An efficient key revocation protocol for wireless sensor networks

Gianluca Dini and Ida Maria Savino
University of Pisa
Dipartimento di Ingegneria dell'Informazione
via Diotisalvi 2, 56100 PISA Italy
{g.dini, ida.savino}@iet.unipi.it

Abstract

In this paper, we present a scalable and secure protocol for key revocation in Wireless Sensor Networks. The protocol guarantees an authenticated distribution of new keys that is efficient in terms of storage, communication and computing overhead. The proposed protocol reduces the number and the size of rekeying messages. It achieves the necessary level of confidentiality and authenticity of rekeying messages by only using symmetric ciphers and one-way functions. Hence, the protocol results scalable, and particularly attractive for large and/or highly dynamic groups.

1 Introduction

Nowadays, small, low-cost sensor nodes are being widely used to build self-organizing, large-scale, wireless networks for various applications, such as environmental surveillance, health monitoring and so on [1].

The management of Wireless Sensor Networks (WSNs) is particularly complex due to the large number of nodes, the limited hardware capabilities of the nodes and the constant exposure of nodes to be compromised. In fact, in order to make WSNs economically viable, sensor nodes typically lack adequate support to tamper-resistance. In particular, sensor nodes can be deployed over open, unmonitored, possibly hostile areas and they are exposed to the risk of being captured by an active adversary.

In a WSN, sensor nodes cooperate towards a given application according to the *group communication* model [5, 9]. Usually, sensor nodes perform in-network processing by reducing large streams of raw data into useful aggregated information. Therefore, compromised nodes could deviate the network behaviour by injecting false data or modifying data of correct nodes. Thus, it must be guaranteed that compromised nodes do not take part in the group communication.

For this reason we assume that group members share a secret symmetric key, called the *group-key*, they use to authenticate messages they broadcast within the group. Upon detecting a compromised sensor node, the current group-key must be revoked and a new one must be distributed to all group members except the compromised one. It follows that, by revoking all keys of a compromised node, it is possible to remove the presence of that node from the group.

This paper presents a key revocation protocol for large, highly dynamic WSNs that follow the group communication paradigm. The protocol is efficient and scalable as it limits both communication and computation overhead during rekeying. The proposed protocol follows a centralized approach where a *Key Service* (*KS*) forces the compromised nodes to leave the group. *KS* revokes the current group-key and redistributes a new one to all sensor nodes except the compromised one. The centralized schemes proposed so far [6, 3, 5] require a number of rekeying messages that grows linearly with the network size, i.e., $\mathcal{O}(n)$ messages. In contrast, the protocol we propose accomplishes the same task with $\mathcal{O}(\log n)$ messages so scaling much better in the case of large WSNs.

The proposed protocol levers on *Logical Key Hierarchy* (LKH), a technique for secure and scalable group rekeying [11, 10]. LKH allows us to achieve a rekeying protocol which requires a number of messages that is a logarithmic function of the network size. Relevant applications of LKH in large-scale distributed systems can be found in [11, 7, 10]. However, these systems use digital signatures based on public key, e.g., RSA [8], to ensure the authenticity of new keys. In contrast, we achieve such a proof of authenticity by means of *key-chains*, an authentication mechanism based on the Lamport's one-time passwords [4]. This lightweight mechanism uses only hash functions that are several orders of magnitude more efficient than public-key cryptography.

The paper is organized as follows. Section 2 illustrates the key-chain mechanism for key authentication. Section 3 details the key revocation protocol and, finally, in Section 4

Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)
0-7695-2593-8/06 $20.00 © 2006 IEEE

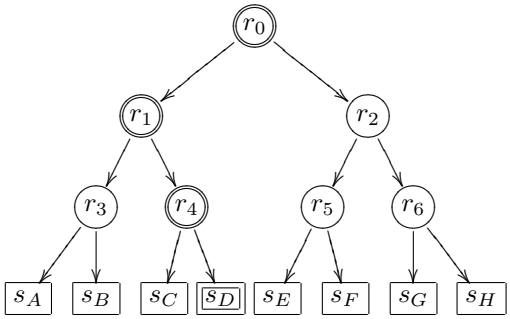

Figure 1. Eviction-tree with m=2 and h=3. Double-circles represent the internal nodes belonging to *Path*(s_D).

we expose our concluding remarks.

2 Key authentication

We consider a sender S that broadcasts symmetric keys and a receiver R that must be able to verify the authenticity of received keys, i.e., they come from S.

