CITY UNIVERSITY OF HONG KONG
香港城市大學

Contactless Side Channels in Mobile Charging Systems: Attacks and Defenses
移動充電系統中的非接觸式側信道：攻擊與防禦

Submitted to
Department of Computer Science
電腦科學系
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
哲學博士學位

by

Ni Tao
倪濤

March 2024
二零二四年三月

Abstract

Recent years have witnessed the explosive development of mobile devices like smartphones and tablets. Along with the growing market of these mobile devices, the demand for novel and fast charging systems has gained much more popularity, *i.e.*, wireless chargers, wireless charging power banks, and multi-port chargers. However, these newly released mobile charging systems expose various vulnerabilities that can be leveraged by adversaries to launch different side-channel attacks, including eavesdropping attacks to violate user privacy and inaudible audio injection attacks to manipulate the voice assistants maliciously. This dissertation focuses on investigating contactless side channels present in different mobile charging systems. My research involves designing attack frameworks to demonstrate the feasibility of potential threats and proposing effective countermeasures to protect against these attacks. Specifically, I leverage two physical phenomena, namely coil whine and magnetic field perturbations, which occur during wireless charging processes. By utilizing these phenomena, I develop an attack framework capable of inferring fine-grained user-smartphone interactions while smartphones are being charged by commonly available wireless chargers. This framework enables us to uncover sensitive information, *i.e.*, unlocking passcodes, launching apps, and sensitive keystrokes. Furthermore, I extend our framework to target power banks that support wireless charging. I employ fast domain adaptation techniques, such as few-shot learning, to enhance the transferability of the attack across various scenarios, including different power banks, smartphones, and battery levels. Additionally, I conduct an in-depth study to explore the security of multi-port chargers, a new type of charging accessory that can charge multiple mobile devices simultaneously. Our research reveals that voltage leakages across neighboring USB ports can compromise user privacy by disclosing their activities on charging mobile devices. In the case of multi-port chargers equipped with USB-C interfaces, I demonstrate that the audio pins of the USB-C port can be exploited to silently activate voice assistants, *i.e.*, Apple Siri, Google Assistant, and OnePlus Breeno, and inject modulated voice commands. To mitigate the risks posed by these attacks, I propose and implement

countermeasures from both hardware and software perspectives. These measures aim to safeguard users from the aforementioned vulnerabilities and protect their privacy and security. Overall, this dissertation provides a thorough examination of contactless side-channel attacks and defenses in order to raise awareness about the potential threats associated with widely used commodity mobile charging systems.

CITY UNIVERSITY OF HONG KONG
Qualifying Panel and Examination Panel

Surname:	NI
First Name:	Tao
Degree:	Doctor of Philosophy
College/Department:	Department of Computer Science

The Qualifying Panel of the above student is composed of:

Supervisor(s)

Prof. XU Weitao	Department of Computer Science City University of Hong Kong

Co-supervisor(s)

Prof. ZHAO Qingchuan	Department of Computer Science City University of Hong Kong

Qualifying Panel Member(s)

Prof. WANG Cong	Department of Computer Science City University of Hong Kong
Prof. LI Zhenjiang	Department of Computer Science City University of Hong Kong

This thesis has been examined and approved by the following examiners:

Prof. XU Weitao	Department of Computer Science City University of Hong Kong
Prof. WANG Cong	Department of Computer Science City University of Hong Kong
Prof. WANG Shiqi	Department of Computer Science City University of Hong Kong
Prof. SHEN Yiran	School of Software Shandong University

Acknowledgements

I want to express my deepest gratitude to my two advisors, Prof. Weitao Xu and Prof. Qingchuan Zhao. Their invaluable guidance has been instrumental in shaping my professional growth as a researcher. Under their wings, they have nurtured my development and shared their unwavering optimism in the face of research challenges. Their generous support and the freedom they afforded me to pursue my research interests have been remarkable during my Ph.D. journey.

I am deeply grateful to Prof. Cong Wang, Prof. Zhenjiang Li, Prof. Yiran Shen, Prof. Kehuan Zhang, and Prof. Shiqi Wang for being my qualifying panel on the thesis committee and giving me valuable comments and suggestions. I would like to thank my collaborators and mentors. I am very grateful to Prof. Xiaokuan Zhang, Prof. Daniel Xiapu Luo, Dr. Chaoshun Zuo, Dr. Lei Xue, and Dr. Jianfeng Li, who have collaborated with me since the start of my dissertation research in side-channel analysis, for their dedication to our collaborative research projects and the enthusiasm they have exemplified to me in pursuing outstanding academic achievements. I also want to thank Prof. Tao Gu, Dr. Guohao Lan, and Dr. Zhenyu Yan for sharing their knowledge and passion with me in our collaborative research in mobile computing. I appreciate all my student collaborators, including Yongliang Chen, Zehua Sun, Mingda Han, Huanqi Yang, Di Duan, Keqi Song, and other teammates. I am also grateful to my friend Dr. Guowen Xu, who gave me illuminating suggestions for research and life.

My special thanks go to my best friend, Sherry Yixin Tu, who has supported me with her joyful spirit, optimistic soul, and delightful sense of humor. I also appreciate my other friends, Tongwei Zhu, Siying Xu, Dr. Anqi Hu, Mario Hou, and other talented young scholars I met at conferences for the joyful and memorable moments.

I extend my heartfelt gratitude to my beloved Sabrina, whose unwavering sincerity, kindness, and optimism have continually brightened my life and warmed me with her enduring companionship. At last, I want to thank my dearest parents, who raised me to be a man of integrity and determination, understood my struggles, and continuously encouraged and supported me unconditionally to pursue my dreams.

Table of contents

Chapter 1

Introduction

The past decade has seen an unprecedented surge in the use of mobile devices, such as smartphones and tablets, enriching our daily lives with millions of applications worldwide. However, the diverse functionalities of these devices have significantly increased power demands, highlighting battery capacity as a critical limitation and spurring the development of various innovative charging methods and accessories. Diverging from traditional mobile charging methods that rely on USB cables for a physical connection, wireless charging has emerged as the most popular solution for modern smartphones. This shift can be attributed to two key factors: *(i) Generalizability:* Wireless chargers offer a more universal charging solution as they can charge smartphones across different brands through electromagnetic induction. This universality overcomes the limitations posed by various charging ports, such as lightning ports in iOS devices and micro-USB/USB-C ports in Android smartphones. *(ii) Reliability:* Previous research has demonstrated that USB connections can potentially expose users to privacy breaches and even enable attackers to inject malicious software. Wireless chargers, by contrast, eliminate the need for a physical connection to the smartphone, thereby safeguarding against potential side-channel attacks that exploit USB cables.

Unfortunately, recent studies [1–3] have shown that wireless charging is susceptible to a range of side-channel attacks, which could lead to the privacy compromise of smartphones being charged. However, the attack models proposed in these studies have practical limitations: either they necessitate tampering with the cables of wireless chargers to access power traces (*e.g.,* current/voltage) [2] or they require specific conditions such as a battery level higher than 80% [1], rendering these threat models somewhat impractical. This raises a critical question: *Is it feasible to conduct side-channel attacks on various mobile charging systems, such as wireless chargers, in a*

contactless manner? In other words, an alternative approach to explore, rather than compromising wireless chargers to monitor power traces, would be to utilize the signals emitted during the wireless charging process to deduce the user's interactions with the charging smartphone. Furthermore, I believe that investigating *contactless side channels* in different mobile charging systems (*e.g.,* wireless chargers [4], power banks [5], and multi-port chargers [6]) across various scenarios opens up new research avenues. These investigations can contribute to the development of effective defense strategies, thereby enriching the field of security research.

In this dissertation, I delve into the realm of *contactless side channels* across three innovative mobile charging platforms: *wireless chargers*, *power banks*, and *multi-port chargers*. At a high level, my approach involves capturing and analyzing the physical signals emitted during the charging process. I then employ advanced deep neural networks to decode both the coarse-grained contexts (*e.g.*, attack triggering, device fingerprinting, and user interface switches) and the fine-grained user interactions (*e.g.*, unlocking passcodes, app launches, and sensitive keystrokes) on the charging smartphone. The contactless side channels I have uncovered exhibit distinct characteristics, which can be categorized into three primary aspects:

- **Non-intrusiveness.** In this thesis, I have identified and analyzed two physical phenomena present in the wireless charging process: *coil whine* and *magnetic field perturbations*. These can be non-intrusively captured using the microphones and magnetometers found in commodity smartphones or small-size attack devices. I also revealed voltage leakages across adjacent USB ports in multi-port chargers, which allows adversaries to eavesdrop on user privacy without physical connections.
- **Practical attack scenarios.** Compared to previous studies that necessitated compromising wireless chargers or power banks for attacks, my research reveals contactless side channels that are more practical in real-life scenarios. This raises important concerns about the security of wireless charging stations and multi-port chargers in public areas. The vulnerabilities we've identified exist in common mobile charging platforms and could be exploited to maliciously violate user privacy.
- **Fine-grained user privacy inference and inaudible injection attacks.** I have shown that contactless side channels in the wireless charging process can be exploited to deduce detailed user information from the charging smartphone in a context-sensitive manner. Additionally, side channels in USB-C audio transmission can be leveraged to discreetly activate the voice assistant of the charging smartphone and inject malicious voice commands inaudibly.

In a nutshell, my work validates the feasibility of exploiting these side channels to compromise user privacy and inject voice commands in a contactless manner. I then provide a concise overview of each contactless side channel I identified in three mobile charging systems, followed by the organization of this dissertation.

1.1 Proposed Contactless Side Channels in Mobile Charging Systems

1.1.1 Contactless Side Channels in Wireless Chargers

There has been a notable surge in smartphones equipped with wireless charging capabilities, including the iPhone series and most Android devices in recent years. A recent survey forecasts that by 2030, the global market for wireless charging will surpass 129 billion [7]. Known as inductive charging, wireless charging employs electromagnetic induction to transfer power from a charger to a smartphone. Predominantly, most smartphones utilize the Qi charging protocol [8], a globally recognized wireless charging standard developed by the *Wireless Power Consortium* (WPC). Qi-certified wireless chargers harness electromagnetic induction to deliver power ranging from 5 to 15 watts, effectively charging smartphones without needing physical connectors. Despite their popularity and convenience, wireless chargers' security and privacy issues remain uninvestigated. Existing works [1–3] reveals that the power traces (*e.g.*, inductive current/voltage) could leak user interactions on the smartphone being wirelessly charged, while usually requiring to compromise wireless chargers physically.

In §2, I 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. I have developed and executed a novel three-stage attack framework, which I refer to as `WISERS`, to illustrate the viability of this newly identified side channel. `WISERS` initially captures both the coil whine and magnetic field perturbations emitted by the wireless charger. Subsequently, it deduces *(i)* inter-interface transitions (e.g., switching from the home screen to an app interface) and *(ii)* intra-interface activities (e.g., keyboard inputs within an app) to construct *user interaction contexts*. This process ultimately uncovers sensitive information. The efficacy of `WISERS` has been rigorously tested with a range of popular smartphones and commercially available off-the-shelf (COTS) wireless chargers, demonstrating its po-

tential impact in real-world scenarios. In particular, `WISERS` can achieve over 90.4% accuracy in inferring sensitive information, such as screen-unlocking passcode and app launch. In addition, I also show that `WISERS` is resilient to a list of impact factors, including different wireless chargers, smartphones, and battery levels.

1.1.2 Contactless Side Channels in Wireless Charging Power Banks

Power banks have increasingly become essential devices for individuals needing to charge their smartphones on the go, particularly when facing low battery levels. Recent studies forecast that by the end of 2030, the global market for power banks will surpass 31 billion [9]. Meanwhile, a growing number of newly-released power banks have begun to support wireless charging. This shift towards wireless charging is prompting smartphone manufacturers to integrate wireless charging modules into their designs and to move away from traditional wired charging methods. This transition is partly driven by the desire to mitigate the security risks associated with USB cables, as highlighted by previous research [10–12, 1, 2]. Although these wireless charging power banks appear to be immune to most reported vulnerabilities in either power banks or wireless charging, I find that the same contactless side channels proposed in §2 also exist in wireless charging power banks that leak user privacy from their charging smartphones without compromising either power banks or victim smartphones.

In §3, I introduced `BankSnoop` to demonstrate the feasibility of exploiting the wireless charging side channel discovered in power banks. `BankSnoop` capitalizes on the coil whine and magnetic field disturbances generated by a power bank during wireless charging of a smartphone. It employs few-shot learning techniques to accurately identify the app running on the smartphone and to decode keystrokes. I assessed the efficacy of `BankSnoop` using standard wireless charging power banks and smartphones. The results indicate a remarkable average accuracy exceeding 90% in recognizing app launches and keystrokes. Additionally, `BankSnoop` showcases notable adaptability across different smartphone models and power banks, maintaining over 85% accuracy in 10-shot learning scenarios. By combining these contactless side channels discovered in the wireless charging process, I want to raise public awareness of the threats in these new mobile charging platforms.

1.1.3 Contactless Side Channels in Multi-Port Chargers

My exploration into contactless side channels in wireless chargers and power banks has further led us to examine other emerging mobile charging accessories. Multi-

port chargers, capable of simultaneously charging multiple mobile devices like smartphones, have skyrocketed in popularity, with millions of units sold in recent years. Yet, this feature, designed for simultaneous charging, could inadvertently introduce security and privacy vulnerabilities. These arise from the interconnected features of the devices being charged; if not meticulously designed and implemented, one device could interact with another during the charging process. Unfortunately, these potential risks have not been extensively studied. My research has uncovered a novel attack surface within the circuit design of multi-port chargers. This vulnerability could allow an adversary to exploit one port to *(i)* surreptitiously monitor the activities of other devices being charged and *(ii)* covertly inject malicious audio commands, particularly if the charging device is equipped with voice assistant capabilities and a USB-C interface.

In § 4, I have developed and implemented **`XPorter`**, an innovative framework specifically designed to analyze and demonstrate the security and privacy vulnerabilities inherent in multi-port chargers. **`XPorter`** ingeniously utilizes fluctuations in voltage signals from adjacent ports to monitor changes induced by user activities on the charging port. This allows for the identification of active apps and the detection of keystrokes on the connected device. In addition, **`XPorter`** is capable of conducting inaudible audio injection attacks from a neighboring port to the charging device through the USB-C interface. I conducted a thorough evaluation of **`XPorter`**'s effectiveness using five commonly used multi-port chargers and five different mobile devices. The results were impressive, showcasing a high degree of accuracy in recognizing the launch of 20 mobile apps (88.7%) and in decoding unlocking passcodes (98.8%). Moreover, **`XPorter`** demonstrated a 100% success rate in executing inaudible audio injection attacks on three different voice assistants. My study also revealed that **`XPorter`** is robust against various influencing factors and highlighted its potential to target multiple victims simultaneously.

1.2 Organization

In the rest of this dissertation, I expand on the details of each contactless side channel in three mobile charging systems. I first discuss *Contactless Side Channels in Wireless Chargers* in § 2. Then, I further investigate *Contactless Side Channels in Wireless Charging Power Banks* in § 3. After that, I explore *Contactless Side Channels in Multi-Port Chargers* in § 4. Finally, I discuss and identify potential vulnerabilities and research opportunities of this research area in future charging systems in § 5, and summarize the contributions as well as conclude this dissertation in § 6.

Chapter 2

Contactless Side Channels in Wireless Chargers

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 chapter, 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, we also show that `WISERS` is resilient to a list of impact factors.

2.1 Introduction

Recent years have witnessed the advance of wireless charging technology for smartphones. Wireless charging standards, *e.g.*, Qi [8] 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 [7].

In this chapter, 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 [13–15, 12] and *wireless* charging [1, 3] 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 [16–19]), 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 [1].

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 [8]). The current changes in the internal coil of the charger will incur electromagnetic forces, based on Ampere's force law, that slightly deform and vibrate 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 fea-

tures (*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 effectiveness 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 (HUMAN-2023-0016-2). 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 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.

2.2 Preliminary

2.2.1 Wireless Charging on Smartphones and Qi Standard

Qi standard [20] is a wireless charging standard widely supported by smartphones [1]. Wireless chargers holding Qi certification could leverage electromagnetic induction to charge smartphones by providing 5-15 watts of power [21]. An illustration of this wireless charging process is presented in Fig. 2.1. 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 [8]. Activities using more power make the smartphone request more power supply from the wireless charger [8, 3]. This charging

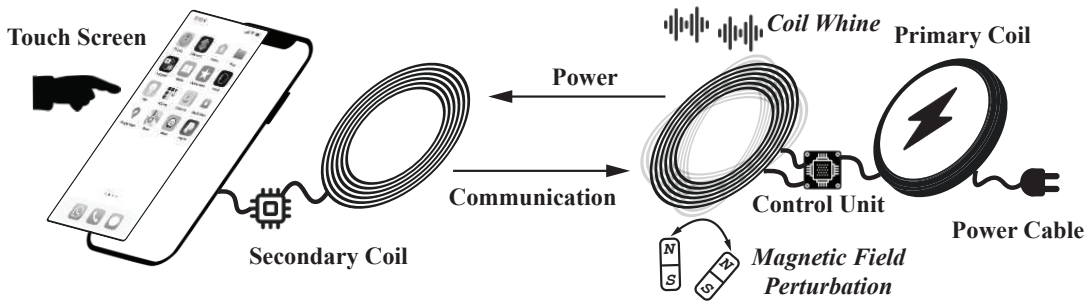


Fig. 2.1 Wireless charging principle.

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.

2.2.2 Physical Phenomena Generated by Wireless Charging

Since a charging system under the Qi standard [20] 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 [22]. 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 Fig. 2.1, 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 [23]. Coil whine can be in different frequency ranges, making it either human audible (between 20 Hz and 20 kHz) or inaudible [24].

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.

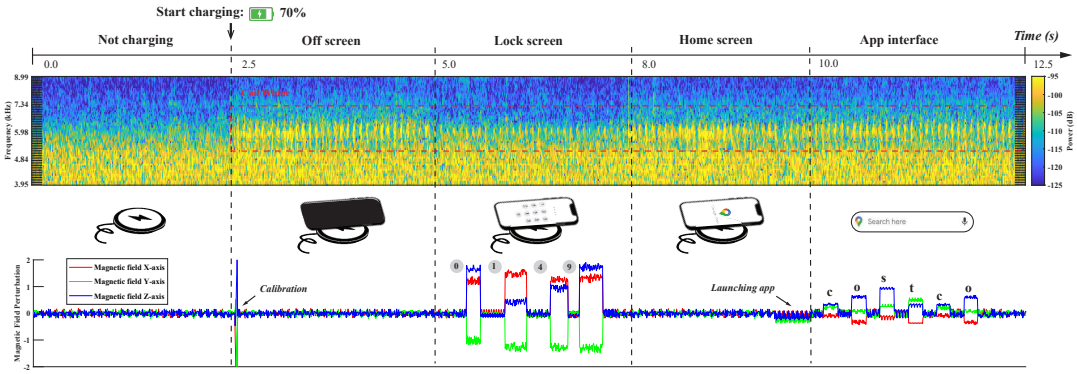


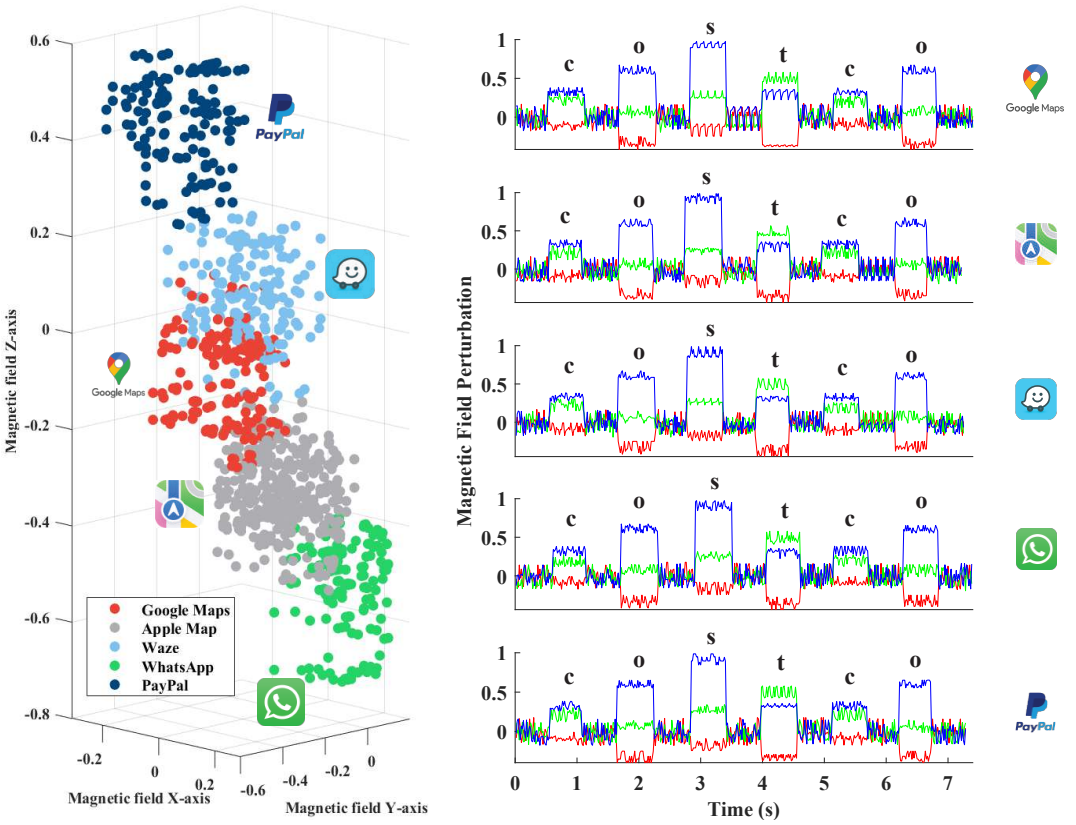
Fig. 2.2 Motivating example scenario: a user places a smartphone with 70% battery left on a wireless charger, unlocks the screen with the passcode (*i.e.*, 0149), clicks app icon to open Google Map, and types "costco" to search for nearby supermarket locations. Upper Figure: the corresponding power spectrum of the coil whine; Lower Figure: the strength and directions in three dimensions of the ambient magnetic field.

2.3 Motivation, Principle, and Threat Model

2.3.1 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 §2.2.1, 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[25] 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 Fig. 2.2. The middle part of Fig. 2.2 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 Fig. 2.2, we calibrate the ambient mag-



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

Fig. 2.3 Magnetic field perturbation in five different apps.

netic 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 Fig. 2.3. Specifically, Fig. 2.3a presents the magnetic field perturbation resulting from the first five seconds after launching different apps, and Fig. 2.3b 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 mag-

netic 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.

2.3.2 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 2.1). The inductive electromagnetic field produces an induced voltage $V_s(t)$ to power the smartphone following Faraday's law (Equation 2.2).

$$\Phi_s(t) = \eta\Phi_p(t) = \eta\frac{\mu_0 N_p I_p(t)}{2r_p}, \tag{2.1}$$

$$V_s(t) = N_s\frac{\Delta\Phi_s(t)}{\Delta t} = \eta\frac{N_s}{N_p}\cdot\frac{\mu_0\Delta I_p(t)}{2r_s\Delta t}, \tag{2.2}$$

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 2.3).

$$\Delta F_p(t) = \Delta I_p(t) L_p \times \Phi_p(t) \tag{2.3}$$

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.

$$\Delta F_p(t) \Rightarrow \Delta\mathcal{S}(A_{cw}, f_{cw}) \tag{2.4}$$

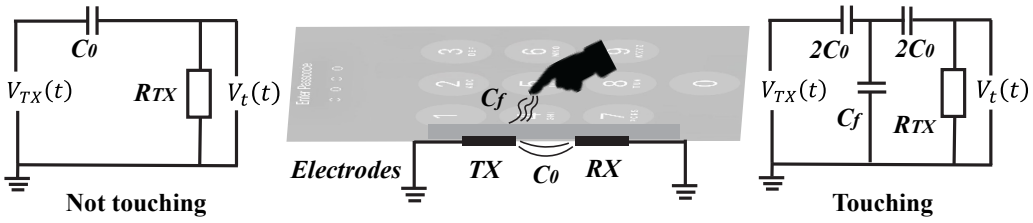


Fig. 2.4 Illustration of finger-coupling effects in a touching event.

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 [8] and the finger-coupling effects [26] 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 Fig. 2.4, 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 2.5), 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.

$$\begin{cases} V_t(t) = V_{TX}(t) \cdot \frac{R_{TX}}{R_{TX}+1/(j2\pi f C_0)} \text{ (Not touching)} \\ V_t(t) = V_{TX}(t) \cdot \frac{R_{TX}}{R_{TX}+1/(j4\pi f C_0)+\Delta Z_f(t)} \text{ (Touching)} \\ \Delta Z_f(t) = 1/(\frac{1}{1/(j2\pi f \Delta C_f(t))} + \frac{1}{1/(j4\pi f C_0)}) \text{ (Impedance)} \end{cases} \tag{2.5}$$

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 2.6) finally produce the perturbations on the inductive electromagnetic field, $\Phi_s(t)$.

$$\Delta I(t) = \frac{V_s(t) + \Delta V_t(t)}{\Delta R(t)} \Rightarrow \Delta\Phi(t) = \frac{\mu_0 N_s \Delta I(t)}{2r_s} \tag{2.6}$$

(a) Cafe (b) Airport

Fig. 2.5 Public wireless charging facility examples.

2.3.3 Threat Model

We consider a common scenario in wireless charging side-channel attacks [1, 3] 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 (Fig. 2.5). 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 [27], and their microphones are sensitive enough with a sampling rate of 44.1 kHz -48 kHz [28] to capture most coil whine generated in charging a smartphone with *commercial-off-the-shelf* (COTS) wireless chargers (*e.g.*, Apple MagSafe Wireless Charger). Additionally, placing this monitoring device in close proximity [29, 30] could also be achieved in public facilities, as shown in the examples of Fig. 2.5.

Moreover, while assuming the adversary can observe the type and initial orientations of 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 and voltage 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.

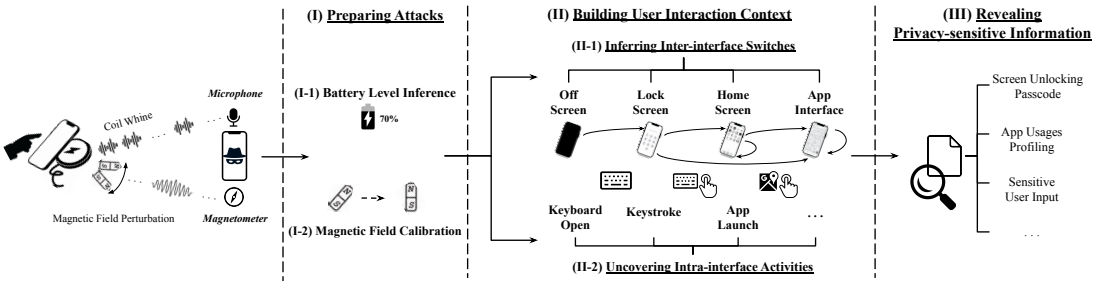


Fig. 2.6 Overview of WISERS.

2.4 Attack Framework

This section presents the details of our proposed three-stage attack framework, WISERS. As shown in Fig. 2.6, *(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 (§ 2.4.1); *(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 (§2.4.2); and *(iii)* the last stage is to utilize the established user interaction context to uncover user privacy (§2.4.3). The implementation of the WISERS prototype is detailed in §2.4.4.

2.4.1 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 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

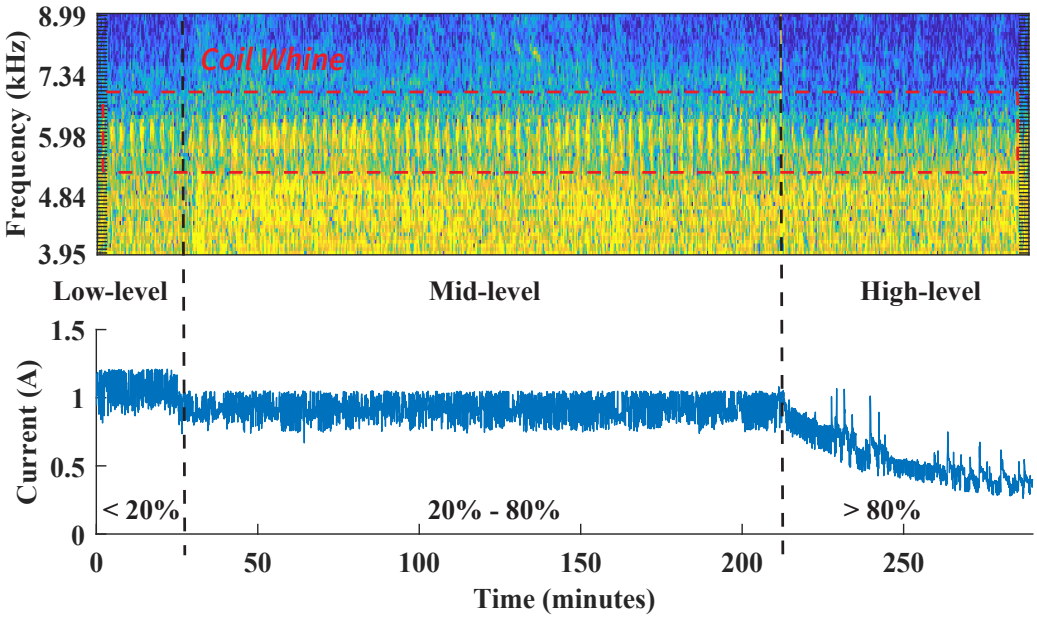


Fig. 2.7 Power spectrum of coil whine from different battery levels.

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 [8]. 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 Fig. 2.7, 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 §2.3.2, 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 Fig. 2.8, 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 [31] to calibrate the coordinates in the X-Y plane using the Equation 2.7, 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.

$$\begin{cases} x_1 = x_2 - d(1 - cos\theta) \\ y_1 = y_2 - dsin\theta \\ z_1 = z_2 \end{cases} \tag{2.7}$$

After calibrating the coordinates to our pre-setting position, 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.

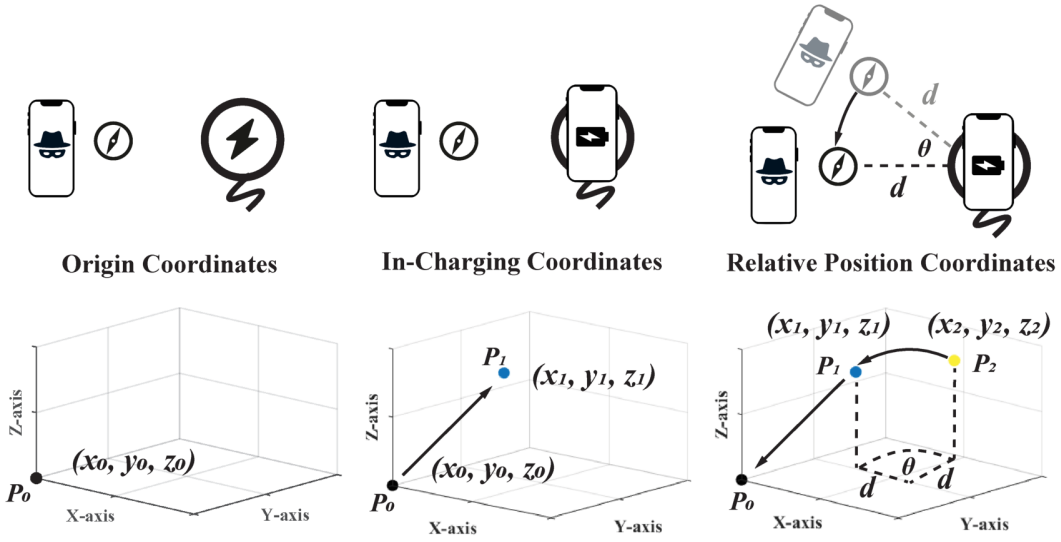


Fig. 2.8 Magnetic field calibration.

