A Cloud-Based Scheme for Protecting Source-Location Privacy against *Hotspot-Locating* Attack in Wireless Sensor Networks

Mohamed M. E. A. Mahmoud and Xuemin (Sherman) Shen
Department of Electrical and Computer Engineering
University of Waterloo, Waterloo, Canada
mmabdels@bbcr.uwaterloo.ca and xshen@bbcr.uwaterloo.ca

Abstract—In wireless sensor networks, adversaries can make use of traffic information to locate the monitored objects, e.g., to hunt endangered animals or kill soldiers. In this paper, we first define a hotspot phenomenon that causes an obvious inconsistency in the network traffic pattern due to a large volume of packets originating from a small area. Second, we develop a realistic adversary model, assuming that the adversary can monitor the network traffic in multiple areas, rather than the entire network or only one area. Using this model, we introduce a novel attack called *Hotspot-Locating* where the adversary uses traffic analysis techniques to locate hotspots. Finally, we propose a cloud-based scheme for efficiently protecting source nodes' location privacy against *Hotspot-Locating* attack by creating a cloud with an irregular shape of fake traffic, to counteract the inconsistency in the traffic pattern and camouflage the source node in the nodes forming the cloud. To reduce the energy cost, clouds are active only during data transmission and the intersection of clouds creates a larger merged cloud, to reduce the number of fake packets and also boost privacy preservation. Simulation and analytical results demonstrate that our scheme can provide stronger privacy protection than *routing-based* schemes and requires much less energy than *global-adversary-based* schemes.

Index Terms—Wireless sensor network privacy, source-location privacy preserving schemes, context privacy, and anonymity.

1. Introduction

A wireless sensor network (WSN) consists of a large number of sensing devices called sensor nodes which are interconnected through wireless links to perform distributed sensing tasks. WSNs have found many useful applications for automatic data collecting [1, 2, 3], such as habitat monitoring, military surveillance, and target tracking, for monitoring the activities of enemy soldiers or valuable assets such as endangered animals. When a sensor node detects a soldier or an endangered animal, it reports the event to the data collector called the *Sink*. This data transmission may occur via multi-hop transmission, where the sensor nodes act as routers. In this paper, we consider habitat monitoring applications where the WSN is deployed for monitoring pandas. For example, a WSN has been deployed by the Save-The-Panda Organization to monitor pandas in a wild habitat [4]. While pandas move in the network, their presence and activities are periodically sensed by the sensor nodes and reported to the *Sink*.

However, WSNs are usually deployed in open and large areas that are unattended and lack of protected physical boundary, which makes the networks vulnerable to security threats. Since the sensed data are typically transmitted through wireless channels, adversaries can eavesdrop on the open and shared wireless medium and make use of traffic information to locate source nodes to hunt pandas. Therefore, preserving source nodes' location privacy is essential due to the easiness of locating pandas and their furs' large market value, e.g., a piece of a panda's fur was sold in China for $66,500 in 2003 [5].

The privacy threats can usually be classified into: *content privacy* and *contextual privacy* [6]. For the *content privacy* threat, the adversary attempts to observe the content of the packets sent in the network to learn the sensed data and the identities and locations of the source nodes. This privacy threat can be countered by encrypting the packets' contents and using pseudonyms instead of the real identities. For the *contextual privacy* threat, the adversary eavesdrops on the network transmissions and uses traffic analysis techniques to deduce sensitive information, including whether, when, and where the data is collected. Actually, the act of packet transmission itself reveals information even if the packets are strongly encrypted and the adversary could not interpret them [7]. The existing source location privacy-preserving schemes can be classified into *global-adversary-based* and *routing-based* schemes. These schemes employ either weak or impractical adversary model.

The *global-adversary-based* schemes [8, 9] assume that the adversary can monitor every radio transmission in every communication link in the network. To preserve source nodes' location privacy, each node has to send packets periodically, e.g., at fixed time slots. If a node does not have sensed data at one time slot, it sends dummy packet, so that the adversary cannot know whether the packet is for a real event or dummy data. However, the assumption that the adversary can monitor the transmissions of the entire network is not realistic, especially when the WSN is deployed in a large area. Moreover, if the adversary has a global view to the network traffic, he can locate pandas without making use of the network transmissions. Transmitting dummy packets periodically consumes a significant amount of energy and bandwidth, and decreases packet delivery ratio due to increasing packet collision, which makes these schemes impractical for WSNs with limited-energy nodes.

