Hindawi
Security and Communication Networks
Volume 2017, Article ID 6235484, 21 pages
https://doi.org/10.1155/2017/6235484

Research Article

Noncooperative 802.11 MAC Layer Fingerprinting and Tracking of Mobile Devices

Pieter Robyns, Bram Bonné, Peter Quax, and Wim Lamotte

Expertise Centre for Digital Media, UHasselt-tUL-imec, Wetenschapspark 2, 3590 Diepenbeek, Belgium

Correspondence should be addressed to Pieter Robyns; pieter.robyns@uhasselt.be

Received 5 January 2017; Accepted 21 March 2017; Published 25 May 2017

Academic Editor: Pascal Lorenz

We present two novel noncooperative MAC layer fingerprinting and tracking techniques for Wi-Fi (802.11) enabled mobile devices. Our first technique demonstrates how a per-bit entropy analysis of a single captured frame allows an adversary to construct a fingerprint of the transmitter that is 80.0 to 67.6 percent unique for 50 to 100 observed devices and 33.0 to 15.1 percent unique for 1,000 to 10,000 observed devices. We show how existing mitigation strategies such as MAC address randomization can be circumvented using only this fingerprint and temporal information. Our second technique leverages peer-to-peer 802.11u Generic Advertisement Service (GAS) requests and 802.11e Block Acknowledgement (BA) requests to instigate transmissions on demand from devices that support these protocols. We validate these techniques using two datasets, one of which was recorded at a music festival containing 28,048 unique devices and the other at our research lab containing 138 unique devices. Finally, we discuss a number of countermeasures that can be put in place by mobile device vendors in order to prevent noncooperative tracking through the discussed techniques.

1. Introduction

Techniques for tracking the whereabouts of IEEE 802.11 standard compliant (Wi-Fi) mobile devices can be employed for several benign use cases, such as positioning [1–4], asset tracking [5, 6], travel time estimation [7], and behavioral modeling [8]. These use cases inspired the creation of numerous startups and commercial products such as Skyhook (`http://www.skyhookwireless.com/`), Wifarer (`http://www.wifarer.com/`), YFind, and Nomi (`http://www.nomi.com/`), as well as a substantial amount of academic research. Wi-Fi based tracking is furthermore often chosen in favor of other technologies because of low battery consumption, low cost, passive monitoring capabilities of personal devices, and widespread presence of Wi-Fi chipsets, which is likely to further increase due to the current Internet of Things (IoT) trend. However, tracking systems can unfortunately also be used *maliciously* to noncooperatively or involuntarily disclose the location of mobile device users to an adversary, compromising their privacy.

From the perspective of the tracked device, location tracking systems can be divided into two categories: cooperative and noncooperative location tracking systems. In cooperative tracking, the mobile device is aware that it is being tracked and actively cooperates with the location tracking system to determine its correct location. In Android phones, for example, when the “high accuracy” location mode is enabled, the device will determine its own location by combining information from nearby Wi-Fi Access Points (APs), cellular networks, or the Global Positioning System (GPS) chipset.

On the other hand, in noncooperative tracking systems, the mobile device is unaware of the fact that it is being tracked, no existing infrastructure is modified, and the mobile device does not cooperate with the location tracking system (this type of tracking is referred to as infrastructureless terminal-based Location Fingerprinting (LF) in Kjærgaard’s taxonomy of location tracking technologies [9] or as “involuntary tracking” or “noncollaborative tracking” in other works). Instead, the location tracking system relies on radio transmissions captured by one or more Monitoring Stations (MSs) to track the location of a mobile device,

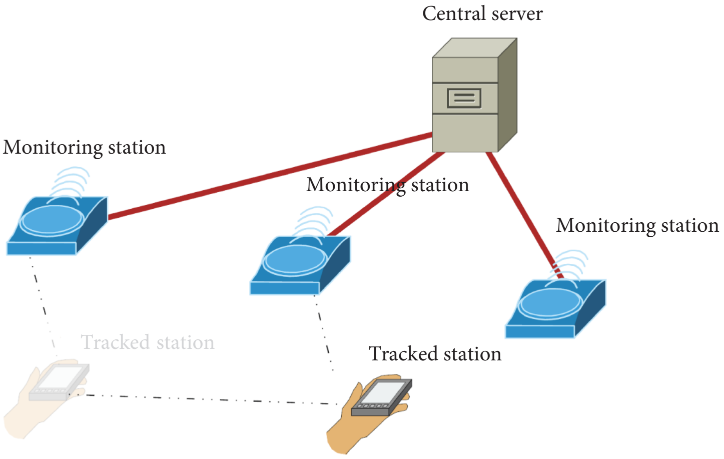


Figure 1: An example of a noncooperative position tracking topology. Several low to medium range Wi-Fi MSs with known positions are connected to a central server via a Local Area Network (LAN) or the Internet. The system can track a roaming target by intercepting signals transmitted by the device and mapping the detection to the location of the corresponding MS.

which also creates the opportunity for malicious exploitation by an adversary. A remote positioning system topology as described by Liu et al. [2] can, for example, be used for this purpose [7, 8, 10, 11].

In this work, we will study two aspects of noncooperative tracking: how mobile devices can be *fingerprinted* and *deanonymized* by a MS based on a single observation and how their *transmission frequency* can be artificially increased so that the number of observations by the MS is increased. We demonstrate a variety of techniques that exploit protocol design flaws and chipset implementation vulnerabilities to achieve these goals. Furthermore, we focus on tracking systems that operate on the data link layer (specifically, the Media Access Control (MAC) sublayer). These systems have the advantage of being deployable in software on commodity hardware as opposed to Physical (PHY) layer based tracking (see [12] and the references therein), where specialized equipment is needed. For higher layer (e.g., transport or application layer) based tracking, the device that is being tracked must be associated with a Basic Service Set (BSS), and the frame payload must be transmitted in plain text. This is rarely the case anymore at the time of writing, as according to WiGLE only 7.95% of wireless networks do not use encryption [13].

The structure of this paper is as follows. Section 2 entails general background information about the principles and topology used in noncooperative tracking. The role of fingerprinting in noncooperative Wi-Fi tracking is discussed as well. In Section 3, we examine the types of device identifiers present on the PHY and MAC layer. Furthermore, we introduce a novel technique that can be used to defeat MAC address randomization based on a single frame observation from the transmitting device. Section 4 will then discuss both novel and existing techniques for increasing the frame transmission frequency from tracked devices. The effectiveness of the discussed techniques is evaluated and compared against methodologies from earlier works in Section 5. A discussion of which countermeasures should be implemented by vendors is detailed in Section 6. In Section 7, we will discuss related work, and finally, conclusions and future work are discussed in Sections 8 and 9.

Our Contributions. We first perform a detailed entropy analysis of 802.11 MAC layer frame bits, specifically of Information Element (IE) bits contained within `Probe Request` frames. We then describe our novel technique, which identifies the frame bits that are the most useful for constructing a per-device fingerprint. To the best of our knowledge, this approach has not been considered before in previous works. Next, we demonstrate how an adversary can exploit this leakage of information to track users involuntarily, without changing the existing wireless network infrastructure and using only commodity hardware. We further demonstrate how this technique can be used to defeat MAC address randomization and deanonymize users.

Following our per-bit analysis, a second novel technique that can be applied to instigate extra transmissions from a tracked device is detailed. More specifically, we show that Hotspot 2.0 devices supporting the 802.11u standard or devices that incorrectly handle 802.11e Block Acknowledgements (BAs) frames can be actively probed, so that their presence is unwittingly exposed to a tracking system.

The aforementioned techniques are experimentally evaluated on two datasets, of which the first was recorded at Glimps 2015, a Belgian music festival in the city of Ghent, and the other at our research lab. Finally, we present several countermeasures against these noncooperative tracking techniques and discuss a number of interesting opportunities for future research.

2. Background

Recall that, in noncooperative tracking, the mobile device is unaware that it is being tracked and does not actively assist the location tracking system in determining its location. In this case, a topology as shown in Figure 1 can be used.

Here, the tracking system consists of several MSs spread out over a number of known and fixed geographical locations. Since the locations of these MSs are known, radio waves transmitted by a target device can be captured by the MS and roughly mapped to the MS's location [8, 11, 14]. In this scenario, the accuracy of the exact location of the tracked device depends on several factors, such as the gain of the receiving MS's antenna, carrier frequency, the transmit power of the tracked device, and environmental factors. A high range results in a higher probability to detect a device's transmission but will decrease the location accuracy and vice versa. If location accuracy is crucial for the application, other techniques such as Time Difference Of Arrival (TDOA) with multiple synchronized MSs could be used to improve the location accuracy [4, 15, 16].

The effectiveness of a noncooperative remote positioning topology depends on three criteria: a monitored device must be *uniquely identifiable*, its identifier must *remain stable over time*, and the identifier must be *transmitted at a minimal frequency*, preferably as often as possible.

Firstly, the unique identification requirement follows from the fact that the location tracking system must be able to differentiate between multiple observed devices. Here we can make the distinction between globally unique identifiers, which are permanently unique versus local identifiers, which are only unique for a limited duration or session. In literature, the study of uniquely identifying a device is commonly referred to as "fingerprinting." We will use this terminology from now on. In order to create a unique fingerprint, MSs must analyze externally observable properties from frames that are transmitted by the target device. When these properties are intended or designed to uniquely identify a device, they are called *explicit identifiers*. An example of an explicit identifier is the MAC address. On the other hand, if the ability to fingerprint a device from this property is unintentional, it is an *implicit identifier* [14]. These implicit identifiers are present in many shapes and forms on both the PHY and MAC layers of 802.11 frames, as will be shown in Section 3.

Secondly, stability of the identifier over time means that the identifier must remain either identical or correlatable to previous values for a sufficiently long duration. Indeed, the same device identifier must be detected by at least two different MSs before assumptions about the path traversed by the device can be made. A globally unique identifier, for example, is always stable.

Finally, the identifier must be transmitted as often as possible in order to increase the probability of detection. This is especially important if the target device is moving fast, since the device might move out of range even before it had the chance to transmit its identifier. We will henceforth refer to this property as the transmission frequency of the device.

3. Identifiers and Fingerprinting

Identifiers are important in fulfilling the requirement that a MS must be able to differentiate between multiple observed devices. Moreover, when explicit identifiers are unavailable (e.g., because of MAC address randomization), the MS can only rely on implicit information to construct a fingerprint for the observed devices. If these fingerprints are unique to such a degree that frames transmitted by a particular device can be linked together without explicit identifiers, the device is said to be deanonymized. In this section, we will explore earlier approaches to PHY and MAC layer fingerprinting. In Section 3.3, we will discuss our own methodology for constructing a fingerprint of a device without using explicit identifiers. Later, in Section 5.2.2, we will see how this approach can be used to deanonymize devices in practice.

3.1. PHY Layer Fingerprinting. Identifying a device based on PHY layer properties typically relies on the analysis of either a raw radio signal transmitted by the device or the baseband signal (e.g., for Wi-Fi, the Physical Layer Convergence Protocol (PLCP) data). Unfortunately for the tracker, these PHY layer properties cannot be easily extracted from a frame, since in conventional Wi-Fi chipsets the PHY layer is implemented in hardware. Therefore, this type of data is typically not accessible via the driver or firmware of the Wi-Fi chipset. A signal analyzer or Software Defined Radio (SDR) can be used instead, in order to capture the raw radio waves and demodulate them in software [23]. However, these types of hardware are more expensive, require more computing power than off-the-shelf Wi-Fi devices, are more sensitive to channel noise and interference, and require more complex pattern matching algorithms to derive a meaningful fingerprint from the radio signal. Several approaches based on the analysis of PHY layer properties have been proposed in earlier work.

Time Domain Analysis. In a time domain analysis of a transmitted signal, devices are classified using differences between the amplitude, the phase, and the frequency of the wave transients. This technique can identify a device given that its fingerprint is known to the classifier [21, 22].

Frequency Domain Analysis. Brik et al. compare small frame imperfections in the modulation domain with an ideal PHY frame modulation. However, a Support Vector Machine (SVM) classifier must be trained to recognize the device beforehand, using a set of 20 frames—MAC address pairs [23]. In the work presented by Corbett et al., a Power Spectral Density (PSD) analysis is performed, which captures the power or spectral density that a signal has over a range of frequencies. The PSD is calculated for each of the devices to be fingerprinted [24].

