SCREENID: Enhancing QRCode Security by Fingerprinting Screens

***Abstract*—Quick response (QR) codes have been widely used in mobile applications due to its convenience and the pervasive built-in cameras on smartphones. Recently, however, attacks against QR codes have been reported that attackers can capture a QR code of the victim and replay it to achieve a fraudulent transaction or intercept private information, just before the original QR code is scanned. In this study, we enhance the security of a QR code by identifying its authenticity. We propose SCREENID, which embeds a QR code with information of the screen which displays it, thereby the QR code can reveal whether it is reproduced by an adversary or not. In SCREENID, PWM frequency of screens is exploited as the unique screen fingerprint. To improve the estimation accuracy of PWM frequency, SCREENID incorporates a model for the interaction between the camera and screen in the temporal and spatial domains. Extensive experiments demonstrate that SCREENID can differentiate screens of different models, types, and manufacturers, thus improve the security of QR codes.**

Index Terms*—screen-camera communication; secure QR code;

I. INTRODUCTION

Quick response (QR) codes are barcodes comprising white and black blocks. In the past years, with the pervasive built-in cameras on smartphones, QR codes have been widely adopted in mobile applications such as communication, payment, etc. Especially, for mobile payment scenarios, the QR code system (hereafter we name a QR code system as QRCode) is almost a standard module for service providers such as AliPay, WeChat, PayPal, etc [1], [38]. Users just need to show their QR codes on smartphones for a quick transaction, which provides convenient and friendly experience.

The QRCode system works in a straightforward way. As shown in the flowchart in Fig. 1, payment transactions begin with the generation of a legal QR code, which includes the ID of the user in the application, e.g., the AliPay account, a time-stamp, and the secret transaction information in the form of an encrypted token. The user then presents the QR code to the cashier to have it scanned into the transaction system, together with other transaction information such as total amount and currency type. On the server side, the merchant extracts the token from QR code, obtains the user ID from the token and then accesses the database maintained by the company to retrieve the stored secret of the payer. This secret is then converted into a new token for verification. As long as the token is deemed valid, the transaction proceeds.

However, the above-mentioned transaction process is far from secure. Recently, researchers have reported that a QR-Code system is susceptible to the Synchronized Token Lifting and Spending (STLS) attack and other similar attacks [2]. In this attack, the adversary first acquires an image of the QR code displayed on the victim's device [18], when the victim is showing the QR code to the cashier, for example. Then the adversary replays the stolen QR code for another transaction, which we call fraudulent transaction. To ensure the success of such an attack, the adversary should also finish the transaction before the legitimate scanning process from the cashier.

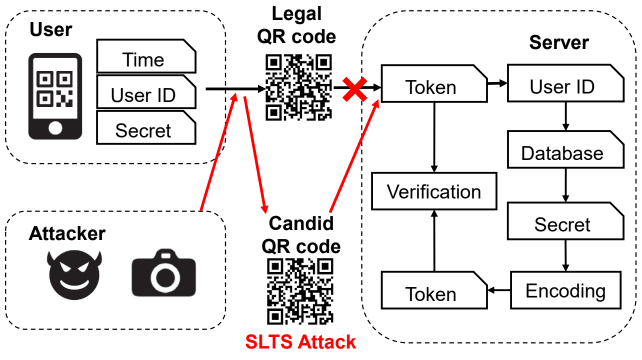


Fig. 1. A flowchart of a typical QRCode system for mobile transaction. Note that an adversary can capture the victim's QR code and then replay it for a fraudulent transaction.

To against the STLS attack, researchers have proposed a lot of solutions. Most existing protection schemes aim at safeguarding the QRCode system by inserting codewords [6], [7], [23] or concealing the original QR codes using visually cryptographic techniques [5], [13], [24], [29], [37], [39]. Unfortunately, attackers can still succeed in replaying the stolen QR code as they do not need to decrypt the message embedded in the QR code.

In this paper, we sought to enhance the security of the QRCode system by identifying the authenticity of a QR code. We propose SCREENID, which embeds a QR code with information of the screen that is displaying it, thereby the generated QR code can reveal whether the screen is corresponding to it. In SCREENID, we utilize the pulse width modulation (PWM) frequency of screens as the unique screen fingerprint. From our experiments, it can hardly find any two smartphones with the same PWM frequency. PWM frequency makes a good candidate for screen fingerprint for the reason that it is adjusted to different value by screen manufacturers. Even for the same manufacture, the PWM frequency shows variances due to the variations in the manufacturing processes. In Section IV, we show that PWM frequencies are different even for phones of the same model. Although the results cannot fully guarantee the security of QRCode system, the probability that any two mobile phones have the same frequency is negligible, therefore SCREENID can successfully enhance the security of QRCode system.

On the receiver side, e.g., the camera in a cashier, the PWM frequency is measured by the rolling shutter effect observed on the cashier's camera. The QR code is legal and the transaction proceeds only if the PWM frequency embedded in the QR code and the one measured from the screen are closely match.

In realizing the SCREENID, we overcome several challenges. First, the interaction between screen and camera involves many influencing factors, such as PWM dimming and screen refresh rate, etc. Obtaining an accurate frequency estimation requires that the interactions among these factors be modeled in the temporal as well as the spatial domains (Sec. VI). Second, it was necessary to compensate for variations of distance and angle between the display device and the scanning device, e.g., a camera (Sec. VII-F). Third, we found that the differences in PWM frequency among smartphones were really small (e.g., $0.1Hz$), particularly when dealing with smartphones of the same model. Thus, we need to increase the frequency resolution, while minimizing the number of captured images (Sec. VII-G).

We verified the efficacy of a SCREENID prototype using 50 screensunder a variety of camera settings and environmental conditions. The usability of the proposed system was also verified in experiments involving ten participants of various ages. We summarize the contribution of this paper as follows:

- We demonstrated the feasibility of using PWM frequency of screens for fingerprint, and the fingerprint can be embedded in to a QR code to check its authenticity. From our dataset, 99.3% pairwise frequency differences of screens are larger than $0.1Hz$.
- We proposed SCREENID, which incurs no additional hardware for the QRCode system and has no requirements for user behavior. SCREENID can be implemented via a simple software update.
- We proposed to model the interaction between the camera and the screen in both temporal and spatial domains and achieved high estimation accuracy of PWM frequency, i.e., within $0.03Hz$.
- We conducted exhaustive experiments and demonstrated the efficacy of the SCREENID system under a variety of operating conditions. The evaluation shows the identifiable success rate (i.e. True Positive Rate) is above 94.3% and the wrongly accepted rate (i.e. False Positive Rate) against attack attempts is below 0.8%.

II. Related Work

A. Visible Light Communication

Many works exploited the interaction between the light and camera. The visual microphone [10] used the rolling shutter effect to measure the vibration of objects to reconstruct the speech. Danakis et al. [9] exploited the rolling shutter effect to increase data rates in light-to-camera communications. LiTell [40], iLAMP [44], and Pulsar [41] have applied visible light communication to indoor localization based on frequencies specific to individual light sources. LiShield [43] uses the flickering of smart LEDs to protect visual privacy.