On the contrary, *routing-based* schemes [6] use weak adversary model assuming that the adversary has limited overhearing capability, e.g., similar to a sensor node's

transmission range, and can monitor only one local area at a time. These schemes assume that the adversary starts from the *Sink* and tries to locate the origin of a transmission by back tracing the hop-by-hop movement of the packets sent from the source node. Once the adversary overhears a transmission made from node A, he moves to A and waits. Then, he overhears a transmission from node B and moves to B to be closer to the source node, and so on until he locates the source node. *Routing-based* schemes try to preserve source nodes' location privacy by sending packets through different routes instead of one route, to make it infeasible for adversaries to trace back packets from the *Sink* to the source node because they cannot receive a continuous flow of packets. However, if the adversary's overhearing range is larger than the sensor nodes' transmission range, the likelihood of capturing a large ratio of the packets sent from a source node significantly increases, e.g., in [6], it is shown that if the adversary's overhearing range is three times the sensor nodes' transmission range, the likelihood of locating pandas is as high as 0.97. Moreover, if pandas stay for some time in one location, the adversary can capture enough packets to locate them even if the packets are sent through different route.

In this paper, we first define a hotspot phenomenon that causes an obvious inconsistency in the network traffic pattern due to a large volume of packets originating from a small area. Hotspots can be formed for different reasons, e.g., when pandas have high density or spend some time in one area due to the availability of food, water, shadow, shelter, etc. Second, we develop a realistic adversary model assuming that the adversary has a partial view to the network traffic by distributing a group of monitoring devices at different observation points. Each monitoring device collects traffic information, including a packet's content, the coordinates of the sending node, and the time of sending the packet. Then, using this model, we introduce a novel attack called *Hotspot-Locating*, where the adversary tries to make use of the traffic inconsistency caused by hotspots to locate pandas by analyzing the data collected from the observation points using traffic analysis techniques such as the nodes' packet sending rates and packets correlation.

Finally, we propose a cloud-based scheme for efficiently protecting source nodes' location privacy against *Hotspot-Locating* attack by creating a cloud with an irregular shape of fake traffic, to counteract the inconsistency of the traffic pattern caused by hotspots, and camouflage the source node within the group of nodes forming the cloud. The fake packets also enable the real source node to send the sensed data anonymously to a fake source node selected from the cloud's nodes to send to the *Sink*. Cryptographic operations are used to change the packets' appearance at each hop to prevent packet correlation and make the source node indistinguishable because the adversary cannot differentiate between the fake and real traffic, i.e., the cloud's traffic pattern looks random for the adversary. Moreover, tracing the packets back to the source node is nearly impossible because the real traffic is indistinguishable and the real source node sends its packets through different fake source nodes.

WSNs may be deployed in areas where human maintaining is impractical and thus recharging or replacing the batteries of sensor nodes may be infeasible or impossible. To reduce the energy cost, clouds are active only during data transmission, the nodes generate fake packets probabilistically, and the intersection of clouds creates a larger merged cloud to reduce the number of fake packets and also boost privacy protection. Moreover, our scheme uses energy-efficient cryptosystems such as hash function and symmetric-key cryptography and avoids the intensive energy consuming cryptosystems such as asymmetric-key cryptography. It also avoids large-scale packet broadcasting and network-wide packet flooding. In order to determine the tradeoff between the energy cost and the strength of privacy protection, some parameters such as the cloud size can be tuned.

Simulation and analytical results demonstrate that the *Hotspot-Locating* attack is a severe threat to source nodes' location privacy because adversaries can locate the source nodes using a limited number of monitoring devices with low overhearing range and simple traffic analysis techniques. *Routing-based* privacy preserving schemes are vulnerable to *Hotspot-Locating* attack because they leak traffic analysis information, i.e., the adversary can correlate packets and observe the high packet sending rates of the sensor nodes near of hotspots. Our scheme can provide much stronger privacy protection than *routing-based* schemes because in addition to varying traffic routes, it can conceal the traffic analysis information. Our scheme also requires much less energy than *global-adversary-based* schemes.

Our main contributions can be summarized as follows. (1) We develop a realistic adversary model; (2) We define a hotspot phenomenon and introduce a *Hotspot-Locating* attack, and we have shown that *routing-based* schemes are vulnerable to this attack; and (3) We propose a novel scheme for protecting source nodes' location privacy against *Hotspot-Locating* attack with a low energy cost.

