Password-Stealing without Hacking: Wi-Fi Enabled Practical Keystroke Eavesdropping

Jingyang Hu*
Hunan University
China
fbhhjy@hnu.edu.cn

Hongbo Wang*
Nanyang Technological University
Singapore
hongbo001@ntu.edu.sg

Tianyue Zheng
Nanyang Technological University
Singapore
tianyue002@ntu.edu.sg

Jingzhi Hu
Nanyang Technological University
Singapore
jingzhi.hu@ntu.edu.sg

Zhe Chen
Fudan University
China
zhechen@fudan.edu.cn

Hongbo Jiang
Hunan University
China
hongbojiang@hnu.edu.cn

Jun Luo
Nanyang Technological University
Singapore
junluo@ntu.edu.sg

ABSTRACT

The contact-free sensing nature of Wi-Fi has been leveraged to achieve privacy breaches, yet existing attacks relying on Wi-Fi CSI (*channel state information*) demand hacking Wi-Fi hardware to obtain desired CSIs. Since such hacking has proven prohibitively hard due to compact hardware, its feasibility in keeping up with fast-developing Wi-Fi technology becomes very questionable. To this end, we propose WiKI-Eve to eavesdrop keystrokes on smartphones without the need for hacking. WiKI-Eve exploits a new feature, BFI (*beamforming feedback information*), offered by latest Wi-Fi hardware: since BFI is transmitted from a smartphone to an AP in clear-text, it can be overheard (hence eavesdropped) by any other Wi-Fi devices switching to *monitor* mode. As existing keystroke inference methods offer very limited generalizability, WiKI-Eve further innovates in an adversarial learning scheme to enable its inference generalizable towards unseen scenarios. We implement WiKI-Eve and conduct extensive evaluation on it; the results demonstrate that WiKI-Eve achieves 88.9% inference accuracy for individual keystrokes and up to 65.8% top-10 accuracy for stealing passwords of mobile applications (e.g., WeChat).

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**; • **Computing methodologies → Machine learning**.

*Both authors contributed equally to this research; this work is done when Jingyang Hu works as a CSC visiting scholar at Nanyang Technological University (NTU).

KEYWORDS

Keystroke inference attack; password-stealing; Wi-Fi sensing; beamforming feedback information; wireless security.

ACM Reference Format:
Jingyang Hu, Hongbo Wang, Tianyue Zheng, Jingzhi Hu, Zhe Chen, Hongbo Jiang, and Jun Luo. 2023. Password-Stealing without Hacking: Wi-Fi Enabled Practical Keystroke Eavesdropping. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS'23), November 26–30, 2023, Copenhagen, Denmark*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3576915.3623088

1 INTRODUCTION

Mobile devices (e.g., smartphones and tablets), along with their software applications, have been increasingly adopted to identify human users in modern societies [5, 17]. Consequently, stealing passwords from these devices becomes almost like *identify theft*, hence attracting diversified eavesdropping attacks, either *direct* [42, 70] or *indirect* [2, 36, 37, 43, 53, 69, 73]. Bearing no need to have a visual on the target screen, the indirect attacks often incur a much higher risk as they leverage side-channels to infer passwords in a stealthy manner. Typical side-channels considered by prior works include acoustic [36, 73], electromagnetic emission [31], indirect vision [11, 53, 57, 69], and motion sensors [9, 37, 43]. Though they have demonstrated successes for particular scenarios, these successes often rely on strong assumptions [34], including i) eavesdropping devices are in proximity to the victim device (e.g., in centimeter scale or within line-of-sight region) [11, 31, 36, 37, 43, 53], ii) rogue software have been implanted to the victim device [9, 37, 73], and iii) the content to eavesdrop has linguistic structure [2, 37, 43, 69].

Among all side-channels, Wi-Fi CSI (*channel state information*) stands out as it appears to be void of all aforementioned weaknesses [34, 67]. Essentially, since keystrokes affect wireless channels as shown in Figure 1, the "twisted" CSIs can be used to infer individual keys involved in typing a password. The practical significance of this type of attack is also backed by the wide adoption of Wi-Fi infrastructure and extensive reach of Wi-Fi signals (thus CSIs). Nonetheless, this seemingly plausible attack actually bears

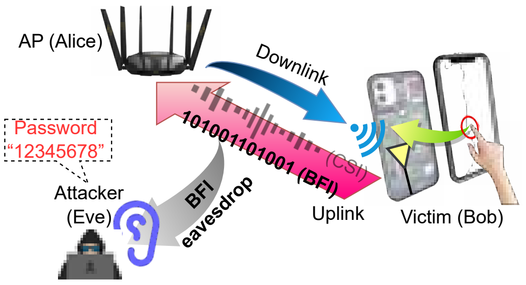


Figure 1: Vision of WiKI-Eve: eavesdropping clear-text BFI (representing downlink channel states) transmitted to the AP, Eve can readily infer the Bob's password typing that physically "hits" the Wi-Fi channel.

one fatal issue: though CSI was hacked[1] from Wi-Fi hardware more than a decade ago [25], only a handful of such hardware have been hacked by far while Wi-Fi standards/technologies are constantly getting upgraded every two or three years.[2] Therefore, it is highly questionable if CSI-based side-channel attacks are able to keep up with the technology developments, hence our passwords appear to remain secure.

Unfortunately, technology developments of Wi-Fi also introduce new vulnerability, as new Wi-Fi hardware (starting from Wi-Fi 5 [21]) piggybacks BFI (*beamforming feedback information*), a compressed *digital* version of *analog* CSI, in clear-text onto control frames. Basically, BFIs are used to feed downlink channel states back to an access point (AP), for the sake of guiding AP beamforming [1]. Though they only account for part of the downlink CSIs concerning the AP side, the fact that on-screen typing directly impacts the Wi-Fi antennas (hence channels) right behind the screen (see Figure 1) allows BFIs to contain sufficient information about keystrokes. Consequently, any device capable of overhearing Wi-Fi traffic (under the *monitor* mode [8]) may obtain BFIs for free. As shown in Figure 1, our proposal aims to take advantage of this new vulnerability, in order to achieve keystroke eavesdropping without the need for hacking the constantly evolving Wi-Fi hardware.

However, we still face two challenges for realizing this idea. On one hand, passwords lack linguistic structure in natural languages (e.g., word structure and occurrence frequency of letters) to serve as prior information and features; this has forced existing password inference methods to either rely on independent keystroke features [34] or leverage transition features between two keystrokes [67]. Nonetheless, as these features have strong environment dependency, the resulting inference methods can hardly be generalized to unseen scenarios. Although supervised learning techniques may address this issue with a dataset containing sufficient training data, gathering such a labeled dataset can be prohibitively difficult due to diversified smartphone models and human typing habits. On the other hand, BFIs, carried by control traffic, can be sparse and sporadic. This relatively minor issue, if not properly addressed, may exacerbate the data deficit challenge for training a password inference model.

To tackle these challenges, we propose WiKI-Eve to steal *numerical* passwords by eavesdropping on keystroke-induced BFI variations. Thanks to BFI's clear-text nature, no low-level hacking is needed on Wi-Fi hardware. Given the lack of linguistic structure in passwords, we follow the canonical way of identifying individual keystrokes, but we leverage a deep learning model with a natural segmentation as input to get rid of the artifacts introduced by rule-based segmentation and environment interference. We exploit *adversarial learning* [22] to extract features relevant only to individual keystrokes; such a cross-domain training is capable of generalizing keystroke inference to unseen scenarios with limited amount of training data, making WiKI-Eve achieve practical inference without having to gather a prohibitively large dataset. Furthermore, we design a sparse recovery algorithm to address the data deficiency issue for training the keystroke inference model. Finally, we implement a prototype of WiKI-Eve using a laptop or a rooted smartphone, and conduct extensive experiments on it to evaluate the performance of WiKI-Eve. In summary, our main contributions are:

- We propose WiKI-Eve as the first WiFi-based *hack-free* keystroke eavesdropping system; leveraging the clear-text BFI, it allows a wide range of Wi-Fi devices to eavesdrop on confidential passwords at ease.
- We innovate in leveraging adversarial learning to remove environment dependencies, rendering WiKI-Eve's inference model generalizable to unseen scenarios.
- We design a sparse recovery algorithm to address the sparsity issue of BFI, handling the data deficiency issue for training the keystroke inference model.
- We conduct extensive evaluations; the results indicate that WiKI-Eve achieves 88.9% accuracy for identifying single numerical keys, and a top-100 accuracy of 85.0% for inferring a 6-digit numerical password.

The paper is structured as follows. Section 2 introduces the background and motivation of our work. Section 3 presents the attack design of WiKI-Eve in detail. Sections 4 and 5 respectively explain WiKI-Eve's implementation and report the extensive evaluations on WiKI-Eve, followed by a discussion on extension from numerical to general keystroke inference. In Section 6, we study the impact of different background traffic on BFI/CSI data flow and discuss defense strategies against WiKI-Eve. Related works are briefly captured in Section 7. Finally, we conclude our paper in Section 8.

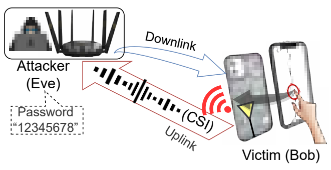
2 BACKGROUND AND MOTIVATION

In this section, we first introduce our *keystroke inference* (KI) attack scenario, contrasting it to those considered by existing Wi-Fi CSI-based proposals. Then we demonstrate the advantages of BFI over CSI for realizing the keystroke eavesdropping.

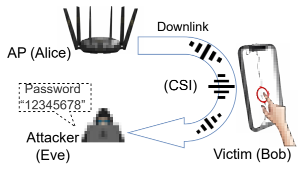
2.1 Attack Scenarios and Methods

We consider a scenario where a victim, Bob, uses his mobile device (smartphone or tablet) to connect to a Wi-Fi access point (AP) with a shared password or even no password protection; this is a reasonable assumption in public places such as shopping malls, office buildings, airports, and restaurants, because such a Wi-Fi access

[1]Instead of high-level software hacking, here we refer to a *low-level hacking*, including firmware patching [52] and driver modification [30], on Wi-Fi hardware.
[2]As a matter of fact, most research proposals driven by Wi-Fi CSIs are still leveraging the 15-year-old Wi-Fi 4 hardware [15].



(a) In-band KI (IKI).



(b) Out-of-band KI (OKI).

