Domien Schepers* and Aanjhan Ranganathan

Privacy-Preserving Positioning in Wi-Fi Fine Timing Measurement

Abstract: With the standardization of Wi-Fi Fine Timing Measurement (Wi-Fi FTM; IEEE 802.11mc), the IEEE introduced indoor positioning for Wi-Fi networks. To date, Wi-Fi FTM is the most widely supported Wi-Fi distance measurement and positioning system. In this paper, we perform the first privacy analysis of Wi-Fi FTM and evaluate devices from a wide variety of vendors. We find the protocol inherently leaks location-sensitive information. Most notably, we present techniques that allow any client to be localized and tracked by a solely passive adversary. We identify flaws in Wi-Fi FTM MAC address randomization and present techniques to fingerprint stations with firmware-specific granularity further leaking client identity. We address these shortcomings and present a privacy-preserving passive positioning system that leverages existing Wi-Fi FTM infrastructure and requires no hardware changes. Due to the absence of any client-side transmission, our design hides the very existence of a client and as a side-effect improves overall scalability without compromising on accuracy. Finally, we present privacy-enhancing recommendations for the current and next-generation protocols such as Wi-Fi Next Generation Positioning (Wi-Fi NGP; IEEE 802.11az).

Keywords: IEEE 802.11, Wi-Fi, Fine Timing Measurement, Privacy, Positioning

DOI 10.2478/popets-2022-0032
Received 2021-08-31; revised 2021-12-15; accepted 2021-12-16.

1 Introduction

IEEE 802.11 is an ever-evolving set of standards, aiming to add new features and enhanced performance to Wi-Fi networks. One of the recent amendments is IEEE 802.11mc, incorporated in IEEE 802.11-2016 [7], and standardized Wi-Fi Fine Timing Measurement (FTM). Wi-Fi FTM is named Wi-Fi Location by the Wi-Fi Alliance, and commonly referred to as Wi-Fi Round-Trip Time (RTT). The measurement protocol enables stations to estimate the distance between them with meter-level precision [11, 52]. Since its standardisation in 2016, Wi-Fi FTM is increasingly being leveraged to build novel location-based services by both industry and academia [17, 20, 23]. For example, Google introduced support for Wi-Fi FTM (named Wi-Fi Round-Trip Time by Google) in Android 9 and expanded its functionality in Android 12 [4]. Google has since released two Android applications (WiFiRttScan [32] and WiFiRttLocator [31]) for developers to evaluate the performance of their indoor positioning, navigation, and retail applications as it predicts *increasing support for Wi-Fi FTM in the future* [31]. Furthermore, Wi-Fi Aware [2], a Wi-Fi Alliance standard that enables proximal Wi-Fi devices to quickly discover, connect, and exchange information without the need for any infrastructure uses Wi-Fi FTM to scan for nearby devices. Google has also released an example application for Wi-Fi Aware (*WiFiNanScan*) [33]. Achieving meter-level positioning precision over widely distributed Wi-Fi access points and smartphones opens up countless opportunities for new applications and Wi-Fi FTM location based applications such as indoor positioning [11, 20, 52], asset tracking, and network management and device onboarding [23]. To summarize, Wi-Fi FTM is a first step towards integrating wireless positioning information into IEEE 802.11 standards and we are already seeing it used as a foundation for next generation standards such as IEEE 802.11az (Wi-Fi Next Generation Positioning [18]) and Wi-Fi Aware [2]. The aim of this work is to proactively evaluate and understand the privacy-implications of massive deployment of Wi-Fi FTM as it stands today and encourage stakeholders to mitigate all identified flaws in their existing and upcoming products.

In this paper, we perform the first privacy analysis of Wi-Fi Fine Timing Measurement (FTM), covering a wide variety of devices from Intel, Qualcomm, and Broadcom. Our analysis reveals the protocol leaks location-sensitive information, for example, a passive eavesdropper is able to physically locate any client by learning its measured distance by simply observing

***Corresponding Author: Domien Schepers:** Northeastern University, E-mail: schepers.d@northeastern.edu
Aanjhan Ranganathan: Northeastern University, E-mail: aanjhan@northeastern.edu

Wi-Fi FTM transactions using commodity Wi-Fi hardware. Specifically, we show that a mobile (client) application using Wi-Fi FTM for positioning or proximity verification transmits sensitive information that can be used to extract client's precise location by *any* device within the communication range of the client/mobile device. Furthermore, an adversary can deanonymize and track clients despite their usage of Wi-Fi FTM MAC address randomization, and fingerprint client stations with firmware-specific granularity. Our client-station identifying fingerprints are based on frame-level parameters and characteristics (e.g., vendor make, response behaviour to invalid frames) that are specific to the device under test and independent of the measurement environment. This means, in contrast to conventional triangulation methods using Received Signal Strength Indicator (RSSI), our location inference technique does not require the attacker to a-priori measure and build a radio signal fingerprint database in the area of interest; thereby significantly lowering the bar for compromising location privacy. In addition, we identify practical scalability limitations in all evaluated devices, restricting the number of concurrent clients or even resulting in a denial-of-service. Throughout our analysis we identified a wide variety of vulnerabilities and information leakages, which were disclosed to their respective vendors and resulted in numerous CVEs and patches by Intel, Qualcomm, Google and more, highlighting the practical and real-world impact of our privacy analysis.

In order to address the identified shortcomings, we present and evaluate the design for a fully passive and privacy-preserving positioning system. Our system leverages existing Wi-Fi FTM infrastructure and requires no hardware modifications. Furthermore, our system hides the presence of a client, and since it does not require clients to transmit and occupy the communication channel, it improves the overall scalability. We implement our design and perform real-world experiments demonstrating clients can perform passive distance measurements and self-positioning with meter-level accuracy, equaling that of traditional Wi-Fi FTM. Finally, to mitigate the identified flaws, we present privacy-preserving recommendations to be considered in the development and implementation of current and next-generation protocols.

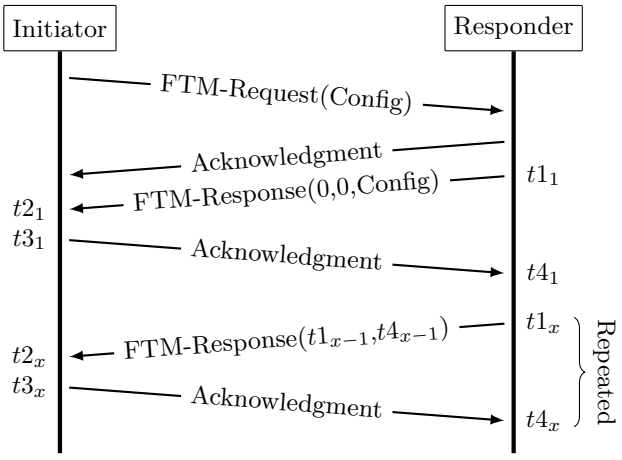


Fig. 1. Example Wi-Fi Fine Timing Measurement session with ASAP=1, and x-number of measurements for a single burst.

2 Background

In this section, we present an overview of standardized Wi-Fi distance measurement and positioning protocols. To date, the IEEE standardized Wi-Fi Fine Timing Measurement (Wi-Fi FTM; IEEE 802.11mc [7]), and is in the process of standardizing Wi-Fi Next Generation Positioning (Wi-Fi NGP; IEEE 802.11az [18]).

2.1 Wi-Fi Fine Timing Measurement

Wi-Fi Fine Timing Measurement (FTM) is defined in the IEEE 802.11mc amendment, and incorporated in IEEE 802.11-2016 [7]. The measurement procedure enables two stations to determine the distance between themselves by measuring the round-trip time of exchanged frames. As such, it is commonly referred to as Wi-Fi Round-Trip Time (RTT). In practice, client stations continuously initiate the measurement procedure through distance- and location-aware applications. The measurement results allow clients to determine their precise location (e.g., using trilateration), and therefore serves well for purposes such as indoor positioning [11, 20, 52], asset tracking, and network management [23]. Furthermore, the Wi-Fi Alliance adopted Wi-Fi FTM as a key feature in Wi-Fi Aware, listing use cases such as geo-fencing and mobile identification [2].

2.1.1 Round-Trip Time Measurement Protocol

IEEE 802.11mc uses a round-trip time measurement protocol in order to determine the distance between two stations. An initiating station starts the measurement session by transmitting a request frame, which includes configuration parameters (e.g., scheduling, channel and operational parameters) and vendor-specific information elements. In order to accommodate for station constraints (e.g., concurrent sessions, higher-priority traffic), stations may request a preferred time window allocation, referred to as a burst instance. That is, a station may request the burst instance to start As Soon As Possible (ASAP) or request a later time window. The responding station, often configured as an access point, responds with a status code verifying the requested parameters. If the stations agree upon the parameters, the responding station will commence the measurement exchange phase, in which the responding station measures round-trip times and shares them in a response frame. Consecutive response frames are spaced apart by a minimum time interval, defined as Min Delta FTM, and allows for stations to prepare for the arrival of new response frames. The session terminates implicitly after the last burst instance as defined by the session configuration parameters. Figure 1 shows an overview of the protocol message exchange, using ASAP=1 mode and x-number of measurements for a single burst.

Based on the arrival and departure timestamps of the frames, the initiating station is able to calculate the in-flight time of the radio signals, and as such determine the distance between itself and the responding station. Specifically, timestamps t_1 and t_3 represent the Time of Departure (ToD) of the response and acknowledgment frames, similarly t_2 and t_4 represent their Time of Arrival (ToA). Timestamps correspond to the moment at which the preamble of the frame appears at the antenna connector. According to the standard, an implementation may capture a timestamp at another point in time, and correct for the expected time differences [7, §6.3.58.1]. The average round-trip time is then calculated using equation:

$$RTT = \frac{1}{n}\sum_{x=1}^{n}((t4_x - t1_x) - (t3_x - t2_x)) \qquad (1)$$

where n is the total number of distance measurements. The initiating station is able to derive the measured distance as $RTT/2$ knowing radio waves travel at the speed of light, or 29.98 centimeter per nanosecond. Since only the initiating station is capable of determining the distance, the network's infrastructure (e.g., access point) is prevented from learning an initiator's distance or position. If the responder wishes to measure the distance, the stations need to switch roles and start a new session.

