Accurate and Efficient Wireless Device Fingerprinting Using Channel State Information

Jingyu Hua*†, Hongyi Sun*†, Zhenyu Shen*†, Zhiyun Qian‡ and Sheng Zhong*†
*State Key Laboratory for Novel Software Technology, Nanjing University, China
†Department of Computer Science and Technology, Nanjing University, China
Email: huajingyu2012@gmail.com, hysquall@gmail.com, shenzhenyuyuyu@gmail.com, sheng.zhong@gmail.com
‡University of California, Riverside, USA
Email: zhiyunq@cs.ucr.edu

***Abstract*—Due to the loose authentication requirement between access points (APs) and clients, it is notoriously known that WLANs face long-standing threats such as rogue APs and network freeloading. Take the rogue AP problem as an example, unfortunately encryption alone does not provide authentication. APs need to be equipped with certificates that are trusted by clients ahead of time. This requires either the presence of PKI for APs or other forms of pre-established trust (e.g., distributing the certificates offline), none of which is widely used. Before any strong security solution is deployed, we still need a practical solution that can mitigate the problem. In this paper, we explore a non-cryptographic solution that is readily deployable today on end hosts (e.g., smartphones and laptops) without requiring any changes to the APs or the network infrastructure. The solution infers the Carrier Frequency Offsets (CFOs) of wireless devices from Channel State Information (CSI) as their hardware fingerprints without any special hardware requirement. CFO is attributed to the oscillator drift, which is a fundamental physical property that cannot be manipulated easily and remains fairly consistent over time but varies significantly across devices. The real experiments on 23 smartphones and 34 APs (with both identical and different brands) in different scenarios demonstrate that the detection rate could exceed 94%.**

***Index Terms*—device fingerprinting, attack detection, authentication, wireless networks**

I. Introduction

WiFi networks are attracting various kinds of attacks due to their extremely high popularity. Among these attacks, rogue Access Points (i.e., rogue APs) and WiFi Freeloading are the most common which bring significant security and privacy threats. A rogue AP is a device set up by an adversary to mimic the legitimate AP in public places such as coffee shops and shopping malls. It usually uses the same identifiers (basic service set identifier (BSSID) and service set identifier (SSID)) as the original AP. Once users are fooled to connect to it, the adversary could launch man-in-the-middle attack and eavesdrop all the network communications. It has been estimated that almost 20% of corporations have rogue APs in their networks [1]. WiFi freeloading refers to that an unauthorized user compromises or bypasses the authentication of an AP and then gets into the private WLAN for free. Note that a freeloader may be stealing more than just bandwidth considering that he has become an insider of the network.

A key point to defend against these attacks is a strong mutual authentication mechanism between clients and APs. In fact, the wireless security enhancement 802.11i RSNA (Robust Security Network Association) does provide optional mutual authentication using traditional cryptographic methods (i.e., digital certificates), which can make both attacks less likely if implemented properly. Unfortunately, as mentioned by Jana et al. [2], wireless networks using 802.11i RSNA still suffers from vulnerabilities due to several practical issues. For instance, as the signal strength is the only criteria for clients to select AP in the current implementation, users can be tricked to connect to a fake AP with a higher signal strength than that of the real one but does not support any security measures such as RSNA. Even worse, as the management and distribution of digital certificates are extremely cumbersome, most networks simply choose to support only user authentication but never AP authentication. As a result, it is still easy for adversaries to deploy fake APs. For the freeloading attack, although most networks authenticate users, the authentication is based on human-determined passwords which are usually quite weak and can be easily compromised or disclosed especially when all users share one password (e.g., WPA2-PSK).

Motivated by the above, researchers have proposed noncryptographic solutions based on device fingerprinting recently. These solutions are not meant to replace cryptographic solutions. Rather, they aim to provide an extra layer of security in light of the difficulties in adopting cryptographic solutions. Their common idea is to identify physical characteristics of the hardware that can be extracted remotely to fingerprint wireless devices [2]. An example scenario where this technique is helpful is the following: when a user goes to a coffee shop visiting frequently, without fingerprinting techniques, he will easily be tricked to connect to a rogue AP with the same identity as the real one. With fingerprinting, if the fingerprint (remembered by the client) changes, the user could be alerted that he is likely connecting to a rogue AP.

While a promising direction, there is no real-world deployment of such fingerprinting techniques to our knowledge. This is due to several important practical issues. First, the solution may require special hardware which hinders the

J. Hua was supported in part by NSFC-61300235. This work was supported in part by NSFC-61425024, NSFC-61402223, the Jiangsu Province Double Innovation Talent Program, and NSFC-61321491. (Corresponding author: Sheng Zhong.)

978-1-5386-4128-6/18/$31.00 ©2018 IEEE

deployment. For instance, Brik et al. [3] propose to use some Radio Frequency (i.e., RF) features such as the frequency error, magnitude error, sync correlation and I/Q offset to distinguish among different wireless devices. However, they require additional hardware to capture and analyze the radio signals. Second, the hardware characteristics can be spoofed and therefore the security guarantee is questionable. Kohno et al. [4] present a rogue AP detecting scheme which leverages the clock screws measured by TCP/ICMP time stamps as new device fingerprints. Unfortunately, Jana and Kasera [2] haven shown that TCP/ICMP timestamps in general could be easily spoofed. They instead measure the Time Synchronization Function (TSF) timestamps in the beacon/probe response frames (from the AP), which are tagged by hardware and somewhat hard to spoof. Nevertheless, it has been shown that it is still possible to spoof such timestamps by modifying the device driver of a fake AP [5].

In this paper, we investigate a novel wireless device fingerprinting approach that can avoid the above problems and is applicable to both rogue APs and WiFi freeloading. In particular, our scheme fingerprints a device by estimating its Carrier Frequency Offset (CFO) compared with the fingerprinting node. CFO is attributed to the oscillator drift, which remains fairly consistent over time but varies significantly across devices. What's more, such oscillator drift is caused by the crystal imperfection and cannot be spoofed by any software. As a result, it could act as an ideal fingerprint.

