On Using a Von Neumann Extractor in Heart-Beat-Based Security

Robert M. Seepers[1], Christos Strydis[1], Ioannis Sourdis[2], and Chris I. De Zeeuw[1]
[1]Dept. of Neuroscience, Erasmus Medical Center, Rotterdam, The Netherlands
[2]Dept. of Computer Science & Engineering, Chalmers University of Technology, Gothenburg, Sweden
[1]{r.seepers, c.strydis, c.dezeeuw}@erasmusmc.nl [2]sourdis@chalmers.se

***Abstract*—The Inter-Pulse-Interval (IPI) of heart beats has previously been suggested for facilitating security in mobile health (mHealth) applications. In heart-beat-based security, a security key is derived from the time difference between consecutive heart beats. As two entities that simultaneously sample the same heart beats may generate the same key (with some inter-key disparity), these keys may be used for various security functions, such as entity authentication or data confidentiality. One of the key limitations in heart-beat-based security is the low randomness intrinsic to the most-significant bits (MSBs) in the digital representation of each IPI. In this paper, we explore the use of a von Neumann entropy extractor on these MSBs in order to increase their randomness. We show that our von Neumann key-generator produces significantly more random bits than a non-extracting key generator with an average bit-extraction rate between 13.4% and 21.9%. Despite this increase in randomness, we also find a substantial increase in inter-key disparity, increasing the mismatch tolerance required for a given true-key pair. Accordingly, the maximum-attainable effective key-strength of our key generator is only slightly higher than that of a non-extracting generator (16.4 bits compared to 15.2 bits of security for a 60-bit key), while the former requires an increase in average key-generation time of 2.5x.**

I. Introduction

Heart-beat-based security (HBBS) has emerged as a method for facilitating security in mobile health (mHealth) applications such as Body-Area Networks (BANs) or Implantable Medical Devices (IMDs) [7], [9], [12]. In HBBS, security keys are generated from the time difference (Inter-Pulse-Interval, IPI) between consecutive heart beats. Previous work has revealed that each IPI contains a certain degree of entropy and is measurable with some consistency throughout the same human body. Two entities which simultaneously sample the same heart beats may, thus, use the IPI for various security functions such as key agreement [15], BAN-device pairing [1], [7], [18] or IMD (-emergency) authentication [9], [12].

As with all biometrics, the measurements taken by two different entities tend to have some disparity between them due to inter-sensor variability (VAR$_{is}$). An exact key match is, thus, not likely to occur and some tolerance to this inter-key disparity is required. Effectively, this entails that the security strength of IPI-based keys depends on both the randomness of the IPI-bits selected for key-generation, as well as the allowed disparity between these keys [13]. Previous work has shown that the least-significant bits (LSBs) in the digital representation of IPIs are essentially random, yet are not usable for key generation due to a high VAR$_{is}$. Conversely, the most significant bits (MSBs) are easier to reproduce by both entities, yet are not random enough for security-key generation due to a high degree of inter-IPI correlation [13].

In this paper, we propose a novel von Neumann (vN) key-generator, which applies a von Neumann entropy extractor [16] on the MSBs of each IPI. The vN extractor produces a bit only when its input (the MSBs) changes value, allowing us to bypass the long sequences of identical values observed in the MSBs [13], yielding bits with increased randomness. By increasing this randomness, it may become possible to use these MSBs for key-generation, potentially leading to the production of stronger keys and a reduction in the key-generation time. Our vN key-generator is evaluated in terms of 1) the minimum entropy of generated keys; 2) the worst-case disparity between keys obtained by multiple entities; 3) the effective key strength; and 4) the key-generation time. To the best of our knowledge, this is the first work which proposes and evaluates the use of an entropy extractor in heart-beat-based security.

The rest of this paper is structured as follows: First, we discuss established work related to heart-beat-based security in Section II, after which we describe the most commonly used (non-extracting) IPI-based key-generator and our proposed vN key-generator in Section III. We evaluate these key-generators at length in Section IV, after which concluding remarks are given in Section V.

II. Related Work

In this Section, we consider related works which address the strength of IPI-based keys in heart-beat-based security. This key strength depends on both the entropy per IPI and the inter-key disparity allowed for a true-key pair, as will be discussed in Section IV-A1. Accordingly, we first discuss relevant studies on the IPI entropy, after which we review a number of related works on the inter-key disparity.

Various studies have evaluated the entropy per IPI (in this work considered to be represented as an 8-bit value) of healthy subjects, hypertensive subjects as well as patients with cardio-vascular disorders (CVDs) [9], [13], [19], all of which conclude that the four least-significant bits (LSBs) of each IPI contain a high degree of entropy. The most significant bits (MSBs), on the other hand, show a substantial reduction in entropy, mostly due to a large correlation between consecutive IPIs [13]. In an attempt to increase the entropy obtained from IPIs, Bao et al. [1] have proposed using the *m*ulti-Inter-Pulse Interval (*m*IPI) for key generation, where $mIPI_{(i,j)}$ is

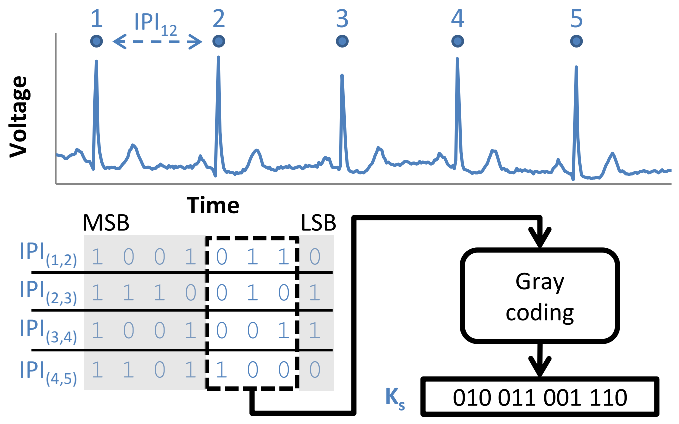


Fig. 1: Baseline (non-extracting) key-generator for generating a security key in heart-beat-based security.