Clock Skew. Inherent drifts in the hardware clock of a device, caused by variations in the manufacturing process, can be measured and then utilized as a fingerprint [18–20]. However, in context of 802.11 this technique relies on timestamps extracted from `Beacon` and `Probe Response` frames, which are only transmitted by APs.

Scrambler Seed. The 802.11 scrambler is a Pseudo-Random Number Generator (PRNG), implemented as a 7-bit Linear-Feedback Shift Register (LFSR), which is XORed with the frame payload in order to obtain a uniform distribution of bit values. It is used in Orthogonal Frequency-Division Multiplexing (OFDM) modulation to improve the Peak-to-Average

Power Ratio (PAPR) and in turn to reduce the packet error rate. Bloessl et al. and Vanhoef et al. show that predictable scrambler states can be used to track mobile devices [25, 35]. However, this assumes that the scrambler seed is variable (this is not the case when Direct Sequence Spread Spectrum (DSSS) modulation is used, for example, when transmitting Probe Requests in the 2.4 GHz band [35]) and that the chipset vendor has implemented the scrambler so that it behaves in a deterministic, predictable manner. Furthermore, if the tracked device cannot be observed continuously, linking the scrambler states of two transmissions originating from the same device becomes increasingly difficult.

Observe that, for each of these PHY based approaches, the implicit identifiers of a device need to be determined in some preprocessing step [12], before the device can be uniquely detected reliably at a later stage. Furthermore, multiple observations are often required before a device can be correctly classified. These properties render PHY based approaches impractical for noncooperatively tracking devices at a large scale.

3.2. MAC Layer Fingerprinting. In contrast to PHY layer fingerprinting, MAC layer fingerprinting does not require raw radio signals to be analyzed in order to uniquely identify a device. Instead, one can use off-the-shelf Wi-Fi hardware such as a USB dongle to process the radio signal and analyze the resulting MAC layer data. For this reason, MAC layer tracking is particularly attractive to the industry, which according to Kjærgaard et al. desires systems that "have low maintenance, allowing positioning of all user devices, regardless of platform and form factor" [1]. The MSs used in MAC layer tracking systems are typically configured in *monitor mode* so that frames are forwarded to user space regardless of their destination MAC, and a Radiotap header with frame metadata is prepended to the packet.

3.2.1. MAC Address. A common identifier for fingerprinting a device on the MAC layer is the MAC address, which is disclosed by devices on a regular basis via transmitted frames, even when the device is not associated with an AP. Examples are `Probe Request` frames, which are transmitted by a client device in order to exchange IEs with the AP and detect their presence. The advantage of tracking a device based on the MAC address is that it is inherently a globally unique identifier per Network Interface Card (NIC), which explains why it is so commonly used in existing tracking systems [7, 8, 10, 11, 14, 30, 33].

However, tracking systems that solely rely on the MAC address as an identifier can be easily thwarted by employing MAC address randomization. With this feature, the device temporarily sets a random, locally unique MAC address before the active scanning procedure (i.e., before transmitting `Probe Requests`) [36]. This may be implemented by the device vendor in several ways, since the procedure is not explicitly stated in the 802.11 standard [37, p. 980]. For example, the MAC address can be randomized for each `Probe Request` or only once before association with an AP. Moreover, some implementations randomize the entire MAC address, whereas others keep the Organizationally Unique Identifier (OUI) part of the MAC address identical (in `wpa_supplicant`, the most popular 802.11 supplicant for Linux and the default supplicant for Android devices, this randomization behavior can be configured through the parameters `mac_addr`, `rand_addr_lifetime` and `preassoc_mac_addr` [38]). Random MACs should have their "locally administered address" bit set to one [37]. After the scanning procedure, that is, when the device is associating with an AP, the nonrandom MAC is typically reused in order to prevent MAC address collisions or network disruptions when roaming [36].

3.2.2. MAC Header Information. Despite the fact that MAC address randomization mitigates the issue that the MAC address can be used as a unique identifier, some devices can still be uniquely identified even when MAC address randomization is enabled. This can be accomplished by exploiting implementation quirks, for example, by correlating the 802.11 frame `Sequence Number` [31, 35, 39] or `Duration` field [32] if these fields are not randomized.

Several other fields and properties from the MAC frame header have been proposed in previous works as implicit identifiers, such as combinations of the frame size [14, 28], "`more fragments`," "`retry`," "`power management`," and "`order`" bits in the header, the authentication algorithms offered, and the used transmission rates [14]. However, the exact uniqueness and stability of these identifiers in context of devices in the unassociated state have not been studied before. In Section 3.3, we will demonstrate our methodology for determining the suitability of an identifier in an automated fashion.

3.2.3. Timing Variations. Differences between implementations or hardware can cause slight timing variations that can in turn serve as useful implicit identifiers. For example, the Timing Synchronization Function (TSF) in `Beacon` frames from an AP can be used to measure the hardware clock skew, which is different for each device due to variations in the manufacturing process [19, 20]. However, this timing information cannot be extracted from non-AP stations (STAs), since they do not transmit the TSF.

For non-AP STAs, timing differences between the observations of `Probe Request` frames have been considered to differentiate between implementations [10, 26, 27]. Even so, these timings are usually not unique per device, and they can be influenced by other factors such as whether the screen is on or whether the Wi-Fi chipset is in sleep mode [17]. A more generalised time based fingerprinting approach where timing information is extracted from the Radiotap headers of the MS was proposed by Neumann et al. [28]. Here, the medium access time, transmission time, and frame interarrival time are derived from the Radiotap header timing information in order to construct an implicit identifier.

3.2.4. Information Elements. IEs are Type-Length-Value (TLV) fields that are embedded in `Probe Requests` and `Beacon frames`. They contain information about the STA such as the supported data rates, Service Set Identifier (SSID), capabilities, vendor specific data and are therefore a plentiful source of implicit identifiers [35].

One example is the SSID IE, which is used in a `Probe Request` to indicate the SSID that is addressed. An SSID of length 0 can be used to indicate the wildcard (any) SSID [37, p. 478]. When a rare SSID, denominated in some works as a Personally Identifying Wireless Network (PIWN) [10], is observed or when multiple common SSIDs are observed in a particular order for a single device, the MS can build a fingerprint based on these observations [14, 30].

3.3. Per-Bit MAC Header Analysis. In each of the approaches from Section 3.2, a relatively small number of bits is chosen as the uniquely identifying information, and the remainder of the frame is discarded. Instead of manually defining which bits are suitable for fingerprinting a device and which are not, it would be interesting to *automatically* learn the suitability of a bit from a set of observations. We will now present our methodology, which aims to accomplish this goal.

In our methodology, a score is assigned to each bit of the MAC frame based on its potential as an identifier. The most suitable bits are then combined to construct a bitmask for each frame field. As such, new or rare fields used by only a handful of devices are automatically incorporated into the resulting fingerprint, independently of the header field order or underlying protocol semantics. If performance is critical, it is possible to perform the per-bit analysis on a small "training" subset and utilize the resulting bitmask to fingerprint the remainder of the dataset (see Section 5.2).

Our approach originates from the observations that we made in Section 2: that is, any set of field bits from an 802.11 frame could potentially be exploited to form a unique identifier, as long as the bits differ sufficiently across devices to become locally unique (bit variability) and remain consistent or at least related (for example, when using the sequence number from 802.11 frames, the value will not be identical in each transmission but it will be related to the previously transmitted frame) over the duration required by the tracking system (bit stability).

For the tracked device, we assume that it may randomize its MAC address for every transmitted frame; that is, MAC address randomization is enabled and correctly implemented. Therefore, our analysis requires only a single frame from the tracked device in order to construct a fingerprint. Moreover, we assume that the tracked device is in the unassociated state, causing it to only receive and transmit Class 1 frames. This class of frames will be discussed in detail in Section 4.1.

Let us now define what a "uniquely identifying bit" entails exactly. We define three metrics that will henceforth be used in our analysis: the *variability* or uniqueness of a bit, the *stability* of a bit, and the *suitability* of a bit.

3.3.1. Calculating the Variability. The variability of a bit is the entropy of that bit measured over multiple MAC frames, each of which is transmitted by a different device. Only one frame should be considered per device in order to prevent a "talkative" device from biasing the result, and only nonrandom MACs should be considered in this learning phase for the same reason. The goal of the variability metric is to indicate which bits provide unique contributions to the fingerprint.

To calculate the variability of the bit at position i in a frame, we will first calculate the discrete probability density function $P(X_i)$ for each bit value at position i. Let X_i be the random variable that represents a bit value. Then $X_i \in \{0, 1, U\}$, where a bit value of U denotes the absence of a bit. The probability density function can subsequently be used to calculate the Shannon (information) entropy:

$$H(X_i) = -\sum_{x \in \{0,1,U\}} P(X_{ix}) \log_3 P(X_{ix}). \qquad (1)$$

Note that we use a base 3 logarithm since we have a tristate bit instead of a conventional bit. The tristate bit is used to ensure that the absence of a bit is also counted as a different bit value. Now $H(X_i)$ will be a value between 0 and 1 where 0 indicates no entropy and 1 indicates maximum entropy. The result can be represented as a variability vector $\mathbf{v}$ per bit:

$$\mathbf{v} = \begin{bmatrix} H_1 & H_2 & \cdots & H_{n-1} & H_n \end{bmatrix}, \qquad (2)$$

where n is the number of bits in the frame and H_n is the entropy of bit n.

3.3.2. Calculating the Stability. We define the stability of a bit as one minus the entropy of that bit measured over multiple MAC frames transmitted by *the same* device. The stability gives an indication of how likely it is that a bit will stay the same over multiple transmissions by the same device. The goal of the stability metric is to exclude those bits that are associated with the same device but change frequently.

The stability for a certain bit of the frame transmitted by a device is calculated analogous to (1). The result can be represented as a stability vector $\mathbf{s}$ per device d:

$$\mathbf{s}_d = \begin{bmatrix} 1 - H_{d1} & 1 - H_{d2} & \cdots & 1 - H_{dn-1} & 1 - H_{dn} \end{bmatrix}. \qquad (3)$$

As opposed to the variability, a higher entropy is unfavorable as it means a lower bit stability. Hence, we measure the stability of bit i as $1 - H(X_i)$. Finally, recall that, contrary to the variability, we calculated the stability *per device*. Hence, the stability vectors $\mathbf{s}_d$ for each device are averaged into a vector $\mathbf{s}$, resulting in the final average stability vector per bit.

3.3.3. Combining Variability and Stability to Form a Fingerprint. Ideally, the bits used in our fingerprint should both be highly variable and highly stable. We will now discuss two possible approaches to combine these metrics into a new metric which we will define as the *suitability*, denoted as the vector $\mathbf{u}$. Both approaches can be used in practice, as we will discuss in the coming sections.

Probabilistic Approach. Since both the variability and stability are values in the interval $[0, 1]$, we can interpret them as probabilities of the bit being suitable for use in a fingerprint. Assuming that variability and stability are independent variables, we can then multiply the variability and stability to obtain the final suitability:

$$\mathbf{u} = \mathbf{v} \odot \mathbf{s}. \qquad (4)$$

For example, a bit with 1.0 variability and 1.0 stability is ideal and therefore has maximum suitability, whereas a bit with 1.0 variability and 0.0 stability is completely unsuitable for fingerprinting a device.

Filtering Approach. Considering that stability is a desired feature, another possibility is to only use the bit variability and completely exclude bits that have a stability less than a user specified threshold λ from the fingerprint.

$$\mathbf{u}_i = \begin{cases} \mathbf{v}_i, & \text{if } \mathbf{s}_i \geq \lambda, \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

For example, if $\lambda = 1$, this approach ensures that the bits that are combined in the fingerprint will always remain stable over time.

The threshold λ essentially determines the tradeoff between uniqueness of the fingerprint and stability of the fingerprint, which will be discussed in Section 3.3.5. Bits with suitability greater than zero are used in a bitmask, which is applied to each frame received from a device. The result of the mask operation is then concatenated and used as the final fingerprint.

3.3.4. Information Element Analysis. In Section 3.2.1 we briefly mentioned that IEs are exchanged between the AP and client STAs via `Probe Requests` and `Probe Responses`. These IEs contain information such as the supported rates, SSID, capabilities, vendor specific data and are therefore a plentiful source of uniquely identifying bits. It would be interesting to know which bits of which IEs are *the most* variable and stable. For that purpose, we can use the previously discussed methodology.

