Chapter 18
Privacy Enhancing Technologies for Wireless Sensor Networks

Chi-Yin Chow, Wenjian Xu and Tian He

Abstract Since wireless sensor networks (WSNs) are vulnerable to malicious attacks due to their characteristics, privacy is a critical issue in many WSN applications. In this chapter, we discuss existing privacy enhancing technologies designed for protecting system privacy, data privacy and context privacy in wireless sensor networks (WSNs). The privacy-preserving techniques for the system privacy hide the information about the location of source nodes and the location of receiver nodes. The data privacy techniques mainly protect the privacy of data content and in-network data aggregation. The context privacy refers to location privacy of users and the temporal privacy of events. For each of these three kinds of privacy in WSNs, we describe its threats and illustrate its existing privacy-preserving techniques. More importantly, we make comparisons between different techniques and indicate their strengths and weaknesses. We also discuss possible improvement, thus highlighting some research trends in this area.

1 Introduction

Privacy is a critical issue when applying theoretical research in wireless sensor networks (WSNs) to scientific, civilian and military applications [35, 45], e.g., environmental sensing, smart transportation and enemy intrusion detection.

C.-Y. Chow · W. Xu
Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
e-mail: chiychow@cityu.edu.hk

W. Xu
e-mail: wenjianxu2@student.cityu.edu.hk

T. He (✉)
Department of Computer Science and Engineering, University of Minnesota,Minneapolis, MN, USA
e-mail: tianhe@cs.umn.edu

H. M. Ammari (ed.), *The Art of Wireless Sensor Networks*,
Signals and Communication Technology, DOI: 10.1007/978-3-642-40009-4_18,

WSNs are vulnerable to privacy breaches because they possess the following characteristics:

- **Wireless communication.** Wireless sensors need to communicate with each other through wireless communication. Wireless communication signals are easy to be tracked or eavesdropped by adversaries. We show later in this chapter that, in some kinds of applications, privacy breaches take place when adversaries are able to track or eavesdrop wireless communication signals even if the content of the transmitted data is protected securely.
- **Open environments.** WSNs are usually deployed in open environments to provide sensing and monitoring services. Such open environments could cause privacy concerns because malicious people can easily approach the system area or even physically access the sensor.
- **Large-scale networks.** The number of sensor nodes in a WSN is often large, so that protecting every node from being compromised by adversaries is difficult. Thus, the privacy enhancing technology designed for the WSN should be able to deal with a situation that the network contains some compromise sensor nodes which can be controlled by adversaries.
- **Limited capacity.** In general, wireless sensors have scarce resources, e.g., limited computational power, constrained battery power, and scarce storage space. As a result, existing privacy enhancing technologies designed for the Internet or wireless networks are not applicable to WSNs.

Due to these limitations, it is very challenging to design secure privacy-preserving techniques for WSNs. It is essential for researchers to study existing privacy enhancing technologies for WSNs, investigate their strengths and weaknesses, and identify new privacy breaches to improve them. To help researchers to understand the state-of-the-art privacy enhancing technologies for WSNs, we category them into three main types of privacy, namely, *system privacy*, *data privacy*, and *context privacy*. These three kinds of privacy are defined as follows:

1. **System privacy** is the ability of a system to protect the information about the setting of its WSN (e.g., the location information of its base station) and the communication information among its network components (e.g., the source node of data).
2. **Data privacy** is the ability of a system to preserve the data content through the course of transmission or in-network aggregation.
3. **Context privacy** is the ability of a system to protect the user location monitored by sensor nodes, or the time when an event is detected by sensor nodes.

For each of these three kinds of privacy, we discuss its threats and then highlight its existing privacy-preserving techniques.

The rest of this chapter is organized as follows: Sect. 2 presents an overview of this chapter. Sections 3, 4, and 5 describe the threat models and solutions of system privacy, data privacy, and context privacy, respectively. Section 6 concludes this chapter and discusses future research directions.

2 Overview of This Chapter

Various WSN applications require different privacy protection techniques. For example, in an event detection application, the location information of source sensor nodes is the sensitive information and can be inferred by adversaries through wireless communication signal analysis even without knowing the transmitted data content. Such event detection applications require system privacy protection. In a data collection application, its sensor node's readings are sensitive and should be protected during the course of transmission. Thus, data collection applications need data privacy protection. In a location monitoring application, the location information of monitored individuals is sensitive and should be protected. Location monitoring applications call for context privacy protection. From these three applications, we can see that different types of WSN applications have their own definition of sensitive information and they require different privacy protection techniques. In other words, exiting privacy enhancing technologies for WSNs are application-oriented because many of them are designed for a particular WSN application.

In this chapter, we identify three main types of privacy for existing WSN applications, namely, system privacy, data privacy, and context privacy. For each type of privacy, we first discuss its threat model and then describe its protection techniques, as depicted in Fig. 1. For system privacy, we mainly discuss privacy-preserving techniques designed to protect the source node of data and the location information of base stations. For data privacy, we survey privacy-preserving techniques for protecting data content during transmission or in-network aggregate processing. For context privacy, we focus on protection techniques designed for preserving the location privacy of people monitored by sensor nodes and the temporal information of events detected by sensor nodes.

3 System Privacy

Tracking wireless communication signals in a WSN could reveal different kinds of sensitive information such as which sensor node generates a reading, which sensor node is the destination of a packet and which sensor nodes are near a base station (or a

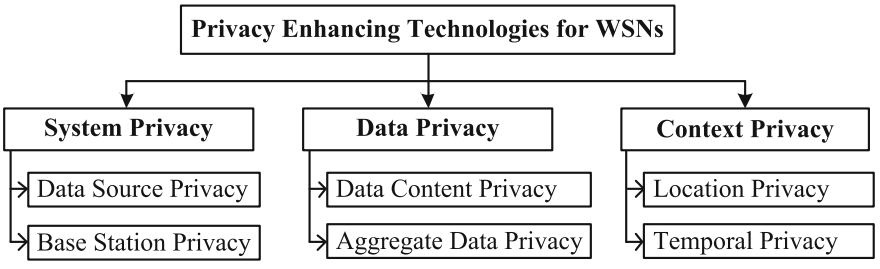

Fig. 1 Our taxonomy of privacy enhancing technologies for wireless sensor networks (WSNs)

sink node). For example, sensors deployed for detecting and monitoring a particular type of endangered animal in a forest (e.g., giant pandas [57]) can be utilized by hunters to locate the monitored animals. After a sensor node detects a target animal, it sends a report to the base station. A hunter can eavesdrop wireless communication signals to trace from the base station back to the source node, even without capturing and analyzing the transmitted data content. This kind of privacy breach would cause serious consequences and is difficult to be detected. More powerful adversaries can also compromise a sensor node and capture packets to learn the routing path from the source sensor node to the base station.

Under the system privacy, we identify two main types of privacy issues, namely, *data source privacy*, and *base station privacy*. Adversaries may locate the data source or the base station by analyzing wireless communication signals. Privacy-preserving techniques designed for protecting data source and base station privacy may prevent such malicious attacks.

3.1 Data Source Privacy

Consider our example where a WSN is deployed for detecting and monitoring a particular type of endangered animal in a forest. After a sensor node detects a target animal, it generates a report about the animal's activities and send the report to the base station through a multi-hop routing protocol [2, 3]. Suppose an adversary is equipped with devices such as antenna and spectrum analyzers that can be used to capture wireless signals between sensor nodes and measure the angle of arrival of signals. And all sensor nodes keep silent until they detect a target animal in their sensing area. After a sensor node detects a target animal, it sends a report to the base station. The adversary would capture wireless communication signals along the data transmission path from the source sensor node to the base station, and then he can carry out a traffic backtracking attack to find the source sensor node by capturing wireless signals for each hop, analyzing their direction, and then identifying the sender of each hop, as illustrated in Fig. 2. In practice, the adversary could use traffic backtracking to locate the endangered animals and bring danger to them, even though he is not able to read the report content. In this section, we discuss existing privacy enhancing techniques for two main types of attacks: local eavesdropping (Sect. 3.1.1) and global eavesdropping (Sect. 3.1.3).

3.1.1 Local Eavesdropping

Phantom routing is a routing protocol designed to prevent the traffic backtracking attack [32, 46]. The main idea of the phantom routing protocol is to select a phantom source node at a location far away from the real one. The protocol consists of two main steps:

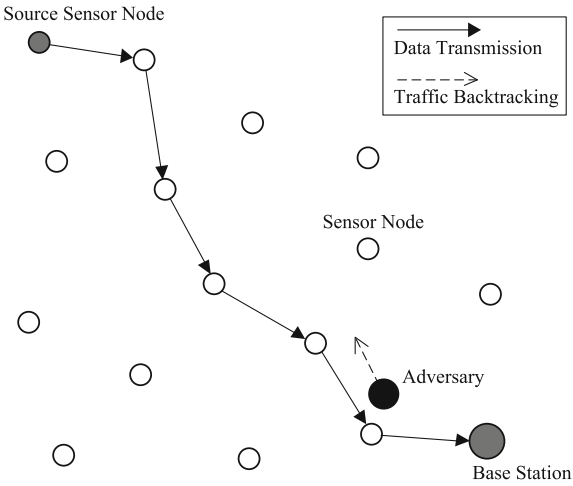

Fig. 2 The traffic backtracking attack in a WSN

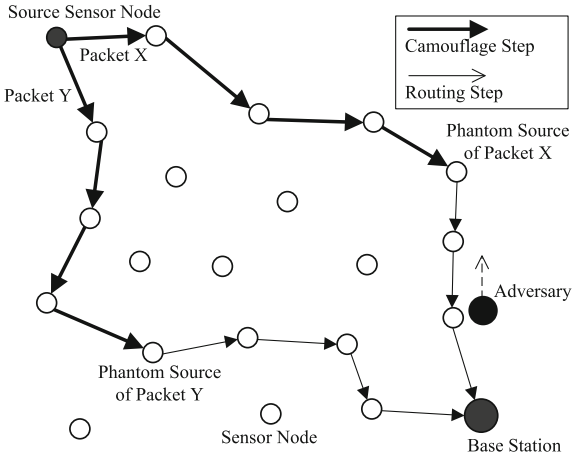

Fig. 3 The phantom routing paths of two data packets from the real source node to the base station

1. **Camouflage step.** This step ensures that a phantom source sensor node is located far away from the real source node. For example, when a sensor node wants to send out a packet, the packet is forwarded to the other node with hop distance h.
2. **Routing step.** This step makes sure that the packet can be delivered to the base station. After forwarding the data packet h hops, a conventional routing protocol (e.g., flooding [39], probabilistic broadcast [19], or single-path routing [28, 33]) is used to route the packet to the base station.

