Uncovering User Interactions on Smartphones via Contactless Wireless Charging Side Channels

Tao Ni*, Xiaokuan Zhang†, Chaoshun Zuo‡, Jianfeng Li§, Zhenyu Yan¶, Wubing Wang‖,
Weitao Xu*, Xiapu Luo§, Qingchuan Zhao*

*City University of Hong Kong, †George Mason University, ‡The Ohio State University,
§The Hong Kong Polytechnic University, ¶The Chinese University of Hong Kong, ‖DBAPPSecurity Co., Ltd

***Abstract*— Today, there is an increasing number of smartphones supporting wireless charging that leverages electromagnetic induction to transmit power from a wireless charger to the charging smartphone. In this paper, we report a new *contactless* and *context-aware* wireless-charging side-channel attack, which captures two physical phenomena (*i.e.*, the coil whine and the magnetic field perturbation) generated during this wireless charging process and further infers the user interactions on the charging smartphone. We design and implement a three-stage attack framework, dubbed WISERS, to demonstrate the practicality of this new side channel. WISERS first captures the coil whine and the magnetic field perturbation emitted by the wireless charger, then infers *(i)* inter-interface switches (*e.g.*, switching from the home screen to an app interface) and *(ii)* intra-interface activities (*e.g.*, keyboard inputs inside an app) to build *user interaction contexts*, and further reveals sensitive information. We extensively evaluate the effectiveness of WISERS with popular smartphones and commercial-off-the-shelf (COTS) wireless chargers. Our evaluation results suggest that WISERS can achieve over 90.4% accuracy in inferring sensitive information, such as screen-unlocking passcode and app launch. In addition, our study also shows that WISERS is resilient to a list of impact factors.**

I. Introduction

Recent years have witnessed the advance of wireless charging technology for smartphones. Wireless charging standards, *e.g.*, Qi [66] introduced by the Wireless Power Consortium (*WPC*), have been widely adopted, and supporting wireless charging has become an almost must-have feature for newly released smartphones. By the end of 2021, there were more than one billion newly released smartphones equipped with a wireless charging module [12].

In this paper, we present a new side channel targeting wireless chargers that can be leveraged to uncover *fine-grained* user interactions with charging smartphones and reveal sensitive information (*e.g.*, screen-unlocking passcode and keyboard inputs). Specifically, this new side-channel attack utilizes the emitted coil whine and perturbations in the ambient magnetic field when charging a smartphone wirelessly. Different from existing side-channel works in *wired* charging [22], [64], [70], [77] and *wireless* charging [34], [75] that require physical access to obtain current traces, this attack can work *contactlessly* and does not require the knowledge of the current readings. It also makes no assumption about compromising the victim's smartphones (*e.g.*, installing a malicious app [19], [45], [48], [83]), and an attacker can launch the attack by placing a measurement device (*e.g.*, a smartphone) in close proximity (*e.g.*, 8in or 20cm) to the victim's smartphone. Moreover, this attack can leak fine-grained information on smartphones even when the battery level is lower than 80%, which is considered impossible in prior work [34].

Our newly discovered side-channel attack stems from two inevitable physical phenomena, *i.e.*, the coil whine and the magnetic field perturbation, that are brought by the power transmission between a wireless charger and a smartphone. A user's interaction with the smartphone in wireless charging, such as typing texts, could change the displayed content on the touchscreen. Changes of displaying content could often change the power supply (the amount of current) in the wireless charger, according to nowadays charging standards (*e.g.*, Qi [66]). The current changes in the internal coil of the charger will incur electromagnetic forces, based on Ampere's force law, that slightly deforms and vibrates the coil, resulting in the coil whine and the magnetic field perturbations surrounding the wireless charger. These two phenomena can be detected by sensing devices nearby.

To study the practicality of this novel side-channel attack, we introduce WISERS, a *WI*reles*S* charg*ER* *S*ensing system that aims to uncover user interactions in a *context aware* manner based on the collected coil whine and magnetic field perturbation traces. To this end, we introduce a novel concept of *user interaction context* to comprehensively describe a series of user interactions with the smartphone in two orthogonal aspects: *(i) inter-interface switches* that represent every switch from one interface (*e.g.*, the home screen) to another (*e.g.*, an arbitrary app UI interface); *(ii) intra-interface activities* that represent actions performed within a UI interface (*e.g.*, typing on a soft keyboard). Specifically, WISERS consists of three stages. First, it senses a set of features (*e.g.*, battery level in a smartphone) impacting the measurement of both the coil whine and the magnetic field perturbation, then configures itself accordingly to prepare an attack. Next, it leverages the coil whine to infer inter-interface switches and utilizes the magnetic field perturbations to uncover intra-interface activities. Based on the inferred switches and uncovered activities, WISERS builds the *user interaction context* and finally interprets particular user interactions to reveal specific sensitive information (*e.g.*, typing the username and password in a particular app).

We have implemented a prototype of WISERS and comprehensively evaluated its performance by analyzing the ef-

fectiveness of each of its stages and demonstrating end-to-end attacks. our prototype uses an iPhone to record the coil whine through its microphone and measure magnetic field perturbations via its magnetometer. As a proof-of-concept, this prototype focuses on three particular intra-interface activities (*i.e.*, app launch, keyboard open, and keystroke) and four types of user interfaces (*i.e.*, off screen, lock screen, home screen, and app interface) to reveal sensitive information including screen-unlocking passcode, cross-app searching content, and app-specific sensitive user inputs. Accordingly, we prepared eight datasets consisting of data traces collected from top 15 apps in each of 24 categories (360 in total) in the App Store as of mid February, 2022. WISERS achieves an accuracy of 92.5% to infer inter-interface switches, 91.8% and 87.9% to recognize an app at launch in the closed-world and open-world setting, respectively, and 99.0% to identify a keyboard open. In respect of uncovering keystrokes ranging from 1 to 15 in length on the screen-unlocking keyboard, the numeric-only keyboard, and the full-size keyboard, WISERS also reaches the accuracy of 94.4%, 92.6%, and 90.6%, respectively, within five attempts.

In addition, we conduct 40 end-to-end attack trials to reveal the aforementioned three types of sensitive information from a series of user interactions. Each series starts by unlocking the screen and ends with typing sensitive information in one of eight popular apps such as WhatsApp, PayPal, and Safari. WISERS captures each user interaction context and reveals sensitive information with a 100% success rate within five attempts. Moreover, we also present an extensive analysis of the practical impact factors, such as different chargers and smartphones. Our results show that WISERS is resilient to a variety of impact factors, indicating that WISERS can be applied on different wireless chargers, battery levels, users, smartphones, and from different distances.

Ethical Consideration. We take ethical considerations seriously in this study. Data collections from volunteer participants were conducted with IRB approval. Screen-unlocking passcodes, cross-app searching content, and privacy-sensitive user input were generated randomly for effectiveness evaluation only, and we only use our own accounts to evaluate keystrokes uncovering inside apps such as WhatsApp. WISERS has never been released to any other parties.

Contributions. We make the following contributions:

- **New side-channel attack vectors.** We introduce a new side-channel attack that leverages the emitted coil whine and changes in the ambient magnetic field during the process of wireless charging to infer fine-grained and sensitive user interactions on smartphones in a *contactless* manner.
- **A new attack system.** We propose WISERS, a three-stage, and context-aware attack framework, and implement a prototype to demonstrate the feasibility of the new side channel. Our prototype introduces a novel concept of user interaction contexts to reveal sensitive information such as screen-unlocking passcode and sensitive user inputs.
- **Extensive evaluation.** WISERS is evaluated extensively, and the results show that it can effectively construct *user interaction contexts* based on the coil whine and the magnetic field perturbation traces. In addition, our study shows that the demonstrated attack is resilient to a list of impact factors.

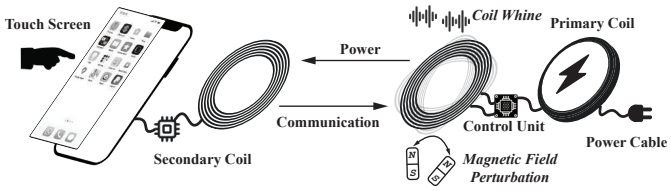


Fig. 1: Wireless charging principle.

II. Background

A. Wireless Charging on Smartphones and Qi Standard

Qi standard [73] is a wireless charging standard widely supported by smartphones [34]. Wireless chargers holding Qi certification could leverage electromagnetic induction to charge smartphones by providing 5-15 watts of power [74]. An illustration of this wireless charging process is presented in Figure 1. When a wireless charger detects a smartphone is put on, the charger initiates a series of communications with the smartphone for power transfer configuration and its control unit converts the input DC to power its coil (primary coil). The primary coil runs an alternating current that incurs alternating voltages in the built-in coil (secondary coil) of the smartphone to achieve the charging purpose. Particularly, during this power transfer phase, the wireless charging unit in the smartphone continuously talks to the control unit in the wireless charger to change the power supply by adjusting the current running in the primary coil. Changes in the power supply are coordinated with the different power requirements of activities performed by the smartphone at charging [66]. Activities using more power make the smartphone request more power supply from the wireless charger [66], [75]. This charging process terminates if the smartphone is taken away or it sends messages to the charger to stop charging, *e.g.*, the battery is fully charged.

B. Physical Phenomena Generated by Wireless Charging

Since a charging system under the Qi standard [73] uses electromagnetic induction to transfer power from the primary coil in the wireless charger to the secondary coil in charging devices, an ambient magnetic field is generated [72]. The dynamically changing current could make the coils vibrate during this charging process, resulting in the coil whine and perturbation in the ambient magnetic field.

Coil whine. Coil whine, *a.k.a.*, electromagnetically induced acoustic noise, is a microphonics phenomenon. As shown in Figure 1, it is generated by the vibration or deformation of coil materials under the excitation of a series of electromagnetic forces, including Maxwell stress tensor, magnetostriction, and Lorentz force [10]. Coil whine can be in different frequency ranges, making it either human audible (between 20 Hz and 20 kHz) or inaudible [71].

Magnetic field perturbation. The dynamic current changes during the wireless charging process can impact the ambient magnetic field and result in magnetic field perturbations. As such, this perturbation can be measured by the changes in the magnetic field over a period of time. At a specific time point, the magnetic field could be described by a vector consisting of the coordinates in a 3D-Cartesian coordinate.

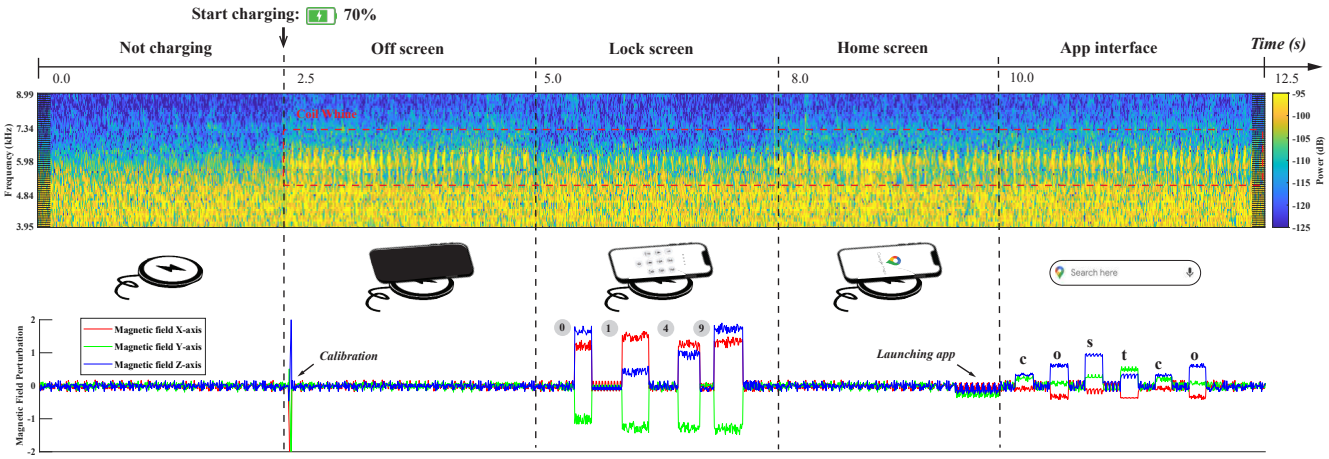
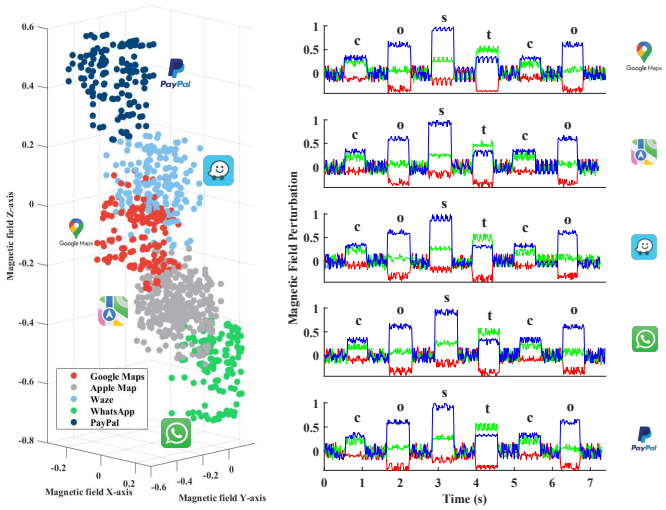


Fig. 2: Motivating example scenario: a user places a smartphone with 70% battery left on a wireless charger, unlocks the screen with the passcode (i.e., 0149), clicks app icon to open Google Map, and types "costco" to search for nearby supermarket locations. Upper Figure: the corresponding power spectrum of the coil whine; Lower Figure: the strength and directions in three dimensions of the ambient magnetic field.