Figure 2: CSI-based keystroke inference (KI) methods.

is often provided for the convenience to customers or visitors. After connecting to the AP for accessing the Internet, Bob happens to have the need to access a sensitive account (e.g., online payment) protected by a *password*, which makes him a target of attack launched by Eve (see Figure 1). We follow the convention [34, 67] to mainly focus on numerical passwords, but we also consider an extension to general KI in Section 5.4.2. From here on, our method diverges from existing ones that either demand a rogue AP to trick Bob into using its service [19, 34], or require setting up extra Wi-Fi communication links to "sense" Bob's typing [2, 67].

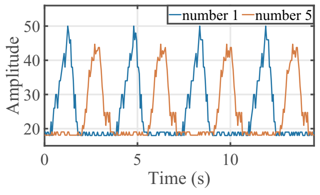
Essentially, WiKI-Eve's attack method allows Eve to launch an attack on Bob regardless of which AP Bob is connected to. It leverages only a laptop equipped with a network interface card (NIC); in fact, WiKI-Eve may even use a mobile device, as far as its Wi-Fi NIC can be switched to the monitor mode [8]. We term our method o-IKI (*overhearing in-band keystroke inference*), named after the IKI method proposed by Li *et al.* [34] where the Wi-Fi link (actually its CSI) between Bob and the AP is exploited for password-stealing, as shown in Figure 2(a). However, WiKI-Eve innovates in getting rid of the need for hacking a Wi-Fi NIC and tricking Bob to use it as an AP. This improvement of o-IKI over IKI is significant because, while the feasibility of hacking the continuous evolving Wi-Fi NICs is questionable (see Section 1), effectively deploying rogue AP has been made extremely challenging due to the increasing alerts raised by individuals and companies on such attacks [7, 13, 61].

Another method known as *out-of-band keystroke inference* (OKI) [2, 67], shown in Figure 2(b), requires Eve to create a separate channel irrelevant to Bob, using Eve's Wi-Fi NIC and another device (e.g., an AP). Eve then infers Bob's keystrokes by observing the CSIs of this channel. Compared with OKI relying on analog CSIs, the digital nature of o-IKI eavesdropping BFI leads to a significantly larger sensing range, while the in-band sensing for KI ensures a sufficiently high signal-to-noise ratio (SNR). Unlike IKI having Eve directly observing data traffic via its rogue AP [34], both o-IKI and OKI require Eve to be able to identify Bob's device: whereas this has proven very difficult to achieve under realistic scenarios for OKI's analog CSI sensing (given the low spatial resolution of Wi-Fi sensing [27, 71]), we shall demonstrate in Section 3.1 that there exists a natural solution for o-IKI's digital BFI eavesdropping.

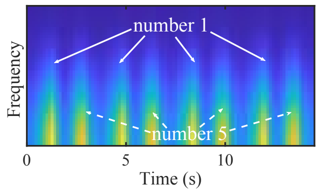
2.2 Why BFI instead of CSI?

BFI actually offers other advantages over CSI in terms of KI attack, apart from its easy acquirement explained earlier. To be specific, BFI behaves *less sensitive* to channel variation than CSI, rendering the sensing outcome more *stable* especially upon IKI's close impact (from on-screen keystrokes) on Wi-Fi channels. This stability stems from the way BFI is generated. Given the downlink CSI represented as $H = Y/X$, where X and Y respectively denote the transmitted (Tx) and received (Rx) signals [34], BFI is generated by partitioning H (hence the channels it represents) into separated Tx and Rx components; only the Tx component is fed back to the AP for guiding AP beamforming [1]. Thanks to this "channel splitting", BFI becomes less susceptible to channel variations caused by IKI's on-screen keystrokes, which otherwise leads to significant ambiguities in CSI-enabled KI.

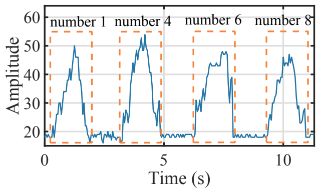
To showcase the superiority of BFI over CSI in KI, we conduct a series of experiments, leveraging iPerf [59] to generate saturated traffic and collecting only raw BFI and CSI samples; this temporarily neglects the sample sparsity issue to be elaborated in Section 3.4. In particular, Figure 3(a) and Figure 3(b) respectively depict the BFI time series and spectrograms for clicking numerical keys '1' and '5'



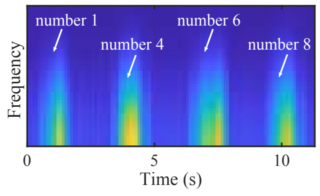
(a) Time series of the same keys.



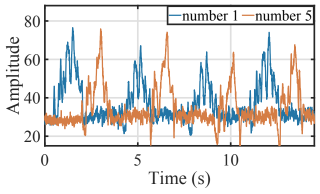
(b) Spectrogram of the same key.



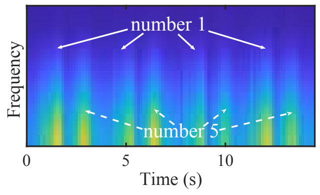
(c) Time series of different keys.



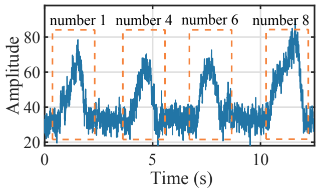
(d) Spectrogram of different keys.



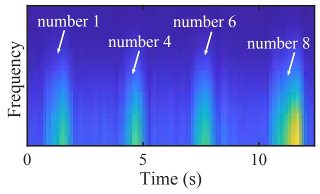
(e) Time series of the same keys.



(f) Spectrogram of the same key.

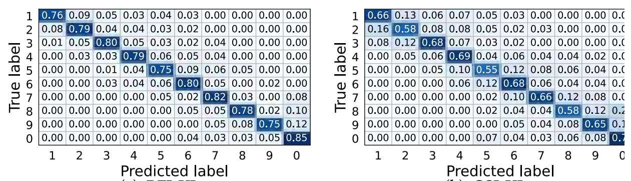


(g) Time series of different keys.



(h) Spectrogram of different keys.

Figure 3: BFI-KI (a)-(d) vs. CSI-KI (e)-(h): whereas BFIs exhibit both consistency for the same key and distinction for different keys, CSI's irregular patterns may cause ambiguities for keystroke inference.



(a) BFI-KI. (b) CSI-KI.

Figure 4: Confusion matrices for BFI- vs. CSI-based keystrol inference, demonstrating the superiority of BFI over CSI i completing this task.

four times. One may readily observe that the BFI patterns rema consistent for clicking the same keys at different times, while tl distinctions between two keys are also pronounced. Additional., Figure 3(c) and Figure 3(d), presenting BFI time series and spectrograms for clicking four different keys, again confirm the remarkable distinctions across these keys. In short, BFI is well-suited for KI with minimal preprocessing.

As a comparison, Figures 3(e) and 3(f), with the same contents as those for Figures 3(a) and Figures 3(b) but for CSIs collected simultaneously with the aforementioned BFIs, fail to indicate either remarkable consistence for the same key or pronounced distinctions between two different keys. Meanwhile, the four-key tests shown in Figures 3(g) and 3(h) also suggest the need for some heavy denoising before using CSIs for KI, as the distinctions between certain keys (e.g., '4' and '6') appear to be overwhelmed by noises. We suspect that such noises cannot be easily eliminated using conventional signal processing techniques, since their wide spectrum may confuse themselves with CSI features, as confirmed by the following KI test with denoised CSI and raw BFI.

We collect both BFI and CSI samples from 20 subjects typing numerical keys '0' to '9'. The same denoising technique in [34] is applied to the CSI samples, while the BFI samples are kept raw. We then use a one-dimensional convolutional neural network (1-D CNN) [33] to perform classification for the sake of KI and evaluate the KI accuracy by cross-validation. Figures 4(a) and 4(b) present the confusion matrices for BFI- and CSI-based KIs, respectively; these results evidently demonstrate that BFI achieves higher accuracy for individual keys, as indicated by the diagonal of the confusion matrix. Overall, the average accuracy achieved by BFI is 78.9%, notably higher than 64.5% achieved by CSI, confirming the benefit of BFI's stability over even denoised CSI in terms of realizing KI.

3 THE DESIGN OF WIKI-EVE

In this section, we introduce the attack strategy of WiKI-Eve. As shown in Figure 5, the whole workflow consists of five steps: i) identifying the victim, ii) determining the attack time when the victim accesses the targeted application service, iii) capturing the victim-associated BFI time series, iv) parsing and restoring the (possibly) sparse BFI series, and finally v) segmenting the BFI series and performing KI to recover the intended password. Key contributions in iv) and v) are respectively presented in Sections 3.4 and 3.3.

3.1 Victim Identification and Attack Timing

Following an implicit assumption of [34], we also allow Eve to have prior knowledge of Bob's device identity (e.g., MAC address). In reality, Eve can acquire this information beforehand by conducting visual and traffic monitoring concurrently: correlating network traffic originating from various MAC addresses with users' behaviors should allow Eve to link Bob's physical device to his digital traffic, thereby identifying Bob's MAC address. It is worth noting that victim identification is only possible through IKI, since the analog nature of OKI [2, 67] forbids the use of header information to differentiate multiple subjects.

Once locked onto Bob's MAC address, Eve waits for the right time (when Bob is about to enter his password) to launch attack. This timing issue can be readily addressed if visual hints are presented (e.g., Bob scan the WeChat Pay QR code or Bob's screen shows the payment page); otherwise, Eve can inspect the requests made to a payment service. Consider the case of WeChat [62], though most of its traffic is secured via application-layer encryption [65], IP addresses are not encrypted for the public Wi-Fi networks targeted by WiKI-Eve. To exploit this vulnerability, Eve creates a database of IP addresses associated with the payment service: though such IP addresses can be dynamic, our experiments reveal that users from the same region are directed to the same IP address within a certain period. Subsequently, upon detecting an IP address recorded in the database, the attack can be launched; the recording of BFI series will be stopped once no more requests to the IP can be observed.

3.2 BFI Analysis and Parsing

We hereby provide more details on how password typing can manifest in BFI to facilitate later developments, by first explaining how BFI is generated. As explained in Section 2.2, BFI is the Tx component of CSI H and is fed back to guide AP beamforming. This is accomplished by SVD (singular value decomposition) [56] that decomposes the channel as $H = USV$. Among these components, only the right matrix V is chosen as BFI, while the other two matrices U and S (representing Rx beamforming and channel gains, respectively) are not. As illustrated in Figure 6, Bob's password typing affects the diffraction pattern of the Wi-Fi signals around

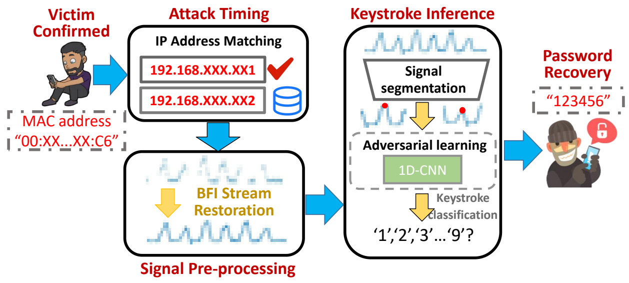


Figure 5: The workflow of WiKI-Eve's attack strategy.