2.1.2 Security and Privacy Features

The IEEE 802.11mc standard provides no confidentiality, integrity or authentication. This is a result of clients not having to authenticate to the access point, an accessibility feature that results in no cryptographic keys being available to secure the exchanged messages. This is a notable change from IEEE 802.11-2012, defining Timing Measurement [6, §10.23.5], the predecessor of Wi-Fi FTM. The outdated protocol uses Received Signal Strength Indicator (RSSI) techniques, and allowed only post-association distance measurements [14]. Given the wireless and open nature of Wi-Fi, a passive adversary is now able to capture any frame within the Wi-Fi FTM protocol message exchange. In order to provide a minimum of privacy and security in IEEE 802.11mc, initiating stations are encouraged to use a randomized MAC address when initiating a new session. The responding station will then run the measurement session with an apparent anonymous client, and as such will make it harder for an adversary to track any client. Wi-Fi FTM MAC address randomization is enforced on Android smartphones, for example, Google Pixel 4 XL.

2.2 Wi-Fi Next Generation Positioning

The IEEE has formed a task group to work on a new and improved distance measurement standard named IEEE 802.11az, also referred to as Wi-Fi Next Generation Positioning (NGP) [18]. Under its goals, it is stated to support a better accuracy than Wi-Fi FTM, while *reducing existing wireless medium use and power consumption and is scalable to dense deployments* [18]. Security and privacy features are not explicitly listed as a goal on the task group website [18]. To date, the standard remains under development, is not publicly available, and is scheduled to be published in 2023. As a result, researchers remain unaware of the inner workings of Wi-Fi NGP, and any security and privacy features, if present, are hard to assess.

In order to allow for the scalability of Wi-Fi NGP, researchers have presented passive localization techniques. That is, in [8, 9], the authors propose a passive localization technique named Collaborative Time of Arrival (CToA), utilizing early IEEE 802.11az fea-

tures with customized Wi-Fi FTM APs to form a high-precision geolocation network. It is unclear if CToA is to be adopted in Wi-Fi NGP, since documents published by the task group in 2019 indicate it is instead set to support a passive ranging mode named Passive Location Ranging [18]. Using this mode, a station is expected to passively measure its differential distance to other sets of stations, and as such perform self-localization. In Section 4.3, we present this type of passive localization system using existing Wi-Fi FTM infrastructure, providing novel insights in its potential positioning accuracy.

Until a final standard of Wi-Fi NGP is defined and made public by the IEEE, researchers, the commercial industry, and enthusiasts are building practical systems using Wi-Fi FTM. As a result, these systems depend on the security and privacy guarantees of Wi-Fi FTM, and having an understanding of them will help in the design and implementation of Wi-Fi NGP.

Fig. 2. Overview of initiating and responding stations (left), and our Wi-Fi FTM accuracy evaluation setup (right) of Appendix A.

3 Wi-Fi FTM Privacy Analysis

In this section, we perform a privacy analysis of Wi-Fi FTM, as defined in IEEE 802.11-2016 [7]. Our analysis identifies shortcomings in the standard and its various implementations, exposing clients to an adversary capable of performing localization and tracking (Section 3.5).

3.1 Methodology

In our privacy analysis, we investigate how the standard and its various vendor-specific implementations preserve user and location privacy. In order to make that assessment, we break down each piece of information that is exposed due to the open nature of the protocol. We then investigate if and how this information can be leveraged by a (passive) adversary, and how it poses a (location) privacy risk for any client of the protocol. For example, the responding station transmits timestamps t_1 and t_4, representing the measured round-trip time, in the clear. As a result, a passive observer can trivially learn their values. We find this is sufficient to track movements of a client station, and even allows for physical localization (Section 3.2). Next, we investigate the privacy-preserving features that are provisioned by the standard, as described in Section 2.1.2. We find that all implementations of Wi-Fi FTM MAC address randomization are flawed and can be easily negated, allowing for deanonymization and novel tracking mechanisms (Section 3.3). Furthermore, we find Wi-Fi FTM frames expose a variety of features that can be leveraged to fingerprint the hardware device and firmware version of a client station, allowing for more targeted localization and tracking of clients (Section 3.4). Finally, we discuss the overall impact of all our findings (Section 3.5).

3.1.1 Adversarial Model

Throughout our analysis, we consider an adversary which has the goal of violating the (location) privacy of clients using the Wi-Fi FTM protocol. We assume the adversary has full knowledge of the protocol specification, given IEEE 802.11-2016 [7] is publicly available. We assume the adversary can eavesdrop and transmit arbitrary Wi-Fi frames; a rather weak assumption since an adversary can achieve this requirement using low-cost commercial off-the-shelf hardware (e.g., a TP-LINK AC600 Archer T2UH dongle which supports monitor mode and frame injection). Finally, the adversary is under no physical location constraints (i.e., the adversary can be at any arbitrary location as long as it is within range of the transmitted radio signals), and we assume the adversary does not physically displace or tamper with any hardware device, or modifies any of its software (e.g., application-layer code, drivers, or firmware).

3.1.2 Experimental Setup and Measurement Accuracy

We evaluate commercially available products including smartphones, access points, and off-the-shelf Wi-Fi cards from a wide variety of vendors. As a responding station, we test Google Wi-Fi (Qualcomm IPQ4019), Google Nest (Qualcomm QCS404), Compulab WILD (Intel AC-8260), and ASUS RT-ARCH13 (Qualcomm IPQ4018). As an initiating station, we test Google

Pixel 4 XL (Qualcomm WCN3990), Xiaomi Mi 10T 5G (Qualcomm QCA6390), and Samsung Galaxy Note 10 (Broadcom BCM4375B1) smartphones, and Compulab WILD. Furthermore, we evaluate Intel AC-8260, AC-8265, AX-200, and AX-210 as initiators and responders. In Figure 2 (left), we present an overview of the devices.

Furthermore, in Appendix A, we perform an extensive accuracy evaluation, as shown in Figure 2 (right). Our results show meter-level ranging accuracy can be achieved using their out-of-the-box configuration, and therefore serves as the expected measurement accuracy throughout the remainder of our paper.

3.2 Location Leakage of Client Stations

The distance between two stations is calculated based on the measured round-trip time (Equation 1), as well as the timestamps taken at the initiating station. Since the round-trip time (i.e., $t_4 - t_1$) is transmitted in the clear, a passive observer trivially learns this value. Notably, the observer may be located anywhere within range of the responding station in order to intercept these timestamps. Timestamps t_2 and t_3, however, are derived only by the initiating station and are not transmitted within the measurement protocol, and hence cannot be captured by an observer. Thus, in order to learn the distance, we must find a means to learn, or estimate, these timestamps. Specifically, we are interested only in their difference (i.e., $t_3 - t_2$), which we refer to as the Initiator Processing Time (IPT). If an observer is able to measure or estimate the IPT value, then it can be subtracted from the round-trip time, and hence one derives the distance measured between the stations. Rather worrisome, if the client station is measuring its distance with several responding stations, an adversary is capable to physically localize the client (e.g., using trilateration).

3.2.1 Benchmarking Wi-Fi Cards

In order to estimate the IPT (i.e., $t_3 - t_2$), we devised a technique to benchmark Wi-Fi cards. In this technique, the responder transmits a static round-trip time (i.e., $t_4 - t_1$) regardless of the physical distance between the stations. Then, we can subtract the result reported by the initiator from our own static round-trip time and determine a value for the IPT (i.e., $t_3 - t_2$). Repeating this process, we obtain an $AverageIPT$ estimate. Specifically, we implemented our own responder supporting measurement sessions in ASAP=0 mode, as seen in Figure 3. In this mode, the stations first establish their configuration parameters and run the measurement session at a later time (Section 2.1). This behavior allows us to exclude configuration and vendor-specific parameters from the first measurement result. As such, we avoid enlarged round-trip times (i.e., parameters result in more data for radio transmission) and different processing times due to vendor-specific implementations (e.g., due to its parameter processing) and hence $AverageIPT$ estimates. However, we find Android does not support the configuration of ASAP=0 mode, and therefore requires us to benchmark it under its default ASAP=1 mode. In this mode, we have to discard the first measurement result in order to exclude the configuration and vendor-specific parameters. Fortunately, we can discard a result by transmitting a response with empty timestamps.

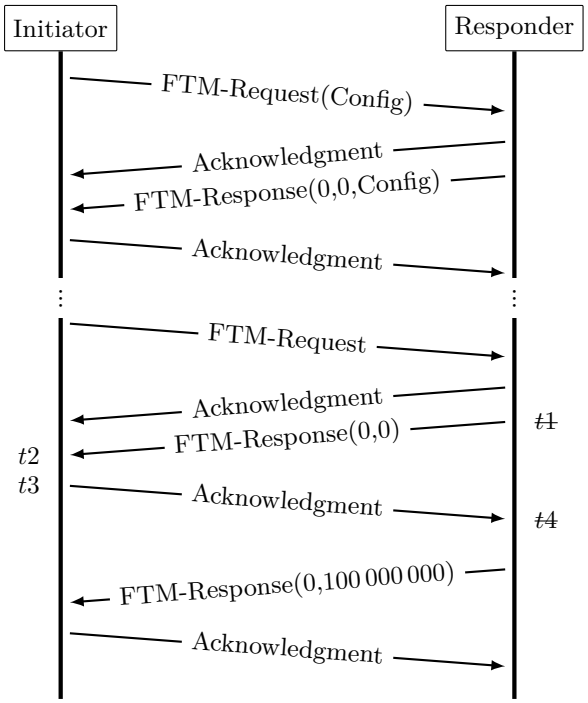


Fig. 3. Wi-Fi Fine Timing Measurement session with ASAP=0. The responder spoofs a round-trip time of $t4 = 100\,000\,000$ ps, allowing us to benchmark an average processing time of $t3 - t2$.

Having obtained an $AverageIPT$ estimate, one can calculate the measured distance from a single response frame using the following equation:

$$2d = d((t_4 - t_1) - AverageIPT) \tag{2}$$

Where function $d()$ maps picoseconds to its respective distance at the speed of light, and the result is twice the distance due to it being the round-trip distance, i.e. $(t_2 - t_1) + (t_4 - t_3)$. The expected accuracy of Equation 2 is tied to the standard deviation of $AverageIPT$.

