EKG-based Key Agreement in Body Sensor Networks

Krishna Kumar Venkatasubramanian, Ayan Banerjee, and Sandeep K. S. Gupta
Department of Computer Science and Engineering
Arizona State University
Tempe, Arizona 85287
{kkv,abanerj3,sandeep.gupta}@asu.edu
(http://www.impact.asu.edu)

***Abstract*—Preserving a person's privacy in an efficient manner is very important for critical, life-saving infrastructures like Body Sensor Networks (BSN). This paper presents a novel key agreement scheme which allows two sensors in a BSN to agree to a common key generated using electrocardiogram (EKG) signals. This *EKG-based Key Agreement (EKA)* scheme aims to bring the "plug-n-play" paradigm to BSN security whereby simply deploying sensors on the subject can enable secure communication, without requiring any form of initialization such as pre-deployment. Analysis of the scheme based on real EKG data (obtained from MIT PhysioBank database) shows that keys resulting from EKA are: random, time variant, can be generated based on short-duration EKG measurements, identical for a given subject and different for separate individuals.**

I. Introduction

The ability to monitor a person's health in real-time is vital in emergency response scenarios such as disasters, battlefield monitoring, and individual ailments (heart-attacks, strokes). Recent developments in low-powered electronics has lead to the design of sensors which can perform patient monitoring [1] [2]. These sensors usually form a network on a person's body in order to collect, process and store health information. In the event of an emergency, first responders can deploy these *Body Sensor Networks* (BSN) on the victims which can then continuously monitor their vital signs, as required. The potentially life-saving nature of services provided by BSNs makes them a critical infrastructure.

As BSNs deal with personal health data, securing them, especially their communication over the wireless link, is equally critical. Lack of adequate security features may not only lead to a breach of patient privacy but also potentially allow adversaries to modify actual data resulting in wrong diagnosis and treatment [3] [4]. Indeed, protecting health data is a legal requirement as per the Health Insurance Portability and Accountability Act (HIPAA) [5] which mandates that all personally identifiable information be protected. Further, emergency situations usually have a certain amount of urgency associated with them, for example the golden hour in the event of a heart attack [6]. Therefore, security features for BSNs should be self-configurable and minimize large initialization (initial setup time) overhead as much as possible.

Sensors rely on cryptographic keys to secure their communication. We define security as the ability to protect data confidentiality, integrity and to authenticate the communicating entities. Key distribution in sensor networks usually requires some form of pre-deployment. Prominent examples include probabilistic key distribution [7], LEAP [8], SPINS [9], and even asymmetric crypto-systems [10]. However, given the progressively increasing size of BSNs (networks of size 190-255 nodes have already been proposed [11] [12]), and the urgency associated with their deployment in handling emergency scenarios, traditional approaches may potentially involve considerable latency during network initialization or any subsequent adjustments, due to their need for pre-deployment.

In this paper we present a novel key agreement scheme called *EKG based Key Agreement (EKA)*, which utilizes electrocardiogram (EKG) signals for generating cryptographic keys. Using EKA, secure inter-sensor communication can be executed in a "plug-n-play" manner, i.e. no network setup is required, keys for communication are generated from the environment (body) as and when needed. Additionally, the keys agreed upon by sensors using EKA meets the **design goals** suggested for physiological value based keys in [13], namely - the keys are *long, random, time variant*, generated from a *universally measurable* physiological stimuli (EKG), and are *distinctive* for different people. Further, EKA has an additional property of being able to generate keys with *low latency* , i.e. requires only a small duration of EKG measurement to generate keys. The *contribution* of this paper is two fold: 1) to show that EKG signals can be used for generating common cryptographic keys between two nodes in BSN, 2) to show that the keys generated meet the aforementioned design-goals, based on data from real patients.

The paper is organized as follows: Section II presents the related work, Section III presents the system model, and the trust and threat assumptions. Section IV presents EKA scheme in detail Section V presents the security analysis. Section VI presents the performance results followed by Section VII which presents the implementation issues pertaining to EKA. Finally Section VIII concludes the paper.

