Future Generation Computer Systems 96 (2019) 283–296

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Feasibility analysis of Inter-Pulse Intervals based solutions for cryptographic token generation by two electrocardiogram sensors[☆]

Lara Ortiz-Martin [b], Pablo Picazo-Sanchez [a,*], Pedro Peris-Lopez [b], Juan Tapiador [b], Gerardo Schneider [a]

[a] *Chalmers, University of Gothenburg, Sweden*
[b] *Universidad Carlos III de Madrid, Spain*

HIGHLIGHTS

- We analyze how to generate the same random token from an ECG in two different sensors.
- Two sensors cannot derive a common token by applying an error correction technique.
- We propose a timed automaton to synchronize two ECG sensors.
- To generate a 32, 64 and 128 bits tokens, sensors should wait 13, 28 and 56.5 seconds.
- We have developed a proof-of-concept implementation of an ECG-based token generator.

ARTICLE INFO

Article history:
Received 11 December 2018
Received in revised form 30 January 2019
Accepted 12 February 2019
Available online 22 February 2019

Keywords:
Biometrics
Security
Privacy
Body sensor networks

ABSTRACT

In this paper we address the problem of how two devices that are sensing the same heart signal can generate the same cryptographic token by extracting them from the Inter-Pulse Intervals (IPIs) of each cardiac signal. Our analysis is based on the use of a run-time monitor, which is extracted from a formal model and verified against predefined properties, combined with a fuzzy extractor to improve the final result. We first show that it is impossible, in general, to correct the differences between the IPIs derived from two captured electrocardiogram (ECG) signals when using only error correction techniques, thus being impossible to corroborate previous claims on the feasibility of this approach. Then, we provide a large-scale evaluation of the proposed method (run-time monitor and fuzzy extractor) over 19 public databases from the Physionet repository containing heart signals. The results clearly show the practicality of our proposal achieving a 91% of synchronization probability for healthy individuals. Additionally, we also conduct an experiment to check how long the sensors should record the heart signal in order to generate tokens of 32, 64 and 128 bits. Contrarily to what it is usually assumed (6, 12, and 24 s for individuals with a heart rate of 80 beats-per-minute), the sensors have to wait 13, 28 and 56.5 s on median, respectively, to derive the same token from both sensors.

1. Introduction

Interest in biometrics has gained momentum in the last years mostly due to the massive use of daily life devices like smartwatches, smartphones and laptops [1,2]. This technology identifies and authenticates people in an automatic way based on biological and behavioral traits [3]. This interest is not temporary. According to a recently published report, global biometric market revenues will reach $34.6 billion annually in 2020, especially in mobile devices [4].

From a technical point of view, biometrics can be classified into two main groups depending on whether they use physiological or behavioral signals. Examples of physiological signals include fingerprints, iris, retina, heart and brain signals, whereas voice, signature analysis or keystroke dynamics are behavioral signals. The main reason why such signals can be easily included in authentication systems is because they exhibit a number of desirable features: they are universal, collectible, unobtrusive, permanent, unique, and difficult to circumvent [5].

[☆] This work was partially supported by the Spanish CAM grant S2013/ICE-3095 (CIBERDINE), the Spanish MINECO grant TIN2016-79095-C2-2-R (SMOG-DEV), and, by the Swedish Research Council (Vetenskapsrådet) under grant Nr.2015-04154 (PolUser: Rich User-Controlled Privacy Policies).

[*] Corresponding author.
E-mail addresses: laortizm@inf.uc3m.es (L. Ortiz-Martin), pablop@chalmers.se (P. Picazo-Sanchez), pperis@inf.uc3m.es (P. Peris-Lopez), jestevez@inf.uc3m.es (J. Tapiador), gersch@chalmers.se (G. Schneider).

https://doi.org/10.1016/j.future.2019.02.021
0167-739X/© 2019 Elsevier B.V. All rights reserved.

The research outcome in this area is that most gadgets, such as smartphones, tablets, wearables and *Implantable Medical Devices (IMDs)*, have been equipped with one or more embedded sensors with the ability to measure biometric parameters from the bearer. Besides having biometrics sensors, most (if not all) of these devices are enhanced with some wireless communication technology, e.g., Bluetooth, WiFi or *Radio Frequency (RF)*, allowing them to share data and to perform remote reconfiguration [6]. All the above has given birth to the so-called *Wireless Body Area Network (WBAN)*.

In the last years, several works have focused on using the heart signal as part of either authentication protocols [8–10], human identification [11,12], or as a key generation algorithm [13–16] to enable secure communications. More concretely, authors use the *Electrocardiogram (ECG)* to extract the time difference between two consecutive heartbeats (R-peaks). These time intervals are referred to as *Inter-Pulse Intervals (IPIs)* or RR-intervals and have been shown to contain some degree of entropy after applying a quantization algorithm (see Section 3.2.1). This makes the IPI values an ideal candidate to generate tokens to be used in cryptographic solutions (e.g., [8,10,17–19]).

In order to obtain a biometric signature based on the heart signal, different sensors such as ECG, *Photoplethysmographic (PPG)* or *Blood Pressure (BP)* can be used. The ECG signal is measured using electrodes usually placed on the chest which detect the tiny electrical changes in the heart and generate a complex digital signal. The PPG detects the pulse of the heart by measuring the amount of light which is reflected in the skin to a photodiode. As a light source, most of the commercial gadgets have a LED on them, e.g., smartwatches and sport wrists. As an example on how these advances may be used for new purposes, some researchers have recently used a BP sensor to get the bearer's heart signal [10]: this sensor can measure the pressure in large arteries in the systemic circulation, so the signal reflects the up and down fluctuation of the arterial pressure which is related to each heartbeat.

Using these sensors is not trivial though, as there are some technical difficulties due to different factors. For example, even when two similar sensors – from the same manufacturer, having the same brand, and with the same capabilities – are measuring the same heart signal in the same part of the body, the resulting signal would likely be different in both sensors due to the noise of the signals, missed data during the gathering phase, delays, or simply because of the bearer's movements [20].

Along the same lines, it has been reported in [21] that both *Heart-Rate Variability (HRV)* and *Inter-Sensor Variability (VAR_{is})* measurements directly affect the processing of the heart signal and, in particular, the peak detection procedure. These issues become crucial when a cryptographic protocol entirely relies on biometric data acquisition to generate random tokens, e.g., random seeds or fresh nonces, to be used for key generation [22] or in authentication procedures [8].

In particular, the problem of signal synchronization is quite relevant in the health sector where expensive medical electrodes are used. Let us consider a real example of measuring ECGs using two different sensors. Fig. 1 shows two ECG signals, channel 1 (ECG_1) and channel 2 (ECG_2), taken from the public database *svdb* [7]. This database is composed of 78 half-hour ECG recordings of supraventricular arrhythmias. The *beats per minute (bpm)* in both signals are the same, or, in the worst case, show a difference of a few bpm. However, at low level, the time differences between two consecutive heartbeats (R-peaks), are slightly different in ECG_1 and ECG_2. Thus, despite sensing the same ECG from the same patient, both channels have different signals, and it is easy to see that even by shifting any of the signals it could not be possible to fully synchronize them.

Authors are somehow aware of this problem and for instance in [20], a miss-detection algorithm is proposed that given two ECGs, authors "manually" add a peak in the place where it was supposed to be whenever it is detected that a peak was missing in order to generate the same token in different devices. Some years later, in [23], authors propose a key-exchange protocol among a Programmer[1] and an IMD where both devices generate the same key from the heart signal. After gathering the same signal, authors apply a *Bose-Chaudhuri-Hocquenghem (BCH)*, which is an *Error Correcting Code (ECC)*, to the generated keys in both devices to finally get the same value.

1.1. Our work

No matter if authentication protocols for WBAN were published [8,11,24], if key distribution schemes based on the heart signals were proposed [9,23] or whether authors assumed that there is a secure communication channel and a shared key is derived from the heart signal to be used afterwards in a cryptographic protocol [10,13,20,25], all these proposals rely on the same assumption: there are two sensors measuring the heart signal and they can derive the same cryptographic token under an IPI-based approach and after applying an ECC algorithm like BCH. Unfortunately, after an in depth analysis (19 databases), we show that the above claim does not hold when only an ECC algorithm is used to correct errors between the two generated tokens.

Motivated by this, we carry out an analysis on the (open) question concerning the generation of a cryptographic token based on the analysis of IPI values from different ECG devices that are sensing the same heart signal. Our analysis is based on the use of a run-time monitor, extracted from a formal model, i.e., a timed automaton, that is verified against predefined properties, combined with a fuzzy extractor (i.e., an ECC) to improve the final result. We show that it is impossible, in general, to correct the differences between the two captured signals when using only the fuzzy extractor, thus being impossible to corroborate previous claims on the feasibility of the approach.

Our proposed method can successfully synchronize two heart signals through IPI values and extract a common token that can be used afterwords as part of a cryptographic protocol, as one more security check in order to proof that both devices are attached to the same body by proving that they are listening to the same heart signal, i.e., they are attached to the same body.

To the best of our knowledge, this is the first work to use a run-time monitor in combination with a fuzzy extractor. In addition, to demonstrate the validity of our approach, we provide a large-scale evaluation of the proposed method over 19 public databases containing heart signals. However, we do not evaluate how good or bad the IPI-based generated random tokens are from a cryptographic point of view; we urge the reader to consult [26] for an in-depth analysis of this issue.

After applying our proposed solution to public databases containing at least two measurements of heart signals (ECG_1 and ECG_2), we conclude that a fuzzy extractor (or another error correction technique) is not enough to correct the synchronization errors between the IPI values derived from two ECG signals captured via two sensors placed on different positions (Section 3). In particular, we show that a pre-processing of the heart signal must be performed before the fuzzy extractor is applied.

1.2. Contributions

In summary, our contributions are:

[1] Device used to (re)configure IMDs.

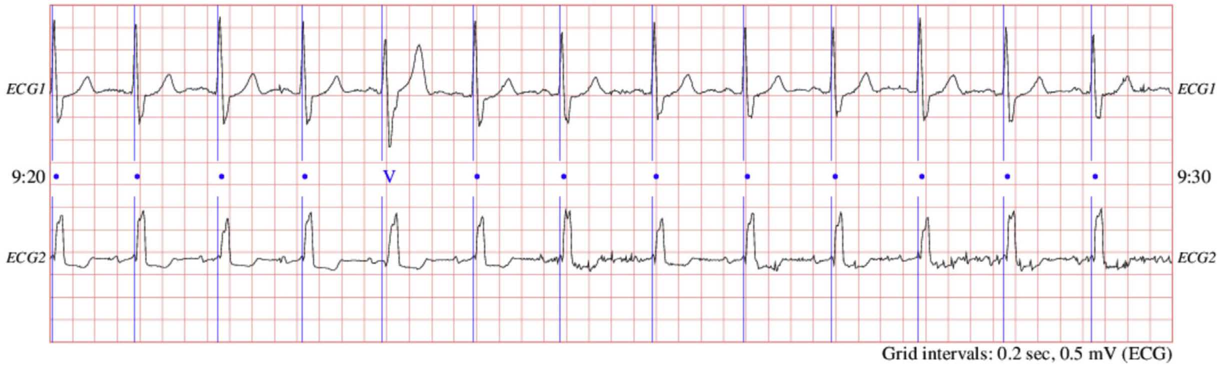

Fig. 1. Two ECG signals from svdb [7] database.

