GyroVib: Sensor-Based Smartphone Identification

Mohammad Tahaei

Informatik IV, Universität Bonn, Germany
tahaei@cs.uni-bonn.de

Abstract

No one can doubt the popularity of mobile phones. As these devices get more popular, protecting people's privacy gets more important. Apple removed unique device ID since iOS7, but it is still possible to find a way to identify uniquely mobile phones. In this paper, we show that a combination of vibrations and sensor readings from gyroscope could identify mobile phones within lab conditions. We extract features from gyroscope readings and use classification methods to identify each phone with 87.5% precision.

1 Introduction

Apple Store had more than 1.4 million apps in 2015 [4], and mobile advertisement is a primary source of making app market a success [16]. Moreover, targeted ads bring higher revenues [16]; hence it is interesting to track users and follow the habits of users to offer them more relevant ads. One of the classical methods to do such tracking is cookies. Websites widely use cookies. Furthermore, with the law of privacy protection by US and Europe [1], the user should give the permission to the website to upload cookie to her device. Due to privacy policy and protection of users, Apple removed access to unique device identifier (UDID) for developers [2], instead it offers two temporary identifiers which are advertisingIdentifier and identifierForVendor.

The main contribution of this paper is to show that it is still possible to generate unique identifiers with a combination of sensors without using Apple official SDK, or access to cookies. We choose to do experiments with gyroscope and vibrator (which to the best of our knowledge has not been investigated before), we vibrate the phone and measure its gyroscope values at the same time. We use eight iPhones to run our experiment. To find out similar samples and same devices, we use data mining methods. GyroVib uses the k-nearest neighbor algorithm to classify devices. The results are promising; it is possible to identify each phone with 87.5% precision.

The rest will be as follows, in Section 2, we give an overview of fingerprinting, then Section 3 provides an overview of previous works. Section 4, defines the attack model and in Section 5 we explain our method. Section 6 will show the results, and in Section 7 we talk about limitations and future works. We end this paper with conclusions at Section 8.

2 Background

In this Section, we give an overview of fingerprinting, what is gyroscope and how it works in iPhone, and which iOS identifiers are available through iOS SDK.

2.1 Fingerprinting

Fingerprinting is one of the most common ways to identify people. It is extensively employed in biometric and forensic sciences. It is interesting to people, governments, and scientists to

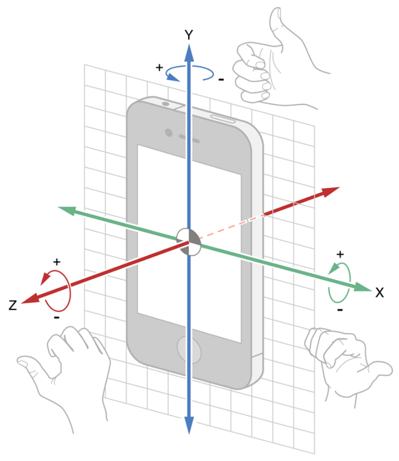

Figure 1: The gyroscope measures rotation around the x, y, and z axes.

find more ways to identify people uniquely, so the research to see how we can fingerprint people is still an ongoing topic.

These fingerprints are not only limited to the human being but also could apply to hardware and software [12, 11, 13, 18, 15, 9]. Two related fingerprints are browser and sensor fingerprinting. Cookies mostly do browser fingerprinting; this helps vendors track a user within different websites. Sensor-based fingerprinting dependents on hardware imperfections, i.e. each hardware has its characteristics also, biases, so it is possible to define such fingerprints for hardware.

2.2 Gyroscope

Apple uses accelerometer and gyroscope to detect events like movements, shakes, or tilting of the device. Accelerometer measures changes in velocity and "The gyroscope measures the rate of rotation around the three axes" [6]. The former is for changes in movement and device orientation, and the latter is for the speed of rotation (Figure 1). Both sensors measure the data in three axis-x, y, and z.

For each gyroscope update, Core Motion of iOS SDK returns a rotation rate in CMGyroData object (x, y, z). To start reading gyroscope, we call startGyroUpdates, this updates gyroData of CMMotionManager with the latest gyroscope activity. Then we assign an update interval to gyroUpdateInterval property (maximum interval 100 times per second for iPhone). Finally, startGyroUpdatesToQueue:withHandler: method will be called after each update of gyroscope based on specified interval.

