Fingerprinting Wi-Fi Devices Using Software Defined Radios

Tien D. Vo-Huu Triet D. Vo-Huu Guevara Noubir

College of Computer and Information Science
Northeastern University
Boston, MA 02115
{tienvh|vohuudtr|noubir}@ccs.neu.edu

ABSTRACT

Wi-Fi (IEEE 802.11), is emerging as the primary medium for wireless Internet access. Cellular carriers are increasingly offloading their traffic to Wi-Fi Access Points to overcome capacity challenges, limited RF spectrum availability, cost of deployment, and keep up with the traffic demands driven by user generated content. The ubiquity of Wi-Fi and its emergence as a universal wireless interface makes it the perfect tracking device. The Wi-Fi offloading trend provides ample opportunities for adversaries to collect samples (e.g., Wi-Fi probes) and track the mobility patterns and location of users. In this work, we show that RF fingerprinting of Wi-Fi devices is feasible using commodity software defined radio platforms. We developed a framework for reproducible RF fingerprinting analysis of Wi-Fi cards. We developed a set of techniques for distinguishing Wi-Fi cards, most are unique to the IEEE802.11a/g/p standard, including scrambling seed pattern, carrier frequency offset, sampling frequency offset, transient ramp-up/down periods, and a symmetric Kullback-Liebler divergence-based separation technique. We evaluated the performance of our techniques over a set of 93 Wi-Fi devices spanning 13 models of cards. In order to assess the potential of the proposed techniques on similar devices, we used 3 sets of 26 Wi-Fi devices of identical model. Our results, indicate that it is easy to distinguish between models with a success rate of 95%. It is also possible to uniquely identify a device with 47% success rate if the samples are collected within a 10s interval of time.

1. INTRODUCTION

Wi-Fi is emerging as the primary medium for wireless Internet access. Cellular carriers are increasingly offloading their traffic to Wi-Fi Access Points (APs) to overcome capacity challenges, limited RF spectrum availability, cost of deployment, and keep up with the traffic demands driven by user generated content. Wi-Fi offloading is facilitated by 3GPP standards for Non-3GPP Access Networks Discovery and Roaming [1], IETF seamless USIM-based strong authentication and secure communication protocols such as EAP-SIM/AKA [10, 13]. Studies forecast a sustained 50% yearly growth in Wi-Fi offloading for many years to come [22, 23, 25]. This trend is paved by the increasing deployments of Hotspot 2.0 (HS2) Access Points enabled by seamless handover across networks implementing the IEEE 802.11u amendment [15]. Moreover, manufacturers of laptops and streaming devices, such as the Apple MacBook Pro and the Roku streaming player, are removing Ethernet ports and entirely relying on Wi-Fi, and several new variants of Wi-Fi are being developed to suit different environments (e.g., IEEE 802.11p for vehicular networking and IEEE 802.11af for TV white spaces).

The ubiquity of Wi-Fi and its emergence as a universal wireless interface makes it the perfect tracking device. The Wi-Fi offloading trend provides ample opportunities for adversaries to collect samples (e.g., Wi-Fi probes) and track the mobility patterns and location of users. The simplest way of tracking users consists of extracting the MAC address of probe packets periodically transmitted by Wi-Fi cards. This is known to be exploited by government agencies, marketing companies, and location analytics firms. In shopping malls for instance, companies such as Euclid Analytics state on their website that they collect "the presence of the device, its signal strength, its manufacturer (Apple, Samsung, etc.), and a unique identifier known as its Media Access Control (MAC) address." [8] to analyze traffic patterns of users over large spatio-temporal durations of time. Another example is by startup Renew, which installed a large number of recycling bins in London with capability to track users. This allows Renew to identify if the person walking by is the same one from yesterday, even her specific route, walking speed [6, 27]. While information about the activities of data analytics and advertizing firms is public by the nature of their business, little is known about governments and cybercriminals Wi-Fi surveillance programs. The threats to privacy exploiting MAC address tracking triggered Apple to include a MAC address randomization feature in its iOS 8 release, receiving significant praise from privacy advocates [14]. Unfortunately, MAC address randomization is only the simplest and easiest to mitigate tracking techniques. Recent attacks demonstrated that it is feasible to infer Android devices routes using zero-permission sensors (i.e., gyroscope, accelerometers, and magnetometer) [18]. Such attack however focuses on devices in vehicles moving along roads and requires the installation of an App. On the other end of the spectrum, an adversary can potentially track a wireless device based on its physical layer characteristics. Variations in the fabrication process of silicon devices result in intrinsic random physical features that can potentially uniquely identify a device in a way difficult to compensate for or clone. The unique characteristics of the physical layer in wireless devices have even been considered for authentication purposes leading to the area of Physically Unclonable Func-

WiSec'16 , July 18–20, 2016, Darmstadt, Germany

DOI: http://dx.doi.org/10.1145/2939918.2939936

tions (PUFs) [16, 19]. While researchers obtained mixed results with the use of physical charateristics of devices for authentication, the potential of fingerprinting for tracking devices is more serious. This is because, unlike in an authentication protocol where the authenticator needs a failure probability exponentially small in the security parameter, a tracking adversary only needs a reasonable probability to breach the privacy of users (e.g., if a given person is at home, office, cafe, entered a street). However, fingerprinting Wi-Fi devices is challenging. In this work, we develop a framework for investigating Wi-Fi devices fingerprinting, at various layers of the network stack and that enables reproducibility and analysis. Our goal is to provide both a solid theoretical and experimental foundation for understanding Wi-Fi devices fingerprinting. Our contributions can be summarized as follows:

- Our first step towards a methodical analysis of Wi-Fi fingerprinting was to develop a full implementation of a Wi-Fi stack for the popular Ettus USRP Software Defined Radio platform. Our platform can iteratively process RF signals and easily analyze a variety of features.
- Our platform extracts all the characteristics from the PHY, MAC, and Link layers that can be exploited for fingerprinting. For instance at the Physical layer, we extract the carrier frequency offset, the sampling frequency offset, transmitter turn on/off transients, scrambling seed.
- We analyzed the potential of each technique and their combination on a set of 93 devices spanning 13 different models including three sets of 26 cards each (from reputable manufacturers).
- We discovered new differentiating factors such as the common seed subsequences, the distributions of carrier and sampling frequency offsets through the Kullback-Leibler distance, and the envelop of frame transients.

In Section 2, we present how the key Wi-Fi features are extracted using our SDR receiver. Section 3, describes our device fingerprinting and classification techniques. Section 4, presents our testbed, evaluation methodology, and results. Finally, we summarize the related and conclusions.

2. FEATURE EXTRACTION

In order to characterize and fingerprint Wi-Fi devices, we develop our own Wi-Fi receiver [32] using Software Defined Radio [11] running on the popular Universal Software Radio Peripheral (USRP [7]). Our receiver is able to decode transmissions of rate up to $54\,\mathrm{Mbps}$[1]. Unlike with commercial Wi-Fi adapters, it is easy to extract physical characteristics of the Wi-Fi signal with our SDR receiver. To better explain and discuss the features that we use for fingerprinting, we first give a brief overview of the procedure of receiving Wi-Fi signal.

2.1 Wi-Fi Receiver using SDR

Our SDR Wi-Fi receiver block diagram is shown in Figure 1. The receiver consists of three main processing tasks: (1) baseband signal reception, (2) OFDM demodulation, and (3) data decoding. Except for the baseband signal reception handled in the USRP, the other two tasks are carried out on the host computer connected to the USRP.

[1]While there is an existing implementation of Wi-Fi receiver [2] on GNU Radio, it is not functional for rates beyond $18\,\mathrm{Mbps}$.

Table 1: Features extracted from our fingerprinting system.

Feature	Extracted from
Scrambling seed	Descrambler
Sampling frequency offset	Channel Estimator
Carrier frequency offset	OFDM Synchronizer
Frame transient	OFDM Synchronizer

OFDM demodulation. After the USRP captures the Wi-Fi signal on its RF front end, it produces a stream of digitized complex samples. On the host computer, we look for the Wi-Fi packets in the digital samples based on the repeated patterns in the preamble of the Physical frame. During this phase, the carrier frequency offset (CFO) is also estimated to correct the mismatched clock between the receiver and transmitter. After that, the samples are transformed into the frequency domain for OFDM demodulation. In order to decode QAM-modulated transmissions, we developed a pilot-based phase tracking technique combined with a decision directed estimation technique to estimate and equalize the channel on individual data subcarriers. Through this method, we obtain the sampling frequency offset (SFO) used in fingerprinting, and also demodulate the complex samples into a binary data sequence.