2.4.2 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 (§2.3.1).

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 Fig. 2.2, the power spectrum of the coil whine could reflect different types of interfaces. Hence, similar to inferring the battery level (§2.4.1), `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 § 2.3.1, 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 [32] 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 [33] 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 [34]), it also normalizes each activity attempt into the same length of time via property-preserving up-sampling (*e.g.*, nearest neighbor interpolation [35]) or down-sampling (*e.g.*, decimation factor [36]) 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 [37] 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 the 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.

2.4.3 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.

2.4.4 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) [38], Gammatone cepstral coefficients (GTCCs) [39], linear prediction cepstrum coefficients (LPCCs) [40], spectral power patterns [41], *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 A.1 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.

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 [42–44], 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.

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.

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 § 2.4.2, a keystroke may last from 0.05 to 0.2 seconds on average [34]. 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.

2.5 Evaluation

2.5.1 Evaluation Setup

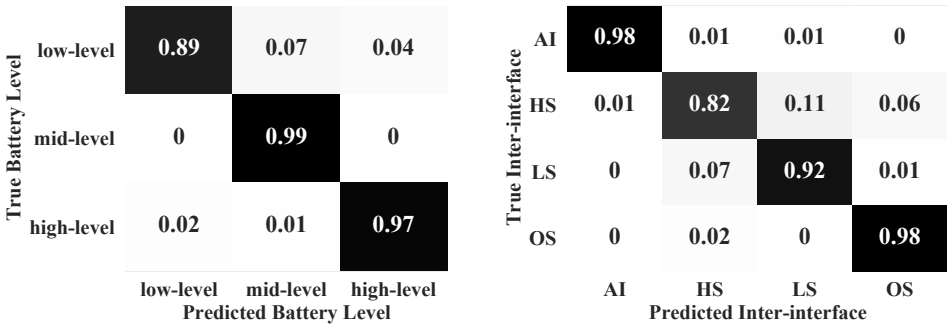
Our evaluation involves two sets of equipment to collect data and process the collected data for training and testing. To collect data, we use an iPhone 11 as the data collector (the attack device) to collect data from an iPhone 13 Pro (the victim device) charging on a 10 W Gikfun wireless charger at 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 11 uses two free apps from Apple's App Store to collect data, *i.e.*, Audio Recorder [45] (version 1.8.1) that uses the iPhone's microphone to record the coil whine, and Sensor Logger [46] (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* [47] 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.

2.5.2 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 Fig. 2.7 in §2.4.1). 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 (§2.4.4), and split these traces into the training set and test set with the ratio of $8:2$.

Results. As shown in Fig. 2.9a, the overall accuracy of battery level inference is 95.0%. Specifically, the low-level, mid-level, and high-level accuracy are 98.7%,



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

Fig. 2.9 Effectiveness evaluation on coil whine based inference. OS: off screen; LS: lock screen; HS: home screen; AI: app interface.

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 Fig. 2.7 in §2.4.1, 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 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(§2.4.2), we evaluate the accuracy of recognizing different types of interfaces. As shown in Fig. 2.9b, 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.

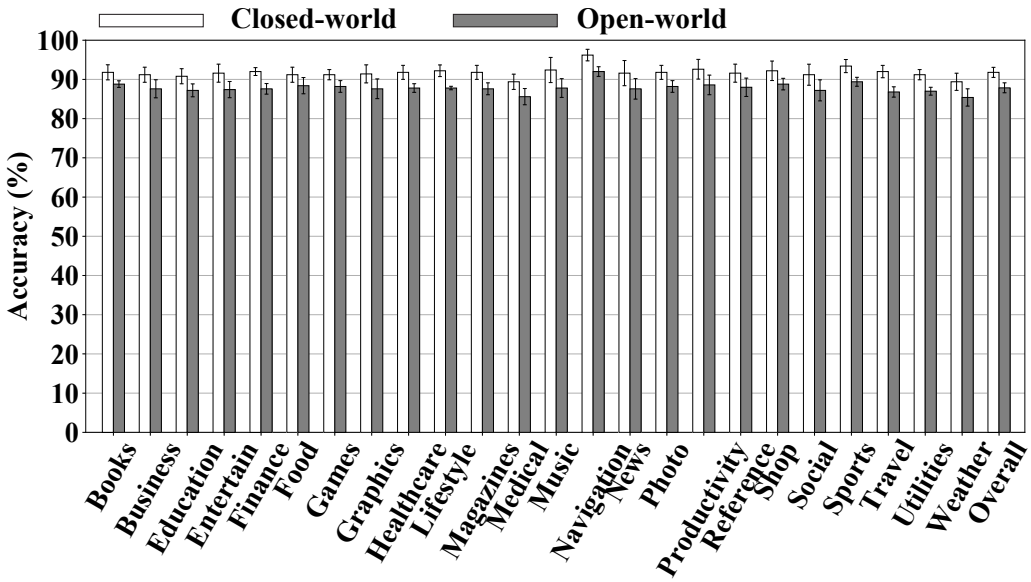
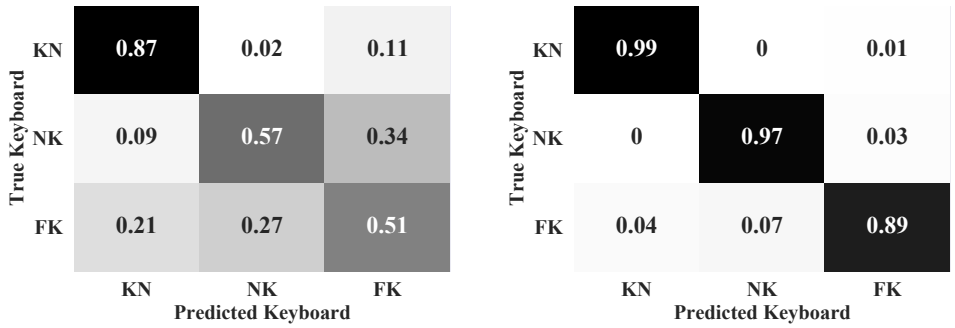


Fig. 2.10 Effectiveness evaluation on app recognition at launch.

2.5.3 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. Like collecting coil whine traces, each magnetic field perturbation trace 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 Fig. 2.10, the effectiveness of recognizing an app at its launch is evaluated in both the closed-world and open-world settings defined. As mentioned in §2.4.4, in both settings, we use the same dataset to train the model, and 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-



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

Fig. 2.11 Effectiveness evaluation of keyboard open: KN for keyboard not open, KO for keyboard open, NK for numeric-only keyboard, FK for full-size keyboard.

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 Fig. 2.10.

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 (§

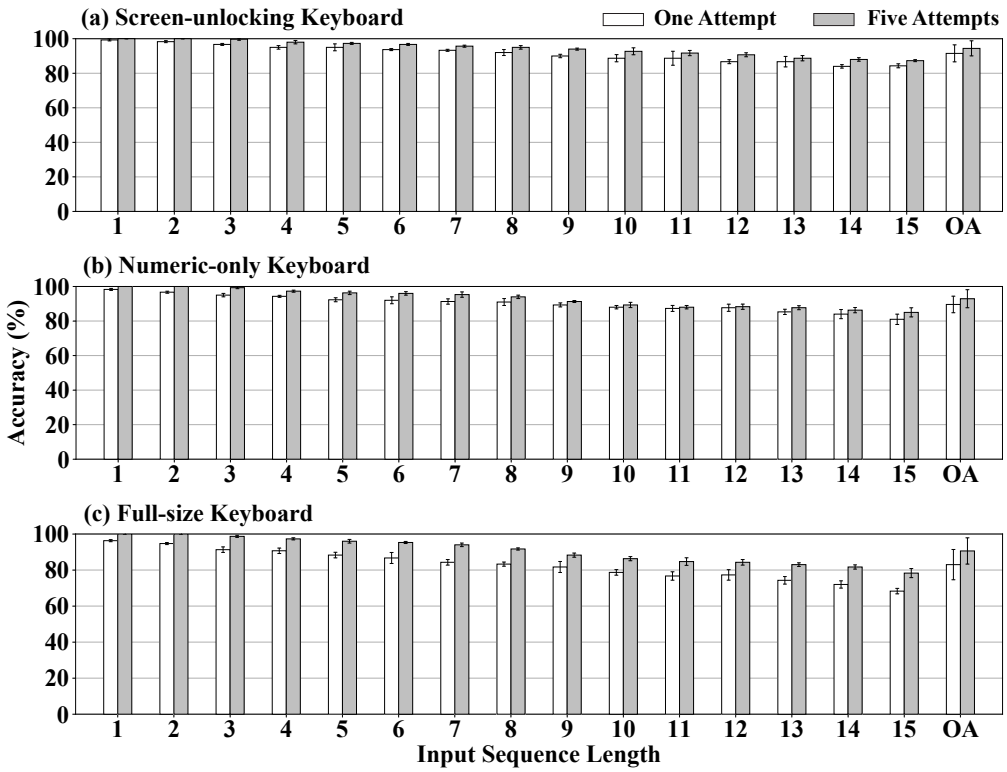


Fig. 2.12 Effectiveness evaluation on keystroke uncovering with different lengths from three keyboards (OA: Overall).

2.4.4); 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 Fig. 2.11a, 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 Fig. 2.11b, 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

Table 2.1 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	○	○	●	●		●	●	○	●	○	✗	✓

(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 §2.4.2. 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.

[1] The full list of the testing cases in keystroke inference and more detailed experiment results of §2.5.4 (end-to-end attacks) are available at: https://github.com/WISERS-SP23/WISERS_Experiment_Details

Results. Fig. 2.12 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 [15], 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%).

2.5.4 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 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 2.1 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 2.1, `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 2.1 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 2.1, 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

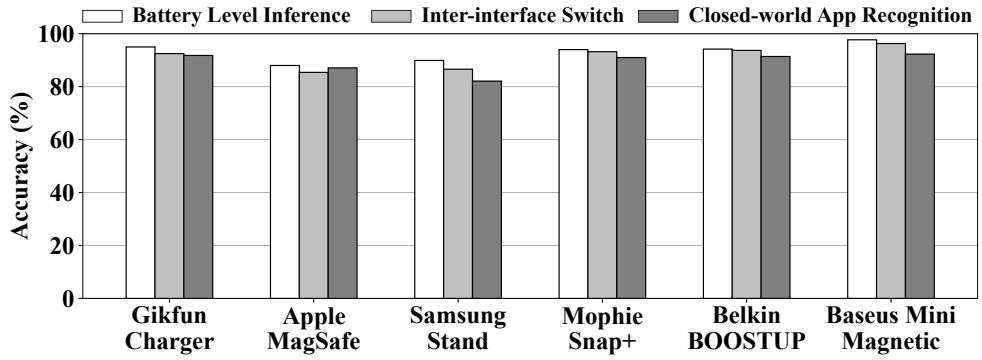
when there is a character in such a word appears consecutively, such as "ee" in the word "freezing", or the corresponding key of a character has relatively smaller space than the other keys, such as "." in the middle of "gmail.com". Similarly, within at most five attempts, all user chatting content is recovered.

2.5.5 Impact Factors

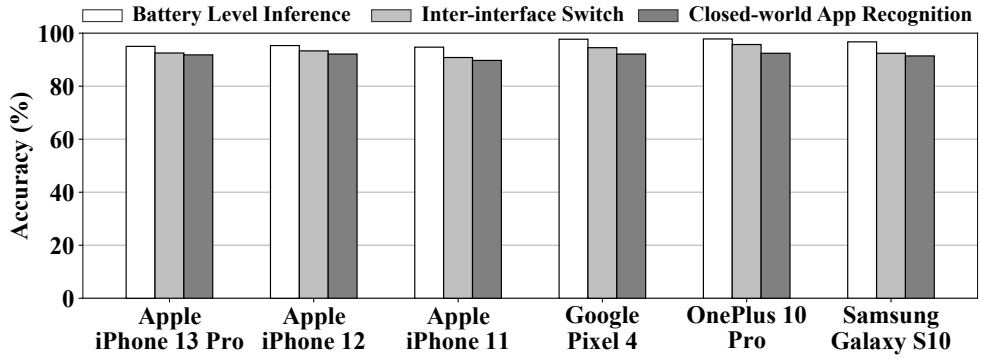
Impacts from different users. Considering previous studies have found that users spend different amounts of time for each keystroke [34], `WISERS` normalizes each keystroke trace to reduce such impacts on recovering keystrokes. To evaluate its performance, we have conducted an IRB-approved user study. 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 § 2.5.1. By using the same classification model trained on these three datasets, as shown in Fig. 2.17 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 §2.5.

As shown in Fig. 2.13a, `WISERS` can achieve high precision in all six evaluations across different wireless chargers. In particular, the accuracy of these chargers in



(a) Different wireless chargers.



(b) Different smartphones.

Fig. 2.13 Impact factor analysis of wireless chargers and smartphone models.

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 have 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 Fig. 2.13b show that these six classification models of different smartphones achieve the accuracy from 94.7% to

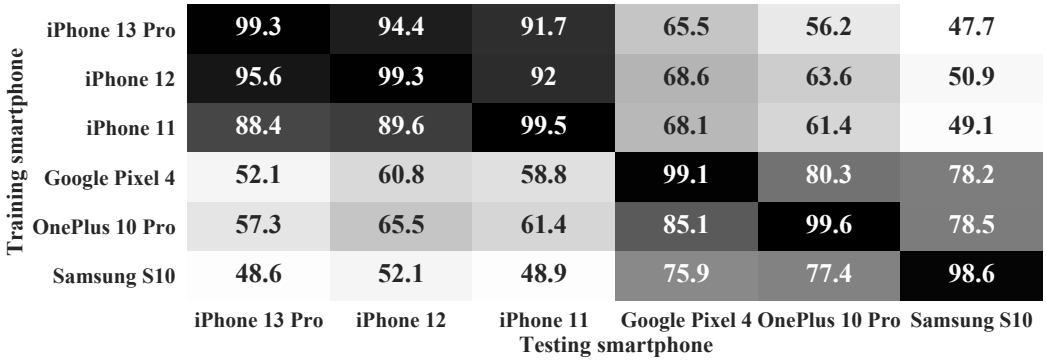
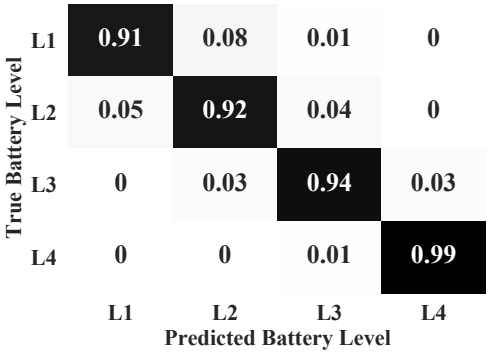


Fig. 2.14 Transferability of different smartphones.

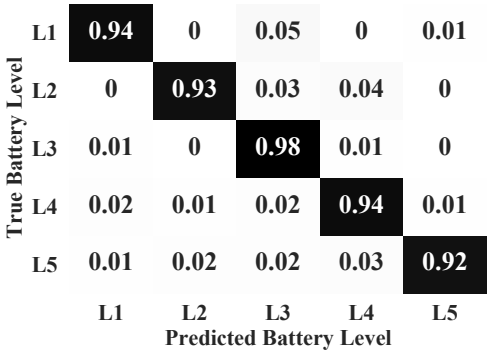
97.8% (1.38% SDV) in battery level inference, 90.8% to 95.7% (1.72% SDV) in inter-interface switch, and 89.7% to 92.4% (0.98% SDV) in closed-world app recognition.

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 Fig. 2.14, 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 Fig. A.2 in the Appendix. In short, though performance might decrease, `WISERS` can also work for cross-device attacks.

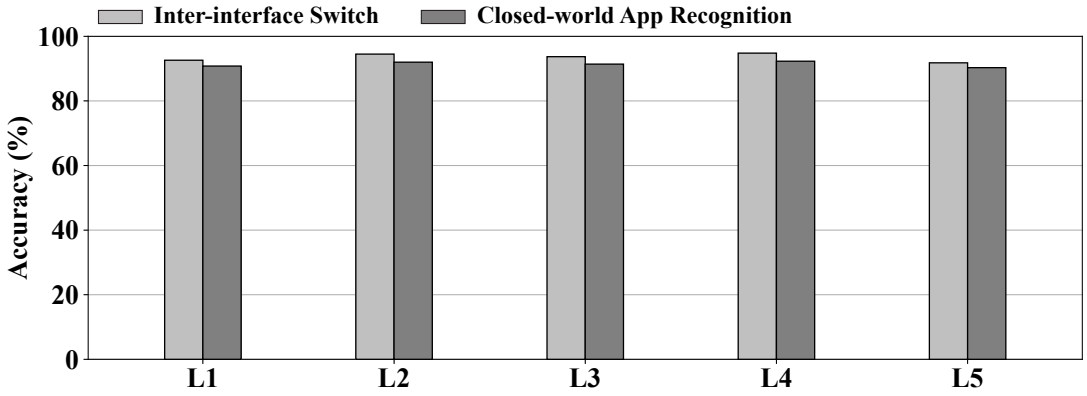
Impacts from different battery levels. According to the wireless charging protocol, the same activity could show different patterns of coil whine and magnetic field perturbations if the smartphone contains different battery levels. We comprehensively investigate 26 commodity wireless chargers with their battery levels in the charging process [48], 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



(a) Mophie Snap+.



(b) Belkin BOOSTUP.



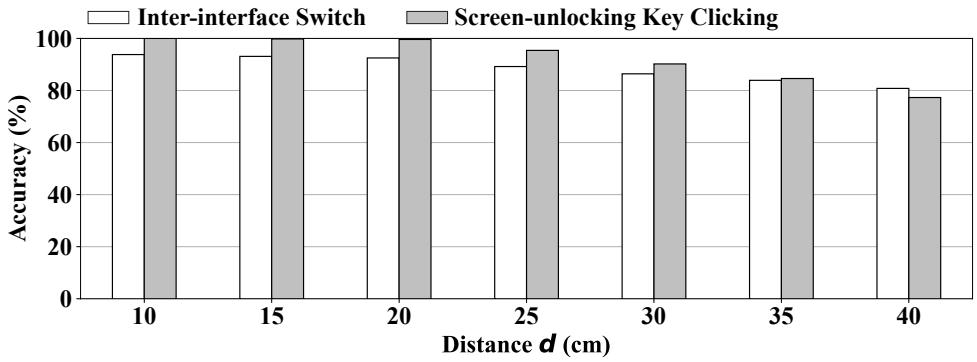
(c) Impact factor analysis of battery levels (Belkin BOOSTUP).

Fig. 2.15 (a) and (b): Battery level inference of wireless chargers with more than three levels. (c) Effectiveness at different battery levels of the Belkin BOOSTUP.

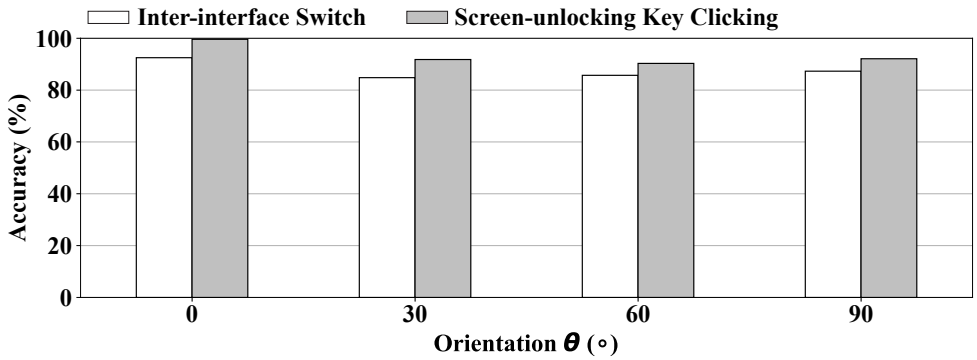
7.5W wireless charger has five levels (the full investigation results are presented in Table A.2 in the Appendix). Therefore, similar to the experiment in the battery level inference, we also evaluated `WISERS` in wireless chargers with more than three charging battery levels. As shown in Fig. 2.15a and Fig. 2.15b, `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 Fig. 2.15c, 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 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.

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 Fig. 2.16a, 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 Fig. 2.16b, 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 smoother accuracy decrease.



(a) Distance.



(b) Orientation.

Fig. 2.16 Impact analysis of distance and orientation.

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 2.2. 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 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

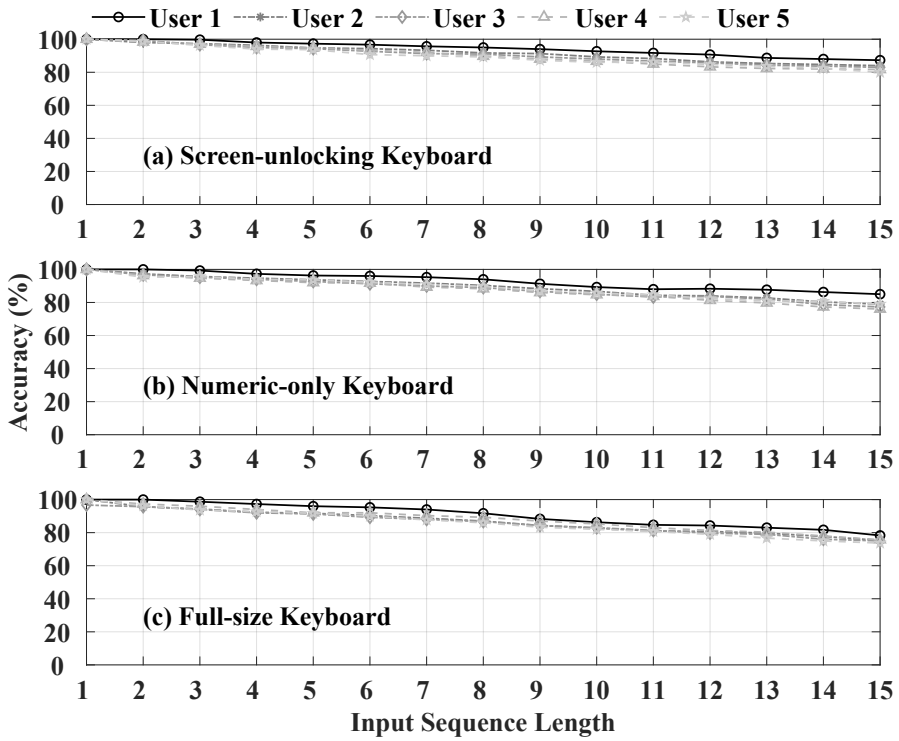


Fig. 2.17 Impact factor of users on uncovering keyboard inputs.

Table 2.2 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

could be observed. As such, the impacts of different apps on soft keyboard keystroke uncovering are not significant.

2.6 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 [49, 50] and apps [51, 52], which raise victims' suspicious, and this attack can be prevented by simply covering the touchscreen with hands [53] or using anti-peep screen protectors [54]. 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 [55]. 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 attack 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 [56]). Similarly, the background apps might mislead `WISERS` under a few specific circumstances (*e.g.*, high run-sleep ratio [57]). 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 [58] and additional sensors [59], such as the accelerometer and gyroscope, which have already been integrated into most commodity smartphones.

2.7 Defense Methods

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 ex-

ploited. 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 [60]. 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 [3]) or dynamically switching the amplitude and frequency [61] of voltage in the primary coil. In addition, it is also feasible to apply intentional electromagnetic interference (IEMI) techniques when starting the wireless charging [62, 55, 63] We will explore these methods in future work.

2.8 Related Works

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 [3], and demonstrated the potential of hijacking and eavesdropping attacks on the Qi wireless charging standard from the induced current traces in limited scenarios [1]. Moreover, EM-Surfing [2] connected an external resistor to the power line of the wireless charger to monitor changes of the induced voltage for app and keystroke recognition. In addition, recent studies also demonstrate the feasibility for malicious audio injections via compromised wireless chargers [64, 65]

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 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 [16, 17, 14, 19]) or compromise the power cable (*e.g.*, USB cable) or power station to monitor power traces (*e.g.*,[13, 10]) 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 victims' device.

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 [66], monitor program execution [67, 68] from micro-controllers, extract secret keys [69–72], recognize the security code from the touchscreen [56], and infer keystrokes [26] from smartphones. Other works use a smartphone as a probe to launch attacks, such as monitoring software launches in laptops [73]. These EM-based attacks are launched in a close proximity, *e.g.*, less than 1cm [72, 71, 69, 70, 66, 56, 67, 68], 2.5–5cm [73], 20–90cm [26]. 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 [74, 75] and in iOS [15, 76]), while some are from a slightly different perspective. For example, magnetic side channels in iOS have been explored to monitor app activities [18] assuming pre-installed malicious apps and channels in Android are leveraged to finger movement trajectories [77, 59, 78] 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 [79–84]. Moreover, sensors in smartphones including accelerometers [85–87], gyroscopes [88, 89], and microphone [90, 91] have also been used to study associated side channels. Unlike these works, `WISERS` focuses on an entirely different and novel wireless charging side channel.

2.9 Summary

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.

Chapter 3

Contactless Side Channels in Wireless Charging Power Banks

Recently, power banks for smartphones have begun to support wireless charging. Although these wireless charging power banks appear to be immune to most reported vulnerabilities in either power banks or wireless charging, we have found the same *contactless* wireless charging side channel in these power banks that leaks user privacy from their wireless charging smartphones without compromising either power banks or victim smartphones. We have proposed `BankSnoop` to demonstrate the practicality of the discovered wireless charging side channel in power banks. Specifically, it also leverages the coil whine and magnetic field disturbance emitted by a power bank when wirelessly charging a smartphone and adopts the few-shot learning to recognize the app running on the smartphone and uncover keystrokes. We evaluate the effectiveness of `BankSnoop` using commodity wireless charging power banks and smartphones, and the results show it achieves over 90% accuracy on average in recognizing app launching and keystrokes. It also presents high adaptability when apply to different smartphone models, power banks, *etc.*, achieving over 85% accuracy with 10-shot learning.

3.1 Introduction

Today, power banks have almost become one of the must-carry-on devices for numerous people to charge their smartphones outdoors if the battery is about to die. Accordingly, we have witnessed the tremendous growth of power bank rental stations in various public spaces (Fig. 3.1), *i.e.*, cafes and airports, making their global market exceed a value of 15.9 billion dollars worldwide (North America 54%, Asia Pacific

21%, Europe 10%, as shown in Fig. 3.2) by the end of 2030 [92]. Recently, many newly released power banks have begun to support wireless charging because of its growing popularity, and these power banks mostly follow the Qi [8] wireless charging standard that is widely supported by different smartphone models running different mobile operating systems (*e.g.*, iOS and Android).

Fig. 3.1 Illustration of power bank rental stations to provide charging services in shared spaces and public facilities.

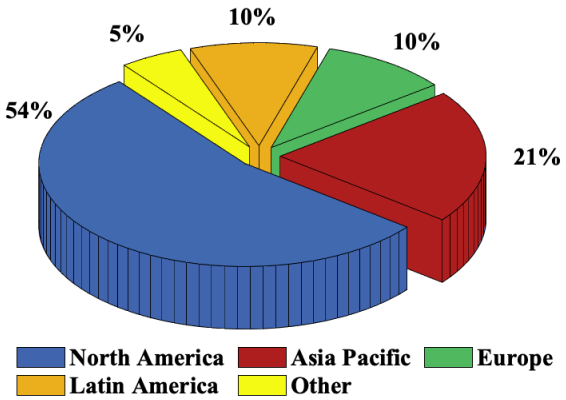


Fig. 3.2 Power bank rental service market categorized by region by the end of 2030 (North America 54%, Asia Pacific 21%, Europe 10%, Latin America 10%, and other regions 5%) based on a recent investigation [92].

While previous studies [1, 3, 2, 10, 12] have reported that either wireless charging or purely cable-based power banks could be leveraged to infer user privacy from the charging smartphones, these studies have not raised sufficient public awareness. This is because it seems plausible for wireless charging power banks to survive those vulnerabilities in daily cases: *(i)* a wireless charging power bank does not require a USB cable that often connects cable-based power banks to smartphones for charging, where the cable can be used to collect side-channel information, *i.e.*, the charging current [10, 12] to eavesdrop on user privacy. *(ii)* unlike wireless chargers, wireless charging power banks

are not connected to the power outlet via the power cable when charging a smartphone, and the power supply is dynamically adjusted to the power bank's battery level. Hence, reported attacks that collect traces (*e.g.*, current and voltage) in the power cable and assume the power supply is stable [1, 2] cannot apply to wireless charging power banks. *(iii)* similar to wireless chargers, heterogeneous wireless charging power banks also rely on well-trained models [1, 3, 2, 10], which are challenging to generalize across different power banks and make the attack cost economically unacceptable in practice.

However, wireless charging power banks are not as robust to privacy leakage as they may appear. In this chapter, we report the same *contactless* wireless charging side channel in power banks that can be exploited to infer user privacy (*e.g.*, app usage, keystrokes) from their charging smartphones without compromising both the power bank and the smartphone in any way. This wireless-charging side channel also leverages two physical phenomena that are essentially rooted in the wireless charging process, *i.e.*, the emitted coil whine and the induced ambient magnetic field disturbance. These two physical phenomena are stemmed from the load changes [2] resulting from smartphone activities (*e.g.*, turning on the screen, receiving notifications), and these changes slightly vibrate the internal coil of a wireless charging power bank. An adversary could leverage the two physical phenomena to determine the device type and battery status of the charging devices, and infer users' activities on the charging smartphone from their unique and distinctive patterns.

We have designed and implemented `BankSnoop` to demonstrate the feasibility of leveraging our reported novel attack surface to launch a contactless, fine-grained, and domain-adaptive attack on wireless charging power banks for the first time. Table 3.1 summarizes a comparison with five state-of-the-art related works [1, 3, 2, 10, 12] from five metrics, which shows `BankSnoop` is more stealthy and practical: *(i)* It requires no prior knowledge of the smartphone and power bank, *(ii)* It has no need to compromise the power bank or install malware into the victim's smartphone, and *(iii)* it achieves good transferability across various attack scenarios. Specifically, `BankSnoop` detects the coil whine and measures the ambient magnetic field disturbance to recognize the content changes displayed on a smartphone's screen and uncover sensitive information through four steps: *First*, it detects the appearance of the coil whine as the indicator to trigger the attack because the coil whine can only be generated when a wireless charging power bank is attached to the smartphone and begins to charge its battery. *Then*, it depends on the power spectrum of the coil whine to recognize the type of power bank and smartphone and then leverages the magnetic field traces to infer their battery levels to specify the attacking conditions since battery levels determine the power con-

Table 3.1 Comparison with related attacks from five metrics: (M1) contactless or not; (M2) no need to compromise devices; (M3) no prior knowledge of charging devices; (M4) fine-grained user privacy inference; and (M5) adaptive to various conditions. ✓: true, ✗: false.

Attacks	Attack surface	M1	M2	M3	M4	M5
Cour *et al.* [1]	Current in the power line	✗	✗	✗	✗	✗
Wu *et al.* [3]	Inductive current	✓	✓	✗	✗	✗
EM-Surfing [2]	Inductive voltage	✗	✗	✗	✓	✗
Charger-Surfing [10]	Current in the USB cable	✗	✗	✗	✓	✗
GhostTalk [12]	Current in the USB cable	✗	✗	✗	✓	✗
`BankSnoop`	Coil whine and magnetic field	✓	✓	✓	✓	✓

sumption that significantly affects the strength and direction of ambient magnetic field disturbance. *Next*, it utilizes the magnetic field disturbance to recognize different user activities resulting from different displaying content on the screen. *Moreover*, it also adopts few-shot learning to quickly adapt pre-trained models to be deployed in new attack scenarios and environments, considering a relatively large number of practical factors in practice

We evaluate the effectiveness of `BankSnoop` with a custom-built portable attacking device, which comprises commercial-off-the-shelf (COTS) electronic components, in uncovering three user privacy information, *i.e.*, app launching, in-app activities, and input keystrokes (*e.g.*, unlocking passcode, keyboard input) from different wireless charging power banks and smartphones configuring with various impact factors (different battery levels, users, screen brightness, *etc.*). Our evaluation shows that `BankSnoop` achieves high effectiveness in coil whine detection (99.0%), charging device fingerprinting (98.3%), battery level inference of both the power bank and the smartphone (99.8%), app launching recognition (93.1%), and keystroke uncovering from the unlocking numeric keyboard (94.9%) and the full-size QWERTY keyboard (86.9%) within ten attempts. In addition, `BankSnoop` also presents high performance and resilience when considering different practical impact factors by exploiting the few-shot learning module with 5-shot and 10-shot adaptation. On average, it achieves over 80% and 85% accuracy in 5-shot and 10-shot when adapting to different scenarios.