the accumulation of all IPIs previously considered for key generation, i.e., $mIPI_{(i,j)} = \sum_{i=1}^{j-1} IPI_{(i,i+1)}; j > i$. While our own experiments confirm the apparent increase in entropy resulting from the use of $mIPI$, we note that it does *not* enhance security. The *m*IPI attempts to increase randomness using a simple addition and, as famously stated by John von Neumann, "any one who considers arithmetical methods to produce random digits is, of course, in a state of sin" [16]. Contrary to the *m*IPI, our proposed key-generator uses a von Neumann extractor [16] which *does* increase the entropy of the generated keys, albeit at the cost of increased key-generation time and inter-key disparity.

The inter-key disparity allowed for a true-key pair (expressed as the Hamming-distance threshold between the pair value, T_{HD}) depends on both the inter-sensor variability VAR$_{is}$ between two entities and the targeted true-key-pair matching rate. While all studies on the subject agree that T_{HD} reduces the strength of IPI-based keys, several methodologies have been followed to model VAR$_{is}$, resulting in a discrepancy across reported results [10]. Zhang et al. [19] have overlooked the issue of VAR$_{is}$, while Chang et al. [2] and Rostami et al. [9] have modeled it as two different leads of the same ECG (Electrocardiogram), all of which result in a minor disparity for a true-key pair. Poon et al. [7] and Bao et al. [1] model VAR$_{is}$ as the difference between an ECG and PPG (Photoplethysmography), showing a significant dissimilarity between generated keys (a 2.06% false-rejection rate has been described for a 128-bit key using $T_{HD} = 48$ bits). Another study has shown a similar disparity (describing a best-case $T_{HD} = 16$ bits for a 60-bit key) by considering VAR$_{is}$ as the difference between ECG and blood-pressure recordings [13]. It is clear from these studies that a substantial VAR$_{is}$ (and, thus, T_{HD}) may be expected when different cardiac signals are measured simultaneously. However, we consider such a model representative for typical mHealth applications, such as a BAN, where sensors are very likely placed at different locations on the same body and measure various cardiac signals. Accordingly, we adopt the model for VAR$_{is}$ described in [13].

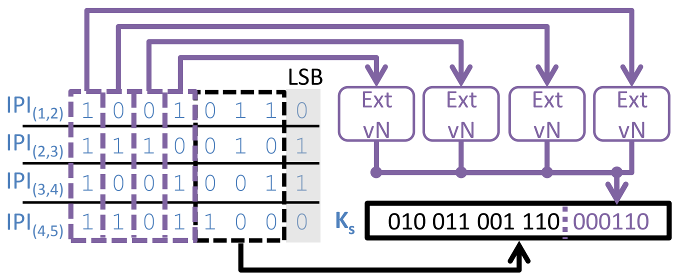


Fig. 2: Proposed von Neumann (vN) key-generator which applies a vN extractor (Ext vN) to each of the 4 MSBs. Gray coding is still applied to the key-bit selection (prior to bit-extraction) but not depicted for simplicity.

III. Key generation with von Neumann extractor

In this Section, we describe the most commonly used (baseline) key generator in heart-beat-based security, after which we discuss the von Neumann (vN) extractor and its application in our vN key-generator.

Figure 1 illustrates the most commonly used method of security-key generation in heart-beat-based security [1], [7], [9], [13], [19]. First, each entity detects a number of heart beats from their cardiac biosignals and calculates the time interval (IPI, in this work considered as an 8-bit value) from consecutive beats, i.e., $IPI_{(i,i+1)} = beat_{i+1} - beat_i$. From these IPIs, a predefined set of bits m is selected (the key-bit selection) to form a key segment: the most-significant bits (MSBs) are discarded due to their inherent low entropy, while LSBs may be discarded due to inter-sensor variability[1] (VAR$_{is}$). Gray coding is applied to the key segment in order to strengthen it against VAR$_{is}$ (reducing the number of bits affected by a small disparity between IPIs), after which n key segments are concatenated to form security key k. It is commonly (and in this work) assumed that k cannot be obtained by a remote adversary, allowing it to be used for various security functions. A prominent example is entity authentication, where the keys are used as entity identifiers and authentication is successful if the identifiers are similar enough, i.e., if the disparity (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).

By extending this baseline key-generator through applying a vN extractor on the MSBs, we hope to achieve an increase in key strength or a reduction in key-generation time, as more (random) bits are obtained per IPI. The vN extractor is a function which takes as input two bits, x_0 and x_1, and produces an output bit $x_{out} = x_0$ iff $x_0 \neq x_1$; else, x_0 and x_1 are both discarded. As long as the probability P($x_0 = 1$, $x_1 = 0$) = P($x_0 = 0$, $x_1 = 1$), this results in the production of quasi-random bits [11] regardless of the probability $P(x_i = 0)$ and $P(x_i = 1)$ as the inputs $(x_0, x_1) \subseteq \{(0,0),(1,1)\}$ are discarded [16]. The extractor may be applied to any input x_s containing an even number of bits: In this case, x_s is first split into $x_s/2$ bit pairs after which extraction is applied to each pair. For example, the input $x_s = 00000100$ is split into bit

[1]Assuming precise and non-drifting sensors, VAR$_{is}$ is the variance between two different sensor measurements of cardiac biosignals, caused by the variable pulse-transition time of ventricular contraction (heart beats) to the rest of the body due to, for example, motion and pressure differences.

pairs 00|00|**01**|00, resulting in a single bit **0** being extracted from the third pair.

