SecOPP+: A Secure Dynamic Scheme for Adding new Nodes in SecOPP Protocol

Hela Maddar[1], Abdelbasset trad[1], Abderrahmen Guermazi[2], Sofienne Ben Othman[1]
[1]PRINCE Research Unit
ISITcom, Hammam Sousse
University of Sousse, Tunisia
[2]ISET Sfax
Tunisia
maddar.hela.si@gmail.com, Abdelbasset.Trad@isigk.rnu.tn
Abderrahmen.Guermazi@tunet.tn, ben_oth_soufiene@yahoo.fr

ABSTRACT: *In this paper, we propose an enhancement to the secure routing protocol SecOPP. SecOPP provides the most security services such as integrity, authentication, confidentiality and freshness of data. All these services are obtained with low energy consummation. In addition, this protocol is characterized by little stockade places and reduced computational cost. Despite its robustness against attacks and its scalability, SecOPP in its current application is static. It does not allow secure addition of new nodes. Each node has a limited life time, when batteries became empty, node dies. We propose to securely add new nodes on network to increase its longevity, making it more reliable and robust. This is well verified through simulations with TOSSIM using NesC language.*

Keywords: Wireless Sensor Network (WSN), Security, Key Management, SecOPP, SecOPP+

Received: 2 June 2013, Revised 3 July 2013, Accepted 9 July 2013

1. Introduction

The miniaturization of sensors nodes, the cost more and more increasingly low, different sensor nodes available and the wireless communication medium used allow sensor networks to be very useful in many applications. They also help to extend existing application. Sensor node can be very useful in many and different application despite their limited resources in computing and energy storage. Generally sensor nodes should collect data and send them to the base station.

For these reasons, Wireless Sensor Networks (WSNs) become increasingly used and they need an extremely security to be efficient. Many routing protocols are proposed to WSNs [1]. However there are many attacks against WSNs and especially against routing protocols which are not secure [2]. In this paper, we are interested in the Secure One Phase Pull diffusion routing protocol (SecOPP) [3] which is a lightweight version of Directed Diffusion[4]. SecOPP protocol is a secure data-centric routing protocol which is deployed with flat WSNs. It permits creation of multi-hop routing paths to disseminate gathered data toward base station. SecOPP provides the most security services. It reduces the diffusion overhead in network, it allows little stockade place and reduced computational cost. In our SecOPP+ scheme, we propose an enhancement of SecOPP by the secure addition of new nodes and that's an essential step to increase its longevity, making it more reliable and robust. SecOPP+ protocol requires

the design of an appropriate key pre-distribution scheme which satisfies security goals and at the same time ensures low computation energy consumption and little stockade places.

This paper is organized as follows: section 2 presents the related work, section 3 describes existing scheme, section 4 describes the proposed scheme, section 5 presents simulation results, and finally section 6 concludes the paper.

2. Related Work

Security is vital for WSN, especially for sensitive applications such as military, medical and industrial WSN. Packets transmitted from one node to another must be protected until the arrival of data to the base station. For a secure network we should ensure security objectives, the main ones include: authentication, confidentiality, data integrity, and freshness data.

These security objectives cannot be applied on the network without security mechanism such as cryptography, hashing, MAC (Message Authentication Code) [5] and others. Also in a WSN, security cannot be used and implemented without a key management mechanism. Key management in WSNs is based on symmetric cryptography and not asymmetric cryptography because resources limitation of sensor nodes (memory size, energy limited, computing capacity…). Key management typically involves four essential tasks: Key generation, key storage, distribution or exchanging key and key verification.

The simplest scheme is to pre-load the same key in all network nodes. This key will be used after deployment to ensure a secure communication. The major disadvantage of this approach is the possibility of compromising a node, so, the entire network can be compromised.

In another solution called naive, all pairs of nodes share the same key. This protocol is resistant against capture, but it does not allow the addition of new nodes in the network and does not allow the passage to the scale.

