

ON THE ESTABLISHMENT OF
PSEUDO RANDOM KEYS
FOR BODY AREA NETWORK SECURITY
USING PHYSIOLOGICAL SIGNALS

by
BESTE SEYMEN

Submitted to the Institute of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

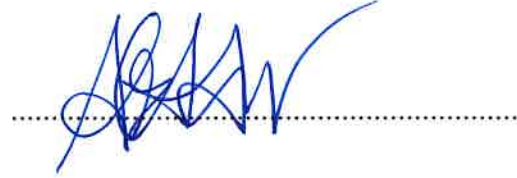
Sabancı University

January 2019

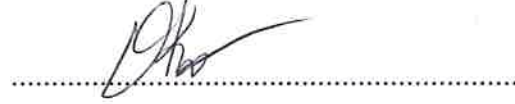
ON THE ESTABLISHMENT OF PSEUDO RANDOM KEYS
FOR BODY AREA NETWORK SECURITY
USING PHYSIOLOGICAL SIGNALS

APPROVED BY:

Prof. Albert Levi
(Thesis Supervisor)



Dr. Duygu Karaođlan Altop
(Thesis Co-Supervisor)



Assoc. Prof. Selim Balcısoy



Assoc. Prof. Cemal Yılmaz



Asst. Prof. Kbra Kalkan akmakçı



DATE OF APPROVAL: 04/01/2019

© Beste Seymen 2019
All Rights Reserved

ABSTRACT

ON THE ESTABLISHMENT OF PSEUDO RANDOM KEYS FOR BODY AREA NETWORK SECURITY USING PHYSIOLOGICAL SIGNALS

BESTE SEYMEN

M.Sc. Thesis, January 2019

Supervisor: Prof. Albert Levi

Co-Supervisor: Dr. Duygu Karaođlan Altop

Keywords: Cryptographic Key Generation, Body Area Network Security,
Physiological Signals, Key Agreement, Bio-cryptography

With the help of recent technological advancements especially in the last decade, it has become much easier to extensively and remotely observe medical conditions of the patients. This observation is done through wearable devices named *biosensors* that act as connected nodes on the Body Area Network (BAN). The main goal of these biosensors is to collect and provide critical and sensitive health data concerning the host individual, communicate with each other in order to make decisions based on what has been captured and relay the collected data to remote healthcare professionals. The sensitive nature of this critical data makes it extremely important to process it as securely as possible. Biosensors communicate with each other through wireless medium that is vulnerable to potential security attacks. Therefore, secure mechanisms for both data protection and intra-BAN

communication are needed. Moreover, these mechanisms should be lightweight in order to overcome the hardware resource restrictions of biosensors. Random and secure cryptographic key generation and agreement among the biosensors take place at the core of these security mechanisms.

In this thesis, we propose SKA-PSAR (Secure Key Agreement Using Physiological Signals with Augmented Randomness) system. The main goal of this system is to produce highly random cryptographic keys for the biosensors for secure communication in a BAN. Similar to its predecessor SKA-PS protocol by Karaođlan Altop et al., SKA-PSAR also employs physiological signals, such as heart rate and blood pressure, as inputs for the keys and utilizes the set reconciliation mechanism as basic building block. Novel quantization and binarization methods of the Secure Key Agreement Protocol of the proposed SKA-PSAR system distinguish it from SKA-PS in a way that the former has increased the randomness of the generated keys. In addition, the generated cryptographic keys in our proposed SKA-PSAR system have distinctive and time variant characteristics as well as long enough bit sizes that can be considered resistant against a cryptographic attack. Moreover, correct key generation rate of 100% and false key generation rate of 0% have been obtained. Last but not least, results of the computational complexity, communication complexity and memory requirements of our proposed system are quite higher as compared to SKA-PS, but this is a cost that needs to be paid for achieving high randomness level.

ÖZET

GÖVDE ALAN AĞLARININ GÜVENLİĞİ İÇİN FİZYOLOJİK SİNYALLER KULLANILARAK SÖZDE RASGELE ANAHTARLAR OLUŞTURULMASI

BESTE SEYMEN

Master Tezi, Ocak 2019

Danışman: Prof. Dr. Albert Levi

Eş-Danışman: Dr. Duygu Karaoğlan Altop

Anahtar Sözcükler: Kriptografik Anahtar Üretimi, Gövde Alan Ağlarında Ağ Güvenliği, Fizyolojik Sinyaller, Anahtar Mutabakatı, Biyo-kriptografi

Son yıllarda yaşanan teknolojik gelişmelerin yardımıyla, hastaların sağlık durumlarını uzaktan gözlemleyebilmek kolaylaştı. Hastaların gözlemlenmesi “biyosensör” adı verilen ve gövde alan ağında birbirine bağlı düğümler halinde bulunan giyilebilir cihazlar ile yapılmaktadır. Biyosensörlerin en önemli görevleri bağlı bulunulan kişiden hassas ve kritik verilerin toplanması, toplanan verilerin biyosensörler arasında iletişim kurularak analiz edilmesi ve ardından sağlık çalışanlarına gönderilmesidir. Toplanan verilerin hassas veriler olması nedeniyle veriler üzerinde yapılan işlemlerin güvenli olması gerekmektedir. Biyosensörler güvenlik saldırılarına açık olan kablosuz ağ üzerinden iletişim kurmaktadır. Bu nedenle, verilerin korunması ve gövde alan ağı içerisindeki iletişimin güvenliğinin sağlanması için bir güvenlik mekanizması gerekmektedir. Buna ek

olarak, biyosensörlerin donanımsal kaynak kısıtlamalarının üstesinden gelinebilmesi için oluşturulan güvenlik mekanizması fazla kaynak gerektirmemelidir. Kriptografik anahtar oluşumu ve biyosensörler arası anahtar anlaşmasının rasgele ve güvenli olması, bu güvenlik mekanizmalarının en önemli öğelerindedir.

Bu tezde, fizyolojik sinyaller kullanılarak güvenli ve rasgeleliği arttırılmış anahtar anlaşması sistemi (SKA-PSAR) önerilmiştir. Bu sistemin temel amacı biyosensörlerin gövde alan ağları içerisinde güvenli iletişim sağlayabilmesi için rasgeleliği yüksek kriptografik anahtarlar üretmektir. SKA-PSAR sistemi de, öncülü Karaoğlan Altop vd. tarafından önerilen SKA-PS protokolü gibi, fizyolojik sinyalleri (kalp atış hızı, kan basıncı, vb.) girdi olarak kullanmakta ve temel yapı taşı olarak küme uzlaşması mekanizmasından yararlanmaktadır. Yeni nicemleme ve ikilileştirme mekanizmaları ile daha rasgele anahtarlar üretilebilmesi, SKA-PSAR sistemini SKA-PS protokolünden ayırmaktadır. Bununla beraber, SKA-PSAR sistemi tarafından oluşturulan anahtarlar ayırt edicilik ve zamansal değişim özelliklerini taşımakta ve aynı zamanda yeterince uzun bit uzunlukları ile kriptografik ataklara karşı dayanıklılık göstermektedir. Buna ek olarak, %100 doğru anahtar oluşturma yüzdesi ve %0 yanlış anahtar oluşturma yüzdesi elde edilmiştir. Son olarak, önerilen protokolün hesaplama karmaşıklığı, iletişim karmaşıklığı and hafıza gereklilikleri SKA-PS protokolüne göre yüksek çıkmıştır; fakat yüksek rasgelelik içeren anahtarlar oluşturulması için bu gereklidir.

to my mother, father and Scarlet

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to Prof. Albert Levi, my advisor and one of the major contributors of this thesis, together with my co-advisor Dr. Duygu Karaođlan Altop. His guidance and endless patience has motivated me immensely. The feedback he has given me for the past couple months has been invaluable and I am extremely grateful for his contributions to this thesis. I would also like to thank my co-advisor Dr. Duygu Karaođlan Altop for her timely and actionable feedback and I am happy to have had the chance to work alongside her. I also extend my gratitude to the jury members Assoc. Prof. Selim Balcısoy, Assoc. Prof. Cemal Yılmaz, Asst. Prof. Kbra Kalkan akmakı for participating in my jury, reviewing my thesis and providing feedback.

TABLE OF CONTENTS

1 Introduction	1
1.1 Motivation	2
1.2 Contributions of the Thesis	3
1.3 Outline of the Thesis	4
2 Background and Related Work	5
2.1 Body Area Networks (BANs)	5
2.2 Biometrics	7
2.3 Physiological Signals in Health Monitoring	8
2.4 Set Reconciliation	9
2.5 HMAC	11
2.6 Bio-Cryptography in BAN Security	12
3 Proposed Key Agreement System: Secure Key Agreement using Physiological Signals with Augmented Randomness (SKA-PSAR)	21
3.1 Proposed IPI Sequence Generation Technique	23
3.2 Secure IPI Sequence Reconciliation (SISR) Protocol	24
3.3 Secure Key Agreement (SKA) Protocol	31
4 Performance Evaluation	37
4.1 Test Environment and Dataset	37
4.2 System Parameters	42

4.3	Correct Key Generation Rate (CKGR) and False Key	
	Generation Rate (FKGR)	44
4.4	Security Analysis	45
4.4.1	Threat Model	45
4.4.2	Randomness of the Generated Cryptographic Keys	46
4.4.3	Distinctiveness of the Generated Cryptographic Keys	48
4.4.4	Temporal Variance of the Generated Cryptographic Keys	49
4.5	Computational and Communication Complexity and Memory Requirements of SKA-PSAR	49
4.5.1	Maximum Number of Candidate IPI Sequences Generated in SISR Protocol	50
4.5.2	Computational Complexity	51
4.5.3	Communication Complexity	52
4.5.4	Memory Requirements	53
5	Discussion On the Comparison of the SKA-PSAR system and the SKA-PS protocol	55
5.1	Disparities between SKA-PSAR and SKA-PS	55
5.2	Performance Comparison of SKA-PSAR and SKA-PS	58
5.2.1	Key Generation Rates: CKGR and FKGR	59
5.2.2	Randomness Tests	60
5.2.3	Maximum Number of Candidate IPI Sequences Generated	60
5.2.4	Distinctiveness and Temporal Variance	61
5.2.5	Computational Complexity, Communication Complexity and Memory Requirements	62
6	Conclusions	66

LIST OF TABLES

2.1 Polynomial Evaluations For the Example Above	10
3.1 Symbols used in SKA-PSAR system	22
3.2 Relation of the bit length and the minimum/maximum IPI values	33
3.3 Gray Codes	33
4.1 Statistics of IPI Distributions	39
4.2 General Statistics of IPI Distributions	39
4.3 Key sizes obtained from SKA-PSAR system	42
4.4 Parameters used our SKA-PSAR system	44
4.5 Correct Key Generation Rates and False Key Generation Rates of SKA- PSAR system	44
4.6 NIST Test Suite Results of SKA-PSAR System	48
4.7 Maximum number of candidate IPI sequences generated in SISR protocol	50
4.8 Average latency on Macbook Pro	52
4.9 Average latency on Raspberry Pi3	52
4.10 Average communication complexity (KB)	53
4.11 Average memory requirement (MB)	54
5.1 Key Generation Rates of SKA-PS and SKA-PSAR	59
5.2 NIST Test Suite Results of SKA-PS and SKA-PSAR	60
5.3 Maximum number of candidate IPI sequences generated	61
5.4 Average latency on Raspberry Pi3	62

5.5 Average latency on Macbook Pro	63
5.6 Average communication complexities (KB)	64
5.7 Average memory requirements (MB)	65

LIST OF FIGURES

2.1	General infrastructure of a BAN	6
2.2	Matching scores of biometrics (retrieved from [34])	8
2.3	ECG-PPG-BP Signals (retrieved from [16])	9
2.4	Feature Generation Method of EKA [44]	15
2.5	Physiological Parameter Generation Technique of SKA-PS	19
3.1	Overview of SKA-PSAR System	23
3.2	IPI Sequence Generation Technique	23
3.3	One Round of our proposed SISR Protocol	26
3.4	Our Proposed SISR Protocol	26
3.5	Our Proposed SKA Protocol	31
3.6	Key Agreement	36
4.1	IPI Distribution of BP signal of Subject 1	40
4.2	IPI Distribution of BP signal of Subject 3	41
4.3	IPI Distribution of BP signal of Subject 8	41

Chapter 1

Introduction

Rapid technological advancements in recent history have allowed medical patients' ongoing conditions and well-being to be observed in real-time through the use of small, low-power wearable devices named *biosensors*. Biosensors act as connected nodes on the body in a network named Body Area Network (BAN) [27, 47, 43]. Through these biosensors, a BAN thoroughly collects critical medical information (blood pressure, heart rate, etc.) about the subject in real-time, sends them to remote healthcare professionals, allowing decisions to be made by the professionals based on what has been captured.

It is important to note that a BAN operates in a wireless environment. Due to this, even though it becomes much easier to remotely monitor the patient and acquire data, several other challenges also come up. Wireless networks are much more susceptible to outside attacks [41] and the critical nature of the stored information makes it extremely important to provide a secure network. This network should satisfy all principles of information security – confidentiality, integrity and availability (CIA). Any security issue in the system, which causes the patient's critical information to be disclosed, may result in the patient being harmed in various ways. For instance, illness of a high-profile individual might be made public that negatively affects his/her life. In another hypothetical scenario, a patient whose heart rate rapidly increases may not receive critical help due to an attacker rigging the network in a way that disguises this sudden change. As a result, BANs must be secured using a lightweight security mechanism.

1.1 Motivation

Biosensors are responsible for sending the collected sensitive information to a central server that is responsible for storing the data and sharing it with health professionals when necessary. Normally, a designated gateway is used to relay data towards central servers and multihop communication may be needed to reach the gateway in a BAN. Also, for pre-evaluations, the data may sometimes be collected to one of the biosensors. Therefore, a secure communication channel between the biosensors is obligatory. Although it is extremely critical for the communication of biosensors in a BAN to be as secure as possible, the power and memory constraints of the biosensors make BAN unsuitable for traditional cryptographic key generation algorithms, such as the ones using public key cryptography. Due to these constraints, a mechanism that makes the use of lightweight key generation protocols for providing the security of communication between biosensors must be employed. Due to limited input and output capabilities of the BAN devices, it would be very helpful if the key generation process is automatized. Utilizing biometrics of the individuals in the aforementioned protocol not only provides automation in key generation, but also produced cryptographic keys become unique to the individuals and differ from person to person. Randomness is one of the most essential characteristics of the cryptographic keys. The cryptographic keys generated from the biometric sources may suffer having sufficient randomness.

There are studies in the literature [44, 17] that tackle some of these characteristics, however a comprehensive protocol that satisfies the hardware and security constraints, especially with adequate randomness of the resulting cryptographic keys, does not exist.

1.2 Contributions of the Thesis

The main objective of this thesis is to develop a secure key agreement system that produces highly random keys for the communication between the biosensors in a BAN. For this purpose, we propose SKA-PSAR (Secure Key Agreement Using Physiological Signals with Augmented Randomness) system, which is based on SKA-PS [17] protocol. SKA-PSAR system is composed of three main parts: (i) IPI Sequence Generation Technique, (ii) Secure IPI Sequence Reconciliation (SISR) protocol and (iii) Secure Key Agreement (SKA) protocol. SKA-PSAR system uses physiological signals (blood pressure, electrocardiogram) on communicating biosensors as inputs and outputs the same symmetric cryptographic key on the communicating biosensors while utilizing the set reconciliation paradigm similarly as they are used in SKA-PS [17]. Moreover, as in SKA-PS, SKA-PSAR also relies on the fact that biosensors placed on the same individual generates similar IPI (Inter-Pulse Interval) values; in other words, the distance between the calculated peaks of the physiological signals are almost identical. SKA-PSAR differs from SKA-PS in that it generates more random keys as the result of the novel quantization and binarization methods.

We evaluated our proposed system on correct key generation rate, false key generation rate, randomness, distinctiveness, temporal variance, computational complexity, communication complexity and memory requirements in a comparative way with SKA-PS. Randomness of the generated keys are evaluated using NIST Test Suite [5]. Hamming Distance metric is utilized to calculate the distinctiveness and the temporal variance of the generated keys. Performance evaluations are measured on Macbook Pro and Raspberry Pi3. All these analyses show that correct key generation rates are high while false key generation does not exist in both models. SKA-PSAR system not only creates highly random cryptographic keys, but also generates long, time variant and distinctive cryptographic keys. On the other hand, the keys generated by SKA-PS possess lower randomness as compared to SKA-PSAR. The computational and communication complexities, and memory requirements are much higher in SKA-PSAR, but this should be considered as a trade-off between randomness and operational performance.

1.3 Outline of the Thesis

The rest of this thesis is organized as follows. Chapter 2 includes the background information for understanding the basis of the work and also the related work. In Chapter 3, we explain our proposed SKA-PSAR system that produces highly random cryptographic keys. Chapter 4 evaluates the performance of our proposed SKA-PSAR system on correct key generation rate, false key generation rate, computational complexity, communication complexity and memory requirements together with the randomness, time variance and distinctiveness of the generated keys. In Section 5, the differences between our proposed SKA-PSAR system and its predecessor SKA-PS protocol are discussed. Finally, Chapter 6 provides conclusions reached by this thesis.

Chapter 2

Background and Related Work

In this chapter, we first explain the Body Area Networks (BANs) with their infrastructure, application areas, and security and privacy concerns in Section 2.1. Then, we define biometrics with the performance evaluation methods in Section 2.2. After that, physiological signals that are used in health monitoring are explained and depicted in Section 2.3. Thereafter, the basic cryptographic building blocks of our proposed system, which are set reconciliation and HMAC (Hash-based Message Authentication Code), are explained in Section 2.4 and Section 2.5, respectively. Finally, in Section 2.6, we discuss about the related work in the literature: We first explain the usage of bio-cryptography in BAN security, and we discuss the key generation methods that utilize physiological signals, including the details of the SKA-PS protocol [17], on which our proposed protocol is built.