The major challenge is that on both mobile devices and APs, there is no software approach to estimate the CFO from the underlying hardware. The state-of-the-art [3], [6] typically requires additional hardware (e.g., vector signal analyzer and USRP) to capture and analyze the raw signals, which prevents them from being applied to fingerprinting in practice. Unlike their work, we propose to mine the CFO indirectly from the Channel State Information (CSI), which is easily measurable using software on off-the-shelf wireless devices [7]. We can achieve this because CFO will contribute a specific phase offset at the receiver signal and the CSI measurement rightly contains a field that records the total phase offset during signal propagation, which includes the offset due to CFO. This task is no-trivial because there are many other factors (e.g., Time of flight) can also contribute to the final offset and we have to filter out their interferences.

In summary, we make the following contributions:

(1) We propose the first CFO-based WiFi device fingerprinting mechanism that are readily deployable on off-the-shelf wireless devices such as smarphones. Experiments show that such fingerprints are consistent over time and locations.

(2) We propose a novel approach to precisely and quickly estimate CFO from CSI, which does not require any additional hardware and is difficult to spoof.

(3) We implement a prototype system to verify the performance of our scheme. The results show that our approach can achieve a high accuracy that can be used in real world for applications such as rogue AP detection.

II. Background of CSI & CFO

CSI (*Channel State Information*): it describes the combined effect of, e.g., scattering, fading and power decay on the signal propagation from the transmitter to the receiver. The IEEE 802.11 standard defines a mechanism to measure CSI between each transmitter-receive (Tx-Rx) antenna pair. The CSI continually captures signal strength and phase information of each OFDM subcarrier. Let X and Y be the receive and transmit signal vectors, respectively, and H and N be the channel matrix and noise vector, respectively. We have $Y = H \times X + N$, where H is a complex vector, called *Channel Frequency Response* (CFR) which reflects the signal gain between the Tx-Rx pair. Those information can be used to achieve reliable communication with high data rate. The CSI measurements are the sampling for the CFR at different subcarriers. At the 2.4Ghz frequency band with bandwidth 20Mhz, the CSI measurements consist of 30 complex values, where each corresponds to a selected subcarrier. Let N_{tx} and N_{rx} be the number of transmit and receive antennas, respectively. There are $30 \times N_{tx} \times N_{rx}$ CSI streams for a received 802.11 frame. The CSI stream for the kth subcarrier between the ith transmit antenna and the jth receive antennas can be expressed as $H_{k,i,j} = |H|e^{-j\phi_{k,i,j}}$, where $|H|$ is the amplitude and $\phi_{k,i,j}$ is the phase part of subcarrier k.

CFO: For OFDM system, the carrier frequency f should be the same between Tx-Rx pair in the ideal situation. But due to the hardware imperfection, there usually exists a offset between the Tx-Rx oscillators, which causes a *Carrier Frequency Offset* (CFO). Since a large CFO will introduce a big noise at the receiver end, the CFO is compensated by the hardware. But there still exists a fractional CFO, Δf_c, after such compensation because the hardware imperfection. The CFO results in a phase shift φ_t at the receiver signal, which can be represented as $\varphi_t = 2\pi\Delta f_c t$, where Δf_c is the CFO after compensation. For commercial WiFi devices, the fractional CFO is inevitable. The fractional CFO can be up to 100 kHz according to IEEE 802.11n standard. For convenience, the word CFO always means the fractional CFO in this article.

III. Threat Model

In this paper, we focus on the following two major threats:

Rogue AP: An adversary sets up an unauthorized AP to masquerade an authorized AP in public places such as airport and coffee shop. We assume that the adversary is power enough to modify the SSID and BSSID fields of each frame to the same values of the authorized AP. Moreover, she/he employs the same authentication strategy (i.e., pre-shared key or 802.1X authentication) as the authorized AP but always lets connecting users pass the authentication. Note that the rogue AP and the authorized AP could be both active at the same time. In this case, as the current AP selection mechanisms use signal strength as the only selection criteria, the user could be still attracted to connect to the rogue one if it has higher signal strength. According to our experiment, if two overlapped WLANs use the same SSID, the mobile device would just show the stronger one in its WLAN list. Once a user

logins into the rogue AP, the adversary is assumed to launch various man-in-the-middle attacks to eavesdrop, collect, and analyze any amount of data without being detected.

WiFi Freeloading: An adversary steals the credential of an authorized user to login to a private WLAN and then does something evil in the name of the legitimate user. Here, the credential might be either a simple password if the WLAN adopts WPA-PSK mode, or a username-password pair if the WLAN adopts WPA-Enterprise mode. The adversary could obtain such credentials through various means. For instance, recently, there are many crowdsourced apps like WiFi Companion for sharing users' WiFi login credentials. When an authorized user of a private WLAN installs such apps, her/his credentials would be stealthily collected and shared with others, certainly including attackers as well.

As we mentioned earlier, both these threats are due to the loose authentication requirement between mobile devices and APs. This paper aims to propose a non-cryptographic solution to significantly enhance these schemes and effectively detects the above threats. We next present our key technique.

IV. Methodology

Our solution tries to estimate the frequency offset (i.e. CFO, which have been introduced in Sec. 2) of a device as its fingerprint to perform attack detection. As mentioned in Sec. 2, CFO is due to the carrier oscillator drift in the WiFi network card, which, in theory, remains consistent over time and locations but varies across devices. Moreover, it's hard to spoof as it's a pure hardware feature and can hardly be affected by any software running on the application processor. As a result, it's an ideal feature for device fingerprinting. Nevertheless, existing solutions [3], [6] to estimate CFO require additional hardware, which prevents them from using in practice.

As section II introduces, the phase fields of CSI measurements capture the accumulated phase offset during the signal propagation. As CFO will result in a phase offset at the receiver, it is straightforward to consider if we could estimate CFO from CSI measurements. If the answer is positive, it means CFO could be estimated without using additional hardware because CSI could be obtained by upper layer application on off-the-shelf wireless devices by just modifying NIC drivers. Nevertheless, we find this task is non-trivial as the recorded phase offsets in CSI are contributed by many other factors besides CFO. In order to precisely estimate CFO, we have to first filter out these unrelated factors. So, below, before presenting the proposed method to estimate CFO from CSI, we first analyze the composition of each phase value in CSI.