Other well-known solutions are based on pre-distribution protocols that can be deterministic or probabilistic. Eschenauer and Gligor [6] propose the first probabilistic scheme. The main idea of this scheme is to distribute randomly a number of keys from a defined set for each node before deployment. After deployment, two nodes have the possibility to exchange messages and should have at least one key in common. Chan, Perring and Sang [7] propose a probabilistic scheme based on the same idea of [6] but they have looked to improve safety under the pressure of attack. In this scheme, nodes should share q key with $q >= 2$ to communicate together.

Unlike probabilistic schemes, deterministic key management protocols ensure that each node is able to establish a communication with any neighbor node. To ensure determinism, protocols such as LEAP [8] OMTK [9] and SecOPP [3], [10] use a common key transitional pre-loaded on all nodes before their deployment. This key is used to generate the necessary keys between neighboring nodes after deployment. This key will be deleted after the creation of communication key.

3. SecOPP Protocol Description

SecOPP [3] [10] is a secure version of the routing protocol One Phase Pull diffusion which is a lightweight variant of Directed Diffusion. SecOPP allows deterministic shared key to secure routing on multi-hop paths. In fact, it provides the most security services, such as integrity, authentication, confidentiality and freshness on exchanging messages. All these services are obtained with low energy consumption, simple storage space and low computational cost. This allows him to share effectively three types of keys: a global key before deployment, group key and pair wise key after deployment. SecOPP is applied in a static network with homogeneous nodes and random deployment; it offers to the WSNs a flat topology.

Before deployment, all nodes are recharged by a global common key shared with the base station. This key is used to establish two key: pair wise Key and group key. The global key shared will be deleted within 10 seconds after deployment, which is less than the time required to reveal information from ordinary sensor [11].

Two phases are necessary for SecOPP working to perform all its functionalities. Table 1 describes the notation used in this description.

3.1 Phase 1: Secure organization of the WSN and group key distribution

The base station sends a broadcast of a '*path discovery*' message encrypted by the global key to trace routing paths, this

Notation	Description
BS	Base station
N	Sensor node
G	Gradient node
*	Broadcast address
$A \| B$	Data *A* concatenated with data *B*
$MAK_k(A)$	*MAC* calculated and encrypted with a key *K*
$Enc_k(A)$	Encrypt data *A* with a key *k*

Table 1. Notation Table

message will be also used to distribute keys groups to allow a Gradient node to communicate securely with its subordinate nodes. To determine the level of the tree, a height field is added to the message:

$$\mathbf{BS \rightarrow *: None \,||\, height \,||\, Enc}_{\textit{global key}}\mathbf{(group\ key) \,||\, MAC}_{\textit{global key}}$$

A node *N* has received the '*path discovery*', rebroadcasts the message after verifying the authenticity, nonce, height, and extraction of the group key. Nodes should increment the value of the height field; generate a group key and updating the MAC. The node resaving the message should apply the same changes.

3.2 Phase 2: pair wise key distribution

After the end of '*path discovery*', a node *N* sends a '*join group request*' to communicate with its Gradient node and to share with him a pair wise key to be able to send its data.

$$\boldsymbol{N \rightarrow G\ *: None \,||\, Enc}_{\textit{groupkey}}\boldsymbol{(pairwiseKey) \,||\, MAC}_{\textit{group key}}$$

When the Gradient node receives this message, it checks the received data and it sends a confirmation message to its subordinate node.

$$\boldsymbol{GH \rightarrow N * N \,||\, GH \,||\, MAC}_{\textit{pairwise key}}$$

After phase 2 all nodes have secure link with their gradients. Despite its robustness against attacks, its scalability, its reduced cost, SecOPP in its current implementation is static. It does not allow the addition of new nodes in the network. Nodes are not sustainable when their batteries are completed they die. And for that we propose SecOPP+ to increase its longevity, make it more reliable and more robust.

4. Proposed SecOPP+ Protocol

SecOPP+ offers the addition of a dynamic scheme for adding new nodes securely in SecOPP given that it is a necessary step to increase its lifetime even for many years since life nodes is limited and the exclusion of damaged nodes outside the network is essential. Tables 2 describe the notation used in our description.

