Peak Misdetection In Heart-Beat-Based Security: Characterization and Tolerance

Robert M. Seepers[1]†, Christos Strydis[1]†, Pedro Peris-Lopez[2], Ioannis Sourdis[3]†, and Chris I. De Zeeuw[1]†
[1]Dept. of Neuroscience, Erasmus Medical Center, Rotterdam, The Netherlands
[2]Dept. of Computer Science, Universidad Carlos III de Madrid, Madrid, Spain
[3]Dept. of Computer Science & Engineering, Chalmers University of Technology, Gothenburg, Sweden
[1]{r.seepers, c.strydis, c.dezeeuw}@erasmusmc.nl [2]pperis@inf.uc3m.es [3]sourdis@chalmers.se

Abstract— **The Inter-Pulse-Interval (IPI) of heart beats has previously been suggested for security in mobile health (mHealth) applications. In IPI-based security, secure communication is facilitated through a security key derived from the time difference between heart beats. However, there currently exists no work which considers the effect on security of imperfect heart-beat (peak) detection. This is a crucial aspect of IPI-based security and likely to happen in a real system. In this paper, we evaluate the effects of peak misdetection on the security performance of IPI-based security. It is shown that even with a high peak detection rate between 99.9% and 99.0%, a significant drop in security performance may be observed (between -70% and -303%) compared to having perfect peak detection. We show that authenticating using smaller keys yields both stronger keys as well as potentially faster authentication in case of imperfect heart beat detection. Finally, we present an algorithm which tolerates the effect of a single misdetected peak and increases the security performance by up to 155%.**

I. INTRODUCTION

Mobile-health (mHealth) is an emerging technology which allows for continuous, remote health care through the use of mobile devices. Body-Area Networks (BANs) may provide continuous patient monitoring through the use of cheap, wearable biosensors [9]. Modern Implantable Medical Devices (IMDs) feature wireless capabilities to allow remote configuration without requiring invasive surgery or data-log broadcasting from a home-monitoring station [3]. Due to the wireless nature of mHealth solutions and the sensitivity of the data transmitted, security has shown to be an important aspect of mHealth. Non-secure communication may allow an adversary to steal private patient data or, worse, alter device parameters or even prevent treatment [2], [9].

The inter-pulse interval (IPI) of heart beats has recently been proposed for securing both wireless IMDs and BANs [15], [13], [12]. In IPI-based security, each sensor measures a heart-related biosignal, for example, cardiac activity using an electrocardiogram (ECG) or blood flow, and forms a security key based on the time difference between successive heart beats. Previous work has shown that this time difference contains a significant degree of entropy, while may be measured remarkably consistent on different locations of a patient's body [12]. These two characteristics allow IPIs to be used as a basis for security aspects such as key agreement or entity authentication.

To the best of our knowledge, there is currently no work which characterizes the security performance (key strength and authentication rate) in case one sensor does not correctly detect the same peaks as another sensor. Peak misdetection may occur due to, among others, the presence of noise in biosignals, preventing an entity from detecting a peak or falsely detecting a non-existent peak [5], [10], [6]. This may lead to a disparity between the generated keys and will, in effect, impact security. Evaluating the security performance as a function of peak-detection rate provides insights into the required peak-detection performance of a sensor in the context of mHealth security.

The remainder of this paper is structured as follows: First, we briefly discuss works related to the generation of security keys using IPIs in Section II. The security performance will be evaluated as a function of peak detection rate in Section III, after which we describe an algorithm which improves the security performance in Section IV. Finally, concluding remarks will be given in Section V.

II. RELATED WORK

Previous work has shown that each IPI contains a number of bits with a high degree of entropy and that a security key may be generated by combining a number of subsequent IPIs [12]. This security key may subsequently be used as an entity identifier (EI) [1]. An evaluation of healthy subjects, hypertensive subjects as well as patients with cardio-vascular disorders (CVDs) at rest has revealed that four bits with a high degree of entropy are available per IPI [12], [1], [17], [13]. While IPIs, thus, contain a number of highly entropic bits, they may be measured with minor discrepancies by sensors on the same body (inter-sensor variability) [12], [13], [15]. To deal with this inter-sensor variability, a tolerance margin is required between IPIs measured by two sensors, limiting the security performance.

III. CHARACTERIZATION

Figure 1 shows a common method of providing entity authentication using IPIs. First, each entity (sensor) detects a number of consecutive peaks from their cardiac biosignals and calculates the time interval (IPI) between these peaks.

