A Secure Scheme for Heterogeneous Sensor Networks

Sujun Li, Weiping Wang, Boqing Zhou, Jianxin Wang, Yun Cheng, and Jie Wu, *Fellow, IEEE*

***Abstract*—Heterogeneous sensor networks (HSNs) consist of a small number of powerful high-end sensors (H-sensors) and a large number of low-end sensors (L-sensors). HSNs are vulnerable to H-sensors replication attack. In this letter, a scheme against the attack is proposed. The analysis and simulation results indicate that the scheme can improve networks' resilience against H-sensors replication attack as compared with existing related schemes.**

***Index Terms*—Heterogeneous sensor networks, H-sensors replication attacks, master-slaver model, EQ method.**

I. Introduction

HETEROGENEOUS sensor networks (HSNs), which consist of a small number of H-sensors (e.g., PDAs) and a large number of L-sensors (e.g., the MICA2-DOT [1]), have attracted much attention due to their better performance and scalability compared with homogeneous sensor networks [2]. In HSNs, H-sensors are in charge of forwarding data to the Base station (BS). Therefore, they are vulnerable to suffer from various attacks; one of the most common attacks is replication attack. This attack once being launched successfully, HSNs will be subject to the following threats: 1. L-sensors will choose these replication nodes as cluster heads and submit their data to them; 2. These replication nodes can communicate with normal H-sensors in the networks, and can forge a great deal of false data. This false data is forwarded to the BS, which not only wastes power and bandwidth of H-sensors, but also lets the BS make wrong judgments.

To enhance the secrecy of HSNs, in the last few years, different pairwise key distribution schemes using symmetric key algorithms have been developed [3]–[5]. Du *et al.* [3] proposed AP-D scheme using asymmetric predistribution key (AP) method. In the scheme, before deployment, an L-sensor and an H-sensor randomly pick $t1$ and $t2$ ($t1 << t2$) keys from a large key pool without replacement, respectively. After deployment, two nodes can establish a pairwise key if the number of common keys between them is greater than or equal to q ($q \geq 1$). The AP method is also used in [4] and [5]. In AP-L [4], the key pool of L-sensors is a subset of H-sensors. In [5], the CSS-SH scheme enhances the resilience by exploiting two dimensional backward key chains constructing disjoint and association key pools. Nevertheless, in the schemes, H-sensors and L-sensors share the same key pool; and H-sensors are needed to participate in the establishment of shared keys between them. As a result, replication H-sensors can easily establish pairwise keys with normal nodes by using compromised keys. Therefore, in HSNs, new schemes against H-sensor replication attack must be developed.

In this letter, a secure scheme against H-sensors replication attack, namely SS-H, is proposed. Main contributions of our scheme are summarized as follows: 1. A new secure communication model, namely master-slaver model, is created, and is realized using new two-dimension backward key chains; 2. A new method, namely *EQ*, for establishing pairwise key between an L-sensor and an H-sensor is presented.

The letter is organized as follows. Section II presents our protocol and Section III analyzes the protocol. The conclusion is given in Section IV.

II. SS-H Scheme

In HSNs, H-sensors serve as cluster heads and form clusters around them [3]. The formation of clusters is as follows: Each L-sensor selects an H-sensor whose Hello message has the best signal strength as its cluster head, and records other H-sensors from which it has received Hello messages, will serve as backup cluster heads in the case that the cluster head fails.

In SS-H, we make use of the following assumptions:

1. Only a limited number of L-sensors may be compromised by an attacker during the short time period of the key establishment between L-sensors [5], [6].

2. BS and H-sensors will not be compromised by an attacker [3]–[5].

A. Two-Dimensional Backward Key Chain

The method for constructing a two-dimensional backward key chain C_j is as follows:

1. A backward key chain, whose length is n, is generated by a generation key g_j as follows:

$$k_j^i = H_1(k_j^{i+1}),\ \text{where}\ k_j^n = H_1(g_j),\ 1 \leq i \leq n-1. \quad (1)$$

Manuscript received October 11, 2016; revised December 1, 2016; accepted January 1, 2017. Date of publication January 10, 2017; date of current version April 7, 2017. This work was supported in part by the HPCSIP Key Laboratory, Ministry of Education, in part by the Research Foundation of Education Committee of Hunan Province, China, under Grant 16A110, in part by the Hunan Provincial Natural Science Foundation of China under Grant 17JJ4102, and in part by the National Natural Science Foundation of China under Grant 61672543. The associate editor coordinating the review of this paper and approving it for publication was P. Li. (*Corresponding authors: Weiping Wang; Boqing Zhou.)*

S. Li is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and also with the Department of Information, Hunan University of Humanities, Science and Technology, Loudi 417000, China (e-mail: sujunli@mail.csu.edu.cn).

