Recovering Fingerprints from In-Display Fingerprint Sensors via Electromagnetic Side Channel

Tao Ni
City University of Hong Kong
taoni2-c@my.cityu.edu.hk

Xiaokuan Zhang
George Mason University
xiaokuan@gmu.edu

Qingchuan Zhao
City University of Hong Kong
qizhao@cityu.edu.hk

ABSTRACT

Recently, in-display fingerprint sensors have been widely adopted in newly-released smartphones. However, we find this new technique can leak information about the user's fingerprints during a screen-unlocking process via the electromagnetic (EM) side channel that can be exploited for fingerprint recovery. We propose FPLogger to demonstrate the feasibility of this novel side-channel attack. Specifically, it leverages the emitted EM emanations when the user presses the in-display fingerprint sensor to extract fingerprint information, then maps the captured EM signals to fingerprint images and develops 3D fingerprint pieces to spoof and unlock the smartphones. We have extensively evaluated the effectiveness of FPLogger on five commodity smartphones equipped with both optical and ultrasonic in-display fingerprint sensors, and the results show it achieves promising similarities in recovering fingerprint images. In addition, results from 50 end-to-end spoofing attacks also present FPLogger achieves 24% (top-1) and 54% (top-3) success rates in spoofing five different smartphones.

CCS CONCEPTS

• **Security and privacy** → **Side-channel analysis and countermeasures**; **Mobile and wireless security**.

KEYWORDS

Electromagnetic side channels; In-display fingerprint sensors; Denoising diffusion model

ACM Reference Format:
Tao Ni, Xiaokuan Zhang, and Qingchuan Zhao. 2023. Recovering Fingerprints from In-Display Fingerprint Sensors via Electromagnetic Side Channel. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23), November 26–30, 2023, Copenhagen, Denmark.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3576915.3623153

1 INTRODUCTION

The in-display fingerprint sensor [27] has rapidly gained popularity in recent years, and major smartphone manufacturers have incorporated this technology into their latest Android smartphones, such as the Samsung Galaxy S22, OnePlus 10 Pro, and Huawei P30 Pro. One key factor driving its popularity is its ability to support a more sleek and modern design, as the in-display sensor can be seamlessly placed under the device's display screen, eliminating the need for a separate button as required by traditional fingerprint sensors. According to a recent market survey [49], smartphones equipped with in-display fingerprint sensors have dominated a significant market share, with nearly 32% and 24% in Asia-Pacific and North America, respectively. The global market for this technology is expected to grow to approximately 11 billion dollars by the end of 2025.

Fingerprints are widely regarded as a secure form of biometric authentication for smartphones as they are less susceptible to being captured, leaked, or spoofed compared to other commonly used authentication methods on the Android platform, such as numeric or alphanumeric passcodes, pattern locks, and face recognition. In particular, unlike passcodes and unlocking patterns, which can be inferred through finger movement-recovering attacks such as smudge attacks [21, 34, 72], shoulder-surfing attacks [8, 37, 67], and wireless side-channel attacks for movement tracking [35, 73], fingerprints are immune to such reported attacks as they are not composed of finger movements. On the other hand, fingerprints are less likely to be leaked or captured in high quality compared to faces, which can be easily spoofed and exploited to bypass dedicated authentication measures [32, 47, 51, 63] in daily scenarios, *i.e.*, facial information exposed in social media or work badges. Additionally, in contrast to faces that can be captured through the front camera by malicious apps or compromised mobile OS platforms, fingerprint data is processed only within the fingerprint sensor module and simply reports success or failure to the mobile OS without any redundant data [27], reducing the risk of using software-based methods [1, 4, 14, 65, 69] for capturing or leaking fingerprints.

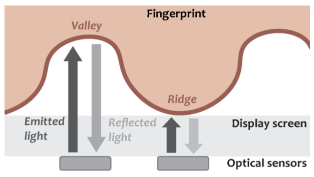
However, we surprisingly found that fingerprints can be leaked from the in-display fingerprint sensor of a smartphone without compromising its hardware and operating system, which has not been reported in prior work, to the best of our knowledge. That is because the in-display fingerprint sensor's authentication process inevitably emits electromagnetic (EM) emanations that contain sufficient information to recover the fingerprint. Specifically, the authentication process begins when a finger is placed on the in-display fingerprint sensor. The sensor then scans the patterns of the pressed finger to generate the fingerprint image and converts it to a data stream for further verification. This series of actions of users and the sensor's electronics will emit EM emanations that can be easily captured by small antennas (*e.g.*, coils). After a thorough investigation, we discovered that the captured EM emanations could reveal the fingerprint image scanned by the in-display fingerprint sensor, and the recovered fingerprint image is of high quality to reconstruct a 3D fingerprint to bypass in-display fingerprint authentication systems on COTS smartphones.

Moreover, our newly discovered EM side-channel attack can overcome many limitations of previous fingerprint-oriented attacks [4, 14, 21, 34, 65, 69, 72]. In particular, in contrast to previous approaches [21, 34, 72], our attack can recover unseen fingerprints from the EM emanations without requiring prior knowledge of the victim's fingerprints. Moreover, unlike previous approaches [4, 19, 20, 65], our attack neither needs to compromise the hardware or software of the victim's smartphone and its in-display fingerprint sensor nor relies on expensive and bulky devices (*e.g.*, oscilloscope) to measure the EM emanations, making it achievable in a common attack scenario, which is on par with prior works on electromagnetic side-channel attacks [3, 38]. In this scenario, an attacker can place a compromised wireless charging power bank in a rental station or modifies a public wireless charging facility and use the wireless charging coil as the antenna to capture EM emanations in close proximity because a smartphone has to attach to those compromised devices for battery charging purposes.

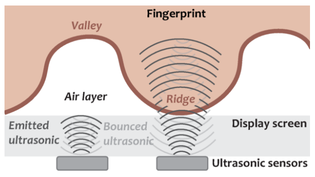
In this paper, we propose a novel framework named FPLogger to demonstrate the feasibility of our reported EM side-channel attack. Specifically, FPLogger first continuously captures nearby EM emanations to monitor if a finger is being placed on the in-display fingerprint sensor to initiate the authentication process. Once a finger-pressing activity on the in-display fingerprint sensor is detected, it then selects segments that contain fingerprint information from the captured EM signals, and adaptively extracts feature maps that reflect the fingerprint patterns. Next, these feature maps are fed into a pre-trained convolutional variational autoencoder (VAE) model to generate the recovered 2D fingerprint images. Furthermore, a denoising diffusion model is deployed to take these recovered 2D fingerprint images and remove noises to improve the recognizability of the output fingerprints. Finally, the recovered 2D fingerprint images are converted into 3D fingerprint pieces via 3D printing, which can be utilized for bypassing smartphone authentication systems if the in-display fingerprint sensors can be spoofed.

We have evaluated the effectiveness of FPLogger on five different smartphones that are equipped with both optical-based (OnePlus 10 Pro, OPPO A96, Xiaomi Redmi K20 Pro, and Huawei P30 Pro) and ultrasonic-based (Samsung Galaxy S10) in-display fingerprint sensors with a compromised wireless charging power bank. Accordingly, the attack distance between the power bank's coil and the in-display fingerprint sensor ranges from 2mm to 10mm due to different smartphones' specifications. Evaluation results show that FPLogger achieves promising performance in recovering fingerprint images with high similarities, ranging from 50.3% to 75.0%, to the original fingerprints. FPLogger also presents promising scalability towards several practical impact factors, including different smartphones, coils for capturing EM emanations, finger types, and attacking distances. Moreover, we conduct 50 trials of end-to-end spoofing attacks, and the results demonstrate that FPLogger can spoof these smartphones with 24% and 54% success rates in one attempt (top-1) and three attempts (top-3), respectively. In addition, we also explore and discuss the potential of attacking in-display fingerprint sensors registered with real fingerprints.

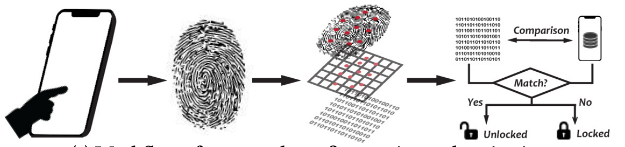
Ethical consideration. We take ethical considerations seriously. All fingerprint pieces are built via a 3D printer using fingerprint images from a public dataset for scientific research [57], and these pieces are only used for fingerprint registration and unlocking the smartphone to collect EM emanations for empirical evaluations. Note that FPLogger and our attacking device have never been released to any other parties.



(a) Optical.



(b) Ultrasonic.



(c) Workflow of a smartphone fingerprint authentication.

Fig. 1: Optical and ultrasonic in-display fingerprint sensors for fingerprint authentication in the smartphone unlocking process.

Responsible disclosure. We have disclosed our findings to the relevant in-display fingerprint sensor and smartphone manufacturers, including OPPO, Vivo, and Samsung. As of this writing, we are working with them closely on practical mitigation solutions. More details (*e.g.*, code, dataset, demo), updates, and appendices will be released at the project website [45]: https://em-fingerprints.github.io.

Contributions. We summarize the contributions as follows:

- **A novel side-channel attack.** We introduce a novel side-channel attack that leverages the EM emanations emitted from the unlocking process of using the in-display fingerprint sensor to unlock a smartphone to recover the user's fingerprint.
- **An end-to-end attack framework.** We propose and implement an end-to-end attack framework, FPLogger, to demonstrate the feasibility of this new side-channel attack. By leveraging the EM emanations emitted during the unlocking process, it recovers fingerprint images and reconstructs 3D fingerprint pieces to spoof the fingerprint authentication system and gain access to the smartphone. To the best of our knowledge, FPLogger is the *first* study to attack in-display fingerprint sensors on smartphones.
- **Empirical evaluation.** FPLogger is evaluated on five commodity smartphones equipped with different in-display fingerprint sensors (optical and ultrasonic). It is also evaluated with a set of impact factors. Our results show that it can effectively recover 3D fingerprints from the EM emanations, and successfully spoof smartphone's fingerprint authentication systems.

2 PRELIMINARIES

2.1 In-display Fingerprint Sensors

The in-display fingerprint sensor is a technology that has been utilized in most newly-released smartphones, enabling users to unlock their devices using their fingerprints without a physical fingerprint sensor on the phone's surface. Instead, an optical or ultrasonic fingerprint sensor (*e.g.*, Synaptics' Clear ID [59]) is embedded beneath the LCD/OLED touchscreen, typically located on the bottom. As is shown in Fig. 1a and Fig. 1b, when a user places their finger on the designated area of the screen, the thin-film transistor (TFT) array of the in-display fingerprint sensor emits either light from back light [2] or ultrasonic waves generated from piezoelectric effect [48] to scan the fingerprint and capture an image of the distinctive *ridges* and *valleys* on the fingerprint pattern. Then, the bounced light or ultrasonic signals are converted to pixel-cell electric currents that represent the signal strength in the gray-scale bitmap, which generates a contour image to describe the fingerprint [48]. Finally, the generated fingerprint image will be compared to the stored fingerprint data to determine whether the user is authorized to unlock the device. This novel unlocking technology has gained popularity because of its seamless and intuitive user experience and its ability to protect against certain types of attacks that other unlocking methods may be vulnerable to, such as shoulder-surfing attacks [8, 67] or eavesdropping attacks [12, 29, 46, 73].

2.2 Smartphone Fingerprint Authentication

Fig. 1c shows a typical workflow of fingerprint authentication for unlocking a smartphone, which comprises four steps. *(i)* The user simply places the finger on the sensor to unlock the smartphone with an in-display fingerprint sensor. *(ii)* The sensor uses emitted light or ultrasonic waves to capture a fingerprint image. This fingerprint image contains the unique patterns of ridges and valleys, highlighted by the shadows and the time it takes for the light or ultrasonic waves to bounce back. *(iii)* Once the sensor has captured the fingerprint patterns, it then applies an algorithm to extract features and convert the image into a biometric data stream, which is a digital representation of the distinguishing characteristics of the fingerprint, *i.e.*, location and depth of ridges and valleys. *(iv)* The extracted biometric data stream is compared to a database of stored fingerprint data to determine if the captured fingerprint matches any authorized users [27, 62]. *(v)* If a match is found, the smartphone is unlocked, and the user gains access to the device. On the contrary, if the fingerprint is unauthorized, the unlocking request is denied, and the device remains locked.