† This work has been supported by the EU-funded project DeSyRe (Grant agreement no: 287611) and would not have been complete without the invaluable feedback from P. Kaklamanis.

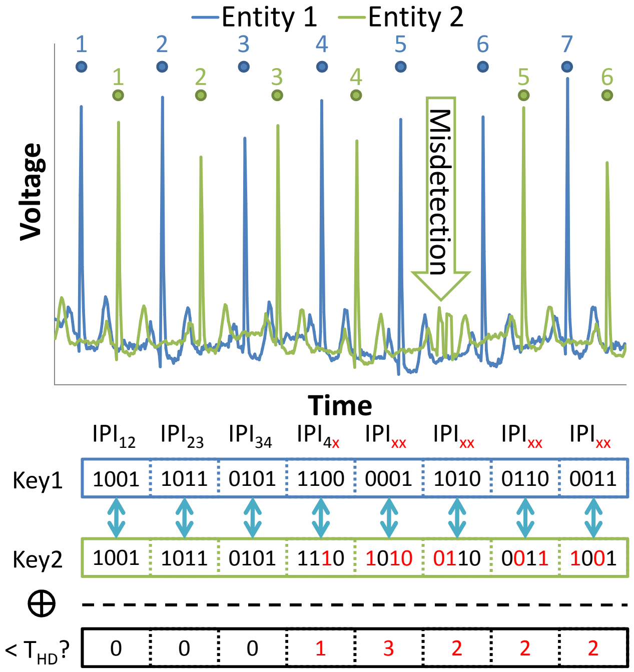
Entity 1
Entity 2
1 2 3 4 5 6 7
1 2 3 4 5 6
Voltage
Misdetection
Time
IPI12 IPI23 IPI34 IPI4x IPIxx IPIxx IPIxx IPIxx
Key1 1001 1011 0101 1100 0001 1010 0110 0011
Key2 1001 1011 0101 1110 1010 0110 0011 1001
⊕
< THD? 0 0 0 1 3 2 2 2

Fig. 1: Key comparison in IPI-based security. A misdetected peak leads to a significant disparity between the keys.

Each entity selects a predefined number of (entropic) bits from each IPI, which is called a key segment and denoted m, and concatenates n key segments to form a security key k. As biosignals are rarely identical, entity authentication is successful if the keys are similar enough, i.e., if the Hamming distance between the keys is smaller than a predefined threshold ($hd(k_1 \oplus k_2) < T_{HD}$, where $hd(x)$ represents the number of non-zero values in x).

Figure 1 illustrates the effect in case one entity misdetects a peak (in this example, entity 2 does not detect peak 5). As a result of this peak misdetection, sensor 2 calculates IPI_{45} using the time difference between peaks 4 and 6, causing a disparity in this key segment. Moreover, this misdetected peak leads to a de-synchronization between the sensors, as the second sensor generates one less IPI than sensor one. This de-synchronization may be observed by comparing key segments $k_1(m+1)$ to $k_2(m)$ for $m = 6, 7, 8$ in the example of Figure 1. A single missed peak may, thus, cause a significant disparity between the two keys.

A. Experimental Setup

For our experiments, we have used the *MIT-BIH arrhythmia dataset*, a commonly used dataset containing subjects with a wide variety of CVDs [11], [7]. The location of the heart-beats ("R" peaks) in the database were detected using an in-house peak-detection algorithm and hand-corrected afterwards to ensure that our baseline data represents a peak detection rate of 100%.

We model the inter-sensor variation using the annotation differences between the ECG and blood-pressure recordings from the *Fantasia dataset* [8]. While previous work has modeled the inter-sensor variation as the time difference between two different leads from the MIT-BIH dataset [13], we consider our model to be more realistic for typical mHealth applications as it takes into account both different biosignals (it is unlikely that all entities have access to the same biosignal) as well as higher inter-sensor variation due to using different measuring equipment. The significant levels of inter-sensor variation in this model prevents us from using the three least-significant bits of each IPI [15].

