Chapter 2
Secure Data Aggregation in Wireless Sensor Networks

Wireless sensor networks (WSNs) usually consists of a large number of sensors which have limited capability in terms of communication, computation and memory [1, 2]. These sensors are deployed in a remote and unsurveillant field and they autonomously form before they engage in a predefined sensing task. WSNs renders possible solutions to many problems in both civilian and military applications, including temperature monitoring, wildfire detection, animal tracking, and battlefield surveillance. Therefore, it is challenging to come up with efficient ways to collect desired data, given the sensors only have simple hardware and software resources. For instance, most of the sensors only have a short lifetime due to the non-rechargeable battery which is a bottleneck for designing WSNs protocols.

Therefore, in order to reduce the communication cost in transmitting data, data aggregation [2, 3] technique is adopted to achieve this goal. Data aggregation aims to combine data from different data collectors so that the total amount of data transmission is curtailed. An example of data aggregation scheme in WSNs is presented in Fig. 2.1. As depicted in the figure, two groups of sensors are structured in the from of cluster and tree before they start gleaning environmental information (e.g., the arrival of tanks, locomotion of battleship) in their targeted regions. When the base station queries the sensor network, each sensor will send its data along a multi-hop path containing a certain number of aggregate sensors (e.g., compute the sum, average) and relay sensors which lead to the base station, instead of uploading data to it directly. From the example, data aggregation reduces the overall data transmissions, therefore improving the bandwidth and service life of the network.

Data aggregation performed by the relay nodes will bring about more benefits with the increase of network scale. However, while data aggregation have the advantage of improving network bandwidth and energy utilization, it will in turn affect other metrics of the network, such as delay and security [3, 4]. First, data aggregation needs data from every sensor to obtain the accurate final aggregation result. If one or more data is not available, then the base station has to wait for the missing data to be uploaded which incurs network delay. Second, since security is a major concern for the most of WSNs applications, it is not possible for researchers and practitioners

© The Author(s) 2017

L. Zhu et al., *Secure and Privacy-Preserving Data Communication in Internet of Things*, SpringerBriefs in Signal Processing,
DOI 10.1007/978-981-10-3235-6_2

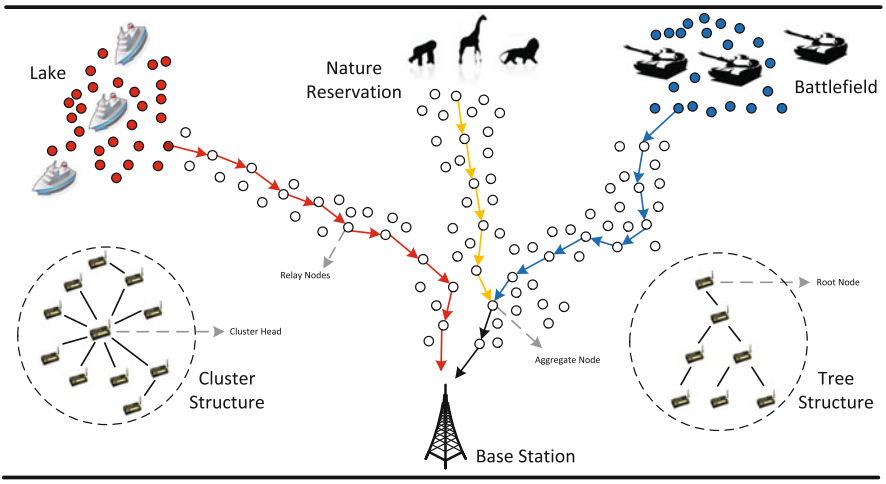


Fig. 2.1 Network model

to trade security for data aggregation. Moreover, there is a natural conflict between security and data aggregation. On one hand, security schemes require the sensors to encrypt the sensed data before transmission and then base station decrypt the received message to obtain plain data [5]. On the other, data aggregation schemes work efficiently with plain data to perform aggregation on aggregate sensors. Because of these conflictive goals in designing WSNs protocols, security and data aggregation must be designed in a collaborative way such that data aggregation can be implemented without compromising security.

The importance of achieving security and data aggregation simultaneously have led many researchers to focus on secure data aggregation problem. In this chapter, we aim to provide an extensive overview of secure data aggregation schemes in WSNs. Specifically, we will introduce the network model with main design goals and cover several eminent work in this area. Secure data aggregation problem has been studied over the past ten years and still has a potential to offer interesting research directions or opportunities which can be adopted in other research areas, such as participatory sensing [6], crowdsourcing [7] and mobile crowdsensing [8]. Meanwhile, we also aim to give a starting point for researchers interested in secure data aggregation by introducing the pertinent work and future research directions.

The remainder of this chapter is organized as follows: in Sect. 2.1, we present the problems, models and goals. Then, we introduce the cryptographic and non-cryptographic secure data aggregation schemes in Sects. 2.2 and 2.3, respectively. Last, we draw our summary.

2.1 Problems, Models and Goals

Before data aggregation, sensor nodes are classified according to their roles in the network and then the network structure is established. During data aggregation, sensor nodes will perceive the environment and transmit data though a path to the base station. In this section, we will briefly introduce the problems, network model, security model and security goals in wireless sensor networks.

Problems WSNs have unique characteristics and constraints which make the former schemes designed for traditional networks infeasible. Therefore, understanding the unique characteristics and constraints of WSNs is very helpful to design practicable schemes in WSNs. These problems come from two aspects: sensors and network.

From the viewpoint of sensors, different networks are formed from of different entities. Sensors consist the majority of WSNs. However, a sensor's capacity is limited in terms of memory, energy and CPU.

A sensor is a small device with a small amount of memory for code and parameters. To illustrate, the common sensor MICA2 has 4 K RAM, 128 K program memory and 512 K flash storage [9]. Given this constraint, the code size of a proposed scheme must be as small as possible.

The energy is the biggest problem in WSNs. Once the sensors are deployed in the filed, it is assumed that their batteries will not be replaced. Therefore, the operations required for sensors must be light-weight.

The CPU in MICA2 is the 16b, 8MHz microcontroller [9]. Generally, the CPUs in sensors are not that powerful and complex operations should be avoided. Since sensors communicate with each other wirelessly, their radio coverage area is also limited. For MICA2, it is 152 meters. Therefore, multi-hop routing is needed for sensors to send reports to base station which will further incur great transmission latency and difficult synchronization among sensors.

From the viewpoint of network, when sensors are deployed in the field, they always form a network autonomously. Several problems also arise.

First, the sensors are in a remote or even hostile environment, like battlefield, and they may be left unattended for a long time. Therefore, sensors are vulnerable to physical attacks from adversaries or natural activities. Second, the topology of the environment may not be clear to owner of the sensors and some sensors may be out of the network unexpectedly due to a bad location from paradrop, drained energy, or being damaged by animals.

Models Here we will introduce node classification, network structure, and different attacks on wireless sensor networks.

Node Classification. Typically, there are four types of sensor nodes in wireless sensor networks which are: (i) sensing nodes, (ii) aggregate nodes, (iii) relay node, and (iv) querier (e.g., base station).

The sensing nodes are the majority of which compose the WSNs and they usually contains sensing unit, processing unit, power unit and transceiver unit [10]. When a sensing task is initiated, sensing nodes sense data from environment and preprocess it

before sending to the next hop in the network. The preprocessing includes mapping, encryption, signature, etc.

The aggregate nodes are the intermediate nodes collecting preprocessed data from downstream sensor nodes, and apply a certain aggregation function onto those data. The aggregation function includes Max, Min, Sum, Average, etc. Then they send the processed data to the next hop along the transmission path until querier receives it.

The relay nodes are simply relay the data they receive to the next hop along the transmission path.

At last, the querier receives all the semi-processed data from aggregate/relay node (s) and perform aggregation function or other pertinent functions on it to derive useful information indicating the occurrence of some events which the base station is expecting to detect. These functions include mapping, decryption, signature verifying, etc.

Network Structure. Since these sensors are deployed in a remote and hostile environment, they have to form a network with a static structure in order to help intranet communication and data transmission. In order to complete data transmission from sensing nodes to base station, the network structure needs to be determined. Generally, the network structure is always divided into two categories: cluster [11–17] and tree [18–27].

A cluster is consisted of cluster members and one cluster head. Cluster members are the normal sensing nodes, and cluster head is elected to be the cluster manager which is responsible for local data aggregation. The role of cluster head can be either determined before they are deployed in the field or adaptively selected according to the network structure. Presumably, cluster head is considered to have stronger communication and computation capability than cluster members. As shown in Fig. 2.1, a cluster-structured network is on the left circled by an imaginary line and the center sensor node inside is the cluster head while the rest of sensor nodes are the cluster members.

A tree is consisted of leaf nodes, intermediate nodes and one root node. Leaf nodes are the normal sensing nodes, and each of them sends data to their own parent node which is the intermediate node. Data aggregation happens at intermediate nodes, performing the aggregation function on data collected from all child nodes and sends the semi-aggregated result upwards. In addition, the intermediate nodes can just relay what they receive can perform no aggregation functions. Root node controls the subtree rooted at itself and awaits for downstream data. As shown in Fig. 2.1, a tree-structured network is on the right circled by an imaginary line and the sensor node on the top is the root node of the tree while the rest of sensor nodes are the intermediate nodes and leaf nodes.

Cluster heads can send data to the querier directly by long range radio transmission. However, this is quite inefficient for sensor nodes with limited energy. In practice, cluster heads from different clusters usually form a tree with relay nodes to transmit their data by multi-hopping through each other, resulting in better energy efficiency. Both cluster structure and tree structure can enable better resource allocation and help improve energy control. The construction [12, 21] of a cluster or a tree is also an important phase prior to data collecting and data aggregation, however,

this chapter mainly introduces secure data aggregation, and the construction phase will not be paraphrased.

