Camera Fingerprinting Authentication Revisited

Dominik Maier[1], Henrik Erb[2], Patrick Mullan[2], and Vincent Haupert[2]

[1]Technische Universität Berlin
[2]Friedrich-Alexander-Universität Erlangen-Nürnberg

Abstract

Authentication schemes that include smartphones gain popularity. Instead of storing keys in app-private storage — cloneable by privileged malware — recent research proposes authentication with hardware fingerprints, arguing they will be harder for attackers to fake. Notably, the use of camera sensor fingerprints has been discussed recently. This paper revisits the eligibility of this camera sensor noise for authentication. The so-called *Photo Response Non-Uniformity (PRNU)* exploits use of production tolerances in the CMOS sensors, commonly used in smartphone cameras, to trace a photo to a specific phone and authenticate its user. We conducted the first large-scale study for PRNU on smartphones, with 56,630 images stemming from individual 3,809 devices across 1036 models. Based on the collected dataset, we reproduce proposed authentication schemes and uncover caveats not discussed in prior work on authentication. In addition, we give constraints an image used for authentication schemes needs to fit, to increase the reliability of the results. We are able to provide novel insights, implement attacks against the proposed schemes and discuss future improvements.

1 Introduction

With the emergence of smartphones, users carry around powerful multipurpose computing devices. As these devices are highly personal, identifying the smartphones typically makes it possible to identify the respective owner. For certain use cases, for example banking apps, smartphone apps replaced true second factors, such as hardware tokens. Some modern banking apps and the start-ups behind them value user experience and time to market over security [27] — and the majority of users seemingly prefers apps over purpose-built hardware devices they have to buy and may lose. In contrast to purpose-built hardware tokens without internet access, on phones privileged malware can misuse any secrets stored in apps. Haupert and Müller show that such app-based transaction schemes for banking — even those relying on two separate devices — can be attacked [29, 30]. Instead of storing secrets that can always be copied off [28], apps can calculate unique fingerprints of phones on the fly. The idea of device fingerprinting is simple: production tolerances in sensors and other input hardware are assumed to be unique enough to be utilized as identifying key for each individual device. Fingerprinting of browsers and mobile phones alike, is already used extensively for marketing purposes [57]. In recent years, researchers and vendors also adapt fingerprinting schemes to identify and authenticate the user and the device. One well-suited sensor is the phone's camera. On photos taken, the imaging sensor leaves imperceptible noise that can be used to identify the respective camera. Ba *et al.* [4] as well as Valsesia *et al.* [56] constructed authentication schemes based on camera fingerprinting. These schemes work on any current smartphone without additional hardware and therefore without impacting usability. Attackers, on the other hand, may also be able to learn the respective fingerprints and abuse them for malicious authentications. In this paper we take a close look at the schemes proposed. We revisit the idea of camera fingerprinting and put effort into providing an in-depth answer to the question, if and how camera fingerprinting enables secure smartphone authentication. An adversarial mindset and the results of a large-scale study unveil drawbacks in camera fingerprinting for authentication. Notably, we provide simple attacks against each attack detection step Ba *et al.* [4] present.

Contributions

In detail, we make the following contributions:

- Our evaluation provides an in-depth view on PRNU camera fingerprinting for authentication. As part of our research, we gathered images from 3,809 unique smartphone devices yielding 56,630 pictures, from 1036 models running both iOS and Android. To the best of our knowledge this is by far the largest dataset of images from smartphones recorded under a controlled environment, e.g. our self-implemented app, allowing for a realistic security assessment. With this, we were able

to reproduce and proof prior assumptions and published results.

- We introduce a forgery resistent camera authentication scheme with potential real-world security benefits, even on phones where keys can be securely stored.
- We present realistic attacks against proposed authentication schemes based on camera fingerprinting. We uncover flaws in defenses of the ABC protocol proposed by Ba *et al.* [4]. Most notably, we elaborate a replay attack and two fingerprint forgery attacks.

2 Background

First we provide background and related work around device and user fingerprinting as well as authentication. We discuss a wide variety of ways to fingerprint devices and users to then dive into details about camera fingerprinting.

2.1 Device Fingerprinting

For the web, fingerprinting users and their browsers in a privacy invading way is well researched and widely adopted for advertisements and tracking [1, 14, 46, 57]. This trend also emerges on phones, where an even larger variety of sensors can be leveraged. Mobile device fingerprinting through apps is of ongoing research interest. In the following, we present the state-of-the-art in device fingerprinting for mobile platforms.

Fingerprinting by Exploiting Properties of Sensors Prior research proves that most hardware sensors on phones can be used to fingerprint specific devices. Yue [60], Das *et al.* [13] and Bojinov *et al.* [6] have all used position sensors, accelerometer and the gyroscope, to fingerprint phones and their users. Hupperich *et al.* [31, 32] go one step further and use the available sensor data for an authentication scheme based on the device fingerprint. Das *et al.* fingerprint the devices using microphones [12]. Zhou *et al.* [62] and Das *et al.* [12] use speakers and microphones modules to uniquely identify a smartphone.

Fingerprinting by Exploiting Human Characteristics Fingerprinting the user directly, recognizing differences in individual movements and behavior, for authentication, is another theme. Nickel *et al.* recognize the users' walking patterns to authenticate them [45]. Frank *et al.* track specific touch behaviors for authentication [17]. Gong *et al.* propose a forgery resistant tracking method for user touches [25]. Bo *et al.* [5] combine touch and sensor fingerprinting to authenticate specific users. Instead of recognizing input, Kurtz *et al.* [35] and Wu *et al.* [59] leverage user settings and files to track iOS and Android devices with high accuracy. Further proving users' actions can be used for unique identification,

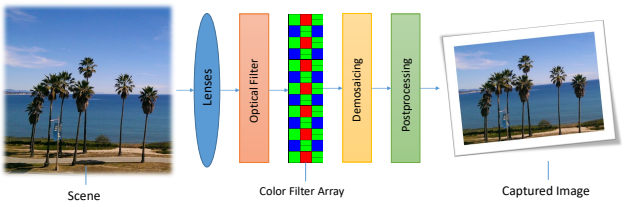


Figure 1: Digital Camera Image Pipeline

Stöber *et al.* [54] identify smartphones by monitoring the network traffic from Facebook, Skype, WhatsApp and Dropbox for 15 minutes. Reaves *et al.* [51] propose the use of phone calls to authenticate the user remotely.

2.2 Smartphone Camera Fingerprinting

With camera fingerprinting, this paper focuses on anomalies in the CMOS sensor of smartphones. Before light reaches the CMOS sensor of a camera, it passes through multiple elements. All of these, as depicted in Figure 1, introduce imperfections specific to a device or make. Light travels through lenses, an anti-aliasing filter and the color filter array. Typical parts of the image pipeline are discussed by Ramanath *et al.* [50]. Like all elements in the pipeline, and other sensors in a phone, the CMOS sensor of every camera is subject to production tolerances. The phone camera's megapixels translate to the number of colored dots it captures. Every pixel in the final photo then consists of intensities of the three colors red, green and blue. These intensities are the amount of light reported by the CMOS sensor at this position. Some CMOS sensor elements will systematically interpret the light impact more or less intense by a slight margin, that might exceed the granularity of discretized pixel values. Ultimately, this leads to a peculiar pattern that can be found with varying strength throughout all images of a given imaging device. This pattern is called *Photo Response Non-Uniformity* (PRNU). Even though, this pattern is weak, usually imperceptible to the human eye, methods have been developed to extract it reliably from images [22, 38]. Further, these works showed as a remarkable property of PRNU that it is highly unique across different devices. This holds even for different devices of the same brand and model. We focus on PRNU in this paper even though CMOS Image Sensor Fixed Pattern Noise is also unique, as discussed by Kim and Lee [34], as it works with brightly lit, captured photos. Lukás *et al.* [38] conclude that other sources of noise, like fixed-pattern-noise and shot noise, are not as well-suited.

For the mathematical explanation of the PRNU, refer to Appendix A. The PRNU was shown to be reasonable stable over the lifetime of a camera [19] and provides a solid way to associate an image to its source device. For the sake of completeness it shall be mentioned that, for our study in Section 3, further post-processing of the estimated fingerprint is done

by our C++ code. Specifically, the code removes non-uniform artifacts, as other implementations do as well. As these optimizations are not fundamental to the concept of PRNU, we do not elaborate on them here, but refer the interested reader to literature like [21].

