

# Probing for Passwords – Privacy Implications of SSIDs in Probe Requests

Johanna Ansohn McDougall, Christian Burkert, Daniel Demmler, Monina Schwarz, Vincent Hubbe, and Hannes Federrath

University of Hamburg, Germany

{johanna.ansohn.mcdougall, christian.burkert, daniel.demmler,  
monina.schwarz, vincent.hubbe, hannes.federrath}@uni-hamburg.de

**Abstract.** Probe requests help mobile devices discover active Wi-Fi networks. They often contain a multitude of data that can be used to identify and track devices and thereby their users. The past years have been a cat-and-mouse game of improving fingerprinting and introducing countermeasures against fingerprinting.

This paper analyses the content of probe requests sent by mobile devices and operating systems in a field experiment. In it, we discover that users (probably by accident) input a wealth of data into the SSID field and find passwords, e-mail addresses, names and holiday locations. With these findings we underline that probe requests should be considered sensitive data and be well protected. To preserve user privacy, we suggest and evaluate a privacy-friendly hash-based construction of probe requests and improved user controls.

**Keywords:** Probe Requests · Wi-Fi Tracking · Privacy Preserving Technologies

## 1 Introduction

To establish a Wi-Fi connection, mobile devices can transmit so-called *probe requests* to receive information about nearby Wi-Fi networks. An access point observing a probe request is led to reply with a probe response, thereby initiating a connection between both devices. While probe requests are used to establish a connection between a mobile device and an AP, they also serve as a means to track, trilaterate and identify devices for attackers who passively sniff network traffic. They can contain identifying information about the device owner depending on the age of the device and its OS. One of those is the preferred network list (PNL), which contains networks identified by their so called Service Set Identifier (SSIDs). Around 23 % of the probe requests contain SSIDs of networks the devices were connected to in the past, according to our measurements. There exist online mapping services like WiGLE<sup>1</sup>, which provide information about geographical locations where SSIDs have been observed. A casual observation of the networks

<sup>1</sup> <https://www.wigle.net>

available in any given residential area returns a multitude of personalised, often descriptive SSIDs used for private networks. Therefore, a query for an SSID might reveal home or work addresses, or other visited locations where users connected to Wi-Fi, and can thereby reveal very personal information about them.

Another application in which probe requests are frequently used is tracking of devices in stores or cities: as probe requests are sent rather frequently, they can be used to trilaterate the location of a device with an accuracy of up to 1.5 metres [24]. Trilateration can also be used to follow the movements of a device and thereby its user over a longer period of time, and track them through a store or city [23]. This is in fact employed in 23% of the stores already [1]. Companies and cities that conduct Wi-Fi tracking take the legal position that only the MAC address contained in probe request is considered personal data according to GDPR Article 4(1) [8,10], which protects personal data from unlawful collection and processing. They therefore maintain that if the MAC address is anonymised before storage, the collection and evaluation of probe requests is GDPR compliant [29]. The randomisation of MAC addresses mitigates linkability via this element. Instead, we focus on looking at what privacy risks originate from probe requests related to the list of SSIDs stored in the PNL. We provide empirical evidence that probe requests should also be considered personal data on the basis of their SSID field, which we find can even contain directly identifying information. We hope to thereby stress the need for a more thorough legal evaluation. We additionally propose changes to the handling of SSID field and mobile OS behaviour to enhance the privacy of users and decrease their trackability to passive sniffers.

To this end, we contribute the following:

- We conduct a field experiment in a German city, recording probe requests of passersby.
- We evaluate their content, with special regard for SSIDs and identifying information.
- We summarise the state of probe requests for different OS versions.
- We propose a hashing of non-wildcard SSIDs in probe request to protect their confidentiality against passive observers.
- We propose changes to the UI design of Wi-Fi selection and PNL management.

This paper is structured as follows: in the next section, we first provide a background on network discovery and privacy implications of MAC addresses. Here, we also compare the privacy features of various Android and iOS versions. Thereafter, we present related work in Section 3. Section 4 explains the experimental setup and our handling of ethical and privacy concerns. We then present the results of our data analysis in Section 5. Section 6 proposes mitigation approaches on both protocol and user interface level. In Section 7, we discuss the findings. Finally, Section 8 concludes the paper.

## 2 Background

In this section we define the underlying technological background of our work.

## 2.1 Network Discovery in 802.11

To establish a Wi-Fi connection between a mobile device and an access point (AP), both devices have to discover each other; either via active or passive discovery:

In a passive discovery, an AP advertises itself by sending out *beacons* containing its SSID, MAC address, the cipher suites it supports and a few other elements [17]. These beacons are sent at an interval of approximately every 100 ms [12], and mobile devices can respond with Wi-Fi association frames.

In an active discovery, mobile devices broadcast *probe requests* to find APs they have previously associated with. Active discovery is also required to connect to so-called hidden networks, for which the AP does not advertise the network, i.e., does not send out beacons. Probe requests sent by most modern devices are typically broadcast and contain the empty wildcard in the SSID field. APs receiving a probe request respond with a *probe response* directed at the sender of the probe request. The probe response contains the SSID of the AP and additional information like supported rates and various capabilities.

The reason both active and passive discovery mechanisms are used is that while APs advertise themselves constantly, scanning for beacons can be rather energy consuming and slow. Additionally, a mobile device scanning for beacons on one channel with a certain frequency might miss beacons sent on another channel. A device actively probing for APs just has to turn on the Wi-Fi radio until it receives the probe response, which typically takes only a few milliseconds [12]. On the other hand, active discovery requires the transmission of packets containing information about the mobile device. While probe requests sent by devices running older OS might contain SSIDs of one or more APs the device has previously been connected to, newer devices transmit only the SSIDs of hidden networks to improve user privacy and make the device less traceable (cf. Section 2.3). Additionally, they omit the real MAC address of the device, instead sending a randomised MAC address.

Probe Requests are sent in bursts, every burst containing several probe requests sent via some or all of the 14 channels of the 2.4 GHz spectrum (and additionally the 5 GHz spectrum if applicable) within a short time span of just a few milliseconds. Whether MAC address randomisation is employed or not, all packets in a burst are sent from the same MAC address.

