# Improving Privacy through Fast Passive Wi-Fi Scanning

Frederik Goovaerts[1], Gunes Acar[2], Rafael Galvez[2], Frank Piessens[1], and Mathy Vanhoef[1,3]

[1] imec-DistriNet, KU Leuven
[2] imec-COSIC, KU Leuven
[3] New York University Abu Dhabi

**Abstract.** Traditionally, Wi-Fi networks are discovered by actively transmitting probe requests. The alternative, passive scanning, is rarely used because it is substantially slower. Unfortunately, active scanning can be abused to track users based on (physical) fingerprints of probe requests. Previous work attempted to address these issues by making active scanning more privacy-friendly. For instance, Franklin et al. proposed to make implementations more uniform (USENIX Security 2006), and Lindqvist et al. suggested to use encrypted probe requests (WiSec 2009). However, a better approach is to make passive scanning faster. This motivates vendors to use passive scanning, increasing the privacy of users.

Motivated by the above insight, we improve the performance of passive scanning. We implement our proposals on Android, and show the average time needed to connect to a known network using passive scanning now matches active scanning. Additionally, we implement a new network-discovery mechanism that drastically decreases scanning times, and present a new method to fingerprint Wi-Fi radios. All combined, our results show that passive scanning is a viable and more privacy-friendly alternative to active scanning.

**Keywords:** Tracking; Anonymity; Passive Scanning; Priority Scanning

## 1 Introduction

Practically all mobile devices discover nearby Wi-Fi networks by actively sending probe requests [10]. Unfortunately, properties of the physical Wi-Fi signal of probe requests can be abused to fingerprint and track devices [4]. These physical properties of the Wi-Fi signal are caused by unique imperfections in each radio transmitter. As a result, whenever a probe request is sent, no matter what its content, it becomes possible to track the sender. Additionally, probe requests may contain the unique MAC address of the transmitter, and can contain other sensitive information [21,10]. We find that in practice people are indeed being tracked based on the Wi-Fi signals of their devices. For example, garbage bins tracked people in London based on probe requests [5].

Previous works tackled this issue by trying to make active scanning more privacy-friendly. For instance, Franklin et al. proposed to make active scanning implementations more uniform [9, §7], Greenstein proposed to encrypt probe requests [11], and Lindqvist et al. suggested a different protocol to encrypted probe requests [16]. More recently, Vanhoef et al. advised to simplify and unify the content of probe requests [23], and Matte et al. suggested to randomize the timings of probe requests [18]. However, our position is that trying to improve active scanning is not the way forward. We believe this because new tracking techniques against active scanning are inevitable, since in general *any* frame being transmitted can be fingerprinted based on physical-layer properties [4]. To defend against this, we should be making passive scanning more performant, such that it becomes a viable alternative to active scanning.

Under passive scanning, a device does not send any frames to discovery networks. Instead, nearby Wi-Fi networks are instead detected by listening for beacon frames that all APs periodically transmit. These beacon frames contain the same information as probe responses, and hence allow a client to determine all relevant parameters of the network. This makes it impossible to track users based on probe requests. Unfortunately, having to wait for beacons on each channel makes passive scanning significantly slower than active scanning, and therefore it is rarely used. We overcome this obstacle by increasing the performance of passive Wi-Fi scanning.

Apart from privacy pitfalls, active scanning also reduces the available bandwidth. That is, the airtime consumed by probe requests and responses can be quite large. For example, in crowded places they take up more than 10% of available airtime [14], and can reduce the throughput of clients by 17%. Hence, apart for privacy advantages, passive scanning would also free up airtime.

Inspired by the privacy and bandwidth benefits of passive scanning, our goal is to increase its speed, such that it becomes a viable alternative to active scanning. This will incentivise vendors to use passive scanning instead, thereby eliminating the privacy downsides of active scanning. To avoid the longer scanning durations under passive scanning, we introduce the concept of priority scans. Under a priority scan, a set of priority channels is scanned first, and networks detected on these channels are returned immediately. We implement these modification on Android to evaluate our techniques in practice. Additionally, we propose a novel scanning technique where networks advertise all neighboring networks that they can detect, and present a new method to fingerprint Wi-Fi radios. The client can then use this information to drastically reduce the average scanning duration.

To summarize, our main contributions are:

- We present a novel method to fingerprint of type of Wi-Fi radio that a device uses (Section 3).
- We improve passive scanning to reduce the time needed to discover and connect to (known) networks (Section 4).
- We implement and test our proposal on Android (Section 5).
- We propose to advertise neighboring networks to drastically improve the discovery time of known networks (Section 6).

Finally, we discuss related work in Section 7, and conclude in Section 8.

## 2 Background

In this section we introduce relevant parts of the 802.11 standard, and we discuss our threat model.

