International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:12, 2009

An Enhanced Key Management Scheme Based on Key Infection in Wireless Sensor Networks

Han Park, *Non-Member, IEEE,* and JooSeok Song, *Member, IEEE*

Abstract—We propose an enhanced key management scheme based on Key Infection, which is lightweight scheme for tiny sensors. The basic scheme, Key Infection, is perfectly secure against node capture and eavesdropping if initial communications after node deployment is secure. If, however, an attacker can eavesdrop on the initial communications, they can take the session key. We use common neighbors for each node to generate the session key. Each node has own secret key and shares it with its neighbor nodes. Then each node can establish the session key using common neighbors' secret keys and a random number. Our scheme needs only a few communications even if it uses neighbor nodes' information. Without losing the lightness of basic scheme, it improves the resistance against eavesdropping on the initial communications more than 30%.

Keywords—Wireless Sensor Networks, Key Management

I. Introduction

IT is clear that a sensor will make our lives more comfortable and safer in the future. As a sensor becomes smaller and cheaper, many applications start to use it. We can use it as not only a fire alarm or a gas leak detector, but also a biochemical attack watcher.

The more we depend on sensors, the more security becomes important. Invalid information from compromised sensors can cause serious problems. There are many security mechanisms for the sensor network. Key management, the mechanisms about establishing a key, is the first step for secure communication.

There are, however, many constraints in wireless sensor networks.

- *Limited capabilities of a sensor node.* Most sensors are tiny and low-priced, so it has limited battery capability, processing power and memory size. It is hard to use public key cryptography like RSA [7] because of the limited computation powers.
- *Physical vulnerability.* A sensor can be compromised physically by an attacker. It may be deployed in public area, thus an attacker can capture a sensor node and take information from its memory. The entire network should be robust even if some nodes are compromised by attackers.
- *No existence of a trusted center.* No existence of a trusted center is one of the significant problems. There is no Key Distribution Center (KDC) for symmetric key cryptography or Certification Authority (CA) for public key cryptography.
- *Broadcasting.* We cannot predict any geographical characteristics and network topologies, so we have to use broadcasting instead of unicasting or multicasting. An attacker can eavesdrop on the network traffics easily.

Han Park is with the Department of Computer Science, Yonsei Univiersity, Seoul, Korea e-mail: ipuris@emerald.yonsei.ac.kr
JooSeok Song is with the Department of Computer Science, Yonsei Univiersity, Seoul, Korea e-mail: jssong@emerald.yonsei.ac.kr

Because of these constraints, it is impractical to use such mechanisms which have been applied in other networks like Kerberos [5], Diffie-Hellman key agreement [6], and public-key distribution [7].

Key Infection [3] is one of the lightweight key management schemes for wireless sensor networks. It broadcasts a secret key in plaintext. Then each node establishes a session key encrypted with the secret key. Communication in plaintext is easy to be eavesdropped, but they have shown that using the initial communications just after deployment is secure sufficiently.

Key Infection, however, has vulnerability when the percentage of the black node is increased. They have evaluated their scheme where the percentage of the black node is 1%, 2%, or 3%, but there are some factors that can increase the percentage. Predicting the deployed area or using more powerful equipment can be a reason that increases the percentage of black node. From this point of view, considering larger value is more reasonable and realistic. In the case of the basic scheme, however, the percentage of compromised node is almost 16% where the percentage of the black node is 7%. To address this vulnerability, we propose a new key management scheme. Our scheme is more secure than basic scheme especially when the percentage of the black node is increased.

At first we will review previous works in this area in section II. We leverage Key Infection, so we will introduce the basic scheme in section III. After then, we will present ours in section IV. We analyze and evaluate our scheme in section V. The conclusion is in section VI.

II. Related Works

In this section, we will review previous works. Many key management schemes have been suggested.