Table 1. Benchmarking and location leakage results for initiating stations (Qualcomm WCN3990, Qualcomm QCA6390, Broadcom BCM4375, Intel AC-8260, Intel AX-200, Intel AX-210). We evaluate the accuracy at a number of distances and present the offset from our prediction to the reported distance in meters.

Card	Avg. IPT	STD	5 m	10 m	15 m	20 m
WCN3990	73 906 ns	1 ns	-1.06	-0.58	-1.02	-1.17
QCA6390	71 954 ns	8 ns	10.54	10.79	10.10	10.76
BCM4375	72 068 ns	9 ns	10.67	10.55	10.68	10.43
AC-8260	73 416 ns	40 ns	1.31	1.89	1.61	2.23
AX-200	71 714 ns	37 ns	1.83	1.20	1.90	2.35
AX-210	73 370 ns	60 ns	0.53	0.29	1.16	1.08

3.2.2 Evaluation

In order to obtain an $AverageIPT$, we benchmark the Wi-Fi cards by running up to 1 000 sessions. We note that in order to benchmark, the firmware needs to report sufficiently large distance measurements. That is, the firmware must not filter out any measurements results exceeding some pre-defined thresholds. Fortunately, for every card we tested, there is a suitable firmware version [38]. In Table 1, we present the results together with their standard deviation. Furthermore, we find that Wi-Fi cards of the same make yield the same benchmarking results. Specifically, we tested 2x Qualcomm WCN3990, 2x Intel AC-8260, and 4x Intel AX-200. Notably, we find Intel has a noticeably larger standard deviation than Qualcomm (for example, 40 ns for Intel AC-8260 and merely 1 ns for Qualcomm WCN3990). Potentially, this is a countermeasure implemented by vendors in which a random time delay is added before transmitting an acknowledgment. As a result, our distance estimations will yield larger standard deviations as well. However, in practice clients run multiple measurements per session (e.g., eight on Android) giving the observer multiple results to take an average from.

Given an $AverageIPT$, we can evaluate the accuracy by placing the stations at various distances (5, 10, 15, and 20 meter) using a Compulab WILD and Google Nest as the responding station, and recover the measured distance. Note the observer is allowed to physically be located anywhere within reception of the responding station, since the location is leaked in part through the response frames transmitted in the clear. In Table 1, we present our results and list the offset from our prediction compared to the reported value. For Qualcomm WCN3390, Intel AC-8260, AX-200, and AX-210 we find that we can learn the distance with meter-level accuracy, at any given distance. Since a passive observer is able to recover the measured distance of other stations, we exposed a rather concerning issue in regards of a client's location privacy. However, we find Qualcomm QCA6390 and Broadcom BCM4375 have a consistent offset from the expected value, i.e., close to a 10 meter overestimate. Since these Android devices are evaluated under ASAP=1 mode, we conjecture proprietary firmware implementations behave differently to our response frame with empty timestamps, introducing minor time offsets to their $AverageIPT$ value. That is, a client may follow a different code execution path when the response frame has to be discarded. Since the offset remains static across any given distance, one can correct for the offset without hindering the performance of our distance inference technique.

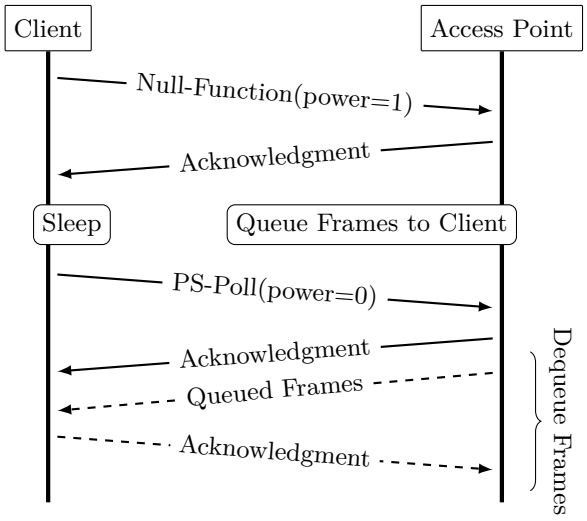


Fig. 4. Power management procedure allowing a client to enter sleep-mode and dequeue frames using Power-Save (PS) Polling.

3.3 Flawed MAC Address Randomization

We inspect all MAC address randomization implementations as provisioned by the standard (Section 2.1.2). Specifically, we manually inspect all frames sent by a station under a variety of circumstances, and identify side-channels which allow a passive observer to recover the original MAC address of a client, link subsequent anonymous Wi-Fi FTM sessions to a unique client, and deploy novel tracking mechanisms (Section 3.3.4).

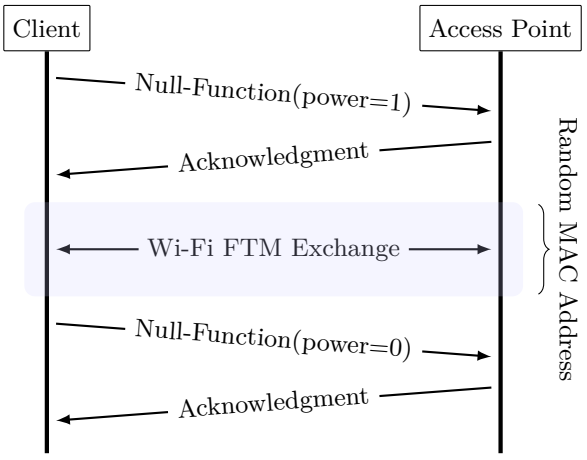


Fig. 5. Intel wraps Wi-Fi FTM exchanges in power management frames using their non-random MAC address, exposing its value.

3.3.1 Power Management Side-Channels

Amendment IEEE 802.11e-2005 [1] introduced Quality-of-Service (QoS) enhancements for wireless LAN networks. It includes power management features such as Unscheduled Automatic Power Save Delivery (U-APSD), allowing a client to enter sleep mode and wake up at an unscheduled time using a Power Save (PS) polling procedure. In a PS-polling procedure (Figure 4), a client station informs the AP that it is entering sleep mode by setting the power management flag in its IEEE 802.11 header. Clients can set the flag in data frames or use an empty Null-Function frame. When the client is in sleep mode, the AP queues frames bound for the client until the client notifies the AP with a PS-Poll frame.

Surveys conducted in October 2020 indicate the amendment is supported by over 98% of APs [37]. Furthermore, we conducted our own survey in September 2021 to measure the support rate among client stations by monitoring for QoS and power-management frames. Out of 1 857 unique clients, 1 566 (84%) had support for IEEE 802.11e. We note this is a lower bound as our survey was performed by walking around a residential neighborhood in Cambridge (Massachusetts, U.S.) and are within range of each client only momentarily; therefore we obtain only a snapshot of its (connection) state.

Abusing Power Management Frames

We find power management frames can lead to the exposure of the client's MAC address. Note the exposed address may be either the hardware or randomized connection-specific address (e.g., Android and iOS have a feature to use a network-specific randomized Wi-Fi MAC address [3, 5]). We find this side-channel to be present within all evaluated devices, for any firmware version. First, Qualcomm and Broadcom clients send a wake-up frame after their measurement request. The wake-up frame is sent from the connection-specific address, despite the usage of Wi-Fi FTM MAC address randomization. Second, Intel clients transmit power management frames to indicate the client goes to sleep before initiating the session, and wakes up when the measurement session completed. In Figure 5, we provide an illustration of this behavior. We conjecture clients use this technique to prevent the loss of data, as the client may switch its channel in order to run the measurement session. As a result of this behavior, the clients leak their MAC address each time a measurement session is performed, making it trivial for a passive observer to recover the connection-specific MAC address.

Interestingly, Broadcom transmits its wake-up frame on an incorrect channel. That is, the card switches to the appropriate channel for the Wi-Fi FTM measurement and only then transmits the wake-up frame to its AP, even though the AP is operating on another channel. As a result, an adversary can easily correlate the frames without the need for channel-hopping.

Additionally, we find request and wake-up frames sent by Qualcomm and Broadcom use sequential sequence numbers, allowing one to attribute the frames to a unique client even on a congested network and under MAC randomization. Correlation of sequence numbers can be generalized, resulting in our second side-channel.

3.3.2 Correlation of Sequence Numbers

All IEEE 802.11 frames make use of a sequence control field in its header, which includes a fragment and sequence number, and is used for duplicate detection and recovery [7]. Different types of traffic use different sequence number counters. If the counters are not managed properly (e.g., by resetting them) when using a newly randomized MAC address, then the sequence numbers may serve as a side-channel. That is, when sequence numbers are sequential, it becomes trivial to link distinct MAC addresses despite their address randomization. While researchers have correlated sequence numbers before [12, 36], we find the problem persists beyond the traditional use cases of MAC address randomization (e.g., scanning for networks). That is, in practice we find the sequence number counters are not properly managed for protocols requiring address randomization, and these shortcomings become more severe in the con-

text of privacy-sensitive protocols such as Wi-Fi FTM. Particularly for Wi-Fi FTM, we find that its measurement request frames use consecutive sequence numbers, despite their MAC addresses being randomized. Furthermore, we find the counter is shared with frames sent under the client's non-randomized address (e.g., data and power-management frames). We find these side-channels to be present within all Intel, Qualcomm, and Broadcom devices, for any firmware version.

As a result, the side-channel allows a passive adversary to recover the non-randomized MAC address of a client. Furthermore, it enables the adversary to link distance measurement sessions to the same seemingly anonymous client, and thereby accurately track client movements. Finally, and rather worrisome, the adversary can correlate distance measurement sessions with regular data frames, since these frames will share the same sequence number counter.

3.3.3 Non-Random MAC Address Randomization

Earlier Intel firmware versions (i.e., version 31 for the Intel AC-8260) generate predictable MAC addresses. That is, each time the firmware is loaded onto the Wi-Fi card, the randomization algorithm will start generating the same set of addresses. This is likely due to an improper seed in the algorithm's pseudo-random number generator. As a result, one can trivially predict the next randomized address generated by the client station.

