TIMAR: An Efficient Key Management Scheme for Ubiquitous Health Care Environments*

Syed Muhammad Khaliq-ur-Rahman Raazi
Department of Computer Engineering, Kyung Hee University, Korea.
raazi@khu.ac.kr

Sungyoung Lee†
Department of Computer Engineering, Kyung Hee University, Korea.
sylee@oslab.khu.ac.kr

Young-Koo Lee
Department of Computer Engineering, Kyung Hee University, Korea.
yklee@khu.ac.kr

ABSTRACT

Wireless Sensor Networks (WSN) are worthy to use in ubiquitous healthcare environments. They provide comfort to patients and make patient monitoring systems more efficient. WSN, when applied in ubiquitous healthcare environments, have different characteristics and security requirements. Number of sensors is very small as compared to other WSN applications and all nodes are close to each other in the network. Possibility of human intervention in ubiquitous healthcare environments and randomness properties of biometric measurements, which are collected in ubiquitous healthcare environments, reduce the security requirements as compared to other WSN applications. Key Management Schemes, proposed for generic WSN, prove to be overly complex and inefficient for ubiquitous healthcare environments. In this paper, we present TIMAR, which is an efficient Key Management Scheme specifically designed for ubiquitous healthcare environments.

Keywords

Humanware, Healthcare, Security, Key Management, Body Area Networks.

1. INTRODUCTION

Sensor networks are resource constrained data centric networks. They are used to sense certain phenomena and then relay the sensed information towards a server using wireless communications [1]. In ubiquitous healthcare applications, sensor nodes are required to relay the sensed information from human body to a central server in real time. Sensed information can be biometric or multimedia data from human body. In turn, this relayed information is used or processed at the central server by some software according to the requirements of the application.

Security is an important in WSN applications. We need to maintain confidentiality, authenticity and message integrity in all communications. Key management plays an important role in security of WSN. However, we need to keep in mind the characteristics of a network and requirements of the applications using the network while designing key management protocol for it. For example, highly secure state-of-the-art mechanisms such as TLS [2] and Kerberos [3] are too heavy to run on resource constrained WSN.

In this paper, we present TIMAR[1], which is a distributed key management scheme specifically designed for ubiquitous healthcare environments. Rest of this paper is organized as follows. Section 2 discusses the ubiquitous healthcare environment in detail and outlines differences between generic WSN applications and ubiquitous healthcare applications. Section 3 outlines the related work followed by section 4, which states the system model and assumptions. Section 5 presents our scheme. Section 6 analyzes our scheme and compares it with other state-of-the-art key management schemes. Section 7 concludes the paper. In this paper we use many abbreviations and notations like WSN for Wireless Sensor Networks. Refer to Table 1 for complete list of notations used in this paper.

2. UBIQUITOUS HEALTHCARE ENVIRONMENTS

Sensor nodes have less memory, computation and communication capabilities. Also, they have limited energy resources. However, applications of WSN may differ in many aspects. Some applications, like ubiquitous healthcare environments, may differ in scale, topology and security requirements from other WSN applications. In this section, we will discuss how

†Corresponding author

[1]TIMAR is a word from urdu language. It is used for someone, who helps in taking care of or takes care of an ill person's health. We have named our scheme so because it helps in taking care of people's health

Table 1: List of Used Notations	
WSN	**Wireless Sensor Network**
MS	**Medical Server**
PS	**Personal Server**
SN^i	**Sensor Node i**
K^i_{bsc}	**Basic Key of Node** i
K_{comm}	**Communication Key**
K^i_{admin}	**Administrative Key** i

ubiquitous healthcare applications differ from generic WSN applications.

Firstly, there is a difference in the number of sensor nodes. In generic applications of WSNs, number of nodes may be in thousands while healthcare applications require very few nodes, which may be less than twenty. If there are too many nodes, it will wouldn't be comfortable for patients, who have to wear these devices, and may not be acceptable to them.

Secondly, there is a big difference in area of use. In generic applications of WSN, nodes may be scattered in large areas like battlefields. On the other hand, all nodes in ubiquitous healthcare applications are placed in a very small area i.e. on a human body. This brings all the nodes in communication range of each other. Researchers have proposed communication protocols keeping in mind such topology [4].

