A key management system with a storage-saving secret sharing scheme in a sensor network

Yoshihide Nagai, Hyunho Kang and Keiichi Iwamura
Department of Electrical Engineering, Tokyo University of Science.
6-3-1 Niijuku, Katsushika-ku, Tokyo 125-8585, Japan.
nagai@sec.ee.kagu.tus.ac.jp, {kang, iwamura}@ee.kagu.tus.ac.jp

ABSTRACT

Generally, sensor networks are used in cryptography. However, it takes a long time because of many processes. We proposed a storage-saving secret sharing scheme to solve this problem. However, we did not show an application and estimation of the process. Therefore, we apply to this scheme to a sensor network. As a result, we show that the proposed scheme is suitable for a sensor network.

KEYWORDS

Secret sharing, Sensor network, Key management

1 INTRODUCTION

Generally, sensor networks are used in cryptography. However, it takes a long time because of many processes. In many sensor nodes, the encryption key uses a large amount of storage. We demonstrate the features of sensor networks. The advantages of sensor networks we are low cost and low energy. The disadvantages are low processing and low storage. Therefore, it needs a secure, fast, and storage-saving scheme.

Kurihara presented a secret sharing scheme using XOR[1]. This implemented high-speed processing and is an ideal secret sharing scheme because it does not leak information from the share, and the recovery and share have the same data length.

Kurihara also developed the RAMP secret sharing scheme using XOR[2]. However, this scheme is not suitable for secure situations because it may leak partial secrets.

Therefore, we proposed a new secret sharing scheme using XOR [3]. This scheme achieves high-speed processing and reduces the data size using pseudo-random numbers.

However, we did not demonstrate an application of this scheme. In this paper, we show a concrete application as an example.

2 SENSOR NETWORK

2.1 Sensor network

A sensor network is constructed by sensor nodes. A sensor node has three types (base station, router, end device). We show the features of each node.

- Base station

  The base station has the functions of a router and managing a hole network.

- Router

  The router has the functions of sensing and relaying data.

- End device

  The end device only has sensing functionality.

2.2 Connecting topology

A sensor network has three types of connecting topology (mesh, star, and cluster tree). A sensor node determines the topology according to the connection conditions and uses.

- Mesh topology
- Star topology
- Cluster-tree topology

ISBN: 978-0-9891305-3-0

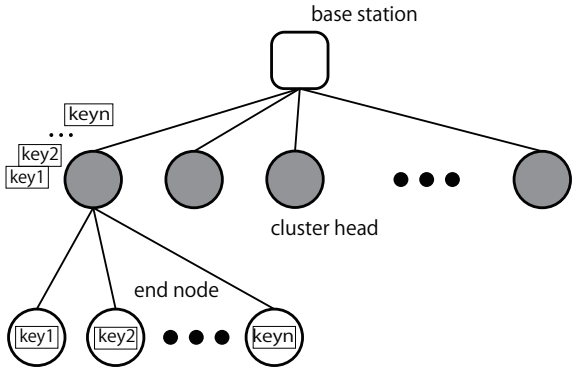

Figure 1: Key management on a sensor network

2.3 Problem

We show the security of a sensor network. Generally, a sensor network is used in cryptography such as AES [4]. However, this is complex for sensor networks [5] owing to low processing ability. Moreover, AES requires symmetric keys. Figure 1 shows key management on a sensor network.

The cluster head has many symmetric keys $key1, 2, \ldots, n$. When the cluster head has many keys, the cluster head has low processing ability. When a large number of end nodes exist, the cluster head may not be able to store the keys.

3 CONVENTIONAL KEY MANAGEMENT SCHEME [6]

In this section, we demonstrate a key management scheme for a sensor network as an example.

3.1 Key management scheme adapting any topology

After the nodes are set with six elements, the nodes share keys, adapting each topology.

- a unique key: k_u
- a global key: k_g
- a random key: k_r
- a ticket: T
- a list of unique keys (only the base station has the list)
- the elapse time of an initial key: t_s

A unique key is the specific key of each node, which is not redundant. A global key is a key for all nodes. A random key is each key from some key pool. Moreover, we assign a key ID to each random key. Therefore, the nodes can distinguish the random keys from each other to exchange key IDs. A ticket is as follows:

$$T = k_{uc}(k_u||t_D||K||r).$$

In this paper, we show key sharing for a cluster-tree topology.