Packet decoding. A chain of three decoding blocks is applied on the binary data sequence produced by the OFDM demodulator. First, it is deinterleaved to scatter possible bit errors and allow the convolutional decoder to correct the errors and decode the data. Finally, the original frame is recovered by the descrambler.

2.2 Fingerprinting Overview

As our Wi-Fi receiver is SDR-based, we can easily allow the extraction of various physical characteristics of the Wi-Fi signal. While features such as constellation error vectors, channel state information, decoding metrics, or transmit spectrum have been used for fingerprinting and achieved good performance in previous work (e.g., [4, 5] and references therein), we found that in our experiments, those features are more affected by the environment than by the device imperfection. We conjecture that the production quality has been improved since then, making devices more identical. In this work, we instead focus on 4 features listed in Table 1 for fingerprinting.

The main idea of our fingerprinting system is as follows. We capture the transmitted frames in the form of a digital signal, and decode them to extract Physical and MAC information. We group the recorded digital samples of frames based on the MAC addresses and perform the classification/characterization for each group. The classification is done by computing the likelihood between features extracted from the samples and features belonging to known Wi-Fi devices. It is worth noting that while existing machine learning algorithms (e.g., SVM) can be used for classification, it remains unclear how to tune parameters for achieving good accuracy. In this work, we design our own classification algorithms specically applied to the chosen physical features.

2.3 Extracting Scrambling Seed

According to the IEEE 802.11a/g standard, a Wi-Fi transmitter shall generate a new random scrambling seed for every transmission of a Physical Layer frame. However, not all chipset manufacturers follow the standard. Recent work [30] discovered that for some chipset manufacturers, scrambling seeds are generated in a free-wheeling mode with some specific shift distances which are used to distinguish the device models. In general, however, reverse engineering the seed generating algorithm is challenging as chipset manufacturers do not disclose their design. In our work,

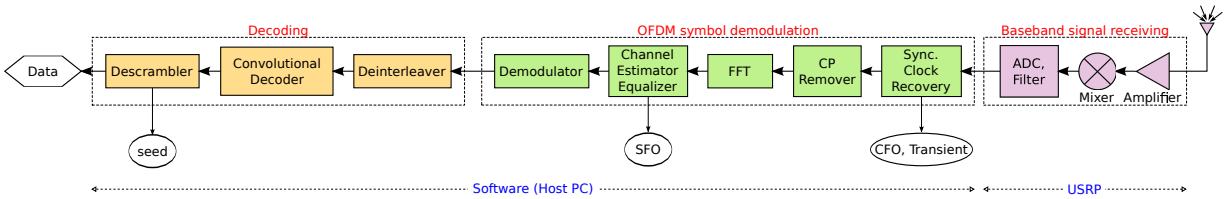


Figure 1: IEEE 802.11a/g SDR receiver block diagram with feature extraction capabilities.

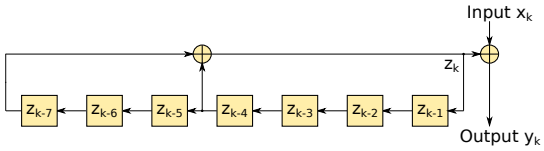


Figure 2: IEEE 802.11a/g scrambler structure.

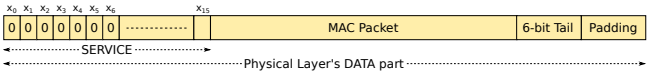


Figure 3: Prior to scrambling, the first 7 bits of SERVICE field in the Physical Layer's DATA part are set to zero.

we aim to design a generic classification technique that evaluates the likelihood of scrambling seed sequences based on the unique seed subsequences as seed signature. Our technique does not rely on the assumption that seeds are generated according to some shift distance, therefore it has the potential of distinguishing models that make use of their own algorithm for seed generation.

2.3.1 Recovering the Scrambling Seed

We briefly review the scrambling and descrambling processes defined in IEEE 802.11a/g/p.

Data scrambling. The binary data is scrambled by a special construction depicted in Figure 2, where a 7-bit linear feedback shift register (LFSR) produces the scrambled bit y_k at output by computing the exclusive-or (modulo-2 sum) of the input bit x_k and the LFSR feedback value z_k. The mathematical description of the scrambler in Figure 2 is given by

$$\begin{aligned} z_k &= z_{k-4} \oplus z_{k-7} \\ y_k &= x_k \oplus z_k \end{aligned} \tag{1}$$

where x_k, y_k are the k-th input and output bits, while z_k represents the feedback of the shift register at that time. We can represent the shift register's content by either a binary sequence $z_{k-1} \dots z_{k-7}$ or a single decimal value

$$s = z_{-1} \cdot 2^6 + \dots + z_{-6} \cdot 2 + z_{-7}. \tag{2}$$

To prepare a packet for transmission, the transmitter prepends the packet by a 16-bit SERVICE field and appends it by tail and padding bits to create the DATA part of the Physical Layer frame (Figure 3). The LFSR is initialized with a new seed value s and the first seven bits of SERVICE field are set to zero, then the whole DATA part is scrambled.

Recovering transmitter's seed. As seen in Equation (1), since $y_k = x_k \oplus z_k$, we have $y_k \oplus z_k = x_k \oplus z_k \oplus z_k = x_k$. Consequently, an identical structure to the one described in Figure 2 is used for the descrambling process, where x_k and y_k switch roles. The unknown scrambling seed can be recovered by the receiver by relying on the fact that the first seven bits of the SERVICE field prior to scrambling are $x_0 = \dots = x_6 = 0$, resulting in a frame after scrambling having the first seven bits $y_0, \dots, y_6$ of SERVICE field as $y_k = x_k \oplus z_k = z_k = z_{k-4} \oplus z_{k-7}$, specifically

$$\begin{aligned} y_0 &= z_{-4} \oplus z_{-7} & y_4 &= z_0 \oplus z_{-3} = y_0 \oplus z_{-3} \\ y_1 &= z_{-3} \oplus z_{-6} & y_5 &= z_1 \oplus z_{-2} = y_1 \oplus z_{-2} \\ y_2 &= z_{-2} \oplus z_{-5} & y_6 &= z_2 \oplus z_{-1} = y_2 \oplus z_{-1} \\ y_3 &= z_{-1} \oplus z_{-4} \end{aligned} \tag{3}$$

From the relations in Equation (3), the receiver can recover the transmitter's original seed s by first computing its bit values

$$\begin{aligned} z_{-1} &= y_6 \oplus y_2 & z_{-4} &= y_3 \oplus z_{-1} \\ z_{-2} &= y_5 \oplus y_1 & z_{-5} &= y_2 \oplus z_{-2} \\ z_{-3} &= y_4 \oplus y_0 & z_{-6} &= y_1 \oplus z_{-3} \\ & & z_{-7} &= y_0 \oplus z_{-4} \end{aligned} \tag{4}$$

then deriving s based on Equation (2).

2.3.2 Seed Patterns

In this subsection, we study the characteristics of seed sequences generated by Wi-Fi devices. We first define the seed pattern $P = (s_1, \dots, s_k)$ as a sequence of seed values s_i corresponding to *consecutive* Physical Layer frames (regardless of frame types) sent by a transmitter. We emphasize that the consecutiveness is a required property in the definition of seed pattern. If a frame is lost in the middle, we observe two separate seed patterns. Based on experimental results, we identify the following classes of seed patterns generated by commercial Wi-Fi devices.

- *Fixed:* The same seed value is used for all transmitted frame: $P = (s, s, \dots, s)$.
- *Incremental:* Seed value is incremented after every transmitted frame: $P = (s, s+1, \dots, s+k)$. Due to lost packets, a seed sequence may contain multiple incremental patterns.
- *Repeated pattern:* A group of seed values is repeated for some number of times. After that, a new group of seed values is selected and repeated in the same manner. The sequence may be observed as $P_1, \dots, P_1, P_2, \dots, P_2, \dots$.
- *Pseudo-random:* Seed values are pseudo-randomly generated: $P = (s_1, \dots, s_k)$ for random s_i and large k.

Characteristics. While seed patterns such as fixed or incremental can be easily recognized by observing a few seed values, the repeated and pseudo-random patterns are less trivial to understand. Based on experiments of multiple transmissions from different Wi-Fi models, we found the following characteristics of non-trivial seed sequences.

- *Non-zero:* Zero is never used by any device as a scrambling seed. This is also advised by the standard to avoid sending biased data stream.

