Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/081L22

Secure Cluster Formation Employing Two-hop Conformity Verification in Wireless Sensor Networks

Gicheol Wang[1], *Sangwon Bang*[2] *and Gihwan Cho*[3,*]

[1] Dept. of Advanced KREONET Service, Korea Institute of Science and Technology Information, Daejeon, Korea
[2] Dept. of Computer and Information, Songwon University, Gwangju, Korea
[3] Dept. of Computer Science and Engineering (Cloud Open R&D Center), Chonbuk National University, Jeonju, Korea

Received: 28 Apr. 2013, Revised: 23 Aug. 2013, Accepted: 24 Aug. 2013
Published online: 1 Apr. 2014

Abstract: Cluster structure is frequently adopted for wireless sensor networks since it enables energy-efficient communication between nodes and extends the network longevity. To build up a cluster structure in a network, a cluster formation protocol should be invoked. During a cluster formation protocol, if compromised nodes survive the screening process, they can make some members have different cluster membership to separate a cluster and consequently lower the average cluster quality. To cope with this problem, this paper presents a novel and secure cluster formation scheme. First, our scheme enhances the average cluster quality by attempting to create two-hop clusters where all members are at most two hops away from each other. Second, our scheme avoids the separation of the clusters through the help and verification of two hop distant nodes. Last, our scheme saves energy consumption amount of nodes by primarily using broadcast transmissions. Experiments show that our scheme greatly reduces the number of generated clusters as compared with a rival scheme and expels more compromised nodes and consumes less amount of energy than the rival scheme.

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

1 Introduction

Currently, applications of sensor networks are quite various ranging from military reconnaissance and some sort of surveillances to smart cities. Cluster structure brings many benefits in such networks since it enables extension of network lifetime [1,2], balanced load among nodes [3], and distribution of key management [4,5].

Cluster structure is made by grouping physically adjacent nodes into a logical bunch of nodes that is called a cluster. The procedure that groups adjacent nodes into a cluster is called cluster formation and a protocol should be invoked by all nodes for the implementation of the procedure. A leader in the cluster which is called a CH (Cluster Head) can be elected by all members and the election is performed by all members invoking a CH election protocol. A CH gathers sensed data from its members and sends the gathered data to the sink. Therefore, it can be easily identified by attackers and the attackers aim at the CH as a compromise target or try to become a CH to get all sensed data and handle them. We have two options to keep attackers from becoming a CH. As the first option, we can keep attackers from joining the initial cluster formation. We employ this option when we want to keep an attacker from joining the initial cluster formation and from having the legal right to become a CH. So, the first option mitigates the vulnerability that an attacker becomes a legal CH candidate. As the second option, we can prevent attackers from predicting and manipulating election results for their benefits. So, the second option mitigates the vulnerability that an attacker facilitates their CH winning in the elections. Because mitigating the first vulnerability also turns into the reduction in the second vulnerability, we only focus on the first option in this paper and skip the second option. Refer to [6,7,8] for more details about the second option. Some schemes [9,10,11] employed shared keys to keep external attackers from joining the cluster formation process. However, they cannot prevent compromised nodes (that is, internal attackers) from joining the cluster formation process. In [12], each node checks the protocol conformity of members using the public key cryptography. The scheme effectively keeps two types of compromised nodes from joining the cluster formation

* Corresponding author e-mail: ghcho@chonbuk.ac.kr

process. However, the scheme generates only small sized clusters which are called cliques and even splits them whenever it identifies a suspicious node. As a result, the scheme increases the number of clusters and decreases the average size of clusters. Furthermore, the scheme induces a lot of communication overhead to check and verify the protocol obedience of members.

We try to resolve above problems in this paper. First of all, our scheme increases the average size of clusters by merging two hop distant nodes. That is, our scheme reduces the number of clusters by forming two-hop clusters where any two nodes are at most two hops away from each other. Second, our scheme minimizes the separation of the clusters by employing the help of two hop distant nodes. Last, our scheme minimizes the unicast communication and employs broadcast communication more frequently to diminish the communication overhead.

We organize this paper as the following. We briefly describe the related work covering secure cluster formation in Section 2. The system and attack model is presented in Section 3. We describe the details of the proposed scheme in Section 4 and provide the simulation results in Section 5. We draw the conclusion in Section 6.

2 Related Work

A probability-based CH election scheme where a node becomes a CH without message exchange was proposed in [1]. The authors attempted to lengthen the longevity of network by assigning CH roles to all nodes in turn. All nodes calculate the threshold value which is based on the probability of being a CH and generate their own random value. Then, they examine if their own random value is lower than the threshold or not. Some nodes having lower values than the threshold become a CH and otherwise nodes become a member of one of the CH nodes. However since this scheme assumes a benign environment, they cannot defeat malicious actions made by external attackers and misbehavior of compromised nodes in the operation of protocol.