In the setup phase, S constructs a *key-chain*, a sequence of values $< K_0; \ldots ; K_i; \ldots ; K_N >$, where each key is linked to other ones by means of a one-way hash function H [2]. We refer to K_0 as the *start-key*, and to K_N as the *end-key*. S randomly chooses the end-key K_N and then iteratively applies the one-way function H for N times in order to obtain the start-key at the beginning of the chain. That is, $K_i = H(K_{i+1}), 0 \leq i < N$.

Each key in the chain (except the start-key) is the hash preimage of the previous one in the direction of the start-key. That is, the key K_i is the hash preimage of K_{i-1}. Therefore, previous keys are not useful in computing the next ones, which instead can be easily obtained from the keys towards the end of the chain. In fact, K_i can be computed by applying the hash function H to the next keys K_j, $j > i$.

S uses the key-chain for authentication purposes as follows. S reveals the keys of the chain starting with the start-key K_0, and ending with the end-key K_N. Assume that S reveals the i-th key of chain K_i to R through an authenticated channel. Then, assume that, later R receives the j-th, $j > i$, key over an insecure channel. R can verify the authenticity of K_j by applying $j - i$ times H to K_j and checking that the result is equal to the K_i. That is, $K_i = H^{j-i}(K_j)$.

We define the *current-key*, K_{cur}, as the last-revealed key that belongs to the chain. That is, K_{cur} corresponds to K_i if S has already revealed K_i but K_{i+1} is still secret. Furthermore, the key that is adjacent to K_{cur} in the direction of the end of the chain is referred as the *next-key*, K_{nxt}.

3 Lightweight key revocation

We assume that each sensor node has a *private-key*, a secret symmetric key that the node shares with *KS*. We denote by $K^p_{s_X}$ the key of sensor node s_X.

In order to remove a compromised node s_L, *KS* maintains a hierarchical structure of symmetric keys, called the *revocation-tree* (Figure 1). In the following we will use the term node to refer a tree node. However, whenever ambiguity could arise, we will use the term group member to refer a sensor node.

KS associates each internal node r_j with a chain composed of N_{r_j} keys, and each leaf with the private-key of a group member. We define *CurrentKeys* the set of the current-keys associated with internal nodes of the revocation-tree. Initially, the current-key of the internal node r_j, $K^{r_j}_{cur}$, corresponds to the start-key, $K^{r_j}_0$. Thus, in the setup phase *CurrentKeys* contains only the start-keys.

A sensor node s_X stores a subset of *CurrentKeys*, *KeyRing*(s_X), defined as follows. Let *Path*(s_X) be the set of internal nodes lying on the path from the root to the leaf associated with s_X. The key-set *KeyRing*(s_X) is composed of the current-keys associated with the internal nodes in *Path*(s_X):

$$\textit{KeyRing}(s_X) = \{K^{r_j}_{cur} \mid r_j \in \textit{Path}(s_X)\}$$

At any time, the key-set in each group member is composed of h keys, where h is the path length. In the case of a balanced tree $h = \log_m(n)$, where n is the network size and m is the tree ariety.

Every group member stores a copy of the current-key associated with the tree root, $K^{r_0}_{cur}$, because it belongs to the key-set of every group member. Thus, this key acts as the group-key.

When a sensor node s_L is compromised, or it is suspected so, all keys belonging to *KeyRing*(s_L) become compromised and *KS* has to renew them. These keys must be securely distributed to all group members, except s_L, as follows. For each internal node r_j in *Path*(s_L), and for each child of r_j, *KS* broadcasts a rekeying message structured as follows. Every message contains the next-key of r_j, $K^{r_j}_{nxt}$. If the r_j's child is a leaf and it is not associated with $K^p_{s_L}$, $K^{r_j}_{nxt}$ is encrypted with the private-key of the corresponding group member. If the r_j's child is an internal node (i.e., r_c) and is not included in *Path*(s_L), $K^{r_j}_{nxt}$ is encrypted with $K^{r_c}_{cur}$. If r_c is included in *Path*(s_L), $K^{r_j}_{nxt}$ is encrypted with $K^{r_c}_{nxt}$. After this procedure, *CurrentKeys* is updated as follows: for each internal node r_j belonging to *Path*(s_L), $K^{r_j}_{cur}$ is replaced by $K^{r_j}_{nxt}$. Then, *KS* removes the leaf associated with $K^p_{s_L}$ from the revocation-tree.

