Generating unique identifiers for smartphones using software

G. Hammouri and B. Sunar

A fast and simple technique for reliably fingerprinting smartphones is presented. This technique extracts the fingerprint by capturing the small variation found in digital imaging sensors used in smartphone cameras. These variations are a result of manufacturing variability and, as such, they are different even for identical digital cameras. The main advantage of the technique presented is that it can quickly generate a fingerprint for a given phone using a mobile software that can be easily downloaded and installed without any changes to the hardware. Furthermore, this technique generates a reliable fingerprint using as little as five images while allowing the fingerprint to be used as a key to any cryptographic application. The techniques are summarised and the experimental data is presented.

Introduction: A large number of physical devices have already been shown to contain some form of unique fingerprint. Examples of such devices are digital circuits, CDs, regular writing paper etc. In general, these fingerprinting features result from variability in the manufacturing process. As expected, this phenomena also affects digital camera sensors. Owing to manufacturing variability, each sensor in a digital camera will experience a minute offset in the light intensity that it registers. While these offsets are too small to affect the performance of the camera they may still be utilised for unique identification. The early work in [1] presented a technique to identify a camcorder from videotaped images. This approach was also used to fingerprint digital cameras in [2]. A survey on this topic can be found in [3]. Here, we further develop this application and introduce a new processing step that allows the generation of a reliable fingerprint suitable for cryptographic applications using only five images. Our technique makes digital camera fingerprinting fast enough for use in smartphone identification.

Digital camera fingerprint: For an extracted value to serve as a fingerprint of a digital camera it needs to satisfy two main conditions:

- *Small intra-variation distance (noise):* That is, the 'distance' between fingerprints extracted from the same camera at different times should be small.
- *Large inter-variation distance:* The 'distance' between fingerprints extracted from different cameras at different times should be large.

Here we use the term 'distance' loosely to refer to any type of distance metric. Using the $L1$ distance it can be shown that when the distance separation between the intra-variation and inter-variation is large the fingerprint will have high entropy which in turn enables the use of the fingerprint in cryptographic applications [4].

Fingerprint extraction process: We model a digital camera (similar to [3]) having an m by n resolution as an m by n matrix I. After taking an image, I_{ij} will represent the intensity of the light captured by the (i, j) camera sensor. Note here that digital cameras do not assign a sensor for every primary light at every pixel location. Each primary colour is only assigned to a number of pixel locations and the rest are filled using a demosaicing algorithm. For now, we will ignore the different colour filters and focus on the sensors. The manufacturing variability effect on the digital image is modelled as a multiplicative factor which is independent for each of the different sensors. If we assume that J represents the intensity matrix captured using ideal sensors, then the actual intensity can be represented by $I_{ij} = J_{ij} + \mu_{ij} \cdot J_{ij}$, where μ_{ij} is the multiplicative factor capturing the manufacturing variability of the (i, j) sensor and will therefore represent the camera's fingerprint. The equation above assumes a noise-free process. To accommodate for noise we introduce ν_{ij} which we assume to affect each sensor independently and in an additive way. The measured intensity for pixel (i, j) can be represented by $I_{ij} = J_{ij} + \mu_{ij} \cdot J_{ij} + \nu_{ij}$. We simplify the above as $I_{ij} = J_{ij} + (\mu_{ij} + \eta_{ij}) \cdot J_{ij}$ where $\eta_{ij} = \nu_{ij} J_{ij}^{-1}$.

The key step in the extraction of the fingerprint is to apply a denoising filter $\mathcal{D}(\cdot)$. We use the filter discussed in [5]. We assume that that the output of the filter (the clean image) represents the ideal image J. Using the output of the filter we can now rewrite the equation above as

$$\frac{I_{ij} - \mathcal{D}(I)_{ij}}{\mathcal{D}(I)_{ij}} = \mu_{ij} + \eta_{ij}$$

where we assume that $J_{ij} = \mathcal{D}(I)_{ij}$. The above equation highlights other sources of noise such as the denoising filter $\mathcal{D}$ which will not work perfectly. Another noise source is the compression performed on the digital image. Since most images produced by digital cameras undergo a compression process and are commonly stored in JPG format, this means that the value we use for each pixel is only an approximation of the actually sensor reading. The reason we highlight these two sources of noise is that they will have a very different nature and distribution when compared with the environmental noise which we have addressed so far. To accommodate for these new noise sources we write $\eta_{ij} = \eta_{E_{ij}} + \eta_{D_{ij}}$ where $\eta_{E_{ij}}$ captures the environmental noise whereas $\eta_{D_{ij}}$ captures the noise resulting from the filter imperfections and the compression process.

These noise sources represent 'the challenge' in retrieving the digital camera fingerprint. To address this problem we use averaging over multiple images originating from the same digital camera. Note that all these images should have the same fingerprint μ_{ij}. However, the noise η_{ij} will be different for each of these images. On the one hand, the environment noise $\eta_{E_{ij}}$ can be expected to follow the same probability distribution for each of the images. On the other hand, $\eta_{D_{ij}}$ will be different depending on the nature of the images used for averaging. For simplicity, we assume that all images used in averaging are plain images of the same surface. This means that the filter imperfections and the compression effects will be almost constant for all the images used. Moreover, this assumption is quite realistic for our setting since we will have full control over the type of the images used by placing the phone on any surface with the camera immediately facing that surface. When this assumption about the images does not hold we can use nonlinear averaging techniques in order to extract the fingerprint.