Our work also relies on the rolling shutter effect to capture the light frequency. Different from previous works which dealt with a single light source (e.g., a light bulb or a LCD screen), one of the challenges we faced is that many light sources (i.e., many pixels on OLED screen) flickering asynchronously at a given frequency. Obtaining accurate frequency estimation requires modeling interactions in temporal and spatial domains.

B. Visual Cryptographic Security

Most previous studies have focused on the encryption of authentication codes [6], [7], [22], [23] directly within barcodes. Chen [6] embedded authentication data including codewords and signatures within QR code. Nonetheless, even those researchers concede that their systems are vulnerable to Replay and STLS attacks [2] without hardware fingerprint. To avoid STLS attacks, POSAUTH [2] requires double scanning which changes users' habits. Zhou et al. [42] utilize various brightness pixels as hardware fingerprint, but is restrict with completely dark environment and pixel aging problem.

Visual cryptography (VC) techniques [30] have also been developed to ensure that messages remain concealed. Essentially, a secret image is embedded within shared images aimed at camouflaging the hidden data, which can be revealed only through the stacking of multiple shared images or capturing with a specific approach [5], [13], [24], [29], [37], [39]. mQR-code [29] utilizes moire patterns to encrypt the original QR code, thereby making the information impervious to extraction unless viewed from a specific position. Nonetheless, those methods necessitate the exchange of key images with a server or restrict the user to a pre-determined position to recover the original QR code.

Unlike the methods mentioned above, our approach utilizes hardware fingerprint to avoid STLS attacks, which does not require additional equipment and allows the user complete freedom in the use of their device.

III. Background

A. Pulse Width Modulation (PWM) Dimming Control

Pulse Width Modulation (PWM) dimming [3] is a common technique which adjusts screen brightness by digital encoding an analog signal to appear for a given period of time, wherein the power is either fully on or fully off. Essentially, the screen brightness is adjusted by regulating the on-time (T_{ON}) in each cycle (T_{pwm}) [35] as shown in Fig. 2. Since human eyes are insensitive to changes of high frequency, we do not notice the flickering, but rather perceive a difference in brightness. Essentially, the screen appears brighter when the on:off ratio is larger.

PWM dimming has several advantages. PWM dimming does not impose chromatographic shifts, as the current is always equal to the full-amplitude current I_{max} or 0. PWM also permits high dimming resolution over a wide range of values. This has led to the widespread adoption of PWM dimming for OLED screens and some LCD screens. Note that it is rapidly gaining popularity for mobile devices.

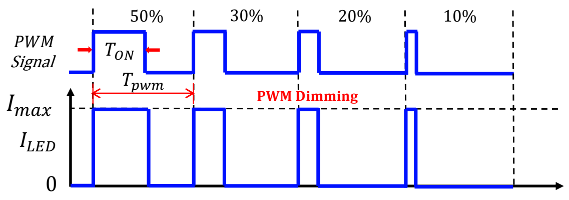


Fig. 2. An illustration of PWM dimming

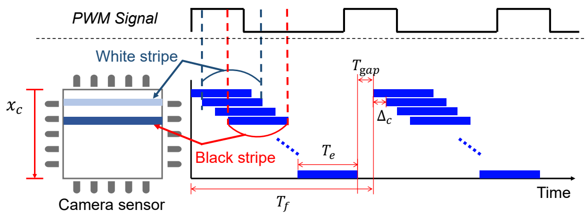


Fig. 3. Sampling performed by camera sensors. Some of the rows sample during the PWM on-time, and others sample during the off-time, respectively producing black (darker) and white (lighter) stripes in the captured image.

B. Rolling Shutter Camera

Complementary metal-oxide semiconductor (CMOS) technology is widely used to fabricate the cameras used in s-martphones. The sequential sampling by different rows in the camera sensor is referred to as a rolling shutter [21]. The latency between the start of each exposure in each row is denoted by Δ_C. Note that Δ_C remains a constant regardless of the ISO and exposure time and whether the sensor is used to capture a still image or video segment. The time during which one frame is captured is denoted by T_f. Note that there is a time gap between two consecutive frames.

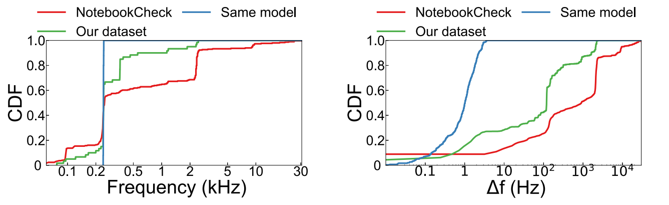
When using a rolling shutter sensor to capture an image, the light intensity indicates the total number of photons received throughout the duration of the exposure. Fig. 3 shows the black (darker) and white (lighter) stripes created by the sensor rows, while recording an image in which PWM signals periodically turn the screen on and off. Examples of this phenomenon are shown in Fig. 6. SCREENID extracts PWM frequency by modeling the stripes produced by the flickering in terms of communication between screen and camera. Details pertaining to the modeling of the stripes and their use in computing the PWM frequency are presented in Section VI.

IV. Preliminary Study

If PWM dimming frequency is to be used as a feature by which to identify screens, then we must first verify its uniqueness and stability.

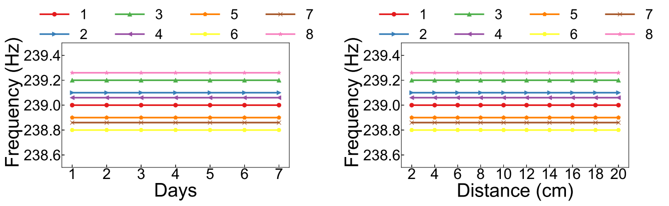
A. Uniqueness of PWM Frequency

We first conducted a preliminary study to assess the u-niqueness of PWM frequencies across screens from different and the same models.We used a light sensor ADPD2212 [11] sampling at $80kHz$ to measure the PWM frequencies of 50 phone screens. We crawled 300 screen benchmarks collected by NotebookCheck [26] (covering mainstream smartphones from 2016 to 2019). Fig. 4(a) illustrates the distribution of PWM frequencies among smartphone screens. We can see that 97% of the PWM frequencies were below $10kHz$ and 68.3% were below $2kHz$. Note that most current smartphones



(a) PWM frequencies. (b) Pairwise PWM frequency differences.

Fig. 4. CDF of PWM frequencies and pairwise differences of 300 screens reported in NotebookCheck [26] and 50 screens (30 are of the same model) we collected.



(a) Time. (b) Distance.

Fig. 5. We measure the PWM frequencies of 8 screens of the same model across days and at various distances to show its stability.

support video capture at 1920×1080 using a frame rate of $30fps$. This means that it should be possible to achieve sampling rates of at least $32.4kHz$, which is far beyond the Nyquist sampling rate ($> 2 \times 10kHz$).

Fig. 4(b) shows the CDF of the pairwise differences in PWM frequency among screens. Note that the frequency resolution in the NotebookCheck dataset is $0.1Hz$ so it cannot distingush screens using a close PWM frequency. In our dataset, the frequency resolution is $0.01Hz$. We can see that among all screens and screens of the same model, 95% pairwise differences are larger than $1.1Hz$ and $0.18Hz$, respectively. The results revealed that a frequency resolution of $0.1Hz$ should be sufficient to differentiate among 99.3% screens.