To verify if a new image was taken with a specific camera, the residual of the new image is calculated as shown in Equation 2. Then this residual is correlated against a reference fingerprint extracted of the camera in question. With the correlation we obtain a value expressing the similarity between the reference fingerprint and the new residual under test. As correlation metric, the normal pearson correlation could be used. However, the *Peak Correlation Energy* ρ has shown to be a more suitable option for this application [21]:

$$\rho_{[I,\hat{K}]} = \mathrm{PCE}(W_i,\, I\hat{K}) \quad . \tag{4}$$

Based on this mathematical foundation, a considerable amount of research went into improvements to increase the quality of the extracted noise pattern [9, 23]. Furthermore, several attacks and counterattacks were discussed. Entrieri and Kirchner [15], Karaküçük *et al.* [33], as well as Li *et al.* [37] show how erasing and spoofing of camera fingerprints is possible and can be mitigated.

2.3 Camera Fingerprinting Authentication

We will briefly discuss two authentication protocols based on PRNU. The idea for both schemes is to authenticate by the unique fingerprint of a smartphone camera.

The elements commonly present in camera fingerprinting authentication schemes are as follows:

Elements

The Verifier
The verifier poses an initial challenge to the smartphone and verifies the final decision.

The Terminal
The terminal can be any device displaying the challenge (QR-Code) to the smartphone. It could either be a POS terminal, a computer screen or some other display and will be fed by the verifier.

The Smartphone
The device to be authenticated. The smartphone takes a photo of the terminal, adding its unique PRNU fingerprint to the image in this process. It will forward the final photo(s) to the verifier.

ABC According to the ABC scheme by Ba *et al.* [4], the verifier and terminal can be implemented on one device. The verifier needs access to a database of all registered phones. The basic interaction between all elements is depicted in Figure 2 and is as follows:

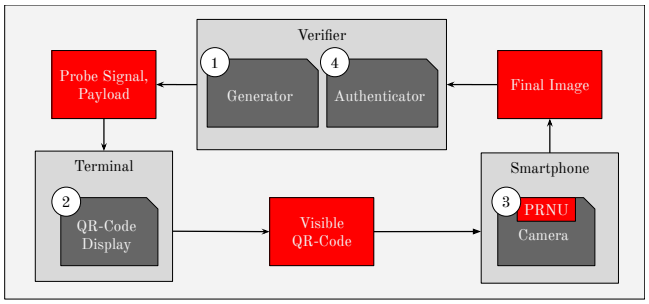


Figure 2: Camera Authentication Building Blocks

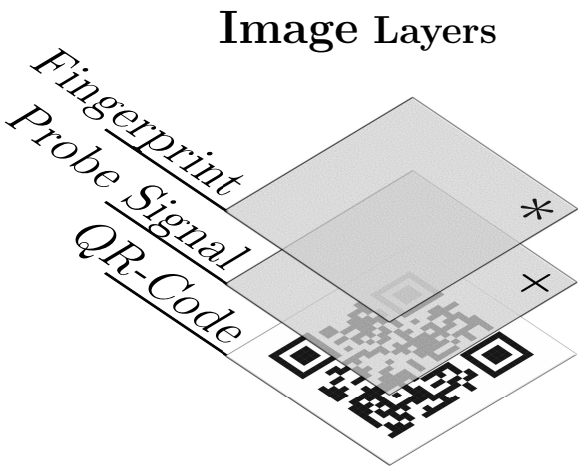


Figure 3: Layers of an image used for ABC authentication [4]

① The *Generator* creates an image with a QR-Code. The QR-Code contains transaction details and a timestamp. Then the probe signal is added to the image and sent to the terminal.

② The image is presented to the user on a display on the *Terminal*.

③ Agreeing with the content, the user captures an image of the screen containing the QR-Code and the probe signal. As a side effect of the image capturing, the PRNU of the camera sensor is added to the photo. The final image is then sent to the authenticator.

④ The *Authenticator* extracts the fingerprint of the final image and compares this to the reference fingerprint of the user stored at the server. If the PCE value is above a certain threshold, the transaction is authenticated.

Based on the apps created for our study, discussed in Section 3, we were able to implement the ABC protocol discussed hereafter. It is based on a registration phase and an authentication phase. For the registration phase, the user uploads one image taken through the smartphone. Ba *et al.* [4] claim that

one image for extracting the reference fingerprint is sufficient. During the authentication phase, the verifier challenges the user to take two images of a screen with two different QR-Codes. Both QR-Codes contain a time stamp and an excerpt of the ongoing transaction. The user then verifies the information about the transaction shown on the smartphone's display, see Figure 2. An additional *probe signal*, designed to survive photographing but not fingerprint removal, gets added to the QR-Code displayed on the terminals' screen. It is specified as *additive* white gaussian noise with a standard deviation of 5. The fingerprint in contrast, is *multiplicative*. A schematic illustration of this pipeline is depicted in Figure 3. The phone then sends the two photos to the verifier. The verifier determines the PCE value between the two images and each image with the reference image. If both values are above a certain threshold and equally high the verifier authenticates the user's request.

RAW vs JPEG Another camera fingerprint-based authentication protocol was proposed by Valsesia *et al.* [56]. They also use the PRNU as physical unclonable property. For their authentication scheme, they assume that only the user or authentication app will have access to the RAW image data of the smartphone. All publicly available images are compressed and, therefore, do not contain the high-frequency components of the fingerprint of a RAW photo. Valsesia *et al.* [56] introduce a way to compress the fingerprint using random projections. This reduces their size and brings a big advantage for transferring and storing the PRNU of multiple cameras. Also, the raw fingerprints do not need to be sent over the network completely. Secret side information for the random projections never leaves the side of the smartphone. The server does not store the compressed fingerprint itself, but uses a fuzzy extractor scheme to create a uniformly random bit string. This prevents an adversary, who gains access to the server, to obtain the stored fingerprints of all users easily. Quiring *et al.* [48, 49] propose the use of Fragile Camera Fingerprints, proofing the fingerprint cannot be recovered from JPEG images, but RAW images alone. To attack these schemes, access to the phone is required, as RAW images rarely get posted to social media.

3 Large-Scale Data Acquisition

To study the real world applicability of the proposed schemes, we implemented a setup for authentication and evaluated it on thousands of images. The dataset was collected from scratch, using apps that resemble implementations of the proposed authentication solutions. We describe here how we designed our apps but first discuss advantages of our new extensive dataset in general, and compare it to other already existing datasets.

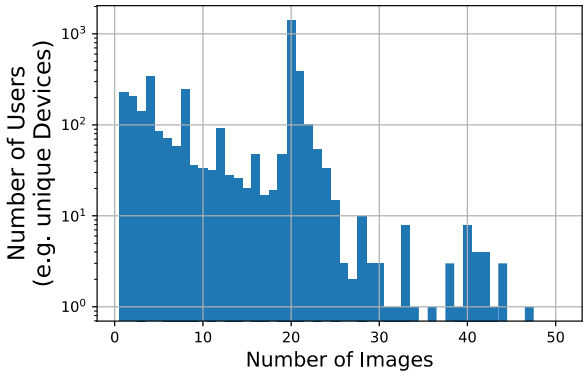


Figure 4: Number of images for individual devices

3.1 Methodology for Data Acquisition

According to Zhang and Zhang [61], around 20 images as training set are a good balance between image count and strength of the PRNU. Our initial tests on a small number of phones showed that additional images still increase the PCE of generated patterns against new images. The PCE for 20 images was usually above the threshold of 60, proposed by Goljan *et al.* [21]. We decided to let participants in our study capture and upload around 20 images to learn their individual PRNU pattern. User were free to quit the study before completion and could upload more images, if desired. This explains why the total number of images is not a multiple of 20. To minimize the workload for participants further, one click on the trigger captured batches of 5 photos in a few seconds of time. Even a small delay between subsequent images within a batch drastically increases the changes of two back-to-back images being unaligned [58]. The apps save *JPEG* files in full resolution and default quality offered by the underlying platform. In Figure 4 the distribution over number of images per unique device is shown. Even though our study thoroughly informed the user about privacy implications and asked to never upload photos of persons, we will not publicly release the dataset at this point, as proper vetting of 56,630 images is infeasible. For reproducibility, access to the dataset can be made available upon request.