The remainder of this paper is organized as follows. We review the related works in Section 2. The network and adversary models are discussed in Section 3. The hotspot phenomenon and the *Hotspot-Locating* attack are discussed in Section 4. We present our privacy-preserving scheme in Section 5. Analytical and simulation results are given in Section 6, followed by a conclusion and future work in Section 7.

2. Related Works

Recently, location privacy in wireless and wired networks has gained more and more attention. Different schemes have been developed to protect users' privacy in location tracking systems [10] which determine the users' positions for location-based services. Location privacy in these schemes is content-oriented where location information is collected and protected as the users' private data. Onion routing [11] provides anonymous communications

for the Internet by hiding the identities of the end users of a communication session. The proposed schemes in [12-14] conceal the nodes' network/MAC addresses in order to achieve anonymous communications for mobile ad hoc networks. However, these schemes employ different network and threat models from the ones suitable for the source location-privacy problem in sensor networks.

The proposed scheme in [15, 16] uses fake packet injection to preserve the location privacy of the *Sink*. The scheme makes it hard for an adversary to deduce the location of the *Sink* by making the directions of both incoming and outgoing traffic at each node uniformly distributed. In [17], Deng et al. propose a scheme for preserving the *Sink*'s location privacy against traffic-rate analysis attacks. Each node has to send packets at a constant rate and the transmissions of the packets are randomly delayed to hide the traffic pattern and the parent-child relationship.

Routing-based schemes preserve source nodes' location privacy by sending packets through different routes to make back tracing the movement of the packets from the *Sink* to the source nodes infeasible. In [6, 18], a random-walk based privacy-preserving scheme, called Phantom, is proposed. Each packet takes a random walk to a random location before it is sent to the *Sink*. However, the scheme fails if the adversary's overhearing range is more than the sensor nodes' transmission range. Moreover, it is very likely that routes will loop around the source node and branch to a random location that is not far from the node. To resolve this problem, the source node can attach the direction of the random walk to the packet header, and each node in the random-walk route forwards the packet to a random neighbor in the same direction. However, once a packet is captured in the random-walk route, the adversary can know the direction information to the source node, which reduces the complexity of tracing the packets back to the source.

Wang et al. [19] present a privacy-aware parallel routing scheme to maximize the time of back tracing the packets to the source nodes. A weighted random stride routing that breaks the entire routing into strides is proposed. In [20], dynamically selected nodes in each route modify the packets to make back tracing packets to the source node difficult, but the adversary can trace the modified packets if there are only one or few transmissions.

G*lobal-adversary-based* schemes [8, 9] assume that adversaries can monitor the traffic of the entire network. Each node has to periodically send packets, and send dummy packets if it does not have sensed data so that it is infeasible for the adversaries to distinguish between the real and dummy packets. However, if the nodes increase the time interval of packet transmission to reduce the energy cost of the dummy packets, the packet delivery delay increases. This is attributed to the fact that if an event is sensed between two time slots, the node should wait the first time slot to transmit the event. To alleviate the tradeoff between the overhead of dummy packets and packet delivery delay, Shao et al. [8] propose a statistically strong source privacy preserving scheme. The nodes send the real packets as soon as possible with keeping them statistically indistinguishable from the dummy packets.

Fan et al. [21] preserve location privacy by using homomorphic encryption operations to prevent traffic analysis in network coding. In [22], each cluster header can filter the dummy packets received from the sensor nodes of its cluster to reduce the number of dummy packets. However, the scheme requires much computation overhead due to using asymmetric-key cryptography, and the packet delivery delay is long because the cluster header sends packets with a fixed rate regardless of the number of events it collects. Mehta et al. [23] formalize the location privacy problem using a global adversary model and compute a lower bound for the overhead required for achieving a given level of privacy protection. The proposed scheme by Alomair et al. [24] can guarantee event indistinguishability by achieving interval indistinguishability, where the adversary cannot distinguish between the first, the middle, or the end of the interval. In [25], dummy packets can be filtered at proxy nodes, and the lifetime of the WSN is analyzed at different proxy assignment methodologies. Hong et al. [26] propose a scheme that can thwart time correlation attack. In this attack, the adversary exploits the time correlation of transmissions in successive links to learn the end-to-end route.