For a disconnected base station, each node shares a key with each around node using a global key and random keys. After the nodes exchange random key IDs with each other, they distinguish the common keys. The nodes connect these common keys and create a link key using a one-way hash function and the connected common keys. All nodes can perform key sharing; therefore, all nodes have a global key.

$$K = h(k_{r1}|| \ldots ||k_{ri}||k_G)$$

$$K : link\_key, h : one\_way\_hash\_function$$

$$k_r : random\_key, k_G : global\_key$$

The router sends a ticket to a base station. A base station decrypts the ticket, checks a unique key against a list of unique keys, and verifies the elapsed deadline of a ticket. When a ticket is correct, a base station stores the unique key of the router as a link key. In the other case, the base station cancels communication.

4 THE STORAGE-SAVING SCHEME[3]

In this chapter, we explain the storage-saving scheme[3]. These techniques implement high-speed recovery and shared processing.

4.1 Distribution algorithm

The proposed scheme can reduce the data size for producing each of the multiple shares.

D picks up t-person from n-person. D defines user ID (i_T) of t-person ($1 \leq T \leq t$).

1. Dealer D gives user i_T a random number r_{i_T} and key_{i_T}.

2. D creates a pseudo-random number $q_{(m,i_T)}$ using the random number r_{i_T} and key_{i_T} ($q_{(m,i_T)} = M(n-1)d$ [bit]).

3. D divides $q_{(m,i_T)}$ into $M(n-1)$ pieces and divides the secret S_m into $n-1$ pieces. D obtains $q_{(m,i_T,j)}, S_{(m,i)}$, and $S_{(m,0)}$ $(0 \leq j \leq n-2)$.

$$q_{(m,i_T)} = q_{(m,i_T,0)} \| q_{(m,i_T,1)} \| \cdots \| q_{(m,i_T,n-2)}$$

$$S_m = S_{(m,1)} \| S_{(m,2)} \| \cdots \| S_{(m,n-1)}$$

$$S_{(m,0)} \in \{0\}^d$$

4. D calculates the coefficient a of the $M(n-1)$ pieces.

$$q_{(m,0,0)}, \cdots, q_{(m,0,n-2)}, q_{(m,1,1)}, \cdots, q_{(m,t,n-2)}$$

$$q_{(m,i_T,j)} = S_{(m,i_T-j)} \bigoplus \left\{ \bigoplus_{h=0}^{k-2} a^h_{(m,h\cdot i_T+j)} \right\}$$

$$(1 \leq T \leq t, 0 \leq j \leq n-2)$$

5. D creates a random number r of $M\{n(k-1) - t(n-1) - 1\}$ pieces. D creates the partial shares $W_{(m,i,j)}$ in $0 \leq i \leq n-1, 0 \leq j \leq n-2$ without $i = i_T$.

$$W_{(m,i,j)} = S_{(m,i-j)} \bigoplus \left\{ \bigoplus_{h=0}^{k-2} (r \cup a)^h_{(m,h\cdot i+j)} \right\}$$

$$(0 \leq i \leq n-1, 0 \leq j \leq n-2)$$

6. D creates the shares $W_{(m,i)}$ from the partial shares $W_{(m,i,0)}, \cdots, W_{(m,i,n-2)}$. D distributes these shares to all users except P_{i_T}.

$$W_{(m,i)} = W_{(m,i,0)} \| W_{(m,i,1)} \| \cdots \| W_{(m,i,n-2)}$$

$$(W_{(m,i)} \in \{0,1\}^{(n-1)d}).$$

4.2 Recovery algorithm

User A wishes to recover one secret. In this case, each of the other users give user A a share. User A can then recover the secret.

Multiple users that reduce the data size

$$P_{i_0}, P_{i_1}, \cdots, P_{i_{t-1}}$$

form the pseudo-random number

$$q_{(m,i_0)}, q_{(m,i_1)}, \cdots, q_{(m,i_{t-1})}$$

$$(q_{(m,i)} = W_{(m,i)})$$

from the random number

$$r_{i_0}, \cdots, r_{i_{t-1}}$$

, and the key that D distributes to multiple users is

$$key\ i_0, key\ i_1, \cdots, key\ i_{t-1}.$$

Then, multiple users give user A the pseudo-random number relative to the secret.

The other users who do not reduce the data size

$$P_{i_t}, P_{i_{t+1}}, \cdots, P_{i_k}$$

give user A a share relative to the secret

$$W_{(m,i_t)}, W_{(m,i_{t+1})}, \cdots, W_{(m,i_k)}.$$

(User A only has shares relative to the secret.)