II. Related Work

The idea of using physiological signals for securing inter-sensor communication was first introduced in [14]. Building upon this initial idea, the authors in [15] [13] propose the use

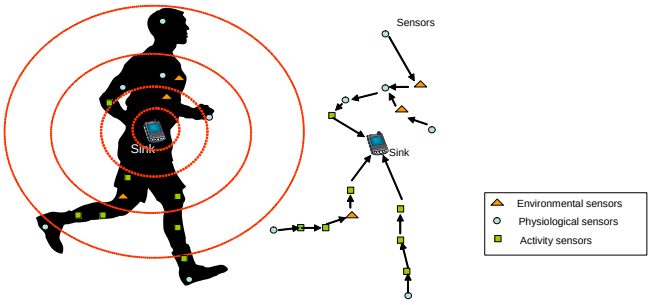

Fig. 1. System Model

of Inter-Pulse-Interval (IPI) to generate cryptographic keys. They derive IPI from the Photoplethysmogram (PPG) and the EKG time series by measuring the time difference between the peaks in the EKG/PPG signal. This series of IPI values were then encoded into binary to form a 128 bit cryptographic key whose hamming distance was shown to vary considerably when measured in two different people and by a few bits for the same person. In most cases the difference could be easily corrected using simple error correction codes.

Though the IPI meets all the requirements of a physiological key source mentioned in [13], it has two principal drawbacks: 1) The value of IPI measured at two different sensors had small differences under the best of circumstances. In most cases an error correction code is required for equalizing the keys; 2) From our own experiments with MIT PhysioBank database (http://www.physionet.org/physiobank/), we found that the inter-peak interval for EKG and PPG signal was on an average 560 msec. Therefore for every second only about 1.8 IPI values can be measured. To be able to use IPI values as keys, the authors in [13] required 67 values, which would take about half a minute of measurement, which makes it considerably slow for the real-time requirements of the BSN. The EKG provides us with an alternative which possesses the properties of IPI without its drawbacks, i.e. it produces identical keys, and measurement for about 5 seconds is enough to generate a key (see Section IV-A), thereby meeting our design goal of *low-latency*.

In [16] the authors suggest the use of EKG signals as a biometric to authenticate users. Their scheme requires the creation of an EKG template and then comparing their current EKG signals with this template to verify identity. The approach cannot be used here because it does not meet the requirements of a physiological key source, in that, the template is never changed. Further, it requires an extensive initialization time with respect to the generation of template with extensive 12 lead EKG collection, which makes it difficult to be used in a plug-n-play manner.

III. System Model

We assume a Body Sensor Network (BSN) to be a network of physiological and environmental monitoring sensors which are worn and/or implanted on a person called the *subject*. The sensors collect health and contextual data at regular intervals and forward it over a multi-hop network to a highly capable

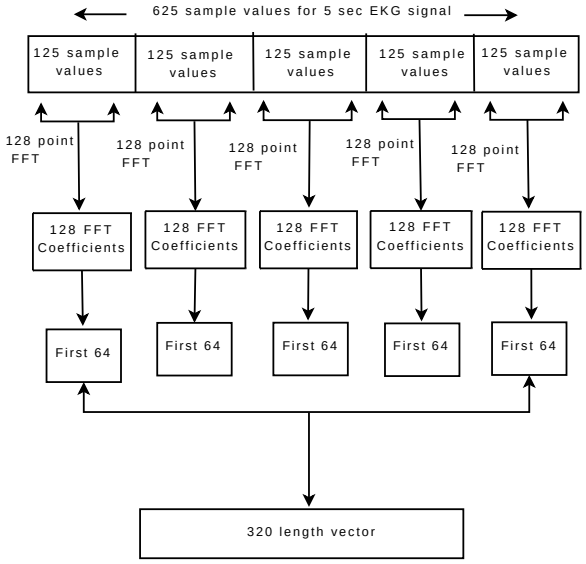

Fig. 2. Feature generation from EKG signal

sink node for further processing. We assume that the sensors *communicate through the wireless medium*, as wires running between sensors in a BSN will make it obtrusive. All sensors are assumed to be able to measure the EKG signals. Already physiological monitoring sensors are becoming multi-modal and are being able to sense multiple types of stimuli [17] (see Figure 1).

