Analyzing & Identifying The Superlative Keying Protocol To Support Sensor Networking

Akula Santha Manohari[1], Ravi Kumar. Singisetti[2]

[1],M.Tech Student(Neural Networks), Gokul institute of Engineering & Technology, Piridi,Bobilli

[2],Asst.Professor, Gokul institute of Engineering & Technology, Piridi,Bobilli.

Abstract

The sensor networks like mobile sensor networks or the network that are built in a hurry are deployed in whimsical and undersigned configuration. For this reason, the sensor in such a network may become eventually adjacent to any other sensor in the network. In order to make a communication between every pair of adjacent sensors, in a secure way in such an arbitrary network, each sensor ,let x in the network be in need to store n-1 symmetric keys that sensor x shares with all the other sensors where n is the number that represents the count of the sensors in the network. When the number n is large and if the availability of storage in each sensor is reserved, then the storage requirement for the keying protocol is a bit hard. Some earlier achievements were made just to redesign this keying protocol and also to reduce the number of keys to be stored in each sensor .Those achievements have exhibited protocols that are accessible to collusion attacks, eavesdropping, and impersonation. We want to present a secure keying protocol that makes each sensor to store (n+1)/2 keys that are less that n-1 keys that are needed to be stored in each sensor in the original keying protocol. In this paper we also want to show that each sensor needs to store at least (n-1)/2 keys, in any of the fully secure keying protocol.

I. Introduction

A wireless sensor network (WSN) consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, etc. and to cooperatively pass their data through the network to a main location.

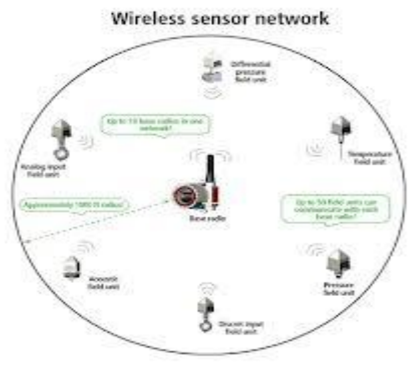


Fig.1: Wireless Sensor Network

Many wireless sensor networks are deployed in random and unintended style. Examples of such networks are networks of mobile sensors [11] and networks that are deployed in a hurry to monitor evolving crisis situations [7] or continuously changing battle fields [1]. In any such network, any deployed sensor can end up being adjacent to any other deployed sensor. Thus, each pair of sensors, say sensors x and y, in the network need to share a symmetric key, denoted Kx,y, that can be used to secure the communication between sensors x and y if these two sensors happen to be deployed adjacent to one another. In particular, if sensors x and y become adjacent to one another, then these two sensors can use their shared symmetric key Kx,y to authenticate one another (i.e. defend against impersonation) and to encrypt and decrypt their exchanged data messages (i.e. defend against eavesdropping).

The probabilistic keying protocol and grid keying protocol are two main keying protocols that were proposed in the past to reduce the number of stored keys in each sensor in the network.Unfortunately, the probabilistic keying protocol suffers from the following problem. The stored keys in any sensor x are independent of the identity of sensor x and so these keys cannot be used to authenticate sensor x to any other sensor in the network. In other word, the probabilistic protocol cannot defend against impersonation.In the grid keying protocol [8], each sensor is assigned an identifier which is the coordinates of a distinct node in a two-dimensional grid. Unfortunately, it turns out that the grid keying protocol is vulnerable to collusion. Specifically, a small gang of adversarial sensors in the network can pool their stored keys together and use the pooled keys to decrypt all the exchanged data messages in the sensor network.

