Revealing Your Mobile Password via WiFi Signals: Attacks and Countermeasures

Yan Meng, Jinlei Li, Haojin Zhu, *Senior Member, IEEE*, Xiaohui Liang, *Member, IEEE*, Yao Liu, *Member, IEEE*, and Na Ruan, *Member, IEEE*

Abstract—In this study, we present WindTalker, a novel and practical keystroke inference framework that can be used to infer the sensitive keystrokes on a mobile device through WiFi-based side-channel information. WindTalker is motivated from an observation that keystrokes on mobile devices will lead to different hand coverage and the finger motions, which will introduce a unique interference to the multi-path signals and can be reflected by the channel state information (CSI). An attacker can exploit the strong correlation between the CSI fluctuation and the keystrokes to infer the user's password input. Compared with the previous keystroke inference approaches, WindTalker neither deploys external equipment physically close to the target device nor compromises the target device. Instead, it employs a more practical setting by deploying a free public WiFi hotspot and collects the CSI data from the target device as long as the device is connected to the hotspot. In addition, to improve inference accuracy and efficiency, it analyzes the WiFi traffic to selectively collect CSI only for the sensitive period where password entering occurs. WindTalker can be implemented without the requirement of visually seeing the target device, or installing any malware on the device. We tested Windtalker on several mobile phones and performed a detailed case study to evaluate the practicality of the password inference towards Alipay, the largest mobile payment platform in the world. Furthermore, we proposed a novel CSI obfuscation countermeasure to thwart the inference attack. The evaluation results show that the performance of WindTalker can be dramatically reduced by adopting the proposed countermeasures.

Index Terms—Channel state information, online payment, password inference, traffic analysis, wireless security

1 Introduction

Smartphones and tablets are widely used for performing privacy sensitive transactions of banking, payment, and social applications. Unlike stationary devices connecting to a secure network and sitting in a physically-secure space, these mobile devices are often carried by a mobile user and connected to a dynamic network where attackers can physically approach the target user's device and launch various direct and indirect eavesdropping attacks. While direct eavesdropping attacks aim at directly observing the input of the target device from its screen or keyboard, indirect eavesdropping attacks, a.k.a. side-channel attacks make use of side channels to infer the inputs on the target devices. Prior works [2], [3], [4], [5], [6], [7], [8], [9], [10] have shown that both types of attacks can be effective in certain conditions. Particularly for the side-channel attacks, it is shown that the PIN number and the words entered at keyboard can be inferred from the electromagnetic signal at radio antenna [2], the acoustic signal at microphone [3], [4], [5], the visible light at camera [6], [7], and the motion status at motion sensors [8], [9], [10]. To access the side channels, these works often assume either external signal collector devices are physically close to the target device (for example, 30 cm in [2]) or the sensors of the target devices are compromised to provide side channel information. However, in a mobile scenario, both assumptions are hardly true and the impact of attacks is thus limited. In addition, the prior works have studied the keystroke inference aiming at achieving a high inference accuracy on a series of keystrokes during a relatively-long period of time. However, the keystrokes on a mobile device are not always highly sensitive. Obviously, the eavesdropping attacker has a greater interest in obtaining the payment PIN number in a short moment than a regular typing information. Therefore, to increase the inference accuracy and efficiency, the application context information needs to be considered in the keystroke inference framework.

In this paper, we present WindTalker, a novel and practical keystroke inference framework that can be used to infer sensitive keystrokes on a mobile device through WiFi signals. WindTalker is motivated from an observation that the typing activity on mobile devices involves hand and the finger motions, which produce a recognizable interference to the multi-path WiFi signals from the target device to the WiFi router that connects to the device. Unlike prior side-channel attacks or traditional CSI based gesture recognition, WindTalker neither deploys external devices close to the target device nor compromises any part of the target device; instead, WindTalker setups a 'rogue' hotspot to lure the target user with free WiFi service, which is easy-to-deploy and

- Y. Meng, J. Li, H. Zhu, and N. Ruan are with the Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: {yan_meng, ricardolee}@sjtu.edu.cn, {zhu-hj, naruan}@cs.sjtu.edu.cn.
- X. Liang is with the Department of Computer Science, University of Massachusetts at Boston, Boston, MA 02125. E-mail: xiaohui.liang@umb.edu.
- Y. Liu is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 02125. E-mail: yliu@cse.usf.edu.

Manuscript received 1 Oct. 2017; revised 14 Oct. 2018; accepted 8 Jan. 2019. Date of publication 16 Jan. 2019; date of current version 6 Jan. 2020. (Corresponding author: Haojin Zhu.) Digital Object Identifier no. 10.1109/TMC.2019.2893338

1536-1233 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

difficult-to-detect. As long as the target mobile device is connected to the rogue WiFi hotspot, WindTalker intercepts the WiFi traffic and selectively collects the channel state information (CSI) between the target device and the hotspot.

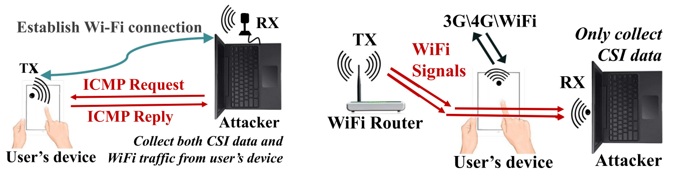
WindTalker has three major technical challenges. i) The impact of the hand and finger movement of keystrokes on CSI waveforms is very subtle. An effective signal analysis method is needed to analyze keystrokes from the limited CSI. ii) The prior CSI collection method requires two WiFi devices, one as a signal sender and the other as a signal receiver, which are deployed close to the victim. A more flexible and practical CSI collection method is highly desirable for the mobile device scenario. iii) The key inference must be done at some selective moments for obtaining a sensitive keystroke, such as payment PIN number. Such context-oriented CSI collection has not been addressed by prior works. The contributions of our paper are as follows.

- We present a practical cross-layer based approach for mobile payment password inference on smartphones using public WiFi architecture. We propose a novel password inference model which analyzes both physical layer information (CSI) and network layer traffic.
- We present a novel ICMP-based CSI collection method, without compromising the victim's device or deploying an external device very close to the victim's device. We develop an IP pool based method to recognize the PIN input period. And we propose an effective keystroke inference algorithm based on the collected CSI.
- We perform extensive evaluations on password inference at the mobile payment platform Alipay, which is secured by the HTTPS protocol and thus traditionally believed to be secure. We investigate the impact of various factors on WindTalker and we demonstrate that WindTalker can infer the PIN number at a high successful rate.
- We introduce some effective countermeasures to thwart the inference attack. Especially, we propose a novel CSI obfuscation algorithm to prevent the attacker to collect the accurate CSI data without the requirements of user's participation. This countermeasure could minimize the impact on the user experience and we perform experiments to prove its ability to reduce the attacking performance of WindTalker.

The remainder of this paper is organized as follows. In Section 2, we introduce the background of this work. In Section 3, we introduce the research motivation by showing the correlation of keystroke and CSI changing. We present the detailed design in Section 4, which is followed by Evaluation, Real-world experiment, Impact of various factors, Countermeasures, Limitations and Related work in Sections 5, 6, 7, 8, 9 and 10, respectively. Finally, we conclude this paper in Section 11.

2 Background

In this section, we introduce our attack scenario, the overview of the keystroke inference methods, and preliminaries of channel state information.



(a) IKI Model (b) OKI Model

Fig. 1. WiFi-based keystroke inference models.

2.1 Scenario

We consider a scenario where a user has a mobile device, such as a smartphone and he or she is using the free public WiFi through the device. It is a very common situation that people could have in the shopping mall, the airport, and restaurants. A WiFi hotspot is set up at a corner or on the ceiling, an unnoticeable location from the user's view. The user searches all the available WiFi signals at her device, and chooses to connect with the WiFi hotspot if the name of the hotspot "looks" good and the hotspot is authentication-free. With the use of application layer security protocol (HTTPS), the user may believe that the Internet traffic is protected from end-to-end such that the content shown at the device and the user's inputs at the device will be only known to herself and the service provider. However, as we will show, our WindTalker framework presents an effective keystroke inference method targeting at the mobile device.

2.2 In-Band Keystroke Inference Model

Different with existing works [2], [11], [12], WindTalker chooses *In-band Keystroke Inference (IKI)* model. As shown in Fig. 1a, WindTalker deploys one Commercial Off-The-Shelf (COTS) WiFi device close to the target device, which could be a WiFi hotspot. The WiFi hotspot provides free WiFi networks for nearby users. When a user connects her device to the hotspot, the WiFi hotspot is able to monitor the application context by checking the pattern of the transmitted WiFi traffic. In addition, the WiFi hotspot periodically sends ICMP packets to obtain the CSI information from the target device. With the meta data of the WiFi traffic collected by hotspot, WindTalker knows when the sensitive operations (such as typing password) happen. And then, the hotspot adaptively launches CSI-based keystroke inference method to recognize sensitive key inputs. To the best of our knowledge, the IKI method we propose is the first one using existing network protocols of IEEE 802.11n/ac standard to obtain the application context and the CSI information at mobile devices.

Note that the existing works about CSI based gesture recognition choose another strategy: *Out-of-band Keystroke Inference (OKI)* model [2]. In this model, the adversary deploys two COTS WiFi devices close to the target device and makes sure the target device is placed right between two COTS WiFi devices. One is the sender device continuously emitting signals and the other one is the receiver device continuously receiving the signals. The keystrokes are inferred from the multi-path distortions in signals.

Compared with OKI model, the proposed IKI model has the following advantages. First, IKI model does not require the placement of both sender and receiver device and can be deployed in a more flexible and stealthy way. Second, in OKI model, the user device is not connected with attacker's

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

device, so that the attacker cannot obtain the WiFi traffic from the user's device. Therefore, OKI model fails to differentiate the non-sensitive operations on mobile devices (e.g., clicking the screen to open an APP or just for web-browsing) from sensitive operation (e.g., inputting the password). Instead, IKI model allows the attacker to obtain both of unencrypted meta data traffic as well as the CSI data to launch a more fine-grained attack.

2.3 Channel State Information

The basic goal of WindTalker is measuring the impact of hand and fingers' movement on WiFi signals and leveraging correlation of CSI and the unique hand motion to recognize PIN. In the below, we briefly introduce the CSI related backgrounds.

WiFi Standards like IEEE 802.11n/ac all support Multiple-Input Multiple-Output (MIMO) and Orthogonal Frequency Division Multiplexing (OFDM), which are expected to significantly improve the channel capacity of the wireless system. In a system with transmitter antenna number N_{TX}, receiver antenna number N_{RX} and OFDM subcarriers number N_s, system will use $N_{TX} \times N_{RX} \times N_s$ subcarriers to transmit signal at the same time.

CSI measures Channel Frequency Response (CFR) in different subcarriers f. CFR $H(f,t)$ represents the state of wireless channel in a signal transmission process. Let $X(f,t)$ and $Y(f,t)$ represent the transmitted and received signal with different subcarrier frequency. $H(f,t)$ can be calculated in receiver using a known transmitted signal via

$$Y(f,t) = H(f,t) \times X(f,t). \tag{1}$$

Since the received signal reflects the constructive and destructive interference of several multi-path signals scattered from the wall and surrounding objects, the movements of the fingers while password input can generate a unique pattern in the time-series of CSI values, which can be used for keystrokes recognition.

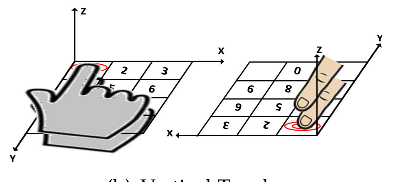
Many commercial devices such as Atheros 9,390 [13], Atheros 9,580 [14] and Intel 5,300 [15] network interface cards (NICs) with special drivers provide open access to CSI value. In this study, we adopt Intel 5,300 NICs, which follows IEEE 802.11n standard [16] and can work in 2.4 GHz or 5 GHz. By selecting a group of 30 OFDM subcarriers of totally 56 subcarriers, Intel 5,300 NICs collect CSI value for each TX-RX antenna pair.