3.3.4 Impact

Our various side-channels show that a passive adversary can link distinct randomized MAC addresses with a unique client, and thereby effectively negate Wi-Fi FTM MAC address randomization efforts, using power-save features as well as sequence numbering. These results have a significant impact in regards of client privacy, as this allows the adversary to (i) track Wi-Fi FTM clients even when their MAC address is randomized, (ii) link measurement sessions under randomized MAC addresses back towards a distinct client station (e.g., thereby enabling an adversary to track client movements), and (iii) link measurement sessions to (data traffic of) an (authenticated) client. In combination with our previous findings, this allows an adversary to localize a particular client which, for example, generates certain data traffic (e.g., authenticating with an attacker-controlled network). Additionally, and rather worryingly, an adversary can leverage these side-channels to expand existing tracking systems (e.g., by integrating Wi-Fi FTM frames into deanonymization algorithms [36]), or build entirely new tracking mechanisms based on the traffic generated by Wi-Fi FTM.

3.4 Fingerprinting Wi-Fi FTM Stations

In recent years, numerous techniques have been introduced for the hardware fingerprinting of Wi-Fi cards [44, 50, 55]. These raise concerns in regards of user and location privacy, which is further stressed by researchers presenting fingerprinting techniques based on Wi-Fi FTM frame exchanges [14]. In this section, we extend their work by (i) analyzing all supported firmware versions, (ii) performing active frame injection, and (iii) fingerprinting responding stations. As such we have features that are observed passively, as well as those that can be actively probed for. For example, we can send invalid frames to a station and fingerprint its responding behavior. From our analysis, we find additional parameters in Wi-Fi FTM frames to identify Wi-Fi cards, its vendor make, or expose specific user configurations. Learning which type of station is running a measurement session aids the adversary in learning the measured distance (e.g., the adversary can select the appropriate benchmark value as discussed in Section 3.2), and aids in identifying a target client station (e.g., for general tracking purposes as discussed in Section 3.3.4).

3.4.1 Fingerprinting Initiating Stations

Through the inspection of request frames sent by an initiating station, we have identified novel fingerprinting parameters, differing even depending per firmware version. In Table 2, we present an overview of all fingerprinting findings, which includes the MAC address randomization flaws identified in Section 3.3.3.

Default Parameters

Wi-Fi cards support a limited set of user-defined parameters, which in turn are supported by wireless tools (e.g., `iw` on Linux). For example, none of the tested cards supported a user-defined Min Delta FTM parameter, specifying the time between consecutive response frames, and therefore a request is sent using a pre-defined value in firmware. We find notable differences in the defaults among all vendors and their cards, as listed in Table 2.

Table 2. Fingerprinting findings for a wide-variety of Wi-Fi FTM initiating stations, listed per supported firmware version.

Wi-Fi Card	Firmware	Random MAC	ASAP-Capable	Window Check	Min Delta FTM	Num. Retrans.
Qual. WCN3990	Unknown	●	○	○	06	4
Qual. QCA6390	Unknown	●	○	○	20	5
Broad. BCM4375B1	Unknown	●	○	●	13	–
Intel AC-8260	Version 31	○	○	○	50	4
Intel AC-8260,8265	Version 34	●	●	○	60	4
Intel AC-8260,8265	Version 36	●	●	○	60	3
Intel AX-200	Version 53 & 55	●	○	○	50	15
Intel AX-200	Version 57 – 59	●	○	●	60	15
Intel AX-210	Version 62 & 63	●	○	●	60	15

Note Intel cards were tested using the same driver and kernel version. Inspecting default values therefore serves as a means of identifying specific firmware versions.

Injecting Invalid FTM Response Frames

An active fingerprinting method is to transmit spoofed response frames towards an initiating station and inspect how the station behaves. Stations may choose to terminate the session upon receipt of an invalid response frame, and thereafter stop acknowledging legitimate response frames. An adversary can use the absence of acknowledgment frames as a side-channel. For example, we can inject an invalid response frame by using out-of-bound t_1 and t_4 timestamps. Alternatively, a response frame may be considered invalid if it is received outside of the negotiated time window (i.e., Min Delta FTM), and therefore would be discarded without an acknowledgment. We find that only the latest firmware versions for the Intel AX-200 and AX-210 (i.e., version 57 and higher) and the Broadcom BCM4375B1, discard frames outside the negotiated time window.

ASAP Capabilities

Certain Intel firmware versions (i.e., version 34 and 36 for the Intel AC-8260 and AC-8265) set the ASAP-Capable flag in the FTM parameter set when initiating a new measurement request. This flags indicates that a station is capable of initiating the session immediately, however, while not specified in the standard, is to be set by a responding station. Newer firmware versions for the Intel AX-200 have corrected this mistake. Rather notably, all tested access points using a Qualcomm Wi-Fi card (e.g., Google Wi-Fi, Nest, ASUS RT-ARCH13), reject measurement requests when the ASAP-Capable flag is set. As a result, these Intel firmware images are incompatible with Qualcomm products. With this fingerprinting feature, one can additionally fingerprint the responding station (i.e., only Qualcomm Wi-Fi cards reject requests with the flag set).

Retransmissions

In practice, initiating stations can be expected to be mobile. For example, a station may move around a shopping center and use Wi-Fi FTM for localization purposes. As such, it will occur that the initiating station makes a request to a responding station that is no longer within range. In said event, the responding station will be unable to acknowledge the request, which results in the initiating station retransmitting the request. Note that an adversary can achieve the same by jamming the acknowledgments from a responding station [48]. Measuring the number of attempted retransmissions gives us a new fingerprinting parameter, given the notable differences among firmware versions, as listed in Table 2. Unlike Intel and Qualcomm, Broadcom does not need to retransmit its measurement requests. This is due to Broadcom sending a request-to-send frame before initiating the session (avoiding frame collisions and ensuring the AP is within range). If no clear-to-send response is received, the session will not be initiated. Similarly for responding stations, one can measure the number of retransmissions made for a response frame that is not acknowledged by the initiator; note an adversary can simply spoof a request frame to trigger responses.

3.4.2 Fingerprinting Responding Stations

Through the inspection of response frames sent by a responding station, one can quickly infer from which make its Wi-Fi card is. Most prominently, a response frame may be sent from its factory MAC address, as well as include vendor-specific information elements. While these trivially reveal the vendor, we are interested in identifying fingerprinting parameters beyond these, which could be firmware-specific. For example, earlier we showed how Qualcomm Wi-Fi cards do not accept measurement requests with the ASAP-Capable flag set. Findings like these will enable an adversary to, for example, better mimic a responding station as to deceive initiating stations, yielding clear security and privacy risks.

Dialog Tokens and Timestamps

At its surface, response frames show two major differences based on its vendor. First, we observe the construction of dialog tokens. Whereas Qualcomm cards respond each sessions with a dialog token value of one, Intel cards use sequential tokens. That is, Intel increases the dialog token by one for each new session. As a minor side-effect, this reveals how many requests the Intel responding station has responded to, which may be of future interest to an adversary. Second, we observe the construction of timestamps. In the first response frame, no timestamps are shared, since a full measurement has not been taken (i.e., the responding station has not measured a t_4 for the acknowledgment yet). However, all Intel cards set the ToD timestamp (i.e., t_1) with a value that increases consecutively for each new session. Additionally, Intel AX-200 cards may set the initial t_4 timestamp to the last round-trip time measured for the client station. This behavior yields a privacy risk in potential next-generation protocols using e.g., some form of opportunistic wireless encryption. That is, an adversary could spoof the MAC address of an initiating station, make a measurement request, and obtain the last-measured round-trip time. We do not observe this finding with Qualcomm Wi-Fi cards. Next, Qualcomm cards set their t_1 and t_4 timestamps according to some internal clock, whereas Intel responds with a t_1 timestamp of zero, and the t_4 timestamp set to the round-trip time. Finally, Qualcomm and Intel cards report timestamps with a precision of 50 and 1 picosecond respectively.

3.5 Discussion

Our analysis exposed numerous shortcomings in the Wi-Fi FTM standard and implementations, having a significant impact on their user and location privacy. Based on our findings, a passive adversary can launch numerous privacy-violating attacks against any client. Since clients broadcast their measurement requests, they are unable to hide their presence, and thus are always at risk. For example, if a client station runs measurements with several responding stations (e.g., APs), an adversary could use trilateration to recover the physical location of the client station. Additionally, even without benchmarking, an observer can track changes in round-trip times to detect the movement of a client (e.g., if the round-trip time decreases by some 6.66 ns, then the client station moved one meter closer to the responding station). Furthermore, an adversary is capable of targeting and tracking distinct clients; even when the client uses randomized MAC addresses, or when numerous other stations are operating on the radio channel. Thus, regardless of the number of clients and APs present in the network, an adversary can perform targeted attacks, and indefinitely track the client station. This is due to our findings negating Wi-Fi FTM MAC address randomization and our fingerprinting techniques, allowing an adversary to identify even which firmware version a client is running.

4 Privacy-Preserving Positioning

In this section, we present a fully passive and privacy-preserving self-positioning system, hiding the presence of a client. Our real-world evaluation achieves meter-level accuracy, equaling that of traditional Wi-Fi FTM.

4.1 Motivation and Goals

In the previous section, we performed a privacy analysis and learned Wi-Fi FTM fails to provide sufficient location privacy to its clients. As a result, its clients are at risk of leaking privacy-sensitive information, enabling an adversary to physically locate and track any client. In order to address these shortcomings, we are motivated to present and evaluate the design for a fully passive and privacy-preserving positioning system. It is our goal to design a fully passive system that hides the presence of a client. In addition to preserving privacy, it increases the overall scalability of the positioning system. Scalability limitations are common in indoor positioning systems due to their bi-directional communication requirements, and Wi-Fi FTM is no exception. In Section 4.2, we identify what these limitations are in practice, since this limitation further motivates the need for a passive self-positioning system. Furthermore, it is our goal to leverage existing Wi-Fi FTM infrastructure, and require no hardware changes. Under these goals, we ensure our positioning system can be deployed on existing devices. We present our design in Section 4.3, and our evaluation in Section 4.4 shows we are able to achieve meter-level accuracy, equaling that of traditional Wi-Fi FTM.