Our large dataset is tailored for investigating authentication schemes on mobile phones, and hence surpasses other image forensic database for this purpose. For instance the large dataset of Goljan *et al.*'s study [21] consists of samples from photo platforms on the internet, however our dataset was only collected through our auth-app, on which we have full control. This tight grip means no (unknowingly) cropped, edited or otherwise altered images impact our training set negatively. The Dresden Database [18] is another image database, popular for image forensic works, containing images from 73 unique devices sampled from 27 models from 4 manufacturers. Both datasets were collected prior to 2010, meaning on

images taken with classic cameras, e.g., not contemporary smartphones in the context of mobile authentication. The VISION database [52] was tailored for mobile devices, however is rather small in comparison to our dataset, with only 35 devices. RAISE [11] is another database often used in works focusing on image forensics. It was designed to offer high-quality raw images from 4 professional DSLR cameras, hence also not suitable for our purposes of vetting PRNU based smartphone authentication schemes.

The authentication platform implemented for our study and the backend consists of three main components, apps for Android and iOS, the Database Server, and a worker server to run PRNU comparisons and evaluations asynchronously.

The apps for iOS and Android can be compiled in different modes: *large-scale study* or *authentication*. In large-scale study mode, both apps for iOS and Android inform the user about the study, goals and the types of photos they are supposed to upload. It provides live information on how their camera compares to that of other phones, including median, mean, maximal and minimal PCE of the user phone's PRNU against other images. This mode was used to gather data about the fingerprinting effectivity and to collect the dataset.

On the workers, the C++ core to extract the PRNU and compare it to images, *MagicFern*, is a speedy reimplementation of the popular Matlab framework by Goljan *et al.* [21] available on their website. It is quick, due to the low level implementation and multi-threading. We benchmarked our tool against the Matlab implementation to ensure we achieve a same level of accuracy. All parts of the platform will be open sourced upon publication.

3.2 Participants

All users participated voluntarily in our large-scale study. The app was available to the public and announced on social media and news outlets. First, it informs the participants about all processes transparently. Then, it instructed them to upload different scenes and never to upload personal or identifying images. The only information gain associated with the image and PRNU is, if a phone participated in our study. Although we advised users to not photograph private entities, like other people's faces, we can not guarantee all users complied due to the large amount of uploaded photos. We will therefore not release the images publicly.

The large-scale study had a total of 3,809 participants from both major mobile platforms over the course of one month. In a platform breakdown, 25% of patterns were learned on iOS, 75% on Android. In total, 1036 different models from 137 manufacturers took part in the study. The top three brands were *Samsung*, followed by *Apple* and *Google*. In total, we collected 56,630 images.

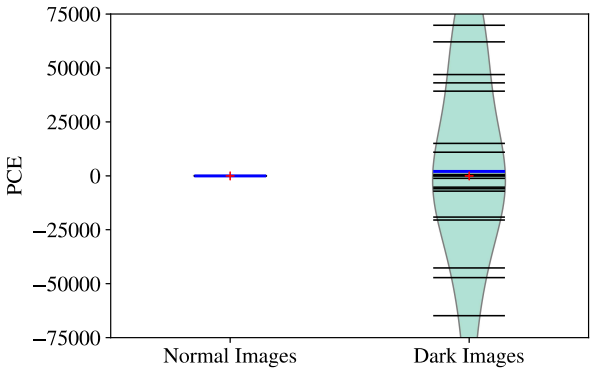


Figure 5: Distribution of estimated PCE values for two datasets. One dataset has only dark images the other one arbitrary sampled images. Further, all images were picked to *not* match a reference fingerprint. Normal images exhibit a low PCE score, as desired. Dark images in turn behave arbitrary, and often even have a large positive value.

4 Evaluation

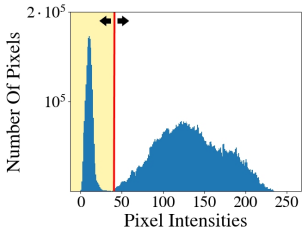
Based on the study, discussed in Section 3, we evaluate and rate the results we obtained from these 56,630 unique images of 3,809 smartphones. We look into the quality of the fingerprints on smartphones and determine constraints to obtain a fingerprint of the highest possible quality. We examine the behavior of the PCE as correlation measure for images and fingerprints under varying characteristics. Images recorded by the same camera should match, hence need to have a high PCE value. Images from different cameras should not match. Their PCE value should be close to zero. Since smartphone cameras produce images of different resolutions calculating the PCE value can be tricky. A common way of overcoming this problem is by cropping out a fixed size patch from all images [21, 23, 24, 36]. We picked a squared patch of size 1024 pixels always centered.

4.1 Effect of Illumination on PRNU

We examined the bad influence of known issues for noise extraction and noise correlation to evaluate the importance of the different constraints. In general, extreme PCE values appear with photos that are too saturated. A setting that is known to be challenging, or even preferable to avoid, if possible, for fingerprint estimation [9, 23].

Influence Of Dark Areas We reproduced previous research [22] showing a clear correlation between image illumination and the PCE value. Our evaluation script inspects all images from each user and compared them to all other images of this user/phone. This should always yield high PCE

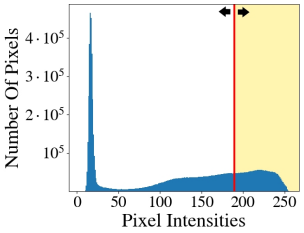
(a) Photo wih large homogeneous areas and a small local standard deviation of 0.006.



(b) Corresponding histogram to the homogenous photo.

Figure 6: Typical scenery image and histogram of its pixel intensities.

(a) Photo of a QR-Code. The standard deviation of 0.174 is very high



(b) Corresponding histogram of QR-Code image.

Figure 7: Typical Image of QR-Code to forward transmission details in an online banking scenario as suggested by [4]. To the left the corresponding histogram of pixel intensities.

values, because they were captured with the same camera, but this oftentimes fails for very dark images. In Figure 5 we plotted the distribution of 21 comparisons of a camera fingerprint from an arbitrary picked phone against random images of different phones by determining the PCE value. The values of normal images are to be expected. They are close to zero with small variance. In strong contrast, the other dataset shows the PCE values of the comparison of only either dark or completely black images from different cameras. The absence of any noise information sometimes randomly triggers very high positive or negative correlations. They generate almost arbitrary PCE values.

To improve the results, we then only took images which met certain constraints, namely an intensity threshold. In Figure 6b we show the histogram of Figure 6a. The varying threshold is represented as the red line. From every image the percentage of the pixels with an intensity below the threshold was determined. This percentage of pixels is illustrated as the yellow area. After this, we varied the constraints an image needed to meet for being taken for fingerprint extraction.

The question arises if there is some sweet spot for the intensity of the illumination of an image or if brighter is always better. In theory, above a certain brightness the pixels of an image are saturated and the fingerprint attenuates. Therefore in the next step, we examined the dependency of the PCE value under the constraint of brightness.

Influence Of Bright Areas The images used for determining the PCE values again needed to fit two constraints. A varying amount of pixel intensities needed to be above a varying threshold. This time we chose a varying threshold from 155 to 255. The threshold is illustrated as the red line in Figure 6b. The yellow area represents the pixel intensities for determining the percentage. The histogram of Figure 6b belongs to the image of Figure 6a. One image per user was compared with all other images of the user, if they fitted the given constraints. From this, the median of all PCE values was determined. Every dot in Figure 8 represents one of the medians. Again we used 100 users and determined the mean over all PCE values per varying threshold. The mean values result in a quadratic curve with a maximum at 173. Therefore we can conclude that one obtains the best results for images that should match, if the threshold for the brightness of the pixels is at 173.

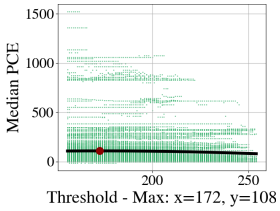


Figure 8: Median PCE for a set of images, where a percentage of pixels is above the depicted threshold (155 to 255)

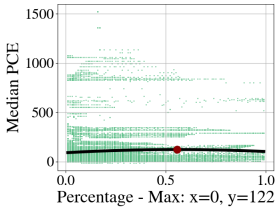


Figure 9: Median PCE for a set of images, where the depicted percentage of pixels is above a certain threshold (155 to 255)

Figure 9 shows the same data as Figure 8. The difference lies in the fact that we look at the varying percentage of pixels with an intensity above a certain threshold. The mean over all PCE values per percentage also result in a quadratic curve. The maximum is located at 56%. We can observe that it is not useful to have images as bright as possible but rather relatively high.

