Open access • Proceedings Article • DOI:10.1145/3212480.3212494

Detecting smartphone state changes through a Bluetooth based timing attack

— Source link

Guillaume Celosia, Mathieu Cunche

Institutions: Institut national des sciences Appliquées de Lyon

Published on: 18 Jun 2018 - Wireless Network Security

Topics: Bluetooth, Timing attack, Mobile device and Wireless

Detecting smartphone state changes through a Bluetooth based timing attack

Guillaume Celosia, Mathieu Cunche

▶ To cite this version:

Guillaume Celosia, Mathieu Cunche. Detecting smartphone state changes through a Bluetooth based timing attack. WiSec '18 - 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks, Jun 2018, Stockholm, Sweden. pp.154-159, 10.1145/3212480.3212494 . hal-01870011

HAL Id: hal-01870011

https://hal.inria.fr/hal-01870011

Submitted on 10 Sep 2018

Detecting smartphone state changes through a Bluetooth based timing attack.

Guillaume Celosia
Univ Lyon, INSA Lyon, Inria, CITI
F-69621 Villeurbanne, France
guillaume.celosia@insa-lyon.fr

Mathieu Cunche
Univ Lyon, INSA Lyon, Inria, CITI
F-69621 Villeurbanne, France
mathieu.cunche@insa-lyon.fr

ABSTRACT

Bluetooth is a popular wireless communication technology that is available on most mobile devices. Although Bluetooth includes security and privacy preserving mechanisms, we show that a Bluetooth harmless inherent request-response mechanism can taint users privacy. More specifically, we introduce a timing attack that can be triggered by a remote attacker in order to infer information about a Bluetooth device state. By observing the L2CAP layer ping mechanism timing variations, it is possible to detect device state changes, for instance when the device goes in or out of the locked state. Our experimental results show that *change point detection* analysis of the timing allows to detect device state changes with a high accuracy. Finally, we discuss applications and countermeasures.

CCS CONCEPTS

• **Security and privacy → Mobile and wireless security**; • **Networks →** *Network privacy and anonymity*;

KEYWORDS

Privacy, Bluetooth, Smartphone, L2CAP, Timing attack, Change point detection

ACM Reference Format:
Guillaume Celosia and Mathieu Cunche. 2018. Detecting smartphone state changes through a Bluetooth based timing attack.. In *WiSec '18: Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks, June 18–20, 2018, Stockholm, Sweden.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3212480.3212494

1 INTRODUCTION

Bluetooth is a widespread radio communications standard in the 2.4GHz ISM band [18]. This wireless protocol makes possible to exchange data over distances ranging from one meter to a hundred meters depending on the device class.

Today, Bluetooth is embedded in most smartphones and tablets. In parallel, a significant fraction of connected objects such as fitness sensors, smart-watches and headsets rely on Bluetooth to connect with the users' smartphones. Bluetooth is thus becoming a key element in many users' life, which are likely to leave their Bluetooth interfaces enabled most of the time.

Because of its massive adoption, Bluetooth has been subjected to various attacks aiming to compromise security and more particularly user's privacy. For instance, Bluetooth has been exploited to track user's location [6, 9, 22], infer user's physical activities [6].

In this paper, we are considering a scenario in which the attacker is interested in learning information on the state of a remote Bluetooth-enabled device. This information could be exploited by the attacker to profile the user or to identify him by correlating this information with visual observations.

Part of the Bluetooth operations are handled by the operating system and are relying on resources shared with the rest of the system. The speed at which the Bluetooth tasks are handled is thus affected by the overall load of the system. Observing the time taken by the system to perform those Bluetooth tasks, such as answering to a request, could reveal information on the load of the system and its state.

Based on this idea, we introduce an active attack allowing a remote attacker to infer information on state changes by leveraging the Bluetooth connectivity of the targeted device. This attack relies on a request-response mechanism (ping) provided by the L2CAP layer of the Bluetooth. By flooding the target with requests while recording the round-trip time (RTT), the attacker can analyze the timing information to detect device state changes. Those state changes can then be correlated with other information sources (visual observations) to gather additional information about the user.