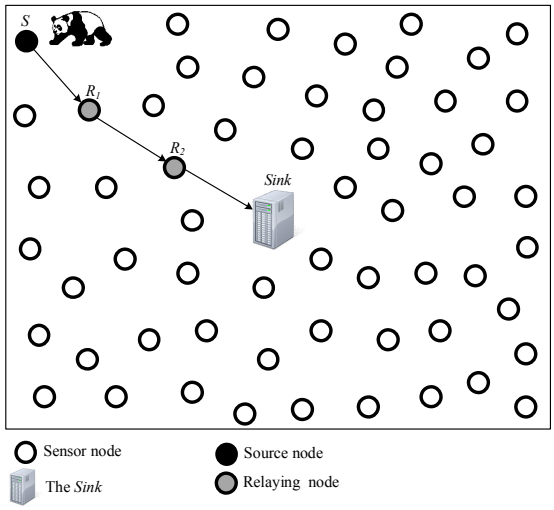

Fig. 1: The architecture of the considered WSN.

3. Network and Adversary Models

3.1 Network Model

As illustrated in Fig. 1, the considered WSN consists of the *Sink* and a large number of homogeneous panda-detection sensor nodes which are randomly deployed in an area of interest. The *Sink* and the sensor nodes are stationary. The sensor nodes are resource-constrained devices with low battery power and computation capacity, but equipped with sensing, data processing, and communicating components. The sensor nodes are interconnected through wireless links to perform distributed data collection. The *Sink* has sufficient computation and storage capabilities [27] to perform two basic functions: a) broadcasting beacon packets to bootstrap our scheme; and b) collecting the data sensed by sensor nodes. Pandas have embedded radio fre-

quency (RF) tags [4], and when a sensor node senses a panda, the node is called a source node and generates and sends event packets to the *Sink*. Each sensor node has a transmission radius of r_S meters and the communication in the network is bidirectional, i.e., any two nodes within the wireless transmission range can communicate with each other. Multihop communication is employed if the distance between a sensor node and the *Sink* is more than r_S, where some sensor nodes (called relaying nodes) act as routers to relay the source node's packets. The *Sink* is the sole destination for all the event packets.

3.2 Adversary Model

The adversary is a hunter who eavesdrops on the wireless transmissions and attempts to make use of the network traffic to determine the locations of pandas to hunt them. The adversary distributes a group of monitoring devices in areas of interest, called observation points, to collect the traffic information in these areas, but he cannot monitor the traffic of the entire network. The adversary analyzes the information collected by the monitoring devices to locate pandas or change the observation points, e.g., to be closer to pandas. For example, Fig. 2 shows that the adversary distributes five monitoring devices in five observation areas named $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, $\mathcal{A}_4$, and $\mathcal{A}_5$.

In addition, the adversary has the following characteristics:

1) **Passive:** The adversary launches only passive attacks to hunt pandas and avoids active attacks to be invisible from the network operator. Disrupting the network proper operation is not beneficial for the adversary to make use of the network transmissions to locate pandas. Passive attacks are more dangerous than active attacks in the sense that they are much more invisible and difficult to detect. Preserving the location privacy of the *Sink* is more important in other applications. For example, in military applications, the network is deployed in hostile environment and the adversary aims to locate the *Sink* to disrupt the network by physically destroying the *Sink* to prevent the enemy from collecting sensitive information.

2) **Well-equipped**: Each monitoring device is equipped with supporting equipments such as antenna and spectrum analyzers. It can intercept the packets in the monitored area and measure the angle of arrival as well as the strength of the signal to accurately determine the locations of the nodes that send packets. However, it cannot determine the location of the receiving node because all the nodes in the transmission range can be the potential receiver of the packet. The monitoring device's overhearing radius (r_A) may be larger than the sensor nodes' transmission radius, e.g., $r_A = \xi \times r_S$ and $\xi \geq 1$, but the adversary cannot monitor the entire network. This is realistic because sensor nodes are cheap and simple devices, but the monitoring devices will be more sophisticated due to the large market value of the pandas' furs. The monitoring devices have sufficient memory for storing all the transmission information in their overhearing range. They also have large energy resource and sufficient computation capability for analyzing the collected data, but it is not sufficient for breaking the encryption algorithm or the hash function.