### 2.1 Network Discovery

An essential task of a wireless devices is discovering nearby networks. The 802.11 standard provides two methods to discover nearby Wi-Fi networks. The first is passive scanning, and the second is active scanning. We briefly introduce both:

**Passive Scanning** All Wi-Fi networks periodically transmit beacon frames. These beacons are used to announce the presence of a network, to synchronize clocks between associated clients, and so on. Since they are periodically broadcasted, clients can passively listen for them to detect nearby networks. Beacons include all network configuration parameters that must be obtained before connecting to a network. In particular, beacons contain the Service Set Identifier (SSID), supported data rates, supported or required security protocols, and so on. By default, APs transmit a beacon every 102.4 ms, though this can be configured differently. Because clients must search for networks on all channels, this makes the default passive scan slow and therefore rarely used.

Some APs can be configured to exclude the SSID in beacons. These are called hidden networks. Doing this has the advantage that nearby clients do not show the SSID (i.e. the network name) in their user interface. To detect hidden networks, active scanning must be used and the SSID of the hidden network must be included in all probe requests. Because including the SSID in probe requests can leak sensitive information about the user, the usage of hidden networks is no longer a recommended practice [19,23,10].

**Active Scanning** With active scanning, the client transmits broadcast probe requests, and in response nearby APs reply with probe responses. These probe responses contain all the information required to connect with a network. As a result, a device can quickly detect nearby networks using active scanning.

Unfortunately, probe requests may contain a significant amount of information about the client's device. For instance, if MAC address randomization is not used, probe requests contain the permanent MAC address of the client. This can be used to trivially track users [3]. Moreover, properties of the physical Wi-Fi signal can also be used to fingerprint and track a device [4]. All combined, whenever a client sends probe requests, it becomes possible to track the user.

Finally, active scanning is the only mechanism able of detecting hidden networks. This is because to detect a hidden network, the client must send a directed probe request that contains the SSID of the particular hidden network. If the hidden network is nearby, it will reply with an (ordinary) probe response.

### 2.2 Frequency Bands and Channels

A Wi-Fi network can operate in various frequency bands, and within one band can operate on several possible channels. The most common frequency bands are the 2.4 GHz and 5 GHz bands. Here, the 2.4 GHz band has 13 channels (in Europe), and out of these 13 channels there are three non-overlapping ones (channel 1, 6, and 11). It is common practice to only use one of these non-overlapping channels, since this avoids cross-channel interference [8].

In the 5 GHz band, there are more than 20 non-overlapping channels. In most regulatory domains (i.e. countries), only a few of these channels can be used freely. All the other channels in the 5 GHz band can only be used if the device supports Dynamic Frequency Selection (DFS). We will refer to these as DFS channels. DFS is a mechanism to avoid interference with radar systems. Among other things, DFS prohibits a device from transmitting until it has listened on the channel fore more than one minute without detecting radar pulses. This means clients are not allowed to immediately send probe requests on DFS channels. Instead, only passive scanning can be used to detect APs on DFS channels.

### 2.3 Threat Model

The adversary we consider aims to identify and track devices when they are not connected to an AP. We assume the adversary controls enough APs, so any probe request that the victim sends will be captured. Additionally, as shown in [4], we must assume the adversary can fingerprint probe requests based on physical properties of any transmitted frame. Put differently, *any* frame sent by a device can be used to identify and track it [4].

## 3 Channel Switch Fingerprinting

In this section we describe a novel method to fingerprint devices. We show how this method allows an adversary to differentiate devices based on the type of internal Wi-Fi radio that a device uses.

### 3.1 Channel Switch Time

When a device is scanning for nearby Wi-Fi networks, it sends one or more probe requests on all non-DFS channels. This means that during a network scan, the Wi-Fi radio is constantly switching channels. Our insight is that the specific type of Wi-Fi radio being used influences how much time it takes to switch from one Wi-Fi channel to another. As a result, if an adversary is capable of accurately measuring the channel switch time, this information can be used to fingerprint the type of Wi-Fi chip being used.

In practice, most devices will use the same (random) MAC address during one network scan over all channels [23,18,17]. In other words, the same (random) MAC address is used to send probe requests over different Wi-Fi channels. This

