Security Management in Wireless Sensor Networks with a Public Key Based Scheme*

Al-Sakib Khan Pathan, Jae Hyun Ryu, Md. Mokammel Haque, and Choong Seon Hong

Department of Computer Engineering, Kyung Hee University
{spathan,jhryu,malinhaque}@networking.khu.ac.kr, cshong@khu.ac.kr

Abstract. This paper proposes an efficient approach for managing the security in WSN. Our approach uses the notion of public key cryptography in which two different keys are used for encryption and decryption of data. Our analysis and performance evaluations show that, our approach is viable with the specifications of today's Berkley/Crossbow MICA2dot motes.

1 Introduction

Among several public key (PK) schemes proposed for wireless sensor networks (WSNs), Elliptic Curve Cryptography (ECC) based algorithms have a proven and acceptable performance for low-powered sensor nodes [1]. However, the use of certificates in such a scheme consumes a huge amount of bandwidth and power. Considering the constrained resources of sensors, here we propose an efficient PK-based security scheme for WSN. Our analysis and simulation results show that our scheme demonstrates good performance for the current generation sensor nodes.

2 Network Assumptions and Preliminaries

The BS is a trusted entity and cannot be compromised in any way. The sensors deployed in the network have resources like modern era sensors (e.g., MICA2 motes [2]). Once the sensors are deployed over the target area, they remain relatively static.

The pseudoinverse matrix or generalised inverse matrix [3] has a very nice property that could be used for cryptographic operations. It is well known that, a nonsingular matrix over any field has a unique inverse. For a general matrix of dimension $k \times n$, there might exist more than one generalized inverse. This is denoted by, $M(k,n) = \{A: A \text{ is a } k \times n \text{ matrix}\}$. Let, $A \in M(k,n)$. If there exists a matrix $B \in M(n,k)$ such that, $ABA = A$ and $BAB = B$, then each of A and B is called a generalized inverse matrix (or pseudoinverse matrix) of the other.

* This paper was supported by ITRC and MIC.

S. Ata and C.S. Hong (Eds.): APNOMS 2007, LNCS 4773, pp. 503–506, 2007.
© Springer-Verlag Berlin Heidelberg 2007

3 Our Proposed Scheme

The first part of our scheme is the key handshaking process and the second part is used for confidential and authenticated data transmissions between two nodes.

Key Handshaking between any Node and Base Station: Let n_i be a node in the network and S be the base station (BS). To derive a shared secret key between the node n_i and the BS, the following operations are performed:

1. Node n_i randomly generates a matrix X with dimension $m \times n$ and its psedoinverse matrix, X_g. These matrices are kept secret in the node.

2. n_i calculates $X_g X$ and sends it to the base station S.

3. In turn, S randomly generates another matrix Y (dimension $n \times k$), and finds out its pseudoinverse matrix Y_g. These matrices are also kept secret in the BS.

4. S calculates $X_g XY$ and $X_g XYY_g$. Then it sends the resultant to n_i.

5. Upon receiving the products of matrices from S, n_i calculates, $XX_g XYY_g = XYY_g$ and sends it back to the base station.

6. Now, both n_i and base station S can compute the common secret key. n_i gets it by calculating $X(X_g XY) = XY$ and the base station gets it by calculating $(XYY_g)Y = XY$. Both of these outcomes are the same matrix (dimension $m \times k$).

Our mechanism ensures that, the individually calculated keys are same and this common key is used for encrypting the messages in the network. The derived common key could be used for node to BS or BS to node secure communications.

Encryption and Decryption of Data for Node-to-Node Communications. The main module in secure node to node communications is a central key generator (CKG) which is located at the base station. The CKG helps any node in the network to decrypt the received encrypted messages from other nodes. If a node n_i wants to send message securely to another node n_j, it uses the key that it has derived using the key handshaking process. Say for example, the encrypted message sent from n_i to n_j is $E_{XY}(M)$. Here, *M* is the message sent from the sender to the receiver.

E_{XY} means the message is encrypted with the key XY which is actually the shared secret key between the base station and the sender n_i. Upon receiving the encrypted message, n_j places its own identity and the identity of the sender to the CKG. In turn, CKG generates a decryption key and transmits it to n_j encrypting it with the secret shared key that it has with n_j. As the CKG in the base station has

prior knowledge about the shared secret keys of both the nodes, it uses that knowledge to generate the decryption key. Now, n_j first decrypts the encrypted message with its shared key, finds out the decryption key, and uses that key to decrypt the message sent from node n_i.

4 Performance Evaluation and Conclusions

To analyze the performance of our security scheme, we considered the specifications of MICA2dot [2] mote platform. In the key handshaking process, we have used linear matrix operations, more specifically matrix multiplication. The complexity of matrix multiplication is very low; hence it could be performed very quickly. In our shared secret key derivation scheme, total number of bits passed is, $n^2 + n(n+k) + mn = n(2n+k+m)$ bits. All the calculations here are linear and can be performed very easily. In the first part, for key handshaking we use public channel for message transmissions. However, capturing the messages like $X_g X$, $X_g XY$, $X_g XYY_g$ and XYY_g could not be helpful to construct the locally computed secret shared key XY. A potential attack could arise in the key handshaking process between a node and the BS, if there exists any sort of identification problem of the participating entities during communications. But, this threat is completely eliminated in our case because; (a) the base station is a trusted entity and could not be compromised in any way and (b) the ids of the communicating nodes are checked by the BS before further communications. In the second part, when the receiver node requests for the corresponding decryption key, the key is not sent as a plain message, instead it is encrypted with the shared secret key of the receiver. So, in no way, any adversary can get the decryption keys for a particular sender-receiver pair.