2.1 Body Area Networks (BANs)

BANs are wireless sensor networks that utilize wearable devices [22, 30, 21], used in healthcare [10, 46], entertainment [1] and military areas [32]. A BAN consists of Body Sensor Units (BSU) and a Body Central Unit (BCU). The former is named biosensor and the functionality of it includes the monitoring of the health of the subject by sensing physiological signals, such as blood pressure (BP) or electrocardiogram (ECG), or by sensing

the motion of the subject. The latter is named an aggregator and it serves as a data collector. The aggregator also communicates with a central server that is responsible for storing the data collected from the biosensors. In addition to these, the general infrastructure of a BAN also includes a health professional, whose responsibility is retrieving and analyzing the data from the central server. Figure 2.1 demonstrates the general infrastructure of a BAN.

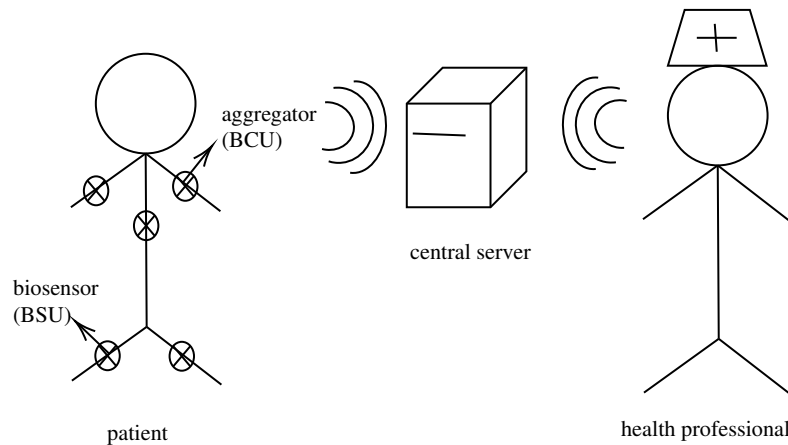


Figure 2.1: General infrastructure of a BAN

There are two kinds of communications in a BAN: intra-BAN and beyond-BAN. Intra-BAN communication involves the communication among the biosensor, and between the biosensors and the aggregator. On the other hand, beyond-BAN communication defines the communication between the central server and the aggregator. In this thesis, beyond-BAN communication will not be in scope.

Since biosensors are communicating with each other using wireless medium, they are prone to both passive and active attacks [41]. A passive attack might violate the confidentiality and the privacy of the collected data and this might result in the data being public and accessible by non-authorized people. An active attack might destroy the integrity, authentication and non-repudiation by enabling the intruder to modify the content or the sender of the data. Since healthcare information is extremely critical and should not be compromised under any means, a security solution is of great importance.

BCUs, such as mobile phones and personal digital assistants (PDAs), are assumed to have more memory and computational power than the BSUs [49]. However, BSUs have limited memory and low computational power compared to BCUs. Due to the power and memory constraints of the biosensors, public key cryptography is not suitable for intra-BAN communications. Therefore, a light-weight secure key creation protocol is needed.

2.2 Biometrics

Biometrics is the study of methods that analyses the human characteristics [29]. One of the important differences between the biometrics and the conventional cryptography is that the traditional cryptography requires to have a known secret, such as a password, or a possession, like a key, while biometrics provides security using distinctive characteristics of individuals. Another difference of biometrics is that it cannot be lost, stolen, forgotten or transferred as the conventional cryptographic keys can be. The biometrics can be classified into distinctive characteristics and behavioral characteristics. The examples of distinctive characteristics are fingerprint, iris recognition and face recognition, and the examples of behavioral characteristics are the gait features and signature.

The performance of a biometric system can be measured using the metrics False Accept Rate (FAR), False Reject Rate (FRR), and Equal Error Rate (EER). FAR is defined as authenticating an unauthorized person, while FRR is being unable to authenticate an authorized person, and EER is described as the rate that both FRR and FAR are equal to each other. There is a tradeoff between the FAR and FRR. Figure 2.2 represents the matching scores based on the similarity measure, which is used to decide if the two biometric trait samples are obtained from the same individual. The right curve in Figure 2.2 illustrates the similarity scores of biometric features obtained from the same individual and the left curve illustrates the similarity scores of biometric features of different persons. Threshold is used to determine the trade off between FAR and FRR.

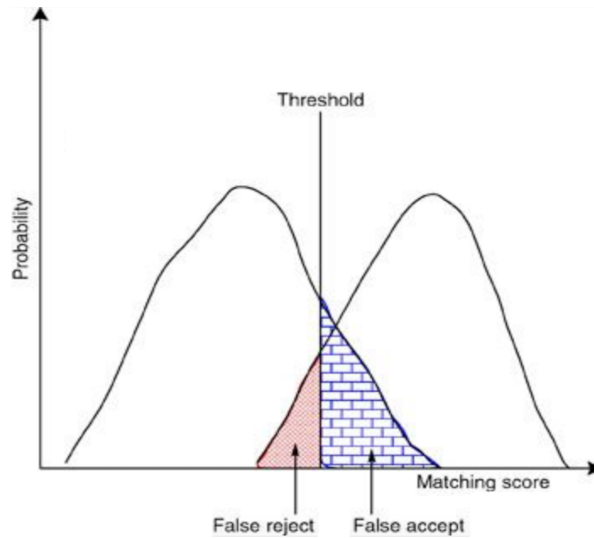


Figure 2.2: Matching scores of biometrics (retrieved from [34])

2.3 Physiological Signals in Health Monitoring

Physiological signals such as blood pressure (BP), electrocardiogram (ECG), oxygen saturation (PPG), body temperature (BT), ballistocardiogram (BCG) and posture muscle activation (EMG) are being utilized in health monitoring systems in order to keep track of the patients' health status.

ECG, which is the graphical representation of electrical activity of the heart in a time period, is one of the most crucial body signals for detecting signs of heart diseases [12]. It is measured using electrodes placed on different parts of the body of an individual. On the other hand, BP is defined as the pressure of the blood on the walls of blood vessels and increase in BP indicates a heavy workload of the individual's heart. The common characteristics of these signals, BP and ECG, is that they represent the cardiac cycle of a human [6]. IPI (inter-pulse interval) is an important indicator in all of these cardiovascular signals and is defined as the time elapsed between the consecutive nerve impulses. The representations of these signals and the concept of IPI can be seen in Figure 2.3. ECG and BP signals are utilized in this thesis since our dataset includes these aforementioned signals.

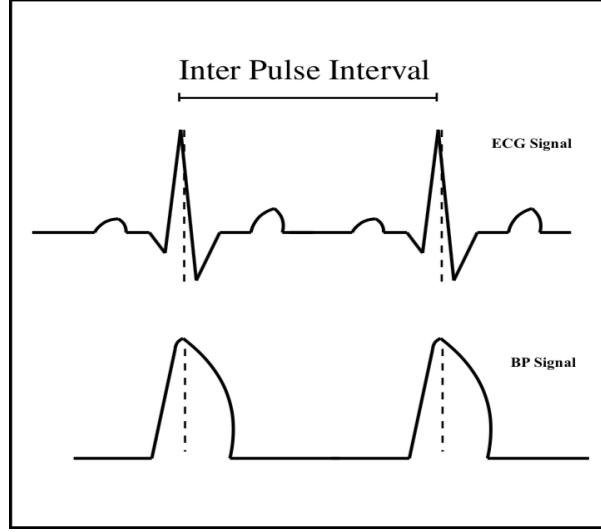


Figure 2.3: ECG-PPG-BP Signals (retrieved from [16])

2.4 Set Reconciliation

Set reconciliation is an approach that enables to reconcile similar sets on different hosts while minimizing the computational and communication complexity [25]. Considering *Host A* and *Host B*, each having a set of length b bitstrings, S_A and S_B , where the difference of S_A from S_B is denoted as ΔA , and the difference of S_B from S_A is denoted as ΔB , with lengths of ΔA and ΔB being indicated as m_A and m_B , respectively, the set reconciliation protocol is explained as follows:

1. *Host A* and *Host B* create characteristic polynomials, defined as the univariate polynomial in Equation 2.1, $X_s(Z)$ of their sets $S = \{x_1, x_2, \dots, x_n\}$ on some field F_q , where q is prime and $q \geq 2^b$.

$$X_s(Z) = (Z - x_1)(Z - x_2)\dots(Z - x_n) \quad (2.1)$$

2. *Host A* evaluates $X_{S_A}(Z)$ and *Host B* evaluates $X_{S_B}(Z)$ using the same evaluation points, where the number of evaluation points is $\bar{m} \geq m_A + m_B + 1$.
3. *Host B* sends its evaluations $X_{S_B}(Z_i)$, $1 \leq i < \bar{m}$, to *Host A*.

4. Combining the evaluations, $\frac{X_{S_A}(Z)}{X_{S_B}(Z)}$ is computed at each evaluation point by *Host B*.
5. The results of the previous step are interpolated in order to recover the coefficients of the reduced rational function, defined as the rational function after simplifying the common factors of the numerator and denominator, $\frac{X_{\Delta_A}(Z)}{X_{\Delta_B}(Z)}$.
6. Factorization of X_{Δ_A} and X_{Δ_B} reveals the elements of Δ_A and Δ_B .

Considering that $q = 97$, $m_A = m_B = 1$, and *Host A* and *Host B* have the following sets $S_A = \{3, 4, 5, 6\}$ and $S_B = \{3, 4, 5, 7\}$, respectively, \bar{m} is calculated as 3, as explained in Equation 2.2. Thus, the evaluation points Z will include 3 points. Letting $Z = \{-1, -2, -3\}$, the characteristic polynomials of *Host A* and *Host B* are created as in Equation 2.3. Polynomial evaluations and their division can be seen in Table 2.1. After recovering the reduced rational function $\frac{X_{\Delta_{S_A}}(Z)}{X_{\Delta_{S_B}}(Z)}$, the roots can be obtained as 6 and 7, since the different set elements of S_A and S_B are less than 2.

$$\bar{m} \geq m_A + m_B + 1, m_A = 1, m_B = 1 \quad (2.2)$$

$$\begin{aligned} X_{S_A}(Z) &= (Z - 3)(Z - 4)(Z - 5)(Z - 6) \\ X_{S_B}(Z) &= (Z - 3)(Z - 4)(Z - 5)(Z - 7) \end{aligned} \quad (2.3)$$

$Z =$	-1	-2	-3
$X_{S_A}(Z)$	64	31	17
$X_{S_B}(Z)$	87	47	62
$X_{S_A}(Z)/X_{S_B}(Z)$	13	44	30

Table 2.1: Polynomial Evaluations For the Example Above

2.5 HMAC

HMAC (Hash-based Message Authentication Code) is a mechanism that provides message authentication utilizing cryptographic hash functions [19]. While HMAC can be applied using any cryptographic hash function, the strength of it highly depends on the security of the underlying hash function. Since SHA-256 is considered as a secure hash function to the date of writing this thesis, it can be used as the underlying hash function for HMAC. In addition to the hash function, HMAC also uses a secret key, whose length can be anything up to the data block size B , in the calculations. Considering a message M , a hash function H , a secret key K , and two fixed and different strings $ipad$ and $opad$, where $ipad$ is equal to the B times of byte $0x36$ and $opad$ is equal to the B times of byte $0x5C$. For calculating HMAC of M in Equation 2.4, the following steps are needed:

1. Zeros is appended to K until the size of K becomes equal to the block size of B , if the size of K is smaller than the block size B .
2. XOR operation is performed between the B -byte result of Step-1 and $ipad$.
3. M is appended to the XOR result from Step-2.
4. H is applied to the result of Step-3.
5. XOR operation is performed between the B -byte result of Step-1 and $opad$.
6. The result of Step-5 and the result of Step-4 is concatenated.
7. H is applied of the result of Step-6.

$$\text{HMAC}(M) = H(K \oplus opad || H(K \oplus ipad || M)) \quad (2.4)$$

2.6 Bio-Cryptography in BAN Security

Bio-cryptography [42] is the combination of biometrics and traditional cryptosystems, where the former contains methods that investigate the human characteristics [29], while the latter involves methods to provide authentication by a secret key. Biometric keys are superior than pure cryptographic keys since they cannot be forgotten, lost or stolen. Hence, bio-cryptography can be applied on biosensors in order to provide the security of the communication among them.

The communication security between the biosensors can be supplied by fuzzy cryptography, which means that the generated keys on the biosensors do not need to be identical but should be similar with a tolerable threshold [15]. Fuzzy cryptography can be also divided into fuzzy based key binding and key generation. Both in key binding and key generation algorithms, physiological signals are utilized to produce pseudo random numbers. However, the difference between the key generation and key binding algorithms is that the former generates the cryptographic keys directly from the pseudo random numbers obtained from the physiological signals [44, 17, 40, 26], while the latter use those pseudo random numbers in order to conceal the cryptographic key generated from the traditional cryptographic algorithms [9, 3, 7, 45]. As the main focus of this thesis is the key generation algorithms, examples from them are discussed in the rest of this section.

Using physiological signals for BAN security was first introduced by Venkatasubramanian in [9]. Also, it has been demonstrated that physiological signals could be suitable sources for cryptographic key generation in the previous works [16, 31]. The authors in [4] used inter-pulse interval (IPI) for the first time as a biometric characteristic in order to identify an individual. On the other hand, the requirements of a cryptographic key is explained in [31]. One of the most important characteristics of a cryptographic key is being random. Ortiz-Martin et al. [28] claims that the IPIs obtained from the physiological signals do not possess sufficient randomness. In order to increase the randomness of the generated keys, Rostami et al. [35] suggest to extract the 4 least significant bits from the quantized IPI values. However, the mean and the standard deviation of the IPI values generated from the physiological signals of each individual vary. For this reason,

extracting fixed amount of bits may decrease the randomness of the generated keys and a dynamic method is needed to determine the bit length retrieved in each IPI value.

Seepers et al. [37] propose a key-exchange protocol that aims to circumvent heartbeat mis-detection by removing IPI values in a block of IPIs, that are far from the mean of the block. Chen [8] studies electroencephalogram (EEG) signals and introduces a transformation method that increases the randomness of the generated binary sequence. In Chen's method, the least significant five bits of the EEG sample amplitude are summed up, the modulo 2 is applied to the sum to obtain one or zero as a result. Bao et al. [2] propose a key generation method for BAN. In this method, they first accumulate m IPI values, and then modulo operation ($\text{mod}(2^p)$) is applied on the accumulation result. Finally, the modulo result is mapped to a smaller range $f : [0, 2^p) \rightarrow [0, 2^q)$ using the Equation 2.5. Seepers et al. [36] propose using Von Neumann extractor to increase randomness in physiological key generation. In this method, Von Neumann extractor, a function that produces output bit $x_{out} = x_0$ if and only if two consecutive input bits x_0 and x_1 is not equal to each other, otherwise x_0 and x_1 are discarded, is applied on the most significant bits of the IPIs.

$$f(m) = \frac{m}{2^{p-q}} \quad (2.5)$$

Moosavi [26] proposes a key generation approach that combines several features of ECG signals as the source of the generated cryptographic key, named as SEF (several ECG features)-based cryptographic key generation. The authors assert that the execution time of SEF is faster than the IPI-based cryptographic key generation protocols. SEF includes a dynamic approach for deciding the number of bits to extract from each ECG feature. For this purpose, the mean (m) and the standard deviation (std) of the feature set are calculated. Then, coefficient of variation (C_v), as seen in Equation 2.6, is obtained. Finally, the number of bits (M) to extract from each feature is decided using the Equation 2.7. Thus, x -bit binary value is generated using the aforementioned method. Finally, the generated bit sequence is strengthened using Fibonacci linear feedback shift register and advanced encryption standard algorithms.

$$C_v = m/std \quad (2.6)$$

$$M = \frac{\ln(std)}{\ln(2)} + C_v \quad (2.7)$$

The authors of SEF assert that the range of each ECG feature differs and also the ranges are not the same in different datasets. Therefore, they apply a dynamic technique in order to determine the bit length that can be extracted from each feature. However, the mean and the standard deviation of the physiological features do not only differ in different datasets but also differ in each individual's data. For this reason, dynamic bit extraction should be applied in each data separately. Also, the authors suggest to retrieve approximately 16 binary values from one heartbeat cycle. For instance: 2 bits from PR, 4 bits from RR, 4 bits from PP, 4 bits from QT and 2 bits from ST interval is retrieved in Motion Artifact ECG dataset. However, the paper does not include any analysis that the obtained bits from different features of ECG signal do not repeat themselves. They only admit that the NIST Test Suite [5] results are better after key strengthening is being applied on the extracted bit sequence from the ECG feature.

Venkatasubramanian et al. [44] propose a key generation algorithm (EKA: ECG-based Key Agreement) that enables two biosensors to create the same cryptographic key utilizing ECG signals. EKA scheme includes two phases: (i) feature generation, and (ii) key agreement. The steps of feature generation method of EKA is given in Figure 2.4. For feature generation, frequency-domain analysis of ECG signal is performed by sampling the ECG signals in both biosensors simultaneously at 125 Hz for 5 minutes. After removing the noise from the measurements, the total of 625 samples divided into 5 parts of 125 samples each. Then, a 128 point Fast Fourier Transform (FFT) is applied on each parts. After, a feature vector is constructed using the first 64 FFT coefficients of each part, thus a total of 320 coefficients are obtained.

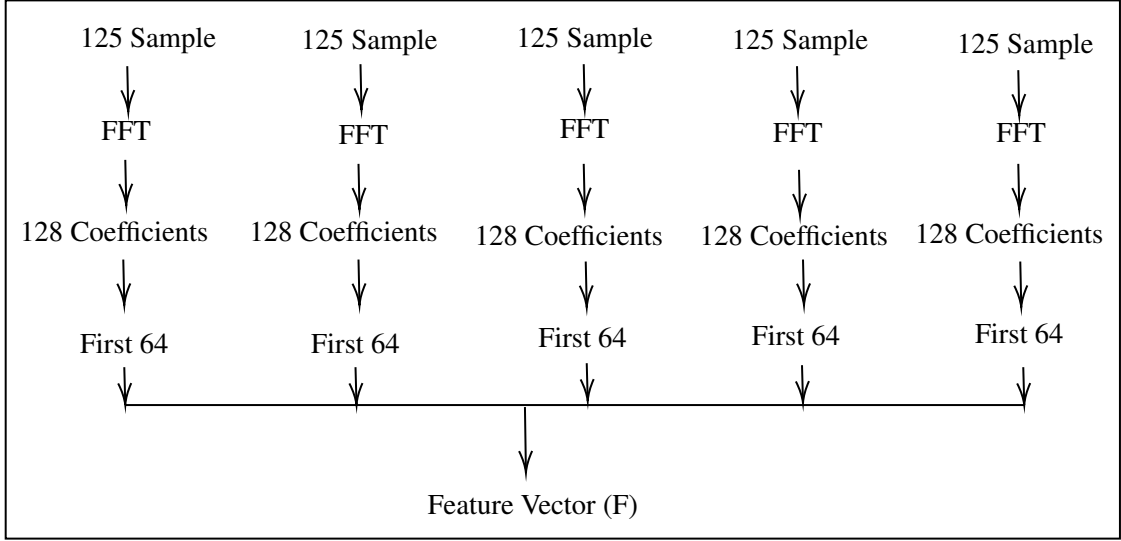


Figure 2.4: Feature Generation Method of EKA [44]

After generating the feature vector F , in order to generate the binary key from F , a quantization method is applied. Vector F , which has 320 coefficients in it, is divided into 20 blocks, each containing 16 coefficients. Then, using exponential quantization function, 4 bit value from each coefficient is obtained. As a result of the quantization method, 20 blocks of 64 bit values are produced.