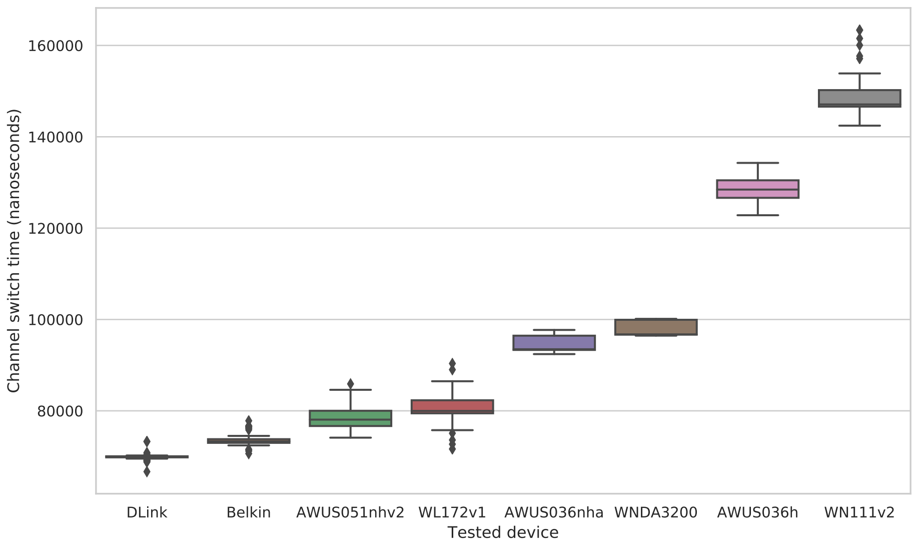


Fig. 1: Box plot of the time between two sequential probe requests sent on different channels, for various types of Wi-Fi radios.

allows an adversary to measure the time between probe requests sent on different channels. Moreover, this time difference can be measured with high accuracy by relying on hardware receive timestamps of commodity Wi-Fi radios. Due to the capture effect, if the victim device is close to the adversary, it is even possible to use a single Wi-Fi radio to receive frames on two adjacent channels. Since multiple channels are scanned in one individual network scan, the adversary can make multiple channel switch timing measurements. This means the average channel switch time can be calculated, reducing the impact of (temporal) noise. All combined, this means commodity Wi-Fi devices can be used to measure the time between probe requests on different channels.

## 3.2 Experiments

We measured the channel switch times of 8 USB Wi-Fi radios. The results of these measurements are shown in Figure 1. We used an Intel Wireless 8265 card to perform the timing measurements. The time difference between two probe requests is calculated based on the hardware receive timestamps of the frames.

From Figure 1 we learn that different types of Wi-Fi radios result in a different average channel switch time. By calculating the average channel switch time over several Wi-Fi channels, this allows us to accurately differentiate different types of Wi-Fi radios.

### 3.3 Countermeasures

One possible defense against our novel fingerprinting technique is to use a new random MAC address on each Wi-Fi channel being scanned. This makes it harder to measure the channel switch time. However, this would still allow an adversary to fingerprint physical properties of the Wi-Fi signal, and hence does not deter more powerful adversaries [4]. Instead, in the remainder of the paper, we improve the performance of passive Wi-Fi scanning, such that it becomes a viable alternative to active scanning.

## 4 Passive Scanning Improvements

In this section we present several techniques to improve the speed of passive scanning. We also define metrics to rigorously evaluate our proposed techniques.

### 4.1 Proposed Scanning Modifications

In this section, we limit ourselves to backwards-compatible modifications on the client. This makes it easier to deploy our proposals, since no network infrastructure needs to be modified. Additionally, this assures that all existing networks remain discoverable. More concretely we propose the following modifications and optimizations:

**Dwell time variation** We first evaluate the impact of the dwell time. The dwell time denotes the total time we listen on each channel for beacon frames. This parameter heavily influences the total duration of a passive scan.

**Incremental scanning** In our second modification, partial scanning results are returned to the Operating System (OS) while the scan is still in progress. We call this incremental scanning. More precisely, after scanning each channel, newly discovered APs are immediately returned to the OS. While reporting these results, the Wi-Fi chip continues scanning the remaining channels. The OS can abort the scan once a known network has been discovered.

**Static priority scanning** This modification is an improvement of the incremental scanning procedure, where we only return discovered APs to the OS after scanning several channels. This removes possible overhead caused by constantly communicating with the OS. At the same time, we prioritize certain channels and scan them first. In particular, we first scan the non-overlapping channels in the 2.4 GHz band (i.e. channel 1, 6, and 11) in a so-called priority scan. After scanning these priority channels, a second scan is performed over the remaining channels. We also examined the 5 GHz band, and observed that channels 36, 40, and 44 are used the most. Hence, in a second variation of static priority scanning, we also include these 5 GHz channels in the (initial) priority scan.

**Dynamic priority scanning** In our last modification, we dynamically determine the set of priority channels that are scanned first. The specific algorithm that is used to select these priority channels is out of scope for this paper. Instead, we refer to related work for algorithms that determine which networks are likely nearby. For example, the presence of nearby networks can be predicted based on one's location [20], the current day and time [22], and so on.

### 4.2 Metrics for evaluation

We now propose several metrics that we will use to quantify the performance of our scanning procedures:

**Scan duration** The scan duration is the total time it takes between listening on the first channel, and the moment collected results are returned to the OS. When using priority scanning, we will refer to the duration of scanning all channels as the full scan time, and the time it takes to scan only the priority channels as the priority scan duration.