Contributions. We summarize the contributions as follows:

- **A novel side-channel attack.** We introduce a new side channel that can be exploited to attack wireless charging power banks in a contactless manner. It leverages

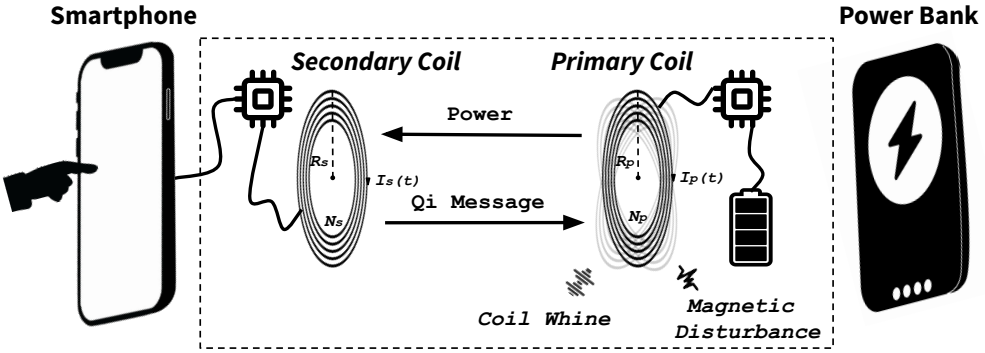


Fig. 3.3 Wireless charging using a power bank.

the emitted coil whine and the induced magnetic field disturbance to reveal sensitive information about the smartphone when wirelessly charged by a power bank.

- **A new attack framework.** We propose and implement a new attack framework, `BankSnoop`, to demonstrate the feasibility of the new side channel and address limitations in the previous wireless charging side-channel attacks.
- **Extensive evaluation.** We conduct an extensive evaluation to demonstrate the effectiveness of `BankSnoop`. The results indicate that it achieves high accuracy in uncovering user privacy and demonstrates great performance for domain adaptation in various attack scenarios.

3.2 Preliminary

3.2.1 Wireless Charging Power Bank

Nowadays, almost all wireless charging power banks are designed to support the Qi wireless charging standard [8]. As shown in Fig. 3.3, once a power bank is attached to a smartphone, it uses electromagnetic induction [93] to transfer power from its coil (*primary coil*) to the coil in the smartphone (*secondary coil*). First, the power bank generates inductive electromagnetic fields $\Phi_p(t)$ and $\Phi_s(t)$ in the primary coil and the secondary coil based on the Biot-Savart law (Equation 3.1). Then, according to Faraday's law, the inductive electromagnetic field $\Phi_s(t)$ generates an induced voltage $U_s(t)$ to charge the smartphone as shown in Equation 3.2:

$$\Phi_p(t) = \frac{\mu_0 N_p I_p(t)}{2R_p}, \Phi_s(t) = \eta \Phi_p(t) \tag{3.1}$$

$$U_s(t) = N_s \frac{\Delta \Phi_s(t)}{\Delta t} = \eta \frac{N_s}{N_p} \cdot \frac{\mu_0 \Delta I_p(t)}{2 R_s \Delta t} \tag{3.2}$$

where $I_p(t)$ is 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 is the energy transmission coefficient, and μ_0 is the magnetic constant.

In the power transfer phase, the control circuit in the power bank continuously communicates with the control unit in the smartphone via Qi message and adjusts the current in the primary coil. That is, when the user is using mobile apps on the charging smartphone, the smartphone will increase the charging speed as the running app consumes more power [8, 3]. Hence, the smartphone will send a Qi message to the wireless charging power bank to request more power supply, which changes the current running in the primary coil of the wireless charger.

3.2.2 Two Physical Phenomena

In the charging process, the wireless charging power bank controls the running current in the coil $I_p(t)$ based on the level of its battery $B(t)$ (Equation 3.3) because it contains limited electricity storage and a continuous large discharging current will inevitably shorten the battery life.

$$I_p(t)\ (\text{A}) = \begin{cases} I_{p_1} & B_0 < B(t) \leq B_1\ \ (\text{mAh}) \\ I_{p_2} & B_1 < B(t) \leq B_2\ \ (\text{mAh}) \\ I_{p_3} & B_2 < B(t) \leq B_3\ \ (\text{mAh}) \\ \cdots & \cdots \end{cases} \tag{3.3}$$

In particular, the adjusting mechanism of the charging voltage $U_s(t) \propto \Delta I_p(t)$ in the Qi wireless charging protocol [8, 3] and the consequent dynamically changed current ultimately result in slight vibrations of the coils, which incites two physical phenomena, *i.e.*, the coil whine and the disturbance of the electromagnetic field.

Coil whine. Coil whine, *a.k.a.*, electromagnetically induced acoustic noise, is a microphonics phenomenon produced by the coils' vibration under the excitation of electromagnetic forces (*e.g.*, Maxwell stress tensor, magnetostriction, and Lorentz force) [23]

based on the Ampere's force Law as:

$$F_p(t) = N_p \Phi_p(t) I_p(t) L_p \propto B^2(t) \tag{3.4}$$

where L_p is the circumference of the primary coil. These forces cause distortion and vibration of the coil, resulting in different coil whines. Specifically, coil whine exists in different frequency ranges, which makes it either human audible (frequency between 20Hz and 20kHz) or inaudible, while it can be captured by sound-recording microphone modules with sufficient sampling frequency [28].

Magnetic field disturbance. Different smartphone-user interactions cause changes in induced current during the charging process [1], which results in the disturbance of the ambient electromagnetic field. Specifically, power-intensive smartphone activities (*e.g.*, screen animation, pressing keyboard) change the load by $\Delta r(t)$ on the secondary coil, which further results in the disturbance of the electromagnetic field $\Delta\Phi(t)$ as shown in Equation 3.5:

$$\Delta I(t) = \frac{U_s(t)}{\Delta r(t)}, \Delta\Phi(t) = \frac{\mu_0 N_s \Delta I(t)}{2R_s} = \frac{\mu_0 N_s U_s(t)}{2R_s \Delta r(t)} \propto \frac{\Delta B(t)}{\Delta r(t)} \tag{3.5}$$

As such, these changes can be measured by monitoring the electromagnetic field over a period of time. In practice, the electromagnetic field at a specific time point can be described as a 3-D (x, y, z) vector that can be captured by magnetometers, which can be utilized for inferring various user activities on smartphones [94, 73, 57].

3.3 Motivation, Principle and Threat Model

3.3.1 A Motivating Example

This section presents a motivating example of launching our newly discovered side-channel attack in a real-life scenario. After attaching a wireless charging power bank to a smartphone, a user unlocks the screen with the password (*e.g.*, "1234"), taps the app icon to open the PayPal, and enters the password "abcde" to access financial functions (*e.g.*, pay, transfer). These actions change the content displayed on the screen from one to another accordingly. As mentioned in §3.2.2, changes on the screen could impact the current in both the primary coil in the wireless charging power bank and the secondary coil in the smartphone, which further influences the ambient electromagnetic

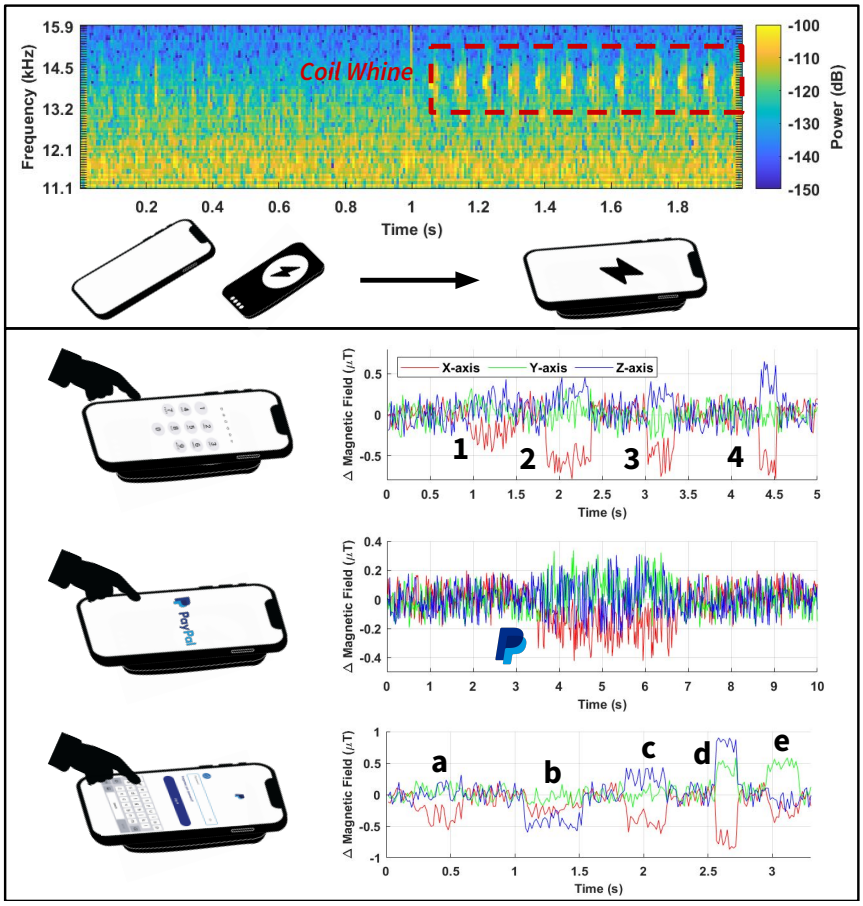


Fig. 3.4 Motivating example scenario: a user charges the smartphone with a wireless charging power bank, unlocks the screen with password "1234", touches the app icon to open PayPal, and types password "abcde" to enter the financial account. Upper part: the power spectrum of the coil whine when the wireless charging process starts. Lower part: user activities and corresponding changes of the magnetic field.

field, and these changes present detectable features that can be used for recognizing corresponding smartphone activities to infer user privacy.

In Fig. 3.4, we present the changes in coil whine and the ambient magnetic field associated with displaying content changes resulting from different user interactions. Specifically, we show the power spectrum of the coil whine within the range from 13kHz to 15kHz and the three-axis magnetic field disturbance of unlocking keystrokes, app launching, and QWERTY keystrokes. As can be seen, the coil whine is detected right after attaching the wireless charging power bank to a smartphone. On the other hand, the magnetic field shows apparent disturbances when launching an app (*i.e.*, PayPal). Moreover, the magnetic field disturbance can also reflect and distinguish different keystrokes on the unlock numeric keyboard (*e.g.*, "1") and the full-size QWERTY key-

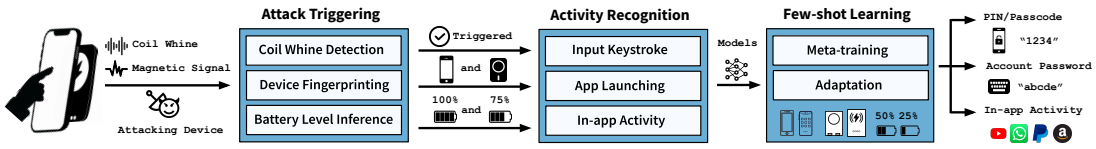


Fig. 3.5 Overview of **BankSnoop**.

board (*e.g.*, "a"). Therefore, the mentioned two physical phenomena, coil whine and the magnetic field disturbance, can be exploited to develop a new contactless wireless charging side-channel attack to reveal the displaying content changes on the screen and uncover sensitive information (*e.g.*, running apps, in-app activities, keystrokes).

3.3.2 Threat Model

We consider a common scenario of using wireless charging power banks to charge smartphones where a victim attaches the wireless charging power bank (*e.g.*, his/her own or borrowed from shareable rental stations) to the smartphone. Then, the victim places the devices on a table and performs a series of interactions (*e.g.*, typing the keyboard on the screen, running apps). Such a scenario is prevalent in daily life in various public spaces like airports or cafes. The attackers can place an attacking device to record the leakage of physical traces in close proximity to the target power bank.

What an attacker cannot do. Unlike many relevant attacks [1, 3, 2, 10, 12], the attacker does not necessarily have prior knowledge of the charging smartphone and the power bank (*e.g.*, model type, battery status). We do not assume the attacker can compromise the power bank or install malware into the victim's smartphone to acquire current/voltage traces either. Also, the attacker has no LoS view of the two devices and does not know the specific time that the wireless charging power bank begins to charge the smartphone. Moreover, it is unlikely for attackers to collect large amounts of data samples from different conditions to train multiple privacy inference models before the attack.

What an attacker can do. We assume the attacker could observe the starting orientations of the charging devices and places a small attacking device to record coil whine and measure the ambient magnetic field disturbance in close physical proximity to the target power bank (underneath the table or side-by-side, *e.g.*, 4in or 10cm, within a certain distance). The attacking device could be small to be hidden in a common elec-

tronic peripheral (*e.g.*, an earbud case) that can be attached beneath a table or put near the power bank without being notified.

3.4 Attack Framework

Fig. 3.5 presents the overview of `BankSnoop`. An attacker first acquires coil whine and magnetic field signals to determine the charging status and trigger the attack to recognize the types and battery levels of both the smartphone and the power bank. Then, the triggered attacking device utilizes magnetic signals for fine-grained activity recognition using pre-trained deep neural network models. Moreover, `BankSnoop` also incorporates a few-shot learning module for quickly adapting to various attack scenarios (*e.g.*, different smartphone models, power banks). Finally, the attacker can infer fine-grained user activities and privacy such as unlocking passcode, sensitive keystrokes, and in-app activities.

3.4.1 Attack Triggering Recognition

To launch this side-channel attack, we need to recognize the triggering condition where a power bank wirelessly charges a smartphone. As mentioned above (§ 3.2), it will generate two physical phenomena, the coil whine and the magnetic field disturbance at such a condition. Specifically, the coil whine appears when the wireless charging starts and performs as an indicator to trigger the attack. Furthermore, the magnetic field disturbance can be used to infer the status of the charging devices. As such, we use them together to detect the charging status by monitoring the coil whine and infer the battery levels of both the power bank and the charging smartphone by measuring the magnetic field.

Coil whine detection. We find the acoustic phenomenon, coil whine, which exists during the wireless charging process. Therefore, we can use the coil whine effect as the attack trigger of `BankSnoop`. In practice, the microphone module on the attack prototype first detects the coil whine, and a high-pass filter is applied to remove noises caused by low-frequency sounds (*e.g.*, human speaking, touchscreen tapping). Next, we obtain the power spectrum of the filtered audio by utilizing Short-time Fourier Transform (STFT) using a periodic Hann window. Then, we extract acoustic features Mel-frequency cepstral coefficients (MFCCs) [38] from the power spectrum and a pre-trained Decision Tree [95] classifier determines the charging status at present (*non-*

charging or *in-charging*). In practice, we leverage MATLAB Audio Toolbox (version 3.0) to extract MFCC features to train the Decision Tree classifier with 10-fold cross-validation for detecting the appearance of coil whine.

Device fingerprinting. Aforementioned wireless charging side-channel attacks [1, 3, 2] usually assume the attacker knows the type of the victim's device, whereas it is impractical and increases the difficulties of launching such an attack in a real-life scenario. Based on Equation 3.4, we know the electromagnetic forces that cause coil whine, are related to the turns and circumference of the coils. Therefore, it is reasonable to exploit the coil whine to fingerprint different power banks and smartphones since their coils have varied characteristics. Following the same procedure, another pre-trained Decision Tree classifier is implemented to determine the device type of both the power bank and the smartphone.

Battery level inference. Unlike wireless chargers with cables that provide a stable charging voltage, wireless charging power banks contain limited electrical energy in the battery. As mentioned in §3.2.2, the power bank adjusts the charging voltage based on its battery status to prevent excessive discharge. Therefore, two battery levels are involved in the wireless charging process supported by the power bank, *i.e.*, the battery level of the smartphone and the battery level of the power bank. As such, `BankSnoop` depends on the inference of the two battery levels for two reasons. *First*, it recognizes the power bank's battery level to understand whether it has power left or not (yes or no) to be sufficient to launch the attack. *Second*, the smartphone's battery level is an essential factor impacting model performance in previous studies of wireless charging side channels such as [1], whose models can only work when the smartphone's battery exceeds 80%. Hence, to enhance `BankSnoop` with the practicality to launch attacks at any battery levels, we leverage the magnetic field disturbance to infer the exact battery level of the smartphone and the power bank to facilitate procedures in the following steps. Specifically, we leverage the magnetometer to collect three-axis magnetic signals and measure the strength of captured 3D magnetic field $Mag_s(t)$ at a specific time point t as shown in Equation 3.6:

$$Mag_s(t) = \sqrt{Mag_x^2(t) + Mag_y^2(t) + Mag_z^2(t)}, \tag{3.6}$$

where Mag_x, Mag_y, Mag_z represent the magnetic field on x, y, z axis, respectively. We obtain the strength differences by deducting the magnetic field when no charging device is presented and then calculate the difference's cumulative distribution function

(CDF). Next, we use the CDF values to train a Decision Tree classifier to infer the battery level of the power bank and the charging smartphone.

To demonstrate the feasibility of battery level inference in the charging process, we conduct a preliminary investigation to answer three research questions as follows:

- **RQ1:** Do different power banks present different battery levels in a wireless charging process?
- **RQ2:** Does the initial battery percentage of the smartphone impacts the inductive charging current?
- **RQ3:** Can CDFs of magnetic field strength differences distinguish the battery levels of a power bank?

To answer research questions *RQ1* and *RQ2*, we use a commodity app, *i.e.*, Amperes 4 [96], to record statistics during the charging status of iPhone 13 Pro starting from 10% charging with four wireless charging power banks: EGO MAGPOWER 2 [97], Anker MagGo [98], Apple MagSafe Battery Pack [99], and Belkin BOOSTCHARGE [100]. We present the current curves of these power banks in Fig. 3.6a and notice that Apple MagSafe Battery Pack and Belkin BOOSTCHARGE show stable current during the charging process, whereas EGO MAGPOWER 2 and Anker MagGo present ladder-like current changes. Additionally, we find the current levels of these two wireless charging power banks correspond to the battery levels that are normally displayed as the number of LED lights on the power banks.

> **Answer to RQ1:** Different power banks present different charging patterns, and some (*e.g.*, EGO MAGPOWER 2, Anker MagGo) present ladder-like battery levels.

Furthermore, we measure the inductive charging current in the *secondary coil* when charging an iPhone 13 Pro with EGO MAGPOWER 2 at different initial battery percentages. Fig. 3.6b shows that the inductive currents present similar ladder-like patterns regardless of the initial battery percentage of the smartphone. Although the initial battery percentage of the smartphone has no impact on the inductive charging current, it still influences the current patterns incurred by smartphone activities, as a prior work [1] has demonstrated. Therefore, in `BankSnoop`, we design the battery level inference module by recognizing the power bank's battery level and the percentage of the smartphone's battery.

Answer to RQ2: The inductive charging current in the *secondary coil* depends on the battery level of the power bank regardless of the smartphone's initial battery percentage.

To answer *RQ3*, we measure the battery levels of two wireless charging power banks (*i.e.*, EGO MAGPOWER 2 and Anker MagGo) that demonstrate ladder-like charging current curves by obtaining the cumulative distribution of the strength differences. Fig. 3.7a and Fig. 3.7b separately present the cumulative distribution plots of EGO MAGPOWER 2 and Anker MagGo, they all show that strength difference patterns are distinctive at different battery levels. Therefore, our proposed CDF-based method is feasible to distinguish the different battery levels of a power bank.

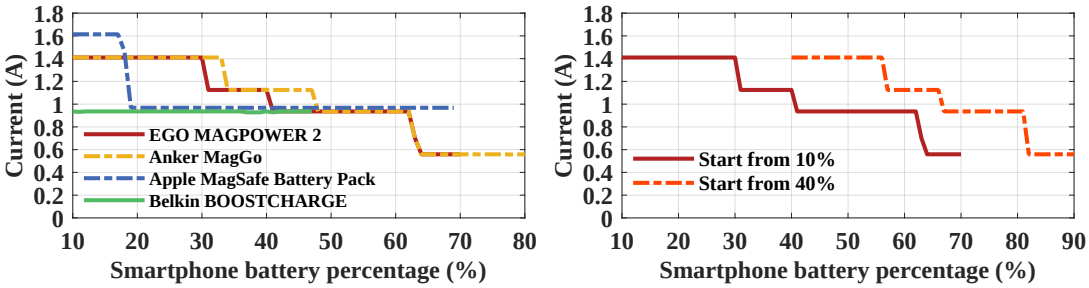
Answer to RQ3: We can use CDFs of the magnetic field strength differences as the measurement to distribute different battery levels of a wireless charging power bank.

3.4.2 Magnetic-based Activity Recognition

Having recognized the attack triggering condition and determined the devices' type as well as the battery levels, `BankSnoop` next exploits the captured 3D magnetic field signals with pre-trained deep learning models to recognize various user activities on the charging smartphone.

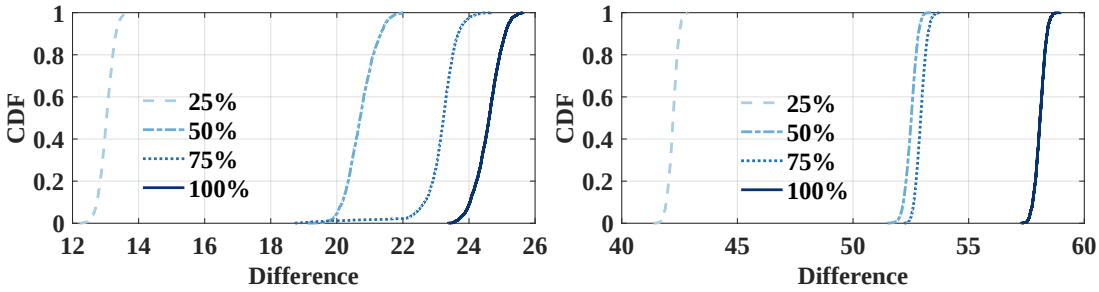
Pre-processing. After obtaining the raw magnetic field signals, we first apply a Savitzky–Golay (S-G) filter to remove noises in the collected sequential magnetic field signals without distorting the signal shapes [33, 101, 102]. Then, we calculate the average values of the first one-second data as the static magnetic field values on three axes, deduce this offset value as the starting coordinate, and further obtain the disturbance resulting from different user-smartphone interactions.

Since each activity has a different length of time in every attempt (*e.g.*, an app launching takes 1-5 seconds [57], a single key-press takes 0.05-0.2 seconds [34]), `BankSnoop` normalizes the processed signals of each activity attempt to the same length of time (*e.g.*, 0.1 seconds) by utilizing up-sampling (*e.g.*, interpolation [103]) or down-sampling (*e.g.*, decimation factor [36]) algorithms.



(a) Different power banks. (b) Different battery percentages.

Fig. 3.6 Charging curves measured from iPhone 13 Pro. (a) Charging with different power banks. (b) Charging at different smartphone battery percentages.



(a) EGO MAGPOWER 2. (b) Anker MagGo.

Fig. 3.7 CDFs of magnetic field strength differences at battery levels 25%, 50%, 75%, and 100% of two power banks.

Activity recognition. As the processed magnetic signals are three-axis one-dimensional time series, we adopt a one-dimensional convolutional neural network (CNN) to build a classifier for activity inference (*e.g.*, app fingerprinting, in-app activity recognition, and single key-press recognition). Specifically, CNN-based deep learning models are utilized in previous side-channel attacks using one-dimensional signals [94, 1] because they can capture temporal (*e.g.*, movements) and spatial (*e.g.*, position) features that reflect user-smartphone interactions from time series and achieve a high classification accuracy [10, 104]. In the CNN-based network, we utilize two convolutional layers to extract temporal and spatial features from the input time series and two batch-normalization layers to standardize the data and stabilize the learning process. Then, two max-pooling layers can reduce the dimension by half, and a dropout layer has been added to prevent overfitting. Finally, the flatten layer converts feature maps to one-dimensional, and the last fully-connected layers output the predicted class with the highest probability.

Implementations. We implement the CNN-based neural networks in Keras 2.3 on the Tensorflow 2.0 framework. We apply the ReLU activation function for two convolution layers and set the pool size as two for each max-pooling layer. In the training stage, we set the batch size as 32 and use the cross-entropy loss and Adam optimizer with an initial learning rate of 0.01 and epoch of 100. The output shape of the last fully-connected layer depends on the corresponding task (*e.g.*, the number of apps and keys on the keyboard).

In the case of keystroke inference, as users often type passwords or keystrokes in sequences of various lengths, we define each interval between two adjacent key presses as a new key class and add it to the training process of the mentioned two soft keyboards. Furthermore, the *softmax* function produces an array that contains the probability of each class and outputs the predicted label with the highest probability value (*a.k.a.*, *argmax*). Hence, we use the output of the *softmax* function and generate predicted sequences with top k (*e.g.*, $k = 5, 10, ...$) highest probabilities, which are also denoted as top-k prediction candidates. We utilize the top-k candidates to evaluate the keystroke inference performance of `BankSnoop` as it is reasonable that an attacker can surmise the correct passwords or keystrokes in a few attempts [26].

3.4.3 Adaptation via Few-shot Learning

Although the CNN-based magnetic signal classifier achieves promising accuracy, its performance can be impacted by shifting conditions. Therefore, previous side-channel attacks try to restrict prerequisites (*e.g.*, smartphone battery percentage over 80% [1]) or train multiple deep learning models for different configurations (*e.g.*, smartphone models [10]). However, these methods not only require large-scale datasets to ensure good performance but also limit attack scenarios. Therefore, considering various attack scenarios in practice, we design a few-shot learning module in `BankSnoop` based on the concept of model-agnostic meta-learning (MAML) [105]. Below, we illustrate our proposed algorithm in two stages: *meta-training* and *adaptation*.

Meta-training. We present the meta-training algorithm for the magnetic signal classifier in Algorithm 1. In the meta-training step, we denote the magnetic signal classifier as f and network parameters as θ. A set of tasks $\mathcal{T}$ are generated from the source dataset $\mathcal{D}_S$ that contains magnetic signal samples collected from various conditions (*e.g.*, different wireless power banks). For each task $\mathcal{T}_i \in \mathcal{T}$, the classifier learns to recognize N classes by using a small number of K (*e.g.*, five or ten) labeled samples of

each class, which is also known as *K*-shot *N*-way classification task. Furthermore, each task $\mathcal{T}_i$ involves a support set $\mathcal{S}_{\mathcal{T}_i}$ and a query set $\mathcal{S}_{\mathcal{Q}_i}$, where $\mathcal{S}_{\mathcal{T}_i}$ disjoints with $\mathcal{S}_{\mathcal{Q}_i}$ ($\mathcal{S}_{\mathcal{T}_i} \cap \mathcal{S}_{\mathcal{Q}_i} = \phi$) and each set contains $K \times N$ samples. The classifier f is initialized with random parameters θ_0 and then being trained by the associated support set $\mathcal{S}_{\mathcal{T}_i}$ of each task $\mathcal{T}_i$. Then, the classifier learns a new task-specific parameters $\theta^{'}_{\mathcal{T}_i}$ which are tuned from the initial parameters θ_0 via updating the gradient descent:

$$\theta^{'}_{\mathcal{T}_i} = \theta_0 - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_{\theta_0}, \mathcal{S}_{\mathcal{T}_i}), \tag{3.7}$$

where α is a preset learning rate of individual tasks and $\mathcal{L}_{\mathcal{T}_i}(f_{\theta_0}, \mathcal{S}_{\mathcal{T}_i})$ is the task-specific cross-entropy loss of f on the support set $\mathcal{S}_{\mathcal{T}_i}$ which is given as follows:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\theta_0}, \mathcal{S}_{\mathcal{T}_i}) = \sum_{(\boldsymbol{x}_j, \boldsymbol{y}_j) \in \mathcal{S}_{\mathcal{T}_i}} \boldsymbol{y}_j \log f_\theta(\boldsymbol{x}_j) + (1 - \boldsymbol{y}_j) \log f_\theta(1 - \boldsymbol{x}_j), \tag{3.8}$$

where $(\boldsymbol{x}_j, \boldsymbol{y}_j)$ is the jth sample in $\mathcal{S}_{\mathcal{T}_i}$. With the task-specific parameters $\theta^{'}_{\mathcal{T}_i}$ of all tasks $\mathcal{T}_i$ in $\mathcal{T}$, we can define a meta-objective function presented as follows:

$$\underset{\theta}{\operatorname{argmin}} \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\theta^{'}_{\mathcal{T}_i}}, \mathcal{S}_{\mathcal{Q}_i}). \tag{3.9}$$

The objection function is proposed to find parameters θ^* that can minimize the sum of task losses in $\mathcal{T}$. We obtain the testing loss of task $\mathcal{T}_i$ by evaluating the performance of the task-specific classifier on the query set $\mathcal{S}_{\mathcal{Q}_i}$. Finally, we obtain θ^* by applying stochastic gradient descent (SGD) [105]:

$$\theta^* \leftarrow \theta_0 - \beta \nabla_\theta \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\theta^{'}_{\mathcal{T}_i}}, \mathcal{S}_{\mathcal{Q}_i}), \tag{3.10}$$

where β is another preset learning rate for SGD optimization. The final outputs of meta-training step is the classifier f_{θ^*} with the optimized parameters θ^*.

Adaptation. After obtaining the optimized initialization parameters θ^*, the magnetic signal classifier can realize fast adaptation to various attack scenarios (*e.g.*, different wireless charging power banks, battery levels, *etc.*) with only $K \times N$ labeled training samples collected from the new scenario to fine-tune the pre-trained model. For example, when a new target dataset $\mathcal{D}_{new}$ that is collected from a different wireless power bank ($\mathcal{D}_{new} \cap \mathcal{D}_{\mathcal{S}} = \phi$), the optimized classifier f_{θ^*} can quickly adapt to this new task

Algorithm 1: Meta-training for magnetic classifier

Input: $\mathcal{D}_{\mathcal{S}}$: source dataset. f: magnetic signal classifier. α and β: learning rate hyperparameters.

Output: f_{θ^*}: trained magnetic signal classifier with optimized parameters θ^*.

$\theta \leftarrow \theta_0$, $f_\theta \leftarrow f_{\theta_0}$ ▷ random initialize f_θ with parameters θ_0

while *not finished* **do**

  $\mathcal{T} \leftarrow$ generate a batch of tasks from $\mathcal{D}_{\mathcal{S}}$

  for *each task* $\mathcal{T}_i \in \mathcal{T}$ **do**

    $\mathcal{S}_{\mathcal{T}_i} \leftarrow K \times N$ support samples from $\mathcal{T}_i$

    $\mathcal{S}_{\mathcal{Q}_i} \leftarrow K \times N$ query samples from $\mathcal{T}_i$ ($\mathcal{S}_{\mathcal{T}_i} \cap \mathcal{S}_{\mathcal{Q}_i} = \phi$)

    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with $\mathcal{S}_{\mathcal{T}_i}$ and loss $\mathcal{L}_{\mathcal{T}_i}(f_\theta, \mathcal{S}_{\mathcal{T}_i})$

    $\theta'_{\mathcal{T}_i} \leftarrow \theta_0 - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_{\theta_0}, \mathcal{S}_{\mathcal{T}_i})$ ▷ obtain task-specific parameters $\theta'_{\mathcal{T}_i}$ of $\mathcal{T}_i$ using gradient descent.