Thirdly, human intervention is possible in ubiquitous healthcare applications. In generic applications of WSN, it is assumed that human intervention is not possible. In some cases, it might become essential to physically replace a compromised node. For example, a node must be replaced immediately if it is the only node measuring a serious patient's heart rate. Also, if a node is compromised in ubiquitous healthcare environments, it is possible to physically remove it or turn it off rather than evicting it through key management protocol.

Lastly, keys don't need to be generated at the sensor nodes in ubiquitous healthcare environments. Ubiquitous healthcare applications sense and relay multimedia and biometric information, which exhibit sufficient randomness properties to be used as cryptographic keys. Such assumption regarding availability of random numbers can not be taken in a generic WSN application. Some researchers have used biometric for key generation [5],[6]. Some argue that in a network formed on human body, keys need not be exchanged. [7],[8],[9]. They assume that two nodes can sense a biometric at the same time. In order to take care of possible error, they rely on error-correcting codes at both the communicating nodes. Apart from extra computations and time synchronization issues, this assumption requires sensor nodes to sense more than one biometric, which may not be feasible in a practical scenario. Also, such schemes do not take into account the nodes, which are not used for sensing biometrics e.g hand-held devices, which may be used in such applications. For more detail, refer to the system model described in 4.

Due to the differences between ubiquitous healthcare applications and generic WSN applications, it is important to have key management schemes, which are designed specifically for ubiquitous healthcare environments.

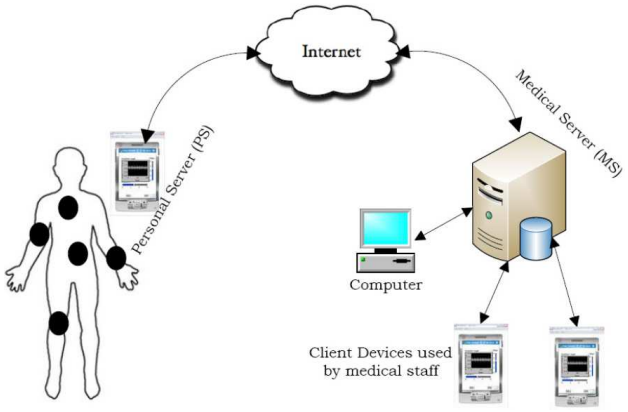

Figure 1: Model of Ubiquitous Healthcare Environment

3. RELATED WORK

Until now, researchers have not focused much on the key management issues of ubiquitous healthcare environments. Most of the related work is in ubiquitous healthcare applications is from WSN paradigm. The simplest key management solution in ubiquitous healthcare environment would be to distribute keys to each pair of communicating nodes before the deployment and then use them for the whole network lifetime. Care must be taken during key assignments otherwise it may result in inefficient security. For example, same key should not be assigned to multiple pair of nodes within a certain area. There are many other such issues and solutions have been proposed, which take care of such issues [10],[11],[12],[13][14].

If we keep on using same keys for longer periods of time, they may come under cryptanalytic attacks. Mica2 is real world example of a sensor node. At full power, its lifetime is expected to be two weeks [15]. In ubiquitous healthcare environments, network lifetime may be long and nodes' batteries can be recharged/replaced. Under such circumstances, it becomes necessary to periodically refresh keys.

Many schemes, which have the support of key refreshment, have been proposed for WSN. Key management scheme of Riaz et. al. [16] requires the base station to provide public keys to the communicating nodes. Drawback of Riaz et. al.'s scheme is that it communicates with the cluster head very frequently, which incurs significant communication overhead. Dutertre et. al. [17] have proposed a lightweight key management solution for wireless sensor networks by leveraging initial trust. Paek et. al. [18] base their scheme on regional and virtual groups. LEAP+ [19] is a localized scheme and one of the state-of-the-art solution for WSN. Drawback of LEAP+, Paek's and Dutertre's scheme is that they assume the network is safe during some initial time period. Also, all the three schemes are not designed keeping in mind the fact that all nodes are in communication range of each other.