Ferreira et al. proposed F-LEACH [9] which protects the cluster formation process of LEACH [1] using pre-assigned keys shared between the sink and nodes. If a node's random value is lower than the probability based threshold, it broadcasts a CH declaration message using the shared keys and the sink verifies the CH declaration using the same keys. Next, the sink creates the verified CH list and broadcasts the list using μTESLA [13]. Last, non-CHs join one of the verified CH nodes. However, since this scheme cannot prevent the compromised nodes from declaring themselves as CHs, it is vulnerable to the compromise of normal nodes. Furthermore, it also cannot prevent an illegal node from joining a cluster since it does not verify the legality of joining nodes.

SecLEACH [10] was proposed to address the illegal join of nodes. In Sec-LEACH, the sink verifies the CH declarations of nodes and the verified CHs also verify the cluster join of nodes. However, since F-LEACH and Sec-LAECH employ pre-assigned keys shared between the sink and nodes to protect the cluster formation, they cannot prevent compromised nodes from declaring themselves as CHs or from joining one of verified clusters without qualification.

A cluster formation scheme where some pre-assigned nodes become CHs and otherwise nodes join one of the CHs directly or indirectly was proposed in [11]. This scheme defeats external attackers quite well since it verifies the CH declarations or the cluster joins using pre-assigned polynomial shares. So, an external attacker having no such shares cannot easily declare itself as a CH or join one of verified CHs. The scheme's prominent feature is a wormhole prevention scheme. If a node recognizes that it has many neighbors, it judges that it is under the wormhole attacks and shutdowns itself. If a node having many neighbors does not shutdown itself, the sink expels it by adding the node to the blacklist report and broadcast the report. In the scheme, since a relay node plays a role of connector between a member and its CH, it can be an attack target. For instance, a compromised node can make many orphan nodes by disturbing the relay node settlement of normal nodes. Furthermore, a compromised relay node can do a DoS (Denial of Service) attack to its serving nodes. Last, attackers are necessarily going to attempt to compromise the pre-assigned CH nodes.

In [12], the authors proposed a cluster formation scheme where each cluster member checks the protocol conformity of other members using public key cryptography and redundancy. First, a physical network is transformed into cliques which are small sized clusters. In a clique, all nodes are mutually reachable from each other using the normal transmission power. Then, each node verifies whether all members share the same view on cluster membership or not. If any inconsistency is found, all members are examined if they obey the protocol operation to recognize and expel compromised nodes. Even though the scheme well recognizes and expels the compromised nodes through such a fastidious check, it induces some disadvantages. First, the number of clusters increases because it creates only small sized cliques and even splits them whenever any suspicious action is found. Even worse, its communication overhead is quite high since it mainly employs unicast communication when it checks the conformity of members.

3 System and Attack Model

3.1 System Model

We assume that sensors are deployed arbitrarily in the mission field without human intervention. After deployment, the locations of nodes are never changed and the network is transformed into clusters to save the energy

of nodes by reducing their transmissions. The sink plays a role of data collection center and a gateway to the infrastructure networks. We add the followings to our assumptions.

First, each node exactly recognizes its neighbors using a wormhole prevention scheme such as [14]. Second, the link between any two neighbors is bidirectional. Third, there is no message loss during the network operation excepting for intentional transmission avoidance of attackers. Forth, we adopt the non-persistent CSMA (Carrier Sense Multiple Access) as a MAC (Medium Access Control) protocol. Fifth, even though public key operations are known too heavy to be performed on a sensor node, a sensor node can perform some lightweight operations such as ECC operations. In [15], the authors proved that such lightweight operations can be performed well on a sensor node. Sixth, all public keys of nodes (that is, certificates) were distributed to the network and they were verified by each node using the CA's public key. Therefore, each node can be uniquely discriminated from other nodes due to its private key.

3.2 Attack Model

Attackers can launch various attacks on networks. Most severe attack is the DoS attack on the physical layer using jamming signals. Because there is no effective way to avoid or prevent this kind of attack, we assume that attackers do not launch this kind of attack. If an attacker succeeds the Sybil attack [16] or the wormhole attack [14], the victim nodes have the wrong list of neighbors and sometimes network does not work due to the wrong list. We assume that the schemes of [17] and [14] defeat those attacks well so that the network is free from those attacks.

In this paper, attackers mean compromised nodes and their attacks mean that they do not obey to the protocol operation. This attack limitation is required for focusing on the cluster formation problem. Particularly, we pay attention to two sorts of attacks which are executable to a cluster formation protocol. First, a compromised node may send a message to a part of members and it is called selective transmission attack hereafter. Second, a compromised node may avoid the transmission of a message and it is called silent attack hereafter. Both attacks make disagreement on the view of cluster membership among members. The disagreement consequently splits a cluster and decreases the average number of members in a cluster.

4 Cluster Formation Employing Two-hop Conformity Check

To help the quick comprehension of our scheme, we define some terms and messages which are used in our scheme in the following. Because each node has already known its neighbors, it also knows which nodes have a lower ID or a higher ID than itself. A node whose ID is lower than itself is considered as a dominant and the node is stored in the list of *dominants*. Also, a node whose ID is lowest among neighbors declares itself as a *cluster separator*.

