Where Is My Tag? Unveiling Alternative Uses of the Apple FindMy Service

Leonardo Tonetto, Andrea Carrara
Department of Computer Science
Technical University of Munich
Munich, Germany
tonetto@in.tum.de, andrea.carrara@tum.de

Aaron Yi Ding
Dept. of Engineering Systems and Services
Delft University of Technology
Delft, Netherlands
aaron.ding@tudelft.nl

Jörg Ott
Department of Computer Science
Technical University of Munich
Munich, Germany
ott@in.tum.de

***Abstract*—Bluetooth trackers, or tags, have quickly become ubiquitous and widely supported by multiple vendors. Beyond their original design of finding lost objects, these devices have the ability to extend the capabilities of current wireless smart devices. Since its launch in 2019, Apple's FindMy enables any devices from their brand to be easily tracked by more than 1 billion active iPhones and iPads on the market. While convenient, these systems may even serve further uses, including as a result of this work, crowd sensing and a side channel for mobile communication. But they also raise privacy concerns for their users. In this paper, we demonstrate how Apple FindMy can be used as a privacy-friendly tool for crowd monitoring, and how it may inadvertently leak information on a person's location in case of deliberate tracking. Additionally, we design and evaluate a proof of concept protocol, using the Apple FindMy and a crafted tag using a simple microcontroller. We show how such system could be used to transmit information at very low bit rates, while the devices transporting the information remain unaware of this covert channel, yielding an *out of band* communication channel.**

***Index Terms*—sensing, location privacy, crowd monitoring, mobile communication, covert exfiltration**

I. Introduction

Bluetooth trackers, or tags, have become ubiquitous, being primarily used to track and find lost objects. This growing pervasiveness allowed manufacturers to create a network of tracker owners to *anonymously* report about any nearby tag, such as Tile and Apple FindMy. This crowdsourced reporting provides obvious primary benefits for its users, but also enables alternative uses for which it was not originally designed.

In this paper, we explore two such alternative uses as the main objective of our work: (1) a fully anonymous crowd sensing system and (2) a covert communication channel built on top of Apple's FindMy system: As an auxiliary finding, our work revealed potential privacy issues that could affect billions of Apple devices [1], for which the only current mitigation is disabling Bluetooth or opting-out of the FindMy service.

Crowd Sensing: Sensing and monitoring different aspects of a (large) crowd may serve numerous purposes, such as steering people flows to safety under pressing conditions. However, automated methods for crowd monitoring, such as image-based tracking, may raise privacy concerns [2]. Individuals are bothered by sensors capturing any form of personal identifiable information (PII) that, if stored permanently, may require explicit consent from those being monitored.

Current solutions to this privacy problem in crowd monitoring may rely on computing all relevant metrics at the edge [2]. However, these systems still handle PII and those being monitored simply have to trust their personal data are being dealt with appropriately. Therefore, a reliable source of crowd data while guaranteeing the privacy of its subjects is still a relevant open problem that we explore with this paper.

We use handcrafted Apple tags enabled by the reverse-engineering work of Heinrich *et al.* [1], in which single board computers and microcontrollers can be used as tags. Apple allows the owner of tags to download all location reports within a week. Through a series of comprehensive analyses, we demonstrate the capability of such system using trackers, or sensory-tags, for coarse crowd monitoring, including determining counts/density and flows.

Covert Data Channel: We also demonstrate how such tracker systems could be used to create a side channel for communication, while sending information *silently* through nearby mobile devices. Our proof-of-concept enables out-of-band communication at low bit-rate, without awareness of those partially carrying the information.

Deliberate Tracking: We explore potential privacy risks associated with Apple FindMy as a side effect of its sensing capabilities. The threats we reveal concern the possibility of exposing location information of a victim from the timestamps contained in each location report. We demonstrate their feasibility through proof of concept examples and discuss possible mitigation approaches to these threats.

Our work exposes and discusses alternative usages for an established secure system, including potential malicious ones. We present an evaluation of the Apple FindMy network and its main properties that are pertinent to the proposed solutions. Furthermore, we design and test a simple protocol, with basic characteristics to ensure a successful transmission of data with our system. Finally, we discuss the implications of this work, along with possible mitigation strategies for users and developers of similar systems. *We reported all uncovered issues to Apple several months prior to the submission of this manuscript.*

Our contributions: (1) We thoroughly analyse the timing and conditions in which location reports are sent for a lost smart tag in the Apple finder network (§ IV). (2) We demonstrate the feasibility and accuracy of using such anonymous location reports for sensing two different aspects of a crowd, namely flow and size/density, from a series of real-world measurements compared to state of the art solutions (§ VI). (3) We reveal the feasibility of timing attacks, using the Apple finder network that could reveal information about a person's whereabouts without their consent (§ VII). (4) We show such tags can be used to transmit information through a side-channel, and we name it TagComm (§ VIII). (5) Complementing the original work by Heinrich *et al.* [1], we provide open source code that enables other devices to be used as sensory-tags as well as be used for covert communication: `macOS` devices and the Amazon Echo [3], [4]. (6) We discuss the privacy and ethical implications of our work (§ IX).

II. Related Work

1) Apple Ecosystem: Recently, various papers evaluated the security of different Apple services. Analysis by Martin *et al.* of the Handoff services, that enables seamless communication between multiple Apple devices under the same iCloud account, reveals how Apple's proprietary protocol can undermine MAC address randomization and allow the identification of devices belonging to a single user [5]. Furthermore, a study by Stute *et al.* demonstrated how the Apple Wireless Direct Link (AWDL) works, including the reverse engineering of its protocols and Wireshark plugins. These examples support the importance of scrutinizing such proprietary systems that may affect users of billions of devices worldwide [6].

2) Bluetooth LE trackers: A recent study by Weller *et al.* evaluated different Bluetooth trackers and their cloud services [7]. Their study revealed a series of security issues, including privacy risks with all products tested, although it did not include Apple's FindMy as no commercial product was available at the time. Focusing exclusively on the Apple service, Heinrich *et al.* dissected how FindMy components work [1]. Their study reverse engineered the protocol used by lost devices, finders and how owners can retrieve available location reports for their tags. Their open source code was used as the foundation for our present paper.

3) Security and Privacy: Security and privacy literature has a vast number of systems that exploit different vectors to covertly exfiltrate data from systems (*e.g.* [8]). Various systems have used keyboard or HDD indicator LEDs [9], as well as inaudible (or indistinguishable) sound from speakers [10], fans or HDDs [11]. In this paper, the proof-of-concept we present enables a covert channel, through which any information can transmitted at low bit rates. To foster further research, we extend [1] by `macOS` support that runs without `root` privilege [3], [4].

