Acoustic Fingerprinting Revisited: Generate Stable Device ID Stealthily with Inaudible Sound

Zhe Zhou, Wenrui Diao, Xiangyu Liu, Kehuan Zhang
Department of Information Engineering
The Chinese University of Hong Kong
{zz113, dw013, lx012, khzhang }@ie.cuhk.edu.hk

ABSTRACT

The popularity of mobile devices has made people's lives more convenient, but threatened people's privacy at the same time. As end users are becoming more and more concerned on the protection of their private information, it is even harder for hackers to track a specific user by using conventional technologies. For example, cookies might be cleared by users regularly. Besides, OS designers have developed a series of measures to cope with tracker. Apple has stopped apps accessing UDIDs, and Android phones use some special permissions to protect IMEI code. However, some recent studies showed that attackers are able to find new ways to get around those limitations, even though these new methods should be improved in order to be practically deployed in large scale. For example, attackers can trace smart phones by using the hardware features resulting from the imperfect manufacturing process of accelerometers. In this paper, we will present another new and more practical method for the adversaries to generate stable and unique device ID stealthily for the smartphone by exploiting the frequency response of the speaker. With carefully selected audio frequencies and special sound wave patterns, we can reduce the impact of non-linear effects and noises, and keep our feature extraction process un-noticeable to phone owners. The extracted feature is not only very stable for a given smart phone, but also unique to that phone. The feature contains rich information, which is even enough to differentiate millions of smart phones of the same model. We have built a prototype to evaluate our method, and the results show that the generated device ID can be used to track users practically.

Categories and Subject Descriptors

D.4.6 [**Operating System**]: Security and Protection—*Invasive software*

General Terms

Security

Keywords

Smartphone; Device Fingerprint; Acoustic Fingerprint

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

ACM 978-1-4503-2957-6/14/11 ...$15.00.
http://dx.doi.org/10.1145/2660267.2660300.

1. INTRODUCTION

Smart phones are playing an increasingly important role in our daily lives, including both work and personal entertainment, which makes the security of smart phones, especially the protection of user privacy, a very important and urgent problem. Smart phone sales are experiencing nearly 40% increasing year on year reported by IDC[8]. However, according to F-secure, a continued 49% raising of mobile threat was witnessed in the last quarter, and 91.3% of them targeted at Android platform, the most popular mobile operating system today[4]. Different from traditional desktop PCs, smart phones often contain more private and sensitive information, like SMS, contacts, location, etc. And studies showed that such sensitive data is the major reason why smart phones are so attractive to attackers [43].

Fortunately, people are becoming better educated to know how to protect their privacy. Statistics from Pew Internet Project show that almost 90% of adult Internet users have taken steps to avoid surveillance by other people or organizations, like clearing cookies, encrypting email, and using aliases [2]. To attract users, major browsers now support various privacy protection features, like "Don't Track", third party cookie disabling, etc. Governments and organizations are also working on laws to protect users' privacy.

However, being able to track users is really profitable in many applications, whether it is used legitimately or maliciously. So, it is not surprising to see that many big companies declare plans to give up using cookie on one side, and work on new tracking technologies on the other side [6]. There are also many studies on the stop-tracking and new tracking technologies in the academia world [36, 32, 39, 28, 41, 29, 35].

Among these new tracking technologies, some suggested to use device IDs when cookie is absent [11], mainly because that device IDs are straightforward and cannot be wiped or reset easily. Typically, many things can be used as device IDs, such as UDID (Unique Device ID) from Apple, IMEI for general mobile phones, Android ID for Android phones, MAC addresses of Wi-Fi and Ethernet network interfaces or Bluetooth modules [40], and so on. Some recent researches also suggested to construct device IDs by using hardware features resulted from imperfect manufacture process, like accelerometers [21] and speakers [19].

But each of these new methods has its own limitations, thus their potential threat could be under-estimated. On one hand, system vendors can easily block the access of a device ID by removing relevant APIs, and on the other hand, some newly discovered device IDs are not mature enough to be deployed in real world production scenarios. For example, Apple ceased the use of UDID recently [3], and on Android, accessing IMEI requires a special permission that could be revoked by Google if necessary (actually, Google made changes to Android permission system from time to time, and recently, they just revoked the permission on SD card writing [1], so there is no guarantee that they would not remove the permissions granting related to IMEI and other possible device IDs). For newly discovered device IDs, like those extracted from accelerometers and speakers, the false positive rates are still too high, and they are not stable and robust enough to give unique IDs to large volume of devices (more details are given in section 8).

To raise an alarm on such alternative user-tracking methods, in this paper we will propose another device ID generation method that is more practical by reaching following requirements better: uniqueness, robustness, and stealthiness. Our basic idea is to leverage frequency responses of speakers on smart phones as hardware-based identification. But our techniques are totally different from previous work and can improve the final results dramatically.

One of our fundamental difference to previous work is the use of high frequency sound. In previous work [19], a piece of music is played, and since its frequency range is normally lower than 10 kHz, it can be easily heard by the smart phone owner. What's more, the majority of environmental noises also fall into this range, which makes the feature extraction difficult and unstable.

By contrast, our method uses audio frequency that is higher than 14 kHz, which is chosen after careful studies of various factors, including the environment noises, characteristics of human hearing, as well as the manufacturing technologies of speakers. For example, as shown by our experiments in section 5, in most cases, there are less noises in higher frequency range. What's more, studies of human hearing indicate that our ears are much less sensitive to sound with higher frequency, which means that people can easily hear a sound with 4 kHz at 30 dB, but can hardly perceive another 16 kHz sound at the same 30 dB (more details in section 3).

More importantly, we have found that speakers' performances are much more diversified at higher frequency range, which helped us be able to get unique feature for each of them with negligible false positive and false negative rates. Ideally, we would expect each speaker performs in the same way. However, this is impossible in real world, so, manufacturers have to optimize speakers with trade-offs among the cost, manufacturing technology, and the perception of human ears. As mentioned above, people are more sensitive to low frequency audio, so the speaker manufacturers focus on the optimization at lower frequency range first, and optimize higher frequency range later only if cost/budget permits. As a result, it is not surprising that the frequency response curves are similar at lower frequency range, but differ to each other dramatically at higher frequency range (more details will be given in section 3).

Another fundamental difference to previous work is that we construct audio stimulus pattern carefully to minimize the impact of non-linear characteristics of speakers and background noises. Instead of playing a piece of random chosen music, as was done in previous work, we choose to output a stable combination of about seventy different frequencies, and later when extracting features, only analyze response at these frequency points. So, noises not on those frequency points can be filtered, and more importantly, the speaker can work in a stable state with its features being exposed steadily and completely. We believe that such design is crucial to get unique and robust device ID.

Contributions. We summarize our contributions as follows:

- We carefully analyzed many different factors that could affect the construction of unique and robust device ID from mobile phone speakers, and proposed to use high frequency sound with special frequency pattern as stimulation to speakers, which can not only make the whole process unnoticeable by the smart phone owners, but also can minimize the impact of background noises and non-linear features.

- We developed novel algorithms to extract and match features from the recorded speaker responses, which is built on self-correlation and cross-correlation functions, instead of using complex machine learning algorithm. We also developed a method to estimate the potential false positive and false negative rates.

- We built a prototype and performed a comprehensive evaluation over the proposed method, and the results showed that the extracted device ID is very stable, with negligible false positive and false negative rates.

Roadmap. The rest of the paper is organized as follows. We list required assumptions and adversary models in Section 2 and then give an overview of our proposed method in section 3. The details of our design in given in section 4, followed by a comprehensive evaluation of the proposed design in terms of different metrics in Section 5. Section 6 presents some real world applications of the proposed techniques. We will discuss the potential limitations in Section 7, and compare our work with prior ones in Section 8. Section 9 concludes the paper.