3 Motivation

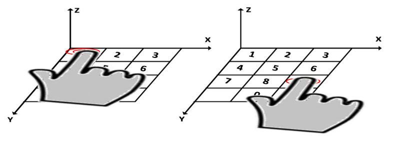
In this section, we illustrate the rationale behind CSI based keystroke inference on smart phones using real-world experiments. Fig. 2a shows the sketch of typical touching screen during the PIN entry for mobile payment (e.g., Alipay or Wechat pay). We particularly focus on the vertical touch and the oblique touch, which are two most common touching gestures [17], [18], [19]. As shown in Figs. 2b and 2c, oblique touch is the most common typing gesture which happens when people press different keys, and vertical touch usually happens when the human continuously presses the same key. Fig. 2d shows the original CSI waveforms from the 21st subcarrier to 30th subcarrier during once

Enter Payment Password
¥100.00
Pay with CCB(7479)

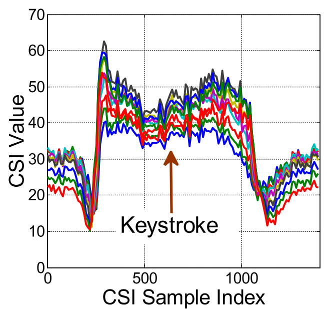
(a) Finger typing



(b) Vertical Touch



(c) Oblique Touch



(d) CSI Waveforms from the 21st to the 30th Subcarrier

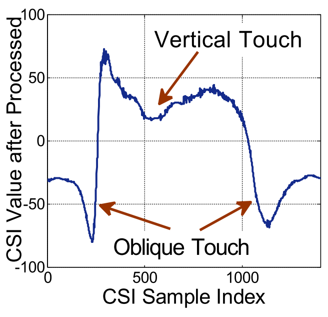


Fig. 2. Finger's influence on CSI.

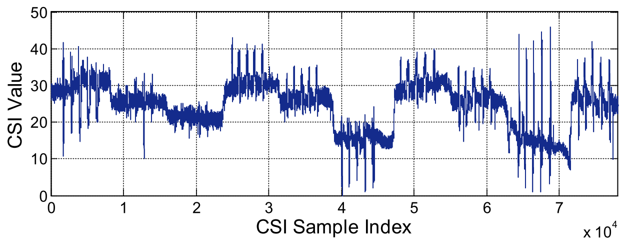
keystroke. It can be seen that CSI waveforms collected by Intel 5,300 NIC are affected by the keystroke and the fluctuations of these ten waveforms are similar. Fig. 2e shows the processed CSI value for the keystroke (the process method is mentioned in Section 4). We find that the pattern of processed CSI value is very closely related to oblique and vertical touch, and this pattern could be used to characterize the corresponding keystroke.

We further investigate how these two common typing gestures influence CSI. Generally speaking, since CSI reflects the constructive and destructive interference of several multi-path signals, the change of multi-path propagation during the PIN entry can generate a unique pattern in the time-series of CSI values, which can be used for keystrokes inference. From our experiments, we found that two main factors contributing to CSI changes are hand coverage and the finger click.

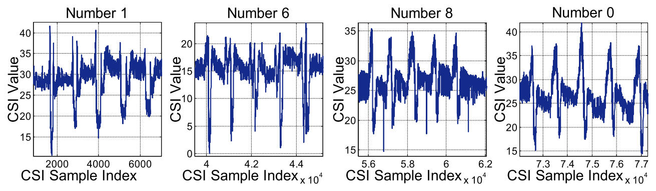
Hand coverage and finger position on a smart phone touchscreen are one of the major factors that cause the fluctuation of CSI waveform. Since time series of CSI waveform reflects the interference of several multi-path signals, different finger position and coverage will inevitably introduce the interference to the WiFi signals and thus lead to the changes of the CSI. We further demonstrate the relationship between CSI variation and finger position/coverage via a series of experiments. Fig. 3a shows a CSI waveform when continuously pressing different number from 1 to 9, followed by 0, each for 5 times. It can be seen that the different coverage leads to the different fluctuation range of the CSI value, which can be exploited for key inference.

Finger click is another important factor that contributes to the fluctuation of CSI. Compared with CSI change caused by the hand coverage, the experiment shows that finger click

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.



(a) Continuously Clicking in the Same Key



(b) Continuously Clicking in Different Keys

Fig. 3. CSI change when typing.

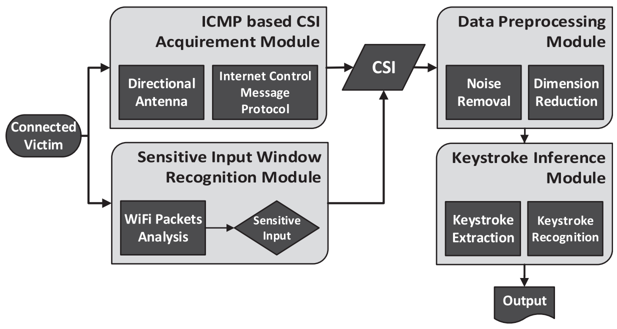

ICMP based CSI Acquirement Module
Directional Antenna
Internet Control Message Protocol
CSI
Data Preprocessing Module
Noise Removal
Dimension Reduction
Connected Victim
Sensitive Input Window Recognition Module
WiFi Packets Analysis
Sensitive Input
Keystroke Inference Module
Keystroke Extraction
Keystroke Recognition
Output

Fig. 4. WindTalker framework.

has a more direct influence on CSI by introducing a sharp convex in Fig. 3b, which corresponds to the quick click's influence on multi-path propagation. This feature can be used to distinguish the oblique touches in the case that the human continuously presses the same key or the adjacent keys, which produce similar CSI values.

4 THE DESIGN OF WINDTALKER

4.1 System Overview

The basic strategy of WindTalker is hitting two birds with one stone. On one hand, it analyzes the WiFi traffic to identify the sensitive attack windows (e.g., PIN number) on smartphones. On the other hand, as long as an attack window is identified, WindTalker starts to launch the CSI based keystroke recognition. As shown in Fig. 4, WindTalker is consisted of the following modules: *Sensitive Input Window Recognition Module*, which is responsible for distinguishing the sensitive input time windows, *ICMP Based CSI Acquirement Module*, which collects the user's CSI data during his access to WiFi hotspot, *Data Preprocessing Module*, which preprocesses the CSI data to remove the noises and reduce the dimension, *Keystroke Extraction Module*, which enables WindTalker to automatically determine the start and the end point of keystroke waveform, and *Keystroke Inference Module*, which compares the different keystroke waveforms and determines the corresponding keystroke.

TABLE 1
Payment Applications and Their Sensitive IP Addresses

Payment application	IP address
Alipay	110.76.15.1xx & 110.75.236.xx
Wechat Pay	182.254.78.1xx
JD Pay	111.13.142.x

4.2 Sensitive Input Window Recognition Module

To distinguish the time window of the sensitive input from that of the insensitive input, WindTalker captures all packets of the victim with Wireshark and records the timestamp of each CSI data. Currently, most of the important applications are secured via HTTPS, which provides end-to-end encryption and prevents the eavesdropper from obtaining the sensitive data such as the password. Our insight is that though HTTPS provides end-to-end encryption, it cannot protect the meta data of the traffic such as the IP address of the destination sever, which can be used to recognize sensitive input window.

In particular, WindTalker builds a Sensitive IP Pool for the interested applications or services. Take the AliPay as an example. During the payment process, the data packets will be directed to a limited number of IP addresses, which can be obtained via a series of trials. In the experimental evaluation, it is shown that, for Alipay users, the traffics of the users under the same network will be directed to the same server IP, which will last for a period (e.g., several days for one round of experiment). Therefore, it is feasible to try to access the interested applications or services at regular intervals and append the obtained IP addresses to the Sensitive IP Pool. This constantly updated pool allows WindTalker to figure out the sensitive input time window.

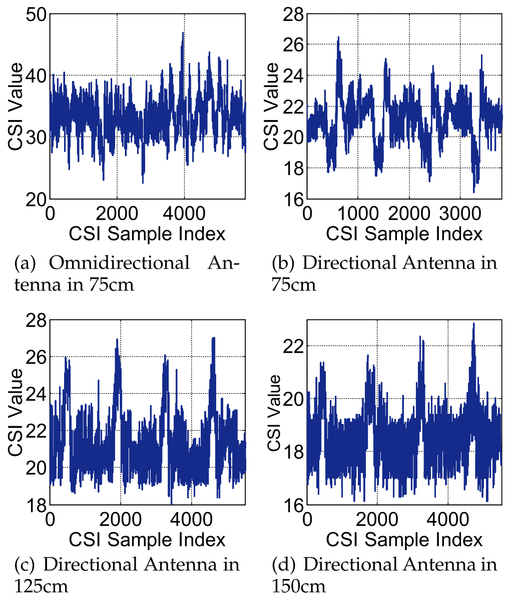
To evaluate this method, we conduct experiments on three popular mobile payment applications (i.e., Alipay, Wechat Pay and JD Pay) and capture the network traffic using WireShark. We completed mobile payment for each application ten times. As shown in Table 1, for a certain application, when the password input process starts, some packets with a specific IP address will happen. This result demonstrates the effectiveness of the sensitive IP pool based method. Therefore, during the attack process, as long as the traffic to the IP addresses contained in the Sensitive IP Pool is observed, WindTalker will extract these traffic, and then record the corresponding start time and the end time, which serve as the start and the end of the Sensitive Input Window. Then, it starts to analyze the CSI data in this period to launch the password inference attack via WiFi signals.

4.3 ICMP Based CSI Acquirement Module

4.3.1 Collecting CSI Data by Enforcing ICMP Reply

Different from the previous works which rely on two devices including both of the sender and the receiver to collect CSI data, we apply an approach that leverages Internet Control Message Protocol (ICMP) in hotspot to collect CSI data during the user accesses to the pre-installed access point. In particular, WindTalker periodically sends a ICMP Echo Request to the victim smartphone, which will reply an Echo Reply for each request. To acquire enough CSI information of the victim, WindTalker needs to send ICMP Echo Request at a high

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.



(a) Omnidirectional Antenna in 75cm (b) Directional Antenna in 75cm (c) Directional Antenna in 125cm (d) Directional Antenna in 150cm

Fig. 5. Antenna performance in public place.

frequency, which enforces the victim to replay at the same frequency. In practice, WindTalker can work well for several smartphones such as Xiaomi, Redmi and Samsung at the rate of 800 packets per second. It is important to point out that this approach does not require any permission of the target smartphone and is difficult to be detected by the victim.

ICMP based CSI collection approach introduces a limited number of extra traffics. For a 98 bytes ICMP packet, when we are sending 800 ICMP packets per second to the victim, it needs only 78.4 KB/s for the attack where 802.11n can theoretically support the transmission speed up to 140 Mbits per second. It is clear that the proposed attack makes little interference to the WiFi experience of the victim.

4.3.2 Reducing Noise via Directional Antenna

CSI will be influenced by both finger movement and people's body movement. One of the major challenges of obtaining the exact CSI data in public space is how to minimize the interference caused by the nearby human beings. We present a noise reduction approach by adopting the directional antenna. Different from omnidirectional antennas that have a uniform gain in each direction, directional antennas have a different antenna gain in each direction. As a result the signal level at a receiver can be increased or decreased simply by rotating the orientation of the directional antenna. WindTalker employs directional antenna to focus the energy toward the target of interest, which is expected to minimize the effects of the nearby human body movement.

WindTalker employs a TDJ-2400BKC antenna working in 2.4 GHz to collect CSI data of the targeted victim, whose Horizontal Plane -3dB Power Beamwidth and Vertical Plane -3dB Power Beamwidth are 30° and 25° respectively.

Fig. 5 shows the comparison of CSI collection with directional antenna and without directional antenna in public place. Figs. 5b, 5c, and 5d show CSI amplitude in the case that a victim is located at 75, 125, 150 cm accordingly away from directional antenna while one human moving nearby. Unique pattern caused by finger click in number 1 can be easily caught from the original CSI waveform without any preprocessing. However, these patterns are submerged in human body's influence on CSI waveform obtained by omni-directional antenna even when the victim and attacker is close as 75 cm, which is shown in Fig. 5a. In the following sections, we only discuss the CSI acquirement with directional antenna.

4.4 Data Preprocessing Module

Before launching keystroke inference module, WindTalker needs to preprocess the CSI data to remove the noises introduced by commodity WiFi NICs due to the frequent changes in internal CSI reference levels, transmission power levels, and transmission rates. To achieve this, WindTalker first turns to wavelet denoising to remove noises from the obtained signals. Then, WindTalker leverages the Principal Component Analysis to reduce the dimensionality of the feature vectors to enable better analysis of the data.