An issue that should be considered beforehand is that although IEs TLVs are usually transmitted in ascending order of the IE type, we have observed that some implementations transmit different orderings of IEs. If the order is disregarded, this would consequently cause the bit variability and stability to be derived from different IE types in some cases.

Therefore, to get an idea of exactly which bits per IE type contain uniquely identifying information, we need to compute their suitability *separately* per IE type. Additionally, the transmitted order must be incorporated back into the fingerprint, because the order itself is a piece of uniquely identifying information. We can achieve this by introducing a "dummy" IE type containing the IE order to the algorithm.

To get an idea of which IEs are the most suitable to be used in a fingerprint, we have performed our bit entropy analysis using a dataset of 200,394 `Probe Requests`. Table 1 shows the total variability and instability for each IE type. Figure 2(a) graphically shows the bit variability for the first 256 bits of each IE type. We can clearly see that the most variable bits come from the SSID IE and the `Vendor Specific` IE. For the SSID IE, also note that each first bit per byte of the SSID field has less entropy than the other bits. This is because all SSID strings that we observed were ASCII encoded. Other useful sources of uniquely identifying bits are the capability IEs: the capabilities of the device will be different depending on the used Wi-Fi chipset.

Table 1: An overview of the total bit entropy sum for each observed IE type.

Information element	Σv_i	$\Sigma 1 - s_i$
AP channel report	0.000	0.000
DS parameter set	0.625	0.411
Extended capabilities	32.790	0.061
Extended supported rates	28.373	1.716
HT capabilities	13.299	0.176
Information element order	40.327	2.529
Interworking	32.491	0.000
RSN information	0.000	0.000
SSID parameter set	87.570	30.590
Supported rates	22.529	1.317
VHT capabilities	10.424	0.000
Vendor specific	285.449	135.288

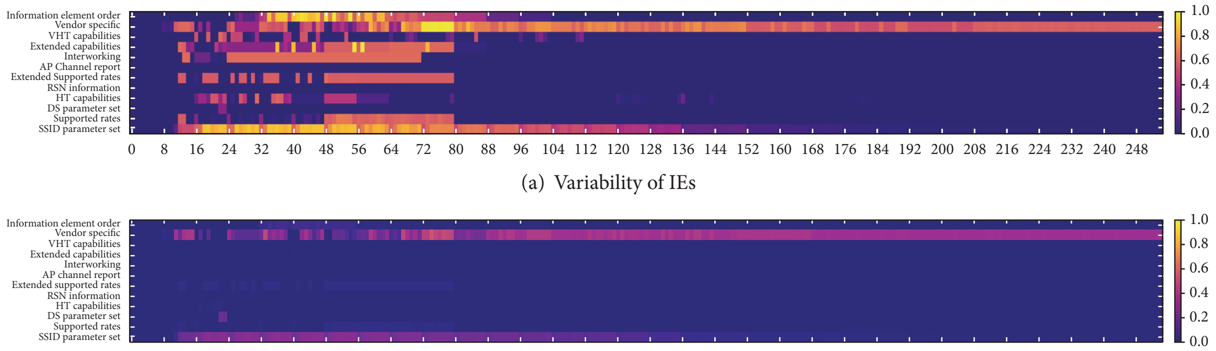
Figure 2(b) shows the bit *instability* (we chose to visualize the instability, since the majority of bits are stable, and plotting all stable bits would hence clutter the heatmap) for the same number of bits. As expected, the SSID IE is unstable because the SSID field in a `Probe Request` is often different for each individual probe. Likewise, a large portion of the `Vendor Specific` IE is very unstable, which means that these bits are unlikely to be useful in long term fingerprints. At the same time, some bits from the `Vendor Specific` IE are stable and highly variable, making them interesting uniquely identifying bits.

Note that some of the capability IEs have a very small amount of entropy. This is an interesting result, because we would not expect capabilities of the Wi-Fi hardware to change. We *believe* that some vendors modify the announced capabilities of the device in order to enforce some kind of behavior from the AP (e.g., announcing support for low data rates only, in order to improve reliability or range at the expense of data rate).

3.3.5. Discussion. The user defined threshold λ can be tweaked according to the needs of the location tracking system. If fingerprints that remain stable for an extended duration are required, λ can be increased. This will cause bits with stability above the threshold to be exclusively incorporated into the fingerprint, trading stability for variability in the process. The tradeoff between variability and stability is depicted graphically in Figure 3.

Figure 3 also shows that, even with $\lambda = 1$, the fingerprint stability is not perfect. This is because some devices transmit an additional IE (for example, a `Vendor Specific` IE) on occasion, resulting in two different fingerprints for a single device. However, even in this case random MACs used by this device can still be mapped to the original MAC address if both fingerprints are associated with the original MAC address at some point in time. An algorithm for associating a random MAC to the original MAC will be discussed in Section 5.

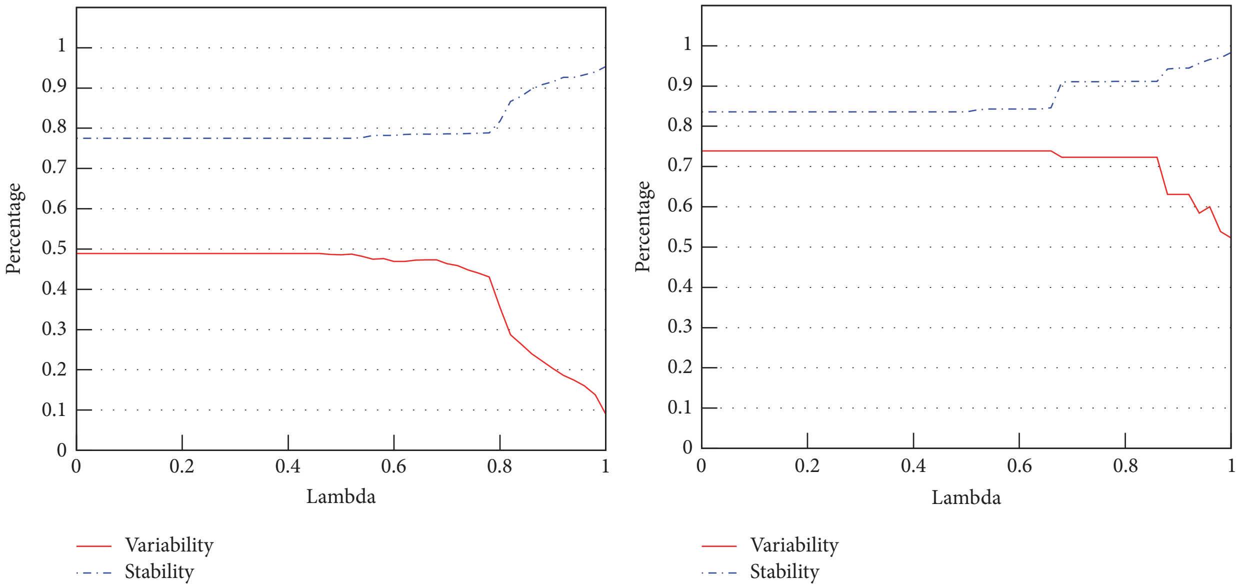
One might argue that, instead of an exact match of the fingerprint bits, a different metric such as the (weighted)



(a) Variability of IEs

(b) Instability of IEs

FIGURE 2: Graphical representation of the per-bit (x-axis) variability (a) and instability (b) for each observed `Information Element` (y-axis). Brighter colors indicate a higher entropy and thus a higher variability and instability.



(a) λ between [0, 1] for a large test set of 10,000 devices

(b) λ between [0, 1] for a small test set of 100 devices

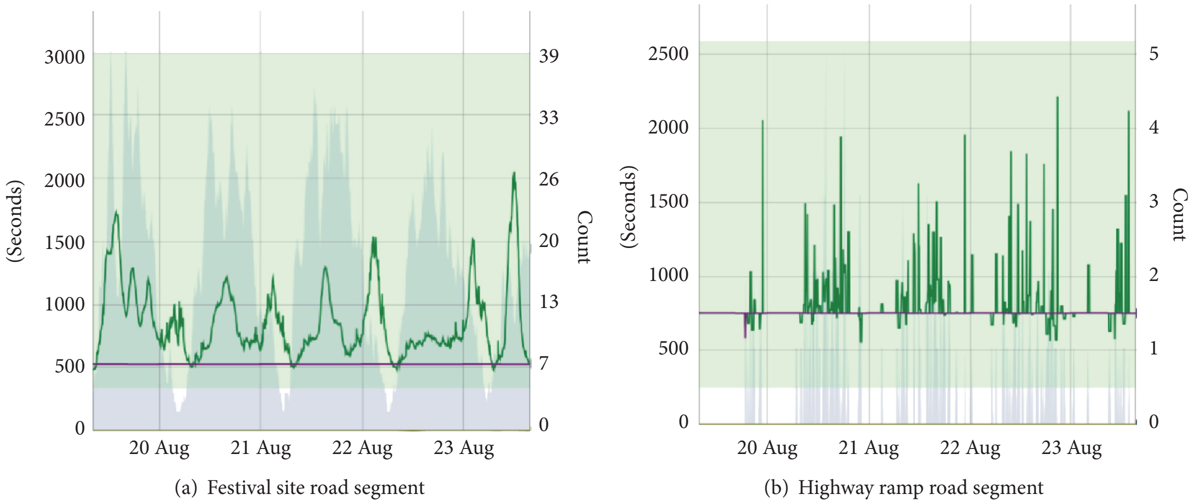
FIGURE 3: Graphical representation of the tradeoff between variability and stability, which can be controlled via λ. Incorporating more stable bits in the fingerprint increases stability of the fingerprint at the expense of fingerprint variability.

Hamming distance or Jaccard similarity coefficient of the bits could be used. However, observe that these metrics are unsuitable for *stable* bits: a stable bit will never change for a specific device. Hence, a change in this bit indicates that a different device is observed, and therefore a different fingerprint should be generated.

4. Transmission Frequency

For noncooperative location tracking, it is desirable that the tracked device transmits radio signals as often as possible. After all, a MS can only fingerprint an observed device when this device is both in range and transmits information that can be used for identification. A device that is moving and infrequently transmits might be “missed” by a MS positioned at a certain location. Moreover, devices might be unassociated and in sleep mode, rarely transmitting frames.

To further clarify the issue, consider the following example use case. In August 2015, we deployed a tracking system on the roads near Pukkelpop 2015, a popular music festival located in Kiewit, Belgium, in order to measure traffic congestion. Figure 4(a) shows the estimated travel time between two points on a road segment near the festival site. The travel time was determined by measuring the time difference between observations of identical mobile devices at the start and end points. Since vehicles were required to drive slowly on this segment, we were able to capture many mobile devices that were observed at both points, resulting in an accurate estimation of the true travel time. Fluctuations between day and night and the increase in congestion at the start and end of the festival can be observed.



(a) Festival site road segment

(b) Highway ramp road segment

Figure 4: Travel time in seconds for a road segment (green line). The purple line shows the travel time in optimal conditions as calculated by Google Maps. The blue graph in the background shows the number of device matches that were involved in the travel time calculation.

Figure 4(b), however, shows the same setup for a segment with a similar distance between the two points, but where the MSs are located near a highway entry and exit ramp. Here, cars were allowed to move faster. Since the transmission frequency of mobile devices located inside the cars remains identical, the probability of matching a device between the start and end points decreases, and so does the accuracy of the travel time estimation.

A possible solution for tracking such rapidly moving devices could be to increase the number of MSs on a road segment [11]. However, the cost of the location tracking system would increase, and the benefit would still be marginal if the device in question transmits its identifier infrequently.

4.1. Instigating Transmissions. To solve the problem of infrequent transmissions by nearby devices, a tracking system can actively try to instigate transmissions from devices in the vicinity. Here, the MS can craft frames which exploit a certain protocol mechanism [11] or vendor specific vulnerability as we will see in Section 5. As an additional benefit, the instigated response might further reveal details about the targeted device [34], which could serve as an implicit identifier. Thus, using this technique, the MS can increase the number of observed devices, increase the number of transmissions per device, and generate more bits as input to a fingerprinting algorithm at the cost of actively transmitting frames and becoming detectable. Let us now determine which frames are allowed to be transmitted and received by a device while in an unassociated state, by investigating the 802.11 standard.