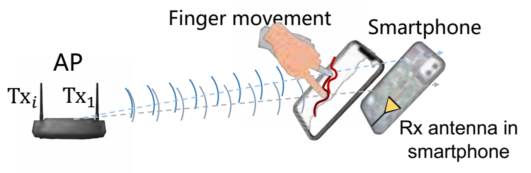


Figure 6: Finger movements cause diffraction on the downlink path, which is manifested in BFI variations.

the phone body. This altered pattern is then reflected in downlink CSI that is in turn decomposed with SVD to obtain BFI V.

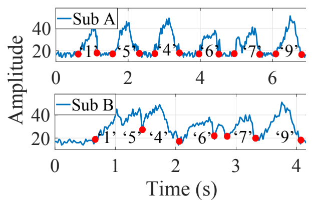
As BFI is transmitted in clear text, Eve can easily intercept it using a Wi-Fi device in monitor mode, along with Wireshark [47]. The frame structure of 802.11ac can be followed to locate BFI in the "VHT beamforming report" field within the Wi-Fi Action frames [1]. To completely extract BFI, the length of the field can be calculated based on the number of Tx and Rx antennas, as outlined in [21]. By continuously recording the BFIs in the Wi-Fi frames from Bob during the time window of Bob's password typing, Eve can obtain a time series of BFI samples correlated with the password. If the BFI time series is too sparse due to a low control frame rate during the time window, WiKI-Eve tries to restore it. To remain focused on the key component, we first explain the KI function in Section 3.3, and postpone the discussion on BFI restoration to Section 3.4.

3.3 Keystroke Inference

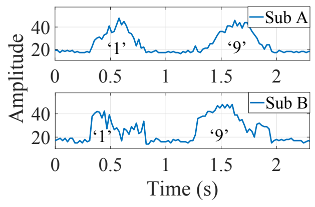
In this section, we elaborate on how WiKI-Eve conducts BFI-KI. We first discuss the drawbacks of previous proposals and explain possible improvements upon them. After that, we specify the signal segmentation on BFI series to kick off KI, which is then followed by the design of the KI neural model and its adversarial learning framework to generalize KI towards unseen scenarios.

3.3.1 What's Wrong with Prior Art? Only two major contributions exist for leveraging Wi-Fi side-channels to steal passwords. The seminal proposal of WindTalker [34] performs classification upon individual keystrokes with rule-based CSI series segmentation. Intuitively, such segmentation should not perform well because it can result in information loss or introduce artifacts. To confirm this suspicion, we ask two subjects to type passwords on their respective smartphones, and Figure 7(a) shows their corresponding CSI series. Apparently, the duration of the keystrokes and the amount of overlap between them vary significantly due to the subjects' distinct typing habits. While rule-based segmentation may be effective for Subject A who types more steadily, it most likely fails for Subject B whose inter-keystroke patterns appear rather messy. In attempting to forcibly assign different sections of the BFI series to individual keys, the segmentation process introduces artifacts (e.g., clipping) to each keystroke, potentially harming the KI performance.

A recent proposal WINK [67] claims to improve the KI performance via series learning. However, it inherits the rule-based segmentation adopted by [34] and hence the same weakness too. Additionally, as linguistic structure cannot be exploited for series learning, WINK argues that transition features between keystrokes may serve as replacements for improving KI accuracy. Unfortunately, factors such as typing habits and smartphone types can affect CSI during the transition period, resulting in different features for the same password. To illustrate this, we ask two subjects to type two keys '1' and then '9' on their phones, and Figure 7(b) shows significant morphological and temporal differences in these two transitions. Therefore, it is very questionable if transition features can ever replace linguistic structure.



(a) Segmentation ambiguity.



(b) Variance in transition features.

Figure 7: Two cases where previous methods fail.

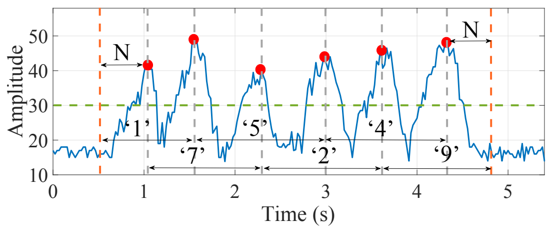


Figure 8: Signal segmentation with overlaps.

To overcome the disadvantages in previous proposals, the rule-based segmentation needs to be replaced with a more sensible method, preferably a data-driven one. Also, as using transition features to replace linguistic structure cannot be reliable, WiKI-Eve falls back to the canonical approach of inferring individual keystrokes as executed by WindTalker. To prevent information loss in segmentation, WiKI-Eve deems the environment-dependent transition periods as different "domains" of the same numerical keystroke. Consequently, an adversarial learning is exploited to train the KI model, aiming to remove domain interference (i.e., environment dependency) and hence generalize KI to unseen scenarios. Note that the data-driven nature of WiKI-Eve also prevents it from taking a series learning perspective, as it would otherwise demand a prohibitively large training dataset whose size grows exponentially with the password length.

3.3.2 Signal Segmentation. In reality, BFI series may not show distinct boundaries between consecutive keystrokes, significantly complicating signal segmentation. Figure 8 provides an example for such a case, where the BFI series displays prominent peaks corresponding to Bob's finger hitting the screen, as well as fluctuations between two peaks representing the transition movement of his fingers. Since the transitions carry information about both the preceding and succeeding keystrokes, segments of neighboring keystrokes should contain the transition. Therefore, we propose to employ an overlapping segmentation method that incorporates all data samples located between two consecutive peaks, from the preceding to the succeeding peaks, instead of the non-overlapping segmentation achieved by windows of rule-defined sizes [34, 67].

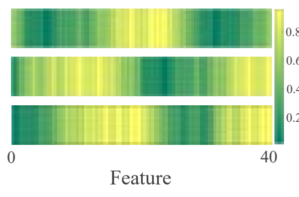
Our segmentation method starts with utilizing the Constant False Alarm Rate (CFAR) algorithm [45] to identify peaks in a BFI series. Suppose Bob typing a K-digit numerical password to produce a BFI series after sparse recovery (will be discussed in Section 3.4) of length L, the CFAR algorithm conducts statistical analysis on the series to determine an adaptive threshold and selects the peaks exceeding this threshold as candidates. Among these candidate peaks, we further eliminate minor ones within a distance of W sampling points from a major peak. We then select the top-K peaks corresponding to the K numbers in the password, assisted by an inter-peak distance of W sampling points, where $W = \alpha \times \frac{L}{K}$. For each peak, we include all the data samples between itself and its two neighboring peaks into the segment corresponding to a single

keystroke; since the first and last numbers in the passwor
preceding and succeeding numbers, we choose to extend
before and after as the segment boundaries, where N
We shall empirically determine the values of α and β in
As demonstrated in Figure 8, this approach effectively pa
BFI series (for password "175249") into segments corresp
individual keystrokes, while preserving the feature-rich t
between keystrokes caused by finger movements.

3.3.3 Adversarial Learning Framework. This section exp
adversarial learning is employed to generalize KI to unseen
Prior to that, we briefly describe the basic design of KI
The classification of time series is a well-established tasl
be effectively addressed using a 1-D CNN. However, as
in Section 3.3.2, the BFI segments may differ in lengt
a challenge to conventional 1-D CNNs. To overcome t
we employ an adaptive average pooling layer [26] to enl
flexibility of 1-D CNNs. To be specific, this layer auto
calculates the appropriate kernel size required to yiel
size output feature map, thus enabling 1-D CNNs to acco
inputs of varying lengths.

In fact, the direct deep learning approach mentioned al
looks the impact of the domain on each keystroke. Her
refers to the context arising from the diversified transiti
the preceding and to the succeeding keystrokes; it includ
tinctions caused by, for example, typing speed, inter-typi
larities, and the adjacent keystrokes. To illustrate this, we
the numerical key '1' in three different domains: '5-1-
and '4-1-2', and present their segments and correspondir
maps in Figure 9. Although the segments of key '1' under
domains, in Figure 9(a), exhibit a high degree of similarity
peak, the '1' in '6-1-8' displays drastic fluctuations duri
tions between neighboring keystrokes, while those in '5
'4-1-2' have rather smooth transitions. Such differences
tributed to larger channel variations induced by finger m
over greater distances between the keys in '6-1-8 '. Add
we show the feature heatmaps for different '1's after the
average pooling layer in Figure 9(b): the same key '1' in different domains exhibit distinct feature maps, thus posing significant challenges to the subsequent keystroke classifier.

The aforementioned domain interference entails the need for a



(b) Feature maps.

Figure 9: Difference in BFI segments and features maps of key '1' indicates the domain dependency of KI.

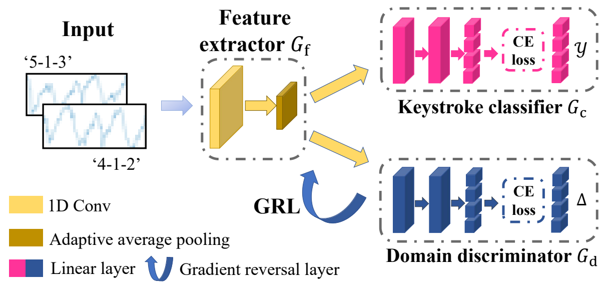


Figure 10: The training strategy enabled by adversarial learning removes domain-specific information.

in [48] could be challenging. Instead, WiKI-Eve aims to achieve a consistent feature space representation in different domains, by harnessing the power of *adversarial learning* [22] to integrate domain adaptation with KI in a unified training process. To incorporate adversarial learning, we revamp the training strategy of 1-D CNN as illustrated in Figure 10, whose training and inference processes are introduced as follows.