2. CHALLENGES FOR SENSOR NETWORKS

As a third example, if the deployed sensor network is mobile, then a detailed plan of the initial topology may be of little value.) In this network, when a sensor x is deployed, it first attempts to identify the identity of each sensor adjacent to x, then starts to exchange data with each of those adjacent sensors. Any sensor z in this network can be an “adversary”, and can attempt to disrupt the communication between any two legitimate sensors, say sensors x and y, by launching the following two attacks:

a)Impersonation Attack:

Sensor z notices that it is adjacent to sensor x while sensor y is not. Thus, sensor z attempts to convince sensor x that it (z) is in fact sensor y. If sensor z succeeds, then sensor x may start to exchange data messages with sensor z, thinking that it is communicating with sensor y.

b)Eavesdropping Attack:

Sensor z notices that it is adjacent to both sensors x and y, and that sensors x and y are adjacent to one another. Thus, when sensors x and y start to exchange data messages, sensor z can copy each exchanged data message between x and y.If the network has n sensors, then each sensor in the network needs to store (n-1) symmetric keys before the network is deployed. If n is large, then the storage requirement, just to store the required shared keys, is relatively large, especially since the size of storage in each sensor is typically small.

To solve this problem, we present the following two results in this paper:

i) Efficiency:There is a keying protocol, where each sensor shares a distinct symmetric key with every other sensor in the network, and yet each sensor needs to store exactly (n+1)/2 symmetric keys, before the network is deployed.
ii) Optimality:In every keying protocol, where each sensor shares a distinct symmetric key with every other sensor in the network, each sensor needs to store at least (n-1)/2 symmetric keys, before the network is deployed.

3. PROPOSED PROTOCOL KEYING CONCEPTS & ALGORITHMS

1) Mutual Authentication: Sensor x authenticates sensor y, and sensor y authenticates sensor x.
2) Confidential Data Exchange: Encrypt and later decrypt all the exchanged data messages between x and y.

Where n denote the number of sensors in our network. Without loss of generality, we assume that n is an odd positive integer. Each sensor in the network has a unique identifier in the range 0…n - 1. We use ix and i_y to denote the identifiers of sensors x and y, respectively, in this network. Each two sensors, say sensors x and y, share a symmetric key denoted $K_{x,y}$ or $K_{y,x}$. Only the two sensors x and y know their shared key $K_{x,y}$. And if sensors x and y ever become neighbors in the network, then they can use their shared symmetric key $K_{x,y}$ to perform above two protocol keying concepts.

Mutual Authentication Protocol:
Before the sensors are deployed in a network, each sensor x is supplied with the following items:
1) One distinct identifier ix in the range 0...n-1
2) One universal key ux
3) (n-1)/2 symmetric keys Kx,y = H(ix/uy) each of which is shared between sensor x and another sensor y, where ix is below iy.
After every sensor is supplied with these items, the sensors are deployed in random locations in the network. Now if two sensors x and y happen to become adjacent to one another, then these two sensors need to execute a mutual authentication

protocol so that sensor x proves to sensor y that it is indeed sensor x and sensor y proves to sensor x that it is indeed sensor y. The mutual authentication protocol consists of the following six steps.

Step 1: Sensor x selects a random nonce nx and sends a hello message that is received by sensor y.

$$x \rightarrow y : hello(ix, nx)$$

Step 2: Sensor y selects a random nonce ny and sends a hello message that is received by sensor x.

$$x \leftarrow y : hello(iy, ny)$$

Step 3: Sensor x determines whether ix is below iy. Then it either fetches Kx;y from its memory or computes it. Finally, sensor x sends a verify message to sensor y.

$$x \rightarrow y : verify(ix, iy, H(ix|iy|ny|K_{x,y}))$$

Step 4: Sensor y determines whether iy is below ix. Then it either fetches Kx;y from its memory or computes it. Finally, sensor y sends a verify message to sensor x.

$$x \leftarrow y : verify(iy, ix, H(iy|ix|nx|K_{x,y}))$$

Step 5: Sensor x computes H(ix|iy|ny|Kx,y) and compares it with the received H(ix|iy|ny|Kx,y). If they are equal, then x concludes that the sensor claiming to be sensor y is indeed sensor y. Otherwise, no conclusion can be reached.

Step 6: Sensor y computes H(ix|iy|ny|Kx,y) and compares it with the received H(ix|iy|ny|Kx,y). If they are equal, then y concludes that the sensor claiming to be sensor x is indeed sensor x. Otherwise, no conclusion can be reached.