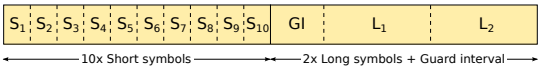
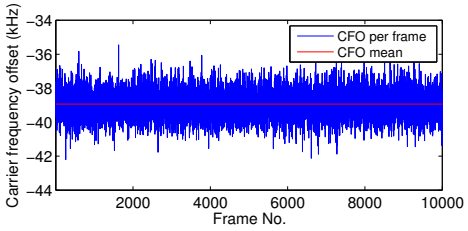
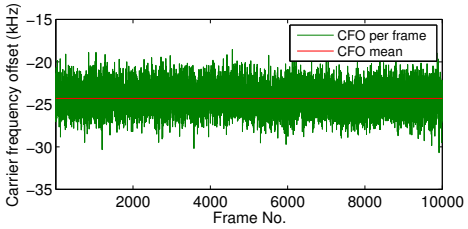


Figure 4: IEEE 802.11a/g/p preamble consists of 10 short symbols and 2 long symbols prepended with an extended guard interval (GI).



(a) D-Link: Mean = -38.9kHz



(b) LinkSys: Mean = -24.3kHz

Figure 5: Estimated CFO between our SDR Wi-Fi receiver and commercial Wi-Fi adapters: D-Link WDA-1320 and LinkSys WMP54G.

- *Uniform:* For a long enough transmission, seed distribution is actually uniform. This property indicates that a good classification algorithm should not solely rely on seed distribution to distinguish them.

- *Retransmission sensitive:* For some Wi-Fi chipsets, a retransmission can terminate the ongoing seed pattern and start a different pattern. This implies that those chipsets take into account the state of retransmission. In the classifier, the algorithm should be able to distinguish this pattern behavior due to retransmissions.

Later in Section 3.1, we discuss our method for classifying the device based on these scrambling seed characteristics.

2.4 Extracting Frequency Offset

The high spectral efficiency of IEEE 802.11a/g/p systems heavily relies on the subcarrier orthogonality, which allows the adjacent subcarriers to overlap each other to increase the bandwidth efficiency. However, the trade-off that OFDM systems make is that they are very sensitive to factors impacting the carrier orthogonality. One notable factor is the carrier frequency offset (CFO) caused by the typical slight frequency difference between the transmitter and receiver crystal oscillators, which directly impacts the down-conversion of the RF signal to baseband signal. The effect of CFO is the shift of the subcarriers in the frequency domain, resulting in the loss of orthogonality at the receiver. Another factor, which received less attention in the community but also greatly reduces the OFDM signal quality, is the sampling frequency offset (SFO) due to the unsynchronized sampling rate between the two RF front ends [21, 29]. The SFO causes the constellation symbols to rotate in the frequency domain. In typical low-cost RF devices, down-conversion and sampling tasks are driven by the same clock. Therefore, CFO and SFO are both usually present in OFDM communications.

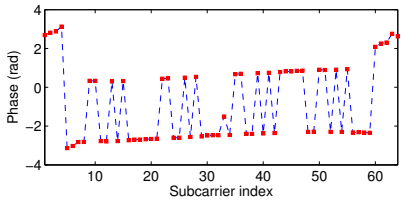


Figure 6: SFO causes phase shift across subcarriers, making phase of the constellation symbols (indicated in Red color) increasing (or decreasing).

In our fingerprinting system, we also exploit these factors to distinguish between Wi-Fi devices.

2.4.1 Carrier Frequency Offset

Due to the presence of a carrier frequency offset θ, every transmitted time-domain symbol s_n at discrete time n is rotated with a phase offset $n\theta$ and seen at the receiver as $r_n = s_n e^{jn\theta}$. In other words, the CFO makes the signals rotate in the time-domain. In order to avoid loss of orthogonality of subcarriers, the CFO must be compensated before the signal is transformed into the frequency domain. We estimate the CFO based on the special structure of the preamble, which consists of two parts: the first part comprises 10 identical short symbols, and the second part is composed of 2 identical long symbols (Figure 4). The repeated preamble symbols lead to an efficient estimation of CFO [26] as follows.

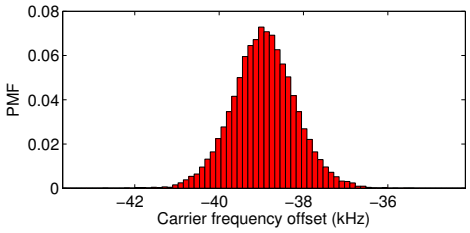
Estimation. Let us focus on short preamble symbols each containing $L = 16$ samples. For every time instant n, we compute the auto-correlation of r_n at lag L as $A = \sum_{k=0}^{L-1} r_{n+k+L} r^*_{n+k}$, where $*$ denotes the complex conjugate. When the frame preamble $\{p_n\}$ is found in the received signal, i.e., $r_n = p_n e^{jn\theta}$, we obtain

$$\begin{aligned} A &= \sum_{k=0}^{L-1} p_{n+k+L} e^{j(n+k+L)\theta} \left(p_{n+k} e^{j(n+k)\theta} \right)^* \\ &= \sum_{k=0}^{L-1} |p_{n+k}|^2 e^{jL\theta} \end{aligned} \tag{5}$$

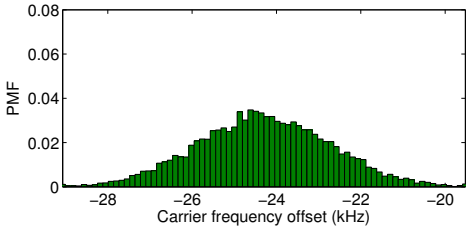
where the second equality is due to the repeated property of preamble symbols $p_{n+k+L} = p_{n+k}$. The CFO value θ is readily estimated as $\theta = \frac{\angle A + m2\pi}{L}$, for some integer m. In practice, since θ is typically smaller than the bandwidth of a subcarrier, m can be safely chosen to be $m = 0$ and hence, $\theta = \frac{\angle A}{L}$. In our SDR Wi-Fi receiver, we estimate the CFO using the following steps: (1) We use short preamble symbols to obtain a coarse estimate $\hat{\theta}$, and correct the rest of the signal with $\hat{\theta}$; (2) Now since the long preamble symbols are also repeated twice, we apply the same approach to compute the fine estimate $\tilde{\theta}$ based on the long preamble symbols, each consisting of $L = 64$ samples. Finally, we derive the final estimated CFO value $\theta = \hat{\theta} + \tilde{\theta}$. Figure 5 shows an example of a CFO recorded in our testbed, which fluctuates around a mean value due to noise in the environment. It is also noted that different Wi-Fi transmitters may create different CFO distributions at the receiver as observed with the LinkSys and D-Link adapters in the above example.

2.4.2 Sampling Frequency Offset

To observe the sampling frequency offset effect, we look into the transformation between the frequency and time domains of the signal. Let $d_1, \ldots, d_N$ be the data symbols on N subcarriers in an OFDM symbol period. For IEEE 802.11a/g/p, $N = 64$. The sender performs the Inverse Fourier Transform on data symbols to obtain



(a) D-Link WDA-1320



(b) LinkSys WMP54G

Figure 7: Histogram of CFO: D-Link vs. Linksys.

the time-domain signal $s_n = \mathcal{F}_n^{-1}(d_k) = \sum_{k=1}^{N} d_k e^{j2\pi kn/N}$ and transmits s_n to the receiver. To simplify the notations, we assume the carrier frequency offset has been compensated and the channel noise is negligible. However, due to the mismatched sampling rate between two RF front ends, the receiver obtains the time-domain samples as $r_n = s_{n(1+\epsilon)}$, where $\epsilon = (f_R - f_S)/f_S$ denotes the relative SFO between the transmitter's sampling frequency f_S and the receiver's sampling frequency f_R [28, 29]. Applying the Fourier Transform on r_n, the receiver obtain the subcarrier symbols $\hat{d}_k = \mathcal{F}_k(r_n)$. By the time-shifting rule, we can rewrite $\hat{d}_k$ as

$$\hat{d}_k = \mathcal{F}_k(s_{n(1+\epsilon)}) = \mathcal{F}_k(s_n)e^{j2\pi kn\epsilon/N} = d_k e^{j2\pi kn\epsilon/N}. \quad (6)$$

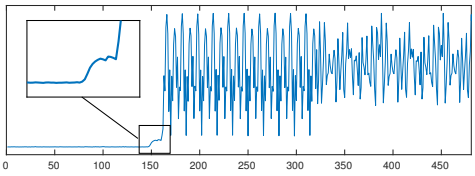
It can be seen that due to the sampling frequency offset ϵ the receiver obtains the rotated version of the original data symbols. Moreover, the phase shift $2\pi kn\epsilon/N$ is proportional to the subcarrier index k and OFDM symbol index $\lfloor n/N \rfloor$. Figure 6 shows an example of a slightly increasing phase shift along the subcarrier indices in an OFDM symbol. This leads to our pilot-aided SFO estimation method described in the following.