W. Wang and J. Wang are with the School of Information Science and Engineering, Central South University, Changsha 410083, China (e-mail: wpwang@mail.csu.edu.cn; jxwang@mail.csu.edu.cn).

B. Zhou and Y. Cheng are with the Department of Information, Hunan University of Humanities, Science and Technology, Loudi 417000, China (e-mail: zbq_paper@163.com; chy8370002@gmail.com).

J. Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

Digital Object Identifier 10.1109/LWC.2017.2650986

2. A forward key chain, whose length is L, is generated by a generation key k_j^i as follows:

$$k_j^{(i,l)} = H_2(k_j^i, k_j^{(i,l-1)}), \text{ where } k_j^{(i,1)} = H_2(k_j^i, k_j^i), 1 \le l \le L. \quad (2)$$

B. Master-Slaver Model

In this model, an H-sensor can calculate the key shared with an L-sensor, but an L-sensor cannot. The details about it are described as follows:

1. Key pool. The key pool, consists of m two-dimensional backward hash key chains, is divided into two parts. One is key pool of H-sensors, namely P_g ($P_g = \{g_j, 1 \le j \le m\}$). The other is key pool of L-sensors, namely P^i, which consists of two parts: a second-dimensional generation key pool, namely P_1^i ($P_1^i = \{k_j^i, 1 \le j \le m\}$) and an ordinary key pool, namely P_2^i ($P_2^i = \{k_j^{(i,l)}, 1 \le j \le m, 1 \le l \le L\}$).

2. Before deployment, each H-sensor picks $t3$ keys from P_g. An L-sensor a^i deployed in the i^{th} phase picks $t1$ and $t2$ keys ($t1 + t2 << t3$) from P_1^i and P_2^i, respectively, which meets the following condition: the number of keys from a two-dimensional backward hash key chain is no more than 1. For example, if key $k_j^{(i,l_1)}$ has been pre-distributed to a^i, then key $k_j^{(i,l_2)}$ cannot be pre-distributed to a^i.

C. Pairwise Key Establishment

After deployment, shared keys between nodes should be established. The above establishment procedure is described as follows:

Between Two L-Sensors: The method of establishing a shared key between L-sensors is as same as in [6]. Their shared keys include the following two parts: 1. Pre-distribution common keys; 2. Shared keys obtained by calculation. For example, if a^i and b^i are pre-distributed keys k_j^i and $k_j^{(i,5)}$, then a^i can calculate their shared keys $k_j^{(i,5)}$ using the formula (2). Once this procedure finishes, each L-sensor saves hashed keys in its key ring. For example, supposing L-sensor a^i is pre-distributed two keys $k_{j_1}^i$ and $k_{j_2}^{(i,l)}$. If this procedure ends, a^i saves the following two hashed keys: $H_2(k_{j_1}^i, ID_{a^i})$, and $H_2(k_{j_2}^{(i,l)}, ID_{a^i})$, where ID_{a^i} is the identity of a^i.

Between Two H-Sensors: If the number of common generation keys between two H-sensors is more than 0, their pairwise key is the hash of all common keys.

Between an L-Sensor and an H-Sensor: In SS-H, a pairwise key between an H-sensor and an L-sensor in its cluster is established using the following *EQ* method:

Step 1. Firstly, an L-sensor a_j^i randomly selects EQ keys, $K_1, \ldots, K_{EQ}$, from its key ring, then calculates the shared key, $K_{a_j^i - CH} = K_1 \oplus \cdots \oplus K_{EQ}$, lastly submits the following information to the cluster head *CH:*

$$inf_{a_j^i} = \{ID_{CH}, ID_{a_j^i}, ID_{K_1}, \ldots, ID_{K_{EQ}}, M_{K_{u_j - CH}}(ID_{K_1} \| \cdots \| ID_{K_{EQ}})\}$$

where ID_{K_j} is the identity of key K_j, $M_K(f)$ represents the authentication code of message f with key K.