    Evaluate $\mathcal{L}_{\mathcal{T}_i}(f_{\theta'_{\mathcal{T}_i}})$ with query set $\mathcal{S}_{\mathcal{Q}_i}$.

  $\theta^* \leftarrow \theta_0 - \beta \nabla_\theta \sum_{\mathcal{T}_i \in \mathcal{T}} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_{\mathcal{T}_i}}, \mathcal{S}_{\mathcal{Q}_i})$ ▷ obtain the optimized parameters θ^* that minimizes all task losses

Output classifier f_{θ^*} with optimized parameters θ^*

$\mathcal{T}_{new}$ and obtain the new parameters θ_{new} in a few gradient descent updates as follows:

$$\theta_{new} = \theta^* - \alpha \nabla_\theta \mathcal{L}_{\mathcal{D}_{new}}(f_{\theta^*}), \tag{3.11}$$

where the α is same to the hyperparameter denoted in Equation 3.7. After the adaptation stage, we obtain the magnetic signal classifier $f_{\theta_{new}}$ with fine-tuned parameters θ_{new} towards the new task $\mathcal{T}_{new}$. In practice, we adopt 5-shot and 10-shot with $N = 120$ for app fingerprinting, $N = 11$ and $N = 33$ for the unlocking and the QWERTY key-pressing recognition, respectively. We set the hyperparameter learning rate α and β as 0.01 and 0.001, respectively. Then, we apply ten gradient descent updates for generating the magnetic signal classifier with cross-task optimized parameters θ^* and ten gradient steps to fine-tune the θ^* to obtain parameters θ_{new} for target datasets in new scenarios.

3.4.4 Portable Attacking Device

As mentioned above, we have implemented a portable attacking device, which consists of four commercial-off-the-shelf (*COTS*) components: an Arduino Nano microcontroller unit (MCU) [106], a microphone module to capture coil whine, a three-axis

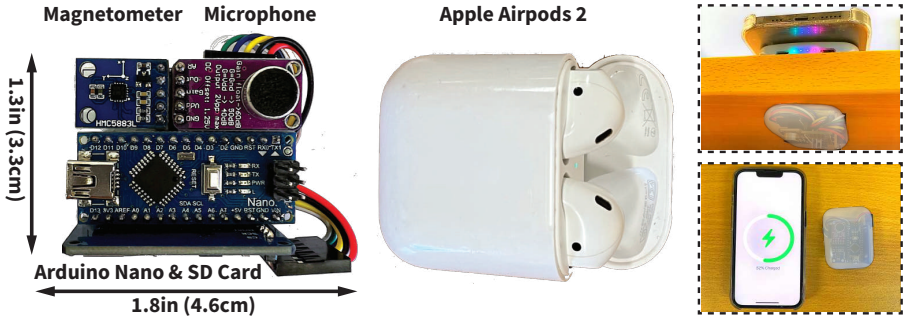


Fig. 3.8 Custom-built attacking device (almost the same size as Apple AirPods 2 charging case) and two common attack scenarios.

magnetometer module to capture magnetic signals, and a micro SD card shield to record collected data. Specifically, we use Adafruit MAX9814 microphone amplifier [107] and HMC5883L magnetometer module [108]. The total size of the attack prototype is approximately 1.8×1.3in (4.6×3.3cm), which is close to the size of an Apple AirPods case. The total cost is approximately 32.5 dollars.

3.5 Evaluation

3.5.1 Experiment Setup

In the primary settings of the experiment[1], we use a full-battery (100%) EGO MAG-POWER2 power bank to charge an iPhone 13 Pro at 80% battery percentage. Then we place the in-charging smartphone on an oak table with a thickness of 0.94in (2.4cm), and stick the attack prototype underneath the table. In addition, the preset sampling frequencies of the microphone and magnetometer are 40kHz and 100Hz, respectively. Moreover, all data processing is conducted on a desktop running Windows 10 with 32GB memory, Intel i7-9700K CPU, and an NVIDIA GeForce RTX 2080Ti GPU. Note that our experiments are conducted in an uncontrolled environment, and low-frequency noises (*e.g.*, human speaking: 50–300Hz; button pressing: 1–10Hz) have little impact since the coil whine has a high-frequency range (*e.g.*, 13–15kHz).

[1] **Ethical consideration**: This work takes ethical considerations seriously, and our IRB has approved it to collect data from human participants (HUMAN-2023-0016-2). More experimental details (*e.g.*, full list of testing sequences) are available at: https://github.com/CityuSeclab/BankSnoop_MobiCom23

3.5.2 Datasets

We build six different datasets on commodity devices in different conditions to demonstrate its effectiveness in §3.5.3 from four commodity wireless charging power banks (P_1–P_4: EGO MAGPOWER 2, Anker MagGo, Apple MagSafe Battery Pack, and Belkin BOOSTCHARGE) and four smartphones (S_1–S_4: iPhone 13 Pro, iPhone 12, iPhone 11, and Samsung S10) to train different models in **BankSnoop** and evaluate their performance in detecting coil whine and devices' type, inferring the battery levels, recognizing app/in-app activities, uncovering keystrokes, and adapting to different attack scenarios with the few-shot learning module.

- $\mathcal{D}_{CW}$: the coil whine dataset contains one-second audio clips in two cases: the smartphone is in-charging and non-charging. In the in-charging condition, we collect samples in screen-off and screen-on status. This procedure is repeated 50 times and then a 0.1 seconds sliding window is applied to perform STFT ($2 \times 50 \times 10 \times 4$ traces).
- $\mathcal{D}_{DF}$: the device fingerprinting dataset follows the same data collection from four smartphones that are being charged by four power banks ($4 \times 4 \times 50 \times 10$ traces).
- $\mathcal{D}_{BL}$: the battery level dataset is collected from two wireless charging power banks (more details in §3.5.3) that present different battery levels (25%, 50%, 75%, and 100%) when charging the four smartphones at four battery percentages (20%, 40%, 60%, and 80%), each charging combination 500 samples ($4 \times 4 \times 500 \times 8$ traces).
- $\mathcal{D}_{App}$: the mobile app dataset is collected from a total of 120 apps from the official iOS and Android store, which contains the top 5 popular free apps from 24 app categories based on the statistics provided by *appfigures* [109] by the end of 2021. We collect the first 0.1 seconds of launching each app and repeat it 100 times ($120 \times 100 \times 8$ traces). Moreover, we select the most popular five apps (*e.g.*, YouTube, PayPal) and collect data when performing five application-specific activities for 100 times to train the classifier for in-app activity recognition ($5 \times 5 \times 100$ traces).
- $\mathcal{D}_{UK}$ and $\mathcal{D}_{QWERTY}$: the two keystroke datasets are collected from two common soft keyboards: unlocking keyboard ($\mathcal{D}_{UK}$) and full-size QWERTY keyboard ($\mathcal{D}_{QWERTY}$). Each key (including backspace, space, *etc.*), as well as the interval key for segmentation, is pressed 100 times ($11 \times 100 \times 8$ traces in $\mathcal{D}_{UK}$ and $33 \times 100 \times 8$ traces in $\mathcal{D}_{QWERTY}$).

3.5.3 Effectiveness

We use accuracy and confusion matrix as the metrics to evaluate **BankSnoop** in coil whine detection, device fingerprinting, battery level inference, app launching/in-app activity recognition, and keystroke uncovering.

Effectiveness of coil whine detection. Based on $\mathcal{D}_{CW}$, we train a Decision Tree classifier to determine the presence of a power bank wireless charging a smartphone. On the testing set (200 samples for each wireless power bank), the Decision Tree classifier achieves an overall accuracy of 99% (EGO MAGPOWER 2: 99.5%, Anker MagGo: 98.5%, Apple MagSafe Battery Pack: 99%, and Belkin BOOSTCHARGE: 100%).

Effectiveness of device fingerprinting. Similarly, we use the captured coil whine in the dataset $\mathcal{D}_{DF}$ to train a Decision Tree classifier to recognize the type of smartphone and the power bank. Fig. 3.9 presents the confusion matrix of the device fingerprinting results of the 16 evaluated combinations (*e.g.*, $S_1 \times P_1$: iPhone 13 Pro charged by EGO MAGPOWER 2, $S_2 \times P_3$: iPhone 12 charged by Apple MagSafe Battery Pack). The results show that **BankSnoop** achieves an accuracy of 98.3% in recognizing the type of charging devices.

Effectiveness of battery level inference. Based on the results of the preliminary study (§3.4.1), we, therefore, utilize the MATLAB Statistics Toolbox to generate the CDFs of magnetic field strength differences from $\mathcal{D}_{BL}$ to develop a Decision Tree classifier to recognize the combination of both the battery level/percentage of the power bank and the in-charging smartphone. Fig. 3.10 presents the confusion matrix of battery level inference results of using the EGO MAGPOWER 2 to charge the iPhone 13 Pro at 16 different battery level combinations (*e.g.*, $S20 \times P100$: smartphone battery at 20%, power bank battery at 100%). It shows that **BankSnoop** achieves 99.8% accuracy in battery level inference.

Effectiveness of app launching recognition. Fig. 3.11 presents the effectiveness of **BankSnoop** in recognizing 120 mobile apps at the app launching stages. We utilize 80% data of each app from $\mathcal{D}_{App}$ to train the recognition model and evaluate its performance with the remaining 20% data. Overall, the recognition model achieves $93.1 \pm 2.9\%$ accuracy on the testing set of traces from 120 apps. Specifically, **BankSnoop** performs the best in identifying apps in categories such as "Books" and "Education" (accuracy 100.0%), and it performs worst in the category "Social Network" (accuracy

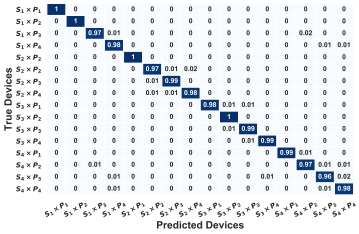


Fig. 3.9 Device fingerprinting results. $S_i \times P_j$: Smartphone S_i charged by the power bank P_j ($i, j = 1, 2, 3, 4$).

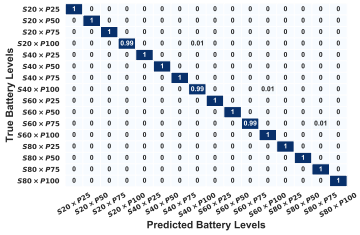


Fig. 3.10 Battery level inference results. $S_{l_1} \times P_{l_2}$: Smartphone battery at $l_1\%$, power bank battery at $l_2\%$ ($l_1, l_2 = 25, 50, 75, 100$).

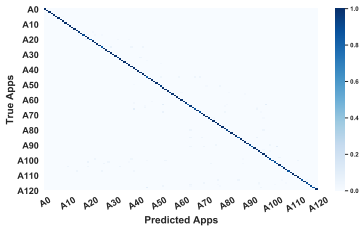


Fig. 3.11 App launching recognition results. A_n: the nth app of the most popular 120 mobile apps ($n = 1, 2, \ldots, 120$).

(a) YouTube (b) PayPal (c) WhatsApp (d) Facebook (e) Spotify

Fig. 3.12 In-app activity recognition results.Evaluated activities of and : ▶–Play, ⏸–Pause, ⏩–Forward, ⏪–Backward, ⏭–Next; : –Scan QR code, $–Pay bills, –Request money, –Get invoices, –Call wallet; : T–Texting, –Send images, –Send videos, –Video call, –Voice messages; : –Thumb-up, –Comment, –Refresh, –Share, –Repost.

$89.8 \pm 2.2\%$). We found that the launching stage of most apps in the category "Social Network" involves fewer animations, which makes them more difficult to be recognized compared with apps that perform launching animations from other categories (*e.g.*, B&N NOOK, Duolingo).

Effectiveness of in-app activity recognition. In $\mathcal{D}_{App}$, we also collect magnetic traces when performing five application-specific activities in five popular apps (YouTube, PayPal, WhatsApp, Facebook, and Spotify) and implement the CNN-based classification model for in-app activity recognition. For example, in YouTube, we evaluate **BankSnoop** in recognizing activities including *play video*, *forward*, *backward*, *pause video*, and *next video*. Fig. 3.12a–Fig. 3.12e present the confusion matrices of the in-app activity recognition results, where we find **BankSnoop** achieves an accuracy of 85.2%, 84.0%, 86.2%, 82.2%, and 81.4% in recognizing the five application-specific activities of the evaluated five mobile apps, respectively. Therefore, we demonstrate the effectiveness of **BankSnoop**, which accurately recognizes not only the launching app but also the fine-grained in-app activities.

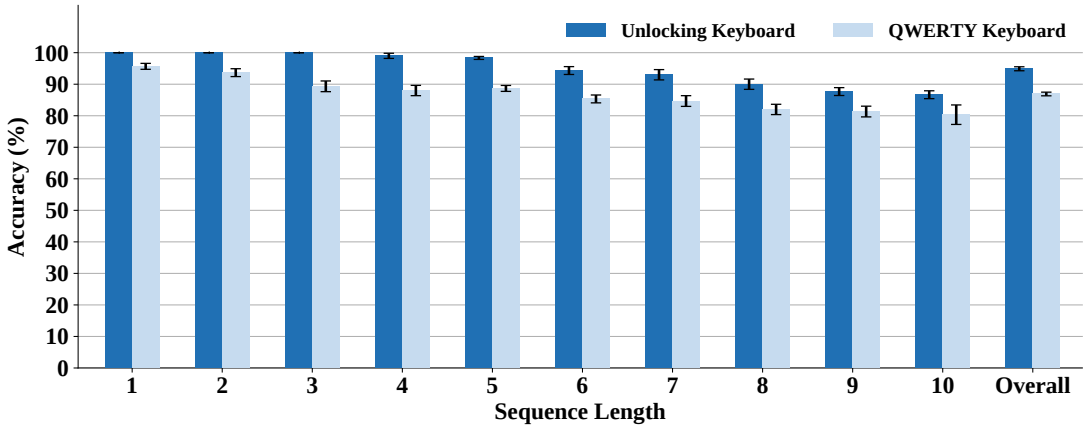
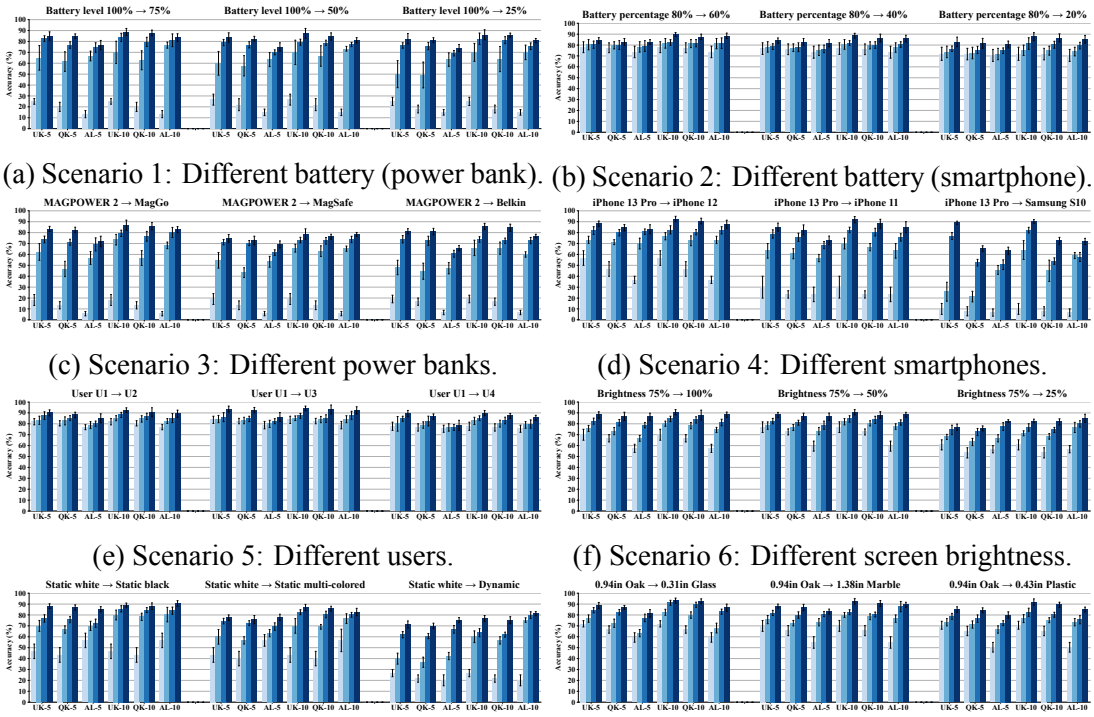


Fig. 3.13 Keystroke uncovering results of the unlocking and the QWERTY keyboards with top-10 candidates.

Effectiveness of keystroke uncovering. The evaluation of `BankSnoop`'s performance on keystroke uncovering is conducted on two datasets,*i.e.*, $\mathcal{D}_{UK}$ and $\mathcal{D}_{QWERTY}$, that are collected from the unlocking keyboard and the QWERTY keyboard. We randomly generate three sequences of numbers and characteristics for each keyboard with lengths ranging from one to ten[1]. Each randomly generated sequence is repeated for 100 times (*e.g.*, three examples of testing cases in length one of the QWERTY keyboard are "c", "o", and "e"). Fig. 3.13 shows the evaluation results on the random sequence testing set. We generate the top-10 candidates of the predicted sequence and obtain the corresponding accuracy if one of the top-10 candidates is correct. The overall accuracy of uncovering sequences in the length of one and ten are 94.9%, and 86.9%, respectively. The top-10 accuracy decreases as the sequence length increases, whereas the keystroke uncovering accuracy is still comparable to other works [30, 26, 34].

3.5.4 Few-shot Learning Evaluation

Baselines. We compare our proposed few-shot learning module with three baselines: *(i)* Source-only (*SO*): we use the model trained from only the source dataset $\mathcal{D}_{\mathcal{S}}$ and evaluate its performance on the target dataset $\mathcal{D}_{\mathcal{T}}$ directly with no adaptation, *(ii)* Target-only (*TO*): we use the few samples (*e.g.*, five or ten) from $\mathcal{D}_{\mathcal{T}}$ to train the CNN-based neural network and evaluate it with the rest samples of $\mathcal{D}_{\mathcal{T}}$, and *(iii)* Transfer-convolutional (*TrC*): Transfer convolutional [110] is one of the most state-of-the-art transfer learning methods for domain adaptation, which assumes that the upper layers' representations of similar problems are transferable [111, 112]. In practice, we freeze



(a) Scenario 1: Different battery (power bank). (b) Scenario 2: Different battery (smartphone).

(c) Scenario 3: Different power banks. (d) Scenario 4: Different smartphones.

(e) Scenario 5: Different users. (f) Scenario 6: Different screen brightness.

(g) Scenario 7: Different wallpapers. (h) Scenario 8: Different table materials.

Fig. 3.14 Few-shot learning module evaluation results (5-shot and 10-shot) in different scenarios. UK: unlocking keyboard keystroke accuracy. QK: QWERTY keyboard keystroke accuracy. AL: app launching recognition accuracy. -K: with K-shot learning, *e.g.*, UK-5 means 5-shot accuracy of unlocking keystroke recognition. – SO, – TO, – TrC, – Our method.

the convolutional layers of the trained model based on $\mathcal{D}_{\mathcal{S}}$ and then fine-tune the fully-connected layers only with the few-shot samples from $\mathcal{D}_{\mathcal{T}}$. Below, we evaluate the adaptation of `BankSnoop` in eight different scenarios.

Scenario 1: Different battery levels of a power bank. To evaluate the performance of `BankSnoop` in different battery levels of the power bank, we collect datasets $\mathcal{D}_{UK}$, $\mathcal{D}_{QWERTY}$ and $\mathcal{D}_{App}$ at the battery level 25%, 50%, 75%, and 100%, and then use datasets collected at the battery level of 100% as the $\mathcal{D}_{\mathcal{S}}$ to build the base model and use the rest samples as $\mathcal{D}_{\mathcal{T}}$. Fig. 3.14a presents the performance of the few-shot learning module at the battery levels of 25%, 50%, and 75%. We can observe that the performance of app launching and keystroke recognition under *SO* is the worst (lower than 25%). The target-only (*TO*) and transfer-convolutional (*TrC*) methods individually improve accuracy by approximately 35% and 45%, but their performance is lower than 80% in most of the cases. Among all scenarios, our proposed few-shot learning

approach achieves the best performance. Specifically, it improves the recognition accuracy of app launching, unlocking and QWERTY key pressing to 75.2%, 83.7% and 82.8% in the 5-shot cases, and 81.8%, 87.5%, and 86.1% in the 10-shot cases. The results demonstrate that **BankSnoop** *can quickly adapt to different battery levels of the wireless charging power bank while maintaining a high accuracy with few samples.*

Scenario 2: Different battery percentages of a smartphone. Previous wireless charging side-channel attacks [1, 10] show that the smartphone battery percentage can impact the model performance. We evaluate the adaptation ability of **BankSnoop** across different smartphone battery percentages by using the datasets collect from the iPhone 13 Pro charged by the EGO MAGPOWER 2 at battery percentage 80% as $\mathcal{D}_{\mathcal{S}}$ and datasets collected from 60%, 40%, and 20% as $\mathcal{D}_{\mathcal{T}}$. Fig. 3.14b presents the evaluation results, where we find the overall activity recognition accuracy decreases to 76.4%, 75.4% and 71.4% when applying models trained from smartphone battery percentage 80% to 60%, 40%, and 20%. Our few-shot learning module improves the accuracy to 83.3%, 82.9%, and 81.7% in the 5-shot cases, and 88.5%, 87.0%, and 86.5% in the 10-shot cases, which also performs better than the three baselines. The results demonstrate the *practicality of deploying* **BankSnoop** *to launch attacks at different smartphone battery percentages with few-shot learning.*

Scenario 3: Different wireless charging power banks. Different wireless charging power banks may present dissimilar coil whine and magnetic signal patterns of a similar task due to the different coil parameters (*e.g.*, coil turns, materials). We evaluate **BankSnoop**'s domain adaptation between different power banks by utilizing datasets of P_1 as $\mathcal{D}_{\mathcal{S}}$ and datasets collected from the other three commodity power banks (P_2–P_4) as $\mathcal{D}_{\mathcal{T}}$. Fig. 3.14c shows the evaluation results of the few-shot learning module with the three power banks. The recognition accuracy of app launching, unlocking passcode and the QWERTY keystroke has been enhanced from lower than 20% (*SO*) to 68.9%, 79.8%, and 78.7% in the 5-shot cases, and 79.3%, 83.7%, and 82.2% in the 10-shot cases, which outperforms about 25% and 8% than the TO and TrC methods. The results indicate that *it is practical for* **BankSnoop** *to attack different power banks and achieve promising accuracy.*

Scenario 4: Different smartphone models. Different smartphones have different configurations of the secondary coil parameters and battery volumes (*e.g.*, iPhone 12: 2815mAh, iPhone 13 Pro: 3095mAh), which results in different patterns of the induced

current changes by user activities. Therefore, prior attacks on smartphones [1, 3, 2, 10] usually trained multiple deep learning models to ensure model performance across different smartphones. Instead, we implement the proposed few-shot learning method by selecting samples collected from S_1 as the $\mathcal{D}_{\mathcal{S}}$ and other datasets collected from another three smartphones (S_2–S_4) as $\mathcal{D}_{\mathcal{T}}$ (charging by EGO MAGPOWER 2). Fig. 3.14d shows the results of adaptation from iPhone 13 Pro (S_1) to other two iPhone models (S_2, S_3), and our few-shot learning method improves the recognition accuracy of app launching, and keystrokes of two keyboards to 78.0%, 86.6%, and 83.5% in the 5-shot cases, and 86.4%, 92.2%, and 89.4% in the 10-shot cases. In particular, the accuracy decreases drastically when we directly apply the model (*SO*) trained for iPhone 13 Pro (S_1) to a Samsung S10 (S_4), which has totally different layouts of soft keyboards. Nevertheless, our method also achieves an accuracy of 72.5% in app launching recognition, 90.2%, and 71.88% in unlocking and QWERTY keyboards' keystrokes recognition. The results demonstrate that **`BankSnoop`** *achieves fast adaptation to smartphones of different platforms (iOS and Android)*.

Scenario 5: Different users. Since smartphone users may have distinctive typing patterns (*e.g.*, speed and movement), we recruit a total of four volunteers (note as U_1, U_2, U_3, and U_4) to join this study (IRB approved) and collect data for evaluation (iPhone 13 Pro charging with EGO MAGPOWER 2) to investigate the impact of different users. Then, we use the dataset U_1 as the $\mathcal{D}_{\mathcal{S}}$ and other three datasets (U_2, U_3, and U_4) as $\mathcal{D}_{\mathcal{T}}$. Fig. 3.14e shows the results of cross-user evaluations with different approaches. We find the overall activity recognition accuracy decreases to 79.7%, 81.6%, and 76.6% when applying the trained models of $U1$ to $U2$, $U3$, and $U4$. The few-shot learning approach improves the accuracy of app launching, unlocking, and QWERTY keystrokes recognition to 83.3%, 91.0%, and 88.9% in the 5-shot cases, and 89.0%, 92.2%, and 90.3% in the 10-shot cases. The results show that *the few-shot learning method improves the models' domain adaptation performance in cross-user evaluations.*

Scenario 6: Different screen brightness. Recent studies have revealed that the brightness of the touchscreen dominates most of the battery consumption [113, 114]. Hence, the brightness might impact the performance of recognizing user activities, especially when the screen brightness varies greatly (*e.g.*, $75\% \rightarrow 25\%$). We collected data from four different brightness (25%, 50%, 75% and 100%) when an iPhone 13 Pro is charging by the EGO MAGPOWER 2. Then we set the dataset of brightness 75% as

the $\mathcal{D}_{\mathcal{S}}$ and other three brightness datasets as the $\mathcal{D}_{\mathcal{T}}$. Fig. 3.14f presents the evaluations of few-shot learning performance in different screen brightness. The accuracy of app launching, unlocking, and QWERTY keystrokes recognition can be enhanced to 83.8%, 84.0%, and 82.5% in the 5-shot cases, and 86.9%, 87.5%, and 85.8% in the 10-shot cases. The results show that **`BankSnoop`** *can quickly adapt to varying brightness conditions.*

Scenario 7: Different wallpapers. Commodity mobile devices' screens are typically buttons with blurred backgrounds. The background picture (*a.k.a.* wallpaper) is displayed by numerous RGB pixels on the OLED touch screen that can exhibit different colors, which induce different power consumption. We consider reducing the wallpapers' impact on **`BankSnoop`** while using fewer data samples and we also explore not only static but also dynamic wallpapers. In practice, we use the data collected from pure white wallpapers as $\mathcal{D}_{\mathcal{S}}$ and other datasets collected from pure black, multi-colored, and dynamic wallpapers as $\mathcal{D}_{\mathcal{T}}$ ($P1 \times S_1$). Fig. 3.14g shows the results of adapting the trained model of static white wallpapers to static black and multi-colored wallpapers, where the recognition accuracy rates are 81.6%, 82.8%, and 81.3% in the 5-shot cases, and 86.7%, 88.0%, 86.9% in the 10-shot cases. Regarding the dynamic wallpaper cases, the dynamic animation adds extra noise to the power consumption of the touchscreen, which results in the worst adaptation performance (*SO* lower than 30%). In addition, our method still achieves an average 77.7% accuracy in the dynamic wallpaper cases, which makes **`BankSnoop`** *adaptive to the vary of different wallpapers, which can realize higher performance with better screen-noise cancellation methods or more shots.*

Scenario 8: Different table surfaces. In our experiment settings, the properties of the table surface (*e.g.*, thickness, materials) may impact the performance of **`BankSnoop`**. Hence, we collect data by placing devices ($P_1 \times S_1$) on three other table surfaces for evaluation: 0.31in (0.8cm) glass, 1.38in (3.5cm) marble, and 0.43in (1.1cm) plastic. Similarly, we use data collected from the oak table as $\mathcal{D}_{\mathcal{S}}$ and evaluate the adaptation performance on datasets of other table surfaces ($\mathcal{D}_{\mathcal{T}}$). Fig. 3.14h shows the evaluation results in three different table surfaces. Activity recognition accuracy, such as unlocking key-press decreases around 25% due to the attenuation of the inductive electromagnetic field. By utilizing few-shot learning, the accuracy of the aforementioned three activities reaches 81.5%, 87.3%, and 86.0% in 5-shot cases, and 87.0%, 92.5%, 90.8%

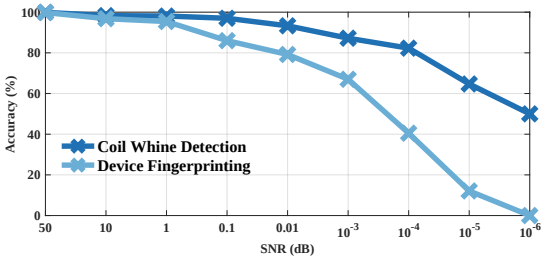


Fig. 3.15 Impact of environmental noise at different signal-to-noise ratios (SNRs).

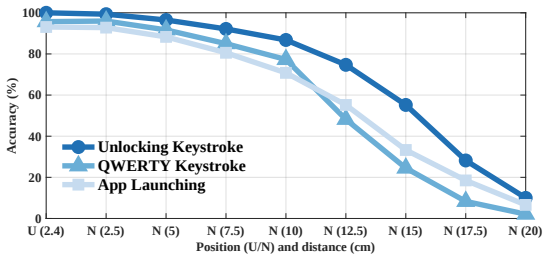


Fig. 3.16 Impact of position and distance (U–underneath, N–near).

in 10-shot cases. The results show that *table surface matters in such a contactless attack, whereas the proposed few-shot learning method still performs well.*

3.6 Discussion

3.6.1 Analysis of Other Impact Factors

Impact of environmental noise. To investigate the impact of environmental noise, we further collect samples (iPhone 13 Pro charged by EGO MAGPOWER 2) with high-frequency environmental noise (*e.g.*, Gaussian white noise [3]) at different signal-to-ratio (SNR) levels. Fig. 3.15 presents the results of coil whine detection and device fingerprinting under noise SNR ranging from 10^{-6} to 50, where we find `BankSnoop`'s performance decreases as we enhance the strength of the environmental noise. In particular, when the strength of the environmental noise is over $10^4\times$ of the coil whine (*i.e.*, SNR $= 10^{-4}$), the performance of `BankSnoop` degraded drastically (*i.e.*, device fingerprinting accuracy $< 40\%$) as such high-frequency noise dominates the captured audio signals.

Impact of position and distance. In practice, an attacker can place the disguised attacking device near the victim's charging smartphone at different distances. To understand the impact of the position and distance, we conducted experiments by placing the attacking device near the targeted charging devices at different distances. Fig. 3.16 presents the evaluations at a distance ranging from 0.98in (2.5cm) to 7.88in (20cm). Although `BankSnoop` achieves similar promising accuracy in different positions (underneath the table or near the smartphone), the overall performance decreases as the changes in the magnetic field can be difficult to monitor when the distance increases. In particular, when the distance is 20cm, `BankSnoop`'s performance decreases to lower

than 20% insufficient strength of the magnetic field disturbance, which remains undetectable by the attacking device.

