International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 6 | ISSN : 2456-3307

# Protected Cluster Formation Using Two-hop Conformity test and Broadcast Communication

**G. Raju [*1], V. Ramesh [2]**

[1]CSE Department, Assistant Professor, Vaagdevi Engineering College ,Warangal, Telengana, India
[2]CSE Department, Assistant Professor, Sri Indu College of Engineering & Technology, Hyderabad, Telengana, India

## ABSTRACT

In wireless sensor networks, cluster is usually adopted as a result of it facilitates the energy savings of nodes and consequently extends the network life. during this paper, we propose a novel and secure cluster formation scheme. First, our scheme generates larger sized clusters to boost the quality of clusters. Second, our scheme produces healthier clusters by evicting a lot of compromised nodes from clusters. Last, our scheme saves the energy of nodes by using broadcast communication. Simulation results prove that our scheme is more robust against compromised nodes and more energy-efficient than different scheme.

**Keywords** : Secure cluster formation; cluster membership agreement, Wireless sensor network

## I. INTRODUCTION

In wireless sensor networks, clustering is adopted to save lots of energy of nodes and consequently to increase the network time period. There are 2 strategies to create a cluster structure for a network. Within the initial method, cluster leaders are initial selected supported a particular metric like the identifier, residual energy, network property, and so on. Then different nodes verify that cluster they belong to supported a particular criterion. Some old-fashioned schemes [1-3] represent this class. during this technique, a compromised node will cheat different nodes as if it's most fitted for the leader in terms of a particular metric. within the second method, all nodes initial type clusters by sharing identical cluster membership and be part of to at least one of these clusters. Then every cluster will elect its leader that is termed CH (Cluster Head) once required. As a result of this technique tries to exclude some compromised nodes throughout the cluster formation; it's safer than the primary technique. Some recent work for secure clump [4-5] represents this class and our theme also takes this approach. Additionally, we only specialize in the secure cluster formation and don't upset the CH election problem.

In this paper, our contribution is as follows. First, our theme creates larger sized clusters within which any 2 nodes will communicate through at the most two-hop transmission power in an exceedingly secure manner. Second, our theme expels a lot of compromised nodes from the cluster using two-hop conformity check and uneven cryptography. Last, our theme reduces the energy consumption of nodes throughout cluster formation by utilizing energy-efficient broadcast communication.

The organization of this paper is as follows. Section two describes previous work regarding secure cluster formation. Section three presents the network and threat model and also the careful description of our theme is bestowed in Section four.

## II. LITERATURE SURVEY

Heinzelman et al. proposed LEACH (Low-Energy Adaptive Clustering Hierarchy) where nodes become a CH alternately and other nodes join a nearby cluster to form clusters [1]. Even though F-LEACH [2] and Sec-LEACH [3] were proposed to protect the cluster formation of LEACH, they do not provide a complete

solution. They cannot prevent compromised nodes from declaring themselves as cluster heads and from joining in any cluster head.

Rifà-Pous et al. proposed a cluster formation method which employs an asymmetric cryptography to build a consensus on the cluster membership [4]. However, this scheme assumes that all nodes conform to the protocol, so any deviation from the protocol can easily break the consensus.

Nishimura et al. proposed a scheme where all nodes give a trust value to each CH candidate and most trusted nodes become a CH [5]. Then other nodes join a nearby cluster to form clusters in the network. However, this scheme causes a lot of communication overhead to build a trust evaluation system. Moreover, this scheme burdens a few CH nodes with a lot of normal nodes for a long time. Consequently, this scheme is not suitable for resource-constrained sensor networks.

Sun et al. proposed a complete solution which checks the protocol obedience of nodes to discriminate compromised nodes from clusters [6]. First, all nodes are grouped into cliques where all members are directly connected. Then, each node investigates if all members agree with the clique membership. If a node finds an inconsistency, it checks whether other nodes in the clique conform to the protocol to identify and remove compromised nodes. However, the scheme enlarges the number of clusters because it initially generates only small sized clusters (i.e. cliques) and splits a cluster whenever a suspicious node is found in the cluster. Furthermore, its communication overhead is quite a lot because it utilizes a lot of uncast communication during the conformity check.