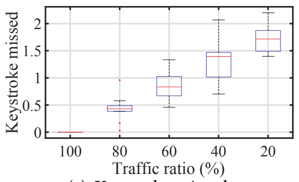
During the *training* phase, we first prepare a dataset consisting of randomly *paired* BFI segments corresponding to the same key (e.g., '1') but under different domains, e.g., two '6-1-8' from different passwords or a pair of '4-1-2' and '5-1-3'. We concatenate the pair as input $\boldsymbol{x}$ and process them through the feature extractor G_f. The resulting features are then fed into both the keystroke classifier G_c and domain discriminator G_d: G_c infers the key y shared by both segments within the pair, and G_d predicts the *domain discrepancy* $\Delta \in \{0, 1\}$, with 0 and 1 denoting the keys from the two segments *are* and *are not* from the same domain, respectively. While G_d aims to improve the accuracy of predicting Δ, the adversarial learning strategy "cheats" G_d by inverting its loss via reversing the gradient during backpropagation using the Gradient Reversal Layer (GRL) [20]; this procedure tends to suppress domain-specific features from the output of G_f and thus allows the 1-D CNN to learn keystroke representations invariant across domains. Denoting the parameters of G_f, G_c, and G_d as θ_f, θ_c, and θ_d, respectively, the above training procedure can be formulated as:

$$(\hat{\theta}_f, \hat{\theta}_c) = \arg\min_{\theta_f, \theta_c} \mathcal{L}(y, \Delta, \boldsymbol{x}), \quad \hat{\theta}_d = \arg\max_{\theta_d} \mathcal{L}(y, \Delta, \boldsymbol{x}),$$

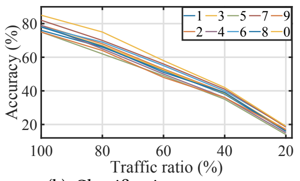
where $\mathcal{L}(y, \Delta, \boldsymbol{x}) = \mathcal{L}_c(y, G_c(G_f(\boldsymbol{x}))) - \lambda \mathcal{L}_d(\Delta, G_d(G_f(\boldsymbol{x}))$, $\mathcal{L}_c$ and $\mathcal{L}_d$ are respectively the cross-entropy losses for G_c and G_d, and λ, a balance factor controlling the trade-off between $\mathcal{L}_c$ and $\mathcal{L}_d$, should have its value empirically determined in Section 4. G_d is discarded during the *inference* phase, and the input of segment pair $\boldsymbol{x}$ is emulated by replicating the original BFI segment.

3.4 Recovering Sparse BFI Time Series

Another potential challenge to WiKI-Eve is traffic sparsity under extreme background traffic conditions: the BFI series may hence become temporally sparse, containing discontinuous samples that negatively impact the KI. To study how sparse traffic affects keystroke missed and classification accuracy, we use iPerf [59] to generate data traffics constantly exchanged between the device and AP. We define five different traffic ratios as 100% (saturated), 80%, 60%, 40%, and 20%. Given a certain ratio, the traffic generation follows a Poisson process [32]. Take the 6-digit password as an example,



(a) Keystroke missed.



(b) Classification accuracy.

Figure 11: Keystroke missing and affected classification accuracy under different traffic rates.

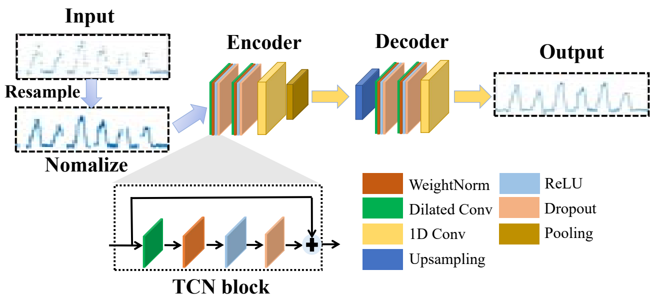


Figure 12: The neural models of WiKI-Eve's SRA.

relies on this representation to reconstruct a non-sparse series. The TCN-AE is trained in a self-supervised manner: we first generate non-sparse BFI series as ground truth using saturated traffic, then we randomly remove data samples to create sparse series that emulate realistic sparsity by following the temporal distribution of real-life BFI series generated under sparse traffic.

4 IMPLEMENTATION AND SETUP

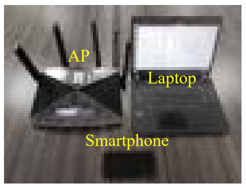
In this section, we elaborate on WiKI-Eve's implementation, as well as introduce the experiment setup and metrics.

System Implementation. Though a rooted smartphone under the monitor mode can act as Eve, Android systems offer minimal support in capturing Wi-Fi traffic at application layer. Therefore, we focus on a laptop implementation in our experiments. We use an Acer TravelMate laptop [29] with an Intel AX210 Wi-Fi NIC [16] supporting 802.11b/g/n/ac as the basis; setting the NIC to the monitor mode, we then use WireShark to capture the BFI series contained in Action No-ACK frames. The captured BFIs are analyzed using Matlab and Python, with the neural models built upon PyTorch 1.7.1 [49]. For the segmentation, the two parameters α and β are set to 0.6 and 0.5, respectively. In the adversarial learning framework, the balance factor λ is set to 0.5. For sparse recovery, the sampling frequency f_s is set to 40 Hz. Our collected data and code for preprocessing the data are publicly available online [63].

Experiment Setup. We recruit 20 subjects, of 12 males and 8 females, between the ages of 20 and 53. All subjects are right-handed and use their own smartphones of various models, including iPhone 13 [3], OnePlus 10T [46], Xiaomi 13 Pro [66], Huawei P40 Pro [28], Samsung Galaxy S20 [51], and Google Pixel 6a [23]. The subjects type a total of 1,500 predefined passwords of 4, 6, and 8 digits, with each length having 5,000 passwords. During typing, background apps remain active to emulate daily smartphone usage. The subjects adopt different postures while typing on the smartphones, such as holding it with one or both hands or placing it on a stand or table. The typing speed of the subjects ranges from 0.5 to 2 cps (*characters per second*). These experiments have strictly followed our IRB.

We conduct experiments and collect BFI series in six environments, including a library, bookstore, auditorium, cafeteria, corridor, and conference room. In each environment, a Wi-Fi router working as an AP for the subjects to connect. Besides BFI collection, we simultaneously obtain CSIs from the AP and another laptop to respectively serve as comparison baselines of WindTalker [34] and WINK [67]. The distance between a subject and the AP ranges from 1 to 10 m, and the distance between the attacker and the subject ranges from 3 to 10 m. Figure 13 shows an example experiment scene and the hardware we use. WiKI-Eve segments the BFI series using the overlapping scheme described in Section 3.3.2, while the baselines conduct segmentation according to their respective proposals [34, 67]. We use 70% of the collected data for training and the remaining 30% for testing.

Metrics. We adopt two metrics for our evaluations, namely keystroke *classification accuracy* and top-N password *inference accuracy*. For single keystroke classification, the classification accuracy measures the percentage of correctly classified keystrokes. For password inference, since an attacker may try multiple passwords to



(b) Adopted hardware.

Figure 13: Evaluative WiKI-Eve: (a) experiment scene in a conference room and (b) hardware configurations.

increase the success rate, we adopt the top-N accuracy as the evaluation metric: the probability of a candidate password is computed as the product of the probability of each key present in the password, then the top-N accuracy is measured by checking if any of the candidates within top-N probability matches the true one.

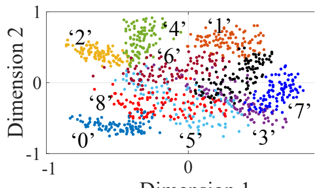
5 EVALUATION

We start with two micro-benchmark studies to demonstrate the effectiveness of WiKI-Eve's building blocks. These are followed by evaluations on overall performance and the impact of practical factors. Finally, we conduct real-world experiments to showcase how WiKI-Eve steals passwords of WeChat Pay, while also extending it to general KI on QWERTY keyboards.

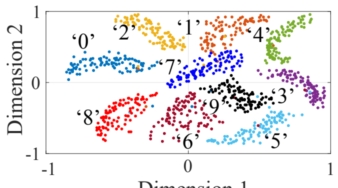
5.1 Micro-benchmark Studies

5.1.1 Domain Adaptation. To demonstrate the effectiveness of WiKI-Eve's adversarial learning framework in Section 3.3.3, we use t-SNE (t-Distributed Stochastic Neighbor Embedding) [41] to visualize the feature maps of 10 numerical keys segmented from 100 random passwords in Figure 14. As shown in Figure 14(a), the normal feature extractor G_f fails to find a domain-invariant feature map: features of different keys apparently get mixed together due to domain interference. In contrast, Figure 14(b) demonstrates that, with adversarial learning, the features of the same keys are consistent across domains and form distinct clusters, indicating that domain-invariant representations have been successfully learned.

5.1.2 Sparse Recovery. We apply SRA to recover the non-sparse BFI time series from passwords typed by a subject, and evaluate its effectiveness using the Root Mean Squared Error (RMSE) between the recovered and the ground truth series. As shown in Figure 15(a), our TCN-AE enabled SRA is able to recover a series with high similarity to the ground truth, capturing details when the subject's finger hits on screen as well as during transition periods. Furthermore, Figure 15(b) illustrates how RMSE changes with the proportion of missing BFI segments: even when 60% of the BFI segments are missing, SRA still achieves a sufficiently low RMSE at 3.3% of the mean amplitude of the ground truth series, indicating a successful recovery and providing a solid basis for WiKI-Eve's ultimate password inference.

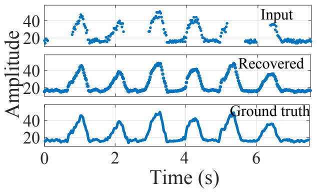


(a) w/o adversarial learning.

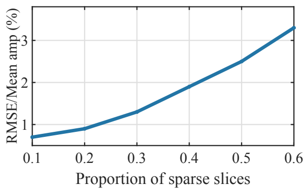


(b) w/ adversarial learning.

Figure 14: t-SNEs of the features output by the feature extractor G_f evidently confirm that adversarial learning results in domain-invariant representations.



(a) Illustration of sparse recovery.



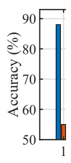
(b) Relative RMSE of sparse recovery.

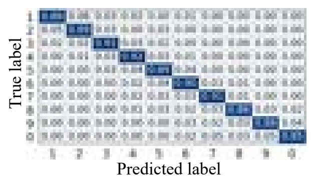
Figure 15: Performance of sparse recovery.