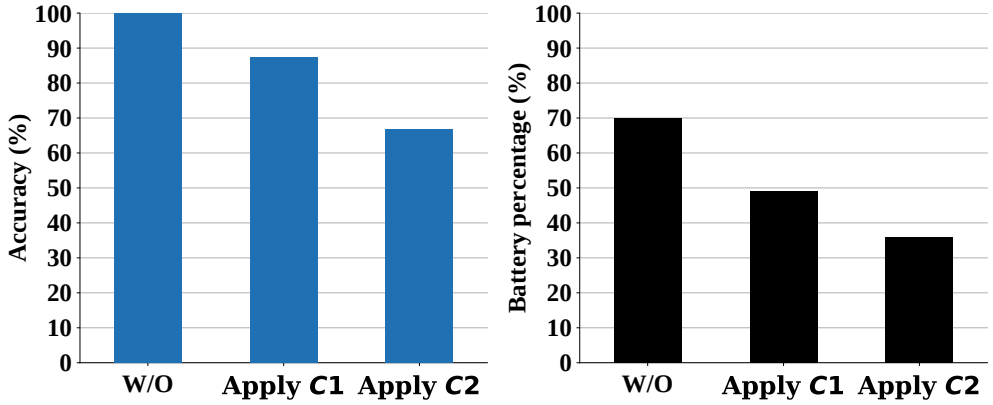
3.6.2 Limitations and Future Works

We have implemented `BankSnoop` to demonstrate the feasibility of the reported contactless side channel. While the results are promising, there still exist several limitations in the current work. First, `BankSnoop` is evaluated by attaching the attacking device underneath the table or putting it next to the target power bank for the proof of concept. Our work has not evaluated its performance in other possible scenarios, such as users holding the charging devices in their hands and performing activities on the run. Theoretically, `BankSnoop` is feasible to apply to those scenarios by adjusting the position and distance of the attacking device within a certain range to capture traces, whereas it inevitably increases the difficulty of launching attacks. Second, we consider a close and practical attacking distance in `BankSnoop` to demonstrate the feasibility because the two physical phenomena will attenuate as the distance increase, which requires tuning the models to adapt to a longer distance. We push these analyses to our future works.

3.7 Defense Methods

C1: Shielding magnetic field. One countermeasure to defend against attacks from `BankSnoop` is to prevent the magnetic traces from being eavesdropped. For example, manufacturers could add thicker cases to commodity wireless charging power bank products to shield the magnetic field to an undetectable degree [115]. Hence, the attacker needs to put the attacking device much closer to the victim or use more sensitive sensors to capture the magnetic traces, which inevitably increases the difficulties and costs of using `BankSnoop` to launch side-channel attacks.

C2: Signal obfuscation. Another countermeasure is to apply signal obfuscation mechanisms in the charging coils to generate indistinguishable current patterns so that the attacker cannot use the collected magnetic traces for user privacy inference. In practice, one can add random current noises (*e.g.*, Gaussian white noises [3]) to the primary coil or utilize a different charging protocol that dynamically switches the frequency and amplitude of the coil current [61] to obfuscate the captured signals. In addition, placing



(a) Recognition accuracy.

(b) Battery after charging.

Fig. 3.17 Results of applying $C1$ and $C2$ to `BankSnoop`.

other charging devices in the vicinity can create extra magnetic fields and obfuscate the captured magnetic field disturbances.

Implementation. We implement $C1$ by wrapping a 0.5 cm thick insulation shield [116] to the power bank and implement $C2$ by leveraging the RIGOL DS 1052E signal generator [117] and then measure the unlocking key-press accuracy as well as the charging efficiency (start from 10%). Fig. 3.17a and Fig. 3.17b show the result, where we know even $C1$ and $C2$ could defend against `BankSnoop`, they also impact the charging efficiency. That is, *C1* also increases the distance of the charging coils and *C2* results in fluctuated coil currents, which both reduce the power transmission efficiency.

3.8 Related Works

Wireless charging attacks. Qi protocol has become the de-facto wireless charging standard for mobile devices [8]. Nevertheless, recent researches reveal security vulnerabilities of Qi-certified wireless charging systems. Cour *et al.* [1] presented a website fingerprinting attack on wireless chargers from its current traces in the power line, which requires a stable charging voltage and a high battery level of the smartphone (*e.g.*, $> 80\%$). Wu *et al.* [3] used a hidden coil to obtain induced current for hijacking the battery and identifying app activities. Moreover, EM-Surfing [2] utilized the induced voltage of an external resistor to monitor privacy leakages, *e.g.*, app usage and keystrokes. Additionally, recent studies have shown that attackers can compromise the wireless charger to achieve voice injections [64, 65]. `BankSnoop` addresses the limi-

tations in these prior works to launch contactless and end-to-end side-channel attacks that achieve fine-grained user privacy inference and realize fast adaptation.

Magnetic side-channel attacks. Recent years have witnessed the development of studies relevant to magnetic-based side-channel attacks. For instance, MagEar [118] utilizes the magnetic flux from the victim's earphone speaker to perform audio eavesdropping attacks. MagSnoop [119] injects malware to capture the sounds in a magnetic secure transmission (MST) process (*e.g.*, Samsung Pay) to recover the tokens of a credit card. In addition, electromagnetic (EM) emanation can be exploited to extract secret keys [69], reconstruct model architectures [66, 120], uncover screen messages [56], keystrokes [121, 26], and 2D fingerprints [122] or camera images [123]. Likewise, `BankSnoop` has demonstrated the feasibility of exploiting two magnetic-induced phenomena to attack wireless charging power banks.

Few-shot learning works. Research efforts [43, 124, 125] presented more practical website fingerprinting methods by exploiting few-shot learning approaches such as Siamese Networks [126] and Triplet Networks [127]. Furthermore, relevant mobile sensing works [128–133] also utilize few-shot learning based on the concept of meta-learning. To the best of our knowledge, `BankSnoop` is the first work that leverages few-shot learning to achieve a practical and domain-adaptive power-based attack.

3.9 Summary

In this chapter, we report the same wireless-charging side channel in wireless charging power banks that can be exploited to launch contactless attacks to infer sensitive information from the charging smartphone, which leverages the coil whine and the magnetic field disturbance stemming from the wireless charging process. We have designed and implemented `BankSnoop`, an attack framework to demonstrate the feasibility of this new side-channel attack. To the best of our knowledge, it is the first attack on wireless charging power banks. Our extensive evaluation suggests that `BankSnoop` is effective in recognizing app launching/in-app activities and uncovering user keystrokes, and the few-shot learning module enables it to adapt to different scenarios while maintaining high accuracy.

Chapter 4

Contactless Side Channels in Multi-Port Chargers

Multi-port chargers, capable of simultaneously charging multiple mobile devices such as smartphones, have gained immense popularity and sold millions of units in recent years. However, this charging-targeted feature can also pose security and privacy risks by allowing one of the simultaneously charging devices to communicate with another one if not properly designed and implemented as these devices are actually interconnected. Unfortunately, such risks have not been thoroughly investigated and we have identified a novel attack surface in the circuit design of multi-port chargers, which allows an adversary to exploit one port to *(i)* eavesdrop on the activities of other devices being charged and *(ii)* inaudibly inject malicious audio commands if the charging device supports voice assistants and USB-C interface.

In this chapter, we design and implement a novel framework, `XPorter`, to analyze and demonstrate the uncovered security and privacy threats in multi-port chargers. Specifically, it leverages the changes in the voltage signals on one neighbor port to monitor the voltage changes of the charging port induced by various user activities, including recognizing the running apps and uncovering keystrokes. Moreover, `XPorter` can also achieve inaudible audio injection attacks from the neighbor port to the charging mobile device via the USB-C interface. We extensively evaluate the effectiveness of `XPorter` using five commodity multi-port chargers and five mobile devices. The evaluation results show its high effectiveness in recognizing the launching of 20 mobile apps (88.7%) and uncovering unlocking passcode (98.8%). Furthermore, `XPorter` achieves 100% success rates in inaudible audio injection attacks on three voice assis-

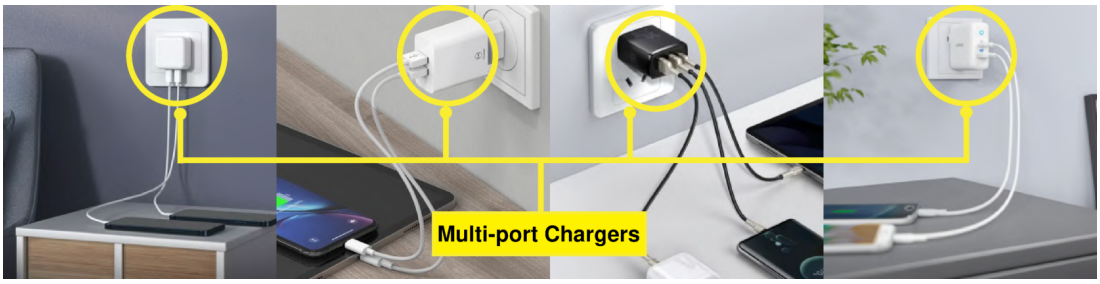


Fig. 4.1 Illustration of multi-port chargers in real-life scenarios. A multi-port charger can support battery charging for multiple mobile devices simultaneously.

tants. In addition, our study also shows that `XPorter` is resilient to various impact factors and presents the potential of attacking multiple victims.

4.1 Introduction

The recent explosive growth of mobile devices, such as smartphones and tablets, has fostered various styles and capabilities of battery-charging accessories whose relevant market has been projected to reach approximately $1,580$ million US dollars by 2022 [134]. Multi-port chargers are one of those representative accessories that provide multiple ports (*e.g.*, two or more USB-C/USB-A ports) for users to charge multiple mobile devices simultaneously. This type of charger is becoming extremely popular over the past five years because people have a growing demand for charging multiple devices with varying charging specifications [134]. Fig. 4.1 shows four typical real-life scenarios that demonstrate the usage of commodity multi-port chargers.

However, this multi-device charging feature exposes an attack surface that could allow one device to conduct malicious actions on other devices when they are charging simultaneously. This vulnerability stems from the fundamental design of the multi-port charger, where all charging ports are connected in parallel and share the same voltage. Consequently, any voltage change in one port could affect all other parallel-connected ports, making it possible to launch attacks on one port to *eavesdrop or inject voice commands* into other connected devices. Previous research has revealed that the voltage changes of a charging mobile device can reveal sensitive information, such as button presses, keystrokes on the unlocking screen, and various running apps on smartphones [10, 135, 1, 2], and these voltage changes can also be exploited to manipulate the charging device's voice assistant, enabling the injection of malicious voice commands and potentially leading to the interpretation of incorrect information [12].

Unfortunately, these severe security and privacy concerns associated with multi-port chargers have been largely neglected. One possible reason for this oversight is that multi-port chargers appear to be immune to these security issues since they are not primarily designed for data transfer, which is an essential attack surface for eavesdropping and voice command injection attacks on other target devices (*e.g.*, USB hubs [11, 136, 137]). Therefore, we aim to *fill this knowledge gap* by analyzing two typical attacks, *i.e.*, *(i)* eavesdropping attacks and *(ii)* inaudible audio injection attacks, as the first step towards shedding light on these previously overlooked threats posed by multi-port chargers and contribute to enhancing their security measures.

We design and implement a novel framework, `XPorter`, to facilitate our study on eavesdropping and audio injection attacks stemming from the communication across (*X*) charging ports of a multi-*PORT* charg*ER*. Specifically, in regard to the eavesdropping attack, `XPorter` first detects the leakage of the voltage signals from one of the neighbor ports and then conducts signal processing to obtain the informative voltage clips to training models to recognize user activities to infer sensitive information on other charging devices. On the other hand, in respect of the inaudible audio injection attack, `XPorter` leverages the USB-C charging interface to activate the voice assistant of the charging device while bypassing the speech verification system and then injects malicious voice commands through a compromised multi-port charger.

We have implemented `XPorter` with a custom-built attacking device to demonstrate the feasibility of the aforementioned two attacks. As a proof of concept, first, `XPorter` aims to eavesdrop on three particular types of sensitive information, *i.e.*, unlocking passcode, launching apps, and sensitive keystrokes, from the charging device due to the fundamental design flaw existing in multi-port chargers. Specifically, we use the attacking device to collect the leaked voltage signals from 20 popular mobile apps and two soft keyboards (*i.e.*, unlocking numeric keyboard and full-size QWERTY keyboard) running on five mobile devices (*i.e.*, iPhone 13 Pro, iPhone 11, OnePlus 10 Pro, Google Pixel 4, and iPad Pro 2019) that are charging with five commodity multi-port chargers from different vendors (*i.e.*, UGREEN 40W dual USB-C charger, Anker 65W dual USB-C charger, Belkin 45W dual wall charger, Apple 35W dual USB-C charger, and ROMOSS 10W dual USB-A charger). Our evaluation results of the eavesdropping attacks show high effectiveness of `XPorter` where it achieves 98.8% in recognizing the unlocking passcode, 88.7% in fingerprinting the 20 mobile apps, and 83.0% in uncovering the alphabetic keystrokes of a QWERTY keyboard. Moreover, it also demonstrates that `XPorter` is resilient to various practical impact factors, including different multi-port chargers, mobile devices, and battery levels of the charging devices. In

addition, we show the potential of eavesdropping on multiple victims' activities and provide efficient countermeasures to smooth out the voltage leakages to defend against `XPorter`.

In respect of demonstrating the inaudible audio injection attack, we evaluate it over three commodity voice assistants, including Apple Siri, Google Assistant, and OnePlus Breeno. Specifically, the attacking device can receive the voice commands remotely from the adversary by Bluetooth and then modulate them to injectable audio clips. Next, it leverages the audio pin of the USB-C interface to automatically activate the voice assistant of the charging smartphone while bypassing the speech verification mechanism that is widely deployed in commodity mobile devices. Finally, the modulated audio clips that contain malicious voice commands would be injected into the charging device to obtain more private information about the device's owner or manipulate the voice-controllable IoT devices (*e.g.*, Apple HomeKit). The extensive evaluation shows that **`XPorter`** achieves 100% success rate in activating the three voice assistants, injecting different voice commands, and 12 trials of end-to-end injection attacks. A demo video is available at https://youtu.be/X9HY9mIDTGw.

Contributions. We summarize the contributions as follows:

- **A novel attack.** We introduce a new attack vector that can be exploited to attack mobile devices charged by a commodity multi-port charger. It leverages the changes of the voltage leakage between the neighbor USB charging ports to reveal sensitive information and the characteristics of the USB-C interface to inject malicious voice commands to charging devices across ports.
- **A new framework.** We propose and implement a new attack framework, **`XPorter`**, to demonstrate the feasibility of the proposed attacks. Specifically, it exploits the leakage of the voltage signal to recognize the unlocking passcode, running apps, and sensitive keystrokes. In addition, it exploits the audio pins of the USB-C interface to inaudibly activate the voice assistant and inject malicious voice commands from the neighbor USB-C port to other charging devices.
- **Comprehensive evaluation.** We comprehensively evaluate the effectiveness of **`XPorter`** with five commodity multi-port chargers and five mobile devices. The results indicate that it performs effectively in eavesdropping on various user activities. Moreover, **`XPorter`** achieves a 100% success rate in activating different voice assistants and inaudibly injecting different voice commands. In addition, we also show the potential of attacking multiple victims and further provide effective countermeasures.

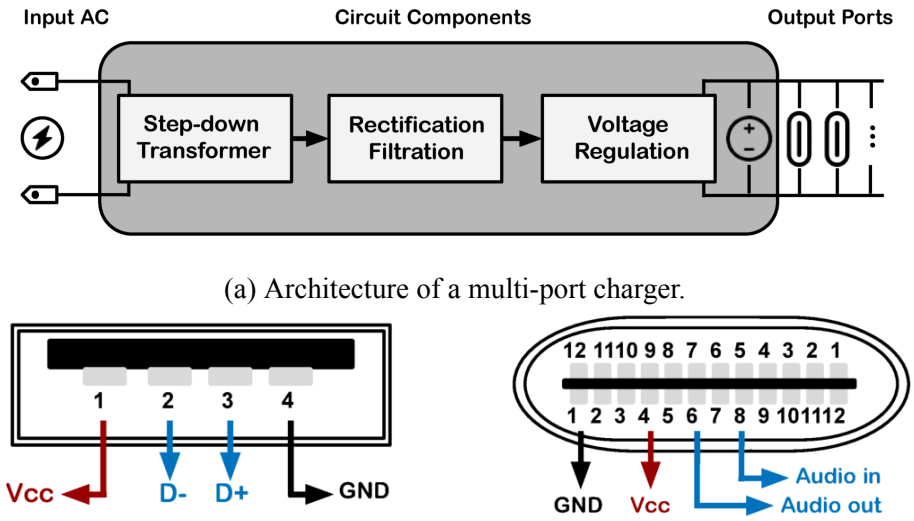
4.2 Preliminary

4.2.1 Multi-port Charger

Nowadays, the multi-port charger allows users to charge multiple mobile devices (*e.g.*, smartphones, tablets) at the same time. Fig. 4.2a shows the basic architecture of a typical multi-port charger, which includes an AC voltage step-down transformer, a rectification circuit, a filtration circuit, a voltage regulation module, and multiple outputs charging ports. First, the step-down transformer converts the high input AC voltage (*e.g.*, 110 V AC) to low AC voltage (*e.g.*, 9 V AC). Then, the rectification circuit removes the negative part of the downgraded AC voltage to produce a partial DC with oscillations, and a filtration circuit suppresses such oscillations to generate a proper DC voltage. Finally, a voltage regulation module eliminates other noise and outputs the DC voltage (*e.g.*, 5 V DC) to the charging ports for powering multiple mobile devices. In particular, as the multi-port charger needs to power two or more devices simultaneously, the output charging ports are parallel connected together so that each of them obtains the same voltage (*e.g.*, 5 V). That is, in a charging process, the voltage changes on one port can induce voltage changes in its neighbor ports.

4.2.2 USB Type-A and USB Type-C Ports

Most commodity multi-port chargers adopt two types of Universal Serial Bus (USB) standards: USB Type-A (*a.k.a.*, USB-A) and USB Type-C (*a.k.a.*, USB-C). The USB-A port is commonly used in different mobile device accessories (*e.g.*, charger, USB-hub), and Fig. 4.2b shows its structure, where two pins (pin 1 and pin 4) are used for charging the battery, and two pins support data transfer. Moreover, the USB-C port has been widely deployed in most Android smartphones and will be mandatorily applied to all smartphones (including iPhone) sold in the European Union by the end of 2024 based on a newly passed legislation [138]. Fig. 4.2c shows the structure of a USB-C port that consists of 24 pins on two sides, and pins on both sides have the same functions because of its rotationally symmetrical structure. Therefore, users have no need to find the correct side to plug into the USB-C port. Furthermore, the USB-C port supports not only battery charging and data transmission but also audio input (pin 8) and audio output (pin 6). As both ports support battery charging, the power traces can be used for inferring user activities on the charging smartphone. In addition, due to the integrated features of USB-C ports, the smartphone is also threatened by potentially injecting inaudible voice commands, as we will demonstrate in this work.



(a) Architecture of a multi-port charger.

(b) USB-A port.

(c) USB-C port.

Fig. 4.2 Architecture of a multi-port charger and USB ports: (a) Circuit of a typical multi-port charger, (b) USB-A port (4 pins), and (c) USB-C port (24 pins on two sides).

4.3 Motivation, Principle and Threat Model

4.3.1 A Motivating Example

We present a motivating example of launching eavesdropping and audio injection attacks through a commodity multi-port charger in this subsection. That is, the user connects the smartphone to one port of the charger for battery charging, unlocks the smartphone with a password (*e.g.*, "1234"), and then launches the app WhatsApp to send a message to others (*e.g.*, "abcde"). This series of activities change the energy consumption of the smartphone battery and further changes the running current in the power line as well as the output voltage provided by the charger. As mentioned in § 4.2.1, the voltage changes in one port can induce voltage changes of other neighbor ports, and these changes present detectable and predictable features that can be exploited for inferring corresponding user activities. In addition, the attacker can exploit the integrated audio pin in the USB-C interface to activate the voice assistant (*e.g.*, Apple Siri) and then inject malicious audio commands (*e.g.*, "open the door").

In Fig. 4.3, we present the changes of voltages in both the user's charging port and a neighbor port when the user performs different activities. Specifically, we show the voltage changes of unlocking password input, app launching, and QWERTY keystrokes. As can be seen, both the voltages of the charging port (grey curve) and the neighbor port (blue curve) present distinctive and synchronized changes when pressing a but-

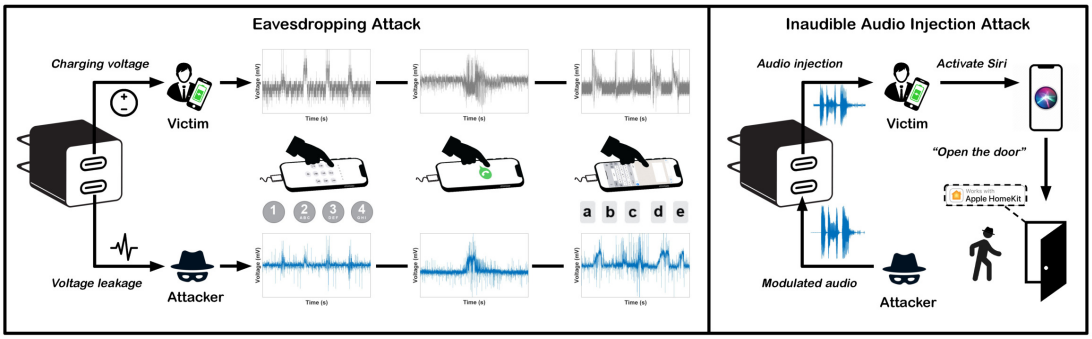


Fig. 4.3 Motivating example scenario of two attacks via **XPorter**. *(i)* Eavesdropping attack (left-hand): When the smartphone is being charged by a multi-port (2-port) charger, the victim performs various activities (*e.g.*, unlocking the smartphone, opening an app, and typing keystrokes), which induces voltage changes (grey color) on the charging port as well as the neighbor port. Meanwhile, the attacker acquires the voltage leakage (blue color) and utilizes it to uncover private information (*e.g.*, passcode, app usage, and sensitive keystroke input). *(ii)* Inaudible audio injection attack (right-hand): Based on a compromised multi-port charger, the attacker can achieve audio injection by activating the voice assistant of the victim's smartphone through the audio pin of the USB-C interface and then injecting malicious voice commands to obtain further user privacy (*e.g.*, "Where's my home?") or control the voice-controlled IoT devices (*e.g.*, "Open the door").

ton to unlock the smartphone, launch apps, or enter keystrokes. In addition, we also show the patterns at the audio input pin of the user's charging port and the neighbor port when we activate the voice assistant Siri and inject the malicious voice command "open the door" into it. As such, Siri will then follow the voice command to open the door of a smart home that is equipped with Apple HomeKit [139].

4.3.2 Fundamental Principles

Below, we illustrate the fundamental principles of voltage leakage and audio injection between two neighbor USB ports of a multi-port charger from the aspect of physics.

Voltage leakage. Due to the parallel-connected architecture of USB ports in multi-port chargers, which allows them to simplify the circuit design by sharing a common output DC voltage, a fundamental design flaw arises, resulting in voltage leakage across neighboring USB ports. In a common battery charging scenario, we denote the output voltage of the charging port as $V_c(t)$ and the voltage of another neighbor port as $V_x(t)$. As these two ports are parallel connected, their relations are shown in

Equation 4.1 as follows:

$$V_x(t) \propto C \cdot V_c(t), \tag{4.1}$$

where C is a mapping factor that reflects the $V_x(t)$ changes with the $V_c(t)$ based on the circuit design between the neighbor USB ports. Note that the magnitude and shape of $V_x(t)$ and $V_c(t)$ may be different, but the mapping factor C only depends on the design of the hardware circuit [140, 11]. That is, for a specific multi-port charger, C is a constant factor between the two neighbor USB ports.

We assume the load of the smartphone is $R_s(t)$ when being charged by a multi-port charger through a USB powerline. Based on Ohm's law, we can present the running current $I_c(t)$ for charging the smartphone in Equation 4.2:

$$I_c(t) = V_c(t)/R_s(t) \propto 1/R_s(t), \tag{4.2}$$

When the user performs different smartphone activities (*e.g.*, running apps, pressing buttons on keyboards), these activities induce different displays of lighter/darker pixels on an OLED touchscreen that consume different amounts of power [10], resulting in load changes $\Delta R(t)$ on the battery of the charging smartphone [2, 3]. As such, these load changes induce the changes of voltage $\Delta V_c(t)$ on the charging port, as well as voltage changes $\Delta V_x(t)$ the neighbor port because of the leakage across ports, which is shown in Equation 4.3:

$$\Delta V_x(t) \propto C \cdot \Delta V_c(t) \propto C \cdot I_c(t) \cdot \Delta R(t). \tag{4.3}$$

Therefore, it is feasible to exploit the voltage leakage of a neighbor USB port to monitor the voltage changes of the charging smartphone and further infer user privacy.

Inaudible audio injection. Since USB-C ports support audio transmission as discussed in §4.2.2, it is feasible to achieve an inaudible audio injection attack towards the voice assistant of the victim's smartphone. The first step is to activate the voice assistant, but most commodity smartphones have a speech verification mechanism that can deny the activation request of a non-owner activating command. Fortunately, the USB-C interface provides a solution to activate the voice assistant [12]. That is, commodity smartphones allow the wire control board of the earphone to activate voice assistant by pressing the button for nearly 1 to 2 seconds, which is also integrated into the functions of a USB-C port. Therefore, the attacker can manipulate the audio pin's

voltage changes to simulate a button-pressing event to inaudibly activate the voice assistant of the victim's smartphone while bypassing the speech recognition system.

After activating the voice assistant through the above method, one can inject a modulated audio signal that contains malicious voice commands to the victim's smartphone across the neighbor USB-C ports of a commodity multi-port charger. Specifically, the modulated audio signal $A(t)$ for injecting voice commands can be denoted as follows:

$$A(t) = \alpha \cdot x(t) + V_{offset}, \tag{4.4}$$

where $x(t)$ is the original audio clip containing the voice command, α is a factor to adjust the amplitude, and V_{offset} is an extra DC offset to compensate for the initial voltage of the port. Then an analog-to-digital converter (ADC) will take the modulated signal and convert it to a digital signal that can be recognized by the audio pin of the USB-C interface.

4.3.3 Threat Model

We consider a common scenario of using multi-port chargers to charge mobile devices (*e.g.*, smartphones) where victims connect their devices to the ports and perform different activities (*e.g.*, unlocking the smartphone, running apps). An attacker can share the multi-port charger with the victims and launch two types of attacks: *eavesdropping attack* and *inaudible audio injection attack*. Such a scenario is ubiquitous in public facilities and shared space, *e.g.*, offices and airports.

Eavesdropping. When launching an *eavesdropping attack*, the attacker monitors the voltage changes of a neighbor port and exploits the voltage traces to infer privacy-sensitive information, *i.e.*, *(i)* digits of the smartphone's unlocking password, *(i)* the victims' app usage and corresponding activities, and *(iii)* sensitive keystrokes of QWERTY keyboard. We assume the attacker can share the neighbor port of a multi-port charger with the victims, but *cannot* compromise *(i)* the commodity multi-port charger to install malicious firmware, *(ii)* the victims' USB power line to monitor current traces, and *(iii)* the victims' smartphone including malware installation.

Inaudible audio injection. When launching an *inaudible voice injection attack*, the attacker can use the USB-C interface to bypass the speech verification and activate the voice assistant (*e.g.*, Apple Siri, Google Assitant) of the victims' smartphones to inject modulated audio commands through the audio signal pin of the neighbor USB-C

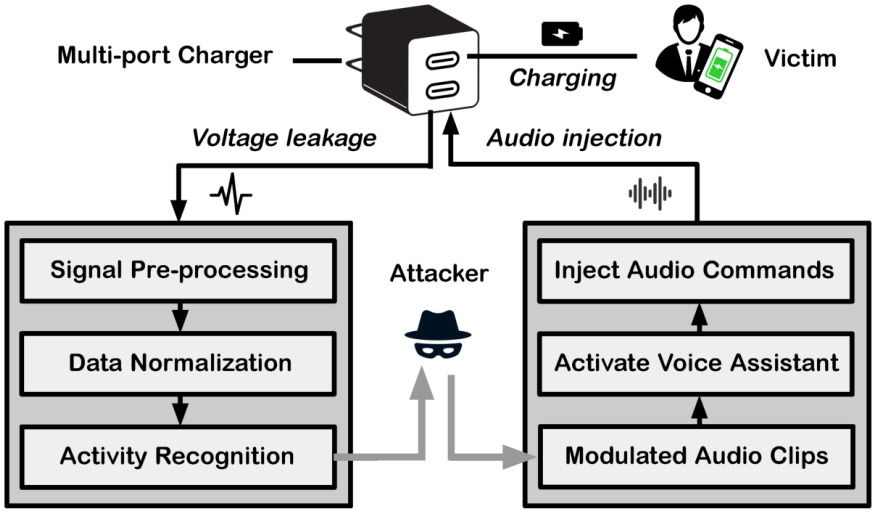


Fig. 4.4 Overview of `XPorter`.

port. As such, we assume that the attacker can first compromise a multi-port charger that has USB-C ports by connecting the audio pins of the two neighbor USB-C ports together. Then, the attacker shares the multi-port charger with the victims in a shared place and then leverages a customized attacking device to achieve the inaudible audio injections. In addition, the attacker can utilize speech synthesis [141] tools (*e.g.*, Google WaveNet [142]) to generate modulated audio commands.

4.4 Attack Framework

4.4.1 Overview of `XPorter`

Fig. 4.4 presents the overview of `XPorter` in launching the eavesdropping attack and audio injection attack. Specifically, an attacker first shares the multi-port charger with the victim and obtains the voltage leakage from a neighbor port. Then, the recorded voltage traces will be processed and normalized for user activity recognition to eavesdrop on privacy information, *i.e.*, unlocking passcode, running app activities, and in-app keystrokes. Moreover, the attacker can exploit the integrated audio pins in USB-C ports to inaudibly activate and inject modulated audio commands into the victim's charging smartphone to maliciously access the voice assistant systems (*e.g.*, Apple Siri, Google Assistant).

4.4.2 Eavesdropping Attack

Below, we present the design and implementation of `XPorter` in launching an eavesdropping attack, which consists of three components as follows: *(i)* signal pre-processing, *(ii)* data normalization, and *(iii)* activity recognition.

Signal Pre-processing. After obtaining the raw voltage signals, we design a signal processing algorithm to handle the acquired voltage leakage as shown in Algorithm 2. Specifically, `XPorter` first exploits a Savitzky-Golay (S-G) filter to remove high-frequency noise in the collected time-series signals (line 2-6) without distorting the signal shapes [33]. We then use the average values of the first one-second data as the DC offset and deduce this offset value in the following signals (line 7). Since the captured voltage signals contain both non-activity and activity-induced voltage changes, we apply a moving-variance window with a given threshold τ (*e.g.*, 0.05) to find the start and end indices of the activity patterns and then segment the signal with privacy information of specific user-smartphone interactions (line 8-18).

Data Normalization. To eliminate the impact of the varied output voltages when charging different mobile devices, we apply methods of data normalization on the segmented voltage signals. Specifically, we normalize the amplitude of the processed voltage signals to the range from 0 to 1 and utilize the decimation factor down-sampling algorithm [143] to reshape these voltage signals to fixed length vectors (*e.g.*, 128×1), and then leverage the dynamic time warping (DTW) algorithm [144] to generate the vectors that maintain the informative patterns for training deep learning models that can recognize fine-grained user activities. Specifically, the DTW algorithm maps output voltage signal $\mathcal{S}$ to the down-sampled $\mathcal{S}^{'}$ by optimizing all admissible paths from S_i to $S_i^{'}$ as shown in Equation 4.5:

$$DTW_q(S_i, S_i^{'}) = \min_{\pi \in \mathcal{P}(S_i, S_i^{'})} \Big(\sum_{(i,j)\in\pi} d(S_i, S_j^{'})^q \Big)^{(\frac{1}{q})}, \tag{4.5}$$

where π is the alignment path of a sequence of K-length index pairs from 0 to K as $((i_0, j_0), (i_1, j_1), ..., (i_{K-1}, j_{K-1}))$, $\mathcal{P}(S_i, S_i^{'})$ is the set of all admissible paths, $d(S_i, S_i^{'})$ is the Euclidean distance between $\mathcal{S}_i$ and $\mathcal{S}_j^{'}$, and q is the power constant.

