Securing Cluster Formation and Cluster Head Elections in Wireless Sensor Networks

Gicheol Wang[1], and Gihwan Cho[2]

[1]Unmanned Aerial Vehicle Systems PMO, Agency for Defense Development, Korea
[2]Corresponding author, Division of Computer Science and Engineering, Chonbuk Nat'l University, Korea
gcwang@add.re.kr, ghcho@chonbuk.ac.kr

***Abstract*:** In wireless sensor networks, clustering plays a very important role for energy savings at each node because it reduces the number of transmissions through TDMA based communication. For secure clustering, it is very crucial to find compromised nodes and remove them during the initial cluster formation process. If some nodes are compromised and survive from the exclusion process of normal nodes, they can make some nodes have a different membership view in the same cluster and consequently separate a cluster into multiple clusters. To resolve these problems, we propose a robust scheme against such attacks in this paper. First, our scheme generates large sized clusters to improve the quality of clusters. Second, our scheme exploits the verification of two hop distant nodes to maintain the quality of the large sized clusters and avoids the separation of the clusters. In addition, our scheme prefers broadcast transmissions to reduce the energy consumption of nodes. We prove that our scheme generates fewer clusters and is more secure and energy-efficient than its rival scheme through security analysis and simulation results. With regard to CH election, we also propose a scheme which securely elects CHs by recognizing the compromised nodes and depriving them of their CH candidacy. To this aim, each node in a cluster calculates reputation values of other CH candidates according to their behavior and distributes them through a broadcast. Then each node extracts substantial reputation values of CH candidates using the distributed reputation values. Next, each node evaluates the substantial reputation values of other CH candidates and excludes some disreputable nodes from CH candidates. The scheme greatly improves non-manipulability and agreement property of CH election results in comparison with other rival schemes. Moreover, the scheme guarantees higher non-manipulability and agreement property than other rival schemes, even in a loss-prone environment.

***Keywords*:** Secure Clustering, Secure Cluster Head Election, Secure Cluster Formation, Wireless Sensor Networks**.**

1. Introduction

Wireless sensor networks frequently employ the cluster structure to reduce energy consumption of nodes and lengthen the network lifetime [1-3]. In addition, distributing key management duty among nodes [4-5] is one of important applications of the cluster structure. Nodes cooperatively generate the cluster structure by combining themselves and their adjacent nodes into a group which is called as a cluster. We usually employ two methods to build a cluster structure for a network. The first method selects cluster leaders based on a specific metric such as identifier, residual energy, network connectivity, and so on. Then, normal nodes determine which cluster they belong to based on a specific metric such as the distance to a CH. This kind of scheme is called as a leader-first scheme and many schemes [1-3], [6-10] fall into this category. In this kind of scheme, a compromised node can cheat other nodes as if it is most suitable for the leader in terms of such a specific metric. The second method first forms clusters by making adjacent nodes share the same cluster membership. Next, each cluster elects its leader which is called as CH (Cluster Head) to serve its cluster members. This kind of scheme is called as a cluster-first scheme and some schemes [11-13] fall into this category. Since this kind of method attempts to expel some compromised nodes during the cluster formation, it is more secure than the first method. Therefore, we also take the cluster-first approach to generate clusters in wireless sensor networks.

In order to securely generate clusters, Sun et al. proposed a scheme which employs protocol conformity check and asymmetric cryptography [12]. It easily prevents two types of attackers from disturbing the operation of the protocol. Because this scheme deals with only small sized clusters (i.e. cliques) and splits them frequently, many clusters are generated and average size of clusters is also decreased. Besides, this scheme causes a lot of communication overhead to check the protocol conformity of nodes. Although the scheme of [13] enhanced the security of [12], it is an immature protocol because it assumes an environment no collisions occur during the protocol operation.

In this article, we propose a novel cluster formation scheme to resolve above problems. First, our scheme provides a way of settling a spreading code and a TDMA (Time Division Multiple Access) schedule in a cluster to avoid inter-cluster collisions as well as intra-cluster collisions. Second, our scheme creates large sized clusters in which any two nodes can communicate through at most two hop transmission power and minimizes the separation. Third, our scheme employs two-hop conformity verification and asymmetric cryptography to preserve the clusters. Last, our scheme minimizes the unicast communication and employs broadcast communication more frequently to reduce the communication overhead.

In a cluster structure, CHs gather data from normal nodes and aggregate them to send to the sink. Therefore, attackers can maneuver the whole network by compromising all CHs in the network. To prevent this, CHs need to be changed periodically through a CH election protocol. However, in an election protocol, attackers undoubtedly attempt to manipulate CH election results and facilitate their wins through the manipulation. To prevent the attackers from fabricating CH election results, a CH election protocol should guarantee important properties such as unpredictability, non-manipulability, and agreement property of the election results. The protocols in [14-16] show that the properties can be met in an environment where only naive attackers exist. However, intelligent attackers can easily break the above properties and forge the CH election results for their benefit.

For such a reason, we also introduce a novel CH election scheme which can deal with the misbehavior of intelligent attackers. At the beginning of every election round, each CH candidate contributes their random number towards generating a common value and the common value is employed for CH election. After the contribution ends, each candidate gives direct reputation values to other candidates considering their behavior during the CH election process and distributes the direct reputation list for other candidates. When each candidate gives a direct reputation value to another candidate, it considers the frequencies of successful and unsuccessful transmissions, the time interval between the last two successful transmissions, and the time interval between the last two unsuccessful transmissions. After receiving the direct reputation lists from all candidates, each candidate can compute indirect reputation values and combined reputation lists for all candidates. Since each candidate maintains the direct reputation lists as the number of members, the combined reputation lists are also generated as the number of members. Each candidate extracts the real reputation list in which each item is the average of the combined reputation values. Each candidate can exclude some other candidates whose real reputation value is lower than the average of the real reputation list. Because the internal attackers are likely to be given a low reputation value, they are prone to be excluded from the CH candidates unless they take special actions. So, the internal attackers maximize their marks and minimize the marks of normal nodes to survive the reputation based exclusion. However, the tactics cannot cause any significant effect to our scheme as long as the normal nodes are more than the internal attackers.

2. Background

2.1 Secure Cluster Formation

Heinzelman et al. proposed LEACH(Low-Energy Adaptive Clustering Hierarchy) in which sensors have a probability of becoming a cluster head without message exchange [1]. This scheme attempted to extend the network longevity by making all nodes play a role of CH alternately. In this scheme, some nodes with a high probability declare themselves as cluster heads and other nodes join in one of them. However, since this scheme assumes no compromised nodes in the network, it has no measure to protect the cluster formation from the compromised nodes.

F-LEACH [2] was proposed to protect the cluster formation in LEACH. In this scheme, when a node declares itself as a cluster head, it employs common keys shared with the sink to request the authentication of the CH declaration to the sink. Then, the sink securely broadcasts the authenticated cluster heads using μTESLA [17]. Normal nodes join in only one authenticated cluster head. However, this scheme has no mechanism to authenticate the normal nodes which join in any cluster. To resolve this problem, Oliveira et al. proposed SecLEACH [3] in which the sink authenticates the cluster head nodes and the cluster heads authenticate the joining nodes. In F-LEACH and SecLEACH, sensors are pre-assigned some keys for authentication before their deployment. However, both F-LEACH and SecLEACH can prevent only external attackers from joining the cluster formation process. In other words, they cannot prevent internal attackers from declaring themselves as cluster heads and from joining in any cluster.

Even though many variants of LEACH [6-10] have been proposed so far, most of them [6-9] focus on balancing the energy consumption over all nodes and extending the longevity of the network. Only the scheme of [10] deals with how to securely elect a CH. However, the scheme cannot prevent a compromised node from declaring itself as a CH because it can cheat other nodes as if it has a short distance to the sink and a large amount of residual energy. Furthermore, the compromised node can declare itself as a CH more frequently than other nodes by fabricating the probability of becoming a CH for its benefit. This is because [10] has no mechanism to verify the probability of becoming a CH.

Liu proposed a cluster formation scheme in which only predetermined nodes declare themselves as cluster heads and other nodes join in any cluster directly or via a relay node [18]. Since any cluster head declaration or any cluster join is authenticated by pre-assigned polynomial share, the scheme prevent any external attacker from participating in the cluster formation. Besides, the scheme has a wormhole prevention mechanism where a node with many neighbors shutdowns itself or the sink expels those nodes from the network by inserting them into the blacklist report and broadcasting it. In this scheme, a compromised relay node can invoke a DoS (Denial of Service) attack by cutting the connection between its cluster head and its serving nodes. Furthermore, predetermined cluster heads become the compromise targets of attackers because their roles are fixed.

Sun et al. proposed a secure cluster formation scheme which checks the protocol conformity of nodes to discriminate malicious nodes from normal nodes [12]. In this scheme, a physical network is transformed into cliques and all members are directly connected to each other in a clique. After the clique formation, each node verifies that all members have the same view of the clique membership. If a normal node finds any disagreement, it performs the protocol conformity check for other nodes in the clique to identify and remove internal attackers. This scheme well finds and removes internal attackers through the protocol conformity check. However, the scheme increases the number of clusters in network because it produces only small sized clusters (i.e. cliques) and separates a cluster whenever a suspicious node is found in the cluster. Moreover, it causes a lot of communication overhead of nodes because it requires a lot of unicast communication during the protocol conformity check. Even though the scheme of [13] has improved the security of [12], it assumed that no collisions would occur during the cluster formation. This assumption cannot be easily satisfied without any special measure such as code separation and TDMA schedule assignment. However, the scheme of [13] has no such a measure.

Nishimura et al. proposed a scheme where all nodes give a trust value to each CH candidate and most trusted nodes become a CH [19]. Otherwise nodes join a nearby cluster to form clusters in the network. 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. Therefore, this scheme is not suitable for resource-constrained sensor networks.

Rifà-Pous et al. proposed a secure cluster formation scheme which is based on public key cryptography [11]. The scheme consists of three phases; cluster discovery phase, cluster head designation phase, and cluster maintenance phase. In the cluster discovery phase, nodes in a cluster attempt to have the

same view on the cluster membership with each other. In the cluster discovery phase, a cluster head is elected considering the number of neighbors and how many times it performed the cluster head role. In the cluster maintenance phase, the elected cluster heads issue an authorization certificate to each member in the cluster. However, this scheme assumes that no nodes deviate from the cluster discovery protocol. For instance, if an attacker transmits its message to a part of all nodes in the cluster discovery phase, the victims have a different view on the cluster membership. As a result, it splits a cluster into multiple ones, and the split clusters elect their cluster head respectively in the cluster head designation phase. Namely, this scheme can generate many clusters under the selective transmission attack.

2.2 Secure Cluster Head Election

Crosby et al. proposed a trust-based CH election scheme where each node gives a trust value to other nodes according to their behavior and highly trusted nodes become CHs [20]. A node's behavior is judged by counting the frequency of successful transmissions of the node and the frequency of unsuccessful transmissions of the node. Namely, the more a node succeeds its transmission, the higher reputation value the node has. When a new CH should be elected, some nodes with a high reputation value are recommended for the CH role by members and one of them is selected as a new CH by the current CH. A compromised CH can insert an innocent victim into a blacklist to take away its candidacy for CH in the cluster. That is, as the number of innocent victims rises up, a compromised node can increase its winning probability thanks to the increasing ratio of compromised nodes to candidates.

