Processing 4 (2004), 430–451.

[21] RASMUSSEN, K., AND CAPKUN, S. Implications of radio fingerprinting on the security of sensor networks. In Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on (Sept 2007), pp. 331–340.

[22] BRICK, V., BANERJEE, S., GRIETSE, M., AND OH, S. Wireless Device Identification with Radiometric Signatures. In Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (2008), MobiCom ’08, pp. 116–127.

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

[24] CANO, P., BATILLE, E., KALKER, T., AND HAITSMAN, J. A Review of Audio Fingerprinting. J. VLSI Signal Process. Syst. 41, 3 (Nov 2005), 271–284.

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

[26] CHEN, C. Signal Processing Handbook. Electrical and Computer Engineering. 1988.

[27] CHENG, M., HUANG, W., AND HUANG, S. R. A silicon microphone for hearing instruments. Journal of Micromechanics and Microengineering 14, 7 (Jul 2004), 859–866.

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

[29] DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B 39 (1977), 1–38.

[30] DESMOND, L. C. C., YUAN, C. C., PHENG, T. C., AND LEE, R. S. Identifying Unique Devices Through Wireless Fingerprinting. In Proceedings of the First ACM Conference on Wireless Network Security (2008), WiSec ’08, ACM, pp. 46–55.

[31] DEY, S., ROY, N., XU, W., CHOUDHURY, R. R., AND NELAKUDITI, S. AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable. In Proceedings of the 20th Annual Network and Distributed System Security Symposium (Feb 2014), NDSS’14.

[32] DIETERICH, T. Ensemble methods in machine learning. In Multiple Classifier Systems, vol. 1857. 2000, pp. 1–15.

[33] DUDA, R. H., L. P., AND STORCK, D. Pattern classification. Pattern Classification and Scene Analysis: Pattern Classification. Wiley, 2001.

[34] ECKERSLEY, P. How Unique is Your Web Browser? In Proceedings of the 10th International Conference on Privacy Enhancing Technologies (2010), PETS’10, pp. 1–18.

[35] EGELIE, M., KRUEGEL, C., KIRDA, E., AND VIGNA, G. PIoS: Detecting Privacy Leaks in iOS Applications. In Proceedings of the Network and Distributed System Security Symposium (NDSS) (San Diego, CA, February 2011).

[36] ENCK, W., GILBERT, P., CHUN, B.-G., COX, L. P., JUNG, J., MCDANIEL, P., AND SHETH, A. N. 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 (Berkeley, CA, USA, 2010), OSDI’10, USENIX Association, pp. 1–6.
[38] Felt, A. P., Ha, E., Egelman, S., Haney, A., Chin, E., and Wagner, D. Android Permissions: User Attention, Comprehension, and Behavior. In Proceedings of the Eighth Symposium on Usable Privacy and Security (2012), SOUPS ’12, pp. 3:1–3:14.

[39] Franklin, J., McCoy, D., Tabriz, P., Neagoe, V., Van Randwyk, J., and Sicker, D. Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting. In Proceedings of the 15th Conference on USENIX Security Symposium - Volume 15 (2006), USENIX-SS’06.

[40] Fujishima, T. Realtime chord recognition of musical sound: A system using common Lisp Music. In International Computer Music Conference (ICMA) (1999), pp. 464–467.

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

[42] Gibler, C., Crussell, J., Erickson, J., and Chen, H. 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 (Berlin, Heidelberg, 2012), TRUST’12, Springer-Verlag, pp. 291–307.

[43] Gouyon, F., Pachet, F., and Delerue, O. On the Use of Zero-Crossing Rate for an Application of Classification of Percussive Sounds. In Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFX-00) (2000).

[44] Guo, F., and Ck River Chieu, T. Sequence Number-Based MAC Address Spoof Detection. In Proceedings of 8th International Symposium on Recent Advances in Intrusion Detection (RAID) (2005), Springer.