The threats faced by the BSN is primarily from adversaries who can eavesdrop on all the traffic within a BSN, inject messages, replay old messages, spoof node identities. The wireless medium is therefore not trusted by the sensors. The adversaries cannot measure EKG signals from the subject whose BSN they are trying to attack and therefore cannot directly know the keys being used. As BSNs are deployed on a person and not in a remote environment, compromising sensors is very difficult without being observed. We therefore assume node compromise to be unlikely. Note that in this work we focus solely on securing inter-sensor communication within the BSN. Communication from the sink onwards can utilize conventional security schemes such as SSL given the considerable capabilities of the entities involved.

IV. EKG Based Key Agreement

Secure communication between sensors in a BSN requires the presence of identical cryptographic keys at the communicating entities. In this section we present **EKG based Key Agreement (EKA)** scheme for enabling two sensors in a BSN to agree upon a *common key*, which is generated based on the EKG time series. The EKA scheme has two main aspects: *feature generation* and *key agreement*, which we now describe.

A. Feature Generation

When two sensors in a BSN want to securely communicate using EKG, they have to first extract features form it. We perform a frequency domain analysis of EKG signals for generating the features. This is because the frequency components

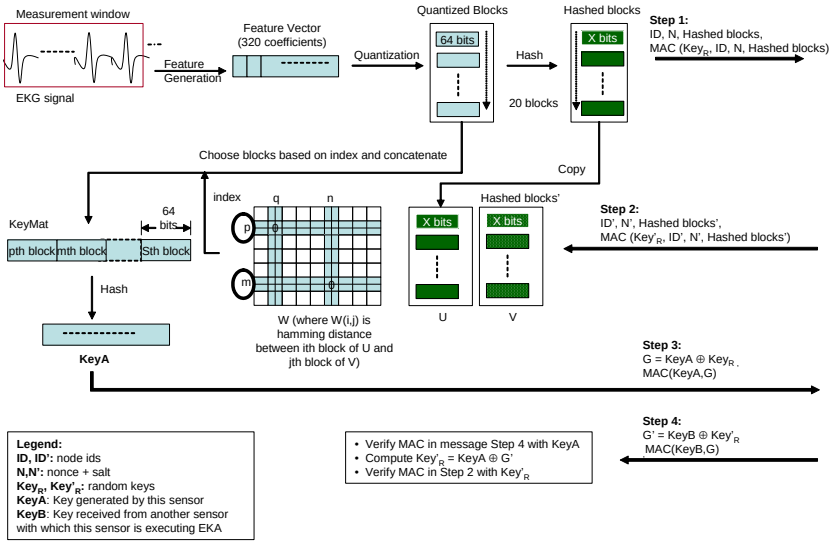

Fig. 3. Key Agreement at an arbitrary sensor in BSN

of physiological signals, at any given time, have similar values irrespective of where they are measured on the body. A time-domain analysis showed that the values of two EKG signals measured at different parts of the body (at different leads) have similar trend but diverse values.

The feature generation is executed by the two sensors, by sampling the EKG signal simultaneously, at a specific sampling rate for a fixed duration of time (125Hz and 5 seconds, respectively in our case). In order to remove measurement artifacts the signal is smoothed by removing the frequency components that do not contribute much to the overall power of the signal. The five second sample of the EKG signal (producing 625 samples) is then divided into 5 parts of 125 samples each. A 128 point Fast Fourier Transform (FFT) is then performed on each of these parts. The first 64 FFT coefficients (due to the symmetric nature of the spectrum) of each of the 5 parts are concatenated to form a feature vector F of 320 coefficients. The process is illustrated in Figure 2.

