Do You See How I Pose? Using Poses as an Implicit Authentication Factor for QR Code Payment

Chuxiong Wu and Qiang Zeng, *George Mason University*

https://www.usenix.org/conference/usenixsecurity24/presentation/wu-chuxiong

This paper is included in the Proceedings of the
33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the
33rd USENIX Security Symposium
is sponsored by USENIX.

Do You See How I Pose? Using Poses as an Implicit Authentication Factor for QR Code Payment

Chuxiong Wu
George Mason University

Qiang Zeng
George Mason University

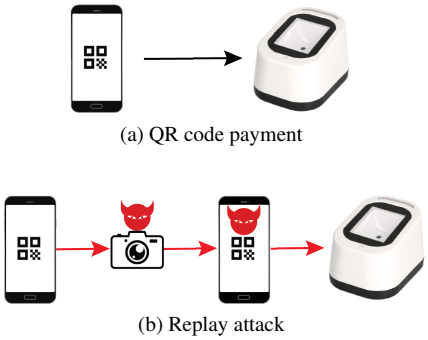
Abstract

QR code payment has gained enormous popularity in the realm of mobile transactions, but concerns regarding its security keep growing. To bolster the security of QR code payment, we propose PQRAUTH, an innovative implicit second-factor authentication approach that exploits smartphone poses. In the proposed approach, when a consumer presents a payment QR code on her smartphone to a merchant's QR code scanner, the scanner's camera captures not only the QR code itself but also the smartphone's poses. By utilizing poses as an additional factor, in conjunction with QR code decoding, the scanner verifies the authenticity of the smartphone presenting the QR code. Our comprehensive evaluation demonstrates the effectiveness of PQRAUTH, affirming its security, accuracy and robustness.

1 Introduction

The widespread adoption of smartphones has significantly changed the way commercial activities are carried out. In particular, Quick Response (QR) codes, which enable contactless mobile payment, have been widely utilized. As many as 1.5 billion shoppers worldwide used QR codes to pay in 2020 and the number is predicted to be 2 billion by 2025 [54]. The market of QR code payment is projected to reach $35.07 billion by 2030 [55]. QR codes have become commonplace among mobile payment service providers, such as Venmo, PayPal, AliPay, and WeChat.

QR code payment provides fast and touch-free experiences, obviating the necessity for carrying cash/credit cards or manual receipt signing. As shown in Figure 1 (a), a consumer only needs to show a QR code generated by a mobile payment app to a scanner to complete the transaction. The code encodes information, such as the user account ID and timestamp [39]. The scanner sends the decoded bits with other transaction information, such as the transaction amount and currency type, to the back-end of the payment system for verification. If the QR code is deemed valid, the payment is authorized.



(a) QR code payment

(b) Replay attack

Figure 1: (a) A consumer presents a payment QR code for a transaction. (b) An attacker sneakingly obtains the consumer's QR code and replays it to initiate a payment, without deciphering the encoded information.

Attacks against QR code payment systems have been frequently reported [8, 39, 46, 52, 74, 80]. As shown in Figure 1 (b), an attacker sneakingly acquires the victim's QR code displayed on the victim's smartphone, and then displays it on the attacker's smartphone to check out. The attacker does *not* need to decipher the encrypted data (such as the user account ID) in the QR code, but simply re-displays the victim's QR code. As the QR code is an exact copy of the code generated by the victim and thus considered valid, the transaction proceeds, but the server charges the victim's account.[1]

Similar to the notorious relay attacks targeting chip card payment systems [4], replay attacks pose a significant threat to QR code payment systems. Replay attacks against QR code payment are not only feasible, but also cheap [8]. Attackers can use affordable commercial off-the-shelf smartphones or cameras to record QR codes [8]. Additionally, replay attacks are often stealthy, as a QR code can be sent to a remote attacker to conduct a fraudulent transaction. Due to the widespread use of QR codes by various entities, including

[1] This is called a *replay attack* in prior work [39, 52, 74], as the QR code is first displayed on the victim's phone and then *re-displayed* on the attacker's phone. We use the same term, and clarify that the QR code is **not** used *twice*.

large retailers like Walmart and Starbucks, as well as financial organizations such as Paypal and AliPay, replay attacks against these systems are on the rise and can cause substantial financial losses [39, 52, 74].

To defeat replay attacks in QR code payment, researchers have proposed various solutions [8, 39, 52, 74, 80]. Unfortunately, these methods are vulnerable, face challenges in real-world deployment, or require users to change their habits. More details about these works are discussed in Section 2.

We present a novel QR code authentication system, named PQRAUTH, which leverages the poses of a smartphone displaying a payment QR code as a second authentication factor. PQRAUTH provides an additional layer of security for QR code payment based on the simple yet solid fact: What the scanner sees (i. e., the observed poses of the smartphone presenting the QR code) should correlate with the actual poses of the *account owner*'s smartphone (sensed by the built-in sensors of the account owner's smartphone).

The poses sensed by the built-in sensors of the account owner's smartphone serve as the *ground truth*. PQRAUTH checks whether the scanner-observed poses of the smartphone displaying the QR code match the ground truth. Specifically, when a consumer displays a payment QR code on the smartphone, the Inertial Measurement Unit (IMU) in the smartphone collects information regarding the smartphone's poses, which is employed as the ground truth. When a scanner scans the QR code, it estimates the smartphone's poses by analyzing the recorded video. By examining whether the scanner-estimated poses match the ground truth, PQRAUTH determines whether the payment should be authorized.

PQRAUTH carries multiple prominent properties. First, PQRAUTH is secure against replay attacks. An attacker who displays a victim's QR code would also need to mimic the poses of the victim's smartphone, *in real time*, in order to fool. However, studies have shown that people tend to overestimate acute angles and underestimate obtuse angles when reproducing angles, even if there are no real-time constraints [14, 27]. Thus, it is unlikely that attackers can reproduce the poses of the victim's smartphone in real time. Second, PQRAUTH is a software-based solution, which requires no special hardware on the consumer side. Third, PQRAUTH provides an implicit authentication factor that does not change any user habits, maintaining the excellent usability of QR code payment. With PQRAUTH, a user is *not* restricted to presenting their smartphone in any specific positions or orientations. Instead, the user presents a payment QR code in the same way as specified by current QR code payment. Fourth, PQRAUTH can be generalized to other scenarios. QR codes are utilized in various other scenarios, including gate access control in buildings, ticketing services, and profile sharing in social media apps. PQRAUTH can also be applied to these scenarios.

It is worth clarifying that PQRAUTH requires zero permissions from smartphone users, as it merely uses zero-permission motion sensors. In addition, the data collection is only conducted while a user is using the QR code, which alleviates privacy concerns.

We build a prototype of PQRAUTH and evaluate its performance in real-world scenarios. Below are some of the questions that our evaluation has studied. *Does the system work for different smartphones and scanners? Is the system resilient to attacks? Is the latency overhead negligible?* The results give positive answers to all the questions.

The contributions of this work include:

1. We propose a novel authentication system named PQRAUTH. It provides an implicit authentication factor for QR code payment without changing user habits.

2. We present a technique that translates 2D images captured by a monocular camera to fine-grained 3D pose information. Plus, an effective method that determines the correlation between pose data is devised.

3. We implement a prototype of PQRAUTH and perform a comprehensive evaluation in real-world environments. Extensive experiments demonstrates its accuracy (>0.99), robustness, and resilience to attacks.

The rest of the paper is organized as follows. We present related work in Section 2 and introduce the background of QR code payment schemes in Section 3. We present the threat model and outline the system overview in Section 4. Section 5 shows the design details. The experiment settings for data collection are presented in Section 6 and the evaluation is presented in Section 7. Section 8 details the user study. We discuss the limitations, improvements, and other potential attacks in Section 9. The paper is concluded in Section 10.

2 Related Work

QR code security. To mitigate replay attacks, most mobile payment apps, such as WeChat and Alipay, set a valid time period for each payment code, which is around 90 seconds. Moreover, mobile payment apps restrict that a code can only be used once, which means any attempts to launch attacks after the code has been used will be in vain. However, as replaying a victim's QR code can be conducted in a matter of sub-seconds, attackers have rich opportunities to have the replayed code scanned before the legitimate user [39, 52, 74]. Moreover, a variant of replay attacks, called *Synchronized Token Lifting and Spending* [8], describes various means to disrupt the legitimate transaction, ensuring the attacker to use the QR code first.

As we aim at a secure and usable authentication approach, we assess QR code security enhancement approaches based on the following criteria: **(C1) resilience to attacks**, such as replay attacks; **(C2) no requirement for special hardware on the consumer side**; **(C3) no alteration of user habits; and (C4) good generalizability.** The results are summarized in Table 1.

Table 1: Comparison. ✓: true, ✗: false, ✓̸: partially true, **?**: unclear.

Technique	C1	C2	C3	C4
mQRCode [52]	✓	✓	✗	✗
ScreenID [39, 74]	✗	✗	✓	✗
AnonPrint [80]	✓	✓	✗	✗
POSAUTH [8]	✓	✓	✗	✗
Secret Handshakes [19]	✗	✓	✗	✓
RF-based technologies [56]	✗	✓	✓	✗
Distance bounding [11]	?	✗	✓	✗
PQRAUTH	✓	✓	✓	✓̸

Existing approaches detect replay attacks by examining unclonable characteristics of either the QR code or the screen displaying the code. For example, mQRCode [52] proposes to use moiré patterns to camouflage the original QR code. This makes it challenging for attackers to decipher the camouflaged code when it is not in a designated position. However, to use mQRcode, users must follow specific orientation and positioning requirements, which requires changes in their established habits (**C3**: ✗). As users needs to learn how to use mQRCode and change their habits, it is difficult to generalize it to other QR code applications (**C4**: ✗).

ScreenID [39, 74] utilizes the pulse width modulation (PWM) frequency of screens to establish a distinct screen fingerprint to enhance QR code security. Nevertheless, attackers can mimic the victim's PWM frequency by using an external lighting device, which can go unnoticed by cashiers in self-checkout scenarios. Alternatively, once knowing the rolling shutter interval of the QR code scanner, which is a constant value, attackers can add rolling stripes over the QR code to mimic the victim's PWM frequency [39, 74] (**C1**: ✗). Furthermore, many smartphones are not equipped with PWM screens, as PWM dimming can cause discomfort such as eye strain, nausea, and headache [1] (**C2**: ✗), which limits the generalizability of ScreenID (**C4**: ✗).

