Practical Obfuscation of BLE Physical-Layer Fingerprints on Mobile Devices

Hadi Givehchian, Nishant Bhaskar, Alexander Redding, Han Zhao, Aaron Schulman, and Dinesh Bharadia

University of California, San Diego
{hgivehch, nibhaska, alredding, h1zhao, schulman, dineshb}@ucsd.edu

Abstract—**Mobile devices continuously beacon Bluetooth Low Energy (BLE) advertisement packets. This has created the threat of attackers identifying and tracking a device by sniffing its BLE signals. To mitigate this threat, MAC address randomization has been deployed at the link-layer in most BLE transmitters. However, attackers can bypass MAC address randomization using lower-level physical-layer fingerprints resulting from manufacturing imperfections of radios. In this work, we demonstrate a practical and effective method of obfuscating physical-layer hardware imperfection fingerprints. Through theoretical analysis, simulations, and field evaluations, we design and evaluate our approach to hardware imperfection obfuscation. By analyzing data from thousands of BLE devices, we demonstrate obfuscation significantly reduces the accuracy of identifying a target device. This makes an attack impractical, even if a target is continuously observed for 24 hours. Furthermore, we demonstrate the practicality of this defense by implementing it by making firmware changes to commodity BLE chipsets.**

1. Introduction

Mobile devices, such as phones and smartwatches, frequently and continuously transmit Bluetooth Low Energy (BLE) beacons. These BLE beacons are used for a variety of applications, including tracking lost devices [21], COVID-19 electronic contact tracing [9], and cross-device continuity used for automated device hand-off and other proximity features [2]. For instance, a typical iPhone continuously transmits more than 800 BLE beacons per minute [19]. These beacons can make it possible to wirelessly track mobile devices: an attacker can setup sniffers to capture beacons in multiple locations and search for beacons from a target device's MAC address. To mitigate this privacy threat, most BLE transmitters randomize their MAC address every 15–20 minutes [7].

Unfortunately, adversaries can bypass MAC address randomization and uniquely identify devices using a different fingerprint in their transmissions: the unique physical-layer fingerprints introduced by hardware imperfections in mobile devices. These are slight imperfections introduced during the manufacturing of the underlying analog hardware—e.g., Carrier Frequency Offset (CFO), I/Q Offset, and non-linearities of power amplifier. Prior work has demonstrated it is feasible to use these imperfections to uniquely identify a mobile device transmitting wireless signals on BLE (and WiFi) in less than a minute, at any desired location, even if there are hundreds of other devices around [19], [8], [53], [26]. However, there is no existing work that demonstrates a practical and effective way to conceal these hardware imperfection fingerprints.

In this work, we demonstrate a practical method to effectively obfuscate CFO imperfections, which are the primary physical-layer fingerprint used to identify BLE devices [19], [53]. Achieving practical physical-layer obfuscation required overcoming several key challenges: First, devices can not observe their own physical-layer fingerprint nor can they observe the fingerprint of the devices around them. Therefore, devices need to obfuscate their fingerprints blindly. Second, altering the fingerprint of a device can make its transmissions harder to receive and demodulate. Therefore, obfuscation is limited in its range to comply with BLE protocol standards. Third, obfuscation cannot be done arbitrarily due to the limited precision available for altering fingerprints in commodity chipsets. Therefore, there are only a finite number of obfuscated fingerprints. Finally, attackers can reverse-engineer the obfuscation behavior of their targets by observing them over extended periods by linking consecutive randomized MAC addresses. The attacker may use different inference methods to identify the obfuscated device. This includes classical machine learning and statistical classifiers as well as black-box classification with neural networks. Some obfuscation strategies that are effective against certain attacks may not be against others.

Our contributions are as follows: (1) We identify the challenges and limitations in obfuscating hardware imperfections in wireless transmissions §2. (2) We provide an empirical model of these imperfections based on a large-scale field data §4.1. (3) We provide a framework to analyze the effectiveness of physical-layer obfuscation, and the theoretical foundations for an optimal physical-layer adversary that has the strongest-possible identification methodology §4.2. (4) We introduce an effective obfuscation strategy such that even the optimal adversary cannot accurately identify a device even after 24 hours of continuous observation §5.1–5.5. (5) We demonstrate the practicality of fingerprint obfuscation by implementing it using commodity BLE chipsets §6. (6) We demonstrate our obfuscation approach is effective using large-scale field data § 7. Finally, (7) we discuss how these analysis and obfuscation strategies can be applied to other wireless protocols §8.4, and other imperfections §8.3.

2. Threat Model

In this study, we consider an attacker that tries to use the physical-layer to wirelessly detect when their target—a user with a mobile device—is present at a specific location. The attacker first fingerprints the target's uniquely identifiable physical-layer characteristics—e.g., Carrier Frequency Offset (CFO)—by passively sniffing packets from the target's raw transmissions with a receiver such as a Software Defined Radio (i.e., *Fingerprinting phase*). Later, the attacker determines if their target is present at a location by sniffing transmissions from all nearby devices. If any fingerprints are close to the target's fingerprint, then the attacker assumes the target was present at that location (i.e., *Identification phase*). Such physical-layer fingerprinting attacks have been demonstrated in previous work on various mobile wireless protocols (e.g., WiFi and Bluetooth) [8], [28], [32], [51], [20], [14], [15], [19].

Why does obfuscation prevent this attack? This attack is only feasible because devices have stable fingerprints: each radio has stable physical imperfections in its transmitter circuitry that are born during manufacturing of its hardware components. However, if a target is obfuscating transmissions by altering their imperfections over time, this attack will trivially no longer be successful.

De-obfuscation requires prolonged observations: An attacker can only de-obfuscate a target that is changing its fingerprint over time if the attacker can observe multiple changes of the same target. The reason is, each time the target changes its fingerprint, it reveals more information about the true fingerprint that is being hidden. For instance, if the obfuscation approach is to alter the fingerprint by adding a magnitude picked uniformly at random, the attacker can recover the target's true fingerprint by averaging them.

Optimal attacker: Since the attacker can use a variety of de-obfuscating approaches (e.g., neural networks), we study an optimal attacker who is aware of our obfuscation methods and can optimally de-obfuscate the fingerprints when identifying the target to achieve the maximum possible identification accuracy. The optimal attacker also can observe their target across enough time periods where any new observations will not substantially improve their ability to identify the target. Further, we assume the optimal attacker has already recovered the true fingerprint of the device during the fingerprinting phase.

3. Challenges

Radio fingerprinting attacks rely on the following properties of radio fingerprints: their stability over time and their distinguishability between different devices. Next, we describe why it is impractical to remove these fingerprints and instead why we need to obfuscate them by periodically changing them.

3.1. It is impractical to remove radio imperfections

The most straightforward way to eliminate the threat of radio fingerprinting attacks would be to calibrate out the imperfections of every radio after they are manufactured. However, this would add significant complexity to the testing process for mobile devices: we would need to precisely and accurately measure and compensate imperfections for *radios used in every single mobile device*. Vendors are not incentivized to do this because it would offer no additional benefits for device performance. Another approach is to improve the quality of the hardware components to reduce the margin of the imperfections. For instance, manufacturers could use high-quality crystal oscillators as sources for the frequency synthesisers in their radios. However, this would increase the cost of each device, and may even consume more power (e.g., using a temperature-compensated oscillator) and would not offer any improvements in device performance. Ideally, we want an approach that can be deployed by vendors on existing hardware that can be incrementally deployed as a firmware change in their chipsets. The practical obfuscation we introduce in this work does not require any changes of the hardware testing or design of radios in mobile devices.

3.2. Strawman: Randomize imperfections

An example of a strawman obfuscation technique is periodically and randomly altering the Carrier Frequency Offset (CFO) of a device's transmissions to obscure hardware imperfections. This is similar to the approach taken in MAC address randomization in the sense that the identifying feature of the device periodically changes to new random values. However, the nature of these imperfections creates unique and different challenges and limitations.

For each packet p, the obfuscation model changes the **original hardware imperfection**, H, (e.g, $H = \{\text{CFO, I offset, Q offset}\}$), randomly into a new **obfuscated hardware imperfection**, H'_p. Once obfuscation is done, the attacker can only observe and use H'_p to infer any information about H or the identity of the transmitter device. Ideally, the obfuscated imperfection (H'_p) should not reveal any information about the original imperfection (H) or the identity of the device. However, such perfect obfuscation is not possible in practice because the obfuscated imperfection H'_p is randomized around the original hardware imperfection H. Given that perfect obfuscation is not practical, we must design the obfuscation in a way that significantly reduces the attacker's ability to uniquely identify a transmitter

The reason why random obfuscation fails is due to the practical limitations on altering the CFO of BLE devices. Random obfuscation is defined as adding a random value h^r to CFO. For example, assume h^r is sampled uniformly at random from the obfuscation set $\{-b : s : b\}$, where $b = 50KHz$ or *range* is the largest value h^r can take and $s = 5KHz$ or *resolution* is the the smallest granularity by which we can change h^r. Figure 1 shows how random obfuscation fails with two devices with a true CFO of 17.5 and -10

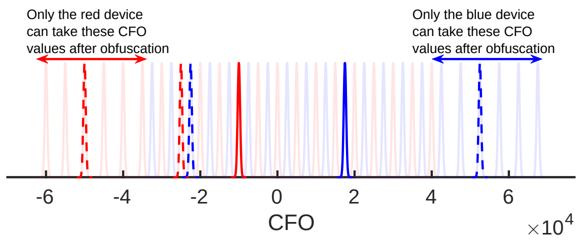

Figure 1: Simple randomized obfuscation will fail in hiding the identity of devices.

KHz. The solid red and blue colors show the original CFO of two devices. The light colored versions show how random obfuscation affects the values of the CFO. Note that instead of a single point, measured CFOs are demonstrated as a Gaussian distribution to represent the measurement noise.

The attacker can easily distinguish these two devices even after obfuscation due to the following reasons (dashed lines show examples of failure cases):

Limited obfuscation range: The range of obfuscation is limited, so there are some CFO values on the right that only the blue device can take and some on the left that only the red device can take. If the obfuscated CFO happens to be one of these, the attacker can easily identify the device. Note that the obfuscation range must be always limited because wireless communication protocol standards impose a limit on the acceptable range of hardware imperfections (e.g., BLE can only have ± 150 kHz [42]).

Limited obfuscation resolution: Even in the range that the obfuscated CFO of both devices can fall into, we see that their CFOs are still distinguishable. This is because the obfuscation resolution is larger than the measurement noise, so the measured obfuscated CFO values never overlap.

Prolonged observation time: The attacker may observe the device for a long time to obtain multiple instances of the obfuscated fingerprint changing. The attacker can then de-obfuscate the observed CFO values from each device, for example, by simply averaging.

Uniqueness of the fingerprint: All the aforementioned challenges are intensified when the true CFO of the devices we are trying to identify are far apart. This is possible because hardware imperfections of different devices usually follow a normal-like distribution (Figure 2). The devices that have rare imperfections (tails of distribution) are easily distinguishable from almost any other device. An effective obfuscation approach must be able to reduce the distinguishability of those extreme devices as well.

Number of nearby devices: If we have a very small number of devices in the environment, it becomes harder to hide a device's true identity by obfuscating. For instance, when the attacker is looking for the device in a private place such as a house, there aren't many devices around to hide amongst them.

4. Analysis Framework

In this section, we present an analytical framework for evaluating how well different obfuscation strategies prevent physical-layer hardware imperfection fingerprinting attacks on BLE devices. This framework is based on an empirically-driven model for CFO fingerprints built from 1,000 BLE devices that we use to analyze how well obfuscation will work in the real world. It also provides a metric to measure how well an optimal attacker can identify a BLE device.

