Int. J. Advanced Networking and Applications
Volume: 6 Issue: 5 Pages: 2435-2439 (2015) ISSN: 0975-0290

Node Legitimacy Based False Data Filtering Scheme in Wireless Sensor Networks

YaFang Liu
College of Environmental Science & Engineering, Ocean University of China, China
Email: lliuyafang@126.com
HaiPeng Qu
College of Environmental Science & Engineering, Ocean University of China, China
Email: haipeng.qu@gmail.com

ABSTRACT

False data injection attack is a serious threat to wireless sensor network. In this paper, a node legitimacy based false data filtering scheme (NLFS) is proposed. NLFS verifies not only message authentication codes (MACs) contains in reports, but also the legitimacy of nodes that endorse the report. The verification guarantees that compromised nodes from different geographical areas cannot collude to inject false data, which makes NLFS has a high tolerance of compromised nodes. In addition, NLFA only utilizes the relationships between node IDs to verify the legitimacy of nodes without other software or hardware overhead. Simulation results show that NLFS can filter 95% false reports within three hops and is resilience to an increasing number of compromised nodes.

Keywords - **compromised node, false data injection, node legitimacy, wireless sensor networks;**

Date of Submission: January 17, 2015 Date of Acceptance: February 15, 2015

1 Introduction

Wireless sensor networks (WSNs) consist of a large number of sensor nodes with limited resources. Sensor nodes are usually deployed at unattended or hostile environments. Therefore, they have a high risk of being captured and compromised. An Adversary can access all keying materials stored in compromised nodes, and utilize these compromised nodes to send bogus reports to the sink causing false alarm, wrong decision, as well as energy waste in forwarding nodes.

Some false data filtering schemes [1-13] have been proposed recently. According to the encrypting mechanisms, the related work falls into two categories: symmetric key based schemes and asymmetric key based schemes. Although asymmetric key based schemes [1-3] offer superior security, most of them are not practical, since these schemes are computation intensive and sensor nodes have limited computing power and restricted memory space. Symmetric key based schemes [3-13] share a general en-route filtering framework. In this framework, nodes first establish key-sharing relationships. When an event happens, at least t nodes collaboratively generate a report. Each node attaches its MAC to the report as an endorsement. Here t is a security threshold. Forwarding nodes utilize the key sharing relationships to verify the correctness of the MACs in the report, and reports that contain wrong MACs will be dropped by the forwarding nodes or sink. However, most of these schemes only consider the correctness of MACs, which makes that any t compromised nodes even from different geographical areas can collude to inject false reports that cannot be detected. Existing schemes [7, 8, 12] that consider the legitimacy of nodes require that every participating node has self-positioning capability. The scheme proposed in [13] utilizes relative positions of sensors to verify the legitimacy of nodes, but it requires fix path between the Sink and each cluster header, which is not practical due to frequent routing changes. In addition, it is probabilistic since it cannot guarantee that every false report will be filtered during the en-route filtering phase.

In this paper, we propose node legitimacy based false data filtering scheme (NLFS). The objectives of NLFS are summarized as follow:

First, NLFS initializes serial ID numbers for nodes in the same cluster; forwarding nodes only utilize the relationships between IDs to verify the legitimacy of nodes without other software or hardware overhead.

Second, NLFS can offer stronger filtering capacity and drop false data within certain number of hops.

Third, NLFS can defend against collaborative false data injection attack launched by compromised nodes from different geographical areas and has a high tolerance of compromised nodes.

2 System model and threat model

2.1 System model

We consider a sensor network composed of a large number of sensor nodes and these nodes are organized into clusters after deployment [14]. We assumes that the sensor nodes are deployed in high density, so that each cluster can contain at least t nodes and one of them is selected as the cluster head. When an event occurs, the cluster head aggregates readings and MACs from its cluster nodes (include itself) and generates the final reports, and then forward these reports to the sink through forwarding nodes. The sink has sufficient computing power and memory space. Fig. 1 illustrates the system model of the sensor network.