SHELL [20] and MUQAMI [21] are lightweight solutions and suit the resource constrained sensor nodes well. Both these schemes are based on combinatorics and Exclusion Basis System (EBS) of matrices [22]. MUQAMI further improves the performance by distributing the key management responsibilities locally. Also, it makes use of key-chains [23], which are based on Lamport's one-time passwords [24]. Drawback of these schemes is that they are designed for large scale WSN while ubiquitous healthcare applications have very few nodes. When applied to small scale networks, like in ubiquitous healthcare applications, their performances drop considerably. Also, EBS based key management schemes are prone to collusion attacks [25].

[26] and [27] have introduced asymmetric cryptography in wireless sensor networks using ECC. Both these schemes try to move the burden of asymmetric cryptography to a trusted server as much as possible. From the perspective of ubiquitous healthcare environment, drawback of these two schemes is that they are designed keeping in mind large number of sensor nodes.

Researchers have focused on the usage of biometric data as keys and authentication codes [5],[6],[7],[8],[9] as already discussed in Section 2. Based on that, we have proposed a complete key management architecture keeping in mind the characteristics and application requirements of WBANs. To our knowledge, this is the first time a key management scheme is proposed specifically for WBANs.

4. SYSTEM MODEL AND ASSUMPTIONS

In ubiquitous healthcare environment, there are some sensor devices, which are capable of measuring biometrics and multimedia data from patients' body. These devices are placed on a patients' body in such a way that they do not hamper their daily routine. Then there is a Personal Server (PS), which can be a laptop or a hand held device. Sensor nodes measure the biometrics and forward the body related information to the PS. In turn, the PS relays this information to the Medical Server (MS), directly or through the internet. One PS is associated with each patient. Multiple PS can be associated with one MS. The MS stores and processes information of all patients, who are associated with it. MS processes each patient's data and generate alert for the concerned people. Also, authorized people can access the required information from the MS. Model of a ubiquitous healthcare environment is presented in Figure 1.

We assume that the PS and all sensor devices are constrained in energy because they use rechargeable batteries. Unlike the generic applications of WSN, we don't need to take care of node compromises. However, we need to take care of confidentiality, message integrity and node authenticity.

5. TIMAR

In our scheme, each node has to refresh key on its turn according to a key refreshment schedule. The key refreshment schedule is issued by the PS, which refreshes it periodically. Our scheme uses three types of keys: communication key K_{comm}, administrative key K_{admin} and basic key K_{bsc}. Communication key K_{comm} is a network wide key and is used to transfer data securely through the network. K_{comm} is managed by the PS itself. Frequent use of K_{comm} makes it vulnerable to cryptanalytic attacks so it must be refreshed regularly. Administrative key K_{admin}, which is also a group key, is used to refresh K_{comm}. Since K_{admin} is not used very frequently, it is less likely to come under cryptanalytic attack.

If K_{admin} is managed by a single node, the node may become an attacker's center of attraction. Also, its energy drainage will be faster as compared to the other nodes. Therefore, we distribute the responsibility of key management throughout the network. If we want to make the network more resilient, we can increase the number of administrative keys.

Third type of key that we use is K_{bsc}. K_{bsc} is used to recover from rare, extreme and unexpected failures. Every node has its own K_{bsc} that is not known to any other node in the network. It is refreshed after every use. MS communicates this key to the PS whenever required.

5.1 Initial Deployment

In the first phase, PS is deployed. Throughout the network lifetime, the PS is connected with the MS. The PS comes pre-loaded with K_{admin}, K_{comm} and identities of all the nodes that are to be deployed in the network. Also, authentication codes, which are used to authenticate the sensor nodes, of all sensor nodes are pre-loaded in the PS. These codes are used to authenticate newly deployed nodes. After the PS is deployed, sensor devices are deployed on various parts of the body. Sensor nodes come pre-loaded with K_{admin} and their respective K_{bsc}. Initial deployment takes place as follows: -

1. PS is deployed and its connection with the MS is established.
2. Sensor nodes are deployed. Each sensor node sends its ID and authentication to the PS in the discovery message. This communication takes place using K_{admin}.
3. After all nodes are deployed, the PS sends initial value of K_{comm} and the key refreshment schedule to all the nodes in the network using K_{admin}.