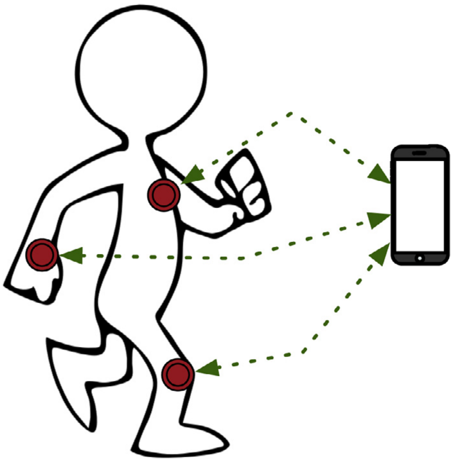

Fig. 2. Body area network.

- We perform an in-depth analysis of the problem of how to synchronize two cryptographic tokens generated by two different ECG sensors that record the same heart signal and use the IPIs as the basics for generating the mentioned tokens. We show how an initial signal pre-processing step is necessary for the error correction algorithm (e.g., fuzzy extractor) to work properly. Our results show that it is not possible to assume that two sensors can derive a common token just by applying an error correction algorithm without having previously synchronized both signals. In summary, this first result gives evidence that the assumptions under which previous IPI-based solutions operate are not correct and does not guarantee that the same token can be extracted from two ECGs sensors (Section 3.2).
- In order to perform the synchronization (at IPI values level) between two ECGs sensors, we have generated a run-time monitor from a timed automaton, which has been verified correct with respect to predefined timing properties. We compare our results before and after applying a fuzzy extractor and demonstrate our improvement in performance (Section 3.3).
- We modified our timed automaton and the monitor in order to extract a token with a given accuracy (namely 32, 64 and 128 bits), in order to gather statistical information on how long it would take (median) to get a token with the requested accuracy. We found that to generate a 32, 64 and 128 bits tokens, a sensor should wait on median 13, 28 and 56.5 s, respectively (for individual with a heart rate of 80 beats-per-minute), instead of 6, 12, and 24 s as reported in previous works, i.e., [8,11,27,28] (Section 3.3.2).
- We have developed a proof-of-concept implementation of an ECG-based token generator by using a BITalino shield[2] (Section 4). This shield has two ECG channels connected using wires and the pre-processing is executed before the token generation (IPI-based approach in our particular case) takes place. The purpose of this proof-of-concept is to shed further light on the technical real difficulties in getting a fully working implementation of such a solution.
- As it was previously stated, the contributions in this article shed light on the feasibility of IPI-based solutions, where two sensors obtain such values from the same organ (in our case the heart). On the other hand, in this article we do not analyze the security of IPI values, which has been widely studied in the literature (e.g., [8,10,18,26]).

1.3. Organization

The rest of this paper is organized as follows, in Section 2 we provide some basic knowledge in order to facilitate the reading of the rest of the paper. Section 3 presents the core of our work, while Section 4 introduces our proof-of-concept implementation of the proposed solution. Section 5 contains a summary of the main published papers in this research area. Finally, we conclude and present future directions for further research in the last Section.

2. Background

In this section we provide some preliminaries on *Body Area Networks (BANs)* and we give a brief overview of the datasets used for the experiments. After this, we yield an overview of related work that has explored how heart signals can be applied to biometrics and cryptography. We also discuss why fuzzy extractors are often used in the literature together with biometrics. Finally, we give some background about modeling and verification of real-time systems focusing on how formal verification is used to verify the run-time monitor that we use in to synchronize two ECG signals.

2.1. Body area networks

With the recent advances on technology, manufacturers are creating small and affordable sensors that people can be equipped with in order to acquire different parameters from their vital signs. For instance, athletes usually wear chest band to measure the heart beats while training or even when they are competing. In the case of elderly people, they might be remotely monitored

[2] http://bitalino.com/en/.

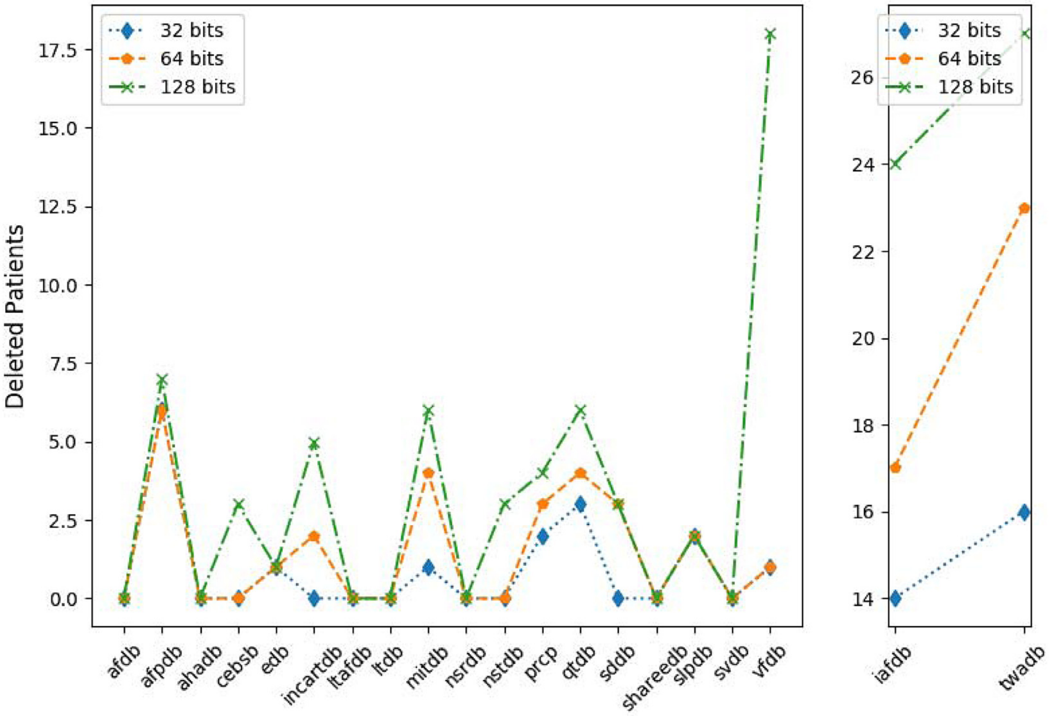

Fig. 3. Deleted patients vs. tokens length.

without the need to be in a medical center. Moreover, nowadays it is common to have smartwatches or sport gadgets equipped with accelerometers, *Global Positioning System (GPS)*, and PPG to measure the heart rate. These devices also have communication modules such as WiFi, Bluetooth or RF.

When all these gadgets are working together, it is said that they are part of a Wireless Body Area Network (WBAN) (Fig. 2). That is, a WBAN is a private network composed of sensors and/or actuators that measure different vital signs and send this information to a central node, typically the bearer's smartphone – which is assumed to be trusted – that acts as a gateway between the WBAN and the Internet [29,30].

2.2. The Physionet repository

Physionet [31] is a public repository composed of different databases about physiologic signals of healthy and patients with diseases. The main purpose of this repository is to allow and encourage researchers to investigate in the study of diseases and physiologic signals. Specifically, in this work we are only focusing on heart signals and, more precisely, in those databases with at least two ECG channels. That being said, popular databases such as *fantasia* [32] or *apnea-ecg* [33] are not considered in our study because there is only one ECG channel in their records. On the contrary, when more than two ECG signals are found in the same file, we are taking the first two signals we found (in a sequential order) in the *.hea* file which is a special file where the metadata of the record is stored.

In order to automate the process, we implemented a script to download 19 databases from the Physionet repository. A description of the databases can be seen in Table 1 where the number of files represents the number of patients we used in our experiments. Apart from that, we computed the average number (median) of R-peaks (heartbeats) that both the first channel of the ECG (ECG_1) and the second channel of the ECG (ECG_2) have. For each database we also included the heart condition (if any) of the patients.

From that table it is interesting to see that the number of peaks, using the well-established Pan–Tompkins algorithm for peak detection [51], is only equal in three databases: *cebsdb*, *qtdb* and *vfdb* whereas the values are almost equal in the *iafdb* and *twadb* databases. All the rest of the databases (14 out of 19) have different number of peaks.

Finally, Fig. 3 shows the number of patients that cannot be considered part of the dataset because they do not reach the minimum number of IPIs which is 8, 16 and 32 to compute the tokens of 32, 64 and 128 bits respectively.

2.3. Heart signals in cryptography

The use of ECG signals and IPI-based approaches for cryptographic applications has been widely studied in the literature. Even though some researchers take more than the 4 *Least Significant Bits (LSBs)* of the IPI to generate cryptographic tokens, (e.g., [52,53]), the vast majority of the research community, e.g., [8,10,11,13,15,17–20,54–57] use the 4 LSBs extracted after applying the quantization algorithm explained in Section 3.2.1, or a slight variation of it proven to contain some degree of entropy. As we try to be as general as possible, we use the 4 LSBs to generate tokens in our experiments. It is worth mentioning that although our work is focused on IPI values, which is the most widely used approach, some authors have proposed alternative solutions which work in a transform domain (e.g., [58] or [16]).

In most of the aforementioned IPI-based works it is assumed that there are two devices listening to the heart signal and they extract a random token which is used afterwards in a cryptographic protocol. However, to the best of our knowledge, no one has performed an in-depth empirical analysis to check if it is indeed possible to extract a common token from the same signal (particularly, the ECG) gathered from different devices over the same body. Our work aims to fill in this gap and focused exclusively on IPI-based approaches.

Table 1
Summary of the databases.

Database	Files	Peaks in ECG$_1$	Peaks in ECG$_2$	Heart condition
afdb [34]	23	49 003	48 294	Atrial fibrillation
afpdb [35]	300	1 817	1 797	Paroxysmal atrial fibrillation
ahadb [36]	2	8 473	8 183	Healthy and ventricular ectopy
cebsdb [37]	60	360	360	Healthy
edb [38]	90	8 852	881	Myocardial and hypertension
iafdb [39]	32	91	88	Atrial fibrillation or flutter
incartdb [40]	75	2 263	2 327	Coronary artery disease
ltafdb [41]	84	110 632	108 205	Paroxysmal
mitdb [42]	48	2 204	2 227	Arrhythmia
nsrdb [43]	18	99 746	10 066	No significant arrhythmias
nstdb [44]	15	2 556	2 544	Mitdb with noise
prcp [44]	10	4 310	3 355	Healthy
qtdb [45]	105	1 044	1 044	Holter recordings
sddb [46]	22	25 969	36 615	Arrhythmia
shareedb [47]	139	95 809	95 896	Hypertension
slpdb [48]	18	21 087	23 892	Sleep apnea syndrome
svdb [7]	70	2 322	2 323	Partial epilepsy
twadb [49]	100	185	184	Myocardial problems
vfdb [50]	22	3 457	3 457	Tachycardia

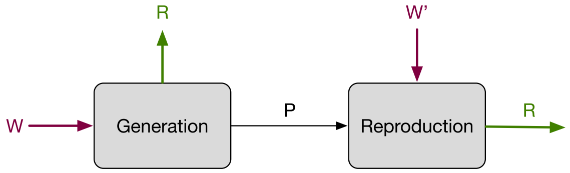

Fig. 4. Scheme of fuzzy extractor [67].