Estimation. In IEEE 802.11a/g, four pilot subcarriers are inserted equally in between the data subcarriers to assist the channel estimation at the receiver. We use the pilot symbols for SFO estimation as follows. Let p_i be the known pilot symbol, and k_i be the index of the i-th pilot carrier ($i = 1, 2, 3, 4$). According to Equation (6), the received pilot symbols in the m-th OFDM symbol (note that $m = \lfloor n/N \rfloor$) are $\hat{p}_{k_i} = p_{k_i} e^{j2\pi k_i m\epsilon}$.

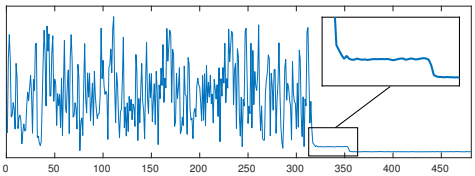
We compute $A_m = \prod_{i=1}^{4} \hat{p}_{k_i} = Ae^{j2\pi Km\epsilon}$, where $A = \prod_{i=1}^{4} p_{k_i}$ and $K = \sum_{i=1}^{4} k_i$ are constants. Similarly, we obtain $A_{m+1} = Ae^{j2\pi K(m+1)\epsilon}$ for the $(m+1)$-th OFDM symbol. Now, the SFO can be estimated by $\epsilon = \angle(A_{m+1}A_m^*)$, where $*$ indicates the complex conjugate. In our SDR Wi-Fi receiver, to reduce the variations due to noise, we compute the average SFO over multiple OFDM symbols within each frame and use it for fingerprinting purpose.

2.5 Extracting Frame Transient

The last physical feature that we use for fingerprinting is the signal transient at the start and the end of each transmitted frame. The signal transient is defined as the portion of the time-domain signal in this duration, in which the transmitted signal envelope changes



(a) Transient at frame start



(b) Transient at frame stop

Figure 8: Signal transients at the start and the stop of a frame transmitted by the Panda Ultra Wireless dongle.

from one stable energy level to another stable energy level. Figure 8a shows an example of the transient observed at the beginning of a frame transmission by the Panda Ultra Wireless dongle, in which the received signal starts to increase from the noise level and finally reaches a stable level after a few preamble samples. The transient at the end of this frame is shown in Figure 8b, where the signal energy drops to the noise level a few samples after the last sample of the frame.

While the IEEE 802.11 standard only requires a maximum period of $2\mu s$ for the signal transient (both at the frame's start and stop periods), no specific behavior of the transient is defined. As a result, the concrete signal shape in such periods is device specific primarily characterized by the hardware manufacturing process and its imperfections (which can be unique across devices of the same model). We exploit these signal transients to identify the devices.

We obtain the signal transient feature for each frame as follows. First, we identify each frame based on its special preamble (Figure 4). In our fingerprinting system, both frame identification and carrier frequency offset estimation (cf. Section 2.4.1) are performed at the same time. Without loss of generality, each detected frame is assumed to start from time 0 and stop at time L with L denoting the number of samples contained in the frame. By parsing the Physical Header, we can determine L. We obtain the frame's transient feature as the composition of the start and stop transients, which are sequences of samples captured in the following time durations:

- *Start transient:* $(r_{-T_1}, \ldots, r_{T_2})$,
- *Stop transient:* $(r_{L-T_3}, \ldots, r_{L+T_4})$,

where T_1, T_2 are the number of samples on the left and right of the first preamble sample, and T_3, T_4 are similarly defined with respect to the last sample of the frame.

3. DEVICE FINGERPRINTING

3.1 Seed classification

While optimal solutions for seed sequence classification deserve a more in-depth study, we are interested in a fast algorithm for finding a suboptimal classification of Wi-Fi devices. Our main idea for classifying Wi-Fi devices is to compute the likelihood of the scrambling seed sequences based on the common subsequences. We first define the likelihood between two seed patterns as follows.

DEFINITION 1 (LIKELIHOOD OF SEED PATTERNS). *Given two seed patterns $P = (s_1, \ldots, s_k)$ and $Q = (t_1, \ldots, t_m)$, let* $\mathrm{LCS}(P, Q)$ *be the longest common subsequence found in P and Q. The likelihood between P and Q is defined as*

$$\mathcal{L}(P,Q) = \begin{cases} \frac{|\mathrm{LCS}(P,Q)|}{\min(|P|,|Q|)} & \text{if } |\mathrm{LCS}(P,Q)| \geq n \\ 0 & \text{otherwise} \end{cases}$$

where $|\cdot|$ denotes the number of seeds contained in a sequence, and n is the likelihood threshold.

The pattern likelihood defined above has the property that for any patterns P and Q, if they are equal to each other or one is a subsequence of the another, then $\mathcal{L}(P, Q) = 1$, otherwise $\mathcal{L}(P, Q) \leq 1$. When $\mathcal{L}(P, Q)$ is close to 1, P and Q are more similar. By including $\min(|P|, |Q|)$ in the formula, we capture such scenarios, where a frame loss may cause a pattern to become a sub-pattern of the another. In such cases, they are considered similar according to the definition. The threshold n is used to control the likeliness decision region. We derive $n = 3$ based on experiments.

Finding seed patterns. As mentioned previously, the seed sequence used by a transmitter observed at the receiver may not comprise values belonging to consecutive frames due to various reasons such as frame loss, frames being incorrectly decoded, or simply the receiver missing the transmitted frame. In order to properly collect the seed patterns, we apply the following rules for processing the received seed values. Let s_{i-1} be the last seed retrieved from the packet trace. We accept s_i as the next seed in the same pattern, if

- $\text{MSEQ}_i = \text{MSEQ}_{i-1} + 1$ and $\text{RETX}_i = \text{false}$; or
- $\text{MSEQ}_i = \text{MSEQ}_i$ and $\text{RETX}_i = \text{true}$,

where MSEQ_i is the value in Sequence Number field of the i-th MAC data frame, and RETX_i is the Retry flag in Frame Control field, indicating whether the corresponding frame is retransmitted. If none of the above conditions holds, we consider that some frames might be missing in between, and therefore, s_i will start a new pattern. Based on the above rules, we can construct from the received seeds sequence multiple sets of patterns: $\mathcal{S} = \{P_1, \ldots, P_k\}$.

Algorithm. We now define the likelihood between two sets of seed patterns as follows.

DEFINITION 2 (PATTERN SETS LIKELIHOOD). *Given two sets of seed patterns $\mathcal{S} = \{P_1, \ldots, P_k\}$ and $\mathcal{R} = \{Q_1, \ldots, Q_m\}$, their likelihood is defined as*

$$\mathcal{L}(\mathcal{S}, \mathcal{R}) = \frac{1}{k} \sum_{i=1}^{k} \max_j \mathcal{L}(P_i, Q_j).$$

The likelihood between two pattern sets also has the property that $\mathcal{L}(\mathcal{S}, \mathcal{S}) = 1$ and $\mathcal{L}(\mathcal{S}, \mathcal{R}) \leq 1$ for any $\mathcal{S}$ and $\mathcal{R}$. This is the basis for our classification algorithm, in which we compute the likelihood between a given set of seed patterns retrieved from a transmitter and a reference set of seed patterns of known Wi-Fi devices. The outcome will be a known Wi-Fi device, which maximizes the likelihood. Our algorithm is given as follows.

ALGORITHM 1 (SEED CLASSIFICATION). *Given N sets of seed patterns $\mathcal{S}_1, \ldots, \mathcal{S}_N$ generated by known Wi-Fi devices, a Wi-Fi transmitter with seed pattern $\mathcal{S}$ is classified to be in class n^* using the following steps:*

1. *For each pattern set $\mathcal{S}_n$, compute $\mathcal{L}(\mathcal{S}, \mathcal{S}_n)$.*
2. *Return $n^* = \arg\max_{n=1\ldots N} \mathcal{L}(\mathcal{S}, \mathcal{S}_n)$.*

3.2 Frequency Offset Classification

While carrier frequency offset (CFO) and sampling frequency offset (SFO) are two distinct features that we use for our fingerprinting system, they are similar from the classifier's point of view. In this subsection, we focus on the classification techniques for the CFO only. The same approach can be directly applied to the SFO.