We summarize our contributions as follows :

- We introduce a timing attack based on the Bluetooth's L2CAP ping mechanism to infer the device state changes;
- We demonstrate on three smartphones that this timing attack is practical and can detect changes between the following states such as idle, locked, active, running an app, inquiring for nearby Bluetooth devices, scanning for Wi-Fi networks;
- We present potential applications and countermeasures.

The paper is organized as follows. Section 2 highlights the related work and a Bluetooth background is provided in Section 3. Section 4 defines our attacker model. Device Bluetooth MAC address acquisition is explained in Section 5. Section 6 describes the attack and Section 7 presents the experimental results. Discussion is provided in Section 8 and Section 9 concludes the paper.

2 RELATED WORK

Timing attacks originally refer to cryptographic side channel attacks in which the attacker attempts to obtain secret information by analyzing the execution time of some tasks [11]. Brumley and Boneh showed [4] that it was possible to perform those attacks from a remote network host. In the context of wireless network, timing attacks have been mounted to recover the secret PIN of a Wi-Fi Protected Setup (WPS) [19].

The idea of inferring information on a device based on timing features of wireless communications has been explored in several contributions. Li et al. implemented WindTalker [12], a keystroke

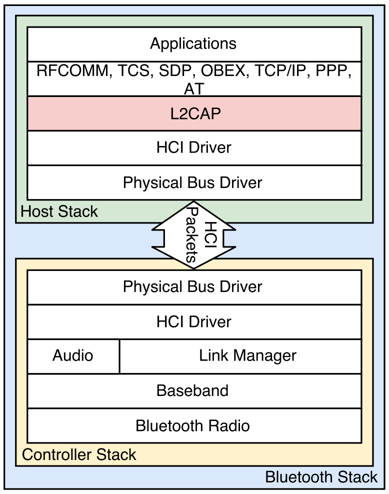

Figure 1: Bluetooth stack and the L2CAP layer.

inference framework that allows an attacker to infer mobile device keystrokes through Wi-Fi-based side-channel information. The side-channel information was collected through an active attack relying on the ICMP ping mechanism. Jamil et al. [10] presented smartphone screen on and off states detection by passively analyzing timing patterns of Wi-Fi probe requests. Our contribution is halfway between Li et al. and Jamil et al. as it leverages a ping mechanism to detect smartphone state changes.

Timing attacks have been exploited to derive information on the status of the operating system and the hosted applications. On Android, the timing of interrupt has been leveraged to infer running application or unlock pattern [7]. Timing attacks leveraging shared FIFO queues have been used to infer status of other process [20]. Similarly to this last work, we leverage the information obtained on the load of a shared resource to infer the status of a device.

Moyers et al. [14] and Adam et al. [1] provide insight into mobile devices battery exhaustion with BlueSmack, a Bluetooth denial of service attack based on the request-response L2CAP layer mechanism that our work differently use to recognize users' smartphones activities.

Wang et al. [21] present activity recognition attacks based on radio technologies by leveraging the influence of the human body on the propagation of radio signals.

3 BACKGROUND

To exchange data over radio communications, the Bluetooth protocol uses a multi-layered stack splitted in two parts [17] as represented in Figure 1. Implemented in microprocessors dongles, the "controller stack" contains the Bluetooth radio interface, while the "host stack" is an operating system part and processes high level data; the L2CAP host layer provides a link between these two parts. This packet-based protocol follows channels communication models where specific commands can be sent between remote devices.

As defined in the Bluetooth specification, ping relies on a request-response mechanism of the L2CAP layer. "Echo Request" command is sent to a remote L2CAP device and solicit an "Echo Response" back. As these requests are only designed to test a link between two Bluetooth devices, they do not require any mutual authentication.