Before deployment, the new sensors are pre-loaded with the global initial key $\boldsymbol{k_{IN}}$. Each node ***Vi*** calculates after deploying its master key: $\boldsymbol{K_{Vi} = f(K_{IN}, ID_{Vi})}$, where $\boldsymbol{f}$ is a pseudo random function. The global key shared will be deleted within 10 seconds after deployment, which is less than the time required to reveal information from ordinary sensor [9]. A new node added ***U*** is preconfigured with the global key $\boldsymbol{k_{IN}}$, this node must also calculate its master key $\boldsymbol{k_u = f(K_{IN}, ID_U)}$ before the deletion of its global key.

Three steps are necessary for the integration of a new node in the network:

a) **Step 1:** A new node added **discovers** its neighbors by spreading the message **join** following:

Notation	Description
ID_i	Sensor node
k_{IN}	Global key
G	Gradient node
U	New node added
V_i	neighbor nodes
*	Broadcast address
$A \| B$	Data A concatenated with data B
$MAK_k(A)$	MAC calculated and encrypted with a key K
$Enc_k(A)$	Encrypt data A with a key

Table 2. Notation Table

$$\boldsymbol{U \rightarrow N^{*}: ID_U \| nonce_U}$$

In this message, ***U*** sends its identity to be authenticated and its nonce generated using a pseudo-random function. The node ***U*** waits for a response from each neighbor ***Vi***. This request is authenticated using the Master key $\boldsymbol{K_{Vi} = f(K_{IN}, ID_{Vi})}$.

b) Step 2: Each node has received the join message, prepares a response message as follows

$$\boldsymbol{Vi \rightarrow U : ID_{Vi} \| Enc_{kVi}(height_{Vi} \| nonce_U \| nonce_{Vi} \| pairwisekey\ k_{U,Vi}) \| MAC_{kVi}(pairwisekey\ k_{U,Vi})}$$

Each node *Vi* that received the message join, prepares a response message that includes its identity $\boldsymbol{ID_{Vi}}$. The height represents the node level. It returns the $\boldsymbol{nonce_U}$ to prove that ***U*** has received the join message to limit replay attacks. A pair wise key is generated by ***Vi*** through a pseudo-random function in order to ensure secure communication between ***U*** and ***V***. These four fields height, $\boldsymbol{nonce_U}$, $\boldsymbol{nonce_{Vi}}$ and are encrypted using ***kVi***. Finally, a MAC (Message Authentication Code) field is calculated to ensure that the pairwise key has not been altered. After this stage, the global key $\boldsymbol{K_{IN}}$ will be deleted from memory for all new nodes.

c) Step 3: The node *U* sends a confirmation message to selected nodes

$$\boldsymbol{U \rightarrow Vi : ID_U \| ID_{Vi} \| nonce_{Vi} \| MAC\,pairwisekey\ k_{U,Vi}}$$

Once the pairwise key is shared, the development of the group key is easy:

$$\boldsymbol{Vi \rightarrow U : Enc_{k_{U,Vi}}(groupkey\ k_{U,Vi}) \| MAC_{k_{U,Vi}}(groupkey\ k_{Vi,U})}$$

The node ***Vi*** transmits its group key already shared with its subordinate to his new son ***U***. So it can communicate with it, and the pairwise key still a specific key to ***U*** that can through it send its data to its parent *Vi*. If the node *Vi* is not a leaf node then it must generate its group key using a pseudo-random function.

d) Step 4: A new added node select its parent Vi based on this criteria

After sending join message by node ***U*** to join the network, neighboring nodes send a unicast response. A node ***U*** choose the node ***V*** the most closely to the base station, it has the minimum height. If we have multiple nodes of the same minimum height, ***U*** establishes a secure communication link with all these nodes.