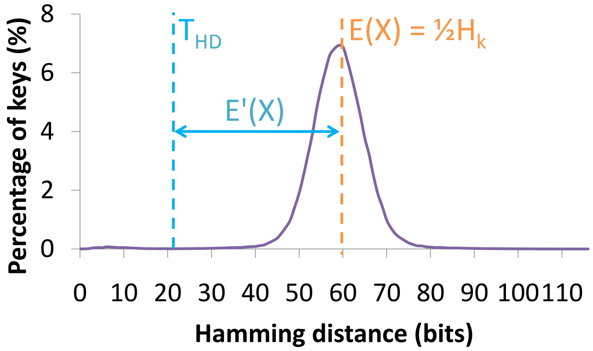

Fig. 2: Key strength as a function of H_k and T_{HD}

The probability of misdetecting a peak, i.e., missing a peak or detecting a non-existent peak, is modeled using a random process with a uniform distribution where peak detections are randomly deleted (or inserted) from the list of peaks of one of the two entities. This generic model allows us to investigate the security performance without relying on a specific peak-detection algorithm. As several peak-detection algorithms report a detection rate of over 99% [5], [6], [10], we evaluate the security performance in terms of entropy, authentication rate (accessibility) and key strength by varying the detection rate from 99% to 100%.

We define the key strength KS as the number of entropic bits which should be known to an attacker in order to successfully authenticate to the IMD with probability $P_{auth} = 0.5$. That is, an attacker would have to mount on average 2^{KS} attacks. KS depends on both the entropy H_k of the generated key and the Hamming-distance threshold T_{HD}. Based on our experiments, in Figure 2 we plot a distribution of Hamming distances between an authentication key and various attacker keys. This distribution $\boldsymbol{X}$ – x being the number of mismatched bits in an n-bit key – is expectedly binomial with an average number of mismatches $E(X) = p_0 n = p_1 n = \frac{n}{2} = \frac{H_k}{2}$. Since, on average, half the number of key bits are mismatched, for successful authentication an attacker would need to try up to:

$$KS = 2 \cdot E(X) - 1 = H_k - 1 \quad bits,$$

the "-1" term accounting for $P_{auth} = 0.5$.

In case a positive T_{HD} is also introduced to the distribution, the average number of mismatched bits would be effectively reduced by the amount of "don't care" T_{HD} bits; essentially $E'(X) = \frac{H_k}{2} - T_{HD}$ (see Figure 2). In this more general case, KS would be calculated as follows:

$$\begin{aligned} KS &= 2 \cdot E'(X) - 1 \\ &= H_k - 2 \cdot T_{HD} - 1 \quad bits. \end{aligned} \tag{1}$$

Consequently, to determine the key strength we have to evaluate the entropy H_k and required Hamming-distance

TABLE I: Entropy results (per bit). Bits 0, 1 and 2 are not usable due to high inter-sensor variation.

Bit #	Entropy results			
	Compr.	Arith. mean	Serial corr.	n_{eq}
0 (LSB)	1.00	1.00	0.99	**1.10**
1	1.00	1.00	1.00	**1.00**
2	1.00	1.00	1.00	**1.00**
3	1.00	1.00	1.00	**1.00**
4	0.98	1.00	0.87	**2.02**
5	0.89	1.00	0.65	**3.80**
6	0.77	1.00	0.50	**5.94**
7	0.60	1.00	0.36	**9.77**

threshold T_{HD}. Without loss of generality, we assume an upper limit of 60 seconds in which entities should authenticate reliably, where we define reliably as successful authentication with probability $1 - 10^{-6}$. Given these constraints, we strive to generate a key which is as secure as possible assuming a fixed, typical heart rate of 60 beats per minute.

B. Evaluation

1) Entropy: The number of fully entropic bits per IPI determines the upper limit H_k of the key strength. To assess the entropy, we model each bit position i within an IPI as a random number generator R_i which generates bits r_i. To assess H_i for each R_i, we use the arithmetic mean, serial correlation and compression test as given by the ENT randomness test suite [16], the results of which are presented in Table I. Conferring with related work, we see that the four least-significant bits (LSBs) of each IPI contain a high degree of entropy, scoring between 0.99 and 1.00 for all tests. Unfortunately, while these bits are highly entropic, our previous work [15] has shown that the three LSBs (bit 0-2) can effectively not be used under our inter-sensor variation model and will, thus, not be considered in this work. From the fifth bit onwards, we can see a steep reduction in available entropy which may be attributed to a higher degree of serial correlation in these bits.

