Communication Pattern Based Key Establishment Scheme in Heterogeneous Wireless Sensor Networks

Daehee Kim[1], Dongwan Kim[1] and Sunshin An[1]
[1] Department of Electronics Engineering, Korea University
Seoul, Korea
[e-mail: dhkim@dsys.korea.ac.kr; dongwank@korea.ac.kr; sunshin@dsys.korea.ac.kr]
*Corresponding author: Sunshin An

Received July 2, 2015; revised December 15, 2015; accepted January 13, 2016; published March 31, 2016

Abstract

In this paper, we propose a symmetric key establishment scheme for wireless sensor networks which tries to minimize the resource usage while satisfying the security requirements. This is accomplished by taking advantage of the communication pattern of wireless sensor networks and adopting heterogeneous wireless sensor networks. By considering the unique communication pattern of wireless sensor networks due to the nature of information gathering from the physical world, the number of keys to be established is minimized and, consequently, the overhead spent for establishing keys decreases. With heterogeneous wireless sensor networks, we can build a hybrid scheme where a small number of powerful nodes do more works than a large number of resource-constrained nodes to provide enhanced security service such as broadcast authentication and reduce the burden of resource-limited nodes. In addition, an on-demand key establishment scheme is introduced to support extra communications and optimize the resource usage. Our performance analysis shows that the proposed scheme is very efficient and highly scalable in terms of storage, communication and computation overhead. Furthermore, our proposed scheme not only satisfies the security requirements but also provides resilience to several attacks.

Keywords: Wireless sensor networks, communication pattern, heterogeneous wireless sensor networks, key establishment, on-demand, broadcast authentication

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST). (No. 2012K1A3A1A09026959).

http://dx.doi.org/10.3837/tiis.2016.03.017 **ISSN : 1976-7277**

1. Introduction

Wireless sensor networks (WSNs) have been getting considerable attentions from a variety of fields such as military, medical and home security [1]. To apply WSNs to these applications successfully, ensuring security is paramount. Undoubtedly, the foundation of security is to establish the required symmetric keys securely and efficiently. However, key establishment in WSNs is not so trivial since sensor nodes have severely limited capabilities in terms of energy, storage, computation, and communication [2].

A diversity of key establishment schemes in WSNs have been suggested [3]. One of the most popular schemes is a random key predistribution scheme (RKP) which preloads a set of keys selected randomly from a common pool and establishes a symmetric key between nodes with the same key [4-6]. Even though RKP is efficient and less complicated than other schemes, it has several drawbacks. First, it is not resilient to node capture attacks since keys of non-compromised nodes can be revealed. Second, a large amount of storages are required to increase the connectivity. Finally, the node authentication is not supported. LEAP [7] is another kind of key distribution mechanism developed for large scale hierarchical sensor networks which is able to establish specific keys to secure various types of unicast and broadcast traffics. Moreover, LEAP can enable the in-network processing to prevent redundant transmission and optimize the resources usage. In addition, LEAP has several strengths that it establishes a small number of deterministic keys and is resilient to a few attacks. Nevertheless, LEAP does not provide the node authentication and is not robust to node capture attacks during deployment.

In order to overcome these problems, heterogeneous WSNs and public key cryptography (PKC) based key establishment schemes have been proposed. Heterogeneous WSNs have different types of sensor nodes and are made up of a small number of powerful high-end sensor nodes (H-nodes) and a large number of low-end sensor nodes (L-nodes), e.g., typical motes. Heterogeneous WSNs offer much more significant benefits than homogeneous WSNs for a variety set of applications [8]. Key establishment scheme can also benefit from such heterogeneous WSNs by exploiting the capabilities of powerful H-nodes. Reference [9] proposes asymmetric pre-distribution key managements based on RKP where powerful H-nodes are assigned more keys and the resource constrained L-nodes get less keys. This leads to increased connectivity and reduced storage overhead in L-nodes. Reference [10] provides hybrid security mechanisms for heterogeneous WSNs. It basically acts in the same way as [9]. Besides, H-nodes can act as the key distribution center (KDC) [11] to establish deterministic keys if available. Although these schemes resolve some of above problems, they are still not resilient to node capture attacks.

PKC based key establishment schemes not only solve these problems gracefully, but also provide enhanced security services such as authentication and integrity using digital signatures. Nonetheless, applying PKC to WSNs has been considered infeasible because of its computational complexity and communication overhead. However, recent researches have demonstrated that PKC is feasible in WSNs. Especially, [12] and [13] show that elliptic curve cryptography (ECC) [14], which is more efficient than RSA [15], is computationally feasible for resource constrained sensor nodes in WSNs. Reference [16] provides an efficient and scalable ECC-based key establishment scheme which is resilient to node capture attacks and has low storage and communication overhead by using ECC. Reference [17] proposes ECC-based key management scheme together with heterogeneous WSNs. It tries to reduce

overall overhead by using heterogeneous WSNs, ECC, and routing information. However, reference [16] and [17] have still high computation overhead due to PKC.

In this paper, we propose a symmetric key establishment scheme called COKES (COmmunication pattern based Key Establishment Scheme) whose primary objective is to minimize the resource usage, such as energy and storage, while satisfying the security requirements, including confidentiality, authentication, integrity, freshness and resilience to possible attacks. Our basic idea starts from the fact that WSNs have particular communication patterns due to the nature of information gathering from the physical world. By considering the communication pattern of WSNs, the number of keys to be established is minimized and, as a result, the storage, computation and communication overhead required for establishing keys decrease. Furthermore, by adopting heterogeneous WSNs, we can build a hybrid scheme with both symmetric key cryptography (SKC) and PKC, which not only provides enhanced security but also reduces the burden of L-nodes. In addition, on-demand key establishment scheme is introduced to support additional communication and optimize the resource usage. Our main contributions are as follows.

- We consider the unique communication pattern of WSNs to minimize the number of keys to be established. By minimizing the number of keys, we can significantly reduce the storage, communication, and computation overhead spent for establishing keys.
- A hybrid key establishment scheme using heterogeneous WSNs, where H-nodes do more works than L-nodes to provide the enhanced security and conserve the resources of L-nodes, is proposed.
- In our scheme, only the fundamental keys are established during the initial key establishment stage. If additional keys become necessary to support extra communication pattern, an on-demand key establishment process is initiated. By establishing keys only when actually required, the establishment of unnecessary keys can be prevented, so that we can save the efforts spent in the key establishment process.
- Our scheme also proposes a robust hybrid broadcast authentication scheme using both PKC and SKC together. Broadcast messages are authenticated by H-nodes using PKC and authenticated by L-nodes using SKC.
- In hostile environment, there can be adversaries from the deployment stage. Unlike other schemes [18-19] that assume to be secure during the initial key establishment process, in our work, every message related to key establishments is protected by authentication tokens, such as digital signatures and message authentication codes (MACs). This makes our scheme more secure even in hostile environment since only legitimate nodes can participate in the key establishment process.

The rest of the paper is organized as follows. Section 2 provides some preliminaries prior to our proposal. In Section 3, we describe the unique communication pattern of heterogeneous WSNs. Our own key establishment scheme is proposed in Section 4. After Section 5 and 6 present the security and performance analysis about our scheme, Section 7 concludes the paper.

2. Preliminaries

2.1 Heterogeneous WSNs

Heterogeneous WSNs [8][20] consist of a small number of powerful H-nodes, such as Everlast [21], and a large number of limited L-nodes, such as MicaZ. Both H-nodes and L-nodes are

powered by batteries, but H-nodes have additional power supplies to operate for a very long time. For example, Everlast is a supercapacitor-operated, solar-powered sensor node, which can run for approximately 20 years without recharge and replacement. In contrast, MicaZ is powered by two AA batteries and can operate for only several months with low duty cycle. Hence, our goal, in terms of energy efficiency, is to minimize the energy of L-nodes, not H-nodes. L-nodes communicate with H-nodes either directly or over multi-hop transmissions. H-nodes reach a base station (BS) via other H-nodes, and they can reach all L-nodes in their cluster by one broadcast due to their long transmission range. Compared to L-nodes, H-nodes have longer transmission ranges, better computation capabilities, larger storages, more energy supplies, and better reliability. By utilizing the powerful capabilities, H-nodes can perform computationally expensive ECC operations, which will be explained in the following subsection.

Hierarchical architectures are utilized to get better scalability and energy efficiency. All H-nodes form a backbone in the heterogeneous WSN which is divided into multiple clusters where each H-node serves as a cluster head (CH). From now on, H-nodes and CHs are interchangeably used. One example of clustering schemes for heterogeneous WSNs can be found in [22]. In heterogeneous WSNs, messages are routed through 2-level hierarchy where the first level is routing between a BS and CHs, called inter-cluster routing, and the second level is routing inside the cluster, called intra-cluster routing. For example, when a message is routed from a BS to a L-node, the message is firstly relayed to the corresponding CH via other CHs over multi-hop communications, and then the CH delivers the message to the L-node in its cluster directly. **Fig. 1** illustrates the architecture of heterogeneous WSNs and routing operations.