Algorithm 2: Signal processing of eavesdropping attack

Input: $\mathcal{V} = [v_{c_1}(t_1), v_{c_2}(t_2), ..., v_{c_m}(t_m)]$: obtained signals from the voltage leakage. o, f: order and frequency of the S-G filter. τ: threshold of the variance.

Output: $\mathcal{S} = [S_1, S_2, ..., S_n]$: filtered voltage signal clips containing specific smartphone activities.

$\mathcal{V}^{'} \leftarrow [], \mathcal{S} \leftarrow []$ ▷ initialize the empty array to record filtered signals and segmented voltage signal clips.

$filter \leftarrow sgolayfilt(o, f)$ ▷ initialize an S-G filter with the given order o and the frequency f.

for *each signal* $v_{c_i}(t_i) \in \mathcal{V}$ **do**

  $v^{'}_{c_i}(t_i) \leftarrow filter(v_{c_i}(t_i))$

  $\mathcal{V}^{'} \leftarrow [v^{'}_{c_1}(t_1), v^{'}_{c_2}(t_2), ..., v^{'}_{c_i}(t_i)]$

$\mathcal{V}^{'} \leftarrow [v^{'}_{c_1}(t_1), v^{'}_{c_2}(t_2), ..., v^{'}_{c_m}(t_m)]$ ▷ the filtered signals.

$\mathcal{V}^{'} \leftarrow \mathcal{V}^{'} - average([v^{'}_{c_1}(t_1), \cdots, v^{'}_{c_f}(t_f)])$ ▷ deduct offset.

$window \leftarrow movvar(\tau, f/10)$ ▷ initialize an moving-variance window with the given threshold τ and size of $f/10$.

for *each filtered signal* $v^{'}_{c_i}(t_i) \in \mathcal{V}^{'}$ **do**

  $\mathcal{R}_{c_i}(t_i) \leftarrow window(v^{'}_{c_i}(t_i))$ ▷ obtain the time-variance signal from the moving-variance window.

  for *each* $r_i \in \mathcal{R}_{c_i}(t_i)$ **do**

    if $\forall r_j \in [r_i, r_{i+f/10}], r_j < r_{j+1}$ and $r_j > \tau$ **then**

      $k_{start} \leftarrow r_i$ ▷ obtain *start* index of the activity.

    else if $\forall r_j \in [r_i, r_{i+f/10}], r_j > r_{j+1}$ and $r_j > \tau$ **then**

      $k_{end} \leftarrow r_{i+f/10}$ ▷ obtain *end* index.

  $S_i \leftarrow [v^{'}_{c_i}(k_{start}), v^{'}_{c_i}(k_{start})]$ ▷ voltage signal clip that contains the specific activity.

  $\mathcal{S} \leftarrow [S_1, S_2, ..., S_i]$

$\mathcal{S} = [S_1, S_2, ..., S_n]$

Output voltage signal clips $\mathcal{S}$ that contain user activities.

To resolve this optimization problem, we need to obtain the quantity $R_{i,j}$ [145] between two timestamps i and j as:

$$R_{i,j} = DTW_q(S_{\rightarrow i}, S'_{\rightarrow j})^q, \tag{4.6}$$

where $S_{\rightarrow i}$ means the time-series voltages obtained up to timestamp i, and we can further obtain $R_{i,j}$ as Equation 4.5:

$$\begin{aligned} R_{i,j} &= \min_{\pi \in \mathcal{P}(S_{\rightarrow i}, S'_{\rightarrow j})} \sum_{(k,l)\in\pi} d(S_k, S'_l)^q \\ &\overset{*}{=} d(S_i, S'_j)^q + \min_{\pi \in \mathcal{P}(S_{\rightarrow i}, S'_{\rightarrow j})} \sum_{(k,l)\in\pi[:-1]} d(S_k, S'_l)^q \\ &\overset{**}{=} d(S_i, S'_j)^q + \min(R_{i-1,j}, R_{i,j-1}, R_{i-1,j-1}), \end{aligned} \tag{4.7}$$

where $*$ denotes the constraints on all admissible paths π, and we set the target length K as 128 and calculate the each $R_{n-1,m-1}$ to retrieve the corresponding $DTW_q(S_i, S'_i)$. After the data normalization process, we then collect the normalized data vectors as the input to train a deep learning classifier for fine-grained user activity recognition.

Activity Recognition. As the processed voltage signals are time series, `XPorter` adopts a one-dimensional convolutional neural network (CNN) with a Long Short-term Memory (LSTM) [146] layer to build a classifier for various activity recognition (*e.g.*, app launching, single key-pressing inference). Specifically, CNN-based neural networks are utilized in various side-channel attacks [10, 1] using one-dimensional time-series signals because the convolutional layers can capture both temporal and spatial features from time-series signals and achieve a promising classification accuracy [104]. Furthermore, as the CNN extracts multiple features from the voltage signal, we use an LSTM layer to learn the order dependence and identify these features.

The topology of our CNN-LSTM model consists of three convolutional layers followed by an LSTM layer, a fully-connected layer, and a softmax layer with a single output for each instance (*e.g.*, key, app). For the three convolutional layers, we use the ReLU as the activation function and add a max-pooling layer to reduce the dimension by half. Then, a flatten layer converts the extracted feature maps to one-dimensional vectors as the valid input for the LSTM layer. After the LSTM layer, a dropout layer with 50% dropout rate is added to regularize the network and prevent overfitting. Fi-

nally, the fully-connected layer and the softmax layer output the predicted class with the highest probability.

Implementation Details. In practice, we implement the first two components, signal pre-processing and data normalization by leveraging MATLAB R2022a Signal Processing Toolbox (version 3.0) that supports reliable toolkits. Then, we implement the CNN-LSTM neural networks for activity recognition in Keras 2.3 on the Tensorflow 2.0 framework. In the training stage, we set the batch size as 32 and use the cross-entropy loss and Adam optimizer with an initial learning of 0.01 and epoch of 100. In particular, the output shape depends on the corresponding task (*e.g.*, the number of apps and the number of keys on a keyboard). Specifically, we study 10 numeric buttons on the unlocking keypad of the touchscreen (10 classes), 20 different mobile apps (20 classes), and the alphabetic keys on the full-size QWERTY keyboard (26 classes).

4.4.3 Inaudible Audio Injection Attack

Apart from the eavesdropping attack, we also present the design and implementation of `XPorter` in launching an inaudible audio injection attack in this subsection. As mentioned in § 4.3.3, the attacker can simply compromise the multi-port charger by connecting the audio pins of the output ports together without modifying the packaging, which results in less suspicion for the victim. Then the attacker connects the attacking device (details in § 4.4.4) to the neighbor USB-C port and conducts three steps to achieve malicious audio injection: *(i)* audio modulation, *(ii)* voice assistant activation, and *(iii)* audio commands injection.

Audio Modulation. As discussed in §4.2.2, the audio signals obtained by the USB-C port are represented by the changing current and voltage of the audio pin. As such, the attacker should first convert the audio clips that contain malicious voice commands to modulated voltage signals and then inject these modulated voltage signals into the victim's smartphone. Specifically, we can exploit Equation 4.4 in §4.3.2 to implement the audio modulation from the audio clips to the recognizable voice commands. To achieve automatic audio modulation, we use an audio board with a Bluetooth module to receive the malicious voice command from the attacker remotely and modulate it to a voltage signal that can be received by the audio pins of the USB-C port and recognized by the voice assistant of the victim's mobile device. Moreover, we also apply a

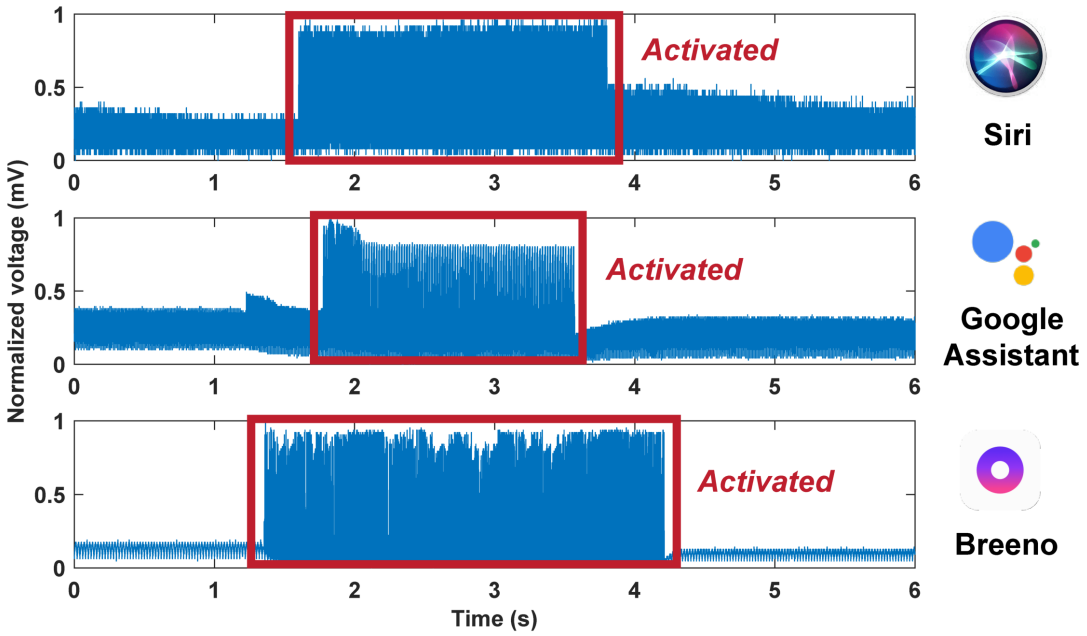
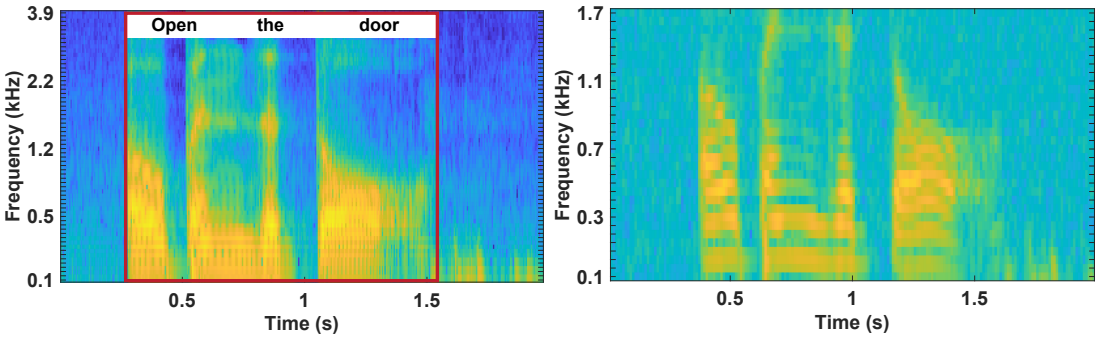


Fig. 4.5 Audio pin voltage signals when activating three commodity voice assistants (Apple Siri, Google Assistant, and Breeno) through the USB-C interface. The red boxes present the voltage changes when the voice assistants are activated.

differential amplifier module to adjust the amplitude of the modulated audio signal to obtain the best configurations for the injection attacks.

Voice Assistant Activation. Previously, inaudible audio injection attacks [28, 147] on smartphones' voice control systems require voice samples from authorized users to generate hotword commands (*e.g.*, "Hey Siri" or "Hello Google") through virtual microphones and speakers to activate the voice assistants. However, these replaying methods can easily be detected and prevented by state-of-the-art verification approaches [41, 148, 149]. Therefore, in §4.3.2, we introduced the headphone button-pressing event that can activate the voice assistant while bypassing the speaker verification system. To verify its practicality, we record the voltage signals of the USB-C audio pin when activating smartphone voice assistants and present the results in Fig. 4.5. In practice, we tested it on three commodity voice assistants (Apple Siri, Google Assistant, and OnePlus Breeno), and we can know that the voltage of the audio pin will boost to a high stage when the voice assistant is activated. In particular, we find that different voice assistants require different patterns of input voltage changes on the USB-C audio pin to activate themselves, *e.g.*, different lasting times and amplitudes.



(a) Modulated audio clip. (b) Injected signal of USB-C.

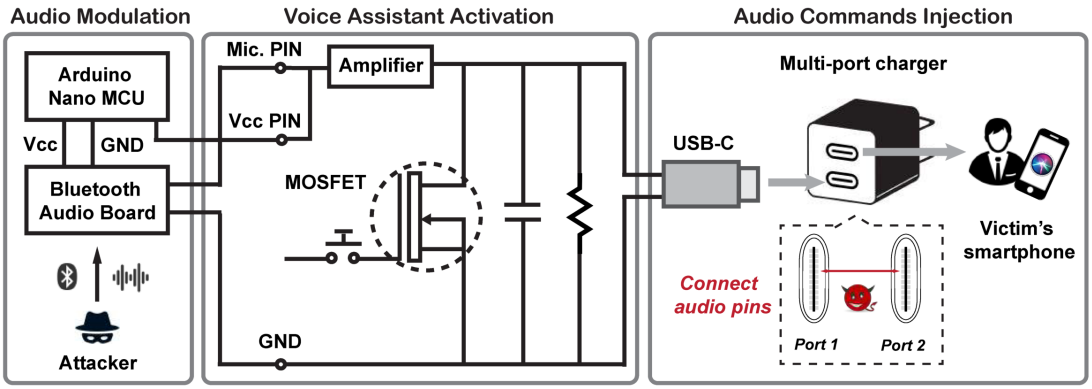
Fig. 4.6 Spectrograms of the modulated audio clip and the voltage signal of the USB-C audio pin when injecting the voice command "Open the door" to Siri through `XPorter`.

To activate the voice assistant through the introduced method and achieve a more generalized audio injection attack, we use a wire control board that contains a MOSFET transistor to manipulate the voltage received by the audio pin of a USB-C port. Specifically, the MOSFET transistor is used to control the current flow between the audio pin and the ground to simulate a fake button-pressing event that produces the same pattern of the input voltage for activating the voice assistants of the charging devices.

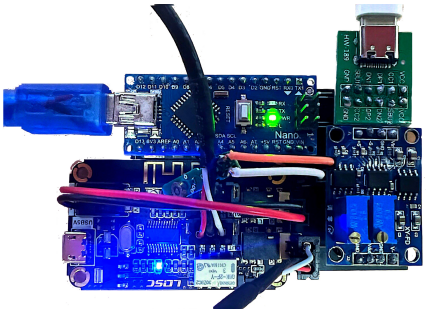
Audio Commands Injection. After obtaining the modulated audio signals and activating the voice assistant, the attacker can inject malicious audio commands through the compromised multi-port charger to acquire user privacy and perform further attacks. For instance, the attacker can send voice commands like "What's my name?" to obtain the victim's private information, make a ghost phone call by injecting "Call my wife", and hack the smart home equipped with a voice control system (*e.g.*, Apple HomeKit) by sending malicious voice commands like "Open the door". Fig. 4.6a and Fig. 4.6b individually present the spectrograms of the modulated audio clip and the injected signal received by the USB-C audio pin when sensing the voice command "Open the door" to Siri through `XPorter`. In particular, we find that despite the modulated audio being distorted in voice command injection, Siri can recognize the command and conduct corresponding responses because the patterns that contain the most important information are maintained as the two spectrograms present (yellow part).

4.4.4 Custom-built Attacking Device

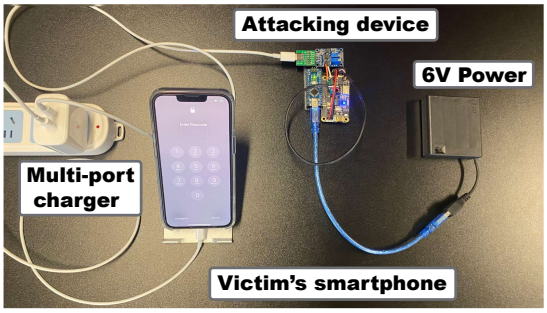
We design and implement a portable attacking device to achieve eavesdropping and inaudible audio injection attacks in `XPorter`, and Fig. 4.7a–Fig. 4.7c show the circuit



(a) Circuit design of the attacking device.



(b) Prototype.



(c) Attack scenarios.

Fig. 4.7 Attacking device in launching eavesdropping and audio injection attacks through a multi-port charger.

design, prototype outlook, and attack scenarios, respectively. First, the attacker can record the voltage leakages from the neighbor USB port to launch various eavesdropping attacks. Second, based on the assumption that the attacker can compromise the multi-port charger by parallel connecting the audio pins of the neighbor USB-C ports, an attacker can connect this attacking device to one of the USB-C ports and then remotely activate the voice assistant of the victim's mobile device and send audio clips that contain malicious voice commands to uncover sensitive information further.

In the prototype, we utilize an Arduino Nano microcontroller to record the voltage leakages and control the MOSFET transistor from a CX-729 wire control board [150] for voice assistant activation, a Bluetooth audio board [151] for receiving voice commands, an AD620 amplifier module [152] for adjusting the amplitude of the recorded voltage signals or modulated audio signals. As a proof-of-concept, we integrate these components in a customized extension PCB board powered by an external battery pack to eavesdrop on user activities as well as inaudibly inject malicious voice commands into the victim's smartphone through the USB-C interface. Note that it is possible to

draw power from the charger to support the attacking device by redesigning the prototype, which can also be implemented smaller and stuffed into the compromised charger to launch attacks directly.

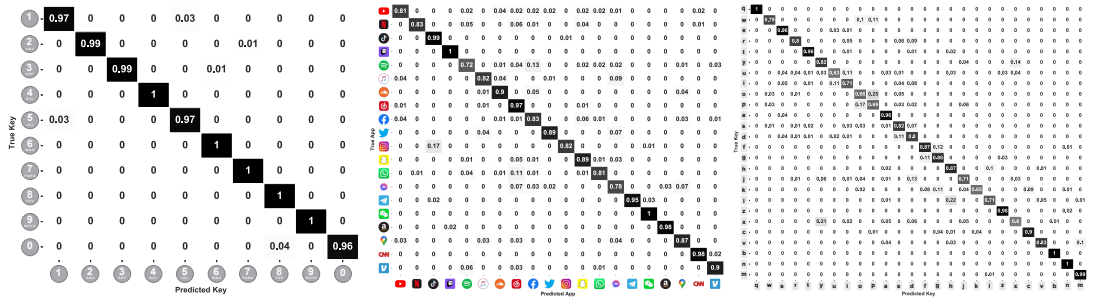
4.5 Evaluation

4.5.1 Effectiveness of Eavesdropping Attack

Experimental Setup. In the primary setting for evaluating the effectiveness of the eavesdropping attacks, we use the UGREEN 40W USB-C port charger[1], which has two USB-C ports for battery charging. Specifically, we first use one port to charge an iPhone 13 Pro as the victim's smartphone and then use the custom-built attacking device to record the voltages of another port when recruiting five participants (three males, two females) to collect data samples perform three common activities: *(i)* entering the password to unlock the smartphone, *(ii)* launching different mobile apps, and *(iii)* typing words in chat apps such as WhatsApp. We follow the same procedure and separately conduct experiments on four other commodity multi-port chargers from different vendors, four other mobile devices, and four other battery levels of the charging device. Moreover, all data processing and model training processes are conducted on a desktop with 32 GB memory and an Intel i7-9700K CPU, and an NVIDIA GeForce RTX 2080Ti GPU.

Effectiveness of Unlocking Password Inference. To evaluate the effectiveness of `XPorter` in inferring unlocking password, we collect voltage signals and obtain the processed data samples from the neighbor output USB-C port while pressing each button (*i.e.*, from 0 to 9) on the unlocking numeric keyboard for 100 times with a time interval of 0.5s. Then, we use 80% data samples to train the proposed CNN-LSTM classifier for determining each input key of the unlocking password and the remaining 20% data samples to evaluate the performance of the trained model with 10-fold cross-validation. Fig. 4.8a shows the confusion matrix of the evaluation results, where `XPorter` achieves 98.8% accuracy in recognizing the ten passcode pins (from 0 to 9) on the unlocking screen. As such, the attacker can precisely detect the victim's unlock-

[1]Note that dual-port chargers are also marketed as multi-port chargers. We adopt it to verify the feasibility of `XPorter`, and also show the potential of attacking multiple devices in §4.6.1. This work takes ethical considerations seriously and has been approved by the IRB (HUMAN-2023-0016-2).



(a) Passcode inference. (b) App fingerprinting. (c) Keystroke recovery.

Fig. 4.8 Effectiveness evaluation of eavesdropping attack. (a): Confusion matrix of recognizing 10 different passcode pins (from 0 to 9) on the unlocking screen. (b): Confusion matrix of fingerprinting 20 mobile apps. (c): Confusion matrix of uncovering 26 different keys on a full-size QWERTY keyboard (from "a" to "z"). Evaluated app list: –YouTube, –Netflix, –TikTok, –Twitch, –Spotify, –Apple Music, –SoundCloud, –Netease Cloud Music, –Facebook, –Twitter, –Instagram, –Snapchat, –WhatsApp, –Messenger, –Telegram, –WeChat, –Amazon, –Google Map, –CNN News, –Venmo.

ing password and then unlock the victim's smartphone to steal more user privacy when the victim's smartphone is left by charging.

Effectiveness of App Fingerprinting. To evaluate the effectiveness of `XPorter` in recognizing mobile app activities, we follow the same data collection procedure and record traces when the in-charging smartphone launches different mobile apps. Specifically, we select 20 most popular mobile apps and launch each of them for 50 times and obtain the first one-second voltages as the data samples for app fingerprinting. Similarly, we also utilize 80% data samples to train the classifier and the rest of 20% data for evaluating the model performance. Fig. 4.8b shows the confusion matrix of the evaluation results, where `XPorter` presents an overall accuracy of 88.7% in fingerprinting 20 popular mobile apps.

Moreover, we find `XPorter` performs the best in recognizing apps such as Twitch and WeChat that have distinguishable voltage patterns due to their customized launching animations that result in more energy consumption, which induces distinctive patterns of the voltage signals. On the contrary, `XPorter` performs the worst in recognizing apps like Spotify (72%) and Messenger (78%) because they adopt the default app launching setup (*i.e.*, white background with a static icon) and consume the lowest energy consumption as they have fewer network requirements and screen animations. Therefore, the changes in the voltage incurred by app launching are milder than other

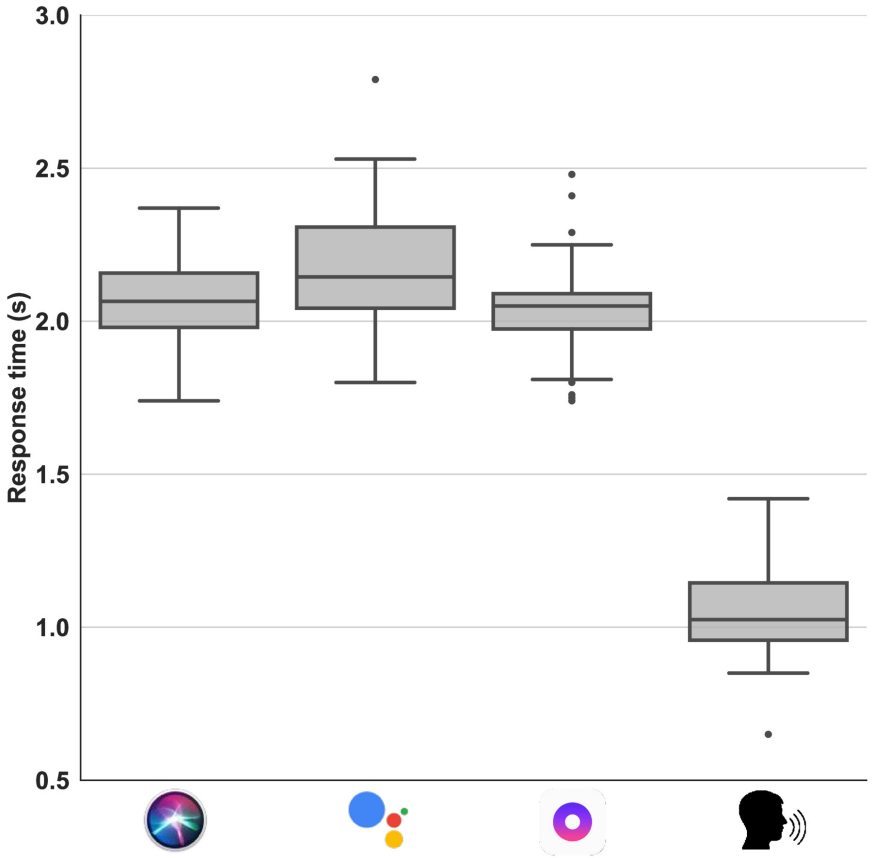


Fig. 4.9 Response time of assistants and human speaking.

apps, which further impacts the performance of **XPorter** in recognizing these apps. Nevertheless, **XPorter** still demonstrates high accuracy in detecting the app usage information of the victim during the charging process stealthily, especially apps containing sensitive information, e.g., Facebook and WhatsApp contain the contact and address information of the users.

Effectiveness of Keystroke Recovery. To achieve more fine-grained eavesdropping attacks, we also evaluate **XPorter** in recovering input keystrokes. Specifically, we collect data samples by typing the keys on the QWERTY full-size keyboard and repeating each key for 100 times, including 26 alphabetic keys from "a" to "z". Likewise, 80% of the collected data samples are used to build the CNN-LSTM classifier for recognizing keys, and 20% data samples are used to evaluate the model's effectiveness. Fig. 4.8c shows the confusion matrix of the evaluation results, where **XPorter** achieves overall 83.0% accuracy in recognizing 26 alphabetic keys (from "a" to "z") on a full-size QWERTY keyboard. In particular, we find most misclassification always happens in two

Table 1: Effectiveness of launching inaudible audio injection attacks via **XPorter**. We test voice commands with different SNR values and conduct 20 trials of end-to-end attacks, including the activation (Act.) and injection (Inj.). (✓/✗: success/ fail).

#	Voice Command	SNR (dB)	Act.	Inj.	Act.	Inj.	Act.	Inj.	#	Voice Command	SNR (dB)	Act.	Inj.	Act.	Inj.	Act.	Inj.
1	Call mom.	20.7	✓	✓	✓	✓	✓	✓	**11**	Where is my home?	19.0	✓	✓	✓	✓	✓	✓
2	Call my wife.	21.2	✓	✓	✓	✓	✓	✓	**12**	What's my ETA?	20.7	✓	✓	✓	✓	✓	✓
3	Call Bob.	20.3	✓	✓	✓	✓	✓	✓	**13**	Open the garage door.	21.5	✓	✓	✓	✓	✓	✓
4	Open Gmail.	19.8	✓	✓	✓	✓	✓	✓	**14**	Turn on the lights.	19.8	✓	✓	✓	✓	✓	✓
5	Open WhatsApp.	20.3	✓	✓	✓	✓	✓	✓	**15**	Turn off all alarms.	20.7	✓	✓	✓	✓	✓	✓
6	Open Paypal.	22.3	✓	✓	✓	✓	✓	✓	**16**	Send a message to...	19.2	✓	✓	✓	✓	✓	✓
7	Check my voicemail.	19.8	✓	✓	✓	✓	✓	✓	**17**	Send a reply email to...	18.8	✓	✓	✓	✓	✓	✓
8	Check my emails.	20.7	✓	✓	✓	✓	✓	✓	**18**	Tell Bob where I am.	20.3	✓	✓	✓	✓	✓	✓
9	Check my wallet.	18.5	✓	✓	✓	✓	✓	✓	**19**	Did I lock the front door?	21.3	✓	✓	✓	✓	✓	✓
10	What's my name?	21.2	✓	✓	✓	✓	✓	✓	**20**	What's my next schedule?	19.5	✓	✓	✓	✓	✓	✓

neighbor alphabetic keys, *e.g.*, nearly 11% testing samples are misclassified in recognizing keys "u" (63%) and "i" (71%) as the voltage patterns incurred by these key-pressing events are close. On the other hand, **XPorter** can detect keys on edge with high accuracy rates, such as "q" (100%), "a" (96%), and "z" (98%) that present distinctive patterns because they have fewer neighbor keys. In short, **XPorter** has demonstrated the ability to infer the victim's keystrokes through the voltage leakage in the charging process, which may contain fine-grained user privacy such as the conversation in chatting apps like WhatsApp, the password for payment in financial apps like PayPal.

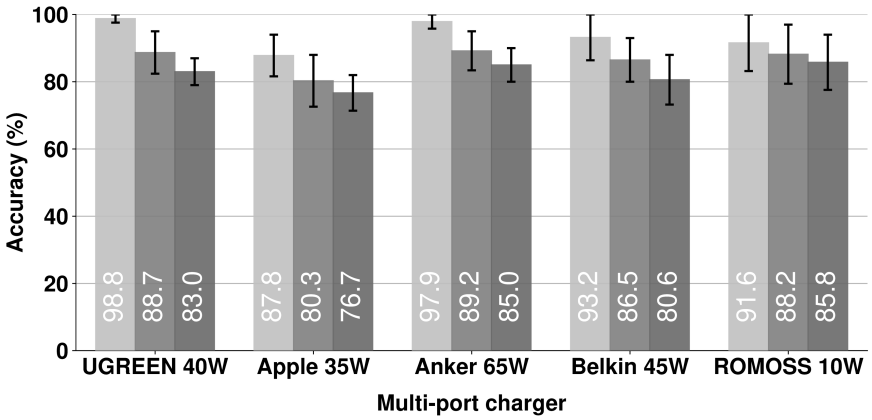
4.5.2 Effectiveness of Audio Injection Attack

Experiment Setup. To evaluate the effectiveness of the audio injection attack, we compromised the UGREEN 40W USB-C port charger by connecting the audio pins of its two USB-C ports together. We also use one port to charge an iPhone 13 Pro correspondingly as the victim's smartphone and then plug the attacking device into another port. Since the attacking device integrates a Bluetooth module for communication, we conduct the evaluation process by controlling the attacking device to activate the voice assistant and inject different modulated audio commands at a non-line-of-sight (NLoS) distance of 5 m.

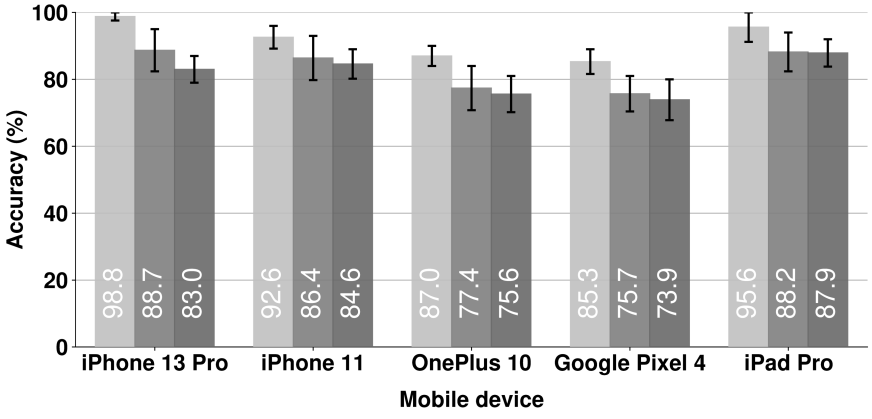
Effectiveness of Voice Assistant Activation. We conduct experiments on smartphones with different voice assistants to demonstrate the **XPorter**'s ability to activate the smartphone's voice assistant while bypassing the speech verification system.