## 2.2 Privacy Implications

A MAC address consists of 6 bytes typically represented in hexadecimal notation, separated by colons, e. g. `01:23:45:ab:cd:ef`. The first three bytes are called the Organizationally Unique Identifier (OUI) and are typically assigned to the manufacturer of the devices. The last three bytes identify the Network Interface Controller (NIC), produced and assigned by the manufacturer. The OUI contains additional information encoded in the two least significant bits (U/L and I/G) of the most significant byte (`01` in our example): The I/G-bit is the least significant bit and specifies whether the recipient is unicast or a multicast. The second-least significant bit, the U/L bit, clarifies whether the address is locally or globally

No.	Time	Source	Info
2284	874.747357275	46:8e:45:03:1c:02	Probe Request, SN=1954, SSID=WootWootKarneval
2285	874.781384907	46:8e:45:03:1c:02	Probe Request, SN=1955, SSID=Wildcard (Broadcast)
2286	874.782041635	46:8e:45:03:1c:02	Probe Request, SN=1956, SSID=TestingThisCrazyIdea
2287	874.782770066	46:8e:45:03:1c:02	Probe Request, SN=1957, SSID=alalalalalong
2288	874.787660896	46:8e:45:03:1c:02	Probe Request, SN=1958, SSID=WootWootKarneval
2294	881.989862003	3e:15:5a:f5:1f:74	Probe Request, SN=404, SSID=Wildcard (Broadcast)
2296	882.031938926	3e:15:5a:f5:1f:74	Probe Request, SN=408, SSID=Wildcard (Broadcast)
2297	882.032516694	3e:15:5a:f5:1f:74	Probe Request, SN=409, SSID=TestingThisCrazyIdea
2298	882.033237567	3e:15:5a:f5:1f:74	Probe Request, SN=410, SSID=alalalalalong
2299	882.038153435	3e:15:5a:f5:1f:74	Probe Request, SN=411, SSID=WootWootKarneval
2300	882.072108931	3e:15:5a:f5:1f:74	Probe Request, SN=412, SSID=Wildcard (Broadcast)
2302	882.074146595	3e:15:5a:f5:1f:74	Probe Request, SN=413, SSID=TestingThisCrazyIdea
2303	882.074876650	3e:15:5a:f5:1f:74	Probe Request, SN=414, SSID=alalalalalong
2306	882.079439335	3e:15:5a:f5:1f:74	Probe Request, SN=415, SSID=WootWootKarneval
2313	887.911035684	12:19:a8:58:f5:39	Probe Request, SN=1100, SSID=Wildcard (Broadcast)
2315	887.913045924	12:19:a8:58:f5:39	Probe Request, SN=1101, SSID=TestingThisCrazyIdea
2316	887.913747853	12:19:a8:58:f5:39	Probe Request, SN=1102, SSID=alalalalalong
2318	887.917433850	12:19:a8:58:f5:39	Probe Request, SN=1103, SSID=WootWootKarneval
2323	887.951525908	12:19:a8:58:f5:39	Probe Request, SN=1104, SSID=Wildcard (Broadcast)

Fig. 1: Three bursts of probe requests sent from the same device. Three different SSIDs and the wildcard SSID, an empty string, are broadcast. Note that the starting sequence number (SN) in the info field is randomised per burst as well.

administered, with a globally administered address being a unique identifier for the physical device, while a locally administered address temporarily overwrites the unique global one in software [34]. Older devices use their universal address to broadcast probe request, which makes them easily trackable. To protect the privacy of users and prevent device tracking, probe requests are often sent from locally administered addresses, employing a technique called MAC address randomisation. Here, the MAC address commonly changes between two bursts, such that each burst will be sent from a new, random MAC address. This behaviour was first introduced in iOS 8 in 2014 [11] and in Android 8 [13] in 2017, albeit the first implementations suffered from information leaks: it was often possible to track devices despite the use of MAC address randomisation [32], for example by the SSIDs they contained. If SSIDs are present in a probe request, either all of them or a subset is contained in a burst, with every packet requesting one SSID. Fig. 1 shows the capture of three bursts of probe requests sent from the same device employing MAC address randomisation but transmitting SSIDs.

While omitting SSIDs and employing MAC address randomisation renders a device less trackable, other fields included in probe requests can spoil the effect: if the sequence number (SN) is not randomised, it is trivial to still follow a device over time. Therefore, a lot of devices randomise their sequence number with the start of every burst, as can also be observed in Fig. 1.

The newer a device and its OS is, the more information is omitted and fields randomised in the probe requests. All the same, various papers still describe how even modern devices can be fingerprinted due to other information contained in them, e. g. in the Information Elements (IE): These non-mandatory parameters contain information on supported rates, network capabilities, and more. Combining the IE parameters, the signal strength and, in some cases, the sequence number, allows to fingerprint individual devices despite MAC address randomisation. [32,30] While efforts are made to reduce the fingerprint of modern

devices, the owners of older devices that don't receive patches introducing MAC address randomisation, sequence number randomisation and SSID omission can easily be tracked.

### 2.3 Differences between Android and iOS versions

Table 1 shows the differences between iOS and Android in supporting Wi-Fi-related privacy features. We compare iOS versions 8, 10, 14 and 15 and Android 8 to 12. Their combined market share comprises about 90 % of the devices [26,27], which makes them representative and provides a good overview over the changes within the last years. In the following, we elaborate on the various features comprised in Table 1.

Table 1: Privacy features for probe requests in different mobile OSs.

	Apple iOS				Android				
	8	10	14	15	8	9	10	11	12
Market Share in %	< 0.1	1.0	35.9	53.4	10.2	13.5	27.0	35.4	1.9
Randomised MAC ...									
- while probing	✓	✓	✓	✓	✓	✓	✓	✓	✓
- per connected SSID	-	-	✓	✓	-	(-)*	✓	✓	✓
- after resetting settings	-	-	✓	✓	-	-	(-)*	(-)*	(-)*
New random MAC after ...	-	-	-	6w	-	-	-	(-) <sup>†</sup>	(-) <sup>†</sup>
Private Address by default	-	-	✓	✓	-	-	✓	✓	✓
Modify distant Network	-	-	-	-	✓	✓	✓	✓	✓
Manually added == hidden	Automatic detection of hidden				✓	-	-	-	-
Probe with SSID	Only if hidden detected				if man. added	If explicitly declared hidden			

\*: Only choosable via Developer Options

†: If use of non-persistent MAC is chosen via Developer Options, a new MAC is set (a) for every new connection establishment (b) every 24 hours, unless a connection is still established or (c) if both the DHCP lease has expired and the device has been disconnected for 4 hours

