International Journal of Engineering & Technology, 7 (4.29) (2018) 192-195

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Fingerprinting Smartphones Remotely via Sensors Data

Ahmed Al-Haiqi[1, 2]*

[1]Department of Electronics and Communication Engineering, Universiti Tenaga Nasional, Kajang, Malaysia
[2]Institute of Power Engineering, Universiti Tenaga Nasional, Kajang, Malaysia
**Corresponding author E-mail: ahmedmubarak@uniten.edu.my*

Abstract

Remote fingerprinting of physical devices has already been shown to be feasible, using tiny deviations in the clock frequencies of fingerprinted machines. Both TCP and ICMP timestamps provided the basis for clock skew estimations. In the context of the emerging urban-sensing paradigm, we note the potential availability of logged sensors data, typically comprising timestamps, and probably labelled with external time reference information. This observation leads to the possible exploit of that new timestamps source to identify individual mobile nodes, and compromise participating users' privacy. Our experiments verify this conclusion, confirming the earlier results published on the same line of research, and pull the attention to a privacy breach that could be mounted remotely, offline and in non-real time, on logs of sensors data with necessary time reference labels.

Keywords: *Timestamps; Device fingerprinting; Urban-sensing; Privacy.*

1. Introduction

Integrating sensors in commodity mobile phones brought about new user experience and applications that were not possible on the public level before. Sensors even introduced new paradigms, given the unprecedented wide penetration of mobile devices into the Earth's population. Urban sensing, people-centric sensing, participatory sensing, and opportunistic sensing are the most common names for the new sensing paradigms, referring essentially to the same idea of next generation wireless sensors networks, where static sensors nodes are replaced with sensing-enabled mobile devices, carried by people [1]. Only imagination can limit the applications spectrum of such systems.

One of the main barriers to the new sensing paradigms is the privacy concerns over potentially sensitive data about the sensed entities [2]. As much as it seems attractively useful, not all entailments are yet understood as of how much those sensing capabilities can reveal on the individual level as well as the aggregated collective level. For example, an adequate amount of sensors data could tell the type of activities, location, or whereabouts of a specific mobile user; it could also disclose a social trend relative to some activity. To track the data of an individual user, it is crucial to distinguish between traces of different devices, utilizing implicit or explicit identifiers, the process known as fingerprinting the device.

Traditionally, in the realm of network security, remote fingerprinting aimed at identifying the operating system or the running software profile of a remote system, in an essential step to mount an informed attack on a specific, and implicitly already identified, target. Software fingerprints are not adequate to discern individual devices, as many users typically share a common set of OS/running services.

Hardware measures are unique across devices to a larger extent. Typical examples are MAC addresses assigned to network interface cards, and IMEI/ESN numbers assigned to mobile phones by manufacturers. These identifiers are deliberately set. Other traits could exist where the basis of identification is a sort of malfunctioning of hardware parts. This is, in a sense, similar to OS fingerprinting, where anomaly behavior with respect to the standard specifications of networking protocols differentiate between operating systems and their versions. Shortness of adhering to standards, and inability to fabricate perfect circuitry, both open opportunities for fingerprinting devices. Kohno et al. introduced the area of remote physical device fingerprinting in their seminal work [3], based on such unintended hardware measures. In particular, they showed that physical devices could be fingerprinted remotely by exploiting the device clock's skew: microscopic deviations in the device clock. Unlike MAC address fingerprints, their technique is applicable remotely over hops and across networks. Further, it does not require the cooperation of the fingerprintee, in terms of running application with special privileges, like what reading IMEI numbers would require.

The basic principle behind clock skew fingerprinting is to exploit the fact that machine clocks have skews that are measurable, and can be assumed constant over time. The skew of a clock refers to the frequency difference between the clock and some reference, possibly another clock or national standards. In essence, the clock skew is now the fingerprint of the physical device, allowing remote identification if timing information is available for each target device.