4.2 Scalability Limitations in Wi-Fi FTM

Indoor positioning systems are often restricted by their scalability limitations, for example due to its bi-

directional communication and resulting spectrum consumption. Recall that, in order for a Wi-Fi FTM client to measure its distance with another station, a minimum of three frames and their acknowledgments are required (i.e., a request and two response frames for a single-shot measurement). Researchers found that a measurement may take up to 30 ms to complete, and as such, a responding station can support approximately 30 single-shot measurements per second [8]. However, the total duration of a session largely depends on the configured parameters (e.g., the Samples Per Burst (SPB) and Min Delta FTM parameters), and thus may vary significantly per its respective setup. Since a single distance measurement is insufficient for determining a position, multiple such measurements have to be performed with several responding stations, potentially operating on the same wireless channel. This introduces a significant overhead into the wireless channel and its available bandwidth, and therefore the Wi-Fi FTM standard scales poorly. For example, hundreds of visitors to a stadium, festival, or shopping center, would be unable to simultaneously perform distance measurements due to these scalability limitations.

In practice, we find numerous other limitations within Wi-Fi FTM and its implementations. First, there are limitations to how many concurrent sessions a responding station will support. For example, we find the latest firmware versions of the Intel AX-200 and AX-210 (i.e., version 57 onward) support only 10 concurrent sessions. Similarly, earlier Intel firmware versions (i.e., version 31 to 55) support up to 32 concurrent sessions, and all tested Qualcomm-based APs support 16 concurrent sessions. Once the maximum number of concurrent sessions is reached, no more are accepted until at least one completes. In a congested network with high demand for measurement requests, this is bound to be a limitation. Furthermore, an adversary can leverage this to perform a denial-of-service by never completing its measurement sessions (e.g., using ASAP=0). Second, we find responding stations reject ASAP=1 requests in a congested network, and instead recommend the initiating station to run ASAP=0 measurements, potentially indefinitely delaying the starting time of the measurement session.

4.3 Design of the Self-Positioning System

We present the design for a fully-passive, privacy-preserving, and scalable Wi-Fi self-positioning system. Our design hides the presence of a client, leverages existing Wi-Fi infrastructure, requires no hardware changes, and significantly improves the overall scalability of Wi-Fi positioning systems. We note that, in [29], the authors presented a passive localization system based on Time-Difference of Arrival (TDoA). In our paper, we extend their work with an hyperbolic solution for joint positioning, and extend their simulation efforts by performing the first real-world evaluation in Section 4.4.

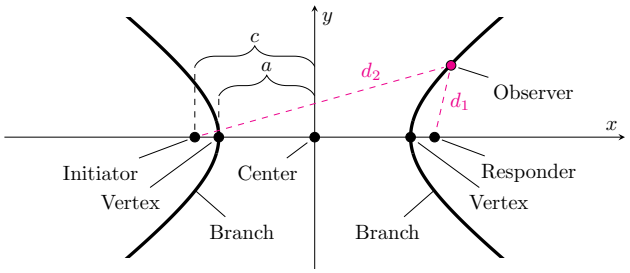


Fig. 6. Hyperbolic localization using Wi-Fi FTM stations. Observers derive a hyperbolic branch with constant $|d_2 - d_1|$.

Throughout history, several hyperbolic navigation systems have been proposed that allowed a receiver to determine its position based on the time difference between two radio signals. A notable example is LORAN, which was developed during World War II [30]. In our design, we apply this concept to round-trip time based systems build upon existing Wi-Fi FTM infrastructure.

4.3.1 Hyperbolic Localization

In hyperbolic localization systems, a receiving station calculates the time difference between two radio signals. Based on the difference, the station obtains an infinite number of possible locations which form a hyperbolic curve. That is, the resulting hyperbola is the set of all points in a plane for which the absolute value of the differences from two distinct points is constant. The distinct points, named foci, are connected by a line segment named the transverse axis, and the midpoint of the axis is the center of the hyperbola. The hyperbola, resulting in two disconnected branches, intersect the transverse axis in two points named the vertices. In Figure 6, we provide an illustration of hyperbolic localization, where the foci are represented by some initiating and responding Wi-Fi station. Any point on the branch has a constant difference of distances to the foci; i.e., value $|d_2 - d_1|$ is always constant. The hyperbola using a horizontal transverse axis with center (h, k) is defined by equation:

$$\frac{(x-h)^2}{a^2} - \frac{(y-k)^2}{b^2} = \frac{(x-h)^2}{a^2} - \frac{(y-k)^2}{c^2-a^2} = 1 \quad (3)$$

For $a \neq 0$ and $b \neq 0$. Solving for y, we obtain equation:

$$y = \frac{b\sqrt{-a^2 + h^2 - 2hx + x^2}}{a} + k \tag{4}$$

If the receiving station, named observer or observing station in Figure 6 and henceforth, is positioned on the transverse axis it can determine the intersection with the hyperbola and determine its measured distance. In order to perform self-positioning, the observer needs to measure its differential distance with several distinct stations (e.g., access points) and obtain a set of hyperbola. Based on the intersection of these hyperbola, the observing station can determine its precise location.

4.3.2 Hyperbolic Localization with Wi-Fi FTM

An observer is able to leverage round-trip time based systems such as Wi-Fi FTM to measure its differential distance to both foci (i.e., the initiator and responder). Using the obtained differential, the observer applies the hyperbolic localization technique to determine its precise location. In essence, differential distance D_{SR} with the sender and receiver is calculated by subtracting the time-of-flight between the sender and observer named T_{SO}, and the receiver and observer named T_{RO}:

$$D_{SR} = c * (T_{SO} - T_{RO}) \tag{5}$$

Where c is the speed of light. These direct values, however, are unknown to an observing station. Fortunately, an observing station is assumed to have knowledge of additional information, allowing us to rewrite the equation. First, the observer knows the distance between the initiating and responding station, named T. For example, access points can share their coordinates using Location Configuration Information (LCI) and Location Civic Report (LCR) information elements fields [7]. Second, the observer knows the round-trip times that were measured by the initiating and responding station. This is due to the open nature of the protocol, in which timestamps t_1 and t_4 are shared in the clear by the responding station in its consecutive response frames. Third, an observer can measure the time-of-arrival of the response frames sent by the responding station, and the acknowledgments sent by the initiating station, named t_5 and t_6 respectively. Given the available information, we can rewrite Equation 5:

$$D_{SR} = c * (t_5 - t_6 - T - (t_1 - t_4)) \tag{6}$$

An observing station can determine all values entirely passively, by decoding the response frames sent by a responding station, and taking its own time-of-arrival timestamps. Note that any local processing time at the initiating station would yield an increased t_6 value, which is subsequently compensated by an increased t_4 timestamp value. Now, given the differential distance, we can substitute value a in Equation 3 with $D_{SR}/2$, and obtain our hyperbolic curves.

Example

Consider a simplified example where the observing station is placed exactly in the center of two access points spaced 18 meters apart. At a simplified speed of light of $3 \cdot 10^8$ m/s, a radio frequency signal travels the distance in 60 ns. Assuming a response frame is immediately acknowledged, the responding station shares timestamps $t_1 = 0$ ns and $t_4 = 120$ ns. Substituting the timestamps in Equation 6, we obtain the following equation:

$$D_{SR} = c * (t_5 - t_6 - 60ns - (0ns - 120ns))$$

The observing station derived its own time-of-arrival timestamps as $t_5 = 30$ ns and $t_6 = 90$ ns, and is then able to calculate its differential distance as follows:

$$D_{SR} = c*(30ns - 90ns - 60ns - (0ns - 120ns)) = c*(0ns)$$

Since the observing station is positioned in the middle (i.e., at the center point of the transverse axis), its differential distance to both stations indeed equals zero. With a differential distance of zero, we obtain a perpendicular line through the center point of the transverse axis, intersecting the position of our observing station.

4.4 Evaluation

We implement and evaluate the design presented in Section 4.3 by leveraging existing Wi-Fi FTM infrastructure, thereby requiring no hardware changes. In our evaluation, we conduct two real-world experiments where we perform passive distance measurements (i.e., our observing station is on the transverse axis), and perform passive self-positioning in a large meeting room. Our experiments show promising meter-level accuracy, equaling the expected accuracy of existing Wi-Fi FTM devices as shown in Appendix A.

4.4.1 Experimental Setup

We configure a Google Wi-Fi AP and a Google Pixel 4 XL smartphone to run Wi-Fi FTM measurement ses-

sions on a 5 GHz channel using 80 MHz in bandwidth. Similarly, we use the configuration with the Intel AX-200 and WILD AP. In order to simulate a real-world configuration, the measurement sessions consists of 7 distance measurements (i.e., SPB=8). The stations are placed at various distances, stationary, and within Line-of-Sight (LoS). Wi-Fi FTM is affected by NLoS conditions only at distances greater than 20 m [16], and therefore we expect no significant impact under these conditions. Initially, we tried using commodity hardware to act as the observing station. However, even with modern Wi-Fi cards such as the Intel AX-200, we are unable to obtain reliable timestamps with close-to nano-second precision. We note these cards, supporting Wi-Fi FTM, are technically capable of measuring accurate time-of-arrival timestamps, and thus their respective vendors can implement our design on existing hardware through a firmware update. Unfortunately, the precise timestamp information is not made available to their respective drivers. Thus, instead, we must find an alternative method of capturing high-precision timestamps. If one were able to obtain accurate timestamps from its Wi-Fi card, passive localization could be done in real-time, as the card can instantly decode the frames to obtain the t_1 and t_4 timestamps. Instead, we use an Ettus Research USRP X310 Software Defined Radio (SDR) to collect samples at 200 MHz. We then use MATLAB to post-process the samples. Fortunately, we do not need to decode the frames, as we are only interested in the frame time-of-arrival difference. Instead, in order to collect the round-trip timestamps t_1 and t_4 as shared by the responding station, we use a TP-LINK AC600 Archer T2UH Wi-Fi dongle. The affordable and widely-available dongle supports IEEE 802.11ac with up to 80 MHz in bandwidth, capable of decoding all frames sent by the initiating and responding stations. Using the USRP, we have a theoretical maximum precision of 5 ns, since the USRP can sample at most 200 million samples per second. This equals close-to 1.5 m of maximum theoretical accuracy we can achieve, based on a sample-level approach. However, as we will take the average over a series of consecutive measurements, this becomes less of a limitation.