C. Hartung, J. Balasalle and R. Han have proposed network-wide shared key scheme [4]. It establishes a session key using a pre-determined master key. Then it removes the master key from the memory. If we assume that it is impossible to eavesdrop on traffic or capture some nodes during the time when the nodes use the master key, it is a very strong mechanism. The time when the nodes use the master key is very short. It needs, however, enough time to finish node deployment phase generally. It means attackers have enough time to take a master key by node capture. If an attacker captures a node and takes

a master key during node deployment phase, entire network is no longer secure even if the time when the nodes use the master key is very short.

L. Eschenauer and V. D. Gligor have proposed Random Key Pre-distribution (RKP) scheme [1]. They use many possible keys called key pool. Each node has some of them in its memory before deployment. After deployment, each node can establish the session key if one node and another node have the same key. It is called shared key. This scheme is based on birthday paradox, and it connects each node with a very high probability. If the sufficient number of node is captured, however, entire key pool can be revealed. It means attackers can take every possible session key. It also uses too much memory inefficiently, so it is impractical to adapt to real applications.

H. Chan, A. Perrig and D. Song have enhanced the basic RKP scheme in various ways [2]. There are three key management mechanisms in the paper. Q-composite Key scheme, one of their three schemes, uses q keys to generate a session key instead of one key. It strengthens the network's resilience against node capture when small numbers of nodes are captured, but it is more dangerous when large number of nodes are captured. Multipath Key Reinforcement scheme, another one of their three schemes, uses random values which are generated by common neighbors to establish a session key. It needs, however, too many communications for establishing the session key. Our scheme has similar ideas that use neighbor nodes and use several keys not only one key. We, however, use more efficient and secure way. Random Pairwise Key, the last one of their three schemes, enhances RKP against node capture by using a pairwise key between two nodes.

R. Anderson, H. Chan and A. Perrig have proposed Key Infection scheme and real world attacker model [3]. Our scheme leverages this scheme, so we will discuss about it in next section.

III. Basic Key Infection Scheme

Key Infection is the lightweight key management scheme for wireless sensor networks. This scheme is quite simple.

1) $i : \{i, k_i\} \rightarrow j$
2) $j : \{j, k_{ji}\}_{k_i} \rightarrow i$

After the node i is deployed, it just broadcasts a secret key in plaintext. If node j hears to the node i's signal, then node j generates a pairwise key and sends it as a response. This response is encrypted by node i's secret key, so node i can decrypt it.

They use the initial communications to exchange a session key between two nodes. It looks very dangerous because it just broadcasts a secret key in plaintext. They, however, have shown that the initial communications after deployment is secure sufficiently even if that is just a plaintext. It is very hard to eavesdrop on the initial communications over the entire deployed area realistically. According to their experiments, only 2.4% of links in entire network are compromised where there is one black node for every 100 white nodes and each node has average four neighbors.

They have also suggested a real world attacker model. In the world of wireless sensor network, we assume very strong attacker model that attackers can monitor all traffics whenever and attack whatever without any restrictions. In the real world, however, attackers can monitor or attack only a small proportion of an entire network area, some times later after node deployment. They suggested new attacker model as followings:

1) The attacker does not have physical access to the deployment site during the deployment phase;
2) The attacker is able to monitor only a small proportion (α) of the communications of the sensor network during the deployment phase. After key exchange is completed, she is able to monitor all communications at will;
3) The attacker is unable to execute active attacks (such as jamming or flooding) during the deployment phase. After key exchange is completed, she is free to launch any kind of attack.

Key infection is not secure under the attacker model we have used, but this scheme is secure enough to apply real applications under the realistic attacker model.

There is, however, a problem in their scheme and the real world attacker model. The problem is about the proportion α, which is stated second. α can be a larger value. There are many factors that make α larger. More times can be needed during the nodes are deployed from starting to end. Attackers can detect the deployment, and install more black nodes to eavesdrop. If attackers can predict the area that sensors will be deployed, the α also can be increased. The basic scheme has assumed 1%, 2%, or 3% of entire nodes as a black node, but it can be too small value and can be increased. When the percentage of the black node is 7% and the average number of neighbor node is 6, eavesdropped link of the entire network is over the 20%. At the point of this view, we need an enhanced key management scheme which has more resistance under the larger number of black nodes.