B. Stability of PWM Frequency

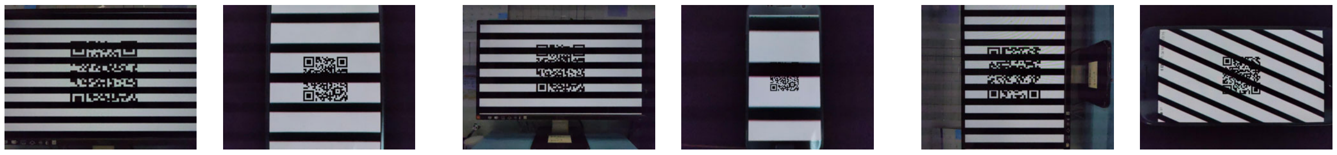
We also assessed the uniformity of PWM frequencies under various conditions. Fig. 5 shows the PWM frequencies of 8 screens (Samsung S7 [12]) of the same model measured in various days and from various distances. The variation in PWM frequency was at most $0.01Hz$ which implies the PWM frequency is stable.

V. Threat Model

In this paper, we aim to enhance the security of a QR code by embedding the screen fingerprint into the displayed QR code. An adversary may destroy the security by launching a STLS attack as following:

An adversary can intercept a QR code without the awareness of the victim during a transaction, for example. Then the stolen QR code can be replayed by the attacker to achieve a successful fraudulent transaction. The whole replay attack process should be finished before the token expires. The attackers may interrupt or delay the legitimate progress for attack by using some social-engineering methods,e.g., talking to the victim or the cashier [2], [20], [27].

We make the following assumptions on the adversary.



(a) lcd screen ($20cm$) (b) oled screen ($5cm$) (c) lcd screen ($40cm$) (d) oled screen ($10cm$) (e) lcd screen ($90°$) (f) oled screen ($90°$)

Fig. 6. Images of LCD and OLED screens captured using a camera. The position of the camera was fixed, while the position and direction of the screens was varied. The stripes in the LCD screen remained the same in all cases, whereas they varied in the OLED screen in terms of width and angle.

- **No access to the victim device**. We consider the adversary can physically obtain the QR code displayed on the victim device from a certain distance. For example, she may capture the QR code using a smartphone or a dedicated camera present at the payment scene.
- **Malware attack**. We assume that the adversary can infect victim's phone with a malicious app, which does not have system privileges but can screenshot or take camera permissions to obtain QR code. For instance, the reflection of QR code on the glass of POS scanner may be sniffed by the malicious app using front camera [2].
- **No limitation to software and hardware.** The adversary may utilize state-of-the-art image photographing, recording and synthesis software techniques. She can also utilize any devices such as high-quality digital cameras and high-speed networks for a replay attack.

VI. Measuring PWM Frequency Using a camera

In this section, we examine the process of measuring PWM frequency using rolling shutter effect. Note that our use of the rolling shutter for frequency measurement must contend with many light sources flickering asynchronously. We addressed this issue by modeling the screen-camera interaction in temporal as well as spatial domain.

A. PWM Dimming of LCD and OLED

LCD and OLED screens both use PWM dimming control; however, the actual dimming methods differ. LCD screens use a single backlight, such that the screen can be regarded as a single light source flickering at its PWM frequency. Figs. 6(a)(c)(e) show the black and white bands caused by the interaction between LCD screen flicker and the rolling shutter. Note the lack of variation in the width and angle of the bands despite variations in the distance and angle between the camera and screen. This indicates the estimates of flicker frequency vary only as a function of rolling shutter speed.

By contrast, the bands in Figs. 6(b)(d)(f) from an OLED screen varied in terms of width and angle, depending on the position of the screen. This can be attributed to the fact that each pixel in an OLED screen is controlled by an LED light source and the PWM cycle of each row is asynchronous. As shown in Fig. 7, the OLED pixels are grouped into rows, the flickering of which is delayed by latency Δ_S from the previous row. Note that PWM latency Δ_S is kept constant among rows in order to prevent fluctuations[1] in current. [14]

[1]A pixel requires a larger current during its PWM duty cycle. By introducing a phase latency between rows, the total current required by the all pixels remain more stable across time.

B. Using a Camera for Sampling

The fact that the brightness of OLED screens varies as a function of time and space means that the rolling shutter effect can be viewed as a process of sampling in the temporal and spatial domains. PWM dimming control produces square wave signals. Since we are primarily interested in frequency f_{pwm}, without a loss of generality, a sinusoidal wave can be used to describe the PWM signals we are interested in, as follows:

$$T(x_s, t) = \cos(2\pi f_{pwm}(t + x_s\Delta_S) + \theta) \tag{1}$$

where $T(x_s, t)$ denotes the signal emitted from the x_s^{th} row of the screen at time t. x_s ranges from $0, 1, ..., X_s - 1$ where X_s indicates the number of rows across the entire screen. θ denotes the initial phase of the signal in the first row.

When using a rolling shutter camera to capture an image from an OLED screen, the screen with X_s rows is mapped to $X_{s\to c}$ rows in the captured picture (as shown in Fig. 8). In other words, we assume that M rows on the camera sensor capture one row from the screen, where the projection ratio M can be denoted as $M = X_{s\to c}/X_s$

Fig. 7 presents an example of image sampling in which the screen and camera are placed in parallel to induce flicker and sampling is performed sequentially from top to bottom. Note that we later remove this assumption and model the interaction under arbitrary screen placements. The red arrows indicate the samples obtained by individual camera rows. $M = 3$ indicates that three camera rows capture the same row on the screen. Assuming that the camera begins sampling at time k, the captured signals can be written as follows:

$$\begin{aligned} R(x_c,t) &= \cos(2\pi f_{pwm}((t + k + x_c\Delta_C) + x_s\Delta_S) + \theta) \\ &= \cos(2\pi f_{pwm}(t + k) + 2\pi f_{pwm}(x_c\Delta_C + \lfloor\frac{x_c}{M}\rfloor\Delta_S) + \theta) \end{aligned} \tag{2}$$

where $R(x_c, t)$ represents the signal received at camera row x_c and time t. x_c ranges from $0...X_c - 1$ where X_c indicates the number of camera rows. Δ_C represents the rolling shutter interval (Fig. 3).

C. Sampling Under Arbitrary Screen Placement

It would be unreasonable to expect all cellphone users to hold their devices at precisely the same angle while the QR code is being read. Thus, we considered sampling with the mobile device held at an arbitrary angle relative to the camera. Fig. 9(a) shows the situation in which the screen is placed at angle α relative to the direction of the rolling shutter. For this arbitrary case, we can obtain a new projection ratio as follows:

$$M = \frac{X_{s\to c}\cos\alpha}{X_s} \tag{3}$$

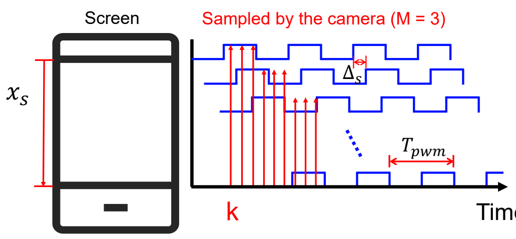


Fig. 7. Asynchronous PWM dimming in OLED screens and example of sampling using a rolling shutter camera where the projection ratio $M = 3$.