(a) Launching apps. (b) Typing "costco".

Fig. 3: Magnetic field perturbation in five different apps.

III. Motivation, Principle, and Threat Model

A. A Motivating Example

This section presents a real scenario that motivates this study. A user puts a smartphone on a wireless charger, unlocks the screen with the passcode, and clicks the app icon to open Google Map to search for wholesale stores by typing "costco" in the search bar. As mentioned in §II-A, these user interactions with the smartphone could impact the current in both the primary coil in the wireless charger and the secondary coil in the smartphone, which results in the coil whine [21] and the magnetic field perturbations surrounding the wireless charger.

Interestingly, the coil whine and magnetic field perturbations appear to reflect corresponding user interactions. We use the microphone and the magnetometer of another smartphone to capture these two physical phenomena stemming from user interactions with the target smartphone, and show the captured data align with corresponding user interactions in Figure 2. The middle part of Figure 2 illustrates the sequence of user interactions, the upper part shows the power spectrum of the coil whine, and the lower part presents the magnetic field perturbations. As can be seen, switches between interfaces (*e.g.*, screen off to lock screen) are more observable in the power spectrum of the coil whine, and finer-grained activities in an interface, such as the app launch and keystrokes, are more noticeable from the ambient magnetic field perturbations. Note that, since it could result in a significant magnetic field perturbation if a smartphone is put on the wireless charger, as shown in Figure 2, we calibrate the ambient magnetic field to better illustrate the association between the following user interactions and magnetic field perturbations.

The observation that magnetic field perturbations could show finer-grained activities raises two additional questions, *i.e.*, *(i)* whether the launches of different apps result in different magnetic field perturbations and *(ii)* whether the same keyboard input in different apps leads to a similar sequence of perturbations. To answer these questions, we further conduct experiments on another four popular iOS apps, including two map apps (*i.e.*, Apple Map and Waze) and two apps delivering totally different services (*i.e.*, one financial app, Paypal, and one chatting app, WhatsApp), and present their results in Figure 3. Specifically, Figure 3a presents the magnetic field perturbation resulting from the first five seconds after launching different apps, and Figure 3b shows the perturbation of typing the same word, *i.e.*, "costco", in different apps. Obviously, launching different apps results in different magnetic field perturbations; the same keystroke produces very similar perturbations across different apps. Therefore, coil whine and magnetic field perturbations could potentially construct a new side channel to infer user interactions with a smartphone when it is being charged on a wireless charger.

B. The Fundamental Principle

The principle of wireless charging. Wireless charging leverages electromagnetic induction to transfer power from the primary coil to the secondary coil. First, the primary coil in the charger generates an inductive electromagnetic field, *i.e.*, $\Phi_s(t)$, in the secondary coil based on the Biot-Savart law (Equation 1). The inductive electromagnetic field produces an induced voltage $V_s(t)$ to power the smartphone following Faraday's law (Equation 2).

$$\Phi_s(t) = \eta\Phi_p(t) = \eta\frac{\mu_0 N_p I_p(t)}{2r_p}, \quad (1)$$

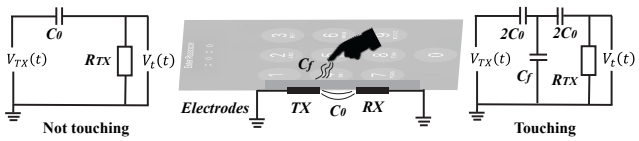


Fig. 4: Illustration of finger-coupling effects in a touching event.

$$V_s(t) = N_s \frac{\Delta\Phi_s(t)}{\Delta t} = \eta \frac{N_s}{N_p} \cdot \frac{\mu_0 \Delta I_p(t)}{2 r_s \Delta t}, \tag{2}$$

where $\Phi_p(t)$ and $I_p(t)$ are the electromagnetic field and the running current in the primary coil, N_p and r_p are the turns and radius of the primary coil, N_s and r_s are the turns and radius of the secondary coil, η and μ_0 represents the energy transmission ratio and the magnetic constant.

The principle of the associations between user interactions and the coil whine. The running current in the coil generates electromagnetic forces that incur vibration and deformation in the coil, which results in the coil whine. In particular, a user interaction could result in a change of the current in the primary coil, $\Delta I_p(t)$, which then changes the electromagnetic forces exerted on the coil, $\Delta F_p(t)$, according to the Ampere's force law (Equation 3).

$$\Delta F_p(t) = \Delta I_p(t) L_p \times \Phi_p(t) \tag{3}$$

where L_p is the length of the primary coil. Therefore, $\Delta F_p(t)$ distorts the amplitude A_{cw} and frequency f_{cw} of the coil whine $\Delta\mathcal{S}(A_{cw}, f_{cw})$ emitted from the wireless charging coil.

$$\Delta F_p(t) \Rightarrow \Delta\mathcal{S}(A_{cw}, f_{cw}) \tag{4}$$

The principle of the associations between user interactions and magnetic field perturbations. User interactions with a smartphone continuously and dynamically change the current in the coil in wireless charging, which leads to magnetic field perturbations. Specifically, for user interaction, such as pressing a button, both the changes in the load of $\Delta R(t)$ on the secondary coil [66] and the finger-coupling effects [32] incur the magnetic field perturbations because the capacitance touchscreen is made of a grid of touch sensors (electrodes). As illustrated by the equivalent circuits in Figure 4, when a finger touches a button, the finger-coupling effect changes the local capacitance of $\Delta C_f(t)$ and results in the changing voltages $V_t(t)$ of this button (Equation 5), which perturbs the corresponding magnetic field. Note that $V_{TX}(t)$ and R_{TX} are the driven voltage and resistor of the touch sensor grid.

$$\begin{cases} V_t(t) = V_{TX}(t) \cdot \frac{R_{TX}}{R_{TX}+1/(j2\pi f C_0)} \text{ (Not touching)} \\ V_t(t) = V_{TX}(t) \cdot \frac{R_{TX}}{R_{TX}+1/(j4\pi f C_0)+\Delta Z_f(t)} \text{ (Touching)} \\ \Delta Z_f(t) = 1/(\frac{1}{1/(j2\pi f \Delta C_f(t))} + \frac{1}{1/(j4\pi f C_0)}) \text{ (Impedance)} \end{cases} \tag{5}$$

Since the key-pressing animation and finger-coupling effects happen together (a detailed description of the key-pressing animation and an investigation of the finger-coupling effects are presented in the Appendix), the change of the current $\Delta I(t)$ and the induced electromagnetic field $\Delta\Phi(t)$ at the certain touching point (Equation 6) finally produce the perturbations on the inductive electromagnetic field, $\Phi_s(t)$.

$$\Delta I(t) = \frac{V_s(t) + \Delta V_t(t)}{\Delta R(t)} \Rightarrow \Delta\Phi(t) = \frac{\mu_0 N_s \Delta I(t)}{2 r_s} \tag{6}$$

(a) Cafe (b) Airport

Fig. 5: Public wireless charging facility examples.

C. Threat Model

We consider a common scenario in wireless charging side-channel attacks [34], [75] where a victim is playing with his or her smartphone while charging it on a wireless charger, such as a public wireless charging station in a Cafe (Figure 5). The adversary can place the attacking device in close proximity (*e.g.*, 8in or 20cm) to the target wireless charger and be aware of the distance and the relative angle between them. The attacking device can record environmental sounds to extract the coil whine and measure the ambient magnetic field, and it is placed before the victim puts the smartphone on the charger. In addition, the attacking device is not required to be professional but could be a commodity smartphone. Most smartphones now come with a magnetometer that can measure the ambient magnetic field accurately [13], and their microphones are sensitive enough with a sampling rate of 44.1 kHz -48 kHz [80] to capture most coil whine generated in charging a smartphone with commercial off-the-shelf (*COTS*) wireless chargers (*e.g.*, Apple MagSafe Charger). Additionally, placing this monitoring device in close proximity [36], [44] could also be achieved in public facilities, as shown in the examples of Figure 5.

Moreover, while assuming the adversary can observe the target wireless charger, we also assume that the adversary *cannot* compromise *(i)* the charging station to monitor current traces in the power cable of a wireless charger before the power conversion, *(ii)* the wireless charger to monitor the current traces in the primary coil after the conversion, and *(iii)* the victim smartphone including modifying hardware or leveraging an installed malicious app or any software vulnerabilities.

IV. Attack Framework

This section presents the details of our proposed three-stage attack framework, WISERS. As shown in Figure 6, *(i)* the first stage is to prepare an attack which includes inferring the battery level left in the charging smartphone and calibrating the magnetic field (§IV-A); *(ii)* the second stage is to build the user interaction context from both the inter-interface switches inferred from the traces of coil whine and the intra-interface activities uncovered from the traces of magnetic field perturbations (§IV-B); and *(iii)* the last stage is to utilize the established user interaction context to uncover user privacy (§IV-C). The implementation of the WISERS prototype is detailed in §IV-D.

A. Preparing Attacks

WISERS aims to identify user interactions with the smartphone based on unique patterns in the traces of coil whine and magnetic field perturbations stemming from a wireless charging process; therefore, its recognition accuracy depends on the precision in recognizing those patterns from the traces. There

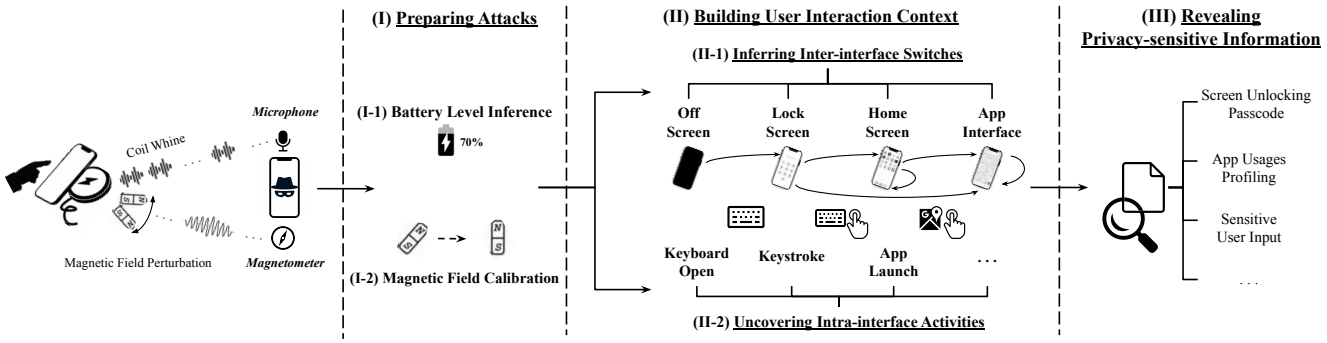


Fig. 6: Overview of WISERS.

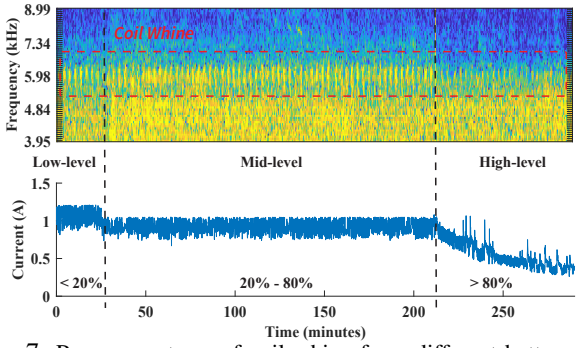


Fig. 7: Power spectrum of coil whine from different battery levels.

are two primary factors, *i.e.*, *(i)* the battery level of the charging smartphone and *(ii)* relative positions between the wireless charger and the measurement device (*e.g.*, magnetometer in a smartphone), that could impact the pattern recognition because changes in these two factors could result in different patterns of the same user interaction. As such, in addition to *identifying the trigger condition* to start the following attacks (*i.e.*, when the smartphone starts charging), WISERS has to *infer the battery level* of the victim's charging smartphone and *calibrate the magnetic field* surrounding the wireless charger.

Identifying the trigger condition. The trigger condition of WISERS to initiate an attack is the moment when a smartphone is put on a wireless charger. While this action simultaneously generates the coil whine and perturbs the magnetic field of the wireless charger, neither of them is sufficient to indicate the trigger condition because of the environmental noise. In particular, various environmental factors could perturb the magnetic field and/or emit sounds with the same frequency range as that of the coil whine in a wireless charger. For example, the frequency of the coil whine in a wireless charger is within 4 to 9 kHz, which is in the same range as the sounds of cutting metal or birds chirping. Therefore, WISERS identifies the trigger condition by capturing an abrupt change of the coil whine and magnetic field perturbation simultaneously. Specifically, it first uses the magnetometer to log the direction and strength of the magnetic field in a time series to identify a significant perturbation and applies a high-pass filter to remove environmental noises, such as low-frequency noise resulting from screen touching or pressing, based on the frequency range of the coil whine of a particular wireless charger. Next, it leverages the Short-term Fourier Transform (*STFT*) and a periodic Hann window function to recognize the abrupt change in the filtered power spectrum.