4) Crowd Monitoring: Assessing and understanding large crowds has been studied with a myriad of sensors and methods, but not without its privacy implications [2]. However, several challenges are still open when it comes to scalability and integration of multiple systems towards a common decision support [12]. The COVID-19 pandemic has stimulated crowd monitoring research, for example, ensuring social distancing [13] as a valuable approach to reduce infections [14].

5) Our Work: In this paper, we further analyse the Apple FindMy service, and we present two proof-of-concept systems, one which allows coarse crowd monitoring, and other that allows side-channel communication as well as their potential risks for users' privacy. Note that, while [1] reverse engineered the client-side managing of tags, we extend our understanding of this system while exposing possible security and privacy leaks FindMy users are currently subject to.

III. Background

In this section, we present basic functionality of Apple FindMy as the underlying system for our current work. Furthermore, we present relevant concepts of Bluetooth LE.

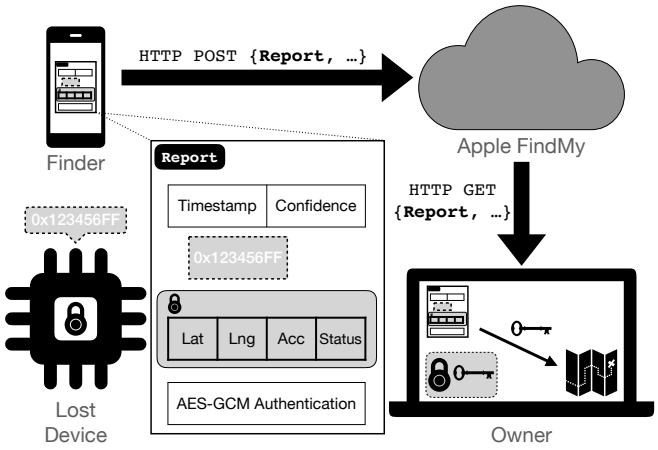


Fig. 1: Delay in sensing and reporting a tag.

A. Apple FindMy Service

This finder network was released in 2019, in which devices that explicitly opt-in are tracked through anonymous crowd-sourced location reports. When devices are marked as lost, their owners receive location reports through their iCloud account and view them, e.g., using the FindMy application [15].

Heinrich *et al.* [1] reverse-engineered this system, allowing a series of Bluetooth Low Energy (BLE) devices to appear as tags inside Apple's FindMy network. Any of such tags will beacon at all times, regardless of the presence of its owner. Furthermore, when marked as *lost* the owner of a tag may then receive any available location report. To get started, an iCloud user, the *owner* of a tag, creates a public-private key pair (e_k, d_k) through a series of API calls, for tracking a device.

1) Beaconing: To enable tracking, a device broadcasts a BLE advertising packet using a specific MAC address that is derived from the above public key e_k. *Finder* devices listen for Apple FindMy beacons and check for each received beacon if the advertised payload and MAC address are a valid "match". A tag derives both payload and MAC address from the public key (e_k) created by its owner (see [1] for details

on their algorithm), so that only valid packets are processed further. *Finder* devices receiving such beacons can then furnish location reports for nearby tags, anonymously to iCloud.

2) Reporting a tag: The reporting process is depicted in Figure 1. A tag broadcasts an appropriate BLE beacon, as described above. A finder device passing by will identify this as a lost device and store a location report, containing (1) the beacon reception time, termed contact (t_c), (2) the confidence about that contact (similar to accuracy in (4)), (3) the public key e_k, (4) the location information, encrypted using e_k and containing geographical coordinates, horizontal accuracy, and status, and (5) an authentication label AES-GCM to validate the report. These reports are uploaded securely (HTTPS POST) after some time to Apple's iCloud, where they are stored until being requested by the tag owner. In addition to these data fields, a *bundle* of reports submitted by a single finder is annotated with the timestamp of when the entire batch was received on the server side (t_r).

3) Reading reports: With the public keys (e_k) of their own tags, users query their iCloud account for available location reports with HTTPS GET requests; each user can then decrypt the location information contained in a report using the corresponding private key (d_k).

B. BLE Advertising and Our Experimental Setup

The BLE standard allows advertising packets of devices to include up to 31 bytes of information. These beacons are broadcast at intervals between 20 ms and 10 s on any of three channels used for advertisements [16]. Their successful reception by a nearby finder device is stochastic: the transmission (TX) power for the advertising packets may influence proper reception and may the distance between finder and tag and the environmental conditions (e.g., radio interference). Also, both finder and tag continuously switch channels and would need to use the same one when a packet is sent.

Our Setup: Given these constraints, and to better understand the conditions in which locations of devices are reported, we carry out a series of experiments to build a thorough understanding of how sensing and reporting work in the Apple FindMy network. We use a series of ESP32 microcontrollers as tags for our experiments. These low-power devices provide programmable BLE support through an API which allows full control of its Bluetooth controller, including TX power and advertising interval, which are often not accessible on other platforms. These experiments allow us to draw observations which set the foundation to the side-channel communication we discuss on the following sections.

IV. FindMy System Characterization

In this section, we present the results of a series of experiments we conducted in controlled environments as well as in the wild. We first present our findings on the behavior of Apple devices when reporting smart tags to the FindMy network. Next, based on observations drawn from our aforementioned findings, we present results from crowd monitoring measurements compared to state of the art alternatives, as well as a possible side-channel attack.

A. Uploading of reports is determined by device settings

As discussed in Section III, finder devices often bundle a series of reports before uploading them to iCloud. To better characterize this behavior, we analyzed the traffic between iCloud and two jailbroken[1] iPhones (7 and 8, on iOS14.6) using an HTTP proxy. With a Bluetooth tracker, continuously beaconing, placed next to these phones for intervals of 72 hours, we tested how different settings influenced the uploading intervals. We present the distribution of these intervals in Figure 2 for various settings, with a clear distinction between being on Wi-Fi (median ∼15 minutes) or Cellular (median ∼3 hours), on power supply or battery. Additionally, with Low Data Mode enabled, the phones uploaded reports less often (median ∼36 minutes), whereas other modes did not affect reporting significantly. Therefore, the phone settings explain differences between the contact time t_c reported and received time t_r, *when a bundle of reports is sent*.

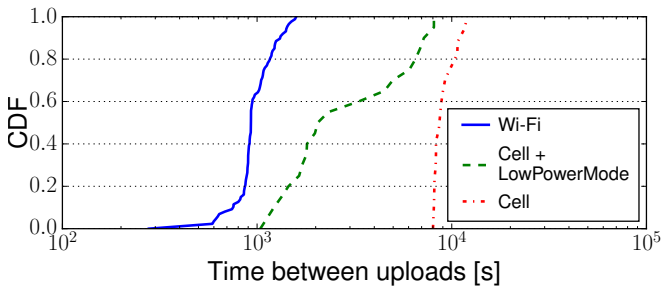


Fig. 2: Delay in sensing and reporting a SmartTag.