**Time-to-connect** The time-to-connect metric measures the time between listening on the first channel, and the moment when the device discovers a known AP. This metric reflects the waiting time that users experience.

**AP discovery rate** We define the AP discovery rate as the amount of APs discovered relative to the total amount of APs discoverable by the device. This metric needs a reference of all available APs in the vicinity to compare a scan result to. In general, there will be a trade-off between discovery rate and scan duration. Similarly, a longer scan time likely results in a better AP discovery rate, but may negatively impact the time-to-connect metric.

**Specific AP discovery rate** This metric measures the rate at which a specific set of APs is discovered, relative to the total amount of scans performed. We will use this metric when a known network is nearby, to measure how frequently the network will be detected by various scanning procedures.

### 4.3 Experimental setup

We implemented our proposed modifications by modifying the user-space Wi-Fi client. In particular, we modified wpa_supplicant. One advantage of this approach is that every Linux and Android device can then be tested with our modifications.

In our specific setup, we implemented our modifications on Android 7.1.1 AOSP, and used a Google Nexus 5X. We changed the `gPassiveMinChannelTime` parameter (and the analogous `Max` parameter) of the Wi-Fi driver to control the dwell time. Fortunately, this did not require any changes to the driver code. Instead, we can modify the driver configuration file that contains these parameters using a user-space script[4].

Our incremental and priority scanning procedures are implemented by modifying wpa_supplicant.[5] This means our modifications can be tested on both Android and Linux, making our results easier to replicate. We implemented incremental scanning by issuing separate scan requests for every individual channel. Although this adds a overhead when communicating with the kernel to initiate each scan, part of this overhead is unavoidable. That is, even if we modified the driver or firmware, we still need to send individual notifications to wpa_supplicant after scanning each channel. When measuring the total scan duration of an incremental scan, we add all the individual scan durations, and exclude elapsed time between individual scans. Note that this measurement still includes the overhead caused by communicating with the kernel for initiating each scan. For priority scanning, we also modified wpa_supplicant to issue two individual scans. The first scan covers the priority channels, and the second scan covers all remaining channels.

## 5 Experimental Results

In this section we experimentally analyze the performance of our scanning strategies. This shows passive scanning can be a viable alternative to active scanning.

### 5.1 Dwell Time Variation

We first analyzed the impact of varying the dwell time when performing a standard passive scan. Interestingly, the total passive scan duration with a dwell time of 100 ms is only slightly longer than the scan duration of a default active scan. This is because most channels in the 5 GHz must be scanned passively, to avoid interfering with other devices such as weather radars [13, §11.1.4.1]. As a result, even when using default active scanning, these channels are still scanned passively. This implies that, when the 5 GHz band is also scanned, there is only a minor slowdown in scanning duration when switching from the default active scanning procedure to passive scanning.

We also measured the influence of the dwell time on the AP discovery rate. This was done at two locations. First at our office, where there were 30 discoverable APs (see Figure 2a). Then at a busy international train station where there were 419 discoverable APs (see Figure 2b). Note that in these figures, the AP discovery rate of a normal passive scan is identical to the discovery rate of a full priority scan. As expected, higher dwell times result in more APs being discovered. With a dwell time of 100 ms, passive scanning matches the discovery rate of default active scanning. For larger dwell times we observed only a slight

[4] This is the file /system/etc/firmware/wlan/qca_cld/WCNSS_qcom_cfg.ini

[5] Our code, including a build for the Nexus 5X, is available at `https://github.com/vanhoefm/nordsec-passivescan`.

increase in the discovery rate. With a dwell time lower than 100 ms, the discovery rate quickly drops, with the discovery rate at 50 ms being roughly half of active scanning. We conclude that the optimal dwell time for passive scanning must be at least 100 ms.

### 5.2 Incremental Scanning

We found that with incremental scanning, the scanning time is slightly higher compared to a standard passive scan. More precisely, the overhead of constantly issuing new scan requests for each channel, reporting the results back to the Operating System, and preparing a new scan, constitutes a 10% overhead. Due to this overhead, we do not consider incremental scanning as a good candidate to replace a default active scan, and will not consider it any further.