2.2 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) [14] is a type of PKC based on elliptic curves over finite fields. ECC is based on the intractability of elliptic curve discrete logarithm problem (ECDLP) which is the equivalent of discrete logarithm problem (DLP) [23] used by RSA [15]. The important thing is that ECDLP is much harder than DLP, and thus ECC needs smaller key size than RSA at the comparable levels of security. For example, ECC with a key size of 160 bits can provide the same level of security as RSA with 1024 bits. Because of the smaller key size, ECC is computationally efficient and can be applied to resource constrained WSNs.

Elliptic curve Diffie-Hellman (ECDH) [24] is a key agreement protocol which allows two parties to share a symmetric key over an insecure channel, using ECC. Let A and B denote the nodes who want to establish a symmetric key with each other, and (x_A, Q_A) and (x_B, Q_B) denote the private and public key pairs of A and B, respectively. To establish a symmetric key, each node sends its public key to the other node. Using the received public key and its private key, each node can establish a symmetric key as follows.

$$A: K_{A,B} = x_A \times Q_B = x_A \times x_B G = x_A x_B G$$

$$B: K_{A,B} = x_B \times Q_A = x_B \times x_A G = x_B x_A G = x_A x_B G$$

Even if the adversary overhears the exchanged messages, it is computationally infeasible to determine the symmetric key unless he can solve ECDLP. However, ECDH requires an additional authentication scheme since it is vulnerable to the man-in-the-middle attack.

2.3 Adversary Model

In general, the primary goal of the adversary is either to get secret information from WSNs or to prevent WSNs from gathering useful information. The best way to attain this goal is to

obtain secret keys and join WSNs. To this end, the adversary is assumed to be able to eavesdrop, modify, drop, inject, and replay messages. Furthermore, the adversary can even tamper a node physically. In this case, he can extract all information from the compromised node, including secret keys, data, and even code. Unlike other schemes [18-19] which assume the adversary is not present during deployment, we assume that the adversary even exists during the initial deployment stage. Hence, the adversary tries to join WSNs during deployment, or interrupt the key establishment process.

3. Communication Patterns in Heterogeneous WSNs

Unlike traditional mobile ad hoc networks, WSNs have asymmetric communication patterns between downlink (DL) and uplink (UL) due to the nature of information gathering from the physical world. DL refers to the direction from a BS to sensor nodes, while UL refers to the reverse direction. Both UL and DL traffic require authentication, integrity, and confidentiality since messages can contain critical data. In this section, we describe three basic communication patterns in heterogeneous WSNs in detail. **Fig. 1** shows an example of three communication patterns in heterogeneous WSNs.

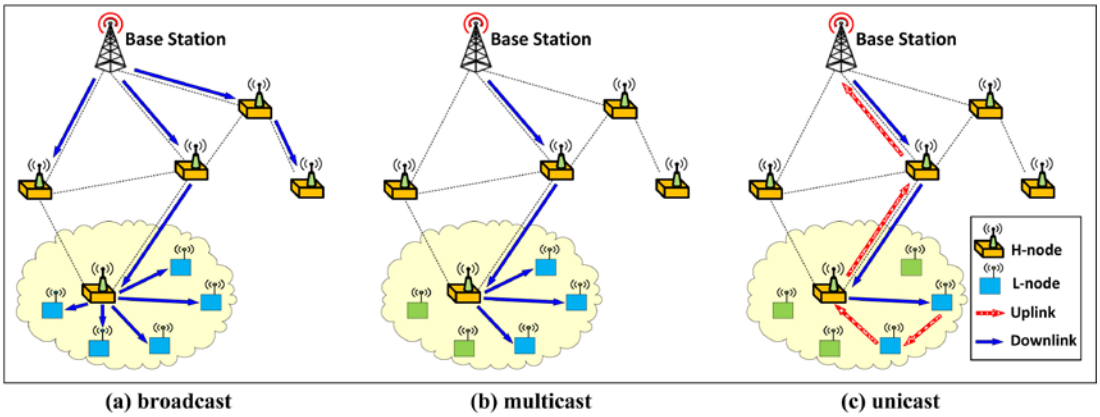


Fig. 1. Communication patterns in heterogeneous WSNs.

3.1 Broadcast

Broadcast is to disseminate messages from a sender to all receivers. In WSNs, broadcast is mainly used to send queries or commands from a BS to sensor nodes. In heterogeneous WSNs, broadcast consists of two levels. Firstly, messages are broadcasted to all CHs, and then, messages are broadcasted inside the cluster. In the first level, it takes multi-hop communications to get to each CH from a BS. In the second level which is from a CH to L-nodes, each CH broadcasts messages to every L-node in its cluster. These broadcast messages reach L-node directly using the long transmission range of the CH

3.2 Multicast

Multicast is to send messages to a group of receivers. The group can be formed in various ways. In WSNs, the group is usually comprised of the nodes with the same conditions, or the nodes within specific geographic regions. In heterogeneous WSNs, there can be two cases of multicast according to the multicast group. The first case is that the multicast group is equivalent to one or more clusters, and the second case is that the multicast group does not

correspond to a cluster which means that some of the nodes in the cluster are group members, but the others in the cluster are not. Assuming the first case, a BS unicasts messages to CHs which have multicast group members, and subsequently each CH broadcasts them to all L-nodes which are group members. In the second case, messages are respectively unicasted or multicasted to the group members inside the cluster instead of broadcast.

3.3 Unicast

Unicast is to transmit messages from a sender to a single receiver. In WSNs, unicast is used for a BS sending queries to a single node in DL and a sensor node reporting data back to a BS in UL. In heterogeneous WSNs, unicast messages are routed as **Fig. 1** (c). In DL, unicast messages are delivered to a L-node over one-hop communication after being routed to a CH over multi-hop communications via other CHs. In UL, messages are routed to a CH over multi-hop transmissions via other L-nodes, and then messages are transmitted to a BS over multi-hop communications through other CHs.

In terms of security, unicast can be achieved in two different ways. The first way is to communicate directly between a BS and a L-node, which means that both a BS and a L-node share a key. The second one is to pass through a CH between a BS and a L-node, which indicates that a BS shares a key with a CH which shares another key with a L-node. The first method is suitable for DL since it prevents a CH from eavesdropping and modifying data, and also removes encryption and decryption overhead of a CH. In contrast, the second method is suitable for UL since it can enable data aggregation in the CH.

4. COKES: COmmunication pattern based Key Establishment Scheme

In this section, we propose our key establishment scheme named COKES. Prior to this, keys to be established are defined first, and then security requirements and system setup for our scheme are described.

4.1 Keys in COKES

Based on the communication patterns described in the previous section, the followings keys are derived to support the basic communications in WSNs. One thing to keep in mind is that a broadcast key, shared by more than two nodes, can provide confidentiality only while a unicast key between two nodes provides both confidentiality and authentication. This is because any node with a broadcast key can generate a valid MAC. Therefore, we need separate keys for broadcast authentication and broadcast confidentiality.

- *Individual unicast key* : a unicast key between a BS and each node, including both H-nodes and L-nodes
- *Intra-cluster unicast key* : a unicast key between a CH and each L-node
- *Inter-cluster broadcast authentication key* : a key for providing broadcast authentication between a BS and CHs
- *Intra-cluster broadcast authentication key* : a key for providing broadcast authentication between a CH and L-nodes
- *Inter-cluster broadcast encryption key* : a key for providing broadcast confidentiality between a BS and CHs
- *Intra-cluster broadcast encryption key* : a key for providing broadcast confidentiality between a CH and L-nodes

Note that multicast keys are not established during deployment because multicast groups change dynamically.

4.2 Security Requirements and System Setup

Under our adversary model, our scheme aims to satisfy four basic security requirements, which are authentication, integrity, confidentiality and freshness. In addition, resilience to several possible attacks has to be guaranteed to serve as the foundation of security in WSNs. Prior to the deployment, a server, such as a BS, creates the keying materials required for the key establishment procedure and preloads them into each node. Each node *i*, regardless of H-nodes and L-nodes, preloads the following keying materials prior to the deployment.

- An individual unicast key shared with a BS, ($K_{BS,i}$)
- A private and public key pair for ECC, (x_i, Q_i)
- A BS's public key, (Q_{BS})
- A certificate, ($Cert_i$)

In addition, a BS preloads the following key, which is used for the inter-cluster broadcast encryption, on all H-nodes.

- An inter-cluster broadcast encryption key shared by all H-nodes and a BS, (K_{BE})

Table 1 shows all other notations used in our scheme.