AnonPrint [80] fingerprints screens with unique brightness levels of pixels across the display area, but it only works in completely dark environment (**C4**: ✗) and requires the smartphone to cover the whole scanner, which negates the advantage of touch-free characteristics of QR code payment (**C3**: ✗). In addition, it is susceptible to pixel aging issues [80].

POSAUTH [8] involves attaching a unique QR code to each POS terminal, which encodes its ID. Without POSAUTH, the consumer directly displays a QR code. With POSAUTH, however, the consumer needs to first scan the POS terminal's QR code and then displays a QR code that encodes the POS terminal's ID. This approach requires changes of user habits (**C3**: ✗). This significant change also harms its generalizability (**C4**: ✗). For example, it can hardly be applied to a busy subway station that requires fast authentication.

Unlike existing approaches, our work is the first that enhances QR code payment security through correlation (without using fingerprints). Correlation-based authentication has led to many famous works [37,38,41,42,61,72,73,75,76]. As interpreted in Section 1, PQRAUTH carries multiple prominent properties. However, we admit that it has a limitation (**C4**: ✓̸): PQRAUTH can only be used with hands-free scanners (or hand-held scanners with built-in IMU, such as smartphones). According to our survey, all the retailers have hands-free scanners (see Section 9).

Alternative methods. One potential approach to QR code payment security is to generate the payment QR code only after the user presents it to a scanner. Like Secret Handshakes [19], which enhances RFID security by activating RFID tags only after a secret gesture is recognized, QR code security could be bolstered by generating the QR code only after detecting secret gestures. However, since replaying a victim's QR code can be executed rapidly [39, 52, 74], it remains a significant threat even if the QR code is generated after secret gestures are recognized (**C1**: ✗). Additionally, enforcing secret gestures alters user habits (**C3**: ✗).

To authenticate the proximity of a smartphone to a QR code reader, a straightforward approach is to utilize radio frequency (RF) based wireless technologies, such as received signal strength indicator, phase-based distance estimation, and radio frequency fingerprinting. However, researchers have repeatedly demonstrated the insecurity of these RF-based proximity-proving techniques [20, 23, 24, 28, 56, 67] (**C1**: ✗). For instance, radio relay attacks [23] against the keyless entry system of modern cars, without cracking crypto-keys, are not only real [65] but also inexpensive ($22) [70]. Furthermore, as wired QR code readers lacking RF capabilities dominate the market [64], the generalizability issue associated with RF-based techniques cannot be overlooked (**C4**: ✗).

Distance-bounding protocols [11], designed to establish an upper bound on the distance between devices, have the potential to counter radio relay attacks [21]. Active research is underway to study their security aspects [17,44], while new attack methods continue to emerge [5, 7] (**C1**: **?**). The protocols are sensitive to even minor processing latency and thus require specialized hardware that is not yet widely available (**C2**: ✗). Moreover, due to the absence of standards, compatibility issues between devices of different vendors hinder the widespread adoption of the protocols (**C4**: ✗).

Motion-based authentication. Human action-based authentication has sparked numerous methods in different domains, spanning from smart homes [37, 53, 59] to drones [61, 71]. While a comprehensive review of all these methods exceeds the scope of this paper, we focus on approaches employing camera-IMU signal correlation for authentication. These methods compare motions estimated from camera data and recorded by IMUs. UniverSense [53] and PosePair++ [59] require users to move an IoT device equipped with an IMU in front of a camera, and IMU readings and acceleration data

estimated from images are used for pairing.[2] G2Auth [71] conducts authentication between drones and users by having users wave a smartphone in front of a drone's camera.

These methods translate 2D images captured by cameras into 2D acceleration data and compare it with IMU readings. In contrast, our work converts 2D images into fine-grained 3D pose information and incorporates it for authentication. IDIoT [58] estimates a plane of possible orientations based on the locations of human body joints in 2D images. However, the orientations cannot be uniquely determined from the joint locations in 2D images. Thus, as admitted by the authors, it cannot recover fine-grained orientation information. Furthermore, it assumes human bodies can be captured by cameras, while during payment a scanner mainly captures a smartphone displaying a QR code.

We introduce a mathematical model to estimate fine-grained 3D poses of smartphone from 2D images. Additionally, we propose a method to establish correlation between the estimated poses and IMU readings. Notably, PQRAUTH stands out as the *first* correlation-based authentication approach that utilizes fine-grained 3D pose data as a distinctive authentication factor.

3 Background

Two main QR code payment schemes have been proposed for different payment scenarios [43].

Consumer-presented QR code mode. To check out, a consumer opens a digital wallet app, e. g., PayPal, generates a unique one-time payment QR code with the app, which is then scanned. The scanner decodes the QR code and sends the decoded data to the payment server. The cashier receives a notification once the payment is verified and prints the sales receipt. Alternatively, in a self-service checkout system, consumers present their payment QR codes to the scanners themselves. The consumer-presented QR code payment procedure is similar to paying with a credit card, where the QR code functions as a one-time credit card.

To scan consumer-presented QR codes for transactions, merchants typically need to apply dedicated scanners and obtain necessary permissions from a Point of Sale (POS) provider [66]. Payment QR codes are not functional with unauthorized scanners.

For convenience, most digital wallet apps, such as Paypal and Venmo, do *not* require consumers to confirm transactions after presenting the payment QR codes to scanners. Some digital wallet apps do *not* need confirmation for transactions below a predefined amount. For example, no password or security verification is required by WeChat Pay for single transactions under RMB 1000 (approximately USD 140) [69]. In other apps, like Alipay [3], consumers have an option to set up a transaction limit, below which no confirmation is required. Despite the great usability (i.e., confirmation is not mandated), the payment is vulnerable to replay attacks, where an attacker replays the QR code presented by a consumer to another authorized scanner.

Merchant-presented QR code mode. In contrast to the consumer-presented QR code mode, upon checking out, the cashier, rather than the consumer, generates a dynamic QR code which contains the bill. The consumer launches the digital wallet app, scans the QR code, and confirms the transaction in the app. Alternatively, the merchant generates a static QR code containing their ID, and the consumer enters the bill after scanning the QR code and confirms the purchase on their smartphone. In this case, the merchant's static QR code can be printed on a portable paper, which is usually adopted by small businesses, such as street food vendors. In the merchant-presented QR code payment scheme, explicit confirmation is *mandatory* for each transaction [2].

In the merchant-presented QR code mode, the QR code obtained by an attacker cannot be exploited for illicit financial gains. Additionally, the security of this mode is strengthened by the mandatory consumer confirmation. In contrast, the consumer-presented QR code payment scheme is vulnerable to replay attacks, as demonstrated in previous research [8, 39, 52, 74, 80]. Therefore, we focus on consumer-presented QR code mode.

4 Threat Model and Overview

4.1 Threat Model

QR code replay attacks. The attacker can obtain the consumer's QR code in several ways. For example, they may physically get in proximity to a victim, and sneakingly record the victim's QR code with a smartphone. Note that the acquired QR code can be sent to a remote attacker to conduct a fraudulent transaction. Alternatively, an attacker may deploy cameras in the environment and take a sneak shot of the victim's QR code remotely [52]. Given that individuals typically have their payment QR codes displayed on their phone screens tens of seconds prior to cashiers scanning them [52], attackers could exploit this window to acquire the victim's QR code and use it before the legitimate scanning process takes place. Such attacks are also described in prior work [39, 52, 74]. Moreover, a variant of replay attacks, called *Synchronized Token Lifting and Spending* [8], describes a variety of means to disrupt a legitimate transaction, while the attacker can acquire and use the QR code.

Mimicry attacks. An attacker who knows the scheme of PQRAUTH may launch mimicry attacks. Note that the QR code generated by the victim contains the victim's account ID. A server only compares the poses of the smartphone being scanned with the sensor data reported by that account owner's

[2] UniverSense and PosePair++ are designed for pairing, but it is straightforward to adapt them to authentication.

smartphone. Consequently, an attacker must accurately replicate the real-time poses of the victim's smartphone, striving to achieve the highest possible resemblance.

Similar to acquiring the victim's QR code, the attacker can employ a camera to observe the poses of the victim's smartphone and mimic the poses simultaneously. However, it is important to note that the average human reaction time exceeds 200ms [26,31,45]. Therefore, it is unlikely for attackers to replicate the poses of the victim's smartphone in real time, and such time difference can be detected by our algorithm. We also assume that attackers have the capability to predict the victim's behaviors through various side-channels, such as leveraging auditory cues like the distinct beep sound produced by a successful QR code scan. This allows them to estimate the timing of the victim's scanning completion and adjust their own smartphone accordingly. However, research reveals that individuals tend to overestimate acute angles and underestimate obtuse angles when reproducing angles, even without time constraints [14,27]. Therefore, even if the attacker predicts the timing of the victim's smartphone movement, it is unlikely that they can accurately replicate the poses.

Perspective distortion attacks. An attacker, who knows the scheme of PQRAUTH and the difficulties of impersonating a user's operations in real time, may choose to perform perspective distortion on the replayed QR code. Specifically, the attacker employs computer vision techniques to analyze the poses of the victim's smartphone and meanwhile distorts the replayed QR code, rendering a visual effect that the replayed QR code's poses correlate with the poses of the victim's smartphone. However, the perspective distortion attacks require utilizing specialized algorithms and ample computing resources to lively infer poses of the victim's smartphone and distort the QR code, which make it less practical in real world. More importantly, the perspective distortion attacks can be easily detected, as the boundaries of the distorted QR code are not parallel to those of the smartphone screen.

A variant of the attack is to record a video or generate a fake video, e.g., using AI, which includes a smartphone that shows the victim's QR code and the poses of the victim's smartphone. Then, a large screen is used to show the video to the scanner. However, with this attack, the phone is not live but rendered using a screen. Either of the two distinct types of techniques can detect such attacks: (1) techniques that detect AI-created videos [40], and (2) techniques that detect whether the part showing phone boundaries belongs to a screen based on, e.g., moire patterns [52] and flickering [39].

Assumptions. Similar to prior work [39,52,74], we have the following assumptions. We assume that the attacker cannot physically get and unlock the victim's smartphone. We also assume a malicious app, if installed on the victim's smartphone, does not manipulate the sensor data, since the current smartphone's sensor security model allows only *read* access to sensitive sensors. We assume that those authorized scanners are not compromised, as retailers usually have strict polices regarding software installed in scanners.