We consider the vN extractor an excellent solution to increase the entropy of the MSBs. Previous work has shown that MSBs suffer from significant correlations across multiple IPIs (inter-IPI correlations) [13], [19]. As the vN extractor only extracts a bit when the value of its input changes, we may increase the randomness of these bits by bypassing the long sequences of identical values observed in them. Moreover, the vN extractor is realized using a single XOR operation, making it suitable for low-power devices such as IMDs. Figure 2 depicts our proposed vN key-generator, where a vN extractor is applied to each of the MSBs. In this arrangement, each extractor obtains an input stream x_s by taking bits from consecutive IPIs in the same bit-position, for example, bit 5 from IPI$_{(1,2)}$ and IPI$_{(2,3)}$. The extracted bits (from the different extractors) are subsequently concatenated to form an additional key segment which is appended to security key k, similar to the baseline (non-extracting) key-generator. This additional key segment increases the number of bits obtained per IPI (18 bits are obtained from 4 IPIs in the example of Figure 2, compared to 12 bits for the baseline in Figure 1), allowing for a decrease in key-generation time.

IV. Evaluation

In this Section, we evaluate and compare the performance of our vN key-generator to that a non-extracting (baseline) key generator in terms of key entropy, inter-key disparity, effective key strength and key-generation time. First, we describe our experimental setup, consisting of our evaluation criteria and considered datasets in Section IV-A1, after which the results of our evaluation will be discussed at length in Section IV-B.

A. Experimental Setup

In this Section, we describe our experimental setup for evaluating the baseline and vN key-generators. First, we introduce the effective key strength KS_{eff} as a figure of merit, as well as its acquisition through measuring the key entropy H_k and the tolerated disparity for a true-key pair T_{HD}. Afterwards, we present the datasets considered in our evaluation.

1) Key Strength: The strength of a key is determined by how hard it is for an attacker to guess it. To quantify the key strength in bits, we define the effective key strength KS_{eff} as the number of *entropic bits* which should be known to an attacker in order to successfully authenticate to the IMD with probability $P_{auth} = 0.5$ [13]. That is, an attacker would have to mount on average $2^{KS_{eff}}$ attacks. To exemplify, in Figure 3 we plot a distribution of Hamming distances between an authentication key and various, randomly selected 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 $E(X) = p_0 \cdot H_k = p_1 \cdot H_k = \frac{H_k}{2}$, where p_0 and p_1 denote the probability of a bit being zero or one (for entropic bits, $p_0 = p_1 = \frac{1}{2}$) and H_k denotes the number of entropic bits in the key (ideally, H_k = n). Since, on average, half the number of entropic bits are mismatched by simply guessing, for successful authentication an attacker would need to try up to:

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

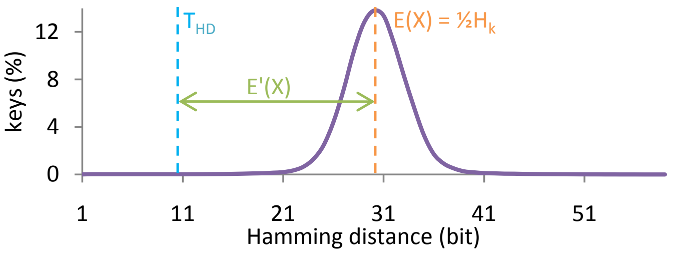


Fig. 3: Key strength KS_{eff} as a function of H_k and T_{HD}.

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

As it is unlikely that keys will be a perfect match due to VAR$_{is}$, we allow entities to authenticate if their keys differ no more than T_{HD} bits, where T_{HD} denotes the Hamming-distance threshold. As a result, the average number of mismatched bits will be effectively reduced by the amount of "don't care" T_{HD} bits; essentially $E'(X) = \frac{H_k}{2} - T_{HD}$ (see Figure 3). In this more general case, KS_{eff} is calculated as follows:

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

Note that KS_{eff} may now assume negative values, signifying that an attacker would require less than one attack on average to guess the key ($2^{KS_{eff}} < 1$). Obviously, a negative KS_{eff} will never exist in practical applications as an attacker would always require at least one attack, i.e., KS_{eff} would be greater or equal to zero. Nevertheless, considering KS_{eff} as a potentially negative value allows us to investigate exactly how far the generated keys are from providing any form of security ($KS_{eff} > 0$).

To determine KS_{eff} we, thus, have to evaluate the entropy H_k and required Hamming-distance threshold T_{HD}, the acquisition of which is described next.

2) Entropy: The upper limit H_k of the effective key strength is determined by the randomness of the key-bit selection (the bits selected per IPI) for key generation. As a first-order estimation of this randomness, we use an arithmetic mean, autocorrelation and compression test over the generated keys (an extension over the tests in the ENT randomness test suite [17] used in [13]), varying the bits selected for key generation.

- The arithmetic-mean test evaluates the average probability of a particular key-bit being one or zero, i.e., (P(x_i = 0), P(x_i = 1)) and, thus, represents the randomness when a bit is sampled from a key. This test reveals a bias in the key bits if $P(x_i = 0) \neq P(x_i = 1)$;
- The autocorrelation test determines the probability of a key-bit being identical to its l^{th} neighboring bit, i.e., $P(x_i = x_{i-l})$, where we choose l = 1, 2, 3, ..., 20 to determine if there are any intra-key correlations. A high value for $P(x_i = x_{i-l})$ indicates repetitive patterns in consecutive IPIs, yielding a reduction in entropy (and security) as the bits in $IPI_{(i,i+1)}$ have predictive value over those in $IPI_{(i+l,i+1+l)}$.