Attacks on WSNs WSNs are vulnerable to various types of attacks. The damage caused by them varies according to the adversary model and network structure. Generally, they are six typical attacks. Basically, the adversary's purpose is to cover his track, interfere with data aggregation process and ultimately mislead the base station to accept false aggregation result.

Node Compromise Attack. One of the severest attacks is node compromise attack [28], where the adversary is in full control of a sensor node (sensing node, aggregate node, or relay node), and can extract and access all the information stored on it, especially the cryptographic keys. This attack can be very harmful no matter what sensor node the adversary may corrupt and he can replay the old messages, selectively forward messages without being noticed.

Selective Forwarding Attack. After a sensing node or intermediate node is compromised, it is up to the adversary which controls them to decide whether or not forwards the sensed data or received messages. Once the adversary succeeded in stopping sensor nodes from sending messages, he can prevent the base station from recording or responding to the targeted event. For example, an adversary could stop a captured sensing node from sending a message indicating the movement of an enemy tank.

Replay Attack. Replay attack [28] is the easiest attack to be launched by an adversary because he does not have to compromise any sensor node or analyze the network traffic. He can records some transmitted data and replay them later on to mislead the aggregate node or base station that this message is sent from some sensor node. For example, an adversary could continuously send a pre-captured report that indicates a significant rise in temperature to trigger a false alarm of forrest fire.

Stealthy Attack. The adversary launches stealthy attack [28, 29] by injecting false data into the network, which will lead to a false final aggregation result. For example, an adversary could inject false trajectory data of illegal cargo ship to avoid being detected. Stealthy attack can also be launched by an adversary compromising a sensing node first.

Spoofed Data Attack. An adversary can also intercept the transmitted data and resend it to the network after an alternation on it. To successfully launch a spoofed data attack, the adversary has to ensure that the modified data will be accepted as valid by aggregate node and base station. For example, an adversary could change the increasing temperature reading caused by a forrest fire to a normal value and let the rampant fire spread. Spoofed data attack can also be launched by an adversary compromising a sensing node first.

Sybil Attack. This attack [28, 30] is where the adversary can present more than one sensor node and wield much power in the network. For example, an adversary can generate multiple sensor nodes to validate a counterfeit event in an witness-based aggregation scheme [31] and the final aggregation result will be affected if the majority of sensor readings are generated by one adversary launching Sybil attack.

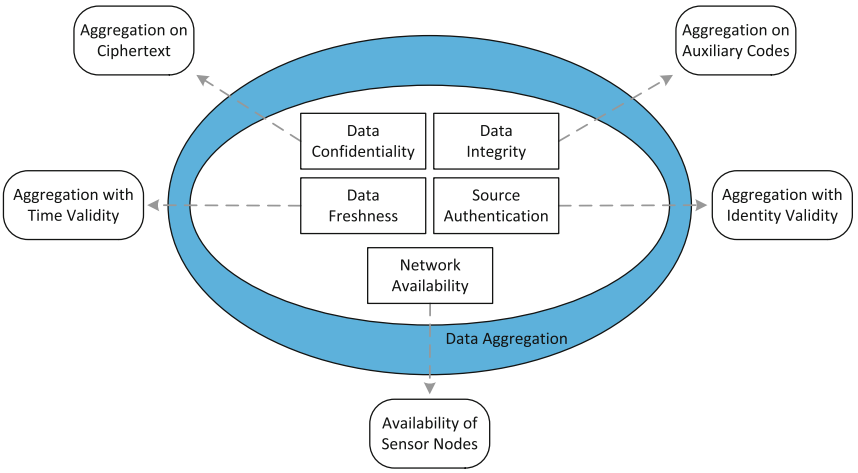


Fig. 2.2 Interaction between WSNs security requirements and data aggregation [4]

Denial-of-Service Attack. It [32, 33] is a standard attack which can be launched at any layer of WSNs by transmitting radio signals of high frequency that interfere with the network radio frequencies and cause network paralysis. DoS attacks can lead to excessive communication or computation and exhaust the battery of sensor nodes which can disable the network permanently. For example, since the availability of aggregate node is much more important than the other sensor nodes, adversaries launching DoS attacks can prevent aggregate nodes from sending data into the higher levels.

Goals Secure data aggregation schemes in wireless sensor networks ought to satisfy the security requirements. Due to the remote and hostile environment and unique characteristics of WSNs, it is challenging to protect sensitive information transmitted by sensor nodes [1]. In this subsection, we present the primary security requirements that are discussed in WSNs and explain how these requirements interweave with data aggregation. These security requirements are data confidentiality, data integrity, source authentication, data freshness, network availability [4]. Figure 2.2 shows the interaction between WSNs security requirements and data aggregation.

Data Confidentiality. In WSNs, data confidentiality is the most important security issue and it ensures that the content of sensed data is not leaked to other sensor nodes or adversaries. The other sensor nodes include neighbouring nodes, aggregate nodes, and relay nodes. The adversaries include internal adversaries and external adversaries which possess different capabilities.

Data confidentiality means the protection of sensitive information including secret keys transferred in the network and it is imperative to construct secure channels amid sensor nodes. Intuitively, the standard way to keep sensitive data secret is to encrypt data using a symmetric key or the receiver's public key. The formal approach one requires that the symmetric key should be established between the sender and

receiver. However, data aggregation functions cannot be performed on encrypted data directly. Therefore, the aggregate node must decrypt the ciphertext first and then aggregates all the plaintexts. Then it encrypts the aggregated data and transmits it to base station. The decryption/encryption of sensed data of aggregate nodes incurs extra computation, leading to not only network delay and energy consumption, but also prevention of data confidentiality from sensing node to base station.

Homomorphic encryption [34] can achieve the direct aggregation on ciphertext without the decryption/encryption, maintaining end-to-end data confidentiality.

Data Integrity. Although data confidentiality ensures that only authorized receivers obtain the plaintext data, it does not guarantee that the final result is not altered. Even if data confidentiality is provided, a compromised sensor node can still corrupt the aggregation and prevent the network from proper function by changing one bit. Moreover, data can still be altered because of the nature of wireless communication channel, even without the interference of adversaries.

Data integrity ensures that a message in transit is not altered. In some applications, data integrity weighs more than data confidentiality and it is sometimes acceptable for adversaries to learn about the intermediate aggregation result or final aggregation result, but not to change any of it. Two common methods of providing data integrity are message authentication codes (MAC) and cyclic codes.

Data Freshness. Providing data confidentiality and data integrity is not enough for secure data aggregation. A compromised sensor node is able to listen to transmitted messages and replay them in subsequent sensing task to disrupt the data aggregation result. Data freshness protects data aggregation against replay attacks by ensuring that the transmitted data is new and recent. In some applications, data freshness is a key requirement. For instance, an adversary can replay a normal sensed data with its pertinent information when a targeted event happens or mislead a normal sensing node by replaying an old distributed key without a proper time stamp.

Source Authentication. Since WSNs use a shared wireless communication medium, sensor nodes need authentication mechanisms to detect illegal messages. Source authentication allows a sensor node to verify whether or not a received message is sent by a legal and claimed sender. Without source authentication, an adversary is able to inject arbitrary data into the network. Moreover, he can impersonate an aggregate node can report false aggregation result.

Network Availability. Network availability is the network usability. First, secure data aggregation schemes must be designed with network availability to prevent excessive communication or computation which exhausts a sensor node's energy. Second, network availability can guarantee the survivability and continuum of network services against Denial-of-Service (DoS) attacks [32, 33].

Network availability is the foundation of enhancing other security aspects in WSNs. The consequence can be catastrophic without network availability. For example, if the availability loss of sensor nodes in critic battlefield could lead to an enemy invasion. Therefore, it is recommended that the network response to the existence of bas sensor nodes and exclude them as soon as possible.

2.2 Cryptographic-Based Data Aggregation Schemes

In this section, we present the cryptographic secure data aggregation schemes [29, 35–45]. These schemes are based on classic cryptography primitives. Though their security goals vary from each other.

Secure Information Aggregation (SIA) [29] is a scheme where the authors construct random sampling mechanisms and interactive proofs for the base station to verify the final aggregation result is a good approximation of the true value, even though the aggregate node and a certain number of sensing nodes are compromised. Their security goal is to prevent stealthy attacks and guarantee that if the base station accepts a final aggregation result, then this result is 'close' to the true aggregation value with high probability. This paper is the first paper on secure information aggregation in WSNs that can handle a malicious aggregate node and sensing nodes.

Specifically, each sensor node with a unique identifier shares a separate secret key with the base station and with the aggregate node. The keys are used to enable encryption and source authentication. Then, they proposed a new approach called aggregate-commit-prove and it consists of three parts: computation of the result, commit to the collected data and report the aggregation result, and prove the correctness of the result. First, aggregate node collects the data from sensors and locally computes the aggregation result, and it can verify the authenticity of each sensed data. Second, the aggregate node commits to the local aggregated data by computing $v_i = H(m_i)$ where m_i is the sensed data of sensing node i. One efficient way of committing to the aggregation data is Merkle hash-tree [46, 47]. In this hash-tree, all the sensed data is at the leave nodes, and the aggregate node then computes hash values of sensed data starting from the leaf nodes; each intermediate node computes the hash value of the concatenation of its two child nodes. The root of the hash-tree is called the commitment of the aggregation data. Third, the aggregate node reports the local aggregation result with the commitment to the base station and proves that the reported result is correct. Figure 2.3 depicts an example of Merkle hash tree construction. SIA provides data confidentiality, data integrity and source authentication.

