Mobile Phone's Wi-Fi Presence for Continuous Implicit Secondary Deauthentication

Adrian Dabrowski and Edgar R. Weippl

SBA Research, Vienna, Austria

Abstract. Authentication is widely considered a core challenge in usable security. We propose an implicit method for secondary (de-)authentication based on device presence. It is suited for scenarios where user inactivity timeouts do not automatically translate well into actual user absence. Mobile phones seems like a natural choice for a secondary token as they are *quasi-nothing-to-carry* (UDS framwork) and personal enough that users are unlikely to share it. Our method does not require system modifications on the users' side, e.g., by installing an additional app, and is therefore cheap and easy to deploy. Having removed the burden of developing and maintaining apps in the fast proceeding field of mobile phone operating systems is a great advantage. Furthermore, our method solves the problem of accurate de-authentication as it automatically logs out the user if the device is not in reach of the Wi-Fi network.
Our method works by monitoring Wi-Fi activity of users' phones regardless if connected to a Wi-Fi network or not. We facilitate passive monitoring on the radio layer (e.g., probing requests and power save mode keep-alive management packets) and the network layer (DHCP and ARP snooping). Additionally, we employ active probing using ARP ping and IPv6 Link Layer Neighbor Discovery. We describe how to implement a monitoring and probing device in a constant time and space manner (e.g., by using a Bloom filter ring) optimal for embedded systems.

1 Introduction

In an 2014 study, Mary Meeker and Liang Wu [21] found that average smartphone users check their phone 150 times a day. Since many find it hard to be without their device, it seems like a good option to include it into authentication procedures. However, a classic app solution requires a new implementation for each mobile phone platform. While an Android and an iOS app solution together would cover 93% of the installed devices worldwide [24], it would nevertheless leave out 7% of the users. The long tail of mobile operating systems and the ever changing landscape are a substantial hurdle in finding an cross-platform solution.

In this paper, we propose the use of mobile phone basic services for a weaker but nevertheless useful secondary and continuous (de-)authentication system. Thus, no app development is needed for every single platform. We evaluate

the usefulness of several wireless protocols supported by modern smartphones. Furthermore, we show how to implement the system in a resource-sustainable way (i.e., constant space and constant time lookups) for embedded systems. Such systems can easily complement and support conventional authentication methods.

By facilitating mobile phones without the need to install additional software, we meet $7\frac{1}{2}$ out of 8 usability criteria in Bonneau's et al. UDS framework [7]. Furthermore, it solves the de-authentication problem that many other schemes (e.g., PIN, passwords) cannot address properly. Other existing de-authentication schemes (e.g., ZEBRA [20]) often solve this problem heuristically, which inherently comes with a non-negligible error rate. It is this error rate that can lead to severe distractions and interruptions for the user [17]. Our system returns deterministic results by passively monitoring the (background) activity of a mobile phone and interrogate actively if necessary.

Our system is especially useful in the following circumstances:

- Users circumvent or share their primary authentication system (e.g., PIN) or token (e.g., RFID, iButton), but a more severe authentication method (e.g., biometrics) is out of scope.

  Example: *A laboratory or workshop offers access to demanded machines only for selected students. However, students share their (RFID) access cards with others.*

- Users (on site) regularly forget to log out or otherwise disappear during a session.

  Example: *Some laboratory machines (as the example above) can only be used according to fixed schedule, after registration, or by passing an introductory lecture. Users log in using their primary authentication method (e.g., iButton, RFID token, PIN, password) and need to stay close to the machine during operation for security reasons. However, users sometimes either forget to log out or simply swap/trade scheduled slots. Unlike desktop computers, idle time does not automatically translate into an unused machine as loading, unloading, and adjusting can take a significant amount of time (Figure 1).*

- A cheap and universal secondary (de-)authentication method is needed.

The remainder of this paper is structured as follows:

Section 2 gives an overview over different transmission systems available in smartphones and compares their key properties. Furthermore, we describe the Wi-Fi stack and its different behaviors. We dive into detail about our Wi-Fi implementation in Section 4 and outline our software in Section 5. Sections 6, 7, and 8 describe our operative, administrative, and usability considerations respectively. Discussion in Section 9, Related Work in Section 10, and Conclusion in Section 11 finalize the paper.