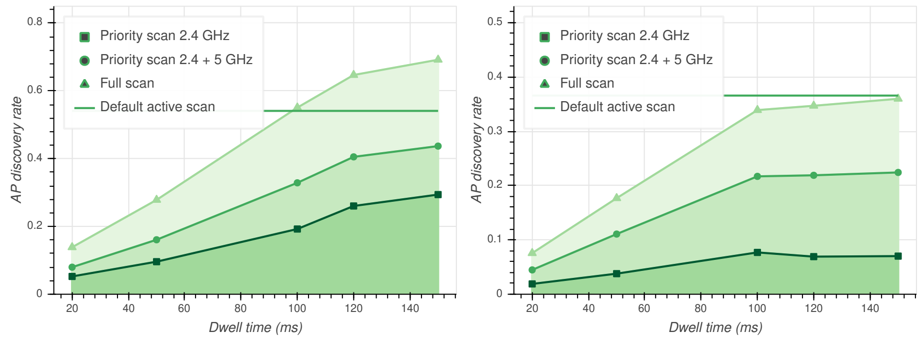
### 5.3 Static Priority Scanning

To evaluate static priority scanning, we first use the 2.4 GHz channels 1, 6, and 11 as static priority channels. Then we included the 5 GHz channels 36, 40, and 44 as well. We evaluated both the scan duration and AP discovery rate:

**Scan duration** We found that the ratio of the priority scan duration, compared to a full scan, approximates the ratio of the scanned channels in the priority scan to the total scanned channels. Note that the full scan duration is the same as a standard passive scan duration. For a small set of priority channels, the priority scan is significantly faster than a default scan. That is, when only including three 2.4 GHz channels, the priority scan takes only 8% of the time. When also including the three selected 5 GHz channels, a priority scan takes 15% of the time of a full scan.

**AP Discovery Rate** Figure 2a and 2b contain the AP discovery rate when using priority scanning at our office and the train station, respectively. The lower AP discovery rate around a 120 ms dwell time at the train station (Figure 2b) is because we had to slightly change our location during the experiment to accommodate passengers. Of interest is how many networks are discovered in a priority scan, compared to a full scan. With the 2.4 GHz priority channel set, 42% of networks are discovered at the office during the priority scan (compared to a full scan), while 22% were discovered at the train station during the priority scan. When also including the 5 GHz priority channels, 61% of networks are discovered at the office compared to a full scan, and 62% are discovered at the train station. This shows that including 5 GHz channels in the priority scan is essential to obtain a high AP discovery rate.

We conclude that, compared to a full passive scan, a priority scan takes 15% of the time, while already discovering close to two-thirds of networks. This shows that priority scanning is a very promising technique. Moreover, a priority scan takes only a fraction of the time, even when compared to the default active scan.



(a) Our office. (b) International train station.

Fig. 2: The AP discovery rate for various passive scanning procedures at (a) our office; and (b) an international train station. The horizontal line denotes the average discovery rate for a default active scan.

### 5.4 Dynamic Priority Channels

To test dynamic priority scanning, we assume the client has one known network, and that it treats the (previous) channel of this network as the single dynamic priority channel. We measure the discovery rate and time-to-connect:

**Specific AP Discovery Rate** For this experiment we used an environment with little interference and close proximity to the AP. We first tested the specific AP discovery rate for a default active scan, and found that the AP is always detected in the first scan (see Figure 3). For our passive implementation, we get comparable results. More precisely, with a dwell time of 100 ms, the discovery rate is around 90%. With a dwell time of 120 ms or higher, the AP is always detected. Based on these results, we conclude that passive scanning with a dwell time of 120 ms or higher matches the performance of active scanning. This again shows that passive scanning can form a practical alternative to active scanning.

**Time-to-connect** We investigate the time-to-connect for dynamic priority scanning, and compare it to the default active scanning procedure. For both procedures, we ran the time-to-connect experiment exactly 50 times. Figure 4a shows the resulting time-to-connect histogram for the default active scanning experiments. The average time-to-connect is between 3.8 and 3.9 seconds, indicating that the AP was always discovered during the first scan. We conjecture that the two separate groups around 3.8 and 3.9 seconds are caused by internal timing constraints in the Wi-Fi chip.

For passive scanning, the time-to-connect heavily depends on the dwell time. In particular, for a dwell time of 100 ms or lower, multiple scan are sometimes needed to discover the AP (see Figure 4b). This significantly increases the average time-to-connect, even though most of the time the network is discovered

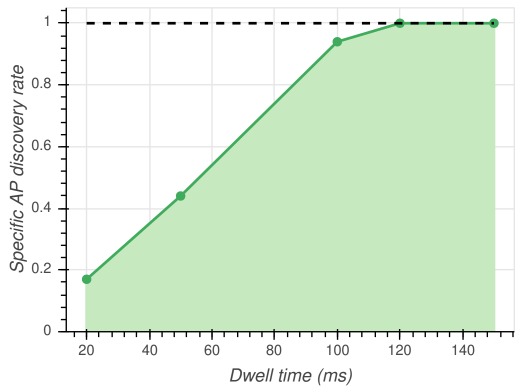
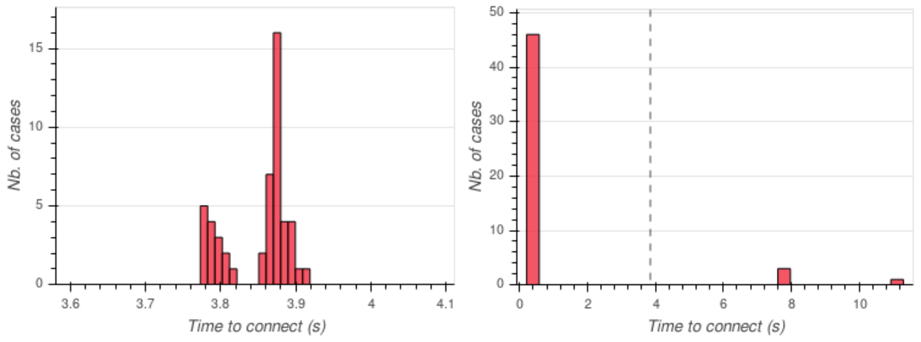