To generate a key from F, it is then quantized into a binary stream. In this regard, we divide F into 20 blocks each containing 16 coefficients. The 16 coefficients in each block are then quantized into binary. We chose to quantize F in small blocks primarily in order to capture the small variations in spectrum which when quantized produced keys with higher level of entropy. We tried linear, fixed and exponential quantization functions in EKA. The best results were obtained when the quantization function used was exponential with 12 steps. The quantization produces 4 bit binary value for each coefficient, resulting in 20, 64 bit blocks at each of the communicating sensors.

B. Key Agreement

Once the feature vector has been divided into blocks at each of the two communicating sensors, they are exchanged between them and processed before a common key can be agreed upon. The key agreement process consists of three phases: *commitment phase*, *processing phase* and *de-commitment phase*.

1) Commitment Phase: In this step the nodes exchange the blocks they generated by quantizing the FFT coefficients. Let $B_{s1} = \{b_1^1, b_2^1, ..., b_R^1\}$, denote the blocks generated at an arbitrary sensor $s1$. Here R is the index of the blocks and $|R| = 20$ in our case. As the blocks form the basis of the final key, they cannot be exchanged in the open. Therefore, each block is hashed using a one-way hash function before being transmitted. The commitment phase is carried out in two steps.

Step 1: $s1 \rightarrow s2 : <ID,\ N,\ hash(b_1^1, N)...hash(b_{20}^1, N),\ MAC(Key_R, ID, N, hash(b_1^1, N)...hash(b_{20}^1, N))>$

Step 2: $s2 \rightarrow s1 : <ID',\ N',\ hash(b_1^2, N')...hash(b_{20}^2, N'),\ MAC(Key'_R, ID', N', hash(b_1^2, N')...hash(b_{20}^2, N'))>$

Here - ID and ID' are node ids, N and N' are nonce for maintaining transaction freshness and also acted as salt to prevent potential dictionary attacks, $hash$ is a one way hash function and assumed to be secure (we utilized $SHA-256$ hash function in our experiments, any future reference to $hash$ assumes the usage of the same hash function) MAC is the message authentication code, and Key_R and Key'_R are random keys generated at each sensor (and at this point are not known to the other sensor). The presence of the MAC commits the sensors to their blocks, and will be used (in later steps) to detect adversaries. Note that we recommend that Key_R and Key'_R be at least 128 bit long in order to prevent brute-force guessing.

2) Processing Phase: Once the hashes of the blocks have been exchanged, each node arranges the received hash values, and the hashes of the local blocks into two 20×64 matrices. Let U and V be the two matrices, respectively. A matrix W of dimensions 20×20 is then computed from U and V, such

that each element $W(i,j)$ is equal to the hamming distance between the i th row of U and the j th row of V, where $1 < i,j < 20$. The matrix W is used to identify the indices of the blocks which are identical at both the sensors. This is done using the function $KeyGen$, described below, to derive the common key.

KeyGen(W)
1. Key = ϕ
2. $KeyMat$ = ϕ
3. while (all $W(i,j) \neq 1$) do
4. (i,j) = $min(W)$
5. if $(min(W) == 0)$
6. $KeyMat = KeyMat + b_i^1$
7. $W(i,k)$ = $1 \; \forall \; 1 \leq k \leq 20$
8. $W(u,j)$ = $1 \; \forall \; 1 \leq u \leq 20$
9. else
10. return $error$
11. end if
12. end while
11. Key = $hash(KeyMat)$
12. return Key

The vector $KeyMat$ is a collection of blocks (not their hashed values) which are identical at the communicating sensors. The common key Key is generated by hashing $KeyMat$. Depending upon the hash function used, the size of key would vary. We recommend the use of a hash function which produces an output as long as Key_R and Key'_R, i.e. at least 128 bits (if the output of the hash is longer we simply drop the extra bits). All commonly used hash functions such as MD5 and SHA256 meet this requirement, fulfilling our design goal of generating *long key*.