"Echo Request" and "Echo Response" packets follow the same format and only differ by the packet's code. Those packets include the following fields:

- BD_ADDR (6 bytes): the destination MAC address;
- Code (1 byte): identifies the command type;
- Identifier (1 byte): used to match "Echo Request" with "Echo Response";
- Length (2 bytes): represents data field length;
- Data: contains uninterpretable data to be sent.

"Echo Request" identifier, length and data fields are copied to "Echo Response" during request-response exchanges.

Bluetooth device MAC address is a manufacturer unique device identifier.

In the context of Bluetooth, it is divided in three parts: the Non-significant Address Part (NAP), the Upper Address Part (UAP) and the Lower Address Part (LAP), which are respectively the 2 most significant bytes, the third most significant byte and the 3 less significant bytes of the MAC address.

The (UAP,LAP) pair is the minimum needed to establish a link with a remote device. In our work, (UAP,LAP) pair-based MAC address is defined as minimal Bluetooth device MAC address whereas complete Bluetooth device MAC address is NAP, UAP and LAP composed.

4 ATTACKER MODEL

We make the following assumptions about the attacker. First, we assume that the targeted device has its Bluetooth interface enabled and is in communication range, meaning that the attacker and the target can exchange Bluetooth frames. In particular, we presume that the attacker can perform L2CAP ping exchanges with the target. Those assumptions can be satisfied using off-the-shelf hardware and open-source software. A 10$ CSR Bluetooth 4.0 dongle along with Bluetooth tools featured GNU/Linux distribution is enough to meet those requirements.

Furthermore, we assume that the attacker knows his target's Bluetooth MAC address. If it is not initially available, this information can be obtained using one of the presented methods in Section 5.

5 MAC ADDRESS ACQUISITION

A prerequisite of our attack is the knowledge of the targeted device Bluetooth MAC address. Indeed, the attacker needs this address in order to send L2CAP ping requests to his target. In the following section, we present several techniques that can be used by the attacker to obtain this MAC address.

Bluetooth inquiry scan. A Bluetooth device can be set in discoverable (as opposed to non-discoverable mode). Once in this mode it will answer to "inquiry" signals thus revealing its MAC address. By performing an inquiry scan, the attacker can obtain the MAC address of nearby devices. Although it is supposed to only be used during pairing process, a number of Bluetooth devices are left in discoverable mode.

CSR8510 chipset, https://www.qualcomm.com/products/csr8510

Bluetooth UAP computation. By default, a Bluetooth device is in non-discoverable mode and does not respond to inquiry signals. Nevertheless, during communications with a previously paired object, Bluetooth frames are exchanged. In those frames, only LAP is available in clear text as part of the header. However, an attacker can derive the UAP by leveraging information provided by the CRC (Cyclic Redundancy Check) and HEC (Header Error Check) fields [16]. By monitoring communications between two Bluetooth devices, the attacker can thus obtain the minimal (UAP,LAP) pair required to send L2CAP ping requests.

BLE public MAC address. BLE devices can advertise their presence by emitting "advertisement" frames. To reduce privacy threats, MAC randomization mechanisms have been introduced [17, sec. 10.8], but in practice many BLE devices still use their public MAC address in those frames [6]. Usually, the public MAC address is shared between the BLE stack and all the other Bluetooth elements, thus these BLE public MAC addresses can be used for reaching the L2CAP layer. As a consequence, collecting MAC addresses found in BLE advertisements can reveal the MAC addresses of nearby Bluetooth devices.

Correlated Wi-Fi MAC address. On mobile devices, Bluetooth and Wi-Fi can be embedded on the same chipset and their MAC addresses are often contiguous [13]. For example, the Bluetooth MAC address of an Apple iPhone smartphone is equal to its Wi-Fi MAC address plus or minus 1. The Wi-Fi MAC address is available in clear text in many frames. In some situations, the device may not expose its true MAC address by using random MAC address, but it may still be possible to obtain the original MAC address [13]. Thus, the attacker can exploit Wi-Fi to obtain the Bluetooth MAC address of a device.