IV. Our Scheme

Our scheme consists of two phases, secret key generation phase and session key establishment phase. In secret key generation phase, each node generates and exchanges a secret key with its neighbor nodes. A session key is established in session key establishment phase. We will explain each phase in detail below.

- Secret Key Generation
  1) Generate a secret key for each node. If there is a node α, it generates k_α.
  2) Broadcast $\{\alpha, k_\alpha\}$ to neighbor nodes in plaintext.
  3) Store neighbor nodes' key in memory. If there are neighbors, node β and node γ, node α should have $\{\beta, k_\beta\}$, $\{\gamma, k_\gamma\}$.
- Session Key Establishment
  1) $\alpha : \{\{\beta, \gamma\}, R\}_{k_\alpha} \rightarrow \beta$
  2) $\beta : \{\{\alpha, \gamma\}, R-1\}_{k_\beta} \rightarrow \alpha$
  3) Session Key: $k_{\alpha\beta} = R \bigoplus k_\alpha \bigoplus k_\beta \bigoplus k_\gamma$

A. Secret Key Generation Phase

After deployment, each node generates a random value. We use it as a secret key of each node. On session key establishment phase, we will calculate a session key using these secret keys. We use *XOR* operations then, so it should be pre-determined fixed length. Length of the secret key depends on what encryption algorithm is used.

After generating the secret key, each node broadcasts their secret key to neighbor nodes in plaintext. There are vulnerabilities because of broadcasting in plaintext. The evaluation in next section, however, shows that our scheme has high resistance against eavesdropping even if we use plaintext. To generate a session key of two nodes, an attacker has to eavesdrop on not only communications between the two nodes, but also all communications including all neighbor nodes of them. According to real world attacker model [3], it is extremely difficult to eavesdrop in practical. It is also difficult with our large α stated previous section.

After hearing the broadcasting messages, each node stores sender's id and key in its memory. We will use them in next phase.

B. Session Key Establishment Phase

We use common neighbors to generate a session key. Common neighbors are the nodes within a communication range of two nodes. Let's assume that node α wants to establish a session key with node β, and there is a common neighbor, node γ.

At first, node α and node β exchange a each node's *neighbors list* and a random value, R. Node α sends its neighbors list and R to node β encrypted with the node α's secret key. R is generated by sender, and it also has same length with other secret keys. Receiving this message, node β also sends its neighbors list and $R-1$ to node α. $R-1$ means that node β successfully decrypted the message and got R from node α. Each node has a secret key for each neighbor node, because they completed secret key exchange when the secret key generation phase. They can decrypt the neighbors list and the random value encrypted by each other's secret key.

After exchanging a neighbors list and random value, each node can calculate a session key. In this case, each neighbors list from node α and β includes node γ's ID, so each node can know what the common neighbor node is. The session key between node α and node β is calculated like below.

$$k_{\alpha\beta} = R \bigoplus k_\alpha \bigoplus k_\beta \bigoplus k_\gamma$$

The session key is not transmitted by network communications. Attackers cannot take a session key by eavesdropping if previous phase is finished in secure. The only way to take a session key is node capture, but we will show that our scheme has high resistance to node capture in next section.

We use neighbor nodes to establish a session key, but only one additional communication for broadcasting its secret key is needed. To establish the session key, we need only three communications. It is a significant advantage for tiny sensors which have very limited battery capability.