B. Over 50% of reports are uploaded within 15 minutes

We ran a set of measurements in the wild, at a large public space. These observations lasted for a total 24 hours, and were conducted on various days from July to September 2021. From these data, we computed the delay between sensing a tag (t_c) and uploading the reports to iCloud (t_r), for which the distribution is depicted in Figure 3. This delay shows a strong mode around 15 minutes, a median of 13.15 minutes, and has 95% of its values between 6 seconds and 8 hours (shaded area).

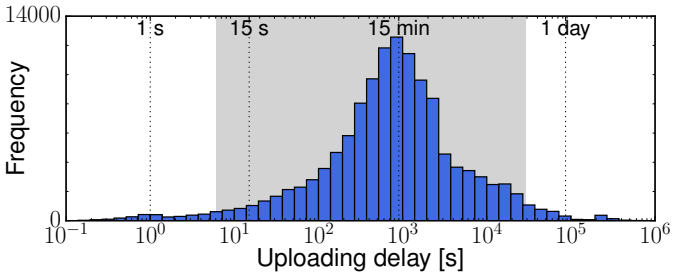


Fig. 3: Distribution of the delay in sensing and uploading.

[1]Required as iCloud HTTPS communication requires certificate pinning.

C. Uploading time uniquely identifies a finder

As reports are bundled, their receiving time t_r is appended on the server side with the precision of milliseconds. This, in turn, allows us to *uniquely* identify a finder for a series of reports as those will contain the same t_r with several decimal points of precision. As discussed above, the uploading of reports may be done hours apart from their actual contact time. It is important to note that each upload may contain up to 255 reports and up to 4 per tag. That is, if a nearby finder is connected to Wi-Fi, only up to 4 location reports will be uploaded every 15 minutes.

D. Short advertising intervals lead to no reports

During our measurements in the wild using short BLE advertising intervals (*e.g.*, 20 ms), we observed that the FindMy network discards *all* location reports for a tag, possibly due to uploads happening too frequently. Although we were not able to precisely determine the best limit, the fastest we could advertise without periods of missing data was 1022.5 ms. For that, in all measurements discussed in Section VI we carried tags configured at that BLE advertising interval (as suggested by Apple [17]) and at maximum TX power (+9 dBm).

Takeaway (§IV): FindMy provides limited but valuable information on nearby finders and that depends on the settings of their mobile devices. Moreover, the reports upload may be delayed from 15 minutes to several hours.

V. General Overview

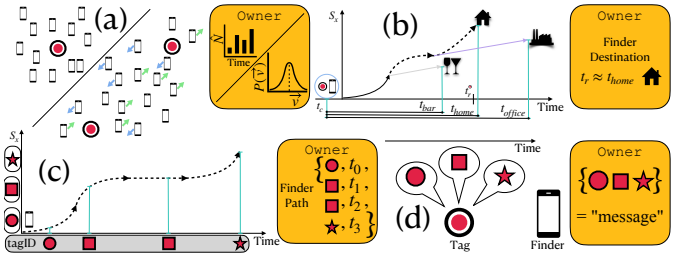


Fig. 4: Overview of alternative uses for the Apple FindMy service. (a) Crowd monitoring. (b) Remote destination inference. (c) Path reconstruction. (d) Covert communication.

Given the observations drawn from the characterization of FindMy (§ IV), we now look at how the spatial and temporal availability of tags can be further exploited to create alternative uses. Using Figure 4 as a guide: (a) In an area with multiple finders, using a *single* tag in a fixed location we can estimate how crowded a monitored area is (§ VI-A), while using *multiple* tags in fixed locations we can study properties of their flow (§ VI-B). (b) Targeting a single finder, using a *single* tag, placed for a short period in proximity with a target, and have this finder move to a commonly visited place, we (or an attacker) can estimate where this target finder could have gone from a list of possible destinations (§ VII-A). (c) Again, targeting a single finder, but this time using *multiple* tags, each placed along any arbitrary area (or paths), we (or an attacker) can estimate the trajectory taken by this finder (§ VII-B). (d) Using multiple tags (or simply emulating multiple tagIDs with a single transmitter) and any arbitrary number of finders, we can encode a message into the sequence these tagIDs are transmitted. As we will discuss, in (b), (c) and (d) the respective finder(s) are unaware of the respective use. Currently, only disabling Bluetooth or opting out of the FindMy service can mitigate this issue.

VI. Crowd Monitoring

We now evaluate how well smart tags on Apple's FindMy network can be used for *crowd monitoring*. We first present our results for crowd size estimates, evaluated against a state-of-the-art image recognition approach. Next, we present results for crowd flow which we evaluate against passive measurements of commonly used Wi-Fi management frames [2].

A. Crowd Size – Using a single tag

For this evaluation, we conducted 8 measurements, of 3 hours each, from July to September 2021 in a large public square in the city of Munich, Germany. During these months, this main square is often crowded due to shops, restaurants and metro stations nearby. Our smart tag setup consisted of an ESP32, advertising at ∼1 second intervals and using high TX power (+9 dBm) for maximum discoverability. We evaluate these measurements against the people count obtained by image recognition, which we describe next.

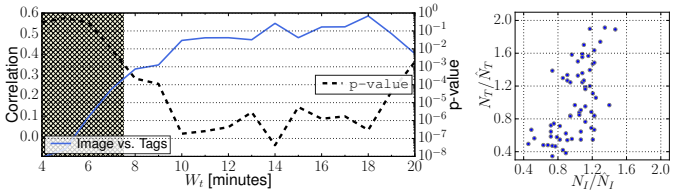
1) Image recognition: The use of images for crowd estimates produces some of the most accurate results with the use of inexpensive hardware [2]. Modern approaches based on Convolutional Neural Networks have quickly become the state of the art for all image recognition tasks, and in spite of their capabilities, such methods are not extensively used due to privacy concerns raised by their usage. Those concerns include regulatory legislation in several countries. For our evaluation, we used images from a publicly available web-cam[2], openly streaming images at 5 seconds per frame, with a 2048x1536 resolution. For that, we use the `Mask R-CNN` [18], from which we extract the total count of persons per frame. `Mask R-CNN` performs multi-class object instance segmentation, detecting and dividing each class instance in the prediction. This segmentation is of key importance for the detection of people in a large environment since they tend to stay in groups. `Mask R-CNN` is able to detect different people that overlap each other, increasing the accuracy of the model [18].

2) Size Measurements Description: For crowd size analysis, we correlate the total number of persons identified using image recognition against our smart tags approach. From the latter, we identify a unique device using the time a set of reports was uploaded to iCloud (see § IV).