2.3 Physical Principles of EM Emanation in Human-Touchscreen Interactions

Emanations from finger-coupling effects. Unlike traditional fingerprint sensors (*e.g.*, button-based), an in-display fingerprint sensor is usually integrated with the capacitive touchscreen, where the burst of EM emanations induced by the finger-coupling effect [29, 41, 46] can be more apparent and detectable for fingerprint recovery. As illustrated by the equivalent circuit changes in Fig. 2a, when the finger is moving toward the touchscreen, the finger-coupling effect changes the local capacitance of ΔC_f and results in the changing voltage $V_t(t)$ of the touchscreen. The voltage changes $\Delta V_t(t)$ (Equation 1) further induce an instant burst of electromagnetic emanation. Note that $V_{TX}(t)$ and R_{TX} are the driven voltage and resistance of the touch sensor grid (electrode), and $\mathcal{F}^{-1}$ is the *Inverse Fourier Transform* (IFT) to represent frequency-related impedance ΔZ_f in time-domain.

$$\begin{cases} V_{t(NT)}(t) = V_{TX}(t) \cdot \frac{e^{-t/(R_{TX}C_0)}}{R_{TX}C_0} \; (\textit{Not touching}) \\ V_{t(T)}(t) = V_{TX}(t) \cdot \frac{e^{-t/(2R_{TX}C_0\mathcal{F}^{-1}(\Delta Z_f))}}{2R_{TX}C_0\mathcal{F}^{-1}(\Delta Z_f)} \; (\textit{Touching}) \\ \Delta Z_f = 1/(\frac{1}{1/(j2\pi f \Delta C_f)} + \frac{1}{1/(j4\pi f C_0)}) \; (\textit{Impedance}) \\ \Delta V_t(t) = V_{t(NT)} - V_{t(T)} \; (\textit{Induce a burst of EM emanation}) \end{cases} \tag{1}$$

Emanations from the in-display fingerprint sensor. When the finger is placed on the in-display fingerprint sensor, the sensor starts to capture the fingerprint image, extract unique features from the patterns, and then checks whether it is from an authorized user. As illustrated in § 2.1, this smartphone authentication process induces current changes of each cell in the transistor array of the in-display fingerprint sensor , which further results in voltage changes $\Delta V_s(t)$ (Equation 2) in the in-display fingerprint sensor module, inciting additional electromagnetic emanations as follows:

$$\Delta V_s(t) = V_{scan} + V_{feature} + V_{auth}, \tag{2}$$

where V_{scan}, $V_{feature}$, and V_{auth} represent the fingerprint sensor's voltage changes in scanning the fingerprint to capture the image patterns, extracting distinctive features, and matching the captured fingerprint with stored fingerprint data, respectively. Note that the optical-based and ultrasonic-based in-display sensors are only different in the emitted signals when scanning fingerprints, without differences in the quality of the radiated EM emanations.

Measurement of the EM emanations. As presented in § 3.1, the electromagnetic emanations from the in-display fingerprint authentication process can be captured and measured by the coil connected to a micro-controller unit (MCU) that has been integrated into a compromised wireless power bank. Fig. 2b shows the equivalent circuit of harvesting the electromagnetic emanations by leveraging the coil, whose impedance Z_c is shown in Equation 3:

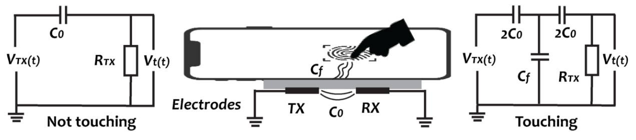
$$Z_c = \frac{1}{R_m} + j2\pi f C_m + \frac{1}{(R_c + j2\pi f L_c)} = e^{-R_m t/C_m} + sin(R_c t/L_c), \tag{3}$$

where C_m, R_m, L_c, and R_c are the impedance parameters (capacitance, resistance, and inductance) related to the materials and turns of the coil. As such, the overall electromagnetic emanations $V_m(t)$ that are scavenged in the coil are the superposition of EM signals emitted from *(i)* the capacitive touchscreen resulting from the finger-coupling effect, *(ii)* the integrated in-display fingerprint sensor when scanning the fingerprint and verification, and *(iii)* workload changes on the CPU and memory, which is measured as:

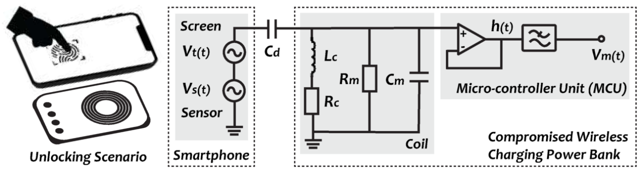
$$\begin{cases} V_c(t) = |\Delta V_t(t)| \, cos(2\pi f_t t) + |\Delta V_s(t)| \, cos(2\pi f_s t) + \sum V_w(f_w, t) \\ V_m(t) = V_c(t) \cdot \frac{e^{-t/(Z_c C_d)}}{Z_c C_d} \cdot h(t) \; (\textit{Captured EM emanation}) \end{cases} \tag{4}$$

where $V_c(t)$ represents the superposition of EM emanations at different frequencies emitted from the mentioned sources, f_t and f_s are the EM frequencies of the touchscreen and the in-display fingerprint sensor, $\sum V_w(f_w, t)$ is the combination of EM signals from the workload changes of CPU and memory, which presents multiple frequencies f_w range from few kHz to GHz [9, 19, 20]. C_d is the distance-related capacitance between the smartphone and the coil ($C_d \propto 1/d$) at vertical direction, and $h(t)$ represents the impulse response function, which involves a voltage amplifier and a low-pass filter in the MCU [29]. Therefore, the strength of the electromagnetic emanations in a smartphone fingerprint authentication can be measured by monitoring the voltage changes of the coil.

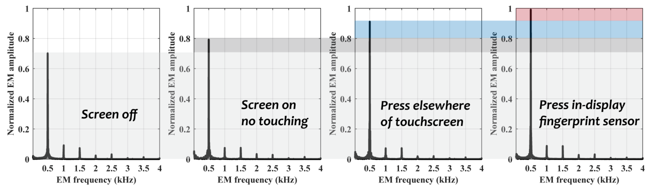
Fig. 2c presents the analysis of captured EM emanations in the frequency domain (*sampling frequency* = 20kHz) when the smartphone is screen off (*silver area*), screen on while not pressing the touchscreen (*grey area*), pressing elsewhere of the touchscreen (*blue area*), or presses the in-display fingerprint sensor (*red area*). It shows that the smartphone radiates EM signals at different frequencies



(a) Finger-coupling effects in pressing in-display fingerprint sensor.


Unlocking Scenario
Screen
Sensor
Smartphone
Coil
Micro-controller Unit (MCU)
Compromised Wireless Charging Power Bank

(b) EM emanation measurements using the coil in a power bank.



(c) Analysis of captured EM emanations in the frequency domain.

Fig. 2: Illustration of finger-coupling effects and EM emanations in unlocking the smartphone using the in-display fingerprint sensor (a) and (b), and frequency analysis of captured EM emanations (c).

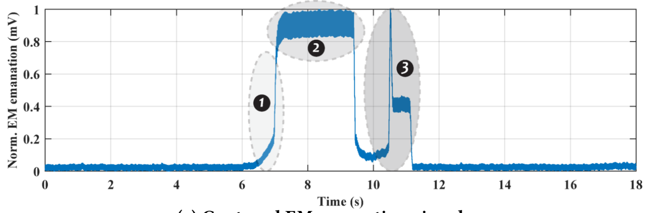
(*e.g.*, CPU and memory [9], touchscreen [56]) at an idle running status. When the user wakes up the smartphone and presses the touchscreen, the finger-coupling effects induce extra EM emanations that lead to the increase of EM signal's strength at approximately $f = 500Hz$ (*silver area* ⟶ *grey area* ⟶ *blue area*). Furthermore, since the in-display fingerprint sensor is integrated with the touchscreen, EM signals emitted from the sensors when scanning the fingerprints for verification could also be superimposed into the captured EM emanations, resulting in a higher EM amplitude (*blue area* ⟶ *red area*) at the similar frequency. Therefore, the captured EM emanations in the coil are the superposition of EM signals from the aforementioned EM sources, which cause the leakage of users' fingerprint information in the in-display fingerprint authentication process.

3 MOTIVATION AND THREAT MODEL

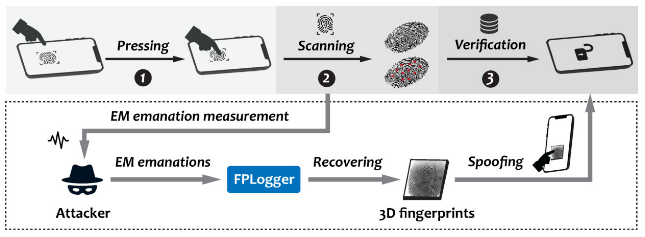
3.1 A Motivating Example

This section presents a real scenario that motivates this study. To unlock and access the smartphone, a user places the finger on the touchscreen, where the in-display fingerprint sensor is located. The sensor scans the fingerprint to determine if it comes from an authorized user and then unlocks the smartphone if a match is detected. However, during this authentication process, an attacker can intercept and collect EM emanations from the smartphone, extract the segment that contains fingerprint information, and recover the 2D fingerprint image and 3D fingerprint. In Fig. 3a, we illustrate the changes of the captured EM emanations in this unlocking process with the in-display fingerprint sensor, and Fig. 3b shows the three stages with different signal patterns as follows:

❶ When the finger approaches the in-display fingerprint sensor, the finger-coupling effects cause a shift in the local capacitance of the touchscreen [29, 41, 46], which leads to a burst of EM emanations. The amplitude of the captured EM emanations reaches the highest when the finger fully contacts the touchscreen.



(a) Captured EM emanation signal.



(b) Three steps of in-display fingerprint recognition.

Fig. 3: A motivating example: a user places the finger on the in-display fingerprint sensor to unlock the smartphone. The attacker can leverage the EM emanations to recover the user's fingerprint to build 3D pieces to spoof the in-display fingerprint sensor.

❷ Once the finger is placed on the sensor, the sensor utilizes light or ultrasonic waves (§ 2.1) to capture the fingerprint image, extract its unique features, and convert it to the data stream [27, 62]. This series of sensor activities also emits EM emanations that contain information reflecting the fingerprint patterns, and these emanations are superimposed with the EM signals from ❶. As such, the attacker can exploit such EM emanations to build 3D fingerprint pieces to spoof the in-display fingerprint sensor of the smartphone.
❸ The data stream of the input fingerprint is compared with stored fingerprint data to determine if it matches a registered fingerprint. If a match is detected, the smartphone will be unlocked. The corresponding changes of the EM emanation signal in this stage primarily result from the processing units and alterations in the display content on the touchscreen when entering the home screen.

In particular, an attacker can first exploit the EM emanation in a screen-unlocking process to trigger the attack and then extract features from the collected emanations that contain fingerprint patterns. Next, the extracted features are used to recover the fingerprint image and build the 3D fingerprint piece to spoof the in-display fingerprint authentication system to unlock the smartphone.

3.2 Threat Model

Attack scenario. We consider a common scenario where an attacker initially places a compromised wireless charging power bank in a rental station or modifies a public wireless charging facility (Fig. 5). Then, the attacker can leverage the wireless charger's coil to capture EM emanations during the screen-unlocking process from the smartphone equipped with an in-display fingerprint sensor (Fig. 6). Subsequently, the attacker can exploit the EM emanations to recover fingerprint images and construct fraudulent 3D fingerprint pieces, which are further used to spoof the in-display fingerprint sensor and unlock the smartphone, enabling unauthorized access to sensitive user privacy. We consider a close attacking distance (*e.g.*, 2mm–10mm) between the smartphone and the coil, which is on par with prior works on electromagnetic side channels [3, 20, 38].

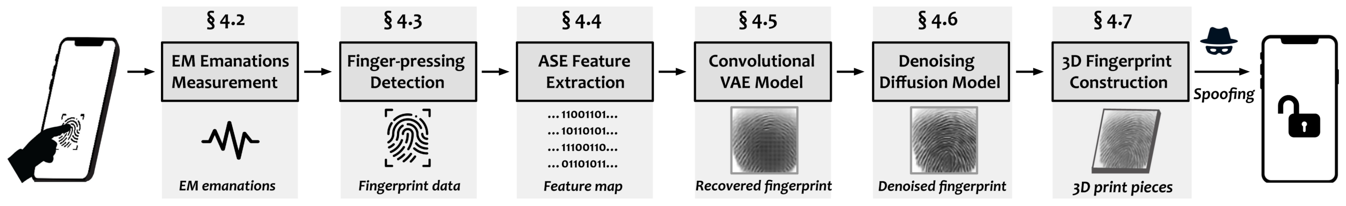


Fig. 4: Overview of FPLogger.

Fig. 5: Illustration of public wireless charging stations and power bank rental services. An attacker can maliciously compromise these facilities to use their coils to capture EM emanations.

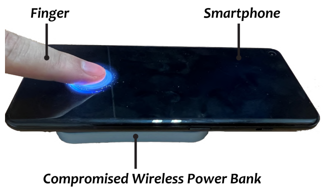


Fig. 6: Attack scenario.

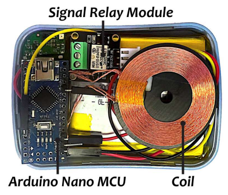


Fig. 7: Compromised power bank.

Assumptions. We assume the victim is using a smartphone with an in-display fingerprint sensor. The attacker ***can*** compromise a wireless charging station (or a power bank) in public rental services, leveraging the coil to capture EM emanations. Note that the attacker ***cannot*** compromise the victim's smartphone, including cracking firmware to obtain stored fingerprint data or access the readings of the in-display fingerprint sensor. Furthermore, the attacker does not have a line-of-sight (LoS) view of the victim's fingers and is unable to capture fingerprint images from the victim's hands through high-definition (HD) cameras.