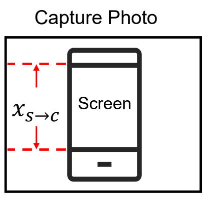
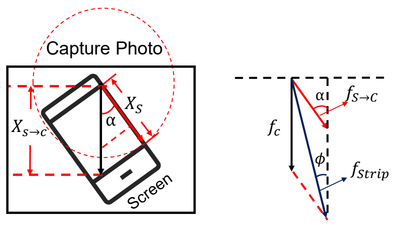


Fig. 8. Projection ratio between screen and camera.



(a) Projection ratio (b) Stripe angle

Fig. 9. Projection ratio and resulting stripe angle in arbitrary direction.

As in Eq.2, the received signal from line x_c at cross-angle α $(0° \leq \alpha \leq 360°)$ can be modeled as follows:

$$R(x_c, t, \alpha) = \cos(2\pi f_{pwm}(t + k) + 2\pi f_{pwm}(x_c\Delta_C + \lfloor \frac{x_c cos\alpha}{M} \rfloor \Delta_S) + \theta)$$

Specifically, the stripe pattern changes as a function of device rotation and distance from the camera (Sec. VI-A). Fig. 9(b) indicates the variation in angle due to rotation and projection ratio M. The scanning performed by the rolling shutter camera and the PWM signal can be considered frequency vectors f_c and f_s, which are derived as $f_c = \frac{1}{\Delta_C}, f_s = \frac{1}{\Delta_S}$.

Due to the degree of projection ratio between the screen and camera, it is possible to convert the actual PWM signal frequency vector into a capture coordinate as $f_{S\to C} = f_s M$. The combined stripe pattern vector (denoted by f_{Strip}) is shown in Fig. 9(b). Thus, the resulting angle of the stripe patterns relative to the horizontal (denoted by ϕ) can be derived as follows:

$$\tan\phi = \frac{f_{S\to C}\cos\alpha + f_c}{f_{S\to C}\sin\alpha} = \frac{\Delta_C M\cos\alpha + \Delta_S}{\Delta_C M\sin\alpha} \tag{4}$$

when $\alpha = 0°$ or $180°$, the refresh direction is the same or the inverse direction of the rolling shutter, such that the stripe patterns are aligned parallel to the bottom of the image.

D. Sampling Using Multiple Frames

The frequency resolution is limited by the length of the data; If we capture only one photo, the frequency resolution is insufficient to determine PWM frequencies of different screens. One intuitive solution is to concatenate multiple frames of a video. However, concatenating frames from an OLED screen is a nontrivial problem, due to the presence of gaps between the frames. The start of exposure in the first row in the previous frame is denoted as t, whereas the start of exposure in the first row in the subsequent frame is $t + T_f$, where T_f denotes the frame duration, as shown in Fig. 3.

Thus, the received signal associated with continuous captures through multiple frames can be denoted as follows:

$$R(x_c, t, \alpha, n) = \cos(2\pi f_{pwm}(t + nT_f + k) + 2\pi f_{pwm}(x_c\Delta_C + \lfloor \frac{x_c\cos\alpha}{M} \rfloor \Delta_S) + \theta) \tag{5}$$

$$(n \in 0, 1, 2, ...)$$

where n denotes the frame number beginning at 0.

Note that Eq. 5 can be treated as a spatial frequency related to row index x_c at specific time $t + nT_f + k$. $\cos(2\pi f_{pwm}(t + nT_f + k)$ can be regarded as the initial phase in the spatial domain, denoted by θ_0. Thus, transforming Eq. 5 into time domain using one sample per line, we can derive the true sampling interval in the communication between screen and camera as follows:

$$\Delta_{s\to c} = \frac{x_c\Delta_C + \lfloor \frac{x_c\cos\alpha}{M} \rfloor \Delta_S}{x_c} \tag{6}$$

where the sampling rates for extracting frequency is denoted by $f_s = 1/\Delta_{s\to c}$.

However, variations among cameras in terms of rolling shutter effects lead to differences in Δ_C and interval Δ_S between the lines in the PWM signal. Variations in frame rates and sampling intervals can also lead to *information duplication* or *information loss*. Essentially, situations involving desynchronization can be divided into two cases:

- **Case1:** Frame duration T_f exceeds the sampling duration $x_c\Delta_{s\to c}$, leading to information loss.
- **Case2:** Frame duration T_f is shorter than the sampling duration $x_c\Delta_{s\to c}$, leading to information duplication.

Fig. 10 presents an example of desynchronization due to variations in frame rates and sampling intervals in three continuous frames. As shown in Fig. 10(a), there exists a time gap $\Delta_T = T_f - x_c\Delta_{s\to c}$ between frame 0 and frame 1 when $\Delta_T > 0$. The signal transmitted during this gap period is not detected in any of the received frames, such that the captured video is non-continuous (incomplete).

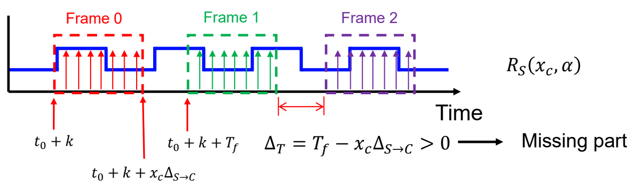
Information duplication can occur in two sequential frames when $\Delta_T < 0$. As shown in Fig. 10(b), frame 0 and frame 1 contain a portion of the tail replicated from frame 0. In these situations, the camera receives identical stripe signals in two sequential frames. Recovering the original signal requires interpolation to deal with *information loss* or the removal of duplicated frame sections and concatenation of the remaining sections to deal with *information duplication*, where the length of the lost or duplicated data is denoted by $\frac{\Delta_T}{\Delta_{s\to c}}$. Specifically, we insert a specific number of zeroes (matching the length of the lost data) into the interval for *information loss*. Conversely, we remove a specific number of tails (matching the length of the data duplication) for *information duplication*.

VII. System Design

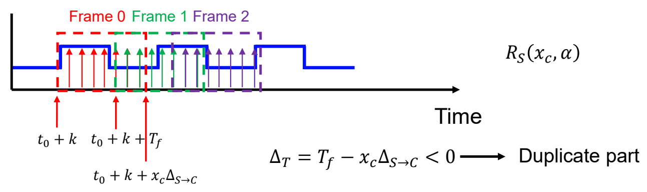
Fig. 11 illustrates the architecture of ScreenID system, comprising two parts: encoding by the sender (smartphone screen) and decoding by the receiver (camera).

A. Encoding

Verifying the authenticity of a screen requires that additional information be embedded in the QR code, namely (i) PWM



(a) Case1: frame duration T_f > sampling duration $x_c\Delta_{s \to c}$



(b) Case2: frame duration T_f < sampling duration $x_c\Delta_{s \to c}$

Fig. 10. Problems associated with desynchronization due to fluctuations in frame rates and sampling intervals.

frequency f_{pwm} and (ii) PWM latency between two rows Δ_S. The size of the QR code to be generated is adjusted automatically based on the resolution and size of the screen, such that the length of the edge is of the fixed number of pixels (580 pixels in our implementation) across screens.