2. Related Work

The work in [3] utilized the TCP timestamp option as the source for timing data. The fingerprinter would use the information carried by TCP headers in the exchanged packets with a target to estimate the clock skew of that device, with respect to its own clock. The authors also considered, to a less extent, the ICMP timestamps. They confirmed that modern computer chips exhibit a detectable, stable, and differentiable clock skews, and showed how to exploit this fact to achieve remote physical device fingerprinting. They also analysed the effect of access technology, to-

pology, and distance through which the target is fingerprinted, and showed that at least the TCP option timestamp is independent of NTP synchronization. Finally, they explained some possible applications for this technique, and provided some potential countermeasures. Swati et al. [7] confirmed similar results through an empirical study that applied the same technique to a range of desktop, laptop, and smartphone devices. They based the estimation of clock skew on TCP and ICMP timestamps. Most recently, Cristea and Groza followed the same line of research, solely focusing on fingerprinting Android smartphones via ICMP timestamps over Wi-Fi channels [4]. This focus was based on the note that Android allows ICMP timestamp requests, and it responds to them.

We further take another step in the same direction, focusing on Android smartphone as the fingerprintee, but from a more specific perspective, and with a major difference. We depend on the timestamps embedded in sensors data as the source of timing information. These data are sent from built-in sensors on modern smartphones, and saved on servers for further processing. In this sense, our technique is more of an offline approach that could be applied to the logs of available sensors data, in the context of urban-sensing applications. Proposed architectures to support emerging sensing paradigms stress frequently on the importance of labelling sensors data with context information, like external network attested time and location stamps, to enhance their credibility and quality [5]. Those meta-data provide for the reference time against which to calculate fingerprinted devices' clock skews.

3. Terminology

Following the nomenclature in NTP specification, we define the offset of two clocks to be the time difference between them, and the skew is the frequency difference between them (first derivative of offset w.r.t. time). Smartphone built-in sensors, like accelerometer, can generate readings in relatively high frequency, but not all sensor records would contain external reference time. Let the trace of sensor records labeled with external network time be T, and all |T| records include both the sensor timestamp (T_s) obtained from the smartphone system clock, and a reference timestamp (T_r), either labeled by the network architecture supporting the sensing paradigm, or queried by the sensing application from NTP servers. $T_s^1$is the timestamp in the first observed record, and T_r^1 is the first reference timestamp. Similarly, for the i^{th} record, the pair of sensor and reference timestamps is T_s^i and T_r^i respectively.

Define

$$x^i = T_r^i - T_r^1$$
$$z^i = T_s^i - T_s^1$$
$$y^i = z^i - x^i$$
$$O_T = \{(x^i, y^i): i \in \{1 \dots |T|\}\}$$

y^i is the observed offset of the i^{th} sensor record, and O_T is the offset set corresponding to the trace T. Now, the slope of the points in O_T is the skew of the smartphone clock.

4. Methodology

Unlike previous works, our method relies on information in the application layer, and is not susceptible to the effects of network technology, topology, and latency. Our experiments assume exclusively a sensing paradigm context, as described in subsection 4.3, where timing information are part of the application content rather than packet timestamps (as in TCP option timestamps), or specifically queried messages (like ICMP time request packets).

4.1. Sensors data

Android platform supports a multitude of built-in sensors, capable of measuring device motion, orientation, and various environmental conditions (e.g., temperature, pressure, illumination and humidity). Programmers can acquire sensors readings through programming structures called *SensorEvent* objects. Besides the sensed raw data, sensor event objects contain a timestamp field that reports the time at which the event happened, in nanoseconds since uptime. Sensors generate events at relatively high rate (typically 5-100 Hz), but we assume lower frequencies of sensing architecture-labelled data. In our experiments, we assume a rate of 1 Hz; a record per second is sent to the server and attested by the architecture with a timestamp. Few minutes are adequate to estimate the clock skew to a precision of few ppm, and we presume 10 minutes on average for all experiment targets. We also exam the effect of monitoring intervals on one Android device.