4.4.1 Wavelet Denoising

We observe that the variation of CSI waveforms caused by finger motion normally appears at the low end of the spectrogram while the frequency of the noise occupies at the high end of the spectrogram. We do not adopt the low-pass filter since high-frequency signal includes some finger motion characters. In this paper, wavelet denoising method is used to remove noise from the raw signal. Different from the traditional frequency analysis such as Fourier Transform, Discrete Wavelet Transform (DWT) is the time-frequency analysis which has a good resolution at both of the time and frequency domains. WindTalker can thus leverage DWT to analyze the finger movement in varied frequency domains. Wavelet denoising includes three main steps as follows:

Discrete Wavelet Transform. Generally speaking, a discrete signal $x[n]$ can be expressed in terms of the wavelet function by the following equation:

$$\begin{aligned} x[n] = &\frac{1}{\sqrt{L}}\sum_k W_\phi[j_0,k]\phi_{j_0,k}[n] \\ &+ \frac{1}{\sqrt{L}}\sum_{j=j_0}^{\infty}\sum_k W_\psi[j,k]\psi_{j,k}[n]. \end{aligned} \tag{2}$$

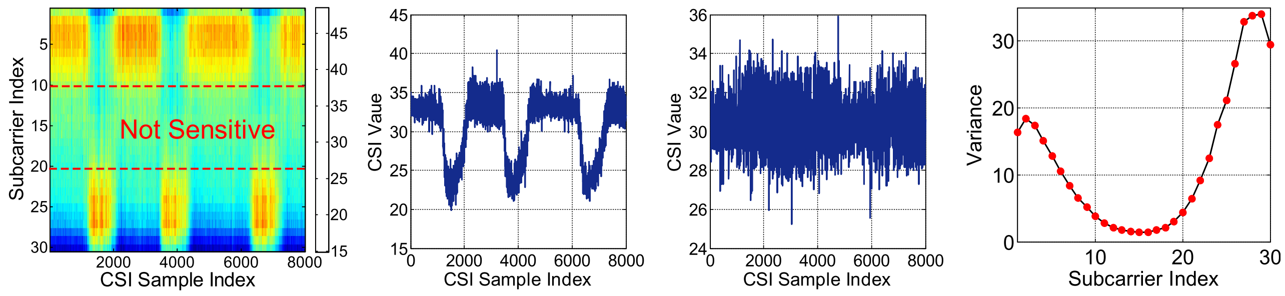
Where $x[n]$ represents the original discrete signal and L represents the length of $x[n]$. $\phi_{j_0,k}[n]$ and $\psi_{j,k}[n]$ refer to wavelet basis. $W_\phi[j_0,k]$ and $W_\psi[j,k]$ refer to the wavelet coefficients. The functions $\phi_{j_0,k}[n]$ refer to scaling functions and the corresponding coefficients $W_\phi[j_0,k]$ refer to the approximation coefficients. Similarly, functions $\psi_{j,k}[n]$ refer to wavelet functions and coefficients $W_\psi[j,k]$ refer to detail coefficients. To obtain the wavelet coefficients, the wavelet basis $\phi_{j_0,k}[n]$ and $\psi_{j,k}[n]$ are chosen to be orthogonal to each other.

During the decomposition process, the origin signal is first divided into the approximation coefficients and detail coefficients. Then the approximation coefficients are iteratively divided into the approximation and detail coefficients of next level. The approximation and the detail coefficients in jth level can be calculated as follows:

$$W_\phi[j_0,k] = \left\langle x[n], \phi_{j_0+1,k}[n]\right\rangle = \frac{1}{\sqrt{L}}\sum_n x[n]\phi_{j_0+1,k}[n] \tag{3}$$

$$W_\psi[j,k] = \left\langle x[n], \psi_{j+1,k}[n]\right\rangle = \frac{1}{\sqrt{L}}\sum_n x[n]\psi_{j+1,k}[n]. \tag{4}$$

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.



(a) CSI Waveforms while Typing Keystroke (b) 1st Subcarrier: Sensitive (c) 16th Subcarrier: Not Sensitive (d) Variance of each Subcarrier

Fig. 6. Subcarrier selection algorithm.

Threshold Selection. After recursive DWT decomposition, the raw signal is broken into detail coefficients (high-frequency) and approximation coefficients (low-frequency) at different frequency levels. Then, the threshold is applied to the detail coefficients to remove their noisy part. The threshold selection is important because a small threshold will remain the noisy components while a large threshold will lose the major information of signals. In this paper, the minimax threshold is chosen based on it's dynamic, effectiveness and simplicity [20].

Wavelet Reconstruction. After the above two steps, we reconstruct the signal to achieve noise removal by combining the coefficients of the last approximation level with all details which have applied threshold. Wavelet selection plays a key role in wavelet decomposition and reconstruction. There are many wavelet bases such as Daubechies and Haar wavelet [20]. In practice, we choose Daubechies D4 wavelet and perform 5-level DWT decomposition in wavelet denoising in our study.

4.4.2 Dimension Reduction

Dimension reduction is essential for keystroke inference via CSI information. For a CSI recording system using Intel 5,300 NICs with N_{TX} transmitter antennas and N_{RX} receiver antennas, it can collect $N_{TX} \times N_{RX} \times 30$ CSI waveforms. It is important to reduce the dimensionality of the CSI information obtained from 30 subcarriers in each TX-RX pair and recognize those subcarriers which show the strongest correlation with the hand and fingers movements. WindTalker adopts PCA to choose the most representative or principal components from all CSI time series. PCA is also expected to remove the uncorrelated noisy components. The procedure of dimension reduction of CSI time series based on PCA includes the following steps.

Subcarrier Selection. We observe that the CSI waveforms from different subcarriers have different sensitivities to CSI variation caused by keystrokes due to frequency diversity. As shown in Figs. 6a, 6b and 6c, some subcarriers amplitudes vary a lot with keystrokes, but others are obtuse. As shown in Fig. 6d, we calculate the variance of each subcarrier. We find that subcarriers 10 to 19 have lower variances, which means they have lower sensitivities with keystrokes. So we discard the lowest ten subcarriers before PCA.

Sample Centralization. We use a matrix H to present original CSI waveform data. For example, in a system with one pair of TX-RX antenna, we will get 30 CSI waveforms from 30 subcarriers. Thus, with sampling rate S and time T, H has dimension of $M \times 30$, where $M = S \times T$. Then we calculate the mean value of each column in H and subtract the corresponding mean values in every column. After the centralization step, we get a processed matrix H_p.

Calculating Covariance Matrix. Calculating the correlation matrix of H_p as ${H_p}^T \times H_p$.

Handling Covariance. Calculating the Eigenvalues and Eigenvectors of Covariance. The Eigenvectors are normalized to unit vectors.

Choosing Main Eigenvalues. Sorting the Eigenvalues from large to small and choosing the maximum k number of Eigenvalues. Then the corresponding k Eigenvectors are used as the column vectors to form a Eigenvector matrix. We will get a Eigenvector matrix whose dimension is $30 \times k$.

Data Reconstruction. Projecting H_p onto the selected k Eigenvector matrix. The reconstruction CSI data stream H_r has the dimension of $M \times k$.

$$H_r(M \times k) = H_p(M \times 30) \times Eigen\ Vectors(30 \times k). \tag{5}$$

With PCA, we can identify the most representative components influenced by the victim's hand and fingers' movement and remove the noisy components at the same time. In our experiment, it is observed the first $k = 4$ components almost show the most significant changes in CSI waveforms and the rest components are noises. In our experiment part, we observed that the first PCA component reserves most changes in CSI while the ambient noise is weak. Thus we only take one PCA component from the first 4 components in the password inference module.

4.5 Keystroke Inference Module

4.5.1 Keystroke Extraction

By processing the wavelet denoising and dimension reduction, it is observed that the CSI data shows a strong correlation with the keystrokes, as shown in Fig. 7a. In the experiment, the sharp rise and fall of the CSI waveform signals are observed in coincidence with the start and end of finger touch. How to determine the start and the end point of CSI time series during a keystroke is essential for keystroke recognition. However, the existing burst detection schemes such as Mann-Kendall test, moving average method and cumulative anomalies [21] do not work well in our situation since the CSI waveform has many change-points during the password input period. Therefore, we propose a novel detection algorithm to automatically detect the start and end

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

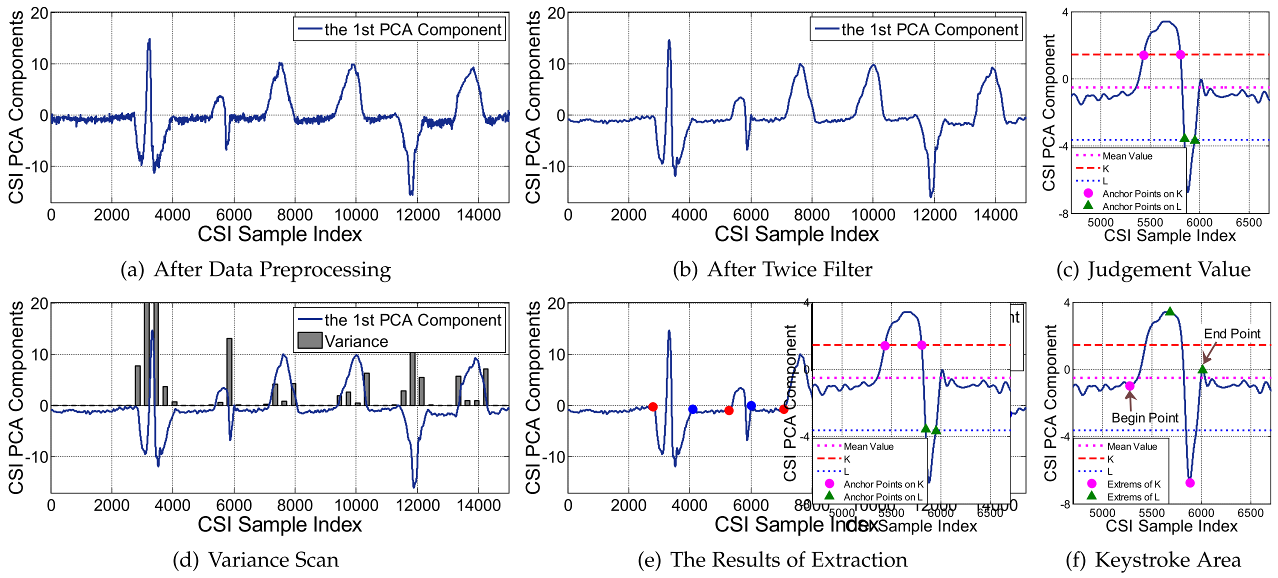


Fig. 7. Keystroke extraction.

point. The proposed algorithm includes the following three steps.

Waveform Profile Building. As shown in Fig. 7a, it is observed that there is a sharp rise and fall which correspond to the finger motions. However, there is a strong noise which prevents us from extracting interested CSI waveform related to the keystrokes. This motives us to perform another round of noise filtering. In the experiment, we still adopt wavelet denoising to make the waveform smooth. After being filtered, the CSI data during the keystroke period are highlighted while the waveform during non-keystroke period becomes smooth, which are shown in Fig. 7b.

CSI Time Series Segmentation and Feature Segment Selection. To extract the CSI waveforms for individual keystrokes, we slice the CSI time series into multiple segments, which be grouped together according to the temporal proximity, and then choose the center of segment as the feature waveform for a specific keystroke. Without loss of the generality, it is assumed that each segment contains W samples. Given the sampling frequency S, and the time duration τ, W can be represented by $S \times \tau$. For the waveform with time duration of T, the number of segments N can be calculated as below

$$N = \left\lceil \frac{T \times S}{W} \right\rceil. \tag{6}$$

It is observed that the CSI segments during the keystroke period show a much larger variance than those happening out of the period, which is shown in Fig. 7d. Motivated by this, we are only interested in the segments with the variance which is larger than a predetermined threshold while removing the segments with the variance under this threshold. The selected segments are grouped into various groups according to the temporal proximity (e.g., five adjacent segments grouped into one group in the practice). Each group represents the CSI waveform of an individual keystroke and the center point of this group is selected as the feature segment of this keystroke. The process of time series segmentation and feature segment selection is shown in Fig. 7d.