2.3 iOS identifiers

As discussed in Section 2.A, fingerprinting is meaningful with hardware and software methods. Currently, Apple provides two software-based identifiers and used to provide a hardware-based identifier till iOS7. These identifiers are:

- advertisingIdentifier: "An alphanumeric string unique to each device, used only for serving advertisements." [3], this value is equal for all vendors, but if the user erases the device this will change. Moreover, the user can reset her advertisingIdentifier from iPhone setting.

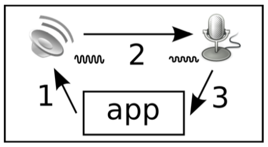

Figure 2: Diagram of sound feedback analysis. Within the confines of the device, the application emits sound via the built-in speaker (1), the sound reaches the microphone in a distorted and attenuated form (2), and the application records the microphone signal and analyzes it (3).

- identifierForVendor: "An alphanumeric string that uniquely identifies a device to the app's vendor." [7], this value is the same for all apps from the same supplier. This value resets if the user deletes all apps from the vendor.

- UDID: a unique device identifier, deprecated since iOS7.

3 Previous Work

In this Section, we give an overview of current research state on mobile sensor fingerprinting.

3.1 Dey et al.

AccelPrint [13] uses the accelerometer to fingerprint devices. The authors vibrate the phone for 10 seconds and measure its accelerometer sensor at the same time. The sensor generates three values for each sample (x, y, z). Next, they get the root of the sum of squares for these values. Since each device has its specific sensor, all data sets are not equally-spaced. Thus they apply a cubic spline interpolation to produce new data sets which are equally-spaced.

To find a proper set of features, they use LibXtract to extract features. First, they find 80 features and at the end come up with 36 most significant features. Finally, for the classification, AccelPrint uses Bagged Decision Tree for ensemble learning. Results for 107 phones/chips/tablets, are: precision and recall 99%.

3.2 Bojinov et al.

In [9], the authors play a specific audio signal through phone's speaker and record it with microphone (Figure 2). Then they divide record intensity with the original intensity (feedback ratio). This ratio is measured at many different frequencies. By computing Fourier coefficients (first two harmonics), they extract two floating-points numbers. With this method, they achieve ˜95% of correct identification with 17 devices using KNN classification.

The authors also introduce a second method with the accelerometer. They measure the accelerometer while the phone is in the resting position, i.e. while it is on the Table, meaning that the velocity has a constant value (no acceleration). This method requires user's collaboration, and she should flip the phone (since they are gathering data for phone faced up and faced down). Results are given for 3583 devices with 15.1% correct identification.

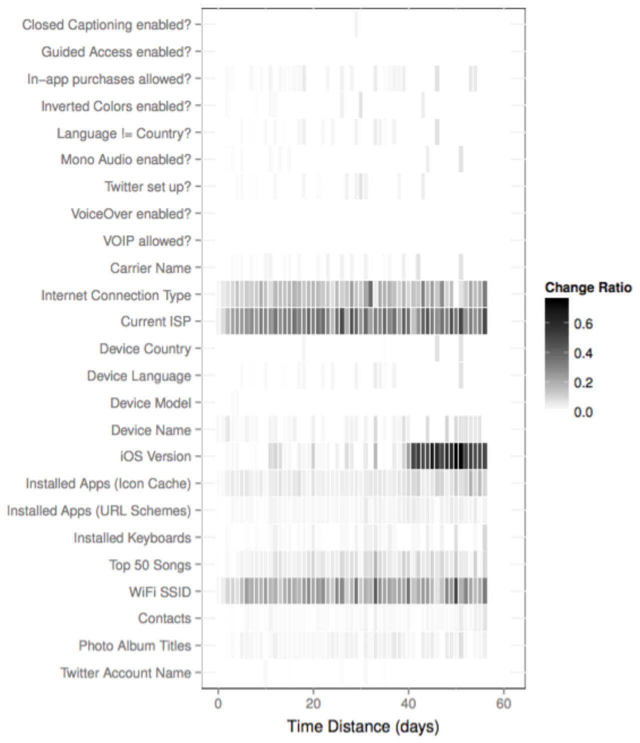

Figure 3: Average Fingerprint Changes per Feature

3.3 Kurtz et al.

[15] uses personalized configurations to identify phones. They found 29 different features from iPhone SDK, which need no permission from the user. These include, for example, device names, language settings, lists of apps installed and most played songs. With these features and 13,000 fingerprints, they could identify each phone with 97% accuracy.