Perfect Ratio Between Percentage And Threshold As a further step we optimize for the perfect ratio between the chosen percentage and threshold. If we vary both variables at the same time we obtain a surface. We varied the percentage of the pixels that need to be above a certain threshold from 0% to 100%. In addition we varied the threshold the pixel intensities needed to be above, from 155 to 255. We then used a box linear filter with the kernel size of 9 to smooth the PCE peaks to make a more general statement. The resulting surface is presented in Figure 10. The global maximum lays at a percentage of 100 and at a threshold of 197. The plot confirms that, with increasing percentage, the PCE value increases as well. However, as depicted in Figure 8, the PCE stops increasing and even decreases for very high values. In

Figure 10 we see, that it falls off very fast after reaching the maximum. Too bright images produce very bad results for fingerprint correlation.

Concluding from this, the more areas of an image contain a high pixel intensity, the better the fingerprint extraction works. To obtain a fingerprint with the highest possible quality, images with 100 percent of the pixel intensities above 197 are the best fit. This means, a possible authentication scheme can optimize screen brightness on the one hand, and reject bad images on the other.

4.2 Scenery as Constraint

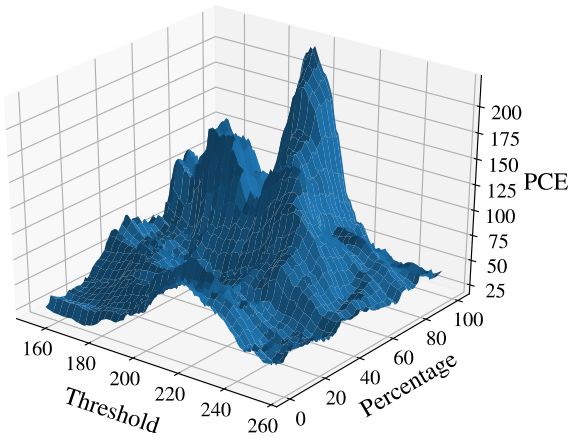


Figure 10: Surface plot describing the behavior of the PCE of images, which meet the constraints of the respective threshold and percentage. A sweet spot is found if most pixels are at an intensity of ≈ 200.

The photographed scenery impacts the quality of the extracted PRNU [9, 23], as well. The ultimate goal of the high-pass filtering, as described in Equation 2, is to suppress the image content. Nevertheless some content, even if weak, always remains, and leaks into the noise residual. This affects the performance of authentication schemes.

From every image of our study, we determined the standard deviation of every pixel intensity to its eight neighbors. We then calculated the mean overall standard deviations and used this value for further comparisons. As in the calculations before, we calculated the PCE value from one image of a user with all other images of the user. Then, we constrained the images used for PRNU extraction to a mean standard deviation above a certain threshold. In Figure 11, all dots of a particular color belong to one user. The plot shows 60 different users. With each dot the constraint of the standard deviation was increased to the point that one image dropped out of the set. In most of the sets of the users, this resulted in an increase of the mean over all PCE values of the images.

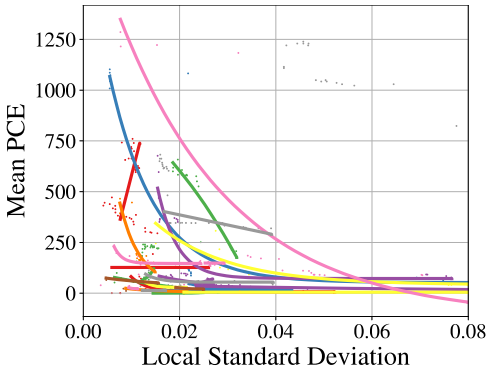


Figure 11: Calculated PCE value as a function of local standard deviation of pixel intensities in an image. The general trend of the downwards sloping curves show that variance impacts the magnitude of PCE.

Furthermore, we fitted a curve into the point clouds. The behavior of the PCE values over a decreasing standard deviation seems to be exponential. We can therefore conclude that the local standard deviation of an image is crucial for PRNU extraction. Homogeneous areas in an image increase the quality of the extracted fingerprint. This results in a higher PCE value for images that should match. Therefore, filtering images under the constraint of the local standard deviation results in a strong increase of the true positive rate.

The local standard deviation of the images of Figure 6a and Figure 7a differ a lot. The QR-Code with a lot of edges and a strong Moiré pattern has a local standard deviation of 0.174. The image of the sky has a local standard deviation of 0.006, which is extremely low, in fact about 30 times less. Therefore, it seems that images containing a QR-Code are not optimal for fingerprint extraction.

5 Discussion

In this section we discuss the impact of our results for camera fingerprinting as authentication scheme. We look into flaws in the authentication protocols based on our findings. We present several possible attacks and defenses.

5.1 Images *DO* Need Constraints

Ba *et al.* [4] do not discuss possible constraints for photos taken for their ABC scheme. As we show in Section 4, the photo content has a strong impact on the PCE quality. Figure 11, for example, shows that the PRNU extraction is negatively affected by high frequency scenery in the image.

Also, concluding from Section 4.1 we can say that the brightness plays an important role for obtaining a good fingerprint resulting in a low false negative and low false positive rate. Images with 100% of the pixel intensities above 197

yield the best results. Since cameras adapt the brightness automatically, taking such a photo might only be possible in extreme lighting conditions. The QR codes used in [4] are default black and white prints. Therefore, depending on the exact exposure and recording, the color settings in such QR codes are prone to produce pictures that are unsuitable for proper PRNU analysis. To improve the true positive rate of an authentication protocol even further, one could add a function for determining the local standard deviation of the challenge and reference images. From Section 4.2 we know that this improves the quality of the extracted fingerprint as well. With the mentioned constraints, we can essentially increase the reliability of an authentication protocol.

Also, we can conclude that rejecting images for fingerprint extraction with a high local standard deviation can decrease the false positive rate and false negative rate. Images with a low local standard deviation in many areas of the image are suited very well for fingerprint extraction. The sample images in Figure 6a and 7a depict histograms of comparable shapes, as seen in the neighboring Figures 6b and 7b. However, the images differ significant with respect to local variance. The sky image has large flat areas, the QR image has many transitions from black to white. The sky image has a very low standard deviation of 0.006, but the QR image has a standard deviation of 0.174. Looking at the plot in Figure 11 we see, that with increasing standard deviation, the quality of the PRNU decreases exponentially.

5.2 QR-Codes for Fingerprint Extraction

The ABC paper does not specify the size of the QR-Code relative to the whole image [4].

We find that images containing QR-Codes have a high local standard deviation. It is almost impossible to extract a correct PRNU from an image area containing a large QR-Code as scenery. The scenery of an image containing a QR-Code has high frequencies: The black and white QR-Codes consist of lot of edges, which leads to very high frequency components in the image. Further, the QR images were taken by capturing the QR-code from a LCD-screen of a monitor. This induced a Moiré pattern. Since the PRNU also is a high frequency signal, it is therefore not possible to differentiate between the scenery and the fingerprint. This leads to artifacts and a deterioration of the PCE value. To somewhat mitigate this problem, our research shows that using grey and white QR-Codes for fingerprint extraction improves the method. For our proof of concept we used a Huawei P20 Pro. The resulting PCE values for images taken by the same camera were five times higher when using grey QR-Codes compared to black QR-Codes. Therefore, we can conclude that grey and white QR codes, while, still being accepted by QR code readers, result in PCE values better suited for authentication, probably as even the grey area still produces enough light for the sensor to add a strong PRNU pattern. The contrast and therefore the edges are not that strong, compared to black and white, and the dark areas still contain light, thus resulting in an increased PRNU.

5.3 Fingerprints Require Multiple Reference Images

The user, in the scheme proposed by Ba et al. [4], only requires the user to record a single image with a smartphone. The image then is sent to the server, which extracts the fingerprint from this single image and stores it for later authentication processes with challenge images.