We compared our shared key derivation scheme with Diffie-Hellman's scheme [4]. We found that, to achieve a security level (complexity) of $2^{49.3}$ in D-H scheme, a key size of 200 bits is needed. On the other hand, to achieve almost the same level security (1/probability) of 2^{48}, 48 bit key is required in our scheme. Likewise, to get security of $2^{59.3}$, $2^{67.4}$, and $2^{74.4}$ in D-H scheme, 300, 400, and 500 bit keys are required respectively. On the other hand, to get security of 2^{60}, 2^{70}, and 2^{75} in our approach, 75, 84, and 105 bit keys are required respectively. In this analysis, the sizes of p in bits for D-H scheme are 200, 300, 400, and 500 respectively. For our approach, (m,n,k) are (4,8,12), (5,9,15), (6,11,14), and (7,12,15) correspondingly. Figure 1(a) shows the level of security to be achieved with required size of the keys in our approach and in D-H scheme. Here in all of these cases, our approach needs keys with much less size than that of D-H scheme. In the figure, along the x-axis we have shown the values of the exponent of 2 to plot the security level for various key sizes. That is, in the figure, a value say, 48 along the x-axis indicates 2^{48}.

The amount of traffic carrying the key related information is also dependent on the size of key. Figure 1(b) plots the key sizes versus the amount of traffic (considering only key related information) needed to pass through the open public channel in our

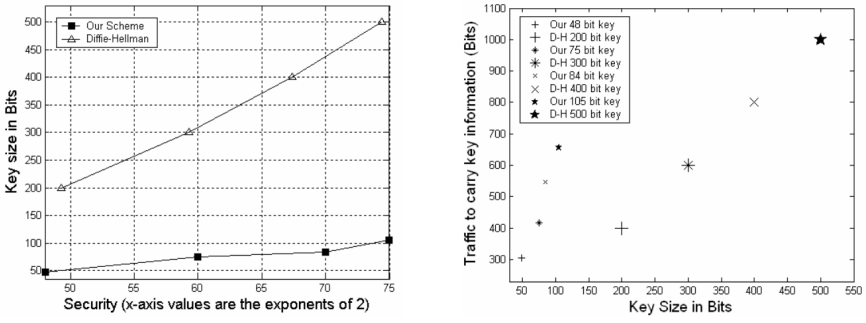

Fig. 1. (a) Required sizes of the key to provide almost same level of security in our key handshaking scheme and D-H scheme **(b)** Key size versus traffic to carry key information

entire scheme and Diffie-Hellman scheme. The data plotted here are based on the fact that, same (or almost same) level of security is to be ensured for both of the schemes.

MICA2dot motes [2] are equipped with 8-bit ATmega128L microcontrollers with 4 MHz clock speed, 128 kB program memory and Chipcon CC1000 low-power wireless transceiver with 433-916 MHz frequency band. According to our calculations, the cost of transmission of one byte is 59.2 μJ while the reception operation takes about half of the transmission cost (28.6 μJ). The power to transmit 1 bit is equivalent to roughly 2090 clock cycles of execution of the microcontroller. We considered a packet size of 41 bytes (payload of 32 bytes, header 9 bytes). With an 8 byte preamble (source and destination address, packet length, packet ID, CRC and a control byte) for each packet we found that, to transmit one packet $49\times 59.2 = 2.9008$ mJ ≈ 2.9 mJ energy is required. Accordingly, the energy cost for receiving the same packet is $49\times 28.6 = 1.4014$ mJ ≈ 1.4 mJ. Considering the same packet size for all the network operations, to set up a shared secret key with the base station each node needs (two transmissions and one reception) $((2\times 2.9)+1.4) = 7.2$ mJ of energy. For node to node communication, the sender needs one transmission (2.9 mJ) and the receiver needs two receptions and one transmission $(((2\times 1.4)+2.9) = 5.7$ mJ). As a whole, the entire scheme could be well-afforded by the energy resources of the current generation sensor nodes. Our scheme is also highly scalable, as any number of new sensors could be added to an existing wireless sensor network whenever needed.

References

1. Malan, D.J., Welsh, M., Smith, M.D.: A Public-Key Infrastructure for Key Distribution In TinyOS Based on Elliptic Curve Cryptography. In: Proc. IEEE SECON, Santa Clara, California, pp. 71–80 (2004)
2. Xbow Sensor Networks, Available at: http://www.xbow.com/
3. Boullion, T.L., Odell, P.L.: Generalized inverse matrices. Wiley-Int. New York (1971)
4. Rhee, M.Y.: Internet Security: Cryptographic Principles, Algorithms and Protocols. Wiley, Chichester (2003)