In the following, we describe the types of messages employed in our scheme. A cluster separator begins the protocol. It is not a CH but only initiates the protocol by broadcasting a *cluster separator message*. If a node has no cluster when receiving a *cluster separator message*, it joins the cluster by broadcasting a *cluster response message*. Upon receiving a *cluster response message*, a receiver checks if the sender is in the *dominants* list. If so, it removes the sender from the list of *dominants* and checks if the list is empty. In case of emptiness, it also becomes a cluster separator. The *cluster separator message* and the *cluster response message* cooperatively determine the cluster border. When a node recognizes that a *cluster separator message* is transmitted to only a part of members, it asks its affiliation to the cluster by broadcasting a *cluster affiliation message*. When a cluster separator asks its members to allow its affiliation to the cluster, it broadcasts a *final cluster message*. If a node recognizes that a cluster separator transmits a *final cluster message* to only a part of members, it collects a proof showing its legality by broadcasting a *solicitation message*.

4.1 Cluster Border Determination

4.1.1 Broadcast of Cluster Separator and Response Message

A node whose ID is the lowest among neighbors becomes a cluster separator and broadcast a cluster separator message. The message type and the separator ID signed by the owner's private key constitute the cluster separator message and the signed ID keeps attackers from launching a spoofing attack. In case of multiple receptions of such a message, only the first message is considered as the cluster separator message and other messages are ignored.

When a node receives a cluster separator message, it first verifies the signed ID. If the verification succeeds, it affiliates to the cluster using a cluster response message. So the cluster separator message and cluster response messages settle the cluster border. The message type, the separator ID, and the signed separator ID constitute the cluster response message. A cluster response message is signed by the owner's private key before the broadcast. This is required for proving that the message is created by the owner and it is not modified by other nodes. The sender ID is also appended to the end of the message. Upon receiving a cluster response message, the receiver verifies the signed message and saves the message in case of successful verification. Especially, the cluster separator

employs the cluster response messages to prove its qualification when it requires other members to accept its join in step 2. In a received cluster response message, the separator ID is first examined if it is same as the receiver's separator. Next, if the cluster response message is not a duplicated message, the receiver rebroadcasts it to defeat a silent attack. If a node receives a cluster response message originated from a different separator, it only checks if the sender is a dominant. If so, the node removes the sender from its dominants and becomes a separator when the list of dominants is empty.

4.1.2 Settlement of Cluster Borders

After the broadcast of cluster separator and response message, each member examines if any node disobeyed the protocol during the process. If any node does so, the following countermeasures start.

1. When a cluster separator is the single connection point among nodes, it can refuse to rebroadcast a cluster response message even though it is a new message. If it happened, the connected nodes through only the separator could not receive the cluster response message and they would have a different list of cluster response messages. To agree the list of cluster response messages, we employ the following measures.
   A. When a node recognizes that it has no cluster response messages from any two hop member, it broadcasts its cluster response message with a doubled transmission power.
   B. A node receiving a cluster response message from any two hop member appends the sender to the member list and sends its own cluster response message to the sender in unicast manner.
2. When a cluster separator launches a silent attack, neighbors can get no messages. If that happens, neighbors exclude the cluster separator from dominants and neighbors. If the list of dominants is empty, they declare themselves as a cluster separator using a cluster separator message.
3. If a cluster separator launches a selective transmission attack when delivering its cluster separator message, the victims cannot get the message. However, the victims may get a cluster response message from other members in the same cluster and recognize that the cluster separator does an attack. It is definite that the aim of the attack is to remove the victims from the cluster.
   A. If a node receives not cluster separator messages and but only cluster response messages, it requires other members to accept its affiliation to the cluster. The message type, the separator ID, and the signed separator ID from a cluster response message constitute a cluster affiliation message. Last, the affiliation requester signs the affiliation message with its private key before transmission.
   B. Upon receiving a cluster affiliation message, the receiver excludes the affiliation requester and the cluster separator from both members and neighbors. This is because the receiver cannot make sure if the affiliation requester is telling a lie or the separator really misbehaved. Therefore, the safest way is to separate them and make them configure their own cluster. Next, the receiver of the cluster affiliation message rebroadcasts the message if the message is not a duplicate.

Sometimes the adjustment of cluster borders makes a node disconnected from its cluster separator. Such a node first examines if there is no dominants among its neighbors and the node becomes a cluster separator in that case.

4.2 Merge and Verification of Cluster Separators

4.2.1 Merge of Cluster Separator

At this step, a cluster separator attempts to join its own cluster by broadcasting a final cluster message. The message type and the list of pre-received cluster response messages constitute a final cluster message. The message is signed by the owner's private key before being broadcasted. A receiver of a final cluster message verifies the signature and checks if the cluster separator has the same list of cluster response messages with it. If so, the receiver accepts the join of the separator and updates its member list. Otherwise, the receiver ignores the final cluster message.

4.2.2 Verification of Cluster Separator Affiliation

After the join of a cluster separator is finished, each member inspects if the cluster separator disobeyed the protocol operation. As soon as any disobedience is detected, the following measure is launched.