2.4. Fuzzy extractor

Juels and Wattenberg were the first who introduced the term fuzzy commitment in [59], where a cryptographic key is extracted from a biometric signal such as an ECG or an *Electroencephalogram (EEG)*. The process of generating this key is through an algorithm called *fuzzy extractor*.

Fuzzy extractors are not only applied to key generation protocols based on biometrics [60–62] but also for generating keys for authentication purposes, by using *Physical Unconlable Functions (PUFs)* [63,64], and for key generation in *Vehicular Ad-Hoc Networks (VANETs)* [65].

Formally, a fuzzy extractor is a function f which takes as input a biometric signal w, and produces a random string R and a public parameter P. Fuzzy extractors are particularly suitable for cryptographic protocols because when the input w' changes slightly, i.e., $w' = w + \epsilon$ for a very small ϵ, the random output R remains invariant [66].

Typically, a fuzzy extractor is composed of two main phases: *generation* and *reproduction* [66]. As it can be seen in Fig. 4, in the generation phase, a biometric signal w is received as input and two parameters are given as output: a secret value R and a public value P. In the reproduction phase, a fresh biometric signal w' is given as input together with the public parameter P, previously generated in the generation phase. If and only if the distance between these two biometric signals – typically the Hamming distance – is less than a given threshold t_r ($Hamming(w, w') < t_r$), then the same output R will be retrieved.

2.5. Modeling and verification of real-time systems

Our application is a typical example of a real-time system, where a number of real-time constraints must be satisfied. Our proposed solution is based on the satisfaction of three important real-time properties concerning: (i) the time between two consecutive peaks of each ECG signal; (ii) the relative time between peaks from the different heart signals; (iii) the total sampling time to return back a valid token. Note that this final requirement is to force the algorithm to finish its execution after a fixed time. We give some upper-bounds of these times in Section 3.3.2.

The design, reasoning and implementation of real-time systems have been addressed by different communities, in particular by formal methods researchers and more specifically those concerned with real-time verification [68]. In that community, the idea is to make an abstract model to represent the real-time system or some specific time constraints of the system, and apply tools to increase the confidence that the model satisfies some properties. One of the most broadly used formalism to model real-time systems is *timed automata* [69], for which reasonable mature tools have been developed to reason about, e.g., UPPAAL [70] and KRONOS [71]. In those tools, one specifies the model as a timed automata and writes properties about it on a real-time logic called *Timed Computation Tree Logic (TCTL)* [72].

The idea is that after performing such verification on the model, one may then write an implementation by taking the timed automaton as a starting point. Depending on the abstraction level of the model, the implementation might be more or less difficult to obtain. Though there is a gap between the model and the implementation, and errors might be introduced when an implementation is obtained from the model, it is clearly an advantage to have a verified model in the first place. As we will see later, in our case the implementation is directly obtained from the model, which gives us quite a high confidence on the correctness of our solution with respect to the specified timing constraints.

3. ECG-based token generation procedure

In this section, we first explain the methodology we have followed to carry out our research. We then explain in detail how we generated tokens from different ECG signals, and demonstrate how a pre-processing phase is needed to agree on the same token generated. Finally, we propose a timed automaton satisfying our properties and create the corresponding monitor in order to synchronize the signals (and thus generate the same token).

3.1. Our methodology

All the experiments presented in this section were run on a Macbook Pro 2.4 GHz with 4 Gb of RAM. The processing of all the patients' signals were implemented on Matlab.

We analyze all the performed experiments and discuss the results obtained after generating two tokens independently (emulating different sensors) in 4 scenarios: 1. running a quantization algorithm (Section 3.2.1); 2. running a fuzzy extractor algorithm (Section 3.2.2); 3. running a run-time monitor (Section 3.3.1), and; 4. running a run-time monitor and a fuzzy extractor algorithm (Section 3.3.2). Finally, from our results we conclude that synchronization of the signals is a must if we want the sensors to derive the same token from the ECG signal.

Regarding the run-time monitor, two specific values need to be computed beforehand: the time between two consecutive peaks from both the same ECG channel and from different channels. To generate an upper-bound of those values with statistical significance, we require the person to be quite and calm. However, due to the fact that we are using the Physionet repository with all the signals already measured, we decided to use the mean of the time interval between R-peaks of each one of the signals as an upper-bound which is a common technique used in medical research [73]. Additionally, we set the maximum time between consecutive peaks for different signals in the case of Physionet repository to $\frac{1}{f_s}$ where f_s is the reading frequency of the device where the signal is gathered. This is due to the fact that this parameter is determined by the physical distance between sensors and in the case of the Physionet databases, all the patients were monitored using wired ECG sensors attached to their chest.

Having computed those numbers, we have verified three main properties: (i) the time between two consecutive peaks of each ECG signal; (ii) the relative time between peaks from the different heart signals, and; (iii) the total sampling time. Similarly [20], we consider that when the time interval between two consecutive peaks from the same signal is longer than the computed upper-bound, then the monitor resets its clocks and considers that there a miss-detected peak was found. Also, when the time interval between two consecutive peaks from different signals is longer than $\frac{1}{f_s}$ then the monitor resets its clocks and considers that those peaks are not synchronized. Finally, we have proved that after t seconds, the final state is always reached and if and only if there are enough synchronized IPI then a token is computed.

3.2. *Debunking ECG-based token generation Myths*

3.2.1. *Token generation algorithm*

Our first experiment goal was to generate as many tokens of 128 bits as possible from both channels (ECG_1 and ECG_2) of the patients of all the databases to know how different they are. In order to process the ECG signal—which is a continuous signal, it must be transformed to a discrete one. This process is known as *quantization* (e.g., uniform or dynamic quantization) and it is one of the most important steps in the token generation based on heart signals [74].

As far as we know, in the context of ECG IPI-based approaches, the dynamic quantization firstly proposed by Rostami et al. in [8] is the most extended in the literature. In a nutshell, their algorithm works as follows. First, the ECG signal is cleaned (i.e., the DC component is eliminated and then the ECG signal is passed through a pass-band filter with 0.67 Hz and 45 Hz cut-off frequencies [75]). Second, R-peaks are extracted from the heart signal by using Pan–Tompkins algorithm [76] and the time difference between R-peaks are computed and thus, the IPIs are generated. Third, the IPIs values are dynamically transformed into values between 0 and 1. Then the data are multiplied by 256 and rounded to the nearest integer. Finally, the Gray code encoding scheme with 8-bit of precision is used to facilitate error correction and the 4 LSBs of each IPIs are extracted to generate a token. This token is computed by appending these 4 LSBs [17,19]; in order to create a 128-bit number at least 32 IPIs should be processed. The source code of the dynamic quantization is freely available at https://github.com/aylara/synchro (i.e., see `getIPIsSignal.m` file).

The pseudocode of the aforementioned IPI extraction algorithm, which is also used in this paper to process the signal and extract the 4 LSBs of the IPIs, is shown as Algorithm 1.

Algorithm 1 IPIs' extraction.

```
procedure IPIGENERATION(record)
    signal ← get_signal(record)
    freq ← get_sampling_frequency(record)
    cleaned_signal ← ECG_pre-processing(record)
    IPIs ← Pan_Tompkins(cleaned_signal,freq)
    IPIs ← dynamic_quantization(IPIs)
    result ← [ ]
    for ipi ∈ IPIs do
        grey← grey_code(ipi)
        $IPI_{NEW}$ ← get_LSB(gray)
        result.append($IPI_{NEW}$)
    end for
    return result
end procedure
```

As mentioned before, we generate as many 128 bits tokens as possible per user per database and the result of this analysis can be seen in the second column of Table 2. Note that this column contains the sum of all the tokens extracted per database for only one channel (ECG_1 or ECG_2). Also, we computed how many of these tokens are similar by calculating the Hamming distance between each pair of tokens from both channels (ECG_1 and ECG_2) before performing any signal processing and compared them pairwise. The results can be seen in the third column of Table 2. It is interesting to see that the number of similar tokens is extremely low in all databases which means that the output of the quantization algorithm cannot directly be used to generate similar tokens in different devices.

As a conclusion, we corroborate our claim that just applying an IPI extraction algorithm like the one presented in (Algorithm 1) composed of the Pan–Tompkins algorithm plus a dynamic quantization to generate tokens is not enough to guarantee that the same token will be generated in two different sensors.

3.2.2. *Fuzzy extractor*

Following the scheme presented in [61], we implemented a fuzzy extractor algorithm, which was specifically adapted to work with ECG signals. The scheme of the fuzzy extractor can be seen in Fig. 5. The fuzzy extractor takes as input two ECG signals (ECG_1 and ECG_2) and two random numbers ($PRNG_1$ and $PRNG_2$). The ECG_1 and the mentioned random numbers are provided in the generation phase since they are needed in the computation of the *Helper Data* (i.e., (s, x)), which is used in the reproduction phase together the ECG_2 signal. The result of the fuzzy extractor is a pair of identical values R. Our fuzzy extractor is publicly available at https://github.com/aylara/synchro (i.e., see `simulation_fuzzyextractor.m` file).

We assign the following values for the parameters m, n, k and t of the BCH:($m = 7$, $n = 127$, $k = 50$, $t = 13$), following the guidelines given in [61]. The parameter $t = 13$ represents a trade-off between the correction capability and the ability the adversary has to break the protocol. Thus this parameter should not be increased arbitrarily since it would increase the success probability of an adversary. This means that the BCH can recover at most 13 different bits from words of 128 bits (i.e., a 10% of the bits). We urge the reader to consult [77] for a detailed description of BCH parameters and their implications. An additional argument for this 10% value ($t = 13$) is that we have empirically demonstrated that is not possible to achieve 90% of similarity

Table 2
Number of tokens of 128-bit tokens generated by Algorithm 1 (column 2); Number of similar tokens after running Algorithm 1 (column 3); Number of similar tokens after running Algorithm 1 + Fuzzy Extractor (FE) (column 4); Number of tokens after running Algorithm 1 + Run-time Monitor (RM) (column 5); Number of similar tokens after running Algorithm 1 + Run-time Monitor (RM) (column 6); Number of similar tokens after running Algorithm 1 + Run-time Monitor + Fuzzy Extractor (RM+FE) (column 7).