Inferring the battery level. After identifying the trigger condition, WISERS next infers the battery level of the charging smartphone. At charging, the smartphone actively communicates with the wireless charger to adapt the power supply based on the battery level of the smartphone following the Qi wireless charging protocol [66]. Currently, COTS wireless chargers often separate the charging process into different stages based on the battery level and provide different supplies (*i.e.*, the amount of current) in each stage, while the number of stages varies among different chargers. For example, as shown in Figure 7, our 10 W Gikfun charger separates the whole charging process into three stages associated with the battery level, *i.e.*, low-level (below 20%), mid-level (between 20% and 80%), and high-level (more than 80%). WISERS infers the battery level by classifying the signal power of the coil whine into different charging stages because different amounts of current generate different patterns of the coil whine. Specifically, after reviewing the acoustic features describing the signal power of a sound, we decide to use all 86 relevant features to model a coil whine trace as a 1×86 vector and adopt the random-forest classification algorithm because of its advances in handling high dimensional feature vectors.

Calibrating the magnetic field. As mentioned in §III-B, user interaction with the smartphone could result in subsequent magnetic field perturbations; however, the perturbation pattern of a specific user interaction varies in different relative positions between a wireless charger and the magnetic field measuring device. To ease the efforts in mapping different patterns of the same interaction in a nearly infinite possible space of the relative positions, WISERS calibrates the coordinates of the magnetic field measured from all possible relative positions between two devices to the coordinates of a pre-setting position.

As shown in Figure 8, before putting a smartphone on the wireless charger, we first place the magnetic field measuring device at a specific position with an attacking distance d and an initial relative angle θ to the wireless charger as the pre-setting position, and set its measured magnetic field coordinates $P_0 = (x_0, y_0, z_0)$ as the origin of the coordinate. After a smartphone is put on the charger, we use a direction vector $\overrightarrow{P_0P_1}$ to represent the magnetic field drifts, where $P_1 = (x_1, y_1, z_1)$ is the new measured magnetic field coordinates. Next, we use circular arc interpolation method [79] to calibrate the coordinates in the X-Y plane using the Equation 7, where θ is the position deviation from a random position $P_2(x_2, y_2, z_2)$ to P_1, to calibrate the coordinates of a measuring device.

$$\begin{cases} x_1 = x_2 - d(1 - cos\theta) \\ y_1 = y_2 - dsin\theta \\ z_1 = z_2 \end{cases} \tag{7}$$

After calibrating the coordinates to our pre-setting position,

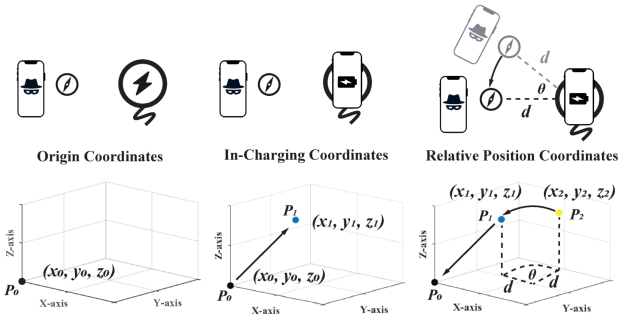


Fig. 8: Magnetic field calibration.

we leverage $\overrightarrow{P_0P_1}$ to reset the coordinates to the origin of the coordinate in the pre-setting position by deducting the offsets to accomplish the magnetic field calibration.

B. Building User Interaction Context

WISERS builds a user interaction context to recognize user interactions. This context combines two orthogonal aspects of user interaction, *i.e.*, inter-interface switches and intra-interface activities. These two aspects are extracted from the coil whine and the magnetic field perturbation.

Inferring inter-interface switches. An inter-interface switch refers to a switch between different interfaces shown on the screen of a smartphone, such as switching from the home screen to an arbitrary app interface. WISERS leverages coil whine to infer inter-interface switches because these switches cause significant changes in the power spectrum of the coil whine than the magnetic field perturbation, which can be seen in the motivating example (§III-A).

Types of inter-interface switches. At a high level, we first systematically categorize smartphone interfaces into four groups: *(i)* off screen interface, *(ii)* lock-screen interface, *(iii)* home screen interface, and *(iv)* app interface. Note that the app interface refers to the general interface of any app. According to these categories, while a series of interactions could involve multiple switches of different lengths in practice, these four types of interfaces could systematically compose six atomic and feasible switches: *(i)* off screen to lock screen, *(ii)* lock screen to home screen, *(iii)* lock screen to app interface, *(iv)* home screen to app interface, *(v)* home screen to home screen, and *(vi)* app interface to app interface.

Inferring inter-interface switches. As observed in Figure 2, the power spectrum of the coil whine could reflect different types of interfaces. Hence, similar to inferring the battery level (§IV-A), WISERS leverages the unique pattern of each specific type of interface from the power spectrum of the coil whine to distinguish them, and then infers the associated inter-interface switches. Specifically, each type of interface is modeled as an 86 acoustic feature vector, and the random-forest classification algorithm is then used to recognize each type of interfaces and switches between them.

Uncovering intra-interface activities. Alongside inferring inter-interface switches, WISERS also aims to recover intra-interface activities. These activities refer to the finer-grained reactions to user interaction within a single interface, such as launching an app, opening a soft keyboard, and typing on a keyboard. As mentioned in §III-A, magnetic field perturbations could reflect user interactions in much finer granularity than the coil whine; therefore, this component aims to achieve the objective by monitoring magnetic field perturbations. In addition, since recovering these activities could be formed as a classification problem, we leverage one of the state-of-the-art classification approaches [65] that trains an Attention-Based Bidirectional LSTM (*AttnBiLSTM*) model, turns it into an embedding model by removing the layers after the embedding layer, and uses the embedding model with a Cosine distance to classify magnetic field perturbations into different patterns, each of which associates with a unique intra-interface activity.

Data pre-processing. WISERS first employs a Savitzky–Golay (*S-G*) filter [17] to remove noises in the collected sequential magnetic field perturbations without distorting the signals. Next, considering each activity may last a different length of time in every attempt (*e.g.*, a single keystroke may take 0.05–0.2 second [76]), it also normalizes each activity attempt into the same length of time via property-preserving up-sampling (*e.g.*, nearest neighbor interpolation [55]) or down-sampling (*e.g.*, decimation factor [33]) algorithms.

Training model. Since the AttnBiLSTM model is trained by taking sequences as inputs, extra data preparation is required to transform a series of magnetic field perturbations into a sequence representing a unique interaction. Considering the magnetic field at a specific time is usually described in a 3D-Cartesian coordinate system, (x, y, z), and the magnetic field perturbation could be modeled as a sequence of traces of the magnetic field in a time-series; each magnetic field dimension of a magnetic field perturbation sequence contains 1D time-series data points. As such, we adopt an approach similar to the one proposed in [50] by applying 1D convolutional neural network (*CNN*) to extract features from the time-series data. To this end, an 1D filter is used to capture temporal correlations on each magnetic field dimension, a max pooling layer to reduce the dimension by half, and a flatten layer is adopted to reshape the feature map to one-dimensional sequences. These sequences are the required legitimate input to train an AttnBiLSTM model. In the AttnBiLSTM model, the embedding layer takes in the input sequences and generates a numerical vector. Next, the bidirectional LSTM layers learn the predictive features from the embedded vectors, and the attention layer captures the dependencies between the features and the output. After that, a full-connected layer produces the classification vectors with the same size as profiled classes. Finally, a soft-max layer generates the probability of each class and outputs the predicted class with the highest probability.

Applying classification. Having obtained the embedding model, WISERS will generate the embedding (e_t) of a new sequence of magnetic field perturbation (s_t), and calculate its Cosine distance to each sequence (s_i^j) of a class C_i. s_t will be classified into class C_i if one the Cosine distances between s_t and s_i^j is lower than the threshold.

C. Revealing Sensitive Information

After inferring inter-interface switches and uncovering intra-interface activities, WISERS can establish the user interaction context accordingly and reveal the user's private information. Specifically, it is designed to take a set of attack plans composed by analysts to reveal particular privacy-sensitive information in the user interaction context. This design ensures WISERS scale to launch a variety of attacks in revealing user

privacy. It is worth noting that this user interaction context is necessary for revealing fine-grained user privacy because either those switches or activities may not always provide fine-grained semantics of user interaction with the smartphone. For example, inter-interface switches can tell whether a newly switched interface is an app interface, but cannot recognize which app it belongs to; similarly, as an intra-interface activity, while a user-typed 4-digit keyboard input could be uncovered, its semantics remains uncertain. Fortunately, the user interaction context could assist in understanding such semantics. For example, we can conclude the uncovered 4-digit user input is a passcode to unlock screen if its inter-interface switches start from the off-screen to the lock screen and then to the home screen or an app interface.

D. Implementation

Our prototype implementation of WISERS primarily focuses on three intra-interface activities, *i.e.*, app launch, keyboard open, and keystroke, though it is able to scale to other activities. In addition, the prototype is implemented atop a set of tools, and its details are elaborated in the following.

Processing the coil whine. WISERS leverages the coil whine to infer the battery level left in the charging smartphone to prepare attacks and the inter-interface switches. To achieve these two objectives, it extracts acoustic features of the coil whine and applies the random forest classification algorithm.

1) Acoustic feature extraction. As mentioned earlier, we use a set of 86 acoustic features to describe the power spectrum of the coil whine, including Mel-frequency cepstral coefficients (MFCCs) [56], Gammatone cepstral coefficients (GTCCs) [1], linear prediction cepstrum coefficients (LPCCs) [46], spectral power patterns [2], *etc.* To extract these features, we depend on the MATLAB Audio Toolbox (version 3.0), which provides reliable algorithms (*e.g.* STFT) and effective toolkits (*e.g.* high-pass and S-G filters). The full list of selected MATLAB functions and parameters is shown in Table III in the Appendix.

2) Random forest classification. WISERS uses the random forest classification algorithm to classify different battery levels and types of interfaces. In particular, we set the number of estimators as 100, limit the maximum depth as 32, and use a 10-fold cross-validation.

Monitoring magnetic field perturbations. WISERS uncovers the intra-interface activities from magnetic field perturbation via an AttnBiLSTM classification algorithm. In particular, it first uses a 1D CNN algorithm to extract the features of each magnetic field perturbation sample, which consists of six magnetic field states in a time series. It then converts this series to a one-dimension sequence to meet the requirement of the AttnBiLSTM algorithm as input. Specifically, this CNN algorithm is configured with three input channels and three output channels, activation function with ReLU, and its kernel size as three and stride as one. In respect of the configuration of the AttnBiLSTM algorithm, we set its batch size as 128, embedding dimension as six, hidden size as 50, and use the Cross Entropy Loss and Adam optimizer with a learning rate of 0.001 and epoch of 300.

Configurations for specific intra-interface activities. The current prototype primarily focuses on three specific intra-interface activities. Accordingly, our prototype applies a set of configurations particular to each of these activities.

1) Adaptive threshold in app recognition at launch for closed-world and open-world settings. To recognize an app being launched, we consider both the closed-world and open-world settings. Since our algorithm requires a threshold on the Cosine distance to classify an app, we propose an adaptive threshold mechanism to ensure its scalability. Following the closed-world and open-world settings proposed in similar works [35], [62], [69], we let $A_T = \{app_T^i\}_{i=1}^{m_T}$ (resp. $A_I = \{app_I^i\}_{i=1}^{m_I}$) be the set comprised of all apps in the training stage (resp. the identification stage). In particular, A_I is the subset of A_T ($A_I \subseteq A_T$) in the closed-world setting, while it could contain apps that are unmonitored in the training stage ($A_I \nsubseteq A_T$) in the open-world setting. Next, we choose the threshold based on the training set property. Specifically, we first produce the threshold set $T = \{threshold_T^i\}_{i=1}^{m_T}$ for each closed-world app_T^i class in the training stage. Next, we select the maximum threshold value $T_{max} = maximum\{T\}$ as the approximate open-world threshold. Accordingly, if the Cosine distance of a new app exceeds T_{max}, the embedding model will classify it as an unmonitored app; otherwise, it will be classified as a monitored closed-world class app_T^i if their distance is shorter than T_i.

2) System keyboards in opening recognition. The prototype of WISERS targets three system default keyboards because several input fields only allow these keyboards instead of custom keyboards. In particular, these keyboards include the screen-unlocking keyboard, the numeric-only keyboard, and the full-size QWERTY keyboard.

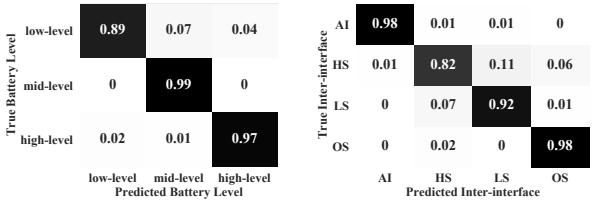
3) Segmentation in uncovering key clicks. In practice, users often type a single word consisting of a sequence of characters of different lengths. Considering the duration of a typing practice contains both key presses and intervals between two presses, we model each interval between two key presses as a special key event, *i.e.*, static key, and leverage this key as an indicator of the segmentation between two key presses. As such, an additional key involved in both training and testing.

Data normalization. As mentioned in §IV-B, a keystroke may last from 0.05 to 0.2 seconds on average [76]. Similarly, users may spend different duration on each interface, and the smartphone could launch an app at different speeds. Therefore, we normalize each coil whine and magnetic field perturbation trace as time series with 0.1-second intervals by applying down-sampling and up-sampling, then use these traces of the same length in both training and testing.