3) Results: To best estimate the time window to aggregate tag reports, we correlated the number of identified finders over different time window (W_t, or bin sizes). Figure 5a depicts how the Pearson correlation value changed with bin

[2]https://www.ludwigbeck.de/webcam (will be **removed** in camera-ready)

sizes, while it also shows the `p-value` for those sizes. The `p-value` estimates the probability the estimated correlation coefficient was due to randomness, and we adopt `p-value` < 0.001 as our confidence interval, below which results are deemed acceptable. From the analysis, we note that only from W_t of 8 minutes we obtained `p-value` below the confidence interval, and the correlation coefficient reaches its maximum value at 18 minutes, with a value of 0.58 which corresponds to a strong correlation between both values. The highest values around 15 minutes could be explained by the expected time an iPhone takes to upload location reports (see § IV). Figure 5b depicts the best relationship between the normalized values for tags (N_T) and from images (N_I), at W_t 18 minutes. Thus, coarse-grained crowd size monitoring with a modest time lag appears feasible.



(a) Correlation and bin sizes (W_t) (b) $W_t = 18$ mins

Fig. 5: Relationship between numbers from tags (N_T) and from images (N_I), for different bin sizes (W_t).

B. Crowd Flow – Using multiple tags

We now evaluate how smart tags could be used to study the flow of a crowd. That is, we explore how well we can measure moving time and waiting time (or dwell time) using a pair of smart tags in a large urban environment. We compare our method with measurements done using Wi-Fi management frames, as those are currently widely adopted by researchers and commercial applications [2].

1) Wi-Fi Management Frames: Mainly due to its simplicity and reported accuracy [2], [19], crowd monitoring using Wi-Fi management frames is extensively used. In principle, all Wi-Fi enabled devices send a series of management frames, often used to search for available access points and to establish/maintain existing connections. These frames contain a device identifier (MAC), which can, in turn, be tracked through space and time while uniquely identifying a mobile device. To mitigate this traceability, since 2014, mobile devices perform MAC randomization at ever increasing rates and new schemes [20]. For our purposes, however, discarding locally managed addresses and subsampling our measurements with only global addresses suffices for our first order approximations of time between vantage points. From our measurements, on average, 26% of management frames captured were from non-random MAC addresses. We measured this using a Raspberry Pi 3, with two external antennas, hopping between the non-overlapping channels 1, 6 and 11.

2) Flow Measurements Description: For crowd flow analysis, we compare the distribution of time intervals a set of devices takes to be observed between two vantage points. These points were 176 meters apart and were chosen at the city center of Munich, Germany, in a commercial area where only pedestrians are allowed. We conducted 3 measurements of 2 hours each on 6/7/8 September 2021. From the measurements of the smart tags, we identify a unique device using the time a set of reports was uploaded to iCloud (t_r, see § IV). If such a bundle of reports contained at least one record at each vantage point, we could then infer the time interval the device took between both locations. Similarly from Wi-Fi frames, this interval corresponds to the time between consecutive records at each observed location.

3) Results: The left panel on Figure 6 shows the histogram of measured times between vantage points, with a mode at 2 minutes from both sources. Furthermore, to meaningfully analyze our measurements, we decompose them into log-normal distributions, using the widely used Gaussian-Mixture Model. This unsupervised learning method decomposes an input set into a pre-determined number of Gaussians. To select the best number of clusters, we used the BIC method [21] which estimates how well a given model explains the variance in the measured value. Our empirical results suggest that the ideal number of clusters for the Wi-Fi and smart tags measurements is 3. Following this classification, the right panel on Figure 6 depicts the similarities in the distributions of the estimated walking and waiting times from tags and Wi-Fi. The average walking time was 2.41±0.04 minutes using tags and 2.28±0.04 minutes using Wi-Fi. That yields an equivalent ∼4.5 km/h walking speed, in line with existing urban pedestrian research [22]. The average estimated waiting time (assuming the mean walking time above) was 19.33±1.44 minutes using tags and 20.54±0.69 using Wi-Fi. A possible interpretation of these values is the expected time pedestrians have spent at shops along the way.

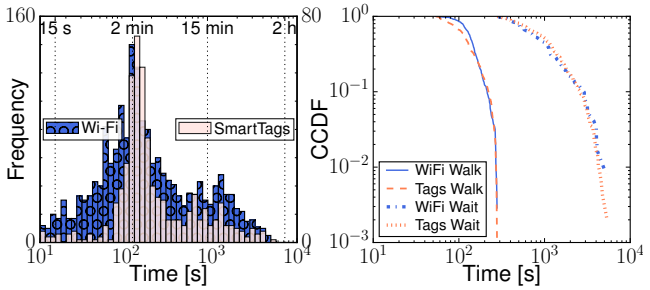


Fig. 6: Crod Flow [Left] Time between vantage points. [Right] Estimated walking and waiting times between vantage points.

Takeaway (§ VI): Bluetooth trackers can be used for crowd monitoring, with comparable results to widely used alternative solutions. These alternatives, however, may disclose personal information and always generate data that needs to be handled with care, such as personal identities. For crowd sizes, estimates with *a single tag* strongly correlate with estimates using state of the art image recognition. Furthermore, different aspects of crowd flow were accurately estimated using multiple

tags when compared to Wi-Fi measurements. In both use-cases, our approach always guarantees the anonymity of the studied subjects given the reporting mechanism and end-to-end encryption of this Apple service.

VII. Deliberate Tracking

In this section, we present proof of concept (PoC) evaluations and possible mitigation strategies to information being leaked by the Apple FindMy service. We demonstrate how this leakage may allow an attacker to track a victim's device through *timing attacks*, enabled by the reporting system implemented by Apple. The experiments we present used only our own devices to avoid disclosing unwanted information from other subjects. Apart from privacy concerns, it was necessary to have control of the phone's settings and times when connected to Wi-Fi or cellular network.

Overview: We demonstrate two examples of information leakage: A) Destination inference, in which the timing between sensing a tag and uploading a report may disclose where a victim could have gone; B) Path reconstruction, in which a sequence of visited places can be precisely inferred. Both examples rely on the bundling of reports (see § III) as well as the difference in uploading delay when connected to different networks (see § IV). *Note that the threats we present only require the victim being near a tag for a brief period of time, and not being tracked by inadvertently carrying a tag.*

A. Remote Destination Inference – Using a single tag

This attack relies on the timing of location reports, but precisely the difference between sensing a tag (t_c) and uploading a report (t_r). Furthermore, the modulation of the TX power can limit the range of BLE beacons, helping ensure only a victim's phone is affected.