Buttyan et al. proposed a cluster head election scheme which conceals the election process from external nodes using cryptographic techniques [14]. However the concealment works for only external attackers since a compromised node can easily unveil the selection result. Moreover, the compromised node can declare itself as a CH even though it is not qualified.

Sirivianos et al. proposed the SANE (Secure Aggregator Node Election) protocol [15] in which all CH candidates in a cluster contribute to the generation of a random value and a CH is elected randomly using the random value. SANE is classified into three sub-schemes according to how to generate and distribute the random value. They are Merkle's puzzle based scheme, commitment based scheme, and seed based scheme.

In Merkle's puzzle based scheme, the current CH establishes pairwise keys with its members using the Merkle's puzzle. Then, each member generates a random value and encrypts it using the pairwise key shared with the current CH. Next, each member adds the encrypted random value to a sum variable and passes the sum variable to another member. This add and pass procedure is continued until all members add their encrypted random value to the sum variable. The last contributor distributes the final sum to all members and the current CH distributes all pairwise keys shared with members to all members in the cluster. All members transform the accumulated sum into a plain sum of random numbers using the pairwise keys. The plain sum is used as a common value for CH election.

In the commitment based scheme, each member transmits its commitment to other members in a peer-to-peer manner. Here, the commitment means the encrypted random value of the sender. For every election round, each member generates a random value, encrypts it with keys shared with other members, and sends the encrypted random values to all members respectively. Next, each member sends the original random value to other members. Receiving members verify the random values using the shared key and sum them to produce an agreed common value.

In the seed based scheme, each member generates a seed value and broadcasts it. The seed value is an initial random value which is employed to produce a new random value in every election round. For every election round, each member broadcasts an availability message. The availability message means the sender's definite intention to take part in the CH election. Upon receiving the message, members store the sender and generate a new random number of the sender using the sender's pre-received seed and the round number of election. After receiving all availability messages, all members can get the sender list and the sum of the new random numbers which is employed as a common value..

Merkle's puzzle based scheme causes a lot of communication and computation overheads for sharing the common value among members. Even though the commitment based scheme and the seed based scheme lowers those overheads, they have a common drawback. Namely, the last contributor in the generation of the common value can predict and manipulate a CH election result and even collapse the agreement property of the CH election result.

Dong et al. proposed a scheme which prevents external attackers from taking part in a CH election through its ID assignment scheme, which tightly binds a node's ID, its commitments, and its polynomial shares [16]. In the scheme, nodes which do not transmit a participation message for CH election (that is, the current round key in their key chain) or explicitly transmit a nonparticipation message are excluded from the CH candidates. The real CH is selected by randomly selecting one node among the rest of the candidates. However, an internal attacker not only can change a CH election result by avoiding the distribution of its participation message but also can generate multiple CH election results by distributing its participation message to only a subset of CH candidates. Even though this scheme has a recovery mechanism to combine multiple election results into one result, it requires the voluntary cooperation of the CH candidates. In the perspective of the attackers, since this mechanism goes against the interests of attackers, they are not going to cooperate with it. Besides, this scheme excludes nodes which do not distribute their participation message more than once from the CH candidates and never allows them to rejoin any CH election. Therefore, this scheme cannot work well in an error-prone environment such as a wireless network.

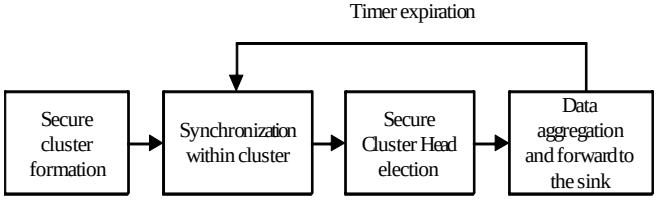

Figure 1. Network operation of clustered sensor networks

Figure 2. Three steps of secure cluster formation

Figure 3. Three steps of secure CH election

Schaffer et al. analyzed the most of the CH election schemes according to their adversarial models and desirable security properties in [21]. They categorized them into five patterns depending on how well they satisfy the desirable properties. Besides, they suggested which countermeasures can be applied to a specific scheme in order to improve the security of the scheme.

3. Network and Threat Model

3.1 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 article, we only cover the secure cluster formation phase and secure CH election phase.

First, the secure cluster formation phase is divided into three steps as shown in Figure 2. In the first step of the cluster formation, 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 as a 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.

Second, the secure CH election phase is divided into three steps as shown in Figure 3. In the first step, each member in a cluster generates a random value and distributes it.

Next, each member gives direct reputation values to other members according to their behavior and distributes the reputation list. Then, each member computes indirect reputation values and combined reputation values of members. In the second step, each member excludes some disreputable nodes from CH candidates.

In the third step, each member randomly elects a CH among the modified candidate list.

3.2 Threat Model

We only focus on two attacks available on a cluster formation protocol. In this article, an attacker means a compromised node which is controlled by attackers. First, an attacker can deliver a message to some nodes while avoiding the delivery to the other nodes using directional antennas. Therefore, this attack is called as selective transmission attack hereafter. In addition, an attacker can completely avoid the delivery of a message. Therefore, this attack is called as silence attack hereafter. When a cluster suffers from the attacks, some nodes in the cluster have a different view on the cluster membership. This splits a cluster into multiple ones and the average size of clusters (that is, average number of members) decreases. The size of clusters greatly affects the probability that a compromised node is elected as a CH on the basis of random election. Assume that there are two clusters and one has a small number of members and the other has more members. When a CH is elected randomly and compromised nodes obey the election protocol, the cluster with a small number of members is likely to elect a compromised node as a CH. Therefore, we need to reduce the number of generated clusters in the cluster formation phase.

A CH election scheme should satisfy the following properties to protect its election process. First, a CH election scheme should provide *unpredictability*. That is, it should be very difficult for a node to predict which node will be elected as a CH. Second, a CH election scheme should provide *non-manipulability*. That is, a node should not be able to modify a CH election result for its own benefit. Last, a CH election scheme should provide *agreement property*. That is, all nodes in a cluster should get the same election result.

We assume that our scheme elects a CH on the basis of a common random value. After a common random value is generated, all members agree with a CH role node using the common value. All members in a cluster contribute to the generation of the common random value by generating and distributing their own random value. Since the common random value can be generated by aggregating random values of all members in the cluster, any other node except for the last member which distributes its own random value cannot predict the common value. This means that a compromised node can predict the common value by delaying the distribution of its random value until all other members distribute their random values. Furthermore, this compromised node can violate the non-manipulability by avoiding its transmission (silence attack). That is, if the compromised node does not transmit its random value, the common value is changed and the CH election result is also changed accordingly. Generally, normal nodes adjust their transmission power so as to deliver their message to all nodes in the cluster. So, the transmission power of normal nodes depends on the maximum hop distance between nodes in the cluster. Hereafter, the maximum hop distance in the cluster is called as the cluster diameter and it is determined by the cluster formation protocol. Therefore, attackers can make multiple common values by decreasing the transmission

power when they transmit their random value to violate the agreement property of a CH election result (selective transmission attack). This is because the random value is received by only a subset of nodes in the cluster. If multiple

common values are generated for a single CH election, the CH election result is split into multiple ones.

Table 1. Function of messages in the cluster formation phase

Message type	Time when message is sent
	Function
Member Report	At the beginning of step 1
	Determination of a cluster's spreading code
In-cluster TDMA Schedule	Upon receiving a Member Report message
	Determination of a cluster's TDMA schedule
CS(Cluster Separator) message	At the beginning of step 2
	Determination of a cluster border
CR(Cluster Response) message	Upon receiving a CS message
	Request for join in a cluster
FCS(Final Cluster Separator) message	At the beginning of the step 3
	Merger of a CS node into a cluster
Solicitation message	When a victim does not receive a FCS message from its CS node
	Acquirement of witnesses for proving its legitimacy
Solicitation Response message	When a node receives a Solicitation message and holds any evidence
	Demonstration of a victim's legitimacy
Attacker Report message	When a victim confirms misbehavior of a CS node
	Exclusion of a CS node from members

4. Secure Cluster Formation and Cluster Head Election

Before describing our cluster formation scheme and cluster head election scheme, we assume the followings. First, any wormhole attack is nullified by a wormhole prevention scheme such as the scheme in [22] so that each node can identify its neighbors correctly. Second, we assume that each node can control its transmission power when they send a message so that two hop distant nodes can receive the message. Last, each node can support lightweight public key operations such as ECC (Elliptic Curve Cryptography) operations. It has been proven in [23] that sensors can well support lightweight public key operations such as ECDSA (Elliptic Curve Digital Signature Algorithm) signature generation and ECDSA verification. Last, the sink plays the role of CA (Certification Authority) for the network and each node holds the public key of the CA.

4.1 Secure Cluster Formation

We first make some definitions and describe types of messages which are employed in our scheme. After the deployment, each node signs its ID with its private key and broadcasts the signed ID with its certificate. After the verification of this message, each node can easily identify the IDs of neighbors since we assumed that a wormhole prevention scheme is working well. A lowest ID node among neighbors becomes a cluster separator. Note that a cluster separator is not a cluster head but just a protocol initiator at each step of the cluster formation phase.

At the step of settlement of code and TDMA schedule, each cluster separator reports its neighbor list as a member list to the sink. The message is called as Member Report. Through the transmission of this message, each cluster determines its spreading code. The sink verifies the member list and fixes the TDMA schedule of the members and transmits the schedule to the members. The schedule message is called as in-cluster TDMA schedule. At the step of merger of cluster members, each cluster separator broadcasts a CS(Cluster Separator) message to determine a cluster border. When a node receives the message, it joins the cluster as a member and responds to the separator using a broadcast message. The message is called as CR(Cluster Response) message. At the step of merger of cluster separator, each cluster separator broadcasts a FCS(Final Cluster Separator) message to require members to allow its join to the cluster. If a member notifies that the final cluster separator does not transmit the message to it but transmits to other nodes, it searches some evidence to claim its legitimacy by broadcasting a Solicitation message. If a node holds the evidence, it provides the evidence to the solicitor using a Solicitation Response message. When the solicitor confirms the malice of the cluster separator, the solicitor reports the cluster separator as a compromised node using an Attacker Report message. Table 1 summarizes the above descriptions.

We explain the detailed process of our cluster formation scheme using Figure 4 through Figure 13 to help the quick comprehension for our scheme.