V. Evaluation

A. Evaluation Setup

Our evaluation involves two sets of equipment to collect data and process the collected data for training and testing. To collect data, we use an iPhone 11 as the data collector (the attack device) to collect data from an iPhone 13 Pro (the victim device) charging on a 10 W Gikfun wireless charger at a distance of 8 inches (20 cm). Except for the analysis on inferring the battery level, the battery of the victim device is set in the *mid-level (*20% *to* 80%*)* in all of our evaluations. Note that, we force close all background third-party apps on the victim device while the remaining system services which provide fundamental functionality. In particular, our iPhone



(a) Battery level inference. (b) Inter-interface switch recognition.

Fig. 9: Effectiveness evaluation on coil whine based inference. OS: off screen; LS: lock screen; HS: home screen; AI: app interface.

11 uses two free apps from Apple's App Store to collect data, *i.e.*, Audio Recorder [5] (version 1.8.1) that uses the iPhone's microphone to record the coil whine, and Sensor Logger [7] (version 1.2.5) that utilizes the iPhone's magnetometer to record the magnetic field perturbations. In respect of data processing, we run all experiments on a desktop that runs Windows 10 with 32GB memory on an Intel i7-9700K CPU and an NVIDIA GeForce RTX 2080Ti GPU.

Datasets. We first build a mobile app dataset (D_{app}) by collecting 360 apps from Apple's App Store, which include the top 15 popular free apps in each app category (24 in total) based on statistics provided by *similarweb* [61] as of mid February 2022, since App Store does not provide such statistics. Based on D_{app}, we next build eight datasets to evaluate the effectiveness of WISERS in its every stage that are elaborated below.

B. Coil-Whine Based Inferences

Inferring battery levels. To evaluate the effectiveness of battery level inference, we build the dataset $D_{battery}$ by collecting coil whine traces from each of the three charging statuses identified in our wireless charger (shown in Figure 7 in §IV-A). Specifically, we put the iPhone on the wireless charger, turn off its screen, wait until the coil whine becomes stable, and collect one-second data. This procedure is repeated 50 times for each of the three cases (3×50 traces in total). Note that, since they are all stable coil whine data, we further divide them into 1,500 traces, each of which lasts 0.1 seconds, for data normalization (§IV-D), and split these traces into the training set and test set with the ratio of $8:2$.

Results. As shown in Figure 9a, the overall accuracy of battery level inference is 95.0%. Specifically, the low-level, mid-level, and high-level accuracy are 98.7%, 89.3%, and 97.0%, respectively. In particular, since the data traces collected for training and testing are well balanced, the relatively lower accuracy when inferring the battery at the low-level is due to the nature of wireless charging strategy and protocol. As shown in Figure 7 in §IV-A, it is less stable at this battery level than that in the other two intervals, which leads to more misclassifications.

Inferring inter-interface switches. To evaluate the effectiveness of recognizing inter-interface switches, we build the dataset D_{switch}. We also collect the coil whine traces of each type of interface for one second after it is stable for 50 rounds. Specifically, these coil whine traces are collected from *(i)* one testing case of the off screen and lock screen, respectively, *(ii)* six testing cases of the home screen, each of which shows a home screen displaying different lines of apps ranging from one to six lines excluding the dock, and 3) 24 testing cases

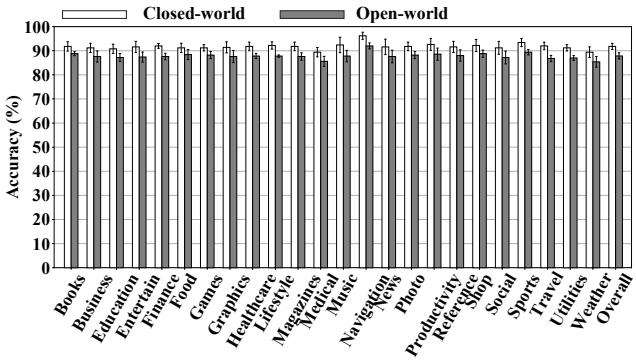
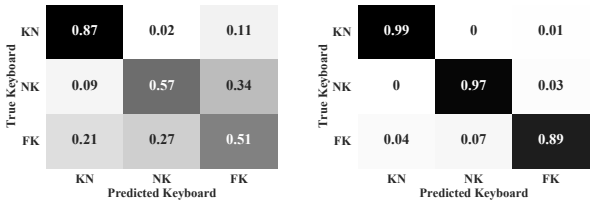


Fig. 10: Effectiveness evaluation on app recognition at launch.



(a) Without keystroke. (b) With one keystroke.

Fig. 11: Effectiveness evaluation of keyboard open: KN for keyboard not open, KO for keyboard open, NK for numeric-only keyboard, FK for full-size keyboard.

of app interfaces that are randomly picked from 24 apps in D_{app}, each of which is the most popular app in its category. Similarly, these traces are split into the training and test set with a ratio of $8:2$.

Results. Since the recognition of inter-interface switch depends on identifying the type of interfaces(§IV-B), we evaluate the accuracy of recognizing different types of interfaces. As shown in Figure 9b, the overall accuracy of interface type recognition is 92.5%. Specifically, the recognition accuracy for the off screen is 98.0%, lock screen is 92.0%, home screen is 82.0%, and app interface is 98.0%. The accuracy of the home screen is lower than other types of interfaces. After investigation, the main reason resides in the similarity between the home screen and the lock screen, where we use the same background that consumes the most power making these two interfaces appear similar in power consumption.

C. Magnetic-Field Based Recognition

Recognizing an app at launch. We build the dataset D_{applch} for this evaluation. Considering that apps on the smartphone are launched one by one, a static image will usually be displayed when an app is being launched, and different apps may vary in launching duration, we choose to collect the magnetometer readings for the first one second after clicking the app icon on the screen to represent the magnetic field perturbations during an app launch. Similar to collecting coil whine traces, each magnetic field perturbation trace also lasts for 0.1 seconds; each app in D_{app} will be repeated 100 times, resulting in 100 traces for each app launch. Since we have 360 apps, D_{applch} consists of 36,000 traces. We also use the 80/20 split to generate the training and test set.

Results. As shown in Figure 10, the effectiveness of recognizing an app at its launch is evaluated in both the closed-world and open-world settings defined. As mentioned in §IV-D, in both settings, we use the same dataset to train the model, and

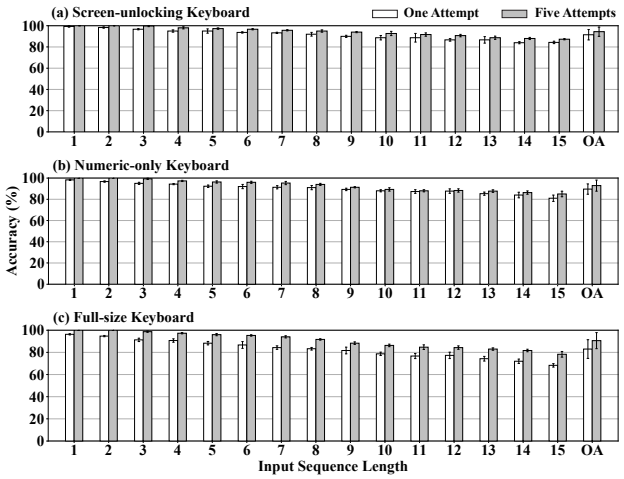


Fig. 12: Effectiveness evaluation on keystroke uncovering with different lengths from three keyboards (OA: Overall).

this dataset is built by randomly picking 80% of traces in 120 apps. Accordingly, to evaluate its effectiveness in the closed-world setting, the test set consists of the rest 20% traces in those 120 apps; and in the open-world setting, the test set includes *(i)* all traces in the rest 240 apps whose traces have not been used to train the model and *(ii)* the test set in the closed-world setting. Overall, the recognition accuracy in this closed-world setting is 91.8% with a standard deviation of 1.28%, and WISERS achieves an overall 87.9% recognition accuracy with a standard deviation of 1.27% in the open-world setting. The high accuracy and small standard deviation in both the closed-world and open-world settings indicate the consistency of WISERS performance across apps in different categories, and this consistency is also observed in the recognition accuracy shown per category in Figure 10.

Specifically, among 24 categories, WISERS performs the best in recognizing apps in "Navigation" (96.2% with 1.48% SDV) and worst in "Medical" (89.4% with 1.95% SDV) in the closed-world setting, and best in "Navigation" apps (92.0% with 1.22% SDV) and worst in "weather" (85.4% with 2.19% SDV) apps in the open-world setting. As such, WISERS can robustly and consistently recognize apps at app launch within 0.1 seconds in both closed-world and open-world settings.

Identifying keyboard open. This experiment involves two datasets (*i.e.*, D_{skopen} and D_{sktype}) to evaluate the effectiveness in identifying whether a keyboard is open and recognizing the type of the keyboard. In particular, these two datasets consist of data traces collected when opening the numeric-only keyboard and full-size keyboard in the same 24 apps that are used to build D_{switch}. Note that the screen-unlocking keyboard can only be opened in the lock screen. Specifically, to build D_{skopen}, we collect one-second data traces in a static interface and one-second data traces after the soft keyboard is open and becomes stable in the same interface, and the collection for each app is repeated 100 times. In respect of building D_{sktype}, we also collect one-second data traces in a static interface and one-second stable data traces after the soft keyboard is open with one keystroke, and this process is repeated 100 times for each app. The collected traces in both datasets will be normalized as 0.1 seconds trace (§IV-D); therefore, in total, each D_{skopen} and D_{sktype} has 2,400 traces, which are split into training and test set with the ratio of $8:2$.

Results. As shown in Figure 11a, while WISERS achieves a precision of 87.0% if there is no keyboard open, it cannot effectively distinguish between the numeric-only keyboard and full-size keyboard with less than 60% accuracy. However, if involving a keystroke, as shown in Figure 11b, it can successfully recognize whether a keyboard is open with 99.0% precision, the numeric-only keyboard with 97.0% accuracy, and the full-size keyboard with 89.0% correctness. Specifically, the main reason why its performance of recognizing the type of keyboard with and without a single keystroke varies is that these two keyboards almost occupy the same amount of area and consume similar amounts of power. Due to the different size of each key, one keystroke could result in an energy consumption burst that could be significant enough to separate these two keyboards.

Inferring keystrokes. This evaluation consists of three datasets for different keyboards: *(i)* screen-unlocking keyboard (D_{kbds}), *(ii)* system numeric-only keyboard (D_{kbdn}), and *(iii)* system full-size keyboard (D_{kbdf}). To build the training set, each key including the static key used for separating keystrokes is clicked 100 times, and each time forms a magnetic field perturbation trace which is normalized to 0.1 as described in §IV-B. To create the test set, we randomly generate a sequence of characters[1] for each keyboard ranging from one character to 15 characters in length, and each sequence generates three testing cases, each of which is repeated 100 times. For example, the three testing cases with one character of the full-size keyboard are "u", "a", and "n". Note that test cases of a keyboard include all its individual keys.

Results. Figure 12 shows the evaluation results on three different soft keyboards where keystroke length ranges from one to 15; the overall accuracy for them are 91.5%, 89.6%, and 83.0%, respectively, with only one guess attempt. The accuracy can increase to 91.5%, 89.6%, and 83.0%, respectively, within five attempts. At a high level, within five attempts, the uncovering success rate of all three keyboards reaches the highest when there is only one character and decreases as the length grows. In particular, within five attempts, WISERS can 100% correctly recover one-character keystroke in all three keyboards, while it achieves a precision of 87.3%, 85.0%, and 78.3% to uncover 15-character keystroke sequence from the screen unlocking keyboard, numeric-only keyboard, and full-size keyboard, respectively. Note that this accuracy is comparable to other works [22], which shows the ability of WISERS in accurate keystroke uncovering. Moreover, five attempts can significantly increase the accuracy in keystroke recovery than the one-time attempt, especially in recovering 15-character keystroke sequence on the full-size keyboard (10% increase from 68.3% to 78.3%).

VI. End-to-End Attacks

End-to-end attack scenarios. The end-to-end success rate is an important metric to evaluate the performance of an attack framework. Since components in WISERS are not independent of each other, we cannot simply multiply the accuracy rates of each component to have the final end-to-end success rate. Therefore, we conduct experiments on end-to-end attacks to obtain the success rate, where each end-to-end

[1]The full list of the testing cases in keystroke inference and more detailed experiment results of §VI (end-to-end attacks) are available at: https://github.com/WISERS-SP23/WISERS_Experiment_Details

TABLE I: End-to-end attack results. BL: battery level, OS: off screen, LS : lock screen, HS: home screen, AI: app interface, AR: app recognition, KO: keyboard opening, UK: unlock-screen keyboard, NK: numeric-only keyboard, and FK: full-size keyboard, PRE: prediction results, T1: one attempt, T5: five attempts.