Keystroke Waveforms Extraction. To extract keystroke waveforms, the key issue is how to determine the start and the end point of CSI time series, which could cover as much keystroke waveform as possible while minimizing the coverage of the non-keystroke portion.

We calculate the average value of the segment samples J, and then choose two key metrics K and L. K is the average value of J and samples' maximum value, while L is the average value of J and samples' minimum value. The intersection of K, L and the CSI waveform serves as the anchor points. On line K, starting from the leftmost anchor point, it performs a local search and chooses the nearest local extremum which is below K as the first start point. Similarly, beyond the rightmost anchor point, it can choose the nearest local extremum which is below K as the first end point. Also, we can perform local searches from two anchor points on line L in order to choose two local extremum beyond L as the second start point and the second end point. Finally, we compare these points respectively. As shown in Figs. 7c, 7f, and 7e, with the lower start point and the higher end point, keystroke waveform can be extracted.

Thus, we can divide a CSI waveform into several keystroke waveform. The ith keystroke waveform K_i from the kth principal component $H_r(:,k)$ of CSI waveforms as follows.

$$K_i = H_r(s_i : e_i, k), \tag{7}$$

where s_i and e_i are the start and the end time of ith keystroke. After keystroke extraction, we use these keystroke waveform to conduct recognition process.

4.5.2 Keystroke Time Domain Feature Extraction

One of the major challenges for differentiating keystrokes is how to choose the appropriate features that can uniquely represent the keystrokes. As shown in Fig. 8, it is observed that different keystrokes will lead to different waveforms, which motivates us to choose waveform shape as a feature for keystroke classification. However, directly using the keystroke waveforms as the classification features leads to a high computational cost in the classification process since

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

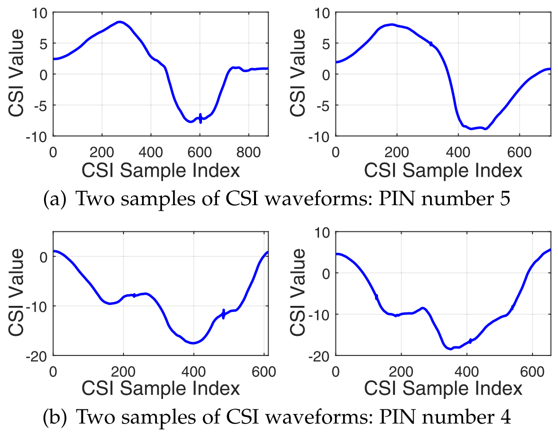


Fig. 8. CSI difference between two number.

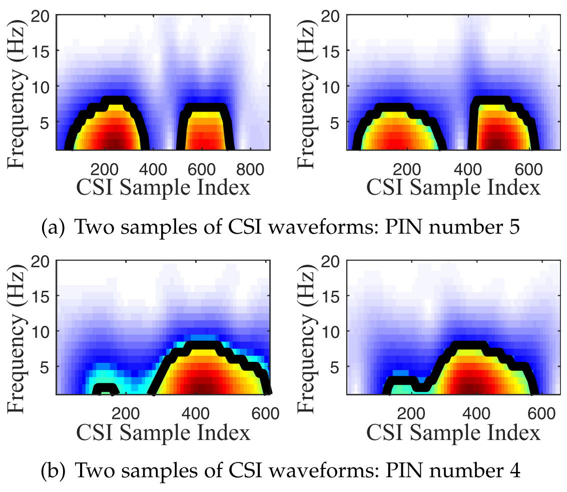


Fig. 9. CSI difference between two number: Frequency domain.

waveforms contain many data points for each keystroke. Therefore, we leverage Discrete Wavelet Transform to compress the length of CSI waveform by extracting the approximate sequence. In the below, we will introduce the details.

Wavelet Compression. As mentioned in Section 4.4.1, a discrete keystroke waveform $K_i[n]$ can be expressed by the following equation:

$$K_i[n] = \frac{1}{\sqrt{L}}\sum_k W_\phi[j_0,k]\phi_{j_0,k}[n] + \frac{1}{\sqrt{L}}\sum_{j=j_0}^{\infty}\sum_k W_\psi[j,k]\psi_{j,k}[n], \tag{8}$$

where L represents the length of $K_i[n]$, $W_\phi[j_0,k]$ and $W_\psi[j,k]$ refer to the approximation and detail coefficients respectively. In the first DWT decomposition step, the length of approximation coefficients is half of L. For the jth decomposition step, the length is half of the previous decomposition step. We use the approximation coefficients to compress the original keystroke waveforms to reduce computational cost. In order to achieve the trade-off between the sequence length reduction and preserving the waveform information, we choose Daubechies D4 wavelet and perform 3-level DWT decomposition in the classification model. Therefore, for ith keystroke, the third level approximation coefficient F_i of K_i is chosen as the feature of the keystroke. After compression, the length of feature F_i is about $1/8$ of $K_i[n]$.

4.5.3 Keystroke Frequency Domain Feature Extraction

Besides CSI waveform shape, the CSI frequency feature can also be used to differentiate keystrokes. The CSI spectrograms in frequency domain is a stable property of CSI streams and is highly correlated to keystrokes. Fig. 9 illustrates the CSI spectrograms corresponding to the CSI waveforms shown in Fig. 8. It is observed that different keystrokes have significantly different CSI spectrograms. Therefore, it's feasible to use CSI spectrogram information as the feature to recognize keystroke.

In this paper, WindTalker first performs Short Time Fourier Transform (STFT) to obtain the two-dimensional frequency spectrograms of CSI. Then, WindTalker calculates the contours of the spectrograms to extract features. To extract the contour, WindTalker first resizes the CSI spectrograms with frequency from 0 to 30 Hz into a m-by-n matrix $M_{CSI}(i,h)$ and normalize the $M_{CSI}(i,h)$ to a range between 0 and 1. Note that, in $M_{CSI}(i,h)$, each column represents the normalized frequency shifts during the ith time slide. Then, WindTalker chooses a pre-defined *threshold* and get the contour $C_{CSI}(i)$, where $i = 1 \dots n$. $C_{CSI}(i)$ is the maximum value j which satisfies that $M_{CSI}(i,j) \geq threshold$. As shown in Fig. 9, the contours are marked by the black lines. It is observed that, between the same keystrokes, the contours of CSI spectrograms have the similar variation trends. Thus we can regard the contours as the frequency domain features of the classification and calculate the similarity between the contours for keystroke recognition.

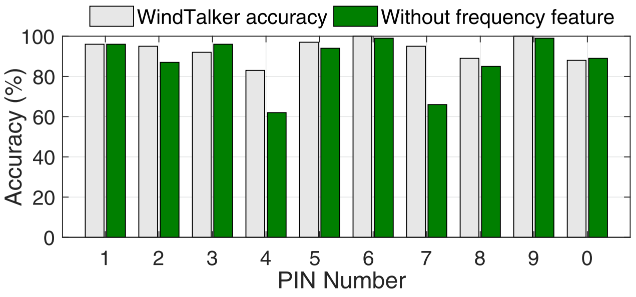
4.5.4 Keystroke Recognition

WindTalker builds a classifier to recognize the keystrokes based on both the time domain feature and the frequency domain feature. To compare the features of different keystrokes, WindTalker adopts the Dynamic Time Warping (DTW) to measure the similarity between two keystrokes.

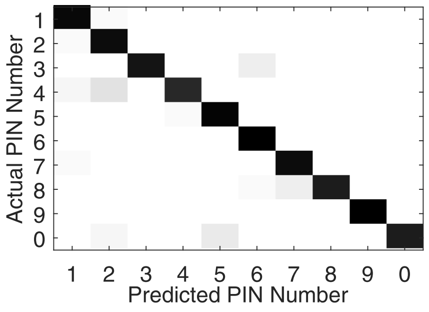
Dynamic Time Warping. DTW utilizes dynamic programming to calculate the distance between two sequences with different lengths. With DTW, the sequences (e.g., time series and spectrogram contours) are warped non-linearly in the time dimension to measure their similarity. The input of DTW algorithm is two sequences and the output is the distance between them. A low distance indicates that these two sequences are highly similar.

By adopting DTW, the classifier gives each keystroke a set of scores, which allows the keystrokes to be differentiated based on the user's training dataset (keystrokes on different numbers). For a certain keystroke K_i, classifier first calculates the DTW distances between the features of K_i and all of the keystroke number's features in dataset in time and frequency domain respectively. Thus, for K_i, we will get two scores arrays $S_T = \{s_{t1}, s_{t2}, \dots, s_{t10}\}$, $S_F = \{s_{f1}, s_{f2}, \dots, s_{f10}\}$, where S_T, S_F represent the scores in time and frequency domain respectively, and s_{tn} refers to the shortest distance between the input keystroke and the certain key number n in time domain. s_{fn} is similar but in frequency domain. Finally the classifier calculates the score $S = \{s_1, s_2, \dots, s_{10}\}$, where $s_n = s_{tn} \times s_{fn}$. The lower the score s_n is, the higher possibility the

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.



(a) Classification Accuracy per key



(b) Color Map of Confusion Matrix (re-drawn)

Fig. 10. Classification accuracy.

certain number n is actual input number. The classifier chooses the key number which has the minimum score (the value n which satisfies s_n is the lowest one) as the predicted key number. Note that the classifier saves all scores of the certain keystroke K_i in order to generate password candidates in Section 5.3.

5 Keystroke Inference Evaluations

5.1 System Setup

WindTalker is built with the off-the-shelf hardware, which is actually a commercial laptop computer equipped with Intel 5300 NIC with one external directional antenna. WindTalker also serves as the WiFi hotspot to attract the users to access to the WiFi. The laptop runs Ubuntu 14.04 LTS with a modified Intel driver to collect CSI data. To collect the CSI data related to the user's touch screen clicks, WindTalker uses ICMP echo and reply to achieve the sampling rate of 800 packets/s. In this evaluation, the distance between the mobile user and the AP is 75 cm and the AP is placed on the left side of mobile phone.

In the experiments, we recruit 20 volunteers to join our evaluation, including 17 males and 3 females. All of the volunteers are right-handed and they perform the touch-screen operations by following their own fashions. During the experiment, the volunteers should participate in the data training phase and keystroke recognition phase by inputting the numbers according to the system hints. In the data training phase, WindTalker records each input and its corresponding CSI data. In the test phase, WindTalker infers the input data based on the observed CSI time series. The training data and testing data collection should be finished within 30 minutes since CSI may change with the change of environment.

We start the evaluation by testing the classification accuracy and the 6-digit password inference accuracy. Then we perform a more specific case study by inferring the password of mobile payment for Alipay in Section 6. Afterwards we investigate various metrics that may influence the inference accuracy of WindTalker including the distance, the direction and the human movement in Section 7. In the current stage evaluation, we only perform user-specific training and will discuss its limitation in Section 9.

TABLE 2
Recovery Rate and Training Loop Times

Loop Times	One	Three	Five	Nine
Recovery Rate	79.5%	86.2%	88.5%	93.5%

5.2 Classification Accuracy

In Section 3, we have shown that different keystrokes may be correlated with different CSI waveforms. In this section, we aim to explore whether the differences of keystroke waveforms are large enough to be used for recognizing different PIN numbers inputs in the real-world setting. We collected training and testing data from 20 volunteers. Each volunteer first generates 50 loop samples, where a loop is defined as the CSI waveform of keystroke number from 0 to 9 by pressing the corresponding digit. After that, we evaluate the classification accuracy of WindTalker through the collected CSI data.

The classification accuracy is evaluated in terms of 10-fold cross validation accuracy. However, in real world scenario, it is not reasonable to collect 50 training samples for one specific PIN number. Therefore, we first divide these 50 loops data into 5 groups evenly. Then, for every 10 loops CSI data, we pick up one loop in turn for the testing data and choose the other 9 loops as the training data. WindTalker adopts the classifier introduced in Section 4.5 to recognize the keystroke. We perform the evaluation on Xiaomi, Redmi and Samsung Note3 smartphone. All of them run with Android 5.0.2. Fig. 10a shows the average classification accuracy of all 20 volunteers in 10 PIN number. It is observed that WindTalker achieves the average accuracy classification of 93.5 percent using combined CSI features. However, if WindTalker only utilizes time-domain feature as [1], the accuracy will drop to 87.3 percent.