Step 2. When *CH* receives all request information from L-sensors in its cluster, it divides the list of key IDs, namely $KList_{ID}$, into the following three categories: 1. These keys can be calculated by the *CH*; 2. These keys can be calculated by *CH*'s neighbors while *CH* cannot; 3. These keys do not belong to the above two categories, namely $KList_{ID}^3$. For these keys in $KList_{ID}^3$, *CH* randomly selects an assistant H-sensor, namely CH_1, from its neighboring H-sensors, and sends the following information to CH_1:

$X_{\text{inf}} = \{ID_{CH_1}, ID_{CH}, KList_{ID}^3, MAC_{K_{CH-CH_1}}(KList_{ID}^3)\}$. When CH_1 receives the above message, it divides the IDs list into three categories as *CH* does. If the third category IDs list $KList'^3_{ID}$ is not empty, then CH_1 performs the same operation as *CH* does, selects an assistant node CH_2 from its neighboring H-sensors and sends the following request information to it:

$$X'_{\text{inf}} = \{ID_{CH_2}, ID_{CH_1}, ID_{CH}, KList'^3_{ID}, MAC_{K_{CH_1-CH_2}}(KList'^3_{ID})\}.$$

Assistant H-sensors repeat the above operations until the third key IDs list is empty.

Step 3. After *CH* securely obtains all key information of $KList_{ID}$, it calculates each shared key $K'_{a_j^i - CH}$ with each L-sensor a_j^i. Then it recalculates the authentication code of $KList_{ID}$ using $K'_{a_j^i - CH}$(see step 1). If the recalculated authentication code is not the same as it received from a_j^i, then it requests a_j^i to retransmit $inf_{a_j^i}$ (see step 1. In the letter, the number of retransmission is not more than T. We will study the practical setting of T in our future research.). Otherwise, *CH* sends the following authentication message to a_j^i:

$$MX_j = \{ID_{a_j^i}, ID_{CH}, MAC_{K'_{a_j^i - CH}}(ID_{a_j^i} || ID_{CH})\}.$$

Step 4. When a_j^i receives the above retransmission message, if the times of retransmission is not more than T, then it retransmits $inf_{a_j^i}$ to *CH*. Otherwise, a_j^i reselects new cluster head. When a_j^i receives information MX_j, it recalculates the authentication code using $K_{a_j^i - CH}$. If the authentication code verifies success, then $K_{a_j^i - CH}$ is their shared key. Otherwise, if the times of retransmission of MX_j are less than T, then a_j^i requests *CH* to retransmit MX_j. Or else, a_j^i reselects new cluster head.

Step 5. When *CH* receives the retransmission of MX_j, if the retransmission times are not more than T, then it retransmits MX_j.

III. Performance and Security Evaluation

In this section, we analyze the performance and security of our scheme. In our analysis and simulations, we use the following setups:

1. Deployment area is 500m×500m.

2. The value of R and r is 160m and 40m, respectively.

3. Nodes deployment includes 3 phases, and nodes are randomly and evenly distributed in the network in each phase. In each phase, H-sensors account for the ratio of is 8%, and the number of nodes deployed and captured is 400.

4. The number of key chains is 6000 ($m = 6000$), and the length of each forward key chain is 20 ($L = 20$).

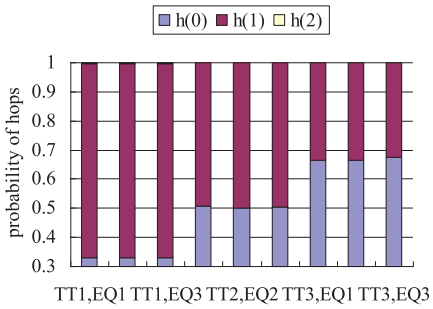

Fig. 1. Distribution of the number of hops as a function of various parameters. In the Fig., $t1 + t2 = 70$, TT1, TT2 and TT3 represent t3=2000, t3=3000 and t3=4000, respectively. EQ1, EQ2 and EQ3 represent $EQ = 1$, $EQ = 2$ and $EQ = 3$, respectively.