A. Decomposing phase values of CSI measurements

Suppose that the fingerprinter has received n frames from the target device, and, for each frame, it has acquired the CSI measurements from the NIC driver. Let us consider the CSI measurement between a specific Tx-Rx pair for the frame at time t. As we mentioned in Sec. 2, for the kth subcarrier, this measurement contains a phase filed $\phi_{t,k}$, which records the measured phase offset of the frame between the transmitter and the receiver at this subcarrier. According to [8], [9], $\phi_{t,k}$ is added up by four offsets due to different factors:

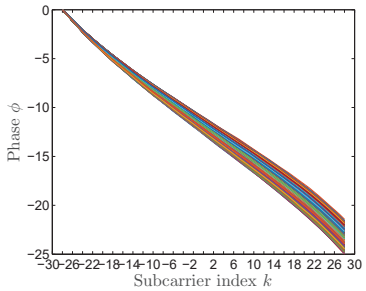


Fig. 1: Integrated phase shifts due to SFO, ToF and FDD over time (Each color curve corresponds to a frame.)

$$\phi_{t,k} = \omega_{t,k} + \theta_{t,k} + \psi_{t,k} + \varphi_t, \tag{1}$$

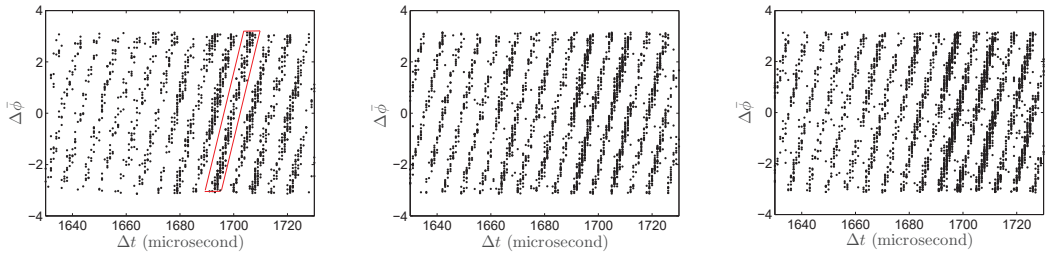
where φ_t is the phase drift due to CFO. The other three components are caused by the following factors:

$\omega_{t,k}$: the phase shift due to the *frame detection delay* (FDD). When a frame arrives at the receiver, it takes some time for the receiver to detect it, which would cause a time delay τ_d in the CSI measuring. Such a time delay inevitably causes a phase shift $\omega_{t,k}$ which is proportional to the frequency. Since the subcarrier index is also proportional to the frequency, it could be mathematically written as $\omega_{t,k} = 2\pi\alpha k\zeta_d$, where α is a constant coefficient, k is subcarrier index, and ζ_d is a value highly related to τ_d. Since ζ_d might vary with time, $\omega_{t,k}$ also varies across frames.

$\theta_{t,k}$: the phase shift due to *Sample Frequency Offset* (SFO). SFO is caused by the out-of-sync of the sample clocks between the transmitter and receiver. Similar to FDD, such out-of-sync also introduces a time delay τ_s, and further causes a phase shift that linearly increases with the subcarrier index. So, we have $\theta_{t,k} = 2\pi\beta k\zeta_s$, where β is constant, k is the subcarrier index, and ζ_s is determined by τ_s.

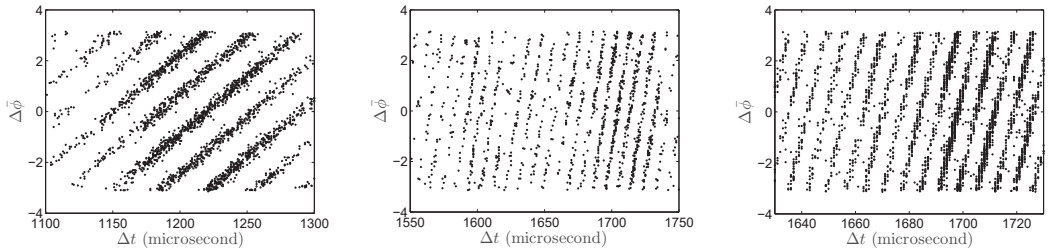
$\psi_{t,k}$: the phase shift due to the *time of flight* (ToF). ToF represents the time for the signal flies from the transmitter to the receiver, which certainly introduces another phase offset. In the absence of multipath, we have $\psi_{t,k} = 2\pi f_k t_p$, where t_p is the line-of-sight(LoS) propagation time from the transmitter to the receiver and f_k is the frequency of the kth subcarrier. However, if considering the multipath effect, there is another item, which is related to the environment, should be included in $\psi_{t,k}$. Since this offset is mainly determined by the time of flight, it is extremely useful in the field of indoor localization. In fact, there have been some work [10], [11] studying how to sanitize the CSI phase domain to perform indoor location. Nevertheless, because our aim in this paper is to extract the CFO component rather than the ToF component, we cannot directly use their phase sanitization algorithms. Now, based on the analysis above, Equation (1) can be rewritten as

$$\phi_{t,k} = k \cdot (2\pi\alpha\zeta_d \ + 2\pi\beta\zeta_s) + \psi_{t,k} + 2\pi\Delta f_c t, \tag{2}$$



(a) Place l_1, Time t_1 (b) Place l_1, Time t_2 (c) Place l_2, Time t_1

Fig. 2: The relationship of $\Delta\bar{\phi}$-Δt for the same device at different time and different places, where the stripe slope is strongly correlated to CFO.



(a) Device 1 (b) Device 2 (c) Device 3

Fig. 3: The relationship of $\Delta\bar{\phi}$-Δt for different devices with the same environment

where the CFO component φ_t have been replaced by $2\pi\Delta f_c t$ as we discussed in section II.