# of Trial	Screen-unlocking Passcode								Cross-app Searching Content											App-specific Sensitive Inputs											
				Inter-interface Switch				PRE (✓/✗)			Inter-interface Switch				Intra-interface Activity				PRE (✓/✗)			Inter-interface Switch				App	Intra-interface Activity					PRE (✓/✗)	
	%	BL	Input	OS	LS	HS	AI	T1	T5	Input	OS	LS	HS	AI	KO	UK	NK	FK	T1	T5	Input	OS	LS	HS	AI		AR	KO	UK	NK	FK	T1	T5
1	64	M	0149	●	●	●	○	✓	✓	whats	○	○	●	○	●	○	○	●	✓	✓	hello world	○	○	●	●	WhatsApp	●	●	○	○	●	✓	✓
2	31	M	0975	●	●	●	○	✓	✓	whatsap	○	○	●	○	●	○	○	●	✓	✓	nice day	○	○	●	●		●	●	○	○	●	✓	✓
3	46	M	032918	●	●	●	○	✓	✓	what	○	○	●	○	●	○	○	●	✓	✓	never mind	○	○	●	●		●	●	○	○	●	✓	✓
4	58	M	310867	●	●	●	○	✓	✓	whatsa	○	○	●	○	●	○	○	●	✓	✓	its freezing	○	○	●	●		●	●	○	○	●	✗	✓
5	87	H	1642185	●	●	●	○	✓	✓	wha	○	○	●	○	●	○	○	●	✓	✓	i really appreciate it	○	○	●	●		●	●	○	○	●	✗	✓
6	54	M	1896	●	●	●	○	✓	✓	teleg	○	○	●	○	●	○	○	●	✓	✓	hello world	○	○	●	●	Telegram	●	●	○	○	●	✗	✓
7	41	M	8261	●	●	●	○	✓	✓	telegram	○	○	●	○	●	○	○	●	✓	✓	nice day	○	○	●	●		●	●	○	○	●	✓	✓
8	51	M	033496	●	●	●	○	✓	✓	tele	○	○	●	○	●	○	○	●	✓	✓	never mind	○	○	●	●		●	●	○	○	●	✓	✓
9	68	M	3179826	●	●	●	○	✓	✓	tel	○	○	●	○	●	○	○	●	✓	✓	its freezing	○	○	●	●		●	●	○	○	●	✓	✓
10	12	L	0123456789	●	●	●	○	✗	✓	telegra	○	○	●	○	●	○	○	●	✓	✓	i really appreciate it	○	○	●	●		●	●	○	○	●	✗	✓
11	65	M	2537	●	●	●	○	✓	✓	payp	○	○	●	○	●	○	○	●	✓	✓	nfawst@gmail.com	○	○	●	●	PayPal	●	●	○	○	●	✓	✓
12	47	M	129540	●	●	●	○	✓	✓	pay	○	○	●	○	●	○	○	●	✓	✓	jfdrgcd@gmail.com	○	○	●	●		●	●	○	○	●	✓	✓
13	90	H	482359	●	●	●	○	✓	✓	pal	○	○	●	○	●	○	○	●	✓	✓	sgjczpoe@gmail.com	○	○	●	●		●	●	○	○	●	✗	✓
14	31	M	4682319	●	●	●	○	✓	✓	paypal	○	○	●	○	●	○	○	●	✓	✓	mcarxbyn@gmail.com	○	○	●	●		●	●	○	○	●	✓	✓
15	85	H	0022446688	●	●	●	○	✗	✓	pa	○	○	●	○	●	○	○	●	✓	✓	oxmlwuyi@gmail.com	○	○	●	●		●	●	○	○	●	✗	✓
16	14	L	3671	●	●	●	○	✓	✓	venmo	○	○	●	○	●	○	○	●	✓	✓	nfawst@gmail.com	○	○	●	●	Venmo	●	●	○	○	●	✓	✓
17	82	H	9430	●	●	●	○	✓	✓	ven	○	○	●	○	●	○	○	●	✓	✓	jfdrgcd@gmail.com	○	○	●	●		●	●	○	○	●	✓	✓
18	46	M	185437	●	●	●	○	✓	✓	venm	○	○	●	○	●	○	○	●	✓	✓	mcarxbyn@gmail.com	○	○	●	●		●	●	○	○	●	✓	✓
19	32	M	7342	●	●	●	○	✓	✓	v	○	○	●	○	●	○	○	●	✓	✓	sgjczpoe@gmail.com	○	○	●	●		●	●	○	○	●	✗	✓
20	71	M	8413620	●	●	●	○	✓	✓	ve	○	○	●	○	●	○	○	●	✓	✓	oxmlwuyi@gmail.com	○	○	●	●		●	●	○	○	●	✗	✓
21	73	M	4869	●	●	●	○	✓	✓	chrom	○	○	●	○	●	○	○	●	✓	✓	www.google.com	○	○	●	●	Chrome	●	●	○	○	●	✓	✓
22	99	H	159628	●	●	●	○	✓	✓	chro	○	○	●	○	●	○	○	●	✓	✓	www.yahoo.com	○	○	●	●		●	●	○	○	●	✗	✓
23	44	M	694330	●	●	●	○	✓	✓	chr	○	○	●	○	●	○	○	●	✓	✓	www.youtube.com	○	○	●	●		●	●	○	○	●	✗	✓
24	45	M	47526401	●	●	●	○	✗	✓	chrome	○	○	●	○	●	○	○	●	✓	✓	www.amazon.com	○	○	●	●		●	●	○	○	●	✓	✓
25	55	M	976013672	●	●	●	○	✗	✓	ch	○	○	●	○	●	○	○	●	✓	✓	www.walmart.com	○	○	●	●		●	●	○	○	●	✓	✓
26	68	M	5198	●	●	●	○	✓	✓	safa	○	○	●	○	●	○	○	●	✓	✓	www.google.com	○	○	●	●	Safari	●	●	○	○	●	✗	✓
27	72	M	257813	●	●	●	○	✓	✓	safari	○	○	●	○	●	○	○	●	✓	✓	www.yahoo.com	○	○	●	●		●	●	○	○	●	✓	✓
28	88	H	751943	●	●	●	○	✓	✓	safar	○	○	●	○	●	○	○	●	✓	✓	www.youtube.com	○	○	●	●		●	●	○	○	●	✓	✓
29	17	L	78787878	●	●	●	○	✗	✓	saf	○	○	●	○	●	○	○	●	✓	✓	www.amazon.com	○	○	●	●		●	●	○	○	●	✓	✓
30	33	M	643185310	●	●	●	○	✗	✓	sa	○	○	●	○	●	○	○	●	✓	✓	www.walmart.com	○	○	●	●		●	●	○	○	●	✗	✓
31	13	L	6263	●	●	●	○	✓	✓	swiss	○	○	●	○	●	○	○	●	✓	✓	013468764189	○	○	●	●	SwissCovid	●	●	○	●	○	✓	✓
32	36	M	330522	●	●	●	○	✓	✓	swi	○	○	●	○	●	○	○	●	✓	✓	167983578654	○	○	●	●		●	●	○	●	○	✓	✓
33	87	H	462183	●	●	●	○	✓	✓	swis	○	○	●	○	●	○	○	●	✓	✓	296794641236	○	○	●	●		●	●	○	●	○	✗	✓
34	41	M	843250	●	●	●	○	✓	✓	swissc	○	○	●	○	●	○	○	●	✓	✓	358784645231	○	○	●	●		●	●	○	●	○	✓	✓
35	17	L	987474501	●	●	●	○	✗	✓	sw	○	○	●	○	●	○	○	●	✓	✓	431654651568	○	○	●	●		●	●	○	●	○	✗	✓
36	24	M	2360	●	●	●	○	✓	✓	lh	○	○	●	○	●	○	○	●	✓	✓	84532761	○	○	●	●	LHSafe	●	●	○	●	○	✓	✓
37	10	L	950718	●	●	●	○	✓	✓	lhsa	○	○	●	○	●	○	○	●	✓	✓	76831025	○	○	●	●		●	●	○	●	○	✓	✓
38	35	M	825134	●	●	●	○	✓	✓	lhs	○	○	●	○	●	○	○	●	✓	✓	68543102	○	○	●	●		●	●	○	●	○	✓	✓
39	60	M	5253	●	●	●	○	✓	✓	lhsaf	○	○	●	○	●	○	○	●	✓	✓	53681279	○	○	●	●		●	●	○	●	○	✓	✓
40	89	H	47654432	●	●	●	○	✗	✓	lhsafe	○	○	●	○	●	○	○	●	✓	✓	46531640	○	○	●	●		●	●	○	●	○	✗	✓

attack trial aims to infer every inter-interface switch and intra-interface activity in a series of user interactions starting from unlocking the screen with a passcode, conducting an across-app search at the home screen, and ending at launching an app and typing privacy-sensitive information. The end-to-end success rate can be calculated as *success rate = (number of success trials) / (number of all trials)*. In particular, we have prepared 40 screen unlocking passcodes that are randomly generated where there are 13 four-digit, 15 six-digit, and 12 custom length passcodes. In respect of cross-app searching keywords, we provide 40 different keywords particular to eight popular apps (five keywords of different lengths for each app): two chatting apps (WhatsApp and Telegram), two financial apps (Paypal and Venmo), two browsers (Chrome and Safari), and two Covid-19 apps (SwissCovid and LHSafe). Specifically, we prepared five arbitrary sentences to type in these two chatting apps, five randomly generated gmail addresses as user accounts to use in two financial apps, five popular website URLs to visit in two browsers, five randomly generated 12-digit Covid case serial numbers for SwissCovid, and five 8-digit mobile numbers for LHSafe. Note that these two Covid-19 apps pop up the numeric-only keyboard when asking for unique sensitive information. In total, there are 40 attack trials, and each trial involves one screen unlocking passcode, one cross-app search keyword, one app, and one app-specific user input.

End-to-end attack results. Table I presents the detailed results of our end-to-end attack. In this attack, WISERS achieves a 100% overall success rate in the 40 end-to-end attacks within at most five attempts in recovering user input without a single wrong character. In addition, even under the strictest standard where only one attempt is allowed to recover user input, more than half of all trials (*i.e.*, 22 out of 40) can still succeed without a single mispredicted character. Each failed case in the remaining trials is in the length of 14 on average, and each only contains *one* mispredicted character. The detailed analysis of each stage is elaborated in the following.

Revealing screen-unlocking passcode. WISERS reveals the screen unlocking passcode from the screen unlocking context, which is established from one intra-interface activity (*i.e.*, keystroke) and two sequences of inter-interface switches. One

sequence of switches starts from the off screen, next to the lock screen, and ends at a home screen, and the other one also starts from the off screen, then to the lock screen, but ends at an app interface. The after-lock screen switching is necessary for this context because this switching indicates the success of the screen unlocking and the correctness of the input passcode. As shown in Table I, WISERS has successfully inferred all inter-interfaces switches and recovered all passcodes within at most five attempts. In particular, in respect of failed ones if using only one attempt, there are four eight-digit, two nine-digit, and two ten-digit passcodes, and all eight failed attempts mispredict only one digit[1].

Revealing cross-app searching content. To reveal such cross-app searching content, we can define its context as inter-interface switches within home screens alongside intra-interface activities of keyboard open and keystroke. Table I show the summary of the attack results[1]. In particular, all inter-interface switches, keyboard opening, and searching content have successfully recognized and recovered without a single failure, achieving a 100% success rate.

Revealing app-specific sensitive inputs. The user interaction context in this procedure involves the interface switching from the home screen to an app interface alongside optional switches between different interfaces in the same app. In respect of intra-interface activities, it requires three activities for accurate uncovering, *i.e.*, app launch, keyboard open, and keystroke. As shown in Table I, all inter-interface switches and keyboard openings in these attempts are accurately recognized and identified[1]. In addition, WISERS needs more attempts to successfully recover a typing word, especially when there is a character in such a word appears consecutively, such as "ee" in the word "freezing", or the corresponding key of a character has relatively smaller space than the other keys, such as "." in the middle of "gmail.com". Similarly, within at most five attempts, all user chatting content is recovered.

VII. Impact Factors

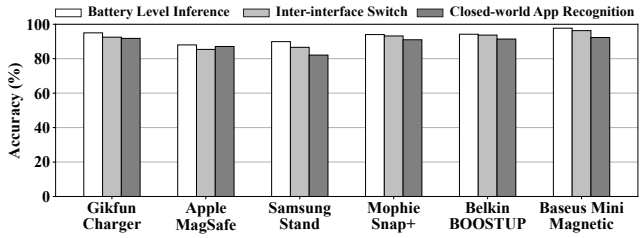
Impacts from different users. Considering previous studies have found that users spend different amounts of time for each keystroke [76], WISERS normalizes each keystroke trace to reduce such impacts on recovering keystrokes. To evaluate its performance, we have conducted an IRB-approved user study. In particular, we have recruited another four volunteers (two males and two females) to join this study in keystroke recovering analysis, and these volunteers were asked to conduct the same experiments following the same procedures when we build our keystroke evaluation dataset (*i.e.*, D_{kbds}, D_{kbdn}, and D_{kbdf}) in §V-A. By using the same classification model trained on these three datasets, as shown in Figure 17 in the Appendix, the accuracy rates on different lengths of sequential keystrokes show a similar trend that slightly decreases as the length grows, and the accuracy difference is within 8% at most between that of these volunteers and ourselves. *Therefore,* WISERS *is practical for cross-user attacks.*

Impacts from different wireless chargers. COTS wireless chargers could be manufactured in different qualities in terms of noise (*e.g.*, coil whine) cancellation and magnetic field shielding, which may make different degrees of impact on our coil whine and magnetic field perturbation-based analysis. To analyze such impacts, we evaluate WISERS on another five popular wireless chargers, *i.e.*, Apple MagSafe Charger (A2140), Samsung Wireless Charger Stand (EP-N5200TBEGGB), Mophie Snap+ 15W (SNP-WRLS-CHGR), Belkin BOOSTUP 7.5W (F7U054), and Baseus Mini Magnetic 15W (BS-W522), and compare the results with that obtained from 10 W Gikfun charger used in our previous effectiveness evaluation. Specifically, WISERS has trained new models with data collected from these five COTS products, and we follow the same procedure and strategy to collect data and evaluate the effectiveness in inferring battery level, recognizing inter-interface switches, and identifying app at launch in the closed-world setting as described in §V.