As soon as the last expected node is discovered or a timer expires, the PS calculates the refreshment schedule and broadcasts it along with K_{comm}. PS does not wait for new nodes after the timer expires. Nodes, which are discovered after the timeout are treated as added nodes and deployed in a manner explained in subsection 5.3

5.2 Re-keying

K_{comm} is refreshed after some time period using K_{admin}. In order to refresh K_{comm}, the PS selects suitable value of a biometric as the new value of K_{comm}. It then encrypts the new value of K_{comm} with K_{admin} and broadcasts it into the network.

Administrative key is also refreshed periodically. When the turn of sensor node i arrives, sensor node i waits for a certain period of time, chooses a suitable value of a biometric as new value for K_{admin}, encrypts it with the current value of K_{admin} and broadcasts it in the network. All nodes can

Table 2: Storage requirements (in bytes) of each type of node in all the three schemes

	Personal Server	Sensor Node
MUQAMI	$(z \times ((l \times (k+m)) + r - (k+m) + 2)) + (4 \times r)$	$(z \times ((k+4) + [(2 \times (l-1) \times (k+m))/r]))$
LEAP+	$z \times (r+2)$	$z \times (r+2)$
TIMAR	$(2 \times z) + (4 \times r)$	$(3 \times z) + 4$

decrypt the new key as they know the old one. When the key refreshment schedule expires, the PS calculates the new schedule, encrypts it in the current value of K_{admin} and broadcasts it into the network.

In some cases, administrative key needs to be refreshed out of schedule. For example, if there is some malicious activity, the PS may decide to refresh K_{admin}. In such scenario, K_{admin} is refreshed as follows: -

1. The PS sends key refresh message to the node, which is supposed to refresh K_{admin} next time.
2. The node immediately chooses a suitable biometric and broadcasts the refresh message in the network.

Sometimes, K_{admin} needs to be refreshed through K_{bsc}. Although it is a rare scenario that K_{admin} is compromised, but we think it is necessary to explain the procedure. In such scenario, K_{admin} is refreshed in the following manner: -

1. PS asks the MS to refresh K_{admin} using K_{bsc} of each node.
2. MS encrypts a new value of K_{admin} in K^i_{bsc} of each sensor node i and sends all these values to the PS using the secure connection established in the initial deployment phase. New values of K^i_{bsc} are also present in these individual messages.
3. PS forwards the individual messages MS has sent to each sensor node. PS can not decrypt these messages because they are encrypted in K_{bsc} of each node separately.
4. MS sends the new value of K_{admin} to the PS using the secure connection established in the initial deployment phase.
5. PS refreshes K_{comm} using K_{admin}.

After this point in time, remaining key refreshment schedule is followed.

5.3 Node Addition

In ubiquitous healthcare environments, sometimes it becomes necessary to place new devices on patient's body or to replace some of the existing ones. Under normal network operation, if a stranger node contacts the PS it is classified as a malicious activity in order to avoid unnecessary drainage of energy from the sensor nodes. However, in the node addition phase, the PS expects messages from new nodes. New nodes are added in the following manner: -

1. MS informs the PS about new sensor nodes. Apart from ID and authentication codes of the new nodes, MS also sends the initial value of K_{admin} that comes pre-loaded in the new nodes. PS switches to node addition phase and starts expecting discovery messages.
2. New nodes are deployed on patient's body. They send their ID and authentication codes to the PS using the pre-loaded value of K_{admin}.
3. As soon as the last node sends its discovery message or a timer expires, PS broadcasts current values of K_{comm} and K_{admin} and the remaining key refreshment schedule for the new nodes. All nodes, except the newly deployed ones, ignore this message.

PS considers the new nodes too when it issues the next key refreshment schedule.

6. ANALYSIS AND COMPARISON

In this section, we will analyze our scheme and compare it with two other state-of-the-art schemes LEAP+ [19] and MUQAMI [21]. SHELL [20] is also one of the state-of-the-art key management solutions for WSNs but it requires services of the neighbouring cluster head nodes and in ubiquitous healthcare applications, presence of a neighbouring CH node can not be guaranteed.