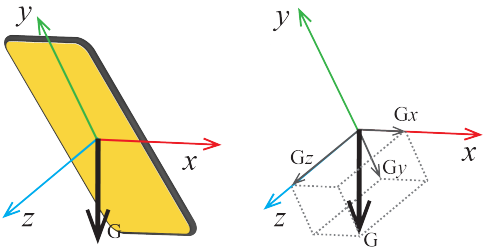


Figure 2: A smartphone's pose is represented as the orientation of the smartphone, which is indicated by the gravitational forces acting along the smartphone reference axes.

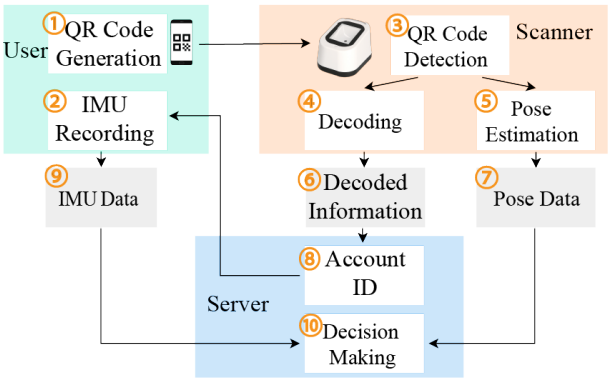


Figure 3: System architecture.

4.2 Overview

Our objective is to enhance the security of QR code payment without altering users' habits. In PQRAUTH, users follow a typical QR code payment process, while a factor of authentication based on poses is added *implicitly*. As shown in Figure 2, a smartphone's *pose* is represented as the orientation of the smartphone, which is reflected by the gravitational forces acting along the smartphone reference axes. The readings from the gravity sensor along the device reference axes, i. e., G_x, G_y, and G_z, represent the projections of gravity, which in turn indicate the orientation of the smartphone.

Figure 3 illustrates the system architecture, consisting of three entities: a smartphone owned by a user, a scanner used by a merchant, and a server that provides authentication services. A representative procedure of PQRAUTH is as follows; note that specific details may vary based on the implementation and payment scenarios.

1. After a bill is generated, the user unlocks the smartphone and launches the payment app to generate a payment QR code (①), and the smartphone starts recording its IMU data (②). The user shows the code to the scanner.

2. Once the code is detected by the scanner (③), a beep alert is emitted to indicate a successful scan, and the user may then put away the smartphone.

3. The scanner decodes the payment code (④) and estimates the poses of the smartphone displaying the code (⑤). The scanner sends the decoded information (⑥) and estimated poses to the back-end server (⑦).

4. Once the account ID is decrypted (⑧), the server fetches the IMU data from the app logged in with that ID (⑨).

5. The server makes a decision by comparing the IMU data with the pose data estimated by the scanner, and sends the payment result to the scanner and the smartphone (⑩). If the pose information estimated by the scanner matches with that recorded by the consumer's smartphone, the payment is authorized; otherwise, it is rejected, and the user is notified to re-generate a payment code and repeat the authentication procedure until the maximum number of attempts is reached.

It is worth clarifying that a payment QR code encodes the consumer's account ID. Thus, the server only compares the scanner-observed pose data with IMU data fetched from the payment app logged in with that account ID. In other words, the decision-making process involves a one-to-one verification problem rather than a one-to-many identification problem, meaning that its accuracy does not decline as the number of users increases.

5 System Design

On the smartphone side, we utilize IMU sensor data to determine the smartphone's poses. On the scanner side, we perform smartphone pose estimation (Section 5.1) from the recorded video and apply data pre-processing to eliminate noise and outliers (Section 5.2). After receiving the pose data from the scanner and the smartphone, the server determines their correlation (Section 5.3). For scanners without an IMU, we perform scanner orientation calibration, which is a one-time effort (Section 5.4).

5.1 Smartphone Pose Estimation

To determine the poses of the smartphone from the recorded video, we perform QR code detection to locate the QR code in the video frame. Within the QR code, three square *position markers* are situated at its top-left, top-right, and bottom-left corners. These markers facilitate the process of locating and orienting the QR code, enabling a QR code reader to identify it accurately.

After locating the QR code, we estimate the pose of the smartphone in the world coordinate system. When a QR code

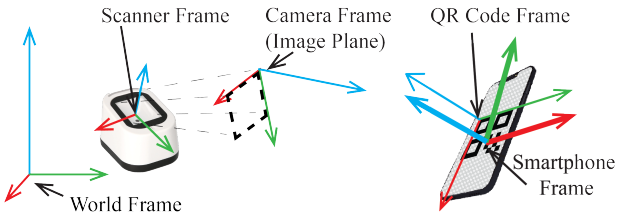


Figure 4: Transformation between coordinates. X-axis: red; Y-axis: green; Z-axis: blue.

is detected by the scanner, the rotation of the smartphone can be obtained by the following equation:

$$R_{World}^{Phone} = R_{World}^{Cam} \cdot R_{Cam}^{QR} \cdot R_{QR}^{Phone} \tag{1}$$

where R_{World}^{Phone} represents the rotation matrix from the world coordinate system to the smartphone coordinate system.

The matrix R_{World}^{Cam} denotes the rotation from the world coordinate system to the scanner camera coordinate system. It can be calculated by Equation 2:

$$R_{World}^{Cam} = R_{World}^{Scanner} \cdot R_{Scanner}^{Cam} \tag{2}$$

where $R_{World}^{Scanner}$ is the rotation matrix from the world frame to the scanner frame, and $R_{Scanner}^{Cam}$ is the rotation matrix from the scanner frame to the camera frame. For scanners with IMU integrated, the scanner frame is defined as their sensor coordinate system. $R_{World}^{Scanner}$ can be obtained from the IMU sensor built in the scanner. For instance, Android provides the `getRotationMatrix()` method, which uses the gravity sensor and the geomagnetic field sensor to get the rotation matrix for a device. Since the camera is fixed in the scanner, the pose of the camera is stable relative to the scanner, and the transformation $R_{Scanner}^{Cam}$ is constant and can be predetermined based on their relative poses. For example, in the case of a smartphone scanner, the camera frame can be obtained by rotating the smartphone's IMU coordinate system by 180 degrees along its x-axis. In summary, R_{World}^{Cam} can be calculated based on the scanner's IMU data. For scanners without IMU integrated, such as a hands-free scanner shown in Figure 4, we propose to perform scanner orientation calibration, which is discussed in Section 5.4.

The rotation matrix R_{Cam}^{QR} represents the rotation of the QR code with respect to the camera frame. We employ a Perspective-n-Point (PnP) based method [35], which minimizes the reprojection error from 3D-2D point correspondences, to estimate R_{Cam}^{QR}. Specifically, a matrix T_{Cam}^{QR} is defined to denote the homogeneous transformation from the camera coordinate system to the QR code coordinate system, as shown in Equation 3. It consists of the 3×3 rotation matrix R_{Cam}^{QR}, and a 3×1 position vector P_{Cam}^{QR}.

$$T_{Cam}^{QR} = \begin{bmatrix} R_{Cam}^{QR} & P_{Cam}^{QR} \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{3}$$

With the transformation matrix T_{Cam}^{QR}, the coordinates of the four corners of a QR code within the image plane can be described by Equation 4, as outlined in accordance with the pinhole camera model [78]:

$$\begin{bmatrix} U_i \\ V_i \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot T_{Cam}^{QR} \cdot \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{4}$$

where $[X_i\ Y_i\ Z_i]$ represents the 3D position vector of the i-th corner of the QR code, and $[U_i\ V_i]$ denotes the corresponding 2D position vector in the image frame captured by the camera. To simplify and without sacrificing generality, we designate the QR code frame's origin as the origin of the world frame. Given that the four corners of a QR code lie in the same plane, their z-axis coordinates Z_i within the QR code frame are uniformly set to zero. In Equation 4, f_x and f_y are the focal lengths expressed in pixel units, while (c_x, c_y) is the principal point which is at the image center. f_x, f_y, c_x, and c_y are intrinsic parameters of a camera and can be obtained via one-time preliminary camera calibration [78].

By localizing the four corners within the image plane and providing their corresponding 3D coordinates, the transformation matrix T_{Cam}^{QR} can be deduced utilizing Equation 4. The rotation matrix R_{Cam}^{QR} is then used to estimate the smartphone's orientation in the world coordinate system, as illustrated in Equation 1.

The matrix R_{QR}^{Phone} is defined as rotation from the QR code coordinate system to the smartphone's coordinate system. We use the smartphone's sensor frame to represent the smartphone's coordinate system since the sensor data are leveraged for authentication. As illustrated in Figure 4, the QR code coordinate system is a right-handed coordinate system with the corner of the middle position marker as the origin, while the x-axis and y-axis are represented by two margins. The sensor coordinate system is defined relative to the smartphone's screen, and its x-axis points to the right, y-axis points up, and z-axis points toward the outside of the screen. Since the coordinate systems of the QR code and the smartphone are predefined, the matrix R_{QR}^{Phone} is predetermined.

We have the following equation that describes the relationship between the gravity measured in the world frame and in the smartphone's frame:

$$\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = R_{World}^{Phone} \cdot \mathbf{gravity} = R_{World}^{Phone} \cdot \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} \tag{5}$$

where g is the constant value of gravity, and $[G_x\ \ G_y\ \ G_z]$ represents the gravity values measured by the smartphone along its three axes.

To compare the rotation information estimated by a scanner with the sensor data recorded by a smartphone, we calculate the gravity values estimated by the scanner with the formula adapted from Equation 5:

$$\begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} = {R_{World}^{Phone}}^{-1} \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = {R_{World}^{Phone}}^{\intercal} \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{6}$$

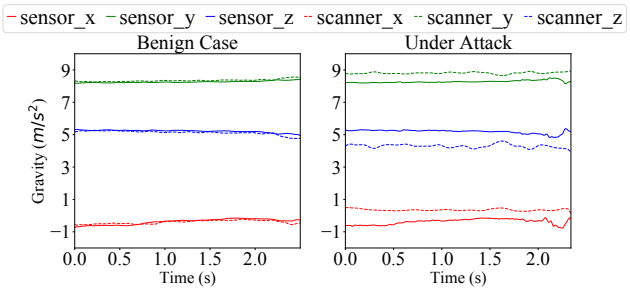


Figure 5: Poses, represented as gravity data, after pre-processing. Solid line: pose data collected by a smartphone sensor; Dashed line: pose data inferred by a scanner. In non-malicious scenarios, there is a robust correlation between the pose data derived from the scanner and that collected from the smartphone. Conversely, in instances of replay attacks, this correlation tends to deteriorate significantly.