6 L2CAP BASED TIMING ATTACK

6.1 Attack description

We describe a timing attack that uses L2CAP layer ping mechanism to infer state changes of a Bluetooth enabled device. By measuring the request-response round trip time (RTT) variations, this attack allows a remote host to identify when the targeted device goes through a change of state.

Given a device identified by its Bluetooth MAC address, the attacker floods his victim Bluetooth interface L2CAP layer by sending "Echo Request" packets in large number. Each of those requests should trigger an "Echo Response" back. Then, the attacker measures the corresponding RTT and uses it to infer information on the device's status.

The idea behind this attack is that the ping flood will stress the Bluetooth stack up to the L2CAP layer. The speed rate at which those ping requests are processed will depend on the target available resources. On mobile operating systems, available resources for the processing of those requests may vary depending on the smartphone state. For instance, those resources will be reduced if an application is running, or if the smartphone is in a power saving mode. As a consequence, the RTT observations can be used to gain information on the device state. Figure 2 presents an illustration of this attack by showing the variations of the RTT when the device goes alternatively in the states *idle* and *locked.*

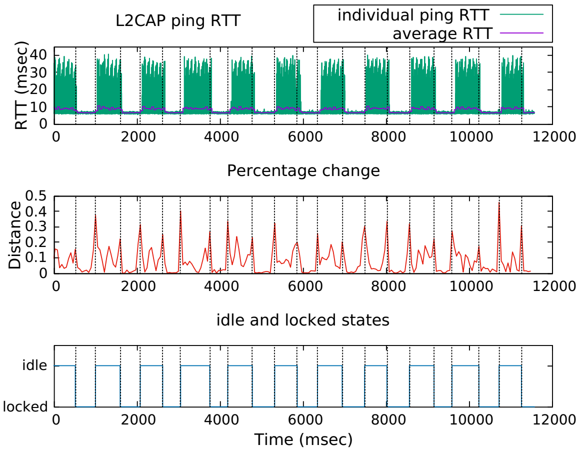

Figure 2: Bluetooth ping flood RTTs (individual and average), percentage change and actual state (ground truth) while alternating between *idle* and *locked* states. The percentage change is computed from the average RTTs. Vertical bars correspond to the times of the actual state changes.

Using this approach, it is possible to run a *state change identification* attack in which the attacker can detect when the targeted device changes its state; for instance, when it goes from *idle* to *locked* state as shown on Figure 2.

6.2 State change identification

Our attack aims at detecting smartphone state changes based on the measured timings. Let $O = \{o_k\}$ be the sequence of observations (ping RTTs) and $T = \{t_k\}$ be their corresponding timestamps. Similarly, let $C = \{c_i\}$ be the sequence representing the state changes: $c_i = 1$ if the device changed state at time t_i, $c_i = 0$ otherwise.

Thus, our *state change identification* problem falls down to inferring the sequence C based on the observation O. As previously observed, the change of state is characterized by the measured RTTs amplitude modifications. Detecting amplitude changes in temporal series is a problem known as *change point detection* [3] for which several methods have been developed. One of them is the *percentage change* algorithm. Based on a sequence of temporal values, it outputs a series of percentage $P = \{p_i\}$ corresponding to the likelihood of a change at a given time; the higher the probability at a given point the most likely the change.

Having this sequence of percentage changes, the next step is to select the actual change points. This is done by setting a threshold value h and by considering change points as all the time points for which the percentage change is above the threshold, in other words time points t_i for which $p_i > h$. In our experiments, we used the percentage change implementation, `pct_change()`, of the Panda Python library. The threshold is identified using an experimental approach as shown in Section 7. In order to cope with the high variations of the RTTs, we rely on the average RTTs over a period of 1 second. Figure 2 shows the variation of the *percentage change* values as the device goes from a state to another.

