Research Article

An Efficient Confidentiality and Integrity Preserving Aggregation Protocol in Wireless Sensor Networks

Liehuang Zhu,1 Zhen Yang,1 Jingfeng Xue,2 and Cong Guo1

1 Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application, School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
2 School of Software, Beijing Institute of Technology, Beijing 100081, China

Correspondence should be addressed to Jingfeng Xue; xuejf@bit.edu.cn

Received 24 December 2013; Accepted 7 January 2014; Published 13 February 2014

Academic Editor: Zhongwen Guo

Copyright © 2014 Liehuang Zhu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Wireless sensor networks (WSNs) are composed of sensor nodes with limited energy which is difficult to replenish. In-network data aggregation is the main solution to minimize energy consumption and maximize network lifetime by reducing communication overhead. However, performing data aggregation while preserving data confidentiality and integrity is challenging, because adversaries can eavesdrop and modify the aggregation results easily by compromised aggregation nodes. In this paper, we propose an efficient confidentiality and integrity preserving aggregation protocol (ECIPAP) based on homomorphic encryption and result-checking mechanism. We also implement ECIPAP on SimpleWSN nodes running TinyOS. Security and performance analysis show that our protocol is quite efficient while persevering both aggregation confidentiality and integrity.

1. Introduction

Wireless sensor networks consist of a large number of sensor nodes which can be deployed to sense, transmit, and process the data collected from the environment in many applications such as battlefield surveillance, health care monitoring, and traffic regulation. The environment data will be transmitted to the base station (BS) hop by hop. Sensor nodes have limited energy storage and their computational ability is not so powerful as base station. We cannot supply energy to the nodes at all times because they are usually deployed in the areas humans hardly reach. If some key nodes cannot work as we supposed, the whole network will break down. The energy of the node is mainly spent on data transmission. On TelosB nodes, sending and receiving one bit data cost 0.72 μJ and 0.81 μJ which are more than 600 times than 1.2 nJ cost on processing one instruction by processor [1]. Therefore, how to reduce communication overhead is the key issue to prolong the life of wireless sensor networks [2, 3].

Other than transmitting data directly to the base station, wireless sensor nodes can perform aggregation operations, such as SUM and AVERAGE, on aggregation nodes. Through this approach, the communication overhead will be reduced largely. An aggregation tree should be built before data aggregation. The topology can be tree-based or cluster-based depending on the applications. Then the sensors need to collect environment data from the monitoring area and transmit the data to the base station hop by hop. When the intermediate nodes receive the data from their child nodes, they aggregate them using aggregation functions.

Sensor nodes should be deployed in the hostile areas in some applications, such as battlefield surveillance and target tracking. If these data was leaked to the enemies, it would bring heavy losses. The adversaries can contact the physical nodes easily and they can tamper the original sensing data so that the base station will receive the false aggregation results. If the base station cannot receive the true data, it will make wrong decisions. Some other attacks also affect the aggregation process like denial of service (DoS) or data selective forwarding. General data aggregation protocols do not consider these attacks, so they are useless in military applications. Designing a data aggregation protocol
should take security into consideration. All above, secure data aggregation mandates three security factors.

1.1. Data Confidentiality. The original sensing data and the intermediate aggregation results should not be disclosed to the unauthorized parties during the transmission process.

1.2. Data Integrity. If the original sensing data or the intermediate aggregation results were tampered by the adversaries or modified due to the poor quality of communications, the base station can detect the changes from final aggregation results.

1.3. Data Freshness. Sensing data received by the base station should represent the very recent state of the environment. If the aggregation values are too old, base station should discard such results.

Researchers have proposed many data aggregation protocols which can provide some secure protections. Most protocols can only provide confidentiality protection [4, 5] or integrity protection [6, 7]. Some protocols claim that they can protect both data confidentiality and data integrity, but some problems still exist. Data must be transmitted in some special forms if data integrity needs to be protected. But data form may be changed when some encryption algorithms are used to protect data confidentiality. They are contradictory in a way. Designing a data aggregation protocol which can provide both data confidentiality and data integrity is a challenge.