Fig. 10b describes the color map of confusion matrix of keystroke inference. For a specific typed number, it gives the corresponding inference results. The darker the area is, the higher the possibility of keystroke inference result is. Intuitively, it is easier for an input number that is confused with the neighboring numbers during the keystroke inference process. We further analyze the impact of the number of training data on recovery rate in WindTalker. Table 2 shows the keystroke inference accuracy increases with the training loop increases. Even if there is only one training sample for one keystroke, WindTalker can still achieve whole recovery rate of 79.5 percent.

5.3 Password Inference

In a practical scenario setting, it may not be easy for WindTalker to get 9 training samples for each PIN number. So in the remaining section, we only use 3 samples per PIN number for training. To illustrate the performance of WindTalker for password Inference, in this part, we ask volunteers to

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

TABLE 3
Recovery Rate and the Number of Candidates

Number of candidates	One	Two	Three
Recovery rate	86.2%	93.4%	96.2%

press 10 randomly generated 6-digit passwords on Xiaomi smartphone and use their corresponding 3 loops as training dataset.

We test totally 500 set of passwords, which include 3,000 keys. As shown in Table 3, with 3 loops as training data, WindTalker can achieve an average 1-digit recovery rate of 86.2 percent. For a 6-digit password in AliPay, the attacker can try several times to recover the password at an increased successful rate. Thus, we introduce a new metric, recovery rate with Top N candidates, which indicates the rate of successfully recovering the password for trying N times and represents a more reasonable metric to describe the capability of the attacker in the practical setting. As shown in Table 3, if we evaluate the 1-digit recovery rate under top 2 and top 3 candidates, the recovery rate can be significantly improved.

We further study how many candidates can help us to succeed in predicting the right 6-digit payment password in WindTalker. In particular, we will investigate the inference accuracy under top N candidates. In the experiment, each 6-digit password will be correlated with six keystrokes $K = \{K_1, K_2, \ldots, K_6\}$. For each keystroke K_i, WindTalker calculates its corresponding score $S_i = \{s_{i,1}, s_{i,2}, \ldots, s_{i,10}\}$. Then, for a given password candidate PIN number $P = \{p_1, p_2, \ldots, p_6\}$, where $p_i \in [0, 9]$, WindTalker calculates the likelihood L between K and P. L is defined by $L = \prod_{i=1}^{6} s_{i,p_i}$. Given a 6-digit password K, for each keystroke K_i, we can obtain 5 candidates with lowest s_i and then generate $5^6 = 15626$ candidate passwords. Then WindTalker sorts these passwords by their likelihoods in ascending order. A successful password inference is defined as that the real password is included in top N candidates. In Fig. 11a, we give the password inference accuracy under top N candidates, where N ranges from 1 to 20. The result is encouraging. It is shown that, given top 1 candidate, the inference accuracy is 41.2 percent. And the inference rate can be significantly improved if given top 5 candidates or top 10 candidates, which correspond to 69.6 and 77.4 percent, respectively. It is also shown in Fig. 11b that, if given enough top N candidates (e.g., set N as 60), the inference accuracy can reach above 85 percent.

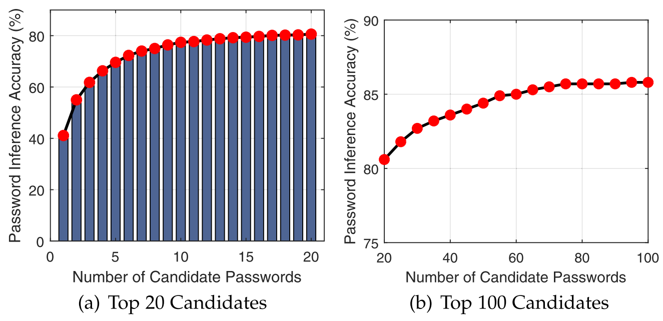


Fig. 11. 6-digit password inference accuracy.

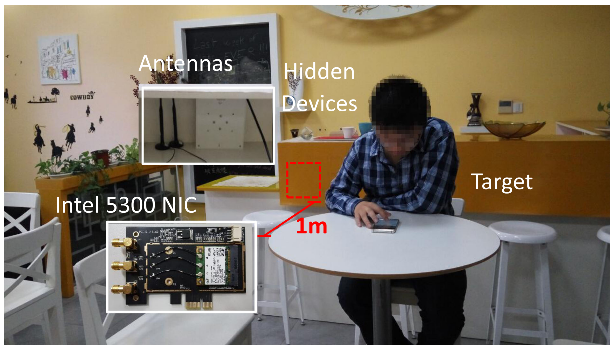


Fig. 12. Real case scenario.

6 Real-World Experiment: Mobile Payment Password Inference Towards Alipay

6.1 System Setup

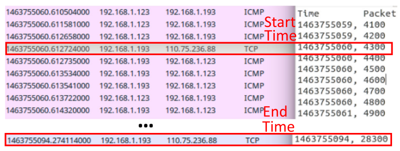
To demonstrate the practicality of the WindTalker, we perform an experimental evaluation of password inference on Alipay, a popular mobile payment platform on Both of Android and iOS system. Alipay is the largest mobile payment company in the world and has 450 million monthly active user including 270 million mobile payment users [22]. As shown in Fig. 12, we deploy a WindTalker system at a cafe-like environment and release an authentication-free WiFi. The AP (including Intel 5,300 NIC and the antennas) is set up behind the counter, which makes it less likely to be detected visually. The victim is 1 meter away from our deployed WiFi devices. When we collect the data, one volunteer walks pass by the victim but none of volunteers walks between the victim and the AP.

To simulate the real-world attack scenarios, the recruited volunteers are required to access to this free WiFi access point and perform the following three phases: 1) Online Training Phase: the volunteers are required to input some randomly generated numbers by following a similar way as Text Captcha. This phase is designed to collect the user's input number and the corresponding CSI data to finish the data training. 2) Normal Use Phase: the volunteers perform the online browsing or use the applications as a normal user. 3) Mobile Payment Phase: when the users use the online shopping applications, it will be ended with the mobile payment. All of the online shopping and mobile payments are secured with HTTPS protocol. According to Alipay mobile payment policy, the mobile users must input the password to finish an mobile payment transactions. The goal of the attacker is to recover the mobile payment password of the volunteers.

6.2 Operations of WindTalker

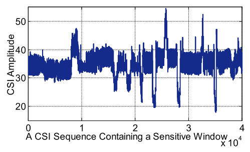
After the volunteers connect to the authentication-free WiFi hotspot, WindTalker triggers ICMP based CSI Acquirement Module to collect the CSI data at the sampling rate of 800 packets/s. WindTalker records the timestamp per one hundred CSI data. Simultaneously, WindTalker utilizes Wireshark to capture and record WiFi traffic packets and their corresponding timestamps. During the real-world experiment, WindTalker collects WiFi traffic data and CSI data in the online

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

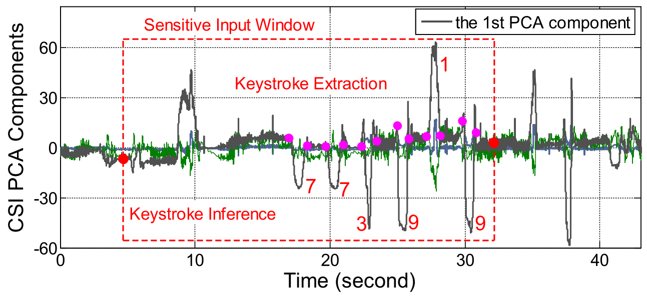


```
1463755060.610504000  192.168.1.123  192.168.1.193  ICMP        Time             Packet
1463755060.611581000  192.168.1.123  192.168.1.193  ICMP Start  1463755059, 4100
1463755060.612658000  192.168.1.123  192.168.1.193  ICMP Time   1463755059, 4200
1463755060.612724000  192.168.1.193  110.75.236.88  TCP         1463755060, 4300
1463755060.612735000  192.168.1.193  192.168.1.123  ICMP        1463755060, 4400
1463755060.613534000  192.168.1.193  192.168.1.123  ICMP        1463755060, 4500
1463755060.613541000  192.168.1.193  192.168.1.123  ICMP        1463755060, 4600
1463755060.613722000  192.168.1.123  192.168.1.193  ICMP        1463755060, 4700
1463755060.614320000  192.168.1.193  192.168.1.123  ICMP End    1463755060, 4800
                                                         Time   1463755061, 4900
                          ...
1463755094.274114000  192.168.1.193  110.75.236.88  TCP         1463755094, 28300
```

(a) Sensitive Input Window Recognition Module



(b) Original CSI: the 30th Subcarrier



(c) Keystroke Inference Module

Fig. 13. WindTalker in case study.

phase. After collecting the data, WindTalker infers the user's mobile payment password in the offline phase.

During collecting ICMP reply packets, WindTalker also collects additional network traffic packets from users' APPs. As pointed out in [15] and [23], only some particular types of packets (e.g., ICMP packets using "HT" rate) will be measured by Linux 802.11n CSI Tool. In our real-world experiments, CSI values will not be extracted from these packets generated by user's APPs. Besides, since CSI is the physical layer information which reflects the wireless channel environment, the CSI measurements are irrelevant to the types of network traffic packets. Thus even some addtional packets were measured by Linux 802.11n CSI Tool, they will only cause the CSI sampling rate vary slightly. Because we can record the timestamp of each CSI value, thus we can use the timestamps to reconstruct CSI stream to eliminate the impact of sampling rate variation. Fig. 13 shows the CSI waveforms reconstructed according to timestamps.

6.3 Recognizing the Sensitive Input Window

To determine the sensitive input window, WindTalker utilizes Wireshark to collect the meta data (eg., IP address) of the WiFi traffic during collecting CSI data. The meta data collected by Wireshark is shown in Fig. 13a. We can find that in the experiment, Alipay applications route their data to a server of some fixed IP address such as "110.75.xx.xx". These IP addresses are used by the Alipay service provider and do not change for one to two weeks. With the traffic meta data, as shown in Fig. 13a, WindTalker obtains the rough start time and end time of the sensitive input window via searching packets whose destination is "110.75.xx.xx". Then, according to the timestamps of CSI data, WindTalker locks the CSI data during this period of time.

6.4 CSI based Password Inference

Fig. 13b shows the original the 30th subcarrier CSI data in Sensitive Input Window. After data preprocessing, Fig. 13c shows the first three principal components of CSI data after PCA. It is found that in the real-world experiment that besides input payment password, victim may have other operations such as selecting credit card for payment in period of time of Sensitive Input. In order to handle this situation, WindTalker only needs to find a continuous keystroke of certain length. In our case, we are interested in continuous 6-bit password input since Alipay chooses 6-digit mobile payment password. Thus after keystroke extraction and recognition process, WindTalker is able to list possible password candidates according to probability. The top three password candidates in this case is 773,919, 773,619, 773,916 while the actual password is 773,919. We carry out the real-world experiment ten times, each time the password is different. Our experiment results show that the attacker can successfully recover 6, 8 and 9 passwords if allowing to try the password input for 5, 10 and 50 times (or Top 5, 10 and 50 candidates). This further demonstrates the practicality of the proposed attack in the practical environment.

7 Impact of Various Factors

There are many factors potentially affecting the CSI. The performance of WindTalker is affected by various factors such as relative position of AP and mobile device, CSI sampling rate, keyboard layout, human movement and temporal factors. Even clicking at the same key, the different distance and direction between AP and the mobile device may also lead to a different CSI. We will investigate the impact of these factors on WindTalker in our experiments.

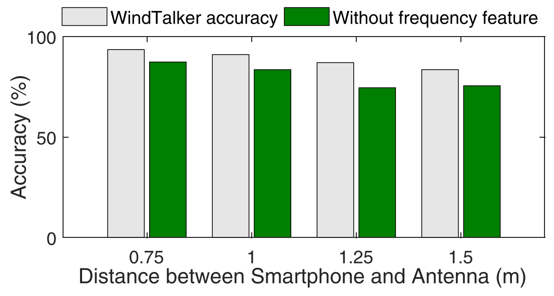
7.1 Distance

In a real scenario, the distance between the victim's mobile device and AP is not fixed. As shown in Fig. 14a, the recovery rate of WindTalker will decrease along with the increase of the distance. However, it is observed that, even if the distance between the antenna of WindTalker and victim's smartphone (i.e., Xiaomi) is enlarged to 1.5 m, WindTalker can still achieve keystroke inference accuracy of 83.5 percent in terms of 10-fold cross validation, which is high enough for launch keystroke inference. Fig. 14b shows that both of CSI shape and degree will change under different distance when pressing the same key. This indicates that WindTalker needs to retrain dataset even for the same victim with different distances. When the distance between antenna and victim is too long, the multiple-path propagate will become more complicated. Thus the collected CSI cannot reflect the victims finger precisely and result in inaccurate inference results. To partially solve these limitations, there are two possible solutions. First, the attacker can fix the location of table and chairs, which will make the user's position relatively stable. The other solution is placing three antennas of Intel 5,300 NIC at different locations to enlarge the effective range of WindTalker. Therefore, when the victim connects to rogue WiFi, WindTalker could dynamically choose the antenna which is closest to the victim to collect CSI data.