Table 1. Notations

Notation	Description
ID_i	the identity of a node *i*
x_i	the private key of a node *i*
Q_i	the public key of a node *i*, which is x_iG
$Cert_i$	the certificate of a node *i*
$K_{i,j}$	the symmetric key between a node *i* and a node *j*
H	the secure one-way hash function
$sig_x(m)$	the digital signature of *m* signed by a node *i* using ECDSA
$MAC_K(m)$	the message authentication code of *m* using a key *K*
$E_K(m)$	the encryption of *m* using a key *K*
ts	the timestamp
seq	the sequence number
R, r	the transmission range of a H-node and a L-node
d	the distance between a CH and a L-node
N_C	the number of nodes in a cluster
S_L, M_L, C_L	the storage, communication, and computation overhead of a L-node
S_H, M_H, C_H	the storage, communication, and computation overhead of a H-node
S_{LEAP}, M_{LEAP}, C_{LEAP}	the storage, communication, and computation overhead of a node in LEAP
$size_{pub}$, $size_{sym}$	the size of a public key and a symmetric key
$size_{cert}$	the size of a certificate for ECC
L	the length of a one-way key chain
deg	the number of neighbor nodes within the transmission range
$C^{ECDSAsig}$	the computation cost of the ECDSA signing
$C^{ECDSAver}$	the computation cost of the ECDSA verification
C^{ECDH}	the computation cost of the ECDH
C^{enc}, C^{dec}	the computation cost of the encryption and the decryption
C^{MAC}, C^{hash}	the computation cost of the MAC and the hash function

4.3 Unicast Key Establishment

Since we assume that the adversary can exist from beginning in WSNs, authentication tokens such as digital signatures and MACs are used in every message exchanged during the key establishment process in order to authenticate the involving nodes. To reduce the computational overhead of L-nodes, a hybrid scheme is used where CHs are authenticated by digital signatures based on ECDSA [25] and L-nodes by a MAC based on SKC. As presented in previous subsection, two unicast keys, which are an individual unicast key and an intra-cluster unicast key, must be generated. Since an individual unicast key is preloaded before deployment, we only need to establish intra-cluster unicast keys.

When sensor nodes are deployed in the field, our key establishment process is started together with clustering. Each H-node broadcasts a message to announce that it is a legitimate H-node. To prove its validity, a digital signature and a certificate based on ECDSA are included in the message. In our scheme, a certificate is composed of an ID of the node, a public key of the node, a type of the node which denotes if the node is a L-node or a H-node, and a digital signature signed by a BS.

$$CH \rightarrow * : < ts,\ ID_{CH},\ sig_{x_{CH}}(H(ts \mid ID_{CH})),\ Cert_{CH} >$$

Upon receiving the message, each L-node verifies a timestamp, a certificate and a digital signature sequentially. First, each L-node verifies the timestamp. If |current time – ts| > T, where T is the maximum allowed time difference, the message is discarded. Next, the certificate is verified using a BS's public key preloaded before deployment. If the certificate is not correct, the message is discarded. Finally, the digital signature is verified using the public key acquired from the certificate. If the message passes all verification processes, the L-node is assured that the message is not modified and comes from a legitimate H-node, which is accepted as its CH. Then, an intra-cluster unicast key between itself and the CH is established using ECDH as follows.

$$K_{CH,i} = x_i \times Q_{CH} = x_i x_{CH} G$$

After establishing the key, each L-node i responds to the CH in order to join the cluster. Since L-nodes are computationally constrained, a MAC, instead of a digital signature, is attached to the message using a newly established key to prove that it is a legitimate L-node.

$$i \rightarrow CH : < ts, ID_i, MAC_{K_{CH,i}}(ts \mid ID_i)),\ Cert_i >$$

Even though L-node does not use a PKC for this message, it has to include its certificate in order to provide the CH with its own public key. When the CH receives this message, it checks out a timestamp and a certificate. If the message is verified successfully, the CH establishes an intra-cluster unicast key using ECDH.

$$K_{CH,i} = x_{CH} \times Q_i = x_{CH} x_i G = x_i x_{CH} G$$

With this key, it verifies a MAC in the received message. If the MAC is verified successfully, the CH is assured that the message is originated from a legitimate L-node and is not altered in transit. As a result, the L-node is accepted as a new cluster member.

In case that a new node is added to the existing WSNs, it can establish an intra-cluster unicast key in the same way as the initial key establishment process

4.4 Broadcast Key Establishment

Now that unicast keys between a CH and each L-node have been established, the remaining thing to do is to establish broadcast keys. As mentioned earlier, a single broadcast key cannot

provide both confidentiality and authentication. Hence, our scheme establishes two kinds of broadcast keys. One is a key for broadcast authentication, and the other is a key for broadcast confidentiality.

First, we consider a key for broadcast authentication. Since broadcast traffic is mainly composed of queries and commands, ensuring authentication is more important than ensuring confidentiality. This is why broadcast authentication has been actively studied in WSNs [26-28]. The easiest way to provide broadcast authentication is to use digital signatures. However, verifying a digital signature for every broadcast message may be heavy burden to L-nodes even though ECDSA is more lightweight than DSA [25]. To resolve this problem, we propose a hybrid broadcast authentication scheme which uses ECDSA from a BS to each CH and SKC based scheme from a CH to L-nodes. Since every CH has a BS's public key, there is no need to establish a new inter-cluster broadcast authentication key between a BS and CHs. The remaining thing to do is to establish an intra-cluster broadcast authentication key between a CH and L-nodes. One plausible solution is to establish a cluster key shared by all cluster members, including a CH and L-nodes. However, as stated earlier, a MAC created using a cluster key does not guarantee that the message is really from the CH since a cluster key is shared by all cluster members, not owned by the CH alone. Instead, one-way key chain [29] is used in our work. One-way key chain is a series of keys $\{K^0, K^1, K^2, \ldots, K^n\}$ where $K^i = H(K^{i-1})$ and $K^0 = H(seed)$ with a randomly chosen value, *seed*. H can be any kind of hash function. Contrary to the generation order, keys are used in the reverse order, in other words, $\{K^n, K^{n-1}, K^{n-2}, \ldots, K^0\}$. Since the generator of the key chain only knows the next key due to the one-wayness, the receivers can be assured that it is from the intended sender. However, in multi-hop environment of WSNs, this is not true since intermediate nodes can manipulate the message when relaying to next nodes. Our scheme overcomes this problem by combining a one-way key chain and heterogeneous WSNs where CHs can broadcast messages to all L-nodes with one hop using its long transmission range. Since broadcast messages are received from a CH directly and only a CH knows a next intra-cluster broadcast authentication key, each L-node can be assured that it is from the real CH.

Broadcast confidentiality can be achieved by using a shared key by all nodes. Since broadcast consists of two levels as described in subsection 3.1, our scheme establishes two different levels of keys which are an inter-cluster broadcast encryption key and an intra-cluster broadcast encryption key. Since an inter-cluster broadcast encryption key is already preloaded in each H-node, we have only to establish an intra-cluster broadcast encryption key which is established together with an intra-cluster broadcast authentication key.

After establishing unicast keys with L-nodes, the CH generates a one-way key chain with a length of L using a randomly chosen value, *seed*, as an intra-cluster authentication key.

$$K^i = H(K^{i-1}), \quad where\ K^0 = H(seed),\ 1 \le i \le L$$

In addition, the CH randomly generates an intra-cluster broadcast encryption key, K_{BE}, which is used to encrypt broadcast messages and shared by all cluster members. The CH delivers K^L and K_{BE} to each L-node securely using an intra-cluster unicast key. Since keys must be confidential to the L-nodes in the cluster, they are encrypted and then a MAC is attached to ensure the authentication and integrity.

$$CH \rightarrow each\ L\text{–}node\ i:$$

$$<ts, ID_{CH}, E_{K_{CH,i}}(K^L), E_{K_{CH,i}}(K_{BE}), MAC_{K_{CH,i}}(ts|ID_{CH}|E_{K_{CH,i}}(K^L)|E_{K_{CH,i}}(K_{BE}))>$$

Upon receiving the message, each L-node verifies a timestamp, the CH's ID, and a MAC. If all verifications succeed, the L-node decrypts an intra-cluster broadcast authentication key and an intra-cluster broadcast encryption key using an intra-cluster unicast key between itself and the CH and stores them for the subsequent broadcast message.

When the CH has a message, *m*, to broadcast, it broadcasts a message together with a next key, K^{i-1}, encrypting with K_{BE}.

$$CH \rightarrow *: <E_{K_{BE}}(m, seq, ID_{CH}, K^{i-1})>$$

When each L-node receives a broadcast message, it decrypts the message with K_{BE}, extracts a next broadcast key, K^{i-1}, and verifies that $K^i = H(K^{i-1})$. If so, the L-node is assured that the message is originated from the CH. Finally, K^{i-1} is updated as a current intra-cluster broadcast authentication key. The sequence number is used for indicating packet losses which are very frequent in WSNs. By comparing a new sequence number to the sequence number of the last received packet, we can identify packet losses and compute the number of lost packets, N_{loss}. In spite of packet losses, our scheme can still authenticate the new message by applying a hash function to the new intra-cluster broadcast authentication key $N_{loss}+1$ times and comparing it with a current intra-cluster broadcast authentication key.

$$\underbrace{H(H(\cdots(H(K^{new}))))}_{N_{loss}+1} = K^{current}$$

where K^{new} and $K^{current}$ denote a newly received intra-cluster broadcast authentication key and a current intra-cluster broadcast authentication key, respectively.

When all intra-cluster broadcast authentication keys in a one-way key chain are used up, a new key chain is generated by the CH, and then the lastly generated key is distributed to each L-node using an intra-cluster unicast key in the same way as initial deployment. In case that a new L-node joins the WSNs, the CH sends the current intra-cluster broadcast authentication key and intra-cluster broadcast encryption key to the new L-node encrypting with an intra-cluster unicast key which has been already established.