Secure Data Aggregation and Verification (SecureDAV) [35] pointed out that bootstrapping keys was a challenge in WSNs and public key cryptosystems were unsuitable for use in resource-constrained sensor networks. They first presented a protocol for establishing cluster keys using verifiable secret sharing and they chose

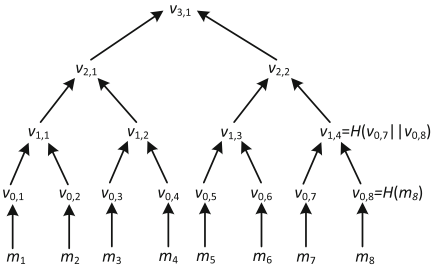


Fig. 2.3 An example of Merkle hash-tree construction [46, 47]

elliptic curve cryptosystems for encryption due to their fast computations and smaller key size. Besides, they proposed SecureDAV protocol that could ensure that the base station never accepted a false aggregation result.

Specifically, they developed a cluster key establishment protocol that all sensor nodes within a cluster shares a secret cluster key and each sensor node only has a share of the secret cluster key. The secret key is used to generate partial signatures (via ECDSA [48]). In their secure data aggregation and verification protocol, the sensing node i transmits the identifier, encrypted data c_i and the hash value h_i of the reading R_i to the cluster head CH_i:

$$c = Enc(k_i^{CH_i}, R_i), \ h_i = H(R_i)$$

where $k_i^{CH_i}$ is the secret key shared between sensor node i and cluster head CH_i. Then, cluster head CH_i aggregates all sensing data to $avg_i = avg_i + (R_j/|CH_i|)$ for each sensing node j in cluster i after decryption, and generates partial signatures and combine them into a whole signature. CH_i transmits the cluster identifier, encrypted message on avg_i and combined signature to base station. Last, base station verifies the signature using public key. SecureDAV provides data confidentiality, data integrity and source authentication.

Concealed Data Aggregation (CDA) [36] provided end-to-end encryption for reverse multicast traffic between the sensor nodes and base station. Aggregate nodes can perform aggregation functions on ciphertexts, which saving the time in costly decryption and encryption operations. They use the encryption transformation and decryption transformation of the privacy homomorphism (PH) [49]. The PH is probabilistic in the sense that the encryption transformation involves some randomness that chooses one ciphertext corresponding to a given plaintext from a set of possible ciphertexts.

Specifically, each sensing node shares a secret key $k = (r, g')$ with base station and it randomly splits its sensed data a into a secret sequence $a_1, \ldots, a_d$ such that $a = \sum_{j=1}^{d} a_j \bmod g'$ and $a_j \in \mathbb{Z}_{g'}$. Then it computes the encryption if a:

$$Enc_k(a) = c = (c_1, c_2, \ldots, c_d) = (a_1 r \bmod g, a_2 r^2 \bmod g, \ldots, a_d r^d \bmod g).$$

Aggregate node collects messages c from sensing nodes and computes the aggregation result S, and sent it to base station:

$$S = \left(\sum_{j=1}^{d} c_{j1} \bmod g', \sum_{j=1}^{d} c_{j2} \bmod g', \ldots, \sum_{j=1}^{d} c_{jd} \bmod g', \right)$$

Base station compute the plaintext y:

$$y = \frac{Dec_k(S)}{n} = \frac{(S_1 \times r^{-1}, \ldots, S_2 \times r^{-2}, \ldots, S_d \times r^{-d}) \bmod g'}{n}$$

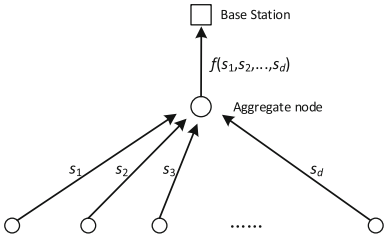


Fig. 2.4 Concealed data aggregation for WSNs with privacy homomorphism [36]

In addition, CDA can be applied to the problem of a concealed movement detection function. For example, there are five sensing nodes which are aware of their relative position, and we set the domain of sensed data into 0, 1. Then a message tuple (0, 0, 0, 1, 1) means that an entity has moved from position 1 to position 2, or vice versa. Figure 2.2 depicts the concealed data aggregation process.

The authors receded that the encryption, decryption and addition operations are more expensive when comparing the clock cycles that are necessary to perform Domingo-Ferrer's PH with those necessary to perform RC5. However, they argued that this disadvantage is acceptable since the aggregator node is the performance bottleneck for a connected WSN with the main objective to balance the energy consumption. At last, CDA only provides data confidentiality. Figure 2.4 shows an example of data aggregation phase in CDA.

Chan et al. [37] focuses on the problem of securely and efficiently performing aggregation queries (such as MEDIAN, SUM and AVERAGE), and presents the first algorithm for provably Secure Hierarchical In-Network Aggregation (SHIA) for general sensor networks and multiple adversarial nodes. This algorithm can guarantee that the adversary cannot have any advantage from manipulating intermediate aggregation aggregations. It supports arbitrary tree-based structure, and retains resistance against aggregation manipulation in the presence of arbitrary numbers of malicious sensor nodes.

The main algorithm is based on performing the SUM aggregation securely and the goal is to compute $a_1 + \cdots + a_n$, where a_i is the data value of sensing node i. Generally, they bulit on the aggregate-commit-prove framework from [29] but extended their single aggregate node model to a fully distributed setting. The algorithm includes computing a commitment structure on the data values as well as the aggregation process. Then the sensor nodes verifies that their contributions are added to the aggregation result by auditing the commitment structure. If the adversary tries to exclude the contribution of one sensing node, this will induce an inconsistency in the commitment result which can be detected by the victim node.

Specifically, the algorithm mainly contains three phases: query dissemination phase, aggregation-commit phase, and result-checking phase.

Query dissemination. The base station broadcasts the query to the all the sensor nodes. An aggregation tree with the base station being the root node, is formed if one is not already present. Tiny Aggregation Service (TAG) [23] is one of various algorithms for selecting the structure of an aggregation tree and each node chooses

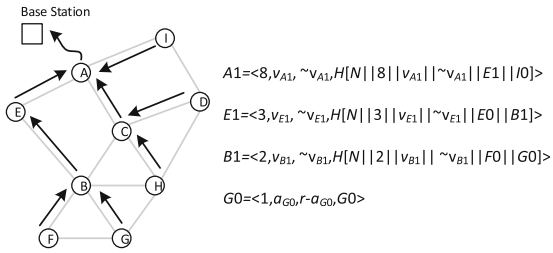


Fig. 2.5 An example of commitment tree in SHIA [37]

the node from which it first heard the tree-formation message as its parent node. After the tree is established, the base station initiates a query request which includes a nonce N to prevent replay attack, and the entire request message is authenticated.

Aggregation commit. In the naive approach, each sensing node sends the number 1, sensed data, the complement of sensed data and ID to its parent node. Each aggregate node performs computes a cryptographic hash of all its inputs (including its own data value if it has any). The hash value is then passed on to the parent node in the aggregation tree along with the intermediate aggregation result. Figure 2.5 depicts a commitment tree consisting of hashes of data values and intermediate aggregation results.

Definition 1 A commitment tree is a tree where each sensing node has an associated label representing the data sent to its parent node. The label has the following format:

$$< count, value, complement, commitment >$$

where count is the number of sensing/contributing nodes in the subtree rooted at this node; value is the SUM computed over all the sensing nodes in the subtree; complement is the aggregate over the COMPLEMENT of the data values; and commitment is the hash value on the concatenation of N, count, value, complement, and labels from the sensing node in one-hop range.

Result-checking. The purpose of this phase is to enable each sensor node i to independently verify that its contribution a_i was added into SUM aggregation result, and the complement $r - a_i$ was also added into the COMPLEMENT aggregation result.

First, the aggregation results are authenticated and broadcast to every sensor node. Each sensor node individually verifies that its data was counted. If so, it sends an authentication code to the base station. When the base station has received all the authentication codes, it then concedes that all sensor nodes' contribution to the aggregation result has been correctly added.

It is worth noting that they gave a definition of ***optimally secure*** and they have proved that SHIA can provide the strongest security bound which can be proven for any secure aggregation scheme without making any assumption about the distribution of sensed data. SHIA only provides data integrity.

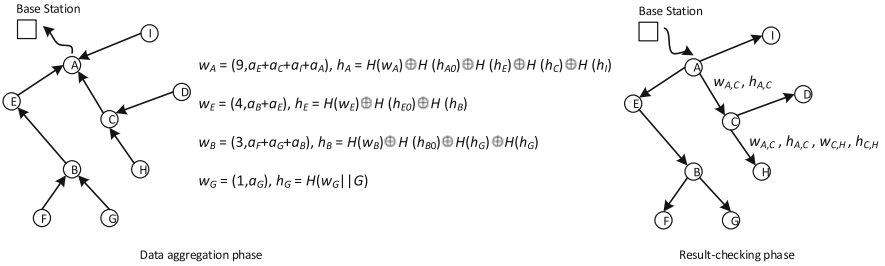


Fig. 2.6 An example of data aggregation phase and result-checking phase in E2IPAP [38]

Definition 2 An aggregation algorithm ***optimally secure*** if, by tampering with the aggregation process, an adversary is unable to induce the base station to accept any aggregation result which is not already achievable by direct data injection.