3) De-commitment Phase: Now that a key has been generated at the communicating sensors, they use it to verify the legitimacy of the blocks received in the commitment phase. To do so, they exchange the following messages:

Step 3: $s1 \rightarrow s2 :< G = Key_R \oplus KeyA, MAC(KeyA, G) >$

Step 4: $s1 \rightarrow s2 :< \widehat{G} = Key'_R \oplus KeyB, MAC(KeyB, G) >$

Here KeyA and KeyB are the keys generated during the processing phase at arbitrary sensors $s1$ and $s2$ (located on the same person), respectively, which should be identical. The sensors first verify the MAC in the messages received in de-commitment phase using their KeyA and KeyB, respectively. If the verification is successful, the nodes extract Key_R and Key'_R by XOR-ing KeyA and KeyB with G and $\widehat{G}$, respectively, which is then utilized to evaluate the MAC in the commitment phase. If the second evaluation is successful, the keys are accepted, otherwise not. Figure 3 shows the execution of three phases.

Given the value of KeyA and KeyB, the two sensors generate temporary keys K_{temp} = $hash(KeyA, l)$ = $hash(KeyB, l)$ for performing actual communication, where l is a random number. This protects the actual key from compromise, the sensors can further generate new temporary keys by varying the value of l from time to time, allowing the key generated using one set of EKG measurements for multiple communication sessions if need be.

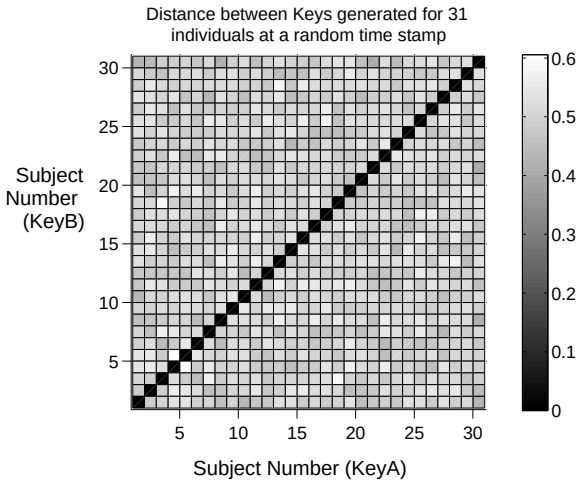

Fig. 4. Distance between Keys generated from Different Subjects

V. Security Analysis

An inherent problem with EKA is the length of the blocks it communicates during the commitment phase. As they are only 64 bits long, their hashes can be easily brute-forced by a capable adversary. One of the ways of overcoming this problem could be to use *key strengthening*. The main idea being that sensors hash each of their 64 bit blocks '2^n' times before transmitting them. An attacker while brute forcing has to generate a candidate block (64 bits long) and hash it '2^n' times before knowing weather the candidate is the actual block. Therefore an additive increase in the number of hash computations for the sender results in a multiplicative increase in the hashing requirements for the adversary. This results in a 2^n fold increase in the processing required for brute-forcing a 64 bit block which is analogous to brute forcing a $64 + n$ bit block. We recommend that sensors choose the value of 'n' as high as possible. However, key strengthening is an expensive proposition. The need to hash each of the 20 blocks 2^n times increases the costs further. For example if we choose 'n' to be 16 (which makes the block as secure as a 80 bit block) the total number of hash operations required to computed would be 131072!. The choice of this value actually used therefore, may depend upon the capabilities of the sensor and the amount of energy available at them, and hostility of the environment in which the subject carrying the sensors is. In a home environment for example, no key strengthening may be required, while in a shopping mall the maximum possible value of 'n' would be necessary. The use of key strengthening is however a stop-gap solution, we are currently working on generating longer blocks during the commitment phase so that brute-forcing them becomes impractical.