We perform smartphone pose estimation for each frame captured by the scanner, resulting in a sequence of data points representing the smartphone's poses during the presentation of the QR code. Each data point in the sequence corresponds to the gravity values along the three axes of the smartphone's sensor coordinate system ($[G_x\ \ G_y\ \ G_z]$). This sequence is then sent to the server for correlation determination.

5.2 Data Pre-processing

Obtaining the smartphone's pose data inferred by the scanner and the user ID in the QR code, the server acquires the IMU data from the user's smartphone for comparison. The pose data determined by the scanner and collected by the smartphone's IMU may both fluctuate and contain noises. Therefore, we perform pre-processing operations on this data.

We use linear interpolation to fill gaps in the data that arise due to uneven sampling. IMU data may contain high frequency noises caused by environmental vibrations, such as sounds [30, 34], we apply a low-pass filter to filter out noises with high frequencies [60]. As the vibration caused by human mobility is mostly less than 10 Hz [15], we select a low-pass Butterworth filter with a cut-off frequency of 10Hz. We apply a median filter to remove outliers and smooth the data [25].

In this manner, two sequences of gravity data are obtained, denoted as: $G_S = \{G_S^{(1)}, G_S^{(2)}, \ldots, G_S^{(n)}\}$ estimated by the scanner, and $G_P = \{G_P^{(1)}, G_P^{(2)}, \ldots, G_P^{(n)}\}$ collected by the smartphone's IMU. Note that each data point in the gravity sequence contains three elements, each measures the force of

gravity that is applied to the smartphone on one of the sensor coordinate system's axes. The gravity data, as inferred by a camera and collected by a smartphone's IMU after pre-processing, is depicted in Figure 5.

5.3 Correlation Determination

After pre-processing, we obtain sequences of gravity data, G_S and G_P. Note that each data point in the gravity sequences contains three elements, each representing the gravity projections along the respective sensor axis.. To determine the correlation between the data sequences obtained from the camera and smartphone, we calculate the Euclidean distances for each attribute in the sequences separately. To account for variations in the length of data n each time a user presents a QR code, we obtain the average distances by dividing the Euclidean distances of each attribute by n. The resulting averaged Euclidean distances are used as features for training a classification model.

The IMU has demonstrated its reliability as an instrument for evaluating various ranges of motion, as noted in previous research studies [9, 18, 51]. Even low-cost IMUs can offer remarkable precision, with a maximum error of approximately 0.5 degrees [33]. Particularly for short time intervals, the influence of IMU drift resulting from integration errors, sensor bias, and sensor noise is minimal [51]. However, the accuracy of smartphone sensors can vary across different devices. To further strengthen the resilience of our system and account for potential inaccuracies in IMU measurements, we propose utilizing correlation to assess the similarity of the **shape** between data sequences. This approach is based on the observation that despite the inherent inaccuracies and drift in IMU sensor data, the overall trend, whether increasing or decreasing, tends to be reliable. Specifically, we employ the Pearson correlation coefficient (PCC) [16], which is a widely used algorithm for quantifying the correlation between two time series. The PCC is calculated as the covariance of two data sequences divided by the product of their standard deviations. The resulting PCC value ranges from -1 to 1. A PCC value of 1 indicates a perfect positive linear relationship between the sequences, while a value of -1 suggests a perfect negative linear relationship. A PCC value of 0 indicates the absence of a linear relationship between the sequences.

In addition to utilizing the Euclidean distances and PCC scores, we also leverage the following features regarding the differences between data sequences: *minimum*, *maximum*, *difference between minimum and maximum*, *average*, *standard deviation*, *median absolute deviation*, *and median*.

We further calculate the fisher score to select fundamental features. The normalized fisher scores are shown in Table 2. Features with a normalized fisher score above 0.1 are selected.

Table 2: Normalized fisher scores of features.

Feature	Pose		
	X-axis	Y-axis	Z-axis
Euclidean	**1.00**	**0.24**	**0.39**
PCC	**0.36**	**0.26**	**0.21**
Minimum	**0.20**	0.10	**0.12**
Maximum	**0.92**	**0.23**	**0.44**
Max-Min Diff	**0.41**	**0.19**	**0.35**
Average	**0.99**	**0.27**	**0.38**
Std	**0.39**	**0.15**	**0.27**
MAD	**0.24**	0.08	**0.24**
Median	**0.79**	**0.26**	**0.79**

5.4 Scanner Orientation Calibration

The smartphone orientation estimation from the recorded video requires the knowledge of the scanner's orientation. (1) Scanners that are equipped with IMU, such as smartphones, can obtain $R_{World}^{Scanner}$ from the sensor data, as described in Section 5.1. (2) For scanners without IMU, we propose to do scanner orientation calibration with the help of an off-the-shelf smartphone, which is a one-time effort. Specifically, a cashier shows a smartphone displaying a QR code to a scanner to be calibrated; meanwhile, the scanner estimates the rotation of the smartphone and the smartphone records its rotation with built-in sensors. Note that the calibration we perform is for extrinsic parameters corresponding to the scanner's orientation. The intrinsic parameters, which are solely determined by the scanner's camera and remain constant after the factory calibration, are not affected. It is also worth noting that recalibration is unnecessary as long as the pose of the scanner remains constant, meaning that $R_{World}^{Scanner}$ remains unchanged, even if the scanner is relocated. For example, on a flat checkout counter, the scanner can be moved without necessitating recalibration.

Based on Equation 1, Equation 2, and Equation 5, we can derive the following:

$$\begin{aligned}\begin{bmatrix}0\\0\\g\end{bmatrix} &= R_{World}^{Scanner}\cdot R_{Scanner}^{Cam}\cdot R_{Cam}^{QR}\cdot R_{QR}^{Phone}\cdot\begin{bmatrix}G_x\\G_y\\G_z\end{bmatrix}\\ &= R_{World}^{Scanner}\cdot Q\end{aligned} \tag{7}$$

where $R_{World}^{Scanner}$ represents the rotation matrix from the world coordinate system to the scanner frame and Q is the multiplication of multiple matrices and vectors that are known. As a rotation matrix has only three rotational degrees of freedom, $R_{World}^{Scanner}$ can be determined with a one-time calibration procedure, even though multiple operations can enhance accuracy.

To evaluate the effectiveness of the proposed scanner orientation calibration, we use a smartphone placed on a holder to serve as a scanner, mimicking the calibration process for hands-free QR code scanners widely adopted by retailers and

Figure 6: Four pairs of smartphones. Each pair contains two identical phones.

merchants. During this process, we consider the IMU sensor data of the scanner as ground truth and compare the calibration results with the IMU sensor data. Our results indicate that, with the one-time calibration, the average angular error along three axes is 0.04 radian (2.3 degrees), demonstrating the effectiveness of the proposed calibration method. Additionally, we evaluate the authentication accuracy of PQRAUTH based on the calibration results. PQRAUTH achieves an equal error rate (EER) of only 0.01 with the one-time calibration, a performance level similar to using the scanner's IMU data for orientation determination. These findings underscore the accuracy and efficiency of the proposed calibration method.

6 Datasets

To evaluate how PQRAUTH performs in determining the correlation between two sequences of pose data, we conduct experiments and build two datasets. Dataet I is collected for normal cases, i. e., without attacks, while Dataset II is collected with mimicry attackers. We recruited 20 participants, including 10 males and 10 females aged range from 18 to 63. It is worth noting that PQRAUTH tackles a verification problem where each individual is matched to a unique identity, rather than an identification problem where one individual is matched against multiple identities. Thus, the accuracy of PQRAUTH remains consistent even as the user base expands. All the participants have prior experience with QR code payment services. The experiments are performed with the approval of an Institutional Review Board (IRB).

6.1 Devices

Without loss of generality, we generate a version-2 (25×25) QR code as the payment code. We employ smartphones as scanners, a strategy consistent with previous work [39, 52, 74, 80]. These smartphones also serve as consumer owned devices for displaying QR codes. To ensure that the attacker's scanner has the same orientation as the victim's scanner, we use two identical smartphone holders with fixed tilt angles. Figure 6 illustrates the smartphones used in our experiments, including Nexus 5X, Pixel 3A, Pixel 4, and Pixel 6.

6.2 Dataset I

To build Dataset I, we instruct the participants to present the payment code to a scanner (Pixel 6) individually. The scanner is positioned on a smartphone holder to emulate a hands-free QR code scanner. The participants are requested to present a QR code with a smartphone (Pixel 3A) to the scanner in the same manner they normally use a QR code payment service. They are allowed to hold the smartphone with any hand and in any way they feel most comfortable. After each successful QR code scan, a notification sound is produced by the scanner to indicate a success, and participants are then free to put away the smartphone. Each participant performs 50 times of authentication operations.

6.3 Dataset II

For the construction of Dataset II, participants are randomly paired, where each pair consists of one individual acting as the victim and the other as the attacker. These roles are later interchanged between them. We ensure that the attacker participant uses identical devices as the victim participant. Specifically, two Pixel 6 smartphones are positioned on two identical holders to serve as scanners. These holders are placed parallel to each other at the same height, with a 70-centimeter distance between them. This distance is selected to ensure that the attacker participant can closely observe the victim participant's authentication actions without interfering with their operations. The victim participant, using a Pixel 3A, is told to perform transactions in the same manner as they would typically do at a retail store. The attacker participant, also using a Pixel 3A, is instructed to imitate the victim's actions as closely as possible. Both scanners provide feedback in the form of beeps if they detect a QR code.

The experimental settings inherently favor attackers, as they can closely and clearly observe the victim's behaviors in real time. By contrast, in real-world scenarios where attacks are launched remotely to maintain stealth, attackers need to use cameras to spy on the victim and then observe the victim in a screen. This introduces additional delays and also results in less clear observation of the victim's actions. Moreover, even if attacks are conducted at the same location, it is unlikely that attackers would have such a clear view, as checkout counters are typically obstructed with boards.

7 Evaluation

7.1 Accuracy

Metrics. We use the *False Acceptance Rate* (FAR) and the *False Rejection Rate* (FRR) as two of the metrics to evaluate the accuracy of PQRAUTH. FAR denotes the percentage of instances where attacks are incorrectly authenticated and a lower FAR indicates higher security. FRR shows the percent-

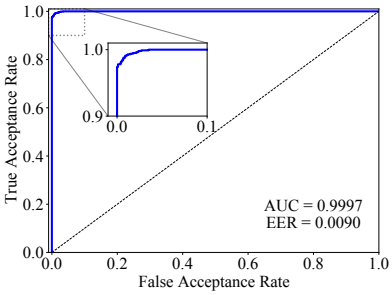


Figure 7: ROC curve, AUC, and EER.