Zhu et al. [38] pointed out that the result-checking phase in [37] suffers from high communication overhead. Particularly, [37] needs $O(h\Delta)$ congestion for each sensor node to verify the aggregation result, and incurs $O(\log_2 n)$ edge congestion and $O(\Delta \log_2 n)$ node congestion (where Δ is the maximum node degree), when the base station disseminates the off-path values down the commitment tree. According to [38], it is important to optimize the result-checking phase and save network energy because the energy consumption is ascribed to the off-path dissemination step. They proposed an integrity preserving protocol Energy Efficient and Integrity- Preserving Aggregation Protocol (E2IPAP) for tree-based sensor networks which can reduce the number of messages per sensor node to $O(\Delta)$, and the edge congestion to $O(\log_2 n)$.

Specifically, in the aggregation-commit phase, each sensor node i sends the data-commitment tuple $< w_i, H_i >$ to its parent node, where w_i contains count, sensed data and the complement data. The aggregate node performs aggregation function, and sends the intermediate result upwards along the tree.

In the result-checking phase, base station broadcasts final aggregation result and final commitment to the network. After checking the validity, each sensor node i sends an authentication code $MAC(K_i, N||OK)$ to base station. Otherwise, it sends $MAC(K_i, N||NK)$ to refuse to accept the aggregation result. E2IPAP provides only data integrity.

Figure 2.6 shows an example of data aggregation phase and result-checking phase in E2IPAP. The node B, E and A all have their own sensed data, therefore, the final aggregation count is 9.

Node compromise attack requires protocols for data aggregation where the intermediate nodes contribute their own data values to the intermediate aggregation result without accessing it. Homomorphic encryption allows aggregation on ciphertext and can provide end-to-end data confidentiality. n-LDA [39] is a protocol for secure data aggregation which offers end-to-end data confidentiality by using homomorphic function and interleaved encryption. Rodhe et al. [39] proposed a layered data aggregation protocol which ensures that an attacker cannot get access to any aggre-

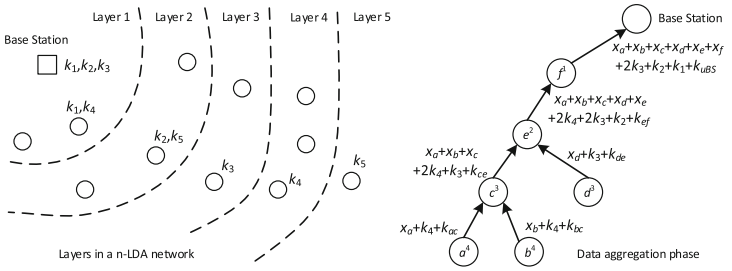


Fig. 2.7 An example of Layers and the aggregation phase in n-LDA [39]

gation result from the network if less than n sensor nodes are captured. When more than n sensor nodes are captured, the attacker can only get access to the aggregation result received by the captured nodes.

Specifically, the sensor nodes are divided into layers according to their hop distance from the base station after a wave algorithm is launched to determine the layers in the network. The nodes in layer i share a common aggregation key k_i which is also known by the base station for $i \leq n$ and by all nodes in layer $i - n$ for $i > n$.

During aggregation process, each node u_i receives encrypted data from nodes in the lower layer and aggregates them with its own encrypted data before choosing a neighbor w_{i-1} from the upper layer and sending the aggregated data. u_i's data is encrypted with k_i and the pairwise key k_{uw} shared with w_{i-1}. The aggregation key will be removed by the corresponding node in layer $i - n$, which provides privacy from layers $i - 1$ to $i - n + 1$. The pairwise key k_{uw} is removed by node w_{i-1}. Finally, base station removes the last keys and obtains the aggregation result. n-LDA provides only data confidentiality. Figure 2.7 depicts a sensor network with five layers in n-LDA.

n-LDA is resilient to node compromise or node failure because the aggregation result is not relayed along a fixed path, instead it can be dynamically chosen. This makes it hard for an adversary to choose which nodes to compromise to achieve his goal. Moreover, the additional information sent with aggregation result reveals nothing about the identity of the contributing sensor nodes.

EPAS [40] proposed an efficient and provably secure encryption scheme on allowing additive aggregation of the encrypted data to minimize the number of messages sent and received by sensor nodes, and maximize network lifetime. The aggregation based on this scheme can be used to efficiently compute statistical data, such as MEAN, and VARIANCE, while curtailing bandwidth savings.

Specifically, they adopted an additively homomorphic encryption scheme using a PRF family and the steps are described as follows.

1. Key Generation:
   (a) A random K is picked from $\{0, 1\}^{\lambda}$ which is set as the decryption key of base station.
   (b) $ek_i = f_K(i)$ for each sensor node i is set as the encryption key.

2. Encryption:

   (a) Given plaintext m_i, nonce r and encryption key ek_i, output:

   $$c_i = Enc_{ek_i}(m_i) = m_i + h(f_{ek_i}(r)) \bmod M.$$

   (b) Set $hdr_i = i$.
   (c) Output ciphertext (hdr_i, c_i).

3. Decryption:

   (a) Given ciphertext (hdr, c) and r, compute $ek_i = f_K(i)$ for $i \in hdr$.
   (b) Compute plaintext:

   $$x_i = Dec_K(c) = \left(c - \sum_{i \in hdr} h(f_{ek_i}(r)) \right) \bmod M.$$

4. Addition of Ciphertexts

   (a) Given two ciphertexts (hdr_1, c_1) and (hdr_2, c_2). compute

   $$c_{1-2} = (c_1 + c_2) \bmod M.$$

   (b) Set $hdr_{1-2} = hdr_1 \bigcup hdr_2$.
   (c) Output aggregation ciphertext (hdr_{1-2}, c_{1-2}).

The standard security goal for encryption is indistinguishability against chosen-ciphertext attack. The security of EPSA is based on the indistinguishability of a pseudorandom function (PRF). They have proved that the scheme is semantically secure against a collusion attack with at most $n - 1$ compromised sensor nodes for a network with n sensor nodes. EPSA provides data confidentiality and data integrity.

Chen et al. [41] pointed out that while these data aggregation schemes based on privacy homomorphism encryption could provide better security and reduce transmission overhead, however, the base station only obtained the aggregation result instead of the individual values from sensing nodes. Two consequences are (1) the usage of aggregation functions is constrained and (2) the base station cannot confirm data integrity and source authentication. Therefore, they proposed the concept of "recoverable" and the base station can recover all sensing data in their design Recoverable Concealed Data Aggregation (RCDA).

Mykletun et al.'s [50] and Boneh et al.'s schemes [51] are reviewed since they are the foundation of the proposed schemes. Specifically, RCDA scheme for homogenous WSN is composed of four phases: Setup, Encrypt-Sign, Aggregate, and Verify. The detailed phases are described as follows.

1. Setup: base station generates the pairs:

   (a) (P_{BS}, R_{BS}): These two keys are generated as in the KenGen function in [50]. $P_{BS} = \{Y, E, p, G, n\}$ and $R_{BS} = \zeta$.

(b) (P_i, R_i): For each sensor node i, the base station generates them by KeyGen function in [51]. $P_i = v_i$ and $R_i = x_i$.

Then, P_i, R_i and H are loaded to sensor node i.

2. Encrypt-Sign: When a sensing node i is ready to send its sensed data to the cluster head (CH_i), it

(a) Encodes sensed data d_i as $m_i = d_i||0^{\beta}$, where $\beta = l \cdot (i-1)$.
(b) Computes signature $\sigma_i = x_i \times H(d_i)$ and $c_i = (r_i, s_i) = (k_i \times G, m_i \times G + k_i \times Y)$, where $k_i \in \{0, \ldots, n-1\}$.
(c) Sends (c_i, σ_i) to CH_i.

After CH_i collects all ciphertext-signature pairs, it can aggregate them as follows:

3. Aggregate ciphertext:

(a) Computes

$$\widehat{c} = (\widehat{r}, \widehat{s}) = \sum_{i=1}^{\eta-1} c_i = \left(\sum_{i=1}^{\eta-1} r_i, \sum_{i=1}^{\eta-1} s_i \right).$$

(b) Computes signature:

$$\widehat{\sigma} = \sum_{i=1}^{\eta-1} \sigma_i.$$

(c) Sends the aggregation result $(\widehat{c}, \widehat{\sigma})$ to base station.

After receiving $(\widehat{c}, \widehat{\sigma})$ from CH_i, base station can recover and verify each sensed data as follows:

4. Verify:

(a) Computes $M' = -t \times \widehat{r} + \widehat{s} = M_1 + \cdots + M_{w-1}$.
(b) Computes $m' = rmap(M') = m_1 + \cdots + m_{w-1}$.
(c) Obtains each sensed data from m': $d_i = m'[(i-1) \cdot l, i \cdot l - 1]$.
(d) Verifies each d_i by checking whether $e_n(\widehat{\sigma}_1, g_2) = \prod_{i=1}^{\eta-1}(h_i, P_i)$ where e_n is bilinear map and $h_i = H(d_i)$.

Base station can recover all sensed data from other clusters and perform any statistical operations after confirming the integrity of all data. RCDA provides data confidentiality and data integrity.

Yang et al. [42] focused on preserving data integrity and proposed an Efficient Integrity-Preserving Data Aggregation Protocol (EIPDAP) where the base station can verify the integrity of aggregation result. Compared with pertinent schemes on preserving data integrity, EIPDAP reduces the communication overhead per node to $O(\Delta)$, where Δ is the degree of the aggregation tree.

The design of EIPDAP is based on the elliptic curve discrete logarithm problem and their algorithm consists of three main phases: query dissemination, aggregation-commit, and result-checking.