DB	Tokens (Algorithm 1)	Similar tokens	Similar tokens (FE)	Tokens (RM)	Similar tokens (RM)	Similar tokens (RM+FE)
afdb	35 690	8 (0.02%)	77 (0.2%)	1549 (%)	847 (54.6%)	1495 (96.5%)
afpdb	14 505	40 (0.27%)	740 (5.1%)	9251 (%)	45 (0.48%)	1196 (12.9%)
ahadb	511	0 (0%)	0 (0%)	15 (%)	2 (13.3%)	14 (93.3%)
cebsdb	2 577	2 (0.07%)	1360 (52.7%)	839 (%)	59 (7.0%)	835 (99.5%)
edb	24 262	21 (0.08%)	497 (2.0%)	3769 (%)	1995 (52.9%)	3706 (98.3%)
iafdb	207	0 (0%)	34 (16.4%)	23 (%)	11 (47.8%)	23 (100%)
incartdb	5 127	0 (0%)	69 (1.3%)	1117 (%)	4 (0.3%)	241 (21.5%)
ltafdb	271 605	185 (0.06%)	1188 (0.4%)	21 337 (%)	92 (0.4%)	870 (4.0%)
mitdb	3 198	0 (0%)	45 (1.4%)	770 (%)	0 (0%)	110 (14.2%)
nsrdb	52 290	1980 (3.7%)	4072 (7.7%)	13 965 (%)	26 (0.1%)	1176 (8.4%)
nstdb	1 160	0 (0%)	8 (0.6%)	171 (%)	0 (0%)	25 (14.6%)
prcp	825	0 (0%)	0 (0%)	22 (%)	2 (9.0%)	20 (90.9%)
qtdb	3 413	1 (%)	216 (6.3%)	1346 (%)	695 (51.6%)	1315 (97.6%)
sddb	21 280	1212 (5.6%)	1732 (8.1%)	1364 (%)	572 (41.9%)	1029 (75.4%)
shareedb	405 775	2638 (0.6%)	4758 (1.1%)	60 440 (%)	297 (0.4%)	3229 (5.3%)
slpdb	8 860	0 (0%)	0 (0%)	172 (%)	27 (15.6%)	169 (98.2%)
svdb	5 710	2 (0.03%)	105 (1.8%)	2984 (%)	9 (0.3%)	315 (10.5%)
twadb	528	0 (0%)	31 (5.8%)	204 (%)	107 (52.4%)	203 (99.5%)
vfdb	2 144	0 (0%)	49 (2.2%)	221 (%)	93 (42.0%)	216 (97.7%)

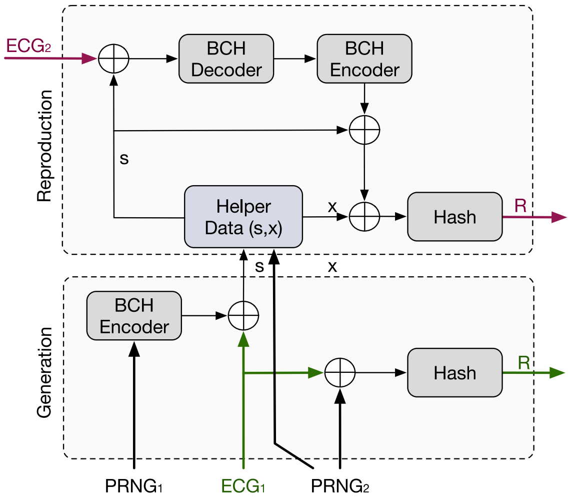

Fig. 5. Fuzzy extractor.

in the tokens generated without our run-time monitor together with the fuzzy extractor (see columns 3 and 4 from Table 2).

In order to check how the fuzzy extractor behaves, we used the output of the Algorithm 1 as input of the fuzzy extractor and computed the Hamming distance between each pair of tokens compared tokens pairwise and the results can be seen in the third column of Table 2. Even if our fuzzy extractor produces a slight improvement, with respect to the results obtained without performing any pre-processing of the signal (see column 2), the results are far from being the expected ones. For instance, the *cebsdb* database which achieves a 52.7% of the similar tokens in both channels is not a good result, i.e., 1 out of 2 generated tokens is random. The reason for getting these poor results stems from the fact that the distance between the IPIs calculated from each sensor clearly exceeds the correction capacity of the fuzzy extractor (BCH encoder). In our experimentation, for words of 128 bits the correction capacity is set to $t = 13$.

3.3. How to generate ECG-based tokens

3.3.1. Timed automata

Timed automata are composed of five main parts: clocks, time-checks, actions, events and states. For our timed automaton, we have defined three different clocks, namely c_1, c_2 and c_3, which are in charge of checking the time properties of the heart beats in our model. Concretely, c_1 checks an upper bound for the execution of the automaton, that is, how long the automaton should be executed; c_2 checks when the peaks from both signals are synchronized or not, and; c_3 checks when there are missed peaks in the same signal.

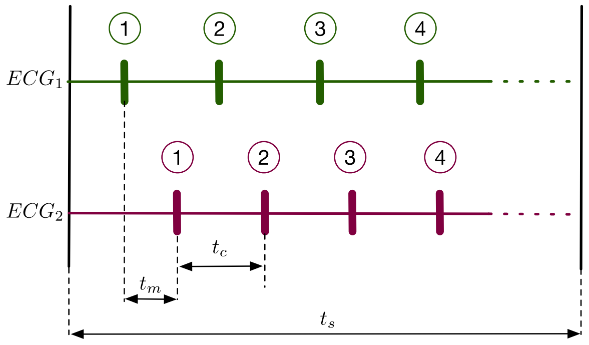

Fig. 6. Time-checks used in the timed automaton.

All the time checks used in the automaton were obtained after having analyzed all databases. We show in Fig. 6 a representation of these time checks. More concretely:

t_c This value varies in time and between each person. In order to compute t_c, as stated in Section 3.1, the person should be in a quite and peaceful environment. For our experiments and following the similar technique proposed in [73], we calculated the mean time between R-peaks of each pair of ECGs (ECG_1 and ECG_2), which is the value assigned to the time-check t_c.

t_m This value is determined by the physical distance between sensors and hence, it is directly affected by the speed of the blood pumped from the heart to the rest of the body. In our particular case, all the databases of the Physionet repository always consider electrodes attached to the chest of the patients, so we forced this value to be less than $\frac{1}{f_s}$ where f_s is the sampling rate. So, $t_m < t_c$, otherwise a missed peak is detected and discarded by the automaton.

t_s This value is a bound that determines how long each “session” of the execution of the monitor should be. We set this

Table 3
Properties of the automaton (Fig. 7).

Clocks	
c_1	Sampling time
c_2	Time between peaks of two signals
c_3	Time between two consecutive peaks (same signal)
Time-checks	
t_c	Time between two consecutive peaks (same signal)
t_m	Time between peaks of two signals
t_s	Sampling time
Actions	
Log	Stores those IPIs which are not synchronized
Reset	Initializes the clocks given as input
ReturnPeaks	Returns the non-synchronized IPI set
Sync	Checks what IPIs are synchronized
Events	
$Peak_{ECG_x}$	R-Peak of ECG_x, where $x \in [1, 2]$
ϵ	No event
States	
E_0	Initial state
E_1	When a peak of the first signal is detected
E_2	When a peak of the second signal is detected
E_3	When one peak of each signal is detected
E_4	When the max time is detected ($c_1 \geq t_s$)

value to be equal to the longest signal encountered in our databases, in order to ensure we consider all the signals.

Table 3 shows all the variables and constants of our automaton (shown in Fig. 7), defined as a tuple $\mathcal{A} = \{L, X, \Sigma, \Delta, F\}$, where:

- $L = \{E_0, E_1, E_2, E_3, E_4\}$ is the set of locations (with E_0 and E_4 the initial and final states, respectively);
- $\Sigma = \{Log, Reset, ReturnPeaks, Sync\}$ are all the actions;
- $X = \{c_1, c_2, c_3\}$ is the set of clocks;
- $\Delta \subseteq L \times X \times \Sigma \times 2^X \times L$ is the transition relation, where $F \subseteq L$ is a set of accepting locations.

The Log action keeps a list of those IPIs which are not synchronized according to our time constraints. Reset initializes the clocks given as input. *ReturnPeaks* returns the list of non-synchronized IPIs. Finally, the Sync action computes the list of IPIs which are synchronized.

Regarding the *events*, we have two types: when a peak comes from the ECG_1 or from the ECG_2. Additionally, we have defined ϵ which means that we do not wait for any event to occur and we force the runtime monitor to check if the condition is satisfied to perform the transition to the corresponding state.

The automaton has 5 *states*. All the clocks and variables are set to 0 in the initial state E_0. Note that whenever $c_1 \geq t_s$ then the computation finishes (the automaton is in state E_4). The rest are intermediate states, ensuring progress in the computation provided the relevant timing constraints are respected ("accepting" or "rejecting" peaks).

We implemented our timed automaton in Uppaal [70], allowing us to validate and verify our model in a formal way. Our verified model was then translated into a *runtime monitor* implemented as Matlab code. The source code of both implementations are available at https://github.com/aylara/synchro (i.e., see `Automaton` and `UPAAL` folders).

We tested our generated runtime monitor with the output of the Algorithm 1. The number of tokens has decreased considerably as it can be seen in the fifth column of Table 2. After that, we then computed the Hamming distance between each pair of tokens compared pairwise and the results can be seen in the sixth column of Table 2. Note that, in general, the number of similar tokens has increased considerably after running the runtime monitor with respect to the third column. However, this improvement does not come for free. The penalty we have to pay is that the number of tokens has decreased per database as it can be seen in fifth column of such a Table.

3.3.2. *Timed automaton & fuzzy extractor*

As already explained, our approach consists of combining our monitor (extracted from our verified timed automaton, for synchronizing the tokens (based on IPI values) extracted from two different ECG signals) and the fuzzy extractor (to correct some bits).

The results can be seen in the last column of Table 2. After applying this solution we can successfully generate the same token from different sensors with high probability in the majority of the databases, i.e., 10 out of 19 databases have a probability higher than 90% of taking two similar tokens generated on different sensors. However, despite of our method improves the current state of the art, it will remains low for 8 databases, namely *afpdb*, *incartdb*, *ltafdb*, *mitdb*, *nsrdb*, *nstdb*, *shareedb* and *svdb* whereas *sddb* achieves a 75.4% of probability that two arbitrary tokens be similar.

From the above results, we can clearly conclude that the best databases to be used to extract cryptographic tokens are the ones with healthy patients. Moreover, our method seems to work reasonably well with those patients whose disease is not severe. Hence, we recommend not to use databases such as *mitdb* which is widely used in the research community for security purposes [8,20,53,78,79] or *nsrdb* [19,80] to mention a few.

Having empirically demonstrated the effectiveness of our proposed method, the only question that remains uncovered yet is how long the run-time monitor needs to listen to the heart signal in order to obtain a token which can be used later on as part of a cryptographic protocol.

We conducted an additional experiment to measure how long the run-time monitor needs to keep listening an ECG signal in order to produce a token of 1. 32 bits (Fig. 8); 2. 64 bits (Fig. 9), and; 3. 128 bits (Fig. 10). To carry out this test, we modified the original timed automaton (Fig. 7) in such a way that, instead of having 3 different clocks (c_1, c_2 and c_3), we only keep c_2 and c_3, and replace c_1 by a counter. By doing so, as soon as the automaton detects that the length of the token is 32, 64 or 128 respectively, then the final state E_4 is reached. Roughly speaking, taking into account that we can only extract the 4 LSBs from an IPI, the automaton will stop when it finds 8, 16 or 32 synchronized IPIs. We have then re-implemented our new automaton in Matlab, getting a new monitor.