After the feature generation is completed on both biosensors, the resulting blocks are exchanged between them. The key agreement method of EKA includes three phases: (i) commitment phase, (ii) processing phase, and (iii) de-commitment phase. In the commitment phase, each block is hashed (using SHA-256) in order not to reveal the key and then the hashed blocks are transmitted to the other biosensor. The transmitted message (M) includes node ids (ID), nonce (N), hashes of the blocks (total 20 blocks), and also a MAC and a random key K_R to detect adversaries. Equation 2.8 gives the message transmitted from A to B , and Equation 2.9 gives the message transmitted from B to A .

$$\begin{aligned}
 M_A = & ID_A \| N_A \| hash(b_1^A), hash(b_2^A), \dots, hash(b_{20}^A) \\
 & \| MAC(Key_R^A \| ID_A \| N_A \| hash(b_1^A), hash(b_2^A), \dots, hash(b_{20}^A))
 \end{aligned} \tag{2.8}$$

$$\begin{aligned}
M_B &= ID_B \| N_B \| \text{hash}(b_1^B), \text{hash}(b_2^B), \dots, \text{hash}(b_{20}^B) \\
&\| \text{MAC}(Key_R^B \| ID_B \| N_B \| \text{hash}(b_1^B), \text{hash}(b_2^B), \dots, \text{hash}(b_{20}^B))
\end{aligned} \tag{2.9}$$

In the processing phase of key agreement, a new matrix W is computed on both biosensors that includes the hamming distances of hashed blocks. Then, this matrix is used to determine the block indices that include the same values on both biosensors. After that, these blocks are hashed in order to create the symmetric keys (Key_A, Key_B) on the biosensors. In the final phase (de-commitment) of EKA, the legitimacy of the blocks are checked by exchanging message G . Equation 2.10 represents the message sent from biosensor A to B , and Equation 2.11 represents the message sent from biosensor B to A .

$$G_A = Key_R^A \oplus Key_A \| \text{MAC}_A(Key_A \| G_A) \tag{2.10}$$

$$G_B = Key_R^B \oplus Key_B \| \text{MAC}_B(Key_B \| G_B) \tag{2.11}$$

Since $Key_A = Key_B$, each biosensor uses their generated keys for the verification of message G . If the verification is successful, they perform an XOR operation using their generated keys to retrieve the Key_R . Finally, Key_R is used to verify the message (M), which was received in the commitment phase. If the second verification is successful, a temporary key K_{temp} is created from the generated keys and a random number l as seen in Equation 2.12. K_{temp} is used for the communication between the biosensors.

$$K_{temp} = \text{hash}(Key_A, l) = \text{hash}(Key_B, l) \tag{2.12}$$

Exchanged hashes in the commitment phase of EKA is only 64 bit long and can easily be broken by a brute force attack. Therefore, the authors of EKA suggest key strengthening, e.g. hashing the blocks 2^n times before exchanging. However, key strengthening increases the computational cost severely as the authors already have noticed.

On the other hand, Shi et al. [40] suggest an energy efficient key agreement protocol, BodyKey, for biosensors. BodyKey utilizes set reconciliation in order to overcome

the variations of biometrics measured by different biosensors placed on the same individual. The purpose of BodyKey is to reduce the energy consumption by exchanging only the necessary information for creating a symmetric key between the biosensors. System model of BodyKey consists of a group of wireless biomedical sensors; one of them has a rechargeable battery and stronger computational power and it acts as a control sensor. The control sensor is responsible for collecting the information from the other sensors and sending it to an external server. BodyKey consists of three steps: (i) feature extraction, (ii) key encoding, and (iii) key decoding.

In the feature extraction step, each biosensor measures the same physiological signals and extracts the biometric features from them. In the key encoding step, the control biosensor creates a symmetric key K from its biometric features X and sends public reconciliation information (PRI) to the other biosensors in BAN. For this purpose, the control sensor generates m original pairs using the ordered set of biometric features as $X = \{(1, X_1), (2, X_2), \dots, (m, X_m)\}$. Then, it computes s integers via Lagrange Interpolation [38], as given in Equation 2.13, where $s = 2(m - t) < m$, t is the threshold, $|X \cap Y| \geq t$, and $X \cap Y$ is the number of the elements in the intersection of X and Y . Then, PRI , as given in Equation 2.14, is constructed using those s integers with a public positive integer λ and a c value as the hashed value of $X + \lambda$, X is the concatenation of m elements. After broadcasting PRI to the other biosensors, the control sensor creates the cryptographic key (K) by utilizing another hash function.

$$f(Z) = \sum_{i=1}^m \left(\prod_{j=1, j \neq i}^m \frac{Z - j}{i - j} x_i \right) \quad (2.13)$$

$$PRI = \{f(m + 1), f(m + 2), \dots, f(m + s), \lambda, c\} \quad (2.14)$$

In the final step, i.e. key decoding, the destination biosensors generate their feature vectors (Y) from their physiological signals. Then, if their feature vectors (Y) are close enough to X , X can be regenerated from their feature vectors Y and the received PRI applying Reed Solomon decoding [24]. For this purpose, the destination sensors combine

their m points with the received s pairs to obtain a group of $s + m$ points. Utilizing Reed Solomon decoding, a polynomial F of degree m such that at least $s + t$ pairs lied on the polynomial is searched. If there is no such polynomial, then the protocol terminates. If F is found, then the verification is performed by constructing hashed value of c' . If verification is successful, then the same K generated by the control sensor can be constructed by the following hash function $H(F + \lambda)$.

For the performance evaluation of BodyKey, 290 subjects are retrieved from PhysioBank Database [20]. The authors report that BodyKey consumes low energy. However, the evaluations do not include any randomness tests.

Karaođlan Altop et al. [17] suggest SKA-PS (Secure Key Agreement using Physiological Signals) protocol. Our proposed SKA-PSAR system is build on the SKA-PS protocol that produces cryptographic keys using physiological signals of the users, such as blood pressure (BP), electrocardiogram (ECG) and photoplethysmogram (PPG). IPI values are utilized in the protocol in order to generate the key between the biosensors that are placed on the same individual. SKA-PS is based on the set reconciliation paradigm [25] that enables to reconcile two similar sets on the different sides of a communication. With the help of set reconciliation, different biosensors on the same individual create the same cryptographic keys by exchanging polynomial evaluations of their IPI sequences.

Physiological parameter generation technique of the SKA-PS protocol includes (i) peak detection, (ii) IPI calculation, (iii) quantization, and (iv) binarization steps, as given in Figure 2.5. In the peak detection step, the peaks of the physiological signals are detected. Then, the time difference between the consecutive peaks of the signals are extracted in order to construct the initial IPI sequences. After that, in the IPI calculation step, each successive g IPI values in the initial IPI sequences are summed up for decreasing the measurement errors of the signals, where g is a system parameter. Afterwards, in the quantization step, circular uniform quantization with a step size is applied on the resulting IPI sequences that are generated from the IPI calculation step: After the IPI calculation step, generated IPI sequences are divided into blocks using a step size s , and each block is mapped to a value from the set $\{0, 1, \dots, 2^{128/\frac{l}{g}-1}\}$, where l/g is the length of the IPI se-

quence. For instance, if $IPI_{min} = 1$, $IPI_{max} = 60$, $s = 8$, and $l/g = 64$, then the partitions will be $\{1 - 8, 9 - 15, \dots, 53 - 60\}$, and the IPI values in the first, fifth, ninth partitions will be assigned to 0, and second, sixth, tenth partitions will be assigned to 1, and so on. Finally, in the binarization step, each quantized IPI value is converted into binary using Gray encoding [23].

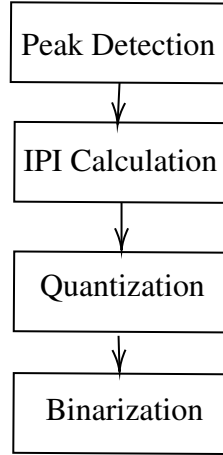


Figure 2.5: Physiological Parameter Generation Technique of SKA-PS

The general methodology of SKA-PS employs two biosensors: source biosensor and conforming biosensor. Source biosensor sends the polynomial evaluations of its characteristic polynomials that are generated using IPI values, to the conforming biosensor. Then, conforming biosensor regenerates the source biosensor's quantized IPI sequence using its own polynomial evaluations together with the polynomial evaluations received from the source biosensor. SKA-PS protocol runs in a round manner and in each round, the source biosensor sends its polynomial evaluations to the conforming biosensor. Conforming biosensor applies set reconciliation in order to reconcile the same set of quantized IPI values of the source biosensor. If the set reconciliation is successful for required number of sets, the key is generated and the conforming biosensor sends a positive acknowledgment together with the key index of the matching sets; otherwise, it sends a negative acknowledgement message to the source biosensor.

The performance of the SKA-PS protocol is evaluated through the match rates (i.e., True Match Rate, False Match Rate and Half Total Error Rate), randomness, distinctive-

ness and temporal variance of the generated cryptographic keys, together with computational and communication complexity, and memory requirements. The authors of SKA-PS report promising performance results; however, their randomness tests rely only on Shannon's entropy [39] and it does not provide good randomness in NIST Test Suite [5]. Moreover, sending the key index in positive acknowledgment causes information leak in SKA-PS. The intruder might use this index information in his/her brute-force attacks.

Chapter 3

Proposed Key Agreement System: Secure Key Agreement using Physiological Signals with Augmented Randomness (SKA-PSAR)

In this chapter, we present our proposed Secure Key Agreement using Physiological Signals with Augmented Randomness (SKA-PSAR) system which is used for creating a highly random cryptographic key between two biosensors: *source biosensor* and *conforming biosensor*. Our proposed SKA-PSAR system is build on the SKA-PS protocol [17] and it aims to enhance the randomness of the cryptographic keys generated by it. The differences and similarities between SKA-PSAR and SKA-PS will be explained in Section 5 in detail.

SKA-PSAR system consists of three main parts: (i) IPI (Inter-Pulse Interval) Sequence Generation Technique, (ii) Secure IPI Sequence Reconciliation (SISR) protocol and (iii) Secure Key Agreement (SKA) protocol. Firstly, IPI Sequence Generation Technique, explained in Section 3.1, is used to produce the IPI sequences from the physiological signals and the generated IPI sequences will be the input of the SISR protocol. Secondly, SISR protocol is described in Section 3.2. In SISR protocol, source biosensor

provides the IPI sequences which will be the source of the cryptographic key. Then, conforming biosensor attempts to regenerate the source biosensor's IPI values by applying the set reconciliation paradigm, which is explained in Section 2.4. Finally, SKA protocol is explained in Section 3.3. In SKA protocol, our novel *quantization* and novel *binarization* methods are applied on the reconciled IPI sequence that is generated as the result of the SISR protocol. As a result of the SKA protocol, source biosensor and conforming biosensor agree on the same cryptographic key. Table 3.1 is provided for the descriptions of the symbols from our proposed SKA-PSAR system and the overview of the SKA-PSAR system is illustrated in Figure 3.1.

Table 3.1: Symbols used in SKA-PSAR system

<i>Symbol</i>	<i>Description</i>
l	Length of the initial IPI Sequence
g	Size of the IPI groups
PP	Reconciled IPI Sequence
CP	Characteristic polynomial (from set reconciliation)
PE	Polynomial evaluations (from set reconciliation)
E	Evaluation points (from set reconciliation)
DE	Divided evaluations (from set reconciliation)
n	Total number of sets utilized for secure key generation
s	Number of elements in each set
r	Required number of sets for secure key generation
u	Number of utilized sets in a specific round
d	Maximum number of different set elements tolerable in set reconciliation
m	Maximum number of different set elements tolerable by SKA-PSAR system
$minBits$	Minimum number of bits needed to represent an IPI
b	Binarization bit length selected in binarization step
bcg	Base Gray code value used in binarization step
K	Generated cryptographic key

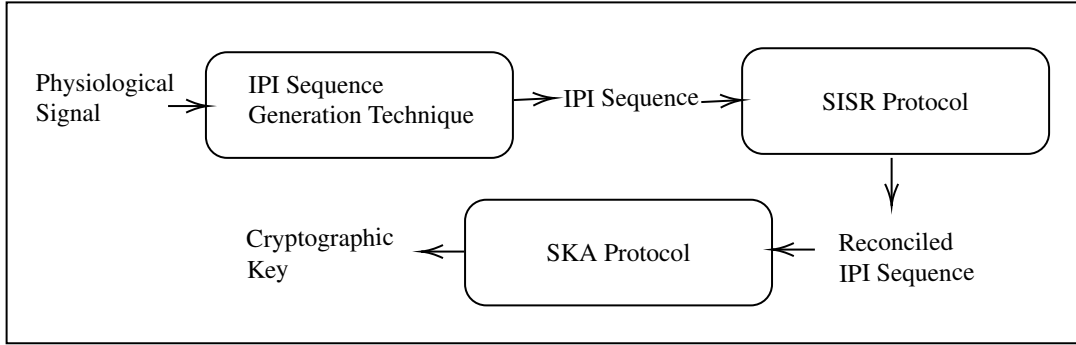


Figure 3.1: Overview of SKA-PSAR System

3.1 Proposed IPI Sequence Generation Technique

Our IPI Sequence Generation Technique, which is adopted from the SKA-PS protocol [17], is illustrated in Figure 3.2 and defined in Algorithm 1. Our IPI Sequence Generation Technique includes peak detection, IPI calculation and IPI accumulation steps. This technique can be used on any cardiovascular physiological signal that is proved to be used as a cryptographic key [18].

First of all, the peaks of the physiological signals that are measured within the same time interval on both biosensors are determined. Then, the duration between the consecutive peaks are calculated on both biosensors. As a result of this calculation, l IPIs are obtained. Then, each successive g IPIs are grouped together and summed up to decrease the measurement errors. After the accumulation operation, l/g IPIs are retrieved from the initial IPI sequence. This final IPI sequence, PP , is used as the input for the proposed SISR protocol, which is explained in detail in Section 3.2.

For example, let our measured IPI sequence be $\{221, 219, 219, 218, 218, 217, 220, 220, 220, 220, 217, 218, 230, 224, 226, 220\}$; assuming $g = 2$, final IPI sequence after grouping will be $\{440, 437, 435, 440, 440, 435, 454, 446\}$.

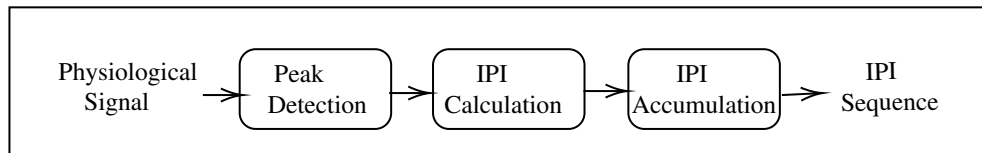


Figure 3.2: IPI Sequence Generation Technique

Algorithm 1 Proposed IPI Sequence Generation Technique

```
1: procedure GENERATEIPISEQUENCE(signal, g)
2:   ipiInitial, ipiGrouped = []
3:   peaks = findPeaks(signal)
4:   for i = 0 to len(peaks) - 1 do
5:     ipiInitial.append(peaks[i + 1] - peaks[i])    ▷ time difference is calculated
6:   end for
7:   for i = 0 to len(peaks)/g - 1 do
8:     groupTotal = 0
9:     for k = 0 to g - 1 do
10:      groupTotal += IPI[(i * g) + k]
11:    end for
12:    ipiGrouped.append(groupTotal)
13:  end for
14:  return ipiGrouped
15: end procedure
```

3.2 Secure IPI Sequence Reconciliation (SISR) Protocol

In this section, we describe our proposed SISR (Secure IPI Sequence Reconciliation) protocol in detail. SISR utilizes set reconciliation paradigm [25], which is described in Section 2.4. The input of our proposed SISR protocol is the IPI sequences calculated as the result of our IPI Sequence Generation Technique, which is described in Section 3.1. Any cardiovascular signal that is proved to be used as a cryptographic key can be used as an input to SISR protocol, as also discussed in Section 3.1. In our experiments, we have utilized BP (Blood Pressure) and ECG (Electrocardiogram) signals for the source and conforming biosensors, respectively, as described in detail in Section 4.

The ultimate purpose of our proposed SISR protocol is to generate a *reconciled IPI sequence* on the communicating biosensors, using the resulting IPI sequence of our proposed IPI Sequence Generation Technique. In brief, the source biosensor shares the polynomial evaluations (PE_s) of its IPI sequence (PP_s) with the conforming biosensor, and with set reconciliation, the conforming biosensor regenerates the source biosensor's IPI sequence (PP_s) using its own IPI sequence (PP_c). The source and conforming biosensors communicate with each other on the wireless medium during the SISR protocol. Considering the security issues on wireless environment, IPI sequences (PP) can not be shared

directly. Therefore, instead of PP , polynomial evaluations (PE) are transferred over the wireless medium to protect the sensitive information from being stolen.