Assuming that the brute forcing of blocks is infeasible, the commitment and de-commitment phases makes it very difficult for adversaries to know the key being agreed upon. There are 4 reasons for this - a) The blocks are not exchanged

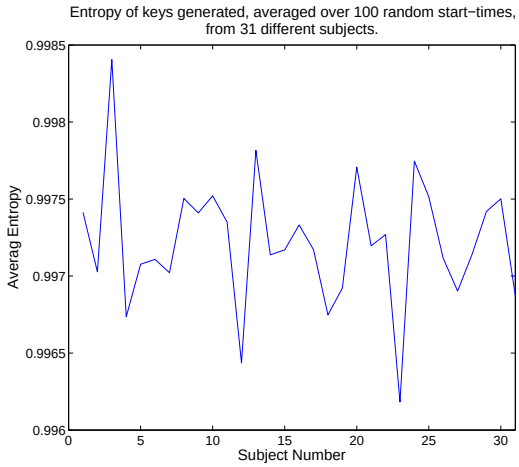

Fig. 5. Average Entropy of Keys for Different Subjects

directly, their hashes are, which gives no indication to their actual values; b) Any modification of the blocks being exchanged during agreement would be caught as soon as the de-commitment phase reveals the random key (Key_R or Key'_R), and the MAC received in commitment phase is verified; c) The random key itself cannot be guessed from the message de-commitment phase as KeyA and KeyB are random enough to prevent guessing (see Section VI) and vice-versa; and d) If an adversary poses as legitimate node (trying to form a key with an arbitrary node $s2$) and replays a previously captured message in Steps 1 and 3, then the message in Step 1 will be rejected outright by $s2$ as the nonce (N) would be identical to a previously used value. If the adversary updates the nonce value during Step 1, it will still not succeed because the KeyB generated by $s2$ would have changed (from the one which was used during the exchange, whose messages are now being replayed by the adversary), prompting $s2$ to reject the whole key agreement process.

VI. Performance Analysis

In this section we analyze the characteristics of the key generated by EKA in terms of our design goals of *randomness, time-variance and distinctiveness*. The analysis utilizes actual EKG data from 31 subjects obtained from the MIT PhysioBank database (http://www.physionet.org/physiobank/) for our validation. Each data value has a time-stamp associated with it, as the data was sampled at 125Hz, there is one EKG value every 8 msec. The EKA implementation and analysis was done using Matlab. In this section section, the notations KeyA and KeyB are used to denote keys generated by arbitrary sensors $s1$ and $s2$ (located in the same BSN) which are in the process of exchanging keys, respectively.

A. Distinctiveness

Our first experiment was to determine if the keys generated by the EKA scheme are *distinctive* for different people, i.e. produced identical keys at the same subject and divergent keys between two different subjects. Data from one lead was used to generate KeyA, and another to generate KeyB. For each subject the EKA scheme was executed over 100 random start-times and the average hamming distance between keys was computed. Figure 4 shows the result at a random start-time. The x-axis represents KeyA of all patients and y-axis represents KeyB for all patients. The colors represent the range within which the actual hamming distance between the two keys falls. It can be seen that all diagonal values are zero, which indicates that the keys generated from EKG signals of the same subject are identical, while keys from two different subjects are however not. The average hamming distance between keys generated from EKG of two different subjects was about 49.99% ($\approx$ 64 bits). As the adversary does not know which 64 bits are different, guessing the correct key would require $\binom{128}{64}$ attempts which is identical to brute-forcing a 124 bit key. The keys agreed upon through EKA meet our design goal of *distinctive*.

B. Randomness

Having long and distinctive keys is not enough, we need to ensure that they are unpredictable as well. To evaluate this, we computed the entropy of the keys generated for each subject over 100 random start-times. We found that in all cases the entropy values were close to 1 (see Figure 5), which signifies that the distribution of 1s and 0s in the key was quite uniform. However, entropy alone does not guarantee randomness, a key of length n with $n/2$ consecutive zeros followed by $n/2$ consecutive ones will also generate a high entropy. To ensure that our keys do not have large runs of zeros and ones, we tested each key generated by EKA using the two-tailed Runs-Test with a significance level of 5%. For each subject we executed the runs test for both KeyA and KeyB generated over 100 start-times. Only about 2% of keys generated failed the Runs-test. To ensure that the keys generated are random enough, once the communicating sensors have computed KeyA and KeyB, they could test the values for randomness. If the values turn out to be non-random the scheme could be executed again with a new set of EKG readings. The keys agreed upon through EKA meet our design goal of *randomness*.