4 DESIGN OF FPLOGGER

4.1 Overview of FPLogger

Fig. 4 presents the overview of FPLogger, which leverages EM emanations from a smartphone to recover 3D fingerprints for spoofing in-display fingerprint sensors. *(i)* FPLogger first collects EM emanations emitted by the victim's smartphone during a screen-unlocking action using the coil of a wireless charging power bank attached to the smartphone. *(ii)* Then, it applies filters and moving-variance windows to detect the finger-pressing event and segment out EM signals with fingerprint information. *(iii)* Next, the obtained EM signal is processed to extract unique features that reflect the input fingerprint patterns. *(iv)* FPLogger uses the pre-trained convolutional VAE model to map extracted features to 2D fingerprint images and *(v)* a denoising diffusion model is deployed to remove noise and generate fingerprints with recognizable resolutions. *(vi)* FPLogger can construct 3D fingerprint pieces using the recovered fingerprint images to spoof the in-display fingerprint sensor of the victim's smartphone and gain access to more sensitive information.

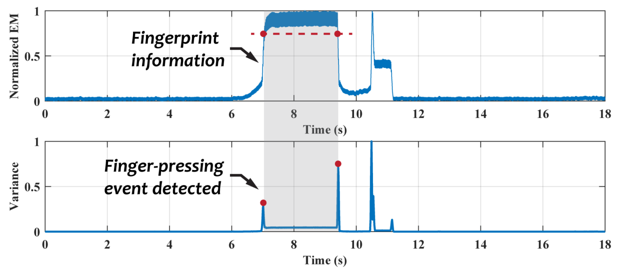
4.2 EM Emanation Measurements

To measure the EM emanations from the smartphone during a screen-unlocking process, we compromise a commodity wireless charging power bank by *(i)* leveraging its coil as an antenna to capture the EM emanations, *(ii)* staffing a tiny micro-controller unit (MCU) to record signals, and *(iii)* integrating a signal relay module to control the start/stop of wireless charging. In practice, we tear up an EGO MAGPOWER 2 wireless charging power bank [39] and connect its coil to an analog input pin of an Arduino Nano MCU [42] with an HFD4/3-S signal relay module [26] as is shown in Fig. 7. As the size of the MCU and signal relay module are significantly smaller than that of the power bank, an attacker can easily insert these hardware components into the power bank and restore it to its original appearance. When the user presses the in-display fingerprint sensor, the signal relay module could suddenly stop and restart the wireless charging process to create a 3–5 seconds interruption for capturing EM emanations from the coil. The power bank samples captured signals with an analog-to-digit converter at a sampling frequency of $20kHz$, which is more than enough to capture signal variances in EM emanations [29] emitted from the touchscreen and the in-display fingerprint sensor (*e.g.*, $500Hz$). Note that FPLogger only requires one interruption to record the EM signals, which is believed not to raise the victim's suspicion because interruptions sometimes happen in a normal wireless charging process due to power fluctuations or firmware glitches [64].

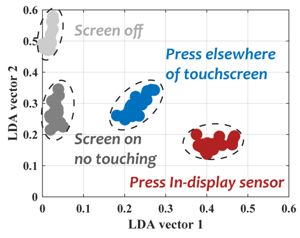
4.3 Finger-pressing Detection

Finger-pressing detection and extraction. When the victim attaches the compromised wireless power bank to the smartphone, FPLogger continuously monitors the captured EM emanations from the coil and applies a *Savitzky–Golay* (S-G) filter to remove noises in the collected sequential EM signals without distorting the signal shapes [7]. The filtered EM signal is then normalized to a range from 0 to 1, and exploits the normalized signal to detect a smartphone-unlocking action to trigger the attack and extract fingerprint information. In practice, we can leverage the signal's variance of the EM emanations to detect a finger-pressing on the touchscreen and further apply a moving variance window to extract the corresponding segment that contains fingerprint information. Fig. 8a shows an example of applying a moving variance window ($threshold = 0.1$) to the captured EM signal ($sampling\ frequency = 20kHz$), where the finger-pressing event can be detected from the appearance of the first peak, and we can extract the EM segment that contains fingerprint information between the first peak and the second peak in the moving-variance signal.

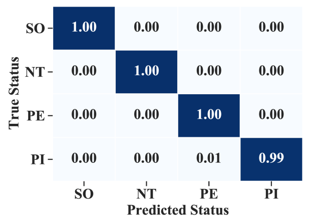
False finger-pressing rejection. After extracting the segment from the captured EM emanations, FPLogger needs to decide whether the finger-pressing is on the in-display fingerprint sensor or other areas of the capacitive touchscreen. In § 2.3, we have demonstrated that the amplitudes of the captured EM emanations differ in four finger-pressing status (Fig. 2c). Therefore, we apply the



(a) Finger-pressing detection and extraction.



(b) LDA of EM amplitude.



True Status	SO	NT	PE	PI
SO	1.00	0.00	0.00	0.00
NT	0.00	1.00	0.00	0.00
PE	0.00	0.00	1.00	0.00
PI	0.00	0.00	0.01	0.99

Predicted Status

(c) Recognition results.

Fig. 8: Finger-pressing event detection and extraction (a), and false finger-pressing rejection based on the LDA of EM amplitude (b) and the recognition results (c) (SO: Screen off, NT: Screen on, no touching, PE: Press elsewhere on the touchscreen, PI: Press the in-display fingerprint sensor).

Fast Fourier Transform (FFT) to the EM emanations and calculate the *Linear Discriminant Analysis* (LDA) [66] of the amplitude of the EM signals in the frequency domain as shown in Fig. 8b, which presents distinguishable features of different finger-pressing scenarios. As such, we can use the LDA features to recognize the finger-pressing status and reject false finger-pressing events resulting from pressing other areas of the touchscreen. In practice, we collect 1,000 EM samples from the four finger-pressing status and use 80% of them to train a k-NN [43] classifier to recognize the four finger-pressing events and the rest 20% samples to evaluate its effectiveness. Fig. 8c shows the results of recognizing finger-pressing status to reject false finger-pressing, where the k-NN classifier achieves 99.75% accuracy. Hence, FPLogger can effectively detect finger-pressing on the in-display fingerprint sensor and reject false positive cases when pressing elsewhere on the touchscreen.

4.4 ASE Feature Extraction

After detecting a finger-pressing event on the in-display fingerprint sensor, FPLogger automatically extracts and selects the segment $F(t)$ that contains the fingerprint information from the captured EM emanations. It then extracts *Adaptively-Selected Envelope* (ASE) features from this informative segment to distinguish between a bit of 0 and 1 of the EM emanations on mobile devices such as smartphones [10]. The envelopes describe features such as amplitude, rising/falling time, and peak/troughs of the captured EM emanations that distinguish different fingerprints, and the ASE approach suits different sampling frequencies and bit rates. To achieve this, we leverage the signal envelope method from the MATLAB (version 2022a) signal processing toolbox to generate the upper envelope curve with peak analysis with a frame length of $0.001f$, where f represents the sampling frequency (*e.g.*, $20kHz$) of the coil for capturing EM emanations. Fig. 9 shows the extracted ASE features from the segmented EM emanation signal in a 0.02 seconds interval.

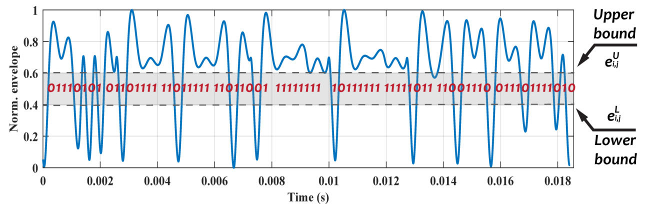


Fig. 9: Adaptive-selected envelope (ASE) features ($e^F_{i,j}(t)$) extracted from EM emanation signal that contains fingerprint information.

Adaptive feature selection. In practice, FPLogger first selects a few envelopes with significant differences depending on the transmitted bit value, which represents the 0/1 data stream generated in fingerprint scanning that reflects the differences in fingerprint patterns (*e.g.*, ridge and valley) [30]. This selection process is carried out each time fingerprint recognition is performed, and a set of envelope features is adaptively chosen based on the segmented EM emanations. As such, FPLogger defines a maximum distinguishable range of $e^L_{i,j}$ (lower bound) to $e^U_{i,j}$ (upper bound) for each envelope $e_{i,j}$ to differentiate a bit value of 0 and 1, where i and j represent the starting and ending indices of the select frame at the timestep t. As such, $e^U_{i,j}$ is initially set to the minimum between these enveloped peaks and further optimized as the minimum value among all positive peaks greater than a given threshold $1.5 \cdot e^L_{i,j}$. For instance, we set $e^L_{i,j}$ and $e^U_{i,j}$ as 0.4 and 0.6 in the frame shown in Fig. 9, respectively. After that, we calculate $e^L_{i,j}$ as the maximum among positive and negative peaks less than $e^U_{i,j}$. This heuristic method enables us to obtain an approximate maximum range that can distinguish the positive and negative peaks caused during the transmission of a bit 1 and 0. FPLogger then selects the top-n_E (*e.g.*, $n_E = 4, 8, \ldots,$ *etc.*) envelopes that have the largest difference less than $e^L_{i,j}$ or exceeds $e^U_{i,j}$.

Fingerprint data extraction. Once the adaptive features are chosen, FPLogger extracts the envelope data $e^F_{i,j}(t)$ that contains the input fingerprint information, *i.e.*, patterns of ridges and valleys from the envelope $e_{i,j}(t)$. Next, FPLogger applies a down-sampling approach [33] to normalize the extracted fingerprint sequence, and concatenates n_F segments from a moving binning window with length $l^F = 0.002$ seconds (*e.g.*, $n_F = e^F_{i,j}(t)/l^F$) to prevent information loss. Note that the down-sampling method, frame length, and binning window can be adjusted based on the required resolution of the target fingerprint images. Therefore, FPLogger exploits such process for each possible fingerprint input and generates sequences reflecting fingerprint features, and the sequences are reshaped to $r_f \times r_f$ resolution 2D arrays to train the models for recovering recognizable 2D fingerprints, where r_f is the resolution of target fingerprint images (*e.g.*, $r_f = 64$). More details about the notations used in ASE feature extraction are listed in the project website [45].

4.5 Convolutional VAE Model

To achieve fingerprint image reconstruction, we train a *convolutional variational autoencoder* (VAE) model to map extracted $r_f \times r_f$ ASE features to $r_f \times r_f$ resolution fingerprint images (*e.g.*, $r_f = 64$). The convolutional layers in the VAE architecture enable the model to effectively capture spatial (*e.g.*, position, peaks/troughs) and temporal (*e.g.*, amplitude, rising/falling time) information in the input data, making it well-suited for fingerprint image reconstructions.

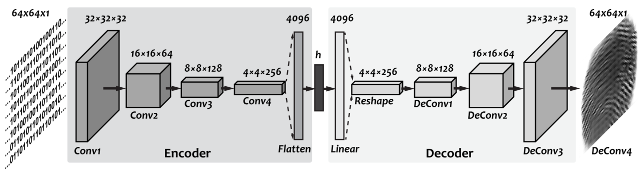


Fig. 10: Architecture of the convolutional VAE model.

The convolutional VAE model can learn latent space representations, handle high-dimensional input data, such as 2D arrays, while avoiding overfitting issues that arise with traditional neural networks. It also leverages dropout layers and regularization methods to help improve the generalization performance. In addition, the KL divergence loss term in the convolutional VAE loss function makes the learned latent space smooth and continuous, which is significant to the image reconstruction tasks [22].

Fig. 10 depicts the architecture of the convolutional VAE model we implemented in FPLogger. The model comprises an encoder and a decoder, both of which are composed of convolutional layers. In particular, the encoder consists of four 2D convolutional layers with 32, 64, 128, and 256 output channels produced by the convolution. Each convolutional layer uses ReLU as the activation function, and the kernel size, stride, and padding are set to 4, 4, and 1, respectively. Then, a flatten layer and a linear layer are used to reshape the feature maps extracted by the convolutional layers to an encoded vector and map it to the decoder. The decoder consists of four transpose 2D convolutional layers with 128, 64, 32, and 1 output channels with the same activation function, kernel size, stride, and padding as the encoder. Finally, an *sigmoid* function maps the decoded features to a value between 0 and 1 to build the grayscale arrays and the forward process in the convolutional VAE model extracts spatial information and encodes the input to latent space representations, and map it for fingerprint image reconstruction. In practice, we select the Adam optimizer [31] and employ the binary cross-entropy (BCE) with the KL divergence regularization [68] to measure the loss of the convolutional VAE model in the training process.

After obtaining the trained convolutional VAE model, we could use it to map extracted feature maps from the captured EM emanations to 2D fingerprint images. Such fingerprint images are often blurry, missing important fingerprint patterns, and therefore cannot be used to launch spoofing attacks directly. Next, we introduce a denoising diffusion model to remove such noises.