First of all, before the SISR protocol starts, both of the biosensors divide their PP s to create n groups of each with s elements. The reason behind this grouping is the security concerns of the generated key. To clarify, as further discussed below, after finding the PP values that will be used to create the key on both of the communicating sides, the IPI values should be placed in the same order to get exactly the same sequence of IPI values at both parties at the very end. However, sorting the entire IPI values decreases the randomness and makes the key weak against brute force attacks. Therefore, IPIs are sorted only in their groups. Besides, biosensors need to agree on r sets for creating the reconciled IPI sequence. The value of r should be selected so as to fulfill the needs of the security level. Using greater r provides better security on the resulting key of the SKA-PSAR system, but less correct key generation rate between biosensors. More information about the selection criteria of the SKA-PSAR system parameters can be found in Section 4.2.

Our proposed SISR protocol runs in a round manner. One round of SISR is illustrated in Figure 3.3 and the steps of SISR are given in Figure 3.4. In each round, all combinations of r sets from u sets are utilized eliminating the combinations from the previous round, starting from $u = r$, until $u = n$, where r is the required set count and n is the maximum set count. To clarify, the index of the sets in the first round will be chosen as $\binom{n}{r}$, which indicates the first r sets of the PP of the biosensors. In the second round, the indices used in the previous round will be removed and the remaining indices will be used, $\binom{n}{r+1} - \binom{n}{r}$, and in the last round, $\binom{n}{n} - \binom{n}{n-1}$ combinations are employed. The process will end when the reconciled IPI sequence is successfully generated or when the total number of sets (n) is reached. The latter case means a protocol failure and reconciled IPI sequence is not generated. As an action to this failure, the subject can try another set of source IPI values, which means running the protocol from scratch with new input.

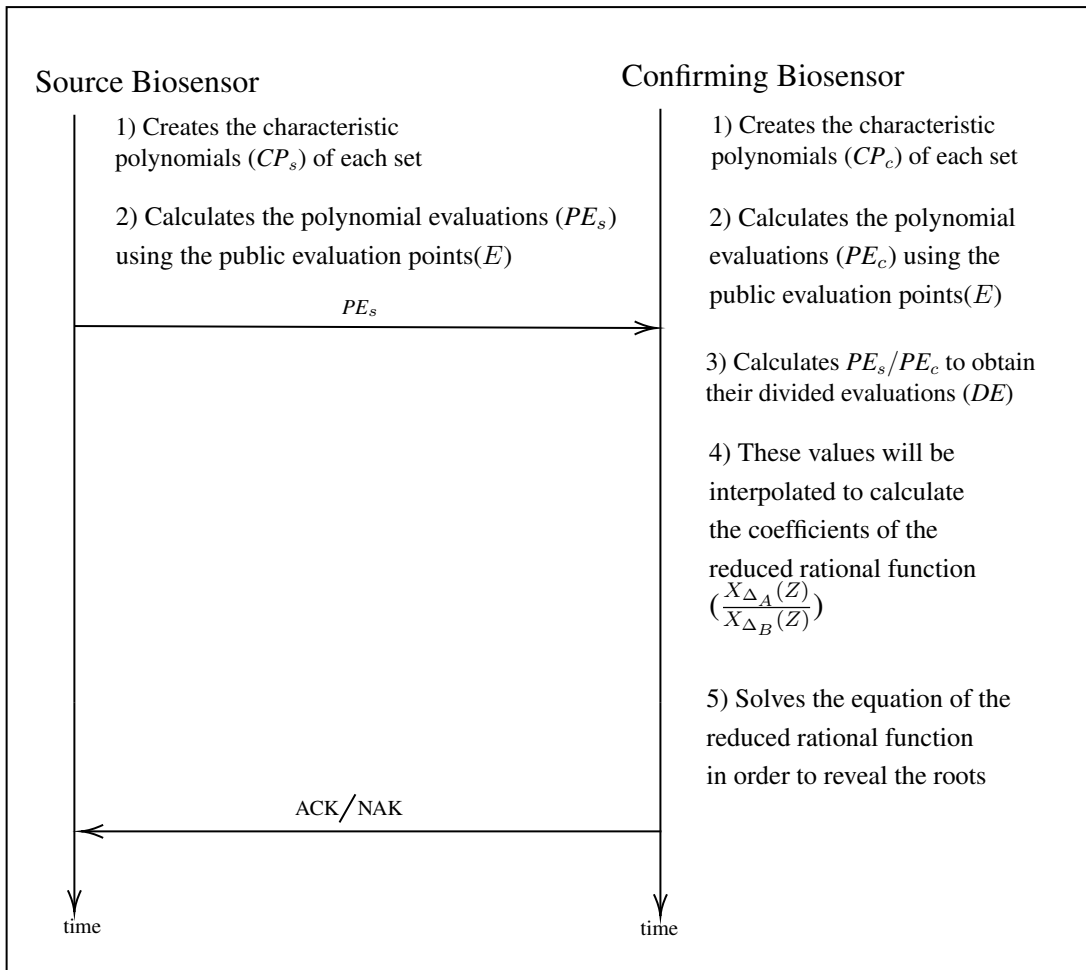


Figure 3.3: One Round of our proposed SISR Protocol

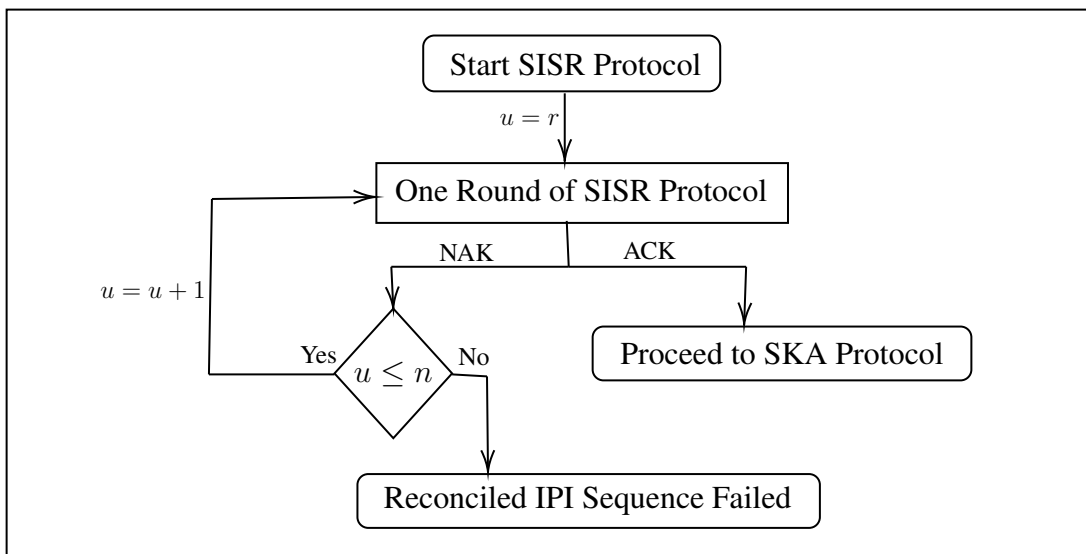


Figure 3.4: Our Proposed SISR Protocol

In each round of our proposed SISR protocol, the source biosensor generates the combinations of r sets from u sets, creates the characteristic polynomials (CP_s) of each set and calculates the polynomial evaluations (PE_s) using the public evaluation points (E). After that, the source biosensor sends its polynomial evaluations of the chosen r sets to the conforming biosensor. Source sensor operations can be seen in Algorithm 2.

Algorithm 2 Source Sensor Operations

```

1: procedure RUNPROTOCOL( $PP_S, r, n, E$ )
2:    $foundKey = false$ 
3:    $u = r$ 
4:   while  $foundKey = false$  and  $u \leq n$  do
5:      $allIndices = selectIndices(u)$ 
6:     for each  $indices \in allIndices$  do
7:        $CP_S = createCP(PP_S, indices)$ 
8:        $PE_S = findPE(CP_S, E)$ 
9:        $send(PE_S)$ 
10:       $foundKey = receive()$ 
11:      if  $foundKey = true$  then
12:        return  $u$ 
13:      end if
14:    end for
15:     $u+ = 1$ 
16:  end while
17:  return  $failed$ 
18: end procedure

```

On the other hand, after receiving the polynomial evaluations of the source biosensor (PE_s) in a particular round, the conforming biosensor calculates the combinations of sets, $\binom{u}{r} - \binom{u-1}{r}$, that will be used in the current round. Then, the conforming biosensor creates the characteristic polynomials (CP_c) of these sets and calculates its own polynomial evaluations (PE_c) using the public evaluation points (E). After that, the conforming biosensor divides PE_s to PE_c to obtain their divided evaluations (DE) which is equal to $\frac{CP_s}{CP_c}$. Then, these values will be interpolated to calculate the coefficients of the reduced rational function, $\frac{X_{\Delta_A}(Z)}{X_{\Delta_B}(Z)}$. Finally, solving the equation of $\frac{X_{\Delta_A}(Z)}{X_{\Delta_B}(Z)}$ reveals the roots that include the different set elements of the source and conforming biosensors. Using the set reconciliation paradigm explained in Section 2.4, the conforming biosensor is able to

regenerate the exact same sets of IPI values of the source biosensor (PP_s) if and only if CP_s and CP_c differ in d elements with each other. More detailed explanations about the selection criteria of parameter d can be seen in Section 4.2 and the conforming biosensor operations can be seen in Algorithm 3.

In the first try of a round, the conforming biosensor applies set reconciliation on original PP_c and PP_s . If set reconciliation is successful, PP_c sequence will be saved and the SISR protocol will continue using the other sets. Contrarily, if the set reconciliation protocol is unsuccessful, a margin parameter (m) is used by the conforming biosensor. A new set will be generated from the original PP_c , where *one of the IPI values in the original set* will increase or decrease by margin. Here, $\pm m$ is applied on each element of the currently processed set (PP_c), $PP_{c[i]}$, where $0 \leq i < s$. The necessity of parameter m is further explained in Section 4.1.

For each set, $s * 2$ additional sets from the original set are obtained by this method, which has the advantage of increasing the true match possibilities of the key agreement between the source and conforming biosensors, but with the disadvantage of additional computational complexity that increases the key agreement latency. Considering that the conforming biosensor has a set of IPI sequences as seen in Equation 3.1, the new sets will be as given in Equation 3.2, for the first try.

$$PP_c = \{443, 437, 435, 442\} \quad (3.1)$$

$$PP'_c[0] = \{442, 437, 435, 442\} \quad (3.2)$$

$$PP''_c[0] = \{444, 437, 435, 442\}$$

In such a case, set reconciliation will be applied between the generated sets PP'_c , PP''_c and PP_s . If set reconciliation is not successful again, using both PP'_c and PP''_c , the next element ($PP_{c[i]}$) of the currently processed set (PP_c) will have an additional value of $\pm margin$ and the conforming biosensor will try to reconcile PP_s again with the newly generated sets, which are provided in Equation 3.3.

$$\begin{aligned}
PP'_c[1] &= \{443, 436, 435, 442\} \\
PP''_c[1] &= \{443, 438, 435, 442\}
\end{aligned} \tag{3.3}$$

For each modification, only one element of the current set changes; the others remain as original. This operation will continue until the margin is applied on all the IPI values in the set or the set reconciliation is successful. As a result of the modifications, $s * 2$ more alternative PP_c values are obtained.

In particular, having $PP_s = \{440, 437, 435, 442\}$ and $PP_c = \{443, 437, 435, 442\}$, as the IPI sequences of the source and conforming biosensors, respectively, characteristic polynomials of the source and conforming biosensors over the field F_{997} will be calculated as given in Equation 3.4, respectively, considering the bound of evaluation points as \bar{m} , where \bar{m} satisfies the condition given in Equation 3.5. With such values, if the source and conforming biosensors use $E = \{-1, -2, -3\}$ and when the characteristic polynomials are evaluated at the evaluation points over F_{997} , the polynomial evaluations will be as given in the Equation 3.6.

$$\begin{aligned}
CP_s(x) &= (x - 440)(x - 437)(x - 435)(x - 442) \\
CP_c(x) &= (x - 443)(x - 437)(x - 435)(x - 442)
\end{aligned} \tag{3.4}$$

$$\bar{m} \geq m_s + m_c + 1 \tag{3.5}$$

$$\begin{aligned}
PE_s &= \{410, 337, 725\} \\
PE_c &= \{562, 125, 93\}
\end{aligned} \tag{3.6}$$

At this point, PE_s will be transmitted to the conforming biosensor and the division of the evaluations will be calculated by the conforming biosensor using Equation 3.7, as $DE = \frac{PE_s}{PE_c} = \{870, 234, 115\}$.

$$\frac{PE_s}{PE_c} = \frac{CP_{\Delta_s}(x)}{CP_{\Delta_c}(x)} \tag{3.7}$$

Algorithm 3 Conforming Sensor Operations

```
1: procedure CONFORMINGPROTOCOLRUN( $PP_C, r, n, m, E$ )
2:    $foundSetCount = 0$ 
3:    $resultIPI, solution = []$ 
4:    $sourceTerminate = false$ 
5:    $keyFound = false$ 
6:   while  $foundSetCount < r$  and  $keyFound = false$  do
7:      $PE_S, indices, sourceTerminate = receive()$ 
8:     if  $sourceTerminate = true$  then
9:       break
10:    end if
11:    for each  $i \in indices$  do
12:      for each  $ipi \in PP_C[i]$  do
13:         $currentSet = PP_C[i]$ 
14:        for each  $action \in [noChange, increase, decrease]$  do
15:           $newIPI = action(ipi, m)$ 
16:           $currentSet.replace(ipi, newIPI)$ 
17:           $CP_C = createCP(currentSet)$ 
18:           $PE_C = findPE(CP_C, E)$ 
19:           $DE = findDE(PE_C, PE_S)$ 
20:           $roots = solveEquation(DE, E)$ 
21:          if  $roots! = []$  then
22:             $foundSetCount++ = 1$ 
23:             $resultIPI.append(currentSet)$ 
24:             $solution.append(roots)$ 
25:            break
26:          end if
27:        end for
28:        if  $roots! = []$  then
29:          break
30:        end if
31:      end for
32:      if  $foundSetCount \geq r$  then
33:         $keyFound = True$ 
34:         $send(ACK)$  ▷ Key is found
35:        break
36:      else
37:         $send(NAK)$  ▷ Key is not found
38:      end if
39:    end for
40:  end while
41: end procedure
```

As explained in the original set reconciliation algorithm [25], the coefficients of the reduced rational function $\frac{CP_{\Delta_s}(x)}{CP_{\Delta_c}(x)}$ can be recovered by interpolating the DE values. Finally, by factoring $CP_{\Delta_s}(x)$ and $CP_{\Delta_c}(x)$, the different elements of the sets are recovered as the roots: 440 and 443. More information about set reconciliation can be found in Section 2.4

3.3 Secure Key Agreement (SKA) Protocol

At the end of the SISR protocol, which is explained in Section 3.2, the conforming biosensor finds the reconciled IPI sequence. The purpose of the Secure Key Agreement (SKA) protocol is to agree on the same cryptographic key that will be used in the communication between the source and the conforming biosensors by using this reconciled IPI sequence. In order to generate the cryptographic key, we employ novel *quantization* and *binarization* methods on the reconciled IPI sequence. Figure 3.5 illustrates the steps of our proposed SKA protocol.

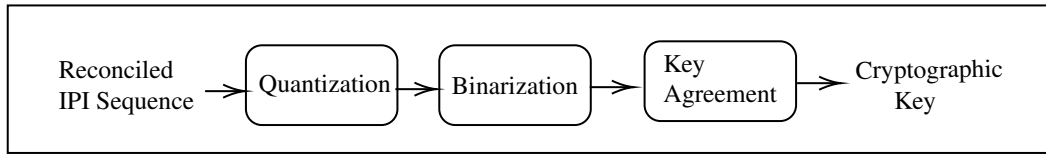


Figure 3.5: Our Proposed SKA Protocol

In our quantization method, each IPI sequence of a subject is quantized using its own IPI range. To do so, for each subject i , minimum IPI_{min}^i and maximum IPI_{max}^i values are found. For each subject, IPI_{min}^i is mapped to zero, IPI_{max}^i is mapped to $(IPI_{max}^i - IPI_{min}^i)$ and other IPI^i values are mapped to linearly within the range $[0, (IPI_{max}^i - IPI_{min}^i)]$ using Equation 3.8

$$IPI_j^i = IPI_j^i - IPI_{min}^i \quad (3.8)$$

The purpose of quantizing the IPI values is to reduce the number of bits required for each IPI while representing them in binary. For instance, in the IPI sequence $\{440, 437, 435, 440, 440, 435, 454, 446\}$, with $min = 435$ and $max = 454$, min IPI value will be mapped to 0, max IPI value will be mapped to $454 - 435 = 19$, and the other values will be

mapped to $(IPI^i - IPI_{min})$. Thus, the quantized IPI sequence after this step will be $\{5, 2, 0, 5, 5, 0, 19, 11\}$.

The next step of SKA after quantization is *binarization*. Binarization step is of great importance due to the fact that the randomness of the generated cryptographic key depends heavily on the binary representation of the IPI sequences. Considering the characteristics of cryptographic keys, we select the number of bits to represent each IPI dynamically. Each IPI sequence includes repetitive IPI values that would potentially reduce the randomness of the generated keys. Therefore, an effective binarization method is needed to circumvent repetitive bits in the generated cryptographic key.

The first step of our binarization method is finding the minimum number of bits to represent each IPI in the IPI sequence, which is called *minBits*. Firstly, *minBits* of each IPI value is calculated. Secondly, bit length, using which the maximum number of IPI values can be calculated in the first step, is chosen as the binarization bit length (b). In case of an equality, the smallest bit length is chosen as b . If IPI_{max}^i is small enough to be represented using b bits, all of the IPI values in the respective sequence can be represented using the same bit length value. Otherwise, if some of the IPI values require more bits to be represented in binary, then those will be represented using the minimum number of bits needed to represent them.