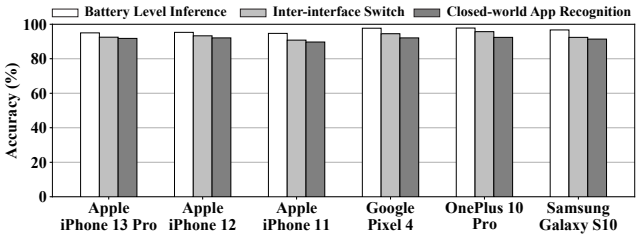
As shown in Figure 13a, WISERS can achieve high precision in all six evaluations across different wireless chargers. In particular, the accuracy of these chargers in battery level inference ranges from 89.9% to 98.0% (2.96% SDV); in inter-interface switch recognition ranges from 85.4% to 96.3% (4.31% SDV); and in the app recognition at launching with a closed-world setting ranges from 82.1% to 92.3% (3.98% SDV), respectively. The results also show that Apple MagSafe does a better job in reducing coil whine, and Samsung Wireless Charger Stand performs better in stabling magnetic field perturbation. *As such,* WISERS *can be applied to other wireless chargers and achieve similar performance.*

Impacts from different smartphone models. Smartphones could come with different energy consumption and management strategies at both the hardware and software levels, even if they are produced by the same manufacturer (*e.g.*, different iPhone models). To evaluate its impact, we follow the same procedures as before to evaluate six different iOS and Android smartphones, *i.e.*, iPhone 13 Pro, iPhone 12, iPhone 11, Google Pixel 4, OnePlus 10 Pro, and Samsung S10. The results in Figure 13b show that these six classification models of different smartphones achieve the accuracy from 94.7% to 97.8% (1.38% SDV) in battery level inference, 90.8% to 95.7% (1.72% SDV) in inter-interface switch, and 89.7% to 92.4% (0.98% SDV) in closed-world app recognition.

Additionally, we also evaluate the model transferability by training a classification model for each smartphone and applying each trained model on all six smartphones. As shown in Figure 14, if applying the classification model to the smartphone this model is trained from, these six models all achieve similar high accuracy ranging from 98.6% to 99.6%; however, if applying a model for other smartphones, the accuracy will decrease at different degrees. Within smartphones running the same OS, the accuracy of the classification model trained for iPhone 13 Pro and iPhone 12 slightly decreases by around 5%, while the model of iPhone 11 deceases roughly 10%. The differences may result from the different screen techniques used in them where iPhone 11 uses LCD while the other two use OLED. On the other hand, the accuracy of the model trained from the three Android smartphones decreases by around 14–23% because of their different keyboard layouts. Furthermore, the accuracy decreases over 30% as we transfer the model trained from an iOS smartphone to an Android smartphone because their screen techniques and UI layouts are extremely different as shown in Figure 19 in the Appendix. In short, though performance might decrease, WISERS can also work for cross-device attacks.



(a) Different wireless chargers.



(b) Different smartphones.

Fig. 13: Impact factor analysis of wireless chargers and smartphone models.

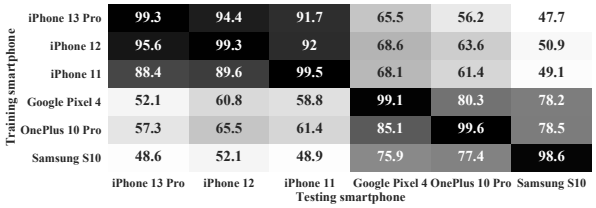
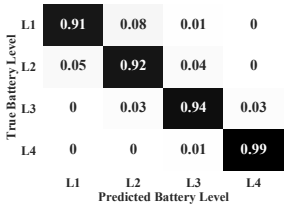


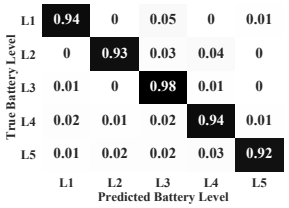
Fig. 14: Transferability of different smartphones.

Impacts from different battery levels. According to the wireless charging protocol, the same activity could show different patterns of coil whine and magnetic field perturbations if the smartphone contains different battery levels. We comprehensively investigate 26 commodity wireless chargers with their battery levels in the charging process [15], and there are 17 wireless chargers that have three levels or less (*e.g.*, Anker 10W charger), eight have four levels (*e.g.*, Mophie Snap+ 15W), and only Belkin 7.5W wireless charger has five levels (the full investigation results are presented in Table IV in the Appendix). Therefore, similar to the experiment in the battery level inference, we also evaluated WISERS in wireless chargers with more than three charging battery levels. As shown in Figure 15a and Figure 15b, WISERS achieves 94.0% and 94.2% accuracy in recognizing battery levels from Mophie Snap+ 15W (four levels) and Belkin 7.5W BOOSTUP (five levels), respectively. We also collected data at different battery levels from Belkin 7.5W BOOSTUP to train five models. As shown in Figure 15c, models trained from level one ($L1$) to level five ($L5$), achieve an accuracy ranging from 91.8% to 94.8% (1.27% SDV) in recognizing inter-interface switches, and an accuracy ranging from 90.3% to 92.3% (0.91% SDV) in app launching, respectively. *Therefore,* WISERS *could be used to launch attacks on different battery levels of a charger regardless of the number of levels a charger has.*

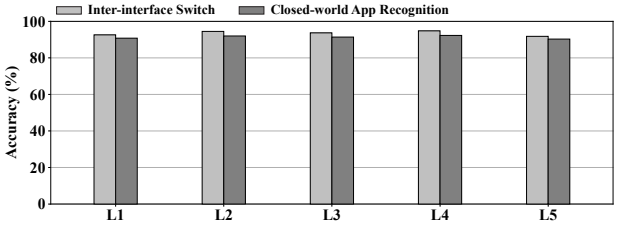
Analysis of other impact factors. In addition to analyzing the above impact factors, we also analyze some other factors, *i.e.*, app interfaces when typing on a keyboard, the distance between the wireless charger and the measuring device, device orientations, and the number of acoustic features. Our evaluation shows that WISERS is also practically resilient to these factors. In particular, recovering keystrokes is resilient to the cross-app impacts where there is only a slight 5% drop in the accuracy on average. Moreover, if increasing the attack distance from 8in (20cm) to 12in (30cm) and 16in (40cm), WISERS can still achieve the accuracy of 86.4% and 80.8% in using coil whine to infer inter-interface switches and 90.2% and 77.3% in leveraging magnetic field perturbations to recover keystrokes, respectively. In addition, WISERS achieves an average accuracy of 85.9% in recognizing inter-interface switches and 91.4% in keystroke recovering when the relative orientation between the charging smartphone and



(a) Mophie Snap+.



(b) Belkin BOOSTUP.



(c) Impact factor analysis of battery levels (Belkin BOOSTUP).

Fig. 15: (a) and (b): Battery level inference of wireless chargers with more than three levels. (c) Effectiveness at different battery levels of the Belkin BOOSTUP wireless charger.

the attack device is switching to 30°, 60°, and 90°. Moreover, we also evaluated the inter-interface switch recognition with a different number of acoustic features and the results show that it achieves accuracy rates of 83.9%, 91.4%, and 92.5% if using 39, 78, and 86 features, respectively. The full results are detailed in the Appendix.

VIII. Discussion

Ramification of our newly found side channel. Our newly discovered wireless charging side channel is orthogonal and complementary to other side-channel explorations (*e.g.*, optical side channel) in user privacy leakages, such as using a tiny camera. For example, tiny cameras may not work in certain circumstances (*e.g.*, insufficient ambient light) and could be easily detected and prevented, whereas our side channel is always present in any environment and much more difficult to be detected. Specifically, if using a tiny camera to capture user inputs, recent studies have demonstrated that tiny cameras can be easily detected by commodity smartphone sensors [57], [59] and apps [6], [27], which raise victims' suspicious, and this attack can be prevented by simply covering the touchscreen with hands [14] or using anti-peep screen protectors [63]. Note that, though our attacking devices could be smartphones that also come with cameras, our attack is less risky to be detected because we can put smartphones on the table by leaving their cameras face-down to avoid raising suspicions [58]. Moreover, our findings are expected to raise public awareness of the privacy leakages from wireless charging.

Limitations. WISERS is resilient to various impact factors but may not achieve the same level of performance as the at-

tack distance increases due to the inevitable signal attenuation of both coil whine and magnetic field. However, it is worth noting that close proximity attack scenarios exist in daily lives where adversaries can exploit this newly-discovered side channel to launch attacks. Additionally, WISERS's performance will decrease if directly using the model trained for one mobile OS to another, and we will tackle this issue in future work. Although WISERS does not take into account the magnetic interference from neighborhood devices, recent studies demonstrated that its effect only appears at a very close distance (*e.g.*, less than 1cm [42]). Similarly, the background apps might mislead WISERS under a few specific circumstances (*e.g.*, high run-sleep ratio [53]). We will evaluate and address them in future work. Moreover, the current prototype may not uncover sensitive user inputs if a system-level auto-filling mechanism exists. For example, the system may auto-fill the user input "ve" to "venmo". Using an AI-based word suggestion mechanism might address this challenge, which is listed as another future work. The current prototype also assumes the awareness of the distance and relative angle between the attacking device and the target charger. This limitation might be alleviated if tweaking existing solutions proposed to detect the distance and angle by using magnetometers [68] and additional sensors [30], such as the accelerometer and gyroscope, which have already been integrated into most commodity smartphones. Further investigation is another future work.

Countermeasures. To defend against this new side-channel attack, our proposed countermeasure is to protect the coil whine and magnetic field perturbations from being eavesdropped and exploited. To prevent eavesdropping, one approach is to use advanced materials to reduce noises (*i.e.*, coil whine) and shield the magnetic field to a certain degree, making them hard to be measured. Unfortunately, this approach may not be applied to existing products, and its effectiveness needs further investigation, let alone its cost–benefit [26]. Hence, we suggest complementing this approach with proactive protection methods. Specifically, since this side channel relies on the fact that a user interaction could be uniquely reflected in the coil whine and magnetic field perturbations, we can proactively introduce additional distortions to the inductive electromagnetic field by adding random noises (*e.g.* Gaussian white noise [75]) or dynamically switching the amplitude and frequency [78] of voltage in the primary coil. We will explore these methods in future work.

IX. Related Work

Wireless charging side channels. There are only a few studies towards understanding side channels in wireless charging, leaving this field largely unexploited. Existing works mostly attempted to conduct website fingerprinting attacks by collecting current traces in the power cable before power conversion in a wireless charger by putting a hidden coil in the proximity of 3.2cm [75], and demonstrated the potential of hijacking and eavesdropping attacks on the Qi wireless charging standard from the induced current traces in limited scenarios [34]. In addition, EM-Surfing [39] connected an external resistor to the power line of the wireless charger to monitor changes of the induced voltage for app and keystroke recognition. Unlike these attempts, WISERS is the *first contactless and context-aware* framework that leverages the emitted acoustic signal and changes in the ambient magnetic field during the process of smartphone wireless charging to infer a variety of user privacy-sensitive interactions with smartphones in general scenarios under much looser assumptions.

Power-based side channels on smartphones. There are many efforts towards exploring power-based side channels on smartphones. These attacks assume adversaries can either compromise victim smartphones (*e.g.*, installing a malware [19], [48], [64], [83]) or compromise the power cable (*e.g.*, USB cable) or power station to monitor power traces (*e.g.*, [23], [77]) to conduct app fingerprinting, location tracking, and privacy-sensitive information extraction. Different from these works, WISERS does not rely on the power profiles from the smartphone and also has no assumptions on victim device compromises.

EM side-channel attacks. Most works studying electromagnetic (*EM*) side channels have to use a special EM probe to collect EM signals to reverse engineer a neural network [9], monitor program execution [29], [49] from micro-controllers, extract secret keys [3], [4], [11], [25], recognize the security code from the touchscreen [42], and infer keystrokes [32] from smartphones. Other works use a smartphone as a probe to launch attacks, such as monitoring software launches in laptops [20]. These EM-based attacks are launched in a close proximity, *e.g.*, less than 1cm [3], [4], [9], [11], [25], [29], [42], [49], 2.5–5cm [20], 20–90cm [32]. Different from these works, WISERS leverages a new side channel and uses a COTS smartphone to uncover user interactions on another victim smartphone in a fine-grained and context-aware manner from similar proximity.

Other side-channel attacks on smartphones. Side-channel attacks on smartphones have been studied extensively in both iOS and Android platform. Some channels are studied from a similar perspective in both platforms (*e.g.*, cache side channels in Android [38], [82] and in iOS [22], [28]), while some are from a slightly different perspective. For example, magnetic side channels in iOS have been explored to monitor app activities [45] assuming pre-installed malicious apps and channels in Android are leveraged to finger movement trajectories [30], [41], [60] requiring additional equipment (*e.g.*, stylus). In addition, some side channels have only been studied in Android, such as using channels based on *procfs* to eavesdrop keystrokes, fingerprint webpages, and hijack UIs [18], [24], [31], [37], [54], [81]. Moreover, sensors in smartphones including accelerometers [8], [40], [52], gyroscopes [47], [67], and microphone [16], [43] have also been used to study associated side channels. Unlike these works, WISERS focuses on an entirely different and novel wireless charging side channel.