Specifically, we utilize three smartphones (iPhone 13 Pro, Google Pixel 4, and OnePlus 10 Pro) with different commodity voice assistant systems (Siri, Google Assistant, and Breeno) by plugging them into the compromised charger and then activating each voice assistant for 50 times. Meanwhile, we record the response time of each trial as well as 50 trials of the response time of activating these voice assistants by speaking hotwords such as "Hey Siri", "Hello Google", and "Hey Breeno". Fig. 4.9 shows the box plot of the response time of the three voice assistants and the human speaking, and we know it takes an average of 2.07, 2.18, and 2.05 seconds to activate Siri, Google Assistant, and Breeno through `XPorter`, respectively. On the other hand, it only needs approximately 1.04 seconds to activate voice assistants by human speaking. Even though more time is required to activate the voice assistant, `XPorter` can bypass the speech verification mechanisms that have been widely deployed in commodity mobile devices, which makes `XPorter` more practical in a real-world scenario. Moreover, as the injected audio commands are voltage signals, `XPorter` cannot be detected and countered by existing defense approaches [149, 148, 41] that are proposed to defend against inaudible audio injections through acoustic signals.

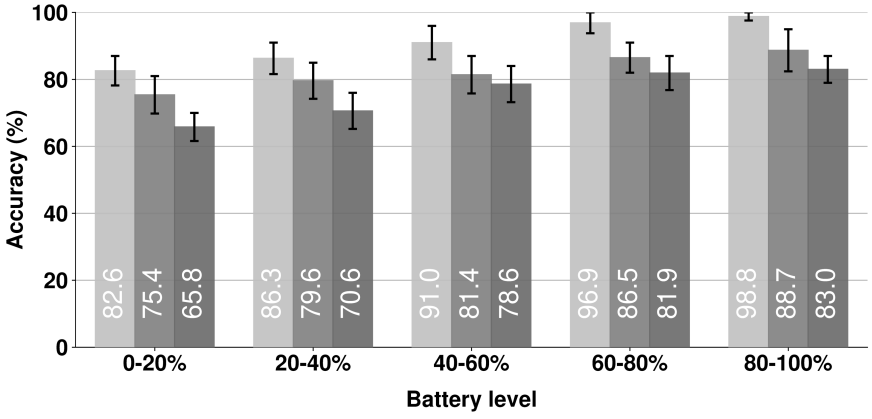
Effectiveness of Audio Commands Injection. To evaluate the effectiveness of inaudible voice commands injection attacks through `XPorter`, we exploit the Google WaveNet API [142] to generate 20 voice commands that have been widely used with high frequency in a quiet environment (SNR $\leq$25 dB), and each of those voice commands is a sentence that contains 2–10 words. Then, we activate the aforementioned three voice assistants (Siri, Google Assistant, and Breeno) using the proposed method and then inject each voice command into them. Once a voice assistant receives the voice commands and provides corresponding feedback, we consider it as one successful attack trial. Table 1 shows the detailed results of the 20 trials of end-to-end inaudible audio injection attacks on the three voice assistants. In all end-to-end attack trials, `XPorter` achieves 100% success rate in activating the three voice assistants, and 100% success rate in injecting different voice commands to compromise user privacy. Therefore, `XPorter` shows competitive performance compared to other state-of-the-art inaudible voice injection attacks [28, 12] and fills the gap between the eavesdropping and the injection attacks via a multi-port charger.



(a) Different chargers.



(b) Different mobile devices.



(c) Different battery levels.

Fig. 4.10 Evaluation results of three practical impact factors on the eavesdropping attacks: (a) Impact of different commodity multi-port chargers, (b) Impact of different mobile devices, (c) Impact of different battery levels of the in-charging device. – Unlocking passcode inference, – App recognition, – Keystroke recovery.

4.5.3 Impact of Practical Factors

Impact of different multi-port chargers. Due to the variety of different multi-port chargers' circuits, the induced voltage leakage presents different patterns. Thus, to evaluate whether `XPorter` can be launched to other multi-port chargers, we conduct further experiments by separately collecting data and training models from four other different commodity multi-port chargers: Apple 35W USB-C compact charger (A2579), Anker 65W smart charger (A2668), Belkin 65W USB-C charger (WCH013), and ROMOSS 2.1A USB-A charger. Fig. 4.10a shows the evaluation results of launching eavesdropping attacks on the five multi-port chargers, where we find `XPorter` achieves high eavesdropping accuracy across different multi-port chargers, *e.g.*, 93.9% in inferring unlocking passcode, 86.6% in recognizing app launching, and 82.2% uncovering keystrokes. In particular, the results show that `XPorter` shows a relatively lower eavesdropping accuracy when applying on the Apple 35W USB-C charger since it presents a relatively high voltage ripple [153] in the charging process so that the voltage changes induced by user activities are overwhelmed. However, the results demonstrate that the voltage leakage is a fundamental design flaw existing in different multi-port chargers, and `XPorter` presents a promising performance in inferring fine-grained user privacy across different commodity multi-port chargers.

Impact of different mobile devices. We use five commodity devices, including four smartphones (iPhone 13 Pro, iPhone 11, OnePlus 10 Pro, and Google Pixel 4) and one tablet (iPad Pro 2019), to evaluate the impact of different mobile devices. Fig. 4.10b shows the results of launching the eavesdropping attacks on different in-charging devices, where we find `XPorter` achieves the highest accuracy in inferring privacy from the iPhone 13 Pro and the iPad Pro but the lowest accuracy in smartphones like the OnePlus 10 Pro and the Google Pixel 4. Because user interactions (*e.g.*, launching apps or pressing keys) with an iPad Pro require more energy consumption due to the large touchscreen and UI components, which induces stronger voltage changes in the charger and voltage leakage. Nevertheless, `XPorter` can be scaled to different mobile devices with average accuracy rates of 91.9%, 83.3%, and 81.0% to recognize the unlocking passcode, the running app, and the keystrokes, respectively.

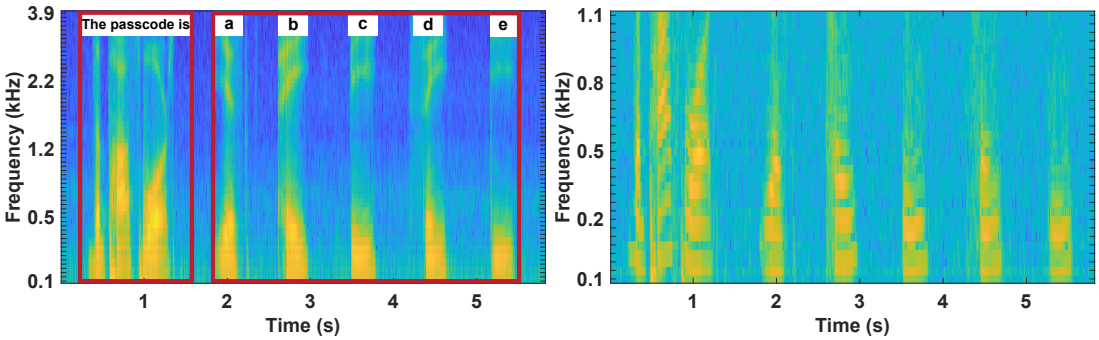
Impact of different battery levels. In practice, the mobile device may have different battery levels when being plugged into the port for charging the battery. To evaluate the impact of different battery levels on the performance of `XPorter`, we follow the same procedure and conduct experiments when the iPhone 13 Pro is at five different bat-

Table 2: Evaluation of inaudible audio injection attacks with different impact factors' combinations. Act. SR.: activation success rate. Inj. SR.: injection success rate.

Multi-port Charger	# of Ports	Type of Ports	Mobile Device	Voice Assistant	Battery Level	Act. SR.	Inj. SR.
UGREEN 40 W	2	2× USB-C	iPhone 13 Pro		80-100%	100%	100%
Anker 65W	3	1× USB-A 2× USB-C	iPhone 13 Pro		40-60%	100%	100%
Belkin 65W	2	2× USB-C	iPhone 13 Pro		60-80%	100%	100%
UGREEN 40 W	2	2× USB-C	Google Pixel 4		20-40%	100%	100%
Anker 65W	3	1× USB-A 2× USB-C	Google Pixel 4		60-80%	100%	100%
Belkin 65W	2	2× USB-C	Google Pixel 4		0-20%	100%	100%
UGREEN 40 W	2	2× USB-C	OnePlus 10 Pro		80-100%	100%	100%
Anker 65W	3	1× USB-A 2× USB-C	OnePlus 10 Pro		60-80%	100%	100%
Belkin 65W	2	2× USB-C	OnePlus 10 Pro		0-20%	100%	100%
UGREEN 40 W	2	2× USB-C	iPad Pro		60-80%	100%	100%
Anker 65W	3	1× USB-A 2× USB-C	iPad Pro		80-100%	100%	100%
Belkin 65W	2	2× USB-C	iPad Pro		20-40%	100%	100%

tery levels: 0–20%, 20–40%, 40–60%, 60–80%, and 80–100%, and Fig. 4.10c shows the experimental results of inferring the three user activities. Specifically, we know when the in-charging mobile device is at a high battery level (*e.g.*, $\geq 60\%$), **XPorter**'s performance of the eavesdropping attack is approximately 15% higher than the lower battery levels (*e.g.*, $\leq 40\%$). Because most of the output voltage of the plugged USB port is used for charging the battery when the device is at a low battery percentage. As such, when the battery is at a low level, the voltage changes induced by user activities could be overwhelmed by the intensive charging voltage. By contrast, when the battery reaches a high level, the charging process slows down, and the charging voltage is constant so that the voltage changes incurred by various user activities would present more distinctive patterns [10, 1, 3]. Despite the impact caused by different battery levels of the charging device, **XPorter** still achieves an overall accuracy of 91.1%, 82.3%, and 76.0% in inferring the unlocking passcode, the running app, and the keystrokes at the five battery levels.

Impact factors on audio injection attacks. Table 2 is the evaluation results of 12 end-to-end inaudible audio injection attacks with combinations of different impact factors. The results indicate that **XPorter** achieves 100% success rate in activating voice assistants and 100% success rate in injecting various voice commands across different multi-port chargers and mobile devices with different battery levels. Therefore,



(a) Original audio. (b) Demodulated voltage.

Fig. 4.11 Spectrograms of the original audio and the demodulated voltage signal of eavesdropping voice mail "The passcode is abcde" through the USB-C interface.

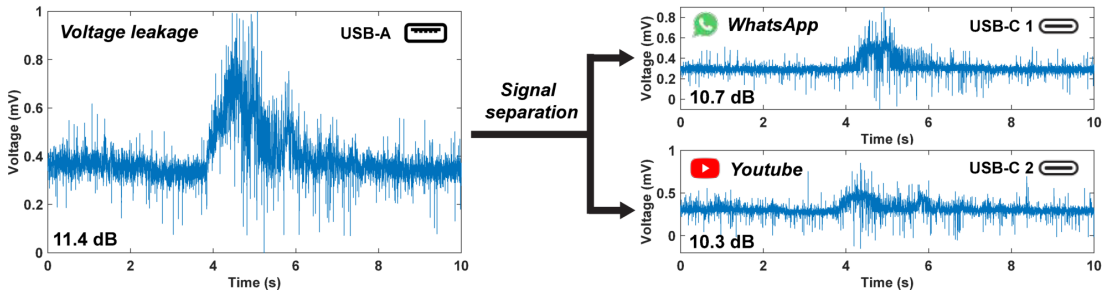
`XPorter` is resilient to the three practical factors in launching the injection attacks and realizes a high success rate.

4.6 Discussion

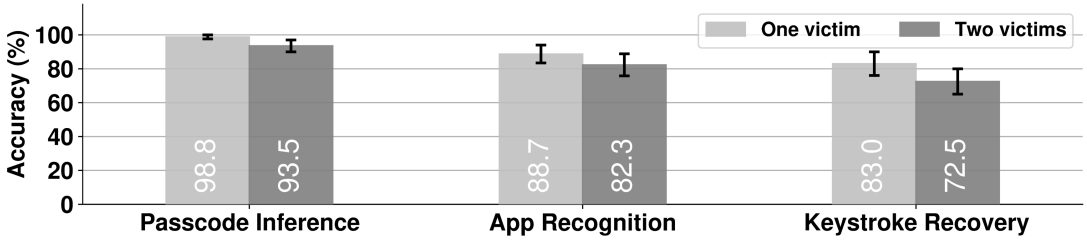
4.6.1 Extending Attacks

Eavesdropping audio through the voltage leakage. The audio pins of a USB-C port also support audio output, allowing for the acquisition of audio data through the analysis of charging power patterns [12]. Therefore, `XPorter` can be extended to obtain the voltage leakage from the audio output pins of the USB-C port so that the attacker can further spy on private information such as sensitive conversations in a phone call and secret messages in voice mails. Fig. 4.11a and Fig. 4.11b individually present the spectrograms of the original audio conversations and the obtained voltage output after applying the demodulation methods through `XPorter` of the voice mail "The passcode is abcde", where we can also find similar patterns that contain sensitive information are presented in the voltage signals. Hence, the attacker can also exploit the voltage leakage as shown in `XPorter` to uncover the secret audio conversations in voice mails and phone calls in a more stealthy way.

Attacks on multiple victims. To explore the feasibility of attacking multiple victims, we leverage the Anker 65W smart charger (2×USB-C, 1×USB-A) to charge two iPhone 13 Pro, and play the two charging smartphones simultaneously (*e.g.*, launching two different apps) while recording the voltage leakages from the neighbor USB-A



(a) Signal separation in the two-victim scenario.



(b) Effectiveness of eavesdropping two victims.

Fig. 4.12 Evaluation of attacking multiple victims.

port. Then, since the voltage leakage is a one-dimensional signal, we apply the blind source separation method (*e.g.*, FastICA [102]) to separate the mixed voltage signal into individual signals to determine the activities of each victim. Fig. 4.12a shows the process of separating the mixed voltage leakage (SNR=11.4 dB) to individual voltage signals when launching WhatsApp (SNR=10.7 dB) and YouTube (SNR=10.3 dB) on the two charging smartphones, respectively. We then conduct extensive experiments to evaluate the effectiveness of eavesdropping on two victims, and Fig. 4.12b shows the results. The accuracy decreases by approximately 5.3–10.5% due to the increase of noise in the individual signals after the source separation, but `XPorter` still achieves acceptable accuracy in uncovering different user activities. In addition, it is also feasible to launch multi-victim audio injection attacks by connecting all the audio pins of USB-C ports together in the compromised multi-port charger. In this case, the attacker can activate voice assistants and inject malicious voice commands into multiple charging devices simultaneously.

4.7 Defense Methods

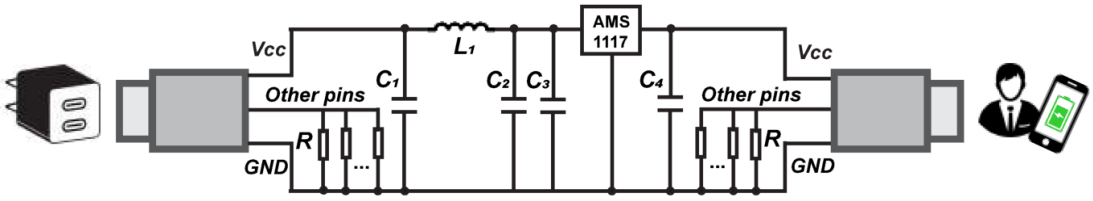
Software-based countermeasures. To prevent the inaudible audio injection attacks from `XPorter`, one software-based solution is to disable the audio transmission func-

tion through the system-level API [154] so that the voice control system cannot detect the voice commands. In addition, since the eavesdropping attacks depend on the captured voltage signals, we can add random noise (*e.g.*, dummy traffic packets [42, 102]) in the services to introduce extra power consumption to obfuscate the voltage traces without influencing the user experience. However, these methods inevitably bring extra energy consumption and may impact the charging efficiency. One effective countermeasure is to leverage the response time of voice assistant to detect potential inaudible audio injection attacks through the USB-C interface because Fig. 4.9 shows the response times of human speech and button-pressing event are different.

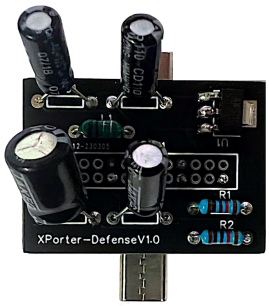
Hardware-based countermeasures. The straightforward way to mitigate the inferences and injections from `XPorter` is to eliminate the voltage leakages in multi-port chargers. Hence, we can connect a physical peripheral between the multi-port charger and the charging devices to smooth out the voltage leakages. For instance, we implement a simple circuit prototype as shown in Fig. 4.13a and Fig. 4.13b with resistors R =10 kΩ, capacitors C_1 = 10μF, C_2 = 1μF, C_3 = 100μF, C_4 = 22μF, and inductor L_1 =0.1 H, and an AMS1117 low-dropout regulator [155]. Fig. 4.13c shows it can smooth the voltage patterns induced by smartphone activities so that the attacker cannot exploit the voltage leakages to infer user privacy through `XPorter`. Another method is that the manufacturer could redesign the hardware by modifying the parallel connection mechanism so that the voltage change of one port cannot induce changes on other neighbor ports. Nevertheless, redesigning the hardware circuits can be a costly endeavor and is not feasible for sold multi-port chargers. Even if hardware modifications are made, it is still ambiguous how users would ascertain whether or not a multi-port charger could be trusted. Therefore, raising public awareness and educating users about the threat of untrusted multi-port chargers is a more effective and economical solution to prevent attacks.

4.8 Related Works

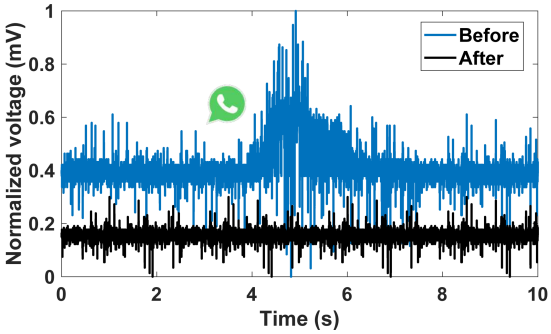
Attacks via charging devices. In Table 3, we summarize the quantified comparisons between `XPorter` and other state-of-the-art attacks via peripheral charging devices, *i.e.*, USB cables [10, 12], wireless chargers [2, 1]. In particular, `XPorter` can launch eavesdropping attacks without compromising devices, but it tampers chargers for audio injection. It is the first work to explore the essential design drawback of a popular charging interface, the multi-port chargers, to investigate their eavesdropping



(a) A simple circuit to smooth out voltage leakages.



(b) Prototype.



(c) Voltage leakages.

Fig. 4.13 Defend against `XPorter` via a simple circuit to smooth out voltage leakages.

Table 3: Quantified comparison with related works via charging devices. "●": yes, "○": no, Acc.: classification accuracy, SR.: injection success rate, UK: unlocking keyboard, FK: full-size QWERTY keyboard, NA: not available or evaluated.

Related works	Target device	No need to compromise		Eavesdropping attacks (Acc.)			Audio injection attacks (SR.)	Potential of attacking multiple victims
		Eavesdropping	Injection	App/Web	Keystroke (UK/FK)	Speech		
Charger-Surfing [10]	USB cable	○	○	○	● (98.7%/NA)	○	○	○
GhostTalk [12]	USB cable	○	○	○	○	● (93.3%)	● (100%)	○
EM-Surfing [2]	Power line of a wireless charger	○	○	● (95.0%)	● (98.3%/96.4%)	● (81.0%)	○	○
Cour *et al.* [1]	Power line of a wireless charger	○	○	● (91.5%)	○	○	○	○
`XPorter` (Our method)	Multi-port charger	●	○	● (88.7%)	● (98.8%/83.0%)	● (NA)	● (100%)	●

and voice injection vulnerabilities. In particular, **XPorter** outperforms these works in three-folds: *(i)* Unlike prior works that need to compromise USB cables [10, 12] or chargers [1, 2] to launch attacks, **XPorter** has no need to compromise victim devices to achieve fine-grained eavesdropping attacks that loosen the assumptions of attackers' ability in [10, 1, 2]. It also reduces the attack efforts to inject malicious voice commands than the prior work [12] because it needs no extra hardware component to be hidden in victims' devices or special USB cable as we have integrated all modules in the custom-built attacking device; *(ii)* **XPorter** is an orthogonal attack framework that can launch both eavesdropping attacks and inaudible voice injections through a single attack surface of the new charging platform; and *(iii)* **XPorter** presents the potential of attacking multiple charging devices simultaneously as we have demonstrated in §4.6.1.

Attacks via other power traces. The power consumption of a smartphone's battery can also be used to infer user privacy [156–158, 101, 4]. That is, an attacker can use pre-installed malware to obtain the battery profile of the victim's smartphone and further uncover user privacy. For instance, POWERFUL [16] exploits the smartphone's battery consumption data to recognize mobile app usage and activities. PowerSpy [17] uses two battery profiles in Android smartphones (*voltage_now* and *current_now*) to determine the motion of the smartphone for tracking the user's location. Furthermore, AppListener [102] leverages RF energy harvesting to capture the emitted RF energy of a Wi-Fi router to recognize fine-grained app activities of a connected smartphone.

4.9 Summary

In this paper, we present a new attack vector for eavesdropping on user privacy and inaudibly injecting voice commands through a commodity multi-port charger. To validate its feasibility and practicality, we design and implement **XPorter**, an attack framework that leverages the voltage leakage of the neighbor ports to infer sensitive information and exploits the USB-C interface to activate voice assistant and inject modulated voice commands into the victim's charging smartphone across the multi-port interface. Our extensive evaluation demonstrates that **XPorter** is effective in inferring fine-grained user privacy and also achieves 100% success rate in launching inaudible audio injection attacks across various impact factors such as different multi-port chargers and mobile devices. We hope our finds can raise public awareness of the vulnerability of multi-port chargers and spur research on detecting forthcoming attacks and new defense methods.

Chapter 5

Future Work

To expand our exploration of this research area in the future, we propose and identify three potential research directions as follows:

Developing secure mobile charging systems and protocols. We have revealed vulnerabilities of three common mobile charging systems (*e.g.*, wireless chargers, wireless charging power banks, and multi-port chargers) and explored the potential risks in this dissertation. To ensure a reliable charging process for numerous mobile devices, one future work of this thesis is to develop secure mobile charging systems and protocols. First, most wireless chargers adopt the Qi wireless charging protocol [8], which has been demonstrated insecure by our investigations [102, 4] and other related works [3, 1, 64]. Therefore, we have reported our discovered contactless side channels to the *Wireless Power Consortium* (WPC) and suggest WPC rethink the security of the current Qi protocol by revising the communications between coils and proposing a more secure protocol. Second, due to the physical characteristics of the inductive charging process, the emitted coil whine and magnetic field perturbations can be captured by our attack frameworks. Hence, we can mitigate the two physical phenomena in wireless charging by redesigning the hardware circuit to reduce electromagnetic (EM) emanations or generating extra noise to bring interference to the leaked physical signals. Combining these two aspects, we can adopt these defense approaches to other charging systems, *i.e.*, charging systems in electric vehicles [159] to prevent potential side-channel attacks [160]. In addition, to address the issue of privacy leakage through neighboring USB ports in multi-port chargers, we intend to share our findings with relevant manufacturers. We aim to assist them in incorporating additional modules to minimize cross-port voltage leakages and suggest smartphone manufacturers disable USB-C audio transmission during charging to enhance privacy protection.

Uncovering user privacy in 2D images and 3D scenes. Following a similar research line, the three works in this dissertation capture the 1D physical signals (*e.g.*, acoustic signals, magnetic field) and leverage deep neural network (DNN) models to infer user interactions (*e.g.*, unlocking the screen, launching apps) and corresponding user privacy on the charging smartphone, which falls in the category of classification tasks. With the development of generative models (*i.e.*, GAN and diffusion models), recent studies have demonstrated that 1D signals can reconstruct 2D images. For instance, `EM Eye` [123] utilizes the EM signals captured in the image-to-signal transmission to recover the 2D images taken by embedded cameras while using GAN to obtain high-quality images. Similarly, `FPLogger` [122] proposes the first side-channel attack against commercial in-display fingerprint sensors through the captured EM emanations and also leverages denoising diffusion models to enhance the patterns of recovered fingerprints, and successfully unlock COTS smartphones with 3D fingerprint pieces. These two works have unveiled the potential of leveraging generative models in our future research to investigate contactless side channels in cyber-physical systems and push the boundaries of inferred user privacy with increased granularity.

Enhancing charging efficiency and moving towards battery-free. Charging efficiency is always a bottleneck in mobile charging systems. One direction of our future work is to investigate how to enhance the charging efficiency in all charging accessories of mobile devices, *i.e.*, USB cables, wireless chargers, power banks, and vehicle charging stations. Therefore, it is necessary to delve into the research of designing an adjustment mechanism in the power consumption of mobile devices to balance the CPU/GPU workloads and increase the battery life, such as dynamic voltage and frequency scaling (DVFS) [161]. Moreover, as many newly-released smartphones adopt fast charging protocols (e.g., AirVOOC [162]), the efficiency and potential adverse effects on battery performance have yet to be investigated. Therefore, we include this as another focus area for future research. In addition, recent studies have focused on techniques that can convert other energy to power mobile devices, *a.k.a.*, energy harvesting. For instance, we can combine solar panels with wireless charging coils to design and implement a wireless charger that only uses sunlight to charge smartphones. Another method is to scavenge radio-frequency (RF) signals from ambient environments and convert them to DC voltage to charge the devices [102]. We also plan to work on increasing the distance of wireless charging (*i.e.*, currently less than 4 cm in Qi protocol) to move towards a charging model with real "wireless" manners at a long-range, and this new concept has been proposed by industries [163] recently.

Chapter 6

Conclusion

This dissertation explores *contactless side channels* in mobile charging systems, which aims to reveal potential user privacy leakage when being charged by wireless chargers or wireless charging power banks and validate the design flaw in multi-port chargers that may lead to inaudible audio injection attacks. In particular, we focus on investigating two physical phenomena emitted from the wireless charging process, the coil whine and the magnetic field perturbations, and leveraging the captured signals to uncover user interactions on the charging devices, *i.e.*, unlocking passcode, app usage, and sensitive user keystrokes. Moreover, we also discovered the design flaw of paralleled USB ports on multi-port chargers that may leak user privacy across neighbor charging ports while the audio transmission pins on the USB-C port can be exploited for injecting malicious voice commands. We design and implement three attack frameworks to demonstrate the feasibility of these contactless side-channel attacks in mobile charging systems, and the results show that our newly discovered side channels could successfully attack these mobile charging platforms.

We believe exploring the *contactless side channels* opens up the security of cyber-physical systems (CPS), such as mobile charging systems, mainly for two reasons. First, the revealed side channels in the aforementioned mobile charging systems could raise public awareness of the potential threats when people choose to charge their devices through untrusted charging platforms. Second, our proposed countermeasures could help manufacturers develop effective defense mechanisms in their products, *i.e.*, redesigning hardware circuits or implementing software-defined defenses to obfuscate attackers with random-induced noise. Therefore, merging these insights paves the way for discovering new vulnerabilities in various other IoT devices, and fostering the development of secure and reliable mobile devices in the IoT era.

References

[1] Alexander S La Cour, Khurram K Afridi, and G Edward Suh. Wireless charging power side-channel attacks. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 651–665, 2021.

[2] Jianwei Liu, Xiang Zou, Leqi Zhao, Yusheng Tao, Sideng Hu, Jinsong Han, and Kui Ren. Privacy leakage in wireless charging. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[3] Yi Wu, Zhuohang Li, Nicholas Van Nostrand, and Jian 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)*, pages 916–929, 2021.

[4] Tao Ni, Xiaokuan Zhang, Chaoshun Zuo, Jianfeng Li, Zhenyu Yan, Wubing Wang, Weitao Xu, Xiapu Luo, and Qingchuan Zhao. Uncovering user interactions on smartphones via contactless wireless charging side channels. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 3399–3415. IEEE, 2023.

[5] Tao Ni, Jianfeng Li, Xiaokuan Zhang, Chaoshun Zuo, Wubing Wang, Weitao Xu, Xiapu Luo, and Qingchuan Zhao. Exploiting contactless side channels in wireless charging power banks for user privacy inference via few-shot learning. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2023.

[6] Tao Ni, Yongliang Chen, Weitao Xu, Lei Xue, and Qingchuan Zhao. Xporter: A study of the multi-port charger security on privacy leakage and voice injection. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2023.

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

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

[9] IMARC Group. Power bank market: Global industry trends, share, size, growth, opportunity and forecast 2022-2027, 2021. https://www.researchandmarkets.com/reports/5562500/power-bank-market-global-industry-trends-share.

[10] Patrick Cronin, Xing Gao, Chengmo Yang, and Haining Wang. Charger-surfing: Exploiting a power line side-channel for smartphone information leakage. In *Proceedings of the 30th USENIX Security Symposium*, 2021.

[11] Yang Su, Daniel Genkin, Damith Ranasinghe, and Yuval Yarom. Usb snooping made easy: crosstalk leakage attacks on usb hubs. In *Proceedings of the 26th USENIX Security Symposium*, pages 1145–1161, 2017.

[12] Yuanda Wang, Hanqing Guo, and Qiben Yan. Ghosttalk: Interactive attack on smartphone voice system through power line. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2022.

[13] Qing Yang, Paolo Gasti, Gang Zhou, Aydin Farajidavar, and Kiran S Balagani. On inferring browsing activity on smartphones via USB power analysis side-channel. *IEEE Transactions on Information Forensics and Security*, 12(5):1056–1066, 2016.

[14] Riccardo Spolaor, Laila Abudahi, Veelasha Moonsamy, Mauro Conti, and Radha Poovendran. No free charge theorem: A covert channel via USB charging cable on mobile devices. In *International Conference on Applied Cryptography and Network Security*, pages 83–102. Springer, 2017.

[15] Patrick Cronin, Xing Gao, Haining Wang, and Chase Cotton. An exploration of ARM system-level cache and GPU side channels. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2021.

[16] Yimin Chen, Xiaocong Jin, Jingchao Sun, Rui Zhang, and Yanchao Zhang. Powerful: Mobile app fingerprinting via power analysis. In *Proceedings of the International Conference on Computer Communications (INFOCOM)*, 2017.

[17] Yan Michalevsky, Aaron Schulman, Gunaa Arumugam Veerapandian, Dan Boneh, and Gabi Nakibly. PowerSpy: Location tracking using mobile device power analysis. In *Proceedings of the USENIX Security Symposium*, 2015.

[18] Nikolay Matyunin, Yujue Wang, Tolga Arul, Kristian Kullmann, Jakub Szefer, and Stefan 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*, pages 135–149, 2019.

[19] Xiaoyong Zhou, Soteris Demetriou, Dongjing He, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, Carl A Gunter, and Klara 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)*, pages 1017–1028, 2013.

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

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

[22] Wikipedia. Inductive charging. https://en.wikipedia.org/wiki/Inductive_charging, 2022.

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

[24] Wikipedia. Electromagnetically induced acoustic noise, 2022. https://en.wikipedia.org/wiki/Electromagnetically_induced_acoustic_noise.

[25] 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.

[26] Wenqiang Jin, Srinivasan Murali, Huadi Zhu, and Ming Li. Periscope: A keystroke inference attack using human coupled electromagnetic emanations. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 700–714, 2021.

[27] Yongyao Cai, Yang Zhao, Xianfeng Ding, and James Fennelly. Magnetometer basics for mobile phone applications. *Electron. Prod. (Garden City, New York)*, 54(2), 2012.

[28] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017.

[29] Seita Maruyama, Satohiro Wakabayashi, and Tatsuya 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)*, pages 620–637. IEEE, 2019.

[30] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na 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.

[31] Xunnian Yang. Efficient circular arc interpolation based on active tolerance control. *Computer-Aided Design*, 34(13):1037–1046, 2002.

[32] Mingtian Tan, Junpeng Wan, Zhe Zhou, and Zhou Li. Invisible probe: Timing attacks with PCIe congestion side-channel. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 322–338. IEEE, 2021.

[33] Jin Chen, Per Jönsson, Masayuki Tamura, Zhihui Gu, Bunkei Matsushita, and Lars Eklundh. A simple method for reconstructing a high-quality NDVI time-series data set based on the Savitzky–Golay filter. *Remote sensing of Environment*, 91(3-4):332–344, 2004.