6.1 Storage Overhead

Considering the storage requirement of a sensor node, only three keys K_{comm}, K_{admin} and K_{bsc} need to be stored. Apart from that, we need to keep into account the storage requirement for the key refreshment schedule. A sensor node can keep track of its turn with the help of two short integers. One has a counter to keep track of its turn and the other indicates the length of time it should wait before it refreshes K_{admin}. If we consider that a short integer requires 2 bytes and key length is z bytes, Then the storage requirement of a sensor nodes in TIMAR becomes: -

$$SR^{TIMAR}_{SN} = (3 \times z) + 4 \quad (1)$$

On the PS, we only need to store two keys K_{admin} and K_{comm}. Apart from that, we also need to store the complete refreshment schedule for K_{admin}. A sensor node's identity can be stored using 2 bytes. Another 2 bytes are required to specify after how much time a node should refresh K_{admin}. So, the storage requirement of a PS in TIMAR can be expressed as: -

$$SR^{TIMAR}_{PS} = (2 \times z) + (4 \times r) \quad (2)$$

where r is the number of nodes in the network formed on the body.

Table 3: Average number of messages transmitted by each type of node when administrative key is refreshed in all the three schemes

	Personal Server	Sensor Node
MUQAMI	$(k+m)\times(1+(1/l))$	$((k+m)/r)\times(1+(1/l))$
LEAP+	r	r
TIMAR	$1/r$	$1/r$

Average storage requirements of a node in LEAP+ is fairly straightforward. Apart from the pairwise key shared with each node in the cluster, it has to store two more keys i.e. its cluster key and the communication key. So, the storage requirements of a node in LEAP+ can be expressed as: -

$$SR_{PS\vee SN}^{LEAP+} = z\times(r+2) \tag{3}$$

In MUQAMI, the PS node has to store one K_{comm} and $K_{cn,ch}$ apart from $K_{ch,sn}$ of all SN nodes and the key-chains $K_{ch,kg}$ of all KG nodes in the cluster. Also, it has to store the EBS matrix [22]. If we consider that storing EBS data for each node takes 4 bytes (2 bytes for storing node identity and 2 bytes for storing key pattern for that node), it takes $4\times r$ bytes to store EBS matrix on the PS. So, average storage requirement of a PS (in bytes) in MUQAMI becomes: -

$$SR_{PS}^{MUQAMI} = (z\times((l\times(k+m))+r-(k+m)+2))+(4\times r) \tag{4}$$

where k and m are EBS parameters and l is the length of the key-chains [23], which are used by MUQAMI for key management. SN nodes have to store k admin keys apart from $K_{ch,sn}$, K_{comm}, K_{bsc} and K_{disc}. So, the average storage requirement of a sensor node in MUQAMI becomes: -

$$SR_{SN}^{MUQAMI} = z\times(k+4) \tag{5}$$

Among the sensor nodes, MUQAMI also requires key generating (KG) nodes to store two key-chains: one for the administrative key, which it generates and one for $K_{ch,kg}$. Also, it has to store $k-1$ EBS keys along with three other keys: K_{comm},K_{bsc} and K_{disc}. So the storage requirement of a KG node in MUQAMI comes out to be: -

$$\begin{aligned} SR_{KG}^{MUQAMI} &= z\times(2\times l+(k-1)+3) \\ &= z\times(2\times(l+1)+k) \end{aligned} \tag{6}$$

Since we have $k+m$ KG nodes out of r nodes inside the cluster, average storage requirement of each node within a cluster comes out to be: -

$$\begin{aligned} & SR_{SN\cup KG}^{MUQAMI} \\ = \; & z\times\frac{(r-(k+m))(k+4)+(k+m)(2(l+1)+k)}{r} \\ = \; & z\times\frac{r\times(k+4)+(k+m)\times(2\times(l+1)-4)}{r} \\ = \; & z\times((k+4)+\frac{2\times(l-1)\times(k+m)}{r}) \end{aligned} \tag{7}$$

Note that $(k+m) << r$ only for large scale networks. Table 2 compares the storage requirements of TIMAR with MUQAMI and LEAP+. It is clear from table 2 that storage overhead of our scheme is negligible as compared to other schemes. This is true not only for sensor nodes but also for the PS.