This research describes a software-based method to determine phones and how several personalized features stay the same during a time. In Figure 3, we can observe how a feature changes over time. For example, installed apps or top 50 songs do not alter over time that much and also are among high valued attributes.

3.4 Goethem et al.

Goethem et al. [18] use accelerometer-based device fingerprinting for multi-factor mobile authentication. Since one of the threats in cell phones is spoofing, the main concern of this paper is to find a way to improve authentication methods. It uses standard APIs in web browsers to vibrate the phone at different intervals during the registration of the user. After the registration step, if the user wants to log into the system, she is asked to give sensor data with random lengths. If these traces matches with the one from registration period, she can log in. The Authors evaluate their method with 15 phones with ten registration traces. The result for the true positive rate is 0.8000 and for false positive rate is 0.0222.

3.5 Das et al.

This method [12] uses a simple Javascript to read accelerometer and gyroscope data in Chrome browser. The authors measure values while playing an inaudible sine wave or a song. Combine features from both sensors provide 44 important features. For sampling, they have around ten samples per phone with 63 phones to test (public, inviting users to participate in the experiment). The F-score for the public experiment is ˜94%.

3.6 Zhou et al.

In this paper [20], the authors represent an acoustic fingerprinting method. In this technique, the microphone records the output from device speaker, and the speaker plays high-frequency audio with specific patterns. The manufacturing of devices results in some traces in each device which can be uniquely identified. To match devices, this method uses a simple distance metric between two vectors, if the distance is over 0.7 then it is considered as the same appliance; Otherwise, it is a new device. The experiment is not only on phones, but speakers from a cellphone vendor are also used (one phone as a host, plugging a new speaker per experiment). The method uses 50 speakers, each 60 features. The error rate of the technique is computed as the sum of false positive and false negatives, which is $1.55 * 10^{-4}$, when the threshold is 0.69.

4 Application Scenarios

4.1 Assumptions

We assume that the adversary is not able to access UDID generated by Apple since it is deprecated from iOS7 and the other two identifies which described in Section 2.C are not persistent and permanent. Following actions are open to any developer from iOS SDK:

- Vibration: it is possible to vibrate the phone for any duration. The only constraint is that the user should turn on the vibration from setting.
- Gyroscope: reading gyroscope values is open to anyone. The highest frequency open to the public is 100 times a second [6].
- Internet access: access to the web is not listed in the setting of apps in iOS [8], so we assume that is it possible to send and receive data from iOS.

4.2 Attack Models

Attack models show what the potential risks and threats to the user are if an adversary generates sensor-based fingerprints like our method. If an adversary generates fingerprints with our model, he will be able to identify the user, and also track the user's activity without her permission [5]. It is also possible to track a user on many apps, meaning that in our model each device has a unique identifier; hence it is possible to track users activity in different apps.

4.2.1 User Identification

The user may like to remain unknown to any service (website, app, company) but with our model, it is possible to relate a device with some identification information (e.g. email, phone or name). The user may like to remove all her information after deleting the app and become

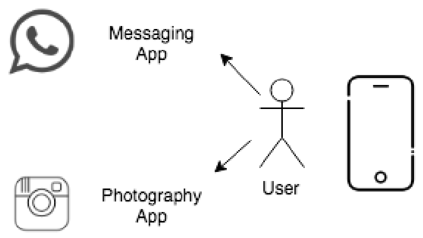

Figure 4: Cross Tracking: identifying the same user within different apps

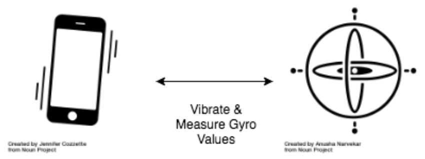

Figure 5: GyroVib: phone vibration and measuring gyroscope values

unknown to the vendor, but in our model, we can identify a user with her phone fingerprint even after deleting and reinstalling the app.

4.2.2 User Tracking

Due to the privacy protection, users can choose if they want to store their cookies on their devices or they want to opt out of this service. Therefore using such tracking service is due to the user's permission, but with our method, it is possible to track user even without her permission.