Proof-of-Concept

For our evaluation, we build a proof-of-concept capable of determining frame arrival times with maximum accuracy. To this end, we use MATLAB and follow its recovery procedure for IEEE 802.11ac frames. This requires us to downsample the 200 MHz raw USRP samples to 80 MHz (i.e., 80 Msps), in order to be able to correctly detect the frames. From the recovery procedure, we determine the frame arrival times (i.e., t_5 and t_6), and calculate their time difference. Simultaneously, we use Wireshark and the TP-LINK Wi-Fi dongle to decode response frames and its round-trip times (i.e., t_1 and t_4). As such, we take an average of both sets of time differences (i.e., $t_6 - t_5$ and $t_4 - t_1$). Having obtained all timestamps, we can substitute the values of Equation 6, and apply its result into Equation 3 in order to obtain our hyperbolic curves.

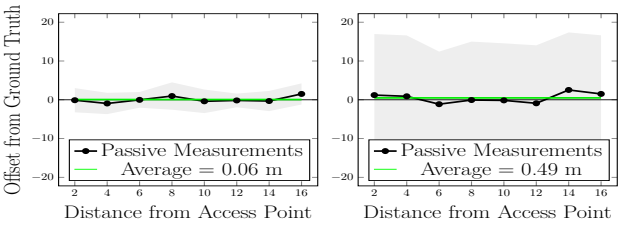


Fig. 7. An observer is able to perform passive distance measurements with meter-level accuracy, as shown for the Pixel 4 XL and Google AP (left) and Intel AX-200 and WILD AP (right).

4.4.2 Evaluating Passive Distance Measurements

We first evaluate the expected accuracy of passive distance measurements by placing the observing station on the transverse axis at a variety of distances ranging from 2 to 16 meter from the access point. Specifically, we capture 10 sessions with SPB=8, though note that a single session may be sufficient for an accurate estimate. Figure 7 presents an overview of our evaluation results, where we plot the offset from the ground truth. That is, if the ground truth distance to the access point is 10 meter, and we obtain a distance of 11 meter, then we have an overestimate or offset from the ground truth of 1 meter. Additionally, we plot the average of our results, shown in green. We have performed the evaluation for two distinct sets of devices. Figure 7 (left) presents results for the Google Pixel 4 XL and Google Wi-Fi AP, both using Qualcomm cards, and Figure 7 (right) presents results for the Intel AX-200 and Compulab WILD AP, both using Intel cards. Our results show we can achieve meter-level accuracy, as we obtain an average over-estimation of 0.06 and 0.49 meter respectively. Notably, we obtain a higher standard deviation for Intel cards, an expected vendor-specific result we addressed in Section 3.2.

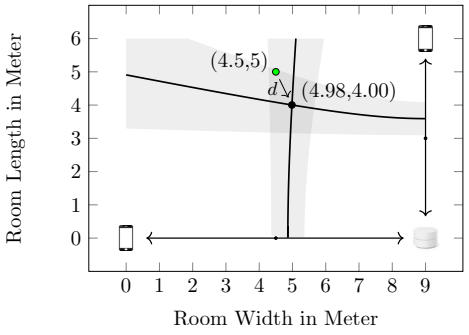


Fig. 8. Top view of a meeting room with stationary Wi-Fi FTM stations. An observing station performs passive self-positioning with meter-level precision (i.e., $d = 1.11$ meter).

4.4.3 Evaluating Passive Self-Positioning

We evaluate the accuracy of self-positioning by placing the observing station off the transverse axis. That is, we place the observing station in a meeting room, and run two sets of measurements from both the width and length of the room. Figure 8 presents an overview of the meeting room, together with the positioning results. We show the ground truth of our observing station in green (i.e., coordinates $(4.5, 5)$). From our measurements, we obtain a differential distance of 0.5926 and -0.3607 meter for the length and width of the room respectively. Calculating the intersection of both hyperbolic curves, we obtain an estimated point at $(4.98345, 4.00311)$, which is a distance $d = 1.11$ meter away from the ground truth.

The results of our real-world evaluation show we are able to achieve a major improvement towards the scalability of clients willing to run passive distance measurements or perform self-positioning, whilst maintaining its expected meter-level accuracy.

5 Discussion

We discuss common countermeasures and learn why they fail to preserve location privacy within Wi-Fi FTM. We then present security and privacy recommendations.

5.1 Countermeasures

5.1.1 Beamforming

Modern devices use beamforming technology to reduce interference and increase data communication rates. It allows a device equipped with multiple antennas to transmit highly directional signals, or *beams*, towards a specific region instead of omnidirectionally. IEEE 802.11ac standardized the sounding and feedback techniques that are used in setting up these beams, enabling compatibility among vendors. Beamforming would seem to enhance the overall privacy of any protocol, since a passive observer would be unable to capture frames if they are not within range, or the interception of, the transmitted beams. However, we find Wi-Fi FTM stations transmit their requests on the 20 MHz channel on which the AP advertises its presence (i.e., its beacon frame), before switching to a negotiated wider bandwidth channel. As a result, the request is not sent using any of the IEEE 802.11ac Very High Throughput (VHT) features, and thus transmitted without beamforming. From an adversarial perspective, the client keeps leaking privacy-sensitive information (e.g., its presence, the AP with which it communicates). As a result, clients remain exposed to attacks identified in our paper. Finally, we note security weaknesses have been demonstrated in beamforming technology. For example, the authors in [42] abused the sector sweep of IEEE 802.11ad in order to launch Man-in-the-Middle (MitM) attacks. Achieving a MitM position would trivially compromise the integrity of any ranging and localization protocol.

5.1.2 Authentication and Encryption

Security features often come with a compromise on usability and accessibility. For example, a design protecting its round-trip times may require a client to authenticate and derive cryptographic keys, exposing its presence and identity. Notably, such a design may not even achieve confidentiality goals in a typical pre-shared key network configuration (e.g., WPA2-PSK) since an adversary can recover the encryption key if the passphrase is known (e.g., the network of an indoor shopping center). Similarly, encryption is futile when a network's group key is used (since everyone on the network derives the same group key), and thus any such design requires special care. If a trade-off is made with anonymity, the design could leverage an enterprise-grade network configuration (e.g., WPA3-Enterprise) to utilize client-specific cryptographic keys and use these to, for example, encrypt and authenticate the round-trip times.

5.1.3 Hiding Transmissions

Hiding the transmission of frames using, for example, Frequency Hopping (FH) or Direct-Sequence Spread Spectrum (DSSS) techniques is no suitable countermeasure due to offline attacks [34]. For example, an adversary can record the entire frequency band and brute force spreading codes. Furthermore, today no such technology is standardized for usage within Wi-Fi networks.

5.2 Recommendations

We present security and privacy recommendations to be considered in the development and implementation of current and next-generation positioning protocols. In general, the recommendations are protocol independent, however, we highlight those which apply to Wi-Fi FTM; these recommendations do not infringe the specification and thereby preserve compatibility among vendors. We encourage vendors to implement our recommendations in their existing Wi-Fi FTM implementations. Today, researchers, industry, and enthusiasts are building real-world systems using Wi-Fi FTM, and in anticipation of the standard being deprecated and replaced by Wi-Fi NGP in 2023, these systems should be kept secure.

5.2.1 Random Time Delays

In a round-trip time measurement design such as Wi-Fi FTM, the initiating station can add a random time delay before transmitting acknowledgments, artificially enlarging the round-trip time by a value known only to itself. While it is not a fundamental countermeasure, it remains a successful means to prevent reliable benchmarking techniques, and therefore protects against a passive observer learning the measured distance. Furthermore, this countermeasure does not rely on the infrastructure and therefore it does not need to be trusted by the client in order to ensure privacy. In practice, an implementation can, for example, randomize the Short Interframe Space (SIFS) interval. In IEEE 802.11ac, the SIFS interval is defined at 16 μs, and a mere 1 μs variation yields a 150 meter one-way distance offset. Potentially some form of delay is implemented by Intel Wi-Fi cards, as we observed a large standard deviation in Table 1. Arguably, even an offset in the 30 ns range is too small as it yields 4.5 meter in one-way distance. Furthermore, such an offset can be averaged over time, especially when the client is stationary for a number of consecutive sessions or the session is configured for a large number of measurements (i.e., Samples Per Burst).

5.2.2 Anonymity and MAC Address Randomization

Any design should allow for an anonymous mode, wherein a client is required to utilize a fully randomized MAC address. Each time a new address is generated, its sequence number counters should be reset or randomized in order to prevent deanonymization (a recommendation often given by researchers to prevent MAC layer based tracking systems [36]), and external network features such as power management scheduling should be ignored during any measurement sessions. Similarly, an implementation should avoid any kind of sequential counter (e.g., dialog tokens in a Wi-Fi FTM response frame) in order to minimize the risk of fingerprinting and identifying a unique client. Notably, when the client has established a connection with the responding station (e.g., an access point), no state information of that connection should be reused within the anonymous mode.

6 Related Work

In recent years, researchers presented ranging and localization systems using the widespread availability of Wi-Fi networks [21, 22, 25–27, 39, 49, 51, 53]. They have studied the security of public WLAN-based positioning systems [43], and the quantification of location privacy [40]. In [34], the authors discuss the location privacy of distance bounding protocols, for example, they analyze which information leaks about the location and distance between two communicating partners [34]. With any positioning system comes the challenge of scalability, and thus researchers discussed the scalability of wireless positioning systems [24]. In order to address this, the authors in [29] presented a passive localization system using Time-Difference of Arrival (TDoA). In our paper, we extend their work with an hyperbolic solution for joint positioning, and we extend their simulation efforts by implementing the design and performing real-world experiments. Further improvements to positioning systems are continuously being made. For example, frequency diversity was proposed to enable a higher accuracy in indoor positioning systems [15], and the IEEE is in the process of standardizing Wi-Fi Next Generation Positioning (Wi-Fi NGP; IEEE 802.11az [18]) in 2023.