4.1.1 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 as cluster separator reports its neighbors to the sink using the *member report* message. In Figure 4, cluster separator 1 generates a *member report* message by listing the signed IDs and signing the list with its private key. Then the

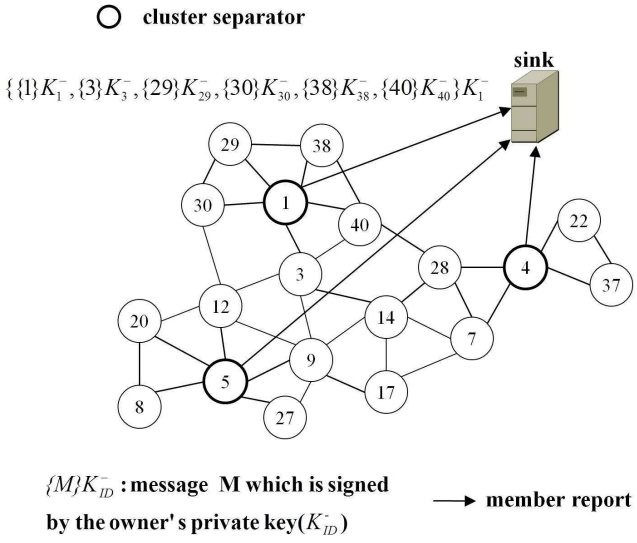

Figure 4. Member report of cluster separators

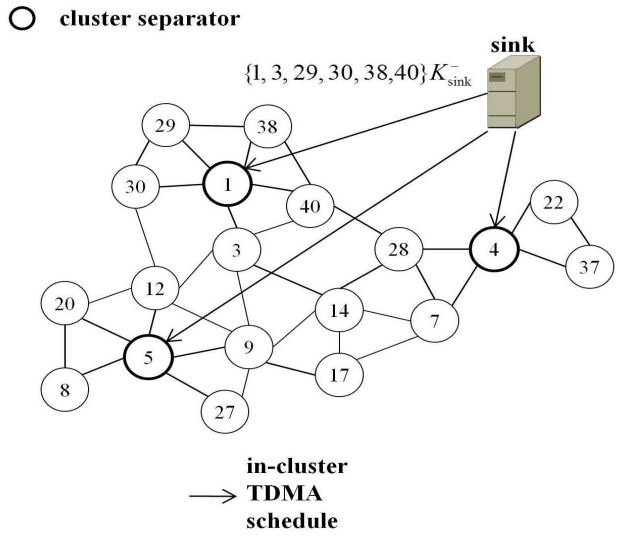

Figure 5. TDMA schedule distribution of sink

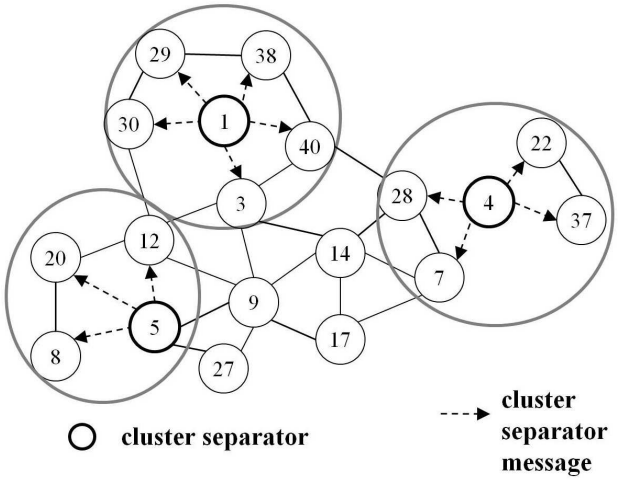

Figure 6. Broadcast of cluster separator message

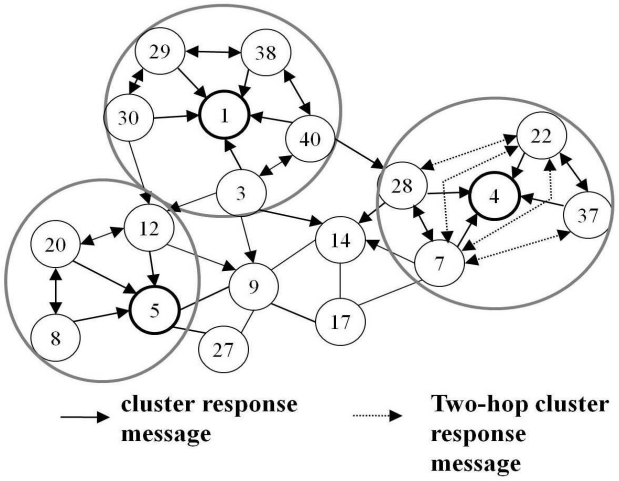

Figure 7. Response to cluster separator message

cluster separator 1 sends the *member report* to the sink. Note that other separators such as nodes 4 and 5 perform the same procedure. The sink verifies the signatures of members using the corresponding public keys. Then the sink fixes 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 to the members of the cluster via the separator as shown in Figure 5. Since all members 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 assigned slots, they do not sleep during the cluster formation phase to hear the messages from other nodes.

4.1.2 Merger of Cluster Members and Verification

In Figure 6, nodes 1, 4, and 5 broadcast a cluster separator message to determine 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. Upon receiving a cluster separator message, the receiver verifies the signed ID using the separator's public key. If the verification is successful, it joins the cluster and notifies its join to other nodes through 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 is under the jurisdiction of the same separator. If a node receives a cluster response message and it has never seen such a message, it rebroadcasts the message. Assuming no attacks 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 attempt the membership disagreement by selectively transmitting its cluster separator message as shown in Figure 6. Node 5 does not send its cluster separator message to nodes 9 and 27 to exclude them from the cluster. Figure 7 shows that each node receiving a cluster separator message broadcasts a cluster response message.

Each node checks if there are some deviations during the exchange of separator and response messages. If a node recognizes such a deviation, it employs the following countermeasures. A malicious node may avoid rebroadcasting the message to induce a cluster membership disagreement. Node 4 might carry out such an attack to veil nodes 7 and 28 from 22 and 37 and vice versa. Because they have already known their cluster members owing to the in-cluster TDMA schedule, they can easily recognize such a deviation. To defeat this kind of attack, nodes 7 and 28 transmit their cluster response message with two hop transmission power since they do not receive a cluster response message from any two hop neighbor. Now, nodes 22 and 37 register the nodes 7 and 28 into their member list and broadcast their own cluster response message with two hop transmission power. Nodes 7 and 28 also register the nodes 22 and 37 into their member list.

A malicious separator may send its cluster separator message to just a part of members to exclude some members. Node 5 invoked such an attack in Figure 6. Therefore, the nodes 9 and 27 cannot receive node 5's separator message. Besides, node 20 broadcasts its cluster response message with two hop transmission power because it does not receive cluster response messages from any two hop neighbor. Here, nodes 9 and 27 identifies that node 5 selectively transmits its separator message. In this case, there are two choices. First, the victims 9 and 27 can ask other members to pass the 5's separator

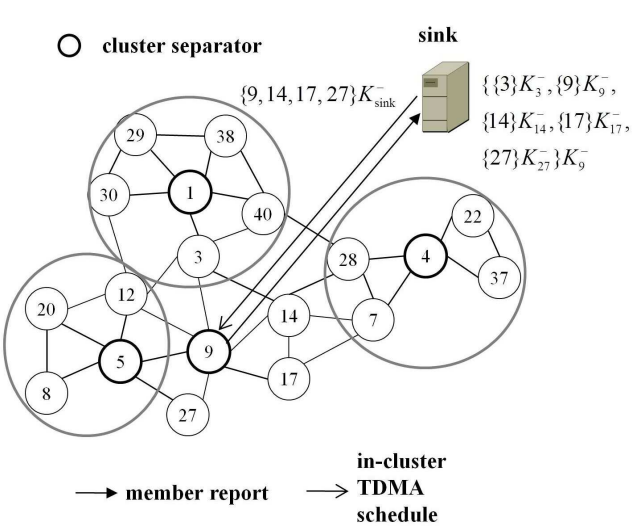

Figure 8. Settlement of spreading code and TDMA schedule

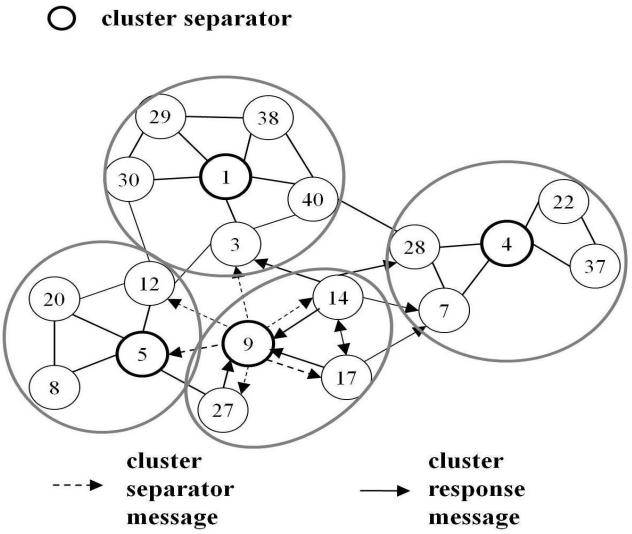

Figure 9. Merger and verification of cluster members

message. However, other members cannot assure whether node 5 deviates from the protocol or node 9 and 27 are telling a lie. Second, the victims 9 and 27 can leave the node 5's cluster and exclude the node 5 from its neighbors to make a new cluster. Regardless of taking any choice, the cluster is split into two and one of them has a malicious CS node. Therefore, we take the second choice in order to save energy consumption.

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 they still have no spreading code, they assign a spreading code by reporting their neighbors to the sink and the sink fixes their TDMA schedule and distributes it to all members in the cluster. For example, in Figure 8, because node 9 waits for 9c time unit which is shortest among neighbors, it first takes the chance to become a separator. Then, the nodes 9, 14, 17, and 27 make a new cluster by exchanging the cluster separator message and the cluster response messages as shown in Figure 9.

4.1.3 Merger of Cluster Separator and Verification

Now, each cluster separator is merged into its 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 10. For the sake of simplicity, we only concentrate on the merger of cluster separator 1 in Figure 10 through Figure 11. 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 ignores the message.