Figure 3 shows an example of the phantom routing where $h = 4$ and a source sensor node sends out two packets X and Y. During the camouflage step (i.e., the first four hops), the routing paths of the packets X and Y (indicated by bold arrows) from the real source sensor node are different. After forwarding the packets 4 hops,

they are forwarded to the base station through a single-path routing protocol. These camouflage paths make an adversary difficult to backtrack to the source sensor node in a given time period. The basic idea is that the adversary can trace back to a certain intermediate node by capturing the transmission signal of one of these two packets, but he may never catch any other packet from the real source sensor node at that intermediate node because the routing paths of other packets do not pass through that node. In this way, the phantom routing could direct the adversary to a place far away from the real source sensor node, and therefore it cuts off the backtracking of the adversary.

We next discuss several techniques that have been proposed for the camouflage step.

1. **Random walk** [32]**.** For each of the first h hops, the sensor node randomly chooses a neighbor node to forward the packet. Since such a random walk may visit a sensor node more than once and the node at the h-th hop (i.e., the phantom source) may remain clustered around the actual source node, the pure random walk is inefficient for leading the phantom source node to be far away from the actual source node [32, 46].
2. **Neighbor-grouping-based directed walk** [32]**.** When a sensor node forwards a packet, it divides its neighbor nodes into two sets S and S'. The node first randomly selects one of these two sets and then randomly forwards the packet to a neighbor node in the selected set. For example, the neighbor nodes can be divided into two sets based on their hop distance to the base station, i.e., S includes all the neighbor nodes with the hop distance smaller than or the same as the sensor node's hop distance and S' includes all the nodes with the hop distance larger than the sensor node's hop distance.
3. **Greedy random walk** [58]**.** Before a sensor node forwards a packet, it uses a Bloom filter [5] to store the identifiers of itself and its neighbor nodes and then forwards the packet with the Bloom filter. After a neighbor node receives the packet, it randomly selects a neighbor node whose identifier is not stored in the Bloom filter as the next hop, adds the identifiers of its neighbor nodes in the Bloom filter, and then forwards the packet with the updated Bloom filter.
4. **Minimum distance** [37]**.** When a sensor node wants to send out a packet, it randomly selects the location of a phantom source node such that the distance between the sensor node and the phantom source node is at least certain distance d_{min}. Figure 4 illustrates this technique in our example, where both packets X and Y are first forwarded to their phantom source nodes that are far away from the source sensor node by at least d_{min}. Then, the phantom source nodes route X and Y using a single-path routing protocol. However, the sensor node may not have the actual location of every node in the system, if an intermediate node that is able to tell that the selected phantom source node does not exist, it becomes the phantom source node. The actual source node can also select all the intermediate nodes on the routing path to a selected phantom node using either an angle-based or a quadrant-based approach.

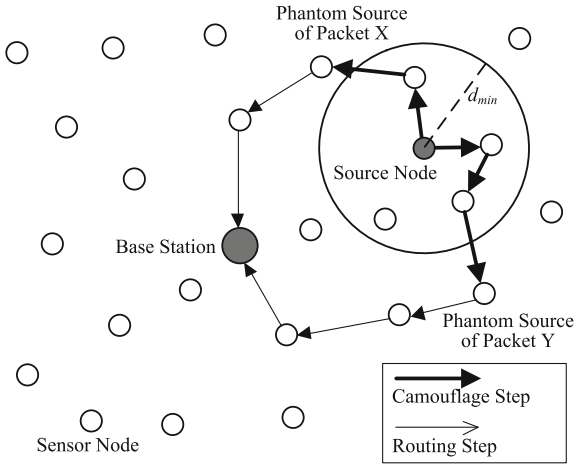

Fig. 4 The minimum distance technique in the camouflage step of phantom routing

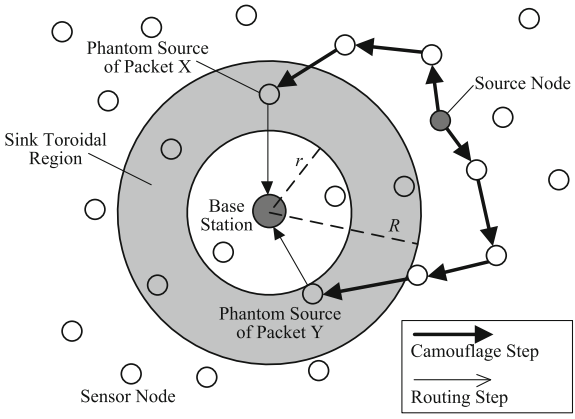

Fig. 5 The sink toroidal region (STaR) technique in the camouflage step of phantom routing

5. **Sink toroidal region** [38]**.** In this technique, a phantom source node is always selected inside a sink toroidal region (STaR) around a base station or a sink. The STaR is defined by two circles centered at the base station with radius r and R, where $r < R$. The intersection area between these two areas constitutes the STaR. A source sensor node randomly selects a location within the STaR as the phantom source, and then forwards its packet to it, as in the minimum distance technique. The STaR area should be large enough such that it is not practical for an adversary to monitor the entire STaR. Figure 5 depicts STaR in our example, where the shaded donut shape is the STaR. Packets X and Y are first forwarded to their randomly selected phantom source nodes inside the STaR and then a single-path routing protocol is used to forward them to the base station.

3.1.2 Discussion

All these techniques share the same goal: randomly finding a phantom source node that is far away from the actual source node. The first three techniques require an intermediate node to make its local decision to find a phantom node, while the last two techniques allow the source sensor node to determine the phantom node, and even all the intermediate nodes in the routing path between itself and the phantom node in the minimum distance technique. The last two techniques are more reliable, energy-efficient, and secure, because the sensor node has more control over the selection of its phantom node. However, if the sensor node does not have a global view of the system, it may not be able to use the minimum distance or STaR technique. It would be useful to extend them to allow intermediate nodes to make their local decision based on the source sensor node's requirements and its performance can be the same as the centralized decision scenario.

3.1.3 Global Eavesdropping

In the previous section, we discussed the privacy-preserving techniques designed for a scenario that an adversary can only eavesdrop on a limited portion of the network at a time, i.e., a *local eavesdropper*. In this section, we consider a stronger adversary called a *global eavesdropper* [41, 51, 59] who is capable to deploy his sensor nodes or compromise sensor nodes to monitor the communication traffic in the entire WSN. With a global view of the network traffic, the adversary can easily infer the location of a source node using the traffic analysis attack.

The basic approach of preventing the traffic analysis attack from global eavesdroppers is to inject dummy traffic into a WSN to make adversaries confused and thus unable to distinguish a real data source from a set of dummy data sources. There are three main kinds of techniques to inject dummy traffic into a WSN:

1. **Periodic collection** [41]. Every sensor node independently and periodically sends out data packets regardless of whether it has real data packets to send out or only dummy data packets. More specifically, each sensor node has a timer that fires at a constant rate. When the timer fires, if the node has a packet in its buffer, it forwards the packet; otherwise, it sends a dummy packet with a random payload. In this way, the adversary is not able to distinguish a real source sensor node from other dummy sensor nodes because they are all sending out data packets in the same manner. The drawback of this solution is that a network with more sensor nodes incurs higher communication overhead.
2. **Source simulation** [41]. Although the periodic collection technique provides optimal source-location privacy, if the constant rate is small, the system delay may be very high; if the rate is large, the system may have too much dummy traffic and suffer from high power consumption. The source simulation technique artificially creates multiple fake traces in the network to hide the traffic generated by real objects. For example, if a WSN is designed to monitor pandas in a forest,

their historical habits and behaviors are studied to create fake traces for pandas. Before deployment, we randomly select a set of sensor nodes and initialize a unique token in each of them. These tokens are transmitted between sensor nodes in the network to simulate the behavior of pandas. Thus, the adversary who wants to trace pandas using the traffic analysis attack probably finds a virtual one.

3. **Probabilistic dummy generation** [51]. The objective of this technique is also to reduce the amount of dummy traffic and latency of the periodic collection technique. This technique uses the exponential distribution to control the rate of dummy generation by creating a sequence of dummy packets such that the time intervals between two consecutive messages follow the predefined exponential distribution. For a packet generated by a real event, the packet is delayed as long as its transmission time also follows the predefined exponential distribution. The whole concept of this technique is based on a statistical property: if two probabilistic distributions are both exponential distributions with very close means, they are statistically indistinguishable from each other. Experimental results show that this technique effectively reduces communication overhead while it can provide the same level of privacy protection as the periodic collection technique.

3.1.4 Discussion

The basic idea of these dummy traffic injecting techniques is to use dummy traffic to hide the real data source sensor nodes. There is a tradeoff between the communication overhead (e.g., bandwidth and power consumption) and the privacy protection [29]. K. Mehta et al. provide a method to estimate a lower bound on the communication overhead needed to achieve a certain level of privacy protection in the network [41]. In addition, some scientists study how to reduce communication overhead for dummy traffic without sacrificing any privacy protection. For example, Y. Yang et al. suggest placing some proxy sensor nodes in a WSN to filter out dummy packets and drop them after a certain number of hops of transmission [59].

There is a new research direction for source data privacy in unattended wireless sensor networks (UWSNs), where critical sensor nodes replicate their readings in a certain number of randomly selected nodes d-hop away from the critical sensor nodes. Recent study has found that there is a tradeoff between source-location privacy and data survivability (i.e., the number of data replicas) [7]. When an adversary finds the data source node, he can destroy the critical node in the system.

3.2 Base Station Privacy

In some WSN applications, such as security monitoring systems, the physical location of a base station (or a sink node) is considered as sensitive information. The main reason is that revealing the physical location of a base station to the adversary may give him a chance to make either physical or denial-of-service attacks to the

base station in order to disable the WSN [15]. However, many routing protocols would reveal obvious traffic patterns in the network. The sensor nodes near the base station forward a greater number of data packets than other sensor nodes that are far away from the base station [16]. Based on such unbalanced traffic, it is easy for the adversary to infer the base station's physical location. The main idea of hiding the location information of the base station is similar to that of the phantom routing and dummy traffic injection techniques, but it focuses on how to make the adversary difficult to distinguish a sink node (or a base station) from other sensor nodes. We describe four techniques for protecting the base station privacy.