If a node in the cluster is compromised, an intra-cluster broadcast encryption key must be updated with a new one, K_{BE}^{new}, as follows. Note that an intra-cluster broadcast authentication key does not have to be updated even when a node is compromised.

$$CH \rightarrow each\ L\text{–}node\ i: <ts, ID_{CH}, E_{K_{CH,i}}(K_{BE}^{new}), MAC_{K_{CH,i}}(ts|ID_{CH}|E_{K_{CH,i}}(K_{BE}^{new}))>$$

4.5 On-demand Key Establishment

Once all kinds of basic keys based on the communication pattern are established, every kind of communication can be started using the established keys. Note that every message is encrypted or authenticated using the symmetric key algorithms with the established shared keys. This is sufficient for most applications. However, there can be very rare case where some applications require another kind of communications in addition to the basic communications. In this case, we can support extra communications using the established keys even though it is rather inefficient. For example, if a L-node wants to communicate with other L-node in the same cluster, it simply sends the message to a CH using an intra-cluster unicast key. Then, the CH relays the message to the other L-node using an intra-cluster unicast key. However, this can incur longer delay and more energy consumptions since it takes a longer path than a direct communication between two L-nodes. To improve the efficiency, our scheme proposes an on-demand key establishment scheme where each CH plays the role of a KDC.

Suppose a simple case where two L-nodes, *i* and *j*, in the same cluster want to communicate with each other directly. To establish a symmetric key between *i* and *j*, *i* sends a request message to the CH using an intra-cluster unicast key.

$$i \rightarrow CH : <ts,\ ID_i, ID_j, MAC_{K_{CH,i}}(ts \mid ID_i \mid ID_j)>$$

Upon receiving the message, the CH verifies a timestamp and a MAC, and then randomly generates a new key, K_{ij}, for *i* and *j*, which is transferred to *i* and *j* using an intra-cluster unicast key, respectively.

$$CH \rightarrow j : <ts, ID_{CH}, ID_i, ID_j, E_{K_{CH,j}}(K_{i,j}), MAC_{K_{CH,j}}(ts \mid ID_{CH} \mid ID_i \mid ID_j \mid E_{K_{CH,j}}(K_{i,j}))>$$

$$CH \rightarrow i : <ts, ID_{CH}, ID_i, ID_j, E_{K_{CH,i}}(K_{i,j}), MAC_{K_{CH,i}}(ts \mid ID_{CH} \mid ID_i \mid ID_j \mid E_{K_{CH,i}}(K_{i,j}))>$$

From now on, L-node *i* and *j* can communicate with each other directly without passing through the CH, using the newly established key. Our on-demand key establishment scheme can be easily extended to establish any kind of key, e.g., a multicast key which is shared by multicast group members. **Fig. 2** (a) shows the procedure to establish a symmetric key between L-nodes *i* and *j* which belong to the same cluster on demand. It is worth noting that if necessary, the key between L-nodes belonging to two different clusters can also be established as depicted in **Fig. 2** (b). Except that the transmission between two CHs is included, the procedure is almost the same as the key establishment between L-nodes in the same cluster. Supposing that node *i*, whose CH is CH_i, wants to set up a key with node *j*, whose CH is CH_j, the detailed procedure is presented as follows.

$$i \rightarrow CH_i : <ts, ID_i, ID_j, MAC_{K_{CH_i,i}}(ts \mid ID_i \mid ID_j)>$$

$$CH_i \rightarrow CH_j : <ts, ID_{CH_i}, ID_i, ID_j, E_{K_{CH_i,CH_j}}(K_{i,j}), MAC_{K_{CH_i,CH_j}}(ts \mid ID_{CH_i} \mid ID_i \mid ID_j \mid E_{K_{CH_i,CH_j}}(K_{i,j}))>$$

$$CH_i \rightarrow i : <ts, ID_{CH_i}, ID_i, ID_j, E_{K_{CH_i,i}}(K_{i,j}), MAC_{K_{CH_i,i}}(ts \mid ID_{CH_i} \mid ID_i \mid ID_j \mid E_{K_{CH_i,i}}(K_{i,j}))>$$

$$CH_j \rightarrow j : <ts, ID_{CH_j}, ID_i, ID_j, E_{K_{CH_j,j}}(K_{i,j}), MAC_{K_{CH_j,j}}(ts \mid ID_{CH_j} \mid ID_i \mid ID_j \mid E_{K_{CH_j,j}}(K_{i,j}))>$$

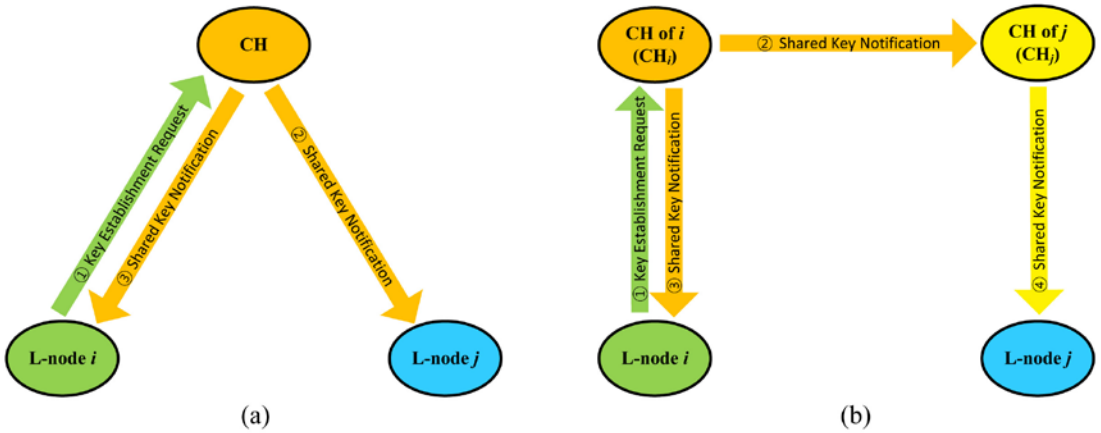


Fig. 2. On-demand key establishment between L-node *i* and *j* (a) when *i* and *j* belong to the same cluster, (b) when *i* and *j* belong to the different clusters.

4.6 Key Refreshment

The keys must be refreshed periodically to prevent various attacks, such as cryptanalytic attacks, in such a way that the key refreshment scheme provides past key secrecy and future key secrecy [30].

In COKES, the keys to be refreshed are an individual unicast key, an intra-cluster unicast key, an inter-cluster broadcast encryption key, and an intra-cluster encryption key. We use a different mechanism depending on whether the key is a unicast key or a broadcast key.

For the unicast keys, a new session key between node *A* and *B* in the *k*-th session ($k \geq 2$), $K_{A,B}^{k}$, is updated as follows.

$$A \rightarrow B: < ts, A, B, k, H(sum_{k-1}), MAC_{K_{A,B}^{k-1}}(ts \mid A \mid B \mid k \mid H(sum_{k-1})) > \quad (1)$$

$$B \rightarrow A: < ts, B, A, H(K_{A,B}^{k}), MAC_{K_{A,B}^{k-1}}(ts \mid B \mid A \mid H(K_{A,B}^{k})) > \quad (2)$$

where $sum_{k-1} = MAC_{k-1}^{1} \oplus MAC_{k-1}^{2} \oplus \cdots \oplus MAC_{k-1}^{N}$, and MAC_{k-1}^{n} is a MAC of the *n*-th message exchanged between node *A* and *B* in the session *k*-1. In other words, sum_{k-1} denotes the XOR sum of MACs in all exchanged messages during the session *k*-1.

Node *A*, who wants to refresh the session key, begins the process by sending the hash value of sum_{k-1} as shown in Eq. 1. Upon receving the message, node *B* verifies a timestamp, a MAC and sum_{k-1}, and then derives a new session key as follows.

$$K_{A,B}^{k} = H(K_{A,B}^{k-1} \mid sum_{k-1}) \quad (3)$$

When node *B* replies to node *A* with a hash value of the new session key as shown in Eq. 2, node *A* can be assured that the unicast key refreshment is successful by verifying the hash value of the new session key. Once a new session key is computed, node *A* and *B* delete the previous session key to prevent the adversary from extracting the previous session key after compromising. The use of MACs for the key refreshement brings us future key secrecy which will be explained in detail in Section 5.4. Past key secrey is ensured by the hash function used in Eq. 3. It is important to note that sum_{k-1} must be identical in both nodes to generate the same key. This can be easily guaranteed for unicast messages by using acknowledgements.

In contrast to unicast messages, since broadcast messages are not guaranteed to be received by all nodes due to the overhead of acknowledgements, the use of MACs for refreshing the broadcast keys might not be possible. Thus, we take a different method that uses a one-way key chain employed from [30]. The BS and the CH generate its own one-way key chain to be used for refreshing the key as follows. Note that the BS is reponsible for refreshing an inter-cluster broadcast encryption key, and the CH is responsible for refrehsing an intra-cluster broadcast encryption key, respectivley.

$$K_{ref}^{i} = H(K_{ref}^{i-1}), \qquad where\, K_{ref}^{0} = H(seed),\ 1 \leq i \leq L$$

Node *A*, which is either a BS or a CH, initiates the broadcast key refreshment for the session *k* ($k \geq 2$) as follows.

$$A \rightarrow B: < ts, A, B, k, E_{K_{A,B}}(K_{ref}^{L-k}), MAC_{K_{A,B}}(ts \mid A \mid B \mid k \mid E_{K_{A,B}}(K_{ref}^{L-k})) > \quad (4)$$

$$B \rightarrow A: < ts, B, A, H(K_{BE}^{k}), MAC_{K_{A,B}}(ts \mid B \mid A \mid H(K_{BE}^{k})) > \quad (5)$$

where K_{BE}^{k} is a new broadcast encryption key for the *k*-th session, node *B* is either each H-node for the inter-cluster broadcast encyption key or each L-node for the intra-cluster broadcast encryption key, and $K_{A,B}$ is a unicast key between node *A* and *B*.