Volume: 6 Issue: 5 Pages: 2435-2439 (2015) ISSN: 0975-0290

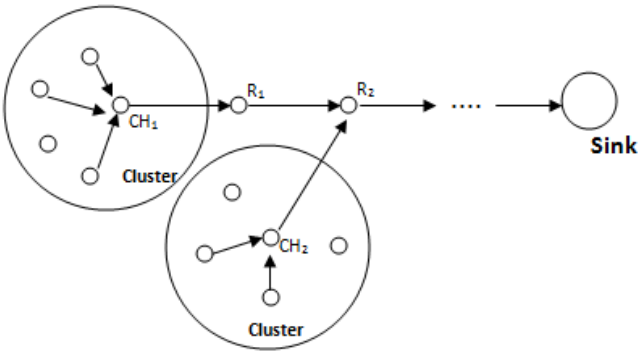

Figure 1. the system model of the sensor network: the circles outline the regions of clusters. CHs are the cluster header and Rs are the forwarding nodes.

2.2 Threat model

Due to cost constraints sensor nodes are not tamper-resistant and the attackers can compromise a node through the radio communication channel or even physically capture a node. We assume the sink employs advanced defensive measures and cannot be compromised. Adversaries have full control of compromised nodes and can utilize these nodes to launch attacks. Same as other schemes, we assume the sensor network has a short safe phase after deployment. During this phase, no node is compromised and it is safe to initialize sensor nodes and distribute authentication keys.

3 The NLFS scheme

NLFS includes four phases: initialization phase, report generation phase, en-route filtering phase and sink verification phase.

3.1 Initialization phase

Before deployment, every node is preloaded with necessary materials to establish symmetric keys. For example, if the key management scheme RSDTMK [15] is employed, nodes will be loaded with a master key and some functions. After deployment, each node gets its ID and establishes symmetric keys with its neighbors. The symmetric key is used to encrypt reports transmitted between two nodes. In this paper, A transmits a report R to B, which means R has been encrypted.

The sink maintains a global pool of authentication keys G= $\{K_{Ai}: 0 \le i \le N-1\}$ and a cluster distribution table. Table.1 shows the content of each row in the cluster distribution table. Each row represents a cluster. The IDs and keys are corresponding.

Table 1. The cluster distribution table

Cluster ID	Node ID	Key
CID_1	ID_1 ,ID_2, ID_3,ID_4	K_{A1} , K_{A2} ,K_{A3} , K_{A4}
CID_2	ID_5,ID_6,ID_7	K_{A5} , K_{A6} , K_{A7}
…	…	…
CID_i	ID_i , ID_j	K_{Ai} , K_{Aj}

3.1.1 Initialization of node ID and key

After deployment, sensor nodes are organized into clusters and each cluster generates a unique cluster ID. The Sink creates a temporary variable SynID and initializes it to 0. Then each cluster head follows steps below to obtain IDs and keys of its cluster nodes. Initially, the cluster distribution table is empty.

Step1: The cluster head sends report R_{UP} {CID, L} to the sink. CID and L denote the cluster id and the size of the cluster respectively.

Step2: When the sink receives the report R_{UP}, it performs the following operations:

1. Select L authentication keys (K_{A1}, K_{A2}, …, K_{Al}) that have not been selected by other nodes from G;
2. Synchronize SynID to SynID +L;
3. Add a record {CID; SynID, SynID-1, …, SynID-L+1; K_{A1}, K_{A2}, …, K_{Al}} to the cluster distribution table;
4. Send report R_{DOWN} {CID; SynID; K_{A1}, K_{A2}, …, K_{Al}} to the cluster.

Step3: When the cluster head receives the report R_{DOWN}, it generates L pairs of key and ID: <SynID, K_{A1}>, <SynID-1, K_{A2}>, …, <SynID-L+1, K_{Al}>, and distributes them to its cluster nodes.

Fig.2 illustrates the interactions between the sink and a cluster. After the initialization, every node gets its ID and key.

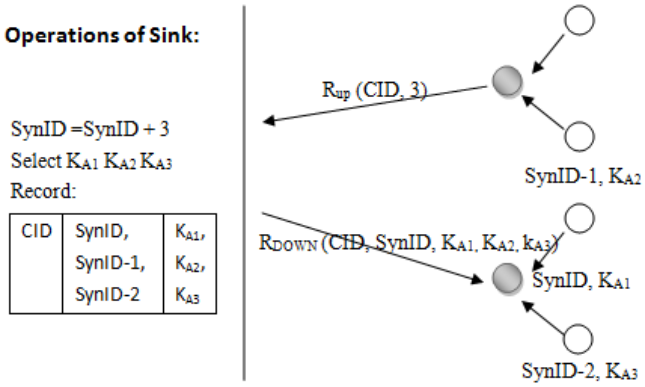

Figure 2. the interactions between the sink and a cluster containing three nodes

3.1.2 Authentication key distribution

To make NLFS adaptive to highly dynamic networks, we adopt the solution proposed in [11]. Every forwarding node forwards the key distribution report R_K to its q most possible next hop nodes.