3.2.1 Multi-Parent Routing

In the multi-parent routing [16], each sensor node has multiple parent nodes. Each sensor node s finds its multiple parents based on their hop distance to the base station. The base station broadcasts a beacon message with a *level* field that is initially set to one. When s receives the beacon message, the value of *level* indicates s's hop distance to the base station. s next increases *level* by one and rebroadcasts the beacon message with the increased *level* value to its neighbors. After a certain time period, s selects all neighbor nodes whose *level* value is less than s's *level* value as its parent nodes. Figure 6 depicts an example, where a WSN consists of seven sensor nodes s_1 to s_7 and each sensor node already finds its *level* value. s_3 have four neighbor nodes (i.e., s_1, s_2, s_6, and s_7), but only the *level* values of s_1 and s_2 are less than s_3's *level* value. Thus, s_1 and s_2 are s_3's multiple parent nodes. Each sensor node erases its *level* value after all the sensor nodes have found their multiple parent nodes. When a sensor node wants to send a data packet, it randomly selects one of its multiple parent nodes to send the packet. Although this technique is similar to the *phantom routing* technique, it aims at spreading out traffic evenly in the whole WSN to make the adversary difficult to infer the physical location of the base station instead of finding an intermediate phantom node. Multi-parent routing often works with random walk and fractal propagation, as will be discussed later, to enhance the privacy-protecting performance.

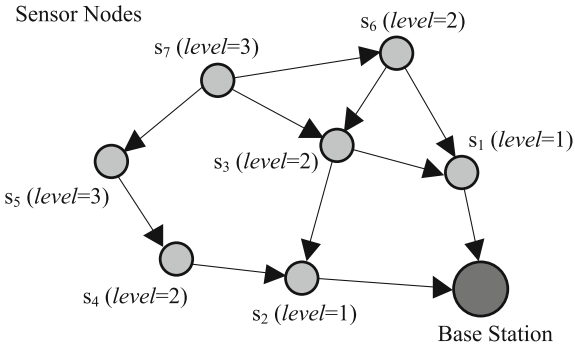

Fig. 6 Multi-parent routing

3.2.2 Random Walk

Although a random walk has been proposed for reliable data transmission [56], it can also be used to protect the base station privacy [16]. The basic idea is that a sensor node has two ways to forward a packet: (1) the node forwards the packet to one of its parent nodes with equal probability or (2) it forwards the packet of one of its neighbors with equal probability. The node uses the first way with probability p or the second one with probability $1 - p$.

3.2.3 Fractal Propagation

The multi-parent routing and random walk techniques can avoid forwarding packets to the base station through the shortest or static path. The fractal propagation technique can further spread out fake packets [16]. When a sensor node overhears that its neighbor node forwards a data packet, it also generates a fake data packet following a predefined probability distribution with a system parameter k and forwards it to a randomly selected neighbor node. After the neighbor node receives the fake packet, it reduces the value of k by one and forwards it to one of its neighbor nodes with the updated k value. When a node receives the fake packet with $k = 0$, it simply drops the packet. As a result, this technique spreads out the communication traffic of a data packet in the network. This technique works due to the fact that the normal nodes can distinguish a fake packet from a real one, while the adversary cannot distinguish since it does not know the encryption key. If we use this technique with the multi-parent routing and random walk techniques, a full picture of the real and fake communication flows triggered by a data packet transmission looks like a fractal shape, as depicted in Fig. 7.

3.2.4 Hiding Traffic Direction Information

The goal of hiding traffic direction information is to make the directions of both incoming and outgoing traffic at a node uniformly distributed [30]. The basic idea is that each sensor node divides its neighbor nodes into two disjoint lists: *closer* and *further lists*, where the closer list contains the neighbor nodes that are closer to a receiver and the further list consists of the neighbor nodes that are further way from the receiver. The distance between two sensor nodes can be measured by their hop distance or their Euclidean distance. When a sensor node forwards a data packet, it selects the next hop from the further list with probability p, and from the closer list with probability $1 - p$. p is used to balance a performance trade-off between communication overhead (i.e., latency and power consumption) and privacy. For example, if p is smaller, the node selects more next hops in the closer list. Thus, the routing paths are shorter and the power consumption is lower, but the receiver's location privacy is weaker. On the other hand, if p is larger, more next hops are elected from the further list. Although the privacy protection is stronger, the routing paths

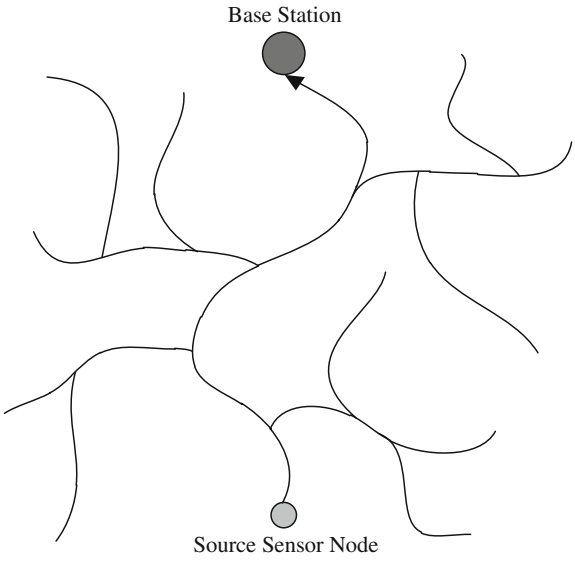

Fig. 7 The real and fake communication traffic trigged by a data packet transmission using the fractal propagation technique with the multi-parent routing and random walk techniques

are longer and the power efficiency is lower. To further protect the receiver-location privacy, fake packets can also be injected with this technique to smooth out the traffic. Whenever a sensor node forwards a packet, it also sends a fake packet to a randomly chosen neighbor in the further list, and the fake packet will be further forwarded away by h hops ($h \geq 2$). Experimental results show that this technique with fake packet injection deliveries more packets than the phantom routing technique [32] and the fractal propagation technique [16], when $p \geq 0.4$.

3.2.5 Discussion

In general, the techniques designed for protecting the base-station- or receiver-location privacy are based on random forwarding and fake packet injection. Some of these techniques (i.e., [16, 30]) require that each sensor node knows its hop distance from a sink through a broadcast message from the sink. An intelligent fake packet injection technique has been designed to protect the base-station privacy during such a topology discovery period [36]. Most of existing techniques for protecting base station privacy only consider stationary base stations. It is interesting to consider a scenario where a base station is able to move to sensor nodes to collect data. There are only a few attempts to tackle this scenario. For example, data forwarded to random nodes and mobile base stations move in the network following some random paths to collect data from sensor nodes [43]. Actually, the mobility of the base station can enhance the privacy of the base station itself [1]. Specifically, the base station relocates itself within the WSN periodically, which obfuscates previous traffic analysis conducted by the adversary. Thus, the privacy of the base station is protected. Nev-

ertheless, more research efforts are needed to tackle the privacy issue of mobile base stations with stationary sensor nodes or even with mobile sensor nodes.

4 Data Privacy

Wireless sensor nodes are usually deployed to monitor surroundings by providing a variety of readings. In many WSN applications, sensor readings may be sensitive information, e.g., a log of identifications of border-crossing vehicles. Such readings must be protected from malicious attacks. This section describes existing privacy-preserving techniques for two main types of data privacy: *data content privacy* and *aggregate data privacy*.

4.1 Data Content Privacy

Many research efforts have mainly focused on how to design encryption and authentication mechanisms for WSNs that consider the computational and power constraints of wireless sensor nodes. Since these techniques are more related to security issues in WSNs, we briefly highlight two well-known suites of security protocols. Interested readers are referred to two survey papers [8, 50].

1. **SPINS** [47]. SPINS consists of two components, a secure network encryption protocol (SNEP) and a micro version of timed efficient streaming loss-tolerant authentication protocol (μTESLA). SNEP provides secure channels between a senor node and a base station for data confidentiality, data authentication for two-party communication, data integrity through data authentication, and guarantee for data freshness (i.e., no adversary replays old data). μTESLA provides authenticated broadcast communication.
2. **Localized Encryption and Authentication Protocol (LEAP)** [60]. LEAP supports the management of four types of keys to provide different security requirements for different kinds of data communications on sensor nodes. (1) Every sensor node has a unique *individual key* that is shared with the base station for their secure communication; (2) the sensor nodes in the same group have a shared *group key* for building a secure broadcast channel from the base station to the whole group; (3) every sensor node shares a *cluster key* with its neighbors for securing local broadcast messages; and (4) every node also shares a *pairwise shared key* with each of its neighbor nodes for secure communication with authentication. In addition, LEAP provides an efficient protocol for establishing and updating these keys, as well as an authentication mechanism for them.

There are two key limitations of security protocols in WSNs [44, 53]. (1) *Constrained resources*. Since sensor nodes usually have limited battery and computational power, only simple and fast methods of cryptography can be used in WSNs.

Wireless communication is a major cause of power consumption, so unnecessary information exchange should be avoided. (2) *Unsecure key storage.* Since sensor nodes are usually deployed in an open area, they have no secure storage for their secret keys. Recent results have shown that pairing-based cryptography (PBC) is suitable for constrained sensors [44, 53]. Based on PBC, authenticated identity-based non-interactive security protocols can be designed for WSNs. Experimental results have shown that PBC is feasible on 8-, 16-, and 32-bit sensor processors and various types of sensor platforms, e.g., MICA2/MICAz, TelosB/Tmote Sky, and Imote2. Therefore, it is important for the sensor network security community to restudy how PBC can be used to enhance the existing suites of security protocols.

4.2 Aggregate Data Privacy

One of the important functions of WSN applications is the support for in-network data processing, which means sensor nodes can collaborate with each other to provide services or answer queries without a centralized server. End-to-end encryption and authentication techniques are not applicable to in-network data processing because intermediate nodes cannot access any by-passing data. In practice, many WSN applications only need to provide aggregate statistics such as SUM, AVERAGE, MIN, or MAX of sensor readings in a certain region or within a certain time period [40]. These applications can employ in-network data aggregation to reduce the amount of raw sensor readings to be reported, so sensor and network resources can be saved. Data aggregation techniques often assume that all sensor nodes in the WSN are trustworthy [49]. However, this assumption may not be realistic because sensor nodes can be compromised by the adversary who wants to steal sensor readings. If the raw sensor readings are passing through and being aggregated on these compromised nodes, this would cause privacy leakage.

Before presenting privacy-preserving data aggregation techniques, we summarize their desired characteristics as follows:

1. **Privacy.** The data generated by a sensor node should be only known to itself. Furthermore, a privacy-preserving data aggregation technique should be able to handle attacks and collusion among compromised nodes, since it is possible that some nodes may collude to uncover the private data of other nodes.
2. **Efficiency.** Data aggregation reduces the amount of traffic in a WSN, thus saving bandwidth and power usage. However, a privacy-preserving data aggregation technique introduces additional computational and communication overhead to sensor nodes. A good technique should minimize such kinds of overhead.
3. **Accuracy.** A privacy-preserving data aggregation technique should not reduce the accuracy of aggregate values.