First, to study the CFO characteristics, we carried out an experiment to collect CFO values corresponding to every received frame from the same transmitter. Figure 5a shows the estimated CFO values between our SDR Wi-Fi receiver and a D-Link WDA-1320 Wi-Fi transmitter. It can be seen that the CFO fluctuates around a mean value of -38.9kHz (roughly 12.5% of the subcarrier bandwidth). The fluctuation is not only caused by the interference in the wireless medium, but also by the internal noise inside the RF front ends. For comparison, we perform a similar measurement for a Linksys WMP54G Wi-Fi transmitter, whose CFO values are shown in Figure 5b. Although the fluctuations of two devices look similar, their mean CFO values are different, which implies that they can be distinguished solely based on the mean CFO. This is the main idea of our first approach for frequency offset classification, specified in Algorithm 2.

ALGORITHM 2 (MEAN FREQUENCY OFFSET CLASSIFICATION). *Given N sets of carrier frequency offsets $\omega_1, \ldots, \omega_N$ corresponding to N known Wi-Fi devices, we identify a Wi-Fi transmitter as the n^* device by the following steps:*

1. *Compute the transmitter's average frequency offset θ over m received frames: $\theta = \frac{1}{m} \sum_{i=1}^{m} \theta_i$.*
2. *Compute $D(\theta, \omega_n) = |\theta - \omega_n|$ for each Wi-Fi device n.*
3. *Return $n^* = \arg\min_n D(\theta, \omega_n)$.*

As shown later in Section 4, this approach can achieve good results for a small set of devices. However, in our testbed evaluation with a large set of devices, we found that many devices can have quite close CFO values, resulting in misclassification. This motivates us to improve the accuracy by studying the distribution of CFO values. First, we compute the probability mass function of the CFO based on CFO values extracted from all received frames. Figures 7a and 7b illustrates an example of CFO histograms for the D-Link WDA-1320 and Linksys WMP54G adapters. In order to justify the difference between two distributions, we use a symmetric variant of the Kullback-Leibler (KL) divergence as the evaluation metrics. Specifically, the symmetric KL divergence between two distributions P and Q is computed by

$$D_{\mathrm{KL}}(P,Q) = \frac{1}{2} \sum_{i=1}^{B} \left(P_i \log \frac{P_i}{Q_i} + Q_i \log \frac{Q_i}{P_i} \right) \qquad (7)$$

where B is the number of discrete values two distributions can take (i.e., the number of bins in the histogram of the distribution).

We note that the KL divergence is only defined for non-zero probability distribution, i.e., $P_i > 0$ and $Q_i > 0$ for all $i = 1 \ldots B$. This requirement, however, may not be satisfied by the CFO distributions for two reasons: (1) Since we build the histogram of CFO based on measurement results, there might exist an empty bin leading to zero probability in that bin; (2) CFO distributions of two devices may only partially overlap or completely not overlap, resulting in the existence of such bins whose probabilities are not non-zero for both distributions. To solve these issues, we perform the following two steps prior to the KL divergence computation:

1. Translate the distribution to the origin, i.e., compute the mean of CFO values and subtract it from all CFO values before computing the probability mass function.

2. Replace zero-probability with non-zero ε-probability ($\varepsilon > 0$ and $\varepsilon \ll 1$).

Our algorithm for identifying a transmitter based on the frequency offset distribution is described in Algorithm 3.

ALGORITHM 3 (F.O. DISTRIBUTION CLASSIFICATION). *Given N sets of frequency offset distributions $Q_1, \ldots, Q_N$ belonging to N known Wi-Fi devices, the task of identifying a Wi-Fi transmitter comprises the following steps:*

1. *Compute the transmitter's frequency offset distribution P.*
2. *Translate P to the origin, replace $P_i = \varepsilon$ for all $P_i = 0$, and recompute P to obtain a proper distribution.*
3. *Compute $D_{KL}(P, Q_n)$ for all $n = 1 \ldots N$.*
4. *Return $n^* = \arg\min_n D_{KL}(P, Q_n)$.*

3.3 Frame Transient Classification

Our frame transient classification is based on the observation that while the signal emitted from the same Wi-Fi transmitter might exhibit differences during the transient periods across the transmitted frames, unique signatures can be obtained via averaging on multiple frames. In our fingerprinting system, we use the amplitude of transient samples as the transmitter's signature.

Let $(r_{i,1}, \ldots, r_{i,T})$ denote the sequence of $T = T_1 + T_2 + T_3 + T_4$ samples containing both start and stop transients of an i-th frame (cf. Section 2.5). The transient $\mathbf{a} = (a_1, \ldots, a_T)$ of a device is obtained by taking the average of sample amplitude over m detected frames: $a_k = \frac{1}{m} \sum_{i=1}^{m} |r_{i,k}|$ for $k = 1 \ldots T$. As we are only interested in the shape of the transient (i.e., the relative change of amplitude across samples), we eliminate the effect of absolute received power by normalizing the samples such that $\sum_{k=1}^{T} a_k = 1$. The similarity of two devices is now computed based on the difference between their transient signatures $\mathbf{a}$ and $\mathbf{a}'$ as follows:

$$D(\mathbf{a}, \mathbf{a}') = \sum_{k=1}^{T} |a_k - a'_k|.$$

Algorithm 4 summarizes our frame transient classification.

ALGORITHM 4 (FRAME TRANSIENT CLASSIFICATION). *Given N sets of transient signatures $\mathbf{a}_1, \ldots, \mathbf{a}_N$ corresponding to N known Wi-Fi devices, we identify a Wi-Fi transmitter as follows:*

1. *Compute the transmitter's transient $\mathbf{a}$ over m received frames.*
2. *Compute $D(\mathbf{a}, \mathbf{a}_n)$ for all $n = 1 \ldots N$.*
3. *Return $n^* = \arg\min_n D(\mathbf{a}, \mathbf{a}_n)$.*

3.4 Combined Classification

To improve the overall accuracy of our fingerprinting system, we combine the above individual features by linearly adding the similarity scores obtained from each feature and classify the devices based on the total score. We note that while the scrambling seed likelihood metric approaches 1 when the seed sequences are similar, the scores produced by other features converge to 0 if devices are alike. Therefore, for integration into the combined classification, we convert the scrambling seed likelihood into the seed score (distance) by computing $D(\mathcal{S}, \mathcal{R}_n) = 1 - \mathcal{L}(\mathcal{S}, \mathcal{R}_n)$. The combined metric is then derived as the sum of weighted individual scores with the weights denoted in Table 2.

Table 2: Features and weights used in the combined method for model classification and device identification.

Feature	Model classification	Device identification
CFO mean value	$\alpha_M = 0.4$	$\alpha_M = 0.3$
CFO distribution	$\alpha_D = 0.2$	$\alpha_D = 0.2$
SFO mean value	$\beta_M = 0.2$	$\beta_M = 0.2$
SFO distribution	$\beta_D = 0$	$\beta_D = 0.1$
Scrambling seed	$\gamma = 0.05$	$\gamma = 0$
Transient	$\tau = 0.45$	$\tau = 0.25$

ALGORITHM 5 (COMBINED CLASSIFICATION). *Given N Wi-Fi devices with known profiles for the feature set of scrambling seed, mean and distribution of carrier and sampling frequency offset, and frame transient, we identify a Wi-Fi transmitter as follows:*

1. *Compute the signatures sig_f of the tested Wi-Fi transmitter (as in Algorithms 1 to 4) for each feature f.*
2. *Compute the score $D_n = \sum_{f \in Features} w_f D_f(sig_f, profile_f)$ for all $n = 1 \ldots N$.*
3. *Return $n^* = \arg\min_n D_n$.*

4. FINGERPRINTING TECHNIQUES PERFORMANCE EVALUATION

4.1 Setup and Methodology

We report on our experimental results for a total 93 Wi-Fi devices of 13 different models, including 6 PCI adapters, 85 USB adapters, and 2 smartphones. Table 3 summarizes all the devices used in our testbed. Our general setup consists of a TP-Link N600 Access Point serving as the base station, and a desktop computer as a wireless transmitter using 91 Wi-Fi devices, among which 6 PCI adapters are directly attached to the computer through PCI slots, and the other 85 USB adapters are connected through USB hubs (Figure 9). For experiments with two smartphones, we associate them to the Access Point and use them as wireless transmitters. On the other side, the Access Point is also connected via the its Ethernet network interface to another desktop computer used as the receiver. We use the *iperf* traffic generator to transmit packets, each of 1500 bytes, between the transmitter and receiver. We carry out the experiments in our lab environment during daily hours, where traffic from other regular Wi-Fi users and human movement are also present. We place no constraint on location of the testbed nodes, and they can be dynamic (e.g., phone held in moving hand while transmitting). Our Wi-Fi testbed experiments were carried out on channel 11 with 20 MHz bandwidth.