[34] Boyuan Yang, Ruirong Chen, Kai Huang, Jun Yang, and Wei 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.

[35] Olivier Rukundo and Hanqiang Cao. Nearest neighbor value interpolation. *arXiv preprint arXiv:1211.1768*, 2012.

[36] David M Kreindler and Charles 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.

[37] Rui Ning, Cong Wang, ChunSheng Xin, Jiang Li, and Hongyi Wu. Deepmag+: Sniffing mobile apps in magnetic field through deep learning. *Pervasive and Mobile Computing*, 61:101106, 2020.

[38] Md Sahidullah and Goutam Saha. Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. *Speech communication*, 54(4):543–565, 2012.

[39] Aniruddha Adiga, Mathew Magimai, and Chandra Sekhar Seelamantula. Gammatone wavelet cepstral coefficients for robust speech recognition. In *Proceedings of the IEEE TENCON*, 2013.

[40] Stephanie McCandless. An algorithm for automatic formant extraction using linear prediction spectra. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 22(2):135–141, 1974.

[41] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. Void: A fast and light voice liveness detection system. In *Proceedings of the 29th USENIX Security Symposium*, pages 2685–2702, 2020.

[42] Jianfeng Li, Hao Zhou, Shuohan Wu, Xiapu Luo, Ting Wang, Xian Zhan, and Xiaobo Ma. FOAP: Fine-grained open-world Android app fingerprinting. In *Proceedings of the 31st USENIX Security Symposium*, 2022.

[43] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew 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)*, pages 1131–1148, 2019.

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

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

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

[47] Similarweb. Top Apps Ranking. https://www.similarweb.com/apps/top/apple/store-rank/us/all/top-free/iphone/, 2022.

[48] ChargingLab. https://www.chargerlab.com/, 2022.

[49] Sriram Sami, Sean Rui Xiang Tan, Bangjie Sun, and Jun 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)*, pages 288–301, 2021.

[50] Rahul Anand Sharma, Elahe Soltanaghaei, Anthony Rowe, and Vyas Sekar. Lumos: Identifying and localizing diverse hidden IoT devices in an unfamiliar environment. In *Proceedings of the 31st USENIX Security Symposium*, 2022.

[51] App Store. Hidden Camera Detector. https://apps.apple.com/us/app/hidden-camera-detector/id532882360.

[52] Google Play. Glint Finder - Camera Detector. https://play.google.com/store/apps/details?id=com.workshop512.glintfinder, 2022.

[53] Matteo Cardaioli, Stefano Cecconello, Mauro Conti, Simone Milani, Stjepan Picek, and Eugen 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)*, pages 1687–1704, 2022.

[54] 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, 2022.

[55] Haoqi Shan, Boyi Zhang, Zihao Zhan, Dean Sullivan, Shuo Wang, and Yier Jin. Invisible finger: Practical electromagnetic interference attack on touchscreen-based electronic devices. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 1548–1548, 2022.

[56] Zhuoran Liu, Niels Samwel, Léo Weissbart, Zhengyu Zhao, Dirk Lauret, Lejla Batina, and Martha 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.

[57] Hao Pan, Lanqing Yang, Honglu Li, Chuang-Wen You, Xiaoyu Ji, Yi-Chao Chen, Zhenxian Hu, and Guangtao 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.

[58] Mingke Wang, Qing Luo, Yasha Iravantchi, Xiaomeng Chen, Alanson Sample, Kang G Shin, Xiaohua Tian, Xinbing Wang, and Dongyao Chen. Automatic calibration of magnetic tracking. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking (MobiCom)*, pages 391–404, 2022.

[59] Hua Huang, Hongkai Chen, and Shan Lin. Magtrack: Enabling safe driving monitoring with wearable magnetics. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services*, 2019.

[60] Tomer Gluck, Rami Puzis, Yossi Oren, and Asaf 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.

[61] Shengqi Yang, Wayne Wolf, Narayanan Vijaykrishnan, Dimitrios N Serpanos, and Yuan Xie. Power attack resistant cryptosystem design: A dynamic voltage and frequency switching approach. In *Design, Automation and Test in Europe*, 2005.

[62] Kai Wang, Richard Mitev, Chen Yan, Xiaoyu Ji, Ahmad-Reza Sadeghi, and Wenyuan Xu. Ghosttouch: Targeted attacks on touchscreens without physical touch. In *31st USENIX Security Symposium (USENIX Security 22). USENIX Association, Boston, MA. https://www. usenix. org/conference/usenixsecurity22/presentation/wang-kai*, 2022.

[63] Qinhong Jiang, Xiaoyu Ji, Chen Yan, Zhixin Xie, Haina Lou, and Wenyuan Xu. GlitchHiker: Uncovering vulnerabilities of image signal transmission with iemi. In *Proceedings of the 32nd USENIX Security Symposium (USENIX Security 23)*, pages 7249–7266, 2023.

[64] Donghui Dai, Zhenlin An, and Lei Yang. Inducing wireless chargers to voice out for inaudible command attacks. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1789–1806. IEEE, 2023.

[65] Zihao Zhan, Yirui Yang, Haoqi Shan, Hanqiu Wang, Yier Jin, and Shuo Wang. Voltschemer: Use voltage noise to manipulate your wireless charger. In *Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24)*, 2024.

[66] Lejla Batina, Shivam Bhasin, Dirmanto Jap, and Stjepan Picek. CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel. In *Proceedings of the 28th USENIX Security Symposium*, 2019.

[67] Alireza Nazari, Nader Sehatbakhsh, Monjur Alam, Alenka Zajic, and Milos Prvulovic. Eddie: EM-based detection of deviations in program execution. In *Proceedings of the Annual International Symposium on Computer Architecture (ISCA)*, pages 333–346, 2017.

[68] Yi Han, Sriharsha Etigowni, Hua Liu, Saman Zonouz, and Athina 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)*, pages 1095–1108, 2017.

[69] Daniel Genkin, Lev Pachmanov, Itamar Pipman, Eran Tromer, and Yuval 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)*, pages 1626–1638, 2016.

[70] Monjur Alam, Baki Berkay Yilmaz, Frank Werner, Niels Samwel, Alenka G Zajic, Daniel Genkin, Yuval Yarom, and Milos 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)*, pages 507–522, 2021.

[71] Pierre Belgarric, Pierre-Alain Fouque, Gilles Macario-Rat, and Mehdi Tibouchi. Side-channel analysis of Weierstrass and Koblitz curve ECDSA on Android smartphones. In *Cryptographers' Track at the RSA Conference*, pages 236–252. Springer, 2016.

[72] Monjur Alam, Haider Adnan Khan, Moumita Dey, Nishith Sinha, Robert Callan, Alenka Zajic, and Milos Prvulovic. One&Done: A single-decryption EM-based attack on OpenSSL's constant-time blinded RSA. In *Proceedings of the USENIX Security Symposium*, pages 585–602, 2018.

[73] Yushi Cheng, Xiaoyu Ji, Wenyuan Xu, Hao Pan, Zhuangdi Zhu, Chuang-Wen You, Yi-Chao Chen, and Lili Qiu. Magattack: Guessing application launching and operation via smartphone. In *Proceedings of the ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, pages 283–294, 2019.

[74] Xiaokuan Zhang, Yuan Xiao, and Yinqian 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)*, pages 858–870, 2016.

[75] Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice, and Stefan Mangard. Armageddon: Cache attacks on mobile devices. In *Proceedings of the 25th USENIX Security Symposium*, pages 549–564, 2016.

[76] Gregor Haas, Seetal Potluri, and Aydin Aysu. iTimed: Cache attacks on the Apple A10 fusion SoC. *Cryptology ePrint Archive*, 2021.

[77] Chao Shen, Shichao Pei, Tianwen Yu, and Xiaohong 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.

[78] Yihao Liu, Kai Huang, Xingzhe Song, Boyuan Yang, and Wei 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.

[79] Kehuan Zhang and XiaoFeng Wang. Peeping tom in the neighborhood: Keystroke eavesdropping on multi-user systems. In *Proceedings of the USENIX Security Symposium*, volume 20, page 23, 2009.

[80] Zhiyun Qian, Z Morley Mao, and Yinglian 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)*, pages 593–604, 2012.

[81] Chia-Chi Lin, Hongyang Li, Xiao-yong Zhou, and XiaoFeng Wang. Screenmilker: How to milk your Android screen for secrets. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2014.

[82] Suman Jana and Vitaly Shmatikov. Memento: Learning secrets from process footprints. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 143–157. IEEE, 2012.

[83] Qi Alfred Chen, Zhiyun Qian, and Z Morley 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.

[84] Wenrui Diao, Xiangyu Liu, Zhou Li, and Kehuan 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)*, pages 414–432. IEEE, 2016.

[85] Adam J Aviv, Benjamin Sapp, Matt Blaze, and Jonathan M Smith. Practicality of accelerometer side channels on smartphones. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2012.

[86] Xiangyu Liu, Zhe Zhou, Wenrui Diao, Zhou Li, and Kehuan Zhang. When good becomes evil: Keystroke inference with smartwatch. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1273–1285, 2015.

[87] Emmanuel Owusu, Jun Han, Sauvik Das, Adrian Perrig, and Joy Zhang. Accessory: Password inference using accelerometers on smartphones. In *Proceedings of the Workshop on Mobile Computing Systems and Applications*, 2012.

[88] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. Gyrophone: Recognizing speech from gyroscope signals. In *Proceedings of the 23rd USENIX Security Symposium*, pages 1053–1067, 2014.

[89] He Wang, Ted Tsung-Te Lai, and Romit Roy Choudhury. Mole: Motion leaks through smartwatch sensors. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 155–166, 2015.

[90] Li Lu, Jiadi Yu, Yingying Chen, Yanmin Zhu, Xiangyu Xu, Guangtao Xue, and Minglu 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.

[91] Huijie Chen, Fan Li, Wan Du, Song Yang, Matthew Conn, and Yu 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)*, 4(3):1–23, 2020.

[92] Spherical Insights. Global power bank rental services market size, share and trends, analysis and forecast 2021 – 2030, 2022. https://www.sphericalinsights.com/reports/power-bank-rental-services-market.

[93] Wikipedia. Inductive charging, 2022. https://en.wikipedia.org/wiki/Inductive_charging.

[94] Rui Ning, Cong Wang, ChunSheng Xin, Jiang Li, and Hongyi Wu. Deepmag: Sniffing mobile apps in magnetic field through deep convolutional neural networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10. IEEE, 2018.

[95] J Ross Quinlan. Learning decision tree classifiers. *ACM Computing Surveys (CSUR)*, 28(1):71–72, 1996.

[96] CrioSoft LLC. Amperes - battery charge info, 2022. https://apps.apple.com/us/app/amperes-battery-charge-info/id1245475416.

[97] EGO INNOVATION LTD. Ego magpower gen.2 6000mah 15w magsafe powerbank, 2021. https://www.egoshop.co/en/products/ego-magpower-15w-magsafe-6000mah-powerbank-1.

[98] ANKER. Anker MagGo, 2020. https://us.anker.com/pages/maggo.

[99] Apple. Magsafe battery pack, 2022. https://support.apple.com/kb/SP846?viewlocale=en_US&locale=en_US.

[100] Belkin. Magnetic wireless power bank 2.5k, 2022. https://www.belkin.com/us/chargers/wireless/boost-charge-magnetic-wireless-power-bank-2-5k/p/p-bpd002/.

[101] Tao Ni, Yongliang Chen, Keqi Song, and Weitao Xu. A simple and fast human activity recognition system using radio frequency energy harvesting. In *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, pages 666–671, 2021.

[102] Tao Ni, Guohao Lan, Jia Wang, Qingchuan Zhao, and Weitao Xu. Eavesdropping mobile app activity via radio-frequency energy harvesting. In *Proceedings of the 32nd USENIX Security Symposium*, 2023.

[103] Rukundo Olivier and Cao Hanqiang. Nearest neighbor value interpolation. *International Journal of Advanced Computer Science and Applications*, 3(4), 2012.

[104] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[105] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1126–1135, 2017.

[106] Arduino Nano. Arduino Nano document, 2022. https://docs.arduino.cc/hardware/nano.

[107] Adafruit. Electret microphone amplifier - MAX9814 with auto gain control, 2021. https://www.adafruit.com/product/1713.

[108] ElectronicWings. HMC5883L magnetometer module, 2022. https://www.electronicwings.com/sensors-modules/hmc5883l-magnetometer-module.

[109] appfigures. Top ranked iOS app store apps, 2021. https://appfigures.com/top-apps/ios-app-store/united-states/iphone/top-overall.

[110] Seyed Ali Rokni, Marjan Nourollahi, and Hassan Ghasemzadeh. Personalized human activity recognition using convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[111] Francisco Javier Ordóñez Morales and Daniel Roggen. Deep convolutional feature transfer across mobile activity recognition domains, sensor modalities and locations. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, pages 92–99, 2016.

[112] Jindong Wang, Vincent W Zheng, Yiqiang Chen, and Meiyu Huang. Deep transfer learning for cross-domain activity recognition. In *Proceedings of the 3rd International Conference on Crowd Science and Engineering*, pages 1–8, 2018.

[113] Mian Dong and Lin Zhong. Chameleon: A color-adaptive web browser for mobile OLED displays. In *Proceedings of the 9th International Conference on Mobile systems, Applications, and Services*, pages 85–98, 2011.

[114] Xiang Chen, Yiran Chen, Zhan Ma, and Felix CA Fernandes. How is energy consumed in smartphone display applications? In *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications*, pages 1–6, 2013.

[115] Danyue Ma, Jixi Lu, Xiujie Fang, Ke Yang, Kun Wang, Ning Zhang, Bangcheng Han, and Ming Ding. Parameter modeling analysis of a cylindrical ferrite magnetic shield to reduce magnetic noise. *IEEE Transactions on Industrial Electronics*, 69(1):991–998, 2021.

[116] US Energy Products. Us energy products (ad3) reflective foam insulation shield, 2023. https://www.amazon.com/US-Energy-Products-Reflective-Insulation/dp/B07R1S669V.

[117] RIGOL. RIGOL DS1052E, 2022. https://www.batronix.com/shop/oscilloscopes/Rigol-DS1052E.html.

[118] Qianru Liao, Yongzhi Huang, Yandao Huang, Yuheng Zhong, Huitong Jin, and Kaishun Wu. MagEar: Eavesdropping via audio recovery using magnetic side channel. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, pages 371–383, 2022.

[119] Myeongwon Choi, Sangeun Oh, Insu Kim, and Hyosu Kim. Magsnoop: listening to sounds induced by magnetic field fluctuations to infer mobile payment tokens. In *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, pages 409–421, 2022.

[120] Henrique Teles Maia, Chang Xiao, Dingzeyu Li, Eitan Grinspun, and Changxi Zheng. Can one hear the shape of a neural network?: Snooping the gpu via magnetic side channel. 2021.

[121] Martin Vuagnoux and Sylvain Pasini. Compromising electromagnetic emanations of wired and wireless keyboards. In *Proceedings of the USENIX Security Symposium*, volume 8, pages 1–16, 2009.

[122] Tao Ni, Xiaokuan Zhang, and Qingchuan Zhao. Recovering fingerprints from in-display fingerprint sensors via electromagnetic side channel. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 253–267, 2023.

[123] Yan Long, Qinhong Jiang, Chen Yan, Tobias Alam, Xiaoyu Ji, Wenyuan Xu, and Kevin Fu. EM Eye: Characterizing electromagnetic side-channel eavesdropping on embedded cameras. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2024.

[124] Mantun Chen, Yongjun Wang, Hongzuo Xu, and Xiatian Zhu. Few-shot website fingerprinting attack. *Computer Networks*, 198:108298, 2021.

[125] Chenggang Wang, Jimmy Dani, Xiang Li, Xiaodong Jia, and Boyang Wang. Adaptive fingerprinting: website fingerprinting over few encrypted traffic. In *Proceedings of the 11th ACM Conference on Data and Application Security and Privacy*, pages 149–160, 2021.

[126] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *Proceedings of the ICML Deep Learning Workshop*, 2015.

[127] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.

[128] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. Metasense: Few-shot adaptation to untrained conditions in deep mobile sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 110–123, 2019.

[129] Guohao Lan, Bailey Heit, Tim Scargill, and Maria Gorlatova. Gazegraph: Graph-based few-shot cognitive context sensing from human visual behavior. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 422–435, 2020.

[130] Shuya Ding, Zhe Chen, Tianyue Zheng, and Jun Luo. RF-net: A unified meta-learning framework for RF-enabled one-shot human activity recognition. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 517–530, 2020.

[131] Pengli Hu, Chengpei Tang, Kang Yin, and Xie Zhang. WiGR: A practical Wi-Fi-based gesture recognition system with a lightweight few-shot network. *Applied Sciences*, 11(8):3329, 2021.

[132] Rui Xiao, Jianwei Liu, Jinsong Han, and Kui Ren. OneFi: One-shot recognition for unseen gesture via COTS WiFi. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 206–219, 2021.

[133] Mingda Han, Huanqi Yang, Tao Ni, Di Duan, Mengzhe Ruan, Yongliang Chen, Jia Zhang, and Weitao Xu. mmsign: mmwave-based few-shot online handwritten signature verification. *ACM Transactions on Sensor Networks*, 2023.

[134] FactMR. USB wall charger market, 2022. https://www.factmr.com/report/2471/usb-wall-charger-market.

[135] Qing Yang, Paolo Gasti, Kiran Balagani, Yantao Li, and Gang Zhou. USB side-channel attack on Tor. *Computer Networks*, 141:57–66, 2018.

[136] Jing Tian, Nolen Scaife, Deepak Kumar, Michael Bailey, Adam Bates, and Kevin Butler. Sok:" plug & pray" today–understanding USB insecurity in versions 1 through C. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 1032–1047. IEEE, 2018.

[137] Federico Griscioli, Maurizio Pizzonia, and Marco Sacchetti. USBCheckIn: Preventing BadUSB attacks by forcing human-device interaction. In *Proceedings of the 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 493–496. IEEE, 2016.

[138] BOONE ASHWORTH. A new EU law would force iphones to adopt USB-C charging, 2022. https://www.wired.com/story/eu-law-usb-c-iphones-lightning/.

[139] Apple. Homekit accessories, 2022. https://support.apple.com/en-us/HT208939.

[140] Yichuang Sun and JK Fidler. Design method for impedance matching networks. *IEE Proceedings-Circuits, Devices and Systems*, 143(4):186–194, 1996.

[141] TensorSpeech. Real-time state-of-the-art speech synthesis for tensorflow 2, 2021. https://github.com/TensorSpeech/TensorflowTTS.

[142] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[143] David M Kreindler and Charles J Lumsden. The effects of the irregular sample and missing data in time series analysis. *Nonlinear dynamics, psychology, and life sciences*, 2006.

[144] Pavel Senin. Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, 855(1-23):40, 2008.

[145] Christopher Tralie and Elizabeth Dempsey. Exact, parallelizable dynamic time warping alignment with linear memory. *arXiv preprint arXiv:2008.02734*, 2020.

[146] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[147] Tiantian Liu, Feng Lin, Zhangsen Wang, Chao Wang, Zhongjie Ba, Li Lu, Wenyao Xu, and Kui Ren. MagBackdoor: Beware of your loudspeaker as a backdoor for magnetic injection attacks. In *Proceedings of the 44th IEEE Symposium on Security and Privacy (SP)*, pages 3416–3431. IEEE Computer Society, 2023.

[148] Zhuohang Li, Cong Shi, Tianfang Zhang, Yi Xie, Jian Liu, Bo Yuan, and Yingying Chen. Robust detection of machine-induced audio attacks in intelligent audio systems with microphone array. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1884–1899, 2021.

[149] Guoming Zhang, Xiaoyu Ji, Xinfeng Li, Gang Qu, and Wenyuan Xu. Eararray: Defending against dolphinattack via acoustic attenuation. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2021.

[150] AliExpress. New original replacement wire control board volume button pcb for pb3 powerbeat earphone, 2022. https://www.aliexpress.com/item/1005003525462944.html.

[151] DROK. Bluetooth board, drok 12v audio receiver bluetooth, 2022. https://www.amazon.com/Bluetooth-DROK-Receiver-Electronics-Headphone/dp/B07P94Z9XR.

[152] ProtoSupplies. Ad620 instrumentation amplifier module, 2022. https://protosupplies.com/product/ad620-instrumentation-amplifier-module/.

[153] ChargerLAB. Review of Apple 35W dual USB-C compact power adapter, 2022. https://www.youtube.com/watch?v=aHdZu-m9y64.

[154] Android Developer. Documentation of manifest permission, 2022. https://developer.android.com/reference/android/Manifest.permission#MODIFY_AUDIO_SETTINGS.

[155] SHIKUES. AMS1117 1A bipolar linear regulator, 2022. https://datasheet.lcsc.com/szlcsc/2001081204_Shikues-AMS1117-1-2_C475600.pdf.

[156] Niels Brouwers, Marco Zuniga, and Koen Langendoen. Neat: A novel energy analysis toolkit for free-roaming smartphones. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys)*, pages 16–30, 2014.

[157] Xiao Ma, Peng Huang, Xinxin Jin, Pei Wang, Soyeon Park, Dongcai Shen, Yuanyuan Zhou, Lawrence K Saul, and Geoffrey M Voelker. eDoctor: Automatically diagnosing abnormal battery drain issues on smartphones. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 57–70, 2013.

[158] Abhinav Pathak, Y Charlie Hu, and Ming Zhang. Where is the energy spent inside my app? fine grained energy accounting on smartphones with eprof. In *Proceedings of the 7th ACM European Conference on Computer Systems*, pages 29–42, 2012.

[159] Tony Nasr, Sadegh Torabi, Elias Bou-Harb, Claude Fachkha, and Chadi Assi. ChargePrint: A framework for internet-scale discovery and security analysis of EV charging management systems. In *Proceedings of the Network and Distributed Systems Security (NDSS) Symposium*, 2023.

[160] Sebastian Köhler, Richard Baker, Martin Strohmeier, and Ivan Martinovic. Brokenwire: Wireless disruption of CCS electric vehicle charging. *arXiv preprint arXiv:2202.02104*, 2022.

[161] Chengdong Lin, Kun Wang, Zhenjiang Li, and Yu Pu. A workload-aware dvfs robust to concurrent tasks for mobile devices. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2023.

[162] Oppo. Airvooc – world leading wireless charging, 2022. https://www.oppo.com/en/newsroom/stories/airvooc-world-leading-wireless-charging/.

[163] Ossia. Cota: Real wireless power, 2022. https://www.ossia.com/cota.

[164] OpenATX. https://github.com/openatx/uiautomator2, 2022.

Appendix A

A.1 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 [2]. 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 A.1. Therefore, $\Delta\Phi(t)$ results in the perturbations on the inductive electromagnetic field $\Phi_s(t)$.

$$\Delta I(t) = \frac{V_s(t)}{\Delta R(t)} \Rightarrow \Delta\Phi(t) = \frac{\mu_0 N_s \Delta I(t)}{2r_s} = \frac{\mu_0 N_s V_s(t)}{2r_s \Delta R(t)} \tag{A.1}$$

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 [164] 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). Fig. A.1 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.

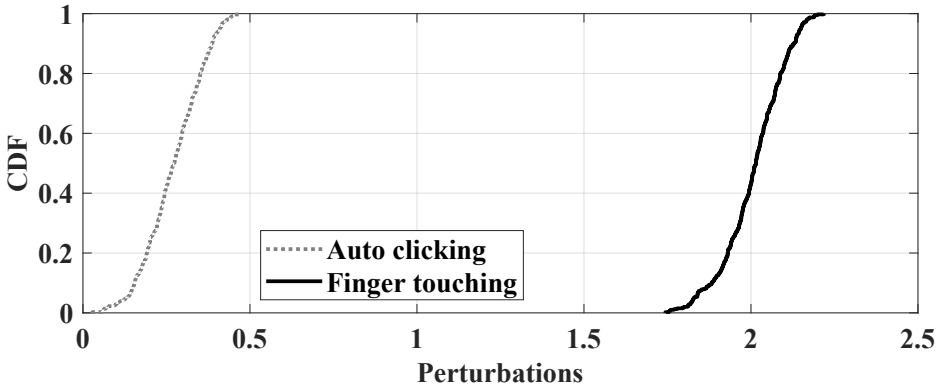


Fig. A.1 CDFs of automatic clicking and human touching.

Table A.1 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 A.2 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

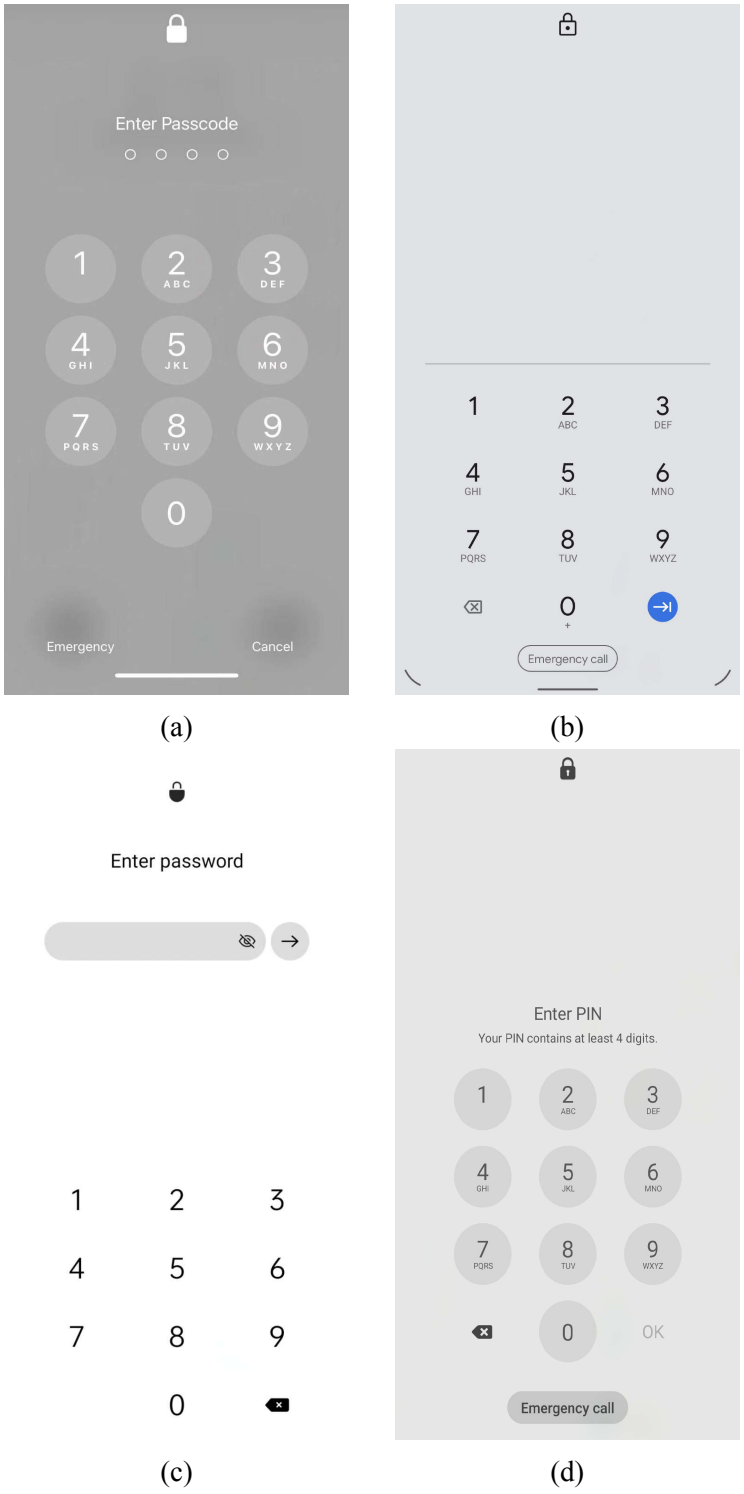


Fig. A.2 Unlocking keyboard layout of different smartphones. (a): iPhone 13 Pro, iPhone 12, and iPhone 11; (b): Google Pixel 4; (c) OnePlus 10 Pro; (d) Samsung S10.

Appendix B

B.1 List of Publications

1. **Tao Ni**, Xiaokuan Zhang, Qingchuan Zhao,“Recovering Fingerprints from In-Display Fingerprint Sensors via Electromagnetic Side Channel”, *ACM Conference on Computer and Communications Security (CCS), 2023*.

2. **Tao Ni**, Xiaokuan Zhang, Chaoshun Zuo, Jianfeng Li, Zhenyu Yan, Wubing Wang, Weitao Xu, Xiapu Luo, Qingchuan Zhao, “Uncovering User Interactions on Smartphones via Contactless Wireless Charging Side Channels”, *IEEE Symposium on Security and Privacy (S&P), 2023*.

3. **Tao Ni**, Guohao Lan, Jia Wang, Qingchuan Zhao, Weitao Xu, “Eavesdropping Mobile App Activity via Radio-Frequency Energy Harvesting”, *USENIX Security Symposium, 2023*.

4. **Tao Ni**, Jianfeng Li, Xiaokuan Zhang, Chaoshun Zuo, Wubing Wang, Weitao Xu, Xiapu Luo, Qingchuan Zhao, “Exploiting Contactless Side Channels in Wireless Charging Power Banks for User Privacy Inference via Few-shot Learning”, *ACM Annual International Conference on Mobile Computing and Networking (MobiCom), 2023*.

5. **Tao Ni**, Yongliang Chen, Weitao Xu, Lei Xue, Qingchuan Zhao, “XPorter: A Study of the Multi-Port Charger Security on Privacy Leakage and Voice Injection”, *ACM Annual International Conference on Mobile Computing and Networking (MobiCom), 2023*.

6. **Tao Ni**, Yongliang Chen, Keqi Song, Weitao Xu, “A Simple and Fast Human Activity Recognition System Using Radio Frequency Energy Harvesting”, *ACM*

International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp) CPD Workshop, ***Best Paper Award****, 2021*.

7. Zehua Sun, **Tao Ni**, Yongliang Chen, Di Duan, Kai Liu, Weitao Xu, "RF-Egg: An RF Solution for Fine-Grained Multi-Target and Multi-Task Egg Incubation Sensing", *ACM Annual International Conference on Mobile Computing and Networking (MobiCom), 2024*.

8. Zehua Sun, **Tao Ni**, Huanqi Yang, Kai Liu, Yu Zhang, Tao Gu, Weitao Xu, "FLoRa+: Energy-Efficient, Reliable, Beamforming-Assisted, and Secure Over-The-Air Firmware Update in LoRa Networks", *ACM Transactions on Sensor Networks (TOSN), 2024*.

9. Zehua Sun, **Tao Ni**, Huanqi Yang, Kai Liu, Yu Zhang, Tao Gu, Weitao Xu, "FLoRa: Energy-Efficient, Reliable, and Beamforming-Assisted Over-The-Air Firmware Update in LoRa Networks", *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), 2023*.

10. Yongliang Chen, **Tao Ni**, Weitao Xu, Tao Gu, "SwipePass: Acoustic-based Second-factor User Authentication for Smartphones", *ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT), 2022*.

11. Di Duan, Zehua Sun, **Tao Ni**, Shuaicheng Li, Xiaohua Jia, Weitao Xu, Tianxing Li, "F^2Key: Dynamically Converting Your Face into a Private Key Based on COTS Headphones for Reliable Voice Interaction", *ACM International Conference on Mobile Systems, Applications, and Services (MobiSys), 2024*.

12. Mingda Han, Huanqi Yang, **Tao Ni**, Di Duan, Mengzhe Ruan, Yongliang Chen, Jia Zhang, Weitao Xu, "mmSign: mmWave-based Few-Shot Online Handwritten Signature Verification", *ACM Transactions on Sensor Networks (TOSN), 2023*.

13. Keqi Song, Zimeng Zhu, Huanqi Yang, **Tao Ni**, Weitao Xu, "MobileKey: A Fast and Robust Key Generation System for Mobile Devices", *ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp) CPD Workshop, 2022*.