Panda: Python Data Analysis Library, https://pandas.pydata.org/

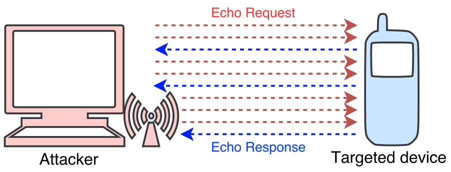

Figure 3: Attack scheme: the attacker floods his target with L2CAP ping requests while measuring the RTTs. Only B ping class Echo Responses are represented.

6.3 L2CAP ping RTTs measurements

This attack requires to be able to send a large number of L2CAP ping requests at regular interval and to measure their corresponding RTTs as described in Figure 3. Our implementation is based on the native l2ping tool "Flood ping" function available in BlueZ, an open source Bluetooth stack for Linux kernel-based operating systems.

This tool was modified to include several new features. A static code review with a dynamic execution analysis revealed that the l2ping "Flood Ping" function did not match with flooding definition; as it waited for an "Echo Response" packet back before reissuing an "Echo Request". We removed this limitation by implementing concurrent threads capable of sending ping requests in parallel. This allowed us to increase the request rate and ensure that they were sent at a regular interval. Then, we optimized the ping identifiers (IDs) management to ensure continuous RTT measurements while maintaining a constant ping requests rate. Indeed, the IDs used for measurements cannot be reused until the corresponding answers have been received. We bypassed this issue by interleaving two ping classes: A ping class requests that are only sent for flooding (ID $\in \{0\}$), and B ping class that are also used for measurements (ID $\in [1; 255]$).

7 EXPERIMENTAL EVALUATION

7.1 Considered devices and states

Tested devices. Experiments have been performed using three devices listed in Table 1. Those three devices cover both Android and iOS, the two most popular mobile operating systems, as well as older (Apple iPhone 4) and more recent devices (Samsung Galaxy A3 and Apple iPhone 6). Apple iPhone 4, the oldest smartphone in our devices set, has an early Bluetooth version. As all Bluetooth versions implement the L2CAP ping mechanism, it shows that the attack can also be efficient on old Bluetooth versions.

Device states. Throughout our experimental study, we focused on a set of states that can be considered as a representative of the standard usages of a smartphone. First, we considered states *locked*, *idle* and *active* that are states in which a device is often set. Then we considered a popular application to represent the states when the device is running an application. We selected another application that actively uses the Bluetooth interface of the device. Finally, we considered two states in which the device uses a wireless interface (Wi-Fi or Bluetooth). The seven considered states are the following:

Table 1: List of tested devices

Device name	OS	Bluetooth version
Apple iPhone 6	iOS 11.0.3	4.2
Samsung Galaxy A3	Android 5.0.2	4.0
Apple iPhone 4	iOS 7.1.2	2.1

- **locked**: the phone is left untouched after the screen has been turned off;
- **idle**: the phone screen is on, but no application is running neither in the foreground nor in the background apart from system specific applications;
- **active**: the phone screen is on, no application is running neither in the foreground nor in the background apart from system specific applications and the user only swipes his finger over the screen;
- **Shazam**: Shazam application is running in the foreground;
- **BLEScanner**: BLEScanner application is running in the foreground;
- **BT inquiry**: an inquiry scan for nearby Bluetooth discoverable devices is performed in the foreground;
- **Wi-Fi Scan**: a scan for nearby Wi-Fi networks is performed in the foreground.

7.2 Experimental protocol

To evaluate the performances of our approach as well as the potential differences between the various states, we conducted series of measurements. For each pair of the states listed above: we alternatively put the smartphone in each of the state for 10 seconds for an overall session of 120 seconds corresponding to a total of 12 changes. Some pairs were not possible as the transition between the states requires to go through one intermediary state. For instance, it is not possible to go directly from the *Shazam* state to the *BT inquiry* state as it is necessary to go through the *active* state first. While performing those manipulations, the device was submitted to active L2CAP ping measurements and the time of transitions were noted manually. The output of each experiment is a set of RTT measurements and a set of timestamps corresponding to changes of states.