All listed versions use MAC address randomisation while probing [19,13,34]. In Android 9 devices, users can choose via Developer Options whether a randomised MAC address should be used while connected. Since Android 10 and iOS 14, private addresses are used by default: They all employ persistent random MAC addresses while connected. Starting with Android 11, one can choose via Developer Options to use non-persistent randomisation per stored SSID during connection. If non-persistent MAC addresses are used in Android, the MAC address is re-randomised either (a) with every new connection establishment, (b) every 24 hours, but without disrupting the network connection to switch to a new MAC address or (c) if the DHCP lease has expired and the device has been disconnected for at least 4 hours. Nevertheless, up to this date, there are no

Android devices that automatically receive a new random MAC address after a certain amount of time by default, without having to modify the Developer Options. Additionally, with persistent private MAC addresses in Android, the default behaviour is to persist a MAC address per SSID even after resetting network settings. This is different in iOS: Starting with iOS 14, removing and adding a network again causes a reset of the network persistent address. In iOS 15, the devices additionally receive a new address when not connected for more than 6 weeks.[2,14,34]

All of the listed Android versions offer to remove any saved network at any given time. This is not the case in iOS: Here, a network can only be removed from the device while in physical proximity of it or by modifying the iCloud Keychain from a MacBook. Without access to a MacBook or physical proximity to the network, it can not be removed without resetting the entire network settings.[21]

When adding a network manually, iOS verifies whether it is a hidden network or not, whereas Android 8 (and earlier) automatically assumes that manually added networks are hidden networks. Therefore, if a network was manually added, Android 8 devices send the SSID in probe requests, while newer Android versions only do so if the added networks were explicitly declared hidden (cf. Fig. 2b in Section 6.2). In iOS, the SSID is only used in probe requests if the network is detected to be a hidden network.[3,20]

### 3 Related Work

In 2013, Cunche et al. showed how to link various devices by their transmitted SSIDs and inferred relationships between users [5]. This work was published before MAC address randomisation was deployed and free transmittal of SSIDs the typical means of network discovery. The authors propose the use of a geolocation-based service discovery instead of active discovery via probe requests. In 2014, MAC address randomisation was first discussed [34] and subsequently tested and published [4]. It was meant as a means to increase privacy, but since it lacked standardisation, all implementations were vulnerable to attacks [32]. Since then, extensive work has been published on probe requests, MAC address randomisation and fingerprinting devices despite MAC address randomisation: 2015, Freudiger et al. gave an overview over the amount of probe requests sent by different devices and analysed the effectiveness of the MAC address randomisation employed in different devices [12]. On a positive note, Freudiger pointed out that recent mobile operating systems only probe for SSIDs of hidden networks. In another influential publication, Vanhoef et al. investigated how well devices can be tracked by combining various fields in probe requests [32]. They also present two attacks that can be used to reveal the real MAC address of a device and summarise that MAC address randomisation is insufficient to impede tracking.

Various other papers in the field attempt to associate probe requests from randomised MAC addresses: Gu et al. use deep learning methods and suggest to encrypt probe requests using the symmetric stream cipher ChaCha20 to protect them from attackers [16]. Tan et al. use minimum-cost flow optimisation to

associate frames and reach an accuracy of more than 80 % [30]. As the use of hidden networks and the amount of devices broadcasting SSIDs are decreasing, both papers put only a minor focus on the transmitted SSIDs.

In 2019, Dagelić et al. [6] observe the occurrence of SSIDs in probe requests at a music festival between 2014 and 2018 and present how easy devices are trackable via probe requests if the devices are fingerprintable. Over the years, the number of MAC addresses they observe increases while the number of SSIDs decreases. They conclude that the use of MAC address randomisation is increasing, as is the number of probe requests that contain the empty wildcard SSID.

An attempt at localisation of criminal groups via probe requests was published by Zhao et al. in 2019 [33]. They build a database of SSIDs like WiGLE and monitor probe requests in different locations in search for specific SSIDs. This methodology allows them to find and track devices belonging to a targeted group.

With respect to protecting the content of probe requests, Pang et al. [22] published an architecture called Tryst in 2007 to conceal confidential information during service discovery. Tryst makes use of access control primitives using symmetric encryption, with which it reveals information to the correct access point while concealing all information not directed at it. It remains unclear how exactly the various SSIDs present in the SEND primitive are concealed from everyone except for the intended recipient. They underline the privacy risks of both APs transmitting SSIDs and mobile devices transmitting probe requests by analysing geoinformation on SSIDs collected in a 2004 data set: they find that about a quarter of the devices probe for SSIDs that uniquely appear in just one city. While Pang et al. also perform geolocalisation of SSIDs like we do, to the best of our knowledge, there has been no publication analysing the content of SSIDs of probe requests as we do in this paper and neither one proposing hash-based anonymisation of probe requests to this date.

In the following, we first introduce the experiment and then strive to demonstrate the privacy implications of the use of such verbose devices on their users.

## 4 Experimental Setup

In this field experiment performed in November 2021, we recorded probe requests in a busy pedestrian zone in the centre of a German city, over the period of one hour, three times in total. We used six off-the-shelf antennae: three for channels 1, 6 and 11 in the 2.4 GHz spectrum and three for channels 36, 40 and 48 in the 5 GHz spectrum. Since our particular focus lies on privacy violations arising from the information contained in the SSID field, we evaluate it with respect to the following:

- The amount of probe requests containing non-empty SSIDs.
- The amount of SSIDs sent per burst.
- Privacy implications of transmitted SSIDs: what potentially personal data can be gleaned from the data set?
- The use of MAC address randomisation.

We calculate an intersection between the data sets of two different days and remove all probe requests by devices that appear in both of them. That way, we strive to isolate the permanent devices in the vicinity of the measurement to have a clearer view on devices more likely to represent human passersby.

#### 4.1 Potential Ethical and Privacy Concerns

A modern smartphone might use MAC address randomisation and refrain from transmitting SSIDs and thereby protect the identity of its user and render itself less trackable. Older devices are often less privacy sensitive, transmit their real MAC address and maybe even known SSIDs. This data can be considered personal data and should therefore only be collected and stored with particular care for the device owner’s privacy. To ensure ethical data aggregation, we submitted our study for approval to the ethics committee of the Informatics department of the University of Hamburg under case number 002/2021. The steps taken to protect the peoples privacy as observed and in accordance with the ethics committee can be found in Appendix A.1.