1) Threat Model: An attacker, who wants to know where a victim has gone after an encounter, performs a timing attack using one tag. The attacker knows the victim's most visited locations, and the victim is only connected to cellular while outdoors and connects immediately to Wi-Fi when reaching her destination. During the encounter, the attacker "tags" a victim's phone by transmitting a series of beacons. The victim's phone, while on cellular, will store the reports until reaching her destination where, on Wi-Fi, it will upload all location reports for the attacker's tag. With the difference between sensing and uploading the reports, an attacker can limit (or pinpoint) the most likely destination of the victim. The victim is unaware this attack is tracking place, and only disabling Bluetooth or the FindMy service can mitigate it.

2) PoC Description: For this, we used an iPhone 12 (iOS 15) and one tag, configured at -6 dBm ensuring only at close proximity our phone would sense our tag. We enabled our tag next to our iPhone for 1 minute, then moved 18.5 km (11.5 miles) to a destination, where we finally enabled Wi-Fi.

3) Results: Our moving time was ∼29 minutes, and the difference between t_c and t_r was ∼35 minutes. Furthermore, we observed similar behavior on sensing and immediately uploading reports once on Wi-Fi, as previously discussed.

B. Path reconstruction – Using multiple tags

Given the bundling of reports (see § III), intentionally positioned *lost* tags can form a sequence of "breadcrumbs" which can then disclose the path followed by a phone. Similarly to the previous example, TX power can be modulated to ensure shorter coverage from each tag. As a proof of concept, we conduct one experiment.

1) Threat Model: An attacker, willing to find out the whereabouts of a victim through an area of interest, places tags at known locations. This attack relies on the victim having her phone connected to the cellular network only while moving and eventually connecting to Wi-Fi after the monitored journey. The victim's phone will then sense these tags, keeping the order of the observed tags. Once uploaded, the location reports disclose where and when the victim had been. The unique t_r, appended by iCloud when receiving a bundle of reports, uniquely identifies a finder device (see § III). The victim is unaware this attack is taking place, and only disabling Bluetooth or the FindMy service can mitigate it.

2) PoC Description: For this example, we used 3 iPhones (7 and 8 on iOS 14.8 and 12 on iOS 15), and placed 3 pairs of tags in a straight line, with each pair at 150 meters away from the next pair, configured at -6 dBm to ensure that only at close proximity devices would sense our tags. We stayed for 5 minutes, at a distance of 2 meters from each pair of tags, then walked to the next location (in ∼2 minutes). We disabled Wi-Fi until reaching a planned location away from the tags to ensure it did not unexpectedly upload any reports.

3) Results: From all three phones, we were able to reconstruct the path and timing taken at each location. Figure 7 depicts the transitions between each state (static or mobile) as well as the corresponding t_c contained in each location report. We also noted that, on all phones, the upload of reports happens within the first 5 minutes of switching to Wi-Fi from cellular-only. With such information, an attacker can reconstruct the set of visits of a victim and obtain an accurate estimation of the time spent at each place. However, if a finder device uploads a bundle of reports before all visits are done, then the attacker will not be able to fully reconstruct a trajectory.

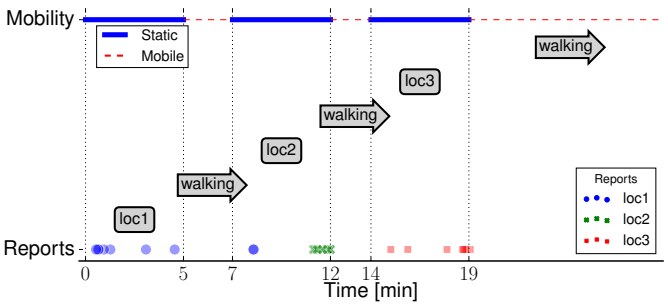


Fig. 7: Path reconstruction.

C. Mitigation

We now discuss mitigation strategies applicable to both path reconstruction and destination inference cases. Until Apple addresses these issues, users can only disable Bluetooth or the FindMy services to prevent their location information unknowingly being leaked, impairing the functionally of the service. From the system's perspective, providers like Apple could (1) reduce the granularity of the *received* timestamps (t_r) or remove them altogether from the location reports, (2) randomize when reports are uploaded, no longer determined by the connectivity available, or (3) initiate the uploading of reports after moving a random distance. None of these systems solutions should impact the functionality of the service, while still protecting the privacy of its users.

VIII. TagComm – Covert Channel Using BLE Trackers

Now, we turn to an illustration of a side-channel communication, built on the FindMy service, which we name *TagComm*. In this section, we delve into the design a simple unicast protocol, which essentially uses the sensing of tags by nearby iPhones to encode and transport information. Figure 8 outlines our proposed design: we create an artificial tag that changes its beaconed tag ID over time (chosen from a pre-defined alphabet, encoding messages as sequences of the transmitted tag IDs, without the awareness of any nearby finder. Our proposed system could, for example, be used by an attacker trying to exfiltrate data from an air gapped system (cf. [8]). Note that, the transmission is end-to-end encrypted as neither the finder nor Apple are able to decode that any specific tag ID is being transmitted. The decoding of the IDs transmitted, and therefore the final message, is only possible by the owner as discussed in Section III.

Overview: Our protocol uses a set of tag IDs and their permutations to encode information. That is, for N available tags we can encode $\lfloor \log_2(N!) \rfloor$ bits of information. Additionally, we include a set of header bits as well as a parity bit to be encoded along with the message payload. These extra bits and a pre-defined number of tags guarantee a transmitted message can *eventually* be recovered, as will be presented next.

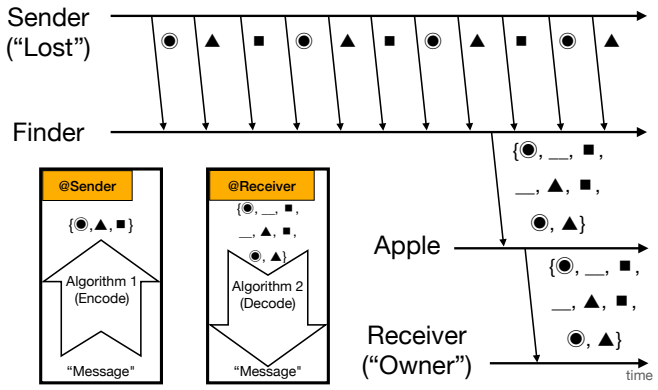


Fig. 8: *TagComm* protocol example, encoding a message as a sequence of tag IDs, silently and securely transmitted by a *finder*.

A. Encoding

In order to maximize the amount of information being sent and provide basic integrity guarantees, we use the permutations of N tag IDs to encode the information we want to transmit. Furthermore, the understanding of the sensing and reporting behavior from Section IV establishes bounds to how fast information can be transmitted.