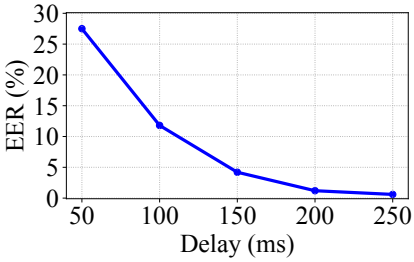


Figure 8: Robotic mimicry attacks with delays.

age of legitimate authentication instances are rejected and a lower FRR shows better usability. We also report *Equal Error Rate* (EER) which corresponds to the point at which the FRR is equal to the FAR. A lower EER indicates a higher accuracy (1-EER) of a system. We also use a *Receiver Operating Characteristics* (ROC) curve to show the accuracy of our system across all possible thresholds and report the *Area Under the Curve* (AUC) of the ROC curve.

Human Mimicry Attacks. From Dataset I, we form the data sequence uploaded by the scanner and the sensor data collected by the smartphone for each authentication as a positive data pair, and label it as 1. From Dataset II, we form the data sequence uploaded by the scanner which scans the attacker-presented QR code and the sensor data collected by the victim's smartphone for each authentication as a negative data pair, and label it as 0. We adopt a strict mechanism, Leave-One-Subject-Out (LOSO), to train an SVM model and evaluate PQRAUTH without user bias. That is, we iteratively choose the data of one subject for testing and use the data of the other 19 subjects to train the system, and report the average performance of these systems.

We compare PQRAUTH with a **baseline** approach that utilizes camera-IMU signal correlation for authentication. UniverSense [53] and G2Auth [71] confine user motions to a 2D plane parallel to the camera, and PosePair++ [59] projects user's motions onto a 2D plane parallel to the camera's image plane to estimate the acceleration data. As the baseline, we also project the smartphone's positions onto a 2D plane parallel to the scanner's image plane to estimate the smartphone's 2D acceleration. The 3D pose data is not used in the baseline. The results of the baseline show that the EER is 0.244.

The results of PQRAUTH are shown in Figure 7. The average EER is 0.009, and the AUC is 0.9997. The low EER indicates that PQRAUTH achieves a high accuracy of 0.991 in distinguishing benign users and attackers. Thus, our approach significantly outperforms the baseline, highlighting the importance of 3D pose data for accurate authentication.

Robotic Mimicry Attacks. To delve deeper into PQRAUTH's vulnerabilities, we also explore how a robot could potentially attack the system. We assume a humanoid robot which can mimic consumer's behaviors perfectly. However, the physical responses of a robot always suffer from delays imposed by physics, and these delays cannot be reduced beyond a fundamental limit [62]. For instance, even leading humanoid robots like NAO, which has a high cost of $9,000, require 200ms to execute a motion command [22]. Another work gives a detailed analysis of the delay and considers the minimum delay as 180 ms (including 80 ms of mechanical delay) in their evaluation [79]. But note the delay does not include the computer vision analysis. One study shows the end-to-end delay from human-waving to robot-waving can take up to 1.72 seconds [13]. Thus, we consider a perfect robot but with a reasonable delay.

To evaluate how resilient PQRAUTH is to robotic mimicry attacks, we construct negative data pairs by shifting the sensor data in Dataset I, and label them as 0. Suppose a robot has a delay of 50 ms, then the sensor data is shifted by the same amount. With the positive data pairs and negative data pairs from Dataset I, we train an SVM model to evaluate the accuracy of PQRAUTH. Figure 8 shows the EERs of PQRAUTH under robotic mimicry attacks with different delays. For mimicry robots with delays greater than 200 ms, PQRAUTH achieves EERs less than 0.012, demonstrating its resilience to robotic mimicry attacks.

7.2 Entropy

Entropy is a widely accepted metric for assessing the security strength of authentication methods, such as passwords [32] and PINs [68]. In the context of PQRAUTH, we employ **entropy** to evaluate the randomness of the poses. This randomness directly reflects the difficulty faced by attackers attempting to guess these poses. The formula for calculating the entropy of a variable x, given the distribution $P(x)$, is expressed as $E = -\sum_{x \in X} P(x) \log P(x)$. To assess the entropy of individual users, we analyze the pose distributions gathered from our dataset. Note that each data point comprises gravity along the three axes of the smartphone sensor coordinate system (see Section 5.2). The gravitational forces along the three axes exhibit totally two degrees of freedom [36]. To quantify the overall entropy, we first compute the average entropy of the gravitational force along each axis and then double it to take into account the degrees of freedom. The resulting average entropy, calculated across all individuals, amounts to

14.4 bits. For comparison, human-chosen passwords typically have an entropy ranging from 14 to 30 bits [32]. Authentication mechanisms based on four-digit PINs exhibit an entropy of 8.4 bits, while those employing six-digit PINs have an entropy of 13.2 bits [68]. These findings underscore the substantial uncertainty due to PQRAUTH, further establishing its potential as a promising biometric authentication factor.

PQRAUTH not only evaluates a single pose presented by a user but also considers the dynamic poses within the authentication process. Consequently, attackers must guess not only the victim's static pose but also a sequence of poses. To measure the unpredictability of the dynamic poses, we employ **approximate entropy** (ApEn), which a metric utilized to quantify the level of randomness and unpredictability within time series data [57]. In essence, given a time series of length N, $ApEn(m,r,N)$ is approximately equal to the negative average logarithm of the conditional probability that two subseries of length m, which are similar within a tolerance specified by $\pm r$ times the standard deviation of the time series, will remain similar for subseries of length $m+1$. ApEn generates a unitless value that falls within the range of 0 to 2, where 0 corresponds to a perfectly regular time series (e. g., a periodic signal), while 2 signifies a completely random time series (e. g., Gaussian noise) [47]. Some prior work has also employed ApEn to gauge the unpredictability of various authentication methods. For instance, For EEG signals, ApEn is 0.25 bits [48], for ECG signals it is under 0.21 bits [29], and for gait at a comfortable speed it is 0.53 [63]. To calculate the total ApEn of PQRAUTH, we first compute the average ApEn for the gravitational force along each axis and then double it. Based on our datasets, the ApEn is calculated to be 0.52 bits. This indicates that the dynamic poses during a QR code payment is more unpredictable compared to EEG [48] and ECG [29], and comparable to gait [63].

7.3 Parameter Study

QR Code Reader. We employ a variety of cutting-edge QR code readers that are publicly available, including ZBar [77], ZXing [81], BoofCV [10], OpenCV [49], and WeChat [50]. The results of our evaluation, which reveal EER_{ZBar}=0.011, EER_{ZXing}=0.012, EER_{BoofCV}=0.010, EER_{OpenCV}=0.010, and EER_{WeChat}=0.009, indicate that there are negligible differences in accuracies across all the QR code readers. PQRAUTH performs effectively regardless of the specific QR code reader employed, demonstrating its robustness.

Classifier. We employ three different classifiers, SVM, kNN, and Random Forest, to train the model. We examine different kernels for SVM, including linear, polynomial, and radial basis function (RBF), and find RBF to be the most effective. We use grid search to determine the optimal hyperparameters for SVM, setting c to 10 and γ to 0.01. We also experiment with different values of k for kNN, ranging from 1 to 20, and find that 5 is the optimal value. We test different numbers of trees for Random Forest, ranging from 50 to 200, and find that 80 is the optimal value. The results, which show EER_{SVM}=0.009, EER_{RF}=0.012, and EER_{kNN}=0.013, indicate that SVM has the lowest EER.

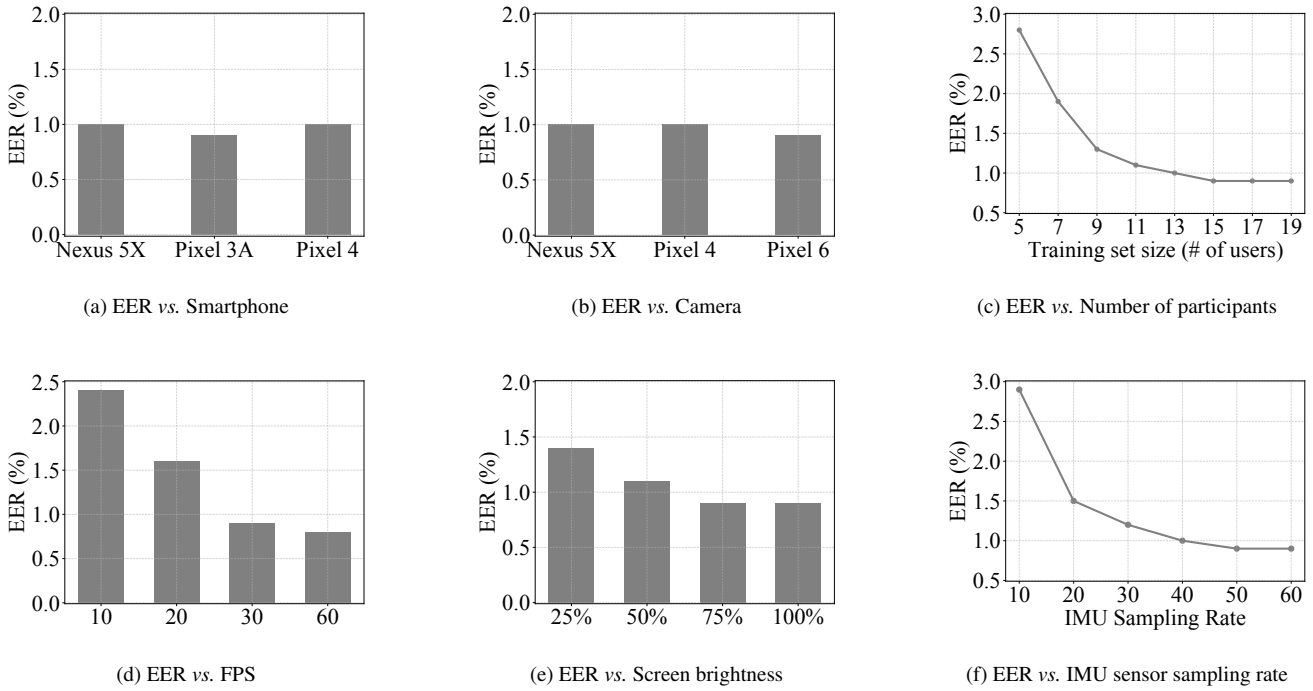
Smartphones. In this study, we investigate the impact of smartphones on PQRAUTH. We utilize smartphones of different sizes and weights, including the Nexus 5X, Pixel 3A, and Pixel 4. Victim participants choose one of the smartphones, and attacker participants are instructed to use smartphones of the same model as those used by the victims to launch mimicry attacks. The results, depicted in Figure 9a, demonstrate no significant performance difference among the smartphones. Therefore, we conclude that regardless of the smartphone used, the accuracy of PQRAUTH keeps high.