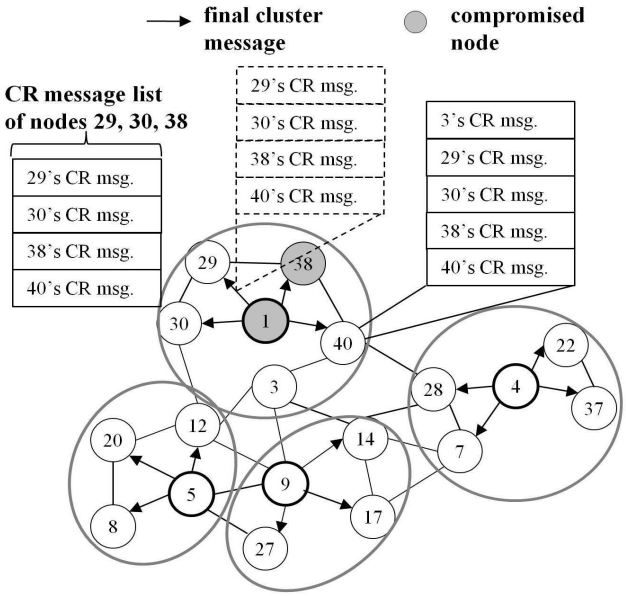

Figure 10. Merger of cluster separator

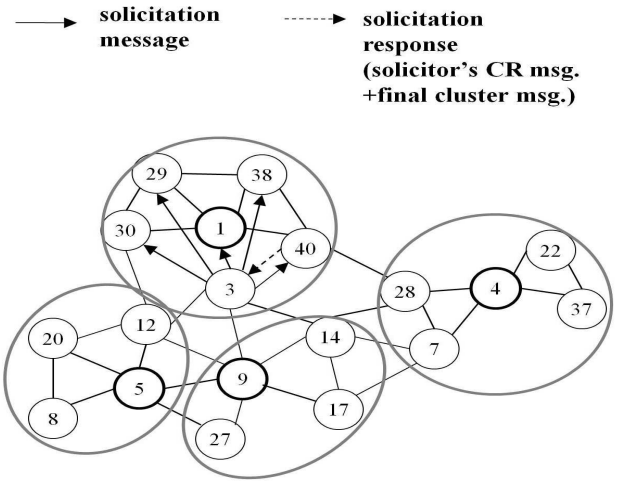

Figure 11. Verification for merger of cluster separator

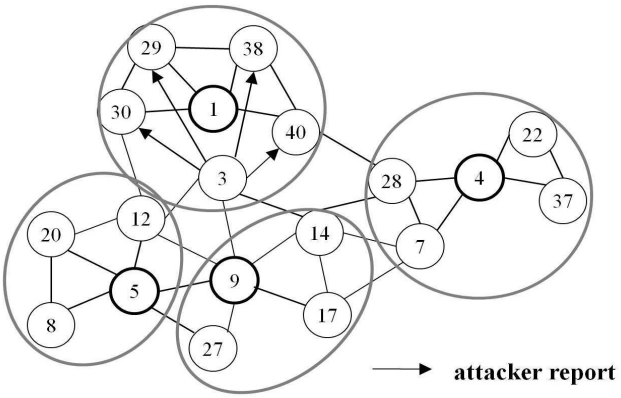

Figure 12. Attacker report distribution

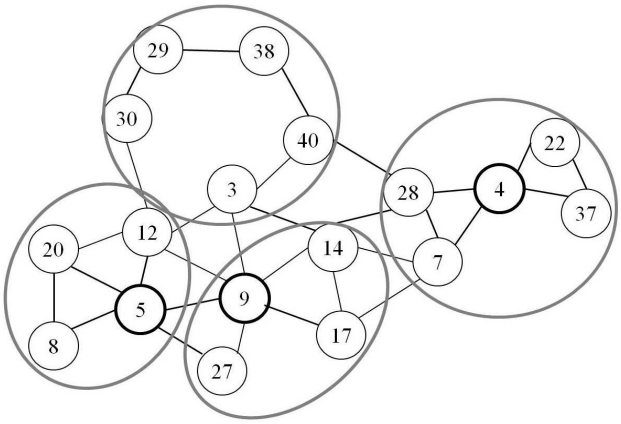

Figure 13. Final clustered network

After the merger of a cluster separator, each node checks whether the cluster separator obeys the protocol or not. If a deviation is recognized, each node employs the following countermeasure. We assume that the cluster separator 1 and node 38 avoided rebroadcasting the cluster response message of node 3 in the previous step to exclude it from the cluster. As shown in Figure 10, the separator 1 broadcasts its final cluster message including the received cluster response messages. As a matter of course, the separator 1 misses 3's cluster response message to cheat other nodes and does not transmit the final cluster message to the node 3. Receivers 29, 30, and 38 compare the received cluster response messages with their own list. Since nodes 29, 30, and 38 find that they are exactly same, they merge the node 1 into their member list. However, the node 40 ignores the message since it finds the disagreement 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 11. Because the receiver 40 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 examines whether the final cluster message includes a cluster response message of any unknown node or not. If such a 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 its cluster response message is not included in 1's cluster response message, it is an evident proof that node 1 deviated from the protocol. Therefore, node 3 reports the node 1 as an attacker using a two hop broadcast message as shown in Figure 12. The attacker report includes 3's cluster response message which is signed by 40's private key and 40's certificate. Recall that all nodes have already exchanged their certificate with neighbors. Receivers 29, 30, and 38 verify the signature. If the verification succeeds, they remove the node 1 from the cluster member list and the neighbor list. As a matter of fact, the node which receives the attacker report cannot assure that the accused node (i.e. node 1) is really responsible for the non-reception of 3's cluster response message. However, in any case, since the cluster separator (i.e. node 1 in this example) is connected to all nodes in the cluster, it is most responsible for the non-reception. Nodes 29, 30, and 38 register the node 3 into their member list because they find a new normal node whose legitimacy is guaranteed by node 40. Finally, we have a clustered sensor network like Figure 13.

4.2 Secure Cluster Head Election Scheme

Our election scheme consists of three steps. In the first step, each node generates its contribution value and broadcasts it. Then, each node gives direct reputation values to other nodes in the same cluster according to how well they conform to the protocol. Protocol conformity is judged by measuring how many times a node successfully transmits its messages during the protocol operation and how many times the node fails in its message transmission during the protocol operation. Therefore, the numbers increase incrementally and they can acquire more correct values with the lapse of time. For such a reason, we can weigh the numbers with a time factor. That is, we can consider the time interval between two successful transmissions and the time interval between two unsuccessful transmissions. Besides, we can diversify a successful transmission into various numbers according to its received signal strength at a receiver side. The less signal strength is measured at a receiver, the smaller the value given to the sender by the receiver. After assigning direct reputation values to all members in the cluster, each node broadcasts the direct reputation list. We describe the first step in subsection 4.2.1 in detail. In the second step, each node generates indirect reputation values of other members using the received direct reputation lists and computes combined values of other members to store it in the combined reputation table. Then, each node's real reputation value is generated by averaging the combined reputation values assigned by other nodes. Lastly, each node computes the average of real reputation values for all members and excludes the nodes whose real reputation value is less than the computed average from CH candidates. The details of the second step are described in the subsection 4.2.2. In the third step, all nodes share a common value by aggregating the contribution values of CH candidates and elect a CH using the common value. The third step is described in subsection 4.2.3.

4.2.1 Transmission of contribution value and reputation list

Figure 14 illustrates the flowchart for the first step in our CH election scheme. Note that Figure 14 was moved into the Appendix due to the big size of the illustration. For every CH election round, each node generates its contribution value and broadcasts it. After all nodes transmit their contribution value, each node computes direct reputation values for other members by evaluating how they conform to the cluster head election protocol. The protocol conformity can be evaluated by how many times the nodes transmit their message and by how many times they fail their message transmissions for the given transmissions. As time goes by, the numbers are likely to increase and the values become more correct. Therefore, they should be weighed by considering the time interval between two successful transmissions and the time interval between two unsuccessful transmissions. Considering the above aspects as a whole, we can produce the equation (2).

$$R_{i,j}^{D} = 1 - \frac{1}{\max[\{w_s sc_{i,j} - w_u uc_{i,j}\}, 0] + 1} \quad (2)$$

$$P_t = \frac{P_r d^4 L}{G_t G_r h_t^2 h_r^2} \quad (3)$$

$$d_r = \sqrt[4]{\frac{P_t}{E_{two\_ray\_amp} \times b}} \quad (4)$$

In (2), $sc_{i,j}$ and $uc_{i,j}$ represent the frequency of successful and unsuccessful transmissions from node *j*, respectively, which are counted by node *i.* Since a compromised node can deliver its message to only a part of nodes in the same cluster by decreasing the transmission power, frequency of the message reception can be diversified. To reflect the effect by this kind of attack, we considered the received signal strength of a message when each node counts the message reception. That is, the value of the $sc_{i,j}$ should depend on the received signal strength whenever a message is received by node *i.* If we assume that the two-ray ground reflection model is used for radio propagation, a node can extract the transmission power of a received message using the equation (3), where P_r is the received power, *d* is the Euclidean Distance, *L* is the system loss, G_t and G_r are antenna gains, and h_t and h_r are antenna heights. If a node can extract the transmission power of a received message, it can also extract the maximum reachable distance of the message (d_r) using the transmission power as shown in the equation (4), where $E_{two\_ray\_amp}$ is the energy consumed by the amplifier and *b* is the bandwidth.

$$w_s = 1 - \frac{\Delta_s}{\Delta_s + \Delta_u} \quad (5)$$

$$w_u = 1 - \frac{\Delta_u}{\Delta_u + \Delta_s} \quad (6)$$

If the maximum reachable distance of the message is the same as the cluster diameter, it means that the sender normally transmits its contribution value. Therefore, node *i* increases the $sc_{i,j}$ by one. Otherwise, the increasing value of $sc_{i,j}$ is determined by dividing the maximum reachable distance by the cluster diameter. Whenever a node *i* recognizes that it receives no messages from a node *j* during a broadcast period (that is, distribution of contribution value or distribution of direct reputation list), it increases the frequency of unsuccessful transmissions for the node (that is, $uc_{i,j}$). In addition, w_s and w_u represent the weights for the frequencies of successful and unsuccessful transmissions, respectively. They can be computed by the equations (5) and (6), where Δ_s is the time interval between the latest two successful transmissions and Δ_u is the time interval between the latest two unsuccessful transmissions.

Then, each node broadcasts the direct reputation list to share it with other members. When a node receives a direct reputation list, it checks whether there is any abnormal value in the direct reputation list or not. If an abnormal value is found, each receiver checks the number of election round. In case of the first round, it replaces all direct reputation values of the list with one and saves the list into the reputation table. Otherwise, it keeps the sender's previous reputation list in the reputation table. If a sender's direct reputation list has no illegal value, each receiver replaces the sender's previous reputation list with the currently received list. Then, each node computes the maximum reachable distance of the message to adjust the frequency of successful transmissions. That is, the frequency of successful transmissions is increased by the number that we can get by dividing the maximum reachable distance of the direct reputation list by the cluster diameter. If members in a cluster receive no messages during a specific period of time, the first step of our scheme ends.

4.2.2 Reselection of CH candidates

Figure 15 demonstrates the flowchart for the second step in our CH election scheme. We put Figure 15 into the Appendix since its size is too big to locate in the main text. First, each member checks if there is any member which avoids the transmission of the direct reputation list. If a node avoids the transmission of its direct reputation list, other nodes increase the silencer's frequency for unsuccessful transmissions by one. Then, each member changes two unsuccessful transmission times for the silencer because the latest unsuccessful transmission time should be changed to the present. Then, each member *i* computes the indirect reputation value of any other member *j* in the same cluster using the equation (7), where *m* is the number of members in the cluster. In addition, each member *i* can compute the combined reputation value for any other member *j* by combining the direct and indirect reputation values of the member *j* as shown in equation (8). After obtaining the combined reputation values, each member stores them into the combined reputation table. Lastly, each member can obtain a member's real reputation value by averaging all combined reputation values of the member as shown in equation (9). Real reputation values for all members are summed and divided by the number of members in order to obtain the average real reputation value. Each member excludes the members from CH candidates whose real reputation value is smaller than the average real reputation

value.