Using SCREENID requires that f_{pwm} and Δ_S are both known in advance. f_{pwm} can be obtained when the user first uses the system and presents his/her phone to the cashier. The cashier then uses the Frequency Extraction scheme to compute f_{pwm}. In practice, the device on the cashier side would use 10 f_{pwm} measurements from which to derive the median f_{pwm}. Estimating PWM latency Δ_S is not a trivial matter; therefore, we propose the method below.

B. Determining PWM Latency Δ_S

PWM latency Δ_S is determined from an image captured at an angle of 90° (i.e., $\alpha = 90°$ in Fig. 9) using a camera with a known configuration. As indicated in Eq. 4, when $\alpha = 90°$, the angle of stripe ϕ is a function of Δ_S that satisfied $\tan\phi = \frac{\Delta_S}{\Delta_C M}$, therefore $\Delta_S = \Delta_C M \tan\phi$, where Δ_C is the rolling shutter interval and M is projection ratio computed using Eq. 3.

In our preliminary analysis, we found little variation in PWM latency Δ_S among phones of the same model; therefore, it was necessary to measure Δ_S only once for each phone model. This could easily be achieved using crowdsourcing where one user of a given phone model performs the calibration and shares the results. Likewise, this information could be provided by the manufacturer when the device is first released.

C. Decoding

Verifying that the captured QR code is authenticated involves embedding the rolling shutter interval Δ_C in camera and PWM latency Δ_S in the QR code for use in estimating the PWM frequency. The QR code is accepted only if the estimated PWM frequency and the PWM frequency embedded in the QR code are a close match lower than a threshold.

The decoding process is performed in two steps: (i) determining the position of the QR code and (ii) extracting the PWM frequency.

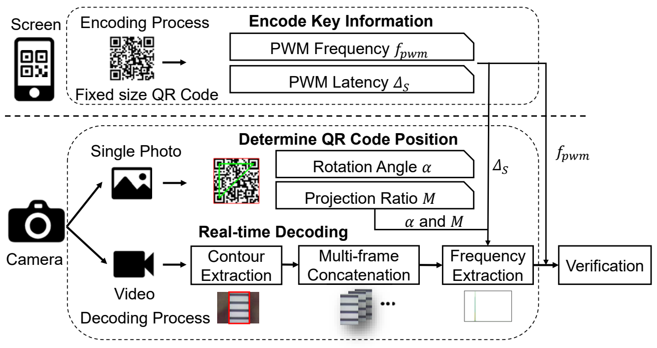


Fig. 11. Overview of SCREENID system.

D. Measuring Rolling Shutter Interval Δ_C

Rolling shutter interval Δ_C must be known a priori in order to compute the PWM frequency. However, we observed inaccuracies in the rolling shutter interval provided by Android Camera2 API [15]. Therefore, we programmed SCREENID to calibrate camera to obtain an accurate indication of rolling shutter interval (denoted as $\widehat{\Delta_C}$). Calibration involves capturing an image of an LCD screen with a known PWM frequency f_{pwm}. Based on Eq. 5, the wavelength (i.e., the width of a pair of black and white stripes) denoted by W_{lcd} can be derived as $W_{lcd} = 1/f_{pwm}/\widehat{\Delta_C}$. Thus, the accurate rolling shutter interval can be estimated as $\widehat{\Delta_C} = 1/f_{pwm}W_{lcd}$. Note that it is easy to calibrate for camera on cashier when carrying out factory examination.

E. Optimizing the Camera Configuration.

The images obtained using the camera serve two purposes. The first purpose involves estimating the position of the QR code and extracting the information embedded. We set the camera to auto-mode to obtain the optimal configuration for ambient lighting conditions to obtain a clear image. The second purpose involves measuring the PWM frequency, which requires a clear indication of the stripes. The following configuration was adopted for capturing video frames:

Exposure time. Many smartphones use PWM dimming only under a low brightness ratio, which tends to extend the exposure time. Note however that an excessively long exposure time would allow the occurrence of more than one PWM cycle, thereby compromising stripes clarity. Thus, we configured SCREENID to operate at an exposure time of $0.3ms$.

ISO. SCREENID was configured to use the highest available ISO, as this tends to enhance contrast between the black and white stripes. High ISO operations also tend to generate noise; however, the fact that most of the noise it at lower frequencies means that it's easily filtered out using a highpass filter.

Aperture. SCREENID was configured to use the largest available aperture as this tends to blur the background, making it easier to extract the contours of the screen.

F. Determining QR Code Position

Estimating the distance and angle between the screen and the camera involves taking a single photo in auto-mode in order to detect the QR code. The position symbols in the the

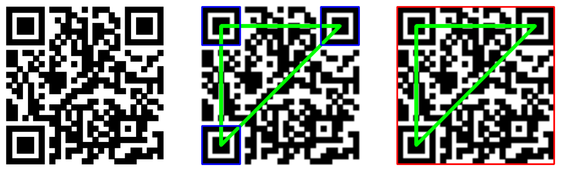

(a) Original code. (b) Rotation angle. (c) Contour.

Fig. 12. Methods used to obtain the rotation angle and projection ratio using a QR code of fixed size

QR code (blue blocks in Fig. 12(b)) to form a right-angle triangle (green in Fig. 12(b), which can be used to estimate rotation angle α.

Locating the position markers makes it possible to extract the contours of the QR code (red in Fig. 12(c)) and measure the size of the QR code in the photo. The size of the displayed QR code is fixed; therefore, it is possible to compute projection ratio M as a fraction of the length of a pixel along the both sides of the original QR code.

G. PWM Frequency Extraction Using Multiple Frames

After extracting essential information from QR code, the camera switches to video mode for frequency extraction. As shown in Fig. 11, this process includes three following steps:

Contour Extraction: Each frame is first transformed into a gray-scale image to enhance contrast. The image then undergoes denoising and binarization, before the boundaries are detected using a function based on the Sobel operator [19].

Multi-frame Concatenation: From among the multiple columns in the contour, we select the longest column as a data sample of screen in a frame to maximize the data length. The frequency resolution (f_{res}) of the Fourier transform is $f_{res} = f_{sample}/N$ [28], where f_{sample} indicates the sampling rates and N indicates the number of samples. Identifying the characteristic PWM frequency requires a frequency resolution of $f_{res} = 0.1Hz$ or smaller. This can be achieved by concatenating columns from multiple frames to increase N, as described in Sec. VI-D.

Frequency Extraction: For a usual condition, the camera of smartphones can sample at $f_{sample} = 80kHz$ with 4000 columns at frame rates of $30fps$. Thus, it requires $\frac{f_{sample}}{f_{res} \times 4000 \times 30} = 6.7$ seconds. Improving system efficiency requires reducing the time required to capture frames, while estimating the PWM frequency with high precision.

This was achieved by exploiting the fact that the sampling rate (e.g., $80kHz$) usually exceeds the Nyquist rate required to reconstruct PWM signals (e.g., $500Hz$). Thus, we down sample a sequence of samples of length N (e.g., $[x_1, x_2, x_3, \cdots, x_N]$) by a factor of K by taking a sample at K intervals, which results in K segments. (e.g., $[x_1, x_{K+1}, x_{2K+1}, \cdots]$, $[x_2, x_{K+2}, x_{2K+2}, \cdots]$, $\cdots$) from the original sequence. The sampling rate of each segment is becomes f_{sample}/K. We then concatenate all of the segments into a single sequence of length N, which retains the same PWM frequency as long as the new sampling rate is $\frac{f_{sample}}{K} > 2f_{pwm}$. We apply the Hanning window [36] to make the concatenation smoother before applying the Chirp-Z transform (CZT) [32] to improve spectral resolution in the frequency range of interest (i.e., f_{pwm}). We named the whole process *Down-sampling CZT* in following text.