Why only study CFO imperfections? This analysis is only focused on CFO imperfections for the following reasons: (1) it is the most distinguishable imperfection [8] sufficient to solely make several BLE devices identifiable [19], (2) observing CFO is available to the attacker with a low barrier of entry (CFO can be obtained from a low-cost commodity receiver chipset [53]). That said, our analysis can be similarly done for many other imperfections. We discuss how the same analysis and design can potentially be expanded to some other hardware imperfections in Section 8.3. Also, since CFO is present in all wireless transmissions, our obfuscation analysis can be used for protocols other than BLE. Lastly, we can study CFO obfuscation in isolation because most radio fingerprints are caused by independent hardware imperfections and do not impact each other (e.g., CFO comes from the oscillator and I/Q offset comes from the RF frontend).

4.1. Modeling real-world CFO imperfections

We collected a large dataset of BLE advertisement beacons from 1,000 real-world devices at several locations (coffee shops, libraries, food courts). We estimate the CFO in each beacon using the BLE fingerprinting methodology from [19]. The MAC address in the beacons was used to group the packets from the same device[1].

Based on this data set, we made the following observations that we use to model the CFO imperfections of real-world devices:

Measured CFO is stable across packets. Although the imperfections measured for different packets with the same MAC address are not exactly the same, the median of standard deviation of CFO for packets from the same device is only 320 Hz (based on 100 packets per device). We refer to these variations as measurement noise.

The distribution of CFO imperfections is Gaussian. We average the CFO of 100 packets with the same MAC address to estimate the CFO of each device. The distribution of CFO across devices looks bell-shaped (Kurtosis is 3.7) as shown in Figure 2, with standard deviation of 9.12 kHz. We also verified that I/Q offsets follow a normal-like distribution. A normal distribution is expected because the source of imperfections is manufacturing variations which often have a normal distribution.

1. Devices in this dataset are likely to have a consistent MAC address over this interval because we observe them for less than a minute.

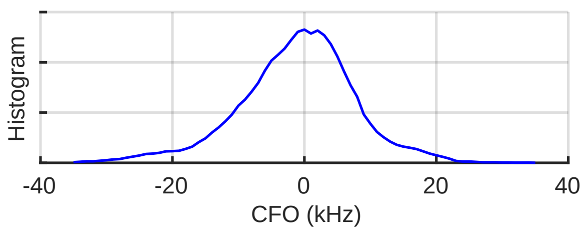

Figure 2: CFO distribution of 1000 real-world BLE devices looks Gaussian.

Modeling the hardware imperfections for analysis. Based on these observations, we assume the original hardware imperfection, $H^{(n)}$, of a device n, comes from a normal distribution $N(\mu, \Sigma)$ where $\mu = 0$ and $\Sigma = 9.12$ kHz are the global average and standard deviation of the original imperfection for all devices in the world. For each packet p, the attacker measures the imperfection $H_p^{(n)}$. $H_p^{(n)}$ has a normal distribution $N(H^{(n)}, \sigma^{(n)} = 320Hz)$ centered around the original imperfection of the device. $\sigma^{(n)}$ represents the measurement noise which depends on factors such as SNR of the received signal, and the resolution and accuracy of the algorithm used by the attacker to measure the imperfections. Note that only $H^{(n)}$ is an identifying signature of the device, and $\sigma^{(n)}$ is not considered as a characteristic of the device itself.

4.2. Analysis metrics and assumptions

We define a metric that represents how well the attacker can possibly de-obfuscate the device's fingerprint. We borrow concepts from the framework proposed by Standaert et al. [46] for analyzing side-channel key recovery attacks and defenses. To analyze the effectiveness of the obfuscation strategy, we can ask two questions: (1) How good is the obfuscation strategy in hiding the original hardware imperfection identity of the transmitter? and, (2) How successful can the attacker be in beating the obfuscation strategy and identifying the transmitter? The first question is about how much information is still leaked after obfuscation, and can be measured with *conditional entropy*. The second question is about how successfully this information can be turned into an attack, and can be measured with the concept of *attacker's success rate*. We choose the success rate as our metric since the eventual goal is to see how well we can prevent the attackers from identifying the devices.

4.2.1. Attacker Success Rate. The goal of hardware imperfection obfuscation is to reduce the attacker's success in identifying the devices based on their hardware imperfections. The common metric to evaluate such attacks is how accurately the attacker can identify the newly received packet (or a group of packets) and decide if the target is the transmitter. In fact, if Y is the random variable representing the ground truth identity of the physical device ($Y = 1$ if the target device is the transmitter and $Y = 0$ otherwise.), and the output of the attacker's decision in identification phase is a random variable Z, the attacker goal is to maximize $Pr(Y = Z)$. Hence, we borrow the notion of *Attacker's success rate* from [46] and define it as follows:

$$Success = Pr(Y = Z) = 1 - Pr(Y \neq Z) \tag{1}$$

This simply represents the probability of identifying the device correctly by the attacker.

4.2.2. Optimal Attacker. The attacker may make their decision (Z) about the identity of the device based on the received packets from the target during fingerprinting phase and a decision rule. The decision rule is any algorithm that the attacker uses to decide if the fingerprints corresponding to one or multiple packets belong to the target and thus, identify the target—e.g., different machine learning classifiers. The choice of decision rule affects the attacker's success rate. Wang et al. [52] consider the maximum accuracy that any attacker can potentially achieve to evaluate website fingerprinting obfuscation. The advantage of such metric is that we can ensure once the obfuscation is deployed, on average no attacker can achieve a better accuracy than what has been evaluated, no matter what algorithm they use for identification. In other words, it is a bound on how successful the attacker can be in de-obfuscating the obfuscation strategy. Hence, we follow a similar approach and use the maximum possible attacker's success rate as the metric to analyze hardware imperfection obfuscation. In Appendix A.1, we prove Maximum A Posteriori (MAP), $C(H'_p) = \max_n Pr(Y = n|H'_p)$, can achieve the maximum success rate (*the optimal attacker*). Thus, we analyze the obfuscation against an attacker who uses MAP as the decision rule to identify the devices.

We make the following assumptions about the attacker:

- The attacker has fingerprinted the original hardware imperfection of the target, $H^{(n)}$, without error.
- The attacker knows the global distribution of the imperfections of devices in the world.
- The attacker knows the exact obfuscation method and parameters that are being used.

Having this information, the attacker can compute the optimal decision rule $C(H'_p) = \max_n Pr(Y = n|H'_p)$ by computing the posterior probabilities and achieve the optimal attacker's success rate. This makes it unnecessary to compare the obfuscation against any other attacker in hardware imperfection fingerprinting literature that uses the imperfections studied in this paper to identify devices. In fact, on average, the optimal attacker would do better than any other attacker who uses a different decision rule—e.g., classical machine learning classifiers [8], [54], distance/statistical classifiers [19], [15], [31], [53], [1], neural networks [26], [43], [36], [30]. Therefore, if obfuscation does well against this optimal attacker, it would do even better against other attackers. All the results and analysis presented throughout the paper is against the optimal attacker, unless otherwise specified. We compare the success rate of such an attacker to when there is no obfuscation, and also the optimal desired success rate (the oracle defense which results in random identification success rate of 0.5).

4.2.3. Analysis setup. The attacker's success also depends on the number of devices around and the uniqueness of target's imperfections. Hardware imperfection fingerprinting attacks are the most successful and threatening when there are only a few devices around, or the target device has a unique and rare imperfection (tails of the normal distribution). Our metric should also represent these two scenarios, to show the obfuscation is effective when fingerprinting attack is a serious problem. To this end, we analyze the optimal attacker's success rate under two circumstances.

First, we consider the attacker is identifying the target when there is only one other obfuscated device around. The reason is that if the obfuscation performs well in confusing the attacker in distinguishing even two devices, it will do even better when there are more devices around. We assume the imperfections of these two devices are randomly drawn from the global imperfection distribution. We call such devices *average devices*. Later in Section 7, we evaluate the obfuscation in practical situations in the field, where there are a different number of devices around of which some percentages are obfuscated and some not. Second, to demonstrate the effectiveness of obfuscation when the target has a rare imperfection, we assume the imperfection of the target device is drawn from the 5% tails of the distribution. We call such a device an *unlucky device* because it is more distinguishable compared to an average device. This demonstrates if the obfuscation helps with preventing the attacker from identifying even the most vulnerable devices.

5. Design and Analysis

Using the analytical framework from the previous section, we now incrementally design an obfuscation scheme that overcomes each of the practical challenges of obfuscating BLE devices (Section 3). We describe the design an obfuscation strategy for which the attacker will not be able to accurately identify the device, even after 24 hours of *continuously* observing the device. We choose 24 hours because it is unlikely for an attacker to *continuously* observe a target for a long time, and targets are unlikely to be stationary for more than a day. Additionally, we found that an attacker's improvement in accuracy over time becomes marginal after only a few hours. We describe the limitations that exist when choosing a random obfuscation value, and we analyze their effect on the success of obfuscation.

5.1. Obfuscation Interval

During a MAC address lifetime, the attacker can identify the device by only looking at the MAC address. Thus, randomizing the imperfections from packet-to-packet in a single MAC lifetime does not have any benefit. Moreover, if the imperfection randomization happens from packet-to-packet within a MAC lifetime, the attacker will have the opportunity to average the randomized imperfections and find the original imperfection of the device. Therefore, only when the MAC address lifetime expires and re-randomizes, the original imperfection H also needs to be re-randomized to a new obfuscated imperfection H', and remains the same for all packets in the next MAC address lifetime. Note that this does not require any changes to the MAC layer.

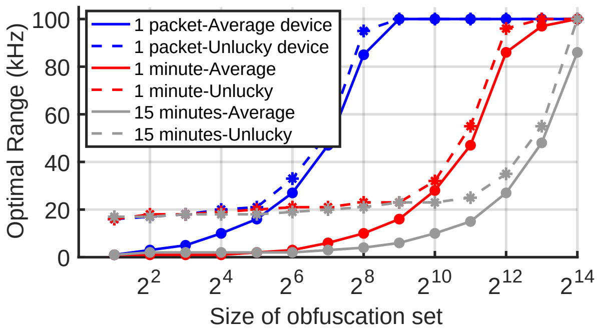

Figure 3: The optimal range for different sizes of obfuscation set (the optimal resolution can be calculated as $resolution = \frac{range}{size}$). We assume the attacker is listening to Contact Tracing beacons which typically transmit 4 packets per second.

5.2. Obfuscation Range

We start with basic discrete uniform obfuscation [37]. The random imperfection h^r is selected uniformly at random from the set $\{a + is\}, i = 0, 1, 2, ..., \frac{b-a}{s}$. The question is how we should choose the variables a, b (range) and s (resolution). Obviously, having a larger range provides better obfuscation. However, we must limit the obfuscation range according to the acceptable amount of hardware imperfection specified in the protocol standards. Specifically, BLE protocol specifies that CFO of the radio should be less than ± 150 kHz [42]. In our field data collection, we did not observe any CFO greater than 36.54 kHz. As a result, the maximum CFO obfuscation range that we consider in our analysis is $b = -a = 100$ kHz. We choose a symmetric obfuscation range ($b = -a$) because the distribution of imperfections is usually symmetric, and the hardware imperfection limits required by the protocol standards are also usually symmetric. Note that the Uniform distribution has the maximum entropy over such set.