C. Temporal Variance

We then evaluated the EKA generated keys for *temporal variance* in order to ensure that a new measurement of the EKG signals would not lead to the same keys. For each subject we computed the KeyA using data from 100 random start times and averaged the distance between the keys (as KeyB is same as KeyA it is not considered here). The minimum distance between any two KeyA's for a subject was about 49.98% ($\approx$ 64 bits). Therefore even if the current value of KeyA or KeyB generated from a subject is known, it does not necessarily make knowing the value of KeyA or KeyB from subsequent measurements any easier. The keys agreed upon through EKA meet our design goal of *temporal variance*.

VII. Implementation Issues

In this section we discuss some of the issues pertaining to the implementation of the EKA scheme. It involves three

main steps: EKG measurement, feature extraction, and key agreement. In order to be able to use EKA, the sensors have to be able to start measuring the EKG values at both the communicating sensors more or less simultaneously. If the measurement start-times are far apart the features derived may not be similar at both ends, due to the time variant nature of EKG. One advantage of using frequency domain features is that the required level of synchronization is not very strict. Our experiments show good results even with a 40 msec skew start of the EKG measurement at either of the sensors. If EKG signals were be analyzed in time domain, the level of synchronization would be limited by the sampling rate (f). For example if the sampling rate is 125Hz then a data value is generated every 8 msec. To ensure no data points are missed at either sensors, the sensor clocks have to be synchronized to within 8 msec. Given that existing synchronization protocol for sensor networks such as [18] can achieve 1 μsec synchronization, any such scheme can be applied here. The feature selection aspect of EKA requires the implementation of 128 point FFT on the sensors. In [19], the authors have proposed the use of dedicated FFT processor in the sensor architecture. Their design can perform variable length FFT (128 point to 1024 point FFT) on a sensor with an energy dissipation of 155nJ per FFT computation at a supply voltage of 350 mV and a clock frequency of 10 KHz. The processor, implemented using 0.18 μm CMOS technology will add only a small footprint to sensors. The key agreement aspect is the simplest of the three steps. It involves the computation of hashes, and comparing blocks of bits, and generating pseudo-random numbers which are relatively simple to implement on any platform such as TinyOS (http://www.tinyos.net). Currently we are in the process of implementing EKA on the Ayushman health monitoring test-bed [20] at the IMPACT Lab at Arizona State University.

VIII. Conclusions and Future Work

In this paper we presented a novel means of using EKG signals for distributing keys to enable secure inter-sensor communication. We further showed that it meets all the design goals we set forth for using physiological values for generating keys. Our security analysis and simulation studies show EKG can be used for generating keys in a BSN. In the future we are planning to improve the security of the EKA scheme by increasing the exchanged block size, implement the scheme on actual hardware, evaluate the key generated in more detail, and analyze the performance of the scheme in terms of computational and other overheads.

Acknowledgements

This research is supported in part by National Science Foundation Grant CNS-0617671 and MediServe Information Systems.

References

[1] K. Lorincz, D. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, S. Moulton, and M. Welsh, "Sensor Networks for Emergency Response: Challenges and Opportunities," *In Proc. of IEEE Pervasive Computing, Special Issue on Pervasive Computing for First Response*, vol. 3, no. 4, pp. 16–23, Oct-Dec 2004.

[2] L. Schwiebert, S. K. S. Gupta, and J. Weinmann, "Research Challenges in Wireless Networks of Biomedical Sensors," July 2001, pp. 151–165, In Proc. of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking.