Among four components, we are only interested in the last one as our goal is to estimate Δf_c. Obviously, if the impacts of the first three components were consistent across frames, we could easily eliminate them by subtracting the measurements of the same subcarrier of two frames. We thereby conduct an experiment to validate whether this hypothesis is established. We collected the CSI measurements of 5000 frames in total. For each frame, we subtracted the phase value of the first subcarrier from those of other subcarriers. By doing so, the phase shift caused by CFO is eliminated as this component is independent with the subcarrier index and remains the same across subcarriers. We then plot the resulted phase values of each subcarrier of all the 5000 frames in the same Phase-Index plot. In particular, the points corresponding to the same frame use the same color and are connected to form a curve. Obviously, if the above assumption is true, the curves of different colors would coincide with each other. Unfortunately, our result in Fig. 1 shows that these curves do not overlap, which indicates that the integrated phase shifts caused by SFO, ToF and FDD are time-varying. Therefore, we have to find a more complicated way to eliminate them.

B. Estimating CFO From CSI

We now describe our technique to estimate CFO from those noisy phase fields of CSI measurements. To make readers easier to follow, we first make a quick overview of our proposal, and then go into details to elaborate why it can eliminate the undesired impacts due to FDD, SFO and ToF.

Our proposal first defines and computes a new phase variable $\bar{\phi}_t = \frac{\phi_{t,-1}+\phi_{t,1}}{2}$ for each frame at time t, where denoting by $\phi_{t,-1}$ and $\phi_{t,1}$ the measured phase values of subcarrier -1 and 1, respectively. Then, for every pair of adjacent frames, it calculates their phase difference $\Delta\bar{\phi}$ (mapped into $[-\pi, \pi]$) and TDoA (Time Difference of Arrival) Δt in microseconds, respectively. Afterwards, it plots all the points $(\Delta\bar{\phi}, \Delta t)$ in the X-Y plane as Fig. 2a shows. As you can observe, these points form a serials of periodical stripes, which are marked by red boxes in Fig. 2a. Moreover, stripes are tilted and seem to have the same slope. Our proposal leverages the method to be introduced in the next section to estimate the slope of these stripes, and finally outputs it as the estimated CFO.

As one example, we present the stripe figures of the same device in Fig. 2 when the time of testing and the position of target device are changed. The results show that the stripe slope remains fairly consistent in different scenarios. We also compare such stripe figures for different target devices in Fig. 3, and find that the stripe slope varies significantly. In summary, from these figures, it seems that the slope of stripes is stable across environment but varies significantly across devices, which is rightly the desired feature of CFO for fingerprinting. We below show that such stripe slope can be regarded as a perfect estimation of CFO.

FDD and SFO removal. Both FDD and SFO cause a time delay in CSI measuring. As we discussed earlier, such time

delays would result in phase shifts which linearly increase with the subcarrier index. According to equation (2), if we sum ϕ_{t,k_1} and ϕ_{t,k_2} where k_1 and k_2 satisfying $k_1 + k_2 = 0$, we can remove the phase shifts due to FDD and SFO. In 802.11n standard [**?**], the CSI measurements record data for 30 subcarriers with indices in [−28,−26,···,−2,−1,1,3,···, 27,28] when the bandwidth is 20 Mhz. There are only two pairs, $[-1, 1]$ and $[-28, 28]$ can meet the requirement. That is why we define the new phase variable $\bar{\phi}_t = \frac{\phi_{t,-1}+\phi_{t,1}}{2}$ in our approach. In fact, for a time point t, $\bar{\phi}_t$ can be expressed as

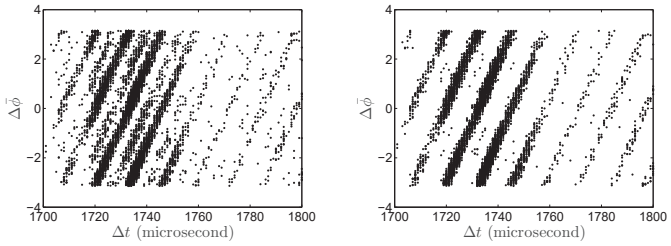
$$\begin{aligned}\bar{\phi}_t &= 2\pi\Delta f_c t + \frac{\psi_{t,-1}+\psi_{t,1}}{2} + (-1+1)\cdot(2\pi\alpha\zeta_d \ + 2\pi\beta\zeta_s) \\ &= 2\pi\Delta f_c t + \frac{\psi_{t,-1}+\psi_{t,1}}{2}.\end{aligned} \tag{3}$$

Fig. 4 compares the stripe figure using the original phase values of subcarrier -28 with the stripe figure using the average phase values of subcarrier -1 and 1. We observe that removing FDD and SFO phase shifts make the stripe figure much clearer.

ToF removal. The best way to removal the ToF phase is to obtain the ToF itself. In fact, there does exist some work [12], [13] that studies how to gain the ToF from CSI measurements. Nevertheless, their methods usually have to scan the entire frequency band in 802.11n, which need the cooperation between the transmitter and receiver. Such cooperation is impractical in our scenario because the device to be fingerprinted might cheat during this process. In our approach, instead of trying to obtain the ToF directly, we take some measures to restrict the variation of ToFs. As you know, the variation of ToF is mainly due to two reasons. The first one is the changing of the relative position between the receiver and the transmitter, and the second one is the changing of the surrounding environment, e.g., there are persons walking around. Thus, we first require both the receiver and the transmitter to keep stationary during the frame collection, which fixes their relative position. Note that this requirement would not significantly affect the user experience because the fingerprinting process could finish within 10 seconds according to our experiments. Second, our approach only selects adjacent frames to calculate the phase differences. This measure is motivated by the fact that the time interval between most adjacent frames is usually less than 3ms, which is so short that the environment could be considered to be static. However, there do exist some adjacent frames whose time intervals are long. Sen et al. [10] proposes a CSI phase sanitization scheme and demonstrate that the transformed phases are static when the environment is stable. The effect of these measures is illustrated by Fig. 5. Therefore, by taking these measures, we can assume the ToF between two adjacent frames is constant, which indicates that when we calculate the phase difference $\Delta\bar{\phi}$, the phase shift caused by ToF can be eliminated as well.