Furthermore, in order to make Figs. 8–10 more readable, we decided to discard some of the outliers and we kept the 70% of the original data. It can be observed that, in order to get a 32-bit token, sensors need to listen approximately for 13 s on median. Similarly, to get a 64-bit or 128-bit token, they should listen the ECG for 28 and 56.5 s, respectively, on median. It is also noticeable that although at a first sight the above mentioned timing values might appear to be excessive, this generation process will only be executed once, typically in the setup phase of the cryptographic protocol (e.g., key generation and synchronization processes between two sensors).

4. Proposed solution

In this Section we provide a proof-of-concept implementation to demonstrate whether two sensors can derive the same token from the heart signal using real hardware. The purpose of the presented proof-of-concept is to show the feasibility of our solution as well as the minimum requirements for generating common

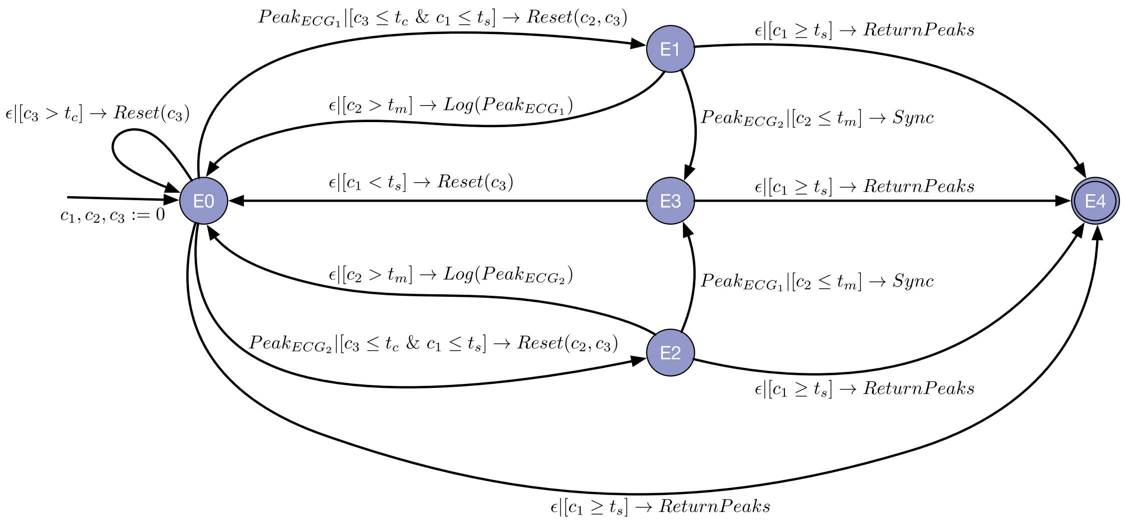
$Peak_{ECG_1}|[c_3 \leq t_c \;\&\; c_1 \leq t_s] \rightarrow Reset(c_2, c_3)$
E1
$\epsilon|[c_1 \geq t_s] \rightarrow ReturnPeaks$
$\epsilon|[c_3 > t_c] \rightarrow Reset(c_3)$
$\epsilon|[c_2 > t_m] \rightarrow Log(Peak_{ECG_1})$
$Peak_{ECG_2}|[c_2 \leq t_m] \rightarrow Sync$
$\epsilon|[c_1 < t_s] \rightarrow Reset(c_3)$
$\epsilon|[c_1 \geq t_s] \rightarrow ReturnPeaks$
E0
E3
E4
$c_1, c_2, c_3 := 0$
$\epsilon|[c_2 > t_m] \rightarrow Log(Peak_{ECG_2})$
$Peak_{ECG_1}|[c_2 \leq t_m] \rightarrow Sync$
$Peak_{ECG_2}|[c_3 \leq t_c \;\&\; c_1 \leq t_s] \rightarrow Reset(c_2, c_3)$
E2
$\epsilon|[c_1 \geq t_s] \rightarrow ReturnPeaks$
$\epsilon|[c_1 \geq t_s] \rightarrow ReturnPeaks$

Fig. 7. Heart based timed automaton.

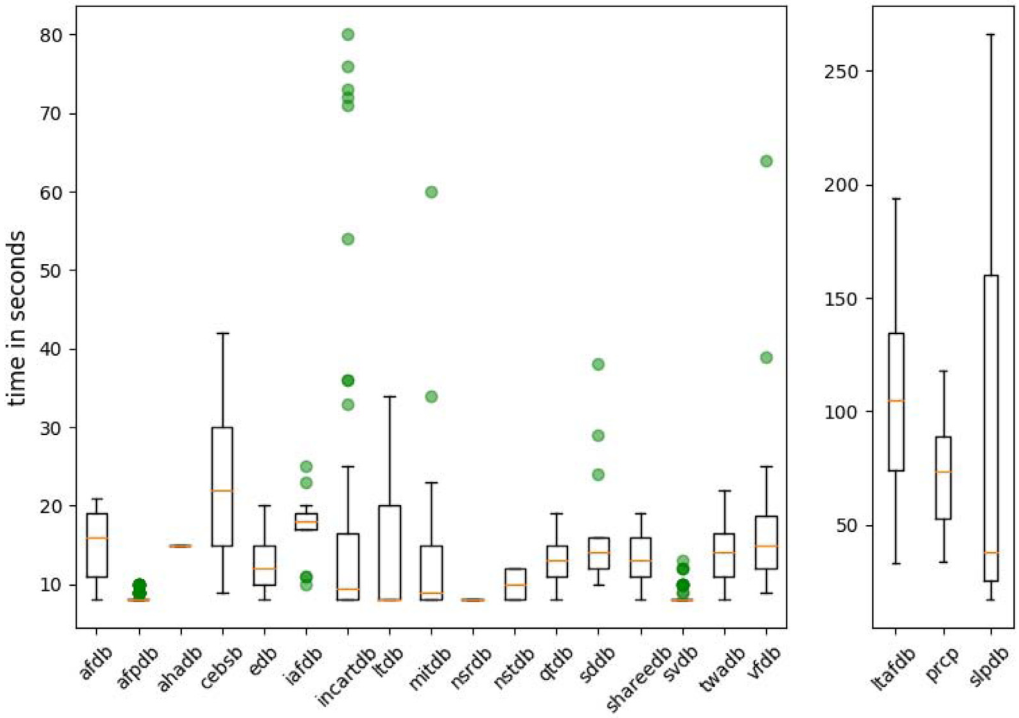

Fig. 8. Time needed to generate a 32-bit token.

tokens on different ECG sensors. For this first approach, and similarly to previous proposals (e.g., [23,81,82]), we assume the communications to be secure between the sensors and the gateway (at least during the set-up phase). Alternatively, we could have used noisy cryptography [83] to share sensitive information (the two ECGs in our particular case) via insecure channels.

A real example in which the above scenario can occur is as follows. Imagine that Alice is wearing a smart T-shirt with an ECG monitor similar to the one proposed in [84]. This T-shirt is already paired with her smartphone which makes the communication channel secure. Additionally, she has a pacemaker which is as well paired with the smartphone. In this scenario, the smartphone is acting as a WBAN gateway due to its computational resources in terms of CPU, storage, memory and communication capabilities. The above example is integrated within what is called body area networks. Another example, perhaps more futurist and within the area of the intelligent and connected cars could be the following. A driver (Bob) holds a smart-watch with an ECG sensor—note that this type of product is already on the market [85]. As for the car, the steering wheel, and as a novelty, also has an ECG sensor [86,87]. As in the above example, both ECG sensors are securely connected to the car's central control system that acts as gateway. The two described examples are completely different but in both scenarios the two sensors must calculate the same cryptographic token (derived from the ECG recorded by each sensor) with the help of the gateway.

Summarizing, our system has three entities: a gateway and two ECG sensors (e.g., smart T-shirt and pacemaker).[3] Fig. 11 provides an architectural view of the system. Once the ECGs signals

[3] Note that it could have also been possible to use one of the sensors as a gateway.

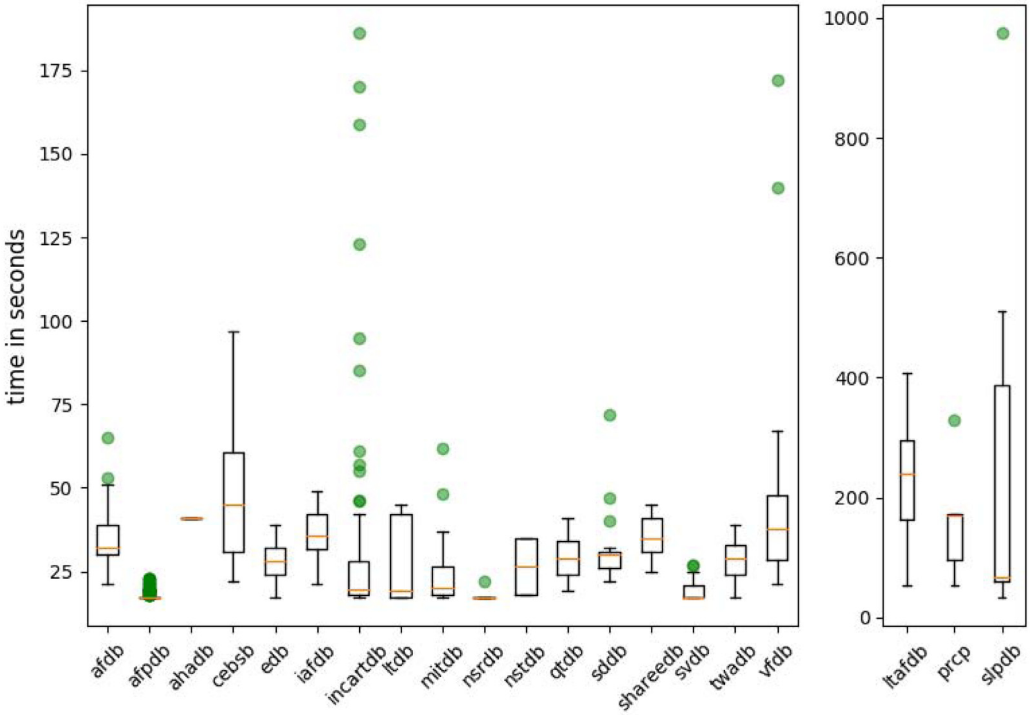

Fig. 9. Time needed to generate a 64-bit token.