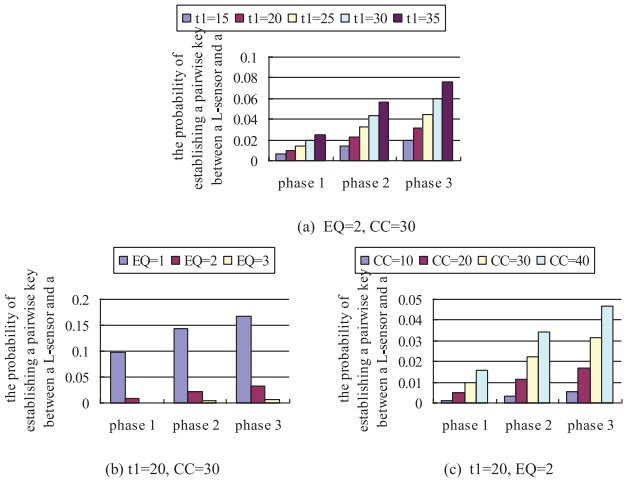

Fig. 2. Function of the probability that a replication H-sensor can establish a pairwise key with an L-sensor and t3. In the Fig., t1+t2=70.

A. Performance

In SS-H scheme, *EQ* method is used to improve resiliency of H-sensor replication attack. In *EQ*, a key path is needed to be established when a head master *CH* has not all keys requested by L-sensors in its cluster (see Section II-C). Therefore, the length of the key path, namely *Hp*, is a very important parameter. In our scheme, the value of *Hp* lies on the probability of keys selected by L-sensors existing in their cluster head's key ring, namely $P_{L\text{-}H}$, and the expectation value of H-sensors' neighbors. Next, we will describe it in detail.

The number of selecting two-dimensional backward key chains for H-sensors and L-sensors is $\binom{m}{t1+t2}$ and $\binom{m}{t3}$, respectively. If there are x common key chains shared between an H-sensor and an L-sensor, then the number of selecting keys from these key chains is : $\binom{m}{x} \cdot \binom{m-x}{2(t1+t2+t3-x)} \cdot \binom{2(t1+t2+t3-x)}{t3-x}$. The number of selecting a key by an L-sensor from the x common key chains is $\binom{x}{1}$. Therefore, the above probability is:

$$P_{L\text{-}H} = \frac{\sum_{x=1}^{t1+t2} \binom{m}{x} \cdot \binom{x}{1} \cdot \binom{m-x}{2(t1+t2+t3-x)} \cdot \binom{2(t1+t2+t3-x)}{t3-x}}{\binom{m}{t1+t2}\binom{m}{t3}} \quad (3)$$

In our analysis, it is supposed that the length of a key path, when an H-sensor *CH* can directly calculate keys requested by L-sensors in its cluster, which is 0; when these above requested keys can be calculated by *CH*'s direct neighbors while *CH* cannot, which is 1; and so on. As a result, the probability of keys being obtained by i hops is:

$$P_{h(i)} = \left(1 - \sum_{i_1=0}^{i-1} P_{h(i_1)}\right) \cdot \left(1 - (1 - P_{L\text{-}H})^{N_H}\right) (i \geq 1) \quad (4)$$

where $P_{h(0)} = P_{L\text{-}H}$, N_H is the expectation value of H-sensors' neighbors.

The formula (3) shows that $P_{L\text{-}H}$ increases with the increase of $t1 + t2$ and $t3$. The formula (4) indicates that the number of hops decreases with the increase of $P_{L\text{-}H}$. This conclusion can be verified by Fig. 1. If the value of $t1 + t2$ is fixed, and $t3$ increases to 4000 from 2000, then $P_{L\text{-}H}$ increase to 0.66 from 0.33 and $P_{h(2)}$ decrease to 0 from 0.003. Also, Fig. 1 shows that cluster heads need about one hop to get all requested keys.

B. Resilience Against H-Sensors Replication Attack

SS-H scheme's resilience against H-sensors replication attack can be evaluated from the following two aspects: 1. The probability that replication H-sensors can establish shared keys with normal H-sensors, namely $PR_{H\text{-}H}$. 2. The probability that replicated H-sensors can establish shared keys with an L-sensor a^i, namely $PR_{L\text{-}H}$.