Different WSN applications make different trade-offs among these performance metrics. The rest of this section discuss four privacy-preserving schemes for data aggregation, namely, cluster-based private data aggregation (CPDA), slice-mix-

aggregate (SMART), secret perturbation, and k-indistinguishable privacy-preserving data aggregation (KIPDA).

4.2.1 CPDA: Cluster-Based Private Data Aggregation

CPDA [25] is designed to support privacy-preserving SUM aggregation in WSNs. In CPDA, sensor nodes are randomly grouped into clusters. For each cluster, algebraic properties of polynomials are used to calculate an aggregate SUM. CPDA guarantees that the data of individual node is not exposed to other nodes. Finally, the intermediate aggregate values in each cluster are further aggregated along a routing path to a base station. In general, CPDA consists of three main steps:

1. **Formation of clusters.** The first step in CPDA is to construct clusters to perform intermediate aggregations. Every cluster consists of one cluster head (CH) and many cluster members. The CH is responsible for calculating intermediate aggregations and reporting their results to a base station. Figure 8 depicts a cluster with three sensor nodes s_0, s_1, and s_2, where s_0 is the CH.
2. **Intermediate aggregation.** This step is based on a random key distribution mechanism proposed in [18]. Consider a cluster with one head and n members, where s_0 is the CH and $s_1, \ldots, s_n$ are other sensor nodes in the cluster. Each node s_i $(0 \leq i \leq n)$ sends a seed A_i to other members in its cluster. As depicted in Fig. 8a, s_0 sends A_0 to s_1 and s_2, s_1 sends A_1 to s_0 and s_2, and s_2 sends A_2 to s_0 and s_1. For each node s_j $(0 \leq j \leq n)$, each node s_i $(0 \leq i \leq n)$ perturbs its own private reading into V_j^i based on s_j's seed and n random numbers generated by s_i, and it then sends $E(V_j^i)$ in an encrypted form to s_j, where $i \neq j$. As illustrated in Fig. 8b, s_0 sends $E(V_1^0)$ and $E(V_2^0)$ to s_1 and s_2, respectively; s_1 sends $E(V_0^1)$ and $E(V_2^1)$ to s_0 and s_2, respectively; and s_2 sends $E(V_0^2)$ and $E(V_1^2)$ to s_0 and s_1, respectively. Each node s_j $(1 \leq j \leq n)$, except the CH, next adds all received V_j^i $(0 \leq i \leq n)$ and its V_j^j together to compute a sum F_j, and then send F_j to its CH. In the running example, s_1 and s_2 send F_1 and F_2 to the CH s_0, respectively. Finally, after the CH receives F_j from each member s_j, it is

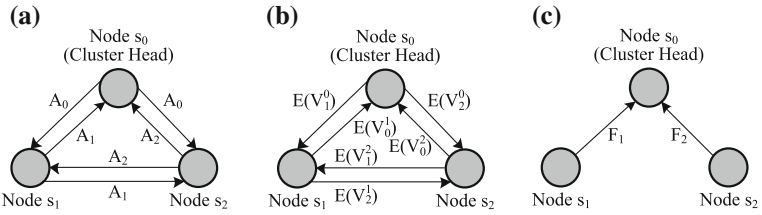

Fig. 8 An example of CPDA within a cluster of three sensor nodes. **a** Broadcast seeds. **b** Send encrypted perturbed values. **c** Send assembled values to CH

able to compute the sum of the original readings from all the nodes in the cluster without compromising the privacy of individual nodes' data value.

3. **Cluster data aggregation.** The cluster head reports its intermediate aggregate sum to the base station. The base station computes an aggregate SUM by summing up all collected intermediate aggregate sums.

CPDA guarantees that if less than $(n-1)$ nodes collude in a cluster of size n, the individual sensor readings in the cluster cannot be disclosed. Therefore, larger average cluster size contributes to better privacy-preservation performance, but it also incurs more computational overhead to compute the intermediate aggregation value. As a result, there is a design tradeoff between the privacy protection and computation efficiency.

Although CPDA can provide private data aggregation, it cannot guarantee data integrity. If an adversary changes the intermediate aggregate result in some clusters, the aggregate result would deviate from the actual one dramatically [26]. CPDA has been extended to iCPDA [27] that can guarantee data integrity through additional piggybacks. In iCPDA, every node in a cluster collects necessary information to calculate an intermediate aggregate result within the cluster. Hence, all the nodes in the cluster can figure out the intermediate aggregated value in the cluster, enabling them to monitor their CH and detect data pollution attacks. Experimental results showed that the communication overhead of iCPDA is a little bit higher than CPDA due to the extra message exchange.

4.2.2 Slice-Mix-Aggregate (SMART)

The basic idea of SMART [25] is to slice readings and use the associative property of addition to compute aggregate SUM. In general, SMART consists of three main steps:

1. **Slicing readings.** Suppose a WSN has N sensor nodes. Each sensor node s_i randomly selects a set of m peers within a certain hop distance. After s_i gets a private reading d_i, d_i is sliced into m pieces. s_i randomly keeps one piece, and then each of the remaining $m-1$ pieces is randomly sent in an encrypted form to a distinct one of the selected m peers. Let d_{ij} be a piece of d_i that is sent from s_i to another sensor node s_j, and hence, $d_i = \sum_{j=1}^{N} d_{ij}$, where $d_{ij} = 0$ if s_j does not receive any piece of d_i.
2. **Mixing slices.** When a sensor node s_j receives k encrypted pieces within a certain time interval, s_j decrypts each piece and sums up all received pieces to compute a mixed value $r_j = \sum_{i=1}^{N} d_{ij}$, where $d_{ij} = 0$ if s_i is not one of the senders of the k pieces. Then, s_j sends r_j to the query server.
3. **Aggregation.** After the query server receives the mixed values from all the sensor nodes within a certain time interval, it sums them up to get the final result $\sum_{j=1}^{N} r_j = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} = \sum_{i=1}^{N} d_i$, where $r_j = 0$ is the query server does not receive any mixed value from node s_j and $d_i = 0$ if node s_i does not report any reading within the time interval.

Unlike CPDA, SMART does not need to form any clusters, so it can reduce the computational cost of cluster-wise data aggregation.

4.2.3 Secret Perturbation

The basic idea of secret perturbation [20] is that each sensor node s_i has a pre-assigned secret S_i, which is only known to s_i and the base station. s_i does not send its original reading d_i to the base station. Instead, s_i only sends a perturbed version of d_i, i.e., $\widehat{d_i} = d_i + S_i$, to the base station. Each intermediate node between s_i and the base station receiving a perturbed value can perform an additive aggregation function on it with other perturbed values sent from other nodes to get a single perturbed value. Since the base station knows the secret of each sensor node, it can subtract the perturbations from a perturbed value to get an actual reading. In general, the *basic secret perturbation scheme* consists of three main steps to compute aggregate SUM:

1. **System initialization.** Given N sensor nodes and the range of each sensor reading $[0, d_{max}]$, the base station selects an integer L, a prime number q and a secure hash function $hash(x)$ such that $max\{2^{L-1}, 2N\} < q < 2^L$, $d_{max} < 2^L$ and $0 \leq hash(x) < 2^L$. It also assigns each sensor node s_i with two secret numbers.
2. **Perturbed aggregation.** When a sensor node s_i receives a query, it gets a reading d_i, uses $hash(x)$ to calculate a perturbed reading value $\hat{d}_i$ and an auxiliary reading value $\hat{A}_i$, and initializes a list of IDs of reporting sensor nodes $list_i = \{i\}$. If s_i is a leaf node or it has no downstream node that reports data, s_i simply sends $\langle \hat{d}_i, \hat{A}_i, list_i \rangle$. On the other hand, if the other node s_j receives $\langle \hat{d}_{i_k}, \hat{A}_{i_k}, list_{i_k} \rangle$ from its downstream nodes, where $k = 0, 1, \ldots, m$ and $m < N$, it computes its own $\hat{d}_j$, $\hat{A}_j$ and $list_j$, and then performs an additive aggregation $\langle \hat{d}_j, \hat{A}_j, list_j \rangle$ on all the received $\langle \hat{d}_{i_k}, \hat{A}_{i_k}, list_{i_k} \rangle$.
3. **Retrieving the original aggregation at a base station.** After the base station s_0 receives the perturbed tuples $\langle \hat{d}_{i_k}, \hat{A}_{i_k}, list_{i_k} \rangle$ from its downstream nodes, where $k = 0, 1, \ldots, m$, $m \leq N$, and the ID of the base station is 0, s_0 computes its $\hat{d}_0$, $\hat{A}_0$ and $list_0$. s_o is then able to find the sum of its reading and the original readings of all the received tuples.

Taiming Feng et al. have extended the *basic secret perturbation scheme* to the *fully-reporting secret perturbation scheme* [20] that does not require sensor nodes to report their IDs by requesting every sensor node to report an actual or dummy reading in the perturbed form. They further proposed the *adaptive secret perturbation scheme* to minimize communication overhead and avoid reporting node IDs.

4.2.4 k-Indistinguishable Privacy-Preserving Data Aggregation (KIPDA)

The KIPDA scheme [21] is designed for MAX/MIN aggregation in WSNs based on the concept of k-anonymity [52]. The basic idea of KIPDA is that each sensor

node reports its actual reading along with $k - 1$ other restricted or unrestricted camouflage values such that the actual reading is indistinguishable among the k values. A base station is able to find an exact aggregate MAX/MIN result based on k-indistinguishable data reported from sensor nodes. We present the four main steps in KIPDA with a running example with $k = 7$ for aggregate MAX, as depicted in Fig. 9.