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

[46] Gouyon, I., and Elissseeff, A. An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, 26 (mar 2003), 1157–1182.

[47] Haitjema, J., and Kalker, T. A Highly Robust Audio Fingerprinting System. In Proceedings of the International Symposium on Music Information Retrieval (2002), pp. 107–115.

[48] Harte, C., Sandler, M., and Gasier, M. Detecting Harmonic Change in Musical Audio. In Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia (2006), AMCM’06, pp. 21–26.

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

[50] Jensen, K. Timbre Models of Musical Sounds. PhD Dissertation. University of Copenhagen, 1999.

[51] Johnston, J. Transform coding of audio signals using perceptual noise criteria. IEEE Journal on Selected Areas in Communications 6, 2 (Feb 1988), 314–323.

[52] Justlin, P. N. Cue utilization in communication of emotion in music performance: Relating performance to perception. Journal of Experimental Psychology: Human Perception and Performance 26, 6 (2000), 1797–1813.

[53] Kelley, P., Consolvo, S., Cranor, L., Jung, J., Saadeh, N., and Wetherall, D. A Comumnd of Permissions: Installing Applications on an Android Smartphone. In Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2012, pp. 68–79.

[54] Kohno, T., Brodoo, A., and Claffy, K. C. Remote Physical Device Fingerprinting. IEEE Trans. Dependable Secur. Comput. 2, 2 (apr 2005), 93–108.

[55] Langley, L. Specific emitter identification (SEI) and classical parameter fusion technology. In WESCON’93. Conference Record, (Sep 1993), pp. 377–381.

[56] Li, T., Ogihara, M., and Li, Q. A Comparative Study on Content-based Music Genre Classification. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2003), SIGIR ’03, pp. 282–289.

[57] Li, Z., Xu, W., Miller, R., and Trappe, W. Securing Wireless Systems via Lower Layer Enforcements. In Proceedings of the 5th ACM Workshop on Wireless Security (2006), WiSe ’06, pp. 33–42.

[58] Logan, B. Mel Frequency Cepstral Coefficients for Music Modeling. In In International Symposium on Music Information Retrieval (2000).

[59] Lyon, G. Nmap: a free network mapping and security scanning tool. http://nmap.org/. Accessed 02/27/2014.

[60] Mahaffey, K., and Hering, J. 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

[61] Mckinney, M., and Brebaart, J. Features for Audio and Music Classification. In Proceedings of the International Symposium on Music Information Retrieval (2003), pp. 151–158.

[62] Misra, H., Ikbal, S., Bourlard, H., and Hermansky, H. Spectral entropy based feature for robust ASR. In IEEE International Conference on Acoustics, Speech, and Signal Processing (2004), vol. 1, pp. 193–196.

[63] Moon, S., Skelly, P., and Towsley, D. Estimation and removal of clock skew from network delay measurements. In Proceedings of the 18th Annual IEEE International Conference on Computer Communications (Mar 1999), vol. 1 of INFOCOM’99, pp. 227–234.

[64] Mowery, K., Bogrenief, D., Yilek, S., and Shachami, H. Fingerprinting Information in JavaScript Implementations. In Proceedings of W2SP 2011 (may 2011), IEEE Computer Society.

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

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

[67] Pang, J., Greenstein, B., Gummadii, R., Seshan, S., and Wetherall, D. 802.11 User Fingerprinting. In Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (2007), MobiCom ’07, pp. 99–110.

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

[69] Peeters, G. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Tech. rep., Ircam, 2004. http://recherche.ircam.fr/equipes/analyse-synthese/peeters/ARTICLES/Peeters_2003_cuidadoaudiofeatures.pdf

[70] Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. Speaker verification using adapted gaussian mixture models. Digital Signal Processing 10, 1-3 (2000), 19–41.
The zero-crossing rate is 

**Zero Crossing Rate (ZCR):** The zero-crossing rate is the rate at which the signal changes sign from positive to negative or back [27]. This feature has been used heavily in both speech recognition and music information retrieval, for example to classify percussive sounds [43]. ZCR for a signal \( s \) of length \( T \) can be defined as:

\[
\text{ZCR} = \frac{1}{T} \sum_{t=1}^{T} |s(t) - s(t-1)|
\]

where \( s(t) = 1 \) if the signal has a positive amplitude at time \( t \) and 0 otherwise. Zero-crossing rates provide a measure of the noisiness of the signal.