Algorithm 1: Encoding input word into sequences

```
Input: S,w; /* Symbols and word encoded */
Output: E;  /* Encoded sequence */
L ← length(S) ;  /* Length of S */
assert(L! >= w) ;  /* From ❶ */
E ← [ceil(w/(L − 1)!)] ;  /* One item list */
for idx ∈ range(L − 1, 0, −1) do  /* From ❷ */
    w −= (E[−1] − 1) ∗ idx! ;  /*E[−1]: last */
    e = ceil(w/(idx − 1)!);
    E.append(e);
```

1) Input to sequence of symbols: Given an input message to be transmitted W and a set of encoding symbols of size N, we iteratively divide the interval of N! to find the corresponding sequence to be used. Note that this requires W < N! (as ❶). For example, for N=5 and W=42, we define an order for the resulting set of symbols, *i.e.*, a<b<c<d<e (❷). Next, we split the interval 5! into 5 equally sized blocks, as depicted in Figure 10a. As 42 is found within the second block, the symbol *b* is removed and set as the first symbol. These steps are repeated until all symbols have been removed, yielding the final sequence *bdeca*. This procedure allows us to encode $\log_2(N!)$ bits of information using N tags. Algorithm 1 systematically describes these steps.

2) Defining N=16: Given ❶, we define the code efficiency as the ratio $\lfloor \log_2 N! \rfloor / \lceil N \log_2 N \rceil$ (as ❷). That is, the maximum number of bits encoded by the minimum number of bits required for all used symbols N. Given these observations, Figure 9 shows the variation in ❷ for different values of N up to 20 tags[3], with its highest efficiency at N=16, which we use in our experiments. This leaves us with a total of 44 bits, and their use will be further described next.

3) Frame and supporting bits: To ensure the integrity of the information being transmitted and to allow the receiver to decode that information, we define a simple frame to our protocol, illustrate in Figure 10b. Three *header* bits (as MSB) distinguish different message types: STX (0b000) the start of transmission, F0 and F1 alternating even and odd frames (0b010 and 0b100, respectively), and EOT end of transmission (0b110). These bits ensure the receiver can deterministically identify the start of a transmission, new frames as well as the end of a transmission. This guarantee

[3]Largest number of permutations that fits in 64 bits.

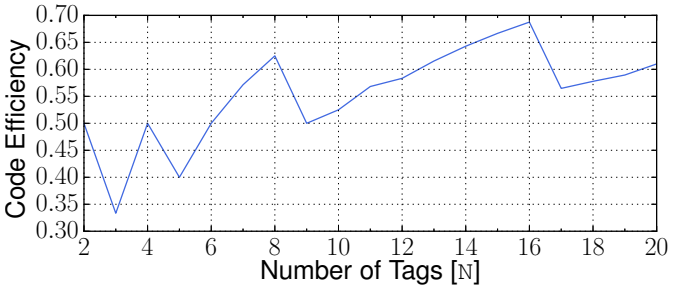


Fig. 9: Code efficiency given the number of symbols (tag IDs) being used to endcode a message, with its maximum at 16.

is achieved by ensuring the initial symbol of a sequence will be unique for each frame type, as a consequence of the values chosen for the header bits. Additionally, a parity bit (as LSB) provides a minimal "checksum" to the message being transmitted once its decoded. Finally, the remaining bits are used for the message payload, which in our setup, consists of 40 bits.

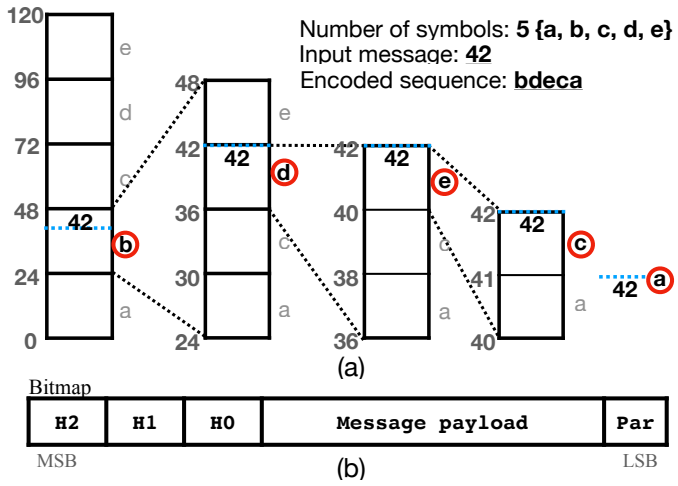


Fig. 10: Protocol definitions. (a) Encoding a value as a sequence of symbols. (b) Frame bitmap.

4) Transmission integrity: To transmit a block of information `W` using Tagcomm, a node will chunk it into words `w` of 40 bits. A transmission will start with an `STX` frame (*i.e.*, header bits set to `0b000`) that will carry the total number of words to be expected in its payload. Next, each word is encoded as a sequence (as described above) as alternating frames of type `F0` and `F1`. Finally, a transmission is terminated with an `EOT` frame. Note that, as the protocol requires all symbols to be transmitted, if a receiver is unable to reconstruct an entire sequence, the corresponding word `w` cannot be retrieved. For details on error recovery, see Section VIII-B.

B. Decoding

Once received by the owner device, location reports for a series of tag IDs should be decoded into its original message. Essentially, this is done by inverting the steps done while encoding a frame into a sequence. Once the different frames are decoded and parity bits verified, the original message can finally be reconstructed.

1) Tags alignment: As the duration of each tag being transmitted is predefined (*e.g.*, t_{tag}), the first step in decoding a message is aligning each received tag in *slots* of size t_{tag}.

2) Frames alignment: As discussed in Section VIII-A, each frame type starts distinct symbols, represented and transmitted in TagComm as tags. Similar to the tags alignment, the duration of how long a frame is sent is also predefined, for example 5 minutes. This way, knowing the expected sequence of frames, *i.e.*, `STX`, `F0`, `F1`, ..., `EOT` (as ❸), and their corresponding starting symbols, we can align all received frames and start the decoding step.

3) Decoding frames: Once the sequences that encode each frame are identified, we can decode the information by reversing the steps explained in Section VIII-A. That is, assuming a predefined order between tags (*e.g.*, ❷), and taking the encoded sequence as input, we can recover the initial message by adding up the partial contributions each symbol had in splitting the `N!` interval, as described in Algorithm 2. After decoding, we then verify the presence of errors in the next step.

Algorithm 2: Decoding sequences into words

```
   Input: S, E; /* Symbols and encoded seq. */
   Output: w;                  /* Decoded word */
1  L ← length(S) ;             /* Length of S */
2  P ← [ ] ;                   /* Empty list */
3  for i, e ∈ enumerate(E) do
4  |  idx ← S.index(c) ;      /* Symbol e index */
5  |  P.append((L − 1 − i) * idx); /*Partial sum*/
6  |  S.pop(idx)
7  w ← sum(P) ;               /* Add up all partials */
```

4) Error Correction: Once each frame has been decoded from the input sequences, we can validate the integrity of the frame with its parity bit. Furthermore, the expected sequence of frame types (*i.e.*, ❸), combined with a parity bit, allows us to recover messages when a single tag (out of a sequence) is not received. The position of the missing tag can be determined when aligning the tags in each frame (see above), and finally verified with the corresponding parity bit, allowing us to reconstruct the orginal message in the next step.