4.6 Denoising Diffusion Model

While the convolutional VAE model is known for its ability to learn latent representations of the images, the recovered fingerprint images often contain noise that can obscure important features, such as the fingerprint patterns of ridges and valleys. These noises may impact the similarities between the recovered fingerprints and the victim's fingerprints, which is crucial for FPLogger to launch the spoofing attack. Therefore, we design and develop a denoising diffusion model based on the concept of *Denoising Diffusion Probabilistic Model* (DDPM) [25] to remove noise and recover the blurred patterns to enhance the clarity of the fingerprint images. It leverages a deep learning architecture to model the noise distribution and iteratively refine the image through a reverse process, resulting in a more accurate representation of the original fingerprint patterns.

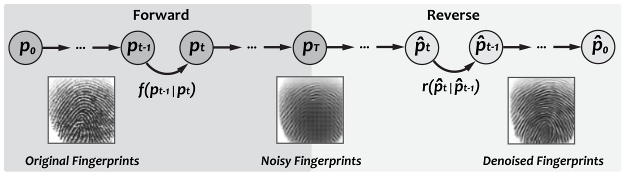


Fig. 11: Illustration of the denoising diffusion model. Note that the noisy fingerprint images are the output of the convolutional VAE model (§ 4.5) and are being denoised in the reverse process.

Fig. 11 shows the proposed denoising diffusion model, which comprises two primary processes: *forward* process and *reverse* process. Note that we use the noisy fingerprints generated from the convolutional VAE model as the input of the denoising diffusion model.

Forward process. The forward process describes the diffusion of noise into original fingerprint images, which effectively learns the noise distribution in the recovered fingerprint images generated by the convolutional VAE model. Given a data point p_0 sampled from the data distribution of a recovered fingerprint image $f(x)$ ($p_0 \sim f(x)$), the forward process models the generation of noisy fingerprints at timestep t from the original fingerprints as follows:

$$f(p_t \mid p_{t-1}) = \mathcal{F}(p_t \; ; \; noise(t, t-1)), \tag{5}$$

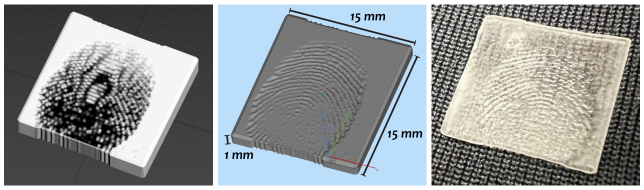
where $noise(t, t-1)$ represents the noises from p_{t-1} to p_t, producing a new latent variable p_t with the distribution $f(p_t|p_{t-1})$. The forward process continues this process to understand the noise distribution in $f(x)$ comparative to the original image $f_0(x)$, which is then utilized in the reverse process for denoising the images.

Reverse process. The reverse process is implemented for denoising the noisy images by exploiting deep learning architecture (*e.g.*, U-Net [55]) to learn the mapping functions between the noisy images and their corresponding clean fingerprint images by minimizing the difference between the original images and the denoised outputs. The reverse process at timestep t can be represented as follows:

$$r(\hat{p}_{t-1} \mid \hat{p}_t) = \mathcal{R}(\hat{p}_{t-1} \; ; \; denoise(t-1, t)), \tag{6}$$

where $denoise(t-1, t)$ is the denoising process from $\hat{p}_{t-1}$ to $\hat{p}_t$ using the deep learning models, *i.e.*, U-Net. Here if we apply the reverse formula for all data points in a noisy fingerprint image from the convolutional VAE model (§ 4.5), we can remove the noisy parts and obtain a more recognizable fingerprint.

Implementation details. In practice, we use the recovered fingerprint images from the training set of the convolutional VAE model as the noisy images and the original fingerprint images to train this denoising diffusion model [54]. Furthermore, we implement a U-Net neural network and use it in the reverse process to denoise fingerprint images. Specifically, this U-Net contains four encoders and four decoders, each consisting of a convolutional block comprising two convolutional layers, with ReLU as the activation function and max-pooling as the pooling method. Note that both forward and reverse processes are relevant since they are trained jointly. In the training process, we use the *Mean Squared Error* (MSE) loss function with Adam optimizer [31] and evaluate the model on the recovered fingerprint images generated by the convolutional VAE model from the testing set. Therefore, the combination of the convolutional VAE model and the denoising diffusion model enables FPLogger to recover more recognizable fingerprint images that retain the essential features and pattern details required for reconstructing 3D fingerprint pieces to spoof smartphones' in-display fingerprint sensors.



(a) 3ds Max. (b) Materialise Magics. (c) Prototype.

Fig. 12: 3D fingerprint pieces construction from 2D images. (a) 3ds Max converts the 2D grayscale fingerprint images to 3D models, (b) Materialise Magics to reshape the 3D models to $15mm \times 15mm \times 1mm$, and (c) A prototype of 3D print fingerprint pieces from 3D printer.

4.7 3D Fingerprint Construction

After obtaining the recovered 2D fingerprint images, FPLogger can reconstruct 3D fingerprint pieces that simulate real fingerprints to spoof in-display fingerprint sensors in commodity smartphones. The 3D fingerprint reconstruction consists of three steps as follows:

Step 1: 2D fingerprint images to 3D fingerprint models. The first step to developing 3D fingerprint pieces is to convert 2D fingerprint images to 3D models with textures of ridges and valleys. As is shown in Fig. 12a, we use Autodesk 3ds Max (version 2021) to draw the fingerprint texture on a square piece based on its grayscale, and then export 3D model files of the fingerprint pieces.

Step 2: Reshape 3D fingerprint models. After building 3D fingerprint models for all images in the dataset, we need to guarantee the size and thickness of the reshaped 3D fingerprint models can be easily attached to the surface of fingers to simulate a real fingerprint. Therefore, as is demonstrated in Fig. 12b, we utilize Materialise Magics (version 21.0) to reshape these 3D fingerprint models to $15mm \times 15mm$ rectangles with the thickness of $1mm$.

Step 3: Printing the 3D fingerprint pieces. At last, we print these well-conditioned 3D fingerprint models with a CNC 3D printer [61] using materials of semi-transparent photopolymer such as light-activated resin. Fig. 12c shows the prototype of a fingerprint piece produced by the 3D printer, where we can observe the texture of ridges and valleys patterns of the fingerprint. Such 3D fingerprint pieces can be exploited to simulate the victim's fingerprint to deceive the in-display fingerprint sensors [17]. Note that each printed 3D fingerprint piece costs approximately 0.05 dollars (5 cents).

5 EVALUATION

5.1 Experimental Setup and Data Collection

5.1.1 Public fingerprint dataset. Human fingerprints are very sensitive personal data, and illegally collecting them may result in ethical violations. Therefore, we utilize fingerprint images from a public dataset, *Sokoto Coventry Fingerprint Dataset* [57] (SOCOFing[1]), to build 3D fingerprint pieces for fingerprint registration and unlocking the smartphone in evaluating FPLogger. In particular, SOCOFing contains 6,000 fingerprint images (each image is gray-scaled with 96×103 pixels) collected from 600 Africans (450 male participants and 150 female participants, fingerprints of both left and right hands) and is released for academic purposes only. Note that each image is grayscale with 96×103 pixels, and we resize the image to 64×64 pixels by removing extra margins and rescaling. Moreover, while this process will result in information loss, we show that it does not affect the authenticity of the fingerprint, and the built 3D fingerprints from these images can be used for fingerprint registration and screen unlocking (§ 5.1.2 and § 6.1).

[1]Fingerprint images are available at: https://www.kaggle.com/ruizgara/socofing. In this study, we spent approximately 40 days in 3D fingerprint construction and 3D printing, 3 months for data collection (*e.g.*, fingerprint registration in the smartphones, EM signal collection), and 20 days for model training, fine-tuning, and evaluation.

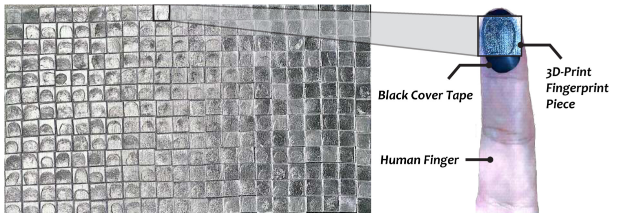
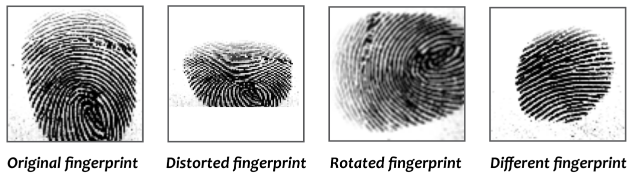


Fig. 13: Illustration of the experimental setup leveraging 3D printed fingerprint pieces to register the user and unlock the smartphone.

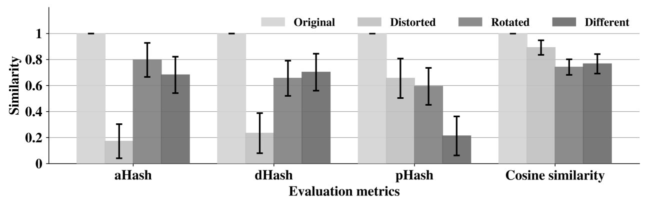
5.1.2 Experimental setup. Fig. 13 shows the experimental setup of leveraging the 3D fingerprint pieces from SOCOFing. As illustrated in § 4.7, we build these thin 3D fingerprint pieces and stick them to a human finger covered with opaque black tape to simulate human finger-pressing actions. Note that black tape is used for covering the real fingerprint of the data collector to avoid interferences, and our institution's IRB board has approved this research. To enable better recovering performance, we press the in-display fingerprint sensor (optical) of a OnePlus 10 Pro at 12 different orientations (*i.e.*, $0°(360°), 30°, 60°, \ldots, 330°$) to collect a total of 72,000 data samples. We separately repeat the same procedure to collect data samples from the other four smartphones equipped with in-display fingerprint sensors (§ 5.3.1) and other four coils (§ 5.3.2), and conduct experiments with the measurement coil in the compromised power bank underneath the smartphone. All data processing, feature extraction, and model training are conducted on a desktop running Windows 10 with 32GB memory, Intel i7-9700K CPU, and an NVIDIA GeForce RTX 2080Ti GPU. In practice, we train the convolutional VAE model and the denoising diffusion model using samples collected from 3D fingerprint pieces of 480 participants (80%, #1–#480), and evaluate the performance and conduct end-to-end spoofing attacks (§ 6) with the samples collected from the other 120 participants (20%, #481–#600). In practice, we initialize the learning rate as 0.001 with a batch size of 32, and train the convolutional VAE model and the denoising diffusion model for 5,000 epochs and 1,000 epochs, respectively. Note that there is *no overlap* of data samples between the training set and the testing set.

5.2 Evaluation Metrics

To evaluate the effectiveness of FPLogger, we investigate four metrics that are used for comparing similarities: average hash [23] (*aHash*), different hash [13] (*dHash*), perceptual hash [15] (*pHash*), and cosine similarity [60]. Fig. 14a shows these four metrics in comparing similarities between the original fingerprint and a distorted fingerprint, a rotated fingerprint (90°), and a different fingerprint. We conducted an experiment on 1,000 different fingerprint images and found that pHash is robust to fingerprint distortion and rotation, and can distribute two different fingerprints. In comparison, the other three metrics still present high similarities between two different fingerprints, which may increase the false positive rate in fingerprint recognition (Fig. 14b). Therefore, based on the empirical results, we select the pHash similarity as the evaluation metric



(a) Comparison with distorted, rotated, and different fingerprints.



(b) Similarity results of four evaluation metrics.

Fig. 14: Investigation to explore the best evaluation metric for comparing image similarity between fingerprints.

for comparing the similarities between output fingerprints from FPLogger and original fingerprints (details in Fig. 20 and § 6.3).

We propose *FingerHash* to calculate pHash similarity between the original and recovered fingerprints in Algorithm 1. It first takes the recovered and original fingerprint images and converts them to gray-scale images (line 1–2). Next, it uses a loop structure to apply the *Discrete Cosine Transform* [52] (DCT) to calculate their pHash values by converting the fingerprint data streams from the spatial domain to the frequency domain (line 3–8). At last, the algorithm calculates the hamming distance between two pHash sequences to obtain the similarity of the two fingerprints (line 9–10).