4.2.3 Cross Tracking

In iOS, it is possible to generate one ID for a company and use it across its applications. However, if the user deletes all its apps, the company should produce a new one for its newly installed apps. On the other hand, in our model, it is possible to generate a fingerprint for one device and use it within many apps, sharing information within them. This means that if the user has installed one app from a company, and deletes all her apps from this company, thinking that her data is removed, and later on she can install a fresh copy with no background information about her, is not a valid assumption.

Furthermore, when a user installs an app, she expects to share her information only with one app, but in our model, it is possible for the vendor to exchange information between apps (Figure 4).

5 Methodology

Our method is based on: generate a device fingerprint with vibrating the phone and at the same time reading its gyroscope values (Figure 5[1]). In the following, we elaborate our method.

[1]https://thenounproject.com/term/gyroscope/145002/
https://thenounproject.com/term/vibration/439978/

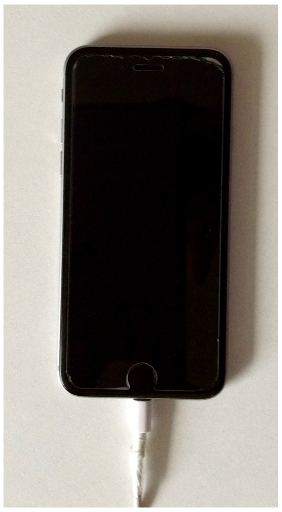

Figure 6: Phone setup on the Table

5.1 Data collection and preprocessing

We use 8 iPhones in our lab (Table 1). To run our experiment, we lay the phone on the Table, and the phone is at rest (no significant movement). There are no unusual shakings, the phone does not have any case (cover), and the table's surface is made of plastic. Each phone vibrates for ten seconds with the same pattern. The vibration pattern is iOS's default pattern (kSystemSoundID_Vibrate). During this period, the gyroscope is active and updating its values. The update interval is 100 times per second which is the highest value for iPhone.

Since each phone has its characteristics (sensors with specific bias or noise), each of them will generate a specific number of samples during the test period (not equally spaced). Though we use cubic spline interpolation [17] to create new data points which are equally spaced. By applying this method comparing data sets are more accurate and reasonable.

Each update of the gyroscope has x, y, z. To have a single value for this and also removing device orientation dependency [11], we use root sum squares (RSS):

$$S = \sqrt{x^2 + y^2 + z^2}$$

We run each experiment, five times per phone, so we in total we have 40 samples.

Model	OS	Quantity
iPhone 6s	iOS 9	4
iPhone 6s Plus	iOS 9	1
iPhone 6	iOS 9	1
iPhone 5s	iOS 9	1
iPhone 5	iOS 9	1
Total		8

Table 1: Distribution of phones

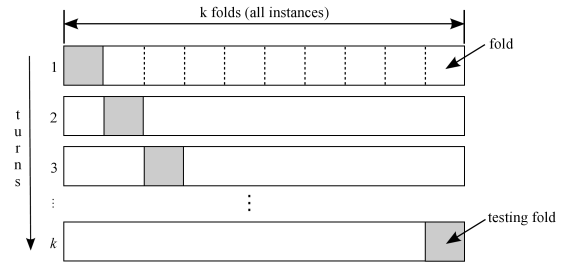

Figure 7: Cross-validation

5.2 Evaluation Metric

We use common classification metrics to evaluate our work. Precision, recall, and F-measure are three different evaluation metrics for our work. Since we do not have enough data to support separate sets for training and testing, we use n-fold cross-validation (Figure 7). In this method, a portion of data is reserved for testing, and the rest is for training. N-fold cross-validation divides the data is in ten random parts, keeping the class distribution approximately the same in all parts. In the next step, it applies the data mining algorithm n times on these parts. In each iteration, one part is for testing and the n-1 parts are for training. Eventually, after running the test n times, the method reports an overall average error. A recommended number for the number of folds is ten and we also use 10-fold cross-validation in our experiment [19].

$$recall = \frac{TP}{TP+FN}$$

$$precision = \frac{TP}{TP+FP}$$

$$F - measure = \frac{2.TP}{2.TP+FP+FN}$$

5.3 Feature Generation

One of the most important factors in every classification problem is finding a good set of features. We use LibXtract [10] to extract features from the dataset. Table 8 shows a description of features, also [13] provides exact formulas to these features. By using these features plus one feature which is the class (the owner of the phone), we have 40 samples, each with 17 attributes. We call each case a fingerprint.