5) Final message reconstruction: Once all frames have been decoded, the total number of expected frames sent as the payload of `STX` frames can be read and verified. Finally, all the information encoded in a series of sequences of tags can be reconstructed.

C. TagComm Experiment

In this section, we describe the set of experiments we conducted to test our TagComm system. Furthermore, we present a series of observations made from the results obtained.

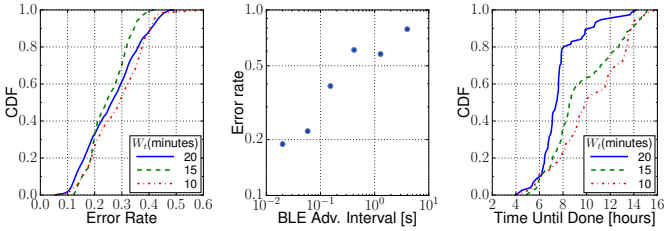
D. Setup

For all measurements, we transmitted a set of 10 randomly generated words of 40 bits. These were transmitted using the protocol described in Section VIII. As previously discussed, we used a single ESP32 as our lost tag, which implemented the transmitter/encoder logic (see § VIII-A. As a *finder*, we had an iPhone 12 (iOS 15) nearby, connected to Wi-Fi at all times. As discussed in Section III, this setting allows us to estimate a best-case scenario given the expected frequency the iPhone would publish location reports for our tag (*i.e.*, within 15 minutes more than 50% of the time).

E. Results

We were able to successfully transmit a set of random words, as described above. To better verify the limits of our system, we varied some of the parameters, such as word duration and BLE advertisement interval.

Definitions: For this analysis, we define the error rate as the fraction of tag slots during which no reports were received. That is, given the tag slot duration (*e.g.*, 30 s), the error rate of a transmission corresponds to tag slot intervals during which a finder was present but no report was sent. Further, we define the time until done (TUD) as the expected minimum time required to decode a complete message, from starting the transmission until it is fully decoded by the owner's device.



(a) (b) (c)

Fig. 11: Error rate and TUD for different settings. (a) CDF of error rate and different frame duration (W_t). (b) Error rate and BLE advertisement intervals. (c) CDF of TUD and W_t.

Frame duration and error rate: The different frame durations we tested produced small differences between error rates. We observed 26.17% for 20 minutes, 24.73% for 15 minutes, and 28.82% for 10 minutes on average, and their distributions are depicted in Figure 11a. This indicates that repeating a tag at a certain position for longer periods of time does not affect the probability it will be detected by a Finder.

Larger BLE advertisement intervals increase error rate: We compare the effect different Bluetooth LE advertising intervals had in the error rate. From our measurements, we note that for larger advertising intervals, less tags were observed per unit of time. Figure 11b depicts the monotonic increase in error rate with increased adv. intervals. This could be explained by the probabilistic nature of these intervals, and to which transmitters do not have any control [16].

Frame duration and time until done: We tested frame window sizes around 15 minutes (10, 15, 20), as previous experiments showed that to be the expected time over 50% of reports take to be published. We measured each configuration for 72 hours, and computed expected values for TUD from 100 different random starting points in each setting. For these measurements, we observed 7.73±0.22 hours with 20 minutes, 9.57±0.30 hours with 15 minutes, and 10.36±0.36 hours with 10 minutes. The distributions of TUD for each configuration is depicted on Figure 11c. Interestingly, using two iPhones on the same iCloud account and placed near a tag, did not produce statistically significant improvements.

F. Mitigation

Currently, users can avoid inadvertently transporting information with a similar system by disabling Bluetooth on their phones or opting out of the FindMy services. From the system side, while keeping the main functionality of the FindMy services, Apple can limit the number of updates issues by a finder, as well as limit the number of available reports per tag or decrease the accuracy of the time stamps used. Notably, Apple currently does not notify users about one of these crafted tags being around, as we did not get a single notification during our experiments, using multiple iCloud accounts.

IX. Discussion

Privacy: To preserve privacy of the individuals part of our crowd sensing experiments, we (1) discard all original MAC addresses from the Wi-Fi measurements, leaving records that can no longer identify the owners of the original devices, (2) we compute the metrics from each image and store only the counts per frame, using publicly available images, and (3) used our own equipment to demonstrate the possible information leakage from the FindMy services. For the communication experiments, by using our own devices and iCloud accounts, we ensure the privacy and resources of other individuals were not affected by our work. However, our work unveils possible attack vectors which could be exploited, compromising the security and privacy of the subjects involved.

Ethics: Our measurements and analysis were designed and executed to minimize exposing information about subjects being studied. Whenever possible, we limited our study to our own devices, and when studying crowds we discarded all identifiable information. However, we understand the methods presented could be used in other unintended ways. Therefore, we believe such study may contribute to the design of future versions of Bluetooth tracking systems.

Crowd Sensing: Our analyses show acceptable results using *a single tag* on the Apple FindMy service to sense aspects of a crowd. More importantly, our system provides privacy guarantees when used with a large group of subjects. Unlike in the deliberate tracking examples, a large group of unknown individuals ensures no single subject can be identified or have further information disclosed.

Extended support to other platforms: To enable further studies with such smart tags system, we extend the support originally implemented by Heinrich *et al.* [1] to `macOS` (*e.g.*, developing of new applications and debugging) and the Amazon Echo [3], [4] (*e.g.*, home IoT turn into sensing device) to be used as a tag.

More Finders may increase Tagcomm delivery guarantees: During our Tagcomm experiments, using a single extra Finder did not yield significant improvements in the reliability of sending messages. For applications deployed in spaces where multiple finders could be passing by may increase further the guarantees messages are transmitted.

X. Conclusion

In this paper, we present how Bluetooth trackers (or tags) can be used beyond their originally designed purpose, of tracking lost devices. We show how crafted tags can be used as crowd sensing devices, with relative estimates of large groups of people. These estimates include crowd size estimated with a *single* tag, and also crowd flow by using multiple tags along a monitored path. Furthermore, we demonstrate through a series of controlled experiments how the Apple FindMy service currently discloses sensitive location information from passive finder devices. An attacker may, in turn, reconstruct a victim's path and visits as well as a possible final destination, currently exposing billions of Apple devices [1].