1. A malicious cluster separator excludes some cluster response messages in the final cluster message and makes the final cluster message unreachable to the victims. This is a trick to expel the victims from the cluster. The victims can defeat the trick using the below measure.
   A. When a node has gotten no final cluster messages, it doubles the transmission power and broadcast a solicitation message so that at most two hop distant nodes can hear it.
   B. A node receiving a solicitation message inspects whether it has the solicitor's cluster response message or not. In case of possession, it first signs the message with its private key and sends the signed message and the final cluster message to the solicitor.

C. If the solicitor receives the signed message and the final cluster message, it first inspects the final cluster message. If there is an unknown response message in the final cluster message, it registers the originator of the response message into the member list. Then, the solicitor checks if its cluster response message disappears from the list of cluster response messages. If so, it reports the cluster separator as a malicious node with a doubled transmission power. Its cluster response message signed by a witness and the signer ID constitute the report message.
D. When a node gets a report about malicious nodes, it attempts to verify the signature. If the cluster response message is never known before, the receiver excludes the cluster separator from its cluster and affiliates the reporter into the cluster member. Actually speaking, a receiver of a malicious node report cannot make sure which node is responsible for the conflict in the list of cluster response messages. Nevertheless, we can make sure that the cluster separator is most responsible for the conflict because it is connected to all members in the cluster.

If a solicitor cannot get any evidence for its legality during a specific period of time, it excludes the separator from the member list and neighbors and again invokes the protocol from the first step.

4.3 Depicted Example

In this section, we present some figures and describe our scheme with those figures to help the understanding for our scheme. The first step of our scheme determines cluster borders and Figure 1 and Figure 2 show the procedure. As shown in Figure 1, nodes 1, 3, and 5 broadcast a cluster separator message because they are lowest ID nodes among neighbors. Among them, a cluster separator (that is, 1) disobeys the protocol. Node 1 makes its cluster separator message unreachable to nodes 4 and 7 to exclude them from members.

As shown in Figure 2, when a node receives a cluster separator message, it replies to the message by broadcasting a cluster response message. When a node receives a cluster response message and it is not a duplicate, the node broadcasts the message again. An attacker may evade the rebroadcast and node 3 is such a node. Due to node 3's evasion of rebroadcast, nodes 9 and 28 have no idea about each other. So, node 9 transmits its cluster response message with a doubled transmission power to search any two hop neighbor. Node 28 appends the node 9 into its members and also broadcasts its own cluster response message with a doubled transmission power. Node 9 also appends the node 28 into its member list. When node 30 receives cluster response messages from nodes 14 and 17, it notifies that all dominants are joined other cluster. Therefore, node 30 now becomes a cluster separator.