Algorithm 1: *FingerHash*: pHash similarity for fingerprints.

```
Input: $f_r$: recovered fingerprint image. $f_o$: original fingerprint
       image. $l_s$: rescaled length (width) of $f_r$ and $f_0$.
Output: $Sim(f_r, f_o)$: Image similarity (0–1) between $f_r$ and $f_o$.
1 $f_r \leftarrow rescale(f_r, l_s)$, $f_o \leftarrow rescale(f_o, l_s)$ ▷ rescale $f_r$ and $f_o$ to
    $l_s \times l_s$ images with an initial $l_s = 32$.
2 $f_r \leftarrow convert\_grayscale(f_r)$, $f_o \leftarrow convert\_grayscale(f_o)$ ▷
    convert fingerprint images to gray-scale $f_r$ and $f_o$.
3 for each point $f_r(i,j) \in f_r$ and $f_o(i,j) \in f_o$ do
4     $F_r = c(u)c(v)\sum_i \sum_j f_r(i,j) cos\left[\frac{(i+1/2)\pi}{N_r}u\right] cos\left[\frac{(j+1/2)\pi}{N_r}v\right]$,
5     $F_o = c(u)c(v)\sum_i \sum_j f_o(i,j) cos\left[\frac{(i+1/2)\pi}{N_o}u\right] cos\left[\frac{(j+1/2)\pi}{N_o}v\right]$.
      ▷ Discrete Cosine Transform to obtain $F_r(u,v)$ and $F_o(u,v)$.
6 $F_r^{doi} = F_r(u,v)[0:8, 0:8]$, $F_o^{doi} = F_o(u,v)[0:8, 0:8]$. ▷ Obtain
    the upper left $8 \times 8$ pixels with high frequencies.
7 $H_r = (F_r^{doi} > average(F_r^{doi}))$, $H_o = (F_o^{doi} > average(F_o^{doi}))$. ▷
    Obtain pHash values from $F_r^{doi}$ and $F_o^{doi}$.
8 $Sim(f_r, f_o) = hamming\_distance(H_r, H_o)$. ▷ Compute similarity
    with Hamming distance between pHash values $H_r$ and $H_o$.
9 Output fingerprint similarity $Sim(f_r, f_o)$ (0–1) between $f_r$ and $f_o$.
```

5.3 Effectiveness of Fingerprint Image Recovery

We evaluate FPLogger's performance in recovering the 2D fingerprint images by comparing the pHash similarity of two output images with the original fingerprints: *(i)* the *recovered fingerprint images* from the convolutional VAE model; *(ii)* the *denoised fingerprint images* from the denoising diffusion model. That is because fingerprint images recorded by the smartphone are stored and encrypted in the in-display fingerprint sensor, whose firmware disallows access by users or the OS [27], and we cannot obtain those images in our current threat model. Thus, we conduct experiments considering different impact factors by following the same procedure: *(i)* different smartphones with both optical and ultrasonic in-display fingerprint sensors, *(ii)* different coils for capturing the EM emanations, *(iii)* different fingers, and *(iv)* different attacking distances.

5.3.1 Evaluation on different smartphones. Since different smartphones usually integrate in-display fingerprint sensors from different vendors (*e.g.*, OnePlus series: Clear ID [59], Samsung Galaxy series: Goodix [50]), we separately collect data to train models by conducting further experiments on four other different commodity smartphones: OPPO A96 (PFUM10), Xiaomi Redmi K20 Pro (M1903F11I), Huawei P30 Pro (VOG-AL00), and Samsung Galaxy S10 (SM-G973F). Note that we select these smartphones based on our investigation of 20 newly-released smartphones [45] , and the results show 85% of them choose optical-based and only 15% choose ultrasonic-based in-display fingerprint sensors. Therefore, we select one smartphone (Samsung Galaxy S10) with an ultrasonic in-display fingerprint sensor, while the other four smartphones (including the OnePlus 10 Pro) are equipped with optical in-display fingerprint sensors. Fig. 15 shows the evaluation results of FPLogger on the five smartphones, where we find FPLogger achieves pHash similarities of 0.565, 0.396, 0.537, 0.546, and 0.472 in recovering fingerprint images captured by the in-display fingerprint sensors of OnePlus 10 Pro, OPPO A96, Redmi K20 Pro, Huawei P30 Pro, and Samsung S10, respectively. After applying the denoising diffusion model, the corresponding pHash similarities are enhanced to 0.750, 0.503, 0.604, 0.707, and 0.543, which reflects an approximate boost of 12.5%–32.7% in fingerprint recovery. In addition, we notice that FPLogger performs well in smartphones like OnePlus 10 Pro and Huawei P30 Pro while presenting lower pHash similarities in the fingerprint samples collected from OPPO A96. This is because the OPPO A96 has a mechanism that verifies whether the finger covers the entire area of the in-display fingerprint sensor before initiating fingerprint recognition [6]. As a result, it is necessary to increase the contact area between the finger with a 3D fingerprint piece and the touchscreen. However, enlarging the contact area can lead to finger-coupling effects that generate additional interference in the collected EM emanations [29]. Nevertheless, FPLogger *have demonstrated the feasibility of recovering fingerprint images from the EM emanations emitted by in-display fingerprint sensors from different commodity smartphones.*

5.3.2 Evaluation on different coils. As we have illustrated in § 2.3, the captured EM emanations are related to the properties (*e.g.*, materials, shapes, and turns) of the receiving coil. Therefore, we investigate the coils' impact by individually collecting data to train models and conducting experiments on four other coils with different characteristics, denoting as $coil_2$ (rectangle, $30mm \times 23mm$, 15 turns), $coil_3$ (rectangle, $42mm \times 29mm$, 27 turns), $coil_4$ (circle, $42mm$, 10 turns), and $coil_5$ (circle, $38mm$, 20 turns). Note that $coil_1$ (circle, $40mm$, 22 turns) is the coil of the compromised wireless charging power bank in § 4.2. Fig. 16 shows the evaluation results of FPLogger using different coils ($coil_1$–$coil_5$) to launch attacks, where FPLogger achieves pHash similarities of 0.565, 0.521, 0.625,

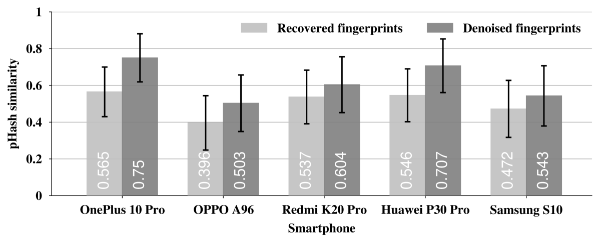


Fig. 15: Evaluation results of different COTS smartphones with the optical or ultrasonic in-display fingerprint sensors.

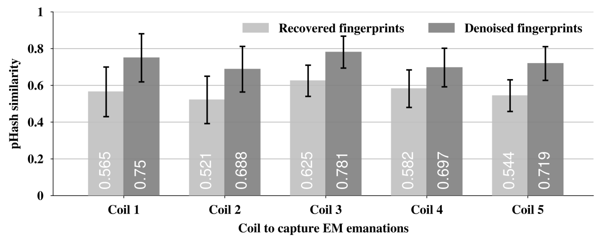


Fig. 16: Evaluation results of different coils with various widths, heights, and turns for EM emanation measurement.

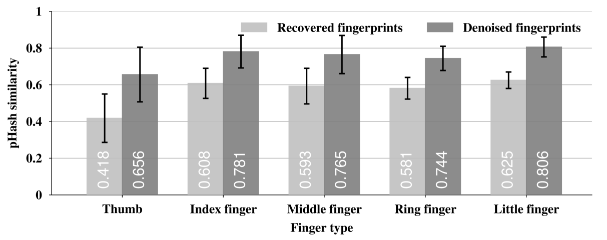


Fig. 17: Evaluation results of different types of the finger on the hand from *thumb* to *little finger*.

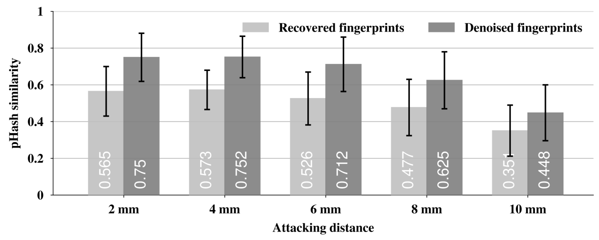


Fig. 18: Evaluation results of different attacking distances between the smartphone and the wireless charging coil.

0.582, and 0.544 in recovering fingerprint images from EM emanations captured by different coils ($coil_1$–$coil_5$). The denoising diffusion model can improve corresponding pHash similarities to 0.750, 0.688, 0.781, 0.697, and 0.719, which shows an increment of 19.8%–32.7% in fingerprint recovery. We observe that different coils show limited impact on the performance of FPLogger because coils' characteristics only affect the EM signal's amplitude resulting from the finger-coupling effect when a user presses the in-display fingerprint sensor. Therefore, FPLogger *can extract fingerprint information from the recorded EM emanations, and use the recovered fingerprints to spoof in-display fingerprint sensors.*

5.3.3 *Evaluation on different finger types.*

The five fingers on a hand have different characteristics, *i.e.*, sizes and shapes. Thus, we also evaluate FPLogger's performance when recovering fingerprints of different fingers: *thumb*, *index finger*, *middle finger*, *ring finger*, and *little finger*. In practice, we divide the testing set into five subsets based on the type of fingers and evaluate the average pHash similarities in each group. Fig. 17 shows the evaluation results of FPLogger in recovering fingerprint images from different types of fingers, where we find that FPLogger achieves pHash similarities of 0.418, 0.608, 0.593, 0.581, and 0.625 in recovering fingerprint images of different types of fingers. Likewise, our proposed denoising diffusion model increases the pHash similarities to 0.656, 0.781, 0.765, 0.744, and 0.806, respectively. Interestingly, we find that FPLogger performs better in recovering fingerprint images of the four fingers, *index finger*, *middle finger*, *ring finger* and *little finger* than fingerprints collected from the *thumb*. The main reason is that in the SOCOFing dataset we used, most fingerprint images collected from *thumbs* usually only contain partial representations of the real fingerprints while ignoring most of the detailed patterns of the fingerprints (*e.g.*, arches, left/right loops, and whorls [53]). Consequently, these images contain limited fingerprint information and are prone to misprediction in the fingerprint recovery process. In summary, FPLogger *achieves acceptable performance in recovering fingerprints from thumbs, and achieves excellent performance in recovering fingerprints from other fingers.*

5.3.4 *Evaluation on different attacking distances.*

We have presented in Fig. 2b that the vertical distance between the smartphone and the coil could impact the capacitance C_d, which further affects the EM emanation measurements based on Equation 4. Hence, we set four other attacking distances (*e.g.*, $4mm$, $6mm$, $8mm$, and $10mm$) between the smartphone and the receiving coils to explore such an impact. Note that the default distance between the smartphone and the compromised wireless charging power bank is $2mm$. Fig. 18 shows the evaluation results of FPLogger at different attacking distances, which indicate FPLogger achieves pHash similarities of 0.565, 0.573, 0.526, 0.477, and 0.351 in recovering fingerprint images from EM emanations captured at different attacking distances, while the denoising diffusion model increases these pHash similarities to 0.750, 0.752, 0.712, 0.625, and 0.448, respectively. The performance in recovering fingerprints decreases as the attacking distance increases. This is because a longer attacking distance between the smartphone and the receiving coil leads to EM emanations with lower amplitudes and more noise. As such, we design the attack scenario of FPLogger when a compromised power bank wirelessly charges the smartphone in proximity settings that follow the research line of similar EM side-channel studies [5, 9, 19, 20, 29, 38]. Note that FPLogger can also work in other directions in a similar attacking distance by re-collecting data and fine-tuning models, *which can recover fingerprints with good performance.*

Summary. In a nutshell, we have demonstrated the feasibility of FPLogger in recovering fingerprints from the emitted EM emanations under different scenarios, *i.e.*, different smartphones, coils, finger types, and attacking distances. The above evaluation results show the recovered fingerprint images, after applying the denoising diffusion models, present acceptable pHash similarities compared with the original fingerprints. Furthermore, we also find considerable deviations in the evaluation results because of the distinctions between different fingerprints. Below, we conduct 50 trials of end-to-end attacks (§ 6) to explore the effectiveness of leveraging the 3D-printed fingerprints from FPLogger's output fingerprints to spoof different smartphones' in-display fingerprint sensors.

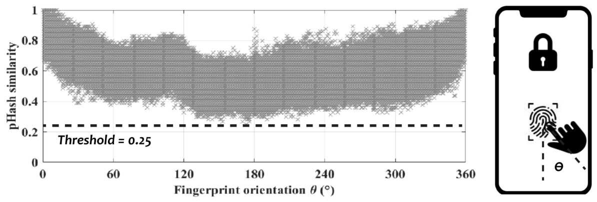


Fig. 19: pHash similarities with different fingerprint orientations (0°–360°) that are recognizable to the in-display fingerprint sensor.