Fig. 1. A possible scenario: The current authorization for the laser cutter at Metalab Vienna, Austria uses iButtons against a central authorization database and a display. Wi-Fi presence information can simply be displayed to the screen if necessary.

2 Technical Background and Comparison of Wireless Systems

Today, mobile phones and especially smartphones with 2G, 3G or 4G transmitters come with additional Wi-Fi, Bluetooth or NFC transmitting technologies. All of them feature unique properties and challenges but they have in common that their basic operation does not need any special third party app.

The main mobile network transmitters for 2G, 3G, and 4G are out of scope for two reasons: First, they operate in a licensed band so that any active transmission violates radio regulations. Second, all of them come with some degree of anti-tracking mechanisms based on temporary identities (e.g., TMSI).

Additionally, ultrasonic sound transmissions [1, 4, 12, 13, 16, 19] would also offer a good method for vicinity presence checks, but this needs certain apps installed and running on the devices.

In the reminder of this section, we will assess each technology based on the following key properties and explain their technical key points (Table 1):

Availability The universality of a system heavily depends on the availability of the used transmission technology. For example, as of November 2015, only two out of 47 phones sold on AT&T web shop came without Wi-Fi and all supported Bluetooth of some kind. The two phones without Wi-Fi were feature phones.

Prevalence How many users regularly turn on the service, or leave it on.

Range This is a very important property. It defines the radius in which a person (with their phone) has to be present. In our examples (c.f. Introduction) the range must be neither too small nor too big.

Blind Scan or Enumeration This describes the ability of enumerating devices without prior knowledge for which device one is looking for.

Presence Check This describes the ability of finding a specific device whose identity is already known (i.e., verifying its presence).

2.1 Near Field Communication (NFC)

NFC has the smallest range of all the transmitting technologies presented here. Due to its inductive working principle the range is typically limited to a single digit centimeter value. Without a special app installed on the smartphone, an NFC reader can read only the Unique ID (UID). Typically, this value has a length between 32 and 128 bits. However, on phones this value is often pseudo-random.

As for the range, this technology would demand the phone to be lying on a reader the whole time. Scan and presence times can be well below 50 ms.

The availability of NFC increased in recent years, as it is featured by numerous companies as a payment service. This led to the inclusion of NFC first in Android phones and later also in iPhones. However, even when enabled, the NFC sensor is usually only active when the screen is turned on. Only very few devices come with a secure element that can be powered passively and independently.

2.2 Bluetooth and Bluetooth LE

Bluetooth comes in several different variants and flavors. The most recent addition to the family of Bluetooth standards is Bluetooth Low Energy (Bluetooth LE or BLE) which allows devices to operate from a single battery for months. Both Bluetooth variants come with a pairing option, i.e., by default, devices can but do not have to authenticate each other via a PIN. However, common mobile operation systems require pairing for data transmission.

As for range, there are multiple classes defined in the standard. Typically, mobile phones are class 2 devices which offer a maximum range of about 10 meters.

A blind scan (discovery) on Bluetooth is a complex stochastic process. As all devices use a frequency hopping scheme, they have to meet by chance while constantly switching between sending and receiving. Additionally, interference, collisions (by multiple devices), and power save sleeping force the scanning device to repeat the scan several times. One scan cycle takes between 1.28 and 2.56

Table 1. Key points about wireless technologies on smartphones

	RFID/NFC	**Bluetooth**	**Wi-Fi**
Availability	7% [2,3]†	≈100%	≈100%
Prevalence*		33% [25]	89% - 98% [26]
Range (indoors)	<10cm	≈10m‡	≈25m
Enummeration	<150ms	2-12 sec	<2s (for /24)
Presence Check	<50ms	2-12 sec	<20ms

* ratio of users having this service enabled most of the time
† based on the numbers of 2014
‡ smartphones are typically class 2 devices

seconds and repeated up to 48 times (61 seconds), depending on the Bluetooth stack implementation. A presence check is slightly less complex, but still a time-costly process compared to other technologies.

On average, only 33% of iPhone users have Bluetooth turned on, as found by a study on 50,000 users [25]. Turning off Bluetooth used to be one of the top points on many *how to conserve phone power* advisory list.

2.3 Wi-Fi

As stated above, Wi-Fi is available on virtually all smartphones and many feature phones sold today. A 2012 study [26] estimates regular Wi-Fi usage of iPhone and Android users at 98% and 89% respectively.