In this paper, we propose an efficient confidentiality and integrity preserving aggregation protocol (ECIPAP) in wireless sensor networks motivated by protocols SIES [8] and SHIA [9]. Our scheme is based on a lightweight homomorphic encryption which is energy efficient. Through the result-checking phase, every node in the wireless sensor network can verify if its data was added to the final aggregation results. We use a random number dissemination mechanism to update the keys stored in sensor nodes so as to guarantee the data freshness against replay attacks. This protocol also uses \( \mu \) TESLA [10] to broadcast authenticated queries along with the random numbers. We implement our protocol on TelosB physical nodes. Through the results of the experiment and the theoretic analysis, we will show the practicability and high efficiency of our protocol.

The paper is organized as follows. In Section 2, we introduce the related work in this research area. Section 3 explains the system model in our protocol. Section 4 describes our protocol in detail. Section 5 analyses the security, experiment, and performance of ECIPAP. We conclude this paper in Section 6.

2. Related Work

Hu and Evan proposed the first integrity-preserving hierarchy data aggregation protocol in 2003 [11]. The main idea of this protocol is delayed aggregation and delayed authentication. Girao et al. proposed an aggregation scheme that can guarantee the end-to-end data confidentiality using symmetric key based homomorphic encryption [12]. After that, a series of secure data aggregation protocols have been proposed. Existing protocols, which claim that they can protect both data confidentiality and data integrity, have some potential risks. Generally, we can think that both confidentiality and integrity preserving protocols should satisfy the following conditions:

(i) The encrypted sensing data should only be decrypted at the base station but not on the intermediate nodes to implement end-to-end confidentiality.

(ii) If the adversaries tamper the sensing data, base station can verify the data integrity through checking the final aggregation results.

(iii) The energy consumption should be reasonable because of the limited energy in each node.

The protocols proposed in [13–16] use pair keys to encrypt the data transmitted between two nodes based on the existing integrity-preserving data aggregation protocols. These protocols achieve hop by hop confidentiality but not end-to-end confidentiality. Based on a confidentiality protection mechanism, paper [17] provides data integrity protection through using a global key \( k^{(r)} \) and a key \( k^{(r)}_{i} \) shared with the base station to compute the authenticated message \( m_{i}k^{(r)} + k^{(r)}_{i} \) of message \( m_{i} \). This protocol is not secure if an adversary compromises a node and gets the global key \( k^{(r)} \), using which the adversary can forge message as wished. Protocol in [18] uses EC-ElGamal to achieve confidentiality protection. It also uses a global key \( k \) and the node’s private key \( z_{i} \) to compute the signature of message \( m_{i} \). However, through compromising a sensor node the adversary can get \( k \) which can be used to tamper the cipher and compute the corresponding signature. Paper [19] proposed a protocol that can protect the data confidentiality. Meanwhile, this protocol also can protect the aggregation data sent by cluster header from being tampered. But adversary can compromise a node and get the public key of the cluster; then the aggregation data can be tampered freely.

Chan et al. presented a provable secure tree-based in-network data aggregation protocol named SHIA [9]. Without assuming a particular data structure, SHIA can detect any manipulation of the in-network aggregation. SHIA has three phases: query dissemination phase, aggregation-commit phase, and result-checking phase. After the queries were broadcast to the whole network, sensor nodes send their environment data and commitment upward. In the result-checking phase, associated off-path values would be passed down to the aggregation tree. Through this way, sensor nodes can verify if their data was indeed added to the final aggregation results. After node \( i \) is convinced, it will send an authentication message \( MAC(K_{i}, N \parallel OK) \), where \( OK \) is a unique identifier and \( K_{i} \) is the key shared with base station, to base station hop by hop. The intermediate nodes perform XOR on these authentication messages and the final result received by base station is \( MAC(K_{i}, N \parallel OK) \oplus \cdots \oplus MAC(K_{i}, N \parallel OK) \). Base station computes this message itself and compares it with the authentication message received. If two messages match, base station accepts the aggregation result. Otherwise, the result will be ignored.
Although this protocol can protect data integrity, the data confidentiality is not guaranteed.