Node *A* begins the process by sending an encrypted K_{ref}^{L-k}. Upon receving the message, node *B* verifies a timestamp, a MAC and K_{ref}^{L-k} by performing $H(K_{ref}^{L-k}) = K_{ref}^{L-k-1}$, and then derives a new session key as follows.

$$K_{BE}^{k} = H(K_{BE}^{k-1} \mid K_{ref}^{L-k}) \tag{6}$$

When node *B* replies to node *A* with a hash value of the new session key as shown in Eq. 5, node *A* can be assured that the broadcast key refreshment is successful by verifying the hash value of the new session key. When the key freshment is done, node *A* and *B* delete the previous session key to prevent the adversary from extracting the previous session key after compromising. Since K_{ref}^{L-k} is kept secret in node *A* only, future key secrecy is guaranteed. The use of a hash function in Eq. 6 privdes past key secrecy. The more detailed explanation about past and future key secrecy will be given in Section 5. 4.

5. Security Analysis

In this section, we analyze our proposed key establishment scheme in terms of security. After investigating that our scheme can satisfy the security requirements, the resilience to attacks and the security of broadcast authentication is analyzed in detail.

5.1 Basic Security Requirements

Our key establishment scheme satisfies four basic security requirements which are authentication, integrity, confidentiality, and freshness. Authentication and integrity are guaranteed by attaching a digital signature with certificates and a MAC on each message. By encrypting the new keys using the established keys, the key itself is not exposed to the adversary, that is, confidentiality is ensured. Freshness is achieved by a timestamp.

5.2 Resilience to Attacks

First possible attack is a man-in-the-middle attack since our scheme for establishing unicast keys is based on ECDH. This is prevented by using a certificate. Since a BS is assumed to be trusted and thus a certificate of each node cannot be forged, the adversary is not able to do the man-in-the-middle attack as long as he does not have private keys of the legitimate nodes.

Most critical attacks in WSNs are node capture attacks. In WSNs, resilience to node capture attacks means that given some compromised nodes, the information of non-compromised nodes is not revealed to the adversary. In our scheme, there can be two types of node capture attacks which are L-node capture attacks and H-node capture attacks. First, let us suppose that a L-node is captured by an adversary. In this case, all information of the L-node, including all its keys which are a private key, a individual unicast key with a BS, an intra-cluster unicast key with a CH, an intra-cluster broadcast authentication key and an intra-cluster broadcast encryption key, is disclosed to the adversary. Every key, except for two broadcast keys, is used only between itself and other node. Therefore, they do not reveal any information about non-compromised nodes. In other words, the adversary is not able to do anything to disrupt the communication between other nodes except the compromised node itself since the adversary does not know any key of other nodes. For an intra-cluster broadcast authentication key, a compromised L-node cannot also reveal any information since the adversary is not able to predict the next broadcast key used for the next broadcast message. In case of an intra-cluster broadcast encryption key, a compromised node shares the key with all nodes in the cluster, and thus the adversary can eavesdrop broadcast messages in the cluster. However, this damage is restricted to the cluster only and broadcast authentication, which is more important than broadcast confidentiality, is still guaranteed.

Next, let us think about H-node capture attacks which are more serious since H-nodes play a critical role and also serve as CHs in heterogeneous WSNs. Due to the importance of H-nodes, the best way to prevent H-node capture attacks is for every H-node to be equipped with a tamper resistant hardware which is also assumed in other works [9][10][17]. This is reasonable since the number of H-nodes is assumed to be much smaller than that of L-nodes. Without a tamper resistant hardware, just as L-node capture attacks, all of the information is leaked to the adversary. These include a private key, an individual unicast key with a BS, intra-cluster unicast keys with all L-nodes in the cluster, an inter-cluster broadcast encryption key, a one-way key chain for intra-cluster broadcast authentication and an intra-cluster broadcast encryption key. Definitely, every communication related to the cluster is affected. However, this is limited to the cluster only. This means that communications between nodes which do not belong to the cluster is not influenced. Furthermore, as soon as H-node capture attacks are detected, the damage can be minimized by letting L-nodes use an individual unicast key shared with a BS instead of an intra-cluster unicast key shared with the CH and by changing their CH to another H-node. In summary, H-node capture attacks are more critical than L-node capture attacks. Hence, the best way to prevent attacks is to be equipped with a tamper resistant hardware. Even without the hardware, the effect is limited to the cluster itself, and data reported by L-nodes can be protected by using an individual unicast key shared with a BS. Note that when a H-node is compromised, other aspects, such as communication and routing, are surely affected, but this is common to every kind of cluster-based WSNs.

Another possible attack is that after compromising a L-node, the adversary lets the L-node impersonate a legitimate H-node to get information from a group of L-nodes. We call this attack a *H-node impersonation attack*. This attack can be avoided by adding a type of the node to a certificate. Suppose that the malicious adversary compromises a L-node and reprograms it to act as a H-node. The reprogrammed L-node announces that it is a legitimate H-node. Upon receiving a message, each L-node can easily detect the fake by verifying the certificate. If the adversary did not modify a type of the node which indicates L-node, each L-node detects it is not a valid H-node. Even though the adversary modified a type of the node to ‘H-node’, each L-node is still able to find out that it is not a legitimate H-node by verifying a digital signature since the adversary cannot forge a new certificate without a BS’s private key.

5.3 Broadcast Authentication

Our scheme provides an efficient and robust broadcast authentication by combining PKC and SKC in heterogeneous WSNs. In inter-cluster broadcast, a broadcast message is authenticated by a digital signature signed by a BS. Since we assume that a BS’s private key is kept secret, the broadcast message is securely authenticated in each CH with its preloaded BS’s public key. In intra-cluster broadcast, a broadcast message is authenticated by a one-way key chain generated by a CH. Since it is computationally infeasible to estimate the next key used for the next message other than the CH, each L-node can be assured that the message is originated from a real CH by verifying $K^i = H(K^{i-1})$. In addition, with the capability of a powerful CH, every intra-cluster broadcast message reaches all L-nodes through one-hop communication. This prevents intermediate L-nodes between the CH and other L-nodes from impersonating as the CH. Besides, some DoS attacks where compromised nodes try to inject false broadcast message, are limited within one hop range. Furthermore, as mentioned in subsection 4.4, our scheme is robust to packet losses which occur frequently in WSNs. Lastly, our scheme does not have several vulnerabilities of μTESLA [26] due to the delay in message authentication since our scheme provides immediate broadcast authentication.

5.4 Past and Future Key Secrecy

Another important security requirements in the key management scheme are past key secrecy and future key secrecy, which are defined as follows [30].

- Past key secrecy: When the current session key is exposed, the past keys should not be revealed.
- Future key secrecy: When the current session key is exposed, the future keys should not be revealed.

We assume that the adversary knows the current session key and can even overhear key refreshment messages in all sessions.

For the unicast keys, a new session key $K_{A,B}^{k}$ is derived from both the current session key and XOR sum of MACs in all exchanged messages during the current session, sum_{k-1}. As you can see in Eq. 1, a hash value of sum_{k-1}, not sum_{k-1} itself, is transferred over the air. Thus, the adversary cannot know a new session key even though overhearing the key refreshment message since sum_{k-1} can be computed only by monitoring every message between node *A* and *B*. Therefore, we can say that our unicast key refreshment scheme provides future key secrecy as long as the adversary does not monitor every exchanged message between node *A* and *B*. Note that the adversary can definitely know the future session key with seamless monitoring after compromising a node, but this cannot be prevented in any key management scheme. Past key secrecy is easily achieved by using a hash function in Eq. 3. Due to the one-way property of a hash function, the adversary cannot find out the past session key even though he knows the current session key.

In case of the broadcast keys, a new session key K_{BE}^{k} is derived using both the current session key, K_{BE}^{k-1}, and a one-way refresh key, K_{ref}^{L-k}, from the broadcast sender. Even though the current session key is revealed, the adversary cannot derive a new session key because he does not know a unicast key between node *A* and *B*, $K_{A,B}$ as shown in Eq. 4. Furthermore, the use of a one-way refresh key prevents any malicious node from initiating the key refreshment since the malicous node cannot know the next refresh key due to its one-way property. However, our scheme fails to provide future key secrecy when the node is physically compromised and the adversary hears the key refreshment messages. Note that this is the same level of security as the group key update protocol in [30]. Similar to the unicast keys, past key secrecy is easily provided by the use of a hash function in Eq. 6.

6. Performance Analysis

In this section, we first investigate the performance of our on-demand key establishment scheme which can incur communication overhead and delay. Then, COKES is analyzed in terms of storage, communication and computation overhead, and compared to LEAP which has very low overhead. Finally, scalability, which is one of the most significant performance metrics in WSNs, is discussed.