5.2 Overall Performance

5.2.1 Classification Accuracy. In this section, we present the accuracy of classifying numerical keys of WiKI-Eve, and compare it with two baseline methods WindTalker [34] and WiPOS [72]. We do not compare WiKI-Eve with WINK [67] in terms of keystroke classification accuracy because WINK is based on series learning that predicts the password as a whole. As shown in Figure 16(a), the classification accuracy of WiKI-Eve for keys '0' to '9' remains steady at around 88.9%, while WindTalker and WiPOS achieve an average accuracy of only 58.2% and 55.1%, respectively, which is significantly lower than that reported in [34], and should hence be further explained in Section 5.2.2. To further analyze the classification accuracy for each key, we present the confusion matrix of WiKI-Eve in Figures 16(b). It is intuitive that each key is most commonly confused with adjacent keys (e.g., the key '5' is most commonly confused with '2', '4', '6', and '8'). Despite the inevitable confusion, the high success rate of classifying individual keys lays a solid foundation for later password inference.

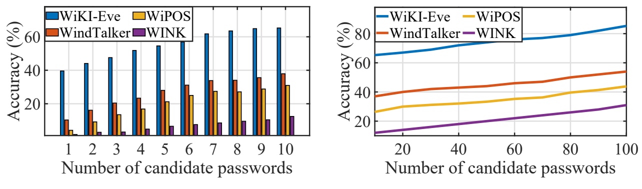
The superiority of WiKI-Eve over WindTalker and WiPOS can be attributed to two reasons. As discussed in Section 2.2, BFI dampens the close impact of IKI from its on-screen keystrokes, making WiKI-Eve more stable than CSI-based approaches. This allows WiKI-Eve to extract consistent features effectively learnable by its neural
models
noises
lapping





(b) Confusion matrix.

Figure 16: Comparing the classification accuracy of WiKI-Eve with WindTalker and WiPOS.



(a) Top-1 to top-10. (b) Top-10 to top-100.

Figure 17: Comparison for password inference accuracy under different numbers of password candidates.

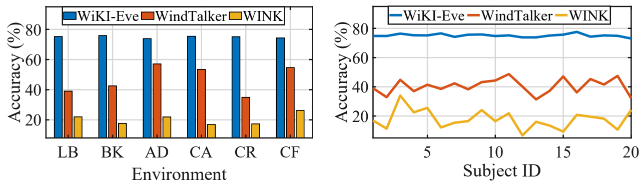
WiKI-Eve with richer domain "context" for its adversarial learning framework. In contrast, WindTalker's and WiPOS's rule-based segmentation potentially discards certain essential parts of the CSI features already overwhelmed by noises, thus failing to obtain full representations for individual keystrokes.

5.2.2 Password Inference Accuracy. Let us further evaluate WiKI-Eve's password inference capability, focusing on 6-digit numerical passwords due to their widespread usage in daily scenarios, but leaving the performance assessment for different password lengths to Section 5.3.6. Figure 17(a) compares the top-1 to -10 accuracy of WiKI-Eve, WindTalker, WiPOS, and WINK: while WiKI-Eve's accuracy varies from 40% to 65% for top-1 to -10 candidates, WindTalker's, WiPOS's, and WINK's only reach 37%, 32%, and 12% for top-10 accuracy, respectively. Figure 17(b) indicates that WiKI-Eve can infer passwords with an 85% success rate in 100 attempts, yet WindTalker, WiPOS, and WINK can only achieve a rate of 54%, 42%, and 31% at the same number of attempts.

The reasons for WiKI-Eve's superiority in Section 5.2.1 also apply to explain WiKI-Eve's much better performance in password inference than all baselines. Additionally, WiKI-Eve has an edge over WiPOS and WINK because o-IKI has a higher SNR than OKI, and the digital nature of BFI prevents fidelity loss of sensing signal. One may notice the performance discrepancy of all baselines from that reported in [34, 67, 72], as also highlighted in Section 5.2.1 for WindTalker. This may stem from their designs failing to properly take into account the influence of domain, thereby limiting their ability to effectively handle diverse data collected from various domains in our experiment setup.

5.3 Impact of Practical Factors

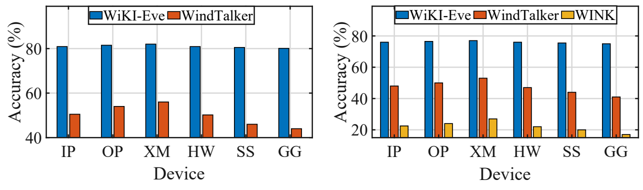
5.3.1 Environments and Subjects. We use the "leave-one-out" strategy [64] to study the impacts of different environments and subjects. This means that the test set consists of all data from one of the 6 environments or one of the 20 subjects, leaving the rest to the training set. Figure 18(a) and 18(b) respectively show the top-100 password inference accuracy for each environment and each subject. Although the testing environments and subjects are unseen during training, WiKI-Eve's top-100 accuracy across all cases is consistently above 75%, thanks to the generalizability of the adversarial learning. Moreover, WiKI-Eve is robust across environments since o-IKI relies on the diffraction pattern around the phone body that are rarely influenced by environment-specific interference. In contrast, the average top-100 accuracy of WindTalker and WINK drops from that in Figure 17 to less than 39% and 18%, due to their limited generalizability to unseen environments and subjects.



(a) Different environments. (b) Different subjects.

Figure 18: Impact of environments and subjects.

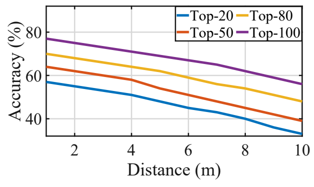
5.3.2 Device Diversity. We again use the "leave-one-out" strategy to evaluate the performance of WiKI-Eve on 6 smartphones specified in Section 4. Figure 19(a) shows that WiKI-Eve can reliably identify keystrokes on different devices, with an average keystroke classification accuracy of over 80%, but WindTalker's accuracy is under 58%. Furthermore, Figure 19(b) indicates that the top-100 password inference accuracy of WiKI-Eve, WindTalker, and WINK is respectively above 76%, below 53%, and below 27%. The consistently high accuracy of WiKI-Eve across different smartphone devices confirms that our adversarial learning framework can generalize to unseen devices. In contrast, the low accuracy of the baselines (evidently worse than the results in Figure 17) highlights their failure on unseen devices. One may also observe some accuracy variations among smartphones, which we attribute to different screen sizes. Specifically, WiKI-Eve achieves the highest accuracy on Xiaomi 13 Pro having the largest screen size (6.73 inch), while on Google Pixel 6a, with the smallest screen size (6.1 inch), it achieves the lowest accuracy. A possible explanation is smartphones with larger screens tend to have larger key distances that result in longer transition periods, thus making the incurred BFI features more distinguishable. Due to the consistently worse performance of the baselines, we do not compare WiKI-Eve with them in subsequent experiments.



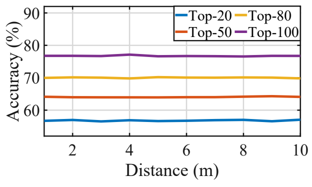
(a) Keystroke classification. (b) Password inference.

Figure 19: Impact of device diversity.

5.3.3 Distance. We evaluate the effect of distances on WiKI-Eve, i.e., the distances from Bob to the AP and from Eve to Bob. Figure 20 presents the top-20, 50, 80, and 100 password inference accuracy at various distances. Figure 20(a) shows that the average accuracy decreases by about 23% as the distance between Bob and the AP increases from 1 m to 10 m, because a longer distance from Bob to the AP weakens the Wi-Fi signal and takes in more interference. On the contrary, Figure 20(b) confirms that the distance between Eve and Bob barely affects the performance of WiKI-Eve, as the digital nature of BFI makes it robust to long-range transmission. Consequently, Eve can eavesdrop stealthily from a long distance without compromising inference accuracy, clearly demonstrating the advantage of WiKI-Eve's o-IKI method.



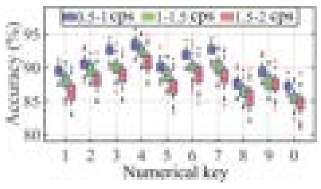
(a) Bob to the AP.



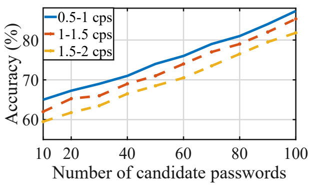
(b) Eve to Bob.

Figure 20: Impact of different distances.

5.3.4 Typing Speed. In this section, we examine how WiKI-Eve's performance varies with typing speeds. Figures 21(a) and 21(b) respectively show the keystroke classification and top-[1, 100] accuracy for tying speed ranges of [0.5, 1.0], [1.0, 1.5], and [1.5, 2.0] cps. As expected, both metric values decrease with higher typing speeds, probably due to stronger inter-typing irregularities. Nevertheless, WiKI-Eve still achieves sufficiently good performance in fast typing case with speed from [1.5, 2.0] cps, with only a minor decrease of around 3% in keystroke classification and less than 7% in password inference accuracy when compared with those in slow typing case with speed from [0.5, 1.0] cps. The relatively consistent performance of WiKI-Eve across different typing speeds is also the consequence of adopting adversarial learning.



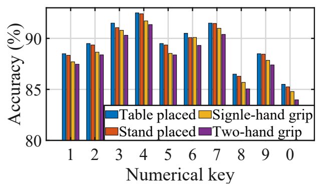
(a) Keystroke classification.



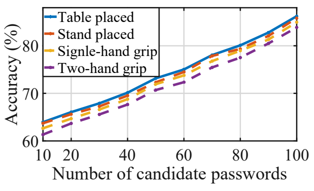
(b) Password inference.

Figure 21: Impact of typing speed.

5.3.5 Typing Scenarios. We further investigate the performance of WiKI-Eve across different typing scenarios, including holding the phone with one or both hands and placing the phone on a stand or a table. Figures 22(a) and 22(b) show that when the smartphone is placed on a stand or a table, WiKI-Eve achieves higher keystroke classification and password inference accuracy, likely due to the stability inherent to these scenarios. Despite the accuracy differences across scenarios, keystroke classification and password inference accuracy variations are less than 2.5%. These consistent results demonstrate that WiKI-Eve is robust to various occlusions and different typing scenarios, further validating the effectiveness of our adversarial learning framework.

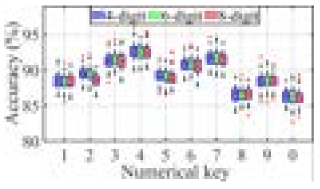


(a) Keystroke classification.

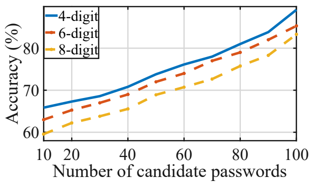


(b) Password inference.

Figure 22: Comparison of 4 different typing scenarios.



(a) Keystroke classification.



(b) Password inference.

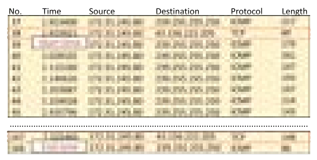
Figure 23: Impact of 3 different password length.