2. ADVERSARY MODEL

This section describes the assumptions required to extract device IDs from smart phone speakers, and the potential adversary/application scenarios where our method may be applicable.

2.1 Application Scenarios

We fully respect that people have the right to protect their privacy, so the most important goal of this paper is not to propose new tracking techniques to invade people's privacy, but instead to bring a new possible privacy attack into people's attention and raise an alarm of such new user tracking method.

However, as a device fingerprinting technology, the proposed user tracking method itself is neutral, so if it is used properly, it could benefit our society to a large extent. For example, it can be used to identify and track stolen phones to support self-destruction functionality that is now required by law in some places. It is also useful to support accurate in-door positioning to provide better shopping experience in supermarket. More details will be given in Section 6.

2.2 Assumptions

The device fingerprinting process actually contains three steps: playing a piece of specially crafted audio, recording the speaker output, and transmitting the preprocessed feature to servers. These three steps can be mapped to three different operations or permissions: playing audio, accessing microphone, and accessing Internet.

- Play audio: According to current Android permission mechanism, playing audio does not require any permission.

- Access to microphone: This is the only necessary permission required by our proposed method, since we have to record the speaker output. However, depending on the specific application scenarios, the microphone permission could be located on the same phone that plays the audio (i.e., self-fingerprinting), or on a different phone (cross-fingerprinting).

- Access to Internet: This permission is unnecessary and can be bypassed because of an existing vulnerability mentioned in [45] by appending the data to a `GET` request. The size of each extracted feature never exceeds 1 KB, so the length limitation of GET request is also not a problem.

3. OVERVIEW

In this section we will introduce the reason why we study sound acoustic fingerprinting of mobile devices though some related work already existed. A brief description on the technical background of our approach will also be presented.

3.1 Three Goals to Be Achieved

We believe that every device fingerprinting technology should achieve the following three goals simultaneously: *uniqueness*, *robustness*, and *stealthiness*. In terms of **uniqueness**, the fingerprints generated from different devices should be different enough from each other, otherwise there would be serious usability problem (imagine that two different users share an identical cookie). **Robustness** means the fingerprints generation method should be able to generate a consistent fingerprints for the same device at different time and under different scenarios. The last goal, **stealthiness**, requires the fingerprints generation process to be unnoticeable by device owners.

Limitations of existing solutions. When considering above goals, we found that existing solutions have various limitations. For example, the work done in [19] needs to play some audible music, which makes it hard to achieve "stealth" goal. In another work that uses accelerometers to track users, there would always be at least 1 device out of 107 wrongly identified, which may not be accurate enough for cookie based applications in real world [21]. More details will be given in related work section 8.

3.2 Our Key Techniques

Our key techniques could be described in a single sentence: we use microphones to record the output from device speakers stimulated by high frequency audio wave with some special pattern. However, it requires more words to explain the rationale behind and how uniqueness, robustness, and stealthiness are achieved by these techniques.

3.2.1 *Be Stealthy with High Frequency Audio*

Common sense tells us that human being cannot hear all sound generated by the world. For example, infrasonic wave produced by earthquake doesn't make any feeling to human but can be detected by machines, which plays an important role in the disaster forecasting. Ultrasonic possesses similar attributes. Figure. 1 shows human's hearable zone [10].

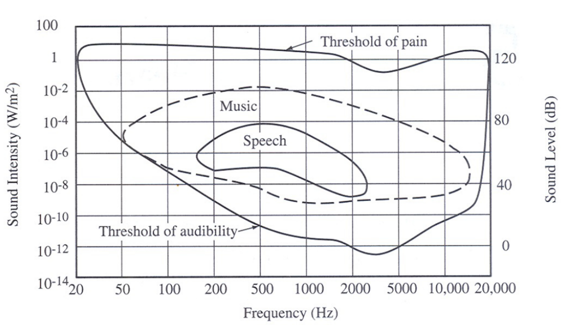

Figure 1: Sound Pressure Level of Human over Frequency.

Normally, people cannot hear our stimulus as shown in Figure 2 for some reasons. Firstly, most people are sensitive to sound from some hundreds Hz to some thousands Hz, and people can only feel little when the sound is lower than 200 or higher than 15 kHz, considering the limited volume of the phone. In other words, you can hear almost nothing if your cell-phone is playing a clip of music with null spectrum between 200 and 15k Hz. Secondly, the energy transform efficiency of the speaker decreases with the increasing frequency. So, volume decreases with the increasing frequency at a given input power. Thirdly, the audio is a wide band audio with energy distributed evenly at multiple frequency component, each of which possesses only little part energy. This point paraphrases why the special audio contains components between 14k and 15k Hz, but cannot be heard.

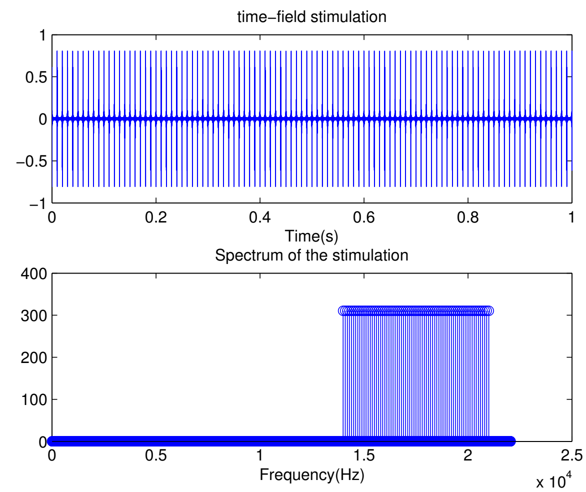

Figure 2: Stimulation.

3.2.2 *Be Unique with High Frequency Audio*

Inside each speaker driver, a flexible cone attached with a coil of wire is mounted on the suspension, which allows it to move freely inside the magnet. The coil, passing electrical currents, creates a varying magnetic field, and the field interacts with the fixed magnet to drive the cone to fluctuate according to the currents [18]. Figure. 3 illustrates the structure of the speaker [18].

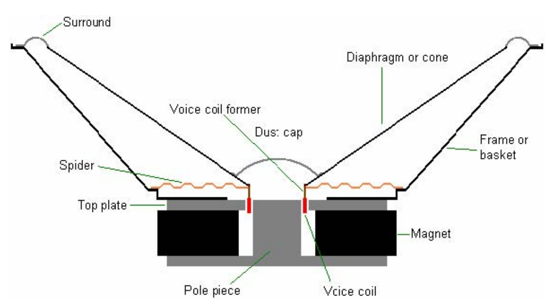

Figure 3: Sectional View to Speaker Driver.

High-end speaker systems may contain more than a single driver to let each driver focus on each frequency band and enhance the quality thereby, because one driver can hardly handle the entire audible frequency range limited by the mechanical feature of the driver. Products from the lower-end speaker market, like those used in our phones, usually have only one driver. Manufactures are capable to control the quality of their product in only a narrow frequency range, while quality outside the important frequency range is less concerned for some reasons.

Firstly, the important frequency range covers most of human's sensitive frequency range, and we are not sensitive to the left frequency range, which leads the quality control outside the main frequency range to be less meaningful.

Secondly, compensating the quality costs a lot, which will increase the overall costs and decrease the competitiveness of the manufactures in terms of price. For example, adding an independent high frequency driver enhances the quality sharply, but it increases the cost multiple times. So phones in the market are often equipped with only one speaker driver.

As a result, manufactures control the sensitive range quality and neglect the insensitive frequency range.