Next, we analyse the effects of the averaging procedure. We use superscripts to indicate different images while using N to represent the number of images used for averaging. We can now write

$$\begin{aligned}\frac{1}{N}\sum_{t=1}^{N}\frac{I_{ij}^{t} - \mathcal{D}(I^{t})_{ij}}{\mathcal{D}(I^{t})_{ij}} &= \frac{1}{N}\sum_{t=1}^{N}\mu_{ij} + \eta_{ij}^{t} \\ &= \mu_{ij} + \frac{1}{N}\sum_{t=1}^{N}\eta_{D_{ij}}^{t} + \frac{1}{N}\sum_{t=1}^{N}\eta_{E_{ij}}^{t}\end{aligned} \tag{1}$$

When the images used are for similar surfaces the term $\alpha_{ij} = (1/N)\sum_{t=1}^{N}\eta_{D_{ij}}^{t}$ will yield a constant which depends on the nature of the used images. The last term $\beta_{ij} = (1/N)\sum_{t=1}^{N}\eta_{E_{ij}}^{t}$ will essentially be an average over samples coming from a random variable. Assuming that $\eta_{E_{ij}}^{t}$ follows a Gaussian distribution, then the standard deviation of the random variable will scale with $(1/\sqrt{N})$. So, even for $N = 4$, the noise level will be dropped in half. The outcome of (1) can now be approximated by $\mu_{ij} + \alpha_{ij} + \beta_{ij}$. For a fingerprint to be useful it has to be reproducible with a low noise rate and has to be different for different cameras. In the equation above, μ_{ij} will be the same even when two different batches of images are used. Similarly, μ_{ij} is expected to be different for different cameras. The third term β_{ij} will be different for different batches of images. However, the effect of β_{ij} can be significantly reduced by increasing the number N of images. The only problem we encounter is with the term α_{ij} which, unfortunately, does not behave like β_{ij}. In fact, this term will have a magnitude which will dominate the value of the fingerprint μ_{ij}.

Essential step: To reduce the effect of α_{ij} we model its effect as an independent random variable, which has the same distribution for each of the (i, j) locations. If μ_{ij} was fixed across the rows or columns then averaging over the rows or columns would actually decrease the effect of α_{ij}. However, recall that we have already assumed that μ_{ij} (the fingerprint) is sampled from some distribution and is independent for each of the mn pixels. This means that averaging over the rows or columns would simply yield another random variable with considerable noise.

Fortunately, it turns out that the camera sensors are typically read through the last row of pixels. After the values of the last row are stored, the charge produced by all the upper rows is shifted down by a single row. Afterwards, the readings of the next to last row are stored. This process continues until all the sensor readings are stored.

Again due to imperfections in the hardware the charge that passes from one row to another is not fully transferred and therefore creates significant dependencies between the row readings. This phenomenon would typically not be desired. However, for our purposes this behaviour gives rise to the most important step of our fingerprinting technique. Owing to the dependencies between the rows the μ_{ij} values will not be independent throughout each column. This means that averaging over the rows would decrease the effect of α_{ij} while giving rise to the effects of μ_{ij}. With this step the left-hand side of (1) becomes

$$\frac{1}{mN}\sum_{i=1}^{m}\sum_{t=1}^{N}\frac{I_{ij}^{t}-\mathcal{D}^{t}(I)_{ij}}{\mathcal{D}^{t}(I)_{ij}}=x_j \tag{2}$$

We have used x_j to denote the fingerprint extracted from the jth column. The full fingerprint is denoted by $X=[x_1 \ldots x_n] \in \mathcal{R}^n$. According to our model, X will capture the extracted fingerprint of a digital camera regardless of the nature of the image used in the extraction process. Naturally, this fingerprint will have some level of noise which we need to reduce as much as possible. In our technique, we place the camera directly on a surface and consecutively capture N images. As we will see from our data, this restriction will cause the L_1 distance between X^1 and X^2 (X extracted using two different surfaces 1 and 2) to be small. At the same time, this will cause the L_1 distance between X, Y (two fingerprints for different digital cameras) to be large. The complete fingerprint extraction procedure is summarised below. Note that the matrix operations are computed element wise.

Algorithm for fingerprint extraction

EX: $X \leftarrow \mathrm{EX}(I^1, \ldots, I^N)$ where $X \in \mathcal{R}^n$ and $I^i \in \mathcal{R}^{m \times n}$

1. Set i = 1 and $A = 0^{m \times n}$ where A_i is the ith row of A.
2. Compute $A = A + \left(I^i - \mathcal{D}(I^i)\right)/\mathcal{D}(I^i)$.
3. Increment i. If $i \leq N$ repeat step 2.
4. Return $X = (1/mN)\sum_{i=1}^{m} A_i$.