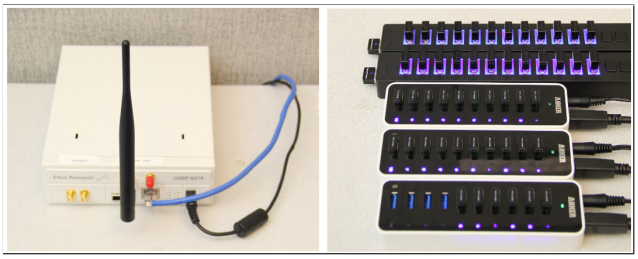


Figure 9: Our testbed consists of a custom made SDR Wi-Fi receiver (left), and 93 Wi-Fi transmitters (right) including 85 Wi-Fi dongles connected to the PC via USB hubs, 6 PCI adapters and 2 phones.

In order to capture the wireless communications for fingerprinting, we run our SDR Wi-Fi sniffer on an Ettus USRP N210 with

Table 3: Wi-Fi devices in our evaluation.

	Model	Quantity	Chipset	Type
1	D-Link WDA-1320	2	Atheros AR2413	PCI
2	Linksys WMP54G	2	Ralink RT2560	PCI
3	TP-Link TL-WN751ND	2	Atheros AR9227	PCI
4	Cisco Linksys AE2500	26	Broadcom BCM43236	USB
5	Panda Ultra Wireless	26	Ralink RT5370	USB
6	TP-Link TL-WN725N	26	Realtek RTL8188CUS	USB
7	Belkin F7D1102	2	Realtek RTL8188CUS	USB
8	Edimax EW-7811Un	2	Realtek RTL8188CUS	USB
9	TP-Link TL-WN321G v4	1	Ralink RT2070	USB
10	TP-Link TL-WN722N	1	Atheros AR9271	USB
11	TP-Link TL-WN821N v4	1	Realtek RTL8192CU	USB
12	Apple iPhone 5	1	Broadcom BCM4334	Phone
13	Nokia Lumia 635	1	Snapdragon 400	Phone
	Total	93		

SBX v3.0 daughterboard. Our Wi-Fi sniffer captures and performs signal processing on the overheard transmissions. During the decoding of Physical Layer frames, we extract the scrambling seeds, carrier and sampling frequency offsets belonging to every frame. Based on the MAC address in each frame, we group the extracted features together and run the fingerprinting algorithms to identify them. The rationale behind grouping based on MAC address is that even though the transmitter can modify his MAC address, it remains the same during the transmission session. We classify the transmitter based on features collected over multiple frames of the transmission. We note that we also use the features of frames with incorrect checksums as they still contain valuable information for classification purpose (as opposed to user data payload which might be uninteresting when corrupted). Moreover, by grouping frames of the same MAC address together, the chance of errors is lower (otherwise we would not see the same address field), and hence the Physical Layer features are more precise with high probability.

Our evaluation consists of extensive experiments over the span of three weeks. Using our SDR Wi-Fi receiver, we collected digital samples and extracted features from each of the 93 Wi-Fi transmitters for at least five different periods of time per day. Each transmission session is carried out in various time durations, ranging from 1 to 100 seconds. We use data collected on the first day as fingerprint profiles and each of the other days as tests.

4.2 Identifying Chipsets by Scrambling Seed

We first evaluate our scrambling seed classification algorithm on all the Wi-Fi devices in our testbed. Using the device profiles recorded on the first day, we compute the likelihood of scrambling seed sequences between the profiles and the tests. Our first observation is that any two devices with chipsets made from the same manufacturer have similar seed patterns during their transmissions. This is explained by the fact that since the scrambling seed sequence is produced by the baseband chipset, the generated seeds are independent of the device brand name, but they are chipset manufacturer specific.

Looking further into the difference in seeds generating mechanisms by different chipsets, we present both the average and the standard deviation of the likelihood between any two models summarized in Table 4. It is an interesting fact that all four chipset manufacturers of our experimental Wi-Fi devices develop their own seed generation completely different from each other, which fall into four classes we defined in Section 2.3. The Realtek chipsets always use a special value 124 as the scrambling seed, while Qualcomm Atheros chooses to increment the seed after every transmitted frame regardless of whether it is retransmitted. Although Broadcom and Ralink generate somewhat random seeds, their strategies are different.

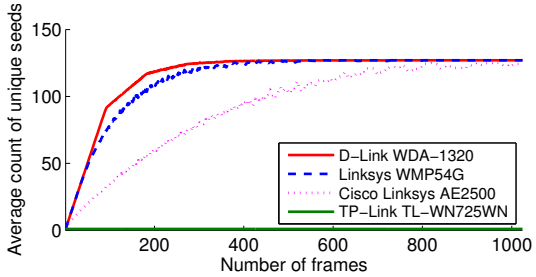


Figure 10: Comparison of seed generating strategies applied by different Wi-Fi adapters. The Y-axis shows the average number of unique seeds counted in every group of consecutive frames specified by the X-axis.

To illustrate the difference in seed generation, we conduct another experiment as follows. We select 4 Wi-Fi adapters (D-Link WDA-1320, Linksys WMP54G, Cisco Linksys AE2500, TP-Link TL-WN725N) corresponding to 4 baseband chipset brands (Atheros, Ralink, Broadcom, Realtek, respectively). In the trace of scrambling seed sequence extracted from each adapter, we count the number of unique seeds generated by the chipset over multiple groups of n consecutive frames, and take the average over those groups. We repeat this step for each value of n from 1 to 1024 and achieve an overall picture of how frequently new seeds are generated by different chipsets, shown in Figure 10. We can observe that, seed generating strategies applied by chipset manufacturers can be distinguished clearly by the curves separation. Even though seeds generated by Atheros (incremental) and Ralink (random) are completely different, they are more similar in terms of seed diversity within small periods of transmitted frames. It is also easily seen that Realtek chipsets use only one seed value for all frames, illustrated by the horizontal line. In contrast, though Broadcom BCM43236 generated seeds are random, they differ from the Ralink strategy. To understand how it produces seeds, we investigate the trace and found that Broadcom BCM43236 tends to "reuse" seeds for some number of frames. For example, we select a range of consecutive frames and read the corresponding seed values as follows: $76, 93, 108, 25, 108, 25, 79, 104, 79, 108, 93, 108, 93, 108, 25, 41, 79, 104, 79, 104, 85, 116, 35, 113, 124, 113, 35, 113$. We can immediately see that seed values $93, 108, 25, 79, 104$ are repeated after a few transmitted frames. This pattern is in fact quite common in the seed sequence generated by Cisco Linksys AE2500 adapter, and that is also illustrated by the slowly increasing average count of unique seeds over consecutive frames, shown in Figure 10.

Moreover, Broadcom seems to apply different seed randomizing mechanisms for different chip generations. This can be seen from the low likelihood between Apple iPhone 5 and Cisco Linksys AE2500 in Table 4. Table 5 summarizes the scrambling seed generating mechanisms discovered during our experiments. We conclude that based on analyzing the scrambling seed sequence received from the transmitter, we can reveal its chipset brand. When this is combined with other techniques, one can narrow down the fingerprinting and classification of a device.

4.3 Frequency Offset Fingerprint

As seen from Section 4.2, we can classify the Wi-Fi devices into different categories with respect to their baseband chipsets. Our goal in this subsection is to further differentiate the devices within the same category based on the frequency offsets extracted from every frame of the transmission.

First, we study the potential of using the average frequency offsets for differentiating the Wi-Fi device models. For this purpose,

Table 4: Average and standard deviation of scrambling seed likelihood (%) between different Wi-Fi adapter models.