The 802.11 standard defines 4 different states that can exist between a pair of STAs. Here, each state defines a class of MAC layer frames that may be exchanged by the transmitting and receiving STAs, as shown in Figure 5. From the MS's point of view, the most interesting frames to consider are *Class 1* frames: these frames do not require authentication or association with an AP in order to be exchanged between peer STAs and are almost always unencrypted (with the exception of `Self-protected Action` frames (used in mesh networks) and some management frames that are transmitted when the robust management frame service (802.11w) is enabled). A MS can utilize this class of frames to either sniff traffic between unassociated STAs or to inject arbitrary frames into the network.

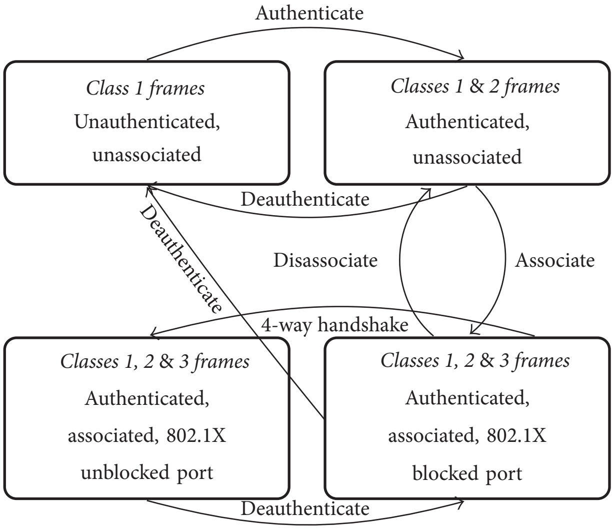


Figure 5: 802.11 nonmesh STA state transition diagram. Each state corresponds to one or more classes of frames that may be exchanged between two STAs [37, p. 1012].

It should be noted, however, that for a device to be able to receive Class 1 frames, it must be tuned to the same channel as the transmitter (since Wi-Fi channels overlap, it is sufficient to be tuned to an overlapping channel). A station will listen on a channel for at least `MinChannelTime`. If the channel is idle, it will switch to the next channel. Otherwise, it will wait until `MaxChannelTime` [37, p. 107] before switching to the next channel [24, 27, 40]. Thus, the MS could wait for the tracked station to switch to its own channel or alternatively transmit Class 1 frame on multiple channels at the same time. In the latter case, monitoring these channels is required as well in order to receive any responses to the stimulus frame, for example, by using multiple interfaces in monitor mode.

4.2. Beacon Frames. Among Class 1 Management Frames [37, p. 1013], a first type that we can utilize to increase the transmission probability of nearby devices is the Beacon frame. Many popular retail stores, hotels, bars, network operators, universities, and so on offer Internet access to their visitors in the form of (often open) wireless network. Such networks can be identified with, for example, the SSIDs "attwifi," "tmobile," and "eduroam." On the other hand, the popularity of SSIDs such as "linksys" and "dlink" can be attributed to the default vendor configuration of certain home APs.

In a tracking system, we can configure the MSs to spoof the aforementioned SSIDs by transmitting crafted Beacon frames matching the SSID *and* Robust Secure Network (RSN) configuration, that is, the security parameters, of the target network. Surrounding devices that automatically connect to known Wi-Fi networks and have one or more of these spoofed SSIDs in their Preferred Network List (PNL) will be more inclined to transmit frames because of automatic connection attempts [10, 11, 17, 33]. For example, Probe Request frames will be transmitted by the device before connecting in order to obtain the capabilities of the (fake) AP.

To increase the maximum number of SSIDs that can be spoofed and to conserve computing power on the MS, one can choose to only spoof Beacon frames and ignore Association Requests from surrounding devices. Further, the Beacon Interval field value can be increased so that Beacons do not need to be transmitted as frequently.

Similar to spoofing popular SSIDs, we can spoof Personally Identifying Wireless Networks (PIWNs) as well [10, 33]. Instead of composing a list of popular SSIDs, we can capture SSIDs from Probe Requests transmitted by a specific device and propagate this information to the other MSs. Hence, the goal here is not to trigger transmissions from as many surrounding devices as possible but to instigate transmissions from one particular device that is associated with this rare SSID. An example of this approach is graphically shown in Figure 6.

Lastly, Vanhoef et al. have recently demonstrated that, by including Hotspot 2.0 IEs in Beacon frames, Windows 10 and Linux clients will transmit Access Network Query Protocol (ANQP) requests using their original MAC address to request more information about the AP [35]. We will discuss Hotspot 2.0 and ANQP in detail in Section 4.4.2.

4.3. Control Frames. The Request to Send (RTS), Clear to Send (CTS), and Acknowledgement (Ack) Control Frames [37, p. 1012] form another type of Class 1 frames that we can use for instigating transmissions. The first two of these frames, RTS and CTS, are exchanged prior to data frames in order to distribute medium reservation information. A STA receiving either one of these frames can extract the period of time that the medium is to be reserved from the Duration field and consequently wait before transmitting in order to avoid collisions [37, p. 824]. As demonstrated by Musa and Eriksson [11], this mechanism can be exploited by transmitting fake RTS frames containing *the receiver's* MAC address as the Transmitter Address (TA) field of the RTS frame. The receiver will then respond with a CTS frame containing its own MAC address, revealing its presence.

Ack frames can be exploited in a comparable manner. According to the 802.11 standard, *each* data frame must be positively acknowledged (although technically, exceptions such as Block Ack frames, which can acknowledge multiple frames at the same time or a "No Ack" policy can be enforced by the transmitting STA) with an Ack frame. Therefore, most Wi-Fi chipsets have implemented the Ack mechanism in hardware as to ensure a timely acknowledgement of received frames. Now, consider what happens if we transmit a data frame with both the Receiver Address (RA) and TA fields set to the tracked device's MAC address. In this case, the receiver's hardware will simply copy the TA field from the data frame (its own MAC) into the RA field of the Ack frame, again revealing its presence.

A disadvantage of using these Control Frames for instigating transmissions is that one is required to know the receiver's MAC address. This might be problematic if it is frequently randomized, since a STA should not respond to its original MAC address after randomization. On the other hand, if the MAC address is randomized infrequently or if the device still responds to its original MAC address, this approach can be used to reliably instigate a response for every transmission.

4.4. Action Frames. Action frames are a type of frame intended for extended management functionalities [37, p. 449]. At the time of writing, there are 20 nonreserved "categories" of Action frames, each offering a different service. For example, Public (Category 4) Action frames are Class 1 frames intended for inter-BSS, intra-BSS, AP to unassociated STA, and Generic Advertisement Service (GAS) communications [37, p. 743].

As we saw earlier, a STA is allowed to transmit and receive Class 1 frames even when it is not a member of any BSS, which makes these frames interesting candidates for location trackers. The 802.11 standard states that, in this case, the "wildcard Basic Service Set Identifier (BSSID)" should be used, which is equal to the broadcast MAC address "ff:ff:ff:ff:ff:ff." In the following sections, we will discuss Block Acks, GAS, Spectrum and Radio Measurement frames, Tunneled Direct-Link Setup (TDLS), and Wireless Network Management (WNM).

4.4.1. Block Acks. In Section 4.3, we have seen that every 802.11 frame must be positively acknowledged on receipt. However, STAs with Quality of Service (QoS) support can choose to acknowledge multiple frames at the same time via Block Acks, reducing overhead. When two communicating STAs both support the Block Ack mechanism, the originator STA can set up a Block Ack exchange by sending an Add Block Ack (ADDBA) Request frame. The recipient STA must then accept or reject the request by responding with

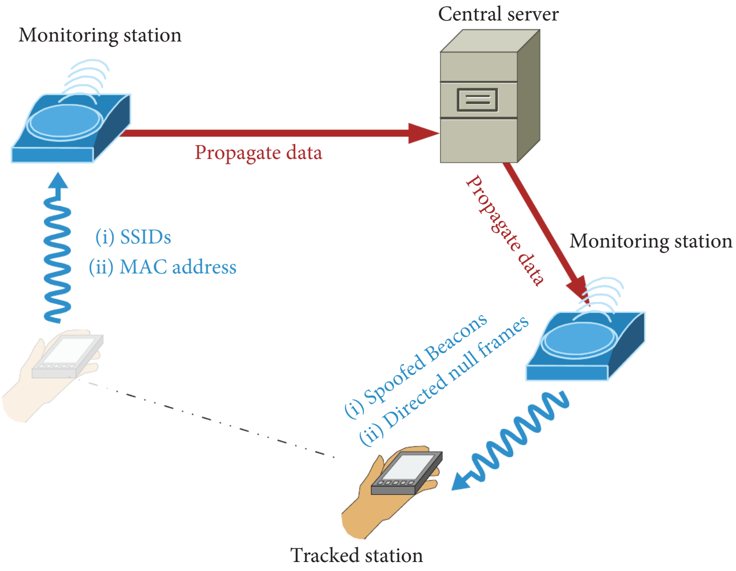


Figure 6: An example topology where data transmitted from the tracked station, such as its MAC address and SSIDs, are captured and propagated to other MSs. This information can subsequently be used to spoof PIWNs and to transmit directed Null Data frames, instigating more transmissions from the tracked station.

an `ADDBA Response` frame. Then, the originator can transmit multiple data frames followed by a `Block Ack Request`, which is acknowledged according to the `Block Ack` policy [37, p. 904].

Unlike normal Acks, `Block Acks` are not Control Frames, but a category of Action frames that was introduced in the 802.11e amendment [41]. An important consequence is that both the transmitter and receiver MAC address fields are present in the frame.

4.4.2. Generic Advertisement Service. In 2011, the IEEE introduced a set of Class 1 Public Action frames under the 802.11u "Interworking with External Networks" amendment. The Interworking amendment, as the name suggests, primarily focuses on enabling information transfer between 802.11 devices and Subscription Service Provider Networks (SSPNs). These are subscription based networks offered by a certain provider, such as cellular networks. The amendment additionally defines a number of new procedures for network discovery and selection, interaction with emergency services, and a QoS mapping from the SSPN's QoS settings to the 802.11 QoS mechanism [37, p. 78]. In 802.11u, an interesting candidate for instigating transmissions is GAS.

GAS allows a STA to discover network properties or services provided by a SSPN, such as whether the network provides Internet uplink. Here, the SSPN is the entity that validates the user's credentials and offers its services to the user through the AP. GAS can also be used to discover the services offered by a peer STA, for example, Wi-Fi Direct or P2P groups [42].

A STA can discover available services by embedding a GAS `Initial Request` inside an 802.11 `Public Action` frame and transmitting it to the peer STA or AP. Since GAS queries may be transmitted *before* association, they allow the mobile device to select the most suitable P2P group or AP before connecting. A unique byte value called the "Dialog Token" is used for matching requests with their corresponding responses in case several GAS Requests are transmitted concurrently.

For querying information from the peer STA (e.g., in case of an AP: whether the network provides Internet access), different advertisement protocols can be used. The default and mandatory supported advertisement protocol is ANQP. Upon receiving a GAS `Initial Request`, the peer STA replies with a GAS `Initial Response`. If the response is too large to fit in one frame, the remainder of the response is queried and delivered with, respectively, GAS `Comeback Requests` and GAS `Comeback Responses`.

With the objective of instigating transmissions in mind, GAS has a few interesting properties. First, a device that has Interworking enabled must support GAS, and GAS queries can be performed peer-to-peer in the unassociated state. Second, ANQP must be supported as the default advertisement protocol per standard definition [37, p. 1145]. ANQP queries can however contain different elements, each with a different purpose. As an example, the "Capability List" element contains the capabilities supported by a STA.

With this knowledge, MSs can create their own ANQP requests, embed them in a GAS `Public Action` frame, and broadcast these frames in order to obtain a GAS `Response` for each device in the vicinity that supports GAS. Since the frame can be broadcast and transmitted in the unassociated state, this technique has the potential to instigate transmissions on demand if the targeted device supports GAS.

4.4.3. Spectrum and Radio Measurement. `Spectrum` and `Radio Measurement` frames, defined in the 802.11k amendment, can be exchanged between a pair of STAs to determine the channel load, the received signal power, noise histograms, and so on. The type of measurement to perform is determined by the `Measurement Type` field of the `Measurement Request` frame.