Here we stress two points. First, X can be labelled X^r, X^g or X^b to represent the fingerprint extracted from the red, green or blue intensity matrix I. Ideally, these three fingerprints will be independent and thus will have a large L_1 distance between them. However, due to the use of a Bayer filter, the red and blue fingerprints will have more internal dependency than the green fingerprint. In fact, because of the Bayer filter layout and our row averaging step, only half of the entries of X^r and X^b will be independent whereas the other half will be reconstructed from the independent entries. On the other hand, every entry X^g will be independent. Thus, we would expect the distance between X^r and X^b to be roughly half of their distances to X^g. This is precisely what our empirical data suggests. The results we report here are for the output of the green filter. The second point is that X can be reliably extracted with some level of noise. This is useful. However, to use X in any cryptographic application we need to guarantee a zero level of noise. This problem is typically encountered when trying to use biometrics to identify humans. 'Fuzzy extractors' were introduced to produce a noiseless version of X [4].

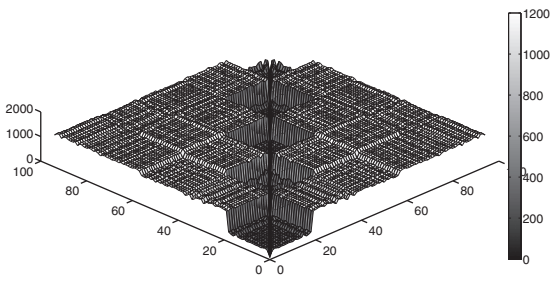

Fig. 1 *L_1 distance between fingerprints extracted from five different iPhone-4 devices using 11 different surfaces*

Experimental validation: To verify our technique, we coded our algorithm and ran it on a several identical iPhones. In particular, we ran our iPhone application on five different iPhone-4 devices placed on a surface and took five images from 11 different surfaces (surfaces included black desktop, blue rug, brown box, purple box, white paper, metallic CD case and so on). For each surface and phone a fingerprint was generated, bringing the number of extracted fingerprints to 55. To validate that the extracted fingerprint was for the device rather than the surface, we measured the L_1 distance between all 55 fingerprints. The distance between the different fingerprints is shown in Fig. 1. The x4- and y-axes represent the number of the fingerprint where each 11 consecutive fingerprints originate from the same phone. The z-axis represents the L_1 distance.

Fig. 1 clearly shows that the L_1 distance is considerably smaller for fingerprints originating from the same phone even when the surfaces are different. The square holes on the surface represent the L_1 distance for fingerprints originating from the same device using different surfaces. Table 1 shows the average L_1 distance for fingerprints coming from different phones. Again it is clear from Table 1 that a large L_1 distance separates fingerprints originating from two different devices even when the pictures are taken over identical surfaces. We end this Section by mentioning that the entire process of extraction including acquisition of images, de-noising, averaging and the extraction of a fingerprint takes < 30 s on an iPhone-4.

Table 1: Average L_1 distance between fingerprints extracted from different devices

iPhone	1	2	3	4	5
1	**362**	1169	1168	1246	1297
2	1169	**469**	1287	1350	1238
3	1168	1287	**384**	1258	1241
4	1246	1350	1258	**396**	1239
5	1297	1238	1241	1239	**453**

Conclusion: We have presented a fast and simple technique for reliably fingerprinting smartphones using the small variation in digital imaging sensors. The technique takes advantage of the way data is collected from the sensors in order to produce a fast algorithm to fingerprint a given phone using only five images. Our technique was validated on an iPhone-4.

© The Institution of Engineering and Technology 2014
16 November 2013
doi: 10.1049/el.2013.3618

G. Hammouri (*Crags Inc., Worcester, MA, USA*)

E-mail: hammouri@alum.wpi.edu

B. Sunar (*Worcester Polytechnic Institute, Worcester, MA, USA*)

References

1 Kurosawa, K., Kuroki, K., and Saitoh, N.: 'CCD fingerprint method – identification of a video camera from videotaped images'. Proc. Int. Conf. Image Processing, Kobe, Japan, October 1999, Vol. 3, pp. 537–540

2 Goljan, M., Filler, T., and Fridrich, J.: 'Camera identification – large scale test', Electronic Media Forensics and Security XI, San Jose, CA, USA, January 2009, *Proc SPIE*, **1254**, pp. 0I-01–0I-12

3 Fridrich, J.: 'Sensor defects in digital image forensics', in (Eds): 'Digital image forensics: There is more to a picture than meets the eye' (Springer, New York, May 2012)

4 Dodis, Y., Ostrovsky, R., Reyzin, L., and Smith, A.: 'Fuzzy extractors: how to generate strong keys from biometrics and other noisy data', *SIAM J. Comput.*, 2008, **38**, (1), pp. 97–139

5 Mihçak, M.K., Kozintsev, I., and Ramchandran, K.: 'Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising'. Prof. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, Phoenix, AZ, USA, March 1999, Vol. 6, pp. 3253–3256