Since its standardization by the IEEE, Wi-Fi Fine Timing Measurement (Wi-Fi FTM; IEEE 802.11mc [7]) has seen wide-deployment and adoption in numerous systems [10, 13, 19, 35, 41, 54], with notable applications such as indoor positioning [11, 20, 52], as well as vehicular positioning [17]. Rather troubling, the protocol is used for security purposes too, for example, as part of network onboarding of Internet of Things (IoT) devices [23]. With the release of commercial products supporting Wi-Fi FTM, researchers performed security analyses [38] and accuracy evaluations [11, 52] indicating meter-level accuracy in low-multipath environments [16]. We extend their work by evaluating modern devices supporting wider bandwidth channels, and additionally show two-meter level accuracy can be achieved on low-multipath 20 MHz bandwidth channels, matching evaluation results obtained in [11]. Furthermore, even though Wi-Fi MAC address randomization has been studied at length [12, 28, 47], and countermeasures to deanonymization and tracking mechanisms were proposed (e.g., randomization of sequence numbers) [36], we find these have not seen their adoption in protocols such as Wi-Fi FTM. Next, researchers have shown fingerprinting techniques to uniquely identify Wi-Fi cards [44, 50, 55], and in [14], the authors performed fingerprinting by including Wi-Fi FTM configuration parameters. In our paper, we extend their work by analyzing specific firmware versions, performing active frame injection, and fingerprinting responding stations. Finally, clients of a positioning system may be susceptible to Man-in-the-Middle (MitM) attacks, even if the network uses IEEE 802.11ad beamforming [42] and when the beacon frames of an AP are not properly protected [45, 46], trivially compromising the security and privacy.

7 Conclusion

In this paper, we performed the first privacy analysis of Wi-Fi Fine Timing Measurement (FTM) as defined in IEEE 802.11mc. We identified numerous weaknesses allowing a passive observer to recover the distance measured by any client, and consequently perform localization. Furthermore, we identified flaws in Wi-Fi FTM MAC address randomization, presented techniques to fingerprint stations with firmware-specific granularity, and identified practical scalability limitations. Altogether, these findings allow an adversary to localize and track individual clients. With the need for a privacy-preserving positioning system, we presented and evaluated a hyperbolic localization system offering meter-level accuracy. Our design hides the presence of a client, leverages existing Wi-Fi FTM infrastructure, requires no hardware changes, and improves overall scalability. Finally, we discussed countermeasures and privacy recommendations for Wi-Fi FTM and Wi-Fi Next Generation Positioning (Wi-Fi NGP; IEEE 802.11az).

Responsible Disclosure

Vulnerabilities are disclosed to their respective vendors. Qualcomm and Android mitigated Wi-Fi FTM MAC address randomization side-channels (Section 3.3.1) in their February 2021 Security Bulletins (CVE-2020-11281 and CVE-2020-11287). Intel mitigated predictable MAC address randomization (Section 3.3.3) in its September 2021 Security Advisory (CVE-2021-0053). Qualcomm mitigated the ASAP-Capability vulnerability (Section 3.4.1) in its February 2021 Security Bulletin (CVE-2020-11280) and is addressed by ASUS.

Availability

We publish a repository [1] where we track support for Wi-Fi FTM, provide practical instructions on how to configure hardware and software, track security and privacy vulnerabilities and their patches, and share code.

Acknowledgments

We would like to thank our shepherd Travis Mayberry and reviewers for their valuable feedback. This work was partially supported by NSF grant 1850264.

References

[1] IEEE Std 802.11e. *Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements*, 2005.

[2] Wi-Fi Alliance. Wi-fi aware. https://www.wi-fi.org/discover-wi-fi/wi-fi-aware, 2020 (Accessed 3 December 2020).

[3] Android. Privacy: Mac randomization. Accessed 03/04/2020 from https://source.android.com/devices/tech/connect/wifi-mac-randomization, 2020.

1 https://github.com/domienschepers/wifi-ftm

[4] Android. Wi-fi location: ranging with rtt | android developers. https://developer.android.com/guide/topics/connectivity/wifi-rtt, 2020 (Accessed 18/06/2020).

[5] Apple. Use private wi-fi addresses in ios 14, ipados 14, and watchos 7. Retrieved 1 December 2020 from https://support.apple.com/en-us/HT211227, 2020.

[6] IEEE Standards Association et al. Ieee std 802.11-2012, ieee standard for local and metropolitan area networks—part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, 2012.

[7] IEEE Standards Association et al. Ieee std 802.11-2016, ieee standard for local and metropolitan area networks—part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, 2016.

[8] Leor Banin, Ofer Bar-Shalom, Nir Dvorecki, and Yuval Amizur. Scalable wi-fi client self-positioning using cooperative ftm-sensors. *IEEE Transactions on Instrumentation and Measurement*, 68(10):3686–3698, 2018.

[9] Leor Banin, Ofer Bar-Shalom, Nir Dvorecki, and Yuval Amizur. High-accuracy indoor geolocation using collaborative time of arrival, 2019.

[10] Leor Banin, Uri Schatzberg, and Yuval Amizur. Wifi ftm and map information fusion for accurate positioning. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2016.

[11] Markus Bullmann, Toni Fetzer, Frank Ebner, Markus Ebner, Frank Deinzer, and Marcin Grzegorzek. Comparison of 2.4 ghz wifi ftm-and rssi-based indoor positioning methods in realistic scenarios. *Sensors*, 20(16):4515, 2020.

[12] Ellis Fenske, Dane Brown, Jeremy Martin, Travis Mayberry, Peter Ryan, and Erik Rye. Three years later: A study of mac address randomization in mobile devices and when it succeeds. *Proceedings on Privacy Enhancing Technologies*, 3:164–181, 2021.

[13] Guangyi Guo, Ruizhi Chen, Feng Ye, Xuesheng Peng, Zuoya Liu, and Yuanjin Pan. Indoor smartphone localization: A hybrid wifi rtt-rss ranging approach. *IEEE Access*, 7:176767–176781, 2019.

[14] Jérôme Henry and Nicolas Montavont. Fingerprinting using fine timing measurement. In *Proceedings of the 17th ACM International Symposium on Mobility Management and Wireless Access*, pages 49–56, 2019.

[15] Berthold KP Horn. Doubling the accuracy of indoor positioning: Frequency diversity. *Sensors*, 20(5):1489, 2020.

[16] Mohamed Ibrahim, Hansi Liu, Minitha Jawahar, Viet Nguyen, Marco Gruteser, Richard Howard, Bo Yu, and Fan Bai. Verification: Accuracy evaluation of wifi fine time measurements on an open platform. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 2018.

[17] Mohamed Ibrahim, Ali Rostami, Bo Yu, Hansi Liu, Minitha Jawahar, Viet Nguyen, Marco Gruteser, Fan Bai, and Richard Howard. Wi-go: accurate and scalable vehicle positioning using wifi fine timing measurement. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pages 312–324, 2020.

[18] IEEE. Ieee p802.11 - next generation positioning study group. Accessed 29/03/2020 from http://www.ieee802.org/11/Reports/tgaz_update.htm, 2020.

[19] Shazal Irshad, Eric Rozner, Apurv Bhartia, and Bo Chen. Rethinking wireless network management through sensor-driven contextual analysis. In *Proceedings of the 21st ACM HotMobile Workshop*, pages 92–97, 2020.

[20] Nicolas Jathe, Michael Lütjen, and Michael Freitag. Indoor positioning in car parks by using wi-fi round-trip-time to support finished vehicle logistics on port terminals. *IFAC-PapersOnLine*, 52(13):857–862, 2019.

[21] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. Spotfi: Decimeter level localization using wifi. In *ACM SIGCOMM computer communication review*, volume 45, pages 269–282. ACM, 2015.

[22] Steven Lanzisera, David Zats, and Kristofer SJ Pister. Radio frequency time-of-flight distance measurement for low-cost wireless sensor localization. *IEEE Sensors Journal*, 11(3):837–845, 2011.

[23] Byung Moo Lee, Mayuresh Patil, Preston Hunt, and Imran Khan. An easy network onboarding scheme for internet of things networks. *IEEE Access*, 7:8763–8772, 2018.

[24] Marc Llombart, Marc Ciurana, and Francisco Barcelo-Arroyo. On the scalability of a novel wlan positioning system based on time of arrival measurements. In *2008 5th Workshop on Positioning, Navigation and Communication*, 2008.

[25] Ahmed Makki, Abubakr Siddig, Mohamed Saad, and Chris Bleakley. Survey of wifi positioning using time-based techniques. *Computer Networks*, 88, 2015.

[26] Ahmed Makki, Abubakr Siddig, Mohamed Saad, Joseph R Cavallaro, and Chris J Bleakley. Indoor localization using 802.11 time differences of arrival. *IEEE Transactions on Instrumentation and Measurement*, 65(3):614–623, 2015.

[27] Andreas Marcaletti, Maurizio Rea, Domenico Giustiniano, Vincent Lenders, and Aymen Fakhreddine. Filtering noisy 802.11 time-of-flight ranging measurements. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pages 13–20. ACM, 2014.

[28] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. A study of mac address randomization in mobile devices and when it fails. *Proceedings on Privacy Enhancing Technologies*, 2017(4):365–383, 2017.

[29] Israel Martin-Escalona and Enrica Zola. Passive round-trip-time positioning in dense ieee 802.11 networks. *Electronics*, 9(8):1193, 2020.

[30] JA Pierce. An introduction to loran. *Proceedings of the IRE*, 34(5), 1946.

[31] Google Play. Wifirttlocator app. https://play.google.com/store/apps/details?id=com.google.android.apps.location.rtt.wifirttlocator, Accessed 15/09/2021.

[32] Google Play. Wifirttscan app. https://play.google.com/store/apps/details?id=com.google.android.apps.location.rtt.wifirttscan, Accessed 15/09/2021.

[33] Google Play. Wifinanscan app. https://play.google.com/store/apps/details?id=com.google.android.apps.location.rtt.wifinanscan, Accessed 24/03/2021.

[34] Kasper Bonne Rasmussen and Srdjan Čapkun. Location privacy of distance bounding protocols. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 149–160, 2008.