The detailed procedures for key distribution process are as follow:

Step1: The cluster head collects IDs and authentication keys from its cluster nodes and generates report R_K {CID; L; L_S; ID_1, K_{A1}; ID_2, K_{A2}; …; ID_i, K_{Ai}}, where L_S is the number of pairs of ID and key contained in R_K and L is the size of the cluster.

Step2: The cluster head forwards R_K to its q forwarding nodes.

Volume: 6 Issue: 5 Pages: 2435-2439 (2015) ISSN: 0975-0290

Step3: When a forwarding cluster header receives R_K, it performs the following operations:

1. Check if its cluster nodes have stored IDs and keys in R_K. If does, it deletes these IDs and keys from R_K and updates L_S.
2. Create node set F. The elements of F are nodes in the forwarding cluster that have not stored any IDs and keys of nodes in the source cluster. The size of set F is L_F.
3. Compare L_F and L_S. If $L_F \leq L_S$, select L_F pairs of ID and key from R_K and distribute them to nodes in F. If $L_F > L_S$, select L_S nodes from F and distribute IDs and keys in R_K to them.
4. Delete IDs and keys that have been distributed from R_K and update L_S. If $L_S > 0$, turn to step2.

Each node in the forwarding cluster stores the ID and key distributed by its cluster header, and also stores CID and L of the source cluster. Fig.3 uses an example to show the process of key distribution where q=1.The key distribution method guarantees that no two keys stored in one node are from the same cluster.

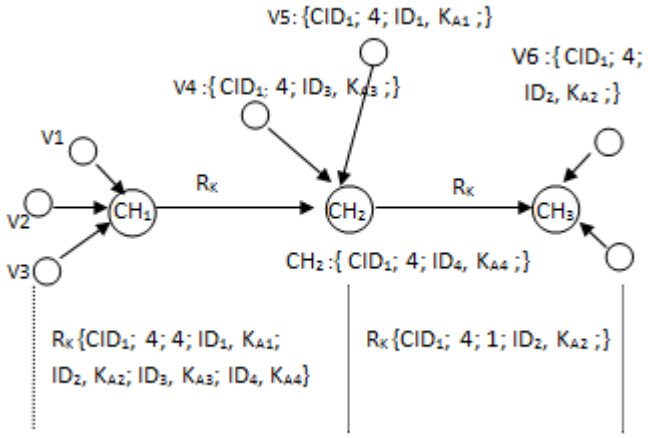

Figure 3. the distribution of keys of a cluster: the source cluster contains four nodes (V1, V2, V3, CH_1), and the report R_K sent to CH_2 contains IDs and keys of nodes in the source cluster. CH2, V5 and V4 select a pair of ID and key in R_K respectively and the rest ID and key (ID_2, K_{A2}) are sent to CH_3.

3.2 Report generation

When an event occurs, the cluster header collects sensor data from sensing nodes in its cluster. Then it generates an accurate description E of the event and forwards E back to the sensing nodes. If the sensing node agrees on E, it computes a MAC over E using its authentication key and then sends the MAC and its ID to the Cluster header. The report R the Cluster header finally generated is as follow:

$$R :\{ E; CID; ID_1, ID_2, \ldots, ID_t; M_1, M_2, \ldots, M_t\}$$

During the en-route filtering phase, if the correctness of a MAC in R is confirmed by a forwarding node F, the MAC and its corresponding ID will be removed from R, and a new MAC computed by node F and ID, CID of node F will be added to R. The report R generated by a forwarding node F is as follow:

$$R: \{E; CID_1; ID_1,\ldots, ID_n; M_1,\ldots, M_n; CID_F; ID_F; M_F\}$$

To provide high security, NLFS requires that at least r ($0< r \leq t$) nodes in R are from the same cluster.

3.3 En-route filtering

NLFS initializes serial ID numbers for nodes in the same cluster. Therefore if the size of a cluster C is L, any two nodes M, N in C should satisfy the following inequality:

$$|ID_M - ID_N| < L$$

When a forwarding node F receives report R, it mainly performs the following verifications:

Node Legitimacy verification: For any node S that endorses the report R, if F has the information of a node P whose cluster ID is the same as S', it computes $|ID_P - ID_S|$. If $|ID_P - ID_S| < L$, we consider node S is legitimate, where L is size of the cluster which p belongs to.