Assuming that images of smartphone cameras are well fitted for PRNU extraction, we have doubts on the assumption that one image is sufficient for creating a high quality reference fingerprint. To our knowledge, this opposes the prevailing opinion in literature. One of the key assumptions in extracting PRNU is that other noises, like shot noise or random noise, can be averaged out over a corpus of N images, as Equation 3 states. In original works [38] up to 50 images were suggested. This has been lowered to 20 by some authors, eg. Zhang et Zhang [61]. With only one image at hand, other sources of noise will essentially persist throughout fingerprint extraction. Our experiments discussed in Section 5.1 suggest that scene content plays a big factor, may it be, because the overall images are too dark, or because they contain too much high frequent content, in textured areas. To leverage this, some authors even suggest to obtain the fingerprint exclusively from images without texture [21]. In our experiments we relaxed this condition and considered also images with texture for extraction, and still got reasonable good results. Nevertheless, in an authentication scheme, may it be for online banking, we argue that no risk should be taken and a solid number of high quality images with little standard deviation should be used for creating the initial fingerprint.

6 Practical Attacks

First, we will introduce the assumed threat model, then we will present three attacks on the ABC scheme, as we implemented it based on [4].

6.1 Threat Model

The attacker assumed by Ba *et al.* [4] is, in our understanding, rather weak. The threats do not factor in malware, which could take photos on the phone and learn fingerprints with ease, even though malware on smartphones is rather common [39,47]. The adversary may gain access to public photos of the victim, is able to sniff the communication channel between victim and verifier, and can fake a display the victim takes images of (phishing). It includes replay attacks, fingerprint forgery attacks (adding a learned fingerprint to photos not taken with the original smartphone) as well as *Man in the*

Middle (MitM) scenarios. The goal of the adversary lies in convincing the victim to authorize a malicious request, or to trick the verifier into authenticating the adversary's request.

The scheme by Valsesia *et al.* [56] relies on an attacker that will not be able to capture RAW images. As this even rules out unprivileged malware with camera access, in our further discussion we focus on the scheme of Ba *et al.* [4].

We assume the following threats.

6.2 Fingerprint Forgery

A well-known approach to tackle fingerprint forgery is to use the triangle test [37]. Ba *et al.* [4] rightfully desist from using this technique on the base, that it comes with high computational effort. A database with all public images for each user would have to be curated and updated constantly. They present a novel technique based on the recording of two images. During the authentication process, the user uploads not one but two images of two different QR-Codes. If an adversary applies the victim's fingerprint on both photos, they still carry their real fingerprint. Hence, their extracted fingerprints have a higher correlation than the comparison with the reference fingerprint stored on the server. By this, Ba *et al.* [4] detect an additional fingerprint on an image. Even if the attacker were not able to erase fingerprints completely, for which multiple ways have been proposed, for example by Bonettini *et al.* [7], attacks are still possible. Using two different smartphones, a single dual camera phone or even just rotating the phone by 90 degrees breaks this defense. For our proof of concept attack, we cropped two, mostly disjoint (they still need to contain the QR-Code), regions from the same image sensor. Different parts of the same sensor have a different PRNU as well.

6.3 Image Reuse

Despite the addition of QR-Codes, we are still able to reuse any images from the victim's phone, thanks to the noisiness of QR-Codes. Due to this high noise, large black areas and the many high-frequency edges of the scenery, as discussed in Section 5, our tests show QR-Codes are almost ignored during PCE calculation. We can use this fact to our favor. After taking a photo of the QR-Code with a second smartphone, including the probe signal (see Section 6.4), we crop the QR-Code from the image and paste it into the victim's image. The QR-Code has to be as small as possible but still large enough to be accepted by the server and containing enough of the probe signal. Because of the small proportion of the QR-Code compared to the whole image, the area containing the fingerprint of the victim is still very big, with the QR-Code having almost no impact. The correlation of the manipulated challenge image and the one stored on the server is high. e were able to create false positive results against our proof of concept implementation this way. It is hard to defend against this attack without adding some sort of image recognition or different forgery detection: if the photographed QR-Code needs to be too big, its noise can result in a high false negative rate, as the fingerprint may be blocked. Addressing this vulnerability, we present a countermeasure in Section 7.3.

6.4 Probe Signal Preserving Fingerprint Removal

In the Section 6.2, we showed that two parts of the image are already enough to forge a foreign fingerprint without failing the authentication process presented by Ba *et al.* [4]. A more elaborate attack first erases the fingerprint of the adversary's camera to then add the fingerprint of the victim. To prevent this, Ba *et al.* [4] introduce an additional probe signal. They specify it as white gaussian noise. This probe signal is applied to the QR-Code displayed by the terminal.

With a black-and-white QR-Code image as base, we assume the signal to be additive. In Figure 3 the resulting layers of noise are depicted. The QR-Code represents the scenery, perceived by the user, containing a lot of high frequency components itself. The probe signal symbolizes the additionally added white gaussian noise. The authors claim that the noise with a standard deviation of 5 is of the same variance as a fingerprint. The top layer represents the fingerprint of the camera sensor, the PRNU. It is added during the recording phase of the image by the varying sensitivity of the camera sensor. Our large-scale test indicates that the standard deviation of the PRNU is a lot lower than 5. A deviation of 5 would result in images with visual noise.

The idea of the probe signal is, that it will be erased if the adversary tries to erase her own fingerprint from the image by using a low-pass filter. If the probe signal is not contained in the challenge image, which is compared to the reference fingerprint and to which the probe signal was added as well, the correlation decreases. By this, Ba *et al.* [4] claim to detect any erasing of a fingerprint from an image.

Our analysis shows that the challenge images can be analyzed with a high-pass filter, to find all noises — both the noise due to the probe signal, as well as the PRNU noise. We can further estimate the fingerprint of the device, for example from other pictures taken on it. With that, we are able to dissect the probe signal from the inherent fingerprint, and clean the attack. This works even if we assume that both signals are of similar strengths. However, on our setup, the noise of the PRNU is much weaker. So simply filtering strong perturbations, with respect to the fingerprinting, suffices. The underlying problem here is, that the probe signal can be considered as a watermark, which is by design detectable, even if the detection should be done on the server.

7 Beyond Camera Fingerprinting Authentication

Besides improving the PRNU extraction in general to improve the significance of the results as proposed in 4 one can undertake further steps. In this section, we propose a transaction security scheme that takes the current state-of-the-art into consideration and improve upon the drawbacks.

7.1 General Camera Fingerprint Improvements

Any fingerprint can be learned by attackers, then be replayed or dropped directly in authenticating code. This is still possible — although harder — after applying probe signals and checking multiple uploaded images, as proposed by Ba *et al.* [4]. Still, attackers crawling images or malware on the phone, can learn the fingerprint and forge correct images to authenticate. Lukás *et al.* [38] note that it is "unlikely that there exists a numerical identification characteristic computed from digital images that could not be compromised by a sufficiently sophisticated opponent". Malware on the defending phone can simply capture 20 images to gain the statistical pattern and then learn the fingerprint. The malware then can create a photo that passes any other requirements, for example including the correct QR-Code for authentication. It can also remove any fingerprints and apply the learned fingerprint on top. However, defenses against fingerprint forgery (and counterattacks) are an ongoing research topic for image forensics. As it is common in the field of security, we are subjected to an arms race between attacker and defender. Trivial anti-forensics for camera fingerprinting have been broken early, for example by the so called *triangle test* proposed in 2010 by Goljan et al. [20], which can be attacked yet again [40]. The arms race in camera fingerprint forgery detection is ongoing with new defensive methods being proposed more than ten years after the paper by Goljan *et al.* [55]. As an alternative to ABC, Ba *et al.* [3] recently proposed CIM, a scheme that combines additional noise in multiple burst photos, as well as accelerometer fingerprinting for a more resistant fingerprint. Additional sensor inputs during photography could even increase the attacker's effort in cloning in the future. For applications that need very high security standards, an authentication scheme could go beyond checking the fingerprint for correctness and also detect possible forgeries on the image itself [10].

7.2 Trusted Secure Camera

Our study shows that the camera fingerprint poses little real world use and leaves room for attackers. The current mitigation approaches proposed by Ba *et al.* [4] can be circumvented (see Section 5) and further defenses can only lead to an arms race that the defender will lose. However, device authentication through images could substantially be increased if the camera output could not be forged. As a matter of fact, cameras that introduce unforgeable cryptographic assets in a captured image have been proposed before, for example *PhotoProof* by Naveh and Tromer [44]. If smartphones were to implement such a secure camera module, secure image based transaction schemes can be implemented. These cameras, however, are state-of-the-art research objects, expensive, and will hardly ever make it into smartphone hardware.