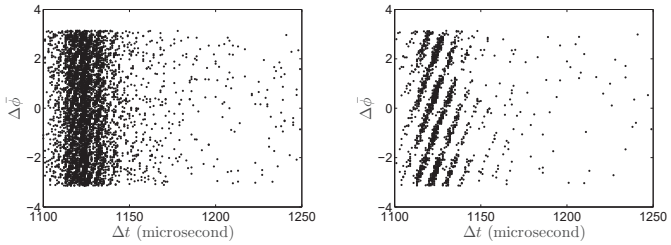
Now, we can rewrite $\Delta\bar{\phi}$ as $\Delta\bar{\phi} = \bar{\phi}_{t_1} - \bar{\phi}_{t_2} = 2\pi\Delta f_c\Delta t$. It's easy to see that Δf_c is just the slope of the linear function between $\Delta\bar{\phi}$ and Δt. That is, in theory the points in those stripe figures should form periodical lines instead of stripes.

The gap between the theory and the practice is attributed



(a) The stripe figure extracted from sub-carrier -28 (b) The stripe figure using the new phase definition $\bar{\phi}_t$

Fig. 4: Effects of our measures to remove FDD and SFO



(a) The stripe figure using all the adjacent frame pairs (b) The stripe figure using adjacent frame pairs selected based on phase sanitization

Fig. 5: Effect of selecting adjacent frames based on phase sanitization

to the various measurement errors and a known firmware issue of Intel 5300 NIC according to [12] and [14]. The measurement errors make the stripes not a smooth line but have some noise points. For instance, if the clock frequency of a device is 10Mhz, then the upper-limit precision to obtain the exact arrival time of a frame at the receiving antenna is $0.1\mu s$. Therefore the interval of any points pair in the stripes figure is $0.1\mu s$ at least in that case. The firmware issue results extra stripes in the figures which makes the $\Delta\bar{\phi}$ and Δt looks that doesn't meet a function relationship. But fortunately, such extra stripes won't affect the slope estimation. And for a purpose of attack detection, it's fine to use a CFO related quantity rather than CFO itself as the fingerprint. In other words, our task is transformed from estimating CFO to estimating the stripe slope from the noisy stripe figure drawn based on CSI measurements, which do not require additional hardware. We will delay our solution to the next section.

V. Implementation

In this section, we introduce the technical details to implement our proposal, especially the approach to estimate the stripe slopes from those noisy figures.

Specifically, when a mobile device (i.e., a laptop or smartphone) wants to fingerprint an AP, we make it first connect to the AP as usual. Then, it leverages the built-in tool, Ping, to send testing data to the AP and collects the CSI measurements of all the response frames. To guarantee the high precision of the obtained fingerprint, we need about 5000 frames. Since the transmission rate could reach 11Mbps for 802.11b, this process

usually takes fewer than 10 seconds. Based on these CSI measurements, we derive the stripe figure described in the last section and then estimate the slope of those stripes contained as the final device fingerprint. The process to fingerprint a mobile device is completely the same except that we have to configure a WiFi P2P connection between the two devices.

Now, we will explain the details of how to estimate CFO in the CFO stripe figure. There are two phases involved : stripe extraction and slope estimation.

A. Stripe Extraction

To estimate the slope, we need first obtain the point sets which constitute each stripe. Our process to extract the slopes can be summarized as three steps which are shown in Fig. 6.

Selecting high dense region. The distribution of the TDoAs of adjacent frames is mainly dominated by the transmission rate of the network, which usually fluctuates within a certain range. As a result, as we show in the first sub-figure in Fig. 6, some interval on Δt-axis may include more points than others which form a high dense region in the stripe figure. It is obvious that the more points a stripe contains, the more accurate the liner fitting will be. So, our first step is to use a sliding window algorithm on Δt-axis to identify such high dense region. In each window of fixed length, we count the total number of points, and then move to the next by sliding exactly one window length. Finally, we select the window with the highest number of points as the high dense region, and all the following processes are applied to this window. The window size is empirically determined by manually checking the stripe figures of thirty devices. Suppose that the average span of stripes on Δt-axis is s. Then, the window length is set to $6s$ in our experiment, which could guarantee that the extracted region contains at least three clear stripes.

Converting to a binary image. Looking at the second sub-figure in Fig. 6, we can observe that most points are concentrated along the central lines of stripes while there are still some outliers sparsely located between different stripes. Obviously, such outliers would bring significant side effects to the slope estimation, and thus have to be removed. For this purpose, our method transforms the high dense region identified in the last step into a binary image, which exploits the great density difference between normal and abnormal points to eliminate the latter ones. In particular, we first divide the region into small rectangles of equal size that each corresponds to a pixel in the newly generated binary image. Then, for each rectangle, we count the insider points. If the total number exceeds a pre-defined threshold, the corresponding pixel in the binary image is set to 1; otherwise 0. By doing so, as we show in the third sub-figure in Fig. 6, most outliers have been removed. For a few of images still containing a large number noises, we can employ some existing image processing schemes such as ALM (Augmented Lagrange Multipliers)-based PCA [15] to further reduce the noises.

Obtaining connected component. Although by now we have obtained a much clearer strip image by taking the above measures, there still exist some noises. Fortunately, we can observe that the normal points now form some connected components which are much longer than those formed by noises. What's more, it seems there is strict one-to-one correspondence between the stripes and the long connected components. We thus use the algorithm described in [16] to identify the k longest connected components in a 2D binary image and then transform the task to estimate the slope of stripes to estimate the slope of these connect components. Of course, in practice, we might get some wrong components that do not correspond to any stripe. Therefore, we need a more slope estimation mechanism as we show below.