1. **System setup.** The base station decides *a set of global real value positions* for a vector with k elements. Each position is assigned to each sensor node to indicate which position in a reported vector should store its actual reading. For each sensor node, the remaining unassigned positions are divided into two disjoint sets: (1) a set of unrestricted positions, where a random camouflage number is generated for each position, and (2) a set of restricted positions, where a camouflage value that is required to be less than or equal to an actual reading for aggregate MAX while a camouflage value that is required to be larger than or equal to an actual reading for aggregate MIN. In our example (Fig. 9), we assume that k is seven, the size of each set of unrestricted positions is two, and the size of each set of restricted positions is four.
2. **Filling camouflage values.** After a sensor node receives a query, it puts its actual reading at its real value position assigned by the base station in a vector. Then, it generates a camouflage value at every unrestricted or restricted position in the vector. Finally, it sends the vector to the base station through its parent node. In the running example, sensor node 1 puts is actual reading (i.e., 23) at the first position in a vector (indicated by an underline) and generates two unrestricted camouflage values 30 and 21 that are put at the fourth and sixth positions in the vector (indicated by shaded cells), respectively, as Fig. 9. Since the query computes aggregate MAX, node 1 generates four camouflage values that are less than the actual reading (i.e., 23). In this example, node 1 puts 18, 15, 20, and 8 at

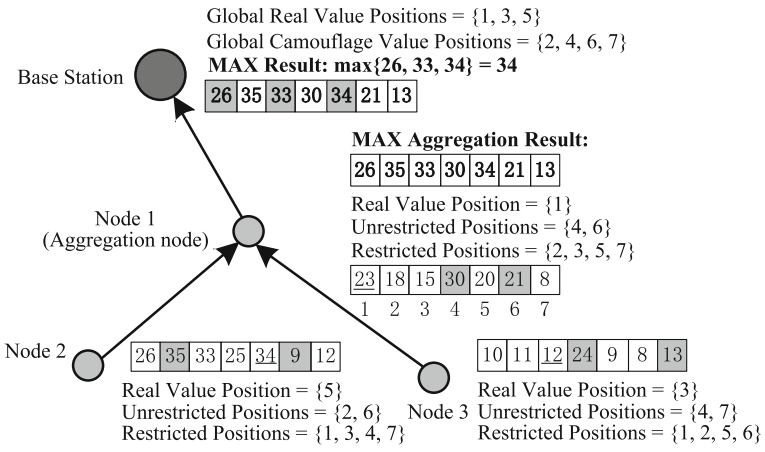

Fig. 9 An example of processing aggregate MAX using KIPDA with $k = 7$

the second, third, fifth, and seventh positions in the vector, respectively. Similarly, nodes 2 and 3 generate their vectors. Since node 1 is the parent node of nodes 2 and 3, nodes 2 and 3 send their vectors to node 1.

3. **Aggregation at an intermediate node.** After a sensor node receives vectors from its children nodes, it computes an intermediate aggregation result vector. For each position in the result vector, the node selects the maximum value at the same position among all the received vectors. The node next sends the result vector to its parent node. For example (Fig. 9), node 1 receives the vectors from nodes 2 and 3. The value at the first position of the result vector is $\max\{23, 26, 10\} = 26$. Similarly, the value at the second position of the result vector is $\max\{18, 35, 11\} = 35$ and so on. After node 1 computes the intermediate result vector $\langle 26, 35, 33, 30, 34, 21, 13\rangle$, the vector is sent to the base station through its parent node.
4. **Retrieving the original aggregation at a base station.** After the base station receives an aggregate vector, it selects the maximum value among the values at the global real value positions in the vector as the query answer. For example (Fig. 9), the base station receives an aggregate vector $\langle \underline{26}, 35, \underline{33}, 30, \underline{34}, 21, 13\rangle$ from node 1. The base station selects the maximum values at the first, third and fifth positions in the vector. Thus, the final aggregate MAX is $\max\{26, 33, 34\} = 34$.

Since the aggregate process of KIPDA does not require end-to-end or link-level encryption, it achieves higher efficiency than the encryption-based data aggregation technique in terms of power consumption and latency. In terms of robustness, KIPDA can tolerate up to a large number of compromised sensor nodes or communication link. However, it is not easy to apply KIPDA to other aggregation functions other than MAX and MIN.

4.2.5 Discussion

All the private data aggregation techniques discussed in this section can provide precise aggregate results if there is no packet loss. In terms of privacy protection, the secret perturbation technique always prevents the adversary from finding out an individual sensor's data or an aggregate result, regardless of the number of compromised sensor nodes. Contrarily, CPDA, SMART and KIPDA can only tolerate up to a certain threshold number of compromised sensor nodes or communication link. In terms of efficiency, experimental results [20] showed that the secret perturbation technique consumes less bandwidth than CPDA and SMART. KIPDA is more efficient than conventional encryption-based data aggregation techniques which incur high latency due to the decryption and re-encryption operations. Finally, CPDA, SMART and secret perturbation techniques are specially designed for aggregate SUM, while KIPDA is tailored for aggregate MAX/MIN. A WSN application should select the best private data aggregation technique based on its requirements for privacy protection and system efficiency.

5 Context Privacy

In this section, we focus on two kinds of context detected by sensor nodes, namely, *location privacy* and *temporal privacy*, in Sects. 5.1 and 5.2, respectively. Location privacy protection techniques are designed to anonymize the location information of people monitored by sensor nodes. Privacy-enhancing techniques for temporal privacy are designed to hide the time when a query is issued by a user or an event is detected by a sensor node.

5.1 Location Privacy

Sensor-based location systems have been proposed to support indoor positioning and monitoring, e.g., [24, 48, 55]. Such indoor monitoring systems can provide many services: (1) *Location-based queries.* They can answer queries like "how many customers on the second floor" and "which shop is the densest one during lunch hours." (2) *Security and control.* The system alerts the administrative staff when it detects that someone enters an office at midnight or the number of people in a room is larger than a system-specified limit. (3) *Resource management.* When the system has detected no people in an certain area for a certain period of time, it turns off some building facilities (e.g., lights, escalators and elevators) to save energy. However, similar to GPS, sensor-based location systems would threaten the user privacy. For example, if an adversary knows the location of a person's office, the adversary can easily determine whether the person is in his or her office by monitoring the sensing information, e.g., the number of people in a sensing area, reported from the sensor deployed in his or her office. Once the adversary identifies an individual's location, the adversary can track the user's movements by monitoring location updates from other sensor nodes [11, 22].

Cricket [48] is a privacy-aware sensor-based location system. Cricket has two strategies to protect the user's location privacy: (1) It deploys sensors in a system area. Every user wears a tag that receives signals from multiple sensors to detect the user's location. This decentralized positioning approach does not require any centralized processing on user location information or store user locations. (2) If a user concerns about location privacy, he or she can turn off the tag, and thus, no monitoring system can know the user's location. However, Cricket [48] is not suitable for location monitoring systems. The main reason is that if many people do not report location information to a location monitoring system, the system cannot provide any meaningful services.

Note that location privacy is different from data source privacy described in Sect. 3.1. Location privacy protection techniques aim at protecting the privacy of individuals' location information collected by source sensor nodes, while the objective of data source privacy protection techniques is to protect the privacy of the location of source sensor nodes themselves. We discuss four privacy enhancing tech-

niques for indoor sensor-based location monitoring systems, and then describe how a privacy-preserving monitoring system provides location-based services without compromising the user location privacy.

5.1.1 Pseudonyms

To protect the privacy of individuals' location information collected by sensors while taking advantage of location-based services, users' true identity should be hidden from the applications receiving their location information. An even-driven middleware has been designed to act as a proxy server between the user and the application to help the user hide his/her real identity [4]. As depicted in Fig. 10, after a user registers his/her interest in a particular location-based service (LBS) with the middleware, the LBS provider receives event callbacks from the middleware when the user enters or exits a certain system-specified area. For example, a shopping mall application is configured to enable an LBS "sending e-coupons to users entering the shopping mall." This application should register certain areas in front of the shopping mall's entrances and wait for the event callback. When a registered user enters the application's registered area, the application sends relevant e-coupons to the user. Since the users' real identity can be anonymized by the middleware, they can enjoy LBS without revealing their real identities. However, it is still risky for a user to use a long-term pseudonym, even if different LBS providers give out different pseudonyms to the same user to avoid collusion. This is because an adversary could identify a user by following the "footsteps" of a pseudonym to or from some places which are strongly associated with the user's real identity (e.g., a residential house). One possible solution is to frequently change a user's pseudonym, but it may significantly degrade the quality of LBS. We discuss a mix-zone approach that can balance between the user location privacy and the quality of services.

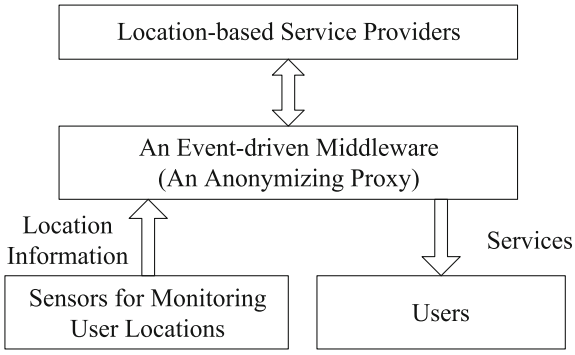

Fig. 10 The middleware model for pseudonyms

5.1.2 Mix Zones

The idea of *mix zones* [4] is derived from the concept of *mix nodes* designed for anonymous communication systems [6]. A *mix zone* is defined as a spatial region with a system-specified maximum number of users who have not registered with any application callback, while an *application zone* is defined as an area where LBS applications could register for event callbacks. An example is depicted in Fig. 11, where there is one mix zone, which is represented by a shaded area, and three application zones, which are represented by white rectangles, an art gallery (A), a book store (B), and a coffee shop (C). Let's use this simple example to illustrate the basic idea of mix zones. Suppose that two users Alice and Bob enter the mix zone from B at the same time, and their identities are mixed; after a certain period, two users exit from the mix zone and appear in C. We cannot infer that these users are Alice and/or Bob or other users located in the mix zone before Alice and Bob entering it. However, if a mix zone has a diameter much larger than the distance the user can reach during one location update period, it might not be able to protect users adequately. For the same example depicted in Fig. 11, A is much closer to B than C. If two users enter the mix zone from A and C at the same time and a user appears in B at the next location update time, an adversary may tell that the user entering B from the mix zone is not the one who emerged at C before. Furthermore, if there is nobody in the mix zone at this time, the user in B can only be the one from A, thus revealing A's identity. To address this privacy issue, two metrics are proposed to measure the level of privacy protection, namely, *anonymity set* and *entropy*. A user can specify the minimum size of his/her anonymity set that is the number of people visiting the mix zone during the same location update period. The user is not willing to reveal his/her location information to any application until the mix zone finds a qualified anonymity set. The entropy is used to measure the level of uncertainty that a hostile adversary knows about a user's location information based on the user's historical movement data.