6.2 Communication Overhead

Communication is the most energy consuming activity in WSN. In ubiquitous healthcare environments, all nodes are in communication range of each other. Therefore, it is sufficient to analyze the average number of messages transmitted by each type of node in every phase.

Initial deployment phase of TIMAR is fairly simple. Every sensor node has to send 1 discovery message each. PS also has to send 1 message, in which it sends K_{comm} and initial key refreshment schedule. Initial deployment phase of MUQAMI is also simple. Every sensor node has to send 1 discovery message each. In return, the PS has to send 1 message to each node in the network, which makes the total number of messages transmitted by the PS equal to r.

In LEAP+'s initial deployment phase, the PS has to send one broadcast message to all nodes in the network. All nodes reply and pair-wise keys are established. After that, it sends its cluster key to each of the r nodes one by one and then broadcasts its group key in the network. Also, it also has to reply to the initial messages sent by other nodes. So, the average number of messages transmitted by PS in the initial deployment phase of LEAP+ can be written as: -

$$\begin{aligned} Avg\_Msg\_Count\_Init_{PS}^{LEAP+} &= (2\times r)+2 \\ &= 2\times(r+1) \end{aligned} \tag{8}$$

The sensor nodes does not have to broadcast the communication key, Therefore, average number of messages transmitted by sensor nodes in the initial deployment phase of LEAP+ can be written as: -

$$Avg\_Msg\_Count\_Init_{SN}^{LEAP+} = (2\times r)+1 \tag{9}$$

Clearly, our scheme has less overhead than the other two schemes. Analysis of node addition phase is similar to the analysis of initial deployment phase.

In the communication key refreshment phase of TIMAR, only the PS needs to broadcast 1 message. It is the same in case of LEAP+ also i.e. only the PS needs to send 1 message and no other node needs to transmit any message. In MUQAMI, the PS will have to send one message to each key-generating node. So, the average number of messages transmitted by the PS in MUQAMI can be expressed as: -

$$Avg\_Msg\_Count\_Rekey\_Comm_{PS}^{MUQAMI} = k+m \tag{10}$$

After receiving key refresh message from the PS, each key-generating node will broadcast this message in the network. There are $k+m$ key-generating nodes among a total of r nodes in a network. Therefore, expression for the average number of messages transmitted by a sensor node for re-

freshment of communication key becomes: -

$$Avg\_Msg\_Count\_Rekey\_Comm_{SN}^{MUQAMI} = \frac{k+m}{r} \quad (11)$$

In order to refresh the administrative key in TIMAR, each sensor node has to send 1 message after r key refreshments. Similarly, the PS also has to send 1 message after r key refreshments in order to issue the new schedule. So, the average number of messages sent by each node for administrative key refreshments in TIMAR becomes: -

$$Avg\_Msg\_Count\_Rekey\_Admin_{SN \vee PS}^{TIMAR} = \frac{1}{r} \quad (12)$$

In order to refresh the administrative key in LEAP+, every node has to send one message to each of r other nodes in the network. For administrative key refreshment in MUQAMI, apart from sending $k+m$ messages to the key generating nodes, PS also has to send one message after every l key refreshments in order to get new seed value for key-chains. So, average number of messages transmitted by PS for refreshment of administrative key in MUQAMI becomes: -

$$\begin{aligned} & Avg\_Msg\_Count\_Rekey\_Admin_{PS}^{MUQAMI} \\ = \; & (k+m) \times (1 + (\frac{1}{l})) \end{aligned} \quad (13)$$

There are $k+m$ key-generating nodes out of the total of r nodes in the network, so the average number of messages transmitted by a sensor node for refreshment of administrative key in MUQAMI can be expressed as: -

$$\begin{aligned} & Avg\_Msg\_Count\_Rekey\_Admin_{SN}^{MUQAMI} \\ = \; & \frac{k+m}{r} \times (1 + (\frac{1}{l})) \end{aligned} \quad (14)$$

If we look at the comparison in table 3, it is evident that our scheme is more efficient than MUQAMI and LEAP+ for the administrative key refreshment phase also.