1. $PR_{H\text{-}H}$: The function of two-dimensional key chain shows that it is not feasible to calculate generation keys pre-distributed to an H-sensor using keys pre-distributed to an L-sensor. That is to say, $PR_{H\text{-}H} = 0$.

2. $PR_{L\text{-}H}$: In the I^{th} capture, supposing the number of L-sensors' pre-distribution keys are not hashed before being compromised is CC^I. After I times capture, the probability of keys from P^i being compromised is

$$PBKC^{(i,I)} = 1 - \left(1 - \frac{t1}{m}\right)^{\sum_{i_1=i}^{I} CC^{i_1}} \cdot \left(1 - \frac{t2}{m \cdot L}\right)^{CC^i}. \quad (5)$$

Therefore, after I times capture, the expected value of the probability of key pool P^i being compromised is about $m' = m \cdot PBKC^{(i,I)}$. $PR_{L\text{-}H}$ can be estimated as follows:

$$PR_{L\text{-}H} = \frac{\sum_{x=EQ}^{\min(m', t1+t2)} \binom{m'}{x} \cdot \binom{m-m'}{t1+t2-x} \cdot \binom{x}{EQ}}{\binom{m}{t1+t2}\binom{t1+t2}{EQ}} \quad (6)$$

where min($t4$,$t5$) represents the minimum of $t4$ and $t5$.

In Fig. 2, in each phase, CC^I is fixed, and it is represented by CC. Formulas (5)-(6) indicate that: 1. $PR_{L\text{-}H}$ is more likely to be affected by $t1$ as comparing with $t2$. 2. $PR_{L\text{-}H}$ decreases dramatically with the increase of EQ. 3. $PR_{L\text{-}H}$ increases with the increase of CC. The above three conclusions can be verified by Fig. 2. For example, in the 3rd phase, when $t1 + t2 = 70$, $EQ = 2$, and $CC = 30$, $t1$ increases to 35 from 15, $PR_{L\text{-}H}$ increases about to 0.076 from 0.02 (see fig. 2(a)). When $t1 = 20$, $t2 = 50$ and $CC = 30$, EQ increases to 3 from 1, $PR_{L\text{-}H}$ decreases about to 0.006 from 0.167 (see fig. 2(b)). When $t1 = 20$, $t2 = 50$ and $EQ = 2$, CC increases to 40 from 10, $PR_{L\text{-}H}$ increases about to 0.046 from 0.005 (see fig. 2(c)).

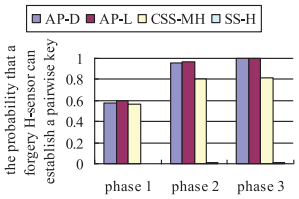

Fig. 3. Comparing the probability that a replication H-sensor can establish a pairwise key with an L-sensor.

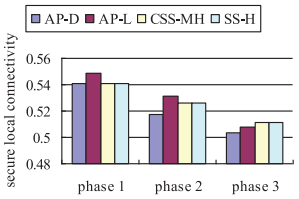

Fig. 4. Comparing the secure local connectivity.

C. Comparison With Existing Scheme

In the section, performance and security between SS-H and schemes *AP-D* [3], *AP-L* [4] and CSS-SH [5] are compared. For the sake of fairness, assumptions in the other three schemes are the same as in SS-H. That is to say, pre-distribution keys of L-sensors are hashed after pairwise key establishment between L-sensors, and *CC* equals to 5. For four schemes, the size of key pool is 6000, and the number of pre-distribution keys of an H-sensor and an L-sensor is 3000 and 70, respectively. In AP-L, the key pool of L-sensors is a subset of H-sensors and its size is 5800. In CSS-SH, an H-sensor only picks keys from P_1^i. In *AP-D*, *AP-L* and CSS-SH, pairwise key can be established between an H-sensor and an L-sensor, and between two H-sensors if the number of their shared keys is no less than 4 and 500, respectively. In SS-H, *EQ* equals to 2. Other parameters settings are as same as Section IV.

