A novel architecture for localized key management in wireless sensor networks

Syed Muhammad Khaliq-ur-Rahman Raazi[1], Sungyoung Lee[1], Young Jae Song[2], Young Koo Lee[1]*
[1] Ubiquitous Computing Lab, Department of Computer Engineering,
Kyung Hee University, 449-701 Suwon, South Korea
{raazi,sylee,yklee}@oslab.khu.ac.kr
[2] Software Engineering Lab, Department of Computer Engineering,
Kyung Hee University, 449-701 Suwon, South Korea
yjsong@khu.ac.kr

Abstract

Wireless sensor networks (WSN) can be used in military surveillance, in which highly confidential data needs to be transmitted. In effect, security becomes a very important aspect in such networks. We present an efficient key management scheme for WSN. Our scheme is an improvement over SHELL [1] and mostly relies on communication within a cluster of nodes†.

1. Relevant Schemes

Our scheme is based on EBS [2], which is a very scalable scheme. To support a set of 'N' nodes, a set of "k+m" keys are required in EBS. Out of the total of "k+m" keys, each node knows a distrinct combination of 'k' keys. In order to evict a compromised node, new keys are distributed using 'm' keys that the node does not know.

We also use a concept of key-chains [3] in our scheme. In key-chains, current key is used to compute the previous one. Previous key can't be used to compute the current key. We use the last key to compute all other keys and store them in a node.

In SHELL [1], every node is authenticated by the command node initially. Then the gateways form their EBS matrices. Keys for each cluster are generated by more than one neighbouring cluster heads. For key distribution, each neighbouring gateway generates one message per individual administrative key in the cluster for each sensor node. All message exchanges are encrypted. For the distribution of communication keys, cluster head sends the communication keys to neighbouring cluster heads. Neighbouring cluster heads then distributes the communication keys as it distributes administrative keys. For addition of new nodes, same procedure is applied after the registration of the new node.

If the gateway is compromised, either it can be replaced or the nodes can join neighbouring clusters. If a sensor node is compromised, the 'k' keys known to it are revoked using the remaining 'm' keys in the EBS matrix.

2. Details of our scheme

We assume that the system has adequate capability of finding out the compromised nodes. Communication between gateways is not necessary in our scheme. Initially, command node authenticates all other nodes to their respective cluster heads.

2.1 Network Initialization

Gateways are deployed and then authenticated by the command node in the first phase. In the second phase, sensors are deployed in the field. Gateways authenticate the deployed sensors from cammand node and then register them. Apart from authentication, command node also indicates which nodes will be used for generating keys. Based on this, EBS matrix is formed both in the command node and the cluster head.

2.2 Initial key distribution

In our scheme, it is clear that the EBS matrix is predominantly calculated by the command node.In our scheme, represenetd of EBS matrix is updated in a manner such that the generator node of a key is represented by '2' as shown in table 1. A '1' in a cell depicts that the key is known to a node and a '0' depicts that the key is not known to a node.

Like shell, cluster head does not know the administrative keys of nodes inside the cluster. Key-generating nodes refresh the administrative keys and communicate them directly to the nodes, who are supposed to know the key. In order to distribute its communication key, cluster head sends it to the key-generating nodes. Key-gerating nodes encrypt them in their generated keys and send them to all the nodes.

	N_0	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9
K_1	2	1	1	1	1	1	0	0	0	0
K_2	1	2	1	0	0	0	1	1	1	0
K_3	1	0	0	2	1	0	1	1	0	1
K_4	0	1	0	1	0	2	1	0	1	1
K_5	0	0	1	0	2	1	0	1	1	1

Table 1: EBS matirx example in our scheme

† This research was supported by the MIC(Ministry of Information and Communication), Korea, Under the ITFSIP(IT Foreign Specialist Inviting Program) supervised by the IITA(Institute of Information Technology Advancement).

2.3 Node addition and re-keying

Re-keying is done periodically or on the request of the cluster head. Cluster head just needs to request the key-generating nodes for re-keying of the administrative keys. Key-generating nodes generate the new keys and change the previous ones with them. For re-keying of the communication key, cluster head just sends the new communication key to the key-generating nodes, which in turn send it to the rest of the nodes.

Key-generating nodes generate new keys with the help of one-way hashing functions. Command node knows the last key of the key-generating node, so it can generate all the keys. Cluster head knows how many times the key has been changed. Whenever new nodes have to be added, command node sends their IDs to the respective cluster heads. Cluster heads halts re-keying and tell the command node how many times a key has been changed. Command node then send the current keys to the new nodes through cluster heads. After authentication, current keys are communicated to the new nodes. Note that the cluster head does not come to know of any key in this process. EBS matrix is also communicated to the cluster head. In case of key-generating node, it will generate the subsequent values of the key.

2.4 Node Compromise

If the cluster head is compromised, we can replace it or redistribute its sensors to neighbouring clusters. Apart from these two options, we can also designate a neighbouring cluster head to take care of this cluster. This is possible as the compromise of cluster head does not reveal any administrative key in our scheme.

If a simple sensor node is compromised, keys that are known to it must be changed. Cluster head asks the respective key-generators to generate the new keys and encrypt them using previous values. Cluster head aggregates these encrypted keys into a new message and send it to the other key-generating nodes, which disseminate the new key to the other nodes, as it happens in the EBS scheme.

In case the key-generating node is compromised, cluster head ceases any re-keying from it and informs the command node the number of times its key has been revoked. Either a new key-generating node can be deployed or the responsibility can be given to an existing node. Apart from that, the cluster head can also take responsibility for generating the key. In any case, the node is provided the current value of the key, which it should generate. Other keys that are known to the compromised nodes are evicted in the same way as they are done in case of simple node compromise.

3. Comparison with SHELL

In our scheme, memory and computation requirement for cluster heads is substantially reduced as compared to SHELL. However, our scheme puts some minimal storage and processing overhead on a very small number of sensor nodes, which have key-generating capability. Computation requirements are reduced on other sensor nodes also. Same is the case with communication overhead. We put a little extra burden on few key-generating nodes and save cluster heads from the long-range inter-cluster communication overheads.

References

[1]. M. Younis, K. Ghumman, and M. Eltoweissy, “Location aware Combinatorial Key Management Scheme for Clustered Sensor Networks,” IEEE Trans. Parallel and Distrib. Sys., vol. 17, No. 8, Aug. 2006.

[2]. M. Eltoweissy, H. Heydari, L. Morales, and H. Sadborough, “Combinatorial Optimization of Key Management in Group Communications,” J. Network and Systems Management, vol. 12, no. 1, pp. 33-50, Mar. 2004.

[3]. G. Dini, I.M. Savino, “An Efficient Key Revocation Protocol for Wireless Sensor Networks” International Workshop on Wireless Mobile Multimedia, Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks, pp. 450-452, 2006.