6.1 On-demand Key Establishment

As you expect, on-demand key establishment may incur communication overhead and delay even though it prevents the establishment of unnecessary keys. To evaluate communication overhead and delay caused by our scheme, we compare our on-demand key establishment scheme with other popular schemes such as a pairwise key establishment scheme in LEAP and

a KDC-based key establishment scheme [11]. Note that the goal of this comparison is to show that our scheme is not superior to others, but rather not inferior to others despite the inherent weaknesses of the on-demand scheme.

For simplicity, suppose that two L-nodes in the same cluster want to communicate with each other. In this case, our scheme needs one transmission (TX) and two receptions (RXs) in L-nodes, and one RX and two TXs in a CH. A pairwise scheme requires two TXs and two RXs. KDC-based scheme demands three TXs and three RXs. Hence, our scheme is worse than a pairwise scheme, and the same as a KDC-based scheme. If communications of L-nodes alone are considered, our scheme is better than both schemes. This is reasonable since H-nodes are assumed to have plenty of energies. Therefore, we can say that our on-demand key establishment scheme has similar or smaller communication overhead than other schemes.

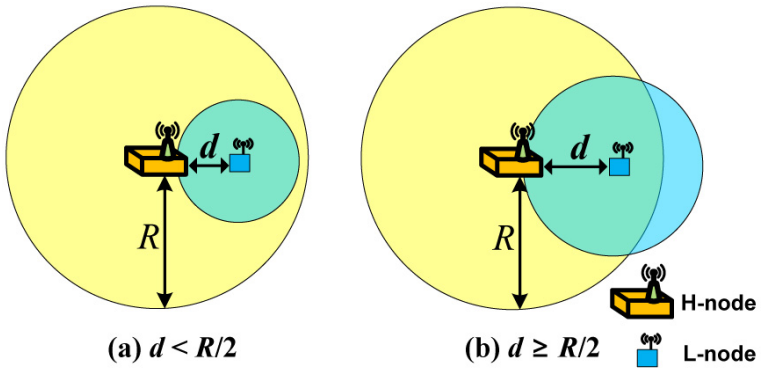


Fig. 3. Ideal models for computing delay in heterogeneous WSNs.

Next, let us think about delay which usually occurs in most on-demand protocols. For example, ad-hoc on-demand distance vector routing (AODV) [31], which is one of the most famous ad hoc routing protocols, has to set up a path on demand before sending a message, and thus the message is delayed until the path is set up. However, our scheme does not have to wait till the key setup is completed since the actual data message can be delivered together with the key setup message using the established keys. Hence, actual delay is believed to be slightly longer than other schemes which have already a direct pairwise key since our message is traversed via a CH. In our heterogeneous WSNs, messages are delivered from a L-node to the CH over multi-hop communications, and then the CH sends messages to the L-node directly, not over multi-hop communications as shown in **Fig. 1**. Surprisingly, this makes our scheme have rather shorter average delay than other schemes where nodes directly communicate. We prove this by showing that the average number of L-nodes, with which a L-node can directly communicate with shorter delay than passing through the CH, is less than half of all L-nodes in the cluster. We first assume that the number of hops between the CH and a L-node is proportional to the distance d between the CH and a L-node for brevity. This implies that every pair of two nodes with the same distance has the same number of hops, and thus has the same delay. For each L-node, direct communications with other L-nodes, which are within a circle with a radius d, have shorter delay than detouring through the CH, since the nodes outside the circle can be reached by the CH over one-hop communication. This is illustrated in **Fig. 3** where the circle with a radius R, which is a transmission range of a CH, denotes a cluster and the circle with a radius d denotes the set of L-nodes with shorter delay than detouring via the CH. The intersection of two circles includes L-nodes with shorter delay than detouring through a CH in this cluster. When $d \leq R/2$ as shown in **Fig. 3** (a), all L-nodes in the circle,

with a radius of d, are included. When $d > R/2$ as shown in **Fig. 3** (b), L-nodes in the intersection of the circles with a radius R and d are included. Hence, the area, which contains the L-nodes with shorter delay, is computed as follows.

$$S(d)=\begin{cases}\pi d^2, & \text{if } d \le \frac{R}{2} \\ d^2\cos^{-1}\left(\frac{2d^2-R^2}{2d^2}\right)+R^2\cos^{-1}\left(\frac{R^2}{2dR}\right)-\frac{1}{2}\sqrt{R^2(2d-R)(2d+R)}, & \text{if } d > \frac{R}{2}\end{cases} \quad (7)$$

where the area S when $d > R/2$ is derived by solving the Circle-Circle Intersection problem [32].

Assuming that L-nodes are deployed randomly according to the Uniform distribution and the number of L-nodes in the cluster is N_C, the average number of L-nodes in the specific region with the area of A is

$$Avg.No.of\ L-nodes=\frac{Area\ of\ the\ region}{Area\ of\ the\ cluster}\times N_C=\frac{A}{\pi R^2}\cdot N_C$$

Hence, we must compute the mean of S in order to obtain the average number of L-nodes with shorter delay. Since S changes with the distance d between the CH and the L-node which ranges from 0 to R, we first define the cumulative distribution function (CDF) of d, which represents the probability that the L-node is within the distance d from the CH, as follows.

$$F_D(d)=P(D\le d)=\frac{Area\ of\ the\ circle\ with\ a\ radius\ d}{Area\ of\ the\ cluster}=\frac{\pi d^2}{\pi R^2}=\frac{d^2}{R^2},\quad where\ 0\le d\le R \quad (8)$$

We then obtain the probability distribution function (PDF) by taking a derivative of the CDF in Eq. 8.

$$f_D(d)=F_D'(d)=\frac{2d}{R^2},\quad where\ 0\le d\le R$$

With this PDF, the mean of area S is calculated as follows.

$$\begin{aligned}E[S]&=\int_0^R S(x)\cdot f_D(x)\,dx\\&=\int_0^{\frac{R}{2}} S_1(x)\cdot f_D(x)\,dx+\int_{\frac{R}{2}}^R S_2(x)\cdot f_D(x)\,dx\end{aligned}$$

where S_1 and S_2 denote the area when $d \le R/2$ and $d > R/2$ in Eq. 7, respectively. Finally, the average number of L-nodes with shorter delay to all L-nodes is obtained as follows using MATLAB.

$$Avg.No.of\ L-nodes\ with\ shorter\ delay=\frac{E[S]}{\pi R^2}\cdot N_C\approx 0.3N_C \quad (9)$$

This result shows that for each L-node, direct communications with at most 30 percent of nodes, on average, give shorter delay in heterogeneous WSNs. For the remaining 70 percent of L-nodes, traversing through the CH gives shorter delay than the direct communication. In summary, each L-node needs to establish a key with only 30 percent of L-nodes for shorter delay and does not need to set up a key with the remaining nodes since the communication through the CH, which has been already established, gives shorter delay than the direct communication.

6.2 Storage Overhead

In this subsection, the required storages for keys of each node are analyzed. In our scheme, every node has an individual unicast key with a BS, a public key, a private key, a BS's public key and a certificate, all of which are preloaded before deployment in common. Each L-node also establishes an intra-cluster unicast key with a CH, an intra-cluster broadcast authentication key and an intra-cluster broadcast encryption key during the key establishment process. If necessary, additional on-demand symmetric keys can be established. Note that the maximum number of on-demand symmetric keys is $0.3N_C$, where N_C is the number of L-nodes in the cluster, from Eq. 9 and generally the actual number of on-demand keys are much less than $0.3N_C$ since each L-node normally communicates with the CH due to the nature of information gathering of WSNs. On the other hand, each H-node establishes N_C intra-cluster unicast keys with L-nodes in its cluster, an inter-cluster broadcast encryption key, an intra-cluster broadcast encryption key, and a one-way key chain of length L for inter-cluster broadcast authentication. Hence, the required storages are written as follows.

$$S_L = 3size_{pub} + 4size_{sym} + size_{cert} + [0.3N_C \cdot size_{sym}] \quad (10)$$

$$S_H = 3size_{pub} + size_{cert} + (N_C + L + 3)size_{sym} \quad (11)$$

The term in brackets in Eq. 10 denotes the overhead caused by on-demand key establishment. In LEAP, each node has the following storage overhead.

$$S_{LEAP} = (3deg + 2 + L) \times size_{sym} \quad (12)$$

where *deg* is the number of neighbor nodes within the transmission range.

Table 2. Simulation parameters

Parameter	Value	Parameter	Value	Parameter	Value
N_C	10~100	$size_{pub}$	160 bits	Hash	5.9 μJ/Byte
R	100 m	$size_{cert}$	688 bits	MAC	5.9 μJ/Byte
r	25 m	ECDSA Verify	45.09 mJ	Encryption	1.62 μJ/Byte
L	100	ECDSA Sign	22.82 mJ	Decryption	2.49 μJ/Byte
$size_{sym}$	128 bits	ECDH	22.3 mJ		

With Eqs. 10-12 and the parameters in **Table 2**, the storage overhead according to N_C is computed and shown in **Fig. 4** and **Fig. 5**. In **Fig. 4**, the overhead of a L-node in COKES is compared to that of LEAP. Note that our performance analysis mainly focuses on the overhead of a L-node which has constrained resources when compared to H-nodes. The 'min', 'avg' and 'max' in parentheses of **Fig. 4** denote the amount of the actual number of on-demand keys, which means zero, a half of maximum number and the maximum number, respectively. As depicted in **Fig. 4**, COKES uses much smaller storage than LEAP irrespective of the number of on-demand keys by at least 88.3%, 77.5%, and 65.1%. This is because COKES considers the communication pattern of WSNs and uses more storage in the CH by adopting hierarchical architecture. Even though a CH uses more memories for storing keys than a L-node, **Fig. 5** shows that COKES is still much better than LEAP by at least 79.5%, 76.1%, and 63.8% in terms of total storage overhead which includes both a CH and all L-nodes in the cluster.