Frequency response presents the quality of a speaker from the perspective of frequency by reflecting the gain or attenuation the speaker provide at each frequency point. Thus, it is easy to conclude that the flatter the response curve is, the better voice quality it will provide. Figure 4, captured from the Internet [5], presents the frequency response of three speakers, which shows that: at low frequency segment, they have similar response curves, while, at high frequency segment, their response curves are different from each other dramatically. Not only the differences between different models of speakers but also the variances between the same model are huge.

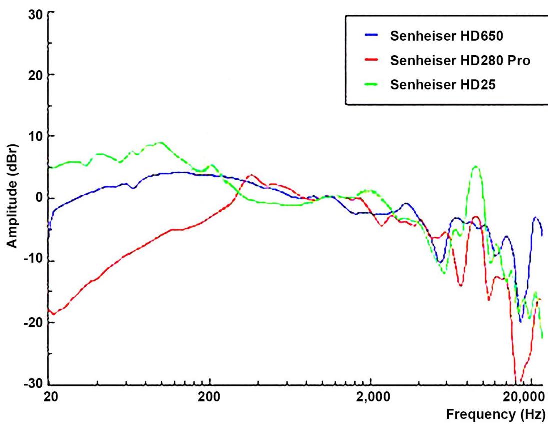

Figure 4: Frequency response of 3 speakers.

Both theoretical analysis and experimental result, which will be shown in the evaluation section, drive us to decide to use the high frequency range response feature, as it carries high variations among each speaker individuals.

3.2.3 Be Robust with Controlled Stimulus Patterns

The sampling data collected by many previous work is just the result of uncontrolled input stimulus. For example, in [21], the accelerometer readings are stimulated by random user movement. In [19], even though the music played could be controlled, but the frequency component combinations and variations are determined by the stimulation as well as the abundant noise permutated in the environment. Due to the non-linear features of speakers, like intermodulations [18], the recorded sound may contain lots of noises that would make the result unstable.

In contrast, we propose to use a controlled audio wave pattern to drive the speaker, so that the results will be more robust to random and non-linear factors, and less vulnerable to noises. The stimulation is shown in Figure 2.

The stimulation lies in a frequency range that interfered only little by the environment. As the spectrum of noise in different environment will be shown in Figure. 12, we found the silent environment in high frequency band provides a perfect test bed for measuring the frequency response of the speaker. It is just the less-interfered environment, controlled stimulation that brought robustness to the scheme.

4. DESIGN

In this section, we will introduce how the scheme works. Specifically, how it generates inaudible stimulation stealthily, calculates frequency response and searches the feature in the database.

4.1 Stimulation Generation

In our scheme, the android phone itself generates appropriate acoustic signal by playing a period of synthetic sound as the stimulation and itself collects the response from the microphone. Comparing with the passive generation, where the response is highly affected by the stimulation provider, active one, in fact, provides plain, pure and noise-less response resulted from a self-controlled stimulation.

We didn't use a wave with continuous flat frequency band because the power of the signal is constrained resulted from a very high PAPR (Peak to Average Power Ratio) in that case. We also didn't adopt a frequency shifted music, because the combinations of complex frequency make the output unstable resulted from the non-linearity attribute of the speaker. Instead, we adopted the stimulation shown in figure. 2. It is consisted of a series of cosine wave from 14 kHz to 21 kHz with 100 Hz gap between neighbor frequency points. In order to play the high frequency sound, we set the sample rate of the PCM format input of the android API at 44100 Hz.

4.2 Frequency Response Generation

A simple frequency response measuring scheme is adopted. Professionals often use DAAS(Digital Audio Analysis Software) to get precise result of the frequency response curve of the speaker. However, acquiring the response feature is infeasible if the phone should be tested in a noise blocked room by an instrument and without phone users' awareness. Therefore, the microphone of the phone itself is adopted to collect the acoustic signal broadcasted by the speaker, though the noise introduced in this way is obviously more than that in a professional way.

Specifically, to get the frequency response, we use the spectrum of recording divided by the spectrum of stimulation. The spectrum of recording is calculated by the FFT. The process of being divided by the stimulation can be neglected since the magnitude of the stimulation at each effective frequency point is constant and the response will be normalized later.

Considering the response feature, at the effective point, the frequency response is calculated with interference of noise brought by the environment, while at the point in the gap between effective points, the response is meaningless because only noise exists. Therefore, only effective points are counted when producing the feature. Besides, in each point, the phase can be neglected comparing with the significance of aptitude. Therefore, we only calculate magnitude instead of considering the complex number.

To save communication bandwidth and storage, in this scheme, only magnitudes of 71 effective frequency points are counted, and it is not the truth that the more points are sampled, the higher entropy will be accumulated, because the power of the stimulation will be allocated to each frequency point, where insufficient power leads to insufficient SNR (signal noise ratio) and an unstable curve thereby.

4.3 Feature Matching

The frequency response can be presented as a curve that can be discretized to some points, thus, a vector. Matching two devices is equivalent to matching the two curves, hence, the vectors the two devices owned. To judge if the two vectors come from the same device, the proof is their similarity. The more similar the two vectors are, the more possible that they come from the same device. Mathematically, the distance between two vectors can be utilized to weight the similarity between two vectors. The shorter the distance is, the more similar they will be. Once the newly received feature is close enough to some existed feature in the database, they will be judged as produced by that device. Otherwise, a new profile will be set up for the new comer.

In the experiment phase, we just use the brute force algorithm to get the most similar feature vector met before and judge if the distance between them reached a predefined threshold(an experimental value 0.7 is set in the experiment phase) to tell if it is a new user or it is just the user the most similar vector represents. Because ultra-large scale data has not been collected and searched, this scheme runs pretty fast. In fact, with the expansion of the scale of the data, matching users one by one becomes time wasting and infeasible. But this never masks the fact that the float vector can be easily fuzzy searched using Locality Sensitive Hashing or k-NN algorithm. In that case, the searching time complexity can be reduced to nearly a constant. [24, 44, 37]

5. EVALUATION

As a practical and feasible fingerprint, the scheme should be inspected in some aspects. For example, fingerprint should be stable as it changes little from time to time, reminding us to check the stability of the frequency response. This section shows our test results to answer the following questions:

- *Performance* Can the scheme be applied to large scale user tracking? Specifically, can a large amount of users be distinguished from each other?
- *Stability* How stable is the response curve? Is it feasible for long term user tracking?
- *Interference* How does the noise in different environment interfere the performance of the scheme?

5.1 Experiment Setting

5.1.1 *Experiment Devices*

The evaluation starts with a small scale experiment among 8 smartphones of different models. And the result shows that they can be distinguished with huge differences. Previous work [19] presents a similar argument. Thus, it is proper to focus on distinguishing phones of the same model. So, we investigate the result of a large scale experiment with phones of the same model.

To prove that phones can be distinguished by only speakers, we designed an experiment bed to emulate multiple phones, among which the only difference is speaker. We conduct the experiments on 50 OEM speakers on a single Samsung Galaxy S3. We modified the Galaxy S3 by converting the soldered speaker interface into a pluggable socket, as shown in Figure 5, then we purchased 50 OEM (Original Equipment Manufacturer) speakers which came from the same assembly line and have continuous Serial Numbers. These speakers were soldered with two-pin plugs, so that they can be easily connected to the phone.

The difference between phones is even larger than the difference between speakers. The difference between phones is the product of all differences of their corresponding subsystems. Components besides speaker, like DAC, chasis also contribute to the overall difference. For example, the chasis is a sound amplifier with distortion. Different chasises may distort the sound in a tiny different way. However most of the difference is contributed by speaker, because it is a mechanical and kinetic component where quality control is more difficult. Therefore, we regard the difference between speakers as the difference between phones in the following part.