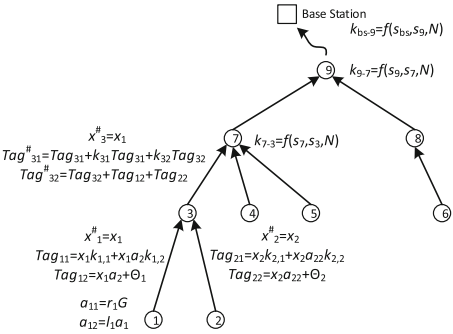


Fig. 2.8 An example of aggregation phase of EIPDAP [42]

Specifically, the base station first broadcasts the query to the network and sends the path-keys and edge-keys encrypted with the secret key shared between itself and nodes to the corresponding nodes. In aggregation-commit phase, each sensor node senses environmental data and calculates two tags before reporting them to parent node. After receiving all the data downstream, the aggregate node performs modulo addition operations and forwards the result to bast station. In the end, base station verifies the integrity of the aggregation result with two aggregation tags. EIPDAP provides data confidentiality and data integrity.

For example, base station calculates the edge-key k_{i-j} for each sensor node using the hash function H:

$$k_{i-j} = H(s_i, s_j, N)$$

where node i is the parent node of node j and s_i is the unique identity of node i. Base station also calculates two path-keys $k_{i,1}$ and $k_{i,2}$:

$$k_{i,1} = \frac{\theta}{k_{path}}, k_{i,2} = \frac{l}{k_{path}},$$

where θ is a point in $E(Z_p)$ and l is an integer. If the path from base station to node i is $bs - 1 - 2 - i$, then $k_{path} = k_{bs-1}k_{1-2}k_{2-i}$. Figure 2.8 shows an example of data aggregation phase of EIPDAP.

Zhu et al. [43] has pointed out that many existing aggregation protocols had to transfer the whole list of sensor nodes ID for base station to know which sensor nodes had contributed sensed data. However, the transmission of these IDs can incur overwhelming communication overhead. Therefore, they proposed a provably secure aggregation scheme perturbation-based efficient confidentiality preserving protocol (PEC2P). The general idea of PEC2P is that after the base station receives the aggregation result and the hash tag, it will go through a brute search on the possible combination of sensor nodes which can form the hash tag. PEC2P can avoid transferring any ID information and allows efficient aggregation of perturbed data.

PEC2P mainly consists of three phases: bootstrapping phase, data aggregation phase, and result retrieving phase. Specifically, in the bootstrapping phase, modulus

M, a collision-resistant cryptographic hash function $H : \{0, 1\} \leftarrow \{0, 1\}$ and a PRF $f : \{0, 1\} \leftarrow \{0, 1\}$ are stored in all sensor nodes. In addition, a unique vector IV_i an a secret key $k_i = IV_i$ are stored in base station and node i.

Each sensor node with combined functions can be logically split into a sensing node and an aggregate node. In data aggregation phase, each leaf sensor node i with environmental data x_i computes it aggregation result $< c_i, hax_i >$ as:

$$c_i = 1,\ hax_i = x_i + H(k_i).$$

Then, it forwards the result to its parent node for data aggregation. At last, i updates its secret key as $k_i = f(k_i)$. Each aggregate node j starts a timer and waits for messages before it expires. Then, it computes partial aggregation result $< c_j, hax_j >$ as:

$$c_j = \sum_{i \in S_j} c_i,\ hax_j = \sum_{i \in S_j} hax_i \bmod M$$

Then, j forwards the aggregation result to base station. Note that the count is used to trace the contributing nodes for base station and synchronization among sensor nodes is not needed. In the result retrieving phase, base station retrieves the contributing ID list and the aggregation result. First, it orderly selects a list IDL of C sensor nodes with their keys and computes:

$$Agg = \left(HAX_{bs} - \sum_{i \in IDL} H(k_i) \right) \bmod M.$$

If $Agg \in [C * vmin, C * v_{max}]$, base station will accept Agg as the aggregation result and update the keys for the C nodes, or it will continue to search for the sensor set.

PEC2P provides data confidentiality and data integrity. Figure 2.9 depicts an example of data aggregation in PEC2P.

Aggregating data while preserving data confidentiality and data integrity is a challenging task for adversaries can eavesdrop on the network and modify the aggregation results readily by launching node compromise attack. Zhu et al. [44] proposed

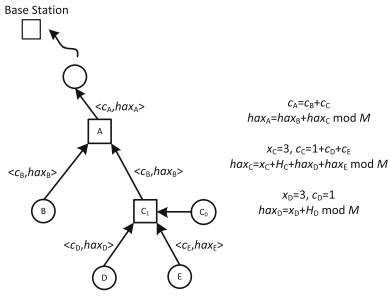


Fig. 2.9 An example of aggregation phase of PEC2P [43]

an efficient confidentiality and integrity preserving aggregation protocol (ECIPAP) based on homomorphic encryption and result-checking mechanism.

Specifically, each sensor node i shares a secret key k_i, a large integer M, and a unique identifier ID_i with the base station. In each data collection task, it broadcasts a query message to the network along with a random number r. Each sensor node i has ensed data $v_i \in [v_{min}, v_{max}]$, $v_i = v_{max} - v$, and $count_i = 1$. Then it computes three temporary keys $k_{i,r}$, $k'_{i,r}$ and $k''_{i,r}$:

$$k_{i,r} = H(k_i, r) \bmod M,\ k'_{i,r} = H(k_i, r+1) \bmod M,\ k''_{i,r} = H(k_i, r+2) \bmod M,$$

and then it encrypts v_i, v_i and $count_i$ as:

$$c_{count_i,i} = Enc(k_{i,r}, count_i, M),\ c_{v_i,i} = Enc(k'_{i,r}, v_i, M),\ c_{\sim v_i,i} = Enc(k''_{i,r}, \sim v_i, M).$$

i computes message authentication code as:

$$MAC_i = H(c_{count_i,i} || c_{v_i,i} || c_{\sim v_i,i} || ID_i),$$

and the data tuple u_i is:

$$u_i = < c_{count_i,i}, c_{v_i,i}, c_{\sim v_i,i}, MAC_i > .$$

When aggregate node receives data tuples downstream, it can perform aggregation functions like SUM, COUNT, AVERAGE, on them. As shown in Fig. 2.10, base station can decrypt the aggregation result as:

$$count_{agg} = Dec\left(U_a \cdot count, \sum_{i=1}^{n} k_{i,r}, M\right),$$

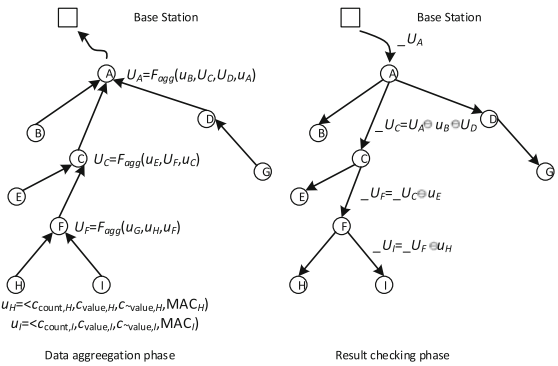


Fig. 2.10 An example of aggregation phase of ECIPAP [44]

$$v_{agg} = Dec\left(U_a \cdot value, \sum_{i=1}^{n} k'_{i,r}, M\right),$$

$$^{\sim}v_{agg} = Dec\left(U_a \cdot^{\sim} value, \sum_{i=1}^{n} k''_{i,r}, M\right)$$

Finally, base station can calculate MAC_{agg}:

$$MAC_{agg} = \bigoplus_{i=1}^{n} m_i \cdot MAC.$$

When the aggregation result reached at base station, base station broadcasts the authenticated aggregated data tuple. To complete result checking, each sensor node will send a checking message using reverse computational operation $\ominus$ to its child nodes. Every sensor node can verify if its own data was added to the aggregation process by comparing its own data to the data sent by parent nodes. If so, node i sends an authentication message $MAC(k_i||r||OK)$ upwards. Otherwise, $MAC(k_i||r||NO)$ will be sent. ECIPAP provides data integrity and source authentication.

Sen-SDA [45] is a practical and secure data aggregation scheme in heterogeneous clustered WSNs, based on the combination of an additive homomorphic encryption scheme, an identity-based signature scheme, and a batch verification technique with an algorithm for filtering injected false data.

Specifically, Sen-SDA five phases: Setup, Private key extraction, Encrypt then sign, Verify, Aggregate then Sign and Verify then Decrypt.

1. Setup: Base station generates $para_1 =< \mathbb{F}_p, \mathrm{E}, \mathrm{q}, \mathrm{P}, P_{Pub}, H_1, H_2 >$ and $para_2 = < \mathbb{F}_p, E, q, P, P_E >$, then it sets the public parameters $para = para_1 \bigcup para_2$. Base station keeps a master secret key s and x.
2. Private key extraction: base station picks a new identifier ID_i for sensor node i and generates $sk_i = (R_i, v_i)$. After performing the clustering algorithm, each sensor node has its own cluster head, Then CH_i keeps a list of member nodes in the ith cluster LCM_i.
3. Encrypt then sign: a sensor node CM_j^i encrypts its message m_j^i with base station's public key and obtains ciphertext:

$$c_j^i =< k_j P, k_j P_{bs} + m_j^i P > .$$

Then i computes a signature with private key SK_j and current timestamp ct_j^i:

$$\sigma_j^i =< R_j, T_j^i, z_j^i >,$$

where $T_j^i = t_j P, z_j^i = v_j + h'_j t_j \bmod q,\ h'_j = H_2(ID_j, ID_i, R_j, T_j^i, ct_j^i)$. Then node i sends $M_j^i =< c_j^i, \sigma_j^i, ID_j, ID_i, ct_j^i >$ to cluster head.