User A obtains the shares of k pieces $W_{t_0}, \cdots, W_{t_{k-1}}$ $(0 \leq t_0 \leq \cdots \leq t_k \leq n-1)$.

1. Separate a share of k pieces into partial shares:

$$W_{t_0} \leftarrow W_{(t_0,0)}, W_{(t_0,1)}, \cdots, W_{(t_0,n-2)}$$

$$\vdots$$

$$W_{t_{k-1}} \leftarrow W_{(t_{k-1},0)}, W_{(t_{k-1},1)}, \cdots, W_{(t_{k-1},n-2)}.$$

2. Create the vector $V_{(t_i,j)}$ using the partial shares.
For partial share $W_{(t_i,j)}$,

$$W_{(t_i,j)} = V_{(t_i,j)} \cdot R_{(k,n)}$$

$$R_{(k,n)} = (S_1, \cdots, S_{n-1}, r^0_0, \cdots,$$

$$r^0_{n-2}, r^1_0, \cdots, r^1_n - 1, \cdots, r^{k-2}_0, \cdots, r^{k-2}_{n-1})^T.$$

3. Create the matrix $M^{(k,n)}_{(t_0,\ldots,t_{k-1})}$ from the vectors:

$$M^{(k,n)}_{(t_0,\cdots,t_{k-1})} = (V_{(t_0,0)}, \cdots, V_{(t_0,n-2)}, \cdots,$$

$$V_{(t_{k-1},0)}, \cdots, V_{(t_{k-1},n-2)})^T.$$

4. Show vector $W_{(t_0,\cdots,t_{k-1})}$ using partial shares:

$$W_{(t_0,\cdots,t_{k-1})} = (W_{(t_0,0)},\cdots,$$
$$W_{(t_0,n-2)},\cdots,W_{(t_{k-1},0)},\cdots,W_{(t_{k-1},n-2)})^T$$
$$W_{(t_0,\cdots,t_{k-1})} = M^{(k,n)}_{(t_0,\cdots,t_{k-1})} \cdot R_{(k,n)}.$$

5. The user calculates the $k(n-1)$-vector $S_{(k,n)}$ of all partial shares through the user transform $M^{(k,n)}_{(t_0,\cdots,t_{k-1})}$ into $G_{(k,n)} = G(M^{(k,n)}_{(t_0,\cdots,t_{k-1})})$ using Gauss–Jordan elimination. The calculation process of Gauss–Jordan elimination is $GF(2)$.

   $G_{(k,n)}$ is represented by

$$G_{(k,n)} = \left\{ \begin{array}{cc} I & \Phi \\ \Phi & \Delta k \end{array} \right\}$$

   where I is the $(n-1)\times(n-1)$ identiy matrix, and Φ is the zero matrix,

$$\Delta k = \left(\begin{array}{ccccc} I & \Phi\|c_1 & \Phi\|c_1 & \cdots & \Phi\|c_1 \\ \Phi & I\|c_1 & \Phi & \cdots & \Phi \\ \Phi & \Phi & I\|c_1 & \cdots & \Phi \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Phi & \Phi & \Phi & \cdots & I\|c_1 \end{array} \right)$$

   where c_1 is $(n-1)$vector consisting of $(1,\cdots,1)^T$.

   The user obtains an equation from equation (1) in Step 4 using Gauss–Jordan elimination

$$S_{(k,n)} = G_{(k,n)} \cdot R_{(k,n)}$$
$$S_{(k,n)} = (S_1, S_2, \cdots, S_{n-1}, *, \cdots, *)^T$$

   ($*$ is the other information).
   The user then has all of the partial secrets.

6. Recover secret S from all partial secrets:

$$S = S_1\|S_2\|\ldots\|S_{n-1}.$$

5 APPLICATION

In this chapter, we demonstrate an application of the proposed scheme.

We define a sensor network constructed by a base station, cluster heads, and end nodes. The network topology is a cluster tree. The base station has enough processing ability and storage capacity. The others have low processing ability and storage capacity. The cluster head is selected from the others.

When the cluster head communicates with the end nodes, the cluster head uses a different key for each node. In this case, the cluster head must know all keys in own cluster. If we assume a large sensor network, it is not efficient for the cluster head to have all keys in storage. In this case, our scheme offers a good solution for key sharing in a sensor network.

We introduce a concrete example. In this example, we assume that the base station knows all the secret keys for all the end nodes, and each end node knows its own secret key.