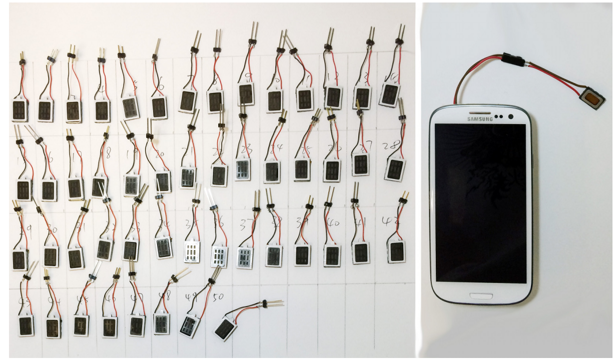

Figure 5: Experiment Equipment.

To each emulated phone, 60 sets of response feature were collected for further evaluation. Thus, totally 3000 vectors have been collected.

5.1.2 *Experiment Environment*

To study the scheme justifiedly, the experiment is conducted in the normal office environment with normal noise level except the interference part. During the experiment, the noise level changes from 50 db to 70 db, that mixed with normal conversation to emulate a real office environment. The volume of the phone is set at 5 out of 7.

5.2 Metrics

The metrics listed are used to evaluate the scheme:

- *Feature Distance* Since the feature is actually a vector in N-space, we simply define the feature distance as the Euclidian distance in N-space listed below:

$$d(p,q) = \sqrt{\sum_{i=0}^{N} (q_i - p_i)^2}$$

where p and q are two feature vectors defined as:

$$p = (p_0, p_1, \cdots, p_{N-1}), q = (q_0, q_1, \cdots, q_{N-1})$$

- *Similarity* We use similarity to measure how likely the two features p, q are coming from the same phone, and it is defined as

$$1 - d(p,q)$$

- *False Positive* We define a case as false positive if phone A is falsely recognized as another phone B based on the input features.
- *False Negative* We define a case as false Negative if no matches can be found in the database for features from phone A that actually does exist in the database.

- *Entropy* The logarithm of (the size of the distinguishable set) to base 2 is the entropy of the scheme. The distinguishable set is the set that all the contained elements can be distinguished from each other by the produced feature.

5.3 Performance

At first, we planned to count the number of errors. So, the 3000 feature vectors are input to the process in a random sort. The output was checked with right answer to count false positive and false negative. Neither false positive nor false negative was found among them. However it can hardly justify the performance of the scheme when the quantity of test cases increases sharply. Consequently, we refer to the distribution of the similarity to calculate the performance in the large scale case.

5.3.1 Distribution of similarity

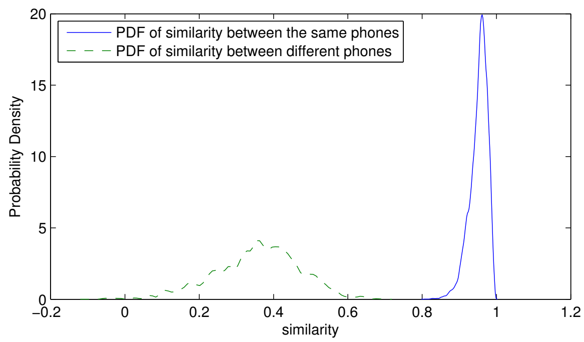

Figure 6: Distribution of Similarities.

We found that there is a gap between similarities of the same phone and the similarities of different phones, which is the main reason of the good performance. We investigated the distribution of similarities between different phones (sim_{corr}) and within the same phones (sim_{self}) respectively. Specifically, in terms of sim_{self}, to each device, comparison between the 60 features results to C_{60}^2 sim_{self}. Thus, totally 50* C_{60}^2 sim_{self} are collected. In terms of sim_{corr}, there are C_{50}^2 devices pairs, where 60*60 similarities can be calculated in each pair. Therefore, totally, $3600 * C_{50}^2$ sim_{corr} are collected. The PDF (probability density function) of the distribution is shown in Figure 6.

The gap between the PDF of sim_{corr} and sim_{self} revealed the reason that we found no false. Specifically, the similarity between different phones spans in a range which has no common part with what of similarity between the same phones. Generally speaking, the maximum value of the sim_{corr} is less than the minimum value of sim_{self}. So, facing a newly arrived feature vector, the similarity between it and its' nearest neighbor is calculated. It can be concluded that they comes from the same device if only this similarity locates at the right side of the gap. Otherwise, the feature comes from an unknown device.

Because the error rate of the scheme is directly linked with the probability distribution over the gap, however, under this setting, the probability of feature's crossing the gap is unknown resulted from lacking of such observation, we shift to get an analytical description of the PDF.

5.3.2 Distribution Fitness

We inspected the two distributions to find their proper distribution type respectively, and found that both of them are unsymmetrical shaped, so we traversed all the common distribution to find the type that fits the observations well. Totally 20 continuous distribution types were tested. After analyzing the fitness, we found that the 2 types of distance derived from observations to feature vectors from either the same phones or different phone pairs fall into Lognormal Distribution well. The fitted distribution is shown in Figure. 7.

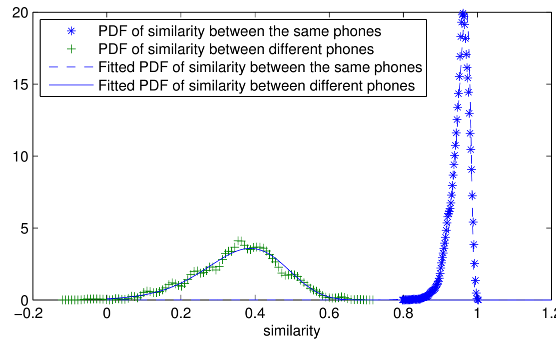

Figure 7: Fitted Distribution of Similarities.

Because distance falls in Lognormal distribution, the similarity, which is 1 - distance, falls in the distribution with the following PDF:

$$f_{self} = \frac{1}{(1 - sim_{self})\sigma\sqrt{2\pi}} e^{-(\ln(1-sim_{self})-\mu)^2/2\sigma^2}$$

Where the fitted parameter gives $\mu = -3.17698, \sigma = 0.546804$.

$$f_{corr} = \frac{1}{(1 - sim_{corr})\sigma\sqrt{2\pi}} e^{-(\ln(1-sim_{corr})-\mu)^2/2\sigma^2}$$

Where the fitted parameter gives $\mu = -0.457726, \sigma = 0.178714$.

5.3.3 Scale

We proved that the distribution can be applied to the large scale case. Doubt may be casted on the assumption that the distribution may be correlated with the quantity of the phones. We argue that the distribution of sim_{corr} changes little with the increasing of device quantity, which implies that the error rate of the scheme doesn't increase when the quantity of the devices increases. Changes of parameters μ and σ according different quantity of devices are shown in Figure. 8.

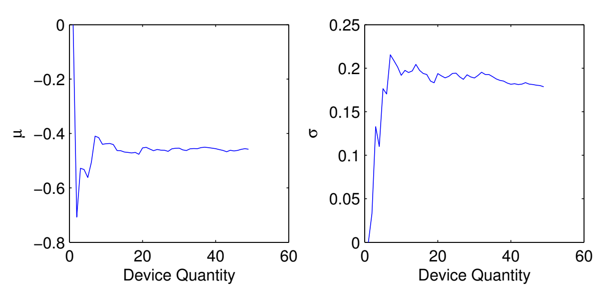

Figure 8: Parameter vs Device Quantity.

As we can see, the parameters converge to constants when the quantity increases. Based on the result, we assume that the model is suited for large scale similarity representation.