- The compression test splits the generated keys into c-sized symbols S and evaluates the frequency of each symbol occuring, i.e., $P(s) = \frac{\sum S=s}{\sum S}$, where $s = 1, 2, 3, ..., 2^c$, S is the value of c consecutive bits and we choose $c = 1, 2, 3, ..., 8$. A high value for $P(s)$ indicates that certain symbols (bit-patterns) s occur more frequently throughout the distribution, indicating correlations between consecutive IPIs.

Based on the probabilities calculated using our tests, we may compute the Shannon entropy for the arithmetic mean (H_{am}), autocorrelation (H_{ac}) and compression (H_c) tests as [14]:

$$H = \sum_i p_i \, log_2 \, p_i \tag{2}$$

where p_i is the probability of a particular event, for example, the probability of a given symbol s in the compression test. As a conservative estimation, we define the minimum entropy $H_{min} = min(H_{am}, H_{ac}, H_c)$. It is shown later on (in Table II) that H_{min} may frequently not equal its maximum value of 1, i.e., the bits in the key are not fully entropic ($p_0 \neq p_1 \neq 0.5$, where p_0 and p_1 denote the probability of a bit being 0 or 1, respectively). In such cases, it is not possible to simply evaluate the randomness of the key H_k by multiplying H_{min} by the number of bits in the key due to the logarithmic nature of H_{min}. For example, a single bit with $H_{min} = 1$ has probabilities $p_0 = p_1 = 0.5$, which is clearly more random than two bits with $H_{min} = 0.5$ each (having $p_0 = 0.89$, $p_1 = 0.11$). To calculate the equivalent entropy of such low-entropic bits, what needs to be done is for a sufficient number of bits n_{eq} to be put together so as to result in a fully entropic symbol S with symbol entropy $H_S = 1$, i.e. it will hold that $p_S = p_{\overline{S}} = \frac{1}{2}$. To construct S, we first obtain p_0 and p_1 from Equation (2), where we set $i = 0, 1$ and $H = H_{min}$. By concatenating n_{eq} of these bits, we obtain $p_S = \frac{1}{2} = p_{max}{}^{n_{eq}}$ where $p_{max} = max(p_0, p_1)$[2], allowing us to solve for $n_{eq} = log_{p^{max}}(\frac{1}{2})$. As each symbol S of length n_{eq} bits provides randomness equivalent to 1 fully entropic bit, each bit shall have equivalent entropy $H_{eq} = \frac{1}{n_{eq}}$. Based on calculating this equivalent entropy for all i individual IPI-bit positions (H^i_{eq}), the entropy of the key-bit selection m may be obtained[3] from $H^m_{eq} = \sum_i H^i_{eq}$ for all $i \; \epsilon \; m$. Finally, as n key-segments are combined to form key k, we obtain $H_k = H^m_{eq} \cdot n$.

3) Hamming-Distance Threshold: The allowed inter-key disparity T_{HD} is determined by both the inter-key disparity VAR$_{is}$ and the desired probability of key-matching. Lowering T_{HD} allows for an increase in KS_{eff} (as an attacker's key is required to be more similar to the actual key), yet also reduces the chance of successful matching for a true-key pair. To determine T_{HD}, we calculate the Hamming distance between true-key pairs generated by two entities and see with what threshold T_{HD} the keys would match *reliably*, where reliable is defined as successful matching with probability $P_{match} = 1 - 10^{-6}$ within a predefined, upper time limit [13]. Without loss of generality, in this work we set the time limit of key generation to 60 seconds. We expect that such a matching criterion will be practically feasible for some of the most safety-critical applications of heart-beat-based security, such as providing IMD-emergency authentication [9], [12]. We evaluate our generator for a 60-bit key as it is allows us to easily assess the effective key strength under our authentication constraints (assuming an average human-heart rate of 60 beats-per-minute), as has been done in prior work [13].

[2]Given that $p_0 = 1 - p_1$, a maximum operator is used so as to get the highest probability between p_0 and p_1. This is the only way that, when concatenating multiple bits n, we can get a combined probability $p^n_{0|1} = 0.5$.

[3]In this work, we have not found any *intra*-IPI dependencies (between IPI bits), permitting H^m_{eq} to be calculated as a linear addition of the H^i_{eq} of the selected IPI-bits [13].

TABLE I: Average bit-error rate due to VAR$_{is}$.

Bit #	0	1	2	3	4	5	6	7
Error rate	0.46	0.29	0.15	0.08	0.04	0.02	0.01	0.00

TABLE II: Entropy-test results for individual IPI-bit positions using the baseline key-generator.

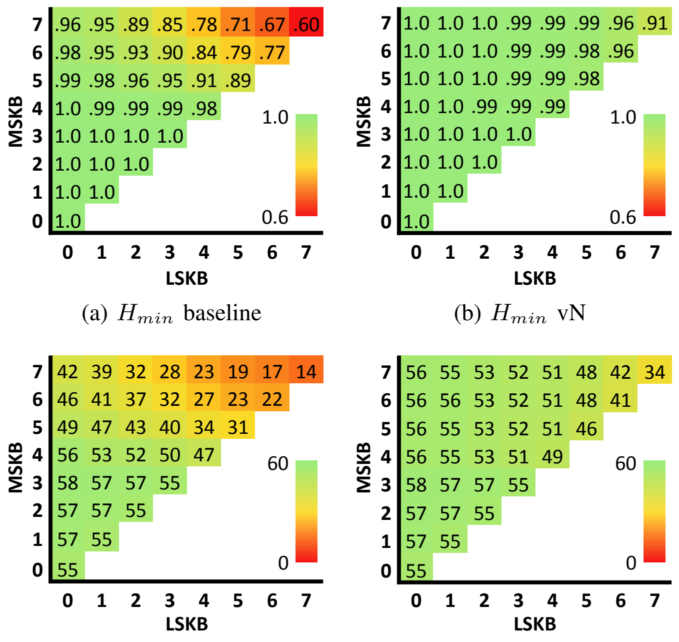
Bit	H_{am}	H_{ac}	H_c	H_{min}	H_{eq}
0	1.00	1.00	1.00	1.00	0.92
1	1.00	1.00	1.00	1.00	0.92
2	1.00	1.00	1.00	1.00	0.92
3	1.00	1.00	1.00	1.00	0.92
4	1.00	0.99	0.98	0.98	0.78
5	1.00	0.91	0.89	0.89	0.52
6	1.00	0.81	0.77	0.77	0.36
7	1.00	0.67	0.60	0.60	0.23