According to the 802.11 standard, support for the `Basic Request` is mandatory and a STA in a BSS should only generate a `Basic Report` in response to a `Basic Request` if the request is received from the AP with which it is associated [37, p. 1048].

4.4.4. Tunneled Direct-Link Setup. A TDLS link can be set up between two peer STAs when they wish to use a feature that is not supported by the BSS itself, for example, a certain high throughput data rate. Frames transmitted in this fashion are said to be transmitted over the TDLS direct link.

Support for the TDLS protocol can be determined by inquiring the STA with a `TDLS Discovery Request`. If the targeted peer STA supports TDLS and if the BSSID is correctly set, it must respond with a `TDLS Discovery Response`.

4.4.5. Wireless Network Management. Another service introduced in 802.11e is WNM. This service is used for assorted management tasks such as requesting diagnostics from a STA, announcing that a STA will enter sleep mode, requesting channel usage information and so on. Two interesting candidates for instigating transmissions are the `Event Request` and `Timing Measurement Request` frames.

The former type can be used to request another STA to report one or more events, such as the `Peer-to-Peer Link` event [37, p. 777]. The latter type is used to synchronize a local clock time between two STAs [37, p. 1131], which could potentially serve as a useful implicit identifier in the context of tracking.

4.4.6. Wi-Fi Direct. A specification for facilitating peer-to-peer communication between two non-AP STAs, named "Wi-Fi Direct," was released to the public by the Wi-Fi Alliance in 2010 [43]. Instead of connecting to a real AP, a STA can take on the role of AP and form a peer-to-peer (P2P) group. Other STAs can discover these P2P groups using passive (`Beacons`) or active (`Probe Requests`) scanning mechanisms [42]. Since the user of a device explicitly needs to cooperate in order to configure the device to scan for P2P groups (in Android 6.0.1, for example, the user must go to the "Wi-Fi Direct" menu under the "Advanced Wi-Fi" settings to scan for P2P groups), we will not consider Wi-Fi Direct further.

4.5. Stimulus Frame Candidates. Of all stimulus frame types discussed above, GAS Requests seem the most promising for instigating transmissions, since unassociated peer STA to peer STA communication is explicitly allowed by the standard for this frame type. This is not the case for all frames we discussed, though it would be interesting to see how a device or chipset reacts when it receives these frames from an unassociated STA such as a MS. For example, `Measurement` frames, TDLS frames, and WNM frames can normally only be transmitted peer-to-peer when *both* STAs are associated with the same BSS. In Section 5.3, we will determine whether these constraints are respected by the device vendor's implementation, and under which conditions the discussed frames are accepted by the receiving STA. Furthermore, we will compare these frames in terms of their ability to increase the transmission frequency of nearby devices.

5. Evaluation

In order to quantify the effectiveness of our IE based fingerprinting technique and of our techniques for instigating extra transmissions from nearby devices, we have performed several experiments. We shall henceforth refer to these experiments as, respectively, the *fingerprinting* and *transmission rate* experiments. The used code and anonymized data sets have been made available publicly at `https://github.com/rpp0/wifi-mac-tracking`, `http://crawdad.org` [44], and `https://wicability.net` [45].

5.1. Attacker Model. We first define the following goals that an attacker attempts to accomplish. In the fingerprinting experiment, the attacker's goal is to uniquely identify as many devices as possible without relying on explicit identifiers. In the transmit rate experiment, the attacker attempts to instigate as many transmissions as possible from nearby devices. We make the following assumptions about the attacker and observed devices:

(i) At least one MS with an interface configured in monitor mode is used to track devices. To determine a trajectory, at least two MSs are required.

(ii) Observed devices may or may not be associated with an AP. The attacker cannot determine whether this is the case.

(iii) The attacker has a minimum amount of information at their disposal: we assume that, in the best case (for the attacker), only a single Probe Request frame is observed by the MS per device. Additionally, explicit identifiers such as the MAC address may be randomized for every transmitted frame.

(iv) Devices may appear or disappear from the tracking system's set of currently observed devices at arbitrary times. Consequently, techniques such as correlating the frame sequence numbers (Section 3.2.2) or scrambler seed (Section 3.1) cannot be used, since these values may have diverged or reset by the next time a device is observed.

Note that the above constraints are typical in noncooperative tracking scenarios, such as tracking the visitors at an event or tracking (smartphones located in) vehicles on the road.

5.2. Fingerprinting Experiments. For evaluating our *fingerprinting technique* (Section 3.3), we have set up a low-cost tracking system consisting of 8 commodity hardware MSs in a remote positioning topology (see Figure 1). For the MS, we have used MikroTik 5RB912UAG-2HPnD devices equipped with an AR9342 chipset (see Figure 10), though any device that supports monitor mode could be used for the same purpose. The stock firmware of the devices was replaced with OpenWRT Chaos Calmer (OpenWRT is a free Linux distribution for embedded devices and can be downloaded at `https://openwrt.org/`). Frames were captured using a custom application with `libpcap` and one interface in monitor mode.

The system was deployed at the Glimps 2015 music festival in Ghent, Belgium, which took place from 10 to 12 December 2015. The MSs were placed at the locations shown in Figure 7. Here, each MS was configured to forward one `Probe Request` frame per unique device to a central server over a secure link. Recall that we designed our fingerprinting technique with the constraint of having only one IE at our disposal in mind. In total, 51,975 `Probe Requests` were analyzed. No data frames were captured during the experiment.

5.2.1. Fingerprint Uniqueness. The effectiveness of our fingerprinting approach from Section 3.3 was evaluated by measuring the ratio of the number of unique fingerprints over the number of unique devices (MAC addresses). Ideally, there should be one fingerprint per device. Each fingerprint is solely based on the bits from a single `Probe Request`. Only nonrandom MACs (28,048 out of 83,055 MACs) were considered, so that the MAC address can be used as a baseline to compare the performance of the fingerprinting algorithm. Furthermore, incorporating random MACs in the analysis would overestimate the observed number of unique devices.

Figure 8(a) plots the overall fingerprint uniqueness for several test set sizes of nonrandom MACs for $\lambda = 0$, so that the variability is maximized and the stability is minimized. Figure 8(b) shows the same experiment, but with $\lambda = 1$ so that the variability is minimized and the stability is maximized. Recall from our discussion in Section 3.3 that the variability represents the uniqueness of the fingerprint and that the stability gives an indication of how likely a fingerprint will remain identical for a given device. It should be mentioned that these results still underestimate the fingerprint uniqueness for two reasons. First, a nonstandard compliant implementation may use MAC address randomization but not set the locally administered bit [35]. As a result, the same device will incorrectly be interpreted as a set of different devices with the same fingerprint, hence underestimating the uniqueness of this fingerprint. Second, although unlikely in practice, `Probe Requests` could be spoofed by an adversary in order to disrupt a tracking system.

From the results shown in Figure 8, we conclude that, for a MS that has observed a small dataset of 50 to 100 devices, the uniqueness of our fingerprint ranges from at least 80.0 to 67.6 percent. If the test set size is increased further, more devices with similar IEs will be encountered eventually, and the uniqueness of the fingerprint will therefore decrease. For large datasets of 1,000 to 10,000 devices, the uniqueness of the fingerprint drops between at least 33.0 and 15.1 percent. These results are encouraging, because a single MS will typically only observe a small set of devices, and the uniqueness for such small sets is high. In Section 9, we describe how the fingerprint uniqueness for large sets of devices could be increased.

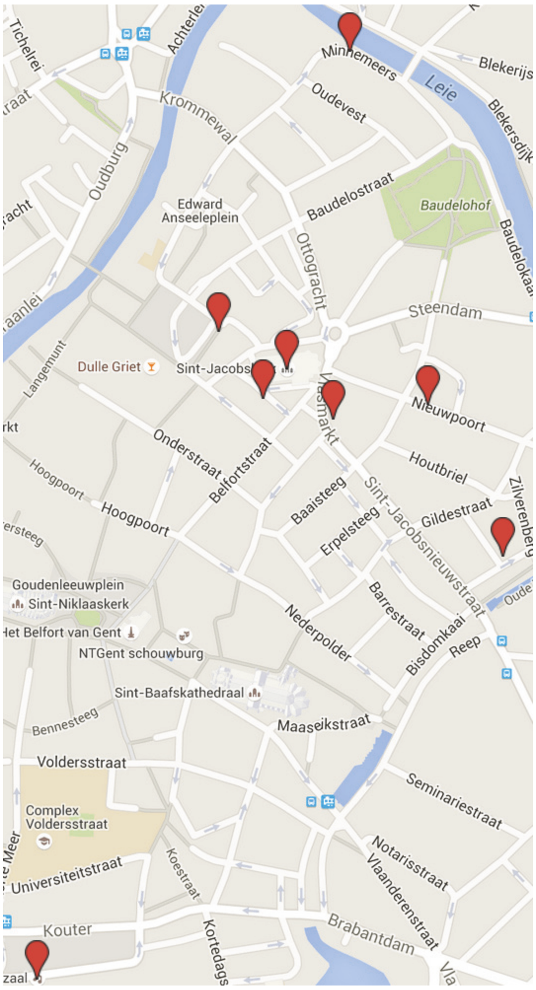
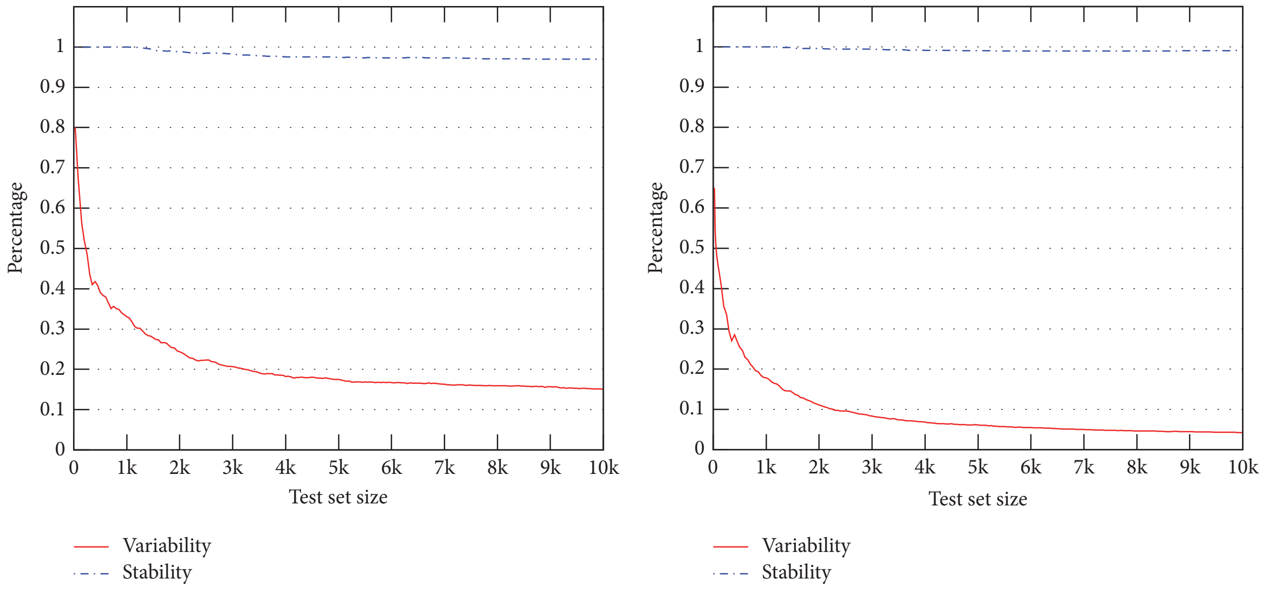


Figure 7: Distribution of MSs for the performed fingerprinting experiments.

5.2.2. Deanonymization. Since the goal of the tracking system is to deanonymize devices, that is, to link random MACs to a single fingerprint, having a large number of overlapping fingerprints is unfavorable.

To overcome this issue, note that we can utilize the fingerprinting algorithm in conjunction with temporal information. After all, the number of devices observed by a MS over a certain period of time is likely to be much smaller than the complete set of observed devices. Furthermore, when a device exposes its nonrandom MAC address at some point in time, random MAC addresses with the same fingerprint near that time are more likely to be associated with that device. The deanonymization of random MAC addresses can thus be performed on a *subset* of devices instead of the complete set. As shown in Figure 8, decreasing the test set size (or



(a) $\lambda = 0$

(b) $\lambda = 1$

FIGURE 8: The variability (solid line) of fingerprints for nonrandom MACs decreases due to collisions as the test set size increases. The fingerprint stability (dashed line) also slightly decreases due to some bits not being considered as unstable during training (e.g., when the training set is small). Results shown for a training set of 1,000 MACs.