Combining homomorphic encryption and secret sharing, Papadopoulos et al. proposed a secure data aggregation protocol SIES which can protect both data confidentiality and data integrity [8]. In this protocol, each sensor node has a secret message $s_i$, a global key $K$, and a private key $k_i$ shared with base station. After the environment data $v_i$ was sensed from monitoring area, each sensor node generates a message $m_i = v_i \parallel s_i$. Sensor nodes use homomorphic encryption to encrypt $m_i$ and send them to their parent nodes. The parent nodes can perform aggregation functions on the encrypted data directly. When the final aggregation result is received by the base station, it will be decrypted and divided into two parts: environment data $V$ and secret SS. Base station computes the SS itself and compares it with the secret received. This protocol also has a problem that if the data $v_i$ of the message $m_i$ was tampered by adversaries, base station cannot detect such changes.

Our protocol improves SIES [8] by using result-checking mechanism motivated by SHIA [9] to protect data integrity instead of secret sharing. We also use homomorphic encryption to allow intermediate nodes perform aggregation directly so as to achieve end-to-end data confidentiality. When the final aggregating result was sent to the base station, the result-checking process will detect whether all the data is added to the final aggregation result.

3. System Model

Different secure data aggregation protocols support different aggregation functions. The security protection abilities are also various and the adversaries can launch a variety of attacks. This section gives the general problem definition, network assumption, and attack model in detail.

3.1. Network Assumption. We assume that an aggregation tree is already set up in the deployment phase. If not ready, TAG [20] can be used to build such tree-based networks. In our protocol, base station BS has unlimited energy and computing ability. BS can broadcast query messages to the whole network using authenticated method $\mu$TESLA [10]. In the aggregation tree, there are two types of nodes, leaf nodes and the intermediate nodes. Leaf nodes only sense environment data and encrypt them, while intermediate nodes not only sense environment data but also aggregate their child nodes’ data. The distance between two nodes is about 10 m and nodes can only communicate with its child nodes or parent nodes.

Each sensor node has a unique ID and an initial key $k_i$. We further assume that sensor nodes can perform symmetric additively homomorphic encryption and collision-resistant hash function.

3.2. Problem Definition. In our protocol, we use an efficient symmetric additively homomorphic encryption algorithm. A key distribution mechanism is also used to generate the keys in each sensor node.

3.2.1. Homomorphic Encryption. As shown in Algorithm 1, $m$ is a large integer. Let $m = \{0, M\}$ be the message that needs to be encrypted, and $k = \{0, M\}$ is the initial key stored in every node before deployment. Parameter $r$ is the random number sent by base station and $H$ is the hash function.

3.2.2. Message Format. The message $m$ sent by sensor node has a fixed format which is a data tuple:

$$\langle \text{count}, \text{value}, \text{value}, \text{MAC} \rangle,$$

where count is the number of the sensor nodes in the subtree. If a sensor node is the leaf node, then count = 1. Parament value is the environment data and it has lower bound $V_{\text{min}}$ and upper bound $V_{\text{max}}$. We also add value = $V_{\text{max}} - \text{value}$ to this data tuple. MAC can be computed as follows:

$$\text{MAC} = H(\text{count} || \text{value} || \text{value} || \text{ID}_i),$$

where $\text{ID}_i$ is node’s unique ID.

3.2.3. Aggregation Function. An intermediate node $i$ has $n$ child nodes $s_1, s_2, \ldots, s_n$. Let $v_1, v_2, \ldots, v_n$ be the data sensed by these child nodes from the monitory area. We define aggregation function $F_{\text{agg}}$ that can be performed on intermediate nodes as shown in Algorithm 2.