Note that, as the state change monitoring was done manually, it does not have the same accuracy as the RTT measurements timestamps. As a result, we considered the granularity of time change detection of one second, which corresponds to the manual measurement expected accuracy.

Based on those measurements, it is possible to evaluate the *state change identification* attack. The performance evaluation has been done using the following metrics: **True Positive Rate**: the ratio of correct inference, **False Positive Rate**: the ratio of incorrect inference. In the case of the *state change identification* attack, a true positive is the detection of an actual change of state, while a false positive is a detection of state change where there is none.

BlueZ: Official Linux Bluetooth protocol stack, http://www.bluez.org/

Shazam, https://www.shazam.com/apps

BLEScanner, https://itunes.apple.com/us/app/ble-scanner-4-0/id1221763603?mt=8

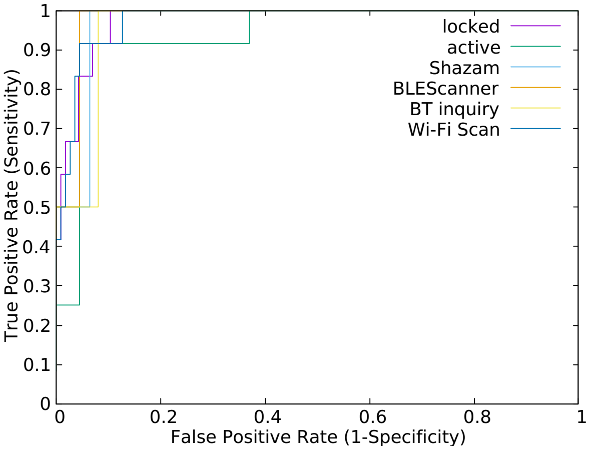

Figure 4: ROC (Receiver Operating Characteristic) curves for the *state change identification* from the *idle* state with the Apple iPhone 6 smartphone.

Flooding parameters. We want to maximize the stress on the Bluetooth stack while maintaining regular measurements. In particular, we want to avoid Bluetooth requests being dropped because the load was too high. Through empirical experimentations we found that sending a request every 6 ms was producing good results in term of RTT variations. To maintain a sufficient granularity of the measurements we set the interval for B ping class requests to 18 ms.

7.3 Results

In the following, we focus on the results obtained with the Apple iPhone 6 device. Its recent Bluetooth version as well as its monitoring ease with low generated noise makes this smartphone the ideal candidate to demonstrate our attack.

The efficiency of our *state change identification* technique has been evaluated using the previously presented protocol. Figure 4 presents the ROC (Receiver Operating Characteristic) curves of this detector for state changes either from or to the *idle* state. Those curves show that our approach is able to detect these changes with high accuracy. For instance, it reaches a True Positive Rate of 0.9167 for a False Positive Rate of 0.0455. Overall, the Area Under the Curve (AUC) is high and close to 1. Similar performances are observed for all the cases except for the *active* state, which is in practice close to the *idle* state as the only difference is that the user swipes his finger over the screen.

Table 2 summarizes the *state change identification* results for the Apple iPhone 6 smartphone by showing the AUC for each state transition. As previously shown for the *idle* case, the AUCs are high and are all above 0.9, demonstrating the good performances of the state change detector.

Comparable performances have been obtained on the two other smartphones. For Apple iPhone 4, AUC $\in [0.9224; 0.9795]$ where the lowest and highest values respectively represent *Wi-Fi Scan* to *locked* and *locked* to *BT inquiry* states transitions. The attack was more difficult to perform on Samsung Galaxy A3 as it only accepts about 10000 "Echo Requests" before resetting the L2CAP connection. For this smartphone experiments, we kept the previous defined protocol with a lower overall session of 60 seconds corresponding to a total of 6 changes. AUC $\in [0.9132; 0.9852]$ represents this smartphone experimental results where the lowest and highest values respectively correspond to *BLEScanner* to *locked* and *Wi-Fi Scan* to *idle* states transitions.