Cameras. This study aims to determine if PQRAUTH works equally well on different scanners. To do this, we use three different smartphones, namely the Nexus 5X, Pixel 4, and Pixel 6, as scanners. Both the victim and attacker participants use the Pixel 3A smartphones to display the QR code. The results, shown in Figure 9b, indicate that there is no significant difference in performance between the three cameras. Therefore, we conclude that cameras have little impact on the performance of PQRAUTH.

Training Dataset Size. We evaluate the impact of training dataset size on the system performance. The training dataset size is defined as *the number of participants* for training. We train PQRAUTH with data from 5 to 19 participants, and test the accuracy of PQRAUTH with data of the reset of participants. The results in Figure 9c shows that the accuracy converges when the number of participants in the training dataset reaches 15. It indicates that PQRAUTH only needs a few data samples in the training phase to determine the threshold.

Camera FPS. We study the impact of camera FPS by varying its frame rate while fixing the camera resolution to 1080P. We have the smartphone's camera to record videos at 10, 20, 30, and 60 FPS, separately. As shown in Figure 9d, the higher FPS, the higher accuracy of PQRAUTH. Especially, a significant improvement can be observed when the FPS is increased from 10 to 20. Considering a higher FPS requires more resources, and the improvement from 30 to 60 FPS is less significant, we set the FPS as 30, which is also the default frame rate of most smartphones.

Camera Resolution. We study the impact that camera resolution has on PQRAUTH, by setting the smartphone's camera to different resolutions. We set the smartphone's resolution to the most popular resolutions supported by mobile devices, including 480P (720×480), 720P (1280×720), and 1080P (1920×1080). The results show that increasing the camera resolution leads to higher accuracy, as evidenced by EER_{480P}=0.016, EER_{720P}=0.011, and EER_{1080P}=0.009. Notably, even at 480P resolution, PQRAUTH still achieves acceptable accuracy, indicating that high-resolution scanners are



(a) EER *vs.* Smartphone

(b) EER *vs.* Camera

(c) EER *vs.* Number of participants

(d) EER *vs.* FPS

(e) EER *vs.* Screen brightness

(f) EER *vs.* IMU sensor sampling rate

Figure 9: Impact of different parameters.

not necessary for its deployment, making it an economically viable solution.

Screen Brightness. We evaluate the performance of PQRAUTH in four screen brightness levels, i. e., the brightness on the consumer side is set to 25%, 50%, 75%, and 100% of the highest level. On the attacker side, the brightness level is set 100% to favor the attacker. Figure 9e shows the corresponding results. The accuracy of PQRAUTH remains consistent under different brightness levels. However, the brighter the screen is, the easier it is to capture the QR code, resulting in less time required for authentication. We thus recommend that smartphones set their screen brightness to the highest level when presenting QR codes. It should be noticed that the recommendation is not specific for PQRAUTH, most mobile payment apps, such as AliPay, PayPal, and Venmo, adjust the screen brightness to the highest level automatically when generating a payment code.

IMU Sensor Sampling Rate. Increasing the sampling rate of the IMU sensor data in smartphones can capture more subtle characteristics, but it also comes with higher burdens such as data collection and power consumption. To determine the optimal sampling rate, we conducted a study by down-sampling the original sensor data and analyzing the performance at different sampling rates. We vary the sampling rate from 10Hz to 60Hz in steps of 10Hz. The results, shown in Figure 9f, indicate that the performance significantly improves when the sampling rate increases from 10Hz to 20Hz, and it stabilizes when the sampling rate reaches 50Hz. Therefore, we choose the sampling rate of 50Hz, which is well within the capabilities of most IMU sensors.

Environmental Brightness. To study the impact of environmental brightness, we use dimmable LED bulbs and adjust their illuminance levels. The recommended lighting levels for retail stores, where mobile payments are frequently happen, range from 200 to 500 lux [6]. We thus change the environmental brightness level from 200 to 500 lux in steps of 100 lux. The results show that the environmental brightness has a very small impact on the system performance and PQRAUTH can work in a wide range of illuminance levels. This is because the smartphone screens are bright enough to clearly show the QR code, under various illuminance levels in buildings.

Real World Retail Stores. We evaluate how PQRAUTH performs in various real world retail stores, including McDonald's, ALDI, CVS, Walmart, and Costco. McDonald's is a fast-food restaurant, ALDI is a grocery store, CVS is a pharmacy, Walmart is a supermarket, and Costco is a warehouse club. PQRAUTH consistently attains a remarkable accuracy rate exceeding 99% across diverse retail stores. There is no discernible variance in PQRAUTH's performance between controlled laboratory environments and real-world retail establishments, underscoring its robustness.

Table 3: Authentication time. Our experiments show that the authentication decision is consistently made before the user operation is complete. Thus, compared to standard QR code payment, the latency overhead of PQRAUTH is zero.

Part	Average Time (ms)	Std (ms)
User Operation	3573	722
Data Transmission	66	8
Data Processing and Decision Making	55	8

7.4 Authentication Time

We measure the time needed for a QR code payment enhanced with PQRAUTH, which starts when the user launches the app and ends when the authentication decision is made or the user operation is complete, whichever is later. The user operation of presenting a QR code is complete when the user hears the beep alert and thus puts away her smartphone (i.e., naturally holding it). Our experiments show that the authentication decision is consistently made before the user operation is complete. In other words, during the time the user puts away her smartphone, the data transmission, processing and decision making are conducted.[3] The breakdown of authentication time is detailed in Table 3. Note that the user operations remain consistent, with or without PQRAUTH. Therefore, PQRAUTH incurs a latency=0, compared to standard QR code payment.

8 User Study

8.1 Recruitment and Design

We recruit 40 participants aged from 18 to 68, including 20 females and 20 males. The participants come from a variety of occupational backgrounds, such as students, lecturers, public servants, university staff, and retirees. Out of the 40 participants, 35 are familiar with the standard QR code payment method, and 32 have used it before. To avoid social desirability bias, we do not make the participants aware of the fact that PQRAUTH is designed by us. Instead, the participants are told to evaluate the usability of different QR code payment methods. We develop a customized app capable of using both the standard QR code and PQRAUTH scheme. The usability study is conducted under an IRB approval and each subject is asked to sign a consent form.

The evaluation procedure consists of three steps: the pre-study instruction, the experiment, and the post-study questionnaire. We first introduce the two authentication methods to the participants by playing a pre-recorded instructional video. We explain the rationale and operations of the two QR code payment systems and tell them the goal of this study is to compare their usability. Each participant is given a smartphone and use our customized app to perform 5 attempts, i. e., showing a QR code to a scanner, for each of the payment methods. Participants are free to ask questions regarding these authentication methods.

In the experiment step, each subject performs another 10 attempts using each payment method. The orders of using each payment method is counterbalanced, i. e., with half of the tasks beginning with standard QR code payment and the other half PQRAUTH. The measurement of the time needed for a QR code payment is described in Section 7.4.

At the end of the study, each subject fills in a questionnaire and judges the two mobile payment methods by answering six questions, which are adapted from the widely-used SUS [12]. The six questions are about usability scales of the two systems. Specifically, the six questions assess (Q1) *if the payment method is easy to learn*, (Q2) *if the payment method is easy to use*, (Q3) *if the payment method is efficient in time*, (Q4) *if they feel the system is secure enough*, (Q5) *if they feel comfortable to use the payment method*, and (Q6) *if they are willing to adopt the payment method*. The scores for each question range from 1 to 5, where 1 indicates strongly disagree and 5 strongly agree. Thus, higher scores indicate better usability. Participants are also encouraged to leave suggestions and comments in the questionnaire.

Table 4: User study results.

	Standard QR Code	PQRAUTH	p-value
Q1	4.95 ± 0.21	4.90 ± 0.30	0.33
Q2	4.85 ± 0.35	4.85 ± 0.35	1.00
Q3	4.70 ± 0.56	4.65 ± 0.33	0.72
Q4	2.15 ± 1.15	4.65 ± 0.57	0.00
Q5	3.05 ± 0.86	4.75 ± 0.70	0.00
Q6	3.85 ± 1.06	4.75 ± 0.54	0.00
Total	23.55 ± 1.80	28.55 ± 1.16	0.00

8.2 Usability Study Results

We use paired *t-tests* to check for differences in the opinions towards each payment method. The p-value of the tests indicates the statistical significance of the results. If the p-value is less than 0.05, we reject the null hypothesis that there is no difference between the means and thus conclude that a significant difference *does* exist. If the p-value is larger than 0.05, a significant difference does *not* exist. Table 4 shows the scores for the standard QR code payment and PQRAUTH.

Participants find PQRAUTH easy to learn and use, and they are satisfied with the payment speed. User perception regarding ease of learning, ease of use, and efficiency in time does not significantly differ between the two payment methods. Participants agree that PQRAUTH is more secure, according to the average score (4.65) towards Q4, which is significantly greater than that (2.15) achieved by the standard QR code payment method ($p = 0.00$). Participants express

[3] It is worth highlighting that, on the scanner side, the frame collection, QR code detection and pose estimation are processed in a *pipeline*, rather than batch-processing the frames after they are all collected. Thus, the computations in the scanner and the user operation are parallel.

(a) Hands-free

(b) Hand-held

Figure 10: Various mainstream scanners.

that they are more comfortable to use PQRAUTH (Q5: 4.75) than a standard QR code (Q5: 3.05). Also, they are more willing to adopt PQRAUTH. The standard QR code payment method achieves 3.85 towards Q6, while PQRAUTH achieves 4.75. In total, PQRAUTH achieves significant higher scores (28.55) than the standard QR code payment method (23.55) due to its perceived higher security.

In summary, users perceive PQRAUTH to be easy-to-learn, easy-to-use, efficient in time, secure, and comfortable, and they are willing to adopt PQRAUTH for mobile payment.