## III. NETWORK AND THREAT MODEL

### A. Network Model

After the deployment of nodes, clusters are formed to facilitate the energy-efficient TDMA communication. After the cluster formation, the network operation is divided into rounds and each round consists of three phases as shown in Figure 1. They are synchronization phase, secure CH election phase, and data aggregation and forward phase. In this paper, we only cover how to protect the cluster formation phase and the phase is divided into three steps. In the first step, each cluster is given a DSSS (Direct Sequence Spread Spectrum) code to avoid the inter-cluster interference when the cluster registers the members into the sink. For instance, the first cluster to register is assigned the first code on a predefined list; the second cluster to register is assigned the second code, and so on. Note that a node which is called separator initiates this registration process. To avoid the intra-cluster interference in a cluster using the same code, the sink settles the TDMA schedule of members in a cluster and distributes the schedule to the members. In the second step, each cluster merges normal members (i.e. non-separator nodes) into the cluster and verifies the merger. In the third step, each cluster merges the cluster separator into the cluster and verifies the merger.

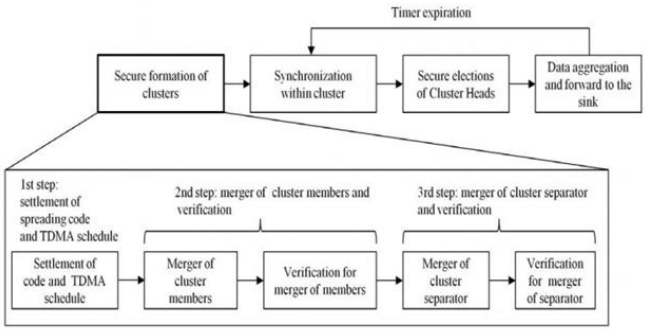



**Figure 1.** Network operation of clustered sensor networks

### B. Assumptions

First, we assume that each node identifies its neighbors correctly via a wormhole prevention scheme. Second, each node can perform lightweight public key operations such as ECC signature and verification. It has been proven in [7] that energy-constrained sensors can perform the public key operations well. Third, the sink plays the role of CA (Certification Authority) for the network and each node holds the public key of the CA.

### C. Threat Model

Even though there are so many attacks available on sensor networks, we only focus on two attacks available on a cluster formation protocol. In this paper, the attacker means the compromised nodes which are controlled by attackers. First, an attacker can deliver a message to some nodes while avoiding the delivery to the other nodes using directional antennas. So, we call this attack selective transmission attack hereafter. In addition, an attacker can completely suppress the delivery of a message. So, we call this attack silent

attack hereafter. When a cluster suffers from these attacks, some nodes in the same cluster have a different view on the cluster membership. This separates a cluster into multiple ones and the average size of clusters (that is, average number of members) shrinks. The size of clusters significantly affects the probability that a compromised node is elected as a CH in case of random election. Assume that there are two clusters and one has a few members and the other has more members. When a CH is elected randomly and compromised nodes conform to the election protocol, the cluster with a few members is prone to elect a compromised node as a CH. So, we need to reduce the number of generated clusters in the cluster formation phase.

## IV. CLUSTER FORMATION USING TWO-HOP CONFORMITY CHECK

### A. Settlement of Code and TDMA Schedule

After the deployment, each node exchanges its signed ID and certificate with neighbors. Then, a lowest ID node which is called cluster separator reports its neighbors as members to the sink using the member report message. In Figure 2(a), cluster separator 1 constructs a member report message by listing the signed IDs and signing the list with its private key. Then the cluster separator 1 sends the member report to the sink. Note that other separators like 4 and 5 follow the same procedure. The sink verifies the signatures of members using the corresponding public keys. Then the sink settles the members of the cluster and TDMA schedule of the cluster. Last, the sink signs the schedule with its private key and distributes the schedule via the separator as shown in Figure 2(b). Because all nodes have the sink's public key, they can get their TDMA schedule in the cluster. Even though each node only transmits its message in only its designated slots, they do not sleep during the cluster formation phase to listen to the messages from other nodes.

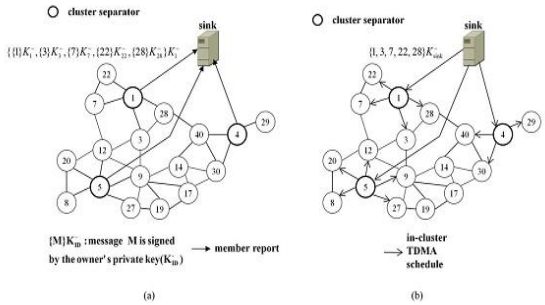



**Figure 2.** Settlement of spreading code and TDMA schedule. (a) Member report of cluster separators (b) TDMA schedule distribution of sink