Step 5: e) Establishment of brothers and son key for a new node added

Other messages received by ***U*** (response message from ***Vi*** after receiving join message from ***U***) will be considered. If the height of node is greater or equal to the height of ***U***, then ***U*** must develop a secure link with this node and becomes its new gradient (father) or brother. If its height is greater than ***U*** its a new gradient else it's a brother. For this, ***U*** generates a group key and sends it to its new subordinate or brother identified by the pair wise key generated by ***V*** in the response message to the join message sent by ***U***.

$$U \rightarrow Vi : ID_U \| Enc_{pairwisekey\ k_{U,Vi}}(nonce_U, height_U, groupkey\ k_{U,Vi}) \| MAC_{pairwisekey\ k_{U,Vi}}$$

$$(groupkey\ k_{U,Vi})$$

Node ***Vi*** sends a confirmation message after verification of ***U*** identity, it height and it nonce *u*

$$Vi \rightarrow U : ID_{Vi} \| ID_U \| nonce_U \| MAC_{groupkey\ k_{Vi,U}}$$

Now, node ***U*** is successfully added in the network. It can communicate securely with its gradient. It can be also a father or brother of other nodes and it can communicate with them safely.

5. Simulation Results

To evaluate functionalities of the proposed key distribution scheme, simulations are made with TOSSIM under TinyOS using NesC language[12][13]. We have created a topology of 25 nodes to assess the performance of our scheme. Using a larger topology of 100 nodes or more will give the same results if the number of neighboring nodes for each node remains the same.

5.1 Links establishment

The first step to do is to add new nodes in the network and to make secure communication links with the old nodes to integrate these new nodes in the network. Once links are established, we succeeded to add new nodes in the network.

Figure 1 shows the degree of linkages in functions of number of new nodes added to the network. The horizontal axis in this figure shows the percentage of the number of new nodes added in percent relative to the total number of nodes in the network. For example, 20% is the addition of five new nodes in our network of 25 nodes.

We can see that the connectivity of new nodes with network is good, if we add a number of nodes equal to half of the network, we have connectivity approximately equal to 80%.

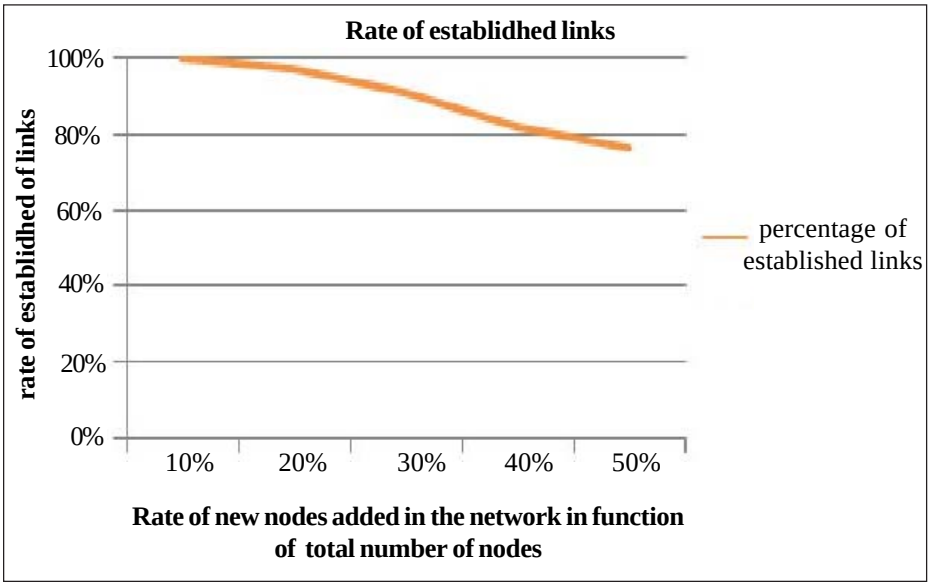

Figure 1. Percentage of establishing links for new nodes in the network

5.2 Key establishment

Once the links are established and new nodes are integrated into the network, we establish the security key to make sure that our scheme is secure and be sure of its efficiency, robustness and scalability.