MAC verification: If node F stores a key that belongs to a node in report R, it computes a new MAC over E using this key. If the new MAC is the same as the corresponding MAC in R, we consider the MAC in R is valid.

When a forwarding Cluster header (CH) receives report R, it distributes R to its cluster nodes. Each cluster node F then performs the following operations:

Step1: Check if R is in correct format and contains at least t different IDs. Inform the CH to drop R otherwise.

Step2: Check if R contains at least r different IDs that are in the same cluster. Inform the CH to drop R otherwise.

Step3: If node F has any of the CIDs in R, it checks the legitimacy of corresponding nodes. If all these nodes are legitimate, it sends their IDs to the CH as a report R_L $\{ID_1, \ldots, ID_i\}$. Inform the CH to drop R otherwise.

Step4: If node F has any of the IDs in R, it checks the correctness of corresponding MACs. If a MAC M_S is confirmed correct, F computes new MAC M_F using its own authentication key and sends report $R_M\{ID_S, M_S; ID_F, M_F\}$ to the CH. Inform the CH to drop R, if there are invalid MACs. If no ID in R is stored in F, F also computes MAC M_F and sends report $R_{MF}\{ID_F, M_F\}$ to the CH.

The CH collects verification reports from its cluster nodes and decides whether to forward the report or not. The detailed procedures are described as follow:

Step1: If the CH receives the message of dropping R, it drops R.

Step2: If the number of reports R_L is less than r or they do not cover all nodes in R, the CH drops R. Step 2 checks whether the legitimacy of every node endorses the report R is verified.

Step3: If the CH does not receive any reports R_M, it forwards R to next hop. Otherwise, for each report R_M, it removes ID_S, M_S from R and adds ID_F, M_F to R.

Step4: The CH checks if R meets the requirements of t and r. If not, it adds the IDs and MACs in reports R_{MF} to R and forwards R to next hop.

3.4 Sink verification

Since the sink has the cluster distribution table, it can verify all MACs and IDs in R.

Volume: 6 Issue: 5 Pages: 2435-2439 (2015) ISSN: 0975-0290

4 Security Analysis

4.1 Filtering capacity

Theorem1. In NLFS, a false report injected by t-1 compromised nodes can be dropped within a / b hops. Where a, b are the size of the largest cluster and the smallest cluster.

Proof: According to the key distribution method, only when a forwarding cluster cannot store all keys of the source cluster, it forwards the rest keys to next hop. Therefore, keys of the source cluster will be stored in clusters that are with a/ b hops from the source cluster. Thus, if totally t-1nodes are compromised, the attacker has to forge a MAC and the corresponding ID. Since NLFS requires a forwarding cluster verifies the legitimacy of all IDs in R, report with invalid ID can be dropped immediately. Even if the attacker can provide valid ID, the report with forged MAC can also be dropped within a / b hops; since the key corresponding to the valid ID must be stored in a node within a/b hops.

4.1 Compromise tolerance

In NLFS, forwarding nodes not only verify the MACs but also the legitimacy of nodes. In order to inject false reports that NLFS cannot detect, the attacker has to compromise t nodes in the same cluster. As illustrated in Fig.3 where t=3 and r =2, nodes C_1, C_2, C_3, C_4, C_5 have been compromised, a forged report generated by any three nodes of them can be filtered since they are in different clusters and the forged report cannot meet the requirement of r. Assume node S1 has also been compromised, a forged report R {e; CID_1; C_1, S_1; M_1, M_S; CID_2; C_2; M_2} is injected through C_1, where nodes C_1, S_1 are in the same cluster and the report R is in correct format . But, it can also be filtered by forwarding nodes of node C_1, since node $C_2$is not legitimate.