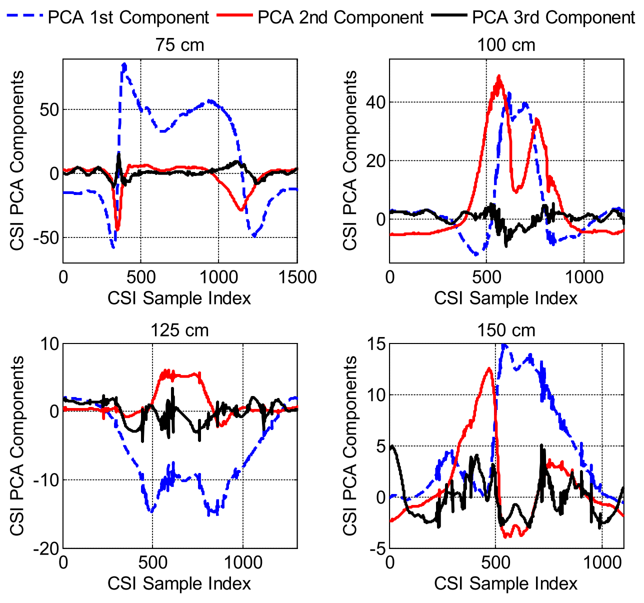
7.2 Direction

The relative direction between the victim and attacker may seriously affect the CSI since different directions mean different multi-path propagation between the transmitter and the receiver. Thus, we show the performance of WindTalker under different directions. Note that the mobile device (i.e., Xiaomi in this experiment) is in front of victim. It is important to point out that, for a right-handed user, WindTalker

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.



(a) Distance



(b) CSI Shape Change by Distance: PIN 1

Fig. 14. Distance's influence in WindTalker.

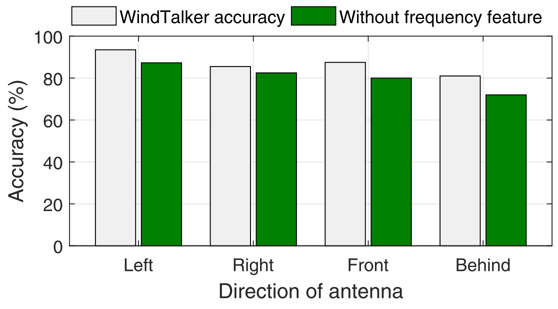


Fig. 15. Accuracy in different direction.

shows a better performance when the AP is on the left side of the victim. This is because it is easier for WindTalker to sense victim's finger clicks and the hand motion. Fig. 15 shows the keystroke inference accuracy of WindTalker in different direction in terms of 10-fold cross validation. It is interesting that WindTalker can achieve a high performance even the AP is deployed behind victims (i.e., 81 percent), which means that the proposed CSI based keystroke inference can work well even if the attacker is behind the user without visually seeing the clicking actions. This represents one of significant merits which cannot be achieved by any previous work. In real-world, the attacker can adjust the position and orientation of directional antenna to overcome the limitations of distance and direction.

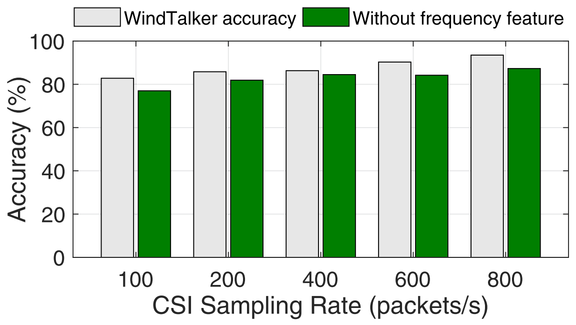


Fig. 16. Effect of CSI sampling rate.

7.3 Smartphone Type

The experiments in Sections 5 and 6 are implemented on Xiaomi, Redmi and Samsung Note3 smartphone. To evaluate the impact of different smartphone types, we recruit ten volunteers to generate 10 loop keystrokes on Xiaomi, Redmi and Samsung Note3. All of these mobile phones run with Android 5.0.2. When using all nine loops data, WindTalker achieves the average classification accuracy of 93.5, 88.3 and 83.9 percent on Xiaomi, Redmi and Samsung Note3 respectively. The experimental result indicates that the WindTalker performance is affected by the smartphone type, because different smartphones may have different relative positions of antennas and working powers. Fortunately, the accuracy of WindTalker on different smartphones are still acceptable for password inference.

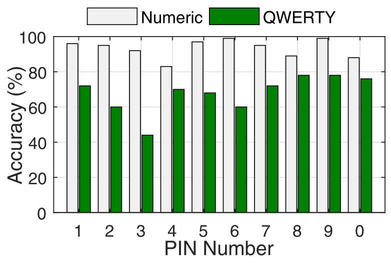
7.4 CSI Sampling Rate

The keystroke recognition accuracy depends on the sampling rate of CSI. When the CSI sampling rate is high, there is more information in the CSI waveform, which increases the keystroke recognition accuracy. Thus, we are interested in how the CSI sampling rate influences the performance of WindTalker. Fig. 16 shows the average classification accuracy of all volunteers with Xiaomi smartphone when varying the sampling rate from 100 packets/s to 800 packets/s. The experiment procedures are the same with Section 5.2 and the antenna is placed at the best position as mentioned in Sections 7.1 and 7.2. From Fig. 16, we observe that the classification accuracy improves when the sampling rate is higher, but the improvement is not significant beyond the sampling rate of 400 packets/s. For instance, with sampling rate of 400 packets/s, the classification accuracy of Xiaomi is 90.3 percent, which is only a slight drop compared to 93.5 percent achieved for a sampling rate of 800 packets/s. Because sampling rate of 400 packets/s is still enough to capture the movement feature of keystroke. When sampling rate reduces to 100 packets/s, the accuracy of Xiaomi reduces significantly to 82.8 percent, as this sampling rate loses the detailed feature of keystroke. In our experiment, we use the sampling rate of 800 packets/s to achieve the best performance of WindTalker. But when facing a high packet loss rate situation, we can use the a lower sampling rate above 100 packets/s to achieve an acceptable performance.

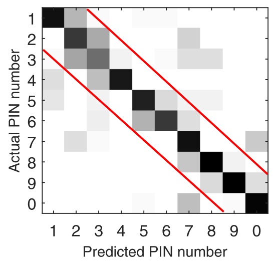
7.5 Keyboard Layout

There are two different keyboard layouts which influence the keystroke recognition accuracy. Besides the numeric

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

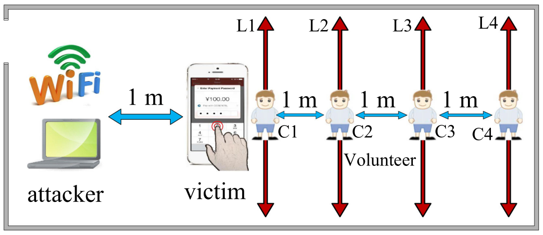


(a) Classification Accuracy on Different Keyboard Layout

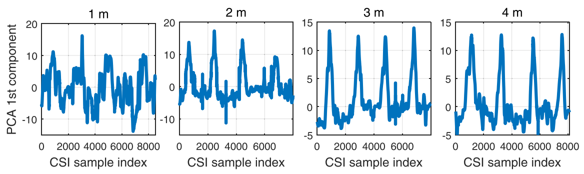


(b) Confusion Matrix about QWERTY Keyboard

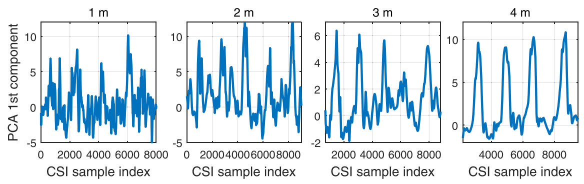
Fig. 17. Keystroke recognition on QWERTY keyboard.



(a) The experiment environment.



(b) The CSI data under human walking scenario.



(c) The CSI data under human arms movement scenario.

Fig. 18. Effect of human movement.

keyboard which is used in most online payment scenarios, there is QWERTY keyboard on which a user can type letters, numbers and special characters. The main difference between the two keyboards is the key space. Comparing to typing numeric keyboard, the hand movement tends to be subtle when typing adjacent keys on QWERTY keyboard, which makes recognizing keystrokes much more difficult since the CSI waveforms become similar.

We are interested in how the QWERTY keyboard influences the recognition of keystroke. For simplicity, we just focus on the digital input on the QWERTY keyboard. We perform experiment on Xiaomi phone and the keyboard layout is provided by Google input method. Fig. 17a shows average classification accuracy on both numeric and QWERTY keyboard. We observe that the accuracy of QWERTY keyboard is 67.8 percent, which significantly drops compared to 93.5 percent of numeric keyboard. Fig. 17b is the confusion matrix of QWERTY keyboard. We observe that most error recognition happened between the adjacent keys. Although the accuracy of QWERTY keyboard is lowered than numeric keyboard, but it still higher than the random guess.

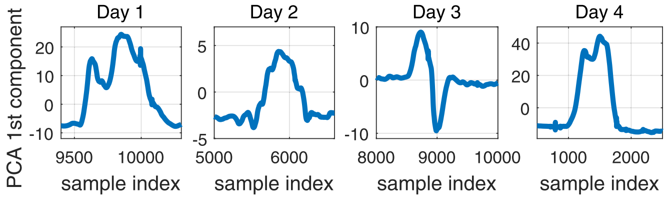


Fig. 19. CSI waveforms of PIN number 1 on different days.

7.6 Human Movement

In some cases, the CSI-based sensing may be affected by the movement of other nearby human. Thus, we evaluate the impact of human walking and human arm movement on the performance of WindTalker. As shown in Fig. 18a, while WindTalker collecting the CSI data to infer the victims keystroke, we recruit a volunteer to walk along four different lines (L1, L2, L3, L4) respectively. The distances between WindTalkers antenna and the midpoints of L1, L2, L3 and L4 are 1 meter, 2 meters, 3 meters and 4 meters respectively. The distance between antenna and the victim is 1 meter. We totally conduct four experiments. In each experiment we ask the victim to continuously generate keystrokes and collect the corresponding CSI, at the same time, the volunteer walks along one of the four lines with the speed of 0.5 m/s. Fig. 18b shows the experimental result. When the distance between antenna and the midpoint of walking mans trajectory is larger than 2 meters, the keystrokes could be easily found from the collected CSI waveforms. However, when the distance is 1 meter (i.e., the walking mans trajectory is very close to the victim), it is hard to extract keystroke waveforms from collected CSI data. The results show that the humans walking will bring additional multiple-path effects into the wireless transmission. However, WindTalker is still effective if only there is no human walking within 2 meters of the WindTalkers antenna.

Besides human walking, we also consider another scenario in which a human stays at a fixed location but waves his/her arms. We conduct four experiments. When the victim continuously generating keystrokes, we ask the volunteer stays at the midpoints (i.e., C1, C2, C3, C4 in Fig. 18a) of above four lines respectively and spins his/her arm with the average speed of 0.91 cycle per second. As shown in Fig. 18c, when the distance between antenna and volunteer is larger than 3 meters, the keystrokes could be recognized from collected CSI data. When the distance is 1 meter (i.e., the victim is very close to another people), the keystrokes is hard to be extracted. Therefore, WindTalker will work normally only if there is no user waves his/her arms within 3 meters of the WindTalkers antenna.