X. Conclusion

We introduce a new *contactless* and *context-aware* wireless charging side channel that utilizes the coil whine and the magnetic field perturbation stemming from the wireless charging process to infer sensitive user interactions on the charging smartphone. We have designed and implemented a three-stage attack framework, WISERS, to demonstrate the practicality of launching attacks using the newly discovered side channel. To the best of our knowledge, it is the *first* attack that makes use of signals in the physical world emitted during the wireless charging process to infer sensitive information from the smartphone. Our extensive evaluation suggests that WISERS is effective in inferring user interactions and resilient to a list of practical impact factors.

ACKNOWLEDGMENT

We sincerely thank our shepherd and all anonymous reviewers for their constructive feedback. This work was supported by CityU APRC grant 9610563, CityU SRG-Fd grant 7005853, NSFC (62101471), Hong Kong GRF (CityU 21201420/11201422), NSF of Shandong province (ZR2021LZH010), and by the Hong Kong ITF Project (GHP/052/19SZ) and the Research and Development Program of Shenzhen under Grants SGDX20190918101201696. Any opinions, findings, and conclusions in this paper are those of the authors and do not necessarily of supported organizations.

REFERENCES

[1] A. Adiga, M. Magimai, and C. S. Seelamantula, "Gammatone wavelet cepstral coefficients for robust speech recognition," in *Proceedings of the IEEE TENCON*, 2013.

[2] M. E. Ahmed, I.-Y. Kwak, J. H. Huh, I. Kim, T. Oh, and H. Kim, "Void: A fast and light voice liveness detection system," in *Proceedings of the 29th USENIX Security Symposium*, 2020, pp. 2685–2702.

[3] M. Alam, H. A. Khan, M. Dey, N. Sinha, R. Callan, A. Zajic, and M. Prvulovic, "One&Done: A Single-Decryption EM-Based attack on OpenSSL'sConstant-Time blinded RSA," in *Proceedings of the USENIX Security Symposium*, 2018, pp. 585–602.

[4] M. Alam, B. B. Yilmaz, F. Werner, N. Samwel, A. G. Zajic, D. Genkin, Y. Yarom, and M. Prvulovic, "Nonce@ Once: A single-trace EM side channel attack on several constant-time elliptic curve implementations in mobile platforms." in *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021, pp. 507–522.

[5] App Store, "Audio Recorder," https://apps.apple.com/us/app/audio-recorder-wav-m4a/id14544888.

[6] App Store, "Hidden camera detector," https://apps.apple.com/us/app/hidden-camera-detector/id532882360.

[7] App Store, "Sensor Logger," https://apps.apple.com/us/app/sensorlogger-csv-export/id15052035.

[8] A. J. Aviv, B. Sapp, M. Blaze, and J. M. Smith, "Practicality of accelerometer side channels on smartphones," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2012.

[9] L. Batina, S. Bhasin, D. Jap, and S. Picek, "CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel," in *Proceedings of the 28th USENIX Security Symposium*, 2019.

[10] A. Belahcen *et al.*, *Magnetoelasticity, magnetic forces and magnetostriction in electrical machines*. Helsinki University of Technology, 2004.

[11] P. Belgarric, P.-A. Fouque, G. Macario-Rat, and M. Tibouchi, "Side-channel analysis of weierstrass and koblitz curve ecdsa on android smartphones," in *Cryptographers' Track at the RSA Conference*. Springer, 2016, pp. 236–252.

[12] Business Standard, "one billion smartphones to have wireless charging globally by 2021 end," https://www.business-standard.com/article/technology/1-bn-smartphones-to-have-wireless-charging-globally-by-2021-end-121070300702_1.html.

[13] Y. Cai, Y. Zhao, X. Ding, and J. Fennelly, "Magnetometer basics for mobile phone applications," *Electron. Prod. (Garden City, New York)*, vol. 54, no. 2, 2012.

[14] M. Cardaioli, S. Cecconello, M. Conti, S. Milani, S. Picek, and E. Saraci, "Hand me your {PIN}! inferring {ATM} {PINs} of users typing with a covered hand," in *Proceedings of the 31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1687–1704.

[15] ChargingLab, https://www.chargerlab.com/.

[16] H. Chen, F. Li, W. Du, S. Yang, M. Conn, and Y. Wang, "Listen to your fingers: User authentication based on geometry biometrics of touch gesture," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, vol. 4, no. 3, pp. 1–23, 2020.

[17] J. Chen, P. Jönsson, M. Tamura, Z. Gu, B. Matsushita, and L. Eklundh, "A simple method for reconstructing a high-quality NDVI time-series data set based on the savitzky–golay filter," *Remote sensing of Environment*, vol. 91, no. 3-4, pp. 332–344, 2004.

[18] Q. A. Chen, Z. Qian, and Z. M. Mao, "Peeking into your app without actually seeing it: UI state inference and novel android attacks," in *Proceedings of the 23rd USENIX Security Symposium*, 2014.

[19] Y. Chen, X. Jin, J. Sun, R. Zhang, and Y. Zhang, "Powerful: Mobile app fingerprinting via power analysis," in *Proceedings of the International Conference on Computer Communications (INFOCOM)*, 2017.

[20] Y. Cheng, X. Ji, W. Xu, H. Pan, Z. Zhu, C.-W. You, Y.-C. Chen, and L. Qiu, "Magattack: Guessing application launching and operation via smartphone," in *Proceedings of the ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2019, pp. 283–294.

[21] Claire, "Should you be concerned if your wireless charger makes an unusual noise," https://global.ipitaka.com/blogs/news/should-you-be-concerned-if-your-wireless-charger-makes-an-unusual-noise?, 2020.

[22] P. Cronin, X. Gao, H. Wang, and C. Cotton, "An exploration of ARM system-level cache and GPU side channels," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2021.

[23] P. Cronin, X. Gao, C. Yang, and H. Wang, "Charger-surfing: Exploiting a power line side-channel for smartphone information leakage," in *Proceedings of the 30th USENIX Security Symposium*, 2021.

[24] W. Diao, X. Liu, Z. Li, and K. Zhang, "No pardon for the interruption: New inference attacks on Android through interrupt timing analysis," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2016, pp. 414–432.

[25] D. Genkin, L. Pachmanov, I. Pipman, E. Tromer, and Y. Yarom, "ECDSA key extraction from mobile devices via nonintrusive physical side channels," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016, pp. 1626–1638.

[26] T. Gluck, R. Puzis, Y. Oren, and A. Shabtai, "The curious case of the curious case: Detecting touchscreen events using a smartphone protective case," in *Proceedings of the IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2017.

[27] Google Play, "Glint finder - camera detector," https://play.google.com/store/apps/details?id=com.workshop512.glintfinder.

[28] G. Haas, S. Potluri, and A. Aysu, "itimed: Cache attacks on the apple a10 fusion SoC," *Cryptology ePrint Archive*, 2021.

[29] Y. Han, S. Etigowni, H. Liu, S. Zonouz, and A. Petropulu, "Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1095–1108.

[30] H. Huang, H. Chen, and S. Lin, "Magtrack: Enabling safe driving monitoring with wearable magnetics," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services*, 2019.

[31] S. Jana and V. Shmatikov, "Memento: Learning secrets from process footprints," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2012, pp. 143–157.

[32] W. Jin, S. Murali, H. Zhu, and M. Li, "Periscope: A keystroke inference attack using human coupled electromagnetic emanations," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021, pp. 700–714.

[33] D. M. Kreindler and C. J. Lumsden, "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*, 2016.

[34] A. S. La Cour, K. K. Afridi, and G. E. Suh, "Wireless charging power side-channel attacks," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021, pp. 651–665.

[35] J. Li, H. Zhou, S. Wu, X. Luo, T. Wang, X. Zhan, and X. Ma, "FOAP: Fine-grained open-world android app fingerprinting," in *Proceedings of the 31st USENIX Security Symposium*, 2022.

[36] M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "When CSI meets public WiFi: inferring your mobile phone password via WiFi signals," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016.

[37] C.-C. Lin, H. Li, X.-y. Zhou, and X. Wang, "Screenmilker: How to milk your android screen for secrets." in *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2014.

[38] M. Lipp, D. Gruss, R. Spreitzer, C. Maurice, and S. Mangard, "Armageddon: Cache attacks on mobile devices," in *Proceedings of the 25th USENIX Security Symposium*, 2016, pp. 549–564.

[39] J. Liu, X. Zou, L. Zhao, Y. Tao, S. Hu, J. Han, and K. Ren, "Privacy leakage in wireless charging," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[40] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When good becomes evil: Keystroke inference with smartwatch," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2015, pp. 1273–1285.

[41] Y. Liu, K. Huang, X. Song, B. Yang, and W. Gao, "Maghacker: eavesdropping on stylus pen writing via magnetic sensing from commodity mobile devices," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services*, 2020.

[42] Z. Liu, N. Samwel, L. Weissbart, Z. Zhao, D. Lauret, L. Batina, and M. Larson, "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*, 2020.

[43] L. Lu, J. Yu, Y. Chen, Y. Zhu, X. Xu, G. Xue, and M. Li, "Keylistener: Inferring keystrokes on QWERTY keyboard of touch screen through acoustic signals," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2019.

[44] S. Maruyama, S. Wakabayashi, and T. Mori, "Tap'n ghost: A compilation of novel attack techniques against smartphone touchscreens," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2019, pp. 620–637.

[45] N. Matyunin, Y. Wang, T. Arul, K. Kullmann, J. Szefer, and S. Katzenbeisser, "Magneticspy: Exploiting magnetometer in mobile devices for website and application fingerprinting," in *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*, 2019, pp. 135–149.

[46] S. McCandless, "An algorithm for automatic formant extraction using linear prediction spectra," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 22, no. 2, pp. 135–141, 1974.

[47] Y. Michalevsky, D. Boneh, and G. Nakibly, "Gyrophone: Recognizing speech from gyroscope signals," in *Proceedings of the 23rd USENIX Security Symposium*, 2014, pp. 1053–1067.

[48] Y. Michalevsky, A. Schulman, G. A. Veerapandian, D. Boneh, and G. Nakibly, "Powerspy: Location tracking using mobile device power analysis," in *Proceedings of the USENIX Security Symposium*, 2015.

[49] A. Nazari, N. Sehatbakhsh, M. Alam, A. Zajic, and M. Prvulovic, "Eddie: Em-based detection of deviations in program execution," in *Proceedings of the Annual International Symposium on Computer Architecture (ISCA)*, 2017, pp. 333–346.

[50] R. Ning, C. Wang, C. Xin, J. Li, and H. Wu, "Deepmag+: Sniffing mobile apps in magnetic field through deep learning," *Pervasive and Mobile Computing*, vol. 61, p. 101106, 2020.

[51] OpenATX, https://github.com/openatx/uiautomator2.

[52] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang, "Accessory: Password inference using accelerometers on smartphones," in *Proceedings of the Workshop on Mobile Computing Systems and Applications*, 2012.

[53] H. Pan, L. Yang, H. Li, C.-W. You, X. Ji, Y.-C. Chen, Z. Hu, and G. Xue, "Magthief: Stealing private app usage data on mobile devices via built-in magnetometer," in *Proceedings of the International Conference on Sensing, Communication, and Networking (SECON)*, 2021.

[54] Z. Qian, Z. M. Mao, and Y. Xie, "Collaborative TCP sequence number inference attack: how to crack sequence number under a second," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2012, pp. 593–604.

[55] O. Rukundo and H. Cao, "Nearest neighbor value interpolation," *arXiv preprint arXiv:1211.1768*, 2012.

[56] M. Sahidullah and G. Saha, "Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition," *Speech communication*, vol. 54, no. 4, pp. 543–565, 2012.

[57] S. Sami, S. R. X. Tan, B. Sun, and J. Han, "Lapd: Hidden spy camera detection using smartphone time-of-flight sensors," in *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2021, pp. 288–301.

[58] H. Shan, B. Zhang, Z. Zhan, D. Sullivan, S. Wang, and Y. Jin, "Invisible finger: Practical electromagnetic interference attack on touchscreen-based electronic devices," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, 2022, pp. 1548–1548.

[59] R. A. Sharma, E. Soltanaghaei, A. Rowe, and V. Sekar, "Lumos: Identifying and localizing diverse hidden IoT devices in an unfamiliar environment," in *Proceedings of the 31st USENIX Security Symposium*, 2022.

[60] C. Shen, S. Pei, T. Yu, and X. Guan, "On motion sensors as source for user input inference in smartphones," in *Proceedings of the International Conference on Identity, Security and Behavior Analysis*, 2015.

[61] Similarweb, "Top Apps Ranking," https://www.similarweb.com/apps/top/apple/store-rank/us/all/top-free/iphone/.

[62] P. Sirinam, N. Mathews, M. S. Rahman, and M. Wright, "Triplet fingerprinting: More practical and portable website fingerprinting with n-shot learning," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019, pp. 1131–1148.

[63] SNHDIGITAL, "3d tempered glass anti-peep privacy screen protector curved compatible with iphone." https://www.amazon.com/Tempered-Anti-Peep-Protector-Friendly-Compatible/dp/B07L43W8KW.

[64] R. Spolaor, L. Abudahi, V. Moonsamy, M. Conti, and R. Poovendran, "No free charge theorem: A covert channel via USB charging cable on mobile devices," in *International Conference on Applied Cryptography and Network Security*. Springer, 2017, pp. 83–102.

[65] M. Tan, J. Wan, Z. Zhou, and Z. Li, "Invisible probe: Timing attacks with PCIe congestion side-channel," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2021, pp. 322–338.