For each bit position i, to quantify the entropy of R_i (in bits), we determine the number n_i^{eq} which, when concatenated from n_i^{eq} IPIs to form one bit string, provide *equivalent randomness* as one bit of entropy [14], i.e., there is a 50% chance of guessing it correctly. For each R_i process, the entropy is defined by Shannon's formula of entropy, $H = -p_0 \cdot log_2(p_0) - p_1 \cdot log_2(p_1)$, which is maximum for $p_0 = p_1 = 0.5$. To obtain n_i^{eq}, we take the most conservative approach by selecting the minimum H_i^{min} value for each R_i in Table I to obtain $p^i_{max} = max(p^i_0, p^i_1)$. By concatenating n_i^{eq} bits, the best chance of guessing all n_i^{eq} bits correctly equals $(p^i_{max})^{n_i^{eq}}$. Accordingly, we obtain the equivalent of one bit of entropy if $n_i^{eq} = log_{p^i_{max}}(\frac{1}{2})$. Table I shows that, as may be expected, the first four bits result in $n_i^{eq} = 1$, i.e., these bits are fully entropic. From the fifth bit onwards, we can see a steep reduction in available entropy, requiring between $n_4^{eq} = 2$ and $n_7^{eq} = 10$ IPIs to generate the equivalent of one bit of entropy. It should be noted that we have found no significant difference in entropy as a function of the peak detection rate and will, therefore, use the results in Table I for both correctly detected and misdetected peaks.

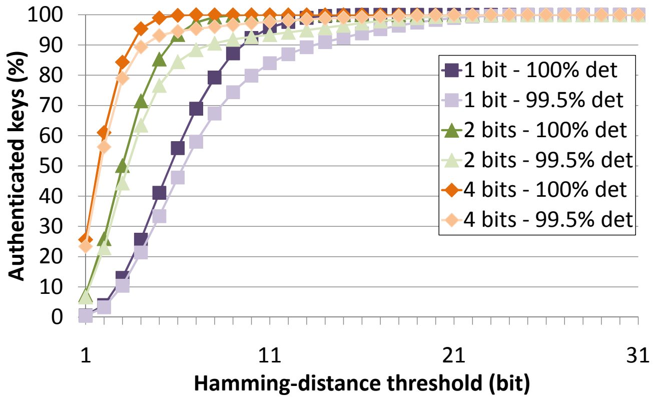

Fig. 3: Authentication rate for a 36-bit key.

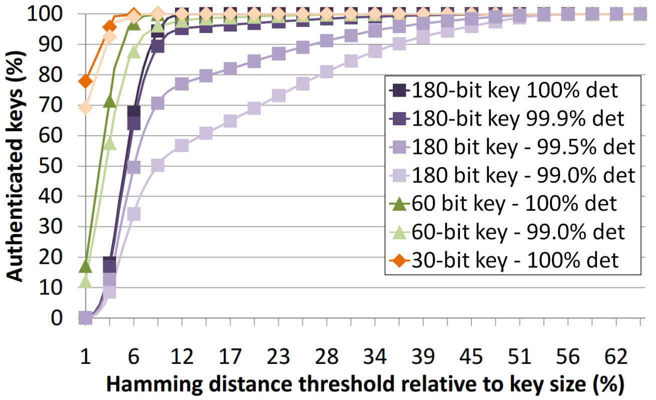

Fig. 4: Authentication rate using three bits (bit 3-5) per IPI.

2) Accessibility: The Hamming-distance threshold T_{HD} should be chosen in such a way that our authentication constraints are met, yet needs to be as small as possible to maximize the key strength. First, in order to evaluate the effect of the peak detection rate and number of bits per IPI on T_{HD}, we vary these parameters while maintaining a fixed key size. Figure 3 depicts the authentication rate for a 36-bit key (other key sizes lead to similar trends, hence are not discussed in detail). In agreement with prior work [15], we find that using more bits per IPI results in a higher authentication rate for a given detection rate. There are two reasons for this: First, more significant bits are less prone to inter-sensor variation, i.e., contribute relatively little to the disparity between two keys. Second, as using more bits per IPI implies using less IPIs for a given key size, there are less noisy bits in total.