With reference to Figure 1, let us assume that the sensor node s_D is compromised. It follows that keys $K^{r_0}_{cur}$, $K^{r_1}_{cur}$,

Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)
0-7695-2593-8/06 $20.00 © 2006 IEEE

and $K_{cur}^{r_4}$ are considered compromised. Thus, *KS* generates and broadcasts the following rekeying messages:

$$m_1 : KS \rightarrow s_C : E_{K_{s_C}^p}(K_{nxt}^{r_4})$$
$$m_2 : KS \rightarrow \{s_A, s_B\} : E_{K_{cur}^{r_3}}(K_{nxt}^{r_1})$$
$$m_3 : KS \rightarrow s_C : E_{K_{nxt}^{r_4}}(K_{nxt}^{r_1})$$
$$m_4 : KS \rightarrow \{s_A, s_B, s_C\} : E_{K_{nxt}^{r_1}}(K_{nxt}^{r_0})$$
$$m_5 : KS \rightarrow \{s_E, s_F, s_G, s_H\} : E_{K_{cur}^{r_2}}(K_{nxt}^{r_0})$$

where $E_K(m)$ is the encryption of message m with the key K. As the private-key held by s_D is not used to encrypt any new key, and all keys in *KeyRing*(s_D) are changed, s_D is no longer able to access the group messages.

Upon receiving the re-keying messages, the group members decrypt them with appropriate keys in order to get the new keys. Then, the group members can immediately verify the authenticity of the new keys by applying H. With reference to Figure 1, let us consider s_B, for example. Upon receiving message m_2, s_B decrypts it with $K_{cur}^{r_3}$ and checks that $K_{cur}^{r_1} = H(K_{nxt}^{r_1})$. Then, s_B decrypts m_4 with $K_{nxt}^{r_1}$ and checks that $K_{cur}^{r_0} = H(K_{nxt}^{r_0})$.

The proposed protocol is efficient in terms of storage and computing overhead. Each sensor node s_X stores only the private-key and h keys belonging to *KeyRing*(s_X). Furthermore, in order to verify the authenticity of the received keys each sensor node performs only an hash function.

Finally, our proposed protocol improves on scalability by reducing both the size and the number of rekeying messages. As to the size of the rekeying messages, our protocol reduces it because *KS* broadcasts only the encrypted key without appending a digital signature for authenticity. As to the number of rekeying messages, it is a logarithmic function of the network size n. In the case of a balanced m-ary tree, *KS* needs to broadcast $m \log_m(n) - 1$ messages. For each internal node r_j on the path of compromised node, *KS* broadcasts m rekeying messages, one for each r_j's child. The exception is made for the internal node having the leaves as children. In this case, *KS* has to transmit $m - 1$ unicast messages, one for each leaf except the one associated with the compromised node. Hence, our protocol improves scalability with respect to the simple solution adopted by μTESLA, for example, where the number of rekeying messages is a linear function of n.

4 Conclusions

With reference to group communication in WSNs, we have presented a key revocation protocol that aims at removing compromised nodes from the group communication. The protocol reduces communication and computational overhead so improving the protocol scalability and increasing the network lifetime.

Acknowledgements

This work has been supported by the Commission of the European Communities under Sixth Framework Programme Project IST-004536 "Reconfigurable Ubiquitous Networked Embedded Systems"(RUNES).

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):293–422, March 2002.

[2] Alfred J. Menezes and Paul C. van Oorschot and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, October 1996.

[3] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *10th ACM Conference on Computer and Communication Security (CCS'03)*, pages 41–57, Washington D.C., USA, October 27–30 2003.

[4] L. Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–772, November 1981.

[5] T. Park and K. G. Shin. LiSP: A Lightweight Security Protocol for Wireless Sensor Networks. *ACM Transactions on Embedded Computing Systems*, 3(3):634–660., August 2004.

[6] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security suite for sensor networks. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)*, pages 189–199, New York, July 16–21 2001. ACM Press.

[7] S. Rafaeli and D. Hutchison. A Survey of Key Management for Secure Group Communication. *ACM Computing Surveys*, 35(3):309–329, September 2003.

[8] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Comm. of the ACM*, 21(2):120, Feb. 1978.

[9] B. Sinopoli, C. Sharp, L. S. S. Schaffert, and S. S. Sastry. Distributed Control Applications Within Sensor Networks. *Proceedings of the IEEE*, 91(8):1235–1246, August 2003.

[10] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner. The VersaKey Framework: Versatile Group Key Management. *IEEE Journal on Selected Areas of Communications (Special Issue on Middleware)*, 17(9):1614–1631, August 1999.

[11] C. K. Wong, M. G. Gouda, and S. S. Lam. Secure Group Communications using Key Graphs. *IEEE/ACM Transactions on Networking*, 8(1):16–30, February 2000.