4.2. Calculating the clock skew

Following the same approach in [3] and [4], we adopt the linear programming technique proposed in [6] to compute the clock skews of fingerprinted devices. Figure 1(a) depicts an example trace of smartphone clock offset against time with respect to the reference clock. In terms of the terminology presented in section 3, the figure is a plot of y values against x values on the horizontal axis. The skew is the slope of the point set, and the chosen approach is to calculate the slope of a line that uppers bound all the points to represent the skew, as shown in Figure 1(b).

The task in this view has many constraints and one linear polynomial objective function, a typical form of linear programming problems. The desired skew is the slope of the line $(ax + b)$, where *'a'* is the slope and '*b'* is the intercept with the *y*-axis. Every point (x^i, y^i) in the set should lie below the line. Hence, $ax^i + b \geq y^i,\ i \in \{1, \dots, |T|\}$ and the distance between the line and the set of points should be minimum, yielding the objective function: $min\{\sum_{i=1}^{|T|}(ax^i + b - y^i)\}$.

Though noisy measurements do occur, the linear programming algorithm avoids their impact by placing the line above all points. Yet, we found that sometimes removing outliers helps in cleaning the figures.

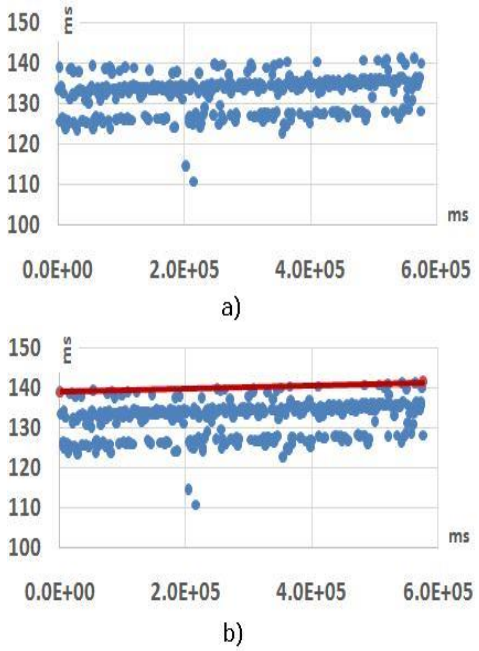


Fig. 1: a) Example clock offset trace, b) the skew as the slope of the upper bound line.

4.3. Experimental setup

We built an Android app that simulates a typical opportunistic sensing application, where, in terminology of [5], the requester initiates a campaign, asking for data relative to some area and/or

time duration, and the appropriate smartphones respond with the required sensors data, potentially labelled with architecture-attested timestamps. These reference times could also be embedded by the sensing application itself, and queried from external sources, like NTP servers. For the purpose of the simulation, we obtained all time references from the system clock of a single workstation. Six Android smartphones were used in the experiments, listed in Table 1. The data are captured for about 10 minutes in each case, labelling the readings once a second with the reference clock time from the workstation, and saving the data in files for later processing. This scenario is a representative case of the expected application usage in the upcoming sensing paradigms architectures, based on the proposed systems in literature.

We also repeated the experiment for Samsung Galaxy S II, introducing some gaps in the sensing intervals. The aim is to investigate the consequences of calculating the skew with discontinuity in time, so that not only different devices can be distinguished, but also the same device can be identified in future logs.