B. Slope Estimation

After the stripe extraction process, we have gotten the point sets for each long connected component. Let the point set forming a specific component c be S. Its slope $\widehat{k_c}$ can be obtained by using the least square method as

$$\widehat{k_c} = \arg\min_{\rho} \sum_{(y,x)\in S} (y + 2\pi x\rho + \omega)^2, \tag{4}$$

where ω is a constant. So, we pick out each connected components and calculate its slope k with this method.

However, the slopes obtained may vary since the connected components may be not exactly the stripe that we desire. In order to solve this problem, we take a voting strategy on those slopes. First, we divide the slopes into different intervals where the length is empirically initialized to 0.1 and increased by 0.01 each time until there is at least one interval contains more than 10% of the obtained slopes. It then counts the number of slopes fall into each interval. Finally the average value of slopes in the densest interval is used as the end result. Since the slopes are a series of real numbers and the number of slopes is small, such process can be fast. The motivation of vote is based on a simple fact that the right values are usually close however the wrong values trend to be varied.

VI. Experiments

In this section, we introduce our experiments for evaluating the real performance of the proposed fingerprint mechanism.

In our experiments, we use a Thinkpad X200 laptop as a fingerprintor to collect CFO fingerprints of testing devices (both APs and smartphones). This laptop is equipped with an Intel 5300 WiFi Card, of which the driver has been modified based on the tool introduced in [7] to report timely CSI measurements for each received frame. In order to fingerprint an AP, we let the laptop connect to it and send it Ping messages. For each response frame, the laptop obtains and stores the corresponding CSI measurements. The sending interval between two Ping messages is 1s by default, which is too long for our fingerprinting purpose. Hence, we use the $-i$ parameter to shorten this interval to 0.002s, which would produce 500 frames every second. For each device, we collect 5000 response frames in total, which usually takes about 10s. The laptop then uses the proposed approach to estimate the CFO as the device's fingerprint. To fingerprint a smartphone, we configure the same laptop to work as a WiFI hotspot.

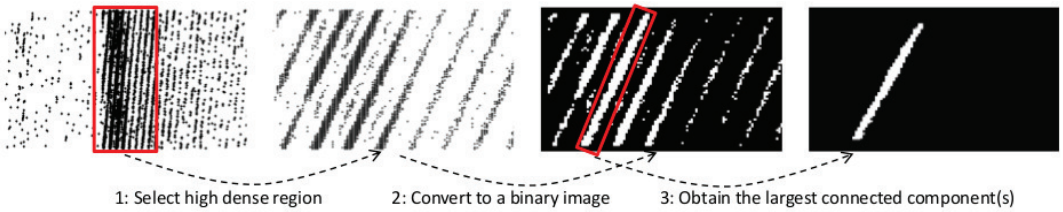


Fig. 6: The main steps to extract stripes from the noisy figure

The phone is then made to connect to the laptop and send it messages the same as the laptop does in AP fingerprinting. The laptop fingerprints the phone based on the collected CSI measurements of the frames received.

We mainly focus on two performance indicators of our proposal. The first one is the stability of the fingerprints (i.e., the estimated CFO) of devices over time and across locations. The second one is the accuracy of our mechanism, including the detection rates of rouge APs and smartphones, and false alarm rates. Our experiments cover 23 smartphones and 34 APs, including both the same and different models. We now present the details and results of our experiments.

A. Stability

The fingerprints in our proposal are CFOs estimated from CSI measurements, which are collected from Commercial Off-The-Shelf(COTS) devices. As much research work [17], [18] mentions, the amplitude domain of CSI is environment sensitive, and even the phase domain can be also affected by the environment changes as we discussed in section IV. Thus, although in theory CFO is considered to be consistent when the relative motions between devices could be ignored in the indoor environments, we still conducted extensive experiments to verify whether CFO is really stable in practice such that can be used as a long-term fingerprint. Our verifications are focused on two aspects: time stability and location stability.

Time Stability: For this aspect, we want to verify whether CFO is consistent over time. Our experiments consider two time scales: one day and one month. Fig. 7 shows the estimated CFO of two APs, NETGEAR R7000 and TP-LINK WDR4300, at different times in one day. Here, we let both APs keep working all day, and record the CSI every 6 hours. We can observe that the CFO estimations of both APs are fairly consistent across different times of the whole day.

To verify the stability of CFO in a long-term, we collected CSI from a smartphone (Xiaomi Note) at different days in one month. As shown in Fig. 8, the CFO estimations in the month are almost the same and the relative differences are below 0.1.

Location Stability: As we mentioned earlier, the phase domain of CSI includes a ToF shift which is highly sensitive to the relative positions of devices and the surrounding environments. To bring in the complex multipath effects, we chose an indoor environment, which is a lab room of $7.7m \times 6.5m$, to verify the location stability of the CFO fingerprints of four APs. For each AP, we put it at four different locations, one

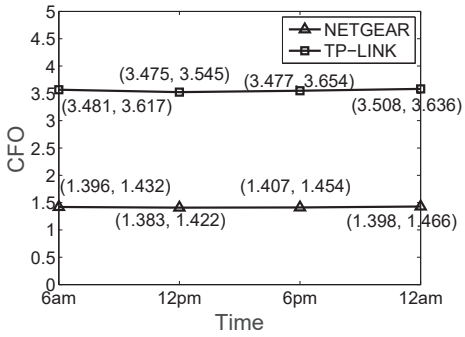


Fig. 7: Estimated CFOs at different times in one day for NETGEAR R7000 and TP-LINK WDR4300, where the value pair in each parentheses denotes the minimum and maximum values among the 15 measurements

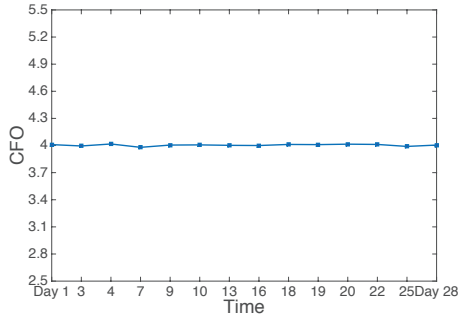


Fig. 8: The estimated CFO from a smartphone, Xiaomi Note, in one month

of which is separated by the wall from the laptop. During each test, we asked two persons to walk around the room to mimic the real environments that are always changing. We performed 15 times of fingerprintings for every AP at each location. The averaged CFO estimations (as well as the minimums and maximums) for these APs at different locations are presented in Fig. 9. It shows that the changes of the CFO estimations due to location changes for the same AP can be neglected compared with the differences between different APs, which demonstrates that our CFO estimations are stable across locations. We conducted similar experiments for four different smartphones and obtained the similar result.