Verification: ScreenID compares the estimated PWM frequency with the one embedded in the QR code. When the difference is less than a given threshold ($0.03Hz$ in current study), the QR code is accepted. Otherwise, it is rejected.

VIII. Evaluation

A. Experiment Setup

We generate version-3 (29×29) QR codes[2] using ScreenID.The size of QR codes is 580×580 pixels. 50 smartphone screens and 5 smartphone cameras are used in our evaluation. Among 50 screens, 10 are LCD and 40 are OLED screens. 30 screens are of the same model.

Unless otherwise stated, we evaluated ScreenID as follows. For each screen, we displayed 40 QR codes where 10 embedded the correct PWM frequency of the screen for authentication while the other 30 embedded that of another screen randomly selected from 50 screens for attack.

We utilize *TPR (True Positive Rate)* and *FPR (False Positive Rate)* as the metrics for performance evaluation, which are defined as follows:

- *TPR* $= \frac{Number\ of\ accepted\ authorized\ QR\ code}{Number\ of\ verification\ attempts}$
- *FPR* $= \frac{Number\ of\ accepted\ unauthorized\ QR\ code}{Number\ of\ attack\ attempts}$

The higher *TPR* is and meanwhile the lower *FPR* is, the better ScreenID performs.

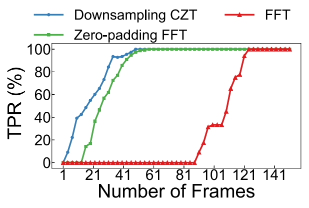
B. Microbenchmark

1) Number of Required Frames: Here we evaluate how many frames are required to accurately estimate the PWM frequency. We used 5 cameras to capture a 60-second video of each of 50 screens. We compare the proposed down-sampling CZT with FFT and zero-padding FFT. The average TPR under various number of frames is reported in Fig. 13(a). FFT needed 121 frames to get enough frequency resolution to achieve 100% TPR. Zero-padding FFT [16] is a well-known method to refine the frequency granularity by padding zeros. Zero-padding FFT and down-sampling CZT perform similarly and needed 65 and 60 frames, respectively. However, the computation complexity of zero-padding FFT is larger than down-sampling CZT [32], so that down-sampling CZT performs faster than zero padding FFT as shown in Fig. 13(b). In the following evaluation, we take a 60-frame video and used down-sampling CZT to compute PWM frequency.

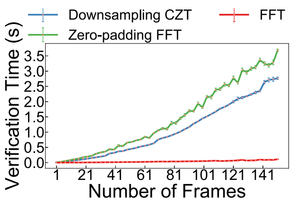
2) Camera Configuration: To find the best exposure time and ISO, we measure the TPR under various settings and show the results in Fig. 14. For the exposure time, we balance the brightness ratios and set exposure time to $0.3ms$. For ISO, we can see that the higher ISO is, the higher the TPR is; therefore, we set ISO to the highest available for the camera.

3) Threshold: ScreenID is authorized only when the difference between estimated frequency and true frequency

[2]Alipay uses version-2 (25×25) QR codes [34] while WeChat uses version-1 (21×21) [34] QR codes. Version-3 QR codes which carry more data are representative of the amount of data needed by these systems.

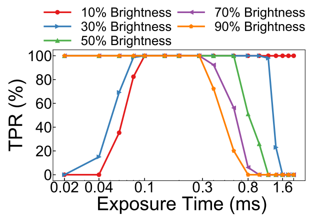


(a) Verification TPR.

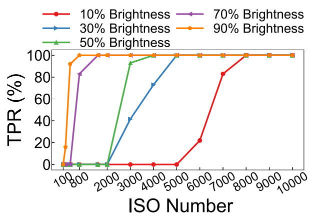


(b) Verification time.

Fig. 13. Verification TPR and time of different frequency extraction schemes.



(a) Impact of exposure time.



(b) Impact of ISO number.

Fig. 14. The TPR on impact of exposure time and ISO number under different brightness ratio.

smaller than the threshold. However, when the threshold become larger, the TPR and the FPR both become higher. Fig. 15 shows the results of TPR and FPR on the impact of threshold. To consider a better trade-off, we select $0.03Hz$ as the threshold where the average TPR is above 94.3%, the FPR is below 0.8% across the same model and 0.5% across our dataset.

4) Processing Time: We measure the processing time of SCREENID on 5 phones. The total computation time is $0.857s$. The feature extraction scheme takes $0.605s$ which accounts for most processing time. Currently we implement down-sampling CZT using OpenCV [4] which can be further optimized by using native DSP SDK [31]. We leave it in our future works.

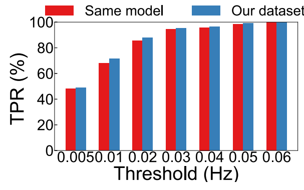
C. Overall Performance

We evaluate the overall performance using 5 cameras and 50 screens. Fig. VIII-C shows the TPR and FPR of 50 screens. The first 30 screens are of the same model. The average TPR is 99.7% and 95.0% for all screens and phones of the same model. The FPR for each screen in the same model is lower than 1.5%, and for other models, the FPR is 0%, respectively.

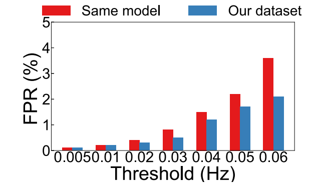
D. Robustness of SCREENID

1) Impact of Screen Brightness: Fig. 17(a) shows the TPR under various brightness ratios. We can see the TPR is above 99.5% when the brightness ranges from $10\% - 90\%$. When the brightness is 100%, the screen is always on so it doesn't flicker. Therefore, SCREENID automatically adjusts the brightness to 50% to ensure PWM dimming is applied.

2) Impact of Capturing Distances: We evaluate SCREENID under various distances ranging from $0cm$ to $26cm$. Fig. 17(b) shows that TPR is 100% when the distance is $10cm - 18cm$. When the captured distance is smaller than $8cm$, some cameras cannot capture the complete QR code so the TPR drops to 0%. When the distance is longer than $22cm$, the true sampling interval $\Delta_{s\to c}$ becomes larger while the projection ratio M becomes smaller. As the result, the wavelength of stripes is larger than the length of the camera sensor so SCREENID cannot estimate the PWM frequency accurately. This result

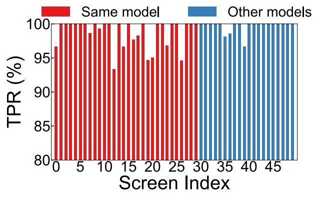


(a) TPR on the impact of threshold

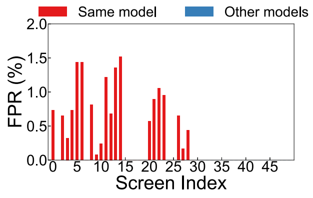


(b) FPR on the impact of threshold

Fig. 15. The TPR and FPR on the impact of threshold of 50 screens we collected (30 are of the same model).