8 DISCUSSION

Attack range. The short range of commodity Bluetooth interfaces might be a limitation to the impact of this attack. However, this range can be significantly extended with the help of custom hardware solutions such as Hering's BlueSniper Rifle [5] or Bluetooone [2].

Correlated attacks. The attack presented in this paper could constitute a first step toward a more elaborated attack. It could be combined with another source of information to identify a user. For instance, the detected state changes of a device could be combined with visual observations of a set of smartphone users [15]; the attacker could correlate observable actions (smartphone in pocket, smartphone in use, smartphone being activated, ...) to identify the owner of a device having a given MAC address.

Countermeasures. Our attack is practical because any remote hosts can perform L2CAP ping requests at a high rate. Therefore, countermeasures should limit the rate at which a remote host can perform these actions or implement flood detection mechanism and blacklisting. Furthermore, the accuracy of the timing measurements is a key element of this attack; thus, introducing a perturbation by enforcing a random back-off time in the response process could reduce its efficiency. Although Bluetooth features mutual authentication mechanisms, the access to the L2CAP services are not limited to authenticated hosts. Restricting the L2CAP services access to authenticated hosts could be another way to prevent this kind of attack.

Applicability to connected devices. Only tested on smartphones, all Bluetooth devices that support this L2CAP request-response mechanism are potentially threatened. Thus, IoT devices [8] are also subject to this attack. By taking advantage of these single-task connected objects, the attacker can define the action that is taking place by analyzing RTTs variations. For instance, RTTs variations in a Bluetooth e-cigarette allow the attacker to know precisely when his target is smoking without having previously fingerprinted this object different states.

9 CONCLUSION

We presented a timing attack that leverages a L2CAP layer request-response mechanism to infer information on the state of a smartphone. This study shows that combining timing measurements with a *change point detection* analysis allow to detect a device state changes with a high accuracy. In this paper, we also highlighted four methods to obtain a device Bluetooth MAC address which represents our timing attack prerequisite.

Our experimental results show that a remote attacker can easily get information on the state of a device, that could be further exploited to obtain additional information on the device or the user.

Table 2: Apple iPhone 6 *state change identification* with AUC (Area Under ROC Curve) values.

from ↓ to →	idle	locked	active	Shazam	BLEScanner	BT inquiry	Wi-Fi Scan
idle	-	0.9756	0.9392	0.9679	0.9775	0.9598	0.9750
locked	0.9881	-	0.9596	0.9643	0.9732	0.9665	0.9808
active	0.9307	0.9894	-	0.9646	0.9662	0.9732	0.9735
Shazam	0.9624	0.9437	0.9583	-	N/A	N/A	N/A
BLEScanner	0.9874	0.9712	0.9777	N/A	-	N/A	N/A
BT inquiry	0.9420	0.9558	0.9643	N/A	N/A	-	N/A
Wi-Fi Scan	0.9591	0.9672	0.9662	N/A	N/A	N/A	-

For instance, the *state change identification* could be used to visually identify the user when he manipulates his device. To protect devices from this timing attack, we have described three potential countermeasures.

In future work, we will extend the attack by developing a method to identify the state of a device using machine learning techniques. This attack can also be extended to other wireless technologies such as Wi-Fi in which request-response mechanisms, such as RTS-CTS [13], could be leveraged to obtain similar results.

ACKNOWLEDGEMENTS

This work was supported by the SPIE ICS-INSA Lyon IoT chair.

REFERENCES

[1] Laurie Adam, Marcel Holtmann, and Martin Herfurt. 2004. Bluesmack. http://trifinite.org/trifinite_stuff_bluesmack.html

[2] Laurie Adam, Marcel Holtmann, and Martin Herfurt. 2004. Bluetooone. https://trifinite.org/trifinite_stuff_bluetooone.html

[3] Michèle Basseville, Igor V Nikiforov, et al. 1993. *Detection of abrupt changes: theory and application.* Vol. 104. Prentice Hall Englewood Cliffs.