B. Attack Detection Accuracy

To evaluate the accuracy of our proposal, we conducted a serials of experiments to test its detection rate and false

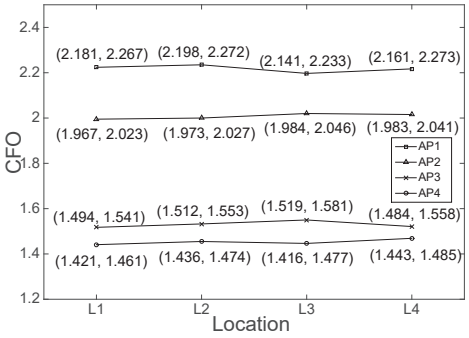


Fig. 9: Estimation of APs at different locations in the lab

alarm rate. To make the results more persuasive, we fingerprint each AP/phone K times, and store all the K fingerprints in a database. Suppose that there are N devices, which indicates we collect $K \cdot N$ fingerprints in total. In each test, we sample a set H_d by randomly pick M of K fingerprints for each device d. Then build a whitelist W by computing the mean value of H_d. The M in our experiment is set to one third of K, which is 5 for $K = 15$ and 2 for $K = 7$. Let S denote the set of remained fingerprints. Then, if comparing one fingerprint in S to the fingerprint of a different device in W, we are actually simulating to detect a Rogue AP or a freeloading attack. On the other hand, if comparing a fingerprint in S to the fingerprint of the same device in W, we are actually simulating the normal communication. So, we can define the detection rate P_d of attacks and the false alarm rate P_f as

$$P_d = \frac{\sum_{i \in S} \sum_{j \in W} [id(i) \neq id(j)] \cdot \text{match}(i,j)}{(K-M) \times (N-1) \times N} \tag{5}$$

and

$$P_f = \frac{\sum_{i \in S} \sum_{j \in W} [id(i) = id(j)] \cdot \text{match}(i,j)}{(K-M) \times N}, \tag{6}$$

where $id(i)$ is the device index of fingerprint i, and $match(i,j)$ equals 1 if Fingerprint i matches Fingerprint j; otherwise 0. The match process calculate the greatest angle difference of $H_{id(i)}$ as T_h and the absolute angle difference between two slopes as d_a. Then decide whether the two fingerprints is matched by comparing d_a to a threshold which is set to the larger of 1° and T_h in our experiments.

We first consider our evaluation for APs. We conducted experiments in 4 different scenarios including a teach-building, a lab room, the university library, and several Starbucks in the downtown and collected 382 fingerprints in total. The details of APs in these environments are shown in Table I. In Starbucks, we did not see the devices so that their brand is unknown. For other scenarios, all the APs share the same brand and model. Note that this actually simulates the worst case where the attacker sets up a rogue AP with the same model as that of authorized APs. Intuitively, devices from the same vendor should have closer fingerprints. According to the results in Table II, the detection rates in all 4 scenarios exceed 94%, while the false alarm rates are below 5.2%.

TABLE I: detailed scenarios of ap experiments

Scenario	AP Brand	Total Fingerprints	# of APs	Value of K
Teaching Building	HUAWEI	112	16	7
Starbucks	Unknown	90	6	15
Lab room	NETGEAR, TP-LINK	120	8	15
Library	HUAWEI	60	4	15

TABLE II: accuracy of ap in different scenes

Scene	Detection Rate P_d	False Alarm Rate P_f
Teaching building	94.07%	4.76%
Starbucks	95.05%	2.31%
Lab room	97.24%	1.47 %
University Library	94.37%	5.11 %

Besides APs, we also conducted experiments to evaluate the accuracy for fingerprinting smartphones. Our experiments cover 23 smarpthones from various brands as shown in Table III where $K = 15$ for all phones. The definitions of detection rates and false alarm rates are similar to those defined for APs. The average detection rate can achieve 94%, while the false alarm rate is just below 3%.

We also do the experiment on the same model phones as illustrated in Table IV. Since the dataset is small, we express P_{df} and P_f as fractions. We can find that our proposal can still well distinguish between devices of the same model.

TABLE III: detail information of phones

Smartphone	Quantity
Meizu	3
Samsung	5
Xiaomi	7
Others	8

TABLE IV: accuracy of same model phones

Smartphone	Quantity	Detection Rate	False Alarm Rate
Xiaomi with the same WiFi NIC	6	149/160	1/70
Samsung S4	3	70/70	0/35

VII. Related Work

The weakness of the existing authentication in 802.11 wireless network has raised many security issues. To fill such security holes, there are some device fingerprint based approaches having been proposed. Unlike traditional cryptographic solutions, the device fingerprint approach use the device-related information to generate the unique fingerprint and such fingerprint can be used to distinguish among devices which preventd the identity spoof in the wireless network.

A wide variety of features have been used as the fingerprint and the different extracting methods lead to different approaches. In [19], the authors use the packet inter-arrival time(IAT) as the feature. In [20], the data rate information in the PHY-layer frame header is used as a feature. In [21], the authors employ a set of wireless parameters to fingerprint a target devices including the inter-arrival time, frame size, transmission rate and etc. Those features are both related to network traffic. A better way to do this is to use a radiometric feature since it is highly related to the device itself.

As an example of RF feature, Hall et al. [22] find that it's possible to use the transient signal to fingerprint wireless device. In [3], a set of radiometric feature are extracted to as the signatures of device, including the phase error, I/Q offset and CFO. The main different between our work from [3] is that we didn't need any additional hardware, which makes our scheme more convenient to deploy on existing devices.