In order to verify the analysis done in this section, we have performed simulation of all the three scheme and recorded the energy consumed in each phase. Values assumed for simulation parameters were: $k = m = 4$, $l = 32$, $r = 16$ and $z = 16$. Simulation was programmed in Tools Command Language (tcl8.0). Calculations for the cost of key-generation were based on [28]. Transmitting and receiving power levels were set to $1mW$ and $0.1mW$ respectively, which is realistic according to [29]. Calculations for computation and communication costs were based on [30] and [15] respectively. Figure 2 shows that simulations results support the analysis done in this section.

7. CONCLUSIONS

From this research, we learn that all applications of WSN are not similar and same solution can not fit in all WSN applications. Ubiquitous healthcare is one such application, whose characteristics and security requirements are different from generic WSN applications. We have presented TIMAR, which is a key management scheme designed specifically for applications in ubiquitous healthcare.

8. ACKNOWLEDGMENTS

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement)" (IITA-2009-(C1090-0902-0002)). This work also, was supported by the Korea Science & Engineering Foundation(KOSEF) grant funded by the Korea government(MEST) (No. 2008-1342), and was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology (2009-0076798). This work is supported by the IT R&D program of MKE/KEIT, [10032105, Development of Realistic Multiverse Game Engine Technology].

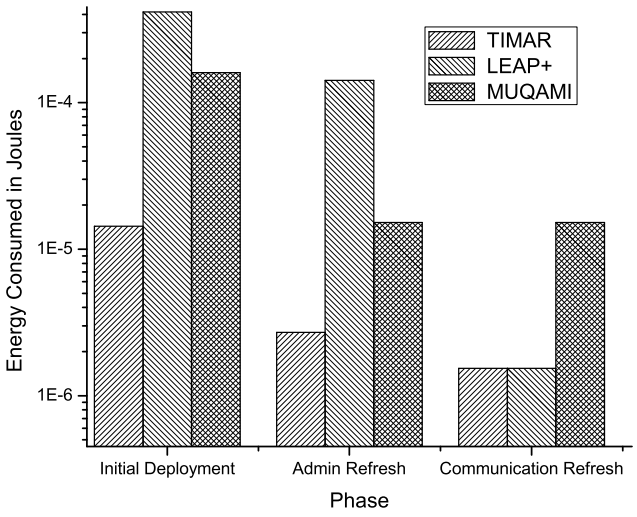

Figure 2: Comparison of Average Energy Consumed by a node in different phases of each scheme

9. REFERENCES

[1] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless microsensor network models," *ACM Mobile Computing and Comm.*, vol. 6, no. 2, pp. 1–8, 2002.

[2] T. Dierks and C. Allen, "The tls protocol version 1.0," 1999.

[3] J. Kohl and C. Neuman, "The kerberos network authentication service (v5)," 1993.

[4] C. Otto, A. Milenkovic, C. Sanders, and E. Jovanov, "System architecture of a wireless body area sensor network for ubiquitous health monitoring," *Journal of Mobile Multimedia*, vol. 1, no. 4, pp. 307–326, 2006.

[5] C. Poon, Y. Zhang, and S. Bao, "A novel biometrics method to secure wireless body area sensor networks for telemedicine and m-health," *IEEE Communication Magazine*, vol. 44, no. 4, pp. 73–81, 2006.

[6] S. Cherukuri, K. K. Venkatasubramanian, and E. K. S. Gupta, "BioSec: A biometric based approach for securing communication," in *in Wireless Networks of Biosensors Implanted in the Human Body, Workshop on Wireless Security and Privacy (WiSPr), International Conference on Parallel Processing Workshops*, 2003.

[7] K. K. Venkatasubramanian and S. K. S. Gupta, "Security for pervasive health monitoring sensor applications," in *ICISIP '06: Proceedings of the 4th International Conference on Intelligent Sensing and*

Information Processing, (Bangalore, India), pp. 197–202, December 2006.

[8] F. M. Bui and D. Hatzinakos, "Biometric methods for secure communications in body sensor networks: Resource-efficient key management and signal-level data scrambling," *EURASIP Journal on Advances in Signal Processing*, 2008.