5.3. Obfuscation Resolution

Having a finer resolution also provides better obfuscation as it causes the measured obfuscated CFO of different devices to be closer to each other, and thus overlap due to measurement noise. As we prefer both large range and high resolution, the question is as follows: With a fixed number of possible choices for the randomly added imperfection values—the *size of obfuscation set*—should we use a large range or a fine resolution? Figure 3 shows the optimal

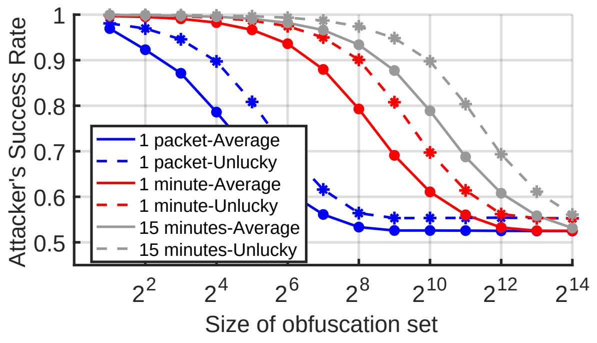

Figure 4: Attacker's success rate for different sizes of obfuscation set when the optimal range and resolution are used.

obfuscation range—the range that achieves the lowest attacker's success rate—for different sizes of obfuscation set. The attacker may receive multiple packets with the same MAC address during a MAC lifetime (15 minutes), and leverage all those packets to reduce the measurement noise. We show the graph when the attacker receives 1 packet, 240 packets (1 minute), 3600 packets (15 minutes). The optimal range increases rapidly as soon as the resolution becomes fine enough to get dominated by the measurement noise, after which a finer resolution does not significantly change the success rate.

We also observe that when the attacker receives more packets, we need a finer resolution because the measurement noise decreases. Therefore, the rapid increase in optimal range happens at larger sizes of obfuscation set. The same pattern holds true for both average and unlucky devices. The only key difference is even when the obfuscation set is small, we at least need to have a moderate obfuscation range to push the unlucky device close to other devices, and thus the optimal range is larger at the beginning. Figure 4 shows the attacker's success rate when the optimal obfuscation range is used. When the attacker receives l times more packets from a device, they can reduce the measurement noise by a factor of $\sqrt{l}$. Consequently, the resolution (and thus, the size of obfuscation set) should also be roughly $\sqrt{l}$ more to degrade the attacker's success to the same value.

Further, note that if the attacker listens to the device for the entire MAC lifetime (15 minutes), obfuscation can still hold them to the lowest possible success rate (0.5) if we have a resolution as fine as $\frac{200KHz}{2^{14}} = 12.2$ Hz. In Section 6, we show we can achieve 15.25 Hz resolution on a commodity BLE chipset. Therefore, we continue the analysis assuming the randomly added CFO is drawn from a continuous distribution, but with a limited obfuscation range. This analysis above is useful for implementations that have a more limited number of possible choices for the obfuscation set (e.g., hardware-based obfuscation [37]).

5.4. Fingerprinting duration

So far, we assumed the attacker is limited in how well they can defeat obfuscation because they only fingerprint packets with the same MAC address. However, attackers can also link consecutive randomized MAC addresses of a device together to refine their fingerprint over a longer period than one MAC address lifetime.

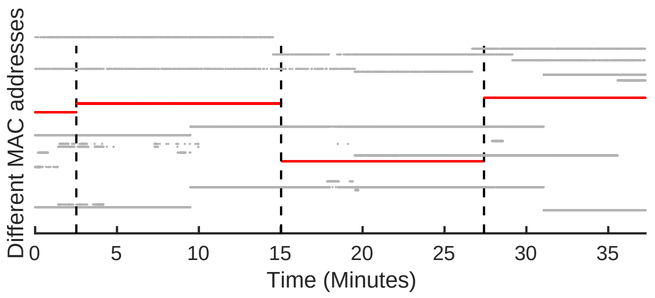

Figure 5: Each red dot is a packet transmitted by the target (Pixel 6 Pro) and gray dots represent other devices. Packets with the same MAC address have same vertical locations. The MAC address randomization events of a device are clearly observable (the target randomizes 3 times). The attacker can use this information to conclude several consecutive randomized MAC addresses actually belong to the same device.

First, we show that the attacker can refine their fingerprints of a device, by following its MAC address randomization through continuous observation. Figure 5 shows that when a particular MAC address disappears, and *immediately* a new one appears, the attacker realizes these two consecutive MAC addresses belong to the same device with a high probability. By repeating this logic, the attacker figures out all red packets belong to the same device, despite randomizing the MAC address three times. Now remember that the obfuscated imperfection corresponding to each MAC address lifetime is one new sample drawn from the obfuscation distribution. Consequently, if the attacker *continuously* observes the device, they know several MAC addresses belong to the same device; and they can refine the measured obfuscated imperfections for all those MAC addresses to de-obfuscate the device's fingerprint.

We evaluated the effectiveness of different obfuscation methods under this prolonged fingerprinting attack.

Continuous uniform distribution. Figure 6 shows how the attacker's success rate (See Appendix A.4) rapidly increases with uniform obfuscation as they observe the device for a longer time. The attacker obtains a new sample of the obfuscated CFO of the device in each MAC lifetime (15 minutes) and gradually breaks the randomness of obfuscation as they get more samples.

Gaussian distribution. Figure 6 shows that the attacker's success increases slower when Gaussian distribution is used compared to uniform obfuscation. In fact, even when the attacker's success for uniform obfuscation is lower than Gaussian at the beginning, it ends up higher than Gaussian after several MAC lifetimes (e.g., uniform with 100 kHz range vs Gaussian with 35 kHz standard deviation).

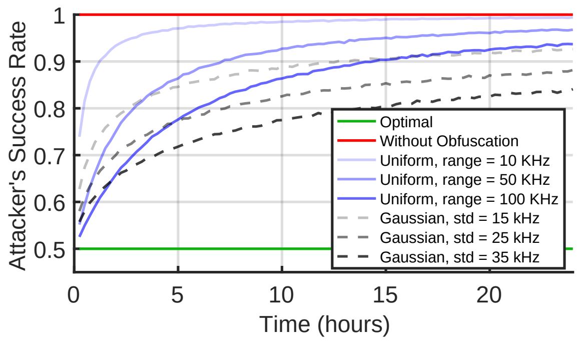

Figure 6: Attacker's success rate for Gaussian and uniform obfuscation for an average device, when the attacker continuously observes the device across multiple MAC lifetimes for hours. Attacker's success rate increases faster over time when Uniform obfuscation is used compared to Gaussian.

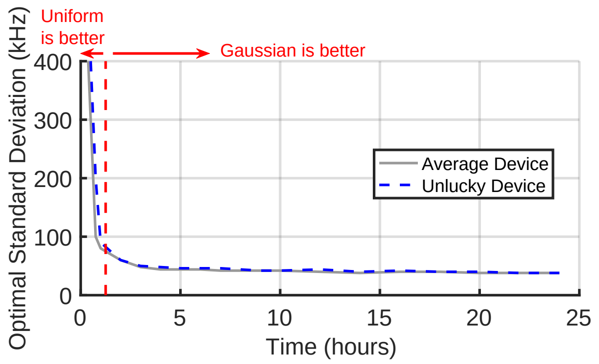

Figure 7: Optimal standard deviation of truncated Gaussian distribution when the attacker observes the device for different time durations, for both average and unlucky devices.

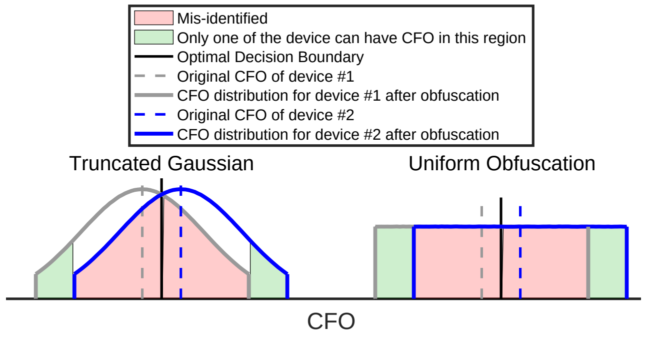
Mis-identified
Only one of the device can have CFO in this region
Optimal Decision Boundary
Original CFO of device #1
CFO distribution for device #1 after obfuscation
Original CFO of device #2
CFO distribution for device #2 after obfuscation
Truncated Gaussian
Uniform Obfuscation
CFO

Figure 8: Comparing Uniform and Gaussian obfuscation.

Truncated Gaussian distribution. Now the natural question is, which distribution should we use for obfuscation? First, note that we must limit the range of Gaussian distribution, so it does not exceed the CFO limits required by the protocol standards. Thus, we truncate the Gaussian probability density function by similar ranges as the uniform distribution. Assume h^r is drawn from a truncated Gaussian distribution $\tilde{N}(0, \sigma_o, b)$. Note that the truncated Gaussian distribution converges to the uniform distribution with range b as the standard deviation σ_o increases to infinity; and for smaller values it is similar to a regular Gaussian.

Figure 7 shows how the optimal standard deviation—the standard deviation of the truncated Gaussian that results in the lowest attacker's success rate—changes if the attacker observes the device for different time durations. As we can see, there isn't a single standard deviation that gives us the best obfuscation when the attacker observes the device for different time durations: If the attacker observes the device for one or a very few MAC lifetimes, a large standard deviation (closer to uniform) is better but as the attacker observes the device for a longer time, smaller standard deviations (closer to Gaussian) provides a better obfuscation.

Intuition. The intuition behind this phenomenon is depicted in Figure 8. The figure shows the distribution of the obfuscated CFO for two devices when uniform and truncated Gaussian distributions are used. The red area shows the probability that the attacker would misidentify the device in one MAC lifetime. This is larger for the uniform distribution as it assigns the probabilities equally across the common range of obfuscated CFO values of both devices, making the attacker completely unable to distinguish the devices. Thus, in short time durations we prefer uniform as the attacker would misidentify more. On the other hand, the green area shows the probability that the attacker can identify the device in one MAC lifetime with certainty (probability of 1). This is because only one of the devices can take the obfuscated CFO values in that region because of limited obfuscation range. This region is also larger for uniform distribution. Consequently, as the attacker observes the device for longer durations of time (more and more obfuscated CFO samples), the probability that at least one of them falls into that region increases quickly, and the attacker can identify the devices faster. Therefore, in that situation, a thin-tailed distribution like Gaussian is more favorable than uniform, in order to make the green region smaller.

5.5. Hierarchy of Obfuscation Distributions

We saw in Figure 6 that no matter what distribution we use, the attacker's success rate increases significantly if they continuously observe the device. In this section, we intend to propose a strategy to maintain the attacker's success much lower even if they continuously observe the device for as long as 24 hours, which essentially makes the identification attack impractical. Second, we observed that depending on how long the attacker observes the device, we may prefer a different obfuscation distribution. So we also intend to take advantage of the best of both uniform and Gaussian distributions to design a better obfuscation strategy. We propose the following obfuscation strategy to achieve both goals mentioned above:

- A random variable μ^r is re-sampled from a uniform distribution once every t^h MAC address lifetimes.
- A random variable h^r is re-sampled from a Gaussian distribution once every MAC address lifetime. The standard deviation of the Gaussian distribution is fixed, and the mean is the random variable μ^r.
- The resulting hierarchical distribution is truncated at ± 100 kHz so that the obfuscated CFO stays within the limits that the BLE standards impose.
- h^r is added to the original imperfection of the device as the random obfuscation value.

This algorithm is presented in Algorithm 1 (Appendix A.8) in more details. We explain the reasoning behind this strategy in the rest of this section.