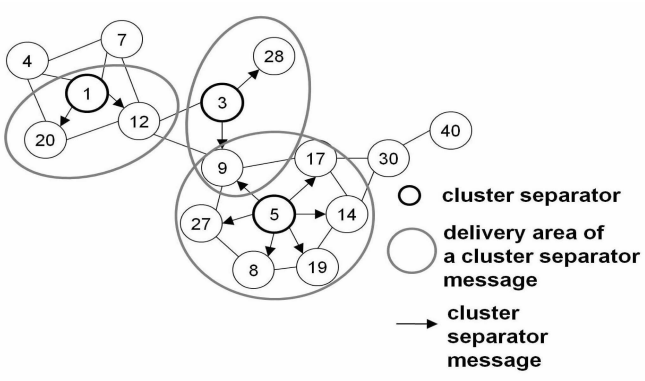

Fig. 1: Broadcast of cluster separator message

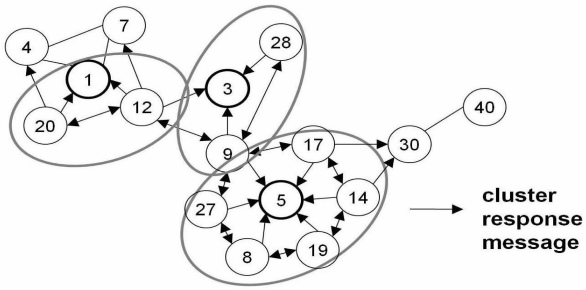

Fig. 2: Broadcast of cluster response message

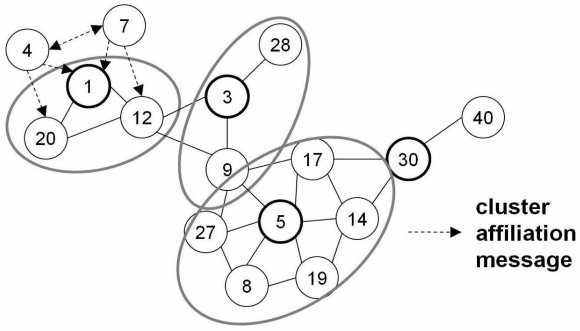

Fig. 3: Broadcast of cluster affiliation messages

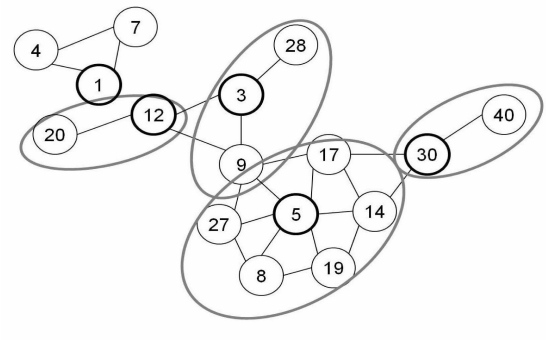

Fig. 4: Separation of conflicting inducers and new cluster formation

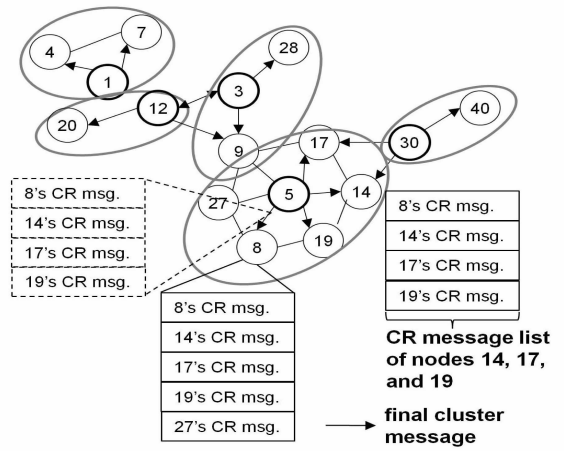

Fig. 5: Distribution of final cluster message

Even though nodes 4 and 7 received no cluster separator message, they now get cluster response messages from other members (that is, nodes 12 and 20). Due to the cluster response messages, they notify that the cluster formation is processing but they are excluded from the process. Now, as shown in Figure 3, they broadcast a cluster affiliation message to request their join to the cluster.

At the standpoint of nodes 12 and 20, the situation is very vague because they cannot assure who is to blame for the conflict. The situation occurs when node 1 does not transmit the cluster separator message or when nodes 4 and 7 tell a lie about their reception of cluster separator message. So, nodes 12 and 20 resolve the problem by removing the conflict inducers (i.e. nodes 1, 4, and 7) from their cluster and neighbors as depicted in Figure 4. Now, 12 and 20 are disconnected from the old cluster separator and they restart the cluster border determination. Node 12 has no dominants because nodes 3 and 9 have already formed their own cluster. So, it declares itself as a cluster separator by broadcasting a cluster separator message and forms a new cluster. Node 30 also broadcasts a cluster separator message like node 12 and we can see the result in Figure 4.

Step 2 merges cluster separators into their own cluster only if they obeyed to the protocol operation. First, cluster separators initiate the step 2 as the step 1. Cluster separators (i.e. 1, 3, 5, and 30) broadcast a final cluster message to start the step 2 as depicted in Figure 5. To pursuit the simplicity of explanation, we focus on the merge of node 5 in the following. Here, nodes 5 and 19 were assumed to evade rebroadcasting 27's cluster response message in step 1 and their aim was to expel 27 from the cluster. Now, the cluster separator 5 broadcasts its final cluster message which consists of cluster response messages of its members (i.e. 8, 14, 17, and 19). Note that the cluster separator 5 intentionally excludes 27's cluster response message in the final cluster message as if it did not transmit its cluster response message. Besides, the cluster separator 5 intentionally makes the final cluster message unreachable to node 27 to delay the detection of its misbehavior. Members of 8, 14, 17, and 19 check if the list of cluster response messages in the final cluster message are same as their own list. In case of 14, 17, and 19, they recognize that they have the same list with that of the received final cluster message and affiliate the separator 5 into their members. Since member 8 recognizes that it has a different list from that of received final cluster message, it ignores the message.

Now, the victim 27 tries to get a proof of its legality by broadcasting a solicitation message with a doubled transmission power as depicted in Figure 6. Fortunately, a member 8 has the victim's cluster response message and it signs the cluster response message with its private key and replies to the victim 27 through the signed message and pre-received final cluster message. So, 27's cluster response message signed by 8's private key and 5's final cluster message constitute a solicitation response message.

As soon as the victim 27 receives the solicitation response message, it examines if the final cluster message contains a cluster response message from an unknown node. If so, it appends the unknown node into its members. Next, the victim 27 examines if 5's final cluster message contains its own cluster response message. If 5's final cluster message is missing its own cluster response message, it is an unavoidable evidence for 5's misbehavior. In this case, the victim 27 advertises that the cluster separator 5 is a malicious node through an attacker report as depicted in Figure 7. The attacker report is propagated to two hop distant nodes (i.e. 14, 17, and 19) and the report consists of the received proof (i.e. 27's cluster response message signed by 8's private key) and witness' ID (i.e. 8). Upon receiving an attacker report, the members (i.e. 8, 14, 17, and 19) first verify the signature and remove their cluster separator 5 from its cluster only if the verification is successful. Because the cluster