4) Used Datasets: Our experiments are based on the ECG-recordings in the *MIT-BIH arrhythmia dataset*, a commonly used dataset in the field which contains the ECG recordings of subjects with a wide variety of CVDs [3], [6]. This dataset contains half-hour recordings of 48 subjects, providing us with 85k IPIs after the exclusion of signal artifacts. We model the inter-sensor variability VAR$_{is}$ through the difference between ECG and blood-pressure recordings as described in Section II and in [12], [13]. That is, a second recording is created by adding VAR$_{is}$ obtained from the *Fantasia dataset* [4] to the ECG recordings from the first ECG-lead of the *MIT-BIH arrhythmia dataset*. To indicate the (average) disparity we may expect due to VAR$_{is}$, we summarize the average bit-error rates for this model in Table I. Note that the disparity is quite severe for the least-significant bit positions (46% and 29% of all bits are expected to mismatch for bit positions 0 and 1, respectively) and decreases exponentially for more significant bits.

B. Experimental Results

In this Section, we evaluate the non-extracting (baseline) and extracting (vN) key-generator in terms of: 1) Entropy of the generated keys H_k; 2) Required tolerance to inter-key disparity for a true-key pair T_{HD}; 3) Effective key strength KS_{eff}; and 4) Key-generation time.

1) Entropy: To evaluate the entropy for the generated keys using the baseline and vN key-generator, we first assess the minimum entropy per-bit H_{min} after which we discuss the resulting key-entropy H_k for both generators. In order to understand the characteristics of the randomness available



(a) H_{min} baseline

(b) H_{min} vN

(c) H_k baseline

(d) H_k vN

Fig. 4: Entropy results for the baseline and vN key-generator. Depicted are: (a) and (b): Minimum-entropy per bit H_{min} results for the baseline and vN key-generator, respectively; (c) and (d): Entropy for a 60-bit key H_k using the baseline and vN key-generator, respectively.

in each IPI, let us first consider the situation where only one bit is selected per IPI for the baseline key-generator. Table II presents the entropy results for the individual tests in Section IV-A2, H_{am}, H_{ac} and H_c and the resulting min-entropy H_{min}. In line with related work, we see that the four least-significant bit (LSB) positions of each IPI contain a high degree of entropy, scoring between 0.99 and 1.00 for all tests. From bit position 4 onwards, we find that the entropy results are gradually decreasing: while H_{am} appears unaffected, we see a substantial decrease in H_{ac} and H_c. These most-significant bits (MSBs), thus, do not show a particular bias, yet show significant correlations between consecutive IPIs (the minimum value for H^i_{ac} and H^i_c were obtained using test parameters $l = 1$ and $c = 8$, respectively), effectively reducing entropy. Table II also presents the equivalent entropy per bit H_{eq} for the various IPI-bit positions. Note that even though H_{min} is considerably high for several bit positions (1.00), H_{eq} is substantially lower with a maximum value of 0.92: Due to the logarithmic scale onto which entropy H_{min} is defined, even a small difference between the maximum-attainable entropy ($H_{min} = 1$) and the actual measured H_{min} results in a significant reduction in H_{eq}.

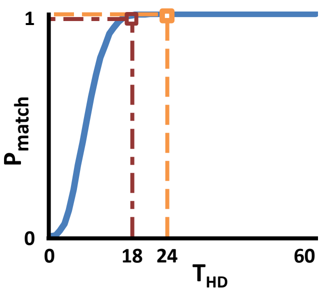
Let us now consider key generation using more than one bit per IPI. Figure 4 presents H_{min} for both the baseline (a) and vN (b) key-generator; The y-axis shows the least-significant IPI-bit used for key-generation (least-significant key-bit, LSKB) while the x-axis shows the most-significant IPI-bit used for key-generation (most-significant key-bit, MSKB). For example, the H_{min} results for a key-bit selection including bits 3-5 are presented for the values ($LSKB = 3$, $MSKB = 5$). Note that the diagonal of Figure 4a equals the results presented in Table II. For the baseline key-generator, it is evident that using fewer LSBs (i.e., increasing $LSKB$) or using more MSBs (increasing $MSKB$) yields a (substantial) reduction in H_{min}, down to 0.6 for ($LSKB$ = 7, $MSKB$ = 7).

By applying the vN extractor to the MSBs (Figure 4b), we observe that the H_{min} of the MSBs is substantially increased. For example, we find a value for H_{min} between 0.99 and 1.00 for any key-bit selection when $LSKB < 5$ (regardless of $MSKB$), considerably higher than the H_{min} value for the baseline key-generator using the same key-bit selection (which may be as low as 0.78). Despite this observed increase in entropy, we *do* find that H_{min} is gradually decreased by including more MSBs in the key-bit selection, down to 0.91 for IPI-bit position 7 ($LSKB = 7$, $MSKB = 7$). We observe this decrease in both H_{ac} and H_c, indicating that the bits produced by the vN extractor still show some correlations. Upon inspection, we find that this correlation is due to *anti*correlation in a subset of input IPIs. That is, a periodic alternation between 1's and 0's is observed in consecutive IPIs, causing the vN extractor to produce a sequence of bits of the same value. Even though this anticorrelation exists, we still obtain a substantial increase in H_{min} compared to the baseline.