Table 1: Smartphones Used in Experiments and Their Clock Skews

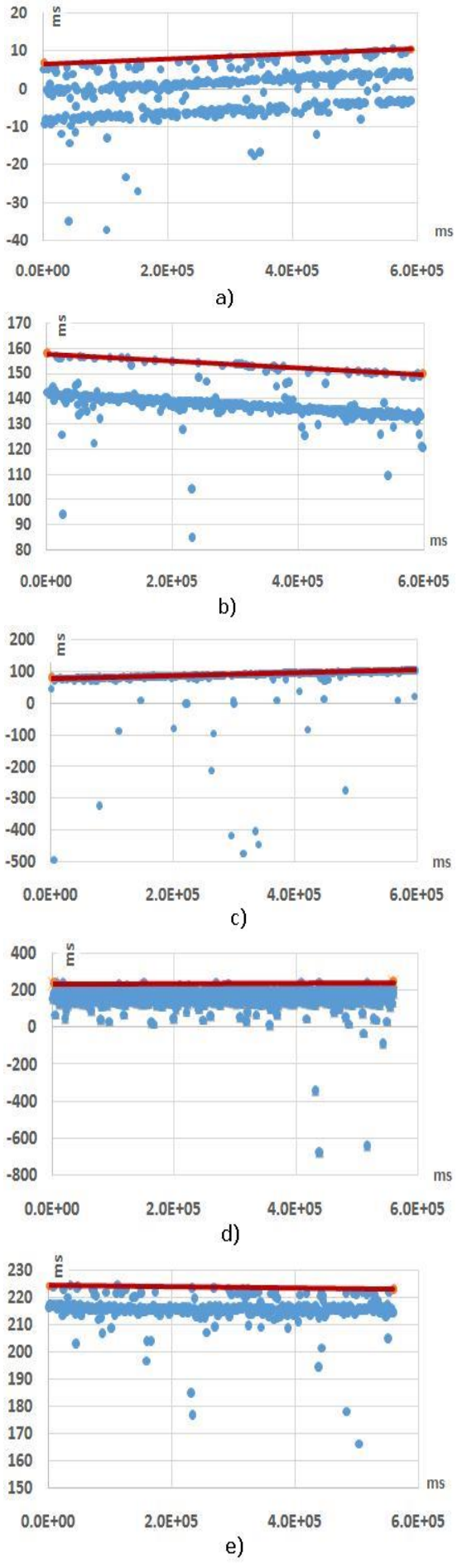
Smartphone	Model no.	Clock Skew (ppm)
Samsung Galaxy S II	GT-I9100G	6.52
Samsung Galaxy S III Mini	GT-I8190	-13.46
Sony Xperia Neo L	MT25i	45.13
Samsung Ace Plus	GT-S7500	15.20
Samsung Galaxy Tab 2	P3100	-2.89
Samsung Galaxy S III	GT-I9300	21.71

5. Results and Discussion

Figure 2(a-f) shows the offsets of the six Android devices (yi values) with respect to the reference clock's time of measurement (xi values), alongside the lines computed using linear programming. We used Octave *glpk()* function to solve the linear programming problems, producing the optimal line equation parameters, of which the slope is the skew estimation. Table 1 lists the skews for the six Android devices, in ppm (part per million), or how many microseconds are deviated each second of reference time. The values are comparable to [4], where the targets are the same class of devices (smartphones). The detectable differences between skew values further confirm the previously found result that clock skews of physical devices are indeed feasible fingerprints. This holds true for desktop, laptop and mobile devices as well.

Figure 3 agrees with the conclusion attested in [3]; the clock skews can be conceived as constant over time, within a precision of around 1 ppm. Unsurprisingly, the skew of Galaxy S II in Fig. 1 is unlike that of the same smartphone in Fig. 3, as the reference clock is different (6 ppm and 14 ppm respectively).

It might be worthy to note, however, as we talk about resolutions of parts per million, that the number of smartphones is on the order of billions, and increasing. In our experiments as well as in previous works, many devices differ in their clock skew estimations by few ppm. Even if we assume a uniform distribution of clock skews over a wide range, we should expect collisions in these values sooner or later. Therefore, this technique of remote device fingerprinting (via sensors data or otherwise), is more effective when applied against limited sets of target devices.