[66] D. Van Wageningen and T. Staring, "The Qi wireless power standard," in *Proceedings of 14th International Power Electronics and Motion Control Conference (EPE-PEMC)*. IEEE, 2010, pp. S15–25.

[67] H. Wang, T. T.-T. Lai, and R. Roy Choudhury, "Mole: Motion leaks through smartwatch sensors," in *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2015, pp. 155–166.

[68] M. Wang, Q. Luo, Y. Iravantchi, X. Chen, A. Sample, K. G. Shin, X. Tian, X. Wang, and D. Chen, "Automatic calibration of magnetic tracking," in *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking (MobiCom)*, 2022, pp. 391–404.

[69] T. Wang, "High precision open-world website fingerprinting," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2020.

[70] Y. Wang, H. Guo, and Q. Yan, "Ghosttalk: Interactive attack on smartphone voice system through power line," in *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2022.

[71] Wikipedia, "Electromagnetically induced acoustic noise," https://en.wikipedia.org/wiki/Electromagnetically_induced_acoustic_noise.

[72] Wikipedia, "Inductive charging," https://en.wikipedia.org/wiki/Inductive_charging.

[73] Wireless Power Consortium, "Download the Qi specifications," https://www.wirelesspowerconsortium.com/knowledge-base/specifications/download-the-qi-specifications.html.

[74] Wireless Power Consortium, "Qi - mobile computing," https://www.wirelesspowerconsortium.com/qi/.

[75] Y. Wu, Z. Li, N. Van Nostrand, and J. Liu, "Time to rethink the design of Qi standard? security and privacy vulnerability analysis of Qi wireless charging," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2021, pp. 916–929.

[76] B. Yang, R. Chen, K. Huang, J. Yang, and W. Gao, "Eavesdropping user credentials via GPU side channels on smartphones," in *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2022.

[77] Q. Yang, P. Gasti, G. Zhou, A. Farajidavar, and K. S. Balagani, "On inferring browsing activity on smartphones via USB power analysis side-channel," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 5, pp. 1056–1066, 2016.

[78] S. Yang, W. Wolf, N. Vijaykrishnan, D. N. Serpanos, and Y. Xie, "Power attack resistant cryptosystem design: A dynamic voltage and frequency switching approach," in *Design, Automation and Test in Europe*, 2005.

[79] X. Yang, "Efficient circular arc interpolation based on active tolerance control," *Computer-Aided Design*, vol. 34, no. 13, pp. 1037–1046, 2002.

[80] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017.

[81] K. Zhang and X. Wang, "Peeping tom in the neighborhood: Keystroke eavesdropping on multi-user systems." in *Proceedings of the USENIX Security Symposium*, vol. 20, 2009, p. 23.

[82] X. Zhang, Y. Xiao, and Y. Zhang, "Return-oriented flush-reload side channels on arm and their implications for Android devices," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016, pp. 858–870.

[83] X. Zhou, S. Demetriou, D. He, M. Naveed, X. Pan, X. Wang, C. A. Gunter, and K. Nahrstedt, "Identity, location, disease and more: Inferring your secrets from Android public resources," in *Proceedings of the ACM SIGSAC conference on Computer and Communications Security (CCS)*, 2013, pp. 1017–1028.

APPENDIX A
SUPPLEMENTARY OF PRINCIPLES AND ANALYSIS

The principle of no human involved activities. Smartphone activities that no human is involved in can produce magnetic field perturbations because of the load changes on the secondary coil when power-intensive activities such as screen animation and message notifications are running on the smartphone [39]. The load changes can be denoted as $\Delta R(t)$, and the corresponding changes of the current $\Delta I(t)$ and the induced electromagnetic field $\Delta\Phi(t)$ are shown in Equation 8. Therefore, $\Delta\Phi(t)$ results in the perturbations on the inductive electromagnetic field $\Phi_s(t)$.

$$\Delta I(t) = \frac{V_s(t)}{\Delta R(t)} \Rightarrow \Delta\Phi(t) = \frac{\mu_0 N_s \Delta I(t)}{2r_s} = \frac{\mu_0 N_s V_s(t)}{2r_s \Delta R(t)} \quad (8)$$

Finger-coupling experimental analysis. To uncover the relationship between the magnetic field perturbations and the finger-coupling effects in a key-pressing event, we conduct controlled experiments by utilizing the Android UiAutomator [51] to click the touchscreen with no human involved automatically. Specifically, we collect key-pressing data from the screen-unlocking keyboard of the OnePlus 10 Pro smartphone and compare the perturbation strengths with the human-touching data using the cumulative distribution function (CDF). Figure 16 shows the CDF results, and we know the perturbations of a finger-touching is usually stronger than an auto-clicking that only causes a screen animation. As automatic clicking is uncommon in real-world scenarios, WISERS can utilize the perturbations resulting from the finger-coupling effects to pinpoint the key-pressing.

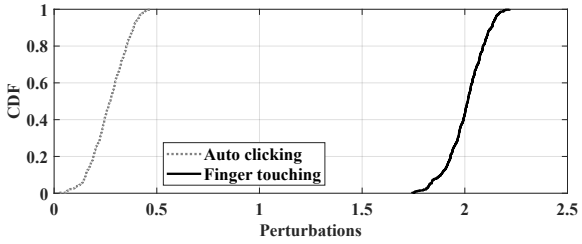


Fig. 16: CDFs of automatic clicking and human touching.

APPENDIX B
ADDITIONAL EVALUATION RESULTS OF IMPACT FACTORS

Keystrokes cross-app analysis. Typing and clicking on the same keyboard in different apps introduce slight differences in keystroke uncovering, as can be seen in Table II. That is because a keyboard usually takes a large area on the screen, generally at one-third to even half of the whole screen, which makes them dominate the power consumption at certain

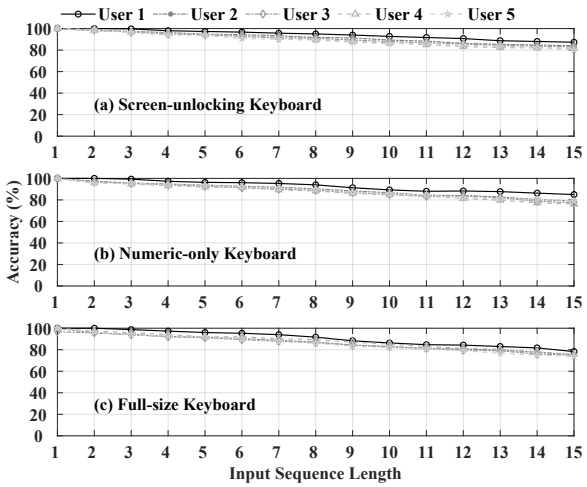


Fig. 17: Impact factor of users on uncovering keyboard inputs.

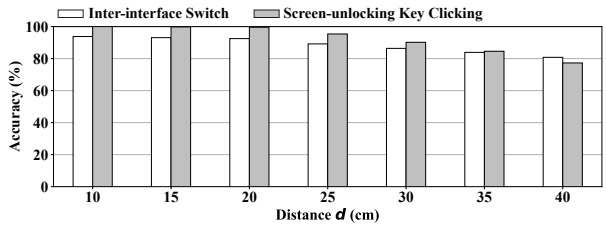
degrees. More importantly, when typing on the screen, normal content other than the keyboard is displayed statically, which introduces limited noises. In particular, in this evaluation, we use the same full-size system keyboard in three different apps (*i.e.*, WhatsApp, Telegram, and Messenger), following the same procedures with the same set of keystrokes ranging from one to 15 characters in length. Moreover, we use the data collected in each app to train a model and use this model to evaluate the accuracy in all three apps. As can be seen, the overall results maintain high accuracy in this cross-app evaluation. Specifically, if using the model trained for itself, all these three apps could achieve an accuracy of around 98%, *i.e.*, 98.6%, 98.3%, and 97.8% for WhatsApp, Telegram, and Messenger, respectively. In addition, if applying to the other two apps, a slight drop (around 5% on average) in the accuracy could be observed. As such, the impacts of different apps on soft keyboard keystroke uncovering are not significant.

TABLE II: Impact factor analysis of different apps on keyboard.

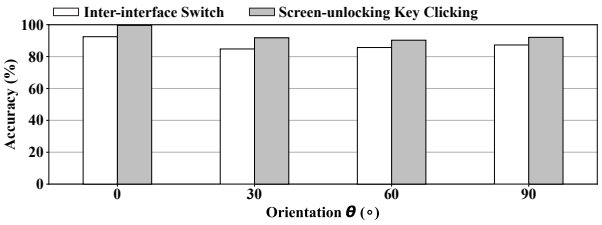
	Full-size Keyboard Key Clicking Accuracy (%)			
	Train\Test	WhatsApp	Telegram	Messenger
Apps	WhatsApp	98.6	93.5	95.1
	Telegram	91.7	98.3	92.9
	Messenger	94.5	93.3	97.8

Attack distance and placement orientation analysis. WISERS leverages the coil whine and magnetic field perturbation to launch attacks; however, their signals will attenuate proportional to the distance or qualitatively decrease in different orientations between the wireless charger and the measuring device. The decreased quality of measurable signals would negatively impact the performance of WISERS. To evaluate the impact of distance, we conduct two experiments on both physical phenomena by placing our measuring device (*i.e.*, a smartphone) from different distances to the wireless charger ranging from 4in (10cm) to 16in (40cm) and in different orientations of placement angles from 0° to 30°, 60°, and 90°. In particular, we follow the same procedures as described earlier in inter-interface switches and keystroke uncovering on the screen-unlock keyboard. As shown in Figure 18a, within 8in (20cm), the accuracy of using magnetic field perturbations to uncover a single key clicking remains 99.6% and then decreases to 77.3%. On the other hand, similarly, the accuracy of using coil whine to infer inter-interface switches starts to

decrease after 8in (20cm) but smoothly drops from 92.5% to 80.8% at 16in (40cm). Moreover, as shown in Figure 18b, the accuracy of recognizing inter-interface switch and keystroke recovering individually decreases by 5.2%–7.7% and 7.5%–9.3% at different placement orientations. Considering we are using a commercial smartphone to conduct this experiment, more sensitive and powerful measuring devices are believed to provide a much smoother accuracy decrease.



(a) Distance.



(b) Orientation.

Fig. 18: Impact analysis of distance and orientation.

APPENDIX C
OTHER EXPERIMENTAL DETAILS

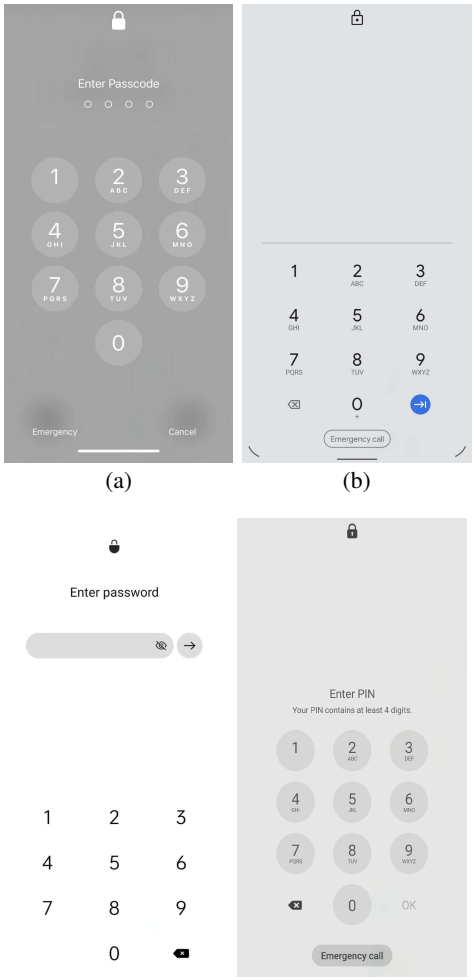
More experimental details, code, and datasets will be released and updated in the public GitHub repository: https://github.com/WISERS-SP23/WISERS_Experiment_Details.

TABLE III: MATLAB functions used in WISERS.

MATLAB function	Toolbox	Parameters
Butterworth filter	Signal Processing	High-pass, fc=3kHz, order=6
STFT	Signal Processing	Hann window=1024, Overlap=256
Audio features	Audio	Overlap length=256
Power spectral features	Signal Processing	Overlap length=256
Savitzky-Golay filter	Signal Processing	order=3, frame length=11

TABLE IV: Charging battery levels of commodity wireless chargers.

Charger	Level	Charger	Level
Gikfun Wireless Charger	3	EGGTRONIC MARBLE	3
Apple MagSafe	4	Apple MagSafe Duo	4
Samsung Charger Stand	4	Baseus Simple Magnetic	4
MOMAX Airbox	2	Meskex 3-in-1 Charger	4
MOMAX Q.MAG PRO 2	3	PYS 3-in-1 Charger	3
PYS BRANO	2	ZMI Wireless Charger	3
PYS MagSafe	4	Xiaomi 20W Charger	3
TESLA Portable	2	Huawei SuperCharge	4
IQOO FlashCharge	3	iWalk Wireless Charger	3
Benks Wireless Charger	3	TEGIC Charger	3
DX Magnetic	3	Mophie MagSafe	3
Mophie Charging Stand	3	Belkin 7.5W BOOSTUP	5
Mophie Snap+ 15W	4	Anker 10W Charger	3



(a) (b) (c) (d)

Fig. 19: Unlocking keyboard layout of different smartphones. (a): iPhone 13 Pro, iPhone 12, and iPhone 11; (b): Google Pixel 4; (c) OnePlus 10 Pro; (d) Samsung S10.