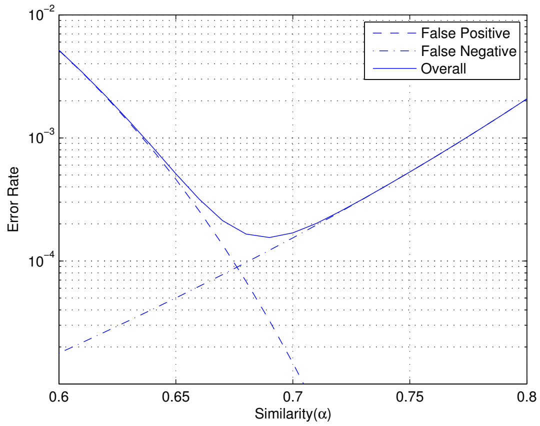

Figure 9: Error Rate vs Similarity (α).

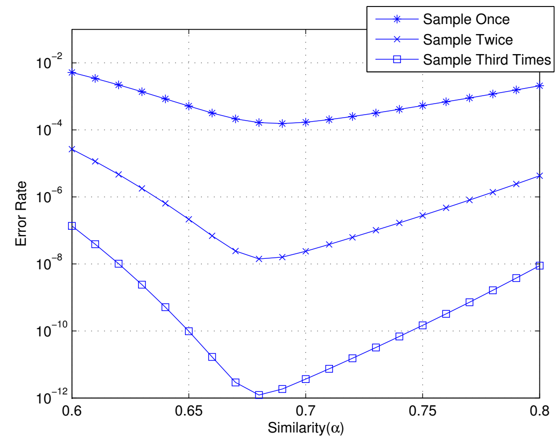

Figure 10: Error Rate vs Similarity (α).

5.3.4 *Error Rate Analysis*

We give the theoretical analysis to the error rate based on the model deducted from the prior part. We analyze the false positive rate and false negative respectively first. We then calculate their sum and analyze the error rate under multiple sampling time case. At last, we show the scheme operator that the parameter can be tuned to satisfy the cookie substitution case.

False Positive Theoretically, if an alien observation to sim_{corr} crossed the gap and fell into the range occupied by the sim_{self}, it may be regarded as being produced by some device already in the database. The probability of this case is $1 - F_{corr}(\alpha)$, where α is the threshold set by server. Curve false positive in Figure. 9 shows the relationship between α and error rate.

There is another case, which also leads to false positive. Feature vector produced by Alice may have a sim_{corr} with Bob's that is higher than sim_{self} of Alice's, which leads server to output Bob. The probability of this case is $\int_{\alpha}^{1} f_{corr}(x)F_{self}(x)dx$, which is preeminently less than $1 - F_{corr}(\alpha)$. As the result, it is neglected when calculating the error rate.

False Negative An observation to sim_{self} may fall into the range belonging to sim_{corr}, which misleads the server to output null instead of the right answer. The probability of this case is $F_{self}(\alpha)$, as it is shown in Curve False Negative of Figure. 9.

Overall Error Rate The error of the scheme is defined by the sum of false positive and false negative. The error rate is calculated by the sum of the two kinds of error rate thereby. It changes according to α, which is shown in Figure 9. The figure tells that lower α brings to more false positive while higher α leads to more false negative. The valley point of the curve locates at 0.69, which implies that setting threshold to 0.69 gives the the best performance.

As we can see, the error rate is around $1.55*10^{-4}$, when the threshold is set at 0.69.

Performance Enhancement Sampling multiple times elevates the performance sharply. Collecting each feature vector costs only little, and noises are regarded as independent, which therefore inspired us to collect feature more than once to decrease the error rate. For example, if we collect 2 samples each time, the error rate decreases sharply because the false positive happens only if both two samples are false positive sample, and the false negative happens only if both two samples are false negative sample. Figure. 10 shows that the error rate of the twice scheme is around $1.41*10^{-8}$, when the threshold is set at 0.68. Hence, $1.23*10^{-12}$ error rate can be achieved if 3 times sampling is adopted.

Biased Case The threshold parameter can be tuned to satisfy different cases. For instance, as the substitution to cookie, the consequences brought by false positive and false negative is not equal. Specifically, clearing cookies often leads to the result that an old user is mistaken as a new comer, which is similar to the false negative, while a piece of cookie will seldom be judged wrongly as other's, which is similar to false positive. As the result, servers' tolerance to false negative is much higher than that of false positive. To this end, the threshold α can be elevated to trade the performance of false negative for the performance of false positive.

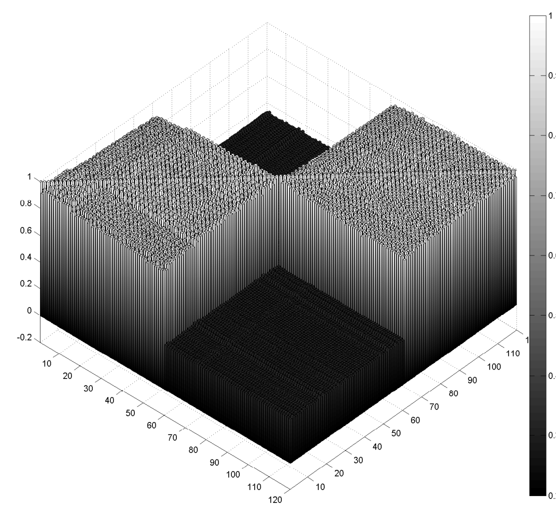

Figure 11: Correlated Similarity.

5.4 Stability

We argue that the higher frequency response feature is a kind of long-term stable and unchangeable feature. To be a kind of identity, the feature should be stable spanning a range of time. However in the case of cookie, different people clear their cookies with different time gaps. Some people never clear their cookies while some never save them, which casts doubt on the stability of the cookie

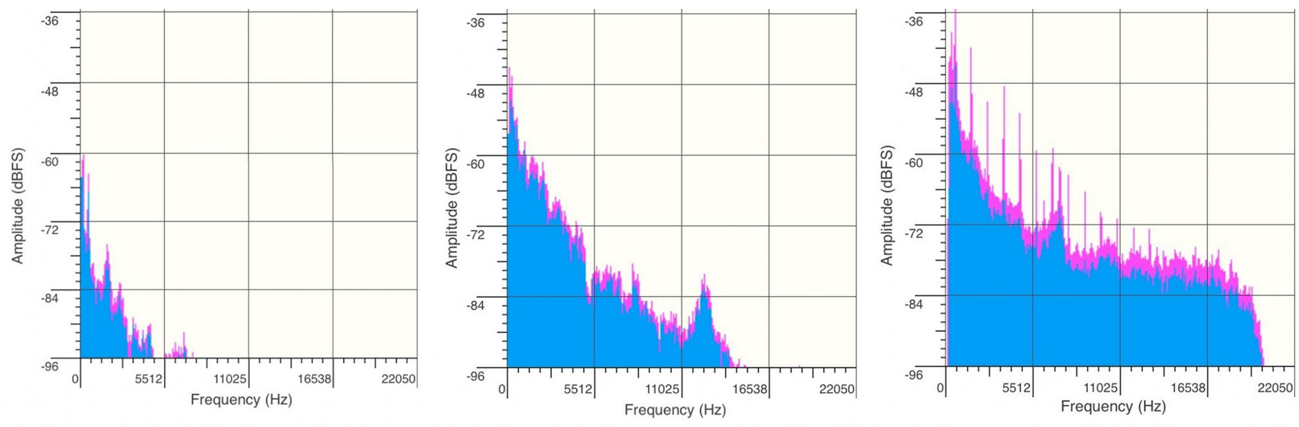

Figure 12: Noise in Office, Street and Metro.

as a kind of identity. To prove our scheme's stability, we chose 2 speakers randomly and collected feature vector every 1 hour to each emulated phone. As the result, we have collected 60 feature vectors to each phone totally. The vectors produced by the first phone are labeled from 1 to 60, while the vectors produced by the second one are labeled form 61 to 120. Figure. 11 shows the similarity between the 120 vectors.