## 5 Data Analysis

Our field data set contains 252 242 probe requests. We found that overall, 23.2% of the probe requests contained SSIDs. Prior measurements done by Dagelić [6] between 2014 (46.7%) and 2018 (12.9%), and also Vanhoef [32] in 2016 (29.9% to 36.4%), revealed higher numbers in 2014 and 2016, from which a decline is absolutely expected. At the same time, the records of 2018 and our measurements do not match up. One explanation might be, that while a measurement at a music festival might record the probe requests of younger people with more recent devices that already omit SSIDs in probe requests, our measurements were taken in the city centre of a touristic city around noon, where perhaps a larger percentage of people kept their (older) devices over a longer period of time. Our numbers do however correlate with the market share of Android devices [26]: 10.2% of Android devices use Android 8, and devices older than Android 8 amount up to 12%. In these devices, manually added networks are considered hidden networks [20] and they are therefore probed for with SSID. Seeing that Android devices make up approximately 70% of the market share, while iOS devices make up around 29% [28], the percentage of SSIDs in the data set is slightly higher than expected.

During our measurement, 116 961 probes (46.4%) were captured in the 2.4 GHz spectrum, of which 28 836 (24.7%) contained at least one SSID. In the 5 GHz spectrum, we recorded 135 281 probes (53.6%), of which 29 653 (21.9%) contained an SSID.

To prepare the probe requests for analysis, we first grouped all requests that were sent from the same MAC address within a period of four seconds into bursts. We then grouped all bursts into clusters of bursts, likely belonging to a single device, if their PNL was equal. We explicitly did not group requests into the same

Table 2: Distribution of the number of SSIDs per cluster.

# SSIDs	1	2	3	4	5	6	7	8	> 8
Share	67.8%	8.2%	4.0%	6.5%	2.7%	2.2%	0.9%	6.6%	1.1%

cluster if their PNL matched only partly to avoid misclassification of distinct devices with partly overlapping PNLs. At the same time, a cluster with only one SSID might contain requests from distinct devices. As can be seen in Table 2, 67.8% of the bursts contain just one SSID, while the remaining 32.2% contain more than one SSID and are unique enough to track devices with it. This is considerably less than Vanhoef et al. recorded in 2016 [32]: In their data set, 53% to 64.8% of the bursts contained a unique PNL.

Of the probe requests containing an SSID, we identified at least 362 devices sending requests from multiple randomised MAC addresses. 542 devices used only one MAC address, and did not employ MAC address randomisation.

In an additional evaluation, we found that the average amount of probe requests sent per unique MAC address was 4.8. This is, again, a legitimate decline in comparison to the capture by Dagelić [6] et al. in 2014 (24.1), 2015 (29.2) and 2017 (6.1) respectively, but is, again, higher than their 2018 count (2.6). For packages including SSIDs, the average amount of probe requests sent per MAC address was 11.2, which confirms that these are likely older devices of which less employ MAC address randomisation.

In the following, we analyse the values contained in the SSID field of probe requests in order to estimate the privacy violations that occur in their commercial collection and analysis.

### 5.1 SSID Contents

As mentioned in Section 2.3, devices running Android 8 and lower treat manually added networks like hidden networks [20]. We conjecture that a lot of the SSIDs in our record originate from users trying to set up a network connection manually by entering both SSID and password through the advanced network settings, and, apparently mistakenly, enter the wrong strings as the SSIDs. The devices then retransmit the PNL with every probe burst. This results in significant additional information for fingerprinting devices compared to the empty wildcard SSID that would be transmitted otherwise.

In the following, we elaborate on our findings of a manual, as well as automated inspection of the encountered SSIDs.

**Password Leaks in SSID Broadcasts.** A small but significant amount of probe requests containing SSIDs potentially broadcast passwords in the SSID field: We identified that 11.8% of the transmitted probe requests contain numeric strings with 16 digits or more, which are likely the initial passwords of popular German home routers (e. g., FritzBox or Telekom home router). This hypothesis

is supported by various cases, in which the numeric strings follow a `PW:`, `WPA:` or `(WPA/WPA2:)`. Also, we repeatedly found both a 16- or 20-digit string and in the same burst additionally the same string, but separated by a space, a dot or a comma every four digits (e.g. `1234567812345678` and `1234 5678 1234 5678`), which is a typical way of improving readability of the initial password on the router case. The multitude of similar spellings support our assumption that the users tried the same way of logging in multiple times with different spellings of the same credentials.

Leaking passwords in SSIDs is especially critical if, along with the password, the device also broadcasts the true SSID either correctly or with a mistype that can be used to infer the true SSID. Only 2.8% of the transmitted SSIDs classified as probable passwords were the only entry in the corresponding PNL. All other probable passwords were transmitted in bursts with other SSIDs that might contain the actual SSID belonging to the password. The assumption that the sniffed passwords correspond to SSIDs that were also transmitted could additionally be verified by setting up fake access points on the fly using the potential credentials we observed. As that would constitute an active attack on the devices and since we are determined to improve user security, not undermine it, we decided against employing fake-AP attacks.

Moreover, Wi-Fi locations can often be gathered using Wi-Fi mapping services like WiGLE, as we demonstrate in Section 5.2. Additionally and with enough criminal energy, an attacker could follow the owner of a talkative device to their home and try out the password in their home network.

**Broadcasts of SSID Mistypes.** In a manual analysis of the data set, we found various devices that broadcast multiple different spellings of presumably the same SSID. We assume that users manually entered them into their devices while trying out for different spelling and capitalisation variations, e.g., `my network`, `MY_NETWORK`, `MyNetwork`. We quantify the amount of mistyped SSIDs by calculating the normalised edit distance between all SSIDs in a burst stemming from a single device. The edit distance defines the minimum amount of operations (insertions, deletions or substitutions of characters) needed to transform one string into another. Since the edit distance can also be calculated over strings of different length, we normalise the result with respect to the longer string length, i.e., we divide the edit distance by the maximum length of the two SSIDs. Similar strings have an edit distance close to 0, while the edit distance is closer to 1 the more strings differ. Before calculating the edit distance, all input strings are transformed to lowercase, as otherwise, the normalised edit distance between SSIDs like `NETWORK` and `network` would evaluate to 1. We set the threshold at which the strings are considered similar and thus treated as mistyped to 0.3. This way, strings that differ in less than 30% of their letters are considered similar. We decided on such a high threshold to accommodate short SSIDs as well as long ones. A manual inspection verified that the results fulfill the typo criteria and nevertheless do not contain distinct network names like `"Fritz!Box 7490"` and `"Fritz!Box 7590"`.

We found that 19.9% of the transmitted SSIDs, stemming from 138 distinct bursts, are similar enough to another SSID in the same burst to be considered a typo. Such a set of constantly transmitted misspelled SSIDs increases the fingerprint of a device drastically and makes tracking it easy.