3.3. Attack Model. The most threatening attack against data aggregation in wireless sensor networks is node compromising launched by the adversaries. Through this attack, adversaries can obtain the keys of the nodes. After that, they can use fake messages to disturb the final aggregation results.

Such stealthy attacks [12] can make base station accept the false aggregation results without being detected. We assume that the adversaries can compromise a fraction of sensor nodes in the network.

Denial-of-service attack is out of our consideration, because this type of attack can be detected easily by the base station when the network works improperly.

4. ECIPAP

We improve SIES [8] by using result-checking mechanism instead of secret sharing. Then we propose an efficient confidentiality and integrity preserving aggregation protocol (ECIPAP). The homomorphic encryption used in our protocol can guarantee end-to-end data confidentiality. Our protocol has three phases: query dissemination phase, data aggregation phase, and result-checking phase.

4.1. Network Deployment. Before the sensor nodes were deployed in the monitory area, each sensor node shared a private key $k_i$, a large integer $M$, and a unique ID, with the base station. The symmetric additively homomorphic encryption algorithm and hash function SHA-1 are also preset. When the aggregation process begins, we assume that the aggregation tree is already set up. We can use TAG [20] to form an aggregation tree.
4.2. Query Dissemination Phase. In each aggregation round, base station BS chooses a random number \( r \) and the query message such as COUNT, SUM, and AVERAGE. Due to its powerful communication ability, it can broadcast this query message to the whole network along with the random number \( r \). When the nodes receive the query messages, they store them in the RAMs and start the data aggregation phase.

4.3. Data Aggregation Phase. Sensor node collects environment data value \( v \) such as temperature and sets \( \text{value} = V_{\text{max}} - v \), \( \text{count} = 1 \). Next it generates the temporary keys \( k_{i,r}, k'_{i,r}, \) and \( k''_{i,r} \):

\[
\begin{align*}
  k_{i,r} &= H(k_i, r) \mod M, \\
  k'_{i,r} &= H(k_i, r + 1) \mod M, \\
  k''_{i,r} &= H(k_i, r + 2) \mod M.
\end{align*}
\]

Then it encrypts these data as follows:

\[
\begin{align*}
  c_{\text{count},i} &= \text{Enc}(\text{count}, k_{i,r}, M), \\
  c_{\text{value},i} &= \text{Enc}(\text{value}, k'_{i,r}, M), \\
  c_{\text{value},i} &= \text{Enc}(\text{value}, k''_{i,r}, M).
\end{align*}
\]

The reason why we use different keys to encrypt count, value, and \( \overline{\text{value}} \) will be shown latter. Sensor nodes can compute message authentication code as follows:

\[
\text{MAC}_i = H \left( c_{\text{count},i} || c_{\text{value},i} || \overline{c_{\text{value},i}} || \text{ID}_i \right). \tag{5}
\]

The data tuple can be created now as

\[
u_i = \left( c_{\text{count},i}, c_{\text{value},i}, \overline{c_{\text{value},i}}, \text{MAC}_i \right). \tag{6}
\]

When the nodes prepare the data tuples ready, they send them to their parent nodes. We show the aggregation process in Figure 1. There are \( N = 10 \) nodes in the network and node \( F \) has two child nodes \( I \) and \( J \). When node \( F \) receives the data tuples \( u_i \) and \( u_j \) sent by node \( I \) and node \( J \), it aggregates them with the data tuple \( u_f \) created by itself: \( u_f = F_{agg}(u_i, u_j, u_f) \). Node \( F \) sends this intermediate aggregation result to its parent node \( C \). The final aggregation result will be transmitted to the base station BS at last. BS can decrypt this aggregation result as follows:

\[
\begin{align*}
  \text{count}_{agg} &= \text{Dec} \left( U_a \cdot \text{count}, \sum_{i=1}^{n} k_{i,r}, M \right), \\
  \text{value}_{agg} &= \text{Dec} \left( U_a \cdot \text{value}, \sum_{i=1}^{n} k'_{i,r}, M \right), \\
  \overline{\text{value}}_{agg} &= \text{Dec} \left( U_a \cdot \overline{\text{value}}, \sum_{i=1}^{n} k''_{i,r}, M \right).
\end{align*}
\]
4.4. Result-Checking Phase. When the final aggregation result is received, base station broadcasts the aggregated data tuple down to the whole network using authenticated method. To enable result checking, each sensor node will send a checking message to its child node. We can regard the result checking as a reverse process of data aggregation. We show this in Figure 2. Base station sends the final data tuple $U'_{f}$ to node $A$. Then node $A$ can remove the data sent by nodes $B$, $C$, and $D$ from $U'_{f}$. We define the reverse computational operation as $\ominus$. For example, we can get data tuple created by node $A$ as $u_{A}^{'} = U'_{f} \ominus u_{B} \ominus u_{C} \ominus u_{D}$. Through comparing $u_{A}^{'}$ with the data tuple $u_{A}$ created by node $A$, node $A$ can verify if its data was added to the final aggregation result.

Every sensor node can verify if its own data was added to the aggregation data by comparing its own data to the data sent by parent nodes. If the result passes the verification, then every sensor node prepares an authentication message $MAC(k_1 \parallel r \parallel OK)$ and sends it to the base station hop by hop. If verification is failed, $MAC(k_1 \parallel r \parallel NO)$ will be sent. When the intermediate sensor nodes receive these authentication messages, they aggregate them using MAC Aggregation Function. The base station also can calculate this authentication message with its own data stored before network deployment. So comparing these two authentication messages can verify if all the sensing data is added to the final aggregation result. If the authentication message passes the verification, then base station accepts this aggregation result. Otherwise, base station just ignores it.

5. Analysis

5.1. Security Analysis. Adversaries can compromise a fraction of sensor nodes in the wireless sensor network. When a sensor node is compromised, its private information such as encryption keys will be leaked. Adversaries can sniff confidential data sent by the sensor nodes. They can launch stealthy attack to make the base station accept false data without being detected.

We assume the adversaries can eavesdrop messages sent between sensor nodes and they already know the range of sensing data. If we use the same key to encrypt both value and $\ominus$ value, the adversaries can get the original sensing data through algebraic operation, such as $value_{ij} = (value_{ij} - value_{ij} - V_{max})/2$. Because we use different keys $k_{ij}$ and $k'_{ij}$ to encrypt count, value, and $\ominus$ value in ECIPAP, data confidentiality can be protected.

Authors in [9] define the direct data injection and optimally secure as follows.

5.1.1. Direct Data Injection. A direct data injection attack occurs when an adversary modifies the data readings reported by the nodes under its direct control, under the constraint that only legal readings in $[V_{min}, V_{max}]$ are reported.

5.1.2. Optimally Secure. An aggregation algorithm is 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.

The message format in our protocol is the same as the message format (vcount, value, complement, commitment) used in SHIA [9]. Adversaries can compromise some nodes and set all their readings to $V_{min}$ or $V_{max}$ so that the final aggregation result is in the range $[\mu V_{min}, \mu V_{max}]$ where $\mu$ is the number of malicious nodes. Obviously, any aggregation


The temperature data DataNum sensed by SimpleWSN nodes are 16-bit hexadecimal integers. We can use the following formula to transform such data to readable Celsius degree:

\[
\text{Temperature (^\circ C)} = \left(\frac{\text{DataNum}}{4096}\right) \times 1.5 - 0.98 - 0.00355.
\]  

(9)

Table 2 describes the parameters used in our experiment. We use sensor nodes to form an aggregation tree whose root is a powerful PC.