As concluded from Figure. 11, there is no obvious decreases in similarity between feature vectors within the same phone collected from the first hour to the last hour. Also, we haven't observed obvious increase in similarity between the two phones from hour to hour, so the experiment concludes that the higher frequency response feature is long-term stable.

5.5 Interference

The higher frequency response is affected by the noise in the higher frequency range, which is pure and silent in most cases. In order to prove the ability to anti interference, we have tested the scheme in different environment with different noise, ranging from office, street, metro station. The result is positive in all cases except the metro station. Figure. 12 shows the spectrum of noise on the air in the 3 environments. We will present both qualitative and quantitative analysis to the anti-interference ability of the scheme.

Qualitative Analysis In the effective frequency range, 14kHz to 21kHz, the environment is silent in the case of office and street, though there are loud human being's voices and other noises, which don't locate at the effective band, so the response feature can be calculated with only little interference. In the case of metro station, the noise spans all the sampling frequency range including the effective band, which overwhelms the signal broadcasted and makes the calculation result meaningless. Thus, we concluded that the scheme works if only the high frequency band is silent.

Quantitative Analysis In this section, we try to find out the highest noise level at which the scheme works. To simplify the problem, we reasonably assume that the feature is absolutely stable and all the distance between the features of the same phone is resulted from the interferences in the environment. The sampled spectrum of signal is denoted as $\vec{X}$, while the noise is denoted as $\vec{N}$. We also assume that there is little correlation between $\vec{X}$ and $\vec{N}$, and the expected mean of the $\vec{N}$ is zero (We assume like this because the noise is often white noise), which leads $\vec{X}$ and $\vec{N}$ to be regarded orthogonal and $\vec{X} \cdot \vec{N} = 0$ thereby. The Similarity calculated in fact is:

$$
\begin{aligned}
& 1 - \sqrt{\left(\frac{\vec{X}}{|\vec{X}|} - \frac{\vec{X}+\vec{N}}{|\vec{X}+\vec{N}|}\right)^2} \\
&= 1 - \sqrt{2 - 2\frac{\vec{X}(\vec{X}+\vec{N})}{|\vec{X}||\vec{X}+\vec{N}|}} \\
&= 1 - \sqrt{2 - 2\frac{|\vec{X}|}{|\vec{X}+\vec{N}|}}
\end{aligned}
$$

We consider the false negative in the interfered environment while neglect the case of false positive, because noise can easily make a feature distorted, but hardly make a feature similar to another. The server outputs right answer when this similarity between the 2 feature vectors is higher than a threshold α. Thus:

$$
\begin{aligned}
& 1 - \sqrt{2 - 2\frac{|\vec{X}|}{|\vec{X}+\vec{N}|}} > \alpha \\
& \Rightarrow \frac{|\vec{X}|}{|\vec{X}+\vec{N}|} > \frac{1+2\alpha-\alpha^2}{2} \\
& \Rightarrow \frac{|\vec{X}|^2}{|\vec{X}|^2+|\vec{N}|^2} > \frac{\alpha^4-4\alpha^3+2\alpha^2+4\alpha+1}{4} \\
& \Rightarrow \frac{SNR}{SNR+1} > \frac{\alpha^4-4\alpha^3+2\alpha^2+4\alpha+1}{4} \\
& \Rightarrow SNR > \frac{1+4\alpha+2\alpha^2-4\alpha^3+\alpha^4}{3-4\alpha-2\alpha^2+4\alpha^3-\alpha^4}
\end{aligned}
$$

Where SNR is $\frac{|\vec{X}|^2}{|\vec{N}|^2}$

The SNR can be calculated in this way according to Parseval's theorem, which indicates that the power of a signal can also be the sum of each frequency component's power, while the power of each component is the square of its amplitude. Therefore, the square to the normal of the vector is just the power.

The relationship between α and error rate, thus Figure. 13, shows the SNR requirement in avoiding false positive in different α setting. As we can see, in the normal setting, thus $\alpha = 0.7$, the SNR requirement is 10 dB. That means the scheme outputs right answer if only the SNR in the effective frequency band is higher than 10 dB. Don't forget that the noise power is only counted for those lo-

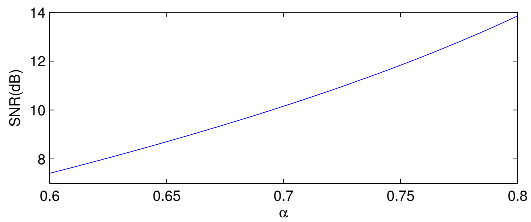

Figure 13: SNR requirement over different α.

cated at the effective frequency points, which possesses only little of the overall noise power.

5.6 Entropy

We calculate entropy in this part, because entropy is important in evaluating an identity scheme. Entropy weights how many information the identity carries, and hence how many devices can be distinguished from each other. Specifically, in order to distinguish a set of devices whose size is N, at least $log_2 N$ bits entropy should be carried during a round of identification. Therefore, we analyze how many devices can be distinguished by deploying our scheme.

After setting the threshold parameter α to the optimized one, the error rate can be calculated accordingly. Thus, the entropy can be calculated if only the relationship between error rate and the distinguishable size is decided. Approximately $1/error\_rate$ devices can be distinguished at the given $error\_rate$, because less than one error will be found expectedly. As the result, we regard all the $1/error\_rate$ devices distinguishable accordingly. The identity carries $-log_2(error\_rate)$ entropy under the settings thereby.

We believe that each feature being transferred back to the server carries entropy. Therefore, with the increasing of feature vectors used linearly, the error rate decreases geometrically and the entropy increases linearly, because of the independence between 2 samplings. As calculated before, the error rate is at $1.55*10^{-4}$, if one feature vector is utilized to make judgement. According to the error rate, 12.6 bits entropy can be achieved in the single sampling case.

6. APPLICATION

The device ID extracted from our proposed scheme can not only replace traditional cookies, but also be used beyond that, and this section will give a brief introduction to some of the applications.

6.1 Stolen phone tracing and self-destruction

Recently, lawmakers in California have approved a bill, which requires all smart phones sold in the state to have anti-theft software installed, so that once the device being lost or stolen, it cannot be used any more, even after a hard reset [16].

However, in order to achieve that goal, the first step is to uniquely identify a device. This is not a trivial task, given the fact that the phone could be reset, re-flashed with different operating system image, or the IMEI code can even be modified via software. In other words, since every piece of current device information is stored in Flash memory, and the Flash memory is under the control of adversaries, nothing can prevent them from modifying such information and defeat the "kill switch" mechanism.

Our speaker-based device ID can help address the challenge. Any changes in the software cannot change our hardware-based device IDs. So, in order to check if current phone has been reported as stolen, the system vendors only need to perform a quick and un-noticeable test, and then look up the generated device ID in the stolen phone's database.

To avoid such detection, the adversaries have to modify the hardware. However the cost is high, because, not only the extra money to buy new parts, but also the skills to replace the part. (especially the phones are becoming more difficult to be disassembled)

6.2 Location information broadcast and relay

Many applications require position information to complete some useful functions. For example, instant message applications can let you know and make friends with people nearby. However, current designs require users to grant the applications to access their current position, which users often decline, either due to the privacy concerns, or avoid overly power consumption used by GPS subsystem.

But with our proposed scheme, applications can easily share and relay position information, and following is a typical scenario. Suppose there were many people in a conference room, but only one of them turned on the location service, so the server could put information of device ID and the location of that conference room into a database. Now the application will periodically play the specially crafted sound, which can be captured and cross-fingerprinted by other phones nearby. Once the device ID is extracted, those smartphones will query the database on the server, and retrieve the location information generated by another phone with GPS turned on. Once a new phone get its location information, it starts to broadcast its identity, and thus the location information can be relayed across the whole conference room.