5.5.1. Stalling the attacker. The attacker obtains a new sample from the obfuscated CFO distribution in each MAC lifetime, and gets one step closer to breaking the obfuscation and identifying the device. Now consider instead of re-randomizing the obfuscation every MAC lifetime, we re-randomize only every t^h MAC lifetimes. The attacker must wait t^h times more to obtain the same number of samples, and thus the same success rate. In fact, we stalled the attacker for t^h times more MAC lifetimes. Having a large t^h will make the attacker wait a longer time to achieve the same success rate. This phenomenon is represented in Figure 9. However, if $t^h = \infty$, this approach is equivalent to permanently shifting the CFO of the device by a fixed random value. Consequently, the new shifted CFO can be used as the fingerprint of the device. Therefore, although we want t^h to be large to stall the attacker more, we cannot make it arbitrarily large because the attacker can identify the device during those t^h MAC lifetimes. Note that although we only consider a fixed t^h in this work, t^h can also be randomized to make it even harder for the attacker.

5.5.2. Combining different distributions to get the best out of both. If during these t^h MAC lifetimes, the randomly added obfuscation value remains fixed, attacker can identify the device during that time. Hence, we need to also randomize the obfuscation during these t^h MAC lifetimes. In fact, we must have a hierarchy of two distributions, D_1 and D_2. Once every t^h MAC lifetimes, a random variable μ^r is sampled from the distribution D_1. For each MAC lifetime in those t^h MAC lifetimes, a random variable h^r is drawn from the distribution $\mu^r + D_2$ and used as the random imperfection obfuscation added to all packets in that MAC lifetime. The question is, how should we choose D_1 and D_2.

We saw in previous section (Figure 7) that if the attacker observes the device for a short time (e.g., less than 1 hour) and gets only a very few samples from the obfuscation distribution, uniform distribution is a better choice. Conversely, if the attacker observes the device for a longer time and gets many samples from the obfuscation distribution, Gaussian is a better choice. Now since t^h should be relatively large to stall the attacker more, D_2 should be Gaussian. On the other hand, since t^h is large, the attacker will get only a very few samples from the first distribution, D_1, as they have to wait t^h MAC lifetimes for each sample. As a result, D_1 should be Uniform. Hence, we use a hierarchy of Gaussian and Uniform distribution as our final obfuscation distribution.

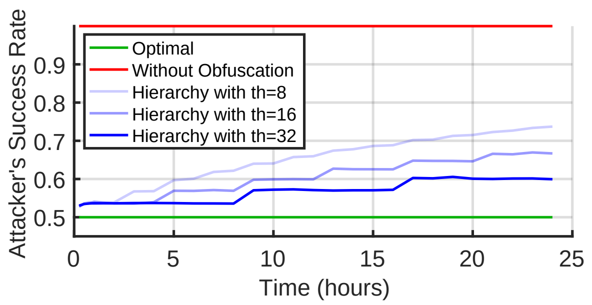

Figure 9: As we increase t^h, the attacker needs to observe the device for a longer time to achieve the same success.

5.5.3. Choosing the distribution parameters. We choose $t^h = 32 = 8 hours$. As we can see in Figure 9, this value is sufficient to keep the attacker's success rate close to optimal even after 24 hours of continuously observing the device, while not being too large that identifying the device during the t^h MAC lifetimes becomes an issue. Moreover, we choose the range of uniform distribution 70 kHz as we did not see any significant improvement by further increasing this parameter, and also leaving some room for the samples from the second distribution to vary on top of the first distribution. We choose the standard deviation of Gaussian 42 kHz by doing an analysis similar to what has been done in Figure 7. The final hierarchical distribution is truncated at 100 kHz to meet the range requirement.

5.5.4. Comparing different distributions. Figure 10 compares the proposed hierarchical distribution of Uniform and Gaussian to other distributions. The red curves represent the minimum attacker's success that we can possibly have at each time duration, if we use a truncated Gaussian distribution (for each time duration, we used the optimal standard deviation from Figure 7 to compute the corresponding success rate). We observe that even when the optimal standard deviation is used for a single truncated distribution, the attacker's success rate increases rapidly, specially for unlucky devices. The gray curves represent the hierarchy of two Gaussians and the blue curves demonstrate the hierarchy of uniform and Gaussian. The hierarchy of Uniform and Gaussian results in lower attacker's success. The reason is that by using hierarchy of distributions, the attacker only gets a very few samples from the first distribution, and we saw earlier that the Uniform distribution provides a better obfuscation than Gaussian in that situation. In summary, Figure 10 shows that the hierarchy of uniform and Gaussian is a better choice than any single uniform or Gaussian distribution as well as the hierarchy of Gaussians.

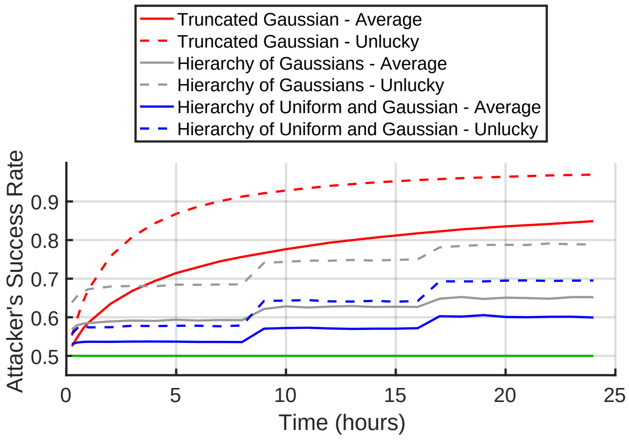

Figure 10: Comparing the attacker's success when different distributions are used for obfuscation.

6. Implementation

We demonstrate the feasibility of implementing our hierarchical CFO obfuscation strategy on two practical hardware platforms: a commodity BLE chipset (TI CC2640), and embedded DSP processors often used in radio chipsets.

Implementation on commodity hardware. Wireless radios can modify the CFO of their transmissions by changing the tuning of their frequency synthesizer, which is the hardware that generates the radio's center frequency. We discovered that a widely-used off-the-shelf BLE chipset—the TI CC2640 [48]—provides an API to control its frequency synthesizer. The CC2640 is prevalent in devices like fitness sensors, tags, and lighting systems. This CC2640 provides API commands to configure the radio's parameters. The frequency synthesizer control command (`CMD_FS`) allows us to set the center frequency of the radio transmitter. `CMD_FS` has two parameters: the BLE advertisement channel (*frequency*) and a fractional frequency offset that is added to channel frequency (*fractFreq*). We can simply use change the *fractFreq* value to add an offset and transmit actual BLE advertisements. *fractFreq* has 16 bits of resolution and a maximum frequency offset of 1 MHz: a resolution of 15.258 Hz. Other BLE chipsets likely also have vendor-specific commands similar to the CC2640. However, not all chipsets will reveal how to alter CFO in their public documentation. We also verified with another leading vendor of BLE chipsets that they have proprietary commands capable of introducing CFO variations.

DSP operations on the baseband signals. The core mathematical operations needed to add random CFO is to perform the following complex multiply with each baseband I/Q sample of a packet transmission:

$$y'(t) = y_o(t) \times e^{j(2\pi f t)}$$

where f, y_o denote CFO, baseband signal respectively.

Unfortunately, we were unable to demonstrate this modification on commodity wireless chipsets because their DSP are property and can not be modified by end users. Instead, we estimate the extra DSP resources that will be required to add this CFO modification. DSPs often already implement CFO correction for removing CFO from received signals. We investigated the Atomix DSP implementation of the WiFi baseband [5] on a TI Keystone 2 DSP. The frequency offset comes by multiplying each baseband sample with a pre-computed sin-wave lookup table. TI's Keystone architecture offers a 4-cycle instruction for complex number multiplication. We can compute sine values in constant time, once each time the host needs to change CFO frequency. We compared the time to the rest of the baseband packet generation and this extra complex multiply adds only a small constant factor.

7. Field Evaluation

In this section, we evaluate how well hardware imperfection obfuscation can confuse an attacker whose goal is to identify a target in practical scenarios. More precisely, in the *fingerprinting phase*, the attacker has fingerprinted the desired hardware imperfection of the target device (CFO) and has come up with a rule to decide if the target device is present upon receiving new packets in future during the *identification phase*. Unless otherwise noted the attacker is the optimal attacker that uses MAP to identify the target, similar to all the analysis so far.

7.1. Dataset

To evaluate the effectiveness of hardware imperfection obfuscation in confusing the attacker in practice, we use a large-scale dataset collected in the field. We collected BLE beacons from 1443 different devices at a public facility through which hundreds of people pass through during the day. These devices include personal mobile devices such as smartphones, smartwatches, wireless earphones, etc. The standard deviation of CFO across all these devices is 9.35 kHz. Note that this dataset is different from the one used to perform the analysis and design the obfuscation. Throughout this section, we use this BLE dataset for evaluation.

Ethical Considerations. We have discussed the details of this research with our IRB office and were told that it does not qualify as human subjects. We perform completely passive data collection of BLE advertisement packets using a SDR. We collect packets on only the BLE advertisement channels, and during the analysis stage we ensure we only use undirected advertisements (beacons). Most of these packets come from ubiquitous BLE applications such as contact tracing and device discovery. These packets contain no personal identifiable information, and the device fingerprints we compute cannot be linked to any individuals.

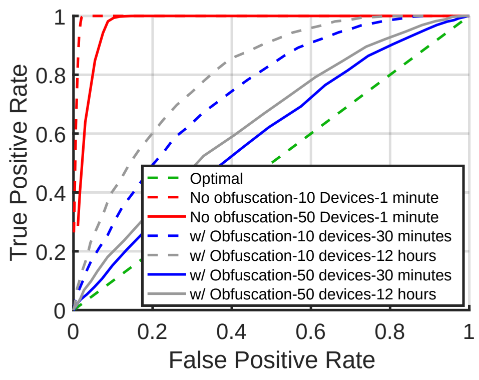

Figure 11: ROC curves of the optimal attacker for different numbers of surrounding devices and time durations.

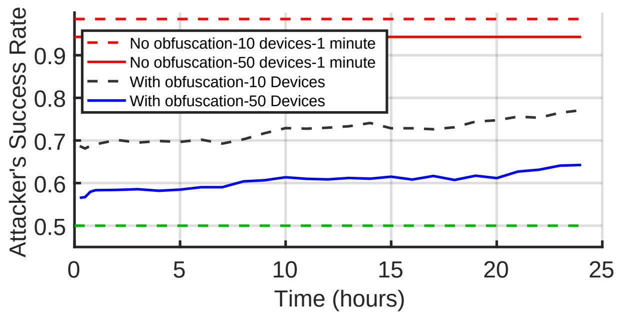

Figure 12: Success rate of the attacker as the time duration of observation increases.

7.2. Data Analysis

We estimate the CFO for each beacon packet. The original CFO of the device is the average CFO across all packets from same MAC address (same device). To evaluate impact of obfuscation, we consider a certain target device and a small varying number of other devices in the environment. We apply our obfuscation methodology to continually modify the CFO of these devices. Our obfuscation strategy uses a hierarchy of a continuous uniform distribution with 70 KHz range, and a Gaussian distribution of 42 kHz standard deviation (the choice of values is described in Section 4.) We also truncate the distribution at a point (*range of obfuscation*) which the CFO of no device in the dataset would exceed the BLE specifications on CFO limits (± 150 kHz). We then perform an identification attack on the target device, and analyze how the identification accuracy changes due to our obfuscation strategy.

In Section 4, we saw that obfuscating an *unlucky* device is significantly harder than an *average* device. Therefore, we only consider *unlucky devices* as target devices for our evaluation. In addition, for each result we perform 10000 tries where we select the target and surrounding devices at random. The reported result is an average of all these tries.

7.2.1. Target identification accuracy. The attacker performs a binary decision-making — identify whether a specific device is the target or not. If the attacker identifies the target when it is not present, a false positive occurs. If the target is present and the attacker does not identify it, a false negative occurs. We consider multiple configurations of total number of BLE devices, and time durations for which the attacker observes the devices. In configurations with obfuscation, we obfuscate 20% of all devices including target devices. Figure 11 shows the ROC curves which represents the True Positive Rate (TPR) at different False Positive Rates (FPR), in each of our configurations. We observe that without obfuscation, the attacker can identify the target with high accuracy in a short time (1 minute). However, with our obfuscation the ROC curves are closer to the ideal green line, indicating that even the strong optimal attacker is significantly more likely of confusing the target.