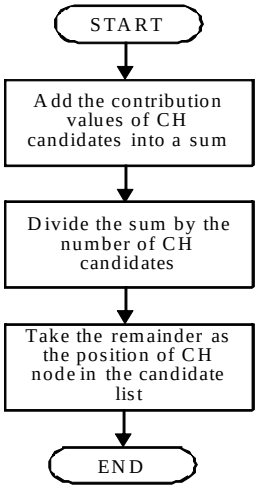

Figure 16. Flowchart for CH election

$$R_{i,j}^{IND} = \frac{\sum_{k=1}^{m-1} R_{i,k}^{D} R_{k,j}^{D}}{m-1} \quad (7)$$

$$R_{i,j} = \frac{R_{i,j}^{D} + R_{i,j}^{IND}}{2} \quad (8)$$

$$R_{real}^{i} = \frac{\sum_{j=1}^{m} R_{j,i}}{m} \quad (9)$$

4.2.3 CH Election

Figure 16 illustrates the flowchart for the third step in our CH election scheme. Each node generates a random value by summing the contribution values of survived CH candidates and obtains a remainder by dividing the random value by the number of the survived candidates. The remainder indicates the position of the CH node in the candidate list.

5. Evaluation

We exploited the simulator ns-2(version 2.27) [24], to evaluate the security and the energy-efficiency of cluster formation schemes and CH election schemes. In our simulation environment, 100 nodes were randomly deployed in a 100 meters × 100 meters area and the sink was located in the position of (50 meters, 175 meters). The simulations employed the energy consumption model in [1]. Each node used non-persistent CSMA and TDMA as its MAC protocol. When a cluster separator communicates with the sink, it employs the non-persistent CSMA to avoid the collision between separators. For intra-cluster communication, each node employs the TDMA to avoid the collision among the members in the same cluster. In addition, each cluster employs a different spreading code to avoid the inter-cluster interference. Table 2 shows the simulation parameters and their values. We ran each scheme 20 times for each number of compromised nodes and averaged the results to draw a statistical value. The network topology and the compromised nodes were changed for each run.

Table 2. Simulation parameters

Parameter	Value
Simulation area	100m.×100m.
Simulation time	1800 sec.(CH election schemes)
CH election period	30 sec. (CH election schemes)
Number of nodes	100~150
Number of compromised nodes	10~50
Compromise time distribution	3~900 sec. (CH election schemes)
Initial energy	20 Joules/battery
Energy consumption model	Energy model of [1]
Bandwidth	1 Mbps
Packet header size	25 bytes
Transmission range	25 meters
Signature Algorithm	ECDSA (Elliptic Curve Digital Signature Algorithm)-160 (cluster formation schemes)
Data Encryption and Decryption Algorithm	AES (Advanced Encryption Standard)-128 (cluster formation schemes)
Hash Algorithm	SHA-1 (cluster formation schemes)
MAC protocol	Non-persistent CSMA, TDMA

In this paper, we have proposed a secure cluster formation scheme and a secure CH election scheme respectively. Therefore, we first performed our simulations to compare and evaluate cluster formation schemes and additional simulations are followed to compare and evaluate CH election schemes. For such a reason, we provide the simulation results in the following subsections respectively.

5.1 Evaluation of Cluster Formation Schemes

In the descriptions of our cluster formation scheme (Section 4.1), we assumed that there was only one attack in a cluster to simplify the explanation about our scheme. However, in the simulation environment, multiple attacks were launched in a cluster to see how those attacks affect the security of a cluster formation scheme. Those attacks are divided into two classes. First class is a precise attack where a compromised node causes an abnormal situation and other nodes can identify or suppose which node is responsible for the situation. This class of attacks is caused by compromised cluster separators. The other class is a vague attack where a compromised node causes an abnormal situation and other nodes cannot suppose which node is responsible for the situation. This class of attacks is caused by all kinds of nodes regardless of their role.

We compare our scheme with Sun's scheme [12] since its aim and strategy are most similar with our scheme. Although other schemes [2-3], [11], [18] provide their secure cluster formation techniques, they are excluded from comparison because their methodology and attackers' aim are significantly different from our scheme. Rifà-Pous' scheme [11] is similar

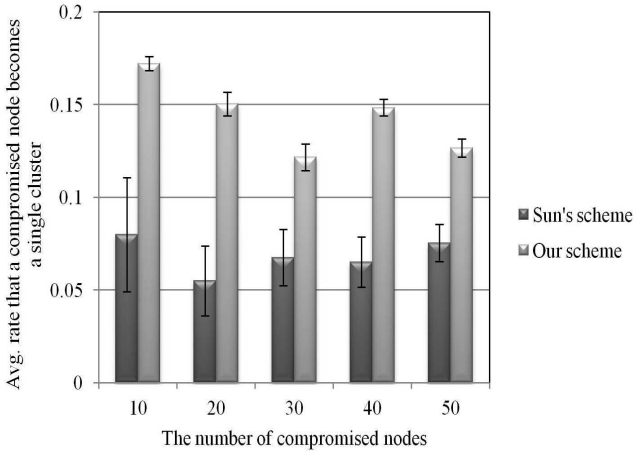

Figure 17. Avg. rate that a compromised node becomes a single cluster

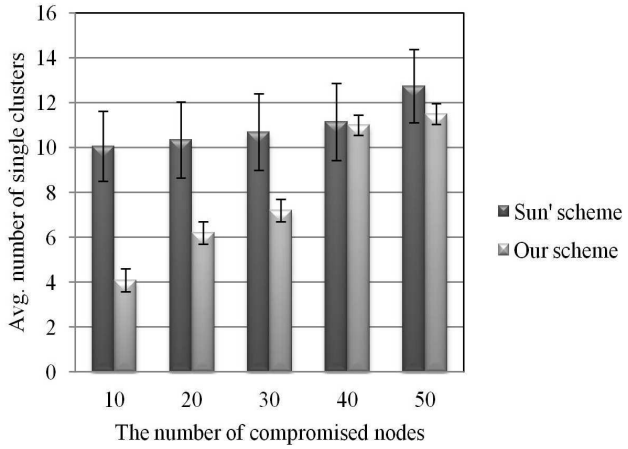

Figure 19. Avg. number of single clusters

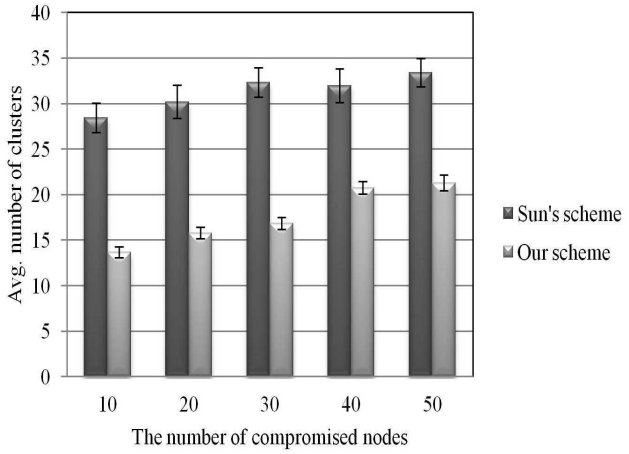

Figure 18. Avg. number of clusters in network

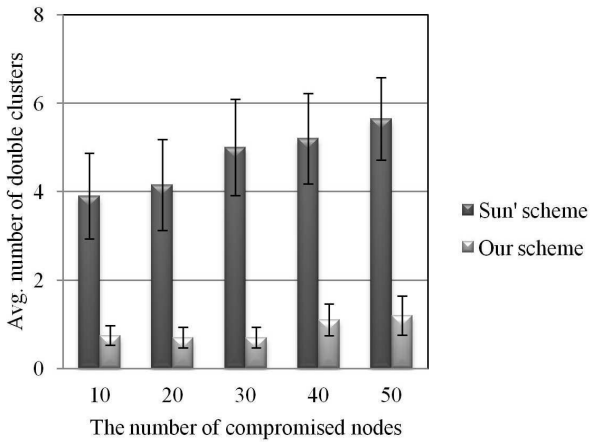

Figure 20. Avg. number of double clusters

to our scheme with regard to cluster formation methodology but it has no defense mechanism against compromised nodes which deviate from the protocol. So, comparison with Rifà-Pous' scheme is unfair. All simulation results have 95% confidence intervals. For the comparison with Sun's scheme, we developed the following metrics.

- Average rate that a compromised node becomes a single cluster: it is computed by counting the single clusters which were compromised nodes by themselves at each run and averaging the fractions. This metric represents how well a cluster formation scheme expels compromised nodes.
- Average number of clusters: it is computed by counting the number of generated clusters at each run and averaging the numbers. This metric represents the resiliency of a cluster formation scheme against the attacks invoked by compromised nodes.
- Average number of members per cluster: it is computed by dividing the sum of all members by the number of clusters at each run and averaging the fractions. This metric represents the quality of generated clusters.
- Average energy consumption per node: it is computed by summing the consumed energy of all nodes during the cluster formation and dividing the sum by the number of nodes. To get a representative value, we averaged the fractions of 20 runs. This metric represents the energy efficiency of a cluster formation scheme.

In Figure 17, we show how many compromised nodes become a single cluster in two schemes as the number of compromised nodes increases. If a compromised node becomes a single cluster, it means that the compromised node is expelled from the cluster. As shown in Figure 17, our scheme outperforms Sun's scheme. Even though the isolation rate of compromised nodes seems to be too small, its performance is rather good because most of compromised separators are isolated. Recall that normal nodes can identify only a precise attack and compromised separators can only invoke such an attack.

Figure 18 shows how many clusters two schemes generate as the number of compromised node increases. Even though both schemes increase clusters as the number of compromised nodes increases, our scheme greatly reduces the number of clusters. This is because our scheme generates larger sized clusters and suppresses the separation of the clusters as possible.

A single cluster consists of only one node and a double cluster consists of only two nodes. Single and double clusters