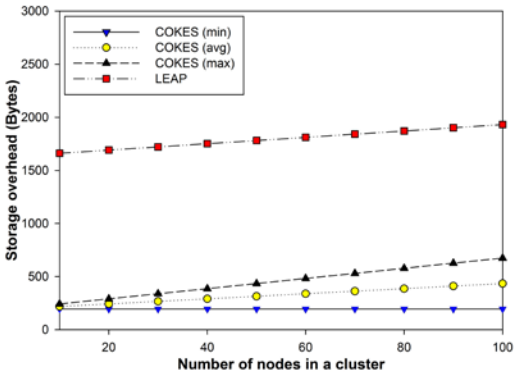


Fig. 4. Storage overhead of a L-node.

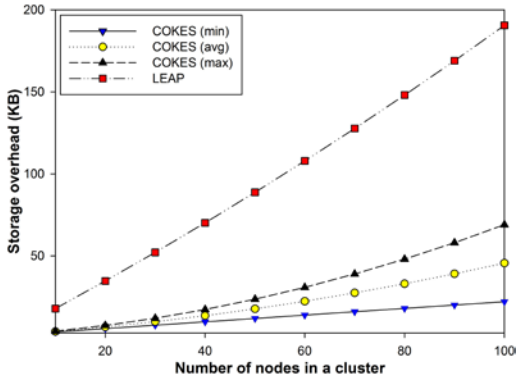


Fig. 5. Total storage overhead.

6.3 Communication Overhead

Communication overhead is closely related to the energy which is the most important resource in WSNs. To decrease the energy consumptions, communication overhead definitely has to be minimized. In this subsection, communication overhead is analyzed in terms of the number of messages exchanged during the key establishment process. Each L-node performs one RX and one TX while establishing an intra-cluster unicast key with a CH. Also, each L-node receives one message for an intra-cluster broadcast encryption key and an intra-cluster broadcast authentication key from a CH. During on-demand key establishment, a requesting L-node transmits one message and each L-node receives a message including a pairwise key. Each H-node performs one TX and N_C RXs for establishing N_C intra-cluster unicast keys with L-nodes in its cluster. In addition, each H-node sends N_C unicast messages containing broadcast keys to each L-node, respectively. During on-demand key establishment, a H-node receives a request and sends messages with a new key to two L-nodes. Therefore, the communication overhead of each node is as follows.

$$M_L = tx + 2rx + [0.3N_C(tx + 2rx)/2] \tag{13}$$

$$M_H = (1 + N_C)tx + N_C \cdot rx + [0.3N_C(2tx + rx)/2] \tag{14}$$

where the division by two in the bracket is from the fact that one key is established between two L-nodes during on-demand key establishment.

The communication overhead of LEAP is as follows.

$$M_{LEAP} = (2deg + 2)tx + (3deg + 1)rx \tag{15}$$

Fig. 6 shows the communication overhead of both a L-node in COKES and a node in LEAP according to the number of nodes in a cluster, using Eqs. 13-15 and the parameters in **Table 2**. Note that TX is counted as 2 since TXs consume twice as many energies as RXs [13]. Without on-demand keys, a L-node has extremely low communication overhead, which is only 1 TX and 2 RXs regardless of the number of nodes in the cluster. When the number of on-demand keys is beyond approximately 75%, LEAP shows better performance than COKES. However, this implies that if the number of on-demand keys is less than 75% of the maximum number, which is exactly what we insist, that is, the actual number of on-demand keys is very small in most applications, COKES is better than LEAP. This results from the fact that COKES establishes only essential keys based on the communication patterns and offloads tasks of L-nodes into H-nodes. Obviously, a H-node has much more communication overhead than LEAP, which is intended to reduce the communication overhead of L-nodes because H-nodes have plenty of resources. However, total communication overhead is also smaller

than LEAP as long as the number of on-demand keys is less than approximately 75% as shown in **Fig. 7**.

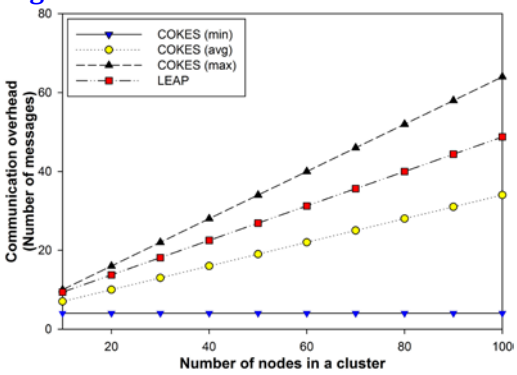


Fig. 6. Communication overhead of a L-node.

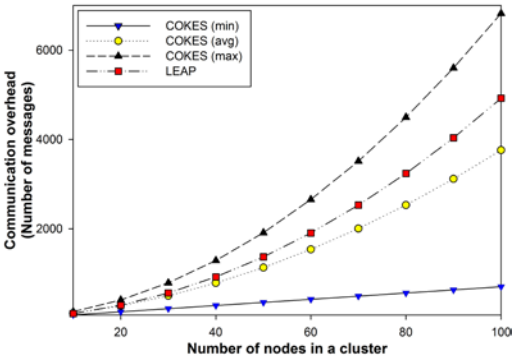


Fig. 7. Total communication overhead.

6.4 Computation Overhead

From our scheme, we can calculate computation overhead of each node which is written as

$$C_L = 2C^{ECDSA_{ver}} + C^{hash} + C^{ECDH} + 2C^{MAC} + C^{dec} + [0.3N_C \times (2C^{MAC} + C^{dec})]$$

$$C_H = C^{ECDSA_{sig}} + N_C \cdot C^{ECDSA_{ver}} + (1+L)C^{hash} + N_C \cdot C^{ECDH} + 2N_C \cdot C^{MAC} + N_C \cdot C^{enc} + [0.6N_C(2C^{MAC} + C^{enc})]$$

The computation overhead of LEAP is as follows.

$$C_{LEAP} = 2deg \cdot C^{MAC} + (deg+1)C^{enc} + (deg+1)C^{dec} + L \cdot C^{hash}$$

Each L-node requires two ECDSA verification operations used to verify a certificate and a digital signature and one ECDH operation, all of which are more expensive than SKC, in order to establish an intra-cluster unicast key with a CH at the beginning. After establishing all kinds of keys, all cryptography operations, including on-demand key establishment, are performed using SKC with the established keys. **Table 2** cited from [13] shows that the energy consumption of PKC is much larger than SKC although ECC is used as PKC. Comparison of computation overhead between COKES and LEAP is given in **Fig. 8** which shows that a L-node has much smaller computation overhead than LEAP by at least 52.1%, 49.3%, and 45.7% with no regard to the number of nodes in the cluster. This is because COKES creates much less keys than LEAP even though three PKC operations are used. In addition, COKES offloads PKC operations into H-nodes which have better computation capability. Definitely, a H-node has far more compuation overhead than LEAP. However, COKES still has less total computation overhead than LEAP by at least 20.4%, 20.0%, and 17.0% as shown in **Fig. 9**.

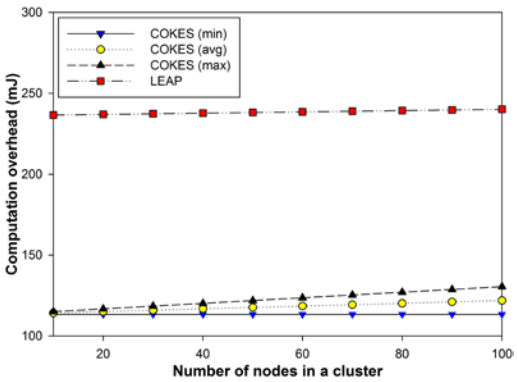


Fig. 8. Computation overhead of a L-node.