9 Discussion

Can PQRAUTH be widely used? To estimate the rotation of a smartphone from the video recorded by a scanner, we need to know the scanner's orientation. Mainstream scanners in the market can be categorized into two groups: hands-free scanners and hand-held scanners, as shown in Figure 10. We admit this limitation: if a store only provides hand-held scanners, PQRAUTH cannot work (unless the scanners are modified to carry an IMU sensor). We conduct an extensive survey of various retail stores in our city, encompassing fast-food restaurants (such as McDonald's and Subway), grocery stores (such as ALDI and Love's), pharmacies (such as CVS and Walgreens), supermarkets (such as Walmart and Target), and warehouse stores. Among the 25 stores equipped with QR code scanners, all provide hands-free scanners, meaning that PQRAUTH can be used in all the stores. Detailed information can be found in Appendix A.

For other applications that use QR codes, such as gate access control in buildings and ticketing services for subways, they typically provide hands-free scanners, as shown in the leftmost photo in Figure 10. Thus, PQRAUTH can be readily applied. For applications that use smartphones as scanners, such as social media apps, PQRAUTH works as well, since smartphones have built in IMU sensors [37].

In addition, if the hands-free scanner is not placed on a flat counter, it needs to be re-calibrated every time the scanner is moved (Section 5.4).

Is network required for the smartphone? Some mobile payment apps, such as Alipay, WeChat, PayPal, and Venmo, support offline payments. To enable PQRAUTH to operate independently of network connections, we put forth a simple yet effective approach. We propose to have the user's smartphone display encrypted IMU data together with the payment QR code on its screen. Alternatively, the encrypted IMU data can be embedded into the payment QR code, simplifying the process for the scanner, which only needs to detect the QR code. The primary idea is to employ the visual channel, rather than the network connection, to transmit the IMU data from the smartphone to the scanner for verification.

The fundamental process for authentication is outlined below. The scanner decodes the QR code and sends the information to the server. The server extracts the consumer's account ID from the received information and deciphers the IMU data using the key unique for each account. It then compares the decrypted IMU data with the poses captured by the scanner.

Repeated attacks. To increase the success rate of mimicry attacks, an adversary could learn a user's motion pattern by forcing the user to present QR code repeatedly. For instance, the attacker might deploy a malicious reader to intentionally fail in reading the QR code, prompting the user to repeatedly display it. As a result, the attacker could observe and learn the user's motion pattern before conducting the replay attack. However, as highlighted in Robotic Mimicry Attacks (Section 7.1), the timing is important. If an adversary shows poses too early or late, the attack will still fail. Thus, one possible countermeasure is to add a random but short delay before a new QR code is generated, such that the mimicry attacker cannot predict the timing of the user's next payment attempt.

10 Conclusion

QR codes have been widely adopted for mobile payment, making their authentication an important problem. We presented PQRAUTH, a highly usable implicit authentication method to enhance the security of QR code payment. It is the first correlation-based approach to enhancing QR code security. PQRAUTH leverages the unclonable, real-time poses when a consumer presents a QR code. Authentication proceeds only if the poses inferred by a scanner match the poses of the consumer's smartphone. Our method attains an accuracy of 0.991 in the real-world scenario where mimicry attacks are launched with zero latency overhead. Extensive experiments demonstrate its accuracy, security, and robustness. PQRAUTH has potential applications to other scenarios that use QR codes, such as gate access control and ticketing services.

Acknowledgements

This work was supported in part by the US National Science Foundation (NSF) under grants CNS-2309550 (CAREER Award) and CNS-2309477. The authors would like to thank the anonymous reviewers for their valuable comments.

References

[1] Alex Ng. Headaches? Eyestrain? Nausea? Own a Smartphone With an OLED Screen? Here's What It Could Be, 2022. https://medium.com/@alexngwrites/headaches-eyestrain-nausea-own-a-smartphone-with-an-oled-screen-heres-what-it-could-be-4e8951c366a6.

[2] Alipay. Merchant-presented Mode Payment, 2021. https://docs.alipayplus.com/alipayplus/alipayplus/product_intro/cscanb.

[3] Alipay. User-Presented Mode Payment, 2021. https://global.alipay.com/docs/ac/ams_upm/introduction.

[4] Ross Anderson and Steven J Murdoch. EMV: Why Payment Systems Fail. *Communications of the ACM*, 57(6):24–28, 2014.

[5] Claudio Anliker, Giovanni Camurati, and Srdjan Capkun. Time for Change: How Clocks Break UWB Secure Ranging. *arXiv preprint arXiv:2305.09433*, 2023.

[6] Archtoolbox. Recommended Lighting Levels in Buildings, 2021. https://www.archtoolbox.com/recommended-lighting-levels/.

[7] Gildas Avoine, Muhammed Ali Bingöl, Ioana Boureanu, Srdjan Čapkun, Gerhard Hancke, Süleyman Kardaş, Chong Hee Kim, Cédric Lauradoux, Benjamin Martin, Jorge Munilla, et al. Security of Distance-bounding: A Survey. *ACM Computing Surveys (CSUR)*, 51(5), 2018.

[8] Xiaolong Bai, Zhe Zhou, XiaoFeng Wang, Zhou Li, Xianghang Mi, Nan Zhang, Tongxin Li, Shi-Min Hu, and Kehuan Zhang. Picking Up My Tab: Understanding and Mitigating Synchronized Token Lifting and Spending in Mobile Payment. In *USENIX Security Symposium (USENIX Security)*, 2017.

[9] Srikanth Sagar Bangaru, Chao Wang, and Fereydoun Aghazadeh. Data Quality and Reliability Assessment of Wearable EMG and IMU Sensor for Construction Activity Recognition. *Sensors*, 20(18):5264, 2020.

[10] BoofCV. Main Page, 2023. http://boofcv.org/index.php?title=Main_Page.

[11] Stefan Brands and David Chaum. Distance-bounding Protocols. In *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1993.

[12] John Brooke. SUS: a "Quick and Dirty' Usability. *Usability Evaluation in Industry*, 1996.

[13] Gerard Canal, Sergio Escalera, and Cecilio Angulo. A Real-time Human-robot Interaction System based on Gestures for Assistive Scenarios. *Computer Vision and Image Understanding*, 149:65–77, 2016.

[14] Linda C Carson and Fran Allard. Angle-drawing Accuracy as an Objective Performance-based Measure of Drawing Expertise. *Psychology of Aesthetics, Creativity, and the Arts*, 7(2):119, 2013.

[15] Wenqiang Chen, Lin Chen, Yandao Huang, Xinyu Zhang, Lu Wang, Rukhsana Ruby, and Kaishun Wu. Taprint: Secure Text Input for Commodity Smart Wristbands. In *The 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[16] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson Correlation Coefficient. *Noise Reduction in Speech Processing*, 2009.

[17] Cas Cremers, Kasper B Rasmussen, Benedikt Schmidt, and Srdjan Capkun. Distance Hijacking Attacks on Distance Bounding Protocols. In *2012 IEEE Symposium on Security and Privacy (S&P)*, 2012.

[18] Antonio I Cuesta-Vargas, Alejandro Galán-Mercant, and Jonathan M Williams. The Use of Inertial Sensors System for Human Motion Analysis. *Physical Therapy Reviews*, 15(6):462–473, 2010.

[19] Alexei Czeskis, Karl Koscher, Joshua R Smith, and Tadayoshi Kohno. RFIDs and Secret Handshakes: Defending Against Gghost-and-leech Attacks and Unauthorized Reads With Context-aware Communications. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS)*, 2008.

[20] Boris Danev, Heinrich Luecken, Srdjan Capkun, and Karim El Defrawy. Attacks on Physical-layer Identification. In *Proceedings of the third ACM Conference on Wireless Network Security*, 2010.

[21] Saar Drimer, Steven J Murdoch, et al. Keep Your Enemies Close: Distance Bounding Against Smartcard Relay Attacks. In *USENIX Security Symposium (USENIX Security)*, 2007.

[22] Sylvain Filiatrault and Ana-Maria Cretu. Human Arm Motion Imitation by a Humanoid Robot. In *2014 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, 2014.

[23] Aurélien Francillon, Boris Danev, and Srdjan Capkun. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2011.

[24] Lishoy Francis, Gerhard P Hancke, Keith Mayes, and Konstantinos Markantonakis. Practical Relay Attack on Contactless Transactions by Using NFC Mobile Phones. *IACR Cryptology ePrint Archive*, 2011, 2011.

[25] N Gallagher and Gı Wise. A Theoretical Analysis of the Properties of Median Filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1136–1141, 1981.

[26] TP Ghuntla, HB Mehta, PA Gokhale, and CJ Shah. A Comparative Study of Visual Reaction Time in Basketball Players and Healthy Controls. *National Journal of Integrated Research in Medicine*, 3(1), 2012.

[27] Robert Gray and David Regan. Accuracy of Reproducing Angles: Is a Right Angle Special? *Perception*, 25(5):531–542, 1996.

[28] Gerhard P Hancke. A Practical Relay Attack on ISO 14443 Proximity Cards. *Technical report, University of Cambridge Computer Laboratory*, 59, 2005.

[29] Andreas Holzinger, Christof Stocker, Manuel Bruschi, Andreas Auinger, Hugo Silva, Hugo Gamboa, and Ana Fred. On Applying Approximate Entropy to ECG Signals for Knowledge Discovery on the Example of Big Sensor Data. In *Active Media Technology: 8th International Conference*. Springer, 2012.

[30] Pengfei Hu, Hui Zhuang, Panneer Selvam Santhalingam, Riccardo Spolaor, Parth Pathak, Guoming Zhang, and Xiuzhen Cheng. Accear: Accelerometer Acoustic Eavesdropping With Unconstrained Vocabulary. In *2022 IEEE Symposium on Security and Privacy (S&P)*, 2022.

[31] Aditya Jain, Ramta Bansal, Avnish Kumar, and KD Singh. A Comparative Study of Visual and Auditory Reaction Times on the Basis of Gender and Physical Activity Levels of Medical First Year Students. *International Journal of Applied and Basic Medical Research*, 5(2):124, 2015.

[32] Christina Katsini, Marios Belk, Christos Fidas, Nikolaos Avouris, and George Samaras. Security and Usability in Knowledge-based User Authentication: A Review. In *Proceedings of the 20th Pan-Hellenic Conference on Informatics*, 2016.

[33] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Closed-form Solutions to Minimal Absolute Pose Problems with Known Vertical Direction. In *Asian Conference on Computer Vision*. Springer, 2010.

[34] Gierad Laput, Robert Xiao, and Chris Harrison. Viband: High-fidelity Bio-acoustic Sensing Using Commodity Smartwatch Accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, 2016.

[35] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision*, 81:155–166, 2009.

[36] Miao Li, Rong-Xin Miao, and Yan-Gang Miao. Degrees of freedom of Gravity. *Journal of High Energy Physics*, 2011(7):1–15, 2011.