Data Exchange Protocol:

After two adjacent sensors x and y have authenticated one another using the mutual authentication protocol described in the previous section, sensors x and y can now start exchanging data messages according to the following data exchange protocol. (Recall that nx and ny are the two nonces that were selected at random by sensors x and y, respectively, in the mutual authentication protocol.)

Step 1: Sensor x concatenates the nonce ny with the text of the data message to be sent, encrypts the concatenation using the symmetric key Kx;y, and sends the result in a data message to sensor y.

$$x \rightarrow y : data(ix, iy, K_{x,y}(ny|text))$$

Step 2: Sensor y concatenates the nonce nx with the text of the data message to be sent, encrypts the concatenation using the symmetric key Kx;y, and sends the result in a data message to sensor x.

$$x \leftarrow y : data(iy, ix, K_{x,y}(nx|text))$$

4. DESIGN & IMPLEMENTATION OF THE SYSTEM

Design is concerned with identifying software components specifying relationships among components. Specifying software structure and providing blue print for the document phase. Modularity is one of the desirable properties of large systems. It implies that the system is divided into several parts. In such a manner, the interaction between parts is minimal clearly specified. Design will explain software components in detail. This will help the implementation of the system. Moreover, this will guide the further changes in the system to satisfy the future requirements.

Class Diagram contains the following elements:

- A class which represents entities with common characteristics or features attributes operations and associations.
- Association, which represent relationship between two or more classes where relationships have common characteristics or features like attributes and operations as in Fig.2.

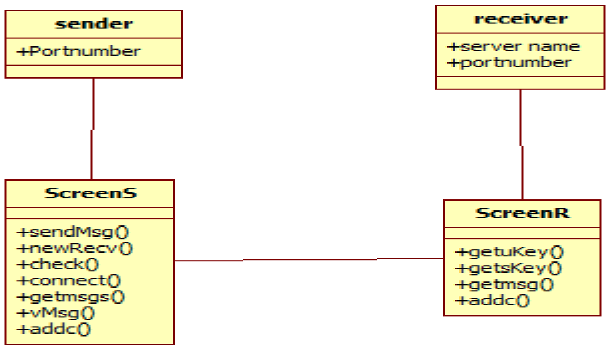


Fig 2: Inter-operational Class diagram for framework

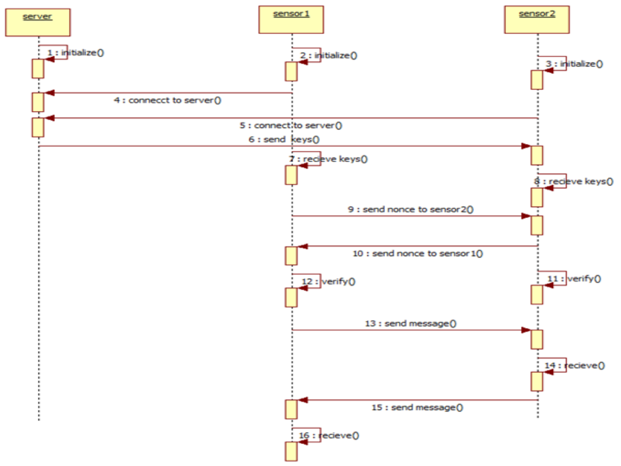
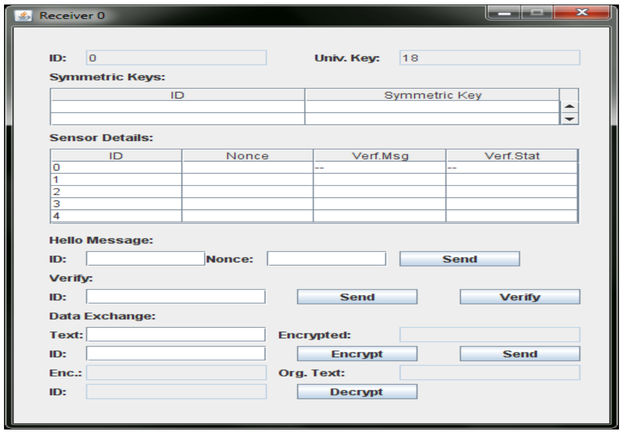