**Additional findings.** We found at least one string that corresponds to a store and the Wi-Fi password of the store’s internal Wi-Fi. We deem this highly likely as it began with the letters ”PW:” and contained the name of the store in the password. We identified 106 distinct first and/or last names, which were propagated 3339 times over the course of the experiment. We found three e-mail addresses that were propagated 36 times. We identified 92 distinct holiday homes or accommodations whose SSIDs users had added to their list of known networks, which were propagated 1257 times. In addition, we found the name of a local hospital broadcast in two different spelling variations 15 times. It is particularly shocking to see such sensitive information like an e-mail address being transmitted openly, let alone the hospital name, from which a potential stay at the hospital can be inferred. At the same time, the name of a person or the hotels and holiday homes in which they have stayed can also be used to draw conclusions about the person.

## 5.2 Geolocation Discoverability and Uniqueness

To provide a better estimate of whether an SSID exists or not, we ran all observed SSIDs through the geolocation lookup API of WiGLE. This way, we were able to find out whether the captured SSIDs correspond to actual APs catalogued by WiGLE. Of course, this approach has one limitation: Mobile devices should, in a perfect scenario, only transmit SSIDs of hidden networks. Those should not be included in the WiGLE map at all, as the service only maps the transmitted SSIDs.

To evaluate the uniqueness of location of the SSIDs we found, we performed an analysis on the coordinates that WiGLE returned. To reduce the accuracy of the location estimation, we limited the amount of decimal places of the coordinates to 2, thereby providing an approximate 1-kilometre radius in which the actual network can be found. This also removed artifacts like multiple networks with the same SSID found within a radius of a few metres, that most likely belonged to the same network. Our input consisted of 1478 unique SSIDs. We had to limit our evaluation to a subset of 1440 SSIDs, as the remaining 38 contained special characters, which WiGLE can’t resolve. We were able to pinpoint 334 SSIDs to one unique location and 377 SSIDs returned multiple locations. 729 (50.6%) of the SSIDs could not be localised anywhere in the world. The latter are either hidden networks that weren’t mapped by WiGLE due to being hidden or mistyped SSIDs.

**Password Evaluation.** To provide an estimate of how many SSIDs contained passwords, we filtered the list for

- strings that contain 16 or more numeric digits and
- strings that contain "pass", "pw", "kennwort" (the German word for password) or "wpa".

Our input consisted of 77 unique strings classified as passwords. The WiGLE evaluation resolved only a single one of the strings to a unique location. We infer that this is strong evidence that the identified strings are in fact actual passwords.

**Typo Evaluation.** We performed the same evaluation on the potential typos we identified. In this analysis, we inserted all spelling variations we found of an SSID into our evaluation, which amounted to 296 unique strings. The hypothesis in this case was that at least half of the SSIDs we identified were in fact typos and would not resolve to an existing SSID. We assumed that it would be more than 50% since quite a few of the potential typos had more than one spelling variation. We assumed that the remaining SSIDs would contain the correct and actual spelling that could resolve to an access point.

Our analysis showed that of the 296 strings classified as potential typos, we were able to resolve approximately 41.9% of the SSIDs and identified 47 unique locations and 66 cases of multiple locations. These results support our hypothesis.

**Limitations.** The evaluation of networks contained in WiGLE is severely limited: Only networks that are *not* hidden appear in it, while probe requests of current devices target *only* hidden networks. Nevertheless, we found a large percentage of the networks do in fact exist. This can have several explanations: (1) multiple networks with the same name exist, and the one in question is in fact a hidden one, (2) the network has been set to hidden recently, and the map still contains the result of a scan from before, (3) the network was manually added in WiGLE.

## 6 Mitigations for Increased Privacy

Our experiments detailed in the previous section suggest that users, presumably mostly by accident or unwillingly, add items to their list of preferred networks, including credentials and sensitive information. Together with legitimately added hidden networks, those threaten user privacy by being sent in plain, making this information observable and the users traceable. To mitigate the issue, we first present a proposal to avoid plain text transmission and then show approaches to limit and control traceable SSIDs through the user interface.

### 6.1 Hashing SSIDs in Probe Requests

Recall that the introduction of the wildcard SSID in probe requests makes active scanning with specific SSIDs only necessary for hidden networks. While some publications consider hidden networks obsolete and no longer recommended [15],

a recent study [25] revealed that in some areas, up to 44 % of the detected networks were hidden. While the WPA3 standard contains improvements to the confidentiality management frames (802.11w) [9], this standard only applies to frames transmitted after a 4-way handshake, and not to frames transmitted without handshakes. Consequently, probe requests are not protected. To rectify this, we propose the following mitigation.

**SSID Hashing.** To circumvent the need to send cleartext SSIDs, we propose to send them in a hashed and salted manner instead. The device emitting the probe request would first salt the SSID using its randomised sender MAC address and the sequence number of the packet and then hash it. It would then send the hash, but omit both salt components as they are included in the frame anyway like so:

$$send(hash(MAC||SN||SSID))$$

Access points of hidden networks would then, upon receiving the probe request, prepend the MAC address and sequence number as salt to their own SSID, hash it, and compare the result with the received hash. If they match, the client was probing for their hidden network. As the MAC address should be chosen randomly and the sequence number changes with every packet, they introduce sufficient entropy and variability in combination to be suitable as salt to make sure the sent information can not be used to track and identify devices through a constant hash value. This mitigation could be employed regardless of whether or not a connection to the network has ever been established before, as it does not require the previous exchange of secrets. Another advantage of this mitigation is that potentially sensitive SSIDs (e. g., containing names or passwords) can only be distinguished from other SSIDs (e. g., generic home router names) with significant effort of brute-forcing the cleartext, thereby improving the privacy of clients and AP operators.

**Attack Model.** We consider an attacker that can monitor all probe requests and has bounded computational power, which makes it impractical for her to find preimages (SSIDs) of the hashes she observes. Introducing the combined MAC address and sequence number salt makes pre-computing hashes impractical, provided that MAC addresses are randomised and sequence numbers are also randomly chosen within their full 12 bit value range. If the randomised MAC address contributes 24 bit of entropy, leaving out the EUI/CID parts as a lower estimate, this totals to 36 bit of salt entropy. Attackers with a priori knowledge of used SSIDs can however, reduce costs significantly compared to true brute-forcing, depending on the number of likely used SSIDs. Consequently, tracking a device whose SSID set is known is therefore feasible.

To determine the practical feasibility of a hash-based mitigation, we first evaluate the computational cost and then calculate the additional bandwidth requirement.