5.3.6 Password Length. We finally examine how password length affects WiKI-Eve's performance. Figure 23(a) demonstrates that the password length does not affect the accuracy of keystroke classification because WiKI-Eve treats each keystroke independently regardless of how many keys are typed. However, it significantly impacts password inference accuracy, as shown in Figure 23(b). For instance, the top-20 and top-100 accuracy for 4-digit numerical passwords is 69% and 89%, respectively, yet it becomes 64% and 83%, respectively, for 8-digit numerical passwords. The accuracy loss is attributed to the increased uncertainty caused by involving more keys. Nevertheless, even for an 8-digit numerical password, the remarkable success rate of 64% after 20 attempts still poses a severe threat to smartphone users.

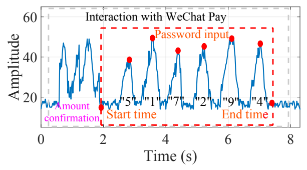
5.4 Real-World Experiment

5.4.1 WeChat Pay Password Inference. To showcase the practicality of WiKI-Eve, we conduct a real-world experiment by acting as Eve to steal password from WeChat Pay, a digital payment service integrated into WeChat [62]. The victim Bob uses an iPhone 13 for his daily activities, typically including WeChat usage, and he is supposed to make a mobile payment transaction with WeChat Pay, for which a numerical password is required, in a conference room of size 5 m × 8 m. The AP is placed on a table and the distance between Bob and the AP ranges from 1.5 to 5 m, as confined by the room layout. Meanwhile, Eve leverages WiKI-Eve to achieve a stealthy eavesdropping at a distance of 3 m from Bob.

Following the method in Section 3.1, WiKI-Eve first identifies Bob's Wi-Fi traffic; this is followed by detecting an IP address "43.156.222.205" coinciding with an entry in a pre-recorded IP database, as shown in Figure 24(a), which in turns starts BFI recording. The recording is stopped once no more requests to that address are made. Subsequently, WiKI-Eve performs SRA on the BFI time series, and the resulting non-sparse BFI series is shown in Figure 24(b). It appears that the BFI series includes not only the 6-digit numerical password but also other keys entered beforehand (e.g., the transfer amount and confirmation), so we extract the last six



(a) Attack timing identification.



(b) Targeted keystroke extraction.

Figure 24: Real-world experiment with WeChat Pay.

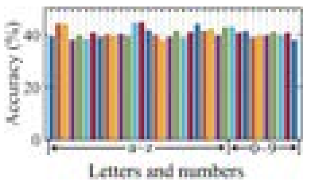
peaks corresponding to the password specifically for WeChat Pay, as highlighted by the red box.

After segmenting the signal, WiKI-Eve initiates the password inference. Since WeChat Pay freezes after five incorrect password inputs, we focus on identifying correct passwords among the top 5 candidates. In the experiment shown in Figure 24(b), the actual password entered by Bob is "517294", and the top 5 candidates are "547294", "517204", "**517294**", "517594", and "517394", indicating a successful password stealing. We conduct 50 such experiments in total, each with a different password. The results indicate that, out of these 50 input passwords, WiKI-Eve achieves a top-5 accuracy of 50%, which is quite close to that shown in Figure 17(a), albeit with a potentially biased statistics given only a small amount of trials. These experiments evidently demonstrate the practicality of WiKI-Eve in real-world scenarios.

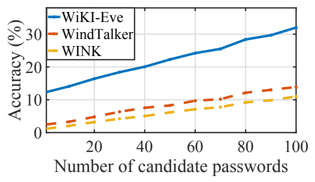
5.4.2 Extending to Virtual QWERTY Keyboard. Many applications need more diversified characters than what a numerical keyboard can offer. Typically, banking applications (e.g., the popular Chase Mobile [14]) handling sensitive financial transactions and identity information demand using a virtual (on-screen) QWERTY keyboard for users to create more secure *alphanumeric* passwords. To test the applicability of WiKI-Eve in such scenario, we conduct keystroke classification experiments on the QWERTY keyboard of Chase Mobile. We collect a dataset of 4,000 pre-defined passwords with varying lengths: 1,500 with 6 characters, 1,500 with 8 characters, and 1,000 with 10 characters. The passwords consist of lowercase letters from 'a' to 'z' and numbers from '0' to '9'. Except for the larger dataset size, we adopt the same experiment settings in Section 4.

Figure 25(a) shows that WiKI-Eve achieves an average keystroke classification accuracy of 40%. Additionally, Figure 25(b) indicates that WiKI-Eve's top-[1, 100] accuracy of 6-character alphanumeric password ranges from 12% to 32%, surpassing WindTalker and WINK whose top-100 accuracy is only 11% and 14%, respectively. Although the accuracy is lower than those in Section 5.2.1 and 5.2.2, it still poses a severe threat to smartphone users. The performance drop on QWERTY keyboards can be attributed to these keyboards having approximately four times more keys than numerical keyboards within the same area. Consequently, the BFI features of clicking different keys are less distinguishable due to their proximity. Additionally, shorter distances (hence shorter transition periods) among keys increase inter-keystroke interference, thereby decreasing KI accuracy.

We also find that KI on a QWERTY keyboard demands a much larger training dataset than on a numerical keyboard. According to Figure 26, WiKI-Eve performs similarly to the baselines when the training set is small. Fortunately, as the training set size increases from 1,000 to 4,000, WiKI-Eve begins to show its strengths: it improves the keystroke classification accuracy from 6% to 40%, and the top-100 accuracy from 6% to 32%, outperforming the baselines by large margins. The need for a large training set can be explained (again) by the drastic increase in the number of keys on QWERTY keyboards, along with the corresponding increase in the number of domains. By employing the adversarial learning framework, WiKI-Eve can fully utilize the training data and perform adequate KI under domain interference. In contrast, WindTalker and WINK struggle with interference and artifacts caused by a large number of domains, barely improving KI performance.

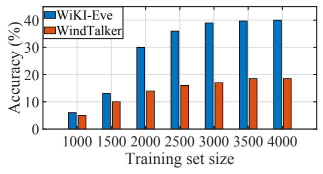


(a) Keystroke classification.

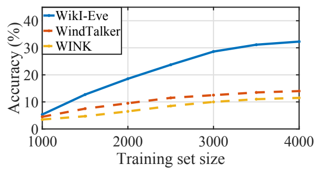


(b) Password inference.

Figure 25: Performance on QWERTY keyboards.



(a) Keystroke classification.



(b) Password inference.

Figure 26: Extending WiKI-Eve to QWERTY keyboards requires more training data.

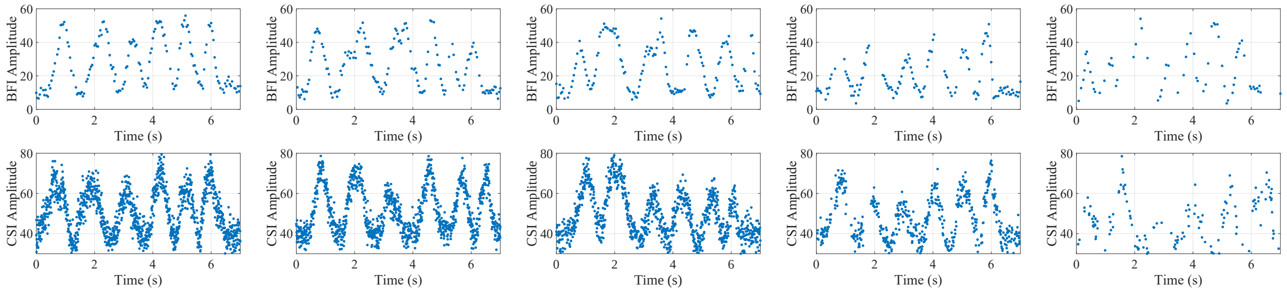
This experiment also reveals a few challenges to be tackled in future for general KI on QWERTY keyboards. First, more diversified password length should be considered, as over 20% of user may have passwords longer than 10 characters [55]. Second, handling more general passwords containing special characters and uppercase letters is also a crucial aspect: typing these characters may require combinations of multiple keys (e.g., "shift" and its paired keys) and thus complicating the BFI series. Third, certain applications have separate keyboard layouts for distinct groups of keys, requiring users to switch between layouts while entering passwords. Performing KI for such applications requires accurate detection of the layout switching, as well as training two separate neural models for each layout, potentially increasing system complexity. Instead of increasing training data in a brute-force manner, other side-channel attacks and social engineering techniques [24] may be combined with WiKI-Eve to enhance its KI capability in tackling these challenges.

6 TRAFFIC IMPACT AND DEFENSE

In this section, we first study the impact of five different background traffic on the sparsity of BFI (and CSI) time series, then we propose four different defense strategies against WiKI-Eve.

6.1 Background Traffic Analysis

To showcase how real-life background traffic intensities affect the sparsity of BFI (and CSI), we set five types of background traffic in (rate) descending order: (artificially) saturated traffic, video conferencing, software update, music streaming, and background chat. Take the 6-digit password as an example, the sparsity of the BFI (and CSI) time series varies with the intensity of the background traffic as shown in Figure 27. Figure 27(a) depicts the dense and continuous time series under saturated traffic. Since video conferencing and software update have their traffic patterns go very close to saturated ones, the resulting time series, as shown in Figures 27(b) and 27(c), again exhibit dense and continuous nature, which can



(a) Saturated traffic. (b) Video conferencing. (c) Software update. (d) Music streaming. (e) Background chat.

Figure 27: BFI (first row) and CSI (second row) time series under real-life background traffic.

be directly used for KI without enhanced by SRA. On the contrary, the time series under two types of low-rate background traffic, as shown in Figures 27(d) and 27(e), apparently behave much sparser, potentially demanding the assistance of SRA.

It is worth noting that, although the BFI (and CSI) time series can be sparse and bursty under real-life background traffic, we barely observe cases where a whole keystroke goes missing due to traffic sparsity. As a result, the BFI time series collected under real-life background traffic, even when sparse and bursty, can almost always be accommodated (hence enhanced) by our SRA presented in Section 3.4. In fact, all existing Wi-Fi-based password eavesdropping scheme [34, 67, 72] have to face the challenge of sparse background traffic, yet we are the first in rising to this challenge, enabling Wi-Fi based password eavesdropping under most real-life traffic conditions. The BFI and CSI data collected under real-life background traffic are online as specified in Section 4. Finally, WiKI-Eve can even steal passwords not sent over the WiFi (e.g., phone unlock) with sufficient background traffic. However, the challenge lies in acquiring the precise timing of the beginning and end of a password input process, for which limited visual cues might be the only feasible approach for now, as discussed in Section 3.1.