From Figure 3, we also note that a decrease in peak-detection-rate results in a decreased number of keys leading to successful authentication for a given T_{HD}, i.e, an increase in T_{HD} is required to maintain the same authentication rate. We can observe that when reducing the number of used bits per IPI, the difference in authentication rate for different detection rates becomes less prominent, i.e., the effect of peak-misdetection decreases. We explain this as follows: Given a certain peak-detection probability P_d, the probability that a misdetection occurs in a key is $P_{md} = 1 - P_d^n$, where n is the number of IPIs used per key. As using more bits

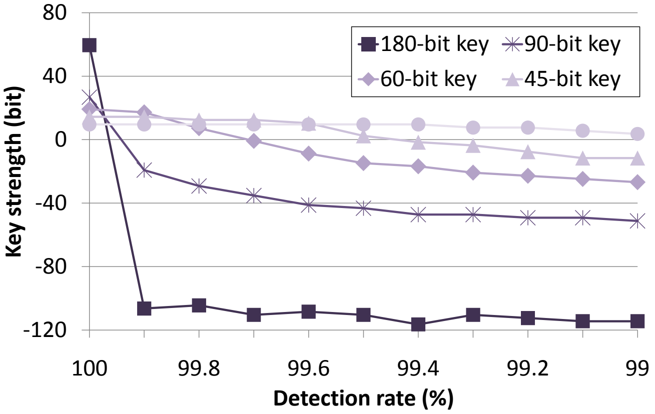

Fig. 5: Key strength as a function of detection rate using three bits (bit 3-5) per IPI.

per IPI results in less samples (i.e. n is reduced), P_{md} is reduced from 16.5% using 1 bit per IPI to 3.9% when using 4 bits per IPI. Moreover, as misdetection results in a desynchronization between two keys, the disparity between the two keys is more significant when misdetection occurs in one of the first IPIs in a key. Similar to P_{md}, the probability of this occurring is higher when fewer bits are used per IPI.

Under our authentication-time constraint of 60 seconds, it is possible to either authenticate using a single key of maximum size (i.e., maximizing the entropy for a single key) or attempt multiple authentications using smaller keys. Figure 4 depicts the percentage of authenticated keys as a function of T_{HD} for various detection rates and key sizes when using three bits per IPI. Note that T_{HD} is presented as a percentage of the total key size, which allows for a more direct comparison of the impact on the key strength for various key sizes. Homologously to Figure 3, we see that using a larger key, i.e., using more IPIs per key, results in a more significant reduction in authentication rate compared to using a smaller key. In particular, a single large key (180-bit) reaches the required authentication reliability of $1-10^{-6}$ using a T_{HD} of 13% to 63% of the keysize for a detection rate of 100% to 99% respectively. Using multiple smaller keys, e.g. three 60-bit keys, this value may be reached using a T_{HD} of 13% and 29% of the keysize for a detection rate of 100% and 99%, respectively. There are two reasons for this drop in T_{HD}: First, the fewer IPIs used per key, the lower P_{md} becomes. Second, the probability of successful authentication within the 60 second time constraint is given as $P_{auth} = 1 - P^{k}_{notauth}$ where $P_{notauth}$ represents the probability a key pair is not authenticated and k is the number of authentication attempts (keys). Consequently, *authenticating using multiple smaller keys provides some tolerance to peak misdetection* as the authentication rate of smaller keys shows smaller variations for a decreasing peak detection rate. As an additional advantage, it is possible that an entity is authenticated sooner than the imposed 60 second constraint as smaller keys may be generated in a shorter timespan.

3) Key strength: Based on the evaluation of entropy and Hamming-distance threshold above, we can now compute the key strength $KS = H_k - 2 \cdot T_{HD} - 1$. For a 100% detection rate, the maximum key strengths (using 60 IPIs) are 13.0, 42.7, 58.6, 68.7 and 74.8 bits, using 1, 2, 3, 4 and 5 bits per IPI, respectively. Without the loss of generality, we will evaluate the effects of peak misdetection using three bits per IPI. Using a different number of bits per IPI leads to very similar trends and is therefore not discussed in detail. Figure 5 depicts the key strength as a function of the peak detection rate for different key sizes. Note that Figure 5 also depicts a number of *negative* key strengths, which indicates that an attacker has to guess less than half of all bits correctly for successful authentication, i.e. it is more likely an attacker will successfully authenticate than not.