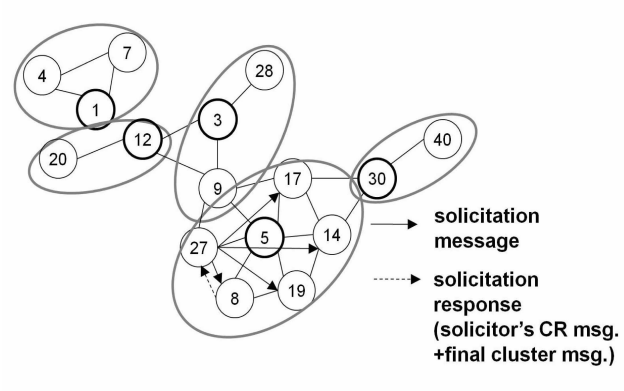

Fig. 6: Exchange of solicitation request and response

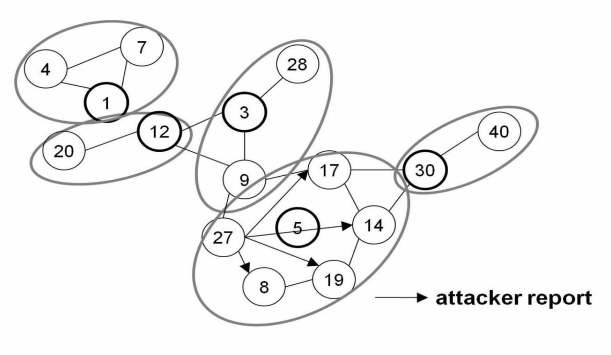

Fig. 7: Distribution of attacker report

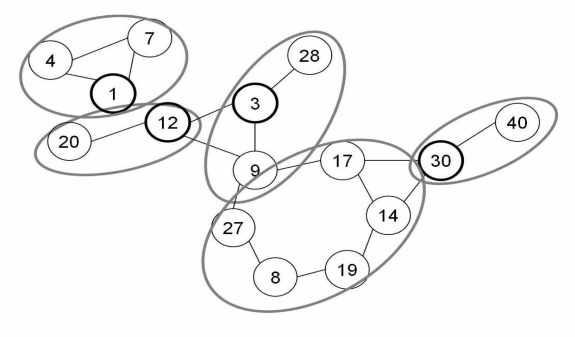

Fig. 8: Finalized cluster formation

Table 1: Parameters for simulations

Simulation parameter	Parameter value
Simulation field	100 meters by 100 meters
Population of nodes	100
Population of compromised nodes	7∼35
Beginning energy	20 Joules
Energy exhaustion model	[1]'s energy model
Wireless bandwidth	1 Mbps
Wireless MAC protocol	Non-persistent CSMA
Digital signature algorithm	ECDSA (Elliptic Curve Digital Signature Algorithm)-160
Algorithm for encryption and decryption	AES(Advanced Encryption Standard)-128
Hash algorithm	SHA-1

separator 5 is connected to all members, it is most responsible for the miss of 27's cluster response message at some members. Now, the members 14, 17, and 19 appends the victim into their members since they can make sure that 27 is a legitimate node that member 8 guarantees the legitimacy.

Finally the whole process for the cluster formation has been completed and we now have a clustered network as depicted in Figure 8.

5 Simulations

We built up the simulation environment using the ns-2 network simulator and evaluate the security and energy-efficiency of our scheme compared to a rival scheme. We deploy 100 nodes arbitrarily in a 100meters by 100meters square field. Each node exhausted their energy following the energy model of [1]. Besides, each node is assumed to employ the non-persistent CSMA MAC protocol. In addition, we assumed that no collisions occurred during the protocol. Parameters for the simulations are listed in Table 1. Only Sun's scheme was compared to our scheme through the simulations. This is because other schemes' cluster formation method and attack method are very different from those of our scheme.

Figure 9 represents the variation in the number of clusters as the number of compromised nodes increases. The variation represents the robustness of two schemes against selective transmission attacks and silent attacks. As Figure 9 represents, two schemes augment the number of clusters linearly in line with the augmentation of compromised nodes. Nevertheless, our scheme attains a great reduction in the number of clusters compared to Sun's scheme. The reason of this result is twofold. First, since our scheme forms two-hop clusters where all members are at most two hops away from each other, the number of clusters decreases naturally. Besides, since our scheme avoids the separation of clusters even when they suffer from attacks, the increase rate of clusters is quite low.