6.2 Defense Strategies

Since WiKI-Eve achieves keystroke eavesdropping by overhearing Wi-Fi BFI, the most direct defense strategy is to encrypt data traffic, hence preventing attackers from obtaining BFI in clear text. In fact, this strategy is commonly used in institutional Wi-Fi deployments, which indeed invalidates the basic assumption required by WiKI-Eve as stated in Section 2.1. However, this strategy may cause trouble for scenarios with high user dynamics, as frequently performing key exchanges substantially increases system complexity. One may consider keyboard randomization [34] as an indirect defense strategy, where a randomly keyboard layout is generated whenever a user attempts to enter password. By shifting the trouble to user side, this strategy, as indicated by [31], forces users to pay more effort when searching for keys on random keyboards, especially affects those used to relying on muscle memory to enter passwords without much visual aid.

A novel strategy against sensing attacks is signal obfuscation. In particular, IRShield [54] leverages IRSs (intelligent reflecting surfaces) installed beside an AP to physically scramble CSI, so as to thwart all sensing attempts. Unfortunately, this proposal goes against the current trend of evovling Wi-Fi towards ISAC (Integrated Sensing And Communications) [27] where legitimate sensing users should be catered. To this end, we suggest to exploit MIMO (multiple-in multiple-out) technology adopted by Wi-Fi hardware to scramble Wi-Fi channels [40]. Acting at physical layer, this strategy can be void of limitations inherent to earlier upper-level digital strategies (e.g., no need for per-user key generation), hence potentially applicable to a much wider range of application scenarios. Of course, this strategy requires hardware or firmware reconstruction, resulting in extra cost compared with digital stratdegyies. Fortunately, realizing ISAC framework by revising Wi-Fi architecture [12] is more and more recognized as a desirable development path.

7 RELATED WORKS

We classify existing KI proposals related to WiKI-Eve into the following five different categories:

Radio-Frequency. WiKey [2] pioneers in leveraging Wi-Fi CSI distortions induced by keystrokes to conduct KI, but the OKI mode used by WiKey is soon exceeded (in SNR) by the IKI mode introduced by WindTalker [34] for password inference, which is followed by Fang et al. [19] who exploit English linguistic structure to infer (non-password) keystrokes from CSI obtained via IKI. Recently, WiPOS [72] uses the OKI model for POS (point of sale) terminal keystroke eavesdropping. SpiderMon [35] attempts to perform passive serial keystroke eavesdropping using signals transmitted by commercial cell towers. WINK [67] also leverages OKI but claims that spatiotemporal analysis could enhance the performance of password inference.

Acoustic. Liu et al. [36] propose to classify keys on a keyboard based on the time difference of arrival of the acoustic signals (generated by pressing and releasing a key) at the two microphones on a smartphone. Similarly, KeyListener [39] performs KI on touchscreen based on different attenuation of the signals (generated by phone speaker) at the two microphones. PatternListener [73] compromises pattern locks by using acoustic signals reflected from fingertips to measure their relative movement and infer the pattern lines. These methods can be deemed as acoustic version of OKI.

Vision. Early vision-based KI attacks depend on directly observing the contents displayed on a screen [42, 70]. To make it more practical, later works explore side-channel visual cues. KI can be achieved by analyzing changes in the device's physical appearance, such as shadows and deformations on the screen [69], as well as backside motions of tablet computers [57]. Moreover, capturing videos of the victim's biometric features during typing, such as finger [53] and eye [11] movements, may also enable KI. Recent work [10] claims to achieve KI even when victims cover the typing hand with the other hand. Although vision-based side-channel attacks have shown a high success rate, the corresponding defense strategies [38, 50] have also grown mature and effective. Compared with the action features required by vision-based KI attacks, WiKI-Eve only requires visual hints (e.g., actions before starting input) rather than the complete input process, as explained in Section 3.1.

Motion Sensors. TouchLogger [9] uses the accelerometer and gyroscope on smartphones to capture phone body movement and infer numerical keys typed on its touchscreen. (sp)iPhone [43] leverages the accelerometer on a nearby phone to detect vibrations from a physical keyboard for enabling KI. Liu et al. [37] further exploit the accelerometer on a smartwatch to capture hand movement and infer keystrokes on POS terminals or QWERTY keyboards.

Electromagnetic Emission. Vuagnoux et al. [60] propose to eavesdrop on keystrokes from wired and wireless keyboards by capturing electromagnetic emissions during their communications. A later work Periscope [31] extends this idea to a broader range of mobile devices by exploiting human-coupled emission from touchscreens to estimate finger movement trajectories and infer numerical passwords. Vulnerabilities in USB data transfers have also been exploited for password-stealing [44] and malicious command execution [58]. Charger-Surfing [18] further demonstrates that, even without any data transfer over USB, variations of consumed power can be exploited to extract private information such as user passwords.

8 CONCLUSION

In this paper, we have proposed WiKI-Eve as the first Wi-Fi based KI attack with no need for hacking or specialized hardware, making it widely applicable to diversified Wi-Fi devices and attack scenarios. Moreover, WiKI-Eve's adversarial learning framework enables KI to be generalized towards unseen domains, further lifting its practical significance. Finally, we propose SRA to restore the sparse BFI series. Our extensive evaluations confirm that WiKI-Eve achieves sufficiently high inference accuracy for both individual keystrokes and numerical passwords, and we also tentatively explore extensions to general keyboards. Our results expose critical vulnerabilities in widely-used applications (e.g., WeChat) and hence underscore an urgent need for enhanced security measures against such risks.

ACKNOWLEDGEMENT

We are grateful to anonymous reviewers for their constructive suggestions. This research is support by National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme grant FCP-NTU-RG-2022-015, as well as MoE Tier 1 grant RG16/22. We also thank ERI@N and NTU-IGP for supporting the PhD scholarship of Hongbo Wang.

REFERENCES

[1] 2010. IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems–Local and Metropolitan Area Networks–Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 10: Mesh Networking. *IEEE P802.11s/D8.0, December 2010* (2010), 1–350.

[2] Kamran Ali, Alex X. Liu, Wei Wang, and Muhammad Shahzad. 2015. Keystroke Recognition using WiFi Signals. In *Proc. of the 21st ACM MobiCom.* 90–102.

[3] Apple Inc. 2023. Buy iPhone 13. https://www.apple.com/sg/shop/buy-iphone/iphone-13. Online; accessed 12 February 2023.

[4] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271* (2018).

[5] The World Bank. 2023. Mobile ID. https://id4d.worldbank.org/guide/mobile-id. Online; accessed 25 March 2023.

[6] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A Theory of Learning from Different Domains. *Machine Learning* 79 (2010), 151–175.

[7] Raheem Beyah and Aravind Venkataraman. 2011. Rogue-access-point Detection: Challenges, Solutions, and Future Directions. *IEEE Security & Privacy* 9, 5 (2011), 56–61.

[8] Jessey Bullock and Jeff T. Parker. 2017. *Wireshark for Security Professionals: Using Wireshark and the Metasploit Framework.* John Wiley & Sons.

[9] Liang Cai and Hao Chen. 2011. TouchLogger: Inferring Keystrokes on Touch Screen from Smartphone Motion. In *Proc. of the 6th USENIX Security HotSec.* 1–9.

[10] Matteo Cardaioli, Stefano Cecconello, Mauro Conti, Simone Milani, Stjepan Picek, and Eugen Saraci. 2022. Hand Me Your PIN! Inferring ATM PINs of Users Typing with a Covered Hand. In *Proc. of the 31st USENIX Security.* 1687–1704.

[11] Yimin Chen, Tao Li, Rui Zhang, Yanchao Zhang, and Terri Hedgpeth. 2018. EyeTell: Video-assisted Touchscreen Keystroke Inference from Eye Movements. In *Proc. of the 39th IEEE S & P.* 144–160.

[12] Zhe Chen, Tianyue Zheng, Chao Hu, Hangcheng Cao, Yanbing Yang, Hongbo Jiang, and Jun Luo. 2023. ISACoT: Integrating Sensing with Data Traffic for Ubiquitous IoT Devices. *IEEE Communications Magazine* 61, 5 (2023), 98–104.

[13] Inc. Cisco Systems. 2023. Cisco Wireless Controller Configuration Guide, Release 8.4. https://www.cisco.com/c/en/us/td/docs/wireless/controller/8-4/config-guide/b_cg84/wireless_intrusion_detection_system.html#rogue-ap-classification. Online; accessed 25 March 2023.

[14] JPMorgan Chase & Co. 2023. Mobile Banking Features with Chase Mobile App. https://www.chase.com/digital/mobile-banking. Online; accessed 25 March 2023.

[15] Intel Corporation. 2008. Intel Ultimate N WiFi Link 5300. https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/ultimate-n-wifi-link-5300-brief.pdf. Online; accessed 28 March 2023.

[16] Intel Corporation. 2023. Intel® Wi-Fi 6 AX201. https://www.intel.sg/content/www/xa/en/products/sku/130293/intel-wifi-6-ax201-gig/specifications.html. Online; accessed 25 March 2023.

[17] Rob Cover. 2015. *Digital Identities: Creating and Communicating the Online Self.* Academic Press.

[18] Patrick Cronin, Xing Gao, Chengmo Yang, and Haining Wang. 2021. Charger-Surfing: Exploiting a Power Line Side-Channel for Smartphone Information Leakage. In *Proc. of the 30th USENIX Security.* 681–698.

[19] Song Fang, Ian Markwood, Yao Liu, Shangqing Zhao, Zhuo Lu, and Haojin Zhu. 2018. No Training Hurdles: Fast Training-agnostic Attacks to Infer Your Typing. In *Proc. of the 25th ACM CCS.* 1747–1760.

[20] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised Domain Adaptation by Backpropagation. In *Proc. of the 32nd ACM ICML.* 1180–1189.

[21] Matthew S. Gast. 2013. *802.11ac A Survival Guide: Wi-Fi at Gigabit and Beyond.* O'Reilly Media, Inc.

[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proc. of 28th NeurIPS.* 2672–2680.

[23] Google LLC. 2023. Pixel 6a. https://store.google.com/product/pixel_6a?hl=en-GB. Online; accessed 10 April 2023.

[24] Christopher Hadnagy. 2010. *Social Engineering: The Art of Human Hacking.* John Wiley & Sons.

[25] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *ACM SIGCOMM Comput. Commun. Rev.* 41, 1 (2011), 53.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 9 (2015), 1904–1916.