5.4 Identification

Each fingerprint (sample) has a class which is its owner, so the problem is finding who is the owner of a fingerprint. This subject is a common problem for supervised learning or classification. There are many tools in the market to do classification, and one of the best ones is Weka [14], which is open source and popular in academia. We use Weka for the classification task. The method to do evaluation is 10-fold cross validation.

#	Feature	Description
1	Mean	The arithmetic mean of the signal strength at different timestamps
2	Standard Deviation	Standard deviation of the signal strength
3	Average Deviation	Average deviation from mean
4	Skewness	Measure of asymmetry about mean
5	RMS	Square root of the arithmetic mean of the squares of the signal strength at various timestamps
6	Max	Maximum signal strength
7	Min	Minimum signal strength
8	Spectral STD	A measure of the standard deviation of a signal's magnitude spectrum
9	Spectral Centroid	Represents the center of mass of a spectral power distribution
10	Spectral Skewness	Represents the coefficient of skewness of a spectrum
11	Crest	Peak to average power ratio
12	Spectral IrregularityK	Measures the degree of variation of the successive peaks of a spectrum
13	Spectral IrregularityJ	Measures the degree of variation of the successive peaks of a spectrum
14	Smoothness	Log of a partial minus the average of the log of the surrounding partials
15	Spectral Rolloff	Defines the frequency below which 85% of the distribution magnitude is concentrated
16	Spectral Flatness	Measures how energy is spread across the spectrum

Figure 8: Extracted features

6 Results

To find the suitable classification method, we apply several Weka classifiers and reporting good ones with over 80% of correctly classified instances (Table 9). Methods we choose come from previous works mentioned in Section 3 and also common classification algorithms. From the results, we can see that the top result is for IBk method (Weka implementation of K-nearest neighbors classifier) with 87.5% of correctly classified instances. KNN is an instance-based learning. In this algorithm, the class of new instances is determined by the class of the neighbor instance. It is possible to have many neighbors, and most common class of nearest k neighbors is assigned to the new instance [19].

To see how the number of instances per phone influences the results, we run IBk method on 2, 3, 4 and five samples per phone. Figure 10 shows the results, and we can observe that with five samples per phone we get the best results.

Each feature has some influence on the result, but some have more impact; hence Table 13 shows the ranking of attributes using Weka "Information Gain Ranking " algorithm.

Table 4 shows confusion matrix. In this matrix, we can see how iPhone models influence the results. We can assume that different models are more distinguishable, but this assumption needs further research and more samples from various models. Figure 11 shows how the highest info gain feature (highestValue) values are distributed. We can see that the values are close to each other in the same phone and this makes it easy for the classifier to predict the class.

7 Limitations and Discussion

There are some questions left which are interesting to answer and could be topics for further research:

- Would this method work on public and a huge number of phones? Since we test the

Classifier	Precision	Recall	F-Measure	Correctly Classified Instances (%)	Incorrectly Classified Instances (%)
NaiveBayes	0.818	0.825	0.815	82.5	17.5
KStar	0.842	0.842	0.821	82.5	17.5
PART	0.827	0.825	0.816	82.5	17.5
Bagging (REPTree)	0.886	0.850	0.829	85	15
J48	0.856	0.850	0.846	85	15
IBk	**0.875**	**0.875**	**0.867**	**87.5**	**12.5**

Figure 9: Results of running several Weka methods on GyroVib data

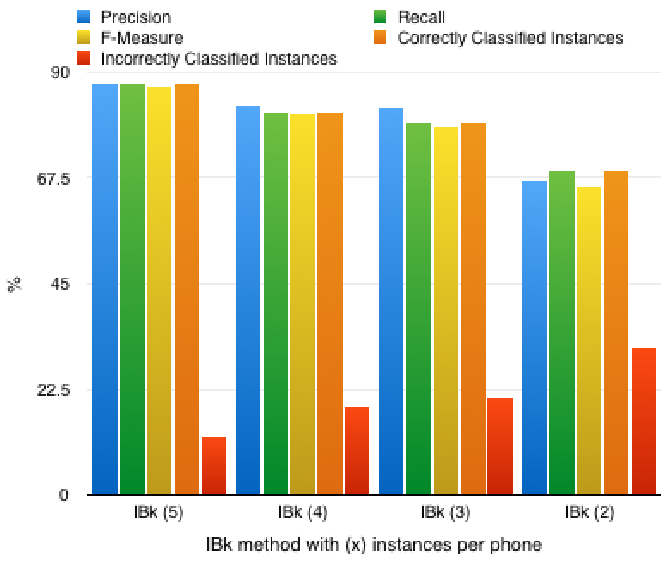

Figure 10: Applying IBk method on 5, 4, 3 and 2 samples per phone

method on eight iPhones, it is still an open question to see if the method is working properly on many different phones.