### B. Merger of Cluster Members and Verification 1) Merger of Cluster Members

In Figure 3(a), nodes 1, 4, and 5 broadcast a cluster separator message to confirm a cluster border at the beginning of the second step. The cluster separator message consists of the type and the separator's ID which is signed by the separator's private key. If a node receives a cluster separator message, it verifies the signed ID using the separator's public key. If the verification succeeds, it joins the cluster and notifies other nodes of its join using the cluster response message. The cluster response message consists of the type, the separator ID, and signed ID received from the separator. A cluster response message proves that the sender exists under the jurisdiction of the same separator. If a node receives a cluster response message and it is not a duplicate message, it rebroadcasts the message. So, if there is no attack in a cluster, all members in the cluster have the same list of cluster response messages (i.e. same membership). However, a cluster separator (i.e. 5) might prevent the membership agreement by selectively transmitting its cluster separator message as shown in Figure 3(a). Node 5 does not send its cluster separator message to nodes 9 and 27 to exclude them from the cluster. Figure 3(b) shows that each node receiving a cluster separator message broadcasts a cluster response message.

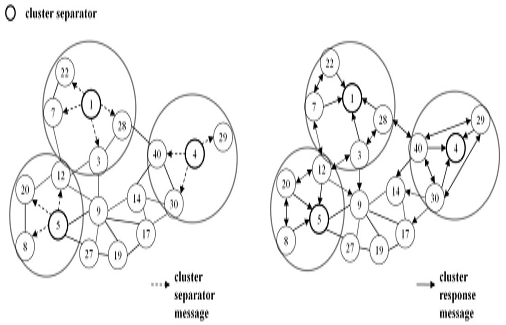



**Figure 3.** Merger of cluster members and verification. (a) Broadcast of cluster separator message (b) response to the cluster separator message

**Verification for the Merger of Cluster Members**

Each node checks if there are some deviations during the exchange of separator and response messages. If a

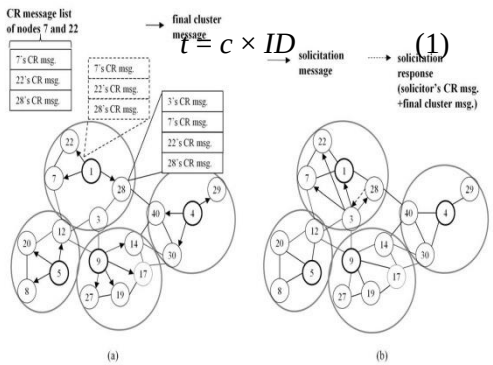
node recognizes some deviations, it employs the following countermeasures.

A malicious node may void rebroadcasting the message to disturb the agreement of cluster membership. Node 4 might carry out such an attack to hide nodes 30 and 40 from 29 and vice versa. Because the nodes 29, 30, and 40 know their cluster members thanks to the in-cluster TDMA schedule, they can easily recognize such an attack. To defeat this kind of attack, node 29 transmits its cluster response message with two hop transmission powers because it does not receive a cluster response message from any two hop neighbor. Now, nodes 30 and 40 register the node 29 into their member list and broadcast their own cluster response message with two hop transmission power. Node 29 also registers the nodes 30 and 40 into its member list.

A malicious separator may send its cluster separator message to just a part of members to exclude some members. Node 5 invokes such an attack in Figure 3(a). So, the nodes 9 and 27 cannot receive node 5's separator message. Besides, nodes 8, 12, and 20 broadcast their cluster response message with two hop transmission power since they do not receive cluster response messages from nodes 9 and 27. So, nodes 9 and 27 recognize that node 5 selectively transmits its separator message. In this case, they have two choices. First, the non-receivers 9 and 27 can ask other members to hand the 5's separator message. However, other members cannot assure whether node 5 is malicious or node 9 and 27 are lying. Second, the non-receivers 9 and 27 can leave the node 5's cluster and remove the node 5 from its neighbors to make a new cluster. We choose the second measure.

After the exchange of separator message and response Message, nodes which belong to no clusters wait for a specific amount of time $t$ as in (1) where $c$ is a little constant.

$$t = c \times ID \quad (1)$$




After the timer expires, they check if they are assigned a spreading code and a TDMA schedule. If not, they assign a spreading code by reporting their neighbors to the sink and the sink settles their TDMA schedule and distributes it to all members in the cluster. For example, in Figure 4(a), because node 9 waits for just 9$c$ time unit which is shortest among neighbors, it first gets the chance to become a separator. Then, the nodes 9, 14, 17, 19, and 27 make a new cluster by exchanging the cluster separator message and the cluster response messages as shown in Figure 4(b).