For example, consider the following quantized IPI sequence $\{5, 2, 0, 5, 5, 0, 19, 11\}$. The IPI value 0 can be represented using a minimum of one bit, the IPI value 2 can be represented using a minimum of two bits, the IPI value 5 can be represented using a minimum of three bits, etc. Therefore, a set of *minBits* values are calculated as $\{1 : 2, 2 : 1, 3 : 3, 4 : 1, 5 : 1\}$ where the first value of each element is the bit length and the second value is the number of IPI values that can be represented using this bit length. Here, the counting shows that two IPI values can be represented using 1 bit, one IPI value can be represented using 2 bits, three IPI values can be represented using 3 bits, one IPI value can be represented using 4 bits and one other IPI value can be represented using 5 bits. In our example, 3 will be selected as the binarization bit length (b). However, given the quantized IPI sequence $\{5, 2, 0, 5, 5, 0, 19, 11\}$, if b is 3, then 11 and 19 cannot be represented using 3 bits

since the maximum number that can be represented using 3 bits is 7. Therefore, 11 and 19 will be represented using the minimum number of bits to represent them, which are 4 bits and 5 bits, respectively. Table 3.2 demonstrates the relation between the bit length and the maximum/minimum IPI values that can be represented with that bit length.

Table 3.2: Relation of the bit length and the minimum/maximum IPI values

min	max	minBits
64	127	7
32	63	6
16	31	5
8	15	4
4	7	3
2	3	2
0	1	1

Gray encoding [23] is a binary representation method that ensures one bit difference between consecutive values. Table 3.3 shows the 4 bit binary representations of the decimal values between 0 – 15 with their corresponding gray code values as an example.

Table 3.3: Gray Codes

decimal	binary	gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

After the necessary bit lengths are calculated for each quantized IPI value, Gray encoding [23] is applied for binarization, as given in Algorithm 4. In order to increase the randomness of the generated cryptographic keys, each quantized IPI value is represented using a different bit sequence. The purpose of representing the same IPI value using a different encoding is to increase the randomness by differentiating the repetitive values.

The differentiation of IPI values are produced by concatenating additional bit sequences to the base Gray code value (bgc) which is the regular Gray code for the corresponding value. Presuming the quantized sequence has the repetitive value of Ω in different indices n times, $\Omega_1, \Omega_2 \dots \Omega_n$, the first time that Ω has been observed in the sequence, i.e. Ω_1 , the binary value of Ω_1 will be calculated as bgc_{Ω} . However, the second time that Ω has been observed, i.e. Ω_2 , the value of Ω_2 will not be represented by only using the base gray code value (bgc_{Ω}) of Ω , but also an additional Gray code value will be concatenated to bgc_{Ω} . The additional Gray codes will start from 1 bit Gray codes $\{0, 1\}$, and continue with 2 bit Gray codes $\{00, 01, 11, 10\}$, and so on, until the repetitions of Ω remain. In the second repetition of Ω , i.e. Ω_2 , the additional value that will be appended is the value from the first index of 1 bit Gray codes, $\Omega_2 = bgc_{\Omega}||0$, while the third repetition of Ω , i.e. Ω_3 , will be represented by $bgc_{\Omega}||1$. Similarly, for the fourth and fifth repetitions, $bgc_{\Omega}||00$ and $bgc_{\Omega}||01$ will be used, respectively. For example, the final binarization result of the quantized IPI sequence $\{5, 2, 0, 5, 5, 0, 19, 11\}$ will be the concatenation of the each IPI's Gray code values, such as $\{111, 011, 000, 1110, 1111, 0000, 11010, 1110\}$.

Algorithm 4 Binarization Method

```
1: procedure BINARIZE(reconciledIPISequence)
2:   binaryResult = ""
3:   bitLengths = []
4:   for each ipi ∈ reconciledIPISequence do
5:     if ipi = 0 then
6:       bitLengths[0]+ = 1
7:     else
8:       bitLength = lower ( $\log_2(ipi) + 1$ )
9:       bitLengths[bitLength]+ = 1
10:    end if
11:  end for
12:  b = bitLengths.index(max(bitLengths))
13:  maxIPIToWrite =  $2^b - 1$ 
14:  grayCodes = createGrayCodes(b)
15:  additionalGrayCodes = createAdditionalGrayCodes()
16:  markedIndex = [0] * len(reconciledIPISequence)
17:  for each ipi ∈ reconciledIPISequence do
18:    if ipi > maxIPIToWrite then
19:      dynamicBitLength = lower ( $\log_2(ipi) + 1$ )
20:      dynamicGrayCodes = createDynamicGrayCodes(dynamicBitLength)
21:      binaryResult+ = dynamicGrayCodes[ipi]
22:    else
23:      binaryResult+ = grayCodes[ipi]
24:    end if
25:    if markedIndex[ipi] ≠ 0 then
26:      binaryResult+ = additionalGrayCodes[markedIndex[ipi] - 1]
27:    end if
28:    markedIndex[ipi]+ = 1
29:  end for
30:  return binaryResult
31: end procedure
```

The key agreement step, which is illustrated in Figure 3.6, of our proposed SKA protocol starts after *binarization*. In order to generate the cryptographic key (K), quantization and binarization methods are applied on the reconciled IPI sequence. After the key is generated on the conforming biosensor, the HMAC of the “KeyGenerated” message with the conforming and source biosensors’ IDs is sent to the source biosensor using the generated key. Then, the source biosensor generates all possible key combinations applying the quantization and binarization methods to its IPI sequences that belong to

that particular round and creates all combinations of the HMAC of the “KeyGenerated” message with the conforming and source biosensors’ IDs using the possible key combinations. After that, source biosensor checks which HMAC from the HMAC combinations is equal to the one received from the conforming biosensor. When the matching HMAC is found, source biosensor sends the HMAC of the “KeyConfirmed” message with the source and conforming biosensors’ IDs using the matching key. Finally, source biosensor and conforming biosensor will start using the generated cryptographic key to secure their communication.

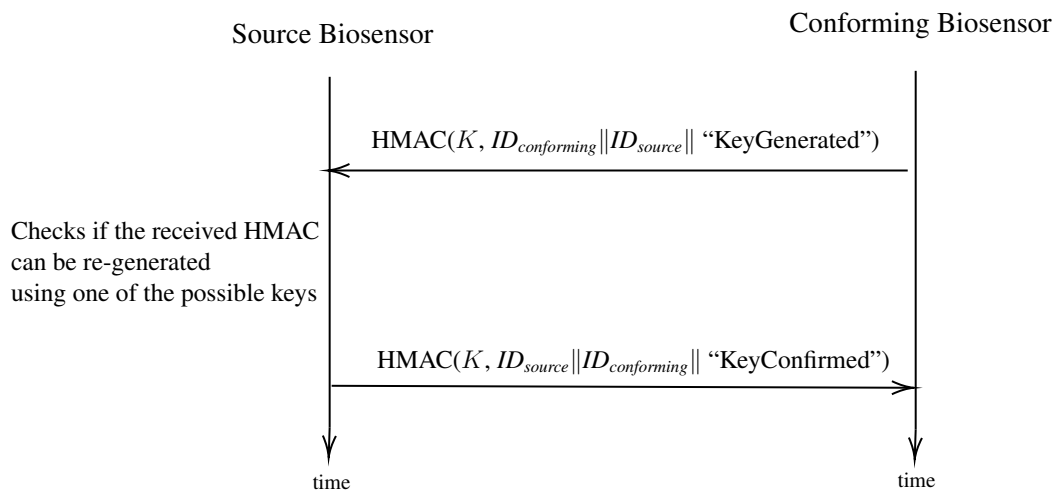


Figure 3.6: Key Agreement

Chapter 4

Performance Evaluation

In this chapter, we discuss the performance measurements related to the security, randomness, distinctiveness and temporal variance of the cryptographic keys generated using our proposed SKA-PSAR (Secure Key Agreement using Physiological Signals with Augmented Randomness) system described in Section 3, together with its key agreement rates, computational complexity, communication complexity and memory requirements. First of all, the test environment and the dataset utilized in our experiments are discussed in Section 4.1. Secondly, the details of the parameters of SKA-PSAR system are explained in Section 4.2. Then, the key agreement rates are given in Section 4.3. Security analyses including temporal variance, distinctiveness and detailed randomness analysis of the generated keys are given in Section 4.4. Finally, computational complexity, communication complexity and memory requirement of SKA-PSAR system are discussed in Section 4.5.

4.1 Test Environment and Dataset

We implemented our SKA-PSAR system using Python programming language version 2.7.13 on MacBook Pro (2.9 GHz Intel Core i5, 8 GB 1867 MHz DDR3, Intel Iris Graphics 6100 1536 MB and MacOS (High Sierra)) and also on Raspberry Pi 3 Model B (1.2 GHz 64-bit quad-core ARMv8 CPU, 802.11n Wireless LAN, Bluetooth 4.1, Bluetooth Low Energy, 1GB RAM), Raspbian OS (Raspbian GNU/Linux 8).

For the experiments, we used BP (Blood Pressure) and ECG (Electrocardiogram) signals obtained from the publicly available PhysioBank MIMIC II Waveform database [20] as the inputs for the source and conforming biosensors, respectively. The performance results do not depend on the choice of the physiological signals for each biosensor. The utilized signals are recorded simultaneously for 5 minutes (sampled at 125 Hz) in order to obtain data from a total of 30 subjects. Then, each signal is divided into two parts and each part is used independently from each other to generate two different IPI (inter-pulse interval) sequences of 80 values each, as explained in Section 3.1. After that, the generated IPI sequences are used as inputs for the SISR protocol of our proposed SKA-PSAR system.

Table 4.1 presents statistics of the data retrieved from the PhysioBank MIMIC II Waveform database [20]. Each row of this table includes the first part of the BP signal of a subject. The general statistics of IPI distributions are given in Table 4.2. These statistics show that the average distinct IPIs belonging to a particular subject from our test data (30 subjects, 2 IPI sequences from each subject, 80 IPI values in each sequence) is approximately 15, and the average range is approximately 30. The minimum IPI of all subjects is 249 and the maximum is 697. Even though the difference between the maximum and the minimum IPI values of all subjects is high, the average IPI range of a single subject, which is calculated as 30, is low. Considering the low range of the IPI values in a particular subject, quantization step in the SKA protocol of our SKA-PSAR system is regulated to obtain acceptable randomness of the resulting cryptographic keys, as explained in Section 3.1. In order to use our proposed SKA-PSAR system, the range of the IPI sequence of an individual should be at least 10, since otherwise the system will be vulnerable to brute-force attacks. The subjects are selected according to this rule.

Table 4.1: Statistics of IPI Distributions

Subject	Range	Min	Max	Number of Distinct IPI
1	38	432	470	11
2	18	420	438	13
3	26	352	378	16
4	10	249	259	11
5	11	355	366	11
6	10	540	550	11
7	33	472	505	10
8	95	356	451	18
9	16	308	324	14
10	37	349	386	21
11	24	392	416	18
12	29	484	513	17
13	20	444	464	15
14	33	482	515	13
15	82	400	482	17
16	12	404	416	12
17	37	470	507	18
18	28	251	279	16
19	25	506	531	19
20	97	600	697	25
21	34	419	453	23
22	33	551	584	20
23	14	460	474	12
24	36	415	451	14
25	21	469	490	19
26	27	361	388	15
27	31	415	446	20
28	70	432	502	15
29	25	548	573	20
30	17	321	338	14

Table 4.2: General Statistics of IPI Distributions

Average Range	Min	Max	Average Number of Distinct IPI
30	249	697	15

We analyzed the IPI distributions in order to detect any repetitive IPI values calculated in the *IPI calculation* step of the IPI Sequence Generation Technique of our proposed SKA-PSAR system. We also analyzed the range between the minimum IPI value and the maximum IPI value obtained from a particular subject. From our observations, we detected that the generated IPI sequences do have repetitions of the same values. Also, we argue that these repetitions reduce the randomness of the resulting keys. Figure 4.1, Figure 4.2 and Figure 4.3 demonstrate the distributions of the calculated IPI values from the BP signals of Subjects 1, 3 and 8, respectively. For instance, one can see from Figure 4.1 that IPI sequence of Subject 1 includes the IPI value of “445” 15 times, where the other IPI values appear only a few times. A similar issue that results in producing non-random cryptographic keys is also seen in Figure 4.2 and Figure 4.3. In order to address the problem of the weak randomness of the generated keys, we propose a novel *binarization* method in Section 3.3.

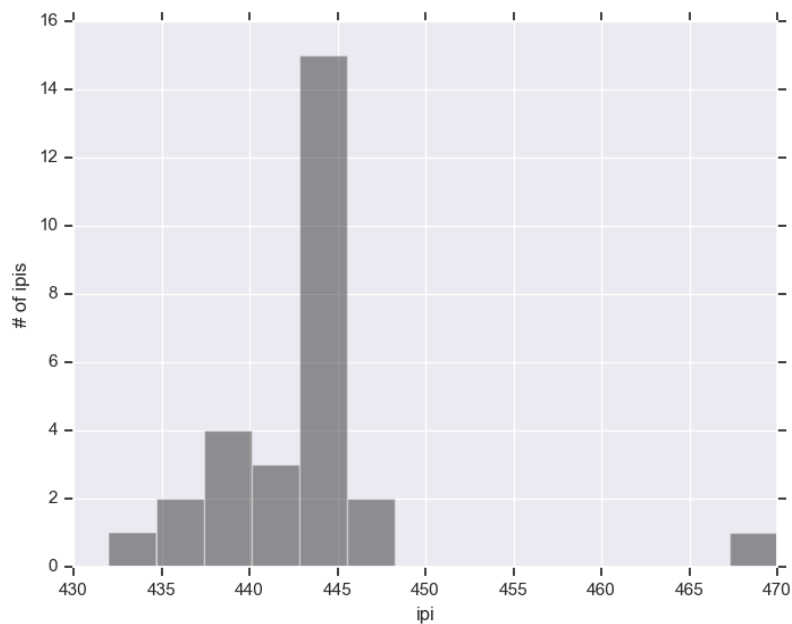


Figure 4.1: IPI Distribution of BP signal of Subject 1

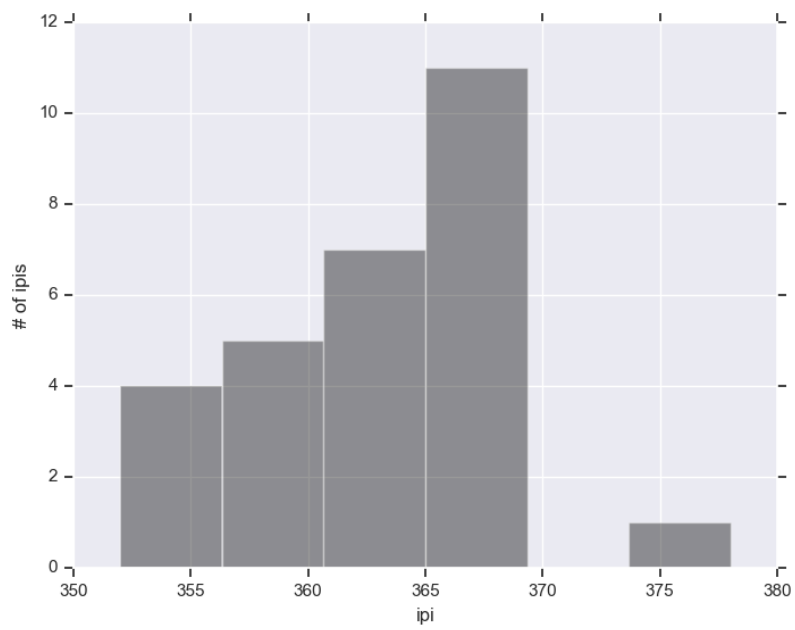


Figure 4.2: IPI Distribution of BP signal of Subject 3

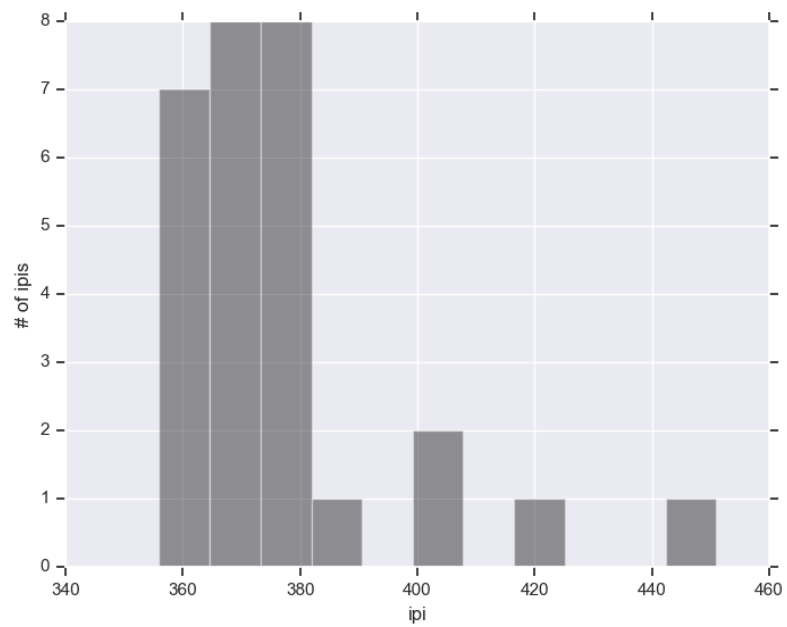


Figure 4.3: IPI Distribution of BP signal of Subject 8

4.2 System Parameters

Security of a cryptographic key depends on the number of effective bits used in the generation of the key. Therefore, the longer the key size, the stronger the key obtained in terms of security. Our proposed SKA-PSAR system produces keys, whose size is determined by the set size (s), the number of utilized sets (r) and the bit length used for representing each IPI of the reconciled IPI sequence produced at the end of SISR protocol, as explained in the binarization step of the SKA protocol of our SKA-PSAR system in Section 3.1. The key sizes that are obtained from SKA-PSAR system is given in Table 4.3. The final length of the key is nb and the strength of this key would be 2^{nb} in case of a regular brute force attack. However, the search space of the attacker is decreased as the result of sorting each set of IPI values in the SISR protocol of our SKA-PSAR system explained in Section 3.2.