In addition, we also present Tagcomm, a proof-of-concept out-of-band communication channel using Apple tags. Using a simple protocol, we demonstrate how various tag IDs can be used to encode any arbitrary information and transmitted over a secure end-to-end encrypted channel, without the knowledge of the phones that handle part of this communication path. Our intent is to raise awareness of such possibility, while discussing possible uses which include side-channel communication that could leak sensitive information from a compromised system.

Future iterations of our work will consider other Bluetooth trackers for crowd sensing, and leverage TagComm to estimate users' behavior. Furthermore, similar systems providing raw reports (*i.e.*, not aggregates over time) will be verified for the vulnerabilities presented here.

Reproducibility: To foster further research, we make our code and sample data openly available [4] along with an extended support for other platforms to be used for either communication or sensing [3].

References

[1] A. Heinrich, M. Stute, T. Kornhuber, and M. Hollick, "Who can find my devices? security and privacy of apple's crowd-sourced bluetooth location tracking system," *Proc. Priv. Enhancing Technol.*, vol. 2021, no. 3, pp. 227–245, 2021. [Online]. Available: https://doi.org/10.2478/popets-2021-0045

[2] A. Draghici and M. van Steen, "A survey of techniques for automatically sensing the behavior of a crowd," *ACM Comput. Surv.*, vol. 51, no. 1, pp. 21:1–21:40, 2018. [Online]. Available: https://doi.org/10.1145/3129343

[3] L. Tonetto *et al.*, "Tags Crowd Sensing," https://home.in.tum.de/~tonetto/smarttag_artifacts.tar.bz2, 2022, [Online; accessed Jan-2022].

[4] L. Tonetto, "Tagcomm," https://home.in.tum.de/~tonetto/tagcomm_artifacts.tar.bz2, 2022, [Online; accessed Jan-2022].

[5] J. Martin *et al.*, "Handoff all your privacy–a review of apple's bluetooth low energy continuity protocol," *Proceedings on Privacy Enhancing Technologies*, vol. 4, pp. 34–53, 2019.

[6] J. Martin, D. Alpuche, K. Bodeman, L. Brown, E. Fenske, L. Foppe, T. Mayberry, E. C. Rye, B. Sipes, and S. Teplov, "Handoff all your privacy - A review of apple's bluetooth low energy continuity protocol," *Proc. Priv. Enhancing Technol.*, vol. 2019, no. 4, pp. 34–53, 2019. [Online]. Available: https://doi.org/10.2478/popets-2019-0057

[7] M. Weller, J. Classen, F. Ullrich, D. Waßmann, and E. Tews, "Lost and found: stopping bluetooth finders from leaking private information," in *WiSec '20: 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2020, pp. 184–194. [Online]. Available: https://doi.org/10.1145/3395351.3399422

[8] A. Dorais-Joncas *et al.*, "Jumping the air gap: 15 years of nationstate effort," https://www.welivesecurity.com/2021/12/01/jumping-air-gap-15-years-nation-state-effort/, 2022.

[9] M. Guri, B. Zadov, and Y. Elovici, "Led-it-go: Leaking (A lot of) data from air-gapped computers via the (small) hard drive LED," in *Detection of Intrusions and Malware, and Vulnerability Assessment - 14th International Conference, DIMVA 2017, Bonn, Germany, July 6-7, 2017, Proceedings*, ser. Lecture Notes in Computer Science, M. Polychronakis and M. Meier, Eds., vol. 10327. Springer, 2017, pp. 161–184. [Online]. Available: https://doi.org/10.1007/978-3-319-60876-1_8

[10] M. Eichelberger, S. Tanner, G. Voirol, and R. Wattenhofer, "Receiving data hidden in music," in *Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications, HotMobile 2019, Santa Cruz, CA, USA, February 27-28, 2019*, A. Wolman and L. Zhong, Eds. ACM, 2019, pp. 33–38. [Online]. Available: https://doi.org/10.1145/3301293.3302360

[11] M. Guri, Y. A. Solewicz, A. Daidakulov, and Y. Elovici, "Acoustic data exfiltration from speakerless air-gapped computers via covert hard-drive noise ('diskfiltration')," in *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II*, ser. Lecture Notes in Computer Science, S. N. Foley, D. Gollmann, and E. Snekkenes, Eds., vol. 10493. Springer, 2017, pp. 98–115. [Online]. Available: https://doi.org/10.1007/978-3-319-66399-9_6

[12] A. M. Al-Shaery, S. S. Alshehri, N. S. Farooqi, and M. O. Khozium, "In-depth survey to detect, monitor and manage crowd," *IEEE Access*, vol. 8, pp. 209 008–209 019, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3038334

[13] C. A. Pouw, F. Toschi, F. van Schadewijk, and A. Corbetta, "Monitoring physical distancing for crowd management: Real-time trajectory and group analysis," *PloS one*, vol. 15, no. 10, p. e0240963, 2020.

[14] S. Chang, E. Pierson, P. W. Koh, J. Gerardin, B. Redbird, D. Grusky, and J. Leskovec, "Mobility network models of covid-19 explain inequities and inform reopening," *Nature*, vol. 589, no. 7840, pp. 82–87, 2021.

[15] Apple, "FindMy App," https://www.apple.com/icloud/find-my/, 2021, [Online; accessed September-2021].

[16] Á. Hernández-Solana, D. P. D. Cerio, A. Valdovinos, and J. L. Valenzuela, "Proposal and evaluation of BLE discovery process based on new features of bluetooth 5.0," *Sensors*, vol. 17, no. 9, p. 1988, 2017. [Online]. Available: https://doi.org/10.3390/s17091988

[17] Apple, "Accessory Design Guidelines - 37.5 Advertising Interval," https://developer.apple.com/accessories/Accessory-Design-Guidelines.pdf, 2021, [Online; accessed September-2021].

[18] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2980–2988. [Online]. Available: https://doi.org/10.1109/ICCV.2017.322

[19] L. Tonetto, M. Untersperger, and J. Ott, "Towards exploiting wi-fi signals from low density infrastructure for crowd estimation," in *Proceedings of the 14th Workshop on Challenged Networks (CHANTS)*. ACM, 2019, pp. 27–32. [Online]. Available: https://doi.org/10.1145/3349625.3355439

[20] E. Fenske, D. Brown, J. Martin, T. Mayberry, P. Ryan, and E. C. Rye, "Three years later: A study of MAC address randomization in mobile devices and when it succeeds," *Proc. Priv. Enhancing Technol.*, vol. 2021, no. 3, pp. 164–181, 2021. [Online]. Available: https://doi.org/10.2478/popets-2021-0042

[21] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics*, pp. 461–464, 1978.

[22] R. L. Knoblauch, M. T. Pietrucha, and M. Nitzburg, "Field studies of pedestrian walking speed and start-up time," *Transportation research record*, vol. 1538, no. 1, pp. 27–38, 1996.