3) **Informed**: The adversary knows the location of the *Sink* and monitors its traffic because it is the destination of all the event packets. To appropriately study privacy, we apply Kerckhoff's principle [28] by assuming that the adversary knows the privacy preserving scheme and the used cryptosystems, but does not know the cryptographic keys.

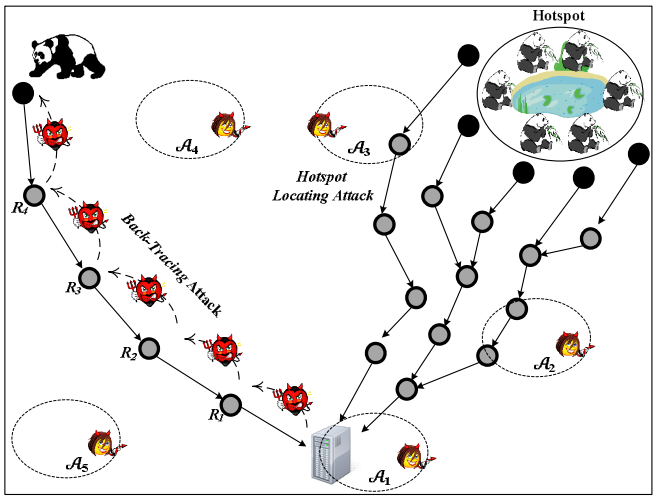

Fig. 2: The adversary model.

4. *HOTSPOT-LOCATING* ATTACK

4.1 Hotspot Phenomenon

A hotspot is formed when a large volume of packets are sent from the sensor nodes of a small area, causing an obvious inconsistency in the network traffic which may last for some time. The adversary attempts to make use of this traffic inconsistency to locate hotspots to hunt pandas. Figs. 3 and 4 can illustrate the hotspot phenomenon. Fig. 3 shows the average packet sending rate of each sensor node when there are no hotspots and using the shortest path routing scheme where the nodes send the sensed data to the *Sink* through the minimum number of relaying nodes. This traffic pattern is obtained when the number of pandas sensed by each sensor node and the time spent by pandas at each node are uniformly distributed. It can be seen that the nodes near of the *Sink* clearly send a significantly larger volume of packets than the nodes further away, and the packet sending rates gradually decrease as we move to the network edges. This is because the nodes at the border only send their sensed data but the other nodes relay the others' data in addition to sending their data.

However, it is not reasonable to assume that pandas spend the same time and have the same density in every area covered by the network. Pandas will have high density in some areas, e.g., a group of pandas live and move together, and will spend longer times at some areas, e.g., due to

the availability of food, water, shadow, shelter, etc, which create areas with large packet sending rates, called hotspots.

Fig. 4 shows that the data transmission is accumulated at a hotspot, and the nodes' packet sending rates are not uniformly distributed. It is obvious that there is much more traffic originating from the hotspot than the rest of the network. The adversary can locate hotspots more successfully when the contrast between the traffic rates of the hotspot and the other areas is large, and when the hotspot lasts for some time. Although the adversary does not have global view to the network traffic, the next section will discuss how he can locate hotspots using distributed monitoring devices with a small overhearing range.

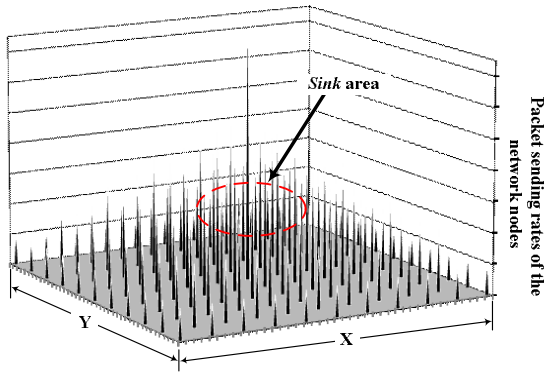

Fig. 3: The packet sending rate of each node without hotspots.

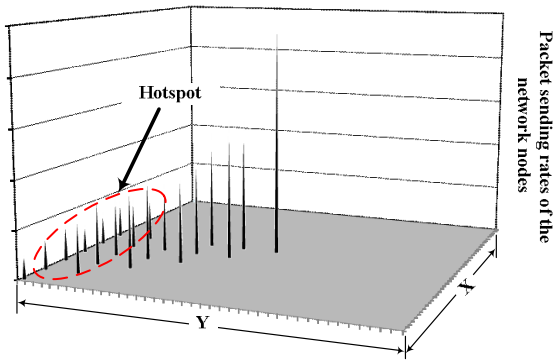

Fig. 4: The packet sending rate of each node with a hotspot.