We will differentiate between Wi-Fi Probe Requests (in unconnected state) from Wi-Fi Network Presence (in a connected/associated state). The client reacts very differently in both states. However, both states offer unique possibilities for our use case (Table 2).

Wi-Fi Probe Requests A client can enumerate all networks completely passively, i.e., by just listening to the beacon packets each access point sends out regularly (typically every 33 to 100 ms). To enumerate all access points, a phone has to tune into each Wi-Fi channel and listen for a particular time. This is considered too time-intensive by phone manufacturers and therefore rarely used anymore. Additionally, this method will not find associated hidden networks.

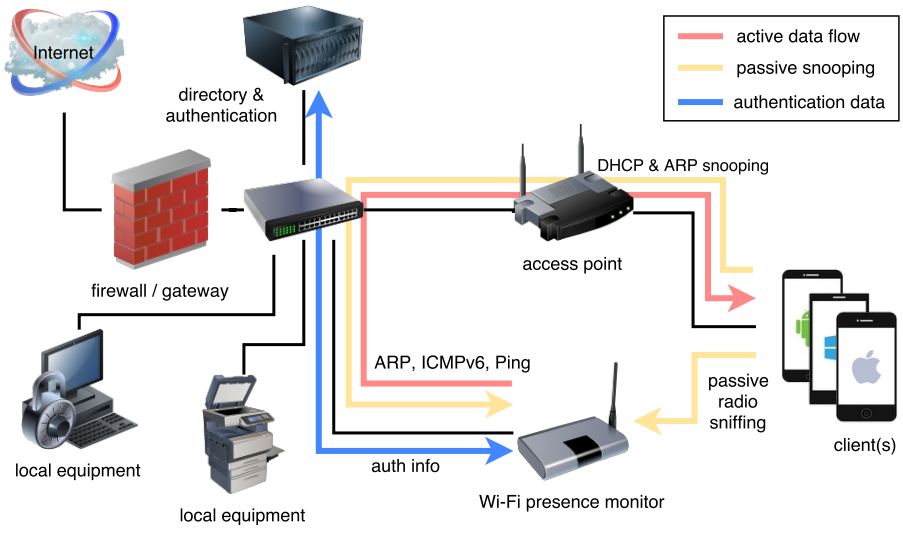

Fig. 2. Simplified setup of a lab LAN with a Wi-Fi; visualization of data collection and traffic flows of the Wi-Fi monitor device.

Instead, mobile phones send out active *Probe Requests* on each channel to speed up discovery. Every Access Point will answer with its public information (e.g., BSSID, supported standards, encryption parameters). Probe requests are also the only method to find (and subsequently to connect) to hidden networks. These networks do not send out the network name in their beacon signal.

Probe Requests contain the 48 bit MAC address that uniquely identifies a phone. Probing intervals and behavior strongly depends on the hardware vendor and the operating system version [11].

Wi-Fi Network Presence Once a phone is connected to a Wi-Fi Network it will stop sending probe requests in *most* cases [11]. Instead, since it is associated to a network, it will answer regular network layer requests. This means that some broadcasts and particular low level network messages (such as a ARP requests) will always trigger a response by the phone. This can be used for a presence check which produces results well within 20 ms, even under bad conditions.

On IPv4, ARP requests are a quick way to do a presence check [18]; however, the IPv4 address has to be known in advance, e.g., sniffed from the DHCP request or read out from the DHCP database. An enumeration of all devices on a typical /24 network using arpscan [14] takes less than two seconds. ICMP Echo (*ping*) is not recommended, as an increasing number of devices are ignoring such requests.

On IPv6, the Wi-Fi MAC address can be directly used to construct link-local addresses in the `fe80::/10` range [15, Appendix A]. This allows to send packets without prior knowledge of other (global) addresses.