In AP-D, AP-L and CSS-SH, H-sensors and L-sensors share the same key pool. Therefore, an adversary can easily successfully launch replication H-sensors attack by capturing L-sensors. Ideally, when there are 8 L-sensors are captured, in this case, $8\times70>500$, then $PR_{H\text{-}H}$ equals to 1. However, in SS-H, previous analysis indicates that no matter how many L-sensors are captured, $PR_{H\text{-}H}$ equals to 0.

In *AP-D*, *AP-L* and CSS-SH, shared keys, between an L-sensor and an H-sensor, are from L-sensors' key pool, and H-sensors are needed to participate in the establishment of pairwise key. If L-sensors are compromised before their pre-distribution keys are not hashed, then replication H-sensors can establish pairwise key with normal L-sensors using these compromised keys. Fig. 3 shows the comparisons of $PR_{L\text{-}H}$ of each scheme. In AP-D and AP-L, the key pool is fixed throughout the lifecycle of HSNs. The size of L-sensors' key pool of AP-L scheme is less than that of AP-D scheme, as a result, $PR_{L\text{-}H}$ of AP-L scheme is higher than that of AP-D scheme. In CSS-MH, key pools of any two phases are different, that is, $P^{i_1} \cap P^{i_2} = \phi$ $(i_1 \neq i_2)$. Therefore, $PR_{L\text{-}H}$ of CSS-MH is less than that of AP-D and AP-L scheme. In SS-H, shared keys, between an H-sensor and an L-sensor, are selected randomly from the key ring of the L-sensor, which will lead to a significant reduction in the value of $PR_{L\text{-}H}$. As shown in Fig. 3, in the 3rd phase, $PR_{L\text{-}H}$ of *AP-D*, *AP-L*, CSS-SH and SS-H is about 1, 1, 0.83 and 0.01, respectively.

In HSNs, secure communication between L-sensors also is very important [3]–[5]. To measure the local connectivity of the four schemes precisely, in here, secure local connectivity (*SPC*) is used, namely $SPC{=}PC{\times}(1\text{-}PRC)$, where *PC* is the probability of pairwise keys establishment between L-sensors, and *PRC* denotes the probability that pairwise keys established between nodes are not compromised after *CC* L-sensors are captured. In AP-L, the size of L-sensors' key pool is less than that of the other three schemes. Therefore, in the 1st phase, secure local connectivity of AP-L is slightly higher than that of the other three schemes. But its secure local connectivity decreases quickly. In CSS-MH and SS-H, their key pools of each phase are different, so their secure local connectivity drops slowly. As shown in Fig. 4, in the 3rd phase, *SPC* of *AP-D*, *AP-L*, CSS-SH and SS-H is about 0.503, 0.507, 0.511 and 0.511, respectively.

IV. Conclusion

In this letter, master-slaver model, is created, and is realized using a new two-dimension backward key chain. In addition, *EQ* method is presented. Analysis and simulation indicate that the master-slaver model can prevent replication H-sensors from communicating with normal H-sensors no matter how many L-sensors are compromised and the *EQ* method can significantly reduce the probability of pairwise key establishment between replication H-sensors and normal L-sensors.

References

[1] *Crossbow Technology Inc.*, Milpitas, CA, USA, 2004. [Online]. Available: www.xbow.com

[2] K. Xu, X. Hong, and M. Gerla, "An ad hoc network with mobile backbones," in *Proc. ICC*, New York, NY, USA, Apr. 2002, pp. 3138–3143.

[3] X. Du, Y. Xiao, M. Guizani, and H.-H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Netw.*, vol. 5, no. 1, pp. 24–34, 2007.

[4] K. Lu, Y. Qian, M. Guizani, and H.-H. Chen, "A framework for a distributed key management scheme in heterogeneous wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 2, pp. 639–647, Feb. 2008.

[5] B. Zhou, J. Wang, S. Li, Y. Cheng, and J. Wu, "A continuous secure scheme in static heterogeneous sensor networks," *IEEE Commun. Lett.*, vol. 17, no. 9, pp. 1868–1871, Sep. 2013.

[6] S. Li, B. Zhou, J. Dai, and X. Sun, "A secure scheme of continuity based on two-dimensional backward hash key chains for sensor networks," *IEEE Commun. Lett.*, vol. 1, no. 5, pp. 416–419, Oct. 2012.