	Model	1	2	3	4	5	6	7	8	9	10	11	12	13
D-Link WDA-1320	1	61±2	0	55±2	0	0	0	0	0	0	58±.5	0	0	1±.6
Linksys WMP54G	2	0	71±3	0	0	48±4	0	0	0	49±3	0	0	2±.8	32±2
TP-Link TL-WN751ND	3	55±2	0	63±5	0	0	0	0	0	0	61±3	0	0	0
Cisco Linksys AE2500	4	0	0	0	88±1	0	0	0	0	0	0	0	0	0
Panda Ultra Wireless	5	0	48±4	0	0	80±4	0	0	0	83±4	0	0	8±3	77±4
TP-Link TL-WN725N	6	0	0	0	0	0	100	100	100	0	0	100	0	0
Belkin F7D1102	7	0	0	0	0	0	100	100	100	0	0	100	0	0
Edimax EW-7811Un	8	0	0	0	0	0	100	100	100	0	0	100	0	0
TP-Link TL-WN321G v4	9	0	49±3	0	0	83±4	0	0	0	83±5	0	0	6±2	69±5
TP-Link TL-WN722N	10	58±.5	0	61±3	0	0	0	0	0	0	58±7	0	0	0
TP-Link TL-WN821N v4	11	0	0	0	0	0	100	100	100	0	0	100	0	0
Apple iPhone 5	12	0	2±.8	0	0	8±3	0	0	0	6±2	0	0	27±5	18±7
Nokia Lumia 635	13	1±.6	32±2	0	0	77±4	0	0	0	69±5	0	0	18±7	71±10

(standard deviation is omitted when zero)

Table 5: Scrambling seed generation methods by Wi-Fi devices.

Class	Model	Seed type
A	Belkin N150, Edimax EW-781Un TP-Link TL-WN725N, TP-Link TL-WN821N	fixed = 124
B	D-Link WDA-1320, TP-Link TL-WN751ND, TP-Link TL-WN722N	incremental
C	Cisco Linksys AE2500	repeated
D	Apple iPhone 5	random
E	Linksys WMP54G, Panda Ultra Wireless, TP-Link TL-WN321G, Nokia Lumia 635	random

we select one device per model in Table 3 and extract the carrier and sampling frequency offsets from every received frame belonging to the same transmitter. To see how the frequency offset changes over a short time, we capture 1 second of the transmission for each device, and repeat this 5 times, each 1 minute after the previous one. The average CFO and SFO are shown in Figure 11.

CFO based distinguishing. We observe that the CFO can be quite different for devices of the same chipset. As an example, Belkin F7D1102 and Edimax EW-7811Un (both use identical chipset Realtek RTL8188CUS) have CFO around 10kHz and -10kHz, respectively. On the contrary, the Belkin's CFO is very close to the Panda Ultra's despite that they use different chipsets. If we repeat the experiment for 20 times and apply Algorithm 2 for all devices, we obtain the average correct detection rate of 17% with the standard deviation of 2%.

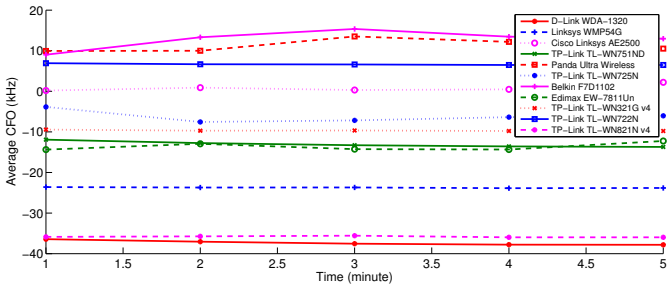
SFO based distinguishing. The average SFO values of the experimented devices, as seen from Figure 11b, are much closer to each other. In the above experiment, we also perform the average SFO based classification and observe that there is a drop of average correct detection rate to 9% with the standard deviation of 2%. This can be explained by the SFO of devices being in a much smaller vicinity, resulting in lower detection accuracy.

4.4 Transient Fingerprint

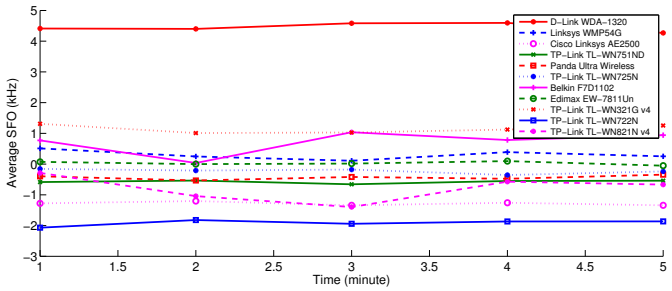
Using the set of 93 devices, we extract the start and stop transients of each frame and apply Algorithm 4 to classify the devices. The results in Table 6 show that most device models can be recognized with a high probability. There are, however, exceptions that the TP-Link TL-WN722N Wi-Fi dongle and the Apple iPhone5 are not correctly classified in all experiments. In fact, they are misclassified to TP-Link TL-WN725N. In the next evaluation, we show that despite individual techniques do not achieve high accuracy, the combined classification can improve significantly the results.

4.5 Device Identification

To evaluate the classification accuracy for a large set of devices, we perform an extensive experiment, in which we collect the scrambling seed, frame transients, CFO and SFO values from all Wi-Fi devices in our testbed. The extracted features of tested devices are collected from 20 experiments during three weeks. For each transmitter, we apply Algorithms 1 to 5 as the classification methods for identifying *both* the models and the devices themselves. We use the weights specified in Table 2. Note that in case of device identification, we set the scrambling seed weight to 0 since the scrambling seed sequence is produced in the same manner by devices of the same chipset, and as a result, negative impact can be created if the scrambling seed is included in the scoring metric. The results are reported in Table 7, where the outcomes are averaged out over 20 experiments.



(a) CFO.



(b) SFO.

Figure 11: Average CFO and SFO of different Wi-Fi devices measured per one-minute transmission during the span of 5 minutes.

First, we see that for the model identification task, even the worst-performance task, the classification based on scrambling seed, can still achieve an accuracy of 87%. The combined method can successfully recognize the device model with a high probability of roughly 95%. The most interesting results are the device identification capability of our fingerprinting system. By exploiting all physical features supported in our system, we can trace the identity of almost half (47%) of the Wi-Fi devices in the testbed.

Table 6: Model identification results for 93 devices using transient classification method.

Model	Model Identification Accuracy (%)
D-Link WDA-1320	100
Linksys WMP54G	100
TP-Link TL-WN751ND	75
Cisco Linksys AE2500	99
Panda Ultra Wireless	91
TP-Link TL-WN725N	97
Belkin F7D1102	75
Edimax EW-7811Un	88
TP-Link TL-WN321G v4	100
TP-Link TL-WN722N	0
TP-Link TL-WN821N v4	75
Apple iPhone 5	0
Nokia Lumia 635	50

Table 7: Fingerprinting results on 93 devices with different classification methods.

Method	Model identification (%)	Device identification (%)
CFO mean value	55±7	17±2
CFO distribution	55±6	10±2
SFO mean value	61±4	9±2
SFO distribution	25±1	1±.1
Scrambling seed	87±1	6±1
Transient	92±1	26±3
All combined	**95±1**	**47±3**

5. RELATED WORK

Previous work considered several approaches to fingerprint and identify radio devices, as well as techniques to prevent cloning and spoofing. Some early work considered the characteristics of the MAC and higher layers of devices. These characteristics where used to detect the location and even the identity of users [20]. For instance, in a conference event hosted on the West coast, overhearing a probe request with id "MIT", is indicative that that a person from MIT might be at the conference, as his Wi-Fi device is trying to associate with an AP of the MIT campus. The assumption that the MAC Sequence number cannot be manipulated by an adversary was used to propose techniques to detect spoofing [12]. Other work, investigated the use of Time of Probe request frames from STA [9]. They measured the received time of those Probe Request frames, and analyzed the interval between them. The goal of this study is to identify different driver behaviors in terms of implementation of the MAC protocol. This work exploits the fact that the standard does not specify the step-by-step behavior for sending the MAC frames (like Probe Requests).

The general problem of uniquely identifying RF devices based on their physical characteristics has been studied for various security applications including authentication and prevention of wormhole attacks as part of the area of Physically Unclonable Functions (PUFs) [16, 19]. Early work demonstrated that it is possible to fingerprint and uniquely identify the CC1000 radios of 10 Cricket Motes operating at 433MHz with an average accuracy of 70% [3]. This work focused on several features of the transient signal of the CC1000 radio. It opened the door for various applications of fingerprinting both in terms of adversarial use such as the potential of invading privacy, and defensive use such as preventing spoofing and wormhole attacks in wireless sensor networks. More recent work investigated the fingerprinting of USRP transceivers using preamble-based identification [24]. This work experimented with seven different USRPs and used a machine learning algorithm (kNN). The paper claims that for each USRP receiver, it is possible to identify the transmitter based on the recorded samples. It is, however, difficult to justify the accuracy of this technique in practice, because during the evaluation the nodes were static. This means that each pair of USRPs has a unique channel with different multipath and RSSI and the system might be mostly identifying the channel. It is unclear how such techniques would perform if the nodes are relocated, or if the environment is changed.