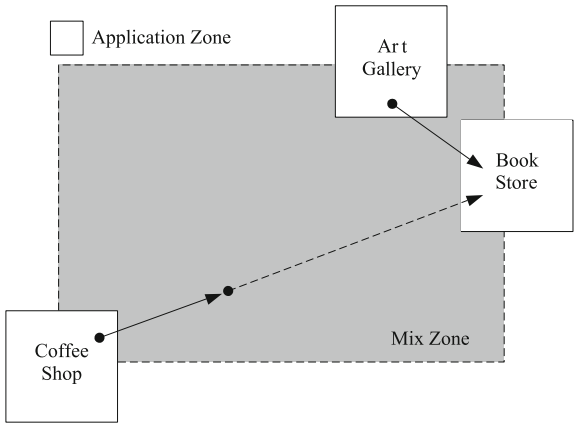

Fig. 11 A sample mix zone for three application zones

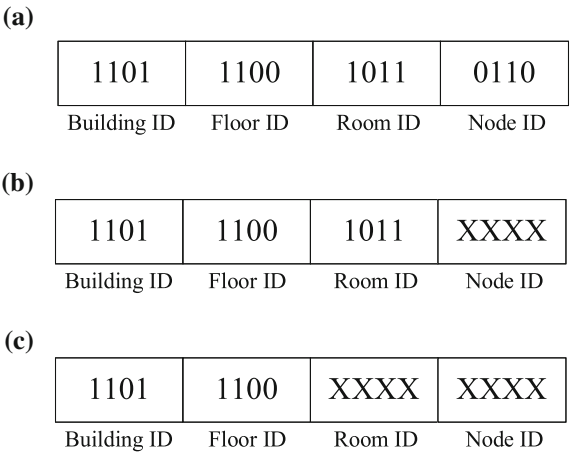

Fig. 12 Examples of node ID cloaking in the hierarchical location perturbation algorithm. **a** Sensor node detected at least k subjects. **b** At least k subjects can only be found in the room level. **c** At least k subjects can only be found in the room level

5.1.3 Hierarchical Location Perturbation

Sensors could be deployed inside buildings to track the locations of individuals, which is used for adaptive computing services. However, the location privacy of individuals may be breached due to the vulnerability of WSN. The basic idea of the hierarchical location perturbation algorithm [22] is to provide less spatial accuracy and perturb the count of subjects in a monitored area. In general, the algorithm consists of two main steps:

Step 1: Defining a hierarchical structure. The system area is partitioned into several physical hierarchies, e.g., rooms, floor, and building. Each sensor node is assigned with a unique hierarchical ID. Figure 12 depicts an example of a hierarchical structure, where the highest level is a building, the second level contains all the floors in the building, the third level contains all the rooms on each floor, and the lowest level contains all the sensor nodes in each room. If each node in the hierarchical structure does not have more than 16 child nodes, the ID of each level can be represented by four bits, as depicted in Fig. 12a.

In each hierarchy, a sensor node is selected as a leader by using a distributed leader election protocol. Each leader keeps track the number of subjects monitored in its corresponding hierarchy. Also, each node knows a system-specified anonymity level k, i.e., the monitoring system can only receive the subject count information of a monitored area containing at least k subjects.

Step 2: Location perturbation. When a sensor node detects the number of subjects that is equal to or larger than k, it cloaks the number of subjects by randomly rounding up or down the subject count to the nearest multiple of k, and sends an exact node ID to the leader in the upper level. Otherwise, the sensor sends the exact number of detected subjects with a cloaked node ID to the leader in the upper level. The leader repeats this step until at least k subjects in a higher level can be found. In our example, if a sensor node can detect at least k subjects, it can report its exact

ID with a cloaked subject count to the leader in the upper level (i.e., the room level) (Fig. 12a). Otherwise, it sends the exact subject count with its cloaked ID "XXXX" to the leader in the room level. Figure 12b shows the case that the leader in the room level can detect at least k subjects. If this is not the case, it further sends the exact sum of the subject counts of its child nodes with its cloaked ID to the leader in the floor level. Figure 12c shows that the leader in the floor level can find at least k subjects, so it sends its exact ID with cloaked room and node IDs and the cloaked subject count to the leader in the building level.

5.1.4 Spatial Cloaking

TinyCasper [9, 11] is a privacy-preserving location monitoring system designed for WSNs. The basic idea of TinyCasper is that sensor nodes can communicate with each other and work together to find k-anonymized aggregate locations that are reported to a server. A k-anonymized aggregate location is defined as (A, N), where A a monitored area that contains at least k people and N is the number of people detected inside A. Figure 13a depicts an example where 17 sensors are deployed in the system, a nonzero number beside a sensor node indicates the number of people detected inside its sensing area, the location of six people are represented by circles, and $k = 3$. After the sensor nodes communicate with other peers, every node with a nonzero people count blurs its sensing area into a cloaked area that contains at least $k = 3$ people. Then, the sensor node reports the cloaked area A along with the number of people N located in A as an aggregate location, i.e., (A, N), to the server. In our example, sensor nodes s_1, s_5, s_6, s_7, and s_{15} detect a nonzero people count, they communicate with nearby peers to blur their sensing area. For example, s_7 needs to communicate with nodes s_5 and s_6 to find three people, and then it blurs its sensing area into a cloaked area that contains the sensing areas of nodes s_5, s_6 and s_7 (represented by the left-bottom shaded rectangle in Fig. 13b. s_7 reports the cloaked area with three people as an aggregate location to the server. In this example, the server receives two three-anonymized aggregate locations (represented by the two shaded rectangles) (Fig. 13b).

Two in-networks spatial cloaking algorithms have been proposed for TinyCasper [9, 11]. (1) The *resource-aware* algorithm employs a greedy approach for sensor nodes to determine their aggregate locations. Its objective is to minimize the communication and computational overhead. (2) The *quality-aware* algorithm aims at enabling sensor nodes to determine their aggregate locations with the minimal cloaked area, in order to maximize the usability of the aggregate locations that are used by the server to provide location-based monitoring services. Spatial cloaking techniques have also been extended to support mobile devices, e.g., smartphones and mobile sensors, through peer-to-peer communication [12, 13, 42].

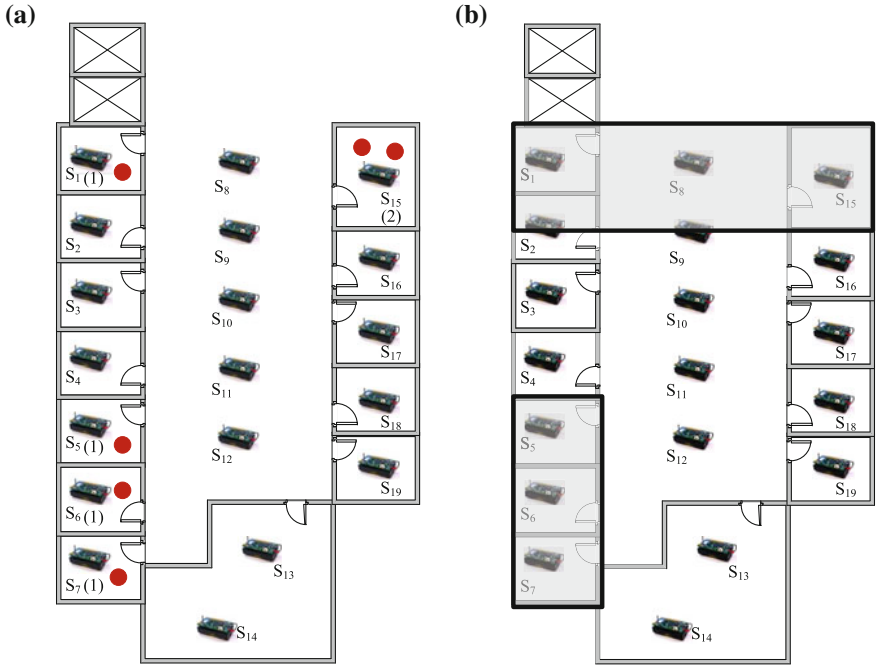

Fig. 13 Example of spatial cloaking in a sensor-based location monitoring system ($k = 3$). **a** Sensor deployment. **b** Two k-anonymized aggregate locations

5.1.5 Discussion

Using pseudonyms may not be secure because an adversary could link the old and new pseudonyms to infer a user's read identity. The effectiveness of the mix zone greatly depends on its size, the user population, the sensing resolution, and the user movement speed. The hierarchical location perturbation algorithm does not consider sensor nodes deployed at the same level in the predefined hierarchy, e.g., the sensor nodes in the same room. When the subject count of a sensor node does not satisfy a certain number, the algorithm goes up to its parent. The algorithm trends to generate cloaked areas larger than necessary, such large aggregate locations would degrade their usability in location monitoring services. On the other hand, the spatial cloaking algorithms do not reply on any hierarchical structure and they can support both indoor and open environments. In addition, since they aim at maximizing the usability of their cloaked areas by minimizing their area, they can be used to provide better location monitoring services. We discuss how a WSN can provide location monitoring services based on cloaked areas in next section.

5.1.6 Privacy-Preserving Location Monitoring Services

As discussed in Sects 5.1.3 and 5.1.4, the privacy-preserving monitoring system can only report aggregate locations for the server to provide monitoring services. To this end, a privacy-preserving aggregate query processor is designed to employ a spatio-temporal histogram to estimate the subject distribution based on aggregate locations reported from sensor nodes [10]. The spatio-temporal histogram partitions the system area into disjoint equal-sized cells that is stored as a two-dimensional array. Each element $M[i, j]$ is an estimator that represents the estimated number of subjects located in its corresponding cell area, where i and j indicate the i-th row and j-th column in the array, respectively. In general, the privacy-preserving aggregate query processing has two main steps.

Step 1: Histogram updates. The server is able to detect the total number of people N_{total} in the system area. Initially, N_{total} is uniformly distributed among all estimators in the histogram. When the server receives an aggregate location (A, N), the basic approach calculates the sum S of the estimators of the cells intersecting the aggregate location area A and uniformly distributes the reported people count N among the cells intersecting A. The difference between S and N is uniformly distributed among the cells outside A. Several optimization techniques have been proposed for an adaptive spatio-temporal histogram, in which the server updates the histogram based on various spatial, temporal, and historical factors.