TABLE I: Comparison table

	Eavesdropping	Node Capture
Network Wide Shared Key	Secure	Not secure
Random Key Pre-distribution	Secure	Not secure
Random Pairwise Key	Secure	Not secure
Key Infection	Not secure	Secure
Our Scheme	Not secure	Secure

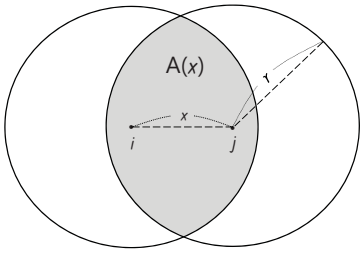

Fig. 1: Overlapped area of two nodes

V. Analysis and Evaluation

We will think about two basic attacks, eavesdropping and node capture. The key management schemes using pre-determined key is strong against eavesdropping, but they have vulnerability against node capture. For instance, in case of random key based schemes such as Random Key Pre-Distribution and Random Pairwise Key, each node has pre-determined key in its memory. It makes harder to take the session key by eavesdropping, but it is vulnerable against node capture. On the contrary, the key management schemes that do not use pre-determined key such as Key Infection is strong against node capture, but they have vulnerability against eavesdropping.

Every node needs enough common neighbors to generate a session key with other nodes. We will show there are enough common neighbors by using mathematics. We then evaluate our scheme in terms of resilience against two possible attacks, node capture and eavesdropping on the initial communications.

A. Expectation Number of Common Neighbors

In our scheme, it is important to guarantee enough common neighbors. We can calculate the expectation number of neighbor node approximately. Multipath Reinforcement scheme [2] also uses common neighbors, so we can use same equations to calculate the expectation number of common neighbor. We need the area within both nodes' communication range, $A(x)$, and the probability distribution function, $F(x)$.

Figure 1 shows the area, $A(x)$. According to Multipath Reinforcement scheme, $A(x)$ and $F(x)$ are:

$$
\begin{aligned}
A(x) &= 2r^2 \cos^{-1}(\frac{x}{2r}) - x\sqrt{r^2 - \frac{x^2}{4}} \\
F(x) &= P(distance < x) \\
&= \frac{x^2}{r^2}
\end{aligned}
$$

We can calculate the expectation area of overlap:

$$
\begin{aligned}
& \int_0^r A(x)F'(x)\,dx \\
= & \int_0^r \{2r^2 \cos^{-1}(\frac{x}{2r}) - x\sqrt{r^2 - \frac{x^2}{4}}\}\frac{2x}{r^2}\,dx \\
= & (\pi - \frac{3\sqrt{3}}{4})r^2 \\
= & 0.5865\pi r^2
\end{aligned}
$$

Equations above show that the expected area of overlap is 0.5865 of a single communication radius. The expected number of common neighbor is $0.5865n$ where n is the number of neighbor for each node. If we use the same example with the Multipath Reinforcement scheme, $n = 60$, there are 35 common neighbors between two nodes. Even if there are only 5 neighbor nodes, we can find 3 common neighbors. It is the sufficient number of common neighbor for our scheme.

B. Against Node Capture

If an attacker captures a node α, she can take session keys and secret keys which are stored in the node. It is impossible to conceal the session keys against node capture since the session keys must be stored in its memory. They cannot be protected by key management algorithms but some other mechanisms. There are such mechanisms for detecting node capture and protecting memory contents against node capture, but that is out of this paper's scope. More important issue in key management is guaranteeing confidentiality of session keys for others. The entire network should be secure even if some nodes are captured.

Let's assume that the attacker captures node α. Then the attacker can take node α's secret key and its neighbor node's secret keys stored in node α. More important issue is that the session key between its neighbor nodes should be secure. In our scheme, node α's neighbor nodes use a random value, R, to generating a session key between themselves. R is not stored in any node's memory. She cannot know R even if she has captured the node α, so she cannot generate the session keys for them. It means remaining network is secure even if some nodes is captured.

Key Infection has very strong resistance against node capture. It uses a pairwise session key for each peer. Our scheme also uses R to make session key as a pairwise key, so it has a same strong point.

C. Against Eavesdropping

If black nodes were deployed before white nodes are deployed, they can eavesdrop on the initial communications. We suppose malicious nodes are uniformly and random deployed, but they cannot cover entire network area.