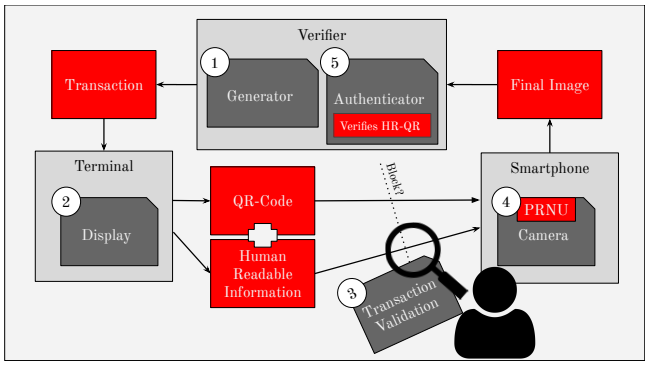


Figure 12: Illustrated scheme of an updated authentication protocol. The user can always cancel the ongoing transaction. The terminal and smartphone can both be adversary-controlled.

On the other hand, trusted execution environments are steadily gaining ground and modules exist in most mobile devices today. Direct access to external hardware is also already feasible, e.g. in ARM TrustZone [2]. Ultimately, by hooking up trusted software to the camera directly, a system for unforgeable camera fingerprints can be built as long as an attacker cannot gain access to the TEE. Images (or videos) taken by this *Trusted Secure Camera* could embed watermarks or signatures like *PhotoProof* [44] inside the image. Another approach is to attach a signature in the metadata before the image leaves the trusted and attested code. An ideal solution only allows write access to the memory region the camera data is written to by the hardware and secure camera stack in trusted code. This *Trusted Secure Camera* would render normal camera fingerprinting redundant and help secure transactions. Since our evaluation shows that almost all images are traceable, drawbacks in privacy are minimal and the feature could also be disabled by users if needed.

7.3 Securing Transactions

In this section we propose a forgery-resistant scheme for secure transactions. Whereas for authentication purposes the methods are flawed, it poses a real benefit for transaction signing. For this, we have extended our camera fingerprinting app built for the large scale study with a banking-like scenario.

In our proof of concept application, our signing app uses optical character recognition (OCR) with a QR-Code as redundancy to ensure that the user really saw the correct transaction contents.

The computer can display the TAN information directly instead of relying on a phone. If malware fakes the on-screen info, the user will immediately see it and is able to react. As depicted in Figure 12, if the content on the display does not match what the user wants to sign, a user can easily identify the faults. The user will stop the transaction automatically if the displayed data is wrong. The transaction does not get signed. The signature is linked to a user's intent. This results in cancellation due to not using the second factor at all. We can achieve this by linking the data displayed to the user in a human and machine readable format. The camera fingerprint then makes sure the user saw the photographed content at least once.

A similar solution to this kind of simple human readable QR (*HR-QR*), has been proposed by Millican and Stanjano [42]. Their so-called *SAVVIcode* combines a QR-Code with a more machine-friendly text above, making it even easier to process without errors. As the codes, just as QR codes, are very noisy, they should be *grey* and white.The key components are as follows:

1. HR-QR: The user needs to know what she scans. The only way to do this is to show the contents in clear text. This human readable QR-Code is then checked by attested code.

2. Authenticity is checked by extracting the PRNU: The same attested code needs to check if the correct camera took the photo at hand, using the camera fingerprint. This will lower the risk that attackers directly insert images in the black box and run the authentication code without the user's knowledge. At the same time, an air gap is enforced, making the phone a true second factor.

The two key components are

- Matrix code is human-readable: To counter MitM schemes, the user needs to know what she scans. The best way to do this is to display the contents in clear text. We add OCR to a usual transaction scheme [41]. Since OCR software is not fail-proof enough, the content is sent along in a separate QR-code or other barcode or matrix code, like [8]. Trusted code, either in the TEE or on the server then checks if both parts match.

- Authenticity is checked by extracting PRNU or Fingerprint: The same, attested, code needs to check if the correct camera took the photo at hand, using the PRNU. This will lower the risk for attackers to directly insert images in the black-box and running the authentication code without the user's knowledge. At the same time, an air gap is enforced.

The camera has the merit, especially in the case of an *HR-QR*, that it contains additional information that can be linked to a user intent. The user will only scan the screen if she wants to sign the authentication. For this study, we implemented an HR-QR scheme. Details are discussed in Appendix B.

Everts *et al.* [16] note on their implementation of smartphone authentication scheme, that a secure element won't prevent malware on the phone to use the credentials. We argue that, using the camera fingerprint as input, this statement no longer holds true. Instead of feeding data to the secure element, the attacker now has to feed an image that satisfies all imposed restrictions. For the future we plan to replace PRNU with a secure camera, as discussed in 7.2. That means, as long as the trusted execution environment can be considered safe, HR-QR is then a secure transaction system, even with compromised smartphones and displays.

8 Conclusion

The attacks on current authentication schemes, presented in this paper, show the shortcomings of smartphone camera fingerprinting We show attacks against defenses discussed by a major authentication protocol leveraging camera fingerprinting, allowing us to impersonate the user, even with a single photo. In our comprehensive large-scale study, collecting 56,630 images, we can substantiate that all current phone cameras work well as PUF, essentially pinning photos to unique devices. After thorough vetting, we can conclude that, in direct comparison with other authentication schemes, the added complexity of camera-fingerprinting based authentication does not add substantial security benefit over secrets stored in the phone's memory. While it does add benefits over weak schemes, such as SMS based authentication [43], possible alternatives exist: pushing an one-time password (OTP) to the phone at the time of authentication through a secured internet connection, through bluetooth or even via sound. In all threat models discussed in Section 6.1, normal app authentication schemes are either good enough, i.e., if there is no privileged access to local contents on the phone, secrets can be stored there — or the camera fingerprinting authentication schemes also fail.

Just as it is the case with fingerprinting in general, it might pose an additional hurdle for attackers, however, the knowledge how to calculate camera fingerprints is readily available. Apart from the special case where the defendant never discloses RAW images and uses them for authentication only, as proposed by Valsesia *et al.* [56], the security of camera fingerprinting is further damped by publicly available images. We do see a benefit for transactions as the photographed screen can replace a secure display. For the future, signing photos in the trusted execution environment of a phone could replace PRNU use-cases completely and provide security benefits.

We hope to enable fruitful future research based on our findings as well as the code published as part of this research.

Acknowledgments

The authors would like to thank Federico Maggi, Stefano Zanero, Tilo Müller, and Christian Riess for feedback and support.

Availability

The PRNU implementation in C++/OpenCV will be open sourced upon publication. Additional source code needed to reproduce results of this study, including our proof of implementation for authentication, as well as evaluation code, will be made available.

References

[1] G. Acar, C. Eubank, S. Englehardt, M. Juárez, A. Narayanan, and C. Díaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, G. Ahn, M. Yung, and N. Li, Eds. ACM, 2014, pp. 674–689. [Online]. Available: https://doi.org/10.1145/2660267.2660347

[2] T. Alves and D. Felton, "Trustzone: Integrated hardware and software security," *ARM white paper*, vol. 3, no. 4, pp. 18–24, 2004.

[3] Z. Ba, Z. Qin, X. Fu, and K. Ren, "Cim: Camera in motion for smartphone authentication," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 11, pp. 2987–3002, Nov 2019.

[4] Z. Ba, S. Piao, X. Fu, D. Koutsonikolas, A. Mohaisen, and K. Ren, "Abc: Enabling smartphone authentication with built-in camera," in *25th Annual Network and Distributed System Security Symposium, NDSS, 2018, San Diego, California, USA, February 18 - 21, 2018*, 2018. [Online]. Available: https://doi.org/10.14722/ndss.2018.23099

[5] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang, "Silentsense: Silent user identification via touch and movement behavioral biometrics," in *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking*, ser. MobiCom '13. New York, NY, USA: ACM, 2013, pp. 187–190. [Online]. Available: http://doi.acm.org/10.1145/2500423.2504572

[6] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," *CoRR*, vol. abs/1408.1416, 2014. [Online]. Available: http://arxiv.org/abs/1408.1416

[7] N. Bonettini, L. Bondi, D. Güera, S. Mandelli, P. Bestagini, S. Tubaro, and E. J. Delp, "Fooling prnu-based detectors through convolutional neural networks," in *2018 26th European Signal Processing Conference (EUSIPCO)*, Sep. 2018, pp. 957–961.

[8] C. Chen, W. Huang, B. Zhou, C. Liu, and W. H. Mow, "Picode: A new picture-embedding 2d barcode," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3444–3458, Aug 2016.