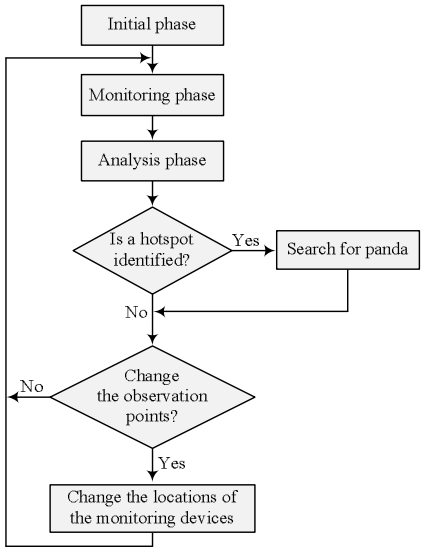

Fig. 5: The flowchart of *Hotspot-Locating* attack.

4.2 Hotspot-Locating Attack

Fig. 5 shows the flowchart of a *Hotspot-Locating* attack using the adversary model discussed in Section 3.2. In the initial phase, the adversary deploys a monitoring device near of the *Sink* and deploys the other devices at initial observation points distributed in the network. For the monitoring phase, the monitoring devices collect traffic information which includes the following tuple $<P_i, X_i, Y_i, t_i>$, where P_i is the content of a packet, (X_i, Y_i) is the coordinates of the sensor node that sent the packet, and t_i is the time of sending the packet. For the analysis phase, the adversary uses traffic analysis techniques to analyze the collected data to decide to (1) search an area for pandas; or (2) change the locations of the monitoring devices, e.g., to move closer to a probable hotspot or move to a more promising area that can lead to a hotspot. If the adversary identifies an area as a hotspot, but no pandas are found, this is called *false positive*.

A wide range of traffic analysis techniques can be employed to analyze the collected data to locate hotspots. In this paper, we consider three techniques, called content correlation, time correlation, and packet sending rates. For content correlation, if a packet is observed in two locations at different times, the adversary can determine that the packet is forwarded from one location to another. For example, from Fig. 2, if a packet is observed at $\mathcal{A}_3$, $\mathcal{A}_2$, and $\mathcal{A}_1$ at times t_1, t_2, and t_3, respectively, where $t_1 < t_2 < t_3$, the adversary can conclude that the packet is forwarded from $\mathcal{A}_3$ to $\mathcal{A}_1$, or $\mathcal{A}_3$ is closer to a hotspot than $\mathcal{A}_2$. To prevent this correlation, the appearance of the packet should change at each hop. For time correlation, if there is a temporal relationship between the packets captured at areas $\mathcal{A}_3$ and those captured at $\mathcal{A}_2$ and $\mathcal{A}_1$, the adversary can conclude that the packets are forwarded from $\mathcal{A}_3$ to $\mathcal{A}_2$, and then to $\mathcal{A}_1$. This temporal relationship can be determined by observing that when a packet is captured at $\mathcal{A}_3$ at time t, a packet is captured at $\mathcal{A}_3$ at $t + \delta$ regardless of the contents of the packets, where δ is the transmission delay. For packet sending rates, the adversary can measure the packet sending rates of the nodes in the monitored areas. From Fig. 2, the adversary can figure out that the traffic rates at areas $\mathcal{A}_4$ and $\mathcal{A}_5$ are much less than those at $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$, and conclude that it is not worth to continue in monitoring $\mathcal{A}_4$ and $\mathcal{A}_5$.

Actually, the adversary attempts to make use of the fact that the nodes at a hotspot send more packets than the nodes further away to identify the hotspot. One way to do this is by using traffic-analysis back tracing (instead of packet back tracing) to move from the *Sink* to a hotspot. The adversary tries to locate a hotspot by tracing the nodes' large packet sending rates caused by the stream of packets flowing from the hotspot towards the *Sink*. Fig. 6 illustrates that the adversary can launch the attack using boundary or inside back tracing. For boundary traffic-analysis back tracing, the adversary can move on the boundary of the large traffic sent from a hotspot. The adversary can identify the boundary easily by observing the large difference in the packet sending rates of the nodes in the two sides of the

boundary. For inside traffic-analysis back tracing, the adversary can follow the high packet sending rates of the nodes that relay the hotspot's packets. The adversary moves the monitoring devices to be closer to the hotspot. These procedures continue until the adversary suspects that an area is a hotspot once he observes large drop in the packet sending rates due to passing the hotspot. Another approach for identifying a hotspot is by observing that the number of outgoing packets from an area is much more than the number of ingoing packets. This attack indicates that even if the adversary does not have global view to the traffic of the network, he can locate hotspots by using traffic-analysis techniques and simple devices.