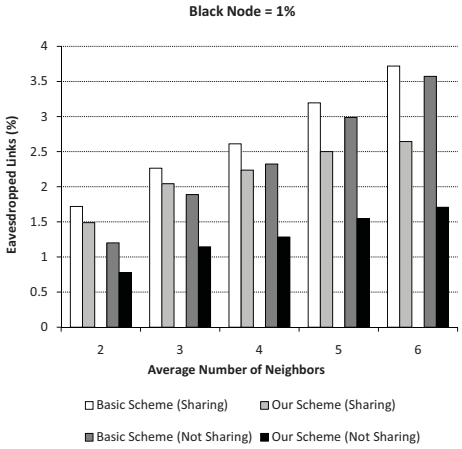

Fig. 3: Sharing mode and not sharing mode

One factor that we should consider is about the sharing of eavesdropped information among black nodes. The basic scheme, Key Infection, does not consider the sharing of information. It is, however, not reasonable because attackers can use more powerful sensors or equipments to eavesdrop on such communications. We analyze our scheme with both two conditions, sharing eavesdropped information among black nodes (sharing mode) and not sharing mode.

At first, we use same conditions with basic scheme. 10,000 white nodes are random and uniformly distributed, and there are 1%, 2%, or 3% of black nodes. The entire node has a communication range of 10 meters. Each result is an average value of 100 simulations. A variable of each simulation is d, the average number of neighbor nodes. We evaluate the percentage of eavesdropped (compromised) links in the entire network.

Figure 2 shows that there are significant improvements where eavesdropped information is not shared regardless of the percentage of black nodes. The simulations where $d = 2$ show about 35% improvements compared with the basic scheme. Where $d = 6$, the improvement rate increases over the 50%.

We can compare our scheme with the basic scheme when attackers share the eavesdropped information (sharing mode) by Figure 3. It shows almost 30% improvements where $d = 6$. The improvement rate is lower than the rate of not sharing mode, but it is also significant improvements.

We also evaluate our scheme under the worse condition. Other conditions are the same with not sharing mode, but there are more black nodes up to 7%. Figure 4 shows the result of the experiment where the average number of neighbor nodes is 4. Our scheme registered only a half number of the link is eavesdropped compared with the basic scheme.

VI. Conclusion and Future Work

Key management is a challenging part in wireless sensor network. Wireless sensor networks have various constraints,

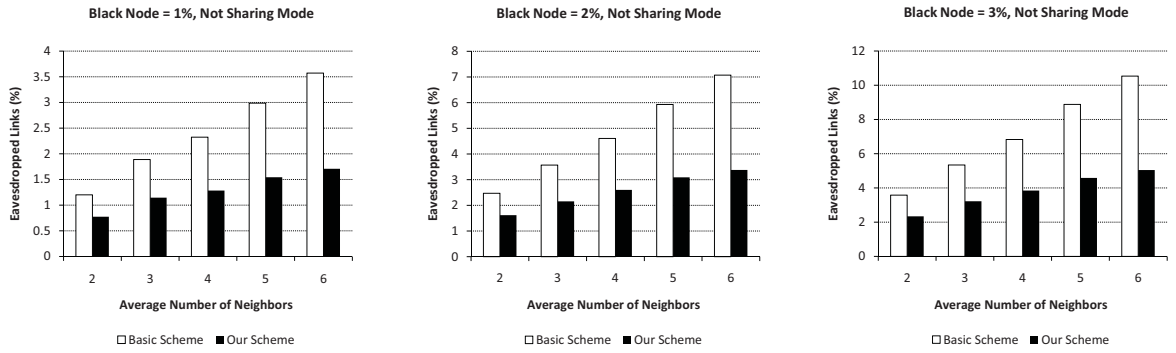

Fig. 2: Simulations with not sharing mode

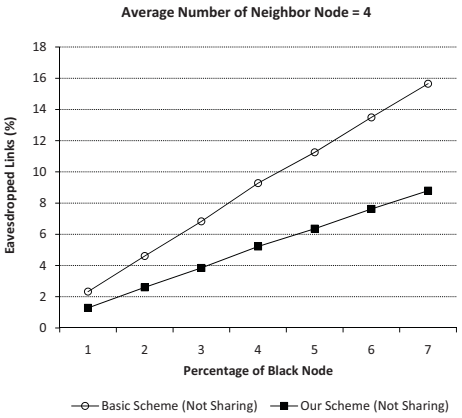

Fig. 4: Percentage of the black node is up to 7%

and we should always consider them.