4. Verify, Aggregate then Sign: After receiving messages from all member nodes, CH_i looks up the list to find the corresponding node ID_j. If it exists and $\{ct_j^i\}_{j=1}^{n_i}$ are valid, CH_i verifies $\{\sigma_j^i\}_{j=1}^{n_i}$. If they are valid, CH_i computes $c_i = \sum_{j=1}^{n_i} c_j^i$. Then CH_i generates a signature:

$$< \tau_i >=< R_i, T_i, z_i >$$

where $z_i = v_i + h_i' t_i \bmod q$. Finally, CH_i sends $M_i =< c_i, \tau_i, ID_i, ID_{bs}, ct^i >$ to base station.

5. Verify then Decrypt: After receiving messages from all cluster heads, base station looks up the list to find the corresponding cluster ID_i. If it exists and $\{ct^i\}_{i=1}^{n_C}$ are valid, base station verifies $\{\tau_i\}_{i=1}^{n_C}$. If they are valid, base station recovers $M_i = (\sum_{j=1}^{n_i})P$ and decrypts $c_i = (c_{i1}, c_{i2})$ as:

$$c_{i2} - x \cdot c_{i2} = \sum_{j=1}^{n_i}(k_j P_{bs} + m_j^i P) - x\left(\sum_{j=1}^{n_i} k_j P\right) = \left(\sum_{j=1}^{n_i} m_j^i\right) P.$$

Table 2.1 Comparison of secure data aggregation protocols

Protocol	Data confidentiality	Data integrity	Source authentication	Network availability
SIA	√	√	√	
SecureDAV	√	√	√	
CDA	√			
SHIA		√		
E2IPAP		√		
n-LDA	√			
EPSA	√	√		
RCDA	√	√		
EIPDAP	√	√		
PEC2P	√	√	√	
ECIPAP		√	√	
Sen-SDA	√	√	√	
Witness		√		
SRDA	√	√	√	
CUSUM-GLR		√	√	
IPERP	√	√	√	√
SDA		√	√	
SDA2014		√	√	

Sen-SDA provide data confidentiality, data integrity and source authentication. Table 2.1 presents the comparison of cryptographic and non-cryptographic secure data aggregation schemes with respect to different security requirements.

2.3 Non Cryptographic-Based Data Aggregation Schemes

In this section, we present the non-cryptographic secure data aggregation schemes [31, 52–57].

A witness-based data aggregation scheme for WSNs is proposed in [31] to solve the assurance problem and assure the validation of the data sent from aggregate nodes to the base station. Some sensor nodes around the aggregate node are selected as witnesses to monitor the data aggregation process.

They proposed to use the concept of witness node. The aggregate node has to provide proofs from several witness nodes, in order to prove the validity of the aggregation result. A witness node also performs data aggregation, but does not send the aggregation result to the base station. Each witness node computes the MAC (proof) of the result, and then sends it to the aggregate node, who must relay these proofs to the base station. If the aggregate node is compromised, and the adversary tries to send a false aggregation result to the base station, it has to forge the proofs on the false one.

Specifically, they used a n out of $m + 1$ voting scheme to determine the validation of the aggregation result. The aggregate node and witness nodes share a secret key with the base station. Assume that we have chosen m witnesses, $w_1, m + 2, \ldots, w_m$, and $k_1, k_2, \ldots, k_m$ are the MAC keys they share with the base station. After receiving the data from the sensor nodes, each witness i compute the aggregation result agg_i and sends $MAC_i = MAC(agg_i, w_i, k_i)$ to the aggregate node. The base station will accept the aggregation result if at least $n - 1$ out of the m witnesses agree with the aggregation result; otherwise it discards the result.

One special case is that n is equal to $m + 1$. After receiving (MAC_1, MAC_2, ..., MAC_m) from the witness nodes, the aggregate node computes

$$MAC_{Agg} = MAC(Agg, k_{Agg}, MAC_1 \bigotimes MAC_2 \bigotimes \ldots \bigotimes MAC_m)$$

where Agg is the aggregation result and k_{Agg} is the MAC key shared by the base station and the aggregate node. After receiving $(Agg, w_1, w_2, \ldots, w_m, MAC_{Agg})$, the base station computes

$$MAC_i = MAC(Agg, w_i, k_i),$$

$$MAC'_{Agg} = MAC(Agg, k_{Agg}, MAC_1 \bigotimes MAC_2 \bigotimes \ldots \bigotimes MAC_m).$$

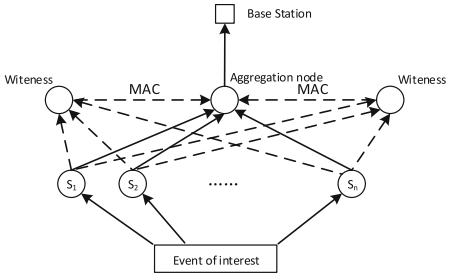


Fig. 2.11 A witness-based approach in wireless sensor networks [31]

If $MAC_{Agg} = MAC'_{Agg}$, then base station deems the aggregation result valid. To make the above scheme robust against the DoS attack, aggregate node should not merge all the MACs, but to send them to the base station. If at least n out of $m + 1$ MACs are matched, the aggregation result is accepted; otherwise, it is discarded. This witness scheme only provides data integrity. Figure 2.11 depicts the witness-based approach in wireless sensor networks.

SRDA [52] is proposed for in cluster-based WSNs to reduce the number of bits in transmission since radio communication is the most energy-consuming behavior for a sensor node. It is done through letting sensor nodes send only the difference (difference data) between raw sensed data value with their reference data value. The reference data is the average value of a series of previous sensor readings.

Specifically, (1) a key distribution protocol is implemented which results in secure peer-to-peer communication links between sensor node pairs. For each session: (2) each sensor node computes its reference value by taking the average of last N sensed data values; (3) sensor node sends the first data packet with value R_{i1} in this session. R_{i1} is taken as the reference value of sensor node i. All packets in the session are encrypted as follows:

(a) Ascertains h, the number of hops to the base station.
(b) Calculates the security margin:

$$S = \left(\frac{1}{h}\right) * 100.$$

(c) Increases the minimum number of rounds of RC6 [58]
(d) Encrypts the data using RC6.

(4) Cluster-head creates a reference entry for each sensor node; (5) for subsequent readings M_j, sensor node sends differential data $M_j - M_{i1}$); (6) cluster-head removes the corresponding reference entry M_{i1}, when the session ends for the sensor node i. SRDA provides data confidentiality, data integrity and source authentication.

To illustrate, let 30 °C denote the temperature measurement of a sensor node. If 25 °C is considered as reference temperature by the cluster head, the sensor node can send only the difference (i.e., 5 °C).

Based on an extended kalman filter which can help set up a normal range of the neighbor's future transmitted aggregated values, the authors [53] further applied an algorithm of combining cumulative summation (CUSUM) and generalized likelihood ratio (GLR), which can utilize the cumulative sum of the deviations between measured values and estimated values, and derive a normal range.

They pointed out that Intrusion Detection Modules (IDM) and System Monitoring Modules (SMM) needed to cooperate with each other in WSNs. When node i raises an alert on node j because of an event E, it starts an investigation on E by collaborating with existing SMMs to decide whether E is dangerous. Specifically, i can invoke sensor nodes around j and ask them for an opinion on E. If the majority of them believe that E may happen, i deems E triggered by some dangerous events.

An EKF based approach requires a function F relating the state s_k at time step t_k, to the state s_{k+1} at time step t_{k+1}. They used Intel Lab Data to plot the relationship between x_k and x_{k+1}. In their experiment, a faulty reading occurs when the reading difference is larger than 10 and a linear function form for F is: $x_{k+1} = x_k + w_k$. Here w_k denotes the process noise at time t_k. They further used Maximum Likelihood Estimation (MLE) to estimate $\sqrt{Q}$ where Q is the variance of distribution that w_k follows and the estimate is

$$\widehat{Q} = \frac{1}{n-1}\sum_{k=1}^{n-1}(x_{k+1} - x_k)^2.$$

The CUSUM-GLR based local detection is described as follows. Consider a sequence of observed random variables $y_0, y_1, \ldots, y_k$ with a probability density $p_\theta(y)$. $\theta = \theta_1$, after the change at t_0. To detect the change of θ, following hypothesis are formed about the parameter θ: $H_0 : \ \theta = \theta_0$, $H_1 : \ \theta = \theta_1$. The log-likelihood ratio is defined as:

$$s_k = ln\frac{p_{\theta_1}(y_k)}{p_{\theta_0}(y_k)},$$

s_k changes from a negative value to a positive value when a change occurs in θ. $S_N = \sum_{i=0}^{N} s_i$ is defined that it first decreases with N, and then increases after θ is changed from θ_0 to θ_1. S_N can detect the change in y_k. CUSUM-GLR provides data integrity and source authentication.

Yang et al. [54] pointed out that a fundamental challenge was how can the base station obtain a good approximation of the fusion result when a fraction of sensor nodes are compromised and more trust was placed on the high-level nodes compared to low-level sensor nodes in an aggregation tree. They proposed a Secure Hop-by-hop Data Aggregation Protocol (SDAP).

The design principles of SDAP is divide-and-conquer and commit-and-attest. First, SDAP uses a probabilistic grouping technique to dynamically partition the nodes into multiple logical groups. A commitment-based hop-by-hop aggregation is performed in each group to produce a group aggregation result. The base station then identifies the suspicious groups based on the set of group aggregation results. Finally,

each suspicious group joins in an attestation process to prove the correctness of its group aggregation result. The final aggregation result is calculated over all the group aggregation results that are either normal or have passed the attestation procedure. SDAP has three phases: query dissemination, data aggregation, and attestation. SDAP can provide data confidentiality, data integrity and source authentication.