Table 3 shows the data received by base station before decryption. The data in count, value, and value columns are encrypted. We know this environment data is larger than the upper bound \(V_{\text{max}}\), so we cannot get any useful information from such messages.

Then, we use the keys shared between sensor nodes and base station to decrypt the aggregation results and show them in Table 4. The total number of nodes used is 5 and the average temperature is about 23.2 degrees Celsius. Base station broadcasts query messages every 1000 million seconds. Our experiment shows the practicability and high efficiency of ECIPAP.

5.3. Performance Analysis. We give a theoretical analysis of the performance in this section. Through comparing the communication overhead between ECIPAP, SHIA, and SIES, we show that ECIPAP not only protect both data integrity and data confidentiality but also reduce the communication overhead in result-checking phase.

We assume every intermediate node in the network has \(d\) child nodes and the height of the aggregation tree is \(h\). The length of the data tuple sent by sensor nodes in ECIPAP, SHIA, and SIES are \(|m|\), \(|m'|\) and \(|m''|\). The length of the
Table 5: Communication overhead of ECIPAP, SHIA, and SIES.

|                          | ECIPAP                              | SHIA                              | SIES                              |
|--------------------------|-------------------------------------|-----------------------------------|-----------------------------------|
| Edge congestion          | $|m|$                                | $h \times (d - 1) \times |m'|$ | $|m''|$                           |
| Node congestion          | $d \times |m|$                           | $d \times h \times (d - 1) \times |m'|$ | $d \times |m''|$                     |
| Data aggregation         | $\sum_{i=1}^{h} d^i \times |m|$ | $\sum_{i=1}^{h} d^i \times |m'|$ | $\sum_{i=1}^{h} d^i \times |m''|$ |
| Authentication           | $\sum_{i=1}^{h} d^i \times |MAC|$ | $\sum_{i=1}^{h} d^i \times |MAC'|$ | $-$                              |
| MAC transmission         | $\sum_{i=1}^{h} d^i \times |MAC|$ | $\sum_{i=1}^{h} d^i \times |MAC'|$ | $-$                              |

Figure 3: Communication overhead in authentication process when node degree $d = 3$.

Figure 4: Total communication overhead in ECIPAP, SHIA, and SIES when node degree $d = 3$.

6. Conclusion

It is difficult to design an aggregation protocol to protect both data confidentiality and data integrity in wireless sensor networks. Existing protocols which claim to achieve this goal still have some problems. In this paper, we improve SIES by using result-checking mechanism instead of secret sharing. We also use homomorphic encryption algorithm to protect data end-to-end confidentiality. We concentrate on the communication overhead cost in the result-checking phase and propose an efficient confidentiality and integrity preserving aggregation protocol (ECIPAP). We implement our protocol on physical nodes running TinyOS. Through this experiment and theoretical analysis, we show the practicability and high efficiency of ECIPAP.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This paper is supported by National Natural Science Foundation of China (no. 61272512, no. 61003262, and no. 61300177), Program for New Century Excellent Talents in University (NCET-12-0047), and Beijing Natural Science Foundation (no. 4121001, no. 4132054).
References

[1] M. M. Groat, W. B. He, and S. Forrest, “KIPDA: k-indistinguishable privacy-preserving data aggregation in wireless sensor networks,” in Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM ’11), pp. 2024–2032, Shanghai, China, April 2011.

[2] M. Li and Y. H. Liu, “Iso-map: energy-efficient contour mapping in wireless sensor networks,” IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 5, pp. 699–710, 2010.

[3] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” Computer Networks, vol. 52, no. 12, pp. 2292–2330, 2008.

[4] T. Feng, C. Wang, W. Zhang, and L. Ruan, “Confidentiality protection for distributed sensor data aggregation,” in Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM ’08), pp. 68–76, Phoenix, Ariz, USA, 2008.