7.7 Temporal Factors

The temporal factors will also affect the performance of WindTalker. Fig. 19 shows how CSI waveform changes on different days. We can observe that these CSI shape patterns look different. The reason is that in different days, the users typing behaviour may be inconsistent and the surrounding environment may change, which may affect the constructive and destructive interference of several multi-path signals. Therefore, in current state, for each keystroke inference, WindTalker needs to update the users CSI profiles to ensure its performance. We leave this limitation for future work.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

8 Countermeasures

8.1 Basic Defense Strategies

8.1.1 Randomizing the Keyboard Layout

One of the most straightforward defense strategies is to randomize the layouts of the PIN keyboard, such that the attacker cannot recover the typed PIN number even if he can infer the keystroke positions on the touchscreen. As pointed out by [7], randomizing the keyboards is effective at the cost of the user experience since the user needs to find every key on a random keyboard layout for every key typing.

8.1.2 Changing Typing Gesture

For WindTalker, collecting the accurate CSI data is essential for achieving high inference success rate. Thus the user can intentionally change his typing gestures or clicking patterns to introduce the unexpected interference to the CSI data. For example, the randomized human behaviors (e.g., human mobility) would introduce more impact on CSI than finger click on wireless signals, which reduce the successful chance of the adversary.

8.1.3 Refusing to Connect to Rogue WiFi

The most thorough defense strategy is refusing to connect to rogue WiFi hotspot. For instance, [24] and [25] proposed a method which can detect a rogue WiFi hotspot. These detection systems suppose that both of the rogue hotspot and the legitimate hotspot have the same SSID. However, if the attacker uses a new SSID that is not observed before by the detection system, it will fail either.

8.1.4 Blocking the ICMP Echo Request

Our CSI based typing inference requires collecting CSI data with a high frequency. According to [26], the data received in the echo message must be returned in the echo reply message. It means that the victims device must reply to the attackers WiFi hotspot when it received the ICMP echo request. A countermeasure for the user is configuring the firewall to detect and block the high-frequency ICMP echo requests. But this countermeasure is rarely used in Android smartphones, because it needs to be implemented at the operation system level and the common users have no access to it[27]. As far as we have tested in 3 mainstream un-rooting smart phones(Xiaomi, Redmi and Samsung), none of them have blocked this kind of ICMP echo request, because it will forbid other hosts to ping the user device and affect the user experience.

8.2 CSI Obfuscation Algorithm

In this section, we propose a novel obfuscation strategy to defend against the CSI based side channel attacks. Our goal is preventing the attackers from collecting the accurate CSI data introduced by users password input. In the ideal case, the strategy can be implemented and deployed at the users side and can be triggered in a user-transparent way as long as any sensitive input time window is observed. This strategy does not need the user's participation and thus minimize its impact on the user experience.

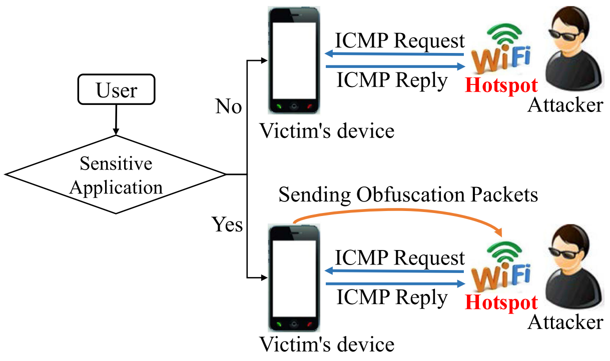


Fig. 20. Obfuscation defense strategy.

8.2.1 Overview of the Basic Idea

The basic idea of the proposed defense strategy is introducing a randomly generated CSI time series sequence to obfuscate the original one. As shown in Fig. 20, during the sensitive input time window, when the attacker collects the CSI data (or original CSI data) from the target user, the user device can randomly generate some CSI data (or obfuscation data) to obfuscate the original CSI data and thwart the side channel attack. According to 802.11n standard [16], when the user does not launch sensitive applications, the attacker can obtain the CSI data by analyzing the training sequence of the preamble of the WiFi packet obtained from the victims device. Without loss of the generality, the original CSI between victim and attacker is estimated as

$$H_1 = \frac{Y_1}{X_1}, \tag{9}$$

where X_1 is the training sequence on transmitter and Y_1 is the training sequence on receiver. In practice, both the transmitter and receiver assume that the training sequence X_1 will not change during the whole communication process.

During the password input progress for a specific mobile payment application, (eg. Alipay), the defense strategy will be launched. The attacker uses the ICMP requests to obtain WiFi packets from the victim, and, at the same time, the users device can also proactively sends the obfuscation packets to the attacker. For instance, the training sequence X_1 in Equation. (9) was changed into

$$X_2 = \Delta H X_1. \tag{10}$$

The revised training sequence will be received by attacker as

$$Y_2 = H_1 X_2 = H_1 \Delta H X_1 = H_2 X_1. \tag{11}$$

From the attacks perspective, it is indistinguishable for the original and obfuscation data. Because the attacker still utilizes original training sequence X_1 to estimate CSI, therefore the attacker would estimate victims CSI as $H_2 = H_1 \Delta H$. It means that the original CSI data will be masked by inserting forged CSI data H_2 into the original CSI sequence H_1. Thus the CSI based side channel attack can be thwarted because the attacker cannot infer the users keystroke by analyzing CSI data.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

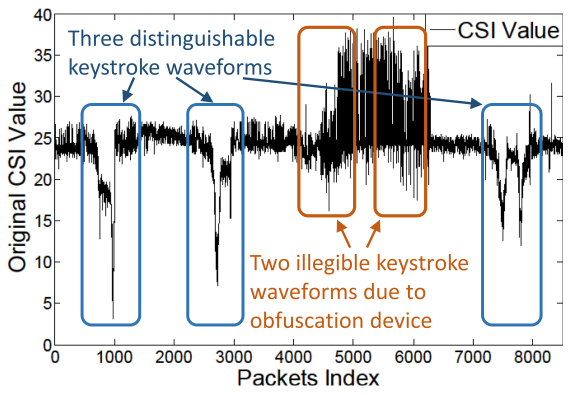


Fig. 21. The 4th subcarrier waveform in the experiment.

8.2.2 *Experiment Evaluations*

We perform an experiment to prove the effectiveness of our proposed strategy. In the ideal case, it should be the users device that generates the obfuscation data. In our experiments, we adopt a mobile phone as the target device and another phone as the obfuscation device to perform the proof-of-concept experiments and evaluate the effectiveness of the proposed defending strategy. In practice, to implement this defense strategy in users devices, we can use Software Defined Radios (SDR) to revise the training sequence of mobile device [28].

In our experiment, both devices are connected to a WiFi hotspot released by WindTalker. The WindTalker uses *ICMP based CSI Acquirement Model* to obtain CSI data H_1 from the victim, and during this period, the victim continuously types PIN numbers. When victim launches sensitive application (e.g., Alipay), the obfuscation device continuously sends packets (e.g., UDP packets) to WindTalker so that the WindTalker receives mixed CSI data. Note that, the obfuscation device is placed at different places to get a different CSI estimated value H_2.

The result is shown in Fig. 21. We can find that without the involvement of obfuscation device, the WindTalker works normally and the finger clicks are easily distinguished in CSI H_1. With the involvement of obfuscation device, the finger click patterns are obfuscated with the forged CSI measurements H_2. So the effectiveness of this defense strategy is demonstrated. The [28], [29] and [30] have discussed how to implement it in the SDR system. To apply this method to mobile phone, the operation system kernel of the phone should be revised [16], which is out of the scope of this work. We will leave it for the future work.

9 Limitations

In this section, we discuss the main limitations of WindTalker. WindTalker's high performance is achieved in an experiment environment. However, if we try to apply WindTalker in anytime and anyplace, we need to overcome the limitations as follows.

Hardware Limitations. In WindTalker, we use Intel 5,300 NIC and Linux 802.11n CSI Tool [15]. In our experiments, it is observed that the system will crash when we perform CSI data collection for iPhone or some versions of android smart phones. This is because, according to the statement of the CSI Tool, it is very easy to crash when one Intel 5,300 NIC works with other NICs (e.g., an iPhone). However, our implementation and evaluation on a wide range of smart phones (including Xiaomi, Redmi and Samsung phones) demonstrate the practicality of the proposed CSI based keystroke inference method. We will leave the issues of improving the compatibility of Intel 5,300 NIC with a wider range of mobile devices to our future work.

Fixed Typing Gesture. Currently, WindTalker can only work for the situation that the victim can only touch the screen with a relatively fixed gesture and the phone needs to be placed in a relative stable environment (e.g., a table). In reality, the user may type in an ad-hoc way (e.g., the victim may hold and shake the phone, or even perform some other actions while typing). We argue that is a common problem for most of the side channel based keystroke inference schemes such as [2], [8], [10]. This problem can be partially circumvented by profiling the victim ahead or performing a targeted attack by applying the relevant movement model as pointed out by [8].

User-specific Training. WindTalker needs to extract the keystroke samples from the victim before launching password inference attack. This requirement is a common assumption for most of the side channel keystroke inference attacks such as [2], [9], [31], [32], [33], [34]. To launch a real-world attack, the attacker can consider the following two strategies. First, WindTalker could leverage some social engineering methods to collect training data from victim. For example, the attacker could implement online training by mimicking a Text Captcha to require the victim to input the chosen numbers. As shown in the Section 5.3, given three training samples per key, WindTalker could achieve 6-digit password inference accuracy of 69.6 percent under top 5 password candidates. The second strategy is using the self-contained structures of collected CSI data. For example, our follow-up work [35] proposes a non-training CSI based keystroke inference system. In this system, the attacker extracts the correlations between the CSI features of keystrokes, and then maps the collected CSI data to a word within a predefined dictionary. Applying this idea in our 6-digital password inference scenario maybe a potential solution, and we leave it for future work.

10 Related Work

10.1 Free Public WiFi with Malicious Behaviors

Free Wi-Fi services provided by public hotspots are attractive to users in a mobile environment when their mobile devices have limited Cellular connection. Existing works [36], [37], [38], [39], [40], [41] have demonstrated it is feasible to deploy a malicious Wi-Fi hotspot in a public area. For example, an iPhone can turn itself into a Wi-Fi hotspot. If the iPhone user changes the session ID to "Starbucks Free Wi-Fi", other people may connect their phones to the iPhone while wrongly believe they are using free WiFi services from a nearby Starbucks. In such a scenario, the attacker can utilize the WiFi network traffic collected by the WiFi hotspot to infer the user's privacy information. Taylor et al. [42] proposed methodology to fingerprint and identify Android apps by analyzing the encrypted HTTPS/LTS traffic. Alan et al. [43] proposed a method to identify mobile apps only using the TCP/IP headers from the apps launch time traffic. Finally, Conti et al. [44] presented a system that using

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

network traffic to identify the specific actions that a user is performing on his/her mobile apps.

Compared with these previous works, our WinderTalker utilizes both network layer traffic and physical layer CSI information to infer the user's sensitive information. In our considered scenarios, attacker lures the users to connect their devices to a fake access point. Then, the attacker eavesdrops the WiFi traffic to identify the sensitive window and selectively analyzes the CSI information to infer the sensitive keystroke information.

10.2 Keystroke Inference Methods

Prior keystroke inference methods utilized the information from various sensors and communication channels, such as motion, camera, acoustic signals, and WiFi signals.

Motion. Owusu et al. [10] presented an accelerometer-based keystroke inference method, which aims to recover six-character passwords on smartphones. Later, Liu et al. [8] applied a similar idea to the smartwatch scenario. Their objective is to track user's hand movement over the keyboard using the accelerometer readings from the smartwatch, and the keystroke inference achieves 65 percent recognition accuracy.

Acoustic Signals. Zhu et al. [5] presented a context-free and geometry-based keystroke inference. They leverage the microphones of a smartphone to record keystrokes' acoustic emanations and the experimental results show that more than 72.2 percent of keystrokes can be accurately recovered. Liu et al. [4] further proposed a keystroke snooping system by exploiting the audio hardware to distinguish mm-level position difference. The accuracy of their system can achieve 94 percent. These works could achieve high accuracy on both digital and QWERTY keyboard. However, compared with these works, WindTalker requires neither a fixed position nor a close distance to the victim. Furthermore, WindTalker can obtain the network traffic information, which improves the practicality in real-world environments.

Camera Based. Yue et al. [7] introduces a camera-based keystroke inference using Google Glass or off-the-shelf webcam. Shukla et al. [6] also presented a video-based attack relies on the spatio-temporal dynamics of the hands during typing. Sun et al. [34] use camera to record tablet backside motion and infer the victim's typing content.