However, in many cases an idle device also gives away information about its presence on the radio layer. In *Constantly Awake Mode* (CAM) a device will constantly listen to the access point for any packets that might be addressed to itself. This provides the best performance, but also consumes the most power. There are multiple *power saving modes* (PSM) for IEEE 802.11/a/b/g/n networks which can be negotiated between client and access point. They are all

Table 2. Wi-Fi Activity and Presence Collection Methods

Wi-Fi State	**Mode**	**Radio Layer**	**Data Link Layer**	**Network Layer**
Disconnected	Passive	Probe Requests		
Connected	Passive	Probe Requests Power Save Polling (Encrypted) Traffic	ARP Snooping	DHCP Snooping
	Active		IPv4 ARP[†] IPv6 Neighbor Sol.[‡]	ICMP Echo[†‡]

[†] via IPv4 address from DHCP snooping
[‡] via link-local address

based on two basic principles respectively a mixture of them: (I) The access point buffers packets for the client which actively polls them in regular intervals. Thus, the client can sleep in between. (II) The client only wakes up at specific intervals and listens for packets. The access points delay incoming messages until the right time slot. Still, the stations have to actively reassure the access point in regular intervals of their presence.

Both methods leave steady proof of the clients existence on the radio layer. Additionally, network layer broadcasts (e.g., ARP) tend to regularly wake up all wireless clients. Wi-Fi is rarely disabled by users as it provides faster service than mobile data and does not count towards the data budget [26].

3 Usage Scenario

Our usage scenario covers situations, where GUI idle time does not translate well into user inactivity, e.g., in a laboratory or workshop where loading and adjusting a machinery or testing of a work piece takes a significant amount of time aside the computer screen. These computers tend to have higher auto-lock and auto-logoff timeouts than on a typical workstation. Additionally, the handling of material and work pieces distracts users from their duty to log-out when leaving.

The larger than usual auto-lock or auto-logout interval eases the attacker in taking control over the computer and/or machinery of a careless user.

Additionally, machines such as laser cutters, CNC millers, or 3D printers are of great demand at certain institutions, but also hold the potential to great disaster when not operated properly or without supervision (e.g., fire). The great popularity and demand amplify the incentives for unethical user behavior, including account sharing, taking over someone else shift, or swapping or selling of machine time (e.g., from a subscription).

4 Wi-Fi implementation

Data collection on the presence monitor device is performed by operating a Wi-Fi radio in monitor mode (and optionally a wired connection). For Linux, many Atheros based radios support this out of the box. For monitoring probe requests, the tuned channel is irrelevant, as clients send them out on all channels. For monitoring power save mode traffic, the frequency has to be tuned to the same channel as the access point. Additionally, if the presence monitor is implemented on Linux the *control* flag has to be set, to receive control traffic as well[1].

[1] On Linux:
```
iw dev wlan1 interface add moni0 type monitor flags control
iw dev moni0 set channel ...
```

4.1 Passive data collection

For both, generic Wi-Fi and LAN data collection only the source MAC address is of importance. This is readable, regardless of the encryption used on the wireless network. It is hashed and stored into a bloom filter (see Section 5).

DHCP and ARP snooping additionally is stored in the MAC/IP lookup table. As a local network offers only a finite number of IPv4 addresses (e.g., a /24 has 256 IPs) the table is fixed size. For larger networks (e.g., a /16) a hash table can be considered. In this case, an (optional) expiry should either follow the lifetime of the bloom filter data or the lifetime of the DHCP leases.

ARP broadcast requests are mostly generated by the clients when trying to resolve the link-local address of the default gateway. They carry the clients correct MAC/IP-tupple as source addresses. DHCP gives away the MAC/IP-tupple during the final phase when performing a duplicate address test.

4.2 Active data collection

The active data collection is primarily used for devices that are actually connected to a local Wi-Fi. It can be done useing a wired connection (as shown in Figure 2 or using a second (or virtual) Wi-Fi interface.

ICMP Echo (also known as *ping*) is the most common way to test the presence of a device. However, recently more and more client devices stopped responding to *ICMP Echo* requests (e.g., modern Windows machines). In general, support for ICMP Echo is optional.

However, devices can not suppress answering to the link-layer address lookup systems such as the Address Resolution Protocol (ARP) [23] and ICMPv6 Neighbor Solicitation [22]. While only available on the local network they are more reliable than ICMP Echo.

ARP requests send a broadcast to the network asking for the MAC address of a specific IP. Since the authorization system already knows the MAC address it needs to translate them to an IP to perform this test. Thus, it is necessary to keep a local IPv4/MAC translation table (see Section 4.1).