[37] Xiaopeng Li, Fengyao Yan, Fei Zuo, Qiang Zeng, and Lannan Luo. Touch Well Before Use: Intuitive and Secure Authentication for IoT Devices. In *The 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[38] Xiaopeng Li, Qiang Zeng, Lannan Luo, and Tongbo Luo. T2pair: Secure and Usable Pairing for Heterogeneous IoT Devices. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020.

[39] Yijie Li, Yi-Chao Chen, Xiaoyu Ji, Hao Pan, Lanqing Yang, Guangtao Xue, and Jiadi Yu. Screenid: Enhancing QRCode Security by Fingerprinting Screens. In *IEEE Conference on Computer Communications (INFOCOM)*, 2021.

[40] Chang Liu and Han Yu. Ai-empowered Persuasive Video Generation: A Survey. *ACM Computing Surveys (CSUR)*, 55(13s):1–31, 2023.

[41] Shrirang Mare, Andrés Molina Markham, Cory Cornelius, Ronald Peterson, and David Kotz. Zebra: Zero-effort Bilateral Recurring Authentication. In *IEEE Symposium on Security and Privacy (S&P)*, 2014.

[42] Shrirang Mare, Reza Rawassizadeh, Ronald Peterson, and David Kotz. SAW: Wristband-based Authentication for Desktop Computers. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2(3):1–29, 2018.

[43] Martin Rupp. Understanding QR codes for Payments, 2022. https://utimaco.com/current-topics/blog/qr-codes-for-payments.

[44] Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Distance-bounding Protocols: Verification Without Time and Location. In *2018 IEEE Symposium on Security and Privacy (S&P)*, 2018.

[45] Daniel V McGehee, Elizabeth N Mazzae, and GH Scott Baldwin. Driver reaction time in crash avoidance research: Validation of a driving simulator study on a test track. In *Proceedings of the human factors and ergonomics society annual meeting*, volume 44. Sage Publications Sage CA: Los Angeles, CA, 2000.

[46] Mingliang Tang. Suspects Caught Using Secretly Photographed Payment Codes to Make Purchases, 2018. https://www.toutiao.com/article/6587918959767454212/.

[47] Luis Montesinos, Rossana Castaldo, and Leandro Pecchia. On the Use of Approximate Entropy and Sample Entropy With Centre of Pressure Time-series. *Journal of Neuroengineering and Rehabilitation*, 15(1), 2018.

[48] Zhendong Mu, Jianfeng Hu, Jianliang Min, and Jinghai Yin. Comparison of Different Entropies as Features for Person Authentication Based on EEG Signals. *IET Biometrics*, 6(6):409–417, 2017.

[49] OpenCV. Qrcode detection and encoding, 2023. https://docs.opencv.org/4.x/df/d77/group__objdetect__qrcode.html.

[50] OpenCV. WeChat QR Code Detector for Detecting and Parsing QR Code, 2023. https://docs.opencv.org/4.x/dd/d63/group__wechat__qrcode.html.

[51] Marcus Valtonen Örnhag, Patrik Persson, Mårten Wadenbäck, Kalle Åström, and Anders Heyden. Trust Your IMU: Consequences of Ignoring the IMU Drift. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[52] Hao Pan, Yi-Chao Chen, Lanqing Yang, Guangtao Xue, Chuang-Wen You, and Xiaoyu Ji. mQRCode: Secure QR Code Using Nonlinearity of Spatial Frequency in Light. In *The 25th Annual International Conference on Mobile Computing and Networking*, 2019.

[53] Shijia Pan, Carlos Ruiz, Jun Han, Adeola Bannis, Patrick Tague, Hae Young Noh, and Pei Zhang. Universense: Iot Device Pairing through Teterogeneous Sensing Signals. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications*, 2018.

[54] Paul Skeldon. Pandemic Drives Global Use of QR Codes for Payments as Shoppers Go Contactless, 2021. https://internetretailing.net/pandemic-drives-global-use-of-qr-codes-for-payments-as-shoppers-go-contactless-22550/.

[55] Pramod Borasi. QR Codes Payment Market Expected to Reach $35.07 Billion By 2030, 2022. https://www.alliedmarketresearch.com/press-release/qr-codes-payment-market.html.

[56] Aanjhan Ranganathan and Srdjan Capkun. Are We Really Close? Verifying Proximity in Wireless Systems. *IEEE Security & Privacy (S&P)*, 2017.

[57] Joshua S Richman and J Randall Moorman. Physiological Time-series Analysis Using Approximate Entropy and Sample Entropy. *American Journal of Physiology-heart and Circulatory physiology*, 278(6), 2000.

[58] Carlos Ruiz, Shijia Pan, Adeola Bannis, Ming-Po Chang, Hae Young Noh, and Pei Zhang. IDIoT: Towards Ubiquitous Identification of IoT Devices through Visual and Inertial Orientation Matching During Human Activity. In *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 2020.

[59] Carlos Ruiz, Shijia Pan, Hae Young Noh, Pei Zhang, and Jun Han. Secure Pairing via Video and IMU Verification: Demo Abstract. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, 2019.

[60] Ivan W Selesnick and C Sidney Burrus. Generalized digital butterworth filter design. *IEEE Transactions on Signal Processing*, 46(6):1688–1694, 1998.

[61] Jonathan Sharp, Chuxiong Wu, and Qiang Zeng. Authentication for Drone Delivery through a Novel Way of Using Face Biometrics. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking (MobiCom)*, 2022.

[62] Yasser Shoukry, Paul Martin, Yair Yona, Suhas Diggavi, and Mani Srivastava. Pycra: Physical Challenge-response Authentication for Active Sensors Under Spoofing Attacks. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2015.

[63] B De La Cruz Torres, MD Sánchez López, E Sarabia Cachadiña, and J Naranjo Orellana. Entropy in the Analysis of Gait Complexity: A State of the Art. *British Journal of Applied Science & Technology*, 3(4), 2013.

[64] Transparency Market Research. Barcode readers market, 2020. https://www.transparencymarketresearch.com/barcode-readers-market.html.

[65] TripWire. Relay Attack against Keyless Vehicle Entry Systems Caught on Film, 2017. https://www.tripwire.com/state-of-security/security-awareness/relay-attack-keyless-vehicle-entry-systems-caught-film/.

[66] Venmo. In-Store QR Codes FAQ, 2023. https://help.venmo.com/hc/en-us/articles/360046392254-In-Store-QR-Codes-FAQ.

[67] José Vila and Ricardo J. Rodríguez. Practical Experiences on NFC Relay Attacks with Android. In *Radio Frequency Identification*, 2015.

[68] Ding Wang, Qianchen Gu, Xinyi Huang, and Ping Wang. Understanding Human-chosen Pins: Characteristics, Distribution and Security. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (AsiaCCS)*, 2017.

[69] WeChat. Instructions, 2023. https://wx.gtimg.com/action/shuaka/help_en.shtml.

[70] Wired. Just a Pair of These $11 Radio Gadgets Can Steal a Car, 2017. https://www.wired.com/2017/04/just-pair-11-radio-gadgets-can-steal-car/.

[71] Chuxiong Wu, Xiaopeng Li, Lannan Luo, and Qiang Zeng. G2Auth: Secure Mutual Authentication for Drone Delivery Without Special User-side Hardware. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2022.

[72] Chuxiong Wu, Xiaopeng Li, Fei Zuo, Lannan Luo, Xiaojiang Du, Jia Di, and Qiang Zeng. Use It-No Need to Shake It! Accurate Implicit Authentication for Everyday Objects With Smart Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 6(3):1–25, 2022.

[73] Chuxiong Wu and Qiang Zeng. Turning Noises to Fingerprint-Free "Credentials": Secure and Usable Drone Authentication. *IEEE Transactions on Mobile Computing*, 2024.

[74] Guangtao Xue, Yijie Li, Hao Pan, Lanqing Yang, Yi-Chao Chen, Xiaoyu Ji, and Jiadi Yu. ScreenID: Enhancing QRCode Security by Utilizing Screen Dimming Feature. *IEEE/ACM Transactions on Networking*, 2022.

[75] Zhenyu Yan, Qun Song, Rui Tan, Yang Li, and Adams Wai Kin Kong. Towards Touch-to-access Device Authentication Using Induced Body Electric Potentials. In *The 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2019.

[76] Heng Ye, Qiang Zeng, Jiqiang Liu, Xiaojiang Du, and Wei Wang. Easy Peasy: A new Handy Method for Pairing Multiple Cots IoT Devices. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[77] ZBar. ZBar Bar Code Reader, 2023. https://zbar.sourceforge.net/.

[78] Zhengyou Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[79] Xuhui Zhou, Ziqi Yang, Yunxiao Ren, Weibang Bai, Benny Lo, and Eric M Yeatman. Modified Bilateral Active Estimation Model: A Learning-Based Solution to the Time Delay Problem in Robotic Tele-Control. *IEEE Robotics and Automation Letters*, 8(5), 2023.

[80] Zhe Zhou, Di Tang, Wenhao Wang, Xiaofeng Wang, Zhou Li, and Kehuan 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 (ACSAC)*, 2018.

[81] ZXing. Zebra Crossing from the ZXing Project, 2023. https://zxing.appspot.com/.

A Scanners Used by Various Retailers

PQRAUTH can work with hands-free scanners (and hand-held scanners with built-in IMU, such as smartphones). We thus survey the scanners used by various retail stores in our city, including fast-food restaurants, convenience stores, pharmacies, supermarkets, and warehouse stores. Table 5 shows the results. Among the 25 stores equipped with QR code scanners, all provide hands-free scanners, meaning that PQRAUTH can be used in all the stores.

Table 5: Scanners used by various retailers

Category	Retailer	Hands-free	Hand-held
Fast Food	McDonald's	✓	✗
	Subway	✓	✗
	Starbucks	✓	✓
	Wendy's	✓	✗
Grocery	ALDI	✓	✗
	Wawa	✓	✓
	Circle K	✓	✓
	Speedway	✓	✓
	Love's	✓	✓
	Trader Joe's	✓	✗
	Tiger Mart	✓	✓
	Aplus	✓	✗
Pharmacy	CVS	✓	✓
	Walgreens	✓	✓
	Kroger	✓	✓
Supermart	Walmart	✓	✓
	Target	✓	✓
	Dollar Tree	✓	✓
	Safeway	✓	✓
	Wegmans	✓	✓
	Giant Food	✓	✓
Warehouse	Costco	✓	✓
	Sam's Club	✓	✓
	Home Depot	✓	✓
	Lowe's	✓	✓