Simulation results for sharing pair wise key and group key with respect to the number of linkages presented in Figure 2 give satisfactory results. Looking at the simulation results, we find that when we add a number of nodes equal to the half of total number of nodes in the network, we share about 68% of pair wise key and 65% of key groups.

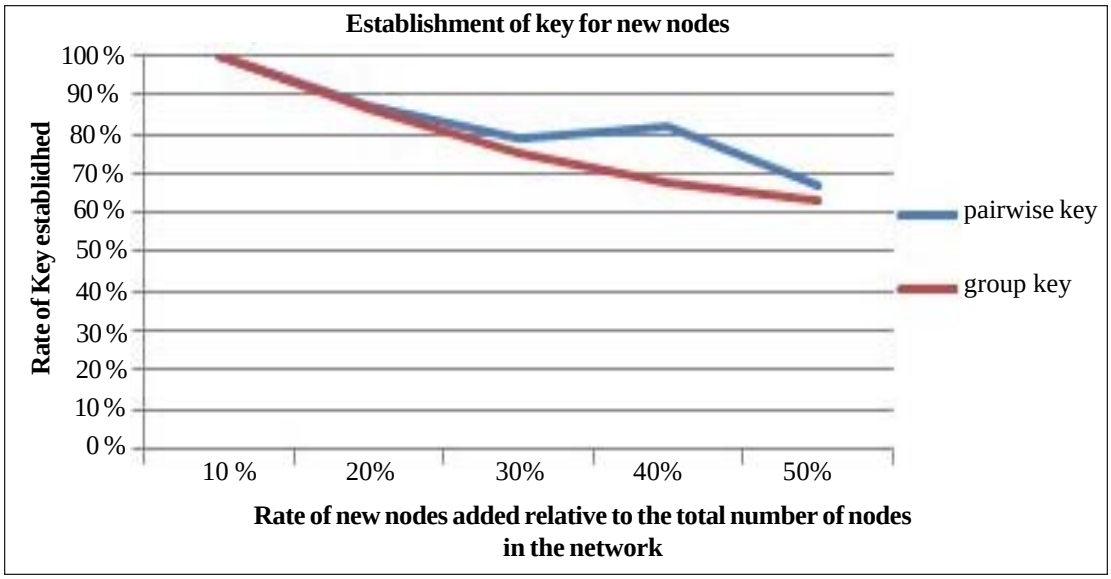

Figure 2. Percentage of key establishment for new nodes

The results are satisfactory and make the principal scheme more scalable keeping its robustness with much greater longevity.

5.3 Time required for key establishment

To be sure of the reliability of our scheme, we calculated the time required for key establishment. This key establishment depends on the time and cannot exceed 10 seconds since it is based on the global key $K_{IN,}$ which will be removed after 10 seconds even if the keys are not well established. Figure 3 represents the time required for creating key in function of the rate of added nodes in the network.