Fig. 3: Specific AP discovery rate for one known network when using passive dynamic priority scanning. The dashed horizontal line marks the average specific AP discovery rate for the default active scanning implementation.

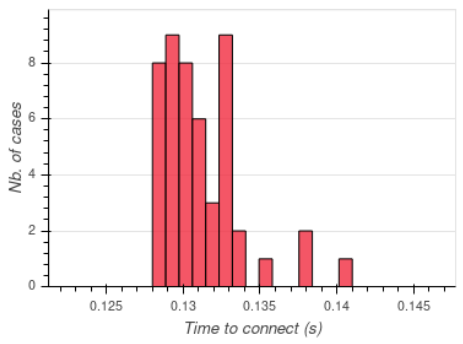


(a) Default active scan.

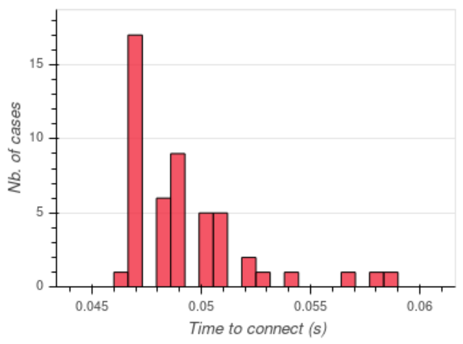
(b) Passive dynamic scan.

Fig. 4: Time-to-connect for (a) default active scanning; and (b) passive dynamic scanning with one priority channel and 100 ms dwelltime. The vertical line denotes the average time-to-connect for the default active scanning procedure.

in the first scan (see Table 1). However, when using a dwell time of 120 ms or higher, we always discover the AP during the first priority scan. Because this is a priority scan, results of the scan are returned nearly instantly, resulting in a very low average time-to-connect. Notice that with a 150 ms dwell time, the AP is again always discovered in the first priority scan. However, the higher dwell time results in a higher scan duration, and hence also in a higher time-to-connect. Finally, the average time-to-connect for active dynamic priority scanning is 0.049 seconds. Although this is faster than passive scanning, this difference is likely not noticeable to users, meaning passive scanning remains a viable alternative to active scanning.



(a) Passive dynamic scanning with one priority channel and 120 ms dwelltime.



(b) Active dynamic scanning with one priority channel.

Fig. 5: Time-to-connect for passive and active scanning.

Table 1: Time-to-Connect for passive dynamic priority scans

| Dwell time (ms) | 50 | 100 | 120 | 150 |
|---|---|---|---|---|
| Average TTC (s) | 5.62 | 0.91 | 0.13 | 0.16 |

We conclude that a dwell time of 120 ms minimizes the average time-to-connect. Moreover, compared to the default active scanning procedure, it results in a lower (i.e. faster) average time-to-connect, while being more privacy-friendly.

## 6 Advertising Neighboring Networks

In this section we propose a novel scanning technique, where APs include basic information of neighboring networks into their beacon frames. We show that providing this information drastically reduces the average scanning time.

### 6.1 Advertising Neighboring Networks

Many APs are capable of detecting nearby networks on different channels, while at the same time providing normal connectivity to clients. This is commonly used to let the AP automatically operate or switch to the least-used channel, and in practice this feature is often called Dynamic Channel Selection (DCS) or Dynamic Channel Assignment (DCA) [24,7]. More importantly for us, this means many APs are capable of scanning for nearby networks without interfering with normal operations.

To reduce the average scanning time of clients, we propose that every AP advertises all neighboring networks in its beacon frames. In practice, a simplified version of the Neighbor Report element can be used for this [13, §9.4.2.37]. In this element the SSID and channel of neighboring networks is included. In case there are many neighboring networks, and there is insufficient space to advertise them all, then only networks with a high signal strength can be included.

When a client is scanning for networks, it can use the neighbor reports inside beacons to optimize the scanning process. In particular, when a neighbor report contains a known SSID, then the client can immediately scan this channel to see if that network is also within range of itself. If so, the client can immediately connect to this network, which greatly reduces the average time-to-connect. In case the network is not with in range of the client, we continue with the normal scanning process.