In IPv6 every interface is assigned a static link-local address constructed deterministically based on the interfaces' link-layer MAC address. These addresses start with `FE80::` and contain the full MAC address together with some filling bits and a special mask. The process is standardized in RFC-4291, Appendix A [15]. Additionally, IPv6 offers deterministic auto-configuration of global IPs based on the MAC address in networks without DHCPv6. Thus, devices can be directly contacted if the MAC address is known. As a replacement to ARP, IPv6 provides a link-layer address resolution protocol known as ICMPv6 Neighbor Solicitation [22]. All devices must respond to these requests. For our purposes, this is the most reliable way to check for a device's presence if it has IPv6 support.

5 A constant time and space implementation

In general, for embedded systems the limited memory is an important issue. Bloom filters [6] meet the criteria by offering a fixed size and constant time

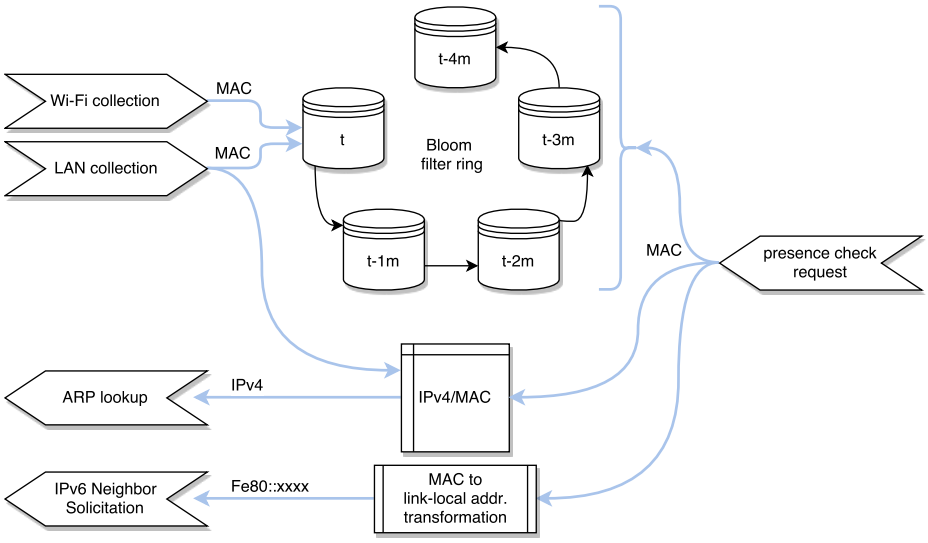

Fig. 3. Visualized data flow ans storage within the monitor and collection unit

characteristics regardless of how many items are inserted. Additionally, Bloom filters also guarantee no false negatives. However, they lack the ability to remove entries (e.g., expiry).

To combat this, we constructed a round ribbon ring of bloom filters (Figure 3). Every time unit t the bloom filters are rotated. Thus, the oldest bloom filter is reset and used for all new incoming data. The retrieval procedure queries all n bloom filters, whereas the computation of the hash only has to occur once. By rotating the filters, all entries expire after $n * t$ time. The time is a trade-off between the auto-logout interval and the automatic data transmissions on the radio network: i.e., smaller values will trigger active device interrogation more often, while longer time intervals tend to capture more of the phone's automatic transmission. For Figure 3 we chose $t = 60s$ and $n = 5$.

The bloom filters hash the incoming data and set their fixed size bit fields accordingly. As bits can overlap between different entries, false positives are possible, but false negatives are not. Broder et al. [8] showed that with just 16 bits of space per expected element, a good implementation has a false positive probability of less than 0.05%. This allows for keeping the bloom filters entirely in RAM.

The characteristics of bloom filters also provide a *denial of service* (DOS) protection against flooding attacks. This *fail open* behavior is preferred for most secondary continued authentication scenarios.

Bloom filters also provide a very privacy-friendly implementation. Although a lot of information about nearby devices could be potentially collected, the construction of the Bloom filter ensures that only information about previously known devices can be read from the collection unit.

As stated above, the IPv4/MAC lookup table is of fixed size as well.

6 Usability Considerations

Our system includes design ideas from Khan et al. [17] and Bonneau et al. [7], and we meet most of the usability criteria in the UDS framework. The system is *memorywise effortless* (U1) and *physically effortless* (U4) as the user does not have to remember or actively do anything. It scales well for a large number of users (U2), as there are no changes for the single user, regardless of how many are known to the system. It is *quasi-nothing-to-carry* (U3) as it employs the omnipresent mobile phone and users do not have to carry any additional device[2]. It is *easy to learn* (U5) and *efficient to use* (U6) as no interaction is required by the user and it produces *infrequent errors* (U7). *Recovery from loss* (U8) is described in Section 8.2 and can be done by the user. Only under certain conditions (i.e., phone lost multiple times in short period of time) this might include a visit to the administrator in person.