(a) TPR for each screen.



(b) FPR for each screen.

Fig. 16. The TPR and FPR of 50 screens where the first 30 (red) are of the same model. The two groups are ordered by their true PWM frequencies.

implies SCREENID works well at the distance from $10-18cm$, which is enough for the mobile payment and can prevent from being candid in a much further distance.

3) Impact of Rotation Angles: We evaluate the impact of rotation angles (i.e., α in Fig. 9(a)) by rotating the screen from $0°-360°$ clockwise. Fig. 17(c) shows that TPR is above 95.2% in all cases. This implies that SCREENID is robust against how users hold the phone.

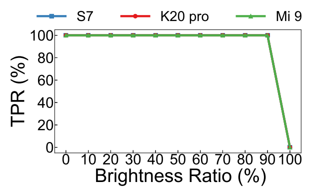
4) Impact of Offset Angles: When the camera is not held right above the screen, the captured photo may be distorted. To evaluate the impact of the distortion, we measure the TPR under various offset angles and show the results in Fig. 17(d). We can see when the offset angle is less than $5°$, TPR is above 91.8%. However, when the offset angle is larger than $10°$, TPR drops to 0%. As we show in the user study, the restriction does not impair the usability of SCREENID in daily use as the users can easily put the phone with an offset angle within $5°$. The result also demonstrates SCREENID can successfully prevent from being sniffed in other viewing angles.

5) Impact of Ambient Light: QR codes are widely used in a variety of environments and the lighting condition is one of the key factor affecting screen to camera communications. We evaluate the impact of lighting condition under following environments: L_A:indoor with light off, L_B, L_C, L_D: outdoor at $9am$,$12am$,$5pm$, L_E:indoor with light on. The results are shown in Fig. 18. It suggests that SCREENID works well under various lighting conditions but the TPR may slightly degrade if the light is too strong (91.3% while using outdoor at $12a.m.$)

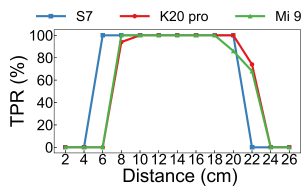
E. User Study

We recruited 10 participants to use SCREENID (7 male and 3 female, ages from $22-56$.) Each participant was requested to hold a smartphone to decode 30 traditional QR codes and 30 SCREENID QR codes. Two types of codes were given alternately to avoid the bias.

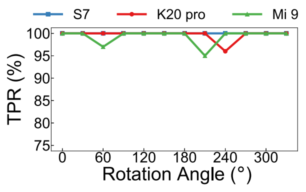
First, all the traditional and SCREENID QR codes were correctly decoded. We measured the time from a QR code was given to that it was successfully decoded and show the CDF in



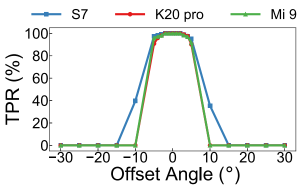
(a) Impact of screen brightness.



(b) Impact of capture distance.



(c) Impact of rotation angle.



(d) Impact of offset angle.

Fig. 17. The TPR under various impact factors.

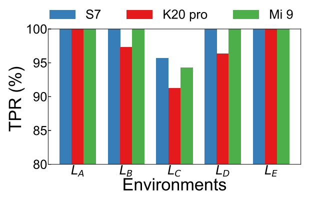


Fig. 18. The TPR on the impact of lighting conditions.

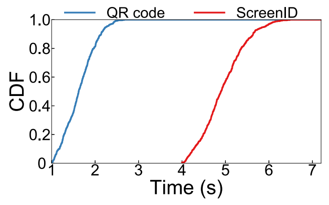
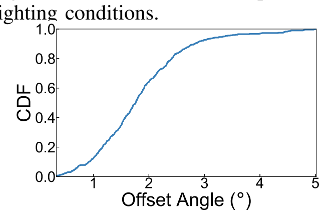
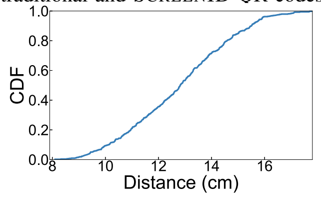


Fig. 19. The required decode time of traditional and SCREENID QR codes.



(a) Offset angle.



(b) Distance.

Fig. 20. The CDF of initial offset angle and distance.

Fig. 19. We can see that traditional and SCREENID QR codes take $1.6s$ and $4.8s$ to decode in average. SCREENID QR codes need a longer time because SCREENID takes a $2s$ video and additional process time around $0.857s$ (Sec. VIII-B4).

As for the hand shaking effect in using SCREENID, there is inevitably a certain amount of camera shake (usually less than $2mm$) [8]. All SCREENID QR codes are correctly verified implying SCREENID is robust against the small shaking. It's because we extract a column from each frame independently so the shift between frames do not impair the accuracy.

Finally, we evaluated if participants can put the phone at the right positions. As shown in Sec. 17(b) and 17(d), the best working distances and offset angles is $10-18cm$ and $-5°$ to $5°$, respectively. We measured the phone positions while participants presented QR codes and show the CDF of distances and offset angles in Fig. 20. We can see that of 95% offset angles are within $4.5°$ and 95% distances are from $10-16cm$, which implies SCREENID can successfully work.

IX. SECURITY ANALYSIS

In this section, we discuss various attacks and the ability of SCREENID to deal with them. We assume that the attacker is able to determine the PWM frequency, but is not cooperating with the cashier responsible for scanning the QR code.

Can attackers mimic the victim's PWM frequency by changing their phone's PWM frequency? PWM dimming is applied to an on-chip display controller [3] using one of two methods [17]. The first is to control the display controller via I^2C interface, and produce corresponding PWM signals. The attackers cannot change their PWM frequency without replacing the display controller chip. The second involves the CPU outputting a PWM signal through a pin connected to display controller. The attackers should have the root permission to modify low-level drivers and identify the CPU pin if he want to alter PWM frequency. However, increasingly strict security restriction on the Android system [25] makes it nearly impossible to modify PWM frequency.

Can attackers mimic the victim's PWM frequency by adding rolling stripes over the QR codes? According to Eq. 5, the stripes on the screen (i.e., the victims screen) are a function of PWM frequency f_{pwm} as well as the rolling shutter interval Δ_C. The fact that the attacker does not know the configuration of the camera on cashier (i.e., Δ_C) makes it impossible to produce the correct rolling stripes over the QR codes; i.e., reflection of QR code on glass of cashier only obtain Δ_C of front camera, not the camera of cashier.

Can attackers mimic the victim's PWM frequency by using an external lighting device? If we assume that the attacker has a lighting device capable of flickering at an arbitrary rate, it might be possible to light their screen at a rate mimicking the PWM frequency of the victim. However, we do not view this kind of attack method as practical, as it would be easily detected by the cashier.