Fig 3: Inter-operational Sequence diagram for the Server & sensors

Sequence Diagrams represents the interactions between classes to achieve a result such as a usecase. The sequence diagram lists objects horizontally and time vertically, and models these messages over time. In this paper the designing of the sequence has been done on both server and the sensors and they were described as in the Fig.3

5. RESULTS

The below are the results obtained


Receiver 0

ID: 0 Univ. Key: 18

Symmetric Keys:

ID	Symmetric Key

Sensor Details:

ID	Nonce	Verf.Msg	Verf.Stat
0		--	--
1			
2			
3			
4			

Hello Message:

ID: Nonce:

Verify:

ID:

Data Exchange:

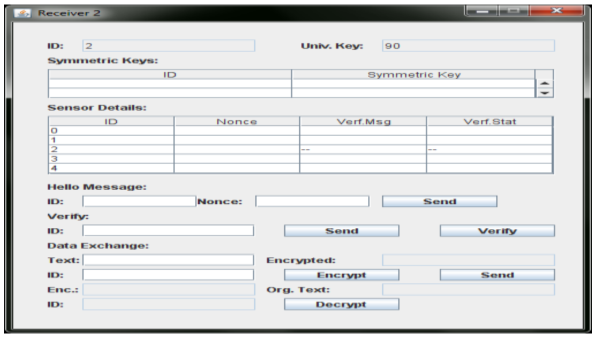
Text: Encrypted:

ID:

Enc.: Org. Text:

ID:

Fig.4: Receiver 0 Communication



ID: 2 Univ. Key: 90

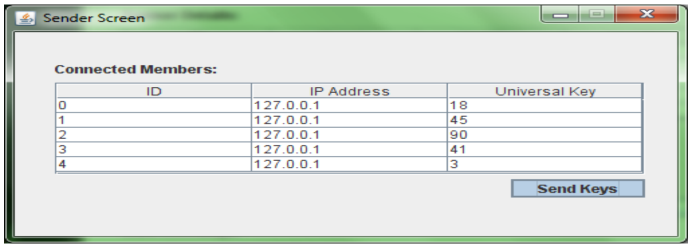
Symmetric Keys:

ID	Symmetric Key

Sensor Details:

ID	Nonce	Verf.Msg	Verf.Stat
0			
1			
2		--	--
3			
4			

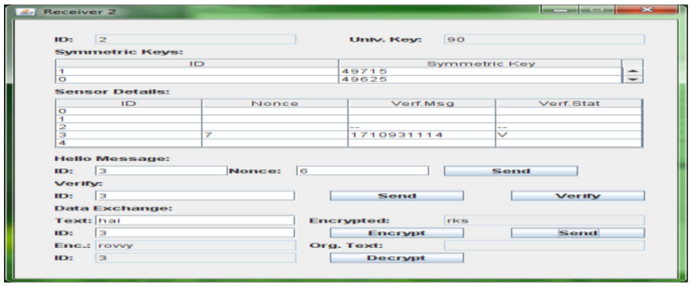
Fig.5: Receiver 2 Communication



Connected Members:

ID	IP Address	Universal Key
0	127.0.0.1	18
1	127.0.0.1	45
2	127.0.0.1	90
3	127.0.0.1	41
4	127.0.0.1	3

Fig.6: Sender Keys



ID: 2 Univ. Key: 90

Symmetric Keys:

ID	Symmetric Key
1	49715
0	49625

Sensor Details:

ID	Nonce	Verf.Msg	Verf.Stat
0			
1			
2		--	--
3	7	1710931114	V
4			

Hello Message:
ID: 3 Nonce: 6

Verify:
ID: 3

Data Exchange:
Text: hai Encrypted: rks
ID: 3
Enc.: rovvy Org. Text:
ID: 3

Fig.7: Receiver2 Authentication

CONCLUSION

In a sensor network with n sensors, each sensor needs to store n-1 shared symmetric keys for the secure communication in between each of the sensors. So the number of symmetric keys that are shared and stored in the sensor network is n (n-1).But theoretically $\binom{n}{2}$=n (n-1)/2 is the optimal number of shared symmetric keys stored in the sensor network for secure communication. Even though there are so many ways that endeavor to lower the number of shared symmetric keys, leads to a loss in security and they are affected by collusion too. This paper proposes a best keying protocol for sensor networks which wants to store only (n+1)/2 shared symmetric keys to each sensor. n(n + 1)/2, is the number of shared symmetric keys stored in a sensor network with n sensors, and for any key distribution scheme which is not vulnerable to collusion, it is close to the optimal number of shared symmetric keys. In addition to the low number of keys stored, and also its capability to withstand collusion between sensors our proposed keying protocol has two advantages .They are it is uniform that means we store the same number of keys in each of the sensor in that network and the other advantage is economy of scale in terms of computation and hence it is apt for a low-power computer where two sensors bind up to each other, there is only requirement of hashing (cheapest and fast method) for computation of a shared symmetric key, the computation of a shared symmetric key requires only hashing, which is a cheap computation and can be done fast. As our protocol has many desirable properties, such as efficiency, uniformity and security, we call this protocol the best keying protocol for sensor networks.

References

[1] S. Sana and M. Matsumoto, "Proceedings of a wireless sensor network protocol for disaster management," in Information, Decision and Control (IDC), 2007, pp. 209 –213.

[2] S. Hynes and N. C. Rowe, "A multi-agent simulation for assessing massive sensor deployment," Journal of Battlefield Technology, vol. 7, pp. 23–36, 2004.

[3] S. S. Kulkarni, M. G. Gouda, and A. Arora, "Secret instantiation in ad-hoc networks," Special Issue of Elsevier Journal of Computer Communications on Dependable Wireless Sensor Networks, vol. 29, pp. 200–215, 2005.

[4] A. Aiyer, L. Alvisi, and M. Gouda, "Key grids: A protocol family for assigning symmetric keys," in Proceedings of IEEE International Conference on Network Protocols (ICNP), 2006, pp. 178–186.

[5] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in Proceedings of the International Symposium on Distributed Autonomous Robotic Systems (DARS), 2002, pp. 299–308.

[6] Y. Fan and J. S. Thompson, "MIMO Configurations for Relay Channels: Theory and Practice," IEEE Trans. on Wireless Communications, Vol. 6, Issue 5, pp. 1774-1786, May 2007.

[7] A. Bletsas, A. Khisti, D. P. Reed, and A. Lippman, "A Simple Cooperative Diversity Method Based on Network Path Selection," IEEE Journal on Selected Areas in Communications, Vol. 24, Issue 3, pp. 659-672, Mar. 2006.

[8] A. Host-Madsen, "Capacity Bounds for Cooperative Diversity," IEEE Trans. on Information Theory, Vol. 52, No. 4, pp. 1522-1544, Apr. 2006.

[9] S. Krco and M. Dupcinov, "Improved Neighbor Detection Algorithm for AODV Routing Protocol," IEEE Communication Letters, Vol. 7, No. 12, pp. 584-586, Dec. 2003.

[10] Intersil Prism WLAN Chipset Application Note: Tutorial on Basic Link Budget Analysis, Available from http://www.sssmag. com/pdf/an9804.pdf, 2006.

[11] J. N. Laneman, "Cooperative Diversity in Wireless Networks: Algorithms and Architectures," Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, Aug. 2002.

Author List:

Akula Santha Manohari received B.Tech in Computer science from JNTUH, Pursuing M.Tech in Neural Networks from Gokul institute of Engineering & Technology Affiliated to JNTUK.. Her research area of interests are Data mining and computer networks

Ravi Kumar. Singisetti Received B.Tech from SISTAM at Srikakulam in the year 2008 and M.Tech from GOKUL institute of Engineering & Technology , piridi ,Bobilli.Presently Working as Ass.Professor in the Dept of CSE at institute of Engineering & Technology.