**Computational Overhead.** Using the SHA-256 hash from Python’s `hashlib` library, we prototyped the hashing and comparison that an AP would have to perform: first hashing a received SSID and then comparing it to a known hash (the AP’s SSID). For baseline reference, we implemented the current string comparison of SSIDs. We performed one million computations in three runs on a Raspberry Pi Model 3, which is likely a good lower estimate of computational power of most routers. A single computation with hashing required on average 11.7 microseconds pure CPU time, while the baseline required an average of 4.6 microseconds. Using hashing therefore increases the time by 153.7%. Considering that the Raspberry Pi does not have cryptographic hardware acceleration, which professional Wi-Fi routers might have, we additionally ran the experiment on a laptop with an Intel i5, where hashing only added an average overhead of 53%.

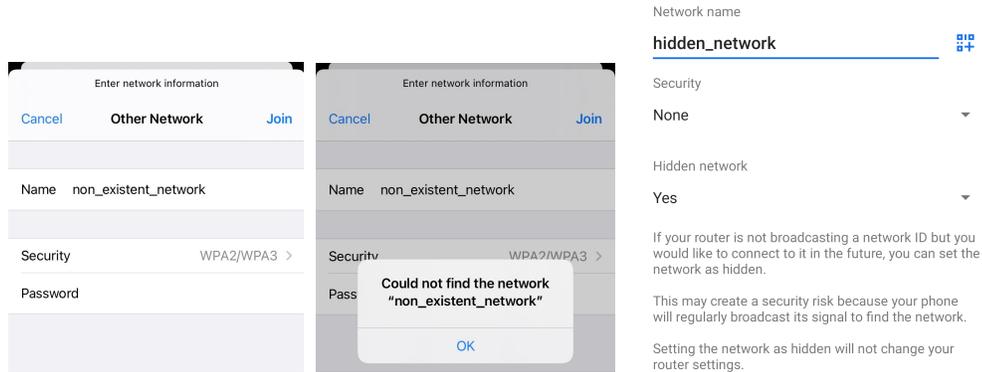
In our experiment in a busy pedestrian zone, we captured around 23 probe requests per second, of which only 23.2% contained an SSID. Therefore, following these figures, an AP would have to hash approximately 5.3 probe requests per second. As a Raspberry Pi can perform around 85 200 hashing operations per second, we deduce that hashing and comparing should be well within the available resources for similarly equipped APs even in much more frequented deployment locations.

**Bandwidth Overhead.** Our proposal may also introduce a bandwidth overhead by always occupying the full 32 bytes available for SSIDs in a probe request frame [18, Sec. 9.4.2.2], whose length would otherwise vary with the actual SSID length. The average length of all packets in our city centre capture is 133.3 bytes, while the average length of packets containing SSIDs is 147.0 bytes. In our capture, the average length of SSIDs was 11.4 bytes. If all SSIDs were transmitted as hashes, it would increase the size by 20.6 bytes, leading to an average size of packets with SSID of 167.6 bytes, which is an increase of 14.01%. Considering that probe requests make up a tiny fraction of the actual transmitted traffic, we consider this an acceptable trade-off for more privacy and less fingerprintability.

## 6.2 Mitigations through User Interface Design

Current iOS and Android version already employ mechanisms to prevent users from accidentally adding items to their PNL, to maintain that list and to change the connection behaviour for individual networks on this list. In the following, we briefly summarise the status quo and suggest further improvements for more accessible and effective controls.

**SSID Entry Safeguards.** Both iOS and Android have safeguards against accidentally adding hidden networks: iOS will only add networks to which a connection can be established at the time of entry (see Fig. 2a). In Android, a manually added network is no longer automatically considered a hidden network. Instead, to enter a hidden network, users have to explicitly select it and then receive a warning about the privacy risks (see Fig. 2b). We suggest to combine both measures for manually adding SSIDs.



(a) Attempting a connection to a hidden network on iOS 15. Contrary to Android devices, iPhones only allow to manually add networks to which a connection can be established. (b) Warning message when adding a hidden network on Android 9 or newer.

Fig. 2: User dialogues mitigating unwanted SSID entry.

**Known SSID Removal.** Being able to remove entries from the list of preferred networks should be possible to reduce traceability and susceptibility for fake AP attacks [32]. However, as mentioned in Section 2.3, removing a known network from proximity is not straightforward in iOS. On Android, in contrast, the list of known networks can be modified directly and at all times.

**PNL Entry Expiry.** In order to maintain its usability, user-facing lists of preferred networks should implement measures against cluttering, which result from steadily adding new SSIDs over time. To avoid cluttering with no longer needed SSIDs, e. g., added during temporary stays, we propose an expiration date for SSID entries: Upon adding a new SSID, the user is prompted to choose when the SSID should be forgotten again. The default expiry date would be *never*, but for SSIDs that are knowingly only in use for a limited period of a few days or weeks, users can choose accordingly. By limiting the life span of a network entry, its negative effects on traceability and the chance to exploit them using fake AP attacks are at least temporally limited. To the best of our knowledge, there is no such mechanism in any OS, but to implement it could significantly reduce the amount of SSIDs accumulated over time.

**Adjustable Auto-Joining.** In addition to networks of limited temporal relevance that benefit from expiry, there are also networks users use only occasionally but regularly, e. g., once every few months or years. While it is convenient to keep them in the PNL, this again increases the risk of fake AP attacks. Instead of removing such networks, preventing the automatic connection to them already effectively reduces the fake AP attack risk. Both iOS and Android offer ways to

disable auto-joining on a per network basis. To make this option more visible and broadly realise privacy gains, we suggest to again prompt users when initially joining a network, whether they want to automatically or manually connect in the future.

**Silencing Probe Requests.** For particularly high privacy demands, disabling probe requests altogether might be an acceptable trade-off. We therefore suggest an advanced network setting, where users are able to choose that their devices do not send active probe requests *at all*, knowing that (a) connection establishment relies only on passive AP announcements and might be slower, (b) the battery usage might be higher, and (c) the connection to hidden networks would be impossible. Such behaviour could also be part of a *reduced visibility* mode that user, e. g., activate in the control center, when they pass through an untrusted area like a shopping centre known for extensive visitor analytics, similar to a do-not-disturb switch.