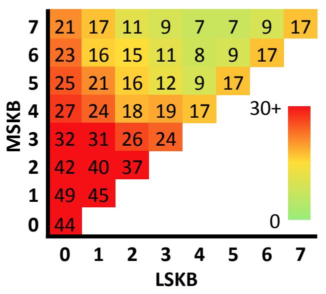
Based on H_{min}, we may now calculate the key entropy H_k as presented in Figure 4c and 4d for a 60-bit key. Limiting the key-bit selection to the four LSBs only ($LSKB < 4$, $MSKB < 4$), we observe a value for H_k between 55 and 58 bits for both generators (close to the maximum H_k = 60 bits for our 60-bit key), again demonstrating the high randomness of these IPI bits. When MSBs are included in the key-bit selection ($MSKB \geq 4$) and, thus, entropy extraction is applied, we find that the vN key-generator consistently produces more random keys than the baseline. This increase in H_k is most apparent when all the MSBs are included in the key-bit selection ($LSKB \geq 0$, $MSKB = 7$): For the baseline, this results in a value for H_k between 42 to 14 bits, compared to 56 and 34 bits for the vN key-generator, i.e., an increase in H_k of up to 2.5x is observed.

2) Hamming-Distance Threshold: Following up on equation 1, we discuss next the Hamming-distance threshold T_{HD} as a function of the bits selected per IPI, the results of which are depicted in Figure 5. First, Figure 5a illustrates the trade-off between T_{HD} (x-axis) and the desired true-key-pair matching rate P_{match} (y-axis) when selecting IPI-bit position 3 only ($LSKB$ = 3, $MSKB$ = 3). It is shown that P_{match} rapidly increases as a function of T_{HD} up to P_{match} = 0.99 (T_{HD} = 18 bits), after which a significant increase in T_{HD} is required to further increase P_{match}. For the true-key-pair matching rate targeted in this work ($P_{match} = 1 - 10^{-6}$), this results in T_{HD} = 24 bits. Recall from Section IV-A1 that the effective key strength $KS_{eff} = H_k - 2 \cdot T_{HD} - 1$: That is to say, requiring a high P_{match} impedes a high KS_{eff}.

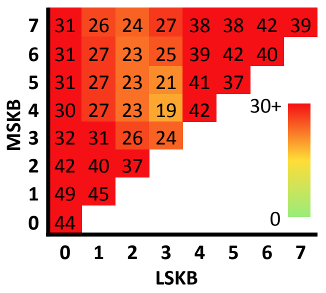
Figure 5b depicts T_{HD} for the baseline key-generator for our target P_{match}. It is clear that the LSBs of each IPI are substantially affected by VAR$_{is}$: When a key-bit selection is made from any of the first three LSBs ($LSKB \leq 2$, $MSKB \leq 2$), we find a high T_{HD} between 37 and 49 bits. Since the MSBs are less susceptible to VAR$_{is}$ (see Table I), we observe a decrease in T_{HD} both when more MSBs are added to the key-bit selection (increasing $MSKB$) and when removing



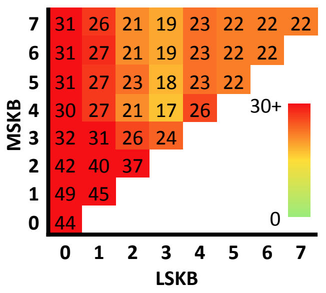
(a) Trade-off T_{HD} and P_{match}



(b) T_{HD} baseline



(c) T_{HD} vN without key-misallignment tolerance



(d) T_{HD} vN with key-misallignment tolerance

Fig. 5: True-key-pair disparity, depicting (a): An example of a trade-off between the Hamming-distance threshold T_{HD} and target key-matching rate P_{match}, here depicted for ($LSKB$ = 3, $MSKB$ = 3); (b): T_{HD} for the baseline key-generator; (c) and (d): T_{HD} for the vN key-generator with- and without tolerance to key-misallignment, respectively.

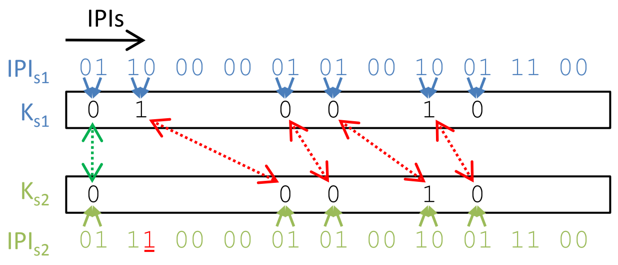
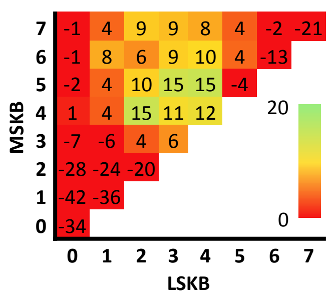


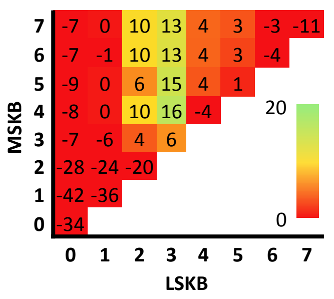
Fig. 6: Key-bit misallignment due to VAR$_{is}$. A single bit-error in the input (IPI_{s2}, bit 4) of the vN extractor causes a shift in key-segments, resulting in a significant disparity between the generated keys if a direct bit-by-bit comparison (illustrated by the dashed arrows) is performed.

the LSBs from this selection (increasing $LSKB$). While IPI-bit position 7 has the lowest average bit-error rate, we find the lowest T_{HD} when selecting IPI bits 4-7 (i.e., including noisier IPI bits in the key-bit selection). We explain this as follows: Using more bits per IPI implies using less IPIs for generating a 60-bit key, less key segments are affected by VAR$_{is}$. Moreover, these key segments are protected from multi-bit errors by Gray coding, allowing for the observed decrease in T_{HD}.