Table 4.3: Key sizes obtained from SKA-PSAR system

s	d	m	r	Minimum	Maximum	Average
4	1	1	7	108	186	140
			8	126	213	162

For calculating the effective key length of the generated cryptographic keys, first, the search space (ss), defined as the number of possible sets that contain sorted, repetitive s numbers, where $0 \leq s \leq v$ and v denotes the range of the IPI values (i.e., $IPI_{max}^i - IPI_{min}^i$) of an individual, is calculated. The formula for calculating the search space is given in Equation 4.1, where r is the required set size parameter of the SKA-PSAR system. Equation 4.2 gives the effective key length of the generated cryptographic keys, where the average range of IPI values that are used to generate this key is v .

$$ss = \binom{v - 1 + s}{s} \quad (4.1)$$

$$Effective\ key\ length = \log_2(ss^r) \quad (4.2)$$

The value of s is important to prevent information leakage of the IPI values. As described in Section 3.2, PP sequences are represented using characteristic polynomials of degree s and evaluated on points E . A polynomial of degree s can be constructed using $s + 1$ linear equations. Thus, the number of exchanged polynomial evaluations should be at most s in order to hide the IPI values from the attacker. However, using s evaluations, a polynomial of degree $s - 1$ can be constructed. Therefore, the degree of the interpolated function to be solved on the conforming biosensor $DE = \frac{CP_s}{CP_c}$ should be at most $s - 1$.

In order to solve this rational function, $degree(\frac{CP_s}{CP_c}) + 1$ simultaneous linear equations are needed. Therefore, $degree(\frac{CP_s}{CP_c}) + 1$ should be equal to $s - 1$; thus $s = degree(\frac{CP_s}{CP_c}) + 2$. When a set of IPI values of the biosensors differ one element in s elements, i.e. $d = 1$, $degree(\frac{CP_s}{CP_c})$ will be 2, s will be 4; and thus the number of simultaneous linear equations are needed to solve DE will be 3. In short, 3 linear equations should be sent from the source biosensor to the conforming biosensor and hence the number of elements of E should be 3, when the element size s of a set is 4 and d is 1. In the case when $d = 2$, s should be at least 6, which means that 5 simultaneous equations are required. Similarly, when $d = 3$, s should be 8 and 7 equations are needed. As a result, the most feasible parameters are $s = 4$ and $d = 1$ for our proposed SISR protocol.

SISR protocol runs using the non-quantized, raw IPI values, a.k.a. IPI sequence, obtained as a result of our IPI Sequence Generation Technique explained in Section 3.1. Therefore, in order to obtain high correct key generation rates, the margin parameter (m), which is clarified in the Section 3.2, is used in our SISR protocol. When s is 4 and d is 1, m is selected as 1. The reason behind choosing m as 1 is not to increase the computational complexity of the protocol by creating lots of candidate IPI sequences. This method enables to increase the maximum number of tolerated different set elements in the set reconciliation to $d + m$.

In the light of the information discussed above, we carefully chose s , d , m and r parameters of our proposed SKA-PSAR system as indicated in Table 4.4, considering the security requirements of the length of the generated keys.

Table 4.4: Parameters used our SKA-PSAR system

s	d	m	r	Effective Key Length (bits)
4	1	1	7	≈ 107
			8	≈ 122

4.3 Correct Key Generation Rate (CKGR) and False Key Generation Rate (FKGR)

In our evaluations, Correct Key Generation Rate (CKGR) represents the successful key agreement ratio between the source and the conforming biosensors of the same person. On the other hand, False Key Generation Rate (FKGR) corresponds to the successful key agreement between the source and the conforming biosensors of different people, which should not occur due to security reasons.

Table 4.5 shows the relation between the system parameters, i.e. set size (s), number of different elements tolerable in set reconciliation (d), margin (m), required number of sets to generate the cryptographic key (r) and total number of sets that can be utilized in the protocol (n), and the key generation rates.

Table 4.5: Correct Key Generation Rates and False Key Generation Rates of SKA-PSAR system

Parameters					Correct Key Generation Rate	False Key Generation Rate
s	d	m	r	n		
4	1	1	7	14	91.6	0
				15	93.3	0
				16	98.3	0
				17	98.3	0
				18	98.3	0
				19	100	0
			8	15	90	0
				16	91.6	0
				17	95	0
				18	98.3	0

When n increases, CKGR also increases, as expected, since more sets are processed during the protocol so that the possibilities of finding the matching r sets increases. On the other hand, FKGR is 0% in all cases, which shows that different subjects' biosensors do not establish correct cryptographic keys among themselves. Moreover, less sets needed to be reconciled when $r = 7$, compared to the case when $r = 8$, therefore CKGR is better when $r = 7$. Results show that 100% CKGR can be achieved among the biosensors when $r = 7$ and $n = 19$, with 0% FKGR.

Also, there is an inverse proportion between the value of the required sets r and the CKGR as seen in Table 4.5: when r increases the total number of subjects who can successfully generate keys decreases. Contrarily, when r increases the effective key lengths of the generated keys increase. Therefore, even though the greater r provides better security in terms of key strength, value of r should also be selected considering the CKGR of the overall system.

4.4 Security Analysis

In this section, we analyze the cryptographic keys generated by our proposed SKA-PSAR system using the security metrics such as randomness, distinctiveness and temporal variance. In Section 4.4.1, the threat model is explained. In Section 4.4.2, we discuss the randomness results obtained from the NIST Test Suite [5] and in Section 4.4.3 and Section 4.4.4, we provide the distinctiveness and the temporal variance of the resulting cryptographic keys, respectively.

4.4.1 Threat Model

The purpose of the adversary is to obtain the cryptographic key that is agreed between the source biosensor and the conforming biosensor as a result of the SKA-PSAR system. Since the communication between the biosensors is on wireless medium, the network packets including the keying materials might be captured and obtained by the adversary. However, the exchanged information do not reveal the key itself, since it includes the

polynomial evaluations that are sent from the source biosensor and the HMAC of the key with an acknowledgement message that is sent from the conforming biosensor. The attacker might try to break the HMAC and apply brute force attack in order to find the matching cryptographic key. Therefore, our proposed SKS-PSAR system parameters are chosen as to provide an effective key size for the generated cryptographic key as explained in Section 4.2. Moreover, she/he might attempt to solve the polynomial evaluations that are sent from the source biosensor to the conforming biosensor, however the SKA-PSAR system parameters are chosen to circumvent this issue, as explained in Section 4.2.

4.4.2 Randomness of the Generated Cryptographic Keys

Being random is one of the most important characteristics of cryptographic keys. Considering this in mind, we made necessary regulations on the proposed SKA-PSAR system, especially on the *quantization* and *binarization* steps of the SKA protocol which is explained in Section 3.3. Representing each IPI using a different bit sequence is one of the regulations made in order to obtain more random keys.

For the analysis of randomness, a Python implementation [11] of NIST Test Suite [5] is utilized, which supplies a command line tool to evaluate the randomness of the generated keys. NIST Test Suite includes 15 randomness tests and each test is given with the input size recommendation in the documentation [5]. We selected seven tests, which are explained below, whose input size is suitable for the cryptographic keys that are generated by our proposed SKA-PSAR system. For the other eight tests, input sizes were not sufficient for the output of our proposed system.

In NIST Test Suite [5] documentation, *P-value* is described as “The probability that a perfect random number generator would have produced a sequence less random than the sequence that was tested, given the kind of non-randomness assessed by the test.” P-value of 1 shows that the binary sequence is a perfect random sequence and P-value of 0 indicates a non-random sequence. In this thesis, P-value is taken as 0.01 as recommended by NIST Test Suite [5] documentation and the binary sequence is accepted as random if the result of its test is greater than or equal to this P-value.

- **Frequency (Monobit) Test:** The purpose of this test is to determine whether the number of ones and zeros in the sequence are almost the same. This test is also used as a prerequisite test to the other tests.
- **Frequency Test within a Block:** The aim of this test is to examine the proportions of ones in M bit blocks. The proportion should be approximately $M/2$ and M is selected as 20, as recommended by the documentation that $M \geq 20$ and $M > .01nb$ should be satisfied, where the total key size (nb) is the multiplication of the selected bit length with $r * s$ in our experiments.
- **Runs Test:** A run is described as the length k of identical bits (zeros or ones) bounded by the opposite bit. This test checks if the number of runs of zeros and ones are expected for a random sequence.
- **Test for the Longest Run of Ones in a Block:** This test analyses whether the longest run of ones within M blocks are expected for a random sequence. The disorder of the longest run of ones implies the disorder of the longest run of zeros. In this test, M is selected as 8, as recommended by the documentation.
- **Non-Overlapping Template Matching Test:** The purpose of this test is to count the occurrences of non-periodic patterns in the given binary sequence.
- **Approximate Entropy Test:** This test aims to compare the frequency of every overlapping m -bit patterns against the expected result for a random sequence. Here, m is selected as 2, as recommended by the documentation that $m < \log^2(n) - 5$ should be satisfied.
- **Cumulative Sums (Cusum) Test:** The purpose of this test is to check if the cumulative sums of partial sequences are as expected for a random sequence.

The results of the corresponding tests of NIST Test Suite are given in Table [4.6](#), which demonstrate that a very high percentage of the generated cryptographic keys possess adequate randomness by successfully passing the tests. The test parameters are selected as recommended by the documentation of the NIST Test Suite.

In monobit frequency, runs and longest run tests, the successfully passing percentage of the generated keys is slightly better when $r = 7$, compared to $r = 8$. On the other hand, the successfully passing percentage of the generated keys is larger when $r = 8$, compared to $r = 7$, in block frequency, non-overlapping template matching and cumulative sums tests.

Table 4.6: NIST Test Suite Results of SKA-PSAR System

NIST Test	$r = 7$			$r = 8$		
	<i>Failed</i>	<i>Passed</i>	<i>Percentage</i>	<i>Failed</i>	<i>Passed</i>	<i>Percentage</i>
Frequency (Monobit)	3	57	95.0	3	56	94.9
Frequency (Block)	4	56	93.3	3	56	94.9
Runs	1	59	98.3	3	56	94.9
Longest Run	1	59	98.3	2	57	96.6
N.O. Template Matching	1	59	98.3	0	59	100.0
Approximate Entropy	0	60	100.0	0	59	100.0
Cusum	4	56	93.3	3	56	94.9

4.4.3 Distinctiveness of the Generated Cryptographic Keys

One of the other important characteristics of cryptographic keys is that they should be different for different users at any particular time. Ideally half of the bits in the same locations of the two bit streams should be different to accept them as distinct. Distinctiveness is calculated using the Hamming Distance metric, by measuring the number of different bits in the same indices of two cryptographic keys belonging to different people. The SKA protocol of our proposed SKA-PSAR system produces cryptographic keys in such a way that each has a different key size, as explained in Section 3.3. Since we need two bit sequences of the same length for Hamming Distance calculation, we used the first t bits of each key pair, where t is the minimum bit length of the two keys in question. Our experiments using Hamming Distance metric show that the cryptographic keys generated by our proposed SKA-PSAR system, which is described in Chapter 3, possess high distinctiveness as the average Hamming Distance calculated between the cryptographic keys are 49.70% and 49.65%, when $r = 7$ and $r = 8$, respectively.

4.4.4 Temporal Variance of the Generated Cryptographic Keys

Another essential characteristic of cryptographic keys is the temporal variance, which is defined as the distinctiveness of the generated cryptographic keys of the same person at the different time intervals. Temporal variance is important in order to be able to renew the cryptographic keys in case of any security problem. In our experiments, we generated two cryptographic keys from the same person in different time intervals, as explained in Section 4.1. Then, utilizing the Hamming Distance metric, the average temporal variance of the generated keys is obtained. The cryptographic keys generated from the SKA protocol of our SKA-PSAR system which is explained in Section 3.3 have diverse key sizes. For this reason, as in Section 4.4.3, the Hamming Distance calculation is performed using pairwise minimum bit lengths of the keys. The measured results are 49.46% and 48.85%, when $r = 7$ and $r = 8$, respectively. These results show that the keys generated by our SKA-PSAR system, which is explained in Section 3, possess high temporal variance.

4.5 Computational and Communication Complexity and Memory Requirements of SKA-PSAR

In this section, we analyze the computational complexity, communication complexity and memory requirements of our proposed SKA-PSAR system. These complexities are directly proportional to the maximum number of candidate IPI sequences generated by the source biosensor and sent to the conforming biosensor for the set reconciliation processes in the SISR protocol phase. The maximum number of candidate IPI sequences generated are discussed in Section 4.5.1. We provide the computational and communication complexities of our proposed SKA-PSAR system in Section 4.5.2 and Section 4.5.3, respectively and we give its memory requirements in Section 4.5.4.

4.5.1 Maximum Number of Candidate IPI Sequences Generated in SISR Protocol

The maximum number of candidate IPI sequences generated with respect to the system parameters, i.e. number of elements in each set (s), maximum number of different set elements tolerable in set reconciliation (d), the margin (m), required number of sets (r) and the total number of utilized sets (n), are given in Table 4.7. In each round of our SISR protocol, $\binom{u}{r} - \binom{u-1}{r}$ candidate IPI sequences are generated by the source biosensor, where u starts from r and is incremented by one in each round until $u = n$. After the polynomial evaluations of the candidate IPI sequences are sent to the conforming biosensor, the conforming biosensor tries to reconcile the same IPI sequences of the source biosensor by trying each IPI sequence. As a result of this, the conforming biosensor needs to do at most $\binom{u}{r} - \binom{u-1}{r}$ comparisons.

Expectedly, the number of candidate IPI sequences increases on both biosensors, as n or r increases. As will be analyzed in the rest of this section, the number of candidate IPI sequences is directly proportional to computational complexity, communication complexity and memory requirements of the overall SKA-PSAR system.

Table 4.7: Maximum number of candidate IPI sequences generated in SISR protocol

Parameters					<i>Number of candidate IPI sequences generated</i>
s	d	m	r	n	
4	1	1	7	14	3432
				15	6435
				16	11440
				17	19448
				18	31824
				19	50388
			8	15	6435
				16	12870
				17	24310
				18	43758

4.5.2 Computational Complexity

Computational complexity of our proposed SKA-PSAR system is measured by the time difference between the start time of the IPI Sequence Generation Technique and the end time of SKA protocol. Here, the start time is defined as the time that the connection has been established between two communicating biosensors, while the end time is defined as the time that both biosensors have generated the symmetric cryptographic keys or the time that the total number of utilized sets (n) are reached. The former scenario represents a successful key agreement whereas the latter scenario shows a failure case of the SISR protocol.

Table 4.8 and Table 4.9 present the average latency of SKA-PSAR system obtained from Macbook Pro and Raspberry Pi3, respectively. Both tables show the relation between the system parameters (s, d, m, r, n) and the average round time together with the average total time, both in seconds. However, Table 4.9 has less data due to the fact that the SKA-PSAR system execution takes too much time on Raspberry Pi3 using the system parameters that are included in Table 4.8 but not included in Table 4.9. The SISR protocol of our proposed SKA-PSAR system runs in a round manner, as explained in Section 3.2. Here, the round time is defined as the time elapsed between the start time of one round and the end time of that round. For generating a key between two biosensors, the proposed SISR protocol runs at least one round and at most $n - r + 1$ rounds, while the SKA protocol runs only once at the end.

Expectedly, when maximum utilized set count n increases, both the round time and the total time increase. The reason behind this is that the number of generated candidate IPI sequences, which is discussed in Section 4.5.1, increases with the increment of n . In addition to these, CKGR, which is explained in Section 4.3, is directly proportional with the value of n . Therefore, computational complexity increases in order to obtain better CKGR between the biosensors. Furthermore, when r increases, the round time and the total time also increase. When $r = 7$ and $n = 18$, CKGR is 98.3% and the average round time is 23.49 s; when $r = 7$ and $n = 19$, CKGR is 100% and the average round time is 133.7 s, while when $r = 8$ and $n = 18$, CKGR is 98.3% and the average round

Table 4.8: Average latency on Macbook Pro

Parameters					<i>Round Time (s)</i>	<i>Total Time (s)</i>
<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
4	1	1	7	14	6.85	20.55
				15	8.76	27.22
				16	23.49	81.22
				17	23.49	81.22
				18	23.49	81.22
			8	19	36.96	133.70
				15	12.87	41.25
				16	17.71	58.63
				17	41.86	148.37
						18

Table 4.9: Average latency on Raspberry Pi3

Parameters					<i>Round Time (s)</i>	<i>Total Time (s)</i>
<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
4	1	1	7	14	59.61	178.85
				15	74.13	216.29
			8	15	102.9	324.01
				16	118.579	358.5

time is 66.81 s on Macbook Pro. The latency of the SKA-PSAR system is higher on the Raspberry Pi3 comparing to the latency of the system on the Macbook Pro, since their computational capabilities and the memory capacities differ substantially as explained in Section [4.1](#).

4.5.3 Communication Complexity

Communication complexity of our proposed SKA-PSAR system is calculated as the amount of data (in kilobytes) exchanged over the network between the communicating biosensors during the SISR protocol and the SKA protocol phases. Table [4.10](#) presents the data sent both from the source biosensor to the conforming biosensor, and from the conforming biosensor to the source biosensor.

The data sent from the source biosensor is greater than the data sent from the conforming biosensor. The reason behind this is that the generation of the candidate IPI sequences

is performed on the source biosensor and those are sent to the conforming biosensor over the wireless network. On the other hand, conforming biosensor sends only a simple response message.

Communication complexity increases when the number of utilized sets (n) increases. Since with the increase of n , more candidate IPI sequences are generated and sent over the network, increase in the communication complexity is expected. Moreover, increment of the required set size (r) from 7 to 8 triggers generation of more IPI sets in order to obtain longer cryptographic keys, as discussed in Section 4.2. Therefore, the communication complexity increases when r increases.

Table 4.10: Average communication complexity (KB)

Parameters					Amount of Data (KB)
s	d	m	r	n	
4	1	1	7	14	49.95
				15	66.72
				16	153.77
				17	153.77
				18	153.77
				19	282.77
			8	15	118.59
				16	158.02
				17	305.26
				18	555.01

4.5.4 Memory Requirements

Memory requirements of our proposed SKA-PSAR system is determined by the amount of memory (in megabytes) allocated for the processes running on the host machines during the IPI Sequence Generation Technique with SISR and SKA protocols. The memory information is retrieved from Macbook Pro using psutil (Python system and process utilities) library [33]. USS (Unique Set Size), which is defined in the psutil documentation as “USS is the most representative metric for determining how much memory is actually being used by a process. It represents the amount of memory that would be freed if the process was terminated right now” [33] is used to determine the memory requirement.