6.3 Indoor tracking

Indoor tracking has a huge market potential, with which supermarket and department store can send coupons and targeted advertisements to their customers. There are already several technologies available, like Bluetooth based iBeacon from Apple [7], and WiFi based solutions [12]. The device ID proposed in this paper can also be used in this scenario. First, whenever the user enters a supermarket, her phone will receive a signal to trigger the inaudible sound playing periodically, which is actually equal to broadcast its device ID from time to time. Such broadcasting will be received by microphones deployed all around the supermarket, then a cross-fingerprinting is performed, and a unique device ID extracted. By correlate the device ID with the microphone location, it is easy to know the route of the user in the supermarket, what her favorite is, and what is still under consideration, etc. With the same technology, it is also possible to associate the purchase history to a specific device ID, simply by putting a cross-fingerprinting microphone near the check-out counter.

7. DISCUSSION

In this section we will discuss the defense methods and limitations of our proposed method.

7.1 Defense

We postulate some practical methods to help users defend this kind of tracking, though we implemented none of them.

Speaker Usage Notification. An indication could be added to warn users that the loudspeaker is working. If the user noticed the indication but heard nothing, there should be someone invoking tracking. The indication could be an icon displayed in the notification bar. It could also be some flash pattern of the LED. The principle is similar to a light near the embedded camera in the laptop indicating that the user's camera is working and privacy may be stolen if the camera is not opened by user himself.

Higher Frequency Blocking. The audio driver developers could cut the non-sensitive sound directly in the speaker mode. In headphone mode, people may feel the change of missing high frequency components, considering the noise isolation and fine device quality. However in speaker mode, where people seldom care the audio quality, adding or deleting higher frequency component has only little effect on quality, because originally, the quality of sound generated by loudspeaker is low and the lack of noise isolation. So, developers could set the cutting frequency of the digital filter to 16 kHz, then the higher frequency component is blocked by the filter and will not play, while the audio quality doesn't decrease much in speaker mode.

7.2 Incomplete Inaudibility

In this paper, the inaudibility focuses on people, especially adults. However, there are also many other individuals that are able to hear or feel the higher frequency audio. For example, infants have a better hearing that may enable them to hear the audio. So, they may cry when they are exposed to the environment filling with the annoying audio. Dog, a kind of creature with much better hearing than human-being, may behave weirdly when it hears the audio.

7.3 Limitation of Stability Experiment

We will indicate the limitations of our work as follow.

Long-term slow changes The duration of our experiment is 60 hours. We haven't kept the experiment for months or years, while a phone usually can be used for years. The phone may experience changes in terms of climate, aging, etc, which may offset the feature slowly. For example, the feature may be different in different air humidity, because the vapor in the air may influence the vibration of coil. We want to compensate this drawback by postulating a fingerprint slow updating technique. We update the fingerprint in the database if the fingerprint can be distinguished but a small constant offset detected, such that the slow changes can be compensated.

Upheavals The stability experiment was conducted in office only. But we know that a phone may encounters range of situations, which may change the feature of the phone rapidly. For example, a sudden dropping to floor may change the mechanical feature of the speaker and thereby change the frequency response. An accidently dropping to water may change the feature too. We haven't come up with practical solution to identify such an upheaval.

7.4 Interference from background noises

Although our proposed scheme has a special design on frequency combinations at about seventy discrete frequency points, it could still fail to extract unique device IDs under environments saturated with high power noise signals, like train station, crowding restaurant, etc.

To overcome such limitation and make our method work even under low Signal-Noise-Ratio (SNR), we may try to use some advanced methods borrowing from communication area, and one example is "Spread-spectrum Communication" [9]. Spread spectrum communication generally makes use of a sequential noise-like signal structure to spread the normally narrow band information signal over a relatively wide band of frequencies. It can even do frequency-hopping where information is sent following a sequence of pseudo random frequencies. The receiver can reproduce the same pseudo random sequence ,thus it is able to correlate the received signals to retrieve the transmitted information [9].

Inspired by the spread-spectrum communication, we can modify the scheme accordingly. In each effective frequency point from 14 kHz to 21 kHz, the original mono tone sine wave is modulated with a pseudo random sequence, such that the energy originally in the frequency point spreads to a frequency range, the width of which is decided by the rate of the pseudo random sequence generation. As the result, the distributed energy decreases the energy density sharply while the overall signal energy keeps unchanged, since the consumed bandwidth increases. Later, the recorded audio data will be sent to a band pass filter and de-spread to recover the sine wave. Finally, the recovered sine wave at each frequency point has different amplitude because the speaker attenuates the signals, which reflects the features of that speaker.

7.5 Device ID for smart phones of different models or from different manufacturers

In this paper, we only evaluated the features of 50 OEM speakers for Samsung Galaxy S3. All the speakers are coming from the same assembly line with continuous Serial Number printed on them. We did not extend our study to smart phones from different manufacturers because of the assumption that speakers from different manufacturers are generally easier to be differentiated, which has been confirmed by previous work [19].

Even in the worst case that above assumption fails, we would propose to incorporate other hardware feature or information into the device ID. For example, the CPU type, memory capacity, operating system version, etc. According to previous studies, an app can get all above information without requesting any special permission [45].

7.6 Detection of audio fingerprinting operation

Although an Android application based on our proposed scheme can disguise itself as a legitimate one by requesting microphone accessing permission for other legal use, it is still possible to detect if such application is trying to perform audio fingerprinting or not. For example, it is required to do Fast Fourier Transform on recorded response in order to generate audio stimulus, so with some code analysis, it is possible to detect the existence of such suspicious operations though it can be hidden into the equalizer processing as if it is enhancing the audio quality. However, if an application's original function include FFT operation, then the detection problem is still difficult to handle.

8. RELATED WORK

Software Fingerprint In terms of software feature, many browser configuration information can be exploited to differentiate devices, such as User Agent, fonts installed, plugin information, benchmark etc [22, 33, 14, 40]. Besides the browser, OS version, Kernel version and application list can all be utilized to distinguish devices. Different implementation to the networking protocol can also be exploited to generate fingerprint, such as TCP initial window size, IP header ID sequence generation [25, 38, 26].

Hardware Fingerprint In terms of hardware feature, a lot of works have been devoted to identifying the devices by exploiting minute differences of the signal produced by the component of the phone. For example, wireless NIC can be distinguished by exploiting feature from RF signal emitted by the transmitter [42, 23, 20, 13, 15, 17]. However they cannot be promoted to Internet tracking usage, since there may be no direct physical link between user and tracer. Data collected from the accelerometer can also be used to distinguish users in [21] with coarse precision without active stimulation. Photos taken by cameras can also be distinguished by pattern and noise [31].

Scheme proposed in [19] also leveraged feature of speaker embedded in the phone to identify users. However, they haven't pointed out how large scale their scheme can be applied to. Besides, the robustness of the cepstral feature is not evaluated, which casts doubt on the feasibility of long-term tracking. What's more, no practical method has been postulated in his scheme in terms of hiding the identification process, while playing a clip of audible music as stimulation will inevitably attract users' attention.

Location Stealing Many researchers have also focused on position stealing method in android devices without corresponding permission. Zhou et al. have studied how to infer the location with public information provided by android without special permission in [45]. Han et al. postulated that accelerometers in smartphones can be utilized to infer location in [27]. Lester et al stated in [30] that techniques have been found to determine if two phones are being carried by the same person. In [34], the author have raised a kind of probabilistic method for positioning to mobile devices in the pocket without GPS information.