- Would this approach work properly on Android, Windows, Blackberry? we tested our method on iOS, which is a fraction of smartphone's market, the method should be tested on several different operating systems.
- Does different surfaces and positions of the phone have an effect on the results? Our testing surface is phone lying on the Table. However, the question of surface and position effect is still open.
- What are the effects of features on the results? We test an attribute set of size 16 which come from similar research [13] in the area, although it is interesting to test a huge set of features and see how they modify the result. Moreover, finding a small set of features which contains the most valuable ones is also interesting.
- Combining hardware (sensor)-based methods with software-based methods: in other research papers there are methods to find fingerprints only with personalized software con-

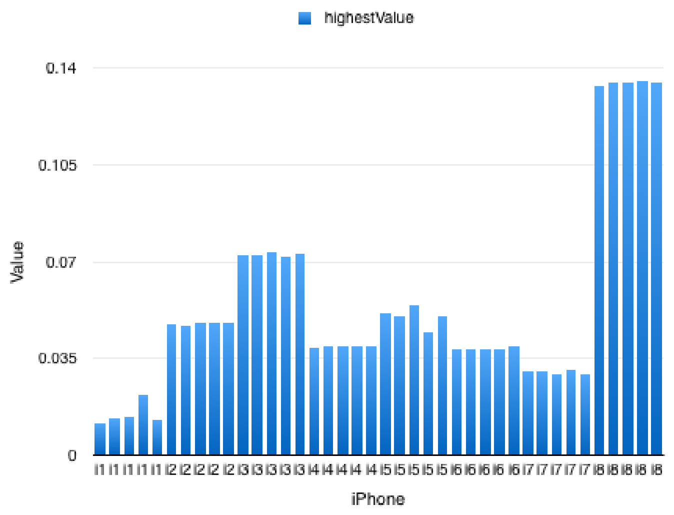

Figure 11: Values of highest info gain feature (highestValue)

a	b	c	d	e	f	g	h	¡- classified as
5	0	0	0	0	0	0	0	a = i1 (5S)
0	5	0	0	0	0	0	0	b = i2 (6S+)
0	0	5	0	0	0	0	0	c = i3 (6S)
0	0	0	2	0	2	1	0	d = i4 (6S)
0	1	0	0	4	0	0	0	e = i5 (5)
0	0	0	1	0	4	0	0	f = i6 (6)
0	0	0	0	0	0	5	0	g = i7 (6S)
0	0	0	0	0	0	0	5	h = i8 (6S)

Figure 12: Confusion Matrix for IBk method (iPhone model included)

figuration, hence here a question arises: if we mix hardware and software method, do we get a very accurate and precise fingerprint?

- How much precision is needed to get a precise fingerprint? If we decrease the accuracy of gyroscope reading or if we change the time of our vibration, would this affect the results? We are using a .15 point float number for gyroscope's readings, but would this also work well with only e.g. .2 floating number?

- Effects of samples per second and sampling period: we used ten seconds with 100 samples per second for our experiments, would decreasing or increasing these numbers affect the results? If yes, how much? There are some discussions in [13] for accelerometers, however, still needs further research.

Recommendations: a list of recommendations to vendors and users should provide enough information to both parties to help them protect users' privacy and security. From the user perspective, she should be aware not to provide extra information and permission to apps and from supplier's perspective, it should be some changes to sensors access, like giving less accurate values to developers or calibrating sensors before releasing them to the market.

Rank	Feature
1	highestValue
2	rolloff
3	lowestValue
4	rmsAmplitude
5	specCetroid
6	std
7	flatness
8	specStd
9	mean
10	crest
11	smoothness
12	avgStd
13	specSkewness
14	specIrregulatiryJ
15	specIrregulatiryK
16	skewness

Figure 13: Ranking of features

8 Conclusions

In this paper, we demonstrate the possibility of generating device fingerprints and identify devices without user or manufacturer permission. Research on fingerprinting devices is currently an ongoing topic in the academia, many combinations of sensors are used to find a unique identifier for phones. Here, we show that one of these combinations is vibrator and gyroscope, we go through different papers which apply similar methods, but no one uses exactly gyroscope and vibration.