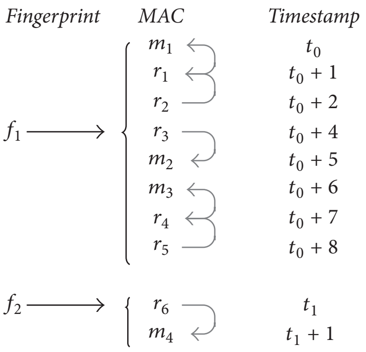


FIGURE 9: An example that shows how the fingerprints f_1 and f_2 can be used in conjunction with temporal information in order to link the random MAC addresses r_1–r_6 to the nonrandom MAC addresses m_1–m_3.

time interval) indeed increases the relative uniqueness of fingerprints, since the probability of observing devices with similar IEs decreases.

Multiple approaches can be considered for determining the subset size. In a naive approach, one could bin devices according to a specified time interval, but then the question remains of how to choose the bin size and how to handle devices that are observed at the boundary of a bin. Alternatively, the fingerprint of the random MAC can be mapped to the MAC that was *closest* in time and has an identical fingerprint for the highest probability of a correct match. In Figure 9, for example, the random MACs r_1–r_5 with fingerprint f_1 are mapped to their corresponding nonrandom MACs m_1–m_3. The algorithm pseudocode is shown in Algorithm 1.

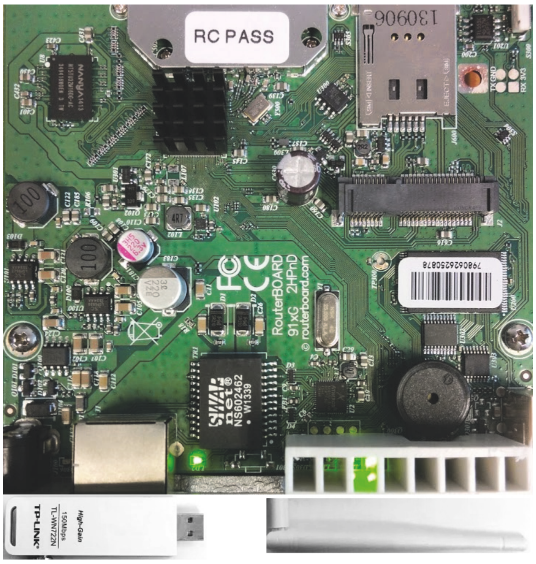


FIGURE 10: Top view of the hardware that was used for the MSs in our experiments.

When using the above algorithm to map a random MAC to its closest nonrandom MAC, we discovered that a matching fingerprint can be found with 99% probability. However, it should be noted that these mappings cannot be guaranteed to be correct as opposed to our experiments where only nonrandom MACs were considered. That is, the information required to validate the accuracy of the mapping (i.e., the true MAC address of the device) is not available in an uncontrolled environment, which is an inherent problem in *noncooperative* deanonymization. One solution to overcome this problem

```
procedure DEANONYMIZE(frame)
    $t \leftarrow$ frame.timestamp
    $m \leftarrow$ frame.addr2    ▷Transmitter MAC
    $f \leftarrow$ fp(frame)    ▷Get fingerprint bitstream(s)
    if is_random_mac ($m$) then
        for $i$ in range (0, len(all_frames)) do
            $t_i \leftarrow$ frame$_i$.timestamp
            $m_i \leftarrow$ frame$_i$.addr2
            $f_i \leftarrow$ fp(frame$_i$)
            if $f_i = f$ then
                $d_i \leftarrow$abs($t - t_i$)
            end if
        end for
        return $m_i$ where $d_i$ is minimal
    end if
    return $m$
end procedure
```

Algorithm 1: Procedure for deanonymizing random MAC addresses.

Table 2: Number of devices per vendor OUI as observed during our evaluation experiments. A combined total of 138 unique devices were observed during the two experiments.

Vendor	Number of devices
Intel	40
Apple	21
Samsung	13
Lenovo/Motorola	11
OnePlus	11
LG	7
Hon Hai	6
Nokia/Microsoft	5
Murata	4
Compal	2
HTC	2
Cisco	2
TP-Link	2
Axis Communications	2
Other	10

could be to install an application on each mobile device partaking in the experiment. This application can provide the true MAC address to the fingerprinter for each random MAC, so that the accuracy of the deanonymization can be determined. A large dataset containing this information would be an interesting contribution for future work.

5.3. Transmission Rate Experiments. To correctly evaluate the *transmission rate increasing techniques* (Section 4.1), a number of complications had to be addressed. A first complication is that, in order to measure the transmission frequency of a device, the device must remain in range of a MS an equal (preferably large) amount of time for each tested technique. This is rarely the case in a field setup (e.g., a music festival or shopping center), since, here, devices are able to roam freely and are typically observed only a few times by a single MS. Secondly, the set of tested devices must be diverse and sufficiently large in order to be representative.

For these reasons, we have chosen to perform the evaluation of the transmission rate increasing techniques at our research lab. Here, devices are more likely to remain in range of the MS for extended durations compared to a field setup. At the same time, there is a healthy model and vendor diversity between devices owned by the researchers (see Table 2).

We have performed two experiments. The goal of the first experiment is to determine under which conditions a device will respond to a stimulus frame. Our second experiment shows how these methodologies compare against each other and against traditional approaches such as `Beacon` spoofing in terms of transmission frequency. Only nonrandom MACs were considered in order to prevent multiple observations of the same device.

5.3.1. Response Conditions. For each of the frames discussed in Section 4.4, we have determined under which conditions and for which STAs they can be used to instigate transmissions. We define four test cases in decreasing order of knowledge required by the MS:

(1) *Known BSSID*: the targeted STA responds to a broadcast stimulus frame only if the `BSSID (addr3)` and `Transmitter Address (addr2)` fields are correctly set to the associated AP. In other words, no frames from unassociated STAs are accepted. This requires the most knowledge by the MS, since the MS must encounter the target while it is associated with an AP, and this AP's BSSID must be spoofed.

(2) *Unicast*: the targeted STA responds to the stimulus frame only if addressed directly. Here, the MS would only need to have knowledge of the target's current MAC address. The BSSID is set to "`ff:ff:ff:ff:ff:ff`" (the wildcard BSSID).

(3) *Broadcast BSSID*: the targeted STA responds to the broadcast stimulus frame when the BSSID field is set to "`ff:ff:ff:ff:ff:ff`." This allows the MS to probe all devices in range.

(4) *Zero BSSID*: the targeted STA responds to the broadcast stimulus frame when the BSSID field is set to "`00:00:00:00:00:00`." This nonstandard behavior might indicate a misinterpretation of the standard or an implementation bug where the BSSID field is not properly validated.

This experiment was performed as follows: first, we used a TP-Link TL-WN722N dongle (Atheros AR9271 chipset) in monitor mode (see Figure 10) to scan for in-range BSSIDs and STAs. For this purpose we have created a Python script that uses the "Scapy" packet manipulation library. The code of this script will be published after acceptance of this work.

After the initial scan, the script continuously transmits each type of stimulus packet for 60 seconds for each of

Table 3: Overview of the number of unique STAs out of 136 that responded to a stimulus frame type (rows), for the 4 test cases that we defined in Section 5.3.1 (columns).

	Known BSSID	Unicast	Broadcast BSSID	Zero BSSID
ADDBA Request	27	7	4	3
GAS Request	4	6	7	6
Basic measurement request	9	1	0	0
CCA Request	11	1	0	0
Channel load request	7	2	2	2
STA statistics request	8	2	2	2
Frame request	8	2	2	2
Link measurement	8	2	2	2
WNM Event Request	2	2	2	2
WNM Timing Measurement Req.	2	2	2	2
TDLS Discovery	0	0	0	0
TDLS Setup	0	0	0	0

the four test cases. Between each test, the experiment was paused for 10 seconds to prevent slow processing of packets from influencing the next test. The experiment was run for a total period of 17,038 seconds, observing 136 unique STAs and 27 BSSIDs. The packet trace of this experiment containing each of the stimulus frames and their responses can be found at Online Resource 1 [46]. There were no other devices transmitting these requests at the time of the tests. The number of devices that sent a response for the stimulus frame per experiment is shown in Table 3. These results will be discussed in the following sections.

5.3.2. ADDBA Request. For the `ADDBA Request` frame (Section 4.4.1) test, we have determined that 27 out of 136 devices responded during the known BSSID test, 7 devices replied to unicast frames, 4 devices responded to a broadcast BSSID `ADDBA Request`, and 3 devices replied in the zero BSSID test.

Among the devices that responded to broadcast and zero BSSID frames were 3 Intel chipsets returning an `ADDBA Response` with error code 37 (request declined), and 2 Axis Communications chipsets responding with a `Block Ack Error` frame (category 131). These devices thus leak their current MAC address to the MS in response to a broadcast ADDBA frame.

5.3.3. GAS Request. In the `GAS Request` frame (Section 4.4.2) experiment we continuously transmitted ANQP queries containing the "vendor specific" element. We have chosen this element specifically because it is the only element supported in peer-to-peer mode by the most popular supplicant for Android and Linux devices, `wpa_supplicant` [47].

We have observed that 4 devices responded in the known BSSID case, 6 devices replied to unicast frames, 7 devices responded in the broadcast BSSID case, and 6 devices in the zero BSSID case. Unlike what we have seen for ADDBA Requests, there is no significant difference in the number of observed devices between the known BSSID and broadcast BSSID tests. This means that if GAS is supported by a device, the device is likely to respond to broadcast peer-to-peer frames.

Support for GAS by a device could be determined in two ways. A first is to look at the Interworking IE transmitted by a device in `Probe Requests`. However, we observed that not all devices that transmit the Interworking IE respond to GAS Requests, and not all devices that respond to GAS Requests transmit the Interworking IE. Therefore, the presence of the Interworking IE in `Probe Requests` is not a useful metric to determine how many devices support peer-to-peer GAS Requests.

A better approach is to look at "Passpoint" certification instead. Devices that support 802.11u are often marketed using the terms "Passpoint" and "Hotspot 2.0". Here, Passpoint is the label that a device obtains when it passes the certification program by the Wi-Fi Alliance, and Hotspot 2.0 is the name of the technical specification that was developed by the Wi-Fi Alliance based on the 802.11u amendment [48].

A full list of 938 Passpoint certified devices can be found via the Wi-Fi Alliance Product Finder tool [49]. It should be noted that this list is merely a lower bound for vulnerable devices. For example, among the devices that responded, we observed two Axis Communications devices that are not Passpoint certified.

5.3.4. Spectrum Management Request. In context of Spectrum Management (Section 4.4.3), we have tested `Basic Requests` and `Clear Channel Assessment (CCA) Requests`.

For `Basic Requests`, no devices responded during the broadcast BSSID and zero BSSID test cases. During the known BSSID test case, however, a response was received for 9 devices from various vendors. Only a single Intel chipset responded with a `Spectrum Management Error` frame during the unicast test. The results for `CCA Requests` were identical, except 11 devices were observed during the known BSSID test.

Given these results, we conclude that although Measurements Reports can be instigated if supported by the device, the BSSID must be known to the MS, and the STA must be associated.

5.3.5. Radio Measurement Request. For Radio Measurement (Section 4.4.3), we have tested `Channel Load`, `STA Statistics`, `Frame`, `and Link Measurement Requests`. For each of these frames, two Axis Communications chipsets responded with error frames during the broadcast BSSID, zero BSSID, and unicast test cases. During the BSSID test cases, we observed responses from 7 devices for the `Channel Load` experiment, and 8 devices for the other experiments. Interestingly, the responses all originated from Motorola and OnePlus smartphones, which suggests that Radio Measurement is only supported by these devices.

5.3.6. WNM and TDLS. For WNM related frames, we were only able to instigate error responses from the two Axis Communications chipsets. Other devices did not respond

Table 4: An overview of the advantages of each technique discussed in Section 4.