### 6.2 Experiments and Results

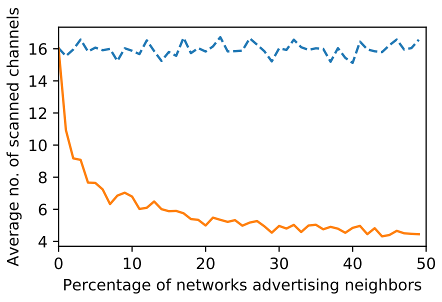
To estimate the benefits of including neighboring networks in beacon frames, we simulated this strategy and determined how much it reduced the average scanning time of a client. This allows us to determine how many APs must support this new feature for it to have a real benefit in practice. Note that in practice APs may not often get updated, meaning it is important that our new scanning strategy works well even if few APs advertise neighboring networks.

To determine the average scanning performance, we determine the number of channels that must be scanned before a known network is detected. In order to have realistic estimations of the number of nearby networks in our simulation, we use real-world Wi-Fi network locations from OpenWifi[6]. This is an open source database of Wi-Fi networks that is normally used for geolocation purposes. Based on this database, we perform the following steps in our simulation:

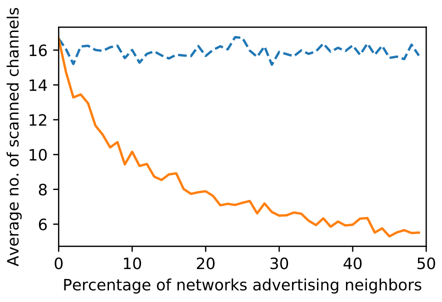
1. We assign a random operating channel to every network. Here we consider a total of 11 possible channels in the 2.4 GHz range, and 20 channels in the 5 GHz range.
2. A given percentage of networks is assumed to implement our proposal and advertise neighboring networks in their beacon frames.
3. We then randomly pick a position in a city, and determine all networks that are within 100 meters of this position. These networks are considered to be within range of the client. We further assume a known network is nearby, since otherwise the scanning time will always equal the duration of a full network scan.
4. Additionally, we assume a second known network is out of range of the client, but within range of one or more nearby APs. This means this network may appear in the neighbor lists of APs, but will not be detected when the client searches for this network on the advertised channel.

Figure 6 shows the resulting number of scanned channels needed to find a known nearby network, in function of how many APs include neighboring networks in their beacon frames. What is surprisingly is that even if only a minor number of APs advertise nearby networks, this already results in a major reduction in scanning time. For instance, in San Francisco our results show that

[6] `https://openwifi.su/download.php?lang=en`



(a) San Francisco, California.



(b) Boulder, Colorado.

Fig. 6: Average number of scanned channels before discovering a known network, when using normal scanning (dashed line) or when using neighbor advertising (solid line). Figure (a) contains the results for a densely populated area in San Francisco, while Figure (b) shows the results of a more sparse area in Boulder.

even if only 5% of APs advertise nearby network, then this already reduces the scanning time by more than half. To study the impact is less dense cities, we performed the same simulation in Boulder, and again found that if a minor number of APs advertise nearby networks, this already results in a major decrease of the scanning time. In particular, when only 10% of networks advertise its neighbors, then the average scanning time is almost halved.

Based on these experiments, we conclude that letting APs advertise nearby networks will result in a major decrease of the average scanning time. As a result, even when using passive scanning, known networks will be quickly found.

## 7 Related Work

Several researchers investigated the discovery process of Wi-Fi platforms and clients [10,6,1,12,18,9]. They conclude that most employ active scanning [10]. Unfortunately, capturing probe requests is trivial, and they provide a significant amount of information, ranging from family names, visited locations, travel routes, and so on [21,2,3].

To prevent tracking, modern devices use MAC address randomization when sending probe requests [23,17]. However, this defense can be (partly) bypassed. First, early implementations of address randomization did not reset the sequence number of probes between different individual scans [10]. Random MAC addresses can then be linked together by their incremental sequence numbers. Another method to link randomized probe requests is by relying on the included Information Elements (IEs). In particular, researchers found that the set of included IEs, their order, and their values, form a reliable fingerprint [23]. They suggest to avoid this by only including essential IEs. Finally, the timing between different network scans can also be used to track devices [18]. More precisely, the Inter-Frame Arrival Time (IFAT) between frames sent on the same channel within an individual scan forms a fingerprint of the device. This can be mitigated by sending probe requests with random delays [18]. In contrast, we look at the time between probe requests sent of *different* channels. Additionally, none of these works focus on improving passive scanning such that it becomes a viable alternative to active scanning.

Other works optimize active scanning by modifying the dwell time [6,1]. Their results indicate that in an environment with many APs, increasing the dwell time leads to a higher AP discovery rate [1]. However, increasing the dwell time past 100 ms did not yield more APs when using active scanning. Kim et al. propose to use the current location of a device to reduce privacy leaks during active scanning [15]. Here a device remembers the location of known networks, and only sends probe requests to networks that are likely nearby. Again, none of these works investigate passive scanning.