In our method, we read the gyroscope values while the phone is vibrating. Our main focus is iPhone and iOS, and since Apple removed unique device identifier from iOS7 to protect user's privacy and security, it is interesting to show that it is still possible to generate new ways to identify phones even without user or vendor's permission.

GyroVib generates 16 features per instance. Using 40 instances from eight iPhones and 16 features and a typical classification algorithm (K-nearest neighbors), we can identify each phone with 87.5% precision.

Other proposed methods ([13]) in the same field offer good results (over 90%). However, no other work is based only on the gyroscope. Our contribution in this paper is offering a sensor-based fingerprint with the gyroscope which has not been done before. The results are promising, but still more work (public evaluation) is needed to prove the method.

Acknowledgment

The author would like to thank Jun.-Prof. Dr.-Ing. Delphine Reinhardt for her insights and comments.

References

[1] Directive 2002/58/ec of the european parliament and of the council. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do? uri=OJ:L:2002:201:0037:0047:en:PDF, July 2002 (accessed July 08, 2016).

[2] ios sdk release notes for ios 7 gm. http://apple.co/29mXVZ2, October 2013 (accessed July 08, 2016).

[3] Asidentifiermanager class reference. http://apple.co/29vnxoX, September 2014 (accessed July 08, 2016).

[4] App store rings in 2015 with new records. http://www.apple.com/pr/library/2015/01/08App-Store-Rings-in-2015-with-New-Records.html, January 2015 (accessed July 08, 2016).

[5] Mitigating fingerprinting in web specifications. http://w3c.github.io/fingerprinting-guidance/, July 2016 (accessed July 08, 2016).

[6] Motion events. http://apple.co/1LE8m67, March 2016 (accessed July 08, 2016).

[7] Uidevice class reference. http://apple.co/29X2vhf, March 2016 (accessed July 09, 2016).

[8] About privacy and location services. https://support.apple.com/en-gb/HT203033, July 2016 (accessed July 25, 2016).

[9] Hristo Bojinov, Yan Michalevsky, Gabi Nakibly, and Dan Boneh. Mobile device identification via sensor fingerprinting. *arXiv preprint arXiv:1408.1416*, 2014.

[10] Jamie Bullock and UCEB Conservatoire. Libxtract: A lightweight library for audio feature extraction. In *Proceedings of the International Computer Music Conference*, volume 43, 2007.

[11] Anupam Das, Nikita Borisov, and Matthew Caesar. Tracking mobile web users through motion sensors: Attacks and defenses. In *Proceedings of the 23rd Annual Network and Distributed System Security Symposium (NDSS)*, 2016.

[12] Anupam Das, Nikita Borisov, and Matthew Cesar. Exploring ways to mitigate sensor-based smartphone fingerprinting. *arXiv preprint arXiv:1503.01874*, 2015.

[13] Sanorita Dey, Nirupam Roy, Wenyuan Xu, Romit Roy Choudhury, and Srihari Nelakuditi. Accelprint: Imperfections of accelerometers make smartphones trackable. In *NDSS*, 2014.

[14] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[15] Andreas Kurtz, Hugo Gascon, Tobias Becker, Konrad Rieck, and Felix Freiling. Fingerprinting mobile devices using personalized configurations. *Proceedings on Privacy Enhancing Technologies*, 2016(1):4–19, 2016.

[16] Ilias Leontiadis, Christos Efstratiou, Marco Picone, and Cecilia Mascolo. Don't kill my ads!: balancing privacy in an ad-supported mobile application market. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, page 2. ACM, 2012.

[17] Sky Mckinley and Megan Levine. Cubic spline interpolation.

[18] Tom Van Goethem, Wout Scheepers, Davy Preuveneers, and Wouter Joosen. Accelerometer-based device fingerprinting for multi-factor mobile authentication. In *International Symposium on Engineering Secure Software and Systems*, pages 106–121. Springer, 2016.

[19] Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.

[20] Zhe Zhou, Wenrui Diao, Xiangyu Liu, and Kehuan Zhang. Acoustic fingerprinting revisited: Generate stable device id stealthily with inaudible sound. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 429–440. ACM, 2014.