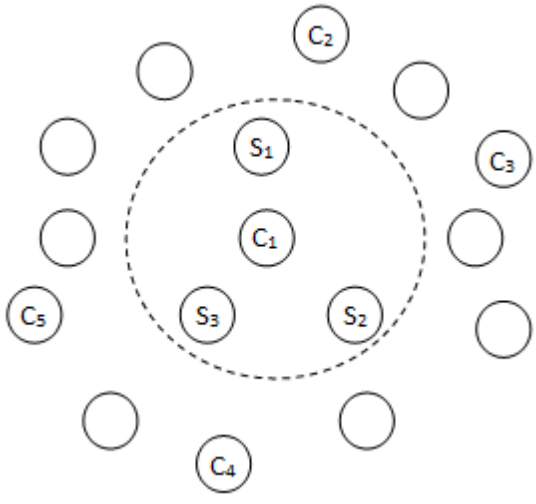

Figure 4. collaborative false data injection attack launched by compromised nodes

5 Simulation results

We study the performance of NLFS by simulation and compare it with SEF [4] and NFFS [13] in terms of filtering capacity, compromise tolerance. We simulate a 200*200 m^2 field, where 1000 nodes are randomly deployed. The transmission range of each node is 20m. The values of security parameters t and r are 5 and 3 respectively.

Fig.4 illustrates how the percentage of false reports filtered increases as the number of traveled hops grows. From Fig.4 we can see the filtering probability of each scheme increases, when the number of traveled hops increases. Furthermore, NLFS has a higher false data filtering capacity. It can drop 95% of false reports within 3 hops, while NFFS needs 10 hops and SEF needs more than 20 hops to achieve the same filtering result.

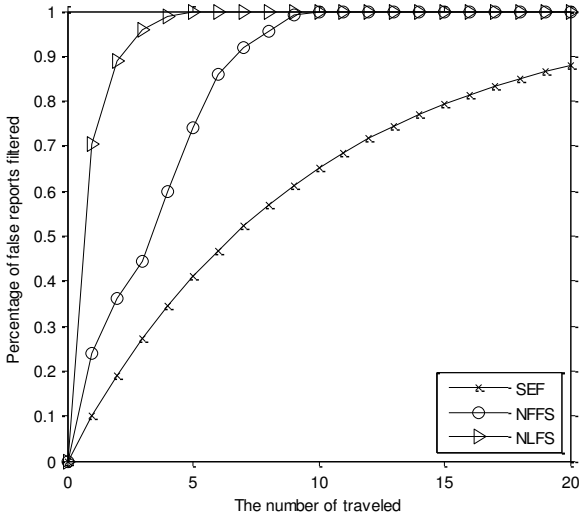

Figure 5. the percentage of false reports filtered as a function of the number of hops they traveled

Fig. 5 illustrates how many compromised nodes every scheme can tolerate. Fig.5 does not show the results of SEF, since it can tolerate less than 15 nodes. The simulation results are averaged over 1000 random tests.

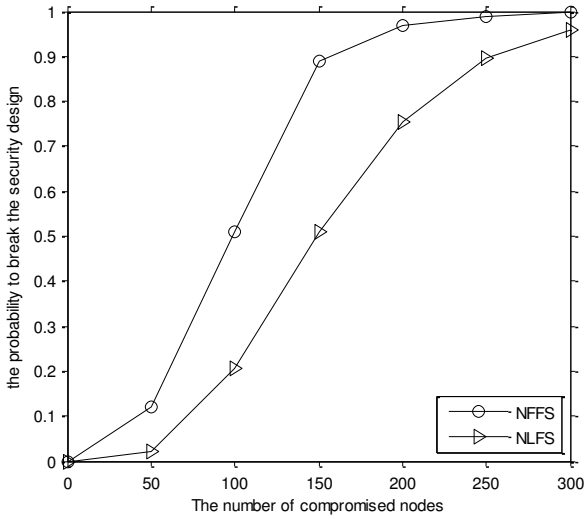

Figure 6. the probability to break NLFS as a function of the number of compromised nodes

Fig.6 illustrates how the percentage of false reports filtered increases as the number of compromised nodes grows. In NFFS, attackers have a great probability of injecting false reports that can only be filtered by the sink through nodes closer to the sink, since these nodes hold

Volume: 6 Issue: 5 Pages: 2435-2439 (2015) ISSN: 0975-0290

fewer auth-keys than nodes closer to clusters. Therefore, as the number of compromised nodes increases, the filtering capability of NFFS decreases quickly, while NLFS decreases much more slowly.