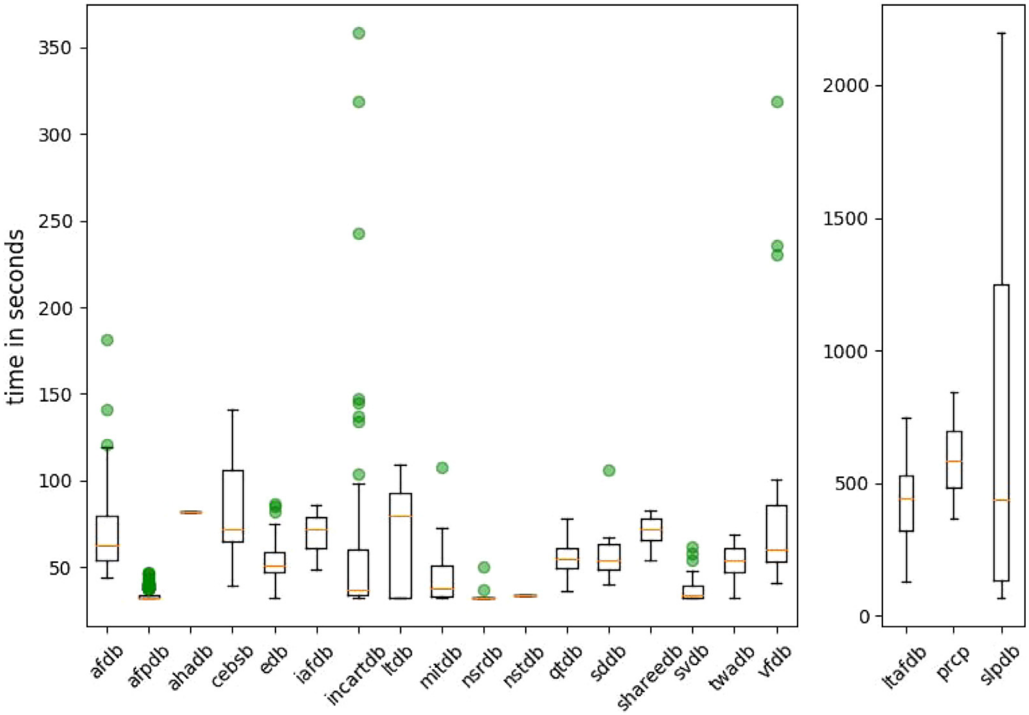

Fig. 10. Time needed to generate a 128-bit token.

have been gathered by the sensors, they are sent to the gateway in order to be synchronized by using the timed-automation (see Section 3.3.1 and Fig. 7). After that, the already synchronized signals are sent back to the sensors, the peak extractor procedure (Algorithm 1) is executed to extract the tokens, and finally the fuzzy extractor (see Section 3.2.2 and Fig. 5) is applied to the processed signals in order to generate the same cryptographic token.

We have basically deployed the scheme presented in Fig. 11 by using a low-cost hardware dedicated for research purposes named BITalino.[4] This shield has two ECG channels and a Bluetooth connection. As in the Alice example, we had to pair our

[4] http://bitalino.com/en/.

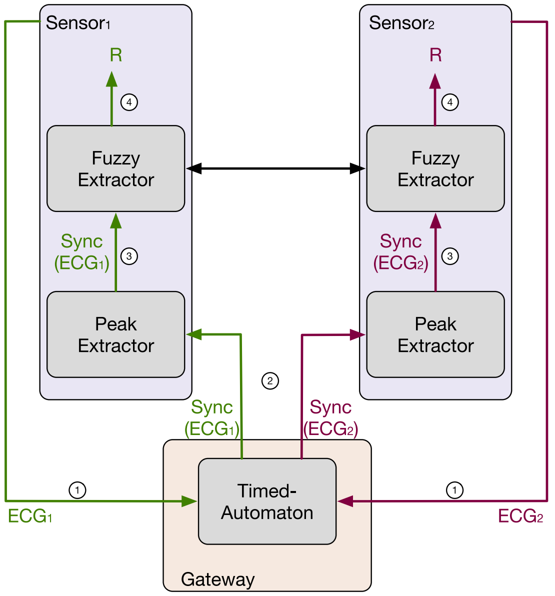

Fig. 11. System model using both a timed automaton and a fuzzy extractor.

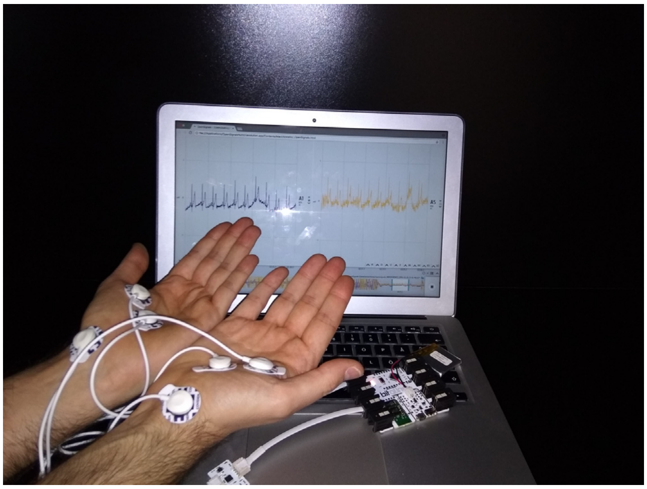

Fig. 12. Proof-of-concept based on the BITalino platform.

hardware with the gateway which, in our case, was a laptop as can be seen in Fig. 12.

More concretely, it works as follows: (1) First, sensors measure the heart signal and send the gathered ECGs to the gateway via a secure communication channel. Once the signal is received by the gateway, the run-time monitor is executed in order to synchronize both signals. (2) After the signals are synchronized, the sensors receive the position of the peaks that must be removed by the gateway. For this communication to occur a secure channel is also needed. (3) When the list of peaks to be removed is received, the sensors proceed to delete them – note that at this point both signals are synchronized in terms of R-peaks – in order to proceed with the token generation. (4) Finally, a fuzzy extractor is applied to the processed signal in order to generate the same token. The helper data can be transmitted from one sensor (generation process) to the other one (reproduction process) without the necessity of a secure channel.

It is worth mentioning that the gateway cannot generate a token by itself to be used in the WBAN; its role is to synchronize the signal and thus helping the sensors to generate a shared token. However, at the end of this protocol, not only the sensors can generate the same token but also the gateway may do it. The source code of our implementation can be downloaded from https://github.com/aylara/synchro.

4.1. Security analysis

In order to solve the problem of creating the same token in two sensors, two aspects need to be addressed: (i) a secure communication channel (sensor(s)–gateway) must be used, and; (ii) sensors have to share their gathered ECGs with the gateway to synchronize them.

Traditionally, authors have assumed that different sensors can extract the same token by measuring the same signal from an organ (in our case the heart). Most authors rely on the fact that, in order to derive the same token, the (active) attacker must be reading the ECG of the bearer at the same time as the devices are and such probability is almost negligible [88]. On the other hand, other proposals (e.g., [8,20]) assumed that the communication can only be established if the devices are close enough (commonly named as neighborhood area): the attacker should be a few centimeters from them and would be easily detected. In these security protocols, the ECG is used to derive a common secret between different sensors and thus, the ECG can be considered as the secret key. The signal must therefore be transmitted over a secure communication channel.

A more recent approach was presented in [81]. In this work, authors use the ECG to securely distribute symmetric cryptographic keys. However, authors use a trusted central node in charge of establishing a secure communication between sensors that want to share some data. In this approach all the devices (two sensors and the gateway) are on the body as they need to record the ECG at the same time and by themselves – in our case only the two sensors have to collect the ECG signal – and the secure channel is established by using a fuzzy commitment scheme [59]. In this work, we demonstrated that only by using a fuzzy commitment scheme is not enough to generate the same token in different devices. In [81] and our proposal, once the cryptographic token synchronized between the sensors is established, they can exchange information directly without involving anyone else. For instance, after the setup phase, smartwatches, wrist-bands and IMD can securely exchange data with each other regardless of the brand, manufacturer or the purpose of the device.

It is important to note that our proof-of-concept implementation is secure if and only if the communication channel between the sensors and the gateway is secure (like most of the proposed solutions in this field). This is because both sensors are sending the ECG to the gateway and thus, an attacker ($\mathcal{A}$) can sniff the communication channel, extract the signals and perform the matching operation. The only extra information that $\mathcal{A}$ would need is the specific instant the tokens have started to be computed. It is also remarkable that the secure channel requirement is only used for the very first time (set-up phase of the protocol); once the two ECG signals are synchronized, there is no longer any need to keep the channel secure.

In this paper we empirically demonstrated that the assumption of two sensors deriving the same token from the heart (ECG signal), is at least questionable. We showed that, in addition to the error correction techniques, a new step is needed before extracting such a token: the synchronization of the signal. To achieve this, there are two options: (1) one of the sensors sends the ECG to the other one in order to synchronize the signal and the latter sends the synchronized signal back to the first one, or; (2) a trusted and external party is used to synchronize the signal and communicate the final decision to the sensors. Either way, the main consequence from a security point of view is that now Eve—a passive attacker, just by eavesdropping on the communication channel might synchronize both ECG signals and extract a common key. To combat this we proposed two main approaches: (1) assume a secure channel in the set-up phased, or; (2) assume that the channel is insecure all the time and use some protection mechanism such as solutions based on noisy cryptography [83].

5. Related work

Several studies have been done in the area of security and privacy applied to biometrics, and in particular where heart signals are involved (e.g., [8,9,19]] In most of these works, there are three main assumptions: (1) bits extracted from the heart signal can be considered random [8,10]; (2) two sensors placed in the same body can generate the same random token from the heart signal [11,18,20,78], and; (3) two sensors should gather 32 consecutive peaks in order to generate a 128 bits nonce which is approximately a 32 s signal [13,19]. As far as we are concerned, this work is the first one that empirically demonstrates that the usual assumptions made in the aforementioned papers regarding the token generation in different devices at the same time are at least questionable.

On the one hand, it is usually assumed that the random numbers derived from IPIs can be directly used on cryptographic applications because of the high entropy degree that the 4 LSBs have. Additionally, some researchers have tried to improve the strength of the entropy per IPI in order to guarantee a higher security level [10,15,18]. However, Ortiz et al. questioned the entropy quality of the IPI values and the dependence of the results on the dataset used [26]. In our work, the involvement of the two sensors is necessary because thanks to this (and the acquisition of the same IPI values derived from the recorded ECGs) they mutually verify they are close to each other (i.e., that they are in the neighborhood area as is commonly named in distance bounding protocols) and additionally they can authenticate each other. Therefore, this kind of distance checking (and mutual authentication if necessary) is made by the participation of both sensors (on the same organ).

On the other hand, researchers have usually assumed that a person equipped with different heart sensors can extract the same nonce from the ECG by using a fuzzy extractor (see [80] for a comparison between fuzzy commitment and fuzzy vault schemes). For example, authors in [8] propose a security protocol where a patient equipped with an IMD and a doctor with a Programmer can extract the same nonce from the patient's ECG. Similar assumptions are made in [9,27,78], just to cite a few of them. Contrarily to any prior proposals, in our work we have demonstrated that error correction algorithms, such as fuzzy extractors, are not enough to claim that sensors placed in different parts of the body can generate the same token using an IPI-based approach.

Recently, some authors have taken into account that some events may occur during the measurement process that increase the difference between the generated tokens. For instance, it is possible that noise appear in the extraction of the signal and the detection of heart peaks will be affected [20]. Because of that, [20] proposes a mechanism to statistically calculate where a peak should be in the ECG signal and they manually add it so that the entropy degree is not affected. Also, other parameters such as HRV and VAR_{is} can alter the peak detection algorithm [21] and they must be taken into account to ensure that the keys are equal enough. In this work, we not only take those issues into consideration but also the heart signal is never modified so that other computations can be applied over the signal such as medical checks of the heart.