Figure 10 represents the variation in the number of cluster members as the number of compromised nodes increases. The variation represents how well two schemes

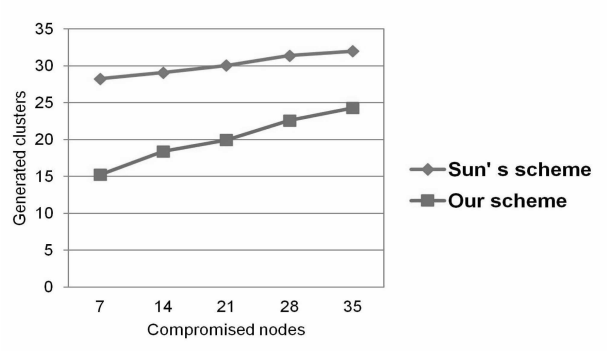

Fig. 9: Generated clusters vs. compromised nodes

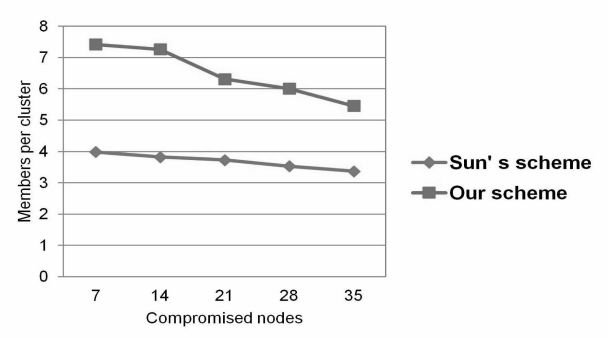

Fig. 10: Members per cluster vs. compromised nodes

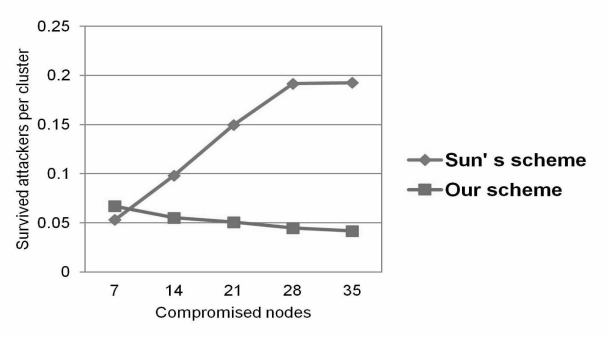

Fig. 11: Survived attackers per cluster vs. compromised nodes

maintain the cluster quality under selective transmission attacks and silent attacks. As depicted in Figure 10, both schemes deteriorate the cluster quality in line with the augmentation of compromised nodes. With regard to Sun's scheme, as soon as a node is recognized as a malicious node or a suspicious node, it is expelled immediately. Therefore, the cluster quality of Sun's scheme is greatly inverse proportion to the increase of

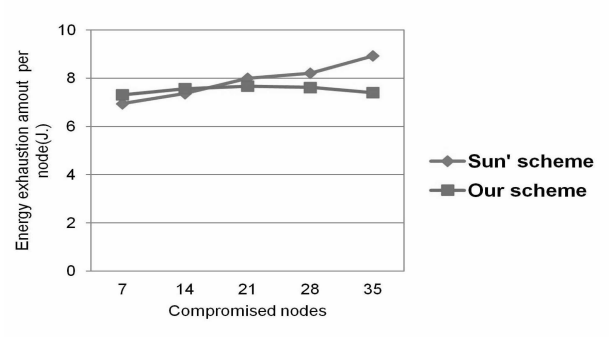

Fig. 12: Energy exhaustion amount per node(J.) vs. compromised nodes

compromised nodes. Contrarily, our scheme is more dullish to the increase of compromised nodes than Sun's scheme since it forms larger sized clusters than Sun's scheme and minimizes the separation of the clusters even under attacks.

Figure 11 shows how many attackers (i.e. compromised nodes) survived after the cluster formation in two schemes. The result represents how well two schemes expel compromised nodes. At a small number of compromised nodes, our scheme allows more attackers to survive from the exclusion mechanism than Sun's scheme. Actually, Sun's scheme well expels attackers using its one hop conformity check, when their population is small. However, because compromised nodes are not going to respond to normal nodes' one hop conformity check, the augmentation of compromised nodes induces a trouble. In other words, if a member identifies a disagreement on cluster membership of any other node, it attempts to reveal which node causes the problem by obtaining related evidences from members analyzing them. Here, if a member having any evidence is a compromised node, it never responds to the request. If all members holding any evidence are compromised, the inspection is going to end with no evidences. In this case, a compromised node which causes the disagreement can detour the one hop conformity check. So, as compromised nodes augment, they can easily detour the conformity check and survive the screening process. Contrarily, since our scheme requests at most two hop distant nodes to offer their holding evidences, our scheme can get more evidences and better detect compromised nodes than Sun's scheme. Particularly, as the population of compromised nodes grows up, our scheme better identifies and take away compromised nodes as depicted in Figure 11.

In Figure 12, how the augmentation of compromised nodes impacts on each node's energy consumption amount is shown. Sun's scheme makes a node perform a protocol conformity check whenever it recognizes a disagreement on cluster membership of any other node.

When a node performs a protocol conformity check, it first asks other members to provide previously received messages with their signature. Assuming the increase of compromised nodes, they all initiate the protocol conformity check to get evidences from members. It requires a large amount of energy consumption at the members since they should respond to all requests of the increased requestors. Our scheme looks like consuming more amounts of energy than Sun's scheme when the number of compromised nodes is small. However, it is a trivial and temporal increase. As compromised nodes augment further, our scheme saves much more amounts of energy than Sun's scheme as depicted in Figure 12. This result is caused by the fact that our scheme rarely employs P2P transmissions and prefers energy-saving broadcast transmissions.