[5] E. Myklebost, J. Girao, and D. Westhoff, “Public key based crypto schemes for data concealment in wireless sensor networks,” in Proceedings of the 41th IEEE International Conference on Communications (ICC ’06), vol. 5, pp. 2288–2295, Istanbul, Turkey, July 2006.

[6] M. Manulis and J. Schwenk, “Security model and framework for information aggregation in sensor networks,” ACM Transactions on Sensor Networks, vol. 5, no. 2, pp. 1–28, 2009.

[7] L. Yu, J. Li, S. Cheng, and S. Xiong, “Secure continuous aggregation via sampling-based verification in wireless sensor networks,” in Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM ‘11), pp. 1763–1771, Shanghai, China, 2011.

[8] P. Papadopoulos, A. Kiayias, and D. Papadias, “Exact in-network aggregation with integrity and confidentiality,” IEEE Transactions on Knowledge and Data Engineering, vol. 24, no. 10, pp. 1760–1773, 2012.

[9] H. Chan, A. Perrig, and D. Song, “Secure hierarchical in-network aggregation in sensor networks,” in Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS ’06), pp. 278–287, Alexandria, Va, USA, November 2006.

[10] A. Perrig, R. Szewczyk, V. Wen, J. D. Tygar, and D. E. Culler, “SPINS: security protocols for sensor networks,” Wireless Networks, vol. 8, no. 5, pp. 521–534, 2002.

[11] L. Hu and D. Evan, “Secure aggregation for wireless networks,” in Proceedings of the IEEE Symposium on Applications and the Internet Workshops (SAINT ’03), vol. 2003, pp. 384–391, Orlando, Fla, USA.

[12] J. Girao, D. Westhoff, and M. Schneider, “CDA: concealed data aggregation for reverse multicast traffic in wireless sensor networks,” in Proceedings of the 40th IEEE International Conference on Communications (ICC ’05), pp. 3044–3049, Seoul, Republic of Korea, May 2005.

[13] J. Pawan and M. Anish, “Efficient secure aggregation in sensor networks,” in Proceedings of the 1st International Conference on High-Performance Computing (HiPC ’04), pp. 40–49, 2004.

[14] E. O. Blaß, J. Wilke, and M. Zitterbart, “Relaxed authenticity for data aggregation in wireless sensor networks,” in Proceedings of the 4th International Conference on Security and Privacy in Communication Networks (SecureComm ’08), pp. 1–10, Istanbul, Turkey, September 2008.

[15] S. Ozdemir and H. Çam, “Integration of false data detection with data aggregation and confidential transmission in wireless sensor networks,” IEEE/ACM Transactions on Networking, vol. 18, no. 3, pp. 736–749, 2010.

[16] M. S. Hung, C.-H. Chen, and P.-C. Li, “A lightweight secure data aggregation protocol for wireless sensor networks,” in Proceedings of the International Conference on Parallel Processing Workshops (ICPPW ’11), pp. 101–107, Taipei, Taiwan, September 2011.

[17] C. Castelluccia, A. C.-F. Chan, E. Myklebost, and G. Tsudik, “Efficient and provably secure aggregation of encrypted data in wireless sensor networks,” ACM Transactions on Sensor Networks, vol. 5, no. 3, pp. 1–36, 2009.

[18] J. Albath and S. Madria, “Secure hierarchical data aggregation in wireless sensor networks,” in Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC ’09), pp. 1–6, Budapest, Hungary, April 2009.

[19] S. Ozdemir and Y. Xiao, “Integrity protecting hierarchical concealed data aggregation for wireless sensor networks,” Computer Networks, vol. 55, no. 8, pp. 1735–1746, 2011.

[20] S. Madden, J. M. Franklin, J. Hellerstein et al., “TAG: a tiny aggregation service for adhoc sensor networks,” in Proceedings of the 5th Annual Symposium on Operating Systems Design and Implementation (OSDI ’02), pp. 131–146, Boston, Mass, USA, 2002.