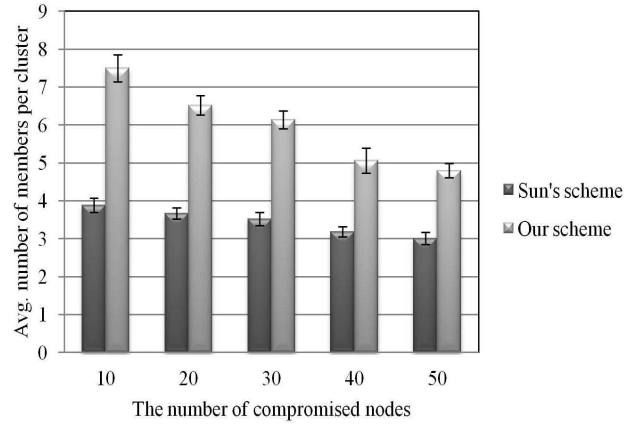

Figure 21. Avg. number of members per cluster

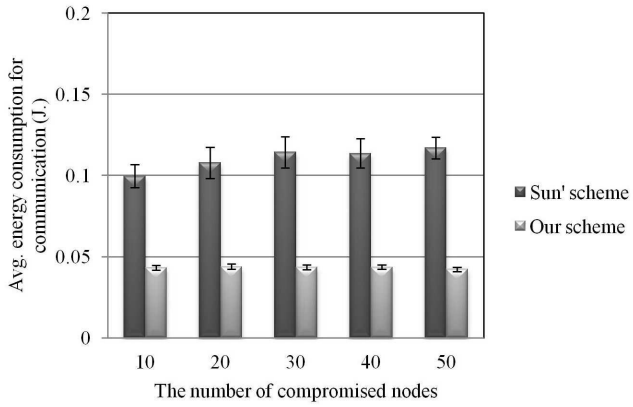

Figure 23. Avg. energy consumption for communication

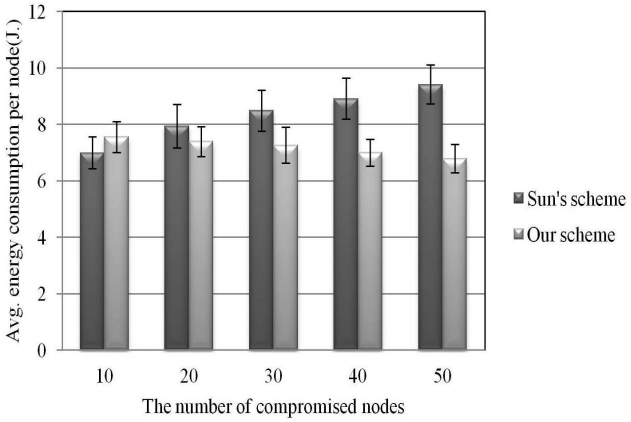

Figure 22. Avg. energy consumption per node

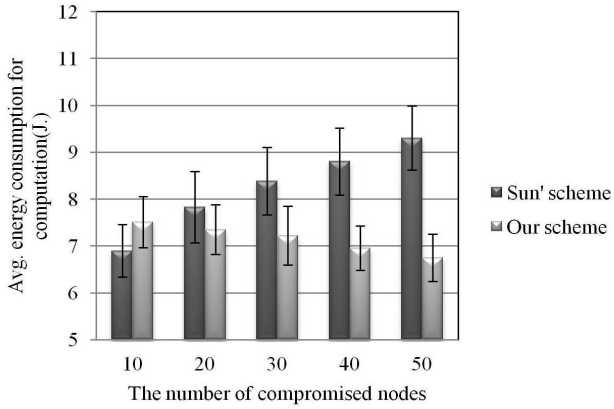

Figure 24. Avg. energy consumption for computation

are almost meaningless in the viewpoint of clustering and we call them as bad clusters. Figures 19 and 20 show how many bad clusters two schemes produce. As shown in Figures 19 and 20, Sun's scheme generates more bad clusters than our scheme. Especially, because Sun's scheme generates more single clusters than our scheme, we can say that it generates more useless clusters in comparison to our scheme.

Figure 21 shows how two schemes have an impact on the average number of members per cluster. The result shows the quality of clusters when the clusters are under attacks. Both schemes decrease the quality of clusters as the number of compromised nodes increases. Sun's scheme excludes a node whenever the node is identified as a compromised node or a suspicious node. So, it decreases the quality of clusters as compromised nodes increase. Our scheme preserves the quality of clusters higher than Sun's scheme because it separates much less nodes from clusters than Sun's scheme.

Figure 22 shows energy-efficiency of two schemes. In Sun's scheme, if a compromised node causes an inconsistency of cluster membership, a normal node starts the protocol conformity check. The normal node requests the neighboring members to send the previously received messages with a signature in unicast manner. So, if compromised nodes increase, more normal nodes start the protocol conformity check and their neighboring members should consume much more amount of energy than the case of less compromised nodes. As the number of compromised nodes increases, our scheme greatly reduces energy consumption of nodes as shown in Figure 22. This is because our scheme employs mainly broadcast transmissions.

Figure 23 shows the amount of energy each node consumed for communication during the cluster formation. As shown in Figure 23, Sun's scheme slightly increases energy consumption as the number of compromised nodes increases. Our scheme greatly decreases energy consumption as well as preserves the amount of energy consumption constantly regardless of the population of compromised nodes. Figure 24 shows the amount of energy each node consumed for computation during the protocol. Both schemes consumed much more energy for computation than that for communication. It shows that the total energy consumption of both schemes highly depends on the energy consumption for the computation of both schemes. As shown in Figure 24, Sun's scheme incrementally increases the amount of energy consumption as the number of compromised nodes increases. This is mainly caused by the increase of one hop conformity checks. Our scheme preserves the amount of energy consumption almost constantly regardless of the increase of compromised nodes. As a result, it greatly reduces the energy consumption of nodes especially under many compromised

Table 3. Qualitative comparison of cluster formation schemes

Schemes / Properties	Sun's scheme	Our scheme
Cluster configuration	Only one-hop clusters	Many two-hop clusters + a few one-hop clusters
Protocol conformity check	Conformity check of one hop neighbors	Conformity check of two hop neighbors
Communication overhead	A large number of unicast transmissions + a small number of broadcast transmissions	A large number of broadcast transmissions + a small number of unicast transmissions
Computation overhead	A large number of ECC encryptions/decryptions + a small number of hash operations	A large number of ECC encryptions/decryptions
Action when a suspicious node is found	Separate the cluster immediately	Delay the separation when a definite evidence is found

nodes.

Table 3 shows the qualitative comparison of our scheme and Sun's scheme. As shown in Table 3, Sun's scheme produces only one-hop clusters where each node is directly connected while our scheme produces many two-hop clusters where each node is connected through at most two hops. In terms of the method for protocol conformity check, Sun's scheme employs only one hop neighbors while our scheme employs two hop neighbors to maintain the two-hop clusters. This maintains the quality of clusters in our scheme high as shown in Figure 21. Sun's scheme employs a lot of unicast transmissions in the protocol operation while our scheme rather employs a lot of broadcast transmissions. This makes the big difference in energy consumption for communication between two schemes as shown in Figure 23. Besides, since our scheme expels more compromised nodes than Sun's scheme, it makes the difference in energy consumption for computation between two schemes as shown in Figure 24. Note that a node which is excluded from a cluster does not consume energy any longer. In Sun's scheme, when a normal node detects a suspicious node, the suspicious node is separated from the original cluster regardless of its legitimacy. Contrarily, our scheme delays the separation and attempts to maintain the two-hop cluster structure as possible.

To compare the memory overhead of two schemes, let Ni be the number of node i's neighbors. In Sun's scheme, because each node i should store messages from all neighbors in each step and the messages are received through four steps, its memory overhead is $4Ni$ messages. If a node i detects that a neighbor j has a different cluster membership, it should receive an extra message from j and store it to complete the verification. If so, node i's memory overhead is $4Ni+1$ messages. Let M be the number of members in a cluster. So, we have an inequality of $4Ni > M(>2Ni) > Ni$ according to the result of Figure 21. In our scheme, each node first stores Ni messages which are sent from its neighbors due to the exchange of certificates. Besides, each node i should store all CR messages sent from its members to agree on the cluster membership. So, each node's storage overhead is $M+Ni$ messages. When a normal node receives no FCS message, it requests other members to send an evidence of its legitimacy. If the node succeeds in getting the evidence, it should also store the evidence to persuade other members of the CS node's misbehavior. In this case, the node's storage overhead is $M+Ni+1$ messages. Therefore, our scheme's storage overhead is lower than Sun's scheme.

5.2 Evaluation of CH Election Schemes

We compared our CH election scheme with the seed based scheme [15], the commitment based scheme [15], and the key chain based scheme [16]. The reason why we chose these schemes for comparison is that they generate a common value and elect a CH using the common value like our scheme. Besides, they suffered from the same attacks. In a cluster, all compromised nodes invoked the same kind of attack. This was essential because the objectives of two attacks (that is, silence attack and selective transmission attack) conflict with each other. To facilitate our comparison, we developed the following metrics.

- Average number of CHs per cluster: it is computed by summing the number of CHs per cluster through all election rounds and dividing the sum by the number of election rounds. This metric represents the resiliency of a CH election scheme against the cluster split trials by compromised nodes.
- CH winning frequency of compromised nodes per election: it is computed by summing the CH winning frequencies of compromised nodes through all election rounds and dividing the sum by the number of election rounds. This metric represents the robustness against the election result modifications by compromised nodes.
- Energy consumption per node: it is computed by summing the expended energy of all nodes during the simulation and dividing the sum by the number of nodes. This metric represents the energy efficiency of a CH election scheme.

Figure 25 shows how many CHs are generated in a cluster as the number of compromised nodes increases. Because the seed based scheme and the commitment based scheme do not properly deal with the selective transmissions of compromised nodes, they greatly increase the number of CHs even under a small number of compromised nodes. The key chain based scheme mitigates the increase of CHs via a protocol that merges multiple clusters into a single cluster when compromised nodes are sparse. However, if compromised nodes increase, they can collapse the operation of the merge protocol. Our scheme hardly increases the number of CHs per cluster regardless of the increase of compromised nodes as shown in Figure 25.

Figure 26 shows the CH winning frequency of compromised nodes as the number of compromised nodes increases. As shown in Figure 26, the seed based scheme and the commitment based scheme increase the CH winning frequency of compromised nodes according to the increase of compromised nodes. This is because compromised nodes can

easily forecast a CH election result and change the CH election result by avoiding the transmission of their contribution values. The key chain based scheme provides better security than the preceding two schemes, because it excludes nodes which avoid transmitting their contribution values more than once. However, if compromised nodes alternately avoid their transmissions, this scheme also allows many compromised nodes to survive the exclusion. Therefore, the key chain based scheme is also vulnerable to the increase of compromised nodes. Although our scheme also increases the CH winning frequencies of the compromised as the number of compromised nodes increases, the increasing rate is very low as shown in Figure 26.

Figure 27 shows the average amount of energy consumed by each node. The seed based scheme provides the best energy efficiency as compared to other schemes, because each node transmits only a short availability message in CH elections. It seems strange that the key chain based scheme reduces the amount of energy consumed by nodes even if the number of compromised nodes increases. This is because the key chain based scheme completely excludes a suspicious node from CH candidates when the node is silent more than once. Therefore, the excluded node will not consume energy any longer. Note that nodes consume almost the same amount of energy in our scheme. This is because compromised nodes in our scheme can participate in the next CH election even though they are excluded in the current election. This strategy ensures that our scheme is able to cope with message losses which occur frequently in the wireless network environment.

We introduced an additional simulation to find how the increase of nodes affects the security and performance of CH election schemes. In the extra simulation, we fixed the number of compromised nodes to 30 and assumed that no message is lost.

Figure 28 shows that the increase of nodes does not have a great impact on the number of generated CHs in all schemes. In Figure 28, the key chain based scheme seems to provide the best performance among others. However, since it produces some clusters having no CH, it is difficult to claim its superiority over other schemes. Considering such an aspect, we can claim that our scheme demonstrates the best performance over other schemes as shown in Figure 28.