7.2.2. Time duration. We analyzed two different time duration configurations in Figure 11. In one configuration the attacker observes the target for a short 30-minute duration — at a public place such as a coffee shop. In another situation the attacker observes for a longer 12 hour duration — such as at their home or work location. We observed that the longer the attacker observes the target, the higher their success in identifying the target. However, obfuscation does significantly limit this increase in success rate.

Figure 12 shows the plot of attacker success rate vs observation time duration when there are 10 and 50 devices around of which 20% are obfuscated. For reference, we also show the success rate when there is no obfuscation and the attacker observes the target for a time duration of just 1 minute. Since the attacker makes a binary decision whether the observed fingerprint belongs to the target device or not, theoretically the best any obfuscation strategy can do is 0.5 (green line). In that oracle case of obfuscation, the attacker basically takes a random guess if the device is their target, and physical layer fingerprints cannot be used by the attacker to differentiate the target. We observe that when using our obfuscation, even after 24 hours of continuous observation, the attacker success rate is around 0.6-0.7 which is far from the unobfuscated situation, and close to the oracle obfuscation. This means that the fingerprints are no longer useful for the attacker to infer the identity of the device and the optimal attacker can barely do better than a random guess even after such a long time.

Further, since the attacker can improve its success over time, we ask how much time the optimal attacker needs to achieve the success rate they would achieve in one minute if the obfuscation was not used (red line). It turns out the obfuscated target must be stationary for more than 10 days so that the attacker can continuously capture BLE packets from them. As a result, the proposed obfuscation makes the attack impractical and significantly discourages the adversary from using hardware imperfections to deploy identification attacks.

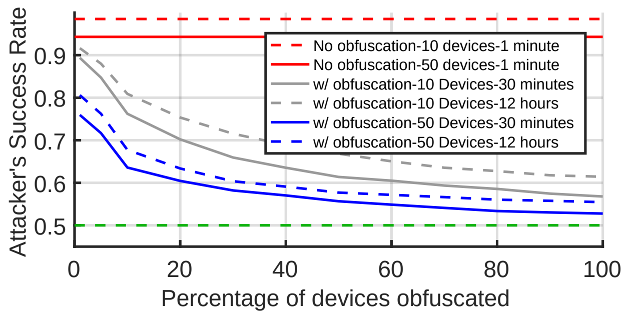

Figure 13: Success Rate of the attacker with varying percentage of obfuscated surrounding devices

7.2.3. Number of obfuscated surrounding devices. If many devices use obfuscation, then it becomes easier for any device to hide from an attacker. We analyze the impact of surrounding obfuscated devices in Figure 13. We plot the attacker success rate in different configurations of number of devices and time duration of observations. For each configuration, we vary the percentage of total devices that are obfuscated. We observe that the success rate approaches the ideal obfuscation as we increase the number of obfuscated devices. This means that as chipset manufacturers start deploying obfuscation, the benefits would keep increasing as more and more obfuscated radios start getting deployed.

7.2.4. Number of total devices. Similar to an increase in surrounding obfuscated devices, an increase in total devices around will also make it harder for an attacker to identify the target, since there is a larger pool of devices to distinguish the target from. We analyzed the impact of number of devices in Figure 11- 13. In particular, we considered two scenarios — the target device is at a home or office location (10 devices), and the target device is at a public place (50 devices). We observed in all the above analyses, the obfuscation significantly reduced the attacker's success in both scenarios and further, it was closer to ideal obfuscation when there were more devices around.

7.2.5. Case Study 1: Obfuscation against a practical weaker attacker. In the analysis thus far, we have considered a strong attacker that has all the details of our obfuscation strategy. However, a practical attacker may be significantly less informed. We evaluate how our obfuscation strategy performs against one such recent attacker as presented in prior work [19]. This attacker averages the CFO across multiple packets and computes Mahalanobis distance, which are compared against a threshold to decide if the received packets belong to the target. Figure 14 shows the ROC curves for the attacker in this situation. We use the same number of devices and time duration configurations as Figure 11. We observe that the obfuscation performs significantly better than our strong attacker situation, almost near optimal performance.

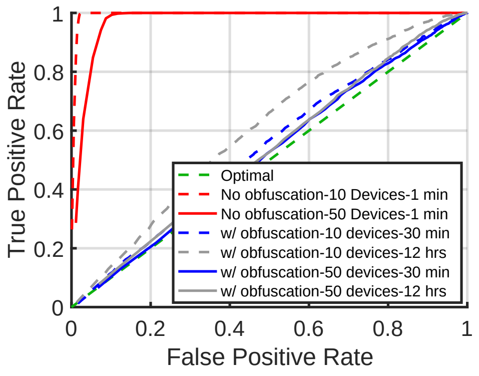

Figure 14: ROC the practical weaker attacker achieves for different number of surrounding devices and time durations.

7.2.6. Case Study 2: Obfuscating a CC2640 chipset. We implemented our obfuscation on a CC2640 chipset as explained in Section 6. Here, we perform a proof-of-concept experiment on this chipset. We set the device to transmit about 3600 BLE packets per MAC address lifetime. We receive the signals with an SDR and measure the CFO for each packet using [19]. Figure 15 shows the measured CFO when the obfuscation is on according to the strategy and parameters in Section 5.5. We observe that as expected, the chip randomizes its CFO after each MAC address lifetime and the CFO never goes beyond the desired limits. We can also observe the hierarchical structure of randomization as proposed in Section 5.5: The mean of the Gaussian distribution from which the random CFOs are picked, changes every 8 hours (shown in gray). As a result, the obfuscated CFOs are distributed around a different value every 8 hours. This prevents the adversary from identifying the device even if they observe it for prolonged durations. We ran both the optimal attacker and the weaker attacker to identify our obfuscated chipset after collecting 24 hours worth of packets. None of them were able to detect the device. We also ran the weaker attack when the obfuscation was off. The attacker was able to identify the device after receiving only 10 packets, while there were on average 19 other BLE devices around per MAC lifetime (15 minutes).

8. Discussion

8.1. Effect of obfuscation on demodulation error

Our obfuscation approach is evaluated with a limited range of CFO alterations (±100 kHz) to ensure compliance to the BLE standard [42]. However, we need to ensure BLE devices do not incur increased bit error rate due to our random CFO alterations. To test this, we measured packet reception and signal strength (RSSI) across a wide range of CFOs between two commodity BLE devices. We transmitted BLE advertisements on channel 39 at 100ms interval using a CC2640, and received them using a Pixel 8 Pro kept at fixed distance for 30 seconds at each CFO setting (-450

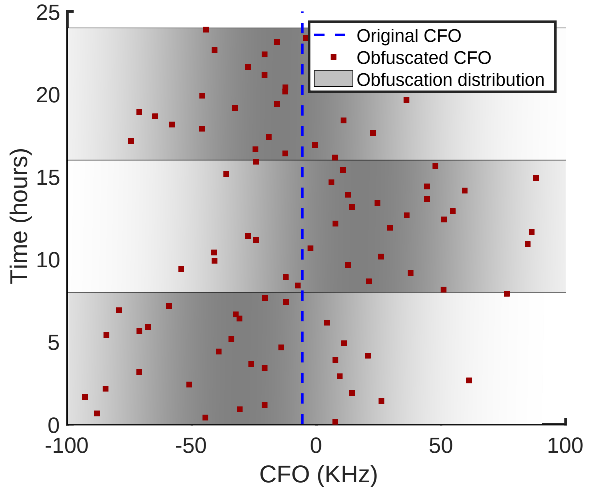

Figure 15: Obfuscated CFO measured for packets transmitted by a commodity TI CC2640 chipset over 24 hours.

KHz – 450 KHz). Figure 16 shows the number of received packets and average RSSI at different CFO values. We see that even large CFOs do not have a noticeable effect on the number of received packets and the RSSI. Even up to ±300 kHz—3 times the range we used in this study—there is no noticeable impact on the packet reception in real-world mobile devices. Receivers may tolerate large CFO offsets because they implement coarse CFO compensation before BLE's GFSK demodulation [47]. Beyond 300 kHz we do observe an expected loss in packets received, which may be due to channel filtering in the chipset. We also tested the receiver CFO tolerance of iPhone 8 and iPhone 10 and observed that at ±150 KHz they could successfully received all packets. These tests show that the packets obfuscated by our strategy can be easily received by real world mobile devices and should not introduce any demodulation performance issues.

8.2. Effect of obfuscation on power consumption

Low power consumption is a fundamental requirement of the BLE protocol. Thus, any obfuscation should not introduce significant additional energy consumption. Fortunately, altering the CFO of transmissions is inherently energy efficient. The hardware architecture of BLE chipsets allows for CFO adjustments without incurring any extra energy consumption. The reason is, BLE transmitters must be able to tune their center frequency when transmitting advertisements and scans on different channels. They achieve this by tuning of the chipset's frequency synthesizer. We simply tune the synthesizer to an offset from the channels' center frequency instead of the true center frequency. This does not add significant extra energy consumption because the synthesizer will inherently keep the same channel and CFO until it is retuned. Also, random CFO generation only happens only once every MAC lifetime (~15 minutes in tandem with the normal randomization for the MAC address). This is infrequent because BLE devices send hundreds of packets per minute [19], so it only happens once every few thousand packet transmissions. Furthermore, since many BLE transceivers use an on-chip CPU to control the radio, the CFO obfuscation can be implemented in the chipset's firmware itself, without incurring any additional energy cost to wake up the device's application processor.

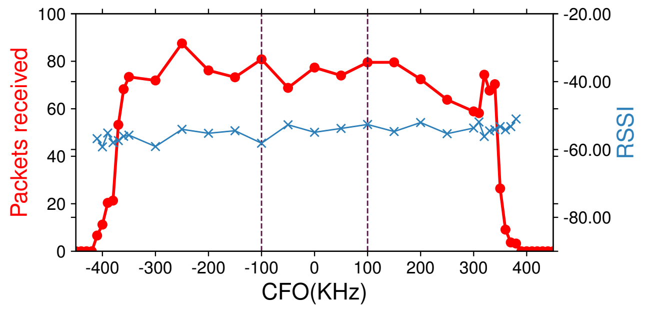

Figure 16: BLE performance across transmitter CFO settings. Dotted lines show the obfuscation CFO range.

8.3. Obfuscating other radio imperfections

In this work, we focused on obfuscating CFO imperfections because they are one of the most distinguishable hardware imperfections in wireless devices [8], [19], [53], [22]. However, there are other physical-layer fingerprinting attacks on BLE (and other protocols) for which we also do not have effective obfuscation strategies. For instance, another unique radio imperfection is the non-linearity of the transmitter's Power Amplifier (PA) [41]. The analysis framework in this work can be used to develop obfuscation strategies for these imperfections as well. The only requirement is that they must be able to be obfuscated by altering them randomly. Several such radio imperfections are listed in Table 1. The primary difference between obfuscating CFO and these other imperfections will be the parameters in the analysis such as the range and resolution of the imperfection, the standard deviation of the imperfection across many devices, and the measurement noise. The obfuscation strategy will likely remain the same.

We encourage vendors and researchers to adopt this analysis framework and obfuscation strategy when designing future chipsets that include physical-layer fingerprint obfuscation. hide user's identity.