## 7 Discussion

**Legal Consideration of SSIDs.** As mentioned in the introduction, Wi-Fi tracking is also used to measure pedestrian flow and count in cities. In two German cities, such measurements have been conducted until the responsible authority started investigations [31]. The authority’s reasoning was that MAC addresses are personal data and it is therefore not legal to record them without legal basis. In both cases, the measurements were ceased. Especially as the continued roll-out of MAC address randomisation might give renewed support to the legal positions that (randomised) MAC addresses should no longer be considered identifying information, we argue that this assessment can and should not be limited to the sender address of probe requests. Considering the wealth of personal and sensitive information we observed in SSID fields, they can constitute identifying information as well and thus require due consideration. For instance, for 334 of the SSIDs we measured, we were able to identify a unique location the SSID originated from. We therefore argue that at least for as long as there are still devices broadcasting SSIDs, probe requests should be considered personal data and not be used for monitoring without legal basis.

**Intentional Password Broadcast.** Some APs might intentionally broadcast their password as SSID to allow visitors an unrestricted yet encrypted Wi-Fi connection, as an alternative to unprotected Wi-Fi. It is a limitation of our password leakage evaluation that we cannot distinguish those intentional from unintentional cases. However, the fact that we could only resolve one of these SSIDs in our geo-lookup suggest that the ratio of SSIDs to actual APs is rather low. Additionally, the notion of intentionally storing passwords in the SSID field should have been superseded by the introduction of OWE (Opportunistic Wireless Encryption): OWE describes the unauthenticated but encrypted connection

between two devices, and is contained in the Wi-Fi specification under the name *Wi-Fi CERTIFIED Enhanced Open™* [7].

**Deployment of Hashed SSID Scheme.** Our proposed hash-based scheme requires modifications to the Wi-Fi implementations: The mobile device has to apply the hashing algorithm to its temporary MAC address, sequence number, and the SSID of the sought-after network before transmission. The AP, upon receiving a probe request, has to apply hashing to the MAC address and sequence number as salt to its own SSID. On one hand, these changes are easy to implement, on the other hand they require a widespread deployment to be effective. While the privacy gain would be worth the deployment effort, it is unfortunately likely that only newer devices would profit from the scheme, while older devices would remain unpatched.

**Limiting Bandwidth Overhead.** Our proposed method of salting and hashing the SSIDs of hidden networks to improve user privacy introduces a bandwidth overhead, (cf. Section 6.1): The average length of a packet containing an SSID would be increased by 14%. This could be addressed by truncating the hash to, e. g., 16 bytes before inserting it into the SSID field. That way, the average packet length would be reduced to 151.6 bytes, which results in an overhead of only 3.2%. At the same time, this reduces the security of the system, as hash collisions become more likely.

**Impact of OS Support Lifespans.** A contributor of non-wildcard probe requests in the wild, besides hidden networks, are legacy devices. While devices running Android 10 or newer use MAC address randomisation and omit SSIDs, older devices do not receive updates long enough to benefit from this improvement. This especially disadvantages people with budgetary constraints or sustainability in mind, who keep their devices longer. This can only be rectified by longer support lifespans that should be legally mandated if manufacturers do not move voluntarily. Such changes with significance to user privacy should be considered like critical security patches and be back-ported to older versions.

## 8 Conclusion

Probe requests are plainly observable to everyone around a sending device. Since they can contain sensitive data, they should be sent more carefully and with privacy in mind. We have collected and analyzed data in a pedestrian zone to gather insight into the status quo of probe requests in the wild. We identified a wealth of personal information in transmitted SSIDs such as potential passwords, holiday homes and e-mail addresses, which should be considered personal and sensitive data. For 334 SSIDs, we were even able to resolve their unique geographic location.

In the area of anonymisation of probe requests, some progress has been made in the last few years, with the latest mobile operating system updates. Nevertheless, we are still facing problems in terms of user privacy. To minimise the amount of personal data that can be sent accidentally, we propose mitigations for both the network layer and the user interface. The first consists of hash-based concealment of SSIDs using a salt constructed from the MAC address and the sequence number of the request. To demonstrate its feasibility we provide estimates of the computational overhead and additional bandwidth requirements. For the latter, we propose to remodel the Wi-Fi handling of mobile devices to add only existing networks in range and to empower their users to take more control over their PNL and minimise the amount of potentially exploitable data.

## Acknowledgements

We would like to thank our reviewers for their valuable and constructive feedback.

## A Appendix

### A.1 Ethical collection of probe requests

Following approval and in coordination with the ethics committee of the informatics faculty of the University of Hamburg, we conformed to the following measures to observe and protect users' privacy rights:

- During the time of the experiment, we set up a well visible sign declaring the undergoing probe request monitoring, including information on how to contact the person in charge.
- We informed and obtained consent from building management to conduct the experiment.
- We provided an option to remove recorded probe requests should participants state their non-consent.
- We used off-the-shelf wireless USB antennae with a limited range to narrow the radius of our measurement
- In case the data set contains personal information, we either anonymise it before storing, or delete it directly after analysing it.
- Any personal data is stored securely, both technically as well as organisationally, to prevent misuse.
- To preserve location privacy, we limit the amount of decimal places of the coordinates returned by WiGLE to 2, thereby providing an approximate 1-kilometre radius in which the actual network can be found.