**Low Energy Rate:** The low energy rate computes the percentage of frames (typically 50ms chunks) with RMS power less than the average RMS power for the whole audio signal. For instance, a musical excerpt with some very loud frames and a lot of silent frames would have a high low-energy rate.

**Spectral Centroid:** The spectral centroid represents the “center of mass” of a spectral power distribution. It is calculated as the weighted mean of the frequencies present in the signal, determined using a Fourier transform, with their magnitudes as the weights:

\[
\text{Centroid}, \mu = \frac{\sum_{i=1}^{N} f_i \cdot m_i}{\sum_{i=1}^{N} m_i}
\]

where \( m_i \) represents the magnitude of bin number \( i \), and \( f_i \) represents the center frequency of that bin.

**Spectral Entropy:** Spectral entropy captures the spikiness of a spectral distribution. As a result spectral entropy can be used to capture the formants or peaks in the sound envelope [62]. To compute spectral entropy, a Digital Fourier Transform (DFT) of the signal is first carried out. Next, the frequency spectrum is converted into a probability mass function (PMF) by normalizing the spectrum using the following equation:

\[
w_i = \frac{m_i}{\sum_{i=1}^{N} m_i}
\]

where \( m_i \) represents the energy/magnitude of the \( i \)-th frequency component of the spectrum, \( w = (w_1, w_2, \ldots, w_N) \) is the PMF of the spectrum and \( N \) is the number of points in the spectrum. This PMF can then be used to compute the spectral entropy using the following equation:

\[
H = \sum_{i=1}^{N} w_i \cdot \log_2 w_i
\]

The central idea of using entropy as a feature is to capture the peaks of the spectrum and their location.

**Spectral Irregularity:** Spectral irregularity measures the degree of variation of the successive peaks of a spectrum. This feature provides the ability to capture the jitter
or noise in spectra. Spectral irregularity is computed as the sum of the square of the difference in amplitude between adjoining spectral peaks [50] using the following equation:

\[
\text{Irregularity} = \frac{\sum_{i=1}^{N} (a_i - a_{i+1})^2}{\sum_{i=1}^{N} a_i^2}
\]

where the \((N + 1)\)-th peak is assumed to be zero. A change in irregularity changes the perceived timbre of a sound.

**Spectral Spread:** Spectral spread defines the dispersion of the spectrum around its centroid, i.e., it measures the standard deviation of the spectral distribution. So it can be computed as:

\[
\text{Spread}, \sigma = \sqrt{\frac{\sum_{i=1}^{N} [(f_i - \mu)^2 \cdot w_i]}{N}}
\]

where \(w_i\) represents the weight of the \(i\)-th frequency component obtained from equation (10) and \(\mu\) represents the centroid of the spectrum obtained from equation (9).

**Spectral Skewness:** Spectral skewness computes the coefficient of skewness of a spectrum. Skewness (third central moment) measures the symmetry of the distribution. A distribution can be positively skewed in which case it has a long tail to the right while a negatively-skewed distribution has a longer tail to the left. A symmetrical distribution has a skewness of zero. The coefficient of skewness is the ratio of the skewness to the standard deviation raised to the third power.

\[
\text{Skewness} = \frac{\sum_{i=1}^{N} [(f_i - \mu)^3 \cdot w_i]}{\sigma^3}
\]

**Spectral Kurtosis:** Spectral Kurtosis gives a measure of the flatness or spikiness of a distribution relative to a normal distribution. It is computed from the fourth central moment using the following function:

\[
\text{Kurtosis} = \frac{\sum_{i=1}^{N} [(f_i - \mu)^4 \cdot w_i]}{\sigma^4}
\]

A kurtosis value of 3 means the distribution is similar to a normal distribution whereas values less than 3 refer to flatter distributions and values greater than 3 refers to steeper distributions.