6 END-TO-END ATTACKS

6.1 End-to-end Attack Evaluation Steps

We conduct 50 trials of end-to-end attacks using the aforementioned five smartphones (each smartphone for 10 trials), including four with optical in-display fingerprint sensors and one with ultrasonic in-display fingerprint sensors. In particular, we randomly select fingerprints from both left and right hands of #481–#600 participants (unseen testing set) and then use the 3D fingerprint pieces generated by FPLogger to spoof the in-display fingerprint sensors. In each attack trial, we press the in-display fingerprint sensor three times and then calculate the top-1 (*T-1*) and top-3 (*T-3*) success rates. Specifically, in calculating the top-3 success rate, we consider one successful unlocking as a successful attacking trial within the three attempts. The reason why we choose the top-3 success rate as the metric is because a typical smartphone authentication system usually allows the user to press the in-display fingerprint sensor several times (*e.g.*, 3–5 attempts) to unlock the smartphone if one attempt is recognized. Among the five smartphones that we have investigated in § 5.3.1, all of their in-display fingerprint sensors allow up to four unlocking attempts. Once the user failed to pass the verification for four times, the smartphone automatically turns off the in-display fingerprint sensor and then switches to the passcode-unlocking screen until the smartphone is successfully unlocked. The time cost of an end-to-end attack is approximately 3 to 5 minutes, including capturing EM emanations (5–6 s), recovering fingerprints (36–48 s), and 3D printing pieces to unlock the smartphone (2.5–3 minutes).

6.2 Threshold of Recognizable pHash

In a real-life scenario, the user can press the in-display fingerprint sensor at any orientation, resulting in different captured fingerprint images. However, the in-display fingerprint sensor can still recognize these fingerprint images and unlock the smartphone. Before launching end-to-end attacks via FPLogger, we first explore the minimum pHash threshold of an authorized fingerprint that is enough to be recognized by the in-display fingerprint sensor. To investigate such a threshold, we compute the pHash values of the 6,000 fingerprint images rotated at an angle from 0° to 360° in Fig. 19, where we know the minimum pHash value for the same fingerprint is approximately 0.25. Once the similarity of the recovered fingerprints exceeds such a threshold, it can theoretically spoof the in-display fingerprint sensor and unlock the victim's smartphone. Furthermore, when the pHash similarity exceeds 0.6, the in-display fingerprint sensor can recognize almost all fingerprints with different orientations to unlock the smartphone. As such, our recovered fingerprint images with pHash similarity ranging from 0.565 to 0.750 as illustrated in § 5.3 could be exploited to launch end-to-end attacks to spoof the smartphone's in-display fingerprint sensors.

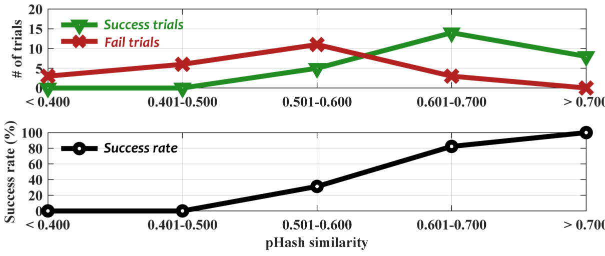


Fig. 20: Top-3 success and fail attack trials, and spoofing success rate (%) with different ranges of pHash similarity.

6.3 End-to-end Attack Results

Table 1 presents the detailed results of the 50 end-to-end attacks, including the pHash similarities of the denoised fingerprints and the attacking success rate in one attempt (*T-1*) and three attempts (*T-3*). We know that the attacking success rate in spoofing different smartphones is approximately 24% (12/50 spoofed) with only one attempt, and 54% (27/50 spoofed) with three attempts of all end-to-end attack trials. In addition, Fig. 20 shows the top-3 success rate with different ranges of pHash similarity, where we also find that the denoised fingerprints with higher pHash similarity to the original fingerprints normally present higher success rates in the spoofing attacks. For instance, trial #4, #21, and #25 present 100% and 100% success rates in one attempt and three attempts, with pHash similarities of 0.781, 0.750, and 0.767, respectively.

Cases with high pHash similarity but low success rate. In particular, we also find several trials present abnormal results, *i.e.*, the denoised fingerprints have high pHash similarities to the original fingerprints but cannot spoof the in-display fingerprint sensor. For instance, trials such as #7, #24, and #26 are with high pHash similarities 0.603, 0.682, and 0.593, but they all failed in spoofing the in-display fingerprint sensor to unlock the smartphone.

Fig. 21 shows original fingerprints and output fingerprints of FPLogger from eight trials of the end-to-end attacks. Four cases (#4, #6, #25, and #36) are fingerprints with high pHash similarities and high spoofing success rates, and other four cases (#7, #24, #26, and #38) with high pHash similarities but low spoofing success rates. We discover that the main reason that leads to these abnormal attack trials is mispredictions from the proposed denoising diffusion model. For instance, in trial #38, the fingerprint contains the pattern "arch", whereas the denoising diffusion model mistakenly predicts this pattern as "left loop". Likewise, the denoising diffusion model also predicts an incorrect "line shape" in the denoised fingerprint while there is a "whorl" in the original fingerprints in the attack trial #7. This is because the denoising diffusion model is trained on a dataset with limited fingerprint samples. As a result, the output fingerprints may contain mispredicted patterns that cannot be recognized by the in-display fingerprint sensor. Note that it is possible to enhance the success rates of the spoofing attacks by leveraging a broader high-resolution fingerprint dataset to train the denoising diffusion model, and we leave it to our future work (§ 7.3).

7 DISCUSSION

7.1 Countermeasures

We formulated the following two directions of countermeasures and discussed them with smartphone manufacturers, and provided practical solutions for mitigating the threat of our attacks.

Table 1: End-to-end attack results, including 50 trials of five commodity smartphones with optical or ultrasonic in-display fingerprint sensors (each 10 trials attacks). We randomly select 3D fingerprint pieces from both left and right hands in the testing set. T-1: one attempt, T-3: three attempts, "○": Spoofing denied, "●": Spoofing succeeded, smartphone unlocked. (trials with * are cases in Fig. 21).

Trial	Smartphone	Sensor Type	Fingerprint No.	Finger Type	Denoised Fingerprint pHash Similarity	T-1 Results	T-3 Results			Spoofing Success ?
							1st	2nd	3rd	
1	OnePlus 10 Pro	Optical	No. 593, Male, Left hand	Index finger	0.688	●	●	●	●	✓
2		Optical	No. 586, Male, Left hand	Little finger	0.593	○	○	○	●	✓
3		Optical	No. 544, Male, Left hand	Middle finger	0.656	○	○	●	●	✓
4*		Optical	No. 506, Male, Left hand	Ring finger	0.781	●	●	●	●	✓
5		Optical	No. 545, Male, Left hand	Thumb	0.422	○	○	○	○	✗
6*		Optical	No. 482, Male, right hand	Index finger	0.625	○	○	●	●	✓
7*		Optical	No. 535, Male, right hand	Little finger	0.603	○	○	○	○	✗
8		Optical	No. 540, Female, right hand	Middle finger	0.531	○	○	○	○	✗
9		Optical	No. 529, Male, right hand	Ring finger	0.656	●	●	●	●	✓
10		Optical	No. 513, Male, right hand	Thumb	0.375	○	○	○	○	✗
11	OPPO A96	Optical	No. 481, Female, Left hand	Index finger	0.313	○	○	○	○	✗
12		Optical	No. 507, Male, Left hand	Little finger	0.359	○	○	○	○	✗
13		Optical	No. 517, Male, Left hand	Middle finger	0.625	○	○	●	●	✓
14		Optical	No. 533, Male, Left hand	Ring finger	0.562	○	○	○	●	✓
15		Optical	No. 561, Male, Left hand	Thumb	0.406	○	○	○	○	✗
16		Optical	No. 566, Male, right hand	Index finger	0.593	○	○	○	●	✓
17		Optical	No. 579, Male, right hand	Little finger	0.656	●	●	●	●	✓
18		Optical	No. 534, Female, right hand	Middle finger	0.453	○	○	○	○	✗
19		Optical	No. 547, Male, right hand	Ring finger	0.563	○	○	○	●	✓
20		Optical	No. 528, Female, right hand	Thumb	0.500	○	○	○	○	✗
21	Redmi K20 Pro	Optical	No. 583, Male, Left hand	Index finger	0.750	●	●	●	●	✓
22		Optical	No. 579, Male, Left hand	Little finger	0.656	○	○	○	●	✓
23		Optical	No. 575, Male, Left hand	Middle finger	0.453	○	○	○	○	✗
24*		Optical	No. 485, Male, Left hand	Ring finger	0.575	○	○	○	○	✗
25*		Optical	No. 511, Male, Left hand	Thumb	0.767	●	●	●	●	✓
26*		Optical	No. 529, Male, right hand	Index finger	0.593	○	○	○	○	✗
27		Optical	No. 562, Female, right hand	Little finger	0.719	●	●	●	●	✓
28		Optical	No. 564, Male, right hand	Middle finger	0.562	○	○	○	○	✗
29		Optical	No. 555, Male, right hand	Ring finger	0.688	○	○	●	●	✓
30		Optical	No. 538, Male, right hand	Thumb	0.593	○	○	○	○	✗
31	Huawei P30 Pro	Optical	No. 532, Male, Left hand	Index finger	0.719	●	●	●	●	✓
32		Optical	No. 540, Female, Left hand	Little finger	0.531	○	○	○	○	✗
33		Optical	No. 547, Male, Left hand	Middle finger	0.781	●	●	●	●	✓
34		Optical	No. 563, Male, Left hand	Ring finger	0.500	○	○	○	○	✗
35		Optical	No. 570, Male, Left hand	Thumb	0.625	○	○	●	●	✓
36*		Optical	No. 492, Female, right hand	Index finger	0.813	●	●	●	●	✓
37		Optical	No. 582, Male, right hand	Little finger	0.688	○	○	●	●	✓
38*		Optical	No. 481, Male, right hand	Middle finger	0.682	○	○	○	○	✗
39		Optical	No. 594, Male, right hand	Ring finger	0.656	○	○	○	●	✓
40		Optical	No. 523, Male, right hand	Thumb	0.565	○	○	○	○	✗
41	Samsung Galaxy S10	Ultrasonic	No. 524, Male, Left hand	Index finger	0.563	○	○	○	○	✗
42		Ultrasonic	No. 531, Male, Left hand	Little finger	0.688	●	●	●	●	✓
43		Ultrasonic	No. 547, Male, Left hand	Middle finger	0.656	○	○	●	●	✓
44		Ultrasonic	No. 557, Male, Left hand	Ring finger	0.593	○	○	○	●	✓
45		Ultrasonic	No. 590, Male, Left hand	Thumb	0.531	○	○	○	○	✗
46		Ultrasonic	No. 596, Male, right hand	Index finger	0.563	○	○	○	○	✗
47		Ultrasonic	No. 588, Male, right hand	Little finger	0.625	○	○	●	●	✓
48		Ultrasonic	No. 495, Male, right hand	Middle finger	0.719	●	●	●	●	✓
49		Ultrasonic	No. 505, Male, right hand	Ring finger	0.521	○	○	○	○	✗
50		Ultrasonic	No. 494, Male, right hand	Thumb	0.656	○	○	○	○	✗

Mitigating the EM emanations. FPLogger exploits EM emanations from a screen-unlocking process to recover victims' fingerprints for spoofing the in-display fingerprint sensors. As such, one potential countermeasure is to eliminate the EM emanations in the screen-unlocking process when pressing the in-display fingerprint sensor. One possible solution is to require endorsement from vendors to redesign the sensor hardware [24] or add proper shielding to mitigate the leakage of the EM emanations [46]. Furthermore, we can apply electromagnetic interference (EMI) to perturb and even cancel out the emitted EM emanations resulting from finger-pressing activities on the touchscreen. Recently, the EMI technique has been widely studied to induce a fake human-touching event (*a.k.a.*, "ghost touch") by intentionally modulating the frequency and strength of the generated EM signals [28, 41, 56]. Hence, we can leverage the modulated EM signals to add extra perturbations in the EM emanations captured by the coil to prevent privacy leakage and shield the security of users' fingerprints. Since the attacker has no prior knowledge of the extra EM interference signal patterns, extracting the informative data incurred by finger-pressing from the perturbed EM emanations is difficult.

Designing a second-factor user authentication. Our research has shown that FPLogger can spoof in-display fingerprint sensors on mainstream smartphones since their authentication systems only authenticate the captured fingerprint images. To mitigate such vulnerabilities, one can leverage other signals from other built-in sensors (*e.g.*, microphone, accelerometer) to design and develop a second-factor user authentication system, *i.e.*, acoustic signals [8, 73], IMU signals [58, 71]. During the screen-unlocking process, smartphones can not only capture fingerprint images via the in-display sensor but also verify the patterns of the user's hand and finger movements through other sensors. As such, these second-factor authentication methods are able to protect against spoofing attacks using recovered 3D fingerprint pieces.