First, it can be seen from Figure 5 that for a 100% detection rate, using a single large key is favoured compared to using multiple smaller keys. As the entropy is increased linearly with the key size ($H_k = n \cdot H_m$, where H_m denotes the entropy in a key segment) and T_{HD} remains the same relative to the key size, KS shows a linear increase in entropy. Furthermore, KS is reduced when the detection rate is decreased. This effect is more noticable when using a single large key: Even a minor decrease in peak detection rate (100% to 99.9%) results in a significant drop in KS (KS goes from from 59 bits to -107 bits, i.e., loses 166 bits. Smaller keys, on the other hand, start with a lower KS for a 100% detection rate (due to a lower H_k), but are more tolerant to peak misdetection as we have previously shown that using multiple smaller keys allows for T_{HD} to remain more constant. Whereas the KS of a 180-bit key is decreased by over 170 bits when the detection rate goes from 100% to 99.5%, the KS of a 45-bit key drops from 13.4 to 1.4 bits. By comparing the maximum key strength possible for a given detection rate to that of a single key with a 100% detection rate, we conclude that *the security performance is significantly reduced as a function of peak-detection rate, resulting in a reduction of, at best, -70% to -303% when the detection rate is reduced from 100% to 99%, respectively.*

IV. Tolerating peak misdetection

While utilizing multiple smaller keys shows a reducting in the effect of misdetected peaks, the overall key strength is significantly reduced. Alternatively, we may attempt to reduce the impact of a misdetected peak (on larger keys), to allow for a potentially higher key strength. To the best of our knowledge, we propose the first, novel method which aims to tolerate a misdetected peak for improving the security performance. As has been described in the previous Section, a misdetected peak manifests as a misallignment in key segments. Consequently, we attempt to tolerate a misdetected peak by resolving this misallignment.

Consider a key k with n segments, where a misdetect occurs in key segment m. As a result, the key segments $1..m$ will be alligned correctly, whereas key segments $m+1..n$ will be misalligned by one, i.e., either $k_1(m+1..n)$ matches $k_2(m..n-1)$ or $k_1(m..n-1)$ matches $k_2(m+1..n)$. As we do not know at which key segment the misdetection occurred (i.e., m is unknown), our algorithm evaluates the minimum Hamming distance between two keys $HD_{min} =$

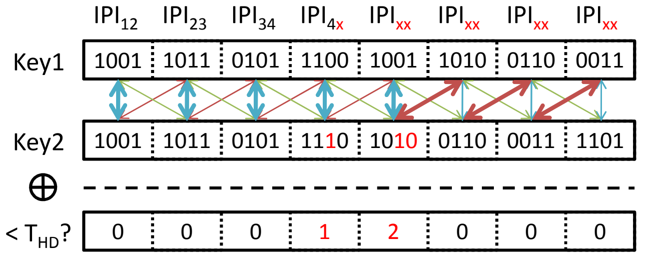

Fig. 6: Tolerating misdetected peaks by allowing comparisons neighbouring key segments.

$hd(k_1(1..m) \oplus k_2(1..m)) + min(hd(k_1(m+1..n) \oplus k_2(m..n-1))\ , hd(k_1(m..n-1) \oplus k_2(m+1..n)))$, for $m = 1$ to n. Consequently, authentication is successful if $HD_{min} < T_{HD}$. An example is given in Figure 6, where $m = 5$ has resulted in the lowest Hamming distance. Tolerating a peak misdetection in this way reduces the maximum value of the key strength, as every combination of key segments can essentially be considered as another authentication attempt or, in other words, an attack. As each key segment is compared to three other segments, as opposed to one, there is an additional $2n$ combinations and, as $KS = log_2(\#attacks)$, the security strength is reduced to $KS = H_k - 2 \cdot T_{HD} - 1 - log_2(2n)$.

Figure 7 shows the key strength using our algorithm for three bits per IPI and various key sizes. Compared to Figure 5, we find that due to the security overhead of allowing more combinations, the maximum key strength (at a 100% detection rate) is reduced between 11.8% for a 180-bit key and 36.7% for a 45-bit key. Thus, our algorithm introduces a relatively high security overhead for smaller keys. As in Figure 5, we observe a significant drop in key strength for a single large key when the detection ratio is lowered from 100% to 99.9% (-90 bits). Upon careful inspection of the generated keys, we have found that a number of keys have fallen victim to multiple misdetected peaks and, as our algorithm tolerates a single misdetect only, there is still a considerable decrease in key strength.