[3] K. Venkatasubramanian and S. K. S. Gupta, *Chapter 15: Security for Pervasive Healthcare*, Y. Xiao, Ed. CRC Press, 2007.

[4] ——, "Security for pervasive health monitoring sensor applications," December 2006, pp. 197–202, In Proc. of the 4th International Conference on Intelligent Sensing and Information Processing.

[5] HIPAA-Report 2003, "Summary of HIPAA Health Insurance Probability and Accountability Act," US Department of Health and Human Service, May 2003.

[6] S. K. S. Gupta, T. Mukherjee, and K. Venkatasubramanian, "Criticality Aware Access Control Model for Pervasive Applications," March 2006, pp. 251–257, In Proc. of 4th IEEE Conference on Pervasive Computing Pervasive Computing and Communications.

[7] L. Eschenauer and V. D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," November 2002, pp. 41–47, In Proc. of the 9th ACM conference on Computer and Communications Security.

[8] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 2, no. 4, pp. 500–528, November 2006.

[9] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar, "SPINS: Security Protocol for Sensor Networks," *Wireless Networks*, vol. 8, no. 5, pp. 521–534, September 2002.

[10] D. J. Malan, M. Welsh, and M. D. Smith, "A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography," October 2004, pp. 71–80, In Proc. of IEEE 2nd International Conference on Sensor and Ad Hoc Communications and Networks.

[11] N. Kern, B. Schiele, and A. Schmidt, "Multi-Sensor Activity Context Detection for Wearable Computing," Eindhoven, The Netherlands, pp. 220–232, November 2003, in Proceedings of European Symposium on Ambient Intelligence.

[12] S. Choi, S. J. Song, K. Sohn, H. Kim, J. Kim, J. Yoo, and H. J. Yoo, "A Low-power Star-topology Body Area Network Controller for Periodic Data Monitoring Around and Inside the Human Body," 2006, pp. 139–140, In Proc. of IEEE 10th International Symposium on Wearable Computing.

[13] C. C. Y. Poon, Y.-T. Zhang, and S.-D. Bao, "A Novel Biometrics Method To Secure Wireless Body Area Sensor Networks for Telemedicine And M-Health," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 73–81, 2006.

[14] S. Cherukuri, K. Venkatasubramanian, and S. K. S. Gupta, "BioSec: A Biometric Based Approach for Securing Communication in Wireless Networks of Biosensors Implanted in the Human Body," October 2003, pp. 432–439, In Proc. of Wireless Security and Privacy Workshop 2003.

[15] S. D. Bao, Y. T. Zhang, and Y.-T. Zhang, "Physiological Signal Based Entity Authentication for Body Area Sensor Networks and Mobile Healthcare Systems," September 2005, pp. 2455–2458, In Proc. of the IEEE 27th Conference on Engineering in Medicine and Biology.

[16] L. Biel, O. Pettersson, L. Philipson, and P. Wide, "ECG Analysis: A New Approach in Human Identification," *IEEE Transaction on Instrumentation and Measurement*, vol. 50, no. 3, pp. 808–812, June 2001.

[17] K. Ouchi, T. Suzuki, and M. Doi, "LifeMinder: A Wearable Healthcare Support System Using User's Context," July 2002, pp. 791–792, In Proc. of 22th International Conference on Distributed Computing Systems Workshops.

[18] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," vol. 36, 2002, pp. 147–163, In Proc. of the 5th symposium on Operating systems Design and Implementation.

[19] A. Wang and A.Chandrakasan, "A 180-mV Subthreshold FFT Processor Using a Minimum Energy Design Methodology," *IEEE Journal on Solid State Circuits*, vol. 40, no. 1, pp. 310–319, January 2005.

[20] K. Venkatasubramanian, G. Deng, T. Mukherjee, J. Quintero, V. Annamalai, and S. K. S. Gupta, "Ayushman: A wireless sensor network based health monitoring infrastructure and testbed," June/July 2005, pp. 406–407, In Proc. of the IEEE International Conference on Distributed Computing in Sensor Systems.