Many existing data aggregation protocols assume that all sensor nodes in the network are fully trusted [18]. However, cryptographic primitives alone cannot provide a sufficient enough solution. Reputation and trust-based framework has proven to be effective in helping decision-making while each sensor node's reputation and trust is built over time. Zhu et al. [55] noticed that when a sensor node had a bad reputation for malfunctioning or suspended due to designated task of entering the period of dormancy, it was applicable to temporarily exclude this sensor node. They proposed an Integrity Preserving Event Report Protocol (IPERP) to validate an event and to add/remove members based on reputation and trust based framework.

Beta Distribution. Several distributions like Beta, Gaussian, Poisson, etc. can represent the reputation of a sensor node [59]. The Beta distribution has been the most prevailing one due to its simplicity, flexibility and strong foundations on the theory of statistics. It can be defined as:

$$P(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$$

where the interactions between i and j have two outcomes $O_{ij} = 1$ for successful and $O_{ij} = 1$ for unsuccessful. α and β are the number of cooperative and uncooperative interactions.

Modeling Reputation and Trust. The reputation of a sensor node j views from node i is given by:

$$R_{ij} = \frac{\Gamma(\alpha_{ij}+\beta_{ij})}{\Gamma(\alpha_{ij})\Gamma(\beta_{ij})} x^{\alpha_{ij}-1}(1-x)^{\beta_{ij}-1} = Beta(\alpha_{ij}+1, \beta_{ij}+1).$$

In the first round, $\alpha_{ij} = 0$ and $\beta_{ij} = 0$ for $\forall i, j$.

Trust is defined as the expectation of one sensor node about the behaviors of another node. The Trust towards node j from the view of node i is given by:

$$T_{ij} = E_{ij} = E(Beta(\alpha_{ij}+1, \beta_{ij}+1)) = \frac{\alpha_{ij}+1}{\alpha_{ij}+\beta_{ij}+2}.$$

Therefore, the range of Trust value is from 0 to 1.

Updating Reputation. When node i node j's reputation, it needs to update α_{ij} and β_{ij}. First, node i uses local observation to update α_{ij} and β_{ij} from time t to time $t+p$:

$$\alpha_{ij}^{t+p} = \alpha_{ij}^{t} + c, \ \beta_{ij}^{t} + uc.$$

where c/uc is the number of the cooperative/uncooperative interactions of j.

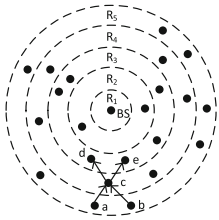


Fig. 2.12 An example of synopsis diffusion over a ring topology [60, 61]

Second, i updates α_{ij} and β_{ij} with second information:

$$\alpha_{ij}^{t+p} = \alpha_{ij}^{t+p} + \frac{\alpha_{kj}^{t+p} + \beta_{kj}^{t+p}}{\beta_{kj}^{t+p} + 1} T_{ik},$$

$$\beta_{ij}^{t+p} = \beta_{ij}^{t+p} + \frac{\alpha_{kj}^{t+p} + \beta_{kj}^{t+p}}{\alpha_{kj}^{t+p} + 1} T_{ik}.$$

When a member node P_r has a bad reputation and is ranked the lowest on the reputation list, CH chooses to rule out this node collaborating with t member nodes in the cluster which P_r is in. Each member node P_i follows a dynamic secret sharing and threshold group signature scheme to remove the node. IPERP provide data confidentiality, data integrity, source authentication and network availability.

Synopsis diffusion [60, 61] combines multi-path routing schemes with duplicate-insensitive algorithms to accurately compute aggregation result regardless of package loss. However, it does not consider the problem of false intermediate aggregation result contributed by compromised nodes which may result in large errors in the final aggregation result. Roy [56] et al. made the synopsis diffusion approach secure against this attack through a novel lightweight verification algorithm by which the base station can determine if aggregation result includes any false contribution.

During the query distribution phase, sensor nodes establish a set of concentric rings centered at base station (BS) according to their distance from BS like in Fig. 2.12. In the aggregation phase, each node generates and broadcasts a local synopsis $SG(v)$ where $SG()$ is the synopsis generation function and v is the sensed value. Each node then combines its own synopsis with child nodes' synopses using a synopsis fusion function $SF()$ and then broadcasts the merged synopsis. Finally, BS combines the received synopses using $SF()$ and then uses the synopsis evaluation function $SE()$ to translate the final synopsis. We now describe the duplicate-insensitive synopsis diffusion algorithm for Count.

In the Count algorithm, each node i generates a local synopsis Q^i which is a bit vector of length $\eta > N$, where N is the upper bound on Count. Each node executes the $CT(i, \eta)$ function to generate Q^i. Algorithm $CT()$ can be interpreted as a coin-tossing experiment and it returns c, the number of coin tosses until the head first occurs or η if $\eta + 1$ tosses have occurred with all tails. In the synopsis generation

function $SG()$, the ith bit of Q^i is set to "1" while all other bits are "0". The synopsis fusion function $SF()$ is the bitwise *Boolean OR* of the synopses being combined. The final synopsis F computed by BS is a bit vector of length η in the form $1^{z-1}0[0, 1]^{\eta-z}$, where z is the lowest order bit in F that is 0. The count of nodes in the network is $2^{z-2}/0.7735$ which is estimated by $SE()$. The Sum algorithm can be extended from computing Sum.

Their goal is to detect the falsified intermediate aggregation attack. Specifically, it is to detect if $\widehat{F}$, the synopsis received at BS is the same as the "true" final synopsis F. A straightforward solution is: BS broadcasts an aggregation query message which includes a random value r and a Seed s. In the aggregation phase, each node i sends authenticate its sensed value by computing $MAC_i(k_i, r, Seed, ID)$. As a result, BS can detect any false "1" bits inserted in the final synopsis F. However, to verify a particular "1" bit, say bit b, BS does not need to receive authentication messages from all of the nodes contributing to bit b. It is sufficient for each node to forward only one MAC corresponding to each "1" bit in the synopsis. SDA [56] provides data integrity and source authentication.

Roy et al. [57] improved their work and presented a attack-resilient algorithm to enable the base station to securely compute count or sum by filtering out the contributions of compromised nodes.

2.4 Summary

Chapter 2 introduces two series of secure and efficient data aggregation schemes in wireless sensor network. Not only do these schemes enable wireless sensors to securely communicate with base station, but also prolong the lifespan of the network.

References

1. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. 2002. A survey on sensor networks. *IEEE Commun. Mag.*, 40 (8): 102–114.
2. J. Yick, B. Mukherjee and D. Ghosal. 2008. Wireless sensor network survey. *Comput. Networks*, 52 (12): 2292–2330.
3. K. Akkaya, M. Demirbas, and R.S. Aygun. 2008. The Impact of Data Aggregation on the Performance of Wireless Sensor Networks. *Wiley Wireless Commun. Mobile Comput.(WCMC) J.*, 8: 171–193.
4. S. Ozdemir and Y. Xiao. 2009. Secure data aggregation in wireless sensor networks: A comprehensive overview. *Elsevier Computer Networks*, 53(12): 2022–2037.
5. L. Hu and D. Evans. 2003. Secure aggregation for wireless networks. *Proc. of the 3rd Symposium on Applications and the Internet(SAINT) Workshops.*, 384–391.
6. J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. 2006. Participatory sensing. *Proc. of the 1st ACM World-Sensor-Web(WSW) Workshops*, 1–5.
7. Jeff Howe. 2006. Crowdsourcing: a definition, crowdsourcing: Why the power of the crowd is driving the future of business. Available via DIALOG. http://crowdsourcing.typepad.com/cs/2006/06/crowdsourcing_a.html