Jana and Kasera [2] leverage the clock skew to fingerprint the device. Despite the clock skew is not a RF feature, it still a device-correlated signature and can achieve a high accuracy. Unfortunately, since the time stamp only exists in the beacon/probe response frame, their scheme can't do a mutual fingerprinting and have no way to detect freeloading attacks.

We notice that there is some work [23] using CSI to identify different users with a high accuracy. However, this scheme, strictly speaking, is not to fingerprint the device, but to fingerprint the channel, which is an essentially different problem. Since CSI usually varies with locations, it seems no way to use CSI as a device fingerprint. Similarly, [24] can also be categorized as channel fingerprinting.

In [25], CFO is elegantly used to implement privacy preserving location verification in LBS system. In [26], the authors also use the CFO as the device fingerprint. However, their scheme requires to employ additional equipments, i.e., the USRP2, to obtain the CFO.

VIII. Conclusion

This paper has demonstrated that CFO can be used as long-term device fingerprints to significantly enhance the current cryptographic authentication mechanism between mobile devices and APs in WLAN. Our experiments show that such fingerprints are fairly consistent over time and locations but vary across devices. As a result, we can use them as extra identities besides MAC addresses to identify devices and further detect rouge APs and users. We solve the challenge to estimate CFO precisely from CSI measurements, which could be obtained by upper-layer applications without using any additional hardware. Our extensive experiments on real APs and smartphones show that our approach could achieve a high detection rate but produce very few false alarms.

References

[1] R. Beyah and A. Venkataraman, "Rogue-access-point detection: Challenges, solutions, and future directions," *IEEE Security and Privacy*, vol. 9, no. 5, pp. 56–61, 2011.

[2] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews." in *Proc. of the 14th ACM international conference on Mobile computing and networking (MobiCom'08)*, 2008, pp. 104–115.

[3] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proc. of the 14th ACM international conference on Mobile computing and networking (MobiCom'08)*, 2008, pp. 116–127.

[4] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," in *Proc. of 2005 IEEE Symposium on Security and Privacy (S&P'05)*, 2005, pp. 211–225.

[5] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proc. of the 3rd ACM conference on Wireless network security (WiSec'10)*, 2010, pp. 169–174.

[6] N. T. Nguyen, G. Zheng, Z. Han, and R. Zheng, "Device fingerprinting to enhance wireless security using nonparametric bayesian method." in *Proc. of 2011 IEEE INFOCOM*, 2011, pp. 1404–1412.

[7] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *Proc. of the ACM SIGCOMM 2010 conference (SIGCOMM'10)*, 2010, pp. 159–170.

[8] J. K. Tan, "An adaptive orthogonal frequency division multiplexing baseband modem for wideband wireless channels," Ph.D. dissertation, Citeseer, 2006.

[9] M. Speth, S. A. Fechtel, G. Fock, and H. Meyr, "Optimum receiver design for wireless broad-band systems using ofdm. part i," *IEEE Transactions on Communications*, vol. 47, no. 11, pp. 1668–1677, 1999.

[10] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "You are facing the mona lisa: spot localization using phy layer information," in *Proc. of the 10th international conference on Mobile systems, applications, and services (MobiSys'12)*, 2012, pp. 183–196.

[11] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *Proc. of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM'15)*, 2015, pp. 269–282.

[12] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-level localization with a single wifi access point," in *Proc. of 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI'16)*, 2016, pp. 165–178.

[13] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity wifi," in *Proc. of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom'15)*, 2015, pp. 53–64.

[14] J. Gjengset, J. Xiong, G. McPhillips, and K. Jamieson, "Phaser:enabling phased array signal processing on commodity wifi access points," in *Proc. of the 20th annual international conference on Mobile computing and networking (MobiCom'14)*, 2014, pp. 153–164.

[15] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[16] R. M. Haralock and L. G. Shapiro, *Computer and robot vision*. Addison-Wesley Longman Publishing Co., Inc., 1991.

[17] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures," in *Proc. of the 20th annual international conference on Mobile computing and networking (MobiCom'14)*, 2014, pp. 617–628.

[18] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. Ni, "We can hear you with wi-fi!" in *Proc. of the 20th annual international conference on Mobile computing and networking (MobiCom'14)*, 2014, pp. 593–604.

[19] K. Gao, C. Corbett, and R. Beyah, "A passive approach to wireless device fingerprinting," in *Proc. of IEEE/IFIP International Conference on Dependable Systems Networks (DSN'10)*, 2010, pp. 383–392.

[20] C. L. Corbett, R. A. Beyah, and J. A. Copeland, "Passive classification of wireless nics during active scanning," *International Journal of Information Security*, vol. 7, no. 5, pp. 335–348, 2008.

[21] C. Neumann, O. Heen, and S. Onno, "An empirical study of passive 802.11 device fingerprinting," in *Proc. of the 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW'12)*, 2012, pp. 593–602.

[22] J. Hall, M. Barbeau, and E. Kranakis, "Enhancing intrusion detection in wireless networks using radio frequency fingerprinting," in *Proc. of the 3rd IASTED International Conference on Communications, Internet and Information Technology (CIIT'04)*, 2004, pp. 201–206.

[23] H. Liu, Y. Wang, J. Liu, J. Yang, and Y. Chen, "Practical user authentication leveraging channel state information (csi)," in *Proc. of the 9th ACM symposium on Information, computer and communications security (ASIACCS'14)*, 2014, pp. 389–400.

[24] J. Xiong and K. Jamieson, "Securearray: Improving wifi security with fine-grained physical-layer information," in *Proc. of 2013 IEEE INFOCOM*, 2013, pp. 441–452.

[25] W. Wang, Y. Chen, and Q. Zhang, "Privacy-preserving location authentication in wi-fi networks using fine-grained physical layer signatures," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1218–1225, 2016.

[26] N. T. Nguyen, G. Zheng, Z. Han, and R. Zheng, "Device fingerprinting to enhance wireless security using nonparametric bayesian method," in *Proc. of 2011 IEEE INFOCOM*, 2011, pp. 1404–1412.