[9] M. Chen, J. J. Fridrich, M. Goljan, and J. Lukás, "Determining image origin and integrity using sensor noise," *IEEE Trans. Information Forensics and Security*, vol. 3, no. 1, pp. 74–90, 2008.

[10] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pp. 1841–1854, Dec 2012.

[11] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "Raise: A raw images dataset for digital image forensics," in *Proceedings of the 6th ACM Multimedia Systems Conference*. ACM, 2015, pp. 219–224.

[12] A. Das, N. Borisov, and M. Caesar, "Do you hear what I hear?: Fingerprinting smart devices through embedded acoustic components," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. ACM, 2014, pp. 441–452. [Online]. Available: http://doi.acm.org/10.1145/2660267.2660325

[13] ——, "Tracking mobile web users through motion sensors: Attacks and defenses," in *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*, 2016.

[14] P. Eckersley, "How unique is your web browser?" in *Privacy Enhancing Technologies, 10th International Symposium, PETS 2010, Berlin, Germany, July 21-23, 2010. Proceedings*, ser. Lecture Notes in Computer Science, M. J. Atallah and N. J. Hopper, Eds., vol. 6205. Springer, 2010, pp. 1–18. [Online]. Available: https://doi.org/10.1007/978-3-642-14527-8_1

[15] J. Entrieri and M. Kirchner, "Patch-based desynchronization of digital camera sensor fingerprints," in *Media Watermarking, Security, and Forensics 2016, San Francisco, California, USA, February 14-18, 2016*, 2016, pp. 1–9. [Online]. Available: http://ist.publisher.ingentaconnect.com/contentone/ist/ei/2016/00002016/00000008/art00022

[16] M. Everts, J.-H. Hoepman, and J. Siljee, "Ubikima: Ubiquitous authentication using a smartphone, migrating from passwords to strong cryptography," in *Proceedings of the 2013 ACM Workshop on Digital Identity Management*, ser. DIM '13. New York, NY, USA: ACM, 2013, pp. 19–24. [Online]. Available: http://doi.acm.org/10.1145/2517881.2517885

[17] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song, "Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 136–148, Jan 2013.

[18] T. Gloe and R. Böhme, "The 'Dresden Image Database' for benchmarking digital image forensics," in *Proceedings of the 25th Symposium On Applied Computing (ACM SAC 2010)*, vol. 2, 2010, pp. 1585–1591.

[19] M. Goljan and J. Fridrich, "Determining approximate age of digital images using sensor defects," in *SPIE Conference on Media Watermarking, Security, and Forensics*, 2011.

[20] M. Goljan, J. Fridrich, and M. Chen, "Sensor noise camera identification: countering counter-forensics," vol. 7541, 2010, pp. 75 410S–75 410S–12. [Online]. Available: http://dx.doi.org/10.1117/12.839055

[21] M. Goljan, J. Fridrich, and T. Filler, "Large scale test of sensor fingerprint camera identification," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2009, pp. 72 540I–72 540I.

[22] M. Goljan and J. J. Fridrich, "Camera identification from cropped and scaled images," in *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, San Jose, CA, USA, January 27, 2008*, ser. SPIE Proceedings, E. J. D. III, P. W. Wong, J. Dittmann, and N. D. Memon, Eds., vol. 6819. SPIE, 2008, p. 68190E.

[23] M. Goljan, J. J. Fridrich, and M. Chen, "Defending against fingerprint-copy attack in sensor-based camera identification," *IEEE Trans. Information Forensics and Security*, vol. 6, no. 1, pp. 227–236, March 2011. [Online]. Available: https://doi.org/10.1109/TIFS.2010.2099220

[24] M. Goljan, J. J. Fridrich, and T. Filler, "Large scale test of sensor fingerprint camera identification," in *Media Forensics and Security I, part of the IS&T-SPIE Electronic Imaging Symposium, San Jose, CA, USA, January 19-21, 2009, Proceedings*, 2009, p. 72540I. [Online]. Available: https://doi.org/10.1117/12.805701

[25] N. Z. Gong, M. Payer, R. Moazzezi, and M. Frank, "Forgery-resistant touch-based authentication on mobile devices," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM, 2016, pp. 499–510.

[26] H. Gueddah, A. Yousfi, and M. Belkasmi, "The filtered combination of the weighted edit distance and the jaro-winkler distance to improve spellchecking arabic texts," in *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*, Nov 2015, pp. 1–6.

[27] V. Haupert, D. Maier, and T. Müller, "Paying the price for disruption: How a fintech allowed account takeover," in *ROOTS*, 2017.

[28] V. Haupert, D. Maier, N. Schneider, J. Kirsch, and T. Müller, "Honey, I shrunk your app security: The state of android app hardening," in *Detection of Intrusions and Malware, and Vulnerability Assessment - 15th International Conference, DIMVA 2018, Saclay, France, June 28-29, 2018, Proceedings*, ser. Lecture Notes in Computer Science, C. Giuffrida, S. Bardin, and G. Blanc, Eds., vol. 10885. Springer, 2018, pp. 69–91. [Online]. Available: https://doi.org/10.1007/978-3-319-93411-2_4

[29] V. Haupert and T. Müller, "(in)security of app-based TAN methods in online banking," Freidrich-Alexander-Universität Erlangen-Nürnberg, Tech. Rep., December 2015. [Online]. Available: https://www1.cs.fau.de/content/insecurity-app-based-tan-methods-online-banking

[30] ——, "On app-based matrix code authentication in online banking," Freidrich-Alexander-Universität Erlangen-Nürnberg, Tech. Rep., 2016.

[31] T. Hupperich, H. Hosseini, and T. Holz, *Leveraging Sensor Fingerprinting for Mobile Device Authentication*. Cham: Springer International Publishing, 2016, pp. 377–396. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-40667-1_19

[32] T. Hupperich, D. Maiorca, M. Kührer, T. Holz, and G. Giacinto, "On the robustness of mobile device fingerprinting: Can mobile users escape modern web-tracking mechanisms?" in *Proceedings of the 31st Annual Computer Security Applications Conference*, ser. ACSAC 2015. New York, NY, USA: ACM, 2015, pp. 191–200. [Online]. Available: http://doi.acm.org/10.1145/2818000.2818032

[33] A. Karakuçük and A. E. Dirik, "Adaptive photo-response non-uniformity noise removal against image

source attribution," *Digital Investigation: The International Journal of Digital Forensics and Incident Response*, vol. 12, 03 2015.

[34] Y. Kim and Y. Lee, "Campuf: Physically unclonable function based on cmos image sensor fixed pattern noise," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3195970.3196005

[35] A. Kurtz, H. Gascon, T. Becker, K. Rieck, and F. Freiling, "Fingerprinting mobile devices using personalized configurations," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, pp. 4–19, 2016.

[36] C. T. Li and R. Satta, "On the location-dependent quality of the sensor pattern noise and its implication in multimedia forensics," in *4th International Conference on Imaging for Crime Detection and Prevention 2011 (ICDP 2011)*, Nov 2011, pp. 1–6.

[37] H. Li, W. Luo, Q. Rao, and J. Huang, "Anti-forensics of camera identification and the triangle test by improved fingerprint-copy attack," *CoRR*, vol. abs/1707.07795, 2017. [Online]. Available: http://arxiv.org/abs/1707.07795

[38] J. Lukás, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 205–214, June 2006.

[39] D. Maier, M. Protsenko, and T. Müller, "A game of droid and mouse: The threat of split-personality malware on Android," *Computers & Security*, vol. 54, pp. 2–15, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2015.05.001

[40] F. Marra, F. Roli, D. Cozzolino, C. Sansone, and L. Verdoliva, "Attacking the triangle test in sensor-based camera identification," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 5307–5311.

[41] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-is-believing: using camera phones for human-verifiable authentication," in *2005 IEEE Symposium on Security and Privacy (S P'05)*, May 2005, pp. 110–124.

[42] J. Millican and F. Stajano, *SAVVIcode: Preventing Mafia Attacks on Visual Code Authentication Schemes (Short Paper)*. Cham: Springer International Publishing, 2015, pp. 146–152. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-24192-0_10

[43] C. Mulliner, R. Borgaonkar, P. Stewin, and J.-P. Seifert, "SMS-based one-time passwords: attacks and defense," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer Berlin Heidelberg, 2013, pp. 150–159.