8. R. Ganti, F. Ye, and H. Lei. 2011. Mobile crowdsensing: Current state and future challenges. *IEEE Communications Magazine*, 49(11): 32–39.
9. Crossbow Technology Inc. Mica2 datasheet, 2006. www.xbow.com/Products/productdetails.aspx?sid=174.
10. M. A. M. Vierial, A.B. da Cunda, and D. C. da S. Junio. 2006. Designing wireless sensor nodes. *Proc. of the 6th International Workshop on Embedded Computer Systems: Architectures, Modeling, and Simulation(SAMOS)*, 99–108.
11. W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. 2002. An application-specific protocol architecture for wireless microsensor networks. *IEEE Trans. Wireless Commun.*,1(4): 660–670.
12. O. Younis and S. Fahmy. 2004. HEED: a hybrid, energy-efficient distributed clustering approach for ad hoc sensor networks. *IEEE Trans. Mobile Comput.*, 3(4): 366–379.
13. Y. Yao and J. Gehrke. 2002. The Cougar approach to in-network query processing in sensor networks. *ACM SIGMOD Rec.*, 31(3): 9–18.
14. S. Chatterjea and P. Havinga. 2003. A dynamic data aggregation scheme for wireless sensor networks. *Proc. of the Program for Research on Integrated Systems and Circuits(RISC)'03*, 41(2): 116–125.
15. V. Mhatre and C. Rosenberg. 2004.Design guidelines for wireless sensor networks: communication clustering and aggregation. *Elsevier Ad Hoc Networks J.*, 2(1): 45–63.
16. P. Popovski, F. H. P. Fitzek, H. Yomo, T. K. Madsen, and R. Prasad. 2004. MAC-Layer Approach for Cluster-Based Aggregation in Sensor Networks. *Proc. of the International Workshop on Wireless Ad-hoc Networks(IWWAN)'04*, 89–93.
17. S. Pattem, B. Krishnamachari, and R. Govindan. 2004. The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks. *Proc. of the 3rd ACM/IEEE International Conference on Information Processing in Sensor Networks(IPSN)*, 28–35.
18. C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. 2002. Impact of network density on data aggregation in wireless sensor networks. *Proc. of the 22nd International Conference on Distributed Computing Systems(ICDCS)*, pp. 457–458.
19. C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. 2003. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Networking*, 11(1): 2–16.
20. B. Krishnamachari, D. Estrin, and S. Wicker. 2002. The impact of data aggregation in wireless sensor networks. *Proc. of the 22nd International Conference on Distributed Computing Systems(ICDCS) Workshops*, 575–578.
21. M. Ding, X. Cheng, and G. Xue. 2003. Aggregation tree construction in sensor networks. *Proc. of the 58th IEEE Vehicular Technology Conference(VTC)*, 58(4): 1049–1059.
22. R. Cristescu, B. Beferull-Lozano, M. Vetterli. 2004. On network correlated data gathering. *Proc. of the 23rd Conference of the IEEE International Conference on Computer Communications(INFOCOM)*, 4: 2571C2582.
23. S. Madden, M. J. Franklin, J. M, and W. Hong. 2002. TAG: A tiny aggregation service for ad hoc sensor networks. *ACM SIGOPS Operating Systems Review, OSDI*, 36(1): 131–146.
24. B. Zhou, L. H. NGOH, B. S. Lee, and C. P. Fu. 2004. A hierarchical scheme for data aggregation in sensor network. *Proc. of the 12th IEEE International Conference on Networks(ICON)*, 525–529.
25. M. Lee and V. W. S. Wong. 2005. An energy-aware spanning tree algorithm for data aggregation in wireless sensor networks. *Proc. of the IEEE PacRrim'05*, 300–303.
26. S. Lindsey, C. Raghavendra, and K.M. Sivalingam. 2002. Data gathering algorithms in sensor networks using energy metrics. *IEEE Trans. Parallel Distrib. Sys.*, 13 (13): 924C935.
27. G. Di Bacco, T. Melodia, and F. Cuomo. 2004. A MAC protocol for delay-bounded applications in wireless sensor networks. *Proc. of Med-Hoc-Net*, 208–220.
28. H. Alzaid, E. F. Juan, and G. Nieto. 2008. Secure data aggregation in wireless sensor network: a survey. *Proc. of the 6th Australasian conference on Information Security(AISC)*, 81: 93–105.
29. B. Przydatek, D. Song, and A. Perrig. 2003. SIA: secure information aggregation in sensor networks. *Proc. of the 1st ACM Conference on Embedded Networked Sensor Systems(SenSys)*, 255C265.

30. J. Newsome, E. Shi, D. Song, and A. Perrig. 2004. The Sybil attack in sensor networks: Analysis and defenses. *Proc. of the 3rd International Symposium on Information Processing in Sensor Networks(IPSN)*, 259–268.
31. W. Du, J. Deng, Y. S. Han, and P. K. Varshney. 2003. A witness-based approach for data fusion assurance in wireless sensor networks. *Proc. of the IEEE Global Telecommunications Conference(GLOBECOM)'04*, 1435–1439.
32. A. Wood and J. Stankovic. 2002. Denial of service in sensor networks. *IEEE Computer*, 35(10): 54–62.
33. J. McCune, A. P. E. Shi, and M. Reiter. 2005. Detection of denial-of-message attacks on sensor network broadcasts. *Proc. of IEEE Symposium on Security and Privacy(S&P)*.
34. D. Westhoff, J. Girao, and M. Acharya. 2006. Concealed data aggregation for reverse multicast traffic in sensor networks: encryption key distribution and routing adaptation. *IEEE Trans. on Mobile Comput.*, 5 (10): 1417–1431.
35. B. Przydatek, D. Song, and A. Perrig. 2003. SIA: Secure information aggregation in sensor networks. *Proc. of the 1st International Conference on Embedded Networked Sensor Systems(SenSys)*, 255–265.
36. D. Westhoff, J. Girao, M. Acharya. 2005. Concealed data aggregation for reverse multicast traffic in sensor networks. *Proc. of the IEEE International Conference on Communications(ICC)'05*, 3044–3049.
37. H. Chan, A. Perrig, and D. Song. 2006. Secure Hierarchical In-network Aggregation for Sensor Networks. *Proc. of the 13th ACM Conference on Computer and Communications Security(CCS)*, 278–287.
38. L. H. Zhu. and M. Li. 2011. An Energy Efficient and Integrity-Preserving Aggregation Protocol in Wireless Sensor Networks. *Proc. of the 30th IEEE International Performance Computing and Communications Conference(IPCCC)*, 1–6.
39. I. Rodhe and C. Rohner. 2008. n-LDA n-Layers Data Aggregation in Sensor Networks. *Proc. of the 28th International Conference on Distributed Computing Systems(ICDCS) Workshops*, 400–405.
40. C. Castelluccia, C. F. Chan, E, Mykletun, and, G. Tsudik. 2009. Efficient and provably secure aggregation of encrypted data in wireless sensor networks. *ACM Trans. Sensor Networks(TOSN)*, 5(3): 1137–1153.
41. C. M. Chen, Y. H. Lin, Y. C. Lin, and H. M. Sun. 2012. RCDA: Recoverable Concealed Data Aggregation for Data Integrity in Wireless Sensor Networks. *IEEE Trans.Parallel and Distributed Systems(TPDS)*, 23(4): 727–734.
42. L. H. Zhu, Z. Yang, M. Li, and D. Liu. 2013. An efficient data aggregation protocol concentrated on data integrity in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2013(7): 1–9.
43. L. H. Zhu, Z. Yang, M. Z. Wang, and M. Li. 2013. ID List Forwarding Free Confidentiality Preserving Data Aggregation for Wireless Sensor Networks. *International Journal of Distributed Sensor Networks*, 9(2013): 59–72.
44. L. H. Zhu, Z. Yang, J. F. Xue, and C. Guo. 2014. An efficient confidentiality and integrity preserving aggregation protocol in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2014(3): 1–8
45. K. A. Shim and C. M. Park. 2015. A secure data aggregation scheme based on appropriate cryptographic primitives in heterogeneous wireless sensor networks. *IEEE Trans. Parallel and Distributed Systems(TPDS)*, 26(8): 2128–2139.
46. R. C. Merkle. 1980. Protocols for public key cryptosystems. *Proc. of the 1st IEEE Symposium on Security and Privacy(S&P)*, 122–134.
47. R. C. Merkle. 1989. A certified digital signature. *Proc. of the 9th Annual International Cryptology Conference(Crypto)*, 218–238.
48. D. Johnson, A. Menezes, and S. Vanstone. 2001. The Elliptic Curve Digital Signature Algorithm(ECDSA). *International Journal of Information Security*, 1(1): 36–63.
49. J. Domingo-Ferrer. 2002. A provably secure additive and multiplicative privacy homomorphism. *Proc. of the 5th Information Security Conference*, 471–483.

50. E. Mykletun, J. Girao, and D. Westhoff. 2006. Public Key Based Cryptoschemes for Data Concealment in Wireless Sensor Networks. *Proc. IEEE International Conference on Communications(ICC)'06.*, 5: 2288–2295.
51. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. 2003. Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. *Proc. of the 22nd International Conference on the Theory and Applications of Cryptographic Techniques(Eurocrypt)*, 416–432.
52. H.O. Sanli, S. Ozdemir, and H. Cam. 2004. SRDA: secure reference-based data aggregation protocol for wireless sensor networks. *Proc. of the 71st IEEE VTC Fall Conference*, pp. 4650–4654.
53. B. Sun, N. Chand, K. Wu, and Y. Xiao. 2007. Change-point monitoring for secure in-network aggregation in wireless sensor networks. *Proc. of the IEEE Global Telecommunications Conference(GLOBECOM)'07*, pp. 936–940.
54. Y. Yang, X. Wang, S. Zhu, and G. Cao. 2006. SDAP: a secure hop-by-hop data aggregation protocol for sensor networks. *ACM Transactions on Information & System Security*, 11(4): 356–367.
55. Liehuang Zhu, Meng Li, and Lejian Liao. 2012. Dynamic Group Signature Scheme based Integrity Preserving Event Report. *Sensor Letters*, 10(8), pp. 1785–1791.
56. Sankardas Roy, Mauro Conti, Sanjeev Setia, and Sushil Jajodia. 2012. Secure Data Aggregation in Wireless Sensor Networks. *IEEE Transactions on Information Forensics & Security(TIFS)*, 7(3): 1040–1052.
57. Sankardas Roy, Mauro Conti, Sanjeev Setia, and Sushil Jajodia. 2014. Secure Data Aggregation in Wireless Sensor Networks Filtering out the Attacker Impact. *IEEE Transactions on Information Forensics & Security(TIFS)*, pp. 681–694.
58. R. L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin. 1998. The RC6 Block Cipher. http://www.princeton.edu/~rblee/RC6/.
59. G. Casella and R. L. Berger. 1990. Statistical Inference. *Duxbury Press*.
60. J. Considine, F. Li, G. Kollios, and J. Byers. 2004. Approximate aggregation techniques for sensor databases. *Proc. of the 20th IEEE International Conference on Data Engineering(ICED)*, 2004.
61. S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson. 2001. Synopsis diffusion for robust aggregation in sensor networks. *Proc. of the 2nd ACM Conference on Embedded Networked Sensor Systems(SenSys)*, 2004.

Springer

http://www.springer.com/978-981-10-3234-9

Secure and Privacy-Preserving Data Communication in Internet of Things

Zhu, L.; Zhang, Z.; Xu, C.

2017, VII, 78 p. 22 illus., 10 illus. in color., Softcover

ISBN: 978-981-10-3234-9