**Spectral Rolloff:** The spectral rolloff is defined as the frequency below which 85% of the distribution magnitude is concentrated [76]

\[
\arg \min_{f_c \in \{1, \ldots, N\}} \sum_{i=1}^{f_c} m_i \geq 0.85 \sum_{i=1}^{N} m_i
\]

where \(f_c\) is the rolloff frequency and \(m_i\) is the magnitude of the \(i\)-th frequency component of the spectrum. The rolloff is another measure of spectral shape that is correlated to the noise cutting frequency [69].

**Spectral Brightness:** Spectral brightness calculates the amount of spectral energy corresponding to frequencies higher than a given cut-off threshold. This metric correlates to the perceived timbre of a sound. Increase of higher frequency energy in the spectrum yields a sharper timbre, whereas a decrease yields a softer timbre [52]. Spectral brightness can be computed using the following equation:

\[
\text{Brightness}_{fc} = \sum_{i=f_c}^{N} m_i
\]

where \(f_c\) is the cut-off frequency (set to 1500Hz) and \(m_i\) is the magnitude of the \(i\)-th frequency component of the spectrum.

**Spectral Flatness:** Spectral flatness measures how energy is spread across the spectrum, giving a high value when energy is equally distributed and a low value when energy is concentrated in a small number of narrow frequency bands. The spectral flatness is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum [51]:

\[
\text{Flatness} = \left(\prod_{i=1}^{N} m_i\right)^{1/N} \quad \frac{1}{N} \sum_{i=1}^{N} m_i
\]

where \(m_i\) represents the magnitude of bin number \(i\). Spectral flatness provides a way to quantify the noise-like or tone-like nature of the signal. One advantage of using spectral flatness is that it is not affected by the amplitude of the signal, meaning spectral flatness virtually remains unchanged when the distance between the sound source and microphone fluctuates during recording.

**Mel-Frequency Cepstrum Coefficients (MFCCs):**

MFCCs are short-term spectral features and are widely used in the area of audio and speech processing [58][76]. Their success has been due to their capability of compactly representing spectrum amplitudes. Figure [10] highlights the procedure for extracting MFCCs from audio signals. The first step is to divide the signal into fixed size frames (typically 50ms chunks) by applying a windowing function at fixed intervals. The next step is to take Discrete Fourier Transform (DFT) of each frame. After taking the log-amplitude of the magnitude spectrum, the DFT bins are grouped and smoothed according to the perceptually motivated Mel-frequency scaling [4]. Finally, in order to decorrelate the resulting feature vectors a discrete cosine transform is performed. We use the first 13 coefficients for our experiments.

---

4Mel-scale approximates the human auditory response more closely than the linearly-spaced frequency bands. [http://en.wikipedia.org/wiki/Mel_scale](http://en.wikipedia.org/wiki/Mel_scale)
Chromagram: A chromagram (also known as harmonic pitch class profile) is a 12-dimensional vector representation of an audio signal showing the distribution of energy along the 12 distinct semitones or pitch classes. First a DFT of the audio signal is taken and then the spectral frequencies are mapped onto a limited set of 12 chroma values in a many-to-one fashion [40]. In general, chromagrams are robust to noise (e.g., ambient noise or percussive sounds) and independent of timbre change.

Tonal Centroid: Tonal centroid introduced by Harte et al. [48] maps a chromagram onto a six-dimensional Hypertorus structure. The resulting representation wraps around the surface of a Hypertorus, and can be visualized as a set of three circles of harmonic pitch intervals: fifths, major thirds, and minor thirds. Tonal centroids are efficient in detecting changes in harmonic contents.