8.4. Does obfuscation only work for BLE?

Our obfuscation strategy targets the underlying source of the fingerprint in RF chain. These hardware impairments impact all types of wireless chipsets such as WiFi, Zigbee and others. In fact, the BLE chipsets in many commodity smartphones shares the RF frontend with WiFi. Consequently, we expect our CFO obfuscation strategy can also be

TABLE 1: Radio hardware imperfections that can be obfuscated with methods similar to ours.

Hardware imperfection	**How is it measured?**	**Source of imperfection**	**How it can be obfuscated**	**Obfuscation Range**
I/Q offset	Physical layer [8], [19], [38]	DC offset in baseband or carrier leakage onto RF output	Adding random values to DC leakage, or I and Q components in the baseband signal	$EVM < -26dB(5\%)$ [24]
I/Q Imbalance	Physical layer [32], [19], [1]	Mismatch between I and Q paths	Adding random values to the logarithm of I and Q amplitude (multiplying gain mismatch between I and Q paths), and random delays between I and Q in the baseband signal	$EVM < -26dB(5\%)$ [24]
PA Non-linearities	Physical layer [31]	Non-linearities of the power amplifier	Adding random values to the smoothness parameter (for definition of this parameter refer to [41]).	3 ± 1 [31]
Clock Skew	Link layer (using TSF [3], [25] or TCP timestamp [27] for WiFi, and arrival time of preamble for Bluetooth [23])	Physical clock drift	Add random values to TSF and TCP timestamp fields for WiFi and randomize slot boundaries for Bluetooth. It can be obfuscated at physical layer only if the obfuscation is applied at the hardware level directly to the clock itself.	Bluetooth: $\pm 20ppm$ BLE: $\pm 50ppm$ [42] WiFi TSF timer: $\pm 100ppm$ [24]

applied on WiFi. To analyze this, we analyzed an additional dataset of WiFi probe packets from 4,673 devices in the wild. This includes devices like laptops, smartphones, and tablets both in indoor and outdoor conditions. We perform a detailed WiFi obfuscation analysis in Appendix A.7. In summary, we found that indeed WiFi transmitters can also be obfuscated effectively. However, implementing CFO obfuscation in WiFi chipsets may require modification of the WiFi DSP code. Fortunately, WiFi receivers include DSP for CFO correction. We would need to apply the same DSP processing to the transmitter side to inject CFO [5].

9. Related Work

Wireless devices can have a variety of physical-layer and link-layer identifiers that can be used for identification and tracking. For instance, although there have been attempts to randomize the MAC address, attacks have been found that can extract unique fingerprints from the bits in the link layer. Hence, we provide a brief summary of other physical-layer and link-layer fingerprints and defenses and how our physical-layer obfuscation fits in the greater landscape of wireless device fingerprinting.

9.1. Device identification and tracking

9.1.1. Physical Layer.

Hardware Imperfections: Different hardware imperfections have been used as a fingerprint to identify wireless devices, including but not limitted to CFO [8], [19], [38], [51], [53], [22], I/Q offset [8], [19], [38], I/Q imbalance [32], [19], [1], and PA non-linearities [31], [39]. Our work demonstrates it is feasible to obfuscate CFO, the most concerning and common of these physical-layer hardware imperfection fingerprints. The obfuscation analysis and methodology can likely be directly applied to these other imperfections.
Channel and Location: Wireless signals can be used to localize devices [56] using measurements such as channel state information (CSI) [29], [50] and received signal strength [16], [10], [44]. The location information can be used as a unique fingerprint to track a device, but it requires the device and environment to be static, unlike physical-layer imperfections that are independent of the location of the device and environment.

9.1.2. Link Layer.

Packet content: Wi-Fi and Bluetooth devices continuously transmit packets containing link layer information that can be used to derive device identifiers. For example, Wi-Fi devices can be identified using sequence numbers and fields such as WPS UUID and WPS IE [17], [49], [34], and BLE devices can be identified as payload changes asynchronous to the MAC address [6], or sequence number increments independent of MAC address randomization [33]. However, software updates can easily fix or change these identifiers, and many have already been addressed by such updates.
Packet timing: The link layer is also responsible for scheduling packet transmissions. Packet timing techniques measure the specific timing properties and use that information as a fingerprint to identify the transmitters. These techniques typically either measure clock skew (drift from ideal timing caused by imperfections in physical clock) using TCP timestamp [27] or TSF timestamp fileds [3], [25], or meaure inter-packet arrival time [16], [35]. Randomizing these fields as well as the timing of the periodic rate of packet transmissions in a hierarchical fashion like we describe in this work can potentially obfuscate these fingerprints at the link layer.

9.2. Physical-layer obfuscation

Recently researchers have proposed methods to obfuscate Channel State information (CSI) to prevent adversaries from localizing WiFi devices [13], [11], [18], [12], [4] or sensing human motions and activities [45], [40], [55]. The overall approach is to apply a random filter to the signal to manipulate peaks and phase changes, or add or move reflecting paths. While these works are similar to ours in that we both manipulate the transmitted signal with randomness, we investigate a completely different problem: obfuscating the hardware imperfection identity of devices. Our threat model, obfuscation strategy, challenges and analysis are completely different.

Prior research on radio frequency fingerprint obfuscation is limited to two studies [1], [37]. Abanto-Leon et al. [1] proposed randomizing the CSI phase to prevent using CSI for device fingerprinting (particularly [32] that uses CSI phase to estimate I/Q imbalance). However, they only analyze the ability of a specific attacker to recover the true phase value in each CSI sample, without analyzing how to practically and effectively obfuscate the underlying physical source of uniqueness as we do in this work for CFO. Further, unlike this work, they did not analyze and evaluate the effectiveness of obfuscation in preventing device identification by an optimal adversary (the attacker may still be able to identify a device even if they cannot recover the original fingerprint; for example, if the obfuscation adds a fixed random value to the fingerprint).

Nikoofard et al. [37] propose an analog approach to BLE imperfection obfuscation. They implemented an integrated circuit that introduces CFO randomization and evaluated their implementation in a lab environment against 20 BLE chipsets. Compared to this work, our obfuscation approach is much more practical because it can be implemented by simply modifying firmware on existing commodity BLE chipsets, without the need to design new chipsets. Additionally, we demonstrate in detail how our approach can effectively hide a target device in a wide range of real-world scenarios. In fact, our work is complementary to theirs as we analyze how one should design the obfuscation strategy, and our obfuscation analysis can be used to evaluate how much the limitations of analog CFO modification (e.g., poor CFO stability) affect the ability to effectively obfuscate a device.

10. Conclusion

In this study, we introduced a framework for analyzing defenses against physical-layer identification attacks on wireless devices. With this framework, we developed a practical defense against the fingerprinting of the Carrier Frequency Offset, the most distinct physical layer fingerprint unique to all wireless devices. Our results showed that this defense strategy can disrupt the most advanced attackers that know our obfuscation methods. Specifically, this defense lead to a significant reduction in the accuracy of an attacker determining whether a device has the same fingerprint as their target. Without obfuscation, the accuracy was nearly 100% with just one minute of fingerprinting the target. However, with obfuscation applied, the accuracy dropped to only 10–20% better than a random guess (not much better than a coin flip), even after 24 hours of continuously observing the target. This defense can be rolled out incrementally, requiring only software modifications on a widely-used Bluetooth Low Energy (BLE) chipset. Moreover, we expect that similar implementations can be tailored for other BLE chipsets or even other protocols, particularly if they provide software-based control over their frequency synthesizer, or if they construct their baseband transmissions with a programmable Digital Signal Processor.

11. Acknowledgements

We would like to thank our shepherd and the anonymous reviewers from IEEE S&P for their insightful comments. We also thank Nadia Heninger for her helpful comments. This work was supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) via [2021-2106240007], National Science Foundation grants NSF-2232481, SaTC-2239163, and NSF-2213689, and a gift from Amateur Radio Digital Communications. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, NSF or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] L. F. Abanto-Leon, A. Bäuml, G. H. Sim, M. Hollick, and A. Asadi. Stay connected, leave no trace: Enhancing security and privacy in wifi via obfuscating radiometric fingerprints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2020.

[2] Apple Inc. Use Continuity to connect your Mac, iPhone, iPad, iPod Touch, and Apple Watch. https://support.apple.com/en-us/HT204681.

[3] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz. On the reliability of wireless fingerprinting using clock skews. In *Proc. ACM conference on Wireless network security*, 2010.

[4] R. Ayyalasomayajula, A. Arun, W. Sun, and D. Bharadia. Users are closer than they appear: Protecting user location from WiFi APs. In *Proc. International Workshop on Mobile Computing Systems and Applications (HotMobile)*, 2023.

[5] M. Bansal, A. Schulman, and S. Katti. Atomix: A framework for deploying signal processing applications on wireless infrastructure. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2015.

[6] J. K. Becker, D. Li, and D. Starobinski. Tracking anonymized bluetooth devices. In *Proc. Privacy Enhancing Technologies*, 2019.

[7] Bluetooth SIG. Bluetooth Technology Protecting Your Privacy. https://www.bluetooth.com/blog/bluetooth-technology-protecting-your-privacy/, Apr. 2015.

[8] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless device identification with radiometric signatures. In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2008.

[9] California Health Care Foundation. Preliminary Research suggests COVID-19 Warning App has slowed Transmission of the Virus. https://www.chcf.org/blog/preliminary-research-suggests-covid-19-warning-app-slowed-transmission-virus/.

[10] Y. Chen, W. Trappe, and R. P. Martin. Detecting and localizing wireless spoofing attacks. In *Annual IEEE Communications Society Conference on sensor, mesh and ad hoc communications and networks*, 2007.

[11] M. Cominelli, F. Gringoli, and R. L. Cigno. Non intrusive wi-fi csi obfuscation against active localization attacks. In *Annual Conference on Wireless On-demand Network Systems and Services Conference (WONS)*, 2021.

[12] M. Cominelli, F. Gringoli, and R. L. Cigno. Antisense: Standard-compliant csi obfuscation against unauthorized wi-fi sensing. *Computer Communications*, pages 92–103, 2022.

[13] M. Cominelli, F. Kosterhon, F. Gringoli, R. L. Cigno, and A. Asadi. IEEE 802.11 CSI randomization to preserve location privacy: An empirical evaluation in different scenarios. *Computer Networks*, 2021.

[14] B. Danev and S. Capkun. Transient-based identification of wireless sensor nodes. In *Proc. ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2009.

[15] B. Danev, D. Zanetti, and S. Capkun. On physical-layer identification of wireless devices. *ACM Computing Surveys (CSUR)*, pages 1–29, 2012.

[16] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee. Identifying unique devices through wireless fingerprinting. In *Proc. ACM conference on Wireless network security*, 2008.

[17] J. Freudiger. How talkative is your mobile device? an experimental study of wi-fi probe requests. In *Proc. ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2015.

[18] L. Ghiro, M. Cominelli, F. Gringoli, and R. L. Cigno. On the implementation of location obfuscation in openwifi and its performance. In *Mediterranean Communication and Computer Networking Conference (MedComNet)*, 2022.

[19] H. Givehchian, N. Bhaskar, E. R. Herrera, H. R. L. Soto, C. Dameff, D. Bharadia, and A. Schulman. Evaluating physical-layer ble location tracking attacks on mobile devices. In *Proc. IEEE Symposium on Security and Privacy (SP)*, 2022.

[20] J. Hall, M. Barbeau, and E. Kranakis. Enhancing intrusion detection in wireless networks using radio frequency fingerprinting. In *Communications, internet, and information technology*, 2004.

[21] A. Heinrich, M. Stute, T. Kornhuber, and M. Hollick. Who Can Find My Devices? Security and Privacy of Apple's Crowd-Sourced Bluetooth Location Tracking System. *Proceedings on Privacy Enhancing Technologies*, 2021(3):227–245, 2021.