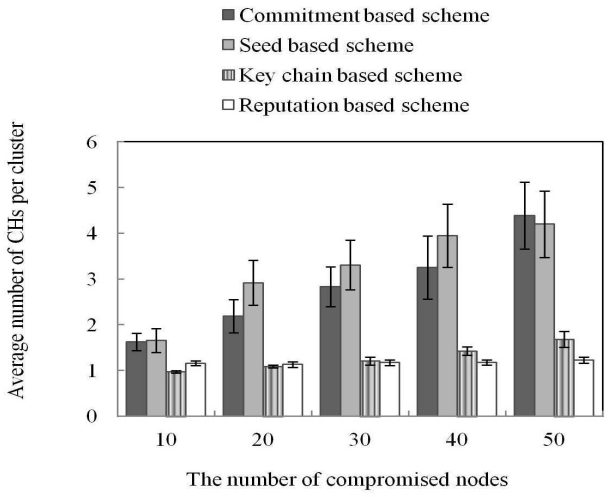

Figure 25. Avg. number of CHs per cluster

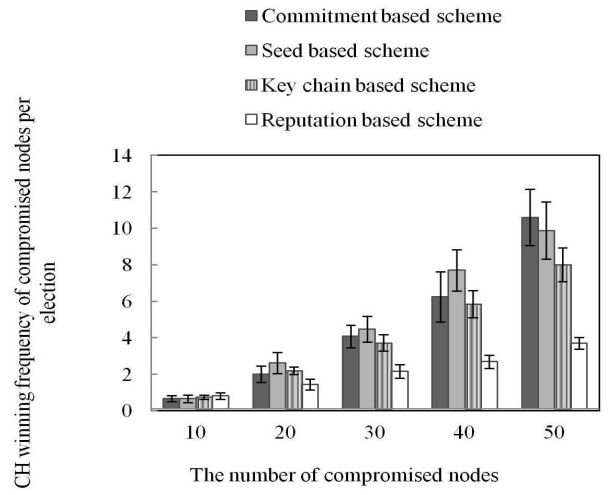

Figure 26. CH winning frequency of compromised nodes per election

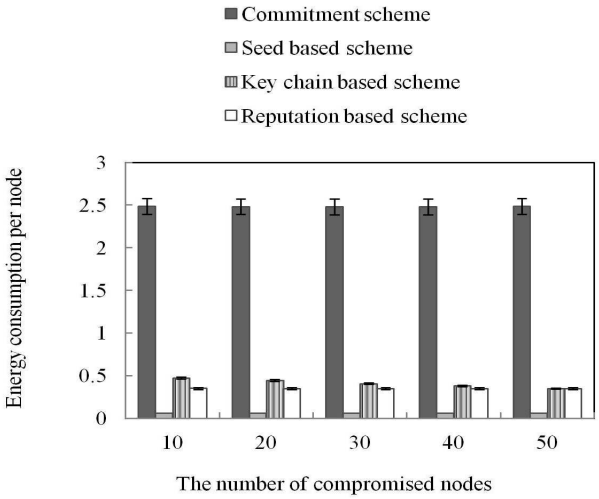

Figure 27. Energy consumption per node

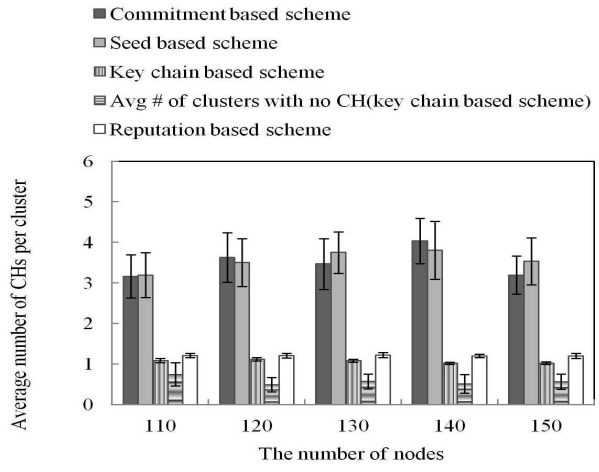

Figure 28. Avg. number of CHs per cluster

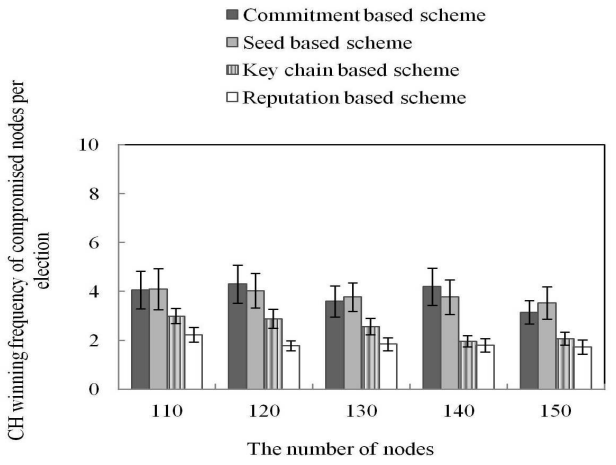

Figure 29 CH winning frequency of compromised nodes per election

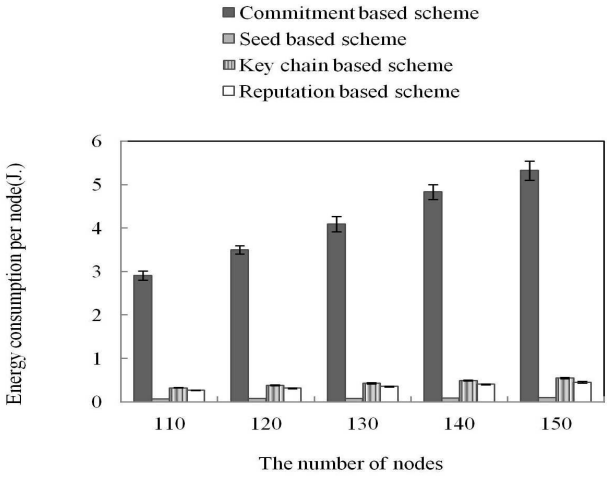

Figure 30. Energy consumption per node

Figure 29 shows the CH winning frequency of compromised nodes as the number of nodes increases. As shown in Figure 29, the increase of nodes reduces the CH winning frequency of compromised nodes in all schemes. This is natural because the number of compromised nodes is fixed and the number of normal nodes increases. Note that our scheme suppresses the CH winning frequency of compromised nodes better than other schemes.

Figure 30 shows the variation of energy consumption as the number of nodes increases. As shown in Figure 30, all schemes increase the energy consumption of nodes as the number of nodes increases. This is because the added nodes also join the CH election protocol and consume their energy resource.

In Table 4, we present the qualitative comparison of our CH election scheme and rival schemes. The key chain based scheme provides the worst unpredictability because the order of being the CH in a cluster is opened to all nodes. Other schemes allow only the last contributor of a common value to predict which node is going to be elected as the CH. Therefore, the last contributor can manipulate the CH election result through the silence attack. Although the key chain based scheme tackles the manipulability by excluding some compromised nodes from CH candidates, the exclusion does not work well when the population of the compromised nodes grows up and they alternately invoke a silence attack. The reputation based scheme excludes malicious nodes very well as long as the majority of nodes are normal nodes in the cluster. If the majority of members in a cluster are normal nodes, they will give a small reputation value to the compromised nodes. Therefore, the compromised nodes' reputation value is likely to be smaller than the cluster's average reputation value and the compromised nodes are likely to be excluded from the CH candidates.

Table 4. Qualitative comparison of CH election schemes

Properties / Schemes	Unpredictability	Non-manipulability	Agreement property	Immunity from message loss
Commitment based scheme	Medium	Medium	Low	Medium
Seed based scheme	Medium	Medium	Medium	High
Key chain based scheme	Low	Low	Medium	High
Reputation based scheme	Low	Low	Low	High

The commitment based scheme and the seed based scheme have no defense mechanism when compromised nodes try to break the agreement property by invoking selective transmission attacks. Therefore, the agreement property in a cluster can be easily broken. Although the key chain based scheme improves the property using a merge protocol, it does not work well when the population of the compromised nodes grows up and they do not cooperate with the protocol. In the reputation based scheme, normal nodes well exclude compromised nodes which launch selective transmission attacks as long as the number of normal members in a cluster is larger than that of compromised nodes. As a result, the CH candidate list of normal nodes is almost same.

In addition, the commitment based scheme and the seed based scheme cannot deal with message loss. Therefore, they are very vulnerable to message loss. In the key chain based scheme, message loss is more critical because some clusters cannot yield a CH due to insufficient recommendation messages. Because our scheme allows the nodes excluded in the current election to participate in the next election again, it is very robust against message loss. Above descriptions are summarized in Table 4.

6. Future Directions for Research

Our cluster formation scheme assumes that there is no message loss during the cluster formation process except for the intentional transmission avoidance of compromised nodes.

Even though it seems to be quite an immoderate assumption, we need this assumption to discriminate a compromised node from normal nodes. Without this assumption, normal nodes cannot discriminate the misbehavior of compromised nodes from message loss. Therefore, the removal of this assumption is an interesting future research item.

Even though we do not provide any simulation result, it is intuitive that our CH election scheme is resilient in an error-prone environment since it allows an excluded node to participate in the next election. In order to prove the resiliency against message loss, we need to introduce additional simulations in a future study.

In a cluster structure, the number of members in a cluster greatly affects energy-efficiency and the network lifetime. The larger a cluster size is, the greater energy-efficiency is. This is because a large sized cluster reduces the number of transmissions in the cluster and consequently reduces the number of clusters (CHs) in the network. In a clustered sensor network, since only CHs performs a long distance transmission, a small number of CHs decreases the number of long distance transmissions. So, we need to devise a scheme which generates larger sized clusters than our scheme and well identifies and excludes some misbehaving nodes.

7. Conclusions

In this paper, we have presented a secure cluster formation scheme and a secure CH election scheme. Our cluster formation scheme generates large sized clusters and suppresses their split by using two-hop conformity check. Besides, our cluster formation scheme mainly employs broadcast communication during the cluster formation to save the energy consumption of nodes. The simulation results represent that our scheme expels more compromised nodes from clusters and suppresses the separation of clusters. Other simulation results represent that our scheme raises the quality of clusters and more energy-efficient than a rival scheme. Our CH election scheme makes manipulation of any CH election result harder than other rival schemes since it excludes any disreputable nodes well using the local trust system. Our CH election scheme also preserves the agreement property of any CH election result by coping well with the misbehavior of any disreputable nodes. Simulation results have proven that Our CH election scheme enhances the non-manipulability and the agreement property of CH election results compared to other schemes. Additional simulation results have shown that the proposed scheme preserves its performance even though the number of nodes increases.