1. Merger of Cluster Separator and Verification
2. Merger of Cluster Separator

Now, we merge the cluster separator into the cluster. First, cluster separators like 1, 4, 5, and 9 broadcast a final cluster message using the received cluster response messages as shown in Figure 5(a). For the sake of simplicity, we only focus on the merger of cluster separator 1 in Figure 5. The final cluster message consists of the type and the list of received cluster response messages. The cluster separator signs the message using its private key before transmitting it. Upon receiving a final cluster message, the receiver verifies the signature and compares the list of cluster response messages with its own list. If they are exactly same, the receiver merges the separator into the cluster. Otherwise, the receiver discards the message.

**Verification for Merger of Cluster Separator**

After the merger of a cluster separator, each node checks if the cluster separator deviates from the protocol. If a deviation is identified, each node employs the following countermeasure.

We assume that the cluster separator 1 avoided rebroadcasting the cluster response message of node 3 in the previous step to exclude it from the cluster. As shown in Figure 5(a), the separator 1 broadcasts its final cluster message including the received cluster response messages. Of course, the separator 1 misses 3's cluster response message to fool other nodes and does not send the final cluster message to the node 3. Receivers 7 and 22 compare the list of cluster response messages with their own list. Because nodes 7 and 22 find that they are exactly same, they merge the node 1 into their member list. However, the node 28 does nothing because it identifies the contradiction of the

two lists. Meanwhile, node 3 identifies the node 1's deviation from the protocol. So, node 3 broadcasts a solicitation message with two hop transmission power to obtain a proof that it broadcasted its cluster response message as shown in Figure 5(b). Because the receiver 28 has 3's cluster response message, it first signs 3's cluster response message by its private key and transmits the signed message along with 1's final cluster message. The signed message and the final cluster message constitute a solicitation response message.

When the node 3 receives the solicitation response, it finds any unknown node in the final cluster message. If an unknown node is found, it registers the unknown node into the member list. Next, node 3 checks whether 1's final cluster message includes its cluster response message or not. If 1's final cluster message misses its cluster response message, it proves the node 1's deviation from the protocol. So, node 3 reports the node 1 as an attacker using a two hop broadcast message The attacker report includes 3's cluster response message which is signed by 28's private key and 28's certificate. Recall that all nodes exchange their certificate with neighbors. Receivers 7, 22, and 28 verify the signature. If the verification succeeds, they remove the node 1 from the cluster member list and the neighbor list. The reason why we remove node 1 from the cluster member is that node 1 is most responsible for the non-reception of 3's cluster response message at nodes 7 and 22. Nodes 7 and 22 register the node 3 into their member list because they find a new normal node.

## V. CONCLUSION

In this paper, we proposed a secure cluster formation theme that generates larger sized clusters and preserves them well. By using the two-hop conformity check, our scheme evicts compromised nodes well and raises the quality of clusters. Besides, employment of broadcast communication saves the energy of nodes throughout the cluster formation. Our simulation results show that our scheme expels additional compromised nodes from clusters than Sun's scheme whereas it reduces the amount of clusters within the network. Alternative simulation results show that our theme raises the quality of clusters and however is additional energy-efficient than Sun's theme. Our scheme may be usefully utilized for initial cluster formation in a cluster based routing protocol.

## VI. REFERENCES


[1]. W. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless micro sensor networks," IEEE Trans. Wireless Communications, vol. 1, no. 4, pp. 660-670, 2002

[2]. A. C. Ferreira, M.A. Vilaca, L. B. Oliveira, E. Habib, H.C. Wong, and A.A. Loureiro, "On the security of cluster-based communication protocols for wireless sensor networks," Proc. of 4th IEEE Int'l Conf. on Networking, Reunion Island, France, Apr. 17-21, 2005

[3]. L. B. Oliveira, H.C. Wong, M. W. Bern, R. Dahab, and A.A. Loureiro, "SecLEACH-a random key distribution solution for securing clustered sensor networks," Proc. of 5th IEEE Int'l Symp. On Network Computing and Applications, Cambridge, Massachusetts, USA, Jul. 24-26, 2006

[4]. H. Rifa-Pous and J. Herrera-Joancomartí, "A Fair and Secure Cluster Formation Process for Ad Hoc Networks," Wireless Communications, vol. 56, no. 3, pp. 625-636, 2011

[5]. I. Nishimura, T. Nagase, Y. Takehana, and Y. Yoshioka, "Secure Clustering for Building Certificate Mangement Nodes in Ad-Hoc Networks," Proc. of 14th Int'l Conf. on Network-Based Information Systems (NBiS), Tirana, Albania, Sep. 07-09, 2011