[22] W. Hou, X. Wang, J.-Y. Chouinard, and A. Refaey. Physical layer authentication for mobile systems with time-varying carrier frequency offsets. *IEEE Transactions on Communications*, pages 1658–1667, 2014.

[23] J. Huang, W. Albazrqaoe, and G. Xing. Blueid: A practical system for bluetooth device identification. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2014.

[24] IEEE. IEEE standard for information technology–telecommunications and information exchange between systems - local and metropolitan area networks–specific requirements - part 11: Wireless LAN Medium Access Control (MAC) and physical layer (phy) specifications. *Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, 2016.

[25] S. Jana and S. K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2008.

[26] T. Jian, B. C. Rendon, E. Ojuba, N. Soltani, Z. Wang, K. Sankhe, A. Gritsenko, J. Dy, K. Chowdhury, and S. Ioannidis. Deep Learning for RF fingerprinting: A massive experimental study. *IEEE Internet of Things Magazine*, 2020.

[27] T. Kohno, A. Broido, and K. C. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, pages 93–108, 2005.

[28] M. Köse, S. Taşcioğlu, and Z. Telatar. Wireless device identification using descriptive statistics. *Communications Fac. Sci. Univ. of Ankara Series A2-A3*, 2015.

[29] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti. Spotfi: Decimeter level localization using wifi. In *Proc. ACM Conference on Special Interest Group on Data Communication (SIGCOMM)*, 2015.

[30] H. Li, C. Wang, N. Ghose, and B. Wang. Robust deep-learning-based radio fingerprinting with fine-tuning. In *Proc. ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2021.

[31] Y. Lin, Y. Gao, B. Li, and W. Dong. Detecting rogue access points using client-agnostic wireless fingerprints. *ACM Transactions on Sensor Networks*, pages 1–25, 2022.

[32] P. Liu, P. Yang, W. Song, Y. Yan, and X. Li. Real-time identification of rogue WiFi connections using environment-independent physical features. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[33] J. Martin, D. Alpuche, K. Bodeman, L. Brown, E. Fenske, L. Foppe, T. Mayberry, E. Rye, B. Sipes, and S. Teplov. Handoff all your privacy–a review of apple's bluetooth low energy continuity protocol. In *Proc. Privacy Enhancing Technologies*, 2019.

[34] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown. A study of MAC address randomization in mobile devices and when it fails. In *Proc. Privacy Enhancing Technologies*, 2017.

[35] C. Matte, M. Cunche, F. Rousseau, and M. Vanhoef. Defeating mac address randomization through timing attacks. In *Proc. ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2016.

[36] K. Merchant, S. Revay, G. Stantchev, and B. Nousain. Deep learning for rf device fingerprinting in cognitive communication networks. *IEEE journal of selected topics in signal processing*, pages 160–167, 2018.

[37] A. Nikoofard, H. Givehchian, N. Bhaskar, A. Schulman, D. Bharadia, and P. P. Mercier. Protecting bluetooth user privacy through obfuscation of carrier frequency offset. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2022.

[38] L. Peng, A. Hu, J. Zhang, Y. Jiang, J. Yu, and Y. Yan. Design of a hybrid rf fingerprint extraction and device classification scheme. *IEEE internet of things journal*, pages 349–360, 2018.

[39] A. C. Polak, S. Dolatshahi, and D. L. Goeckel. Identifying wireless users via transmitter imperfections. *IEEE Journal on Selected Areas in Communications*, 2011.

[40] Y. Qiao, O. Zhang, W. Zhou, K. Srinivasan, and A. Arora. {PhyCloak}: Obfuscating sensing from communication signals. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2016.

[41] C. Rapp. Effects of HPA-nonlinearity on a 4-DPSK/OFDM signal for a digital sound broadcasting signal. *ESA Special Publication*, pages 179–184, 1991.

[42] Y. Rekhter and T. Li. Core Specification 5.3. Technical report, Bluetooth SIG, July 2021.

[43] G. Shen, J. Zhang, A. Marshall, L. Peng, and X. Wang. Radio frequency fingerprint identification for lora using deep learning. *IEEE Journal on Selected Areas in Communications*, pages 2604–2616, 2021.

[44] Y. Sheng, K. Tan, G. Chen, D. Kotz, and A. Campbell. Detecting 802.11 MAC layer spoofing using received signal strength. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[45] P. Staat, S. Mulzer, S. Roth, V. Moonsamy, A. Sezgin, and C. Paar. Irshield: A countermeasure against adversarial physical-layer wireless sensing. *Proc. IEEE Symposium on Security and Privacy (SP)*, 2022.

[46] F.-X. Standaert, T. G. Malkin, and M. Yung. A unified framework for the analysis of side-channel key recovery attacks. In *Annual international conference on the theory and applications of cryptographic techniques*, 2009.

[47] W. Sun, J. Paek, and S. Choi. CV-Track: Leveraging carrier frequency offset variation for BLE signal detection. In *Proc. ACM Workshop on Hot Topics in Wireless (HotWireless)*, 2017.

[48] Texas Instruments. *CC2640 SimpleLink™ Bluetooth® Wireless MCU*, July 2016. SWRS176B.

[49] M. Vanhoef, C. Matte, M. Cunche, L. S. Cardoso, and F. Piessens. Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms. In *Proc. ACM on Asia conference on computer and communications security*, 2016.

[50] D. Vasisht, S. Kumar, and D. Katabi. Decimeter-level localization with a single wifi access point. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2016.

[51] T. D. Vo-Huu, T. D. Vo-Huu, and G. Noubir. Fingerprinting Wi-Fi devices using software defined radios. In *Proc. ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2016.

[52] T. Wang and I. Goldberg. {Walkie-Talkie}: An efficient defense against passive website fingerprinting attacks. In *Proc. USENIX Security Symposium*, 2017.

[53] J. Wu, Y. Nan, V. Kumar, M. Payer, and D. Xu. BlueShield: Detecting spoofing attacks in Bluetooth Low Energy networks. In *International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2020.

[54] W. Yan, T. Voigt, and C. Rohner. RRF: A robust radiometric fingerprint system that embraces wireless channel diversity. In *Proc. ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2022.

[55] Y. Yao, Y. Li, X. Liu, Z. Chi, W. Wang, T. Xie, and T. Zhu. Aegis: An interference-negligible rf sensing shield. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2018.

[56] F. Zafari, A. Gkelias, and K. K. Leung. A survey of indoor localization systems and technologies. *IEEE Communications Surveys & Tutorials*, pages 2568–2599, 2019.

Appendix A. Additional details

A.1. Attacker optimality: Maximum A Posteriori (MAP) maximizes the attacker's success rate

In this section, we prove the optimality of the Maximum A Posteriori (MAP) attacker used throughout the paper. In other words, we prove that an attacker who uses MAP as the decision rule, achieves the maximum success rate.

Bayes Risk or *Bayes Error* is the minimum possible average error one can achieve when applying a prediction rule. The Bayes risk of the attacker to classify the device, for the 0-1 loss, is defined as

$$\begin{aligned} R^* &= \underset{C}{inf} E\Big[L(Y, C(H_p))\Big] = \underset{C}{min} E\Big[I(Y \neq C(H_p))\Big] \\ &= \underset{Z}{min} E\Big[I(Y \neq Z)\Big] = \underset{C}{min} Pr(Z \neq Y) \\ &\rightarrow Maximum\, Success = 1 - R^* \end{aligned} \tag{2}$$

where $L(.)$ is the 0-1 loss

$$L(a,b) = \begin{cases} 0 & \text{if } a \neq b \\ 1 & \text{if } a = b \end{cases} \tag{3}$$

In fact, no attacker can achieve a success rate better than $1 - R^*$ for any classifier ($1 - R^*$ is the maximum possible success rate that one can achieve).

The rule that achieves the Bayes risk is referred to as the *Bayes Rule* (The Bayes rule is the classification rule that minimizes the average probability of error). It can be proven that Maximum A Posteriori (MAP) is the Bayes rule for the 0-1 classification loss:

$$\begin{aligned} R^* &= \underset{Z}{min} E\Big[I(Y \neq Z)\Big] = \underset{Z}{min} E_{H_p}\Bigg\{E\Big[I(Y \neq Z)\Big|H_p\Big]\Bigg\} \\ &\rightarrow z^* = C*(H_p) = \underset{z}{\text{argmin}}\, E\Big[I(Y \neq z)\Big|h_p\Big] \\ &= \underset{z}{\text{argmin}} \sum_{n=1}^{N} Pr(Y = n|H_p = h_p) I(z \neq n) \\ &= \underset{z}{\text{argmin}} \sum_{n=1}^{N} \Big(1 - Pr(Y = n|H_p = h_p)\Big) I(z = n) \\ &= \underset{n}{\text{argmax}}\, Pr(Y = n|H_p = h_p) \\ &\rightarrow R^* = 1 - E\Big[\underset{n}{\max} Pr(Y = n|H_p)\Big] \end{aligned} \tag{4}$$

Thus, MAP—$\underset{n}{\max} Pr(Y = n|H_p)$—achieves the maximum success rate ($1 - R^*$). This means that no attacker can achieve a better success rate no matter what decision rule or classifier they use. Hence, if the obfuscation is effective against such an attacker, on average it will do as good or even better against other attackers who use a different decision rule.

Example of computing MAP rule by the attacker: To use MAP as the decision rule, the optimal attacker needs to be able to compute posterior probabilities $Pr(Y = n|H_p)$ to obtain $\underset{n}{\max} Pr(Y = n|H_p)$ and identify the target. The knowledge of the attacker outlined in Section 4.2 enables the attacker to do that. For example, when there is no obfuscation, we can write (assuming the probability of receiving a packet from each device is equal)

$$\begin{aligned} Pr(Y = n|H_p = h_p) &= \frac{Pr(H_p = h_p|Y = n)Pr(Y = n)}{\sum_{j=1}^{N} Pr(H_p = h_p|Y = j)Pr(Y = j)} \\ &= \frac{Pr(H_p = h_p|Y = n)}{\sum_{j=1}^{N} Pr(H_p = h_p|Y = j)} \\ &= \frac{G(h_p, H^{(n)}, \sigma^{(n)})}{\sum_{j=1}^{N} G(h_p, H^{(j)}, \sigma^{(j)})} \end{aligned} \tag{5}$$

Similarly, we need to compute $Pr(Y = n|H'_p = h'_p)$ when the obfuscation is applied. For instance, when the discrete uniform obfuscation is applied, we can write

$$\begin{aligned} Pr(Y = n|H'_p = h'_p) &= \frac{Pr(H'_p = h'_p|Y = n)}{\sum_{j=1}^{N} Pr(H'_p = h'_p|Y = j)} \\ &= \frac{\sum_i Pr(H'_p = h'_p|i, Y = n)Pr(i)}{\sum_{j=1}^{N}\sum_i Pr(H'_p = h'_p|i, Y = j)Pr(i)} \\ &= \frac{\sum_i G(h'_p, H^{(n)} + a + is, \sigma^{(n)})}{\sum_{j=1}^{N}\sum_i G(h'_p, H^{(j)} + a + is, \sigma^{(j)})} \end{aligned} \tag{6}$$

Note that it is possible to achieve the maximum success rate only if the attacker precisely knows the obfuscation method and distribution that is used, as well as the original hardware imperfection distribution and measurement noises. In fact, the reason for making assumptions about the capabilities and knowledge of the attacker was to have the best possible attacker that can achieve the highest possible success rate.