## 8 Conclusion

In this paper we argued that improving active scanning, such that it is more privacy-friendly, is not the best way forward. Instead, we believe a better approach is to improve the speed of passive scanning. We hope this makes vendors more incentivized to use passive scanning as the new default.

In particular, we proposed incremental and priority scanning. We implemented these modifications on Android, and evaluated their performance using several metrics. These experiments indicate that passive scanning can match and even surpass the speed of active scanning. For instance, when using static or dynamic priority scanning, the average time-to-connect is lower than default active scanning. This makes passive scanning a practical alternative to active scanning, improving privacy, and freeing up airtime.

## Acknowledgments

Gunes Acar and Mathy Vanhoef hold a Postdoctoral fellowship from the Research Foundation Flanders (FWO). This work is partially supported by the Research Fund KU Leuven and by the Center for Cyber Security at New York University Abu Dhabi (NYUAD).

## References

1. Arcia-Moret, A., Molina, L., Montavont, N., Castignani, G., Blanc, A.: Access point discovery in 802.11 networks. In: IFIP WD (2014)
2. Barbera, M.V., Epasto, A., Mei, A., Perta, V.C., Stefa, J.: Signals from the crowd: uncovering social relationships through smartphone probes. In: IMC (2013)
3. Bonne, B., Barzan, A., Quax, P., Lamotte, W.: WiFiPi: Involuntary tracking of visitors at mass events. In: WoWMoM Workshop (2013)

4. Brik, V., Banerjee, S., Gruteser, M., Oh, S.: Wireless device identification with radiometric signatures. In: MobiCom (2008)
5. Campbell-Dollaghan, K.: Brave new garbage: London's trash cans track you using your smartphone (2013)
6. Castignani, G., Arcia, A., Montavont, N.: A study of the discovery process in 802.11 networks. ACM Mobile Computing and Communications Review **15**(1) (2011)
7. Cisco: Dynamic channel assignment (dca). Last retrieved 1 August 2019 from `www.cisco.com/c/en/us/td/docs/wireless/controller/technotes/8-1/mobility_express/b_RRM_White_Paper/b_RRM_White_Paper_chapter_0100.pdf`
8. Cisco Systems: Channel deployment issues for 2.4-GHz 802.11 WLANs. Retrieved 16 July 2018 from `xenguard.com/library/wifi/wifi-channels.pdf` (2004)
9. Franklin, J., McCoy, D., Tabriz, P., Neagoe, V., Randwyk, J.V., Sicker, D.: Passive data link layer 802.11 wireless device driver fingerprinting. In: USENIX Sec (2006)
10. Freudiger, J.: How talkative is your mobile device? an experimental study of Wi-Fi probe requests. In: WiSec (2015)
11. Greenstein, B., McCoy, D., Pang, J., Kohno, T., Seshan, S., Wetherall, D.: Improving wireless privacy with an identifier-free link layer protocol. In: MobiSys (2008)
12. Gupta, V., Beyah, R., Corbett, C.: A characterization of wireless NIC active scanning algorithms. In: WCNC (2007)
13. IEEE Std 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Spec (2016)
14. Khoury, P.: Multiple BSSID support. In: IEEE 802.11-16/0586r1 (2016)
15. Kim, Y.S., Tian, Y., Nguyen, L.T., Tague, P.: LAPWiN: Location-aided probing for protecting user privacy in Wi-Fi networks. In: CNS (2014)
16. Lindqvist, J., Aura, T., Danezis, G., Koponen, T., Myllyniemi, A., Mäki, J., Roe, M.: Privacy-preserving 802.11 access-point discovery. In: WiSec (2009)
17. 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), 365–383 (2017)
18. Matte, C., Cunche, M., Franck, R., Vanhoef, M.: Defeating MAC address randomization through timing attacks. In: WiSec (Jul 2016)
19. Microsoft: Non-broadcast wireless SSIDs: Why hidden wireless networks are a bad idea. Retrieved 16 July 2018 from `blogs.technet.microsoft.com` (2008)
20. Nicholson, A.J., Noble, B.D.: Breadcrumbs: Forecasting mobile connectivity. In: MobiCom (2008)
21. Pang, J., Greenstein, B., Gummadi, R., Seshan, S., Wetherall, D.: 802.11 user fingerprinting. In: MobiCom (2007)
22. Peddemors, A., Eertink, H., Niemegeers, I.: Predicting mobility events on personal devices. Pervasive and Mobile Computing (2010)
23. 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 (2016)
24. ZyXel: Dynamic channel selection (dcs). Last retrieved 1 August 2019 from `https://www.zyxel.com/uploads/Dynamic_Channel_Selection_4.20.pdf`