Figure 5c depicts T_{HD} for the vN key-generator. When the key-bit selection includes any of the MSBs ($LSKB \geq 0$, $MSKB \geq 4$) we find a substantial increase in T_{HD} compared to the baseline by up to 33 bits (for $LSKB = 6$, $MSKB = 7$). In sheer contrast to the baseline generator, we find that



(a) KS_{eff} baseline



(b) KS_{eff} vN

Fig. 7: effective key strength KS_{eff} results for a 60-bit key using (a) the baseline and (b) the vN key-generator, respectively.

including the MSBs shows an *increase* in T_{HD} for the vN key-generator. We explain this observed increase in T_{HD} with the help of Figure 6, which depicts the bit-extraction process for two entities (s_1 and s_2). Depicted for both entities are the input IPI-bits to the vN extractor (IPI_{s1}, IPI_{s2}) as well as the bits extracted from these input bits (K$_{s1}$, K$_{s2}$). Note the disparity in IPI_{s1} and IPI_{s2} in the 4^{th} input IPI-bit. Due to this disparity, s_2 does not generate a bit using IPIs 3 and 4, whereas $s1$ does, causing a shift in key-indexing. A naive bit-by-bit comparison on the generated keys, as is indicated by the arrows in Figure 6, will accordingly result in a significant inter-key disparity.

Fortunately, there are various methods of tolerating this shift in key indexing through order-invariant matching [5], [8]. As such, we re-evaluate T_{HD}[4] assuming such an order-invariant matcher is used, the result of which is presented in Figure 5d. It is clear that tolerating these key-misallignments results in a considerable improvement in T_{HD} by up to 20 bits (comparing Figures 5c and 5d). However, we still find a considerable increase in T_{HD}, even if key-shifting is tolerated, with respect to the baseline (Figure 5b). For example, the minimum T_{HD} for the vN key-generator is 17 bits ($LSKB = 3, MSKB = 4$), compared to a minimum T_{HD} of 7 bits ($LSKB = 4$, $MSKB = 7$) for the baseline. As bit-extraction increases the number of IPIs required for key generation, there are more IPIs which may be affected by VAR_{is}, effectively causing an increase in the disparity between the generated keys.

3) Effective Key Strength: Based on the aforementioned results, we may now calculate $KS_{eff} = H_k - 2 \cdot T_{HD} - 1$, depicted for both key-generators in Figure 7. For the baseline, it can be observed that KS_{eff} is minimal (even < 0) when the key-bit selection includes mostly LSBs or MSBs per IPI (for ($LSKB < 3$, $MSKB < 3$) and ($LSKB \geq 5$, $MSKB \geq 5$)). As discussed before, the LSBs of each IPI suffer significantly from inter-sensor disparity, resulting in a considerable increase in T_{HD} and, therefore, reduction in KS_{eff}. The MSBs, on the other hand, result in a reduction in KS_{eff} due to the limited key-entropy H_k. As a result, we find a maximum effective key strength of 15.2 bits when bits 2-4 are selected from each IPI.

[4]Strictly speaking, this no longer refers to the Hamming distance. However, as the threshold still reflects the tolerated (order-invariant) disparity in a true-key pair, we will retain T_{HD} as the notation.

Similar to the baseline, we find that the vN key-generator (Figure 7b) results in a low KS_{eff} when the key-bit selection includes mostly the LSBs or MSBs. Despite the considerable increase in H_k when the vN extractor is applied to the MSBs, we have found a substantial increase in T_{HD} for these keys. Accordingly, the contribution of the MSBs to KS_{eff} is limited, resulting in only a slight increase in KS_{eff} of 1.2 bits (16.4 bits when bits 3-4 are used per IPI, compared to 15.2 bits for the baseline).

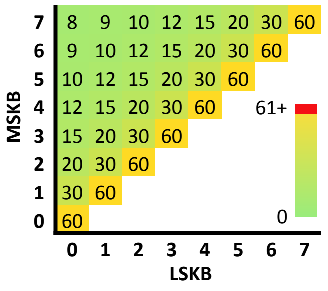
4) Key-Generation Time: We describe next the key-generation time T_{kg} for the baseline and vN key-generators. For the baseline key-generator, a fixed number of bits is obtained per IPI, resulting in a deterministic key-generation time given by $T_{kg} = \frac{60}{MSKB-LSKB+1}$ as depicted in Figure 8a. Given that at least one bit is obtained per IPI and an average heart rate of 60 BPM, the baseline key-generator always respects the 60-second constraint given in Section IV-A1.

In contrast, the vN key-generator extracts – by design – a non-deterministic number of bits from the MSBs as a bit is only produced if these MSBs change value. This non-deterministic behavior results in a variable T_{kg}, i.e., we differentiate between T_{kg}^{max}, T_{kg}^{avg} and T_{kg}^{min} depending on the bit-extraction case, as depicted in Figures 8b, 8c and 8d, respectively. Note that we find a disparity between T_{kg}^{max}, T_{kg}^{avg} and T_{kg}^{min} when MSBs are included in the key-bit selection, which is more substantial when the key-bit selection is shifted to the MSBs (i.e., increasing $LSKB$ and $MSKB$).