7 Operative Considerations

By using an integrated feature of all modern smartphones, no additional client application has to be developed. This is an important benefit in the rapidly changing smartphone landscape.

Probe requests are used by all major smartphone operating systems for Wi-Fi discovery [11]. However, our system is not dependent on that. iOS 8 introduced random source MAC probe requests whenever the screen is turned off. Freudiger et al. [11] nevertheless described how to de-anonymize them. Where de-anonymization is not possible or the mobile phone uses an entirely passive Wi-Fi discovery, the only change for the user is to enable auto-connecting to the lab's Wi-Fi. Most users will do this anyway for faster access speed and to save data traffic from the monthly plan [26].

We put the bloom filter into the collection unit for two reasons: First, it saves traffic between the collection unit and the directory server (e.g., the directory and authentication server can be external). Second, the proposed method has low and constant space properties, whereas the collected stream of MAC addresses can be massive and may contain information on other devices (e.g., bystanders) as well.

The authentication server or the machine on which the user is logged-in polls the existence of users regularly to verify their status (e.g., once a minute, usually not more often than t) by asking the collection unit for any evidence that the user's device might have left. The collection device can then answer whether it has seen the device in the last $n * t$ time frame. Under normal conditions, most

[2] UDS describes this case specifically as *almost* offering the specific benefit. Therefore we count it as $\frac{1}{2}$-achievement. The full *nothing-to-carry* is for authentication schemes that do not require any physical artifact.

devices will have introduced network traffic on their own (Wi-Fi probe requests, Wi-Fi power management frames, or an active Internet communication such as polling mail servers or instant messenger servers).

Active probing by the collection unit is only needed when the device requested is not in the local database (i.e., had no Internet or radio activity in the collection period). Additionally, the collection unit can initiate an active probe when the device requested by the authentication server is about to expire (e.g., in the last bucket of the Bloom filter ring).

The return packets of active probing (such as IPv4 ARP requests or IPv6 ICMP Neighbor Solicitation) do not need to be handled on protocol level. Their responses will be caught by the radio or network monitoring anyway and serve as an evidence for the existence of that device.

The encryption of the Wi-Fi does not need special handling. The radio layer data acquisition is operating on the 802.11 headers which are unencrypted. ARP, DHCP, and active collection is conducted via the wired interface and therefore bypasses the encryption.

Special care must be taken on networks which support multiple frequency bands (e.g., 2.4 Ghz as well as 5 Ghz). They are effectively two wireless networks sharing the same network-level broadcast domain. Multi-band clients send probe requests on both bands, however other packets (such as power safe polling or user data) are transmitted only on one of them. In this case, the collection unit should be equipped with one radio for each band.

8 Administrative Considerations

Our scheme requires to save the phone's MAC address in the user record (on the authentication or directory server). There are two main procedures to consider: (I) Initial registration of the users, and (II) update in case of phone replacement.

Since this is a secondary authentication scheme, we assume that other functions such as expiry are already take care of.

8.1 Initial Registration

All major mobile phone operating systems offer a built-in way to display the wireless MAC address to the end user without third party-apps. On Android, this is in the *Settings* > *About Phone* > *Status* menu, on iOS in *Settings* > *General* > *About*, and on Windows Mobile in *Settings* > *About* > *More Info*.

However, there are more user-friendly ways than spelling out a 12-digit hexadecimal number. The user can be presented with a personalized one-time URL (e.g., in the form of an QR code) which she/he has to access with the phone while connected to the wireless network. This allows the directory server to record the MAC address based on the Source-IP of the request (via a local ARP table lookup).

Those steps can be integrated into the initial enrollment process to the facilities, which typically has to be done in-person and manually anyway.

8.2 Phone Replacement

In the US in 2014, 49% of consumers had replaced their phones after a year [10]. Thus, for a larger user base, a manual update process does not scale well. Where possible, we suggest a self-service solution. A user can use his/her credentials to log on to a website and update the MAC address. However, to prevent misuse this should include a moratorium period (e.g., users can only change their MAC by themselves once in several months) and a second authentication method (e.g., e-mail or SMS token). Users who do not meet this requirements (e.g., changed their phone more often) need to show up in person and be updated manually.