	Beacon spoofing	RTS/Null Data	GAS Request	ADDBA Request
Response contains current MAC	●	●	●	●
Response contains implicit identifiers	●			
Immediate reply from device	◗*	●	●	●
Can be broadcast	●		●	◗†
Requires no knowledge about device			●	●
Commonly supported	●	●		●

*With an implementation dependent delay (see [17]) and only if at least one SSID in the device's PNL is guessed.

†Implementation dependent.

Table 5: Comparison between the average number of frames received by the MS during the control test and the technique tests for 32 unique devices. The average number of frames received highly varies between devices but is greater than the control test for each technique except the Beacon technique.

	Average	Standard deviation
Control test	*155.8*	*224.7*
Common Beacons	148.2	219.4
Null Data	956.3	4529.8
ADDBA	168.9	233.5
GAS	13219.2	48693.7

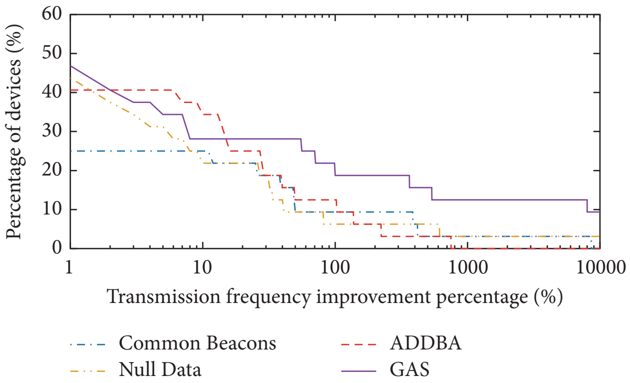


Figure 11: The percentage of observed devices in function of the percentage of improvement compared to the control test in log scale. The GAS technique is the most likely to trigger a response and instigates the most responses compared to the control test: for 3 to 9 out of 32 devices (9.37 to 28.1 percent) there is an improvement of 50 to 10,000 percent.

in any of the test cases. No devices responded to TDLS frames. A possible explanation is that these protocols are rarely supported (in Wireshark, the de facto standard tool for packet inspection, WNM `Event Request` IE parsing is not fully supported, as can be observed from the provided traces). We decided to mention these results nevertheless, since they could be of value to future work.

5.3.7. Discussion and Comparison. Based on our experimental results, we hypothesize that `GAS Request` frames will be the most effective to instigate transmissions in practice. These frames can be broadcast per standard definition and will trigger responses regardless of whether the targeted STAs are associated with an AP. However, the GAS protocol must be supported by the receiving device. A second interesting type is the ADDBA frames, since some Intel chipsets similarly respond to broadcast frames. This behavior can additionally be used as an implicit identifier. In Table 4, we compare the techniques using these frames to previous approaches.

For the comparison experiment, our setup consists of a single MS that forwards all captured frames to a central server. The MS hardware is identical to the hardware used in Section 5.2, that is, a MikroTik RB912UAG-2HPnD equipped with an AR9342 chipset (see Figure 10).

To compare the effectiveness of the techniques from Section 4 in a realistic tracking scenario, we have measured the number of Class 1 frames that the MS received over a total period of 8 hours. Only devices with nonrandom MACs that sent more than 100 frames (32 devices) were considered. The tests for each technique were interleaved in order to mitigate the effects of changing channel conditions. As such, the tests were performed intermittently for a period of 5 minutes each.

Besides novel techniques, we have tested the techniques used in previous works, that is, `Beacon` spoofing (Section 4.2) and directed `Null Data` frames (Section 4.3). Recall that we assume the MS has no knowledge about nearby devices, analogous to the "broadcast BSSID" test case from Section 5.3.1. For the `Null Data` technique, however, we had to relax this assumption and allow unicast, since responses cannot be observed otherwise. For the `Beacon` technique, our MS spoofed several local and popular networks, such as "linksys," "TELENETHOMESPOT," and "Proximus_FON."

The effectiveness of each technique was determined by comparing the number of received Class 1 frames from the device against the control test, where no special techniques were used. Table 5 shows the average number of frames received by the MS during the control test and technique tests. For each test, the standard deviation is high because the STA transmit behavior highly differs between implementations. We can also see that, on average, only the Beacon spoofing technique performs worse than the control test. This can be attributed to a number of reasons: the tracked device must have one of the spoofed SSIDs in its PNL (therefore, the choice of which SSIDs to spoof in a tracking system impacts performance significantly), the resulting `Probe Request`'s transmission is not immediate, the `Beacons` themselves cause more channel contention, and some devices stop probing after association [17].

Figure 11 graphically shows, for each technique, the percentage of the 32 observed devices in function of the percentage of improvement compared to the control test. A log scale was chosen since some devices responded to *each* stimulus frame, vastly increasing their transmission frequency. In general, GAS frames appear to be the most favorable: unlike `Null Data` frames, GAS frames can be broadcast, appear to be supported by the most devices, and also instigated the most transmissions. ADDBA frames were slightly less effective, since not all implementations respond to this type of frame. Null Data frames performed similarly but require knowledge of the tracked device's MAC address. Finally, `Beacons` are the least favorable technique, which can be attributed to the fact that only 3 devices in our lab responded to the spoofed SSIDs. However, a `Probe Request` contains more implicitly identifying information (e.g., IEs) than other responses.

5.4. Practical Location Tracking. Now that we have detailed and evaluated several techniques for fingerprinting a device and instigating transmissions from it, the question remains of how to apply these techniques in practice. The fingerprinting approach from Section 3.3 can be used to create a unique identifier for devices observed by the MSs, based on a single `Probe Request`. Temporal information can then be used to link random MACs with their corresponding nonrandom MAC (see Section 5.2.2).

If the transmission frequency of tracked devices is low or if the tracked devices are moving fast, the techniques that we discussed in Section 4.1 can be used (note that when a device is moving fast, we can only use broadcast (or unicast if the target MAC is nonrandom and known), since roaming devices are rarely associated with one particular AP for an extended duration. As such, we cannot exploit the "known BSSID" assumption in this case). ADDBA frames and GAS Requests are the most effective for instigating transmissions, but their responses do not contain as much uniquely identifying information as `Probe Requests`. Therefore, these techniques will work best for nonrandom MACs. On the other hand, spoofed Beacon frames can instigate `Probe Requests` and are therefore useful for both nonrandom and random MACs, but this approach does not increase the transmission frequency significantly.

6. Countermeasures

To prevent MAC layer based tracking systems from tracking a user's device without their knowledge through the techniques discussed in this work, several countermeasures can be pu t in place by vendors of mobile devices. We will now briefly discuss these countermeasures. Ideally, they should be combined instead of being considered separately.

(i) *Enable MAC address randomization*: since the MAC address is a globally unique identifier, it must always be completely randomized for every transmitted frame when actively scanning for APs. In order to not break roaming functionality, the real MAC address can still be used when associating with an AP, on the condition that the device's PNL does not contain SSIDs that can be guessed by an attacker (see Section 4.2). Devices that use `wpa_supplicant` can enable this countermeasure through the options `mac_addr`, `rand_addr_lifetime`, and `preassoc _mac_addr`.

(ii) *Reduce Probe Request frequency*: since `Probe Request` timing information can be used as an implicit identifier [30], these frames should be transmitted infrequently and at random intervals.

(iii) *Avoid directed Probe Requests*: `Probe Request` frames ideally must only contain an `SSID Parameter Set` IE with the broadcast or empty (null) SSID in order to prevent leakage of the device's PNL or connection history. Alternatively, the device can choose to only scan for networks passively [27, 30, 33].

(iv) *Defer transmission of IEs*: instead of transmitting all IEs for every `Probe Request`, the device should only share this information with an AP in the association stage, since `Probe Requests` contain identifying information [35]. This limits the tracker's opportunities to instances where the user manually connects or where an SSID from the PNL is known.

(v) *Ignore broadcast Class 1 frames*: peer-to-peer Class 1 frames transmitted to `ff:ff:ff:ff:ff:ff` should be ignored. A receiver should only respond to such a request if the frame is directly addressed to its *current* MAC address.

(vi) *Randomize sequence numbers*: the `Sequence Number` field in the MAC header should be randomized while the STA is unassociated in order to prevent deanonymization [31, 35].

(vii) *Validate state machines*: a STA should only interpret a frame when it is received in the correct state. For example, an `ADDBA Request` frame should not be accepted from unassociated STAs, as this would imply a transfer of data frames.

(viii) *Trusted locations*: the user should be allowed to select trusted geographical locations where the Wi-Fi chip is exclusively enabled, in order minimize the risk of being tracked [33].

7. Related Work

In previous work, various noncooperative 802.11 MAC layer based tracking algorithms and topologies have been proposed. Abbott-Jard et al. have implemented a noncooperative tracking system to estimate travel durations for vehicles on the road by solely monitoring MAC addresses [7]. Bonné et al. have created a similar tracking system for tracking movement patterns of event visitors [8]. Musa and Eriksson have implemented a tracking system for vehicles using a probabilistic approach [11].

For *fingerprinting* devices on the MAC layer, an approach where a combination of transmitted network data, SSIDs, specific MAC header fields, and transmission rates is used was discussed by Pang et al. However, in this work MAC header fields on their own were not yet considered practical to distinguish users uniquely [14]. Neumann et al. have used several parameters from the Radiotap header to learn a fingerprint for a device [28]. Cunche et al. utilized the rarity and frequency of SSIDs to fingerprint and link devices [30].

The works by Chernyshev et al. and Bonné et al. [29, 33] use SSIDs to map observed devices to a set of visited geolocations using databases such as https://wigle.net [13]. Vanhoef et al. apply entropy clustering techniques to a selection of IEs and sequence numbers to identify a device but do not consider the bit-level entropy and assume a continuous observation of the tracked devices [35]. Other works made use of timing differences in `Probe Requests` [10, 26, 27].

One of the first works that aims to improve MAC layer fingerprinting by *eliciting more transmissions* from devices was published by Bratus et al. [34]. Here, the reaction of a STA to stimulus `Deauthentication` frames, `Beacons`, `Probe Responses`, and failed authentications was observed and used to fingerprint the device. However, it is assumed that the STA is associated with an AP or joining a BSS. The work presented by Musa and Eriksson [11] deserves a special mention because it demonstrates a number of preliminary techniques to instigate transmissions in context of noncooperative tracking on the MAC layer. They have spoofed `Beacon frame` SSIDs to increase the frequency of `Probe Requests` from unassociated devices and additionally used injected RTS frames for eliciting CTS frame responses. Similarly, introducing a known or common SSID to increase the transmission frequency of nearby unassociated devices is mentioned in the studies performed by Cunche [10] and Bonné et al. [33]. An overview of all relevant works and their features is given in Table 6.

8. Conclusions

Despite efforts by vendors for implementing MAC address randomization, we have shown how a device can nevertheless be fingerprinted and deanonymized, even if the device is not cooperating or not associated with an AP. In our approach we have discussed how a tracking system can combine implicitly identifying IE bits from a single `Probe Request` frame to form a fingerprint that is at least 80.0 to 67.6 percent unique for small sets of 50 to 100 devices and at least 33.0 to 15.1 percent unique for large sets of 1,000 to 10,000 devices. We have evaluated these results using two datasets. The first dataset was recorded at Glimps 2015 and the second at our research lab, containing, respectively, 28,048 and 138 unique devices. Additionally, we have discussed and compared our work against previous works that aim to achieve similar goals. An overview of these works was given in Table 6.

Further, we have shown how these fingerprints can be combined with temporal information and how extra frames can be instigated by a tracking system when a device sends frames infrequently or not at all. We have demonstrated how a MS can exploit protocol design flaws and implementation vulnerabilities to achieve this goal, given that nearby devices support the respective protocols. More specifically, we have studied a wide array of Class 1 frames, such as Beacon, RTS/CTS, Null Data, GAS Request, ADDBA Request, and other Action frames. Such frames can be actively injected by a MS to expose the presence of nearby devices more frequently and reveal more implicitly identifying information on both the PHY and MAC layers to the tracking system. We have experimentally determined that GAS frames are particularly interesting in this regard, as these frames can be broadcast and used to instigate transmissions on demand from Hotspot 2.0 and Passpoint compatible devices while unassociated. Compared to the control test, we measured a transmission rate improvement of 50 to 10,000 percent for 3 to 9 out of 32 devices (9.37 to 28.1 percent). ADDBA frames can be exploited in a similar fashion in some implementations.