Finally, it was stated in [8,11,27,28], that sensors have to keep listening the ECG for about 30 s to generate a 128 bits token. In our work we have proved that in order to generate a 128 bits token, two sensors should be reading the heart signal for almost 1 min (56.6 s) on median, time which we have obtained experimentally and it is in general much larger that it was previously claimed.

6. Conclusion

In this paper we tested whether error corrections algorithms, including fuzzy extractors, can be used alone to claim that two different sensors are able to derive equal tokens from two ECG signals measured at different parts of the body by using an IPI-based approach as proposed in many previous works [8,10,11,15,27]. We run the experiments against 19 public databases from Physionet repository, and we can clearly conclude that a pre-processing of the heart signal is mandatory for generating the same token. Because of that, we proposed a run-time monitor, based on a timed automaton, to synchronize both ECG signals before the peaks are computed and before the fuzzy extractor takes place. Finally, we run once again the same experiments and errors are reduced to zero in many of the tested databases.

Additionally, we also conducted one more experiment to check how long the sensors should record the heart signal in order to generate tokens of 32, 64 and 128 bits and, contrarily to what it is usually assumed (6, 12, and 24 s for individual with a heart rate of 80 bpm), the sensors have to wait 13, 28 and 56.5 s on median respectively to derive the same token from two ECG sensors.

Future work. There have been two main decisions we made which might be improved in the future.

Timed Automaton. The mean value has been used as the upper-bound time for RR intervals as it has been previously proposed in medial research [73], and the consequence is that certain peaks are missed and the run-time monitor does not synchronize as many peaks as it might do. We plan to further research on this line and refine our timed automaton to achieve better results, specially in those databases that do not perform as well.

Extend to other biometrical signals. We have focused on heart signals by using two channels (ECG_1 and ECG_2). We plan to extend our analysis and proposal to other physiological signals like Photoplethysmograms (PPGs), Blood Pressure (BP) or even using the Electroencephalograms (EEGs), as proposed in [89].

References

[1] H. Hamidi, An approach to develop the smart health using internet of things and authentication based on biometric technology, Future Gener. Comput. Syst. 91 (2019) 434–449.

[2] S. Kumar, S.K. Singh, Monitoring of pet animal in smart cities using animal biometrics, Future Gener. Comput. Syst. 83 (2018) 553–563.

[3] J. Wayman, A. Jain, D. Maltoni, D. Maio, An Introduction to Biometric Authentication Systems, 2005, pp. 1–20.

[4] A.M. Intelligence, The global biometrics and mobility report: the convergence of commerce and privacy, in: Tech. rep., Acuity Market Intelligence, 2015, URL http://www.acuity-mi.com/GBMRIntroPreview.pdf.

[5] A. Eng, L.A. Wahsheh, Look into my eyes: a survey of biometric security., in: S. Latifi (Ed.), ITNG, 2013, pp. 422–427.

[6] T. Denning, A. Borning, B. Friedman, B.T. Gill, T. Kohno, W.H. Maisel, Patients, pacemakers, and implantable defibrillators: human values and security for wireless implantable medical devices, in: Conference on Human Factors in Computing Systems, 2010, pp. 917–926.

[7] S.D. Greenwald, R.S. Patil, R.G. Mark, Improved detection and classification of arrhythmias in noise-corrupted electrocardiograms using contextual information, in: [1990] Proceedings Computers in Cardiology, 1990, pp. 461–464.

[8] M. Rostami, A. Juels, F. Koushanfar, Heart-to-heart (h2h): authentication for implanted medical devices, in: CCS '13, 2013, pp. 1099–1112.
[9] C.C.Y. Poon, Y.-T. Zhang, S.-D. Bao, A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health, IEEE Commun. Mag. 44 (4) (2006) 73–81.
[10] R.M. Seepers, C. Strydis, I. Sourdis, C.I.D. Zeeuw, Enhancing heart-beat-based security for mhealth applications, IEEE J. Biomed. Health Inf. 21 (1) (2017) 254–262.
[11] S.D. Bao, C.C.Y. Poon, Y.T. Zhang, L.F. Shen, Using the timing information of heartbeats as an entity identifier to secure body sensor network, IEEE Trans. Inf. Technol. Biomed. 12 (6) (2008) 772–779.
[12] C. Camara, P. Peris-Lopez, J.E. Tapiador, Human identification using compressed ECG signals, J. Med. Syst. 39 (11) (2015) 148.
[13] G.H. Zhang, C.C.Y. Poon, Y.T. Zhang, Analysis of using interpulse intervals to generate 128-bit biometric random binary sequences for securing wireless body sensor networks, IEEE Trans. Inf. Technol. Biomed. 16 (1) (2012) 176–182.
[14] I. Vasyltsov, C. Bak, Method for seamless unlock function for mobile applications, in: 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2016, pp. 2614–2617.
[15] R.M. Seepers, C. Strydis, I. Sourdis, C.I.D. Zeeuw, On using a von neumann extractor in heart-beat-based security, in: 2015 IEEE Trust-com/BigDataSE/ISPA, Vol. 1, 2015, pp. 491–498.
[16] L. González-Manzano, J.M. de Fuentes, P. Peris-Lopez, C. Camara, Encryption by heart (ebh)—using ecg for time-invariant symmetric key generation, Future Gener. Comput. Syst. 77 (2017) 136–148.
[17] D.K. Altop, A. Levi, V. Tuzcu, Deriving cryptographic keys from physiological signals, Pervasive Mob. Comput. 39 (2017) 65–79.
[18] I. Vasyltsov, S. Lee, Entropy extraction from bio-signals in healthcare iot, in: IoTPTS, 2015, pp. 11–17.
[19] G. Zheng, G. Fang, R. Shankaran, M. Orgun, J. Zhou, L. Qiao, K. Saleem, Multiple ecg fiducial points based random binary sequence generation for securing wireless body area networks, IEEE J. Biomed. Health Inf. PP (99) (2016) 1–1.
[20] R.M. Seepers, C. Strydis, P. Peris-Lopez, I. Sourdis, C.I.D. Zeeuw, Peak misdetection in heart-beat-based security: characterization and tolerance, in: 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2014, pp. 5401–5405.
[21] W.-H. Lin, D. Wu, C. Li, H. Zhang, Y.-T. Zhang, Comparison of heart rate variability from ppg with that from ecg, in: ICHI 2013, 2014, pp. 213–215.
[22] L. Yao, B. Liu, G. Wu, K. Yao, J. Wang, A biometric key establishment protocol for body area networks, Int. J. Distrib. Sens. Netw. (2011).
[23] R.M. Seepers, J.H. Weber, Z. Erkin, I. Sourdis, C. Strydis, Secure key-exchange protocol for implants using heartbeats, in: Proceedings of the ACM International Conference on Computing Frontiers, 2016, pp. 119–126.
[24] S.-D. Bao, L.-F. Shen, Y.-T. Zhang, A novel key distribution of body area networks for telemedicine, in: IEEE International Workshop on Biomedical Circuits and Systems, 2004., 2004, pp. 1–17.
[25] S. Cherukuri, K.K. Venkatasubramanian, S.K.S. Gupta, Biosec: a biometric based approach for securing communication in wireless networks of biosensors implanted in the human body, in: 2003 International Conference on Parallel Processing Workshops, 2003. Proceedings., 2003, pp. 432–439.
[26] L. Ortiz-Martin, P. Picazo-Sanchez, P. Peris-Lopez, J. Tapiador, Heartbeats do not make good pseudo-random number generators: an analysis of the randomness of inter-pulse intervals, Entropy 20 (2) (2018).
[27] F. Xu, Z. Qin, C.C. Tan, B. Wang, Q. Li, Imdguard: securing implantable medical devices with the external wearable guardian, in: IEEE INFOCOM, 2011, pp. 1862–1870.
[28] S.R. Moosavi, E. Nigussie, S. Virtanen, J. Isoaho, Cryptographic key generation using ecg signal, in: 2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC), 2017, pp. 1024–1031.
[29] S.M.R. Islam, D. Kwak, M.H. Kabir, M. Hossain, K.S. Kwak, The internet of things for health care: a comprehensive survey, IEEE Access 3 (2015) 678–708.
[30] M. Ambigavathi, D. Sridharan, Energy efficient and load balanced priority queue algorithm for wireless body area network, Future Gener. Comput. Syst. 88 (2018) 586–593.
[31] A.L. Goldberger, L.A.N. Amaral, L. Glass, J.M. Hausdorff, P.C. Ivanov, R.G. Mark, J.E. Mietus, G.B. Moody, C.K. Peng, H.E. Stanley, Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals, Circulation 101 (23) (2000) e215–e220.
[32] N. Iyengar, C. Peng, R. Morin, A.L. Goldberger, L.A. Lipsitz, Age-related alterations in the fractal scaling of cardiac interbeat interval dynamics, Am. J. Physiol.-Regul. Integr. Comparative Physiol. 271 (4) (1996) R1078–R1084.
[33] T. Penzel, G.B. Moody, R.G. Mark, A.L. Goldberger, J.H. Peter, The apnea-ecg database, in: Computers in cardiology 2000, 2000, pp. 255–258.
[34] G. Moody, R. Mark, A new method for detecting atrial fibrillation using R-R intervals, Computers in Cardiology.
[35] G. Moody, A. Goldberger, S. McClennen, S. Swiryn, Predicting the onset of paroxysmal atrial fibrillation: the computers in cardiology challenge 2001, in: Computers in Cardiology 2001. Vol.28 (Cat. No.01CH37287), 2001, pp. 113–116.
[36] E. Institute, AHA Database Sample Excluded Record, URL https://physionet.org/physiobank/database/ahadb/, 2018.
[37] M.A. García-González, A. Argelagós-Palau, M. Fernández-Chimeno, J. Ramos-Castro, A comparison of heartbeat detectors for the seismocardiogram, in: Computing in Cardiology Conference (CinC), 2013, 2013, pp. 461–464.
[38] A. Taddei, G. Distante, M. Emdin, P. Pisani, G. Moody, C. Zeelenberg, C. Marchesi, The european st-t database: standard for evaluating systems for the analysis of st-t changes in ambulatory electrocardiography, Eur. Heart J. 13 (9) (1992) 1164–1172.
[39] Physionet, Intracardiac Atrial Fibrillation Database, URL https://physionet.org/physiobank/database/iafdb/, 2018.
[40] Physionet, St.-Petersburg Institute of Cardiological Technics 12-lead Arrhythmia Database, URL https://physionet.org/physiobank/database/incartdb/, 2018.
[41] S. Petrutiu, A.V. Sahakian, S. Swiryn, Abrupt changes in fibrillatory wave characteristics at the termination of paroxysmal atrial fibrillation in humans, Europace 9 (7) (2007) 466–470.
[42] G.B. Moody, R.G. Mark, The impact of the mit-bih arrhythmia database, Eng. Med. Biol. Mag., IEEE 20 (3) (2001) 45–50.
[43] Physionet, The MIT-BIH Normal Sinus Rhythm Database, URL https://physionet.org/physiobank/database/nsrdb/, 2018.
[44] G.B. Moody, W. Muldrow, R.G. Mark, A noise stress test for arrhythmia detectors, Comp. Cardiol. 11 (3) (1984) 381–384.
[45] P. Laguna, R. Mark, A. Goldberg, G. Moody, Database for evaluation of algorithms for measurement of QT and other waveform intervals in the ecg, in: Computers in Cardiology, Vol. 1997, 1997, pp. 673–676.
[46] S.D. Greenwald, The Development and Analysis of a Ventricular Fibrillation Detector (Ph.D. thesis), Massachusetts Institute of Technology, 1986.
[47] P. Melillo, R. Izzo, A. Orrico, P. Scala, M. Attanasio, M. Mirra, N. De Luca, L. Pecchia, Automatic prediction of cardiovascular and cerebrovascular events using heart rate variability analysis, PLoS One 10 (3) (2015) e0118504.
[48] Y. Ichimaru, G. Moody, Development of the polysomnographic database on cd-rom, Psychiatry Clin. Neurosci. 53 (2) (1999) 175–177.
[49] T.P..C.i.C.C. Moody GB, T-wave alternans. computers in cardiology, Neurology 35 (2008) 505–508.
[50] S.D. Greenwald, The deveLopment and Analysis of a Ventricular Fibrillation Detector (Ph.D. thesis), Massachusetts Institute of Technology, 1986.
[51] C. Vastarouchas, S. Kapoulea, C. Psychalinos, Ecg signal acquisition for the pan-tompkins algorithm using current-mirror filters, in: IEEE International Conference on Electronics, Circuits and Systems, 2016, pp. 317–320.
[52] S. Peter, B. Pratap Reddy, F. Momtaz, T. Givargis, Design of secure ecg-based biometric authentication in body area sensor networks, Sensors 16 (4).
[53] S. Pirbhulal, H. Zhang, W. Wu, S.C. Mukhopadhyay, Y. Zhang, Heart-beats based biometric random binary sequences generation to secure wireless body sensor networks, IEEE Trans. Biomed. Eng. (2018) 1–1.
[54] R. Altawy, A.M. Youssef, Security tradeoffs in cyber physical systems: A case study survey on implantable medical devices, IEEE Access.
[55] G. Chen, Are electroencephalogram (eeg) signals pseudo-random number generators? J. Comput. Appl. Math. 268 (2014) 1–4.
[56] B. Petchlert, H. Hasegawa, Using a low-cost electroencephalogram (eeg) directly as random number generator, in: Advanced Applied Informatics (IIAIAAI), 2014 IIAI 3rd International Conference on, 2014, pp. 470–474.
[57] J. Szczepanski, E. Wajnryb, J. Amigó, M.V. Sanchez-Vives, M. Slater, Biometric random number generators, Comput. Secur. 23 (1) (2004) 77–84.
[58] K. Venkatasubramanian, Venkatasubramanian, A. Banerjee, S.K.S. Gupta, Ekg-based key agreement in body sensor networks, in: IEEE INFOCOM Workshops, 2008, pp. 1–6.
[59] A. Juels, M. Wattenberg, A fuzzy commitment scheme, in: Conference on Computer and Communications Security, 1999, pp. 28–36.
[60] L.M. Dinca, G. Hancke, User-centric key entropy: Study of biometric key derivation subject to spoofing attacks, Entropy 19 (2).
[61] H. Kang, Y. Hori, T. Katashita, M. Hagiwara, K. Iwamura, Performance analysis for puf data using fuzzy extractor, in: Y.-S. Jeong, Y.-H. Park, C.-H.R. Hsu, J.J.J.H. Park (Eds.), Ubiquitous Information Technologies and Applications, 2014, pp. 277–284.
[62] N. Li, F. Guo, Y. Mu, W. Susilo, S. Nepal, Fuzzy extractors for biometric identification, in: 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 667–677.
[63] C. Herder, L. Ren, M. van Dijk, M.D. Yu, S. Devadas, Trapdoor computational fuzzy extractors and stateless cryptographically-secure physical unclonable functions, IEEE Trans. Dependable Secure Comput. 14 (1) (2017) 65–82.
[64] A. Schaller, T. Stanko, B. Škorić, S. Katzenbeisser, Eliminating leakage in reverse fuzzy extractors, IEEE Trans. Inf. Forensics Secur. 13 (4) (2018) 954–964.