9 Discussion and Limitations

Our method is based on the hardware address (MAC) of the involved devices. MAC addresses are not very secure (e.g., a jail-broken or rooted device can spoof another MAC). Therefore, we propose the method only as secondary (supplementary), implicit, and continuous (de-)authentication method. It must not be used without a strong primary authentication method. In situations where user inactivity is not an appropriate indicator for an automatic logout (e.g., the above mentioned lab environment) this method can supplement other systems.

To increase the authentication strength one could include a capability fingerprint based on the Wi-Fi probe requests. They contain information such as the supported frequencies, speeds, standards, or antenna configurations, e.g., for multipath transmission.

We do not know, if the randomization of source MAC for probe requests will serve as precedent. In Apple's implementation the randomization is only active when the phone is not connected and the screen is turned off. The same random value is used until the screen is turned on again. Then the phone uses its real MAC address again. Everything else (incl. the sequence number) persists, which makes it nonetheless trackable [11]. Future implementations might change this.

Since randomization is turned off when connected, these phones can easily participate in the scheme under the condition that they associate with the Wi-Fi in questions. Phones usually do this automatically when the Wi-Fi network has been used before.

10 Related Work

Khan et al. [17] did empirical research on the convenience of implicit authentication methods. They pointed out that in-transparency and false rejection rates can introduce severe interruptions in the work flow. In their 2015 article, De Luca and Lindqvist [9] also stressed the priority of avoiding a negative impact on the user experience.

The usability-deployability-security (UDS) evaluation framework by Bonneau et al. provides an systematic overview and comparability between different authentication methods [7].

Mare et al. [20] also presented an automated authentication mechanism that focused on detecting and de-authenticating departing users. However, they need to distribute additional tokens to each user.

Warsaw Hackerspace uses a Wi-Fi-based system [5] to display (nick) names of members currently in the lab. This serves as an information for others weather the lab is open and if a friend of theirs is currently there.

11 Conclusion

We present an easy-to-use and lightweight system to complement a primary authentication mechanism. It works without additional hardware or software on the user's side by facilitating standard (smart) phone Wi-Fi behavior as a *quasi-nothing-to-carry* device. Even when not associated with a network, a Wi-Fi-enabled phone sends out identifiable signals that allow for simple and reliable tracking of a user. This information can be used to strengthen the primary authentication by including a second factor and to timely detect gone users. This is especially useful in environments where idle time on the console does not actually translate into a user doing nothing (such as supervising a machine, loading, unloading, adjusting, or simply monitoring).

In detail, we examined multiple wireless transmission services typically included in smartphones and assessed their suitability for presence detection. We chose Wi-Fi, even though it behaves completely differently in the associated and unassociated state. However, both states produce enough evidence of a specific phone to detect its sudden absence. In the unassociated state regular probe requests reveal the identity of a phone. In the associated state the regularly sent-out power-save-keep-alive messages and the actual traffic are used as passively collected indicators. Additionally, we actively probe using IPv4 Address Resolution Protocol (ARP) and IPv6 Neighbor Solicitation if necessary.

We designed an embedded wireless collection unit that uses a cyclic ring out of Bloom filters to construct a scalable, privacy-friendly, and self-expiring lookup database with constant space and time properties.

Our scheme fulfills $7\frac{1}{2}$ out of 8 possible usability benefits in the UDS evaluation framework by Bonneau et al. [7]. We also took ideas from Khan et al. [17] to not distract the user or introduce interruptions.

References

1. chirp.io (Lets teach the machines to sing), http://chirp.io/, accessed Dec 9th 2015
2. Forecast installed base of NFC-enabled phones worldwide from 2013 to 2018, http://www.statista.com/statistics/347315/nfc-enabled-phone-installed-base/, accessed Dec 5th 2015
3. Mobile phone users worldwide from 2013 to 2019, http://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/, accessed Dec 9th 2015
4. Nearby - Google Developers, https://developers.google.com/nearby/, accessed Dec 9th 2015