Table 4.11 presents the memory requirements of the source biosensor and the conforming biosensor. With the increase of the utilized set size (n), the memory requirement of both biosensors slightly increase. However, the conforming biosensor uses more memory compared to the source biosensor. The reason behind this is that the conforming biosensor generates the candidate IPI sequences in each round of the SISR protocol as well as receiving the candidate IPI sequences generated by the source biosensor.

Table 4.11: Average memory requirement (MB)

Parameters					<i>Source Biosensor</i>	<i>Conforming Biosensor</i>
<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
4	1	1	7	14	12	38
				15	12	38
				16	12	38
				17	12	38
				18	12	38
			19	13	39	
			8	15	13	38
				16	14	39
				17	15	40
				18	17	41

Chapter 5

Discussion On the Comparison of the SKA-PSAR system and the SKA-PS protocol

In this chapter, we first discuss the general differences between our proposed SKA-PSAR system and the baseline SKA-PS protocol [17], in Section 5.1 by comparing the details of the *quantization* and *binarization* methods, the system parameters and the protocol implementations. Then, we analyze the performance differences of SKA-PSAR system and SKA-PS protocol in Section 5.2, in terms of key generation rates, randomness analysis, distinctiveness and temporal variance results, computational complexity, communication complexity and memory requirements.

5.1 Disparities between SKA-PSAR and SKA-PS

In this section, we compare our proposed SKA-PSAR protocol which consists of three main parts: (i) IPI Sequence Generation Technique, (ii) SISR protocol, (iii) SKA Protocol, with the baseline SKA-PS protocol [17]. IPI Sequence Generation Technique of SKA-PSAR system has equivalent mechanisms up to the quantization step of Physiological Parameter Generation Technique in SKA-PS protocol. SISR protocol of SKA-PSAR sys-

tem corresponds to the set reconciliation part of SKA-PS protocol and SKA Protocol of SKA-PSAR system corresponds to the parts after the set reconciliation paradigm in SKA-PS protocol. There are significant differences between SKA-PSAR system and SKA-PS protocol including but not limited to the *quantization* and *binarization* methods applied to generate the cryptographic keys from a sequence of IPI values, the data on which the set reconciliation is applied and the security mechanism of the acknowledgement messages exchanged during the key agreement protocol.

One of the most important differences between SKA-PSAR and SKA-PS is in the *quantization* and *binarization* algorithms that is used to generate the cryptographic key to be used among the biosensors for secure communication. In the quantization step of SKA-PS, a circular uniform quantization method is used [17, 16] as described in detail in Section 2.6. In this quantization method, different IPI values are represented using the same quantization value. Therefore, the randomness of the generated cryptographic keys decrease. In addition to these, utilizing IPI_{min} and IPI_{max} values in the quantization step is selected from the overall dataset. However, each individual's IPI range ($IPI_{max} - IPI_{min}$) varies, as explained in Section 4.1. Hence, instead of using the range of the dataset, IPI values belonging to a particular person should be quantized within each other. On the other hand, in our proposed quantization method, which is explained in Section 3.1, each IPI in the IPI sequence of an individual i is mapped to $(IPI_k^i - IPI_{min})$, where $0 > k > len(IPI^i) + 1$. In other words, step size is chosen as 1 in the range of a particular subject. The aim of our quantization method is to reduce the bit length of each IPI before the binarization step. Thus, the randomness of the generated cryptographic key from the SKA-PSAR system will be increased.

The other significant difference between SKA-PSAR and SKA-PS is in the *binarization* method. In the binarization step of SKA-PS, $2^{128/\frac{L}{g}}$ bit Gray encoding is applied on each quantized IPI value. In contrary, binarization step in SKA Protocol of SKA-PSAR system utilizes unique binarization for each subject dynamically, as explained in Section 3.3 and as represented in Algorithm 4. Depending on the IPI values in the IPI sequence, binarization bit length (b) changes for each subject and also utilized bit length

of each IPI from the same subject might differ than each other. After the representation bit length is obtained, Gray code is applied on the IPI values. Since the randomness depends on the binary representation of the generated keys, binary representation of the reconciled IPI sequence is of great importance. Another difference of SKA-PSAR system and SKA-PS protocol is in the length of the generated key: Resulting cryptographic key agreed in SKA-PS is fixed size, 128 bit, while the size of the agreed key in SKA-PSAR dynamically changes depending on the distribution of the IPI values and the binarization bit length (b) as explained in detail in Section 4.2.

Moreover, the input of the SKA-PS protocol is the quantized IPI values generated from the quantization step of the physiological parameter generation method of the SKA-PS protocol. Contrarily, the input of our proposed SISR protocol is the IPI sequence including raw IPI values obtained from the physiological signals through IPI Sequence Generation Technique. These IPI values are used in the SISR protocol of the SKA-PSAR system without being quantized. While using raw IPI values from a diverse range as the input of our SISR protocol provides better randomness of the generated keys, it causes the reduction of CKGR compared to the baseline SKA-PS protocol that uses IPI values from a small range.

Nevertheless, by adding a new margin parameter (m), the CKGR of our SKA-PSAR system is improved by increasing the number of different set elements that can be tolerated in set reconciliation. For instance, if the source biosensor has the following set of IPI values $PP_s = \{443, 437, 435, 442\}$ and the conforming biosensor has the following set $PP_c = \{443, 437, 438, 440\}$, where $d = 1$, $m = 1$, the number of tolerable different elements are $d + m = 2$ and $s = 4$, utilizing set reconciliation, the conforming biosensor will be able to retrieve the PP_s using our proposed SISR protocol of the SKA-PSAR system. On the other hand, SKA-PS protocol is designed to handle the IPI sets that differ each other by only a single set element when the set size (s) is 4. Therefore, in SKA-PS protocol the conforming biosensor will not be able to regenerate the IPI sequence of the source biosensor that was provided in the example above.

Another important difference between our proposed SKA-PSAR system and SKA-PS protocol is that the security mechanism of the response messages sent from the conforming biosensor to the source biosensor. In SKA-PS, the response message includes the acknowledgement about whether the key generation is successful or not and also the key index that belongs to the processed key in the current round. However, this method allows the malicious intruder to learn the index of the generated key that might be used in her/his brute force attacks. On the other hand, SKA-PSAR is designed to hide this information from the attackers by sending the HMAC of a success message with the generated key.

5.2 Performance Comparison of SKA-PSAR and SKA-PS

In this section, the performance of our proposed SKA-PSAR system and SKA-PS protocol are compared. For the comparison, we have re-executed SKA-PS protocol with the dataset that we have used for the performance evaluation of our proposed SKA-PSAR system. For the performance analyses, we use the Python programming language (Python 2.7.13) and implemented SKA-PS both on Raspberry Pi 3 Model B (1.2 GHz 64-bit quad-core ARMv8 CPU, 802.11n Wireless LAN, Bluetooth 4.1, Bluetooth Low Energy, 1GB RAM), Raspbian OS (Raspbian GNU/Linux 8) and on MacBook Pro (2.9 GHz Intel Core i5, 8 GB 1867 MHz DDR3, Intel Iris Graphics 6100 1536 MB and MacOS (High Sierra)) then we compare the average latency of the SKA-PS protocol on Raspberry Pi3 and MacBook Pro. In our implementation on Raspberry Pi boards, two Raspberry Pi simulating biosensors, communicate with each other on the wireless medium.

In Section [5.2.1](#), key generation rates of both models are discussed. In Section [5.2.2](#), randomness test results which are based on NIST Test Suite [\[5\]](#), of the generated keys from the SKA-PSAR system and SKA-PS protocol are given. In Section [5.2.4](#), distinctiveness and temporal variance results of the models are compared. In Section [5.2.5](#), computational complexity, communication complexity and memory requirements results are compared.

5.2.1 Key Generation Rates: CKGR and FKGR

CKGR (Correct Key Generation Rate) which is the match rate of reconciled IPI sequences measured by the biosensors placed on the same individual and FKGR (False Key Generation Rate) which is the match rate of reconciled IPI sequences measured by the biosensors placed on different individuals, obtained from the SKA-PS protocol and SKA-PSAR system are given in Table 5.1. The 100% CKGR of SKA-PS is achieved when $n = 14$ and $r = 11$. On the other hand, in SKA-PSAR the 100% CKGR is reached when $n = 19$ and $r = 7$, as explained in Section 4.3. The value of r is chosen considering not only the effective key length of the generated cryptographic keys but also the FKGR value. When r decreases, FKGR increases in both models. Minimum value of FKGR can be obtained using the selected r values in Table 5.1. In SKA-PSAR system smaller required set (r) values can be chosen compared to the SKA-PS protocol. However, in order to generate the cryptographic key, more utilized sets (n) are needed in SKA-PSAR system.

Table 5.1: Key Generation Rates of SKA-PS and SKA-PSAR

	Parameters					CKGR	FKGR
	s	d	m	r	n		
SKA-PS	4	1	-	11	11	83.3	0
					12	91.6	0
					13	96.6	0
					14	100.0	0
SKA-PSAR	4	1	1	7	14	91.6	0
					15	93.3	0
					16	98.3	0
					17	98.3	0
					18	98.3	0
				8	19	100	0
					15	90	0
					16	91.6	0
					17	95	0
					18	98.3	0

5.2.2 Randomness Tests

The randomness test results of the SKA-PS protocol and our proposed SKA-PSAR system that are obtained using NIST Test Suite [5] are given in Table 5.2. The detailed explanations of the tests can be found in Section 4.4.2. Even though, the publication of SKA-PS protocol [17] does not include randomness test results from the NIST Test Suite, we re-executed the SKA-PS protocol using the dataset that we have used to test our SKA-PSAR system in order to obtain SKA-PS randomness results. These results show that the keys generated in the SKA-PS protocol possess low randomness, since 53.07% of the cryptographic keys passed 7 tests from the NIST Test Suite on average. On the other hand, the randomness results of the keys generated in SKA-PSAR system present that the resulting keys possess high randomness as 96.6% of the generated cryptographic keys successfully passed 7 tests of the NIST Test Suite.

Table 5.2: NIST Test Suite Results of SKA-PS and SKA-PSAR

<i>Test</i>	SKA-PS			SKA-PSAR					
	<i>r</i> = 11			<i>r</i> = 7			<i>r</i> = 8		
	<i>Fail</i>	<i>Pass</i>	%	<i>Fail</i>	<i>Pass</i>	%	<i>Fail</i>	<i>Pass</i>	%
Frequency (Monobit)	34	26	43.3	3	57	95.0	3	56	94.6
Frequency (Block)	31	29	48.3	4	56	93.3	3	56	94.9
Runs	23	37	61.6	1	59	98.3	3	56	94.9
Longest Run	60	0	0	1	59	98.3	2	57	96.6
N.O. Template Matching	15	45	75	1	59	98.3	0	59	100.0
Approximate Entropy	0	60	100.0	0	60	100.0	0	59	100.0
Cusum	34	26	43.3	4	56	93.3	3	56	94.9

5.2.3 Maximum Number of Candidate IPI Sequences Generated

Table 5.3 gives the maximum number of candidate IPI sequences generated in each round of the SKA-PS protocol, which depends on the the total number of sets (n). In order to achieve 100% CKGR in the SKA-PS protocol, maximum 364 candidate IPI sequences are generated. To obtain 100% CKGR in the SKA-PSAR system, 50388 candidate IPI sequences are generated, as explained in Section 4.5.1. This significant difference affects the computational complexity of the models, as will be discussed in Section 5.2.5.

Table 5.3: Maximum number of candidate IPI sequences generated

	Parameters					<i>Number of candidate IPI sequences</i>
	<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>	
SKA-PS	4	1	-	11	11	1
					12	12
					13	78
					14	364
SKA-PSAR	4	1	1	7	14	3432
					15	6435
					16	11440
					17	19448
					18	31824
				8	19	50388
					15	6435
					16	12870
					17	24310
					18	43758

5.2.4 Distinctiveness and Temporal Variance

Distinctiveness is calculated using the hamming distance of the keys generated from the physiological signals of different individuals. Approximately half of the bits that are located in the same indices of two binary sequences should be different in order to obtain the ideal distinctiveness. The average distinctiveness of the generated keys using our proposed SKA-PSAR system is measured as 49.70% and 49.65%, when $r = 7$ and $r = 8$, respectively. On the other hand, the distinctiveness result of SKA-PS protocol is 49.95% when $r = 11$.

Temporal variance is measured using the hamming distance of the cryptographic keys that belongs to the same individual and are generated in different time intervals. To obtain optimal temporal variance, half of the bits in the same locations of the binary sequences should be different. The measured temporal variance results of our proposed SKA-PSAR system are 49.46% and 48.85%, when $r = 7$ and $r = 8$, respectively. Contrarily, the temporal variance result of SKA-PS protocol is 26.04% when $r = 11$.

5.2.5 Computational Complexity, Communication Complexity and Memory Requirements

In this section, we compare computational and communication complexity of the SKA-PS protocol together with its memory requirements with that of our proposed SKA-PSAR system. The computational complexities of our proposed SKA-PSAR system and the SKA-PS protocol that is measured on Raspberry Pi 3 boards are given in Table 5.4. The average latency of our proposed SKA-PSAR system and the SKA-PS protocol on Macbook Pro can be seen in Table 5.5. The latency results imply that both models runs on Macbook Pro 10 times faster than on Raspberry Pi3. The results also show that the computational complexity of SKA-PSAR is significantly higher than that of SKA-PS. In order to reach the same CKGR values SKA-PSAR spends approximately 3000 times more time as compared to SKA-PS. The reason behind this difference between SKA-PS and SKA-PSAR is that the number of candidate IPI sequences generated in each round of SKA-PSAR is greater than that of SKA-PS, which is discussed in Section 5.2.3 in light of Table 5.3.

Table 5.4: Average latency on Raspberry Pi3

	Parameters					<i>Round Time (s)</i>	<i>Total Time (s)</i>
	<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
SKA-PS	4	1	-	11	11	0.04	0.04
					12	0.07	0.08
					13	0.11	0.13
					14	0.35	0.45
SKA-PSAR	4	1	1	7	14	59.61	178.85
					15	74.13	216.29
				8	15	102.9	324.01
					16	118.579	358.5

Table 5.5: Average latency on Macbook Pro

	Parameters					<i>Round Time (s)</i>	<i>Total Time (s)</i>
	<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
SKA-PS	4	1	–	11	11	0.005	0.005
					12	0.007	0.008
					13	0.011	0.013
					14	0.040	0.051
SKA-PSAR	4	1	1	7	14	6.85	20.55
					15	8.76	27.22
					16	23.49	81.22
					17	23.49	81.22
					18	23.49	81.22
					19	36.96	133.70
				8	15	12.87	41.25
					16	17.71	58.63
					17	41.86	148.37
					18	66.81	253.65

Communication complexity is measured by the amount of data (in kilobytes) exchanged over the network between the communicating biosensors. The average communication complexity of our proposed SKA-PSAR system and the SKA-PS protocol is given in Table 5.6. The communication complexity of SKA-PSAR is 80 – 125 times greater than that of SKA-PS for same CKGR values as the number of generated candidate keys in SKA-PSAR is greater than that of SKA-PS, as discussed in Section 5.2.1. Therefore, the biosensors of the SKA-PSAR system need to communicate more in order to exchange the keying materials.

Table 5.6: Average communication complexities (KB)

	Parameters					<i>Amount of Data (KB)</i>
	<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>	
SKA-PS	4	1	–	11	11	0.20
					12	0.40
					13	1.19
					14	3.60
SKA-PSAR	4	1	1	7	14	49.95
					15	66.72
					16	153.77
					17	153.77
					18	153.77
				8	19	282.77
					15	118.59
					16	158.02
					17	305.26
					18	555.01

Memory requirement is described as the amount of memory (in megabytes) allocated for the process running on the host machines during the protocols. The results include all of the memory requirement of the data and the additional Python libraries, such as OS, socket, numpy and psutil. The memory requirements of our proposed SKA-PSAR system and the SKA-PS protocol is given in Table 5.7. In the SKA-PS protocol, the difference between the memory requirements of the source and the conforming biosensors are negligible. Even though, on both models the conforming biosensor generates the candidate IPI sequences in each round as well as receiving the cryptographic keys generated by the source biosensor, the conforming biosensor’s memory requirement is 3-to-4-fold greater than the source biosensors in our proposed SKA-PSAR system. The reason behind this is that the number of generated keys and the computational complexity in SKA-PSAR is greater as compared to the SKA-PS protocol.

Table 5.7: Average memory requirements (MB)

	Parameters					<i>Source Biosensor</i>	<i>Conforming Biosensor</i>
	<i>s</i>	<i>d</i>	<i>m</i>	<i>r</i>	<i>n</i>		
SKA-PS	4	1	-	11	11	11	11
					12	11	11
					13	11	11
					14	11	11
SKA-PSAR	4	1	1	7	14	12	38
					15	12	38
					16	12	38
					17	12	38
					18	12	38
				19	13	39	
				8	15	13	38
					16	14	39
					17	15	40
					18	17	41

To sum up, our SKA-PSAR system performs worse in computational, communication complexities and memory usage metrics as compared to SKA-PS for the same CKGR values; however, this is a cost that we pay for achieving high randomness rates. Thus, our SKA-PSAR system provides a trade-off between randomness of the keys versus operational performance without sacrificing from security.

Chapter 6

Conclusions

Body Area Networks (BANs) operate in a wireless environment and are prone to cryptographic attacks. The hardware constraints of the biosensors restrain the usage of conventional cryptographic key generation algorithms. Therefore, a light-weight key generation algorithm is needed to protect the sensitive information exchanged between the communicating biosensors.

In this thesis, we have proposed a novel secure key agreement system that utilizes physiological signals for BANs: Secure Key Agreement using Physiological Signals with Augmented Randomness (SKA-PSAR). Security of the communication between two biosensors placed on the same individual is provided by generating the same cryptographic key among them by utilizing their physiological signals on the SKA-PSAR system. The biggest difference between our proposed SKA-PSAR system and the existing protocols in the literature is that the former is a comprehensive system that provides improved randomness.