As the diversity between devices increases in terms of capabilities and supported protocols, measures must be taken by vendors in order to prevent this kind of unsolicited location tracking by third parties. Among those are randomizing the 802.11 sequence number in conjunction with the MAC address during scanning, limiting the amount of information sent in `Probe Requests`, and preventing replies to broadcast Class 1 frames.

9. Future Work

We believe there are a number of interesting opportunities for future work concerning the subject of noncooperative tracking. The fingerprinting techniques discussed in this paper can be combined with approaches from other works in order to further improve the fingerprint uniqueness. Here, it is essential that identifiers with high variability and high stability are employed. These metrics can be determined using our proposed bit entropy approach. As an example, it would be particularly interesting for a tracking system to have a reliable and generic PHY layer identifier that can be derived from a single frame observation. To the best of our knowledge, such identifiers have yet to be discovered (recall that the PHY layer identifiers we discussed in Table 6 require a preprocessing step, multiple frame observations, or assumptions about a specific implementation). A hybrid PHY and MAC layer tracking approach could thus be interesting to examine in future work. Such systems would however require additional hardware (e.g., SDRs such as the RTL-SDR, HackRF, and BladeRF), changes to the existing infrastructure, and more complex fingerprinting algorithms in order to analyze the raw PHY radio signal. Another potential solution to increase the fingerprint uniqueness would be to include "correlatable features," such as the frame sequence number or scrambler seed (discussed in Section 2) in the fingerprint. However, for this to be effective, the targeted device must be continuously observed, since a gap in the observations would make it infeasible to link related sequence numbers or scrambler states.

Further, our techniques for instigating extra transmissions from nearby devices have revealed differing behavior in terms of timing and response conditions between several devices (see Table 3). These features could also be used as an implicit identifier for future fingerprinting techniques. Additionally, an increased transmission rate can benefit both PHY and MAC layer fingerprinting approaches that depend on a preprocessing or learning phase where multiple frame observations are required. Example use cases are the works described in [21–25, 35]. Here, the classification of observed devices can be sped up by instigating transmissions, that is, by giving the classifier more data to work with in a shorter time frame. Finally, besides 802.11e and 802.11u, other amendments

Table 6: An overview of 802.11 PHY and MAC layer noncooperative fingerprinting techniques.

	PHY				*MAC*									
	Clock skew	Scrambler seed	Time domain	Frequency domain	MAC address	Sequence number	Duration field	Timing	MAC header bits	SSIDs	Information elements	Payload data	Stimulus frames
Kohno et al. [18]	●											●	
Arackaparambil et al., Jana and Kasera [19, 20]	●												
Hall et al., Ureten and Serinken [21, 22]			●										
Brik et al., Corbett et al. [23, 24]				●									
Bloessl et al. [25]		●											
Desmond et al., Franklin et al. [26, 27]								●					
Neumann et al. [28]								●	●				
Cunche, Chernyshev et al., Cunche et al. [10, 29, 30]					●					●			
Pang et al. [14]					●				●	●	◗	●	
Guo and Chiueh [31]						●							
Cache [32]							●						
Abbott-Jard et al., Bonné et al. [7, 8]					●								
Musa and Eriksson, Bonné et al. [11, 33]					●								●
Bratus et al. [34]									●				●
Vanhoef et al. [35]		●				●					●		
This work					●			●			●		●

will work their way into the set of commonly used protocols. This will lead to more diversity among device capabilities and hence more implicit identifiers and opportunities to instigate transmissions. It would prove useful to investigate these future protocols for similar privacy and security flaws.

Disclosure

The authors declare that only `Probe Request` frames were collected at the music festivals. No data frame payloads were captured in order to respect the privacy of the visitors. Furthermore, the supplementary datasets provided with this work were anonymized, so that there is no traceability to the true devices or persons involved in the experiments.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by a Ph.D. Grant of the Research Foundation Flanders (FWO). The authors would like to thank The Safe Group for helping with the setup of MSs and the Pukkelpop and Glimps organizations for allowing them to perform their experiments.

References

[1] M. B. Kjærgaard, M. V. Krarup, A. Stisen et al., "Indoor positioning using wi-fi–how well is the problem understood?" in *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation*, vol. 28, p. 31, 2013.

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007.

[3] T. S. Prentow, H. Blunck, K. Gronbaek, and M. B. Kjærgaard, "Estimating common pedestrian routes through indoor path networks using position traces," in *Proceedings of the 15th IEEE International Conference on Mobile Data Management (MDM '14)*, vol. 1, pp. 43–48, IEEE, July 2014.

[4] R. Yamasaki, A. Ogino, T. Tamaki, T. Uta, N. Matsuzawa, and T. Kalo, "TDOA location system for IEEE 802.11b WLAN," in *Proceedings of the Wireless Communications and Networking Conference*, vol. 4, pp. 2338–2343, IEEE, March 2005.

[5] Ekahau: Asset Tracking & Management, https://www.ekahau.com/blog/2010/04/27/wi-fi-site-surveys-passive-active-rtls/.

[6] J.-H. Youn, H. Ali, H. Sharif, J. Deogun, J. Uher, and S. H. Hinrichs, "WLAN-based real-time asset tracking system in healthcare environments," in *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '07)*, p. 71, IEEE, October 2007.

[7] M. Abbott-Jard, H. Shah, and A. Bhaskar, "Empirical evaluation of Bluetooth and Wifi scanning for road transport," in *Proceedings of the 36th Australasian Transport Research Forum (ATRF '13)*, Queensland, Australia, October 2013.

[8] B. Bonné, A. Barzan, P. Quax, and W. Lamotte, "WiFiPi: involuntary tracking of visitors at mass events," in *Proceedings of the IEEE 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM '13)*, pp. 1–6, Madrid, Spain, June 2013.

[9] M. B. Kjærgaard, "A taxonomy for radio location fingerprinting," in *Location- and Context-Awareness*, pp. 139–156, Springer, 2007.

[10] M. Cunche, "I know your MAC address: targeted tracking of individual using Wi-Fi," *Journal of Computer Virology and Hacking Techniques*, vol. 10, no. 4, pp. 219–227, 2014.

[11] A. B. M. Musa and J. Eriksson, "Tracking unmodified smartphones using wi-fi monitors," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems (SenSys '12)*, pp. 281–294, ACM, Ontario, Canada, November 2012.

[12] B. Danev, D. Zanetti, and S. Capkun, "On physical-layer identification of wireless devices," *ACM Computing Surveys*, vol. 45, no. 1, article 6, 2012.

[13] WiGLE.net: Statistics, https://wigle.net/stats.

[14] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall, "802.11 User fingerprinting," in *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, pp. 99–110, ACM, September 2007.

[15] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '00)*, vol. 2, pp. 775–784, March 2000.

[16] S. A. Golden and S. S. Bateman, "Sensor measurements for Wi-Fi location with emphasis on time-of-arrival ranging," *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1185–1198, 2007.

[17] J. Freudiger, "How talkative is your mobile device?: an experimental study of Wi-Fi probe requests," in *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '15)*, pp. 8:1-8:6, ACM, New York, NY, USA, 2015.

[18] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 2, pp. 93–108, 2005.

[19] C. Arackaparambil, S. Bratus, A. Shubina, and D. Kotz, "On the reliability of wireless fingerprinting using clock skews," in *Proceedings of the 3rd ACM Conference on Wireless Network Security (WiSec '10)*, pp. 169–174, ACM, Hoboken, NJ, USA, March 2010.

[20] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE Transactions on Mobile Computing*, vol. 9, no. 3, pp. 449–462, 2010.

[21] J. Hall, M. Barbeau, and E. Kranakis, "Radio frequency fingerprinting for intrusion detection in wireless networks," *IEEE Transactions on Defendable and Secure Computing*, 2005.

[22] O. Ureten and N. Serinken, "Wireless security through RF fingerprinting," *Canadian Journal of Electrical and Computer Engineering*, vol. 32, no. 1, pp. 27–33, 2007.

[23] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM Annual International Conference on Mobile Computing and Networking (MobiCom '08)*, pp. 116–127, ACM, September 2008.

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

[25] B. Bloessl, C. Sommer, F. Dressler, and D. Eckhoff, "The scrambler attack: a robust physical layer attack on location privacy in vehicular networks," in *Proceedings of the International Conference on Computing, Networking and Communications (ICNC '15)*, pp. 395–400, IEEE, 2015.

[26] L. C. C. Desmond, C. C. Yuan, T. C. Pheng, and R. S. Lee, "Identifying unique devices through wireless fingerprinting," in *Proceedings of the 1st ACM Conference on Wireless Network Security*, pp. 46–55, ACM, April 2008.

[27] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting," in *Proceedings of the 15th Conference on USENIX Security Symposium (USENIX-SS '06)*, 2006.

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

[29] M. Chernyshev, C. Valli, and P. Hannay, "On 802.11 access point locatability and named entity recognition in service set identifiers," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 3, pp. 584–593, 2016.

[30] M. Cunche, M.-A. Kaafar, and R. Boreli, "Linking wireless devices using information contained in Wi-Fi probe requests," *Pervasive and Mobile Computing*, vol. 11, pp. 56–69, 2014.

[31] F. Guo and T. C. Chiueh, "Sequence number-based MAC address spoof detection," in *Recent Advances in Intrusion Detection*, pp. 309–329, Springer, 2005.

[32] J. Cache, "Fingerprinting 802.11 implementations via statistical analysis of the duration field," 2006, http://uninformed.org/?v=5.

[33] B. Bonné, P. Quax, and W. Lamotte, "Your mobile phone is a traitor!—raising awareness on ubiquitous privacy issues with SASQUATCH," *International Journal on Information Technologies & Security*, vol. 6, no. 3, 2014.

[34] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, "Active behavioral fingerprinting of wireless devices," in *Proceedings of the 1st ACM Conference on Wireless Network Security (WiSec '08)*, pp. 56–61, ACM, Alexandria, VA, USA, April 2008.

[35] 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 *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pp. 413–424, ACM, Xi'an, China, June 2016.

[36] M. Gruteser and D. Grunwald, "Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis," *Mobile Networks and Applications*, vol. 10, no. 3, pp. 315–325, 2005.

[37] IEEE Computer Society, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE 802.11 Standard, 2012.

[38] J. Malinen, "Wpa supplicant configuration options," https://w1.fi/cgit/ hostap/plain/wpa_supplicant/wpa_supplicant.conf.

[39] G. Chandrasekaran, J.-A. Francisco, V. Ganapathy, M. Gruteser, and W. Trappe, "Detecting identity spoofs in IEEE 802.11e wireless networks," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '09)*, pp. 1–6, IEEE, December 2009.

[40] C. L. Corbett, R. A. Beyah, and J. A. Copeland, "Using active scanning to identify wireless NICs," in *Proceedings of the IEEE Information Assurance Workshop*, pp. 239–246, IEEE, June 2006.

[41] IEEE Computer Society, *Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements*, IEEE 802.11e Standard, 2005.

[42] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with Wi-Fi direct: overview and experimentation," *IEEE Wireless Communications*, vol. 20, no. 3, pp. 96–104, 2013.

[43] Wi-Fi Alliance Technical Committee, "P2P Task Group: Wi-Fi Peer-to-Peer (P2P) v1.5," Technical Specification, Wi-Fi Alliance, 2014.

[44] D. Kotz, T. Henderson, and C. McDonald, "CRAWDAD: A Community Resource for Archiving Wireless Data at Dartmouth," http://crawdad.org.

[45] P. Robyns, B. Bonné, P. Quax, and W. Lamotte, "POSTER: assessing the impact of 802.11 vulnerabilities using wicability," in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '16)*, pp. 217-218, ACM, Darmstadt, Germany, July 2016.

[46] P. Robyns, "Online Resource 1: pcap file of the stimulus frame response conditions experiment," 2016, http://research.edm.uhasselt.be/~probyns/mactracking/stimulus_experiment_results.pcap.

[47] J. Malinen, "Wpa supplicant official source code repository," https://www.w1.fi/cgit/hostap/tree/src/p2p/p2p_sd.c?id=fb09ed338919db09f3990196171fa73b37e7a17f#n384.

[48] Wi-Fi Alliance: Wi-Fi CERTIFIED Passpoint, https://www.wi-fi.org/discover-wi-fi/wi-fi-certified-passpoint.

[49] "Wi-Fi Alliance: Product Finder search results," https://www.wi-fi.org/product-finder-results?sort_by=default&sort_order=desc&categories=4&capabilities=1.