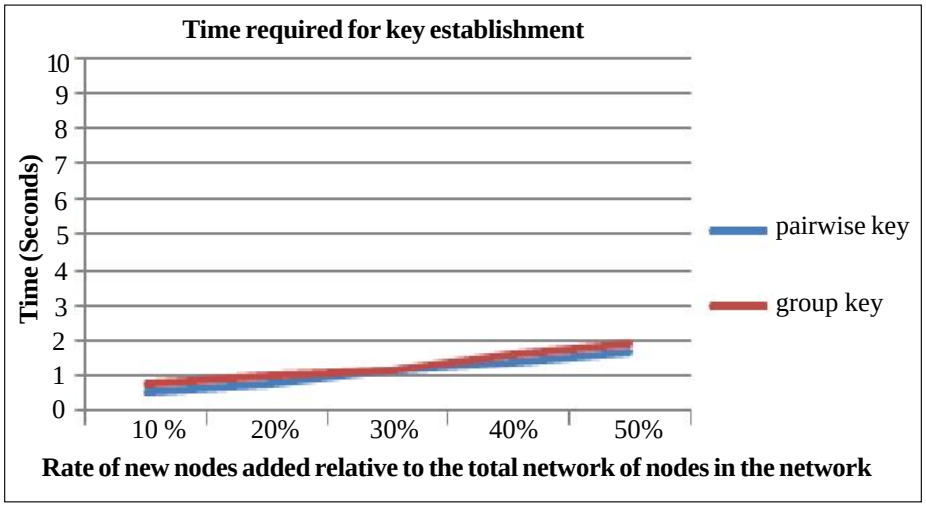

Figure 3. Time required for key establishment for new nodes

It is clear from the curve of time required for key establishment for new nodes that group key establishment takes longer time for contribution than pair wise key, which is normal since in our scheme the sharing of pair wise key is done before the division of group key. We also note that the addition of a number of nodes equal to half of network nodes number does not exceed even 2 seconds, which improves the reliability of our scheme and its security.

6. Conclusion

In this paper, we have proposed SecOPP+ protocol that enhances SecOPP performance and allows adding securely new nodes on network which increases its longevity, makes it more reliable and robust. SecOPP+ provides security services with low energy consummation. Simulation results show that new nodes are successfully integrated in the network, they are easily able to establish links with the existing network. For example, we have a connectivity of 80% when we added a number of nodes equal to the half of our network nodes and we obtained a success key establishment with a minimum time lower than 2 seconds.

References

[1] Akkaya, K., Younis, M. A Survey on Routing Protocols for Wireless Sensor Networks, Department of Computer Science and Electrical Engineering University of Maryland, Baltimore County Baltimore, *MD* 21250.

[2] Karlof, C., Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures, *Elsevier's Ad-Hoc Networks Journal*, Special Issue on Sensor Network Applications and Protocols, 1 (2–3) 293–315.

[3] Guermazi, A., Abid, M. (2010). SecOPP-a low-cost energy secure multi-hop routingprotocol for wireless sensor networks, *Communication in Wireless Environments and Ubiquitous Systems*: *New Challenges* (ICWUS) *IEEE*.

[4] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin , John Heidemann, Fabio, Silva. (2003). Directed diffusion for wireless sensor networking, *IEEE/ACM Transactions on Networking*, 11 (1), February.

[5] Zia, T., Zomaya, A. (2006). Security Issues in Wireless Sensor Networks, *International Journal of Computer Science and Information Security*, Systems and Networks Communications (ICSNC).

[6] Eschenauer, L., Gligor, V. D. (2002). A key-management scheme for distributed sensor networks, *In*: Proceedings of the 9th ACM conference on Computer and communications security. K. Elissa, *Title of paper if known, unpublished*, November.

[7] Chan, H., Perrig, A., Song, D. (2003). Random key predistribution schemes for sensor Networks, *In*: IEEE Symposium on Security and Privacy, Berkeley, California, p. 11-14, May.

[8] Jang, J., Kwon, T., Song, J. (2007). A Time-Based Key ManagemenProtocol for Wireless Sensor Networks, p. 314–328.

[9] Deng, J., Hartung, C., Han, R., Mishra, S. (2005). A Practical Study of Transitory Master Key Establishment for Wireless Sensor Networks, *In*: Proc First IEEE Int'l Conf Security and Privacy for Emerging Areas in Comm. Networks (SecureComm '05), Sept.

[10] Guermazi, J A., Abid, M. (2011). An Efficient Key Distribution Scheme to Secure Data-Centric Routing Protocols in Hierarchical Wireless Sensor Networks: Procedia Computer Science, (5) 208–215.

[11] Anderson, R., Kuhn, M. (1996). Tamper resistance cautionary note, *In*: Proceedings of the 2nd USENIX Workshop on Electronic Commerce, (96) 1–11.

[12] TinyOS: (2010). http://www.tinyos.net/.

[13] NesC: (2010). A Programming Language for Deeply Networked Systems. http://nescc.sourceforge.net/.