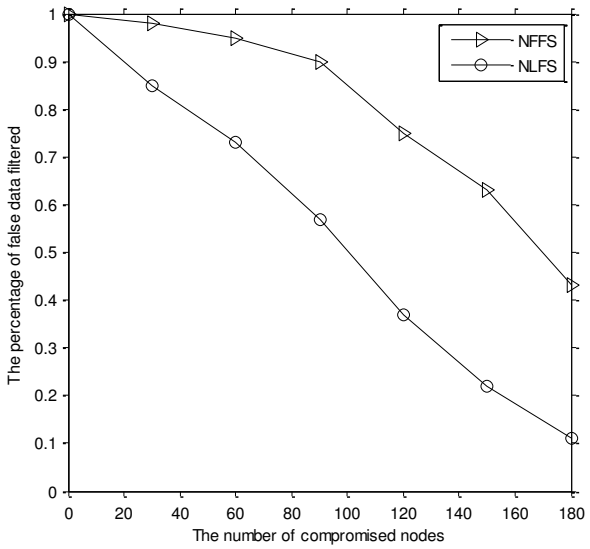

Figure 7. The percentage of false reports filtered as a function of the number of compromised nodes.

6 Conclusion

In this paper, we propose a node legitimacy based scheme to address false data injection attack. Our novel solution is to initialize serial ID numbers for nodes in the same cluster and verify the legitimacy of nodes using IDs stored in forwarding nodes. Compared with others, NLFS can filter false reports much earlier and can tolerate more compromised nodes, since it verifies not only MACs in the reports, but also the legitimacy of nodes. However, NLFS is more complicated than SEF and NFFS in the en-route filtering phase and we will further improve the scheme and make it more resilient and efficient.

REFERENCES

[1] H.D. Wang, Q. Li, Achieving robust message authentication in sensor networks: a public-key based approach. *WIRELESS NETWORKS, 16*(4), 2010, 999-1009.

[2] Y. Zhang, W. Liu, W. Lou, Location-based compromise-tolerant security mechanisms for wireless sensor networks. *IEEE Journal on Selected Areas in Communications, 24*(2), 2006, 247-260.

[3] R. Lu, X. Lin, H. Zhu, BECAN: a bandwidth-efficient cooperative authentication scheme for filtering injected false data in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems, 23*(1), 2012, 32-43.

[4] F. Ye, H. Luo, S. Lu, L. Zhang, Statistical en-route filtering of injected false data in sensor networks. *IEEE Journal on Selected Areas in Communication, 23*(4), 2005, 839-850.

[5] S. Zhu, S. Setia, S. Jajodia, An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks, *ACM Transactions on Sensor Networks(TOSN), 3*(3), 2007.

[6] M Ma, Resilience of sink filtering scheme in wireless sensor networks, *Computer Communications, 30*(1), 2006, 55-65.

[7] E. AYDAY, F. DELGOSHA , F. FEKRI, Location-aware security services for wireless sensor networks using network coding, *Proc. 26th IEEE Conf. on Computing and Communicating* , Alaska, USA, 2007, 1226−1234.

[8] K. Ren, W.J. Lou, Y.C. Zhang, LEDS: Providing location-aware end-to-end data security in wireless sensor networks, *IEEE Transactions on Mobile Computing, 7*(5), 2008, 585-598.

[9] F. Yang , X. Zhou, Q. Zhang, Multi-dimensional resilient statistical en-route filtering in wireless sensor networks, *Proc. 5th International Conf. on Grid and Pervasive Computing*, Hualien, TAIWAN, 2010, 130-139.

[10] Nghiem, T.H. Cho, A multi-path interleaved hop-by-hop en-route filtering scheme in wireless sensor networks, *Computer Communications, 33*(10), 2010, 1202-1209.

[11] Z. Yu, Y. Guan, A dynamic en-route filtering scheme for data reporting in wireless sensor networks, *IEEE/ACM Transactions on Networking, 18*(1), 2010, 150-163.

[12] Z. LIU, J. WANG, Geographical information based false report filtering scheme in wireless sensor networks, *Journal of Communications, 33*(2), 2012, 156-163.

[13] J. Wang, Z. Liu, S. Zhang and X. Zhang, Defending collaborative false data injection attacks in wireless sensor networks, *Information Sciences, 254*, 2014, 39-53.

[14] P. Kuila, P.K. Jana, A novel differential evolution based clustering algorithm for wireless sensor networks, *Applied soft computing, 25*, 2014, 414-425.

[15] F. Gandino, B. Montrucchio, M. Rebaudengo, Key management for static wireless sensor networks with node adding, *IEEE Transaction on industrial informatics, 10*(2), 2014, 1133-1143.