Figure 14 depicts an example of a basic histogram where the system area is divided into 25 cells and $N_{total} = 100$. Initially, N_{total} is uniformly distributed among the 25 cells, so the value of each estimator is 4 (Fig. 14a). Figure 14b shows an aggregate location with $N = 20$ people and its cloaked area A is represented by a bold rectangle. The sum of the four estimators of the four cells intersecting A is $4 \times 4 = 16$. Since 20 is uniformly distributed among the four cells intersecting A, each estimator in these four cells is set to $20/4 = 5$. The difference $16 - 20 = -4$ is evenly distributed

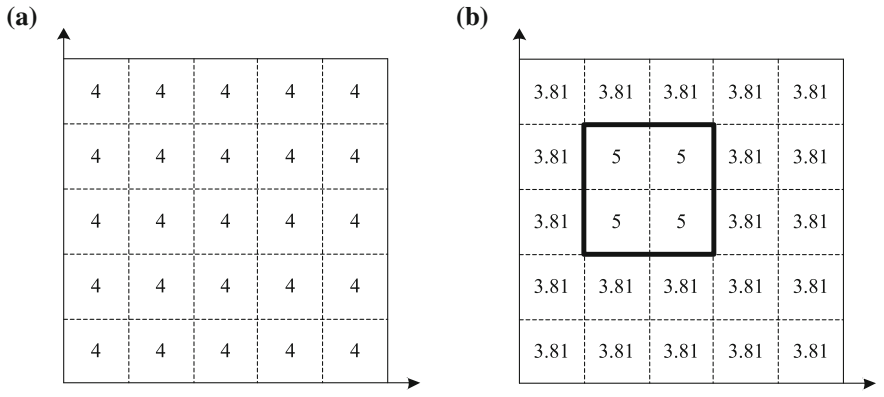

Fig. 14 An update on the basic spatio-temporal histogram. **a** Initial histogram. **b** An aggregate location with 20 people

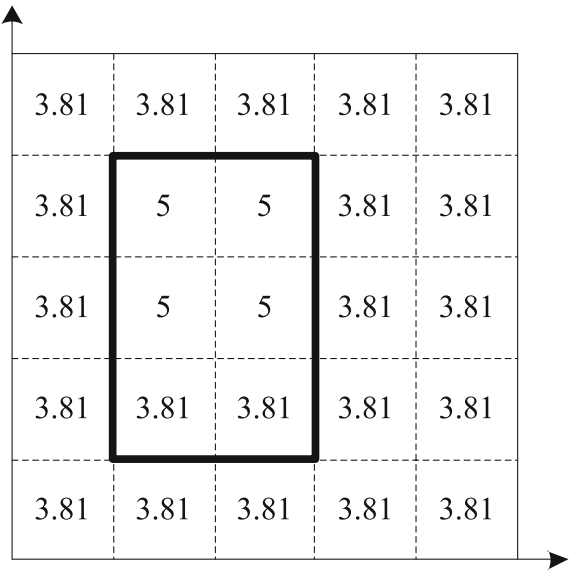

Fig. 15 Privacy-preserving aggregate query processing

among the estimators of the cells outside A, so each of those estimators is set to $4 + (-4/21) = 3.81$.

Step 2: Aggregate query processing. The query processor is able to answer aggregate queries. Given an aggregate query with a query range, the query processor calculates the sum of the estimators of the cells intersecting the query range and returns it as a query answer. Figure 15 depicts an example of an aggregate query with a spatial query range which is represented by a bold rectangle. The query answer is the sum of the estimators of the cells intersecting the spatial query range, i.e., $5 \times 4 + 3.81 \times 2 = 27.62$.

Recently, a stochastic flow model [54] has been designed to provide location monitoring services in an indoor environment based on the sensor node's actual subject count. Since the flow model considers the network topology, it may provide more accurate location monitoring services. It would be very useful to apply the flow model to TinyCasper.

5.2 Temporal Privacy

In this section, we discuss two privacy-preserving techniques for temporal privacy in WSNs, namely, *adaptive delaying* [31] and *probabilistic sampling* [23]. The adaptive delaying technique is mainly designed for protecting the event time or the packet transmission time, while the probabilistic sampling technique aims at protecting the user's query patterns and the unusual event time.

5.2.1 Adaptive Delaying

In some WSN applications, the time when an event is detected or data is transmitted by a sensor node is considered as sensitive information. An adversary would be able to infer temporal information by merely capturing the arrival time of data packets at some sensor nodes. Packet delaying techniques have been applied to anonymous network communication. Most of these early techniques rely on the concept of pool mixes [14], where the pool mix waits until it buffers a certain number of packets before taking mixing action to protect their anonymity. Thus, the pool mix delays packets before forwarding them. The concept of pool mixes has been extended to delay individual incoming packets according a probabilistic distribution before sending them out [17, 34].

P. Kamat et al. [31] have further used packet delaying to protect the temporal privacy of detected events and transmitted packets. Suppose a source node S detects an event and generates a packet at some time X and sends the packet in an encrypted form to a destination node R, and a compromised node E monitors traffic arriving at R. The adaptive delaying technique can protect temporal privacy in three network scenarios:

1. **Two-party single-packet network.** S obfuscates the time X by storing the packet in its local buffer for a random time period Y before transmitting it to R. E witnesses the packet arrives at a time $Z = X + Y$, while R can decrypt the packet to know the actual event detection or packet generation time X.
2. **Two-party multiple-packet network.** Suppose S generates a stream of n packets $p_1, p_2, \ldots, p_n$ at times $X_1, X_2, \ldots, X_n$ and delays them by $Y_1, Y_2, \ldots, Y_n$, respectively, before sending them to R. E observes the packets at $Z_1, Z_2, \ldots, Z_n$, where $Z_i = X_i + Y_i$ $(1 \leq i \leq n)$. Similar to the first scenario, R can decrypt each packet and know its actual event detection or packet generation time. If the system needs to maintain packet ordering, Y_j has to be at least the time until all previous packets $p_1, p_2, \ldots, p_i$ (where $1 \leq i < j \leq n$) were transmitted. Thus, there is an ordering of $Z_1 < Z_2 < \ldots < Z_n$. Otherwise, Y_j can be independent of each other and independent of the event detection or packet generation times.
3. **Multihop network.** Suppose S sends a stream of packets to R through m intermediate nodes $S \rightarrow N_1 \rightarrow N_2 \rightarrow \ldots \rightarrow N_m \rightarrow R$. The delay of a packet p_i is $Y_i = Y_{0,i} + Y_{1,i} + \ldots + Y_{m,i}$, where $Y_{k,i}$ denotes the delay time of i-th packet at k-th intermediate node and $Y_{0,i}$ denotes the delay time of i-th packet at S. Consequently, E observes the arrival of each packet p_i at $Z_i = X_i + Y_i$.

Delaying packets protects temporal privacy, but it increases burden on the buffer at an intermediate node, especially the nodes close to the base station. When a buffer is full, a replacement strategy is needed to replace a victim packet in the buffer for a newly received packet. The victim packet is transmitted immediately instead of dropping it. Four replacement strategies have been designed to choose a victim packet from a buffer as follows:

1. **Longest delayed first (LDF).** The packet has been delayed in the buffer for the longest time is selected as the victim packet.

2. **Shortest delay time first (SDTF).** The packet has been delayed in the buffer for the shortest time is selected as the victim packet.
3. **Longest remaining delay first (LRDF).** The packet with the longest remaining delay time in the buffer is selected as the victim packet.
4. **Shortest remaining delay first (SRDF).** The packet with the shortest remaining delay time in the buffer is selected as the victim packet.

A rated-controlled adaptive delaying mechanism is used to adjust the delay process based on the employed replacement strategy. In the adversary model of simulations, the adversary estimates the creation time of each packet. The total estimation error for m packets is calculated as the *mean square error*, i.e., $\sum_{i=1}^{m}(x_i' - x_i)^2/m$, where x_i is the true creation time for packet i and x_i' is estimated by the adversary. The larger this value is, the better the temporal privacy is protected. At the same time, tolerable end-to-end delivery latency for each packet should be maintained. Simulation results show that LRDF is the best replacement strategy in terms of both temporal privacy and latency.

5.2.2 Probabilistic Sampling

Data can be collected from sensor nodes in an on-demand or a periodic manner. The on-demand manner is easy for an adversary to infer when a user issues a query or a sensor node detects an event. Although the periodic manner can protect the temporal privacy of queries issued by users and events detected by sensor nodes, its power consumption is very high if the data collection time interval is short. However, it may not satisfy a user-specified deadline if the time interval is long. To this end, a probabilistic sampling technique is designed [23]. The basic idea is to carefully report data at random times to blend user requests and events with the routine traffic, thus making user requests and events indistinguishable to an attacker. In this technique, the day is divided into time intervals, *TimeInterval*, and each time interval contains the same number of equal-sized time slots. Figure 16 depicts an example, where the time slot is one second and the day is divided into 86,400 time slots. Each time interval is five seconds, so it contains five time slots and the day has 17,280 time intervals. This technique has three important policies:

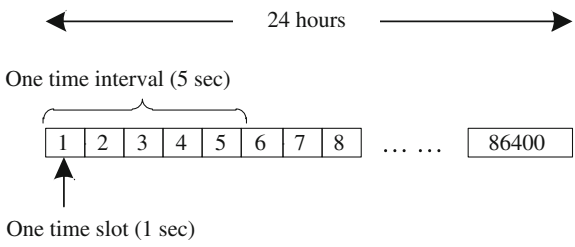

Fig. 16 Probabilistic sampling

1. A time slot is a short time period, in which at most one data report or query can transmit or issue, respectively. The base station and the sensor node have the same probability P (e.g., $1/(TimeInterval \times 2)$) of issuing a query or initiating a data report, respectively.
2. A user should only be able to issue one query every time interval.
3. When there are too many time slots between two reports from a sensor node, P should be increased until the sensor node can generate a data report. After that, P is reset to its original value.

With the careful design of the generation probability of automatic data reports, the privacy of user queries and unusual events in the network is protected. Experimental results show that the probabilistic sampling technique can effectively reduce the chance that an adversary can identify whether a user issues a query in a time interval.

5.2.3 Discussion

The adaptive delaying technique [31] does not consider any user- or system-specified deadline for a query or data collection. It would be more interesting if this technique can be extended to be time-constrained. In addition, it is important to investigate whether such a time constraint degrades its privacy protection. On the other hand, although the probabilistic sampling technique [23] considers user-specified deadlines, it is not clear how to adjust the probability P to meet the deadline and maximize the accuracy of a query answer. More sophisticated models are needed to answer these questions.

6 Conclusion and Future Directions

In this chapter, we highlighted the existing privacy enhancing technologies for wireless sensor networks (WSNs). We categorized these technologies into three main types of privacy, namely, *system privacy*, *data privacy* and *context privacy*. For each type of privacy, we presented its major privacy-preserving techniques. In the context privacy, the user location privacy is a new type of privacy in WSNs. There are three main open privacy issues in privacy-preserving location monitoring services. (1) Existing solutions only aim at protecting snapshot location privacy. It is important to study continuous location monitoring privacy to avoid tracking a target user's location by analyzing consecutive snapshots of aggregate locations reported from sensor nodes. (2) The state-of-the-art privacy-preserving aggregate query processor can only support range queries. It is essential to design more advanced query processors to support more complex analysis, e.g., data mining. (3) There is a tradeoff between user privacy and data accuracy. For example, a higher anonymity level would lead to a larger aggregate location area that degrades the user location accuracy. It would

be very interesting to derive a theoretical model to balance such a tradeoff and find an appropriate privacy protection level for a desired level of accuracy.