However, we can see that our algorithm does lead to significant security improvements: For a 90-bit key, employing our algorithm leads to an increase in key strength of up to 56 bits compared to the baseline. Moreover, we find that our algorithm allows for keys to maintain their maximum key strength for a wider range of detection rates. For example, a 45-bit key using our algorithm provides 8.5 bits of security irrespective of the detection rate. Our algorithm, thus, significantly improves the key strength by allowing the correction of a single misdetected peak. Overall, we conclude that *our method of tolerating a peak misdetection may achieve an increase in maximum key strength of up to 155%.*

V. Conclusions

In this paper we have characterized the effect of misdetecting a heart beat on the security performance of IPI-based security. We have shown that the security performance is significantly reduced as a function of the peak-detection rate (-70% to -303%). We have presented a method which overcomes single peak-misdetections and increases the security performance by up to 155% for detection rates between 99% and 100%. As future work, we will work on improving the security performance of our algorithm. Given that the problem of peak misdetection bears similarities with order-invariance problems, a solution might be sought in the use of fuzzy extractors [4].

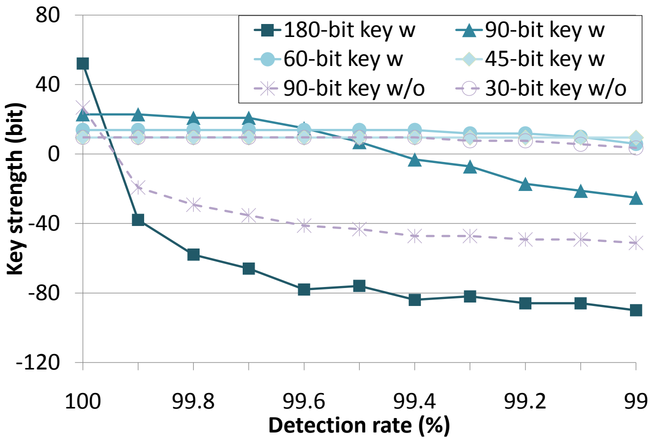

Fig. 7: Key strength as a function of detection rate with and without our algorithm using 3 bits (bit 3-5) per IPI.

References

[1] S.-D. Bao et al. Using the timing information of heartbeats as an entity identifier to secure body sensor network. In *T-ITB, pp. 772-779*, volume 12. IEEE, 2008.

[2] T. Denning et al. Absence makes the heart grow fonder: new directions for implantable medical device security. In *HotSec*, 2008.

[3] T. Denning et al. Patients, pacemakers, and implantable defibrillators: Human values and security for wireless implantable medical devices. In *SIGCHI*, pages 917–926, 2010.

[4] Y. Dodis et al. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, pages 523–540, 2004.

[5] J. M. Fard et al. A novel approach in r peak detection using hybrid complex wavelet (hcw). *Int J Cardiol*, 124(2):250–253, 2008.

[6] A. Ghaffari et al. A new mathematical based qrs detector using continuous wavelet transform. *CEE*, 34(2):81–91, 2008.

[7] A. L. Goldberger et al. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

[8] N. Iyengar et al. Age-related alterations in the fractal scaling of cardiac interbeat interval dynamics. *AJP-Regu*, 271(4):R1078–R1084, 1996.

[9] M. Li et al. Data security and privacy in wireless body area networks. *Wireless Communications, IEEE*, 17(1):51–58, 2010.

[10] J. P. Madeiro et al. An innovative approach of qrs segmentation based on first-derivative, hilbert and wavelet transforms. *MED ENG PHYS*, 34(9):1236–1246, 2012.

[11] G. B. Moody and R. G. Mark. The impact of the mit-bih arrhythmia database. *IEEE Eng Med Biol*, 20(3):45–50, 2001.

[12] C. C. Poon et al. A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health. *IEEE Commun. Mag.*, pages 73–81, 2006.

[13] M. Rostami et al. Heart-to-heart (h2h): authentication for implanted medical devices. In *ACM CCS*, pages 1099–1112, 2013.

[14] R. M. Seepers. Technical report: Assessing equivalent entropy using Ent. `http://www.erasmusbrainproject.com/techreport/TR-EBP-14-1.pdf`, 2014.

[15] R. M. Seepers et al. Adaptive entity-identifier generation for imd emergency access. In *ACM CS2*, pages 41–44, 2014.

[16] J. Walker. Ent a pseudorandom number sequence test program, jan 2008.

[17] G.-H. Zhang et al. Analysis of using interpulse intervals to generate 128-bit biometric random binary sequences for securing wireless body sensor networks. *T-ITB*, 16(1):176–182, 2012.