9. CONCLUSION

This paper exhibited that there are differences between speaker individuals of the smart phone, which is reflected on the differences between the response curves. It is the differences that enable applications to generate unique identity according to response curve. The identity proves to be eligible as a kind of long-term tracking proof, for its' stability. The identity also proves to be entropy sufficient to incorporate all the phones in the world. In terms of anti-interference, both practical experiments and theoretical analysis are conducted to show that the scheme works in common situations except what with annoying high power noise. Besides the identity, more seriously, the location of the device may be exposed, resulted from the narrow broadcasting range of the sound wave. To calculate the error rate, we analyzed the distribution model of the similarity, which is calculated by fitting the similarity between identities to some probabilistic model and choosing the most overlapped one. We decide the entropy according to the size of the distinguishable device pool calculated by the error rate.

10. ACKNOWLEDGEMENTS

We thank our shepherd Christina Pöpper for her valuable time and guidance on the preparation of the final version, and anonymous reviewers for their comments on the draft of the paper. We also want to thank Zhou Li from RSA Laboratories for his insightful suggestions.

11. REFERENCES

[1] Android 4.4 blocked some SD card accessing. http://lifehacker.com/android-kitkat-blocks-some-access-to-micro-sd-cards-1524997895.

[2] Anonymity, Privacy, and Security Online. http://www.pewinternet.org/2013/09/05/anonymity-privacy-and-security-online/.

[3] Apple set deadline for UDID. http://www.engadget.com/2013/03/21/apple-sets-a-may-1st-cutoff-for-app-submissions-that-use-udid/.

[4] F-secure Mobile Treat Report. https://www.f-secure.com/static/doc/labs_global/Research/Mobile_Threat_Report_Q1_2013.pdf.

[5] Frequency response of 3 speakers. http://community.edmdistrict.com/showthread.php?7944-Beats-Studios-doesn-t-sound-right.

[6] Google may stop tracking user by cookie. http://www.usatoday.com/story/tech/2013/09/17/google-cookies-advertising/2823183/.

[7] iBeacon. http://en.wikipedia.org/wiki/IBeacon.

[8] IDC Worldwide Smartphone Shipments. https://www.idc.com/getdoc.jsp?containerId=prUS24645514.

[9] Spread spectrum. http://en.wikipedia.org/wiki/Spread_spectrum.

[10] UIUC Physics 406 Acoustical Physics of Music. http://courses.physics.illinois.edu/phys406/Lecture_Notes/P406POM_Lecture_Notes/P406POM_Lect5.pdf.

[11] UUID of Android. http://developer.android.com/reference/java/util/UUID.html.

[12] Wifi Positioning System. http://en.wikipedia.org/wiki/Wi-Fi_positioning_system.

[13] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz. On the reliability of wireless fingerprinting using clock skews. In *Proceedings of the third ACM conference on Wireless network security*, pages 169–174. ACM, 2010.

[14] K. Boda, Á. M. Földes, G. G. Gulyás, and S. Imre. User tracking on the web via cross-browser fingerprinting. In *Information Security Technology for Applications*, pages 31–46. Springer, 2012.

[15] K. Bonne Rasmussen and S. Capkun. 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*, pages 331–340. IEEE, 2007.

[16] A. Boxall. Smartphone kill switch bill passes in california. http://www.digitaltrends.com/mobile/smartphone-kill-switch-bill-passes-in-california/, May 2014.

[17] 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*, pages 116–127. ACM, 2008.

[18] S. Brown. *Linear and Nonlinear Loudspeaker Characterization*. PhD thesis, WORCESTER POLYTECHNIC INSTITUTE, 2006.

[19] A. Das, N. Borisov, and M. Caesar. Fingerprinting smart devices through embedded acoustic components. *arXiv preprint arXiv:1403.3366*, 2014.

[20] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee. Identifying unique devices through wireless fingerprinting. In *Proceedings of the first ACM conference on Wireless network security*, pages 46–55. ACM, 2008.

[21] 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 (Feb 2014), NDSS*, volume 14, 2014.

[22] P. Eckersley. How unique is your web browser? In *Privacy Enhancing Technologies*, pages 1–18. Springer, 2010.

[23] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker. Passive data link layer 802.11 wireless device

driver fingerprinting. In *Proc. 15th USENIX Security Symposium*, pages 167–178, 2006.

[24] A. Gionis, P. Indyk, R. Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999.

[25] L. G. Greenwald and T. J. Thomas. Toward undetected operating system fingerprinting. In *Proceedings of the first USENIX workshop on Offensive Technologies*, pages 1–10. USENIX Association, 2007.

[26] L. G. Greenwald and T. J. Thomas. Understanding and preventing network device fingerprinting. *Bell Labs Technical Journal*, 12(3):149–166, 2007.

[27] J. Han, E. Owusu, L. T. Nguyen, A. Perrig, and J. Zhang. Accomplice: Location inference using accelerometers on smartphones. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, pages 1–9. IEEE, 2012.

[28] B. Krishnamurthy and C. Wills. Privacy diffusion on the web: a longitudinal perspective. In *Proceedings of the 18th international conference on World wide web*, pages 541–550. ACM, 2009.

[29] B. Krishnamurthy and C. E. Wills. Generating a privacy footprint on the internet. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 65–70. ACM, 2006.

[30] J. Lester, B. Hannaford, and G. Borriello. "are you with me?" - using accelerometers to determine if two devices are carried by the same person. In *Pervasive*, pages 33–50, 2004.

[31] J. Lukas, J. Fridrich, and M. Goljan. Digital camera identification from sensor pattern noise. *Information Forensics and Security, IEEE Transactions on*, 1(2):205–214, 2006.

[32] J. R. Mayer and J. C. Mitchell. Third-party web tracking: Policy and technology. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 413–427. IEEE, 2012.

[33] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham. Fingerprinting information in javascript implementations. In *Proceedings of Web*, volume 2, 2011.

[34] L. T. Nguyen and Y. Zhang. Probabilistic infrastructureless positioning in the pocket. In *Mobile Computing, Applications, and Services*, pages 311–330. Springer, 2012.

[35] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 541–555. IEEE, 2013.

[36] F. Roesner. Detecung and defending against third–party tracking on the web. 2012.

[37] M. Slaney and M. Casey. Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *Signal Processing Magazine, IEEE*, 25(2):128–131, 2008.

[38] M. Smart, G. R. Malan, and F. Jahanian. Defeating tcp/ip stack fingerprinting. In *Proceedings of the 9th USENIX Security Symposium*, volume 6, 2000.

[39] A. Soltani, S. Canty, Q. Mayo, L. Thomas, and C. J. Hoofnagle. Flash cookies and privacy. In *AAAI Spring Symposium: Intelligent Information Privacy Management*, 2010.

[40] K. Takeda. User identification and tracking with online device fingerprints fusion. In *Security Technology (ICCST), 2012 IEEE International Carnahan Conference on*, pages 163–167. IEEE, 2012.

[41] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. Adnostic: Privacy preserving targeted advertising. In *NDSS*, 2010.

[42] O. Ureten and N. Serinken. Wireless security through rf fingerprinting. *Electrical and Computer Engineering, Canadian Journal of*, 32(1):27–33, 2007.

[43] T.-E. Wei, A. B. Jeng, H.-M. Lee, C.-H. Chen, and C.-W. Tien. Android privacy. In *Machine Learning and Cybernetics (ICMLC), 2012 International Conference on*, volume 5, pages 1830–1837. IEEE, 2012.

[44] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, pages 311–321. Society for Industrial and Applied Mathematics, 1993.

[45] X. Zhou, S. Demetriou, D. He, M. Naveed, X. Pan, X. Wang, C. A. Gunter, and K. Nahrstedt. Identity, location, disease and more: inferring your secrets from android public resources. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 1017–1028. ACM, 2013.