5. Now at Hackerspace, https://at.hackerspace.pl/
6. Bloom, B.H.: Space/Time Trade-offs in Hash Coding with Allowable Errors. Commun. ACM 13(7), 422–426 (Jul 1970), http://doi.acm.org/10.1145/362686.362692
7. Bonneau, J., Herley, C., Oorschot, P.C.v., Stajano, F.: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy. pp. 553–567. SP '12, IEEE Computer Society, Washington, DC, USA (2012), http://dx.doi.org/10.1109/SP.2012.44
8. Broder, A., Mitzenmacher, M.: Network applications of bloom filters: A survey. Internet mathematics 1(4), 485–509 (2004)
9. De Luca, A., Lindqvist, J.: Is Secure and Usable Smartphone Authentication Asking Too Much? Computer 48(5), 64–68 (2015)
10. Entner, R.: 2014 US Mobile Phone sales fall by 15cycle lengthens to historic high (Feb 2015), http://reconanalytics.com/2015/02/2014-us-mobile-phone-sales-fall-by-15-and-handset-replacement-cycle-lengthens-to-historic-high/, accessed Dec 11th 2015
11. Freudiger, J.: How Talkative is Your Mobile Device?: An Experimental Study of Wi-Fi Probe Requests. In: Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks. pp. 8:1–8:6. WiSec '15, ACM, New York, NY, USA (2015), http://doi.acm.org/10.1145/2766498.2766517
12. Goodin, D.: Meet "badBIOS" the mysterious Mac and PC malware that jumps airgaps. Ars Technica (2013)
13. Hanspach, M., Goetz, M.: On covert acoustical mesh networks in air. arXiv preprint arXiv:1406.1213 (2014)
14. Hills, R.: ARP scanning and fingerprinting tool, http://www.nta-monitor.com/tools-resources/security-tools/arp-scan, accessed Dec 10th 2015
15. Hinden, R., Deering, S.: IP Version 6 Addressing Architecture. RFC 4291 (Draft Standard) (Feb 2006), http://www.ietf.org/rfc/rfc4291.txt
16. Karapanos, N., Marforio, C., Soriente, C., Capkun, S.: Sound-proof: Usable two-factor authentication based on ambient sound. In: 24th USENIX Security Symposium (USENIX Security 15). pp. 483–498. USENIX Association, Washington, D.C. (Aug 2015)
17. Khan, H., Hengartner, U., Vogel, D.: Usability and Security Perceptions of Implicit Authentication: Convenient, Secure, Sometimes Annoying. In: Eleventh Symposium On Usable Privacy and Security (SOUPS 2015). pp. 225–239. USENIX Association, Ottawa (Jul 2015)
18. Kuznetsov, A., Yoshshifuji, H.: iputils, http://www.skbuff.net/iputils/, accessed Dec 10th 2015
19. Lee, H., Kim, T.H., Choi, J.W., Choi, S.: Chirp Signal-Based Aerial Acoustic Communication for Smart Devices. In: Proc. of IEEE Conf. on Computer Communications (INFOCOM), Hong Kong SAR, PRC (2015)
20. Mare, S., Markham, A.M., Cornelius, C., Peterson, R., Kotz, D.: Zebra: Zero-effort bilateral recurring authentication. In: Security and Privacy (SP), 2014 IEEE Symposium on. pp. 705–720. IEEE (2014)
21. Meeker, M., Wu, L.: 2014 internet trends. KPCB (2014)
22. Narten, T., Nordmark, E., Simpson, W., Soliman, H.: Neighbor Discovery for IP version 6 (IPv6). RFC 4861 (Draft Standard) (Sep 2007), http://www.ietf.org/rfc/rfc4861.txt
23. Plummer, D.: Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission

on Ethernet Hardware. RFC 826 (INTERNET STANDARD) (Nov 1982), http://www.ietf.org/rfc/rfc826.txt
24. Statista: Smartphone OS worldwide by installed base in 2014 (2015), http://www.statista.com/statistics/385001/smartphone-worldwide-installed-base-operating-systems/, accessed Nov 30 2015
25. Thompson, D.: iBeacon: Is Bluetooth On? And Other Insights from Empatika (03 2014), http://beekn.net/2014/03/ibeacon-bluetooth-insights-empatika/, accessed Dec 1st 2015
26. Wehmeier, T.: Understanding todays smartphone user (2012), http://www.informatandm.com/wp-content/uploads/2012/08/Mobidia_final.pdf, accessed Nov 30 2015