[9] T. Falck, H. Baldus, J. Espina, and K. Klabunde, "Plug 'n play simplicity for wireless medical body sensors," *Mob. Netw. Appl.*, vol. 12, no. 2-3, pp. 143–153, 2007.

[10] G. Li, J. He, and Y. Fu, "A hexagon-based key predistribution scheme in sensor networks," in *ICPPW '06: Proceedings of the 2006 International Conference Workshops on Parallel Processing*, (Washington, DC, USA), pp. 175–180, IEEE Computer Society, 2006.

[11] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, (Washington, DC, USA), p. 197, IEEE Computer Society, 2003.

[12] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 42–51, ACM, 2003.

[13] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 41–47, ACM, 2002.

[14] S. A. Çamtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 15, no. 2, pp. 346–358, 2007.

[15] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 162–175, ACM, 2004.

[16] R. Shaikh, S. Lee, M. Khan, and Y. Song, "LSec: Lightweight security protocol for distributed wireless sensor networks," in *11th IFIP International Conference on Personal Wireless Communications PWC'06*, vol. 4217 of *LNCS*, (Spain), pp. 367–377, September 2006.

[17] B. Dutertre, S. Cheung, and J. Levy, "Lightweight key management in wireless sensor networks by leveraging initial trust," Tech. Rep. SRI-SDL-04-02, SDL, 2004.

[18] K.-J. Paek, J. Kim, C.-S. Hwang, and U.-S. Song, "An energy-efficient key management protocol for large-scale wireless sensor networks," in *MUE '07: Proceedings of the 2007 International Conference on Multimedia and Ubiquitous Engineering*, (Washington, DC, USA), pp. 201–206, IEEE Computer Society, 2007.

[19] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Trans. Sen. Netw.*, vol. 2, no. 4, pp. 500–528, 2006.

[20] K. Ghumman, "Location-aware combinatorial key management scheme for clustered sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 8, pp. 865–882, 2006. Senior Member-Younis„ Mohamed F. and Senior Member-Eltoweissy„ Mohamed.

[21] S. M. K. Raazi, A. M. Khan, F. Khan, S. Lee, and Y. Song, "MUQAMI: A locally distributed key management scheme for clustered sensor networks," in *Trust Management*, vol. Volume 238/2007 of *IFIP International Federation for Information Processing*, pp. 333–348, Springer Boston, 2007.

[22] M. Eltoweissy, M. H. Heydari, L. Morales, and I. H. Sudborough, "Combinatorial optimization of group key management," *J. Netw. Syst. Manage.*, vol. 12, no. 1, pp. 33–50, 2004.

[23] G. Dini and I. M. Savino, "An efficient key revocation protocol for wireless sensor networks," in *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, (Washington, DC, USA), pp. 450–452, IEEE Computer Society, 2006.

[24] L. Lamport, "Password authentication with insecure communication," *Commun. ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[25] M. R., M. R., and E. M., "TKGS: Threshold-based key generation scheme for wireless ad hoc networks," in *IEEE International Conference on Computer Communication and Networking(ICCCN'04)*, October 2004.

[26] H. Qiang, C. Johnas, K. Hisashi, L. Bede, and Z. Jinyun, "Fast authenticated key establishment protocols for self-organizing sensor networks," in *WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, (New York, NY, USA), pp. 141–150, ACM, 2003.

[27] P. Kotzanikolaou, E. Magkos, D. Vergados, and M. Stefanidakis, "Secure and practical key establishment for distributed sensor networks," in *Security and Communication Networks*, Wiley InterScience, 2009.

[28] D. Seetharam and S. Rhee, "An efficient pseudo random number generator for low-power sensor networks," in *LCN '04: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, (Washington, DC, USA), pp. 560–562, IEEE Computer Society, 2004.

[29] G. Xing, C. Lu, Y. Zhang, Q. Huang, and R. Pless, "Minimum power configuration in wireless sensor networks," in *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, (New York, NY, USA), pp. 390–401, ACM, 2005.

[30] R. Venugopalan, P. Ganesan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Encryption overhead in embedded systems and sensor network nodes: modeling and analysis," in *CASES '03: Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems*, (New York, NY, USA), pp. 188–197, ACM, 2003.