One protocol chooses the cluster heads. Here, the cluster head does not know each secret key of the end nodes, but the cluster head knows each node ID. The number of end nodes in a cluster is n. The following process is performed for the cluster heads to obtain the secret key of each end node.

1. Each cluster heads sends all node IDs $ID_i(1 \le i \le n)$ belonging its own cluster to the base station.

2. The base station determines an initial value and a secret key and sends these to each cluster head by encrypted communication using the cluster head's secret key. Each cluster head decodes it using its own secret key and stores them.

3. The base station generates a pseudo-random number q_i to use as an initial value and a secret key. The base station calculates α_i according to the equation:

$$q_i = W_{(i,0)} = S_{(i,0)} \bigoplus \alpha_i \ (S_{(i,0)} \in \{0\}^d).$$

4. The base station creates $W_{(i,1)}$, setting $S_{(i,1)}$ as the key of each end node in our scheme. $W_{(i,1)}$ is the mean of the following equation:

$$W_{(i,1)} = S_{(i,1)} \bigoplus \alpha_i.$$

   The base station sends $W_{(i,1)}$ to each cluster head.

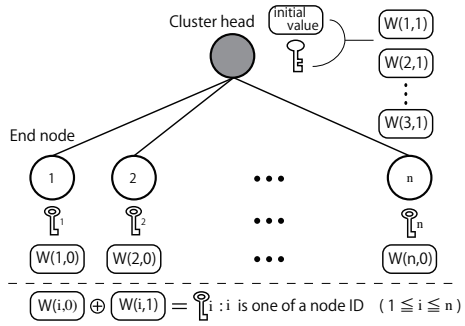

Figure 2: The proposed scheme on a sensor network

5. Each cluster heads sends $W_{(i,1)}$ to the end nodes of the node ID ID_i. Here, each cluster head does not store $W_{(i,1)}$. Each end node of node ID ID_i stores $W_{(i,1)}$.

In key exchange, each end node of node ID ID_i sends $W_{(i,1)}$ to the cluster head. The cluster head generates q_i using the initial value and secret key. The cluster head creates $S_{(i,1)}$ using the shares by means of the following equation:

$$S_i = q_i \bigoplus W_{(i,1)}.$$

Therefore, a cluster head shares secrets among each node. The secrets to use the secret keys achieve encryption. The cluster head does not have the secret keys of n nodes. Moreover, a secret key is not leaked directly when a cluster head is stolen.

6 ESTIMATION OF THE STORAGE CAPACITY

We show the amount of data stored on a sensor node. In a conventional key management scheme, a router has k_u, k_G, $p * k_r$, and T (p denotes the number of a key pool). Therefore, the router has $(p+3)$ keys. An end device has same number of keys. However, a router has an initial key and initial vector in the storage-saving scheme. An end device has a unique key of each end device and a share of the unique key. Table 1 summarizes this situation.

When p of the conventional scheme is increased to gain security, the storage-saving scheme obtains a storage-saving effect.

Table 1: Storage capacity

	SCIS2010 [6]	GCCE2013 [3]
Router	$(p+3) \times key$	$initial\_key$ $+initial\_vector$
End device	$(p+3) \times key$	$key + share$

7 CONCLUSION

In this paper, we have discussed the problems of a sensor network. As demonstrated in the application, the storage-saving scheme can solve these problem. In our future work, we discuss proof of security.

References

[1] Jun Kurihara, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. On a fast (k,n)-threshold secret sharing scheme. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E91-A(9):2365–2378, September 2008.

[2] Jun Kurihara, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. A new (k,n)-threshold secret sharing scheme and its extension. In *Proceedings of the 11th international conference on Information Security*, ISC '08, pages 455–470. Springer-Verlag, 2008.

[3] Yoshihide Nagai, Hyunho Kang, and Keiichi Iwamura. A storage-saving secret sharing scheme suitable for sensor networks. In *GCCE '13: Proceedings of the 2nd IEEE Global Conference on Consumer Electronics*, Oct 2013.

[4] ZigBee Alliance. Zigbee alliance. *WPAN industry group, http://www. zigbee. org/. The industry group responsible for the ZigBee standard and certification*, 2010.

[5] Kasumi Toriumi, Yoshio Kakizaki, and Keiichi Iwamura. Fast implementation of the advanced encryption standard using atmega1281. In *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, pages 342–346. IEEE, 2011.

[6] Yuta Ohami, Yoshiro Kakizaki, and Keichi Iwamura. A key management method adaptable to some topology on sensor network. *SCIS2010*, 3C2-5, 2010 (Japanese).