6 Concluding Remark

This paper presents a novel scheme which securely forms two-hop clusters where all members are at most two hops away from each other. First, our scheme attempts to group nodes into two-hop clusters and then conserves them from separation through the help and verification of two hop distant nodes. Finally, initiators for cluster formation process are incorporated into clusters after they succeed in passing the protocol conformity check. As simulation results show, our scheme reduces the number of generated clusters as compared to Sun's scheme and enhances the cluster quality. Besides, as other simulation results show, our scheme takes more compromised nodes away from network than Sun's scheme. Last, another simulation result shows that our scheme saves more amount of energy than Sun's scheme.

References

[1] W. Heinzelman, A. P. Chandrakasan, H. Balakrishnan, IEEE Transactions on Wireless Communications, **1**, 660-670 (2002).
[2] O. Younis and S. Fahmy, IEEE Transactions on Mobile Computing, **3**, 366-379 (2004).
[3] G. Gupta and M. Younis, Proceedings of the 10th International Conference on Telecommunications, **2**, 1577-1583 (2003).
[4] G. Wang, K. Song, G. Cho, IEICE Transactions on Information and Systems, **E93-D**, 1560-1571 (2010).
[5] G. Wang and G. Cho, IEICE Transactions on Communications, **E92-B**, 612-615 (2009).
[6] M. Sirivianos, D. Westhoff, F. Armknecht, J. Girao, Proceedings of International Symposium On Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 1-10 (2007).
[7] Q. Dong and D. Liu, Proceedings of IEEE 6th Annual Communication Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 108-116 (2009).
[8] G. Wang and G. Cho, IEICE Transactions on Communications, **E93-B**, 2925-2935 (2010).
[9] A. C. Ferreira, M.A. Vilaca, L. B. Oliveira, E. Habib, H.C. Wong, A.A. Loureiro,Proceedings of the 4th IEEE International Conference on Networking, **3420**, 449-458 (2005).
[10] L. B. Oliveira, H.C. Wong, M. W. Bern, R. Dahab, A.A. Loureiro, Proceedings of the 5th IEEE International Symposium On Network Computing and Applications, 145-154 (2006).
[11] D. Liu, Proceedings of the 27th International Conference on Distributed Computing Systems, 40-48 (2007).
[12] K. Sun, P. Peng, P. Ning, C. Wang, Proceedings of the 22nd Annual Conference on Computer Security Applications, 131-140 (2006).
[13] A. Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar, Wireless Networks, **8**, 521-534 (2002).
[14] Y.C. Hu, A. Perrig, D.B. Johnson, IEEE Journal of Selected Areas in Communications, **24**, 370-380 (2006).
[15] A.S. Wander, N. Gura, H. Eberle, V. Gupta, S.C. Shantz, Proceedings of the 3rd International Conference on Pervasive Computing and Communications, 324-328 (2005).
[16] J.R. Douceur, Proceedings of the 1st International Workshop on Peer-to-peer Systems, 7-8 (2002).
[17] B. Parno, A. Perrig, V. Gilgor, Proceedings of 2005 IEEE Symposium on Security and Privacy, 49-63 (2005).

Gicheol Wang received Ph. D degree in Computer Science and Statistics from Chonbuk Nat'l University, Jeonju, Korea, in 2005. He worked for CAIIT(Center for Advanced Image and Information Technology) at Chonbuk Nat'l University, Jeonju, Korea, as a Postdoctoral Research Fellow from Jan. 2006 to Dec. 2007, for the Research Center for Ubiquitous Information Appliances at Chonnam Nat'l University, Gwangju, Korea, as a Postdoctoral Research Fellow from Jan. 2008 to Dec. 2008. From Jan. 2009, he joined to the Dept. of Computing and Networking Resources at KISTI(Korea Institute of Science and Technology Information), Daejeon, Korea, and he is currently serving as a senior research scientist. His current research interests include ad hoc networks, sensor networks, vehicular networks, network security, and mobile computing.

Sangwon Bang received the B.S., the M.S., and the Ph. D degree in Computer Science and Statistics from Chonnam University, Gwangju, Korea in 1985, 1988, and 1995 respectively. From Mar. 1987, he joined to the Department of Computer and Information at Songwon University, Gwangju, S. Korea, as a professor. His current research interests include mobile computing, software engineering, and multimedia.

Gihwan Cho received Ph. D degree in Computer Science from Newcastle upon Tyne, England in 1996. He worked for ETRI(Electronics and Telecommunications Research Institute), Daejeon, S. Korea, as a senior member of technical stuff, and from Sep. 1997 to Feb. 1999, for the Dept. of Computer Science at Mokpo National University, Mokpo, S. Korea, as a full time lecturer. From Mar. 1999, he joined to the Division of Electronics and Information Engineering at Chonbuk National University, Jeonju, S. Korea, as a professor. His current research interests include mobile computing, computer communication, security of wireless networks, sensor networks, and distributed computing system.