7.2 Attacks on Real Fingerprints

Our end-to-end attack evaluation in § 6 has demonstrated FPLogger's effectiveness in recovering 3D-printed low-resolution fingerprints (*i.e.*, 64*dpi*) in the public dataset due to ethical concerns of evaluating with real persons' high-resolution fingerprints. It is also interesting to explore and understand whether a recovered low-resolution fingerprint from FPLogger can still spoof an in-display

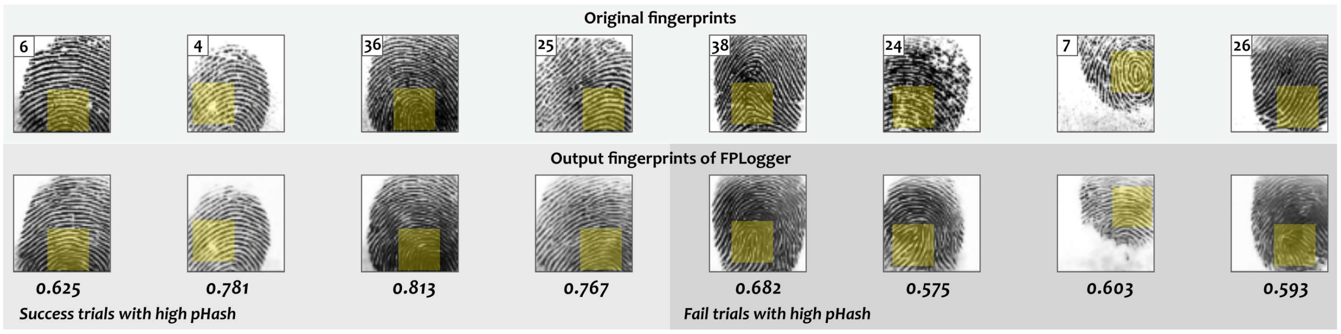

Original fingerprints
6
4
36
25
38
24
7
26
Output fingerprints of FPLogger
0.625
0.781
0.813
0.767
0.682
0.575
0.603
0.593
Success trials with high pHash
Fail trials with high pHash

Fig. 21: Illustration of denoised and original fingerprints in eight end-to-end attack trials with high pHash similarity (marked *). Four of them successfully spoof the in-display fingerprint sensor, while the others are not. Yellow boxes: Similar/mispredicted patterns.

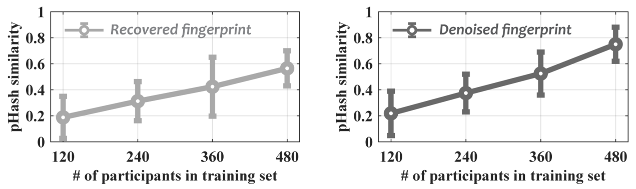


Fig. 22: pHash similarity of recovered and denoised fingerprints with different numbers of participants in the training set (§ 7.3).

fingerprint sensor if the corresponding registered fingerprint is as high-resolution as in daily normal scenarios, where a COTS smartphone's in-display fingerprint sensor often scans the user's fingerprint with resolution ranging from 300*dpi* to 550*dpi* [30]. As such, unlike 64*dpi* 3D fingerprints used for both registration and testing in § 6, the smartphone's registered fingerprints in this new evaluation have resolutions ranging from 300*dpi* to 550*dpi*, whereas the recovered ones for testing are 64*dpi*. In the end, we conducted these new end-to-end experiments within the scope of the approved IRB by registering the five smartphones used in § 6 with five real fingerprints from our own, collecting EM emanations to recover fingerprint images, and reconstructing fingerprint pieces to spoof the in-display fingerprint sensors.

As shown in Table 2, *FPLogger can successfully recover and reproduce all fingerprints, and, surprisingly, it even unlocks two smartphone models in three attack trials though it fails in the rest trials.* These results are reasonable and meet our expectations primarily due to the pattern differences in detail between a high-resolution image ($\geq$ 300*dpi*) and a low-resolution image (64*dpi*), which results in a failed similarity comparison. The evaluation results and hardware specification also support our hypothesis because *(i)* FPLogger failed in all trials against Samsung Galaxy S10 as its sensor scans fingerprints with the highest resolution, 550*dpi* [30], while *(ii)* randomly succeeded in two of other four smartphones, *i.e.*, OnePlus 10 Pro and Huawei P30 Pro, which scan relatively lower fingerprint's resolution (300–363*dpi* [30]), offering a marginal chance for FPLogger recovered 3D-printed low-resolution fingerprints to spoof them. It should be noted that the ability of the current version of FPLogger to generate high-resolution fingerprints is constrained by the training dataset that only contains low-resolution fingerprint images. We push the discussion of the related limitation to § 7.3.

7.3 Limitations and Future Works

We have implemented FPLogger to demonstrate the feasibility of leveraging the EM emanations to recover fingerprints to spoof commodity in-display fingerprint sensors. While the results are promising, there still exist several limitations in the current work. First, as a proof of concept, FPLogger is developed and evaluated in a public dataset that includes limited fingerprint images with low resolutions. Fig. 22 shows the pHash similarities of the recovered and denoised fingerprints increase when the size of the training set increases, which suggests that if there is a larger training dataset with more high-resolution fingerprint images, FPLogger can achieve better performance, especially when coming to recovering real fingerprints from COTS smartphones. Second, we consider a close attacking distance in FPLogger to capture fine-grained EM emanations to demonstrate its feasibility. While this attacking distance is on par with the research line of electromagnetic side channels [3, 5, 9, 20, 38], it is possible to enhance FPLogger to work on a setting with a longer distance, whereas EM emanations from the in-display fingerprint sensor could be interfered with when being blocked and attenuated by other electronic devices, resulting in the loss of fingerprint information. We leave them to our future studies and will seek further IRB approval.

Table 2: End-to-end attack results, including 25 trials of using recovered 3D fingerprint pieces to unlock smartphones registered with five real fingerprints (§ 7.2). FP: fingerprint, "○": Fail, "●": Success.

Smartphone	Register FP *dpi*	Test FP *dpi*	T-3 Results of Five Real Fingerprints				
			FP 1	FP 2	FP 3	FP 4	FP 5
OnePlus 10 Pro	300–363	64	○ ○ ●	○ ○ ○	○ ○ ●	○ ○ ○	○ ○ ○
OPPO A96	300–363	64	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○
Redmi K20 Pro	300–363	64	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○
Huawei P30 Pro	300–363	64	○ ○ ○	○ ○ ○	○ ○ ●	○ ○ ○	○ ○ ○
Samsung S10	550	64	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○	○ ○ ○

8 RELATED WORKS

Fingerprint spoofing attacks. Fingerprint recognition has become a widely adopted authentication method for mobile devices (*e.g.*, smartphones, tablets), laptops, and IoT facilities (*e.g.*, biometric door locks [16, 18]). Recent studies have revealed the vulnerabilities in fingerprint-based authentication systems. For instance, various spoofing attacks [21, 34, 72] leverage smudges left on the touchscreen to recover the fingerprint and then spoof the fingerprint recognition sensors of the users' devices with artificial fingerprint pieces made from materials like glue [11] and gelatin [53]. Moreover, other works [4, 14, 65, 69] have demonstrated that an attacker can compromise the untrusted OS (*e.g.*, Android 6.0/7.0) by installing malicious apps to bypass the smartphone's fingerprint authentication system. However, smudge-based spoofing attacks [11, 21, 34, 72] require prior knowledge of the users' fingerprints left on the touchscreen or leveraging a high-definition camera to take fingerprint images. In addition, attacks [4, 14, 65, 69] need to pre-install malware and can only be effective in old-version Android OS. In this paper, we present FPLogger, the first attack framework to spoof the most advanced in-display fingerprint sensors adopted in newly-released smartphones via EM emanations.

Electromagnetic side channels. There are many efforts towards exploiting electromagnetic side channels of the smartphone to uncover user privacy. For instance, an attacker can leverage the EM emanations to infer cryptographic keys [19, 20], secret message content [38, 70], unlocking passcodes [29], neural network architectures [3, 40], and hardware or software usage in other embedded systems [5, 9]. Furthermore, Ni *et al.* [44, 46] utilizes the perturbations of the EM emanations to uncover user privacy, *i.e.*, app usage and input keystrokes, when a wireless charger is charging the smartphone. MagEar [36] demonstrated the feasibility of using the coils in a headphone to eavesdrop on audio conversations in others' earphones or smartphone earpieces. Likewise, MagSnoop [10] exploits the sound effect induced by the electromagnetic field to infer the credit card tokens in the payment process of Samsung Pay. Our work, FPLogger, leverages the electromagnetic emanations in the unlocking process to recover the 3D fingerprints and successfully spoofs state-of-the-art in-display fingerprint sensors.

9 CONCLUSION

In this paper, we present a novel side-channel attack for spoofing the in-display fingerprint sensors of commodity smartphones by leveraging the emitted EM emanations in a screen-unlocking process. To validate its feasibility, we design and implement FPLogger, an attack framework that utilizes the coil of a wireless charging power bank to capture EM emanations from the smartphone when the user presses the in-display fingerprint sensor, and then exploits the extracted feature maps to recover fingerprint images and reconstruct 3D fingerprint pieces to launch the spoofing attacks. To the best of our knowledge, FPLogger is the *first* attack framework for recovering fingerprints and spoofing the most advanced in-display fingerprint sensors. Our extensive evaluation suggests that FPLogger can recover fingerprints at a promising level that can be used for building the 3D fingerprint pieces, and the results from end-to-end attacks also present FPLogger achieves 24% (top-1), and 54% (top-3) success rates in spoofing the different smartphones' in-display fingerprint sensors. We hope our findings can raise public awareness of the vulnerability of in-display fingerprint sensors and spur research on detecting forthcoming side-channel attacks and developing new defense approaches.

ACKNOWLEDGMENTS

We sincerely thank all anonymous reviewers for their constructive feedback. This work was supported by CityU APRC grant 9610563, Hong Kong RGC (CityU 21219223, C1029-22G). Any opinions, findings, and conclusions in this paper are those of the authors and not necessarily of the supported organizations.

REFERENCES

[1] Asaf Ashkenazi and Dmitry Akselrod. 2007. Platform independent overall security architecture in multi-processor system-on-chip integrated circuits for use in mobile phones and handheld devices. *Computers & Electrical Engineering* (2007).
[2] Sanghoon Bae, Yan Ling, Weiping Lin, and Hong Zhu. 2018. 76-2: Invited paper: optical fingerprint sensor based on a-Si: H TFT technology. In *SID Symposium Digest of Technical Papers*, Vol. 49. Wiley Online Library, 1017–1020.
[3] Lejla Batina, Shivam Bhasin, Dirmanto Jap, and Stjepan Picek. 2019. CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel. In *Proceedings of the 28th USENIX Security Symposium.*
[4] Antonio Bianchi, Yanick Fratantonio, Aravind Machiry, Christopher Kruegel, Giovanni Vigna, Simon Pak Ho Chung, and Wenke Lee. 2018. Broken Fingers: On the Usage of the Fingerprint API in Android.. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium.*
[5] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. 2018. Screaming channels: When electromagnetic side channels meet radio transceivers. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS).* 163–177.
[6] OPPO Care. 2022. How To enable In-Display Fingerprint Sensor on your OPPO phone - OPPO Care. (2022). https://youtu.be/EjqFfHtYZxY.
[7] Jin Chen, Per Jönsson, Masayuki Tamura, Zhihui Gu, Bunkei Matsushita, and Lars Eklundh. 2004. A simple method for reconstructing a high-quality NDVI time-series data set based on the Savitzky–Golay filter. *Remote sensing of Environment* 91, 3-4 (2004), 332–344.
[8] Yongliang Chen, Tao Ni, Weitao Xu, and Tao Gu. 2022. SwipePass: Acoustic-based Second-factor User Authentication for Smartphones. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)* (2022).
[9] Yushi Cheng, Xiaoyu Ji, Wenyuan Xu, Hao Pan, Zhuangdi Zhu, Chuang-Wen You, Yi-Chao Chen, and Lili Qiu. 2019. Magattack: Guessing application launching and operation via smartphone. In *Proceedings of ACM Asia Conference on Computer and Communications Security (AsiaCCS).* 283–294.
[10] Myeongwon Choi, Sangeun Oh, Insu Kim, and Hyosu Kim. 2022. MagSnoop: listening to sounds induced by magnetic field fluctuations to infer mobile payment tokens. In *Proceedings of MobiSys.* 409–421.
[11] Chaos Computer Club. 2004. How to fake fingerprints? (2004). https://biometrics.mainguet.org/alive/site_archive/CCC_01_How_to_fake_fingerprints.htm.
[12] Patrick Cronin, Xing Gao, Chengmo Yang, and Haining Wang. 2021. Charger-Surfing: Exploiting a Power Line Side-Channel for Smartphone Information Leakage.. In *Proceedings of the 30th USENIX Security Symposium.* 681–698.
[13] Joan Daemen and Craig Clapp. 1998. Fast hashing and stream encryption with PANAMA. In *Fast Software Encryption: 5th International Workshop, FSE'98 Paris, France, March 23–25, 1998 Proceedings.* Springer, 60–74.
[14] Thom Does and Mike Maarse. 2016. Subverting Android 6.0 fingerprint authentication. *University of Amsterdam, Amsterdam* (2016), 22.
[15] Ling Du, Anthony TS Ho, and Runmin Cong. 2020. Perceptual hashing for image authentication: A survey. *Signal Processing: Image Communication* (2020).
[16] Foxgard. 2023. Foxgard Smart Fingerprint Door Lock. (2023). https://www.amazon.com/Foxgard-Fingerprint-Keyless-Biometric-Storage/dp/B08Y8QTN7T/.
[17] Sergiu Gatlan. 2019. Samsung Galaxy S10 Fingerprint Scanner Tricked with 3D Print. (2019). https://www.bleepingcomputer.com/news/security/samsung-galaxy-s10-fingerprint-scanner-tricked-with-3d-print/.
[18] GeekTale. 2023. GeekTale Smart Door knob. (2023). https://www.amazon.com/GeekTale-Fingerprint-Biometric-Cloakroom-Apartments/dp/B0BCJKNTGS/.
[19] Daniel Genkin, Noam Nissan, Roei Schuster, and Eran Tromer. 2022. Lend Me Your Ear: Passive Remote Physical Side Channels on PCs. In *Proceedings of the 31st USENIX Security Symposium.* 4437–4454.
[20] Daniel Genkin, Lev Pachmanov, Itamar Pipman, Eran Tromer, and Yuval Yarom. 2016. ECDSA key extraction from mobile devices via nonintrusive physical side channels. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS).* 1626–1638.
[21] R Blanco Gonzalo, Barbara Corsetti, Ines Goicoechea-Telleria, Anas Husseis, Judith Liu-Jimenez, Raul Sanchez-Reillo, Teodors Eglitis, Elakkiya Ellavarason, Richard Guest, Chiara Lunerti, et al. 2018. Attacking a Smartphone Biometric Fingerprint System: A Novice's Approach. In *Proceedings of the International Carnahan Conference on Security Technology (ICCST).* IEEE, 1–5.
[22] Kang Han and Wei Xiang. 2022. Inference-Reconstruction Variational Autoencoder for Light Field Image Reconstruction. *IEEE Transactions on Image Processing* 31 (2022), 5629–5644.
[23] Sam Farisa Chaerul Haviana, Dedy Kurniadi, et al. 2016. Average hashing for perceptual image similarity in mobile phone application. *Journal of Telematics and Informatics* 4, 1 (2016), 12–18.
[24] Jiaji He, Xiaolong Guo, Mark Tehranipoor, Apostol Vassilev, and Yier Jin. 2021. EM side channels in hardware security: Attacks and defenses. *IEEE Design and Test* 39, 2 (2021), 100–111.
[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[26] HONGFA. 2023. HFD4 SUBMINIATURE SIGNAL RELAY. (2023). https://www.hongfa.com/Product/Item/HFD4.