Can attackers happen to have a phone with the same PWM frequency as that of the victim's phone? In the proposed system, the threshold we select is $0.03Hz$. As shown in Fig. 4(b), 99.8% of the pairwise difference in frequency is larger than $0.03Hz$. This would require that an attacker use 450 phones to achieve 90% success rate. Such an attack would indeed be possible; however, it would be prohibitively expensive. SCREENID is meant to enhance the security of QR codes; however, it cannot guarantee security. Note that SCREENID does not need customized techniques so that it can meanwhile incorporate with existing authentication schemes.

In summary, SCREENID provides robust protection against attacks using hardware (not easily manipulated) and joint features (f_{pwm}, Δ_S, and Δ_C), which are difficult to compromise at the same time. It is expected that PWM dimming will become the overwhelming standard as OLED gain popularity [33] thanks to their light weight, wide viewing angle, and high color accuracy [3].

X. CONCLUSION

In this study, we proposed SCREENID to enhance the security of QR codes by fingerprinting the screens displaying the QR codes. Only QR codes displaying on its specific screen are accepted. SCREENID exploits PWM frequency as a unique screen fingerprint. Extensive experiments demonstrate the efficacy of SCREENID in differentiating screens to enhance the security of QR codes in real-world situations.

References

[1] Alipay. Alipay: Experience fast, easy and safe online payments., 2020.

[2] X. Bai, Z. Zhou, X. Wang, Z. Li, X. Mi, N. Zhang, T. Li, S.-M. Hu, and K. Zhang. Picking up my tab: Understanding and mitigating synchronized token lifting and spending in mobile payment. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 593–608, 2017.

[3] M. Barr. Pulse width modulation. *Embedded Systems Programming*, 14(10):103–104, 2001.

[4] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[5] X. Cao, L. Feng, P. Cao, and J. Hu. Secure qr code scheme based on visual cryptography. In *2016 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE 2016)*. Atlantis Press, 2016.

[6] C. Chen. Qr code authentication with embedded message authentication code. *Mobile Networks and Applications*, 22(3):383–394, 2017.

[7] Y.-W. Chow, W. Susilo, G. Yang, M. H. Au, and C. Wang. Authentication and transaction verification using qr codes with a mobile device. In *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, pages 437–451. Springer, 2016.

[8] K. J. Connolly. *The psychobiology of the hand*. Cambridge University Press, 1998.

[9] C. Danakis, M. Afgani, G. Povey, I. Underwood, and H. Haas. Using a cmos camera sensor for visible light communication. In *2012 IEEE Globecom Workshops*, pages 1244–1248. IEEE, 2012.

[10] A. Davis, M. Rubinstein, N. Wadhwa, G. Mysore, F. Durand, and W. T. Freeman. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 33(4):79:1–79:10, 2014.

[11] A. Devices. Adpd2212 datasheet, 2016.

[12] DeviceSpecifications. Device specifications, 2019.

[13] W.-P. Fang. Offline qr code authorization based on visual cryptography. In *2011 Seventh International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 89–92. IEEE, 2011.

[14] J. Genoe, K. Obata, M. Ameys, K. Myny, T. H. Ke, M. Nag, S. Steudel, S. Schols, J. Maas, A. Tripathi, et al. 30.2 digital pwm-driven amoled display on flex reducing static power consumption. In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pages 488–489. IEEE, 2014.

[15] GoogleDevelopers. Sensor rolling shutter skew of camera2, 2020.

[16] S. Hilbert. Fft zero padding, 2013.

[17] T. Instruments. Lm36923 highly efficient triple-string white led driver, 2015.

[18] JobTubeDaily. Your mobile money could be stolen by this attack!, 2018.

[19] N. Kanopoulos, N. Vasanthavada, and R. L. Baker. Design of an image edge detection filter using the sobel operator. *IEEE Journal of solid-state circuits*, 23(2):358–367, 1988.

[20] T. Li. Qr code scams rise in china, putting e-payment security in spotlight., 2019.

[21] C.-K. Liang, L.-W. Chang, and H. H. Chen. Analysis and compensation of rolling shutter effect. *IEEE Transactions on Image Processing*, 17(8):1323–1330, 2008.

[22] K.-C. Liao and W.-H. Lee. A novel user authentication scheme based on qr-code. *Journal of networks*, 5(8):937, 2010.

[23] J. Lu, Z. Yang, L. Li, W. Yuan, L. Li, and C.-C. Chang. Multiple schemes for mobile payment authentication using qr code and visual cryptography. *Mobile Information Systems*, 2017, 2017.

[24] 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)*, pages 110–124. IEEE, 2005.

[25] T. Mohini, S. A. Kumar, and G. Nitesh. Review on android and smartphone security. *Research Journal of Computer and Information Technology Sciences*, 2320:6527, 2013.

[26] NotebookCheck. Pwm ranking - the best displays for the eyes, 2019.

[27] B. O'Donnell. Steals money sneakily by scanning people's qr code., 2019.

[28] A. V. Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

[29] H. Pan, Y.-C. Chen, L. Yang, G. Xue, C.-W. You, and X. Ji. mqrcode: Secure qr code using nonlinearity of spatial frequency in light. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–18, 2019.

[30] P. Punithavathi and S. Geetha. Visual cryptography: A brief survey. *Information Security Journal: A Global Perspective*, 26(6):305–317, 2017.

[31] Qualcomm. Hexagon dsp sdk, 2019.

[32] L. Rabiner, R. Schafer, and C. Rader. The chirp z-transform algorithm. *IEEE transactions on audio and electroacoustics*, 17(2):86–92, 1969.

[33] statista. Shipments of flexible and rigid amoled panels worldwide from 2015 to 2022, 2018.

[34] J. Steeman. Qr code data capacity, 2019.

[35] H. Sugiyama, S. Haruyama, and M. Nakagawa. Brightness control methods for illumination and visible-light communication systems. In *2007 Third International Conference on Wireless and Mobile Communications (ICWMC'07)*, pages 78–78. IEEE, 2007.

[36] A. Testa, D. Gallo, and R. Langella. On the processing of harmonics and interharmonics: Using hanning window in standard framework. *IEEE Transactions on Power Delivery*, 19(1):28–34, 2004.

[37] S. Thamer and B. Ameen. A new method for ciphering a message using qr code. *Comput. Sci. Eng.*, 6(2):19–24, 2016.

[38] WeChatPay. Wechat pay: Beyond payment, leading mobile payment platform, 2020.

[39] C.-N. Yang, J.-K. Liao, F.-H. Wu, and Y. Yamaguchi. Developing visual cryptography for authentication on smartphones. In *International Conference on Industrial IoT Technologies and Applications*, pages 189–200. Springer, 2016.

[40] C. Zhang and X. Zhang. Litell: robust indoor localization using unmodified light fixtures. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 230–242, 2016.

[41] C. Zhang and X. Zhang. Pulsar: Towards ubiquitous visible light localization. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 208–221, 2017.

[42] Z. Zhou, D. Tang, W. Wang, X. Wang, Z. Li, and K. Zhang. Beware of your screen: Anonymous fingerprinting of device screens for off-line payment protection. In *Proceedings of the 34th Annual Computer Security Applications Conference*, pages 77–88, 2018.

[43] S. Zhu, C. Zhang, and X. Zhang. Automating visual privacy protection using a smart led. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 329–342, 2017.

[44] S. Zhu and X. Zhang. Enabling high-precision visible light localization in today's buildings. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, pages 96–108, 2017.