References

1. U. Acharya, M. Younis, Increasing base-station anonymity in wireless sensor networks. Ad Hoc Netw. **8**(8), 791–809 (2010)
2. K. Akkaya, M. Younis, A survey on routing protocols for wireless sensor networks. Ad Hoc Netw. **3**(3), 325–349 (2005)
3. J.N. Al-Karaki, A.E. Kamal, Routing techniques in wireless sensor networks: A survey. IEEE Wirel. Commun. **11**(6), 6–28 (2004)
4. A.R. Beresford, F. Stajano, Location privacy in pervasive computing. IEEE Pervasive Comput. **2**(1), 46–55 (2003)
5. B.H. Bloom, Space/time trade-offs in hash coding with allowable errors. Commun. ACM **13**(7), 422–426 (1970)
6. D.L. Chaum, Untraceable electronic mail, return addresses and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
7. C.W. Chen, Y.R. Tsai, Location privacy in unattended wireless sensor networks upon the requirement of data survivability. IEEE J. Sel. Areas Commun. **29**(7), 1480–1490 (2011)
8. X. Chen, K. Makki, K. Yen, N. Pissinou, Sensor network security: A survey. IEEE Commun. Surv. Tutor. **11**(2), 52–73 (2009)
9. C.Y. Chow, M.F. Mokbel, T. He, TinyCasper: A privacy-preserving aggregate location monitoring system in wireless sensor networks, in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2008
10. C.Y. Chow, M.F. Mokbel, T. He, Aggregate location monitoring for wireless sensor networks: A histogram-based approach, in *Proceedings of the IEEE International Conference on Mobile Data Management*, 2009
11. C.Y. Chow, M.F. Mokbel, T. He, A privacy-preserving location monitoring system for wireless sensor networks. IEEE Trans. Mob. Comput. **10**(1), 94–107 (2011)
12. C.Y. Chow, M.F. Mokbel, X. Liu, A peer-to-peer spatial cloaking algorithm for anonymous location-based services, in *Proceedings of the ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2006
13. C.Y. Chow, M.F. Mokbel, X. Liu, Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. GeoInformatica **15**(2), 351–380 (2011)
14. G. Danezis, The traffic analysis of continuous-time mixes, in *Proceedings of the International Conference on Privacy Enhancing Technologies*, 2005
15. J. Deng, R. Han, S. Mishra, Intrusion tolerance and anti-traffic analysis strategies for wireless sensor networks, in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*, 2004
16. J. Deng, R. Han, S. Mishra, Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks. Pervasive Mob. Comput. **2**(2), 159–186 (2006)
17. C. Díaz, B. Preneel, Taxonomy of mixes and dummy traffic, in *Proceedings of the International Information Security Workshops*, 2004
18. L. Eschenauer, V.D. Gligor, A key-management scheme for distributed sensor networks, in *Proceedings of the 9th ACM conference on Computer and communications, security* (2002)
19. P. Eugster, R. Guerraoui, S. Handurukande, P. Kouznetsov, A. Kermarrec, Lightweight probabilistic broadcast. ACM Trans. Comput. Syst. **21**(4), 341–374 (2003)
20. T. Feng, C. Wang, W. Zhang, L. Ruan, Confidentiality protection for distributed sensor data aggregation, in *Proceedings of the IEEE INFOCOM*, 2008
21. M.M. Groat, W. Hey, S. Forrest, KIPDA: k-indistinguishable privacy-preserving data aggregation in wireless sensor networks, in *Proceedings of the IEEE INFOCOM*, 2011

22. M. Gruteser, G. Schelle, A. Jain, R. Han, D. Grunwald, Privacy-aware location sensor networks, in *Proceedings of the USENIX International Workshop on Hot Topics in Operating Systems*, 2003
23. J. Guerreiro, E.C.H. Ngai, C. Rohner, Privacy-aware probabilistic sampling for data collection in wireless sensor networks, in *Proceedings of the International Wireless Communications and Mobile Computing Conference*, 2011
24. A. Harter, A. Hopper, P. Steggles, A. Ward, P. Webster, The anatomy of a context-aware application. Wireless Netw. **8**(2–3), 187–197 (2002)
25. W. He, X. Liu, H. Nguyen, K. Nahrstedt, T. Abdelzaher, PDA: Privacy-preserving data aggregation in wireless sensor networks, in *Proceedings of the IEEE INFOCOM*, 2007
26. W. He, X. Liu, H. Nguyen, K. Nahrstedt, T. Abdelzaher, iPDA: an integrity-protecting private data aggregation scheme for wireless sensor networks, in *Proceedings of the IEEE Military Communications Conference*, 2008
27. W. He, X. Liu, H. Nguyen, K. Nahrstedt, T. Abdelzaher, A cluster-based protocol to enforce integrity and preserve privacy in data aggregation, in *Proceedings of the IEEE International Conference on Distributed Computing Systems Workshops*, 2009
28. C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva, Directed diffusion for wireless sensor networking. IEEE/ACM Trans. Netw. **11**(1), 2–16 (2003)
29. A. Jhumka, M. Leeke, S. Shrestha, On the use of fake sources for source location privacy: Trade-offs between energy and privacy. Comput. J. **54**(6), 860–874 (2011)
30. Y. Jian, S. Chen, Z. Zhang, L. Zhang, Protecting receiver-location privacy in wireless sensor networks, in *Proceedings of the IEEE INFOCOM*, 2007
31. P. Kamat, W. Xu, W. Trappe, Y. Zhang, Temporal privacy in wireless sensor networks: Theory and practice. ACM Trans. Sens. Netw. **5**(4), 1–24 (2009)
32. P. Kamat, Y. Zhang, W. Trappe, C. Ozturk, Enhancing source-location privacy in sensor network routing, in *Proceedings of the IEEE International Conference on Distributed Computing Systems*, 2005
33. B. Karp, H.T. Kung, Gpsr: Greedy perimeter stateless routing for wireless networks, in *Proceedings of the ACM International Conference on Mobile Computing and Networking*, 2000
34. D. Kesdogan, J. Egner, R. Büschkes, Stop-and-Go-MIXes providing probabilistic anonymity in an open system, in *Proceedings of the International Workshop on Information Hiding*, 1998
35. N. Li, N. Zhang, S.K. Das, B. Thuraisingham, Privacy preservation in wireless sensor networks: A state-of-the-art survey. Ad Hoc Netw. **7**(8), 1501–1514 (2009)
36. X. Li, X. Wang, N. Zheng, Z. Wan, M. Gu, Enhanced location privacy protection of base station in wireless sensor networks, in *Proceedings of the International Conference on Mobile Ad-hoc and Sensor Networks*, 2009
37. Y. Li, J. Ren, Source-location privacy through dynamic routing in wireless sensor networks, in *Proceedings of the IEEE INFOCOM*, 2010
38. L. Lightfoot, Y. Li, J. Ren, Preserving source-location privacy in wireless sensor network using STaR routing, in *Proceedings of the IEEE Global Communications Conference*, 2010
39. H. Lim, C. Kim, Flooding in wireless ad hoc networks. Comput. Commun. **24**(3), 353–363 (2001)
40. S. Madden, M.J. Franklin, J.M. Hellerstein, W. Hong, TAG: A tiny aggregation service for ad-hoc sensor networks. ACM SIGOPS Oper. Syst., Rev. **36**(SI), 131–146 (2002)
41. K. Mehta, D. Liu, M. Wright, Protecting location privacy in sensor networks against a global eavesdropper. IEEE Trans. Mob. Comput. **11**(2), 320–336 (2012)
42. M.F. Mokbel, C.Y. Chow, Challenges in preserving location privacy in peer-to-peer environments, in *Proceedings of the International Workshop on Information Processing over Evolving Networks*, 2006
43. E.C.H. Ngai, I. Rodhe, On providing location privacy for mobile sinks in wireless sensor networks, in *Proceedings of the ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2009
44. L.B. Oliveira, D.F. Aranha, C.P.L. Gouvêa, M. Scott, D.F. Câmara, J. López, R. Dahab, TinyPBC: pairings for authenticated identity-based non-interactive key distribution in sensor networks. Comput. Commun. **34**(3), 485–493 (2011)

45. N. Oualha, A. Olivereau, Sensor and data privacy in industrial wireless sensor networks, in *Proceedings of the International Conference on Network Architectures and Information Systems Security*, 2011
46. C. Ozturk, Y. Zhang, W. Trappe, Source-location privacy in energy-constrained sensor network routing, in *Proceedings of the ACM International Workshop on Security of Ad hoc and Sensor Networks*, 2004
47. A. Perrig, R. Szewczyk, V. Wen, D. Culler, J.D. Tygar, SPINS: Security protocols for sensor networks. Wireless Netw. **8**(5), 521–534 (2002)
48. N.B. Priyantha, A. Chakraborty, H. Balakrishnan, The cricket location-support system, in *Proceedings of the ACM International Conference on Mobile Computing and Networking*, 2000
49. R. Rajagopalan, P. Varshney, Data-aggregation techniques in sensor networks: A survey. IEEE Commun. Surv. Tutor. **8**(4), 48–63 (2006)
50. J. Sen, A survey on wireless sensor network security. Int. J. Commun. Netw. Inf. Secur. **1**(2), 55 (2009)
51. M. Shao, Y. Yang, S. Zhu, G. Cao, Towards statistically strong source anonymity for sensor networks, in *Proceedings of the IEEE INFOCOM*, 2008
52. L. Sweeney, *k*-anonymity: A model for protecting privacy. Int. J. Uncertainty, Fuzziness Knowl. Based Syst. **10**(5), 557–570 (2002)
53. P. Szczechowiak, A. Kargl, M. Scott, M. Collier, On the application of pairing based cryptography to wireless sensor networks, in *Proceedings of the ACM International Conference on Wireless Network Security*, 2009
54. S. Wang, X.S. Wang, Y. Huang, Tracking the dynamic distribution of people in indoor space with noisy partitioning sensors, in *Proceedings of the IEEE International Conference on Mobile Data Management*, 2012
55. R. Want, A. Hopper, V. Falcao, J. Gibbons, The active badge location system. ACM Trans. Inf. Syst. **10**(1), 91–102 (1992)
56. A. Woo, T. Tong, D. Culler, Taming the underlying challenges of reliable multihop routing in sensor networks, in *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems*, 2003
57. World Wildlife Fund: http://wwf.panda.org [Accessed: April 24, 2012]
58. Y. Xi, L. Schwiebert, W. Shi, Preserving source location privacy in monitoring-based wireless sensor networks, in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing*, 2006
59. Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, G. Cao, Towards event source unobservability with minimum network traffic in sensor networks, in *Proceedings of the ACM International Conference on Wireless Network Security*, 2008
60. S. Zhu, S. Setia, S. Jajodia, LEAP: Efficient security mechanisms for large-scale distributed sensor networks, in *Proceedings of the ACM International Conference on Computer and Communications Security*, 2003