References

[1] W. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” IEEE Trans. on 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] G. Wang, K. Song, and G. Cho, “DIRECT: Dynamic Key Renewal Using Secure Cluster Head Election in Wireless Sensor Networks,” IEICE Trans. on Information and Systems, vol. E93-D, no. 6, pp. 1560-1571, Jun. 2010

[5] G. Wang and G. Cho, “Clustering-based Key Renewals for Clustered Sensor Networks,” IEICE Trans. on Communications, vol. E92-B, no. 2, pp. 612-615, Feb. 2009

[6] S. Kang and T. Nguyen, “Distance Based Thresholds for Cluster Head Selection in Wireless Sensor Networks,” IEEE Communications Letters, vol. 16, no. 9, pp. 1396-1399, Sep. 2012

[7] V. Katiyar, N. Cand, G. C. Gautam, and A. Kumar, “Improvement in LEACH Protocol for Large-scale Wireless Sensor Networks,” Proc. of Int’l Conf. on Emerging Trends in Electrical and Computer Technology, pp. 1070-1075, Mar. 2011

[8] P. Ren, J. Qian, L. Li, Z. Zhao, and X. Li, “Unequal Clustering Scheme based LEACH for Wireless Sensor Networks,” Proc. of Fourth Int’l Conf. on Genetic and Evolutionary Computing, pp. 90-93, Dec. 2010

[9] M. Saadat, R. Saadat, ang G. Mirjality, “Improving Threshold Assignment for Cluster Head Selection in Hierarchical Wireless Sensor Networks,” Proc. of Int’l Symposium on Telecommunications, pp. 409-414, Dec. 2010

[10] Y. Han, M. Park, and T. Chung, “SecDEACH: Secure and Resilient Dynamic Clustering Protocol Preserving Data Privacy in WSNs,” Proc. of the 2010 Int’l Conf. on Computational Science and Its Applications, Lecture Notes in Computer Science, vol. 6018, pp. 142-157, 2010

[11] H. Rifà-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

[12] K. Sun et al., “Secure Distributed Cluster Formation in Wireless Sensor Networks,” Proc. of 22nd Annual Computer Security Applications Conference (ACSAC ’06), pp. 131-140, 2006

[13] G. Wang, D. Kim, and G. Cho, “A Secure Cluster Formation Scheme in Wireless Sensor Networks,” Int’l Journal of Distributed Sensor Networks, vol. 2012, Article ID 301750, 14 pages, 2012

[14] L. Buttyan and T. Holczer, “Private Cluster Head Election in Wireless Sensor Networks,” Proc. of the Fifth IEEE Int’l Workshop on Wireless and Sensor Network Security (WSN ’09), IEEE, pp. 1048-1053, 2009

[15] M. Sirivianos et al., "Non-manipulable Aggregator Node Election Protocols for Wireless Sensor Networks," Proc. of Int'l Sympo. on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt '07), Cyprus, pp. 1-10, Apr. 2007

[16] Q. Dong and D. Liu, “Resilient Cluster Leader Election for Wireless Sensor Networks,” Proc. of IEEE 6th Annual

Comm. Society Conf. on Sensor, Mesh and Ad Hoc Communications and Networks(SECON), pp108-116, 2009

[17] A. Perrig et al., "SPINS: Security Protocols for Sensor Networks," Wireless Networks, vol. 8, no. 5, pp. 521-534, Sep. 2002.

[18] D. Liu, "Resilient Cluster Formation for Sensor Networks," Proc. of 27th Int'l Conf. on Distributed Computing Systems (ICDCS '07), pp.40-48, 2007

[19] 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

[20] G.V. Crosby and N. Pissinou, "Cluster-based Reputation and Trust for Wireless Sensor Networks," Proc. of the 4th IEEE Consumer Communications and Networking Conference (CCNC '07), pp. 604-608, Jan. 2007

[21] P. Schaffer, K. Farkas, A. Horvath, T. Holczer, L. Buttyan, "Secure and Reliable Clustering in Wireless Sensor Networks: A Critical Survey," Computer Networks, vol. 56, no. 11, pp. 2726-2741, Jul. 2012

[22] Y.C. Hu, A. Perrig, and D.B. Johnson, "Wormhole Attacks in Wireless Networks," IEEE Journal on Selected Areas in Communications, vol. 24, no. 2, pp. 370-380, 2006

[23] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S.C. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," Proc. of 3rd Int'l Conf. on Pervasive Computing and Communications, Kauai, Hawaii, USA, Mar. 8-12, 2005

[24] The network simulator-ns-2, Available: http://www.isi.edu/nsnam/ns/, Dec. 2012

Appendix

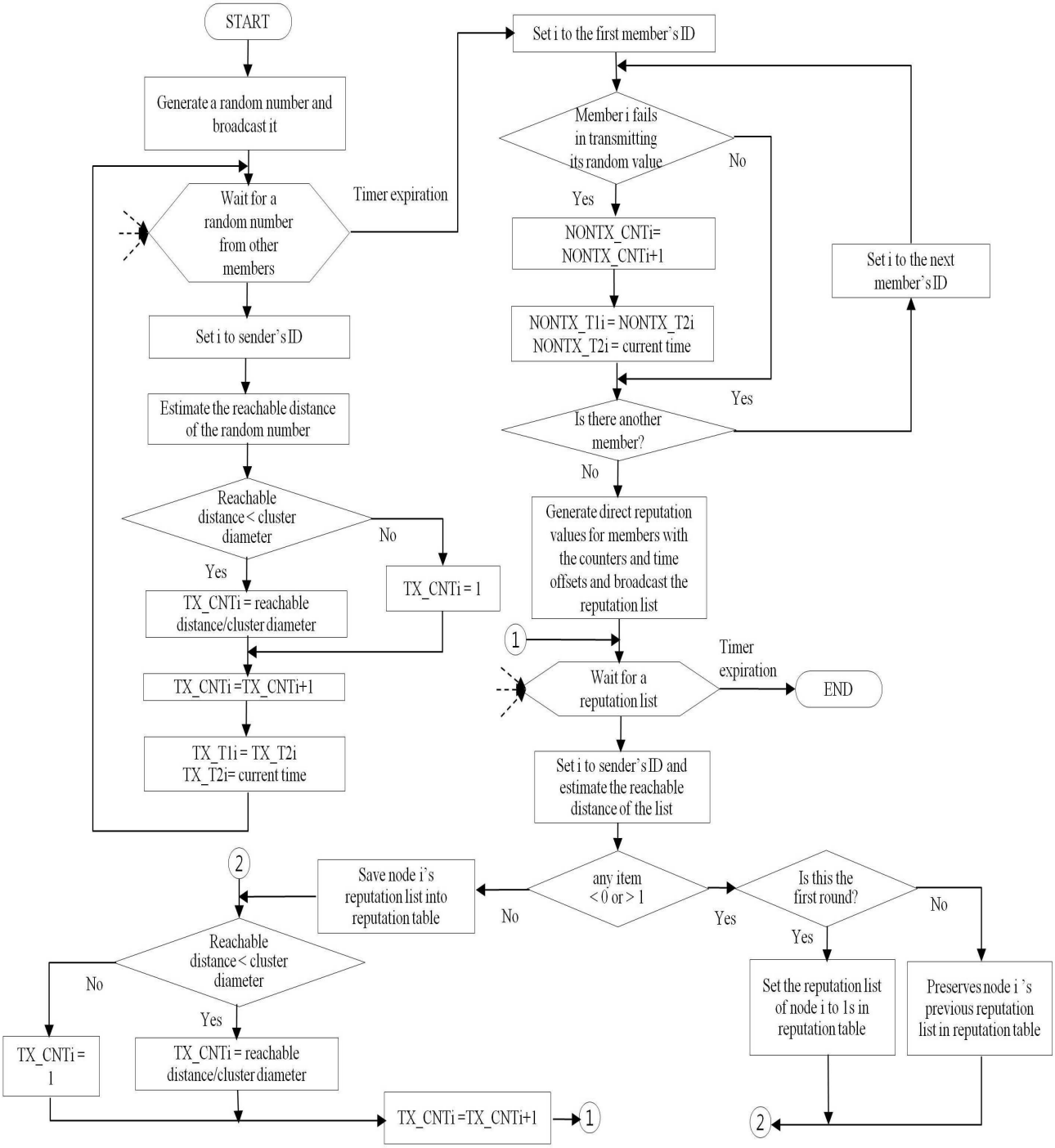

Figure 14. Flowchart for transmission of contribution value amd direct reputation list

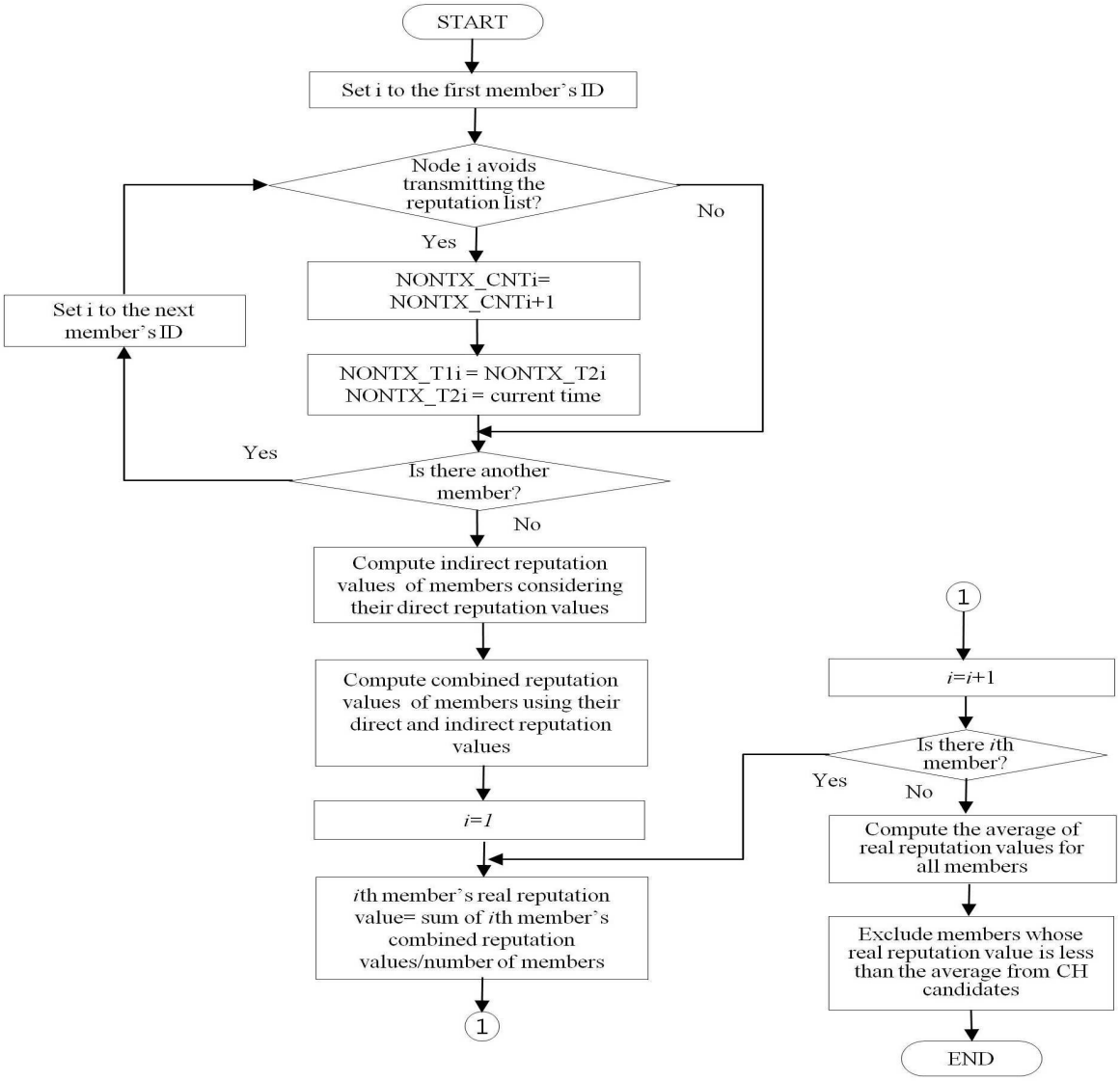

Figure 15. Flowchart for reselection of CH candidates