For the performance analysis of the SKA-PSAR system, BP and ECG signals of 30 individuals are retrieved from the publicly available PhysioBank MIMIC II Waveform database [20]. We analyzed the performance of our proposed SKA-PSAR system through correct and false key generation rates, randomness, distinctiveness and temporal variance of the generated keys together with the computational and communication complexity, and memory requirements. By using our proposed system, successful key agreement

rate of 100% between the biosensors of the same person (correct key agreement rate) and the key agreement rate of 0% between the biosensors of different people (false key agreement rate) are achieved. Also, 96.6% of the generated cryptographic keys from the SKA-PSAR system successfully passed 7 tests of NIST Test Suite [5] that proves the high randomness of the generated keys. As the cryptographic key characteristics require that the generated keys belong to the same individual should not be the same in different time intervals (*temporal variance* property) and also the generated keys of the different individuals should not be similar to each other (*distinctiveness* property), Hamming Distance metric is applied on the generated keys in order to measure the distinctiveness and temporal variance. The average Hamming Distance is calculated as 49.7% and 49.4% for distinctiveness and temporal variance, respectively. Since 50% difference is considered as perfect for both performance criteria, our results show that the generated keys possess almost perfect distinctiveness and temporal variance. All in all, evaluations of the computational complexity, communication complexity and memory requirements of the system show that our proposed SKA-PSAR system provides a good trade-off between security and complexity such that it is possible to reach 100% correct and 0% false key generation rates within 133.70 seconds of system run time and using less than 300 KBytes of data communications and 39 Mbytes of memory usage. On the other hand, 91.6% correct and 0% false key generation rates are possible to be achieved in 20.55 seconds with 50 Kbytes of data communications and 38 Mbytes of memory usage.

Our SKA-PSAR system is built on SKA-PS [17] with some important differences that yield higher randomness level. Other than comparison of randomness level, we also compared SKA-PSAR and SKA-PS in terms of other performance metrics. It is possible to achieve good correct and false key generation rates, distinctiveness and temporal variance rates. However, SKA-PS computational, communication complexities and memory usage is much better than SKA-PSAR. Thus, we can say that SKA-PSAR provides a trade-off between high key randomness and operational performance without sacrificing from security.

Bibliography

- [1] Arvind, D. K. and Bates, A. “The Speckled Golfer”. In: *Proceedings of the ICST 3rd International Conference on Body Area Networks*. BodyNets '08. Tempe, Arizona: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, 28:1–28:7. ISBN: 978-963-9799-17-2. URL: <http://dl.acm.org/citation.cfm?id=1460257.1460295>.
- [2] Bao, S. D., Poon, C. C. Y., Zhang, Y., and Shen, L. “Using the Timing Information of Heartbeats as an Entity Identifier to Secure Body Sensor Network”. In: *IEEE Transactions on Information Technology in Biomedicine* 12.6 (Nov. 2008), pp. 772–779. ISSN: 1089-7771. DOI: [10.1109/TITB.2008.926434](https://doi.org/10.1109/TITB.2008.926434).
- [3] Bao, S. D., Shen, L. F., and Zhang, Y. T. “A novel key distribution of body area networks for telemedicine”. In: *IEEE International Workshop on Biomedical Circuits and Systems, 2004*. Dec. 2004, pp. 1–17. DOI: [10.1109/BIOCAS.2004.1454091](https://doi.org/10.1109/BIOCAS.2004.1454091).
- [4] Bao, S. D., Zhang, Y. T., and Shen, L. F. “Physiological Signal Based Entity Authentication for Body Area Sensor Networks and Mobile Healthcare Systems”. In: *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. Jan. 2005, pp. 2455–2458. DOI: [10.1109/IEMBS.2005.1616965](https://doi.org/10.1109/IEMBS.2005.1616965).
- [5] Bassham, L. E., Rukhin, A. L., Soto, J., Nechvatal, J. R., Smid, M. E., Barker, E. B., Leigh, S. D., Levenson, M., Vangel, M., Banks, D. L., Heckert, N. A., Dray, J. F., and Vo, S. *SP 800-22 Rev. 1a. A Statistical Test Suite for Random and Pseudo-*

- random Number Generators for Cryptographic Applications*. Tech. rep. National Institute of Standards & Technology, 2010.
- [6] Biel, L., Pettersson, O., Philipson, L., and Wide, P. “ECG analysis: a new approach in human identification”. In: *IEEE Transactions on Instrumentation and Measurement* 50.3 (June 2001), pp. 808–812. ISSN: 0018-9456. DOI: [10.1109/19.930458](https://doi.org/10.1109/19.930458).
- [7] Bui, F. M. and Hatzinakos, D. “Biometric Methods for Secure Communications in Body Sensor Networks: Resource-efficient Key Management and Signal-level Data Scrambling”. In: *EURASIP J. Adv. Signal Process* 2008 (Jan. 2008), 109:1–109:16. ISSN: 1110-8657. DOI: [10.1155/2008/529879](https://doi.org/10.1155/2008/529879). URL: <https://doi.org/10.1155/2008/529879>.
- [8] Chen, G. “Are electroencephalogram (EEG) signals pseudo-random number generators?” In: *Journal of Computational and Applied Mathematics* 268 (2014), pp. 1–4. ISSN: 0377-0427. DOI: <https://doi.org/10.1016/j.cam.2014.02.028>. URL: <http://www.sciencedirect.com/science/article/pii/S037704271400123X>.
- [9] Cherukuri, S., Venkatasubramanian, K. K., and Gupta, S. K. S. “Biosec: a biometric based approach for securing communication in wireless networks of biosensors implanted in the human body”. In: *Proceedings of the International Conference on Parallel Processing Workshops*. 2003, pp. 432–439.
- [10] Curtis, D. W., Pino, E. J., Bailey, J. M., Shih, E. I., Waterman, J., Vinterbo, S. A., Stair, T. O., Guttag, J. V., Greenes, R., and Ohno-Machado, L. “SMART-An Integrated Wireless System for Monitoring Unattended Patients”. In: *Journal of the American Medical Informatics Association : JAMIA* 15.1 (Jan. 2008), pp. 44–53. ISSN: 1067-5027. DOI: [10.1197/jamia.M2016](https://doi.org/10.1197/jamia.M2016).
- [11] Gerhardt, I. *Random Number Tests*. URL: <https://gerhardt.ch/random.php> (visited on 11/12/2018).

- [12] Goldberger, A. L., Goldberger, Z. D., and Shvilkin, A. *Clinical Electrocardiography: A Simplified Approach: Expert Consult*. 7th edition. Orlando: Elsevier, 2012.
- [13] Gu, Y. Y., Zhang, Y., and Zhang, Y. T. “A novel biometric approach in human verification by photoplethysmographic signals”. In: *4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine, 2003*. Apr. 2003, pp. 13–14. DOI: [10.1109/ITAB.2003.1222403](https://doi.org/10.1109/ITAB.2003.1222403).
- [14] Hong, T., Bao, S. D., Zhang, Y. T., Li, T., and Yang, P. “An improved scheme of IPI-based entity identifier generation for securing body sensor networks”. In: *33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2011, Boston, MA, USA, August 30 - Sept. 3, 2011*. 2011, pp. 1519–1522. DOI: [10.1109/IEMBS.2011.6090366](https://doi.org/10.1109/IEMBS.2011.6090366). URL: <https://doi.org/10.1109/IEMBS.2011.6090366>.
- [15] Juels, A. and Wattenberg, M. “A Fuzzy Commitment Scheme”. In: *Proceedings of the 6th ACM Conference on Computer and Communications Security. CCS '99*. Kent Ridge Digital Labs, Singapore: ACM, 1999, pp. 28–36. ISBN: 1-58113-148-8. DOI: [10.1145/319709.319714](https://doi.org/10.1145/319709.319714). URL: <http://doi.acm.org/10.1145/319709.319714>.
- [16] Karaođlan Altop, D., Levi, A., and Tuzcu, V. “Deriving cryptographic keys from physiological signals”. In: *Pervasive and Mobile Computing* 39 (2017), pp. 65–79. ISSN: 1574-1192. DOI: <https://doi.org/10.1016/j.pmcj.2016.08.004>, URL: <http://www.sciencedirect.com/science/article/pii/S1574119216301304>.
- [17] Karaođlan Altop, D., Seymen, B., and Levi, A. “SKA-PS: Secure key agreement protocol using physiological signals”. In: *Ad Hoc Networks* 83 (2019), pp. 111–124. ISSN: 1570-8705. DOI: <https://doi.org/10.1016/j.adhoc.2018.09.003>. URL: <http://www.sciencedirect.com/science/article/pii/S1570870518306425>.

- [18] Karaođlan, D. and Levi, A. “A Survey on the Development of Security Mechanisms for Body Area Networks”. In: *The Computer Journal* 57 (Jan. 2014), pp. 1484–1512. DOI: [10.1093/comjnl/bxt077](https://doi.org/10.1093/comjnl/bxt077).
- [19] Krawczyk, H., Bellare, M., and Canetti, R. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104. 1997. URL: <https://www.rfc-editor.org/info/rfc2104>.
- [20] Kreiseler, D. and Bousseliot, R. “Automatisierte EKG-Auswertung mit Hilfe der EKG-Signaldatenbank CARDIODAT der PTB”. In: *Biomedizinische Technik/Biomedical Engineering* 40.1 (2009), pp. 319–320.
- [21] Laerhoven, K. V., Villar, N., and Gellersen, H. “A layered approach to wearable textile networks”. In: *2003 IEE Eurowearable*. Sept. 2003, pp. 61–66. DOI: [10.1049/ic:20030148](https://doi.org/10.1049/ic:20030148).
- [22] Lukowicz, P., Anliker, U., Ward, J., Troster, G., Hirt, E., and Neufelt, C. “AMON: a wearable medical computer for high risk patients”. In: *Proceedings. Sixth International Symposium on Wearable Computers*. Oct. 2002, pp. 133–134. DOI: [10.1109/ISWC.2002.1167230](https://doi.org/10.1109/ISWC.2002.1167230).
- [23] Mano, M. and Ciletti, M. *Digital Design: With an Introduction to the Verilog HDL*. 5th ed. London: Prentice Hall, 2012.
- [24] Massey, J.L. “Theory and practice of error control codes”. In: *Proceedings of the IEEE* 74 (Oct. 1986), pp. 1293–1294. DOI: [10.1109/PROC.1986.13626](https://doi.org/10.1109/PROC.1986.13626).
- [25] Minsky, Y., Trachtenberg, A., and Zippel, R. “Set Reconciliation with Nearly Optimal Communication Complexity”. In: *IEEE Trans. Inf. Theor.* 49.9 (Sept. 2006), pp. 2213–2218. ISSN: 0018-9448. DOI: [10.1109/TIT.2003.815784](https://doi.org/10.1109/TIT.2003.815784). URL: <https://doi.org/10.1109/TIT.2003.815784>.
- [26] Moosavi, S. R., Nigussie, E., Levorato, M., Virtanen, S., and Isoaho, J. “Low-Latency Approach for Secure ECG Feature Based Cryptographic Key Generation”. In: *IEEE Access* 6 (2018), pp. 428–442. ISSN: 2169-3536. DOI: [10.1109/ACCESS.2017.2766523](https://doi.org/10.1109/ACCESS.2017.2766523).

- [27] O’Donoghue, J. and Herbert, J. “Profile based sensor data acquisition in a ubiquitous medical environmet”. In: *Proceedings of the Pervasive Computing and Communications Workshops (PerComW)*. IEEE Computer Society, Washington, Pisa, Italy, 13–17 March 2006, pp. 570–574.
- [28] Ortiz-Martin, L., Picazo-Sanchez, P., Peris-Lopez, P., and Tapiador, J. “Heartbeats Do Not Make Good Pseudo-Random Number Generators: An Analysis of the Randomness of Inter-Pulse Intervals”. In: *Entropy* 20.2 (2018). ISSN: 1099-4300. DOI: [10.3390/e20020094](https://doi.org/10.3390/e20020094), URL: <http://www.mdpi.com/1099-4300/20/2/94>.
- [29] Pankanti, S. *Biometrics: Personal Identification in Networked Society*. 1st edition. Springer US, 2006.
- [30] Paradiso, R., Loriga, G., and Taccini, N. “A wearable health care system based on knitted integrated sensors”. In: *IEEE Transactions on Information Technology in Biomedicine* 9.3 (Sept. 2005), pp. 337–344. ISSN: 1089-7771. DOI: [10.1109/TITB.2005.854512](https://doi.org/10.1109/TITB.2005.854512).
- [31] Poon, C. C. Y., Zhang, Y. T., and Bao, S. D. “A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health”. In: *IEEE Communications Magazine* 44.4 (2006), pp. 73–81. ISSN: 0163-6804. DOI: [10.1109/MCOM.2006.1632652](https://doi.org/10.1109/MCOM.2006.1632652).
- [32] Ragesh, G and Kaliaperumal, B. “An Overview of Applications, Standards and Challenges in Futuristic Wireless Body Area Networks”. In: *International Journal of Computer Science Issues* 1694-0784 9 (Jan. 2012), pp. 180–186.
- [33] Rodola, G. *psutil documentation*. Available at <https://psutil.readthedocs.io/en/latest/>. Accessed: 2018-06-15. 2018.
- [34] Ross, A. and Jain, A. K. “Biometrics, Overview”. In: *Encyclopedia of Biometrics*. Ed. by Li, S. Z. and Jain, A. Boston, MA: Springer US, 2009, pp. 168–172. ISBN: 978-0-387-73003-5. DOI: [10.1007/978-0-387-73003-5_182](https://doi.org/10.1007/978-0-387-73003-5_182). URL: https://doi.org/10.1007/978-0-387-73003-5_182.

- [35] Rostami, M., Juels, A., and Koushanfar, F. “Heart-to-heart (H2H): authentication for implanted medical devices”. In: *Proceedings of the ACM SIGSAC conference on Computer and communications security*. Berlin, Germany: ACM, New York, USA, 2013, pp. 1099–1112.
- [36] Seepers, R. M., Strydis, C., Sourdis, I., and Zeeuw, C. I. D. “On Using a Von Neumann Extractor in Heart-Beat-Based Security”. In: *2015 IEEE Trustcom-BigDataSE-ISPA*. Vol. 1. Aug. 2015, pp. 491–498. DOI: [10.1109/Trustcom.2015.411](https://doi.org/10.1109/Trustcom.2015.411).
- [37] Seepers, R. M., Weber, J. H., Erkin, Z., Sourdis, I., and Strydis, C. “Secure Key-exchange Protocol for Implants Using Heartbeats”. In: *Proceedings of the ACM International Conference on Computing Frontiers*. CF ’16. Como, Italy: ACM, 2016, pp. 119–126. ISBN: 978-1-4503-4128-8. DOI: [10.1145/2903150.2903165](https://doi.org/10.1145/2903150.2903165), URL: <http://doi.acm.org/10.1145/2903150.2903165>.
- [38] Shamir, A. “How to Share a Secret”. In: *Commun. ACM* 22.11 (Nov. 1979), pp. 612–613. ISSN: 0001-0782. DOI: [10.1145/359168.359176](https://doi.org/10.1145/359168.359176), URL: <http://doi.acm.org/10.1145/359168.359176>.
- [39] Shannon, C. E. “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423. ISSN: 0005-8580. DOI: [10.1002/j.1538-7305.1948.tb01338.x](https://doi.org/10.1002/j.1538-7305.1948.tb01338.x).
- [40] Shi, J., Lam, K., Gu, M., Li, M., and Chung, S. “Towards Energy-Efficient Secure Communications Using Biometric Key Distribution in Wireless Biomedical Healthcare Networks”. In: *2009 2nd International Conference on Biomedical Engineering and Informatics*. Oct. 2009, pp. 1–5.
- [41] Siddiqui, M. S. and Hong, C. S. “Security issues in wireless mesh networks”. In: *Proceedings of the International Conference on Multimedia and Ubiquitous Engineering*. IEEE Computer Society, Washington, Seoul, Korea, 26–28 April 2007, pp. 717–722.

- [42] Uludag, U., Pankanti, S., Prabhakar, S., and Jain, A. K. “Biometric cryptosystems: issues and challenges”. In: *Proceedings of the IEEE* 92.6 (June 2004), pp. 948–960. ISSN: 0018-9219.
- [43] Varshney, U. “Pervasive Healthcare and Wireless Health Monitoring”. In: *Mobile Networks and Applications* 12.2 (June 2007), pp. 113–127. ISSN: 1572-8153. DOI: [10.1007/s11036-007-0017-1](https://doi.org/10.1007/s11036-007-0017-1). URL: <https://doi.org/10.1007/s11036-007-0017-1>.
- [44] Venkatasubramanian, K. K., Banerjee, A., and Gupta, S. K. S. “EKG-based key agreement in Body Sensor Networks”. In: *IEEE INFOCOM Workshops 2008* (2008), pp. 1–6.
- [45] Venkatasubramanian, K. K., Banerjee, A., and Gupta, S. K. S. “Plethysmogram-based secure inter-sensor communication in Body Area Networks”. In: *MILCOM 2008 - 2008 IEEE Military Communications Conference*. Nov. 2008, pp. 1–7. DOI: [10.1109/MILCOM.2008.4753199](https://doi.org/10.1109/MILCOM.2008.4753199).
- [46] Wood, A. D., Stankovic, J. A., Virone, G., Selavo, L., He, Z., Cao, Q., Doan, T., Wu, Y., Fang, L., and Stoleru, R. “Context-aware wireless sensor networks for assisted living and residential monitoring”. In: *IEEE Network* 22.4 (July 2008), pp. 26–33. ISSN: 0890-8044. DOI: [10.1109/MNET.2008.4579768](https://doi.org/10.1109/MNET.2008.4579768).
- [47] Yang, G. Z. *Body Sensor Networks*. first. Springer-Verlag London, 2006.
- [48] Yuzuguzel, H., Niemi, J., Kiranyaz, S., Gabbouj, M., and Heinz, T. “ShakeMe: Key Generation from Shared Motion”. In: *13th IEEE International Conference on Pervasive Intelligence and Computing* (2015), pp. 2130–2133.
- [49] Zhong, L., Sinclair, M., and Bittner, R. “A phone-centered body sensor network platform cost, energy efficiency and user interface”. In: *International Workshop on Wearable and Implantable Body Sensor Networks (BSN’06)*. Apr. 2006, pp. 179–182. DOI: [10.1109/BSN.2006.4](https://doi.org/10.1109/BSN.2006.4).