[27] Jingzhi Hu, Tianyue Zheng, Zhe Chen, Hongbo Wang, and Jun Luo. 2023. MUSE-Fi: Contactless MUti-person SEnsing Exploiting Near-field Wi-Fi Channel Variation. In *Proc. of the 29th ACM MobiCom.* 1–15.

[28] Huawei Device Co., Ltd. 2023. HUAWEI P40 Pro. https://consumer.huawei.com/en/phones/p40-pro/. Online; accessed 10 April 2023.

[29] Acer Inc. 2023. Acer TravelMate Laptops for Business. https://www.acer.com/sg-en/laptops/travelmate. Online; accessed 25 March 2023.

[30] Zhiping Jiang, Tom H Luan, Xincheng Ren, Dongtao Lv, Han Hao, Jing Wang, Kun Zhao, Wei Xi, Yueshen Xu, and Rui Li. 2021. Eliminating the Barriers: Demystifying wi-fi Baseband Design and Introducing the Picoscenes Wi-Fi sensing Platform. *IEEE Internet of Things Journal* 9, 6 (2021), 4476–4496.

[31] Wenqiang Jin, Srinivasan Murali, Huadi Zhu, and Ming Li. 2021. Periscope: A Keystroke Inference Attack Using Human Coupled Electromagnetic Emanations. In *Proc. of the 28th ACM CCS*. 700–714.

[32] John Frank Charles Kingman. 1992. *Poisson Processes*. Vol. 3. Clarendon Press.

[33] Serkan Kiranyaz, Turker Ince, Osama Abdeljaber, Onur Avci, and Moncef Gabbouj. 2019. 1-D Convolutional Neural Networks for Signal Processing Applications. In *Proc. of the 44th IEEE ICASSP*. 8360–8364.

[34] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. 2016. When CSI Meets Public WiFi: Inferring Your Mobile Phone Password via WiFi Signals. In *Proc. of the 23rd ACM CCS*. 1068–1079.

[35] Kang Ling, Yuntang Liu, Ke Sun, Wei Wang, Lei Xie, and Qing Gu. 2020. Spider-Mon: Towards Using Cell Towers as Illuminating Sources for Keystroke Monitoring. In *Proc. of the 39th IEEE INFOCOM*. 666–675.

[36] Jian Liu, Yan Wang, Gorkem Kar, Yingying Chen, Jie Yang, and Marco Gruteser. 2015. Snooping Keystrokes with mm-level Audio Ranging on a Single Phone. In *Proc. of the 21st ACM MobiCom*. 142–154.

[37] Xiangyu Liu, Zhe Zhou, Wenrui Diao, Zhou Li, and Kehuan Zhang. 2015. When Good Becomes Evil: Keystroke Inference with Smartwatch. In *Proc. of the 22nd ACM CCS*. 1273–1285.

[38] Ziwei Liu, Feng Lin, Chao Wang, Yijie Shen, Zhongjie Ba, Li Lu, Wenyao Xu, and Kui Ren. 2023. CamRadar: Hidden Camera Detection Leveraging Amplitude-modulated Sensor Images Embedded in Electromagnetic Emanations. *Proc. of the 23rd ACM UbiComp* 6, 4 (2023), 1–25.

[39] Li Lu, Jiadi Yu, Yingying Chen, Yanmin Zhu, Xiangyu Xu, Guangtao Xue, and Minglu Li. 2019. KeyListener: Inferring Keystrokes on QWERTY Keyboard of Touch Screen through Acoustic Signals. In *Proc. of the 38th IEEE INFOCOM*. 775–783.

[40] Jun Luo, Hangcheng Cao, Hongbo Jiang, Yanbing Yang, and Zhe Chen. 2024. MIMOCrypt: Multi-User Privacy-Preserving Wi-Fi Sensing via MIMO Encryption. In *Proc. of the 45th IEEE S&P*. 1–19.

[41] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data Using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.

[42] Federico Maggi, Alberto Volpatto, Simone Gasparini, Giacomo Boracchi, and Stefano Zanero. 2011. A Fast Eavesdropping Attack against Touchscreens. In *Prof. of the 7th IAS*. IEEE, 320–325.

[43] Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. 2011. (sp)iPhone: Decoding Vibrations from Nearby Keyboards using Mobile Phone Accelerometers. In *Proc. of the 18th ACM CCS*. 551–562.

[44] John V. Monaco. 2018. SoK: Keylogging Side Channels. In *Proc. of the 39th IEEE S&P*. 211–228.

[45] Ramon Nitzberg. 1972. Constant-false-alarm-rate Signal Processors for Several Types of Interference. *IEEE Trans. Aerospace Electron. Systems* 1 (1972), 27–34.

[46] OnePlus. 2023. OnePlus 10T 5G. https://www.oneplus.com/sg/10t. Online; accessed 10 April 2023.

[47] Angela Orebaugh, Gilbert Ramirez, and Jay Beale. 2006. *Wireshark & Ethereal Network Protocol Analyzer Toolkit*. Elsevier.

[48] Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2010. Domain Adaptation via Transfer Component Analysis. *IEEE Transactions on Neural Networks* 22, 2 (2010), 199–210.

[49] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv preprint arXiv:1912.01703* (2019).

[50] Sriram Sami, Sean Rui Xiang Tan, Bangjie Sun, and Jun Han. 2021. LAPD: Hidden Spy Camera Detection Using Smartphone Time-of-flight Sensors. In *Proc. of the 19th ACM SenSys*. 288–301.

[51] Samsung. 2023. Samsung Galaxy S20 Series. https://www.samsung.com/sg/news/local/galaxy-s20-launch/. Online; accessed 10 April 2023.

[52] Matthias Schulz, Daniel Wegemer, and Matthias Hollick. 2017. Nexmon: The C-based Firmware Patching Framework. https://nexmon.org

[53] Diksha Shukla, Rajesh Kumar, Abdul Serwadda, and Vir V. Phoha. 2014. Beware, Your Hands Reveal Your Secrets!. In *Proc. of the 21st ACM CCS*. 904–917.

[54] Paul Staat, Simon Mulzer, Stefan Roth, Veelasha Moonsamy, Markus Heinrichs, Rainer Kronberger, Aydin Sezgin, and Christof Paar. 2022. IRShield: A Countermeasure Against Adversarial Physical-layer Wireless Sensing. In *Proc. of the 43rd IEEE S & P*. 1705–1721.

[55] statista. 2023. Average Number of Characters for a Password in the United States in 2021. https://www.statista.com/statistics/1305713/average-character-length-of-a-password-us/. Online; accessed 25 March 2023.

[56] Gilbert W. Stewart. 1993. On the Early History of the Singular Value Decomposition. *SIAM Rev.* 35, 4 (1993), 551–566.

[57] Jingchao Sun, Xiaocong Jin, Yimin Chen, Jinxue Zhang, Yanchao Zhang, and Rui Zhang. 2016. Visible: Video-assisted Keystroke Inference From Tablet Backside Motion. In *Proc. of the 23rd ISOC NDSS*.

[58] Dave Jing Tian, Grant Hernandez, Joseph I Choi, Vanessa Frost, Christie Raules, Patrick Traynor, Hayawardh Vijayakumar, Lee Harrison, Amir Rahmati, Michael Grace, et al. 2018. ATtention Spanned: Comprehensive Vulnerability Analysis of AT Commands Within the Android Ecosystem. In *Proc. of the 27th USENIX Security*. 273–290.

[59] Ajay Tirumala. 1999. iPerf: The TCP/UDP Bandwidth Measurement Tool. *http://dast.nlanr.net/Projects/Iperf/* (1999).

[60] Martin Vuagnoux and Sylvain Pasini. 2009. Compromising Electromagnetic Emanations of Wired and Wireless Keyboards. In *Proc. of the 18th USENIX Security*, Vol. 8. 1–16.

[61] Chen Wang, Xiuyuan Zheng, Yingying Chen, and Jie Yang. 2016. Locating Rogue Access Point using Fine-grained Channel Information. *IEEE Transactions on Mobile Computing* 16, 9 (2016), 2560–2573.

[62] WeChat. 2023. WeChat - Free Messaging and Chatting App. https://www.wechat.com/. Online; accessed 28 March 2023.

[63] WiKI-Eve. 2023. https://github.com/Nest-Fi/WiKI-Eve. Online; accessed 6 August 2023.

[64] Tzu-Tsung Wong. 2015. Performance Evaluation of Classification Algorithms by k-fold and Leave-one-out Cross Validation. *Pattern Recognition* 48, 9 (2015), 2839–2846.

[65] Songyang Wu, Yong Zhang, Xupeng Wang, Xiong Xiong, and Lin Du. 2017. Forensic Analysis of WeChat on Android Smartphones. *Digital Investigation* 21 (2017), 3–10.

[66] Xiaomi. 2023. Xiaomi 13 Pro. https://www.mi.com/sg/product/xiaomi-13-pro/. Online; accessed 10 April 2023.

[67] Edwin Yang, Qiuye He, and Song Fang. 2022. WINK: Wireless Inference of Numerical Keystrokes via Zero-Training Spatiotemporal Analysis. In *Proc. of the 29th ACM CCS*. 3033–3047.

[68] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. 2019. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural computation* 31, 7 (2019), 1235–1270.

[69] Qinggang Yue, Zhen Ling, Xinwen Fu, Benyuan Liu, Kui Ren, and Wei Zhao. 2014. Blind Recognition of Touched Keys on Mobile Devices. In *Proc. of the 21st ACM CCS*. 1403–1414.

[70] Qinggang Yue, Zhen Ling, Xinwen Fu, Benyuan Liu, Wei Yu, and Wei Zhao. 2014. My Google Glass Sees Your Passwords. *Prof. of the Black Hat USA* (2014).

[71] Shujie Zhang, Tianyue Zheng, Hongbo Wang, Zhe Chen, and Jun Luo. 2022. Quantifying the Physical Separability of RF-based Multi-Person Respiration Monitoring via SINR. In *Proc. of the 20th ACM SenSys*. 47–60.

[72] Zijian Zhang, Nurilla Avazov, Jiamou Liu, Bakh Khoussainov, Xin Li, Keke Gai, and Liehuang Zhu. 2020. WiPOS: A POS Terminal Password Inference System Based on Wireless Signals. *IEEE Internet of Things Journal* 7, 8 (2020), 7506–7516.

[73] Man Zhou, Qian Wang, Jingxiao Yang, Qi Li, Feng Xiao, Zhibo Wang, and Xiaofeng Chen. 2018. PatternListener: Cracking Android Pattern Lock using Acoustic Signals. In *Proc. of the 25th ACM CCS*. 1775–1787.