[65] X. Li, J. Liu, Q. Yao, J. Ma, Efficient and consistent key extraction based on received signal strength for vehicular ad hoc networks, IEEE Access 5 (2017) 5281–5291.
[66] Y. Dodis, L. Reyzin, A. Smith, Fuzzy extractors: how to generate strong keys from biometrics and other noisy data, in: EUROCRYPT, 2004, pp. 523–540.
[67] I. Natgunanathan, A. Mehmood, Y. Xiang, G. Beliakov, J. Yearwood, Protection of privacy in biometric data, IEEE Access 4 (2016) 880–892.
[68] R. Alur, T.A. Henzinger, Logics and models of real time: a survey, in: Real-Time: Theory in Practice, REX Workshop, Vol. 600, 1991, pp. 74–106.
[69] R. Alur, D.L. Dill, A theory of timed automata, Theoret. Comput. Sci. 126 (2) (1994) 183–235.
[70] G. Behrmann, A. David, K.G. Larsen, J. Hakansson, P. Petterson, W. Yi, M. Hendriks, Uppaal 4.0, in: Proceedings of the 3rd International Conference on the Quantitative Evaluation of Systems, 2006, pp. 125–126.
[71] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, S. Yovine, Kronos: a model-checking tool for real-time systems, in: Computer Aided Verification, Vol. 1427, 1998, pp. 546–550.
[72] R. Alur, T.A. Henzinger, Real-time logics: complexity and expressiveness, Inform. and Comput. 104 (1) (1993) 35–77.
[73] M.P. Tulppo, T. Makikallio, T. Takala, T. Seppanen, H.V. Huikuri, Quantitative beat-to-beat analysis of heart rate dynamics during exercise, Am. Physiol.-heart Circulatory Physiol. 271 (1) (1996) H244–H252.
[74] A. Calleja, P. Peris-Lopez, J.E. Tapiador, Electrical heart signals can be monitored from the moon: security implications for ipi-based protocols, in: R.N. Akram, S. Jajodia (Eds.), Information Security Theory and Practice, 2015, pp. 36–51.
[75] C. Camara, P. Peris-Lopez, L. Gonzalez-Manzano, J. Tapiador, Real-time electrocardiogram streams for continuous authentication, Appl. Soft Comput. 68 (2018) 784–794.
[76] J. Pan, W.J. Tompkins, A real-time qrs detection algorithm, IEEE Trans. Biomed. Eng. BME-32 (3) (1985) 230–236.
[77] Y.S. Han, BCH Codes, in: Graduate Institute of Communication Engineering, National Taipei University, 2016.
[78] G. Zheng, G. Fang, R. Shankaran, M.A. Orgun, Encryption for implantable medical devices using modified one-time pads, IEEE Access 3 (2015) 825–836.
[79] T. Bai, J. Lin, G. Li, H. Wang, P. Ran, Z. Li, D. Li, Y. Pang, W. Wu, G. Jeon, A lightweight method of data encryption in bans using electrocardiogram signal, Future Gener. Comput. Syst. 92 (2019) 800–811.
[80] G. Zheng, G. Fang, M.A. Orgun, R. Shankaran, A comparison of key distribution schemes using fuzzy commitment and fuzzy vault within wireless body area networks, in: PIMRC, 2015, pp. 2120–2125.
[81] A. Sammoud, O. Hamdi, M.A. Chalouf, A. Bouallegue, A new protocol for an efficient and green biometric-based security key establishment in wbans, in: IWCMC, 2018, pp. 762–767.
[82] Z. Zhang, H. Wang, A.V. Vasilakos, H. Fang, Ecg-cryptography and authentication in body area networks, IEEE Trans. Inf. Technol. Biomed. 16 (6) (2012) 1070–1078.
[83] P. Tuyls, B. Skoric, T. Kevenaar, Security with Noisy Data: Private Biometrics, Secure Key Storage and Anti-Counterfeiting, Springer-Verlag, 2007.
[84] W. Wu, S. Pirbhulal, A.K. Sangaiah, S.C. Mukhopadhyay, G. Li, Optimization of signal quality over comfortability of textile electrodes for ecg monitoring in fog computing based medical applications, Future Gener. Comput. Syst. 86 (2018) 515–526.
[85] A. Carman, Withings' new smartwatch has an EKG sensor to compete with the Apple Watch, URL https://goo.gl/tbrmQz, 2019.
[86] C. Carreiras, A. Lourenço, A. Fred, R. Ferreira, ECG Signals for biometric applications – are we there yet? in: Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics, SciTePress, 2014, pp. 765–772.
[87] B. Secure, ECG Biometrics for the Connected Car (White Paper), URL https://goo.gl/mXC8xG, 2018.
[88] E.K. Zaghouani, A. Jemai, A. Benzina, R. Attia, Elpa: a new key agreement scheme based on linear prediction of ecg features for wban, in: 2015 23rd European Signal Processing Conference (EUSIPCO), 2015, pp. 81–85.
[89] P. Bagade, A. Banerjee, J. Milazzo, S.K.S. Gupta, Protect your bsn: No handshakes, just Namaste!, in: 2013 IEEE International Conference on Body Sensor Networks, 2013, pp. 1–6.

Lara Ortiz-Martin is a PhD. student in Computer Science at Universidad Carlos III de Madrid. She received a MSc. degree in Computer Science from the same university. Her current research interests include systems security, applied cryptography and biometrics. She is also working on web technologies in the industry.

Pablo Picazo-Sanchez received the PhD. degree in Computer Science from the Carlos III University of Madrid, in 2016. Currently he works at Chalmers University, Sweden, in a postdoc position in the Formal Methods division. His current research interests include systems security, applied cryptography and web security.

Pedro Peris-Lopez holds an Associate Professor position at Universidad Carlos III de Madrid. He has a M.Sc. in Telecommunications Engineering (2004) and Ph.D. in Computer Science (2008) from University Carlos III of Madrid. His research interests are in the fields of cryptography, computer forensics, signal processing, and artificial intelligence. Nowadays, his research is mainly focused on Implantable Medical Devices (IMD) and Biomedical applications. He has published many articles (40) in International Journals with impact factor and papers (41) in International Conferences of recognized prestige (peer- reviewed; 2–4 reviewers). His works have a high impact: the whole of his works have more than 3341 cites and his h-index is 26 (04/2018 - Google Scholar). For additional information see: www.lightweightcryptography.com

Juan Tapiador is Associate Professor of Computer Science in the Com- puter Security (COSEC) Lab at Universidad Carlos III de Madrid, Spain. His main research interests include systems security, malware analysis, reverse en- gineering, and anomaly and intrusion detection.

Gerardo Schneider received a PhD degree in Computer Science from the University Joseph Fourier (thesis done at the VERIMAG laboratory), Grenoble (France), in 2002. From 2003 till 2009 he was a researcher at Uppsala University (Sweden), Irisa/INRIA Rennes (France), and the University of Oslo (Norway). He joined the Department of Computer Science and Engineering at the Univer- sity of Gothenburg (Sweden) in July 2009, where he has been a full professor since July 2014. His research interests include formal verification (static and runtime verification, model checking, testing), the specification and analysis of normative documents, and security & privacy.