[44] A. Naveh and E. Tromer, "Photoproof: Cryptographic image authentication for any set of permissible transformations," in *2016 IEEE Symposium on Security and Privacy (SP)*, May 2016, pp. 255–271.

[45] C. Nickel, T. Wirtl, and C. Busch, "Authentication of smartphone users based on the way they walk using k-NN algorithm," in *2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, July 2012, pp. 16–20.

[46] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," in *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*. IEEE Computer Society, 2013, pp. 541–555. [Online]. Available: https://doi.org/10.1109/SP.2013.43

[47] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "TESSERACT: Eliminating experimental bias in malware classification across space and time," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 729–746. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/pendlebury

[48] E. Quiring and M. Kirchner, "Fragile sensor fingerprint camera identification," in *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*, Nov 2015, pp. 1–6.

[49] E. Quiring, M. Kirchner, and K. Rieck, "On the security and applicability of fragile camera fingerprints," in *Computer Security - ESORICS 2019 - 24th European Symposium on Research in Computer Security, Luxembourg, September 23-27, 2019, Proceedings, Part I*, ser. Lecture Notes in Computer Science, K. Sako, S. Schneider, and P. Y. A. Ryan, Eds., vol. 11735. Springer, 2019, pp. 450–470. [Online]. Available: https://doi.org/10.1007/978-3-030-29959-0_22

[50] R. Ramanath, W. E. Snyder, Y. Yoo, and M. S. Drew, "Color image processing pipeline," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 34–43, Jan 2005.

[51] B. Reaves, L. Blue, H. Abdullah, L. Vargas, P. Traynor, and T. Shrimpton, "Authenticall: Efficient identity and content authentication for phone calls," in *26th USENIX Security Symposium, USENIX*

Security 2017, Vancouver, BC, Canada, August 16-18, 2017., 2017, pp. 575–592. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/reaves

[52] D. Shullani, M. Fontani, M. Iuliani, O. Al Shaya, and A. Piva, "Vision: a video and image dataset for source identification," *EURASIP Journal on Information Security*, vol. 2017, no. 1, p. 15, 2017.

[53] M. Simkin, D. Schröder, A. Bulling, and M. Fritz, *Ubic: Bridging the Gap between Digital Cryptography and the Physical World*. Cham: Springer International Publishing, 2014, pp. 56–75. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11203-9_4

[54] T. Stöber, M. Frank, J. B. Schmitt, and I. Martinovic, "Who do you sync you are?: smartphone fingerprinting via application behaviour," in *Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, WISEC'13, Budapest, Hungary, April 17-19, 2013*, ser. WiSec '13. New York, NY, USA: ACM, 2013, pp. 7–12. [Online]. Available: http://doi.acm.org/10.1145/2462096.2462099

[55] S. Taspinar, M. Mohanty, and N. Memon, "PRNU based source attribution with a collection of seam-carved images," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 156–160.

[56] D. Valsesia, G. Coluccia, T. Bianchi, and E. Magli, "User authentication via prnu-based physical unclonable functions," *IEEE Trans. Information Forensics and Security*, vol. 12, no. 8, pp. 1941–1956, 2017. [Online]. Available: https://doi.org/10.1109/TIFS.2017.2697402

[57] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "FP-STALKER: tracking browser fingerprint evolutions," in *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*. IEEE Computer Society, 2018, pp. 728–741. [Online]. Available: https://doi.org/10.1109/SP.2018.00008

[58] B. Wronski, I. Garcia-Dorado, M. Ernst, D. Kelly, M. Krainin, C.-K. Liang, M. Levoy, and P. Milanfar, "Handheld multi-frame super-resolution," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 28:1–28:18, Jul. 2019. [Online]. Available: http://doi.acm.org/10.1145/3306346.3323024

[59] W. Wu, J. Wu, Y. Wang, Z. Ling, and M. Yang, "Efficient fingerprinting-based android device identification with zero-permission identifiers," *IEEE Access*, vol. 4, pp. 8073–8083, 2016. [Online]. Available: https://doi.org/10.1109/ACCESS.2016.2626395

[60] C. Yue, "Sensor-based mobile web fingerprinting and cross-site input inference attacks," in *Security and Privacy Workshops (SPW), 2016 IEEE*. IEEE, 2016, pp. 241–244.

[61] C. Zhang and H. Zhang, "Exposing digital image forgeries by using canonical correlation analysis," in *Pattern Recognition (ICPR), 2010 20th International Conference on*, Aug 2010, pp. 838–841.

[62] Z. Zhou, W. Diao, X. Liu, and K. Zhang, "Acoustic fingerprinting revisited: Generate stable device ID stealthy with inaudible sound," *CoRR*, vol. abs/1407.0803, 2014. [Online]. Available: http://arxiv.org/abs/1407.0803

A The PRNU

In this part of the Appendix, we derive the mathematical foundation of PRNU, as it is well established in literature, most prominently in [38] and [21]. Furthermore, we explain the standard way to compare two noise patterns to determine if they were made by the same camera.

The image I, read out at the camera-interface, differs from the perfect ideal image I_0 that might exist at the sensor without any imperfections or perturbations. First i.i.d. sources like shot-noise or dark-current-noise and random-noise are additionally in the image. These noises are summarized in θ. Then, the described PRNU as multiplicative source of noise is casted on the image, expressed in the following equation with K. The imaging formation pipeline catering for those additional corruptions is given by

$$I = I_0 + I_0 K + \theta \ . \tag{1}$$

The mentioned additive noise sources θ can be averaged out over multiple images. However, the multiplicative noise K, the PRNU we are seeking for, is not a random noise for a given pixel at position (x,y). By averaging a number of photos and filtering them to reduce other noise, the PRNU can be extracted.

The procedure introduced by Lukás *et al.* [38] starts with suppressing the image content of the single images, by taking the difference of the low-pass filtered version of the image and the image itself. This is described with

$$W_I = I - F(I) \ , \tag{2}$$

where F could any arbitrary low-pass filter, in practice, however usually a wavelet filter is used.

The residuals of this filter operation are weighted by the pixel intensities and averaged over N samples like

$$\hat{K} = \frac{\sum_{i=1}^{N} W_I^i I^i}{\sum_{i=1}^{N} (I^i)^2} \ . \tag{3}$$

The superscript i in this equation is an image index going from 1 to N. Also it is worth pointing out, that the product

$W^i I^i$ and similar operations are element-wise, e.g. at pixel positions (x, y) and not regular matrix multiplications. Finally, we are left with the estimated PRNU fingerprint $\hat{K}$.

B HR-QR Implementation

The apps developed for this paper also include a practical implementation of the HR-QR scheme. For the actual implementation of an human-readable QR code, we needed to have an error resilient solution. OCR results are never perfect and other values may falsely be factored into the comparison. With the knowledge that this allows for (easier to spot) attacks by exchanging single numbers or letters, we suggest the use of the *Jaro-Winkler distance* D_{jw}, as a metric to check string similarities [26]. Given the strings of the text X and a string of the QR Code Y, the metric is calculated with

$$D_{\text{jw}}(X,\, Y) = D_j(X,\, Y) + l \cdot p \cdot (1 - D_j(X,\, Y)) \quad , \qquad (5)$$

where p is a prefix coefficient, which promotes the chains with the longest common prefix, and l being the common prefix between X and Y. We use the default value of 0.1 for p. Further, the distances in D_{j} are calculated by

$$D_{\text{j}}(X,\, Y) = \frac{1}{3}\left(\frac{c}{|X|} + \frac{c}{|Y|} + \frac{c-t}{c}\right) \quad , \qquad (6)$$

with c the number of characters that match between X and Y and t the number of transpositions.

As more text is potentially in the users' view, for our proof of concept we extracted the distance for substrings, as shown in algorithm 1. A future implementation might instead refrain to an algorithm already fit for substring matching, like an *Smith-Waterman algorithm* or use a way to transmit the human-readable part intact while still being easy to read, similar to the proposals of Millican and Stajano [42] and Simkin *et al.* [53].

Algorithm 1 MaxJaro

```
1: procedure MAXJARO(check, text)
2:     maxSimilarity ← JARONWINKLERDISTANCE(check, text)
3:     for each substr of text where LEN(substr) = LEN(check) or LEN(check) + 1 do
4:         similarity ← JARONWINKLERDISTANCE(check, substr)
5:         maxSimilarity ← MAX(max, similarity)
6:     end for
7: return maxSimilarity
8: end procedure
```