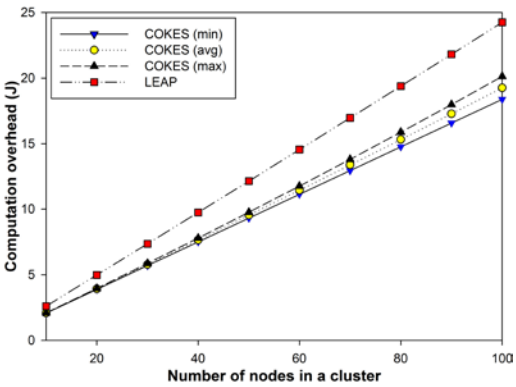
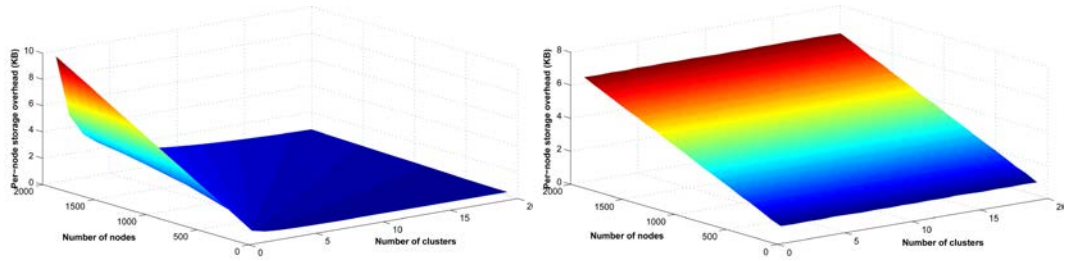


Fig. 9. Total computation overhead

6.5 Scalability

Scalability is the ability for WSNs to perform the given tasks normally as the number of nodes increases. This is very important in WSNs which usually consists of a very large number of sensor nodes. To be scalable, the number of keys, the number of exchanged messages and the number of computations performed by each node must be kept relatively small when the number of nodes increases in WSNs. In COKES, as the number of sensor nodes increases and, at the same time, the number of clusters increases, the overhead remains relatively small as shown in **Fig. 10** (a) which displays per-node storage overhead according to the number of nodes and the number of clusters. This is because our scheme uses hierarchical architecture. In contrast, per-node storage overhead of LEAP continuously increases as the number of nodes increases irrespective of the number of clusters as shown in **Fig. 10** (b). Furthermore, COKES has low per-node overhead even without considering a hierarchical architecture as shown in previous subsections. Therefore, we are sure that COKES is more scalable than LEAP. Note that we do not refer to per-node communication overhead and per-node computation overhead because they shows the same result as per-node storage overhead.



(a) COKES (b) LEAP

Fig. 10. Per-node storage overhead.

7. Conclusion

In this paper, we proposed COKES, a key establishment scheme for WSNs. COKES tries to minimize overhead required for establishing keys while meeting the security requirements by considering the unique communication patterns of WSN and taking advantage of heterogeneous WSNs. The only tradeoff against these benefits is the increased cost due to the use of H-nodes. However, we insist that it would be reasonable since our scheme needs much smaller number of H-nodes than L-nodes.

We are trying to apply COKES to border surveillance WSNs using real sensor nodes. This will enable COKES to be evaluated under specific applications and real environments. Finally, we expect COKES to be used as a cornerstone of security in a variety of applications in WSNs.

References

[1] I. F. Akyildiz, and W. Su, Weilian, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102-114, August, 2002. Article (CrossRef Link).

[2] X. Chen, K. Makki, K. Yen and N. Pissinou, "Sensor network security: a survey," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 2, pp. 52-73, 2009. Article (CrossRef Link).

[3] J. Zhang, and V. Varadharajan, "Wireless sensor network key management survey and taxonomy," *Journal of Network and Computer Applications*, vol. 33, no. 2, pp. 63-75, March, 2010. Article (CrossRef Link).

[4] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. of 9th ACM Conf. on Computer and Communications Security*, pp. 41-47, November 18-22, 2002. Article (CrossRef Link).
[5] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. of 2003 Symposium on Security and Privacy*, pp. 197–213, May 11-14, 2003. Article (CrossRef Link).
[6] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Transactions on Information and System Security*, vol. 8, no. 2, pp. 228–258, May, 2005. Article (CrossRef Link).
[7] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 4, pp. 500–528, November, 2006. Article (CrossRef Link).
[8] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting heterogeneity in sensor networks," in *Proc. of 24th Annual Joint Conf. of the IEEE Computer and Communications Societies*, pp. 878-890, March 13-17, 2005. Article (CrossRef Link).
[9] X. Du, Y. Xiao, M. Guizani, and H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 24-34, January, 2007. Article (CrossRef Link).
[10] Traynor, R. Kumar, H. Choi, G. Cao, S. Zhu, and T. La Porta, "Efficient hybrid security mechanisms for heterogeneous sensor networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, pp. 663-677, June, 2007. Article (CrossRef Link).
[11] G. J. Popek, and C. S. Kline, "Encryption and secure computer networks," *ACM Computing Surveys*, vol. 11, no. 4, pp. 331-356, December, 1979. Article (CrossRef Link).
[12] D. Malan, M. Welsh, and M. Smith, "A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography," in *Proc. of 1st Annual IEEE Communications Society Conf. on Sensor and Ad Hoc Communications and Networks*, pp. 71–80, October 4-7, 2004. Article (CrossRef Link).
[13] A. Wander, N. Gura, H. Eberle, V. Gupta, and S. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," in *Proc. of 3rd IEEE Int. Conf. on Pervasive Computing and Communications*, pp. 324-328, March 8-12, 2005. Article (CrossRef Link).
[14] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987. Article (CrossRef Link).
[15] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, February, 1978. Article (CrossRef Link).
[16] W. Abdallah, N. Boudriga, D. Kim, and S. An, "An efficient and scalable key management mechanism for wireless sensor networks," in *Proc. of 16th Int. Conf. on Advanced Communication Technology*, pp. 687-692, February 16-19, 2014. Article (CrossRef Link).
[17] X. Du, M. Guizani, Y. Xiao, and H. Chen, "A routing-driven Elliptic Curve Cryptography based key management scheme for Heterogeneous Sensor Networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1223-1229, March, 2009. Article (CrossRef Link).
[18] K. Lu, Y. Qian, M. Guizani, and H. Chen, "A framework for a distributed key management scheme in heterogeneous wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 2, pp. 639-647, February, 2008. Article (CrossRef Link).
[19] R. Azarderakhsh, A. Reyhani-Masoleh, and Z. Abid, "A key management scheme for cluster based wireless sensor networks," in *Proc. of IEEE/IFIP Int. Conf. on Embedded and Ubiquitous Computing*, pp. 222-227, December 17-20, 2008. Article (CrossRef Link).
[20] D. Kim, S. Kang, and S. An, "Energy efficient time synchronization for target tracking in heterogeneous sensor networks," in *Proc. of 8th Int. Conf. for Internet Technology and Secured Transactions*, pp. 238-243, December 9-12, 2013. Article (CrossRef Link).
[21] F. Simjee and P. H. Chou, "Everlast: long-life, supercapacitor-operated wireless sensor node," in *Proc. of 2006 Int. Symposium on Low Power Electronics and Design*, pp. 197-202, October 4-6, 2006. Article (CrossRef Link).

[22] X. Du, and F. Lin, "Maintaining differentiated coverage in heterogeneous sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2005, no. 4, pp. 565-572, September, 2005. Article (CrossRef Link).
[23] W. Leveque, *Elementary theory of numbers*, Dover, 1990.
[24] Certicom Research, "Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography," *Version 2.0*, May, 2009. Article (CrossRef Link).
[25] F. K. Cameron and D. G. Patrick, "Digital signature standard (DSS)," *U.S. Department of Commerce, FIPS PUB* 186-4, 2013.
[26] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521-534, September, 2002. Article (CrossRef Link).
[27] D. Liu, and P. Ning, "Multilevel μTESLA: Broadcast authentication for distributed sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 4, pp. 800-836, November, 2004. Article (CrossRef Link).
[28] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4554-4564, October, 2009. Article (CrossRef Link).
[29] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770-772, November, 1981. Article (CrossRef Link).
[30] H. Alzaid, D. Park, J. G. Nieto, C. Boyd, and E. Foo, "A forward and backward secure key management in wireless sensor networks for pcs/scada," *Sensor Systems and Software*, pp. 66-82, 2010. Article (CrossRef Link).
[31] C. E. Perkins, and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90-100, February 25-26, 1999. Article (CrossRef Link).
[32] E. W. Weisstein, "Circle-circle intersection" *MathWorld-A Wolfram Web Resource*, Available: http://mathworld.wolfram.com/Circle-CircleIntersection.html. Article (CrossRef Link).

Daehee Kim received the B.S. degree in Electronics Engineering from Yonsei University, Korea, in 2003 and M.S. degree in Electronic and Computer Engineering from Korea University, Korea, in 2006. Currently, he is working for Ph.D. degree on Electronic and Computer Engineering in Korea University, Korea. His research interests include wireless sensor networks, LTE, 5G, and security in wireless networks.

Dongwan Kim received the B.S. degree from Korea University, Seoul, Korea, in 2003 and M.S. degree from POSTECH, Pohang, Korea, in 2006 and Ph.D. degree in Electronic and Computer Engineering from Korea University, Seoul, Korea in 2015. He is working for Samsung electronics ltd., Suwon, Korea, from 2006 to now. His research interests are in wireless communication design, signal processing techniques applied to the next-generation wireless communications.

Sunshin An received the B.S. degree from Seoul National University, Korea in 1973, and the M.S. degree in Electrical Engineering from KAIST (Korea Advanced Institute of Science and Technology), Korea in 1975 and the Ph.D. degree in Electric and Information from ENSEEIHT, France in 1979. He joined the faculty of Korea University in 1982, where he is currently a Professor of Electronic and Computer Engineering. Prior to joining Korea University, Prof. An was Assistant Professor of Electronic Engineering in Ajou University, Suwon, Korea. He was with NIST (National Institute of Standards and Technology) in U.S.A., as a visiting scientist in 1991. His research interests include the distributed system, communication networks and protocols, information network, intelligent network, multimedia communication system, wireless sensor network and mobile RFID network.