Prior work on fingerprinting Wi-Fi devices investigated the second-order cyclic features of OFDM signals [17]. This work looked at the spectral cross-correlation of the signal based on the assumption that signals of most of communication systems today have periodicity, such as modulated sinusoidal carrier, and cyclic prefix. However, the devices locations were static and it is unclear if this technique too fingerprinted the channel or the devices and how it would perform when the devices are relocated. Moreover, their experiments were performed for a relatively small set of 6 devices.

The closest and best performing related work on fingerprinting Wi-Fi devices considered a combination of frequency offset, transients, and constellation errors for IEEE802.11b cards. A performance accuracy of 99% was reported for a matching of two recordings. While they experimented with a relatively large number of devices (138), all the devices were located in the stable, static and RF-insulated Orbit Lab environment (almost no noise, stable temperature) [4]. It is unclear if this impressive performance is due to the stability of laboratory environment between the two runs (e.g., absence of noise boosted the performance of error vector magnitude or stable laboratory temperature led to a stable CFO), or to other unknown factors. Since then no other work (see [5] and references therein) was able to reproduce these results so far.

Our work, not only provides a framework for repeatable experimentation with Wi-Fi fingerprinting on a low-cost flexible hardware/software platform, it also devised new features (e.g., scrambling seed) and techniques (Kullback-Leibler divergence) across the various blocks of an RF transceiver chain. We achieved high accuracy for Wi-Fi model classification. While our results for device identification reduce to almost half, we believe that further improvements are possible on our platform such as a more careful analysis of the scrambling seed algorithm and the per-carrier frequency domain RF front end characteristics, or using directional antennas to scan and focus on different directionns [31].

6. CONCLUSION

We developed a set of techniques for RF fingerprinting Wi-Fi devices. Our techniques span several blocks of a Wi-Fi receiver. Our results indicate that identifying Wi-Fi devices is possible (with results spanning 44%-50%), and potentially feasible with hardware implementation of lower cost than Wi-Fi chipsets. Further, improvements of the proposed techniques are also possible, for example with more careful analysis and reverse engineering of scrambling seed algorithms, CFO and SFO patterns. Through this work, we hope that the research community can build on our results and tools to better understand the potential and privacy-invasion risks emanating from Wi-Fi devices fingerprinting.

Acknowledgements. This material is based upon work supported by the National Science Foundation under Grant No. NSF/CNS-1409453.

References

[1] 3GPP TS 24.312. Access Network Discovery and Selection Function (ANDSF) Management Object (MO). http://www.3gpp.org/DynaReport/24312.htm, 2014.

[2] B. Bloessl, M. Segata, C. Sommer, and F. Dressler. Decoding IEEE 802.11a/g/p OFDM in Software using GNU Radio. In *19th ACM International Conference on Mobile Comput-*

ing and Networking (MobiCom 2013), Demo Session, pages 159–161, Miami, FL, October 2013. ACM.

[3] K. Bonne Rasmussen and S. Capkun. Implications of radio fingerprinting on the security of sensor networks. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 331–340, Sept 2007.

[4] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless device identification with radiometric signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, pages 116–127, New York, NY, USA, 2008. ACM.

[5] B. Danev, D. Zanetti, and S. Capkun. On physical-layer identification of wireless devices. *ACM Comput. Surv.*, 45(1):6:1–6:29, Dec. 2012.

[6] S. Datoo. This recycling bin is following you. http://qz.com/112873/this-recycling-bin-is-following-you/, Quartz, August 2013. Accessed: May, 2015.

[7] Ettus Research. USRP: Universal software radio peripheral.

[8] Euclid Analytics. Privacy statement. http://euclidanalytics.com/about/privacy-statement/. Accessed: May, 2015.

[9] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. Van Randwyk, and D. Sicker. Passive data link layer 802.11 wireless device driver fingerprinting. In *Proceedings of the 15th Conference on USENIX Security Symposium - Volume 15*, USENIX-SS'06, Berkeley, CA, USA, 2006. USENIX Association.

[10] Free Forfait Mobile. FreeWiFi secure EAP-SIM. http://mobile.free.fr/assistance/261.html, August 2010.

[11] GNU. Gnu radio. http://www.gnuradio.org.

[12] F. Guo and T.-c. Chiueh. Sequence number-based mac address spoof detection. In A. Valdes and D. Zamboni, editors, *Recent Advances in Intrusion Detection*, volume 3858 of *Lecture Notes in Computer Science*, pages 309–329. Springer Berlin Heidelberg, 2006.

[13] H. Haverinen and J. Salowey. Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM). RFC 4186 (Informational), January 2006.

[14] L. Hutchinson. iOS 8 to stymie trackers and marketers with mac address randomization. http://arstechnica.com/apple/2014/06/ios8-to-stymie-trackers-and-marketers-with-mac-address-randomization/, June 2014. Accessed: May, 2015.

[15] IEEE. IEEE 802.11 Interworking with External Networks. http://standards.ieee.org/findstds/standard/802.11u-2011.html.

[16] S. Katzenbeisser, Ünal Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann. PUFs: Myth, fact or busted? a security evaluation of physically unclonable functions (pufs) cast in silicon. In *Cryptographic Hardware and Embedded Systems-CHES, Springer*, 2012.

[17] K. Kim, C. Spooner, I. Akbar, and J. Reed. Specific emitter identification for cognitive radio with application to ieee 802.11. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1–5, Nov 2008.

[18] S. Narain, T. D. Vo-Huu, K. Block, and G. Noubir. Inferring location and traffic patterns without permissions using side-channels. In *Proceedings of IEEE Symposium on Security and Privacy*, 2006.

[19] NXP. PUF - physical unclonable functions protecting next-generation smart card ics with sram-based pufs. http://www.nxp.com/documents/other/75017366.pdf, 2013.

[20] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall. 802.11 user fingerprinting. In *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, MobiCom '07, pages 99–110, New York, NY, USA, 2007. ACM.

[21] T. Pollet, P. Spruyt, and M. Moeneclaey. The BER performance of OFDM systems using non-synchronized sampling. In *Global Telecommunications Conference, 1994. GLOBECOM '94. Communications: The Global Bridge, IEEE*, pages 253–257 vol.1, Nov 1994.

[22] Qualcomm. 3G LTE Wifi offload framework: Connectivity Engine (CnE) solution, July 2013. http://www.qualcomm.com/media/documents/3g-lte-wifi-offload-framework.

[23] M. Ramsay. Wi-Fi offload rising amid soaring data traffic. http://www.wirelessweek.com/News/2012/07/technology-WiFi-Offload-Rising-Amid-Soaring-Data-Traffic/, July 2012.

[24] S. U. Rehman, K. W. Sowerby, and C. Coghill. Analysis of impersonation attacks on systems using RF fingerprinting and low-end receivers. *Journal of Computer and System Sciences*, 80(3):591 – 601, 2014. Special Issue on Wireless Network Intrusion.

[25] B. Rooney. Data-hungry 4G users gorge on Wi-Fi, report finds. http://blogs.wsj.com/tech-europe/2013/09/19/data-hungry-4g-users-gorge-on-wi-fi-report-finds/, September 2013.

[26] T. Schmidl and D. Cox. Robust frequency and timing synchronization for OFDM. *Communications, IEEE Transactions on*, 45(12):1613–1621, Dec 1997.

[27] Z. M. Seward and S. Datoo. City of london halts recycling bins tracking phones of passers-by. http://qz.com/114174/city-of-london-halts-recycling-bins-tracking-phones-of-passers-by/, Quartz, August 2013. Accessed: May, 2015.

[28] M. Sliskovic. Carrier and sampling frequency offset estimation and correction in multicarrier systems. In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 1, pages 285–289, 2001.

[29] M. Sliskovic. Sampling frequency offset estimation and correction in OFDM systems. In *Electronics, Circuits and Systems, 2001. ICECS 2001. The 8th IEEE International Conference on*, volume 1, pages 437–440, 2001.

[30] M. Vanhoef, C. Matte, M. Cunche, L. Cardoso, and F. Piessens. Why MAC Address Randomization is not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms. In *ACM AsiaCCS*, Xi'an, China, May 2016.

[31] T. D. Vo-Huu, E.-O. Blass, and G. Noubir. Counter-jamming using mixed mechanical and software interference cancellation. In *Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '13, pages 31–42, New York, NY, USA, 2013. ACM.

[32] T. D. Vo-Huu, T. D. Vo-Huu, and G. Noubir. SWiFi: An Open Source SDR for Wi-Fi Networks High Order Modulation Analysis. Technical report, 2015.