[35] Maurizio Rea, Traian Emanuel Abrudan, Domenico Giustiniano, Holger Claussen, and Veli-Matti Kolmonen. Smartphone positioning with radio measurements from a single wifi access point. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 200–206, 2019.

[36] Pieter Robyns, Bram Bonné, Peter Quax, and Wim Lamotte. Noncooperative 802.11 mac layer fingerprinting and tracking of mobile devices. *Security and Communication Networks*, 2017, 2017.

[37] Domien Schepers, Aanjhan Ranganathan, and Mathy Vanhoef. Let numbers tell the tale: measuring security trends in wi-fi networks and best practices. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 100–105, 2021.

[38] Domien Schepers, Mridula Singh, and Aanjhan Ranganathan. Here, there, and everywhere: security analysis of wi-fi fine timing measurement. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 78–89, 2021.

[39] Ian Sharp and Kegen Yu. Indoor toa error measurement, modeling, and analysis. *IEEE Transactions on Instrumentation and Measurement*, 63(9), 2014.

[40] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. Quantifying location privacy. In *2011 IEEE symposium on security and privacy*, pages 247–262. IEEE, 2011.

[41] Minghao Si, Yunjia Wang, Shenglei Xu, Meng Sun, and Hongji Cao. A wi-fi ftm-based indoor positioning method with los/nlos identification. *Applied Sciences*, 10(3):956, 2020.

[42] Daniel Steinmetzer, Yimin Yuan, and Matthias Hollick. Beam-stealing: intercepting the sector sweep to launch man-in-the-middle attacks on wireless ieee 802.11 ad networks. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 12–22, 2018.

[43] Nils Ole Tippenhauer, Kasper Bonne Rasmussen, Christina Pöpper, and Srdjan Čapkun. Attacks on public wlan-based positioning systems. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, 2009.

[44] O Ureten and Nur Serinken. Bayesian detection of wi-fi transmitter rf fingerprints. *Electronics Letters*, 41(6):373–374, 2005.

[45] Mathy Vanhoef, Prasant Adhikari, and Christina Pöpper. Protecting wi-fi beacons from outsider forgeries. In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 155–160, 2020.

[46] Mathy Vanhoef, Nehru Bhandaru, Thomas Derham, Ido Ouzieli, and Frank Piessens. Operating channel validation: preventing multi-channel man-in-the-middle attacks against protected wi-fi networks. In *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 34–39, 2018.

[47] Mathy Vanhoef, Célestin Matte, Mathieu Cunche, Leonardo S Cardoso, and Frank 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*, pages 413–424, 2016.

[48] Mathy Vanhoef and Frank Piessens. Advanced wi-fi attacks using commodity hardware. In *Proceedings of the 30th ACSAC Conference*, pages 256–265, 2014.

[49] Deepak Vasisht, Swarun Kumar, and Dina Katabi. Decimeter-level localization with a single wifi access point. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 165–178, 2016.

[50] Tien Dang Vo-Huu, Triet Dang Vo-Huu, and Guevara Noubir. Fingerprinting wi-fi devices using software defined radios. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 3–14, 2016.

[51] Sigit Basuki Wibowo, Martin Klepal, and Dirk Pesch. Time of flight ranging using off-the-self ieee802. 11 wifi tags. In *Proceedings of the International Conference on Positioning and Context-Awareness (PoCA'09)*, 2009.

[52] Shihao Xu, Ruizhi Chen, Yue Yu, Guangyi Guo, and Lixiong Huang. Locating smartphones indoors using built-in sensors and wi-fi ranging with an enhanced particle filter. *IEEE Access*, 7:95140–95153, 2019.

[53] Chouchang Yang and Huai-Rong Shao. Wifi-based indoor positioning. *IEEE Communications Magazine*, 53(3):150–157, 2015.

[54] Yue Yu, Ruizhi Chen, Liang Chen, Guangyi Guo, Feng Ye, and Zuoya Liu. A robust dead reckoning algorithm based on wi-fi ftm and multiple sensors. *Remote Sensing*, 11(5):504, 2019.

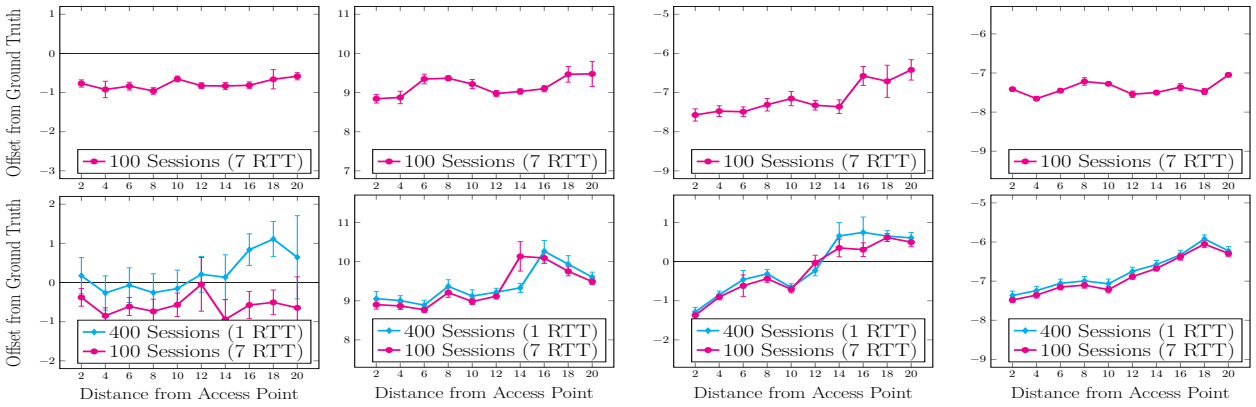
[55] HL Yuan and AQ Hu. Preamble-based detection of wi-fi transmitter rf fingerprints. *Electronics letters*, 46(16):1165–1167, 2010.

A Wi-Fi FTM Accuracy

In order to understand the accuracy provided by Wi-Fi FTM (IEEE 802.11mc), we perform an extensive accuracy evaluation. In [16], the authors demonstrated that meter-level accuracy can be obtained in low-multipath environments. However, few wide-bandwidth configurations were evaluated, in part due to it not being supported by commercial products existing at the time. In our work, we extend their evaluation with a wide-variety of modern devices and physical-layer configurations.

A.1 Experimental Setup

We evaluated commercially available products and off-the-shelf Wi-Fi cards. As responding stations, or APs, we evaluate Google Wi-Fi (Qualcomm IPQ4019), Google Nest (Qualcomm QCS404), Compulab WILD (Intel AC-8260), and ASUS RT-ARCH13 (Qualcomm IPQ4018). As an initiating station, a Google Pixel 4



(a) Google Wi-Fi AP. **(b)** Google Nest AP. **(c)** Compulab WILD AP. **(d)** ASUS RT-ARCH13 AP.

Fig. 9. Accuracy evaluation using a Google Pixel 4 XL (top) and Intel AX-200 (bottom), with a variety of responding stations. Stations are configured at 80 MHz in bandwidth on a 5 GHz channel, and evaluated at distances between two and twenty meter.

XL (Qualcomm Snapdragon 855) smartphone running Android 11, Compulab WILD, NVIDIA Jetson Nano (Intel AC-8265), and a Dell Latitude E5470 and Lenovo ThinkPad P1 (Intel AX-200). Unless noted otherwise, stations are configured to a 5 GHz channel with 80 MHz in bandwidth, as this is the default configuration for modern out-of-the-box devices. In Figure 2, we presented an overview of all devices (left) and our accuracy evaluation setup (right). We placed all devices in an empty parking lot within line-of-sight of each other, at close-to 60 centimeter off the floor. We measure their out-of-the-box accuracy without performing any kind of calibration, and run each session with 7 measurements (i.e., 8 samples per burst, noted 7 RTT or SPB=8). This is the default configuration on Android and therefore a good real-world example. In addition, we use Intel cards to evaluate single-shot measurements (i.e., only a single measurement, noted 1 RTT or SPB=2) since these cards allow for more customized session configurations.

A.2 Accuracy Evaluation

In Figure 9, we present measurement accuracy plots for each combination of initiating and responding stations, at distances ranging from two to twenty meters. We plot the average measurement result, as well as its standard deviation, for its offset from the ground truth. That is, accurate systems will have their offsets to be close to zero, as they have measured the expected distance. Ideally we observe a steady offset from the ground truth, as this would allow for a simple calibration method in practice. Generally, we observe a slight under or overestimate from the ground truth distance, which are consistent in regards to their respective distance from the ground truth (e.g., a consistent 10 meter offset). It shows some form of calibration remains important in practice for most devices, as is further shown by, for example, Android which discusses techniques on how to properly perform a calibration [4]. A notable exception is the Google Pixel 4 XL and the Google Wi-Fi AP (Figure 9a, top), which out-of-the-box yields meter-level accuracy.

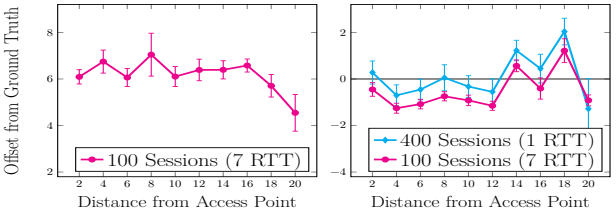


Fig. 10. Two-meter-level accuracy can be achieved on a 20 MHz bandwidth channel, as shown by the Pixel 4 XL and Intel AX-200 (left), and Intel AX-200 and Google Wi-Fi (right).

Furthermore, we performed an accuracy evaluation for systems using narrow bandwidth channels (i.e., 20 MHz). In Figure 10, we present the results for two distinct setups: a Google Pixel 4 XL and the Intel AX-200 serving as an AP on the Lenovo ThinkPad P1 (left), and the Intel AX-200 as a client with a Google Wi-Fi AP (right). Note we are unable to operate an access point in the 5 GHz band using the Intel AX-200, therefore this AP is configured to the 2.4 GHz band (unlike the Google Wi-Fi AP operating in the 5 GHz band). We observe modern Wi-Fi cards allow for up to two-meter accuracy on a 20 MHz bandwidth channel, using a legacy preamble; matching the results in [11].