[4] David Brumley and Dan Boneh. 2005. Remote timing attacks are practical. *Computer Networks* 48, 5 (2005), 701–716.

[5] Humphrey Cheung. 2005. How To: Building a BlueSniper Rifle - Part 1. http://www.tomshardware.co.uk/how-to-bluesniper-pt1,review-1224.html

[6] Aveek K. Das, Parth H. Pathak, Chen-Nee Chuah, and Prasant Mohapatra. 2016. Uncovering Privacy Leakage in BLE Network Traffic of Wearable Fitness Trackers. *ACM HotMobile (the 17th International Workshop on Mobile Computing Systems and applications)* (Feb. 2016). http://escholarship.org/uc/item/52h8734r

[7] Wenrui Diao, Xiangyu Liu, Zhou Li, and Kehuan Zhang. 2016. No pardon for the interruption: New inference attacks on android through interrupt timing analysis. In *Security and Privacy (SP), 2016 IEEE Symposium on.* IEEE, 414–432.

[8] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems* 29, 7 (2013), 1645–1660.

[9] Taher Issoufaly and Pierre Ugo Tournoux. 2017. BLEB: Bluetooth Low Energy Botnet for large scale individual tracking. In *Next Generation Computing Applications (NextComp), 2017 1st International Conference on.* IEEE, 115–120.

[10] Shuja Jamil, Sohaib Khan, Anas Basalamah, and Ahmed Lbath. 2016. Classifying Smartphone Screen ON/OFF State Based on Wifi Probe Patterns. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16).* ACM, New York, NY, USA, 301–304. https://doi.org/10.1145/2968219.2971377

[11] Paul C Kocher. 1996. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Annual International Cryptology Conference.* Springer, 104–113.

[12] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. 2016. When CSI meets public WiFi: Inferring your mobile phone password via WiFi signals. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.* ACM, 1068–1079.

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

[14] Benjamin R Moyers, John P Dunning, Randolph C Marchany, and Joseph G Tront. 2010. Effects of wi-fi and bluetooth battery exhaustion attacks on mobile devices. In *System Sciences (HICSS), 2010 43rd Hawaii International Conference on.* IEEE, 1–9.

[15] Le T. Nguyen, Yu Seung Kim, Patrick Tague, and Joy Zhang. 2014. IdentityLink: user-device linking through visual and RF-signal cues. ACM Press, 529–539. https://doi.org/10.1145/2632048.2636072

[16] Michael Ossmann. 2014. Discovering the Bluetooth UAP. http://ubertooth.blogspot.fr/2014/06/discovering-bluetooth-uap.html

[17] Bluetooth SIG. 2014. *Specification of the Bluetooth System v4.2.* Version. https://bugs.tizen.org/jira/secure/attachment/13248/output.pdf

[18] Axel Sikora and Voicu F Groza. 2005. Coexistence of IEEE802. 15.4 with other Systems in the 2.4 GHz-ISM-Band. In *Instrumentation and Measurement Technology Conference, 2005. IMTC 2005. Proceedings of the IEEE*, Vol. 3. IEEE, 1786–1791.

[19] Stefan Viehböck. 2011. Brute forcing wi-fi protected setup. *Wi-Fi Protected Setup* 9 (2011).

[20] Pepe Vila and Boris Köpf. 2017. Loophole: Timing attacks on shared event loops in chrome. In *USENIX Security Symposium.*

[21] Shuangquan Wang and Gang Zhou. 2015. A review on radio based activity recognition. *Digital Communications and Networks* 1, 1 (2015), 20–29.

[22] Ford-Long Wong and Frank Stajano. 2005. Location privacy in bluetooth. In *European Workshop on Security in Ad-hoc and Sensor Networks.* Springer, 176–188.