[27] David Hsu. 2016. Fingerprint Sensor Technology And Security Requirements. (2016). https://semiengineering.com/fingerprint-senor-technology-and-security-requirements/.

[28] Qinhong Jiang, Xiaoyu Ji, Chen Yan, Zhixin Xie, Haina Lou, and Wenyuan Xu. 2023. GlitchHiker: Uncovering Vulnerabilities of Image Signal Transmission with IEMI. In *Proceedings of the 32st USENIX Security Symposium*, Vol. 23.

[29] Wenqiang Jin, Srinivasan Murali, Huadi Zhu, and Ming Li. 2021. Periscope: A Keystroke Inference Attack Using Human Coupled Electromagnetic Emanations. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 700–714.

[30] Hyun-Joon Kim-Lee, Seog Woo Hong, Dong Kyun Kim, Jinmyoung Kim, Hong Suk Kim, Seok-Whan Chung, Eun-Hyoung Cho, Hae-Sung Kim, and Byung-Kyu Lee. 2020. On-screen fingerprint sensor with optically and electrically tailored transparent electrode patterns for use on high-resolution mobile displays. *Microsystems & Nanoengineering* 6, 1 (2020), 98.

[31] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[32] Mohammad Rami Koujan, Michail Christos Doukas, Anastasios Roussos, and Stefanos Zafeiriou. 2020. Head2head: Video-based neural head synthesis. In *Proceedings of the 15th IEEE International Conference on Automatic Face and Gesture Recognition*. 16–23.

[33] David M Kreindler and Charles J Lumsden. 2016. The effects of the irregular sample and missing data in time series analysis. In *Nonlinear Dynamical Systems Analysis for the Behavioral Sciences Using Real Data*.

[34] Hoyeon Lee, Seungyeon Kim, and Taekyoung Kwon. 2017. Here is your fingerprint! Actual risk versus user perception of latent fingerprints and smudges remaining on smartphones. In *Proceedings of ACSAC*. 512–527.

[35] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. 2016. When CSI meets public WiFi: Inferring your mobile phone password via WiFi signals. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 1068–1079.

[36] Qianru Liao, Yongzhi Huang, Y Zhong, H Jin, and K Wu. 2022. MagEar: eavesdropping via audio recovery using magnetic side channel. In *Proceedings of MobiSys*. 371–383.

[37] Jian Liu, Chen Wang, Yingying Chen, and Nitesh Saxena. 2017. VibWrite: Towards finger-input authentication on ubiquitous surfaces via physical vibration. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 73–87.

[38] Zhuoran Liu, Niels Samwel, Léo Weissbart, Zhengyu Zhao, Dirk Lauret, Lejla Batina, and Martha Larson. 2020. Screen gleaning: A screen reading TEMPEST attack on mobile devices exploiting an electromagnetic side channel. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*.

[39] EGO INNOVATION LTD. 2021. EGO MAGPOWER Gen.2 6000mAh 15W magsafe powerbank. (2021). https://www.egoshop.co/en/products/ego-magpower-15w-magsafe-6000mah-powerbank-1.

[40] Henrique Teles Maia, Chang Xiao, Dingzeyu Li, Eitan Grinspun, and Changxi Zheng. 2022. Can one hear the shape of a neural network?: Snooping the GPU via Magnetic Side Channel. In *Proceedings of the 31st USENIX Security Symposium*.

[41] Seita Maruyama, Satohiro Wakabayashi, and Tatsuya Mori. 2019. Tap'n ghost: A compilation of novel attack techniques against smartphone touchscreens. In *Proceedings of the 40th IEEE Symposium on Security and Privacy (SP)*. 620–637.

[42] Arduino Nano. 2022. Arduino Nano Document. (2022). https://docs.arduino.cc/hardware/nano.

[43] Tao Ni, Guohao Lan, Jia Wang, Qingchuan Zhao, and Weitao Xu. 2023. Eavesdropping Mobile App Activity via {Radio-Frequency} Energy Harvesting. In *32nd USENIX Security Symposium (USENIX Security 23)*. 3511–3528.

[44] Tao Ni, Jianfeng Li, Xiaokuan Zhang, Chaoshun Zuo, Wubing Wang, Weitao Xu, Xiapu Luo, and Qingchuan Zhao. 2023. Exploiting Contactless Side Channels in Wireless Charging Power Banks for User Privacy Inference via Few-shot Learning. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*.

[45] Tao Ni, Xiaokuan Zhang, and Qingchuan Zhao. 2023. EM Fingerprints: Attack In-Display Fingerprint Sensors via Electromagnetic (EM) Side Channel. (2023). https://em-fingerprints.github.io.

[46] Tao Ni, Xiaokuan Zhang, Chaoshun Zuo, Jianfeng Li, Zhenyu Yan, Wubing Wang, Weitao Xu, Xiapu Luo, and Qingchuan Zhao. 2023. Uncovering User Interactions on Smartphones via Contactless Wireless Charging Side Channels. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 3399–3415.

[47] Yuval Nirkin, Yosi Keller, and Tal Hassner. 2019. Fsgan: Subject agnostic face swapping and reenactment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 7184–7193.

[48] Hyun Sung Park, Hyundo Shin, Jisoo Kim, Yudeok Seo, Dongjin Seo, and Yongjo Kim. 2020. P-129: Operating Frequency and Sensitivity Prediction of In-Display Ultrasonic Fingerprint Sensing Systems. In *SID Symposium Digest of Technical Papers*, Vol. 51. Wiley Online Library, 1851–1854.

[49] Nandini Patil. 2023. In-Display Fingerprint Sensors Market. (2023). https://issuu.com/npatil29/docs/in-display_fingerprint_sensors_market.

[50] Chang Peng, Mengyue Chen, and Xiaoning Jiang. 2021. Under-display ultrasonic fingerprint recognition with finger vessel imaging. *IEEE Sensors Journal* (2021).

[51] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. 2018. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of ECCV*. 818–833.

[52] K Ramamohan Rao and Ping Yip. 2014. *Discrete cosine transform: algorithms, advantages, applications*. Academic press.

[53] Aditya Singh Rathore, Weijin Zhu, Afee Daiyan, Chenhan Xu, Kun Wang, Feng Lin, Kui Ren, and Wenyao Xu. 2020. SonicPrint: A generally adoptable and secure fingerprint biometrics in smart devices. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 121–134.

[54] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF CVPR*. 10684–10695.

[55] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

[56] Haoqi Shan, Boyi Zhang, Zihao Zhan, Dean Sullivan, Shuo Wang, and Yier Jin. 2022. Invisible Finger: Practical Electromagnetic Interference Attack on Touchscreen-based Electronic Devices. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*. 1246–1262.

[57] Yahaya Isah Shehu, Ariel Ruiz-Garcia, Vasile Palade, and Anne James. 2018. Sokoto coventry fingerprint dataset. *arXiv preprint arXiv:1807.10609* (2018).

[58] Dai Shi, Dan Tao, Jiangtao Wang, Muyan Yao, Zhibo Wang, Houjin Chen, and Sumi Helal. 2021. Fine-grained and context-aware behavioral biometrics for pattern lock on smartphones. *Proceedings of the ACM IMWUT* (2021).

[59] Synaptics. 2017. Synaptics Brings World's First In-Display Fingerprint Sensors for Smartphones to Mass Production with a Top Five OEM. (2017). https://www.synaptics.com/company/news/Clear-ID-mass-production.

[60] Zhiqiang Tao, Hongfu Liu, Huazhu Fu, and Yun Fu. 2017. Image cosegmentation via saliency-guided constrained clustering with cosine similarity. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 31.

[61] R QIDI TECHNOLOGY. 2023. R QIDI TECHNOLOGY X-CF Pro Industrial Grade 3D Printer. (2023). https://www.amazon.com/QIDI-TECHNOLOGY-Industrial-Specially-11-8x9-8x11-8/dp/B09L7XCW3B/.

[62] Danny Thakkar. 2023. Fingerprint Reader Technology Comparison: Optical Fingerprint Scanner; Capacitive-based Fingerprint Reader and Multispectral Imaging Sensor. (2023). https://www.bayometric.com/fingerprint-reader-technology-comparison/.

[63] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2016. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE/CVF CVPR*. 2387–2395.

[64] Dries Van Wageningen and Toine Staring. 2010. The Qi wireless power standard. In *Proceedings of 14th International Power Electronics and Motion Control Conference (EPE-PEMC)*. IEEE, S15–25.

[65] Tao Wei and Yulong Zhang. 2015. Fingerprints On Mobile Devices: Abusing And Leaking. (2015).

[66] Petros Xanthopoulos, Panos M Pardalos, Theodore B Trafalis, Petros Xanthopoulos, Panos M Pardalos, and Theodore B Trafalis. 2013. Linear discriminant analysis. *Robust data mining* (2013), 27–33.

[67] Guixin Ye, Zhanyong Tang, Dingyi Fang, Xiaojiang Chen, Kwang In Kim, Ben Taylor, and Zheng Wang. 2017. Cracking android pattern lock in five attempts. In *Proceedings of the Network and Distributed System Security (NDSS) Symposium*.

[68] Dong Yu, Kaisheng Yao, Hang Su, Gang Li, and Frank Seide. 2013. KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In *Proceedings of the IEEE ICASSP*.

[69] Muhammad Rehman Zafar and Munam Ali Shah. 2016. Fingerprint authentication and security risks in smart devices. In *Proceedings of the 22nd International Conference on Automation and Computing (ICAC)*. 548–553.

[70] Zihao Zhan, Zhenkai Zhang, Sisheng Liang, Fan Yao, and Xenofon Koutsoukos. 2022. Graphics peeping unit: Exploiting EM side-channel information of GPUs to eavesdrop on your neighbors. In *Proceedings of IEEE Symposium on Security and Privacy (SP)*. 1440–1457.

[71] Xinchen Zhang, Yafeng Yin, Lei Xie, Hao Zhang, Zefan Ge, and Sanglu Lu. 2020. TouchID: User authentication on mobile devices via inertial-touch gesture analysis. *Proceedings of the ACM IMWUT* (2020).

[72] Yang Zhang, Peng Xia, Junzhou Luo, Zhen Ling, Benyuan Liu, and Xinwen Fu. 2012. Fingerprint attack against touch-enabled devices. In *Proceedings of the 2nd ACM workshop on security and privacy in smartphones and mobile devices*. 57–68.

[73] Man Zhou, Qian Wang, Jingxiao Yang, Qi Li, Feng Xiao, Zhibo Wang, and Xiaofeng Chen. 2018. Patternlistener: Cracking android pattern lock using acoustic signals. In *Proceedings of ACM CCS*. 1775–1787.