A.2. Discrete Uniform - Multiple Packets

Consider the discrete uniform obfuscation is used. If the attacker gets l packets $\{p_1, p_2, p_3, ..., p_l\}$ with the same MAC address, assuming the measurement noise for each packet is identical and independent, we can compute the attacker's optimal success rate by computing (See equations 2,4,5,6)

$$\begin{aligned} &Pr(H'_p = \{h'_{p_1}, h'_{p_2}, h'_{p_3}, ..., h'_{p_l}\}|Y = n) \\ &\quad = \prod_{j=1}^{l} Pr(H'_p = h'_{p_j}|Y = n) \\ &\quad = \prod_{j=1}^{l} \sum_i Pr(H'_p = h'_{p_j}|i, Y = n)Pr(i) \\ &\quad = \prod_{j=1}^{l} \sum_i G(h'_{p_j}, H^{(n)} + a + is, \sigma^{(n)})Pr(i) \end{aligned} \tag{7}$$

A.3. Continuous Uniform - Multiple Packets

Consider the continuous uniform obfuscation is used, and the attacker gets l packets $\{p_1, p_2, p_3, ..., p_l\}$ with the same MAC address. The success rate of the optimal attacker is calculated according to Section 4.2 and by replacing $Pr(H'_p = \{h'_{p_1}, h'_{p_2}, h'_{p_3}, ..., h'_{p_l}\}|Y = n)$ in equation 6 as follows:

$$\begin{aligned} &Pr(H'_p = \{h'_{p_1}, h'_{p_2}, h'_{p_3}, ..., h'_{p_l}\}|Y = n) \\ &\quad = \prod_{j=1}^{l} Pr(H'_p = h'_{p_j}|Y = n) \\ &\quad = \prod_{j=1}^{l} \int Pr(H'_p = h'_{p_j}|h^r, Y = n)f(h^r)\, dh^r \\ &\quad = \prod_{j=1}^{l} \int_{-b}^{b} G(h'_{p_j}, H^{(n)} + h^r, \sigma^{(n)})\frac{1}{2b}\, dh^r \end{aligned} \tag{8}$$

A.4. Uniform - Multiple MAC Addresses

Assume the attacker has measured imperfections $\{h'_1, h'_2, h'_3, ..h'_t\}$ for t different MAC addresses corresponding to the same device over time. We use the equations in Section 4.2 to calculate the attacker's optimal success rate (or equivalently Bayes risk) over time as they get packets with different MAC addresses (and new obfuscated imperfections) every 15 minutes. For uniform distribution we can write

$$\begin{aligned} &Pr(H' = \{h'_1, h'_2, h'_3, ..h'_t\}|X = n) \\ &\quad = \prod_{i=1}^{t} Pr(H' = h'_i|X = n) \\ &\quad = \prod_{i=1}^{t} \int_{-b}^{b} G(h'_i, H^{(n)} + h^r, \sigma^{(n)})\frac{1}{2b}\, dh^r \end{aligned} \tag{9}$$

A.5. Truncated Gaussian Obfuscation - Multiple MAC Addresses

Assume the attacker has measured imperfections $\{h'_1, h'_2, h'_3, ..h'_t\}$ for t different MAC addresses corresponding to the same device over time. We calculate the optimal attacker's success rate for truncated Gaussian distribution according to Section 4.2 and by replacing $Pr(H' = \{h'_1, h'_2, h'_3, ..h'_t\}|X = n)$ in equation 6 as follows ($\tilde{G}$ is the truncated Gaussian function):

$$\begin{aligned} &Pr(H' = \{h'_1, h'_2, h'_3, ..h'_t\}|X = n) \\ &\quad = \prod_{i=1}^{t} Pr(H' = h'_i|X = n) \\ &\quad = \prod_{i=1}^{t} \int_{-b}^{b} G(h'_i, H^{(n)} + h^r, \sigma^{(n)})\tilde{G}(h^r, 0, \sigma_o)\, dh^r \end{aligned} \tag{10}$$

A.6. Hierarchy of Gaussian Distributions

Assume a hierarchy of Gaussian distributions $N(0, \sigma_1)$ and $N(0, \sigma_2)$ is used as the obfuscation method, and the attacker has measured imperfections $\{h'_1, h'_2, h'_3, ..h'_t\}$ for t different MAC addresses corresponding to the same device. The success rate of the optimal attacker, is calculated according to Section 4.2 and by replacing:

$$\begin{aligned} &Pr(H' = \{(h'_1, ..., h'_{t^h}), (h'_{t^h+1}, ..., h'_{2t^h}), ..., h'_t\}|X = n) \\ &\quad = Pr(H' = \{h'_{\lfloor \frac{t}{t^h} \rfloor t^h+1}, ..., h'_t\}|X = n)\times \\ &\quad \prod_{j=1}^{\lfloor \frac{t}{t^h} \rfloor} Pr(H' = \{h'_{(j-1)t^h+1}, ..., h'_{jt^h}\}|X = n) \\ &\quad = \int Pr(H' = \{h'_{\lfloor \frac{t}{t^h} \rfloor t^h+1}, ..., h'_t\}|X = n, \mu^r)f(\mu^r)\, d\mu^r \times \\ &\quad \prod_{j=1}^{\lfloor \frac{t}{t^h} \rfloor} \int Pr(H' = \{h'_{(j-1)t^h+1}, ..., h'_{jt^h}\}|X = n, \mu^r)f(\mu^r)\, d\mu^r \\ &\quad = \int \Big(\prod_{i=\lfloor \frac{t}{t^h} \rfloor t^h+1}^{t} Pr(H' = h'_i|X = n, \mu^r)\Big) f(\mu^r)\, d\mu^r \times \\ &\quad \prod_{j=1}^{\lfloor \frac{t}{t^h} \rfloor} \int \Big(\prod_{i=(j-1)t^h+1}^{jt^h} Pr(H' = h'_i|X = n, \mu^r)\Big) f(\mu^r)\, d\mu^r \\ &\quad = \int \Big(\prod_{i=\lfloor \frac{t}{t^h} \rfloor t^h+1}^{t} G(h'_i, H^{(n)} + \mu^r, \sigma_2)\Big) G(\mu^r, 0, \sigma_1)\, d\mu^r \times \\ &\quad \prod_{j=1}^{\lfloor \frac{t}{t^h} \rfloor} \int \Big(\prod_{i=(j-1)t^h+1}^{jt^h} G(h'_i, H^{(n)} + \mu^r, \sigma_2)\Big) G(\mu^r, 0, \sigma_1)\, d\mu^r \end{aligned} \tag{11}$$

A.7. Evaluating obfuscation for Wi-Fi devices

While the analysis in the paper was limited to BLE devices, the obfuscation strategies will also work for other protocols like Wi-Fi as the hardware imperfections like CFO are very similar. To demonstrate this possibility, we use a dataset of 4673 Wi-Fi devices, including phones, laptops, tablets and others. The wireless signal from these devices was collected across real-world indoor and outdoor environments. We estimate CFO for 100 frames from each device and use the average as the true CFO of the device. Figure 17 shows that the distribution of CFO for these WiFi devices also looks Gaussian like BLE devices. We repeat the same evaluation as in Section 7 on these 4673 Wi-Fi devices. To evaluate the impact of obfuscation, we consider a certain target device and a small varying number of other devices in the environment. We apply our obfuscation methodology with the same parameters as in Section 7, to

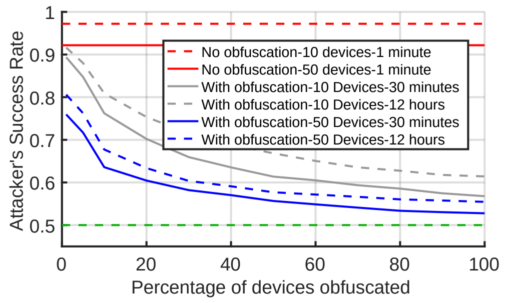

Figure 20: Success Rate of the attacker vs the percentage of obfuscated devices in the field.

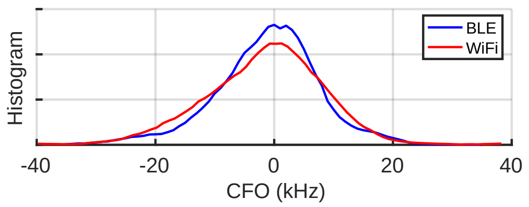

Figure 17: CFO distribution of BLE and WiFi devices.

continually modify the CFO of these devices. Figures 18, 19, 20 are similar evaluations for Wi-Fi devices as Figures 11, 12, 13. We observe that the results are very similar to the BLE dataset, demonstrating that the obfuscation strategy works well even for WiFi devices. While in these experiments for WiFi devices we used the same parameters as BLE devices, in practice we will need to retune them. Particularly, we will need to modify parameters such as obfuscation range to match WiFi standards and requirements [24], and repeat a similar set of analysis as in Section 4 to obtain the appropriate obfuscation parameters for WiFi devices.

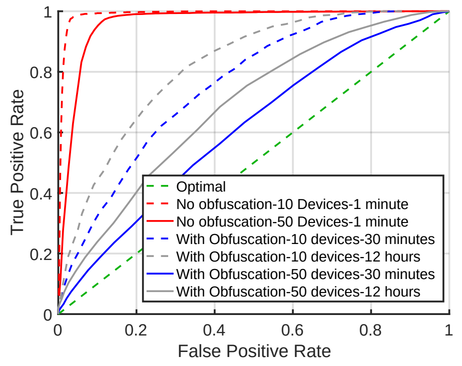

Figure 18: ROC that the attacker can achieve for different number of devices around and different time durations.

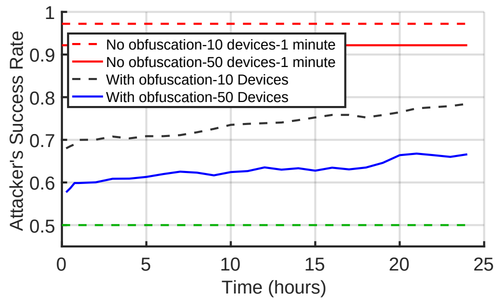

Figure 19: Success rate of the attacker across time when different number of devices are around.

A.8. Hierarchical Obfuscation Algorithm

Algorithm 1 Hierarchy of obfuscation distributions

```
Require: $t^h, b_1, b_2, \sigma_2$
  $t^h$: Number of MAC lifetimes to wait before resampling from the first distribution.
  $U(-b_1, b_1)$: Uniform distribution with the range of $\pm b_1$.
  $\tilde{N}(\mu^r, \sigma_2, b_2)$: Truncated Gaussian distribution with mean $\mu^r$ and standard deviation $\sigma_2$, truncated at $\pm b_2$.
  $Counter \leftarrow 0$
  $\mu^r \leftarrow U(-b_1, b_1)$
  while 1 do
      $h^r \leftarrow \tilde{N}(\mu^r, \sigma_2, b_2)$
      while MAC lifetime is not expired do
          Send packets with the added CFO set to $h^r$
      end while
      $Counter \leftarrow Counter + 1$
      if $Counter \geq t^h$ then
          $Counter \leftarrow 0$
          $\mu^r \leftarrow U(-b_1, b_1)$
      end if
  end while
```

Appendix B. Meta-Review

B.1. Summary

This paper investigates obfuscation schemes to mitigate radio hardware fingerprints caused by manufacturing imperfections by establishing a probabilistic model using real-world BLE beacons to capture the attack success rate based on different threat models and environments. It further proposes a hierarchical combination of different distributions for randomization. The implementation is done using SDR, and a commodity radio chipset. The evaluation shows the robustness of the proposed obfuscation scheme.

B.2. Scientific Contributions

- Addresses a long-known issue.
- Provides a valuable step forward in an established field.

B.3. Reasons for Acceptance

1) This paper addresses a long-known issue in the robustness of Bluetooth fingerprinting.
2) This paper proposes a mathematical model for capturing the robustness of fingerprinting and shows the effectiveness using an SDR-based PoC as well as a commodity radio chipset implementation.