## References

1. Acar, C.: Whitepaper: Smart Store (2018), <https://www.ehi.org/de/studien/whitepaper-smart-store/>

2. Apple Inc.: Use private Wi-Fi addresses on iPhone, iPad, iPod touch, and Apple Watch, <https://support.apple.com/en-us/HT211227>
3. Apple Inc.: Apple Platform Security (2021), [https://manuals.info.apple.com/MANUALS/1000/MA1902/en\\_US/apple-platform-security-guide.pdf](https://manuals.info.apple.com/MANUALS/1000/MA1902/en_US/apple-platform-security-guide.pdf)
4. Bernardos, C.J., Zuniga, J.C., O’Hanlon, P.: Wi-Fi internet connectivity and privacy: Hiding your tracks on the wireless Internet. In: IEEE Conference on Standards for Communications and Networking (CSCN). pp. 193–198. IEEE (2015). <https://doi.org/10.1109/CSCN.2015.7390443>, <http://ieeexplore.ieee.org/document/7390443/>
5. Cunche, M., Kaafar, M.A., Boreli, R.: Linking Wireless Devices Using Information Contained in Wi-Fi Probe Requests. *Pervasive and Mobile Computing* **11**, 56–69 (2018). <https://doi.org/10.1016/j.pmcj.2013.04.001>, <https://linkinghub.elsevier.com/retrieve/pii/S1574119213000618>
6. Dagelić, A., Perković, T., Čagalj, M.: Location Privacy and Changes in WiFi Probe Request Based Connection Protocols Usage Through Years. In: International Conference on Smart and Sustainable Technologies (SpliTech). pp. 1–5. IEEE (2019)
7. Dan Harkins: Wi-Fi CERTIFIED Enhanced Open™: Transparent Wi-Fi® protections without complexity, <https://www.wi-fi.org/beacon/dan-harkins/wi-fi-certified-enhanced-open-transparent-wi-fi-protections-without-complexity>
8. Deutscher Bundestag: Datenschutzrechtliche Zulässigkeit des WLAN-Trackings (2021), <https://www.bundestag.de/resource/blob/538890/3dfae197d2c930693aa16d1619204f58/WD-3-206-17-pdf-data.pdf>
9. Ebbecke, P.: Protected Management Frames enhance Wi-Fi Network Security (2020), <https://www.wi-fi.org/beacon/philipp-ebbecke/protected-management-frames-enhance-wi-fi-network-security>
10. European Union: Regulation (EU) 2016/679 – General Data Protection Regulation. Official Journal of the European Union **L119**, 1–88 (May 2016), <http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L:2016:119:TOC>
11. Fenske, E., Brown, D., Martin, J., Mayberry, T., Ryan, P., Rye, E.C.: Three Years Later: A Study of MAC Address Randomization In Mobile Devices And When It Succeeds. PETS’2021 pp. 164 – 181 (2021)
12. Freudiger, J.: How Talkative is Your Mobile Device? An Experimental Study of Wi-Fi Probe Requests. In: WiSec’15. ACM (2015). <https://doi.org/10.1145/2766498.2766517>, <https://doi.org/10.1145/2766498.2766517>
13. Google Android Documentation: Implementing MAC Randomization (2021), <https://source.android.com/devices/tech/connect/wifi-mac-randomization>
14. Google Android Documentation: MAC Randomization Behavior (2022), <https://source.android.com/devices/tech/connect/wifi-mac-randomization-behavior>
15. Goovaerts, F., Acar, G., Galvez, R., Piessens, F., Vanhoef, M.: Improving Privacy Through Fast Passive Wi-Fi Scanning. In: Secure IT Systems. pp. 37–52. Springer (2019)
16. Gu, X., Wu, W., Gu, X., Ling, Z., Yang, M., Song, A.: Probe Request Based Device Identification Attack and Defense. *Sensors* **20**(16) (2020). <https://doi.org/10.3390/s20164620>, <https://www.mdpi.com/1424-8220/20/16/4620>
17. Harkins, D., Kumari, W.A.: Opportunistic Wireless Encryption. RFC 8110 (Mar 2017). <https://doi.org/10.17487/RFC8110>, <https://rfc-editor.org/rfc/rfc8110.txt>

18. IEEE: IEEE Std 802.11 - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications (2020), <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9363693>
19. Martin, J., Mayberry, T., Donahue, C., Foppe, L., Brown, L., Riggins, C., Rye, E.C., Brown, D.: A Study of MAC Address Randomization in Mobile Devices and When It Fails. PETS'2017 **4**, 268–286 (2017)
20. Martinez, S.: Update add network dialog to not make networks hidden by default (2018), <https://android.googlesource.com/platform/packages/apps/Settings/+8bc3fa0649a3ecff5e42fb0d14ddb8ff6f7f7507>
21. McElhearn, K.: How to Remove Wi-Fi Networks from Your Mac and iOS Device. The Mac Security Blog (2021), <https://www.intego.com/mac-security-blog/how-to-remove-wi-fi-networks-from-your-mac-and-ios-device/>
22. Pang, J., Seshan, S.: Tryst: The Case for confidential Service Discovery. In: Hot-Nets'07 (2007)
23. Post, T.: Scharfe Kritik an Frequenzmessung. Kieler Nachrichten (2017), <https://www.kn-online.de/Lokales/Eckernfoerde/Datenschutz-Diskussion-um-WLAN-Tracking-in-Eckernfoerder-Innenstadt>
24. Redondi, A.E., Cesana, M.: Building up Knowledge through passive WiFi Probes. Computer Communications **117**, 1–12 (2018)
25. Schepers, D., Ranganathan, A., Vanhoef, M.: Let Numbers Tell the Tale: Measuring Security Trends in Wi-Fi Networks and Best Practices. In: WiSec '21. p. 100–105. ACM (2021). <https://doi.org/10.1145/3448300.3468286>, <https://doi.org/10.1145/3448300.3468286>
26. statcounter: Mobile & Tablet Android Version Market Share Worldwide - Dec 2021 (2021), <https://gs.statcounter.com/os-version-market-share/android/mobile-tablet/worldwide#monthly-202112-202112-bar>
27. statcounter: Mobile & Tablet iOS Version Market Share Worldwide - Dec 2021 (2021), <https://gs.statcounter.com/ios-version-market-share/mobile-tablet/worldwide/#monthly-202112-202112-bar>
28. statcounter: Mobile operating system market share worldwide (2021), <https://gs.statcounter.com/os-market-share/mobile/worldwide>
29. Struever, A.: Vorteile und Nachteile WLAN-Tracking – Ist WLAN-Tracking DSGVO-Konform? (2019), <https://www.expocloud.com/de/blog/vorteile-und-nachteile-wlan-tracking-wifi-tracking>
30. Tan, J., Chan, S.H.G.: Efficient Association of Wi-Fi Probe Requests under MAC Address Randomization. In: INFOCOM'21. pp. 1–10. IEEE (2021)
31. Unabhängiges Landeszentrum für Datenschutz: 5.4.8 - Offline-Tracking / Ortung von Mobiltelefonen in Fußgängerzone. Unabhängiges Landeszentrum für Datenschutz (2021), <https://www.datenschutzzentrum.de/tb/tb37/kap05.html>
32. Vanhoef, M., Matte, C., Cunche, M., Cardoso, L.S., Piessens, F.: Why MAC Address Randomization is Not Enough: An Analysis of Wi-Fi Network Discovery Mechanisms. In: Asia CCS '16. p. 413–424. ACM (2016). <https://doi.org/10.1145/2897845.2897883>, <https://doi.org/10.1145/2897845.2897883>
33. Zhao, F., Shi, W., Gan, Y., Peng, Z., Luo, X.: A localization and tracking scheme for target gangs based on big data of Wi-Fi locations. Cluster Computing **22**(1), 1679–1690 (2019)
34. Zúñiga, J.C., Bernardos, C.J., Andersdotter, A.: MAC Address Randomization. Tech. rep., IETF (Jul 2021), <https://datatracker.ietf.org/doc/html/draft-zuniga-mac-address-randomization-01>