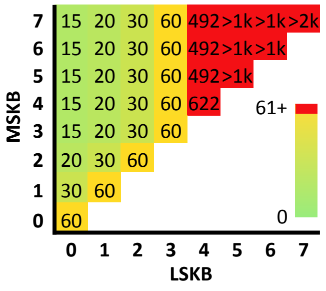
To determine if the vN key-generator is capable of producing keys within our 60-seconds constraint, we consider the maximal key-generation time T_{kg}^{max} (Figure 8b). When an LSB is included in the key-bit selection ($LSKB < 4$) we observe that adding MSBs to key-bit selection ($MSKB \geq 4$) does not yield an improvement in T_{kg}^{max}: As the MSBs are first passed through the vN extractor (which changes the number of extracted bits dynamically, based on its input), it may occur that a 60-bit key is formed from the LSBs alone, yielding no improvement to T_{kg}^{max}. Note, however, that all key-bit selections which include an LSB result in a feasible $T_{kg}^{max} \leq 60$ seconds. If the generated keys consist solely of the MSBs ($LSKB \geq 4$), we observe a value for T_{kg}^{max} between 492 and 2,552 seconds, i.e., basing the key solely on the MSBs results in an infeasible T_{kg}^{max} for the vN key-generator. These long key-generation times are attributed to the high degree of serial correlation in the input IPI-bits, yielding a minimal number of extracted bits (as revealed in Section IV-B1).

We assess the typical key-generation time of the vN key-generator by considering T_{kg}^{avg} (Figure 8c). For a feasible key-bit selection ($T_{kg}^{max} < 60$, found for $LSKB < 4$), we find an average increase in T_{kg}^{avg} between 1.2x and 3.4x compared to baseline. Furthermore, if we compare the T_{kg}^{avg} when both generators produce their respective strongest keys (as discussed in the previous Section), we find that the vN key-generator has an increased (average) key-generation time of 2.5x (up to 50 seconds).

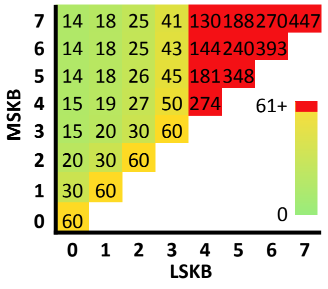
Besides, by calculating T_{kg}^{avg}, it is now possible to deduce the average extraction rate of the vN extractors in the MSB positions, which defines how many bits are extracted (on average) per input bit. By selecting the MSBs individually ($LSKB = MSKB \geq 4$) from Figure 8c, we find that an average of 274, 348, 393 and 447 input-bits are required to extract 60 bits for IPI-bit positions 4, 5, 6 and 7, respectively. Accordingly, the extraction rates for these respective bit-positions are 21.9%, 17.3%, 15.3% and 13.4%.



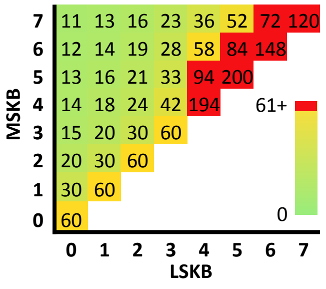
(a) T_{kg} baseline



(b) T_{kg}^{max} vN



(c) T_{kg}^{avg} vN



(d) T_{kg}^{min} vN

Fig. 8: Key-generation time T_{kg} for both key-generators: (a) depicts T_{kg} for the baseline key-generator; (b), (c) and (d) depict the T_{kg}^{max}, T_{kg}^{avg} and T_{kg}^{min} for the vN key-generator, respectively.

V. Conclusion

In this paper, we have introduced a novel method of generating security keys in heart-beat-based security by applying a von Neumann entropy extractor on the most-significant bits (MSBs) of each IPI. This extractor produces bits with increased randomness with an average bit-extraction rate between 13.4% and 21.9% (when applied to IPI-bit positions 7 and 4, respectively), resulting in a significant increase in key-entropy H_k of up to 2.5x. However, the increase in H_k is directly coupled to an increase in the average disparity for a given true-key pair, requiring a greater tolerance T_{HD} to this inter-key disparity. We have found this trade-off between H_k and T_{HD} to dramatically impact the effective key strength KS_{eff} for a vN key-generator, yielding only slightly stronger keys than the non-extracting generator (16.4 bits compared to 15.2 bits for a 60-bit key), while requiring an increase in key-generation time of up to 2.5x. This work has shown that entropy extraction in heart-beat-based security is not trivial, given the complex interplay between entropy and inter-key disparity.

VI. Acknowledgement

This work has been supported by the EU-funded projects DeSyRe (Grant agreement no: 287611) and SHARCS (Grant agreement no: 644571).

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] S.-Y. Chang, Y.-C. Hu, H. Anderson, T. Fu, and E. Y. Huang. Body area network security: robust key establishment using human body channel. In *Proceedings of the USENIX conference on Health Security and Privacy*, pages 5–5, 2012.

[3] 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.

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

[5] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006.

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

[7] 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.

[8] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial & Applied Mathematics*, 8(2):300–304, 1960.

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

[10] M. Rushanan et al. Sok: Security and privacy in implantable medical devices and body area networks. *Proceedings of the IEEE S&P*, pages 529–539, 2014.

[11] M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33(1):75–87, 1986.

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

[13] R. M. Seepers et al. Peak misdetection in heart-beat-based security characterization and tolerance. *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2014.

[14] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[15] K. K. Venkatasubramanian, A. Banerjee, and S. K. S. Gupta. Pska: usable and secure key agreement scheme for body area networks. *ITB, IEEE Trans. on*, 14(1):60–68, 2010.

[16] J. von Neumann. Various techniques used in connection with random digits. *Monte Carlo Method, National Bureau of Standards Applied Math*, pages 36–38, 1951.

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

[18] F. Xu et al. Imdguard: Securing implantable medical devices with the external wearable guardian. In *INFOCOM*, pages 1862–1870. IEEE, 2011.

[19] 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.