WiFi Signal Based. Using WiFi signals to infer the keystroke draws a large research attention because it offers device-free and non-invasion advantages. Ali et al. [2] proposed a keystroke inference systems called WiKey, which uses the CSI waveform pattern generated by finger's unique motion to distinguish keystrokes on a external keyboard. Zhang et al. [11] presented WiPass, which can work in mobile device to detect the graphical unlock passwords. Tan et al. [12] also proposed WiFinger, which leverage CSI from COTS device to capture the user's fine-grained finger gesture. Compared with our work, WiKey, WiPass and WiFinger don't utilize the network traffic information, thus these schemes work on the OKI keystroke inference model and they can not recognize the user's sensitive input window.

11 Conclusion and Future Work

In this paper, we have proposed a novel side-channel attack model named WindTalker, which can be used to infer a victim's mobile password via WiFi signals. WindTalker is a cross-layer inference system, which utilizes both network layer traffic information and physical layer CSI information. Our experiments on Alipay shows that WindTalker can be effective in recognizing the victim's password on smart phones. Compared with previous works, WindTalker neither deploys external devices close to the target device nor compromises the target device. Furthermore, we proposed the CSI obfuscation based countermeasure and performed the experiment to prove the effectiveness of this countermeasure method.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61672350, U1401253, No. 61702330) and the National Science Foundation (No. 1553304, No. 1618893). The preliminary version of this paper titled "When CSI Meets Public WiFi: Inferring Your Mobile Phone Password via WiFi Signals" was published in the *Proc. of ACM CCS*, 2016 [1].

References

[1] M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "When CSI meets public WiFi: Inferring your mobile phone password via WiFi signals," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1068–1079.

[2] K. Ali, A. X. Liu, W. Wang, and M. Shahzad, "Keystroke recognition using WiFi signals," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, 2015, pp. 90–102.

[3] D. Balzarotti, M. Cova, and G. Vigna, "ClearShot: Eavesdropping on keyboard input from video," in *Proc. IEEE Symp. Secur. Privacy*, 2008, pp. 170–183.

[4] J. Liu, Y. Wang, G. Kar, Y. Chen, J. Yang, and M. Gruteser, "Snooping keystrokes with mm-level audio ranging on a single phone," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, 2015, pp. 142–154.

[5] T. Zhu, Q. Ma, S. Zhang, and Y. Liu, "Context-free attacks using keyboard acoustic emanations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 453–464.

[6] D. Shukla, R. Kumar, A. Serwadda, and V. V. Phoha, "Beware, your hands reveal your secrets!" in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 904–917.

[7] Q. Yue, Z. Ling, X. Fu, B. Liu, K. Ren, and W. Zhao, "Blind recognition of touched keys on mobile devices," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 1403–1414.

[8] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When good becomes evil: Keystroke inference with smartwatch," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1273–1285.

[9] P. Marquardt, A. Verma, H. Carter, and P. Traynor, "(sp) iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers," in *Proc. 18th ACM Conf. Comput. Commun. Secur.*, 2011, pp. 551–562.

[10] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang, "ACCessory: Password inference using accelerometers on smartphones," in *Proc. 12th Workshop Mobile Comput. Syst. Appl.*, 2012, pp. 1–6.

[11] J. Zhang, X. Zheng, Z. Tang, T. Xing, X. Chen, D. Fang, R. Li, X. Gong, and F. Chen, "Privacy leakage in mobile sensing: Your unlock passwords can be leaked through wireless hotspot functionality," *Mobile Inf. Syst.*, vol. 2016, 2016, Art. no. 8793025.

[12] S. Tan and J. Yang, "WiFinger: Leveraging commodity WiFi for fine-grained finger gesture recognition," in *Proc. 17th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2016, pp. 201–210.

[13] S. Sen, J. Lee, K.-H. Kim, and P. Congdon, "Avoiding multipath to revive inbuilding WiFi localization," in *Proc. 11th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2013, pp. 249–262.

[14] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity WiFi," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, 2015, pp. 53–64. [Online]. Available: http://doi.acm.org/10.1145/2789168.2790124

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

[15] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, “Tool release: Gathering 802.11 n traces with channel state information,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, pp. 53–53, 2011.

[16] IEEE Std. 802.11n-2009: Enhancements for higher throughput, 2009. [Online]. Available: http://www.ieee802.org

[17] H. Benko, A. D. Wilson, and P. Baudisch, “Precise selection techniques for multi-touch screens,” in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2006, pp. 1263–1272.

[18] C. Forlines, D. Wigdor, C. Shen, and R. Balakrishnan, “Direct-touch vs. mouse input for tabletop displays,” in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2007, pp. 647–656.

[19] F. Wang, X. Cao, X. Ren, and P. Irani, “Detecting and leveraging finger orientation for interaction with direct-touch surfaces,” in *Proc. 22nd Annu. ACM Symp. User Interface Softw. Technol.*, 2009, pp. 23–32.

[20] S. Sardy, P. Tseng, and A. Bruce, “Robust wavelet denoising,” *IEEE Trans. Signal Process.*, vol. 49, no. 6, pp. 1146–1152, Jun. 2001.

[21] C. C. Holt, “Forecasting seasonals and trends by exponentially weighted moving averages,” *Int. J. Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.

[22] J. Russell, “Alipay, china’s top mobile payment service, expands to the U.S,” [Online]. Available: https://techcrunch.com/2017/05/09/alipay-first-data-us-point-of-sale-ex pansion/, Accessed on: May 9, 2017.

[23] D. Halperin, “Linux 802.11n CSI tool,” [Online]. Available: http://dhalperi.github.io/linux-80211n-csitool/faq.html

[24] O. Nakhila, E. Dondyk, M. F. Amjad, and C. Zou, “User-side Wi-Fi evil twin attack detection using SSL/TCP protocols,” in *Proc. 12th Annu. IEEE Consum. Commun. Netw. Conf.*, Jan. 2015, pp. 239–244.

[25] O. Nakhila and C. Zou, “User-side Wi-Fi evil twin attack detection using random wireless channel monitoring,” in *Proc. IEEE Military Commun. Conf.*, Nov. 2016, pp. 1243–1248.

[26] J. Postel, et al., “Internet control message protocol,” *Internet Engineering Task Force, RFC 792*, Sep. 1981. [Online]. Available: http://www.rfc-editor.org/rfc/rfc792.txt.

[27] D. Vecchiato and E. Martins, “Experience report: A field analysis of user-defined security configurations of android devices,” in *Proc. IEEE 26th Int. Symp. Softw. Rel. Eng.*, Nov. 2015, pp. 314–323.

[28] Y. Mao, Y. Zhang, and S. Zhong, “Stemming downlink leakage from training sequences in multi-user MIMO networks,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 1580–1590.

[29] S. Fang, Y. Liu, W. Shen, H. Zhu, and T. Wang, “Virtual multipath attack and defense for location distinction in wireless networks,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 566–580, Feb. 2017.

[30] Y.-C. Tung, S. Han, D. Chen, and K. G. Shin, “Vulnerability and protection of channel state information in multiuser MIMO networks,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 775–786.

[31] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, “Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds,” in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, 2009, pp. 199–212.

[32] B. Chen, V. Yenamandra, and K. Srinivasan, “Tracking keystrokes using wireless signals,” in *Proc. 13th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2015, pp. 31–44.

[33] J. Wang, K. Zhao, X. Zhang, and C. Peng, “Ubiquitous keyboard for small mobile devices: Harnessing multipath fading for fine-grained keystroke localization,” in *Proc. 12th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2014, pp. 14–27.

[34] J. Sun, X. Jin, Y. Chen, J. Zhang, R. Zhang, and Y. Zhang, “VISIBLE: Video-assisted keystroke inference from tablet backside motion,” in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2016, pp. 1–15.

[35] Y. Liu, S. Zhao, Z. Lu, S. Fang, I. Markwood, and H. Zhu, “No training hurdles: Fast training-agnostic attacks to infer your typing,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 1747–1760.

[36] H. Li, Z. Xu, H. Zhu, D. Ma, S. Li, and K. Xing, “Demographics inference through Wi-Fi network traffic analysis,” in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[37] N. Cheng, X. O. Wang, W. Cheng, P. Mohapatra, and A. Seneviratne, “Characterizing privacy leakage of public WiFi networks for users on travel,” in *Proc. IEEE Int. Conf. Comput. Commun.*, Apr. 2013, pp. 2769–2777.

[38] H. Li, H. Zhu, S. Du, X. Liang, and X. Shen, “Privacy leakage of location sharing in mobile social networks: Attacks and defense,” *IEEE Trans. Depend. Secure Comput.*, vol. 15, no. 4, pp. 646–660, Jul./Aug. 2018.

[39] Y. Fan, Y. Jiang, H. Zhu, and X. Shen, “An efficient privacy-preserving scheme against traffic analysis attacks in network coding,” in *Proc. IEEE Int. Conf. Comput. Commun.*, Apr. 2009, pp. 2213–2221.

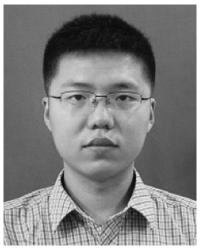
[40] B. Konings, C. Bachmaier, F. Schaub, and M. Weber, “Device names in the wild: Investigating privacy risks of zero configuration networking,” in *Proc. IEEE 14th Int. Conf. Mobile Data Manage.*, 2013, pp. 51–56.

[41] N. Xia, H. H. Song, Y. Liao, M. Iliofotou, A. Nucci, Z.-L. Zhang, and A. Kuzmanovic, “Mosaic: Quantifying privacy leakage in mobile networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 279–290, 2013.

[42] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, “AppScanner: Automatic fingerprinting of smartphone apps from encrypted network traffic,” in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2016, pp. 439–454.

[43] H. F. Alan and J. Kaur, “Can android applications be identified using only TCP/IP headers of their launch time traffic?” in *Proc. 9th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, 2016, pp. 61–66.

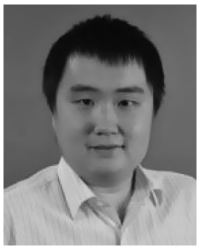
[44] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, “Analyzing android encrypted network traffic to identify user actions,” *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 1, pp. 114–125, Jan. 2016.



Yan Meng received the BS degree in electronic and information engineering from the Huazhong University of Science and Technology, in 2016. He is working toward the PhD degree in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His research interests include wireless network security and social network privacy.

Jinlei Li received the BS degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University of China, in 2018. He is working toward the master’s degree in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His research interests are wireless network security and network privacy.

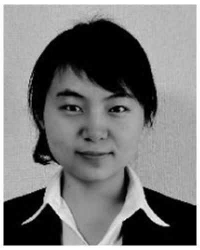
Haojin Zhu received the BSc degree from Wuhan University, China, in 2002, the MSc degree from Shanghai Jiao Tong University, China, in 2005, both in computer science and the PhD degree in electrical and computer engineering from the University of Waterloo, Canada, in 2009. He is currently a full professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. His current research interests include network security and privacy protection. He received the IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award (2014) due to his contribution to wireless network security and privacy, and was a distinguished member of the IEEE INFOCOM 2015 technical program committee. He is a senior member of the IEEE.



Xiaohui Liang received the BSc degree in computer science and engineering and the MSc degree in computer software and theory from Shanghai Jiao Tong University (SJTU), China, in 2006 and 2009, respectively. He is currently working toward the PhD degree in the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include applied cryptography, and security and privacy issues for e-healthcare system, cloud computing, mobile social networks, and smart grid. He is a member of the IEEE.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.

Yao Liu received the PhD degree in computer science from North Carolina State University, in 2012. She is currently an assistant professor with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL. Her research is related to computer and network security, with an emphasis on designing and implementing defense approaches that protect emerging wireless technologies from being undermined by adversaries. Her research interest also lies in the security of cyber-physical systems, especially in smart grid security. She is a member of the IEEE.



Na Ruan received the BS degree in information engineering, the MS degree in communication and information system from the China University of Mining and Technology, in 2007 and 2009 respectively, and the DE degree from the Faculty of Engineering, Kyushu University, Japan, in 2012. In 2013, she joined the Department of Computer Science and Engineering of Shanghai Jiaotong University as an assistant professor. Her current research interests are in wireless network security and game theory. She is a member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: Shanghai Jiaotong University. Downloaded on August 30,2023 at 11:58:29 UTC from IEEE Xplore. Restrictions apply.