We present that the basic scheme, Key Infection, has a weakness when the proportion of the eavesdropped links is increased. It uses real world attacker model instead of previous unrealistic model. We point out that the realistic attacker model is too loose. They have assumed that only a small proportion of the entire network is eavesdropped during the deployment phase. The proportion, however, can be a larger value even if eavesdropping on all links is not realistic. The basic scheme has assumed that the percentage of the black nodes is only 1%, 2%, or 3%, but it is more reasonable that the percentage can be increased up to 5% or 7%.

To address this problem, we propose a new key management scheme for wireless sensor networks based on Key Infection. Our scheme uses common neighbors to establish a session key. It makes eavesdropping more difficult. We evaluate the resistance of the entire network against eavesdropping on the initial communications. With the basic scheme, where the black node's ratio is 7% and the average number of neighbor node is 6, the percentage of the eavesdropped link of the entire network is about 23%. It is too high value for secure communication. Our scheme with same condition registers 11%, it is more than 50% improvements. Our scheme also shows steady improvement rates regardless of the various conditions.

Our scheme has some limits. Secret key generation phase and session key establishment phase are secure only if they use initial communications. There, however, can be some situations that they cannot use initial communications. For instance, node addition is the certain issue. We can also use our scheme when some nodes are added, but there is too much vulnerability. If we allow node addition, we cannot distinguish whether added nodes are white nodes or black nodes. Not supporting node addition can be a more practical solution in this case.

ACKNOWLEDGMENT

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MEST)(2009-0076476).

REFERENCES

[1] L. Eschenauer and V. D. Gligor, *A Key-management Scheme for Distributed Sensor Networks*, Proceedings of the 9th ACM conference on Computer, 2002.
[2] H. Chan and A. Perrig and D. Song, *Random Key Predistribution Schemes for Sensor Networks*, IEEE Symposium on Security and Privacy, 2003.
[3] R. Anderson, H. Chan and A. Perrig, *Key Infecition: Smart Trust for Smart Dust*, 12th IEEE International Conference on Network Protocols (ICNP), Oct. 2004.
[4] C. Hartung, J. Balasalle, and R. Han, *Node Compromise in Sensor Networks: The Need for Secure Systems*, Technical Report CU-CS-988-04, Dept. of Computer Science, University of Colorado at Boulder, 2004.
[5] B. C. Neuman and T. Ts'o, *Kerberos: An authentication service for computer networks*, IEEE Communications magazine, 1994.
[6] W. Diffie and M. E. Hellman, *New Directions in Cryptography*, IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp: 644-654.
[7] R. L. Rivest, A. Shamir, L. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM, 1978.
[8] B. A. Forouzan, *Cryptography and Network Security*, McGraw Hill, 2008.

Han Park received the B.S. degree in computer science from Yonsei University, Seoul, Korea, in 2009. He is currently a graduate student of computer science at Yonsei University, Seoul, Korea. His research interests include cryptography, network security, and social network.

JooSeok Song received the B.S. degree in electrical engineering from Seoul National University, Seoul, Korea, in 1976, and the M.S. degree in electrical engineering from KAIST, Korea, in 1979. In 1988, he received the Ph.D. degree in computer science from University of California at Berkeley. He had been an assistant professor of Naval Postgraduate School, California, USA, from 1988 to 1989. He is currently a professor of computer science at Yonsei University, Seoul, Korea. His research interests include cryptography, network security, and data communications.