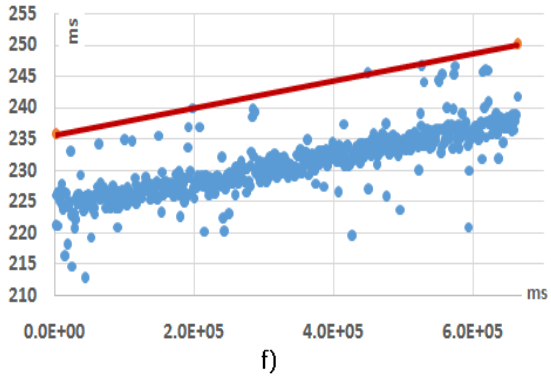


Fig. 2: Clock offsets for a) Samsung Galaxy SII; b) Samsung Galaxy SIII Mini; c) Sony Xperia Neo L; d) Samsung Ace Plus; e) Samsung Galaxy Tab2; f) Samsung Galaxy SIII.

3.4.1. Figure captions

Figures must be numbered using Arabic numerals. Figure captions must be in 8 pt Regular font. Captions of a single line (e.g. Figure 2) must be centred whereas multi-line captions must be justified (e.g. Figure 1). Captions with figure numbers must be placed after their associated figures, as shown in Figure 1.

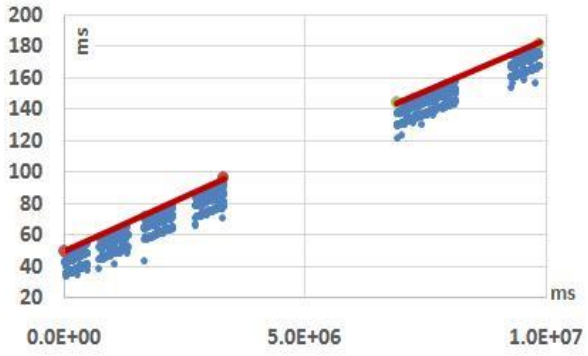


Fig. 3: Clock offset of Galaxy S II for discontinuous measurements.

6. Conclusions

Estimating clock skews from logged sensors data (with necessary reference timestamps), and hence fingerprinting smartphones, is but one class of information that sensors data in general can reveal. Nevertheless, it holds its implications on the design of privacy-preserving urban-sensing architectures. Even encrypting the data is not a sufficient remedy, as they eventually have to be decrypted at mediate or at least final beneficiaries and those data are not meant to convey identification information of individual users' smartphones. Imposing a level of abstraction (e.g. aggregation) on the data may or may not be feasible, depending on the purpose of the sensing campaign in the first place (the level of details requested), and implies trusting the abstracting party.

One obvious approach, in the context of the scenario examined here, is to restrict the external labels of time nature when the shared data contains timestamps, though proposed architectures for sensing paradigms often require some kind of architecture-level credibility, provided via such external reference time/location stamps.

References

[1] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G. S. Ahn," The rise of people-centric sensing", *IEEE Internet Comput*, Vol.12, No.4, (2008), pp.12-21.

[2] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell," A survey of mobile phone sensing", *IEEE Commun. Mag.*, Vol.48, No.9, (2010), pp.140-150.

[3] T. Kohno, A. Broido, and K. C. Claffy," Remote physical device fingerprinting", *IEEE Trans. Dependable Secur. Comput*, Vol.2, No.2, (2005), pp.93-108.

[4] M. Cristea and B. Groza," Fingerprinting Smartphones Remotely via ICMP Timestamps", *IEEE Commun. Lett*, Vol.PP, No.99, (2013), pp.1-3.

[5] J. Bruke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, M. B. Srivastava," Participatory sensing", *Proceedings of the ACM SenSys Workshop World-Sensor-Web*, (2006), pp:1-5.s

[6] S. B. Moon, P. Skelly, and D. Towsley," Estimation and removal of clock skew from network delay measurements", *Proceedings of the INFOCOM'99 18th Annu. Joint Conf. IEEE Computers and Communications Societies*, (1999), pp: 227-234.

[7] S. Sharma, H. Saran, and S. Bansal," An empirical study of clock skew behaviour in modern mobile and hand-held devices", *Proceedings of the 3rd Int. Conf. of Communication Systems and Networks COMSNETS*, (2011), pp: 1-4.