The required time for locating a hotspot depends on many factors such as the number of monitoring devices, the overhearing radius of the monitoring devices, and the effectiveness of the traffic-analysis techniques. Obviously, this attack can be launched successfully with a low false positive rate when the adversary uses a large number of sophisticated monitoring devices, the inconsistency in the network traffic is very obvious, and the hotspot lasts for some time. This is because to perform the traffic-rate analysis, each monitoring device has to stay at its observation point for some time so that it can collect sufficient packets for computing the traffic rate. However, even if the adversary uses few and simple monitoring devices, it may take some time to locate a hotspot, but the adversary still has a good chance to hunt pandas when hotspots last for some time. Moreover, even if the adversary could not find pandas, the fact that pandas have appeared in some areas is still beneficial for the adversary because they will very likely visit these areas again, i.e., it is a kind of studying pandas' behaviors.

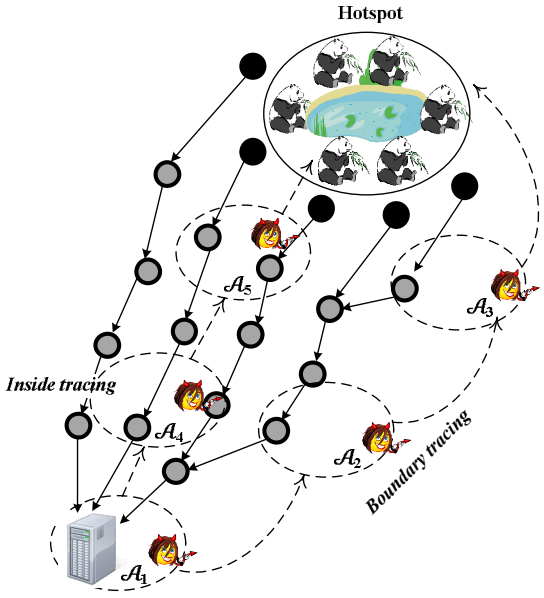

Fig. 6: Inside and boundary back tracing for locating hotspots.

Routing-based privacy preserving schemes are vulnerable to *Hotspot-Locating* attack because they are designed to make receiving a continuous stream of packets by the adversary difficult, but they cannot conceal the traffic-analysis information such as the nodes' packet sending rate and packet correlation. Leaking traffic-analysis information will enable the adversaries to launch *Hotspot-Locating* attack successfully. *Routing-based* schemes may work more effectively if there are no hotspots because there is no inconsistency in the network traffic, but as we discussed earlier, it is not expected that the network traffic will be uniform all the time.

5. Cloud-Based Privacy Preserving Scheme

5.1 Pre-deployment Phase

Before deploying the network, each sensor node A is loaded with a unique identity ID_A, a shared key with the *Sink* K_A, and a secret key d_A that is used to compute a shared key with any sensor node using identity-based cryptography (IBC) based on bilinear pairing. The network operator generates a prime p, a cyclic additive group (G_a), and a cyclic multiplicative group (G_m) of the same order p such that an efficiently computable bilinear pairing $\hat{e}$: $G_a \times G_a \rightarrow G_m$ is known. The bilinear mapping has the following properties:

→ **Bilinear**: $\hat{e}(a \cdot P, b \cdot Q) = \hat{e}(b \cdot P, a \cdot Q) = \hat{e}(P, Q)^{ab}$, $\forall P, Q \in G_a$, $\forall a, b \in Z_P^*$.

→ **Non-degeneracy**: $\hat{e}(P, Q) \neq 1_{Gm}$, $\forall P, Q \in G_a$.

→ **Symmetric**: $\hat{e}(P, Q) = \hat{e}(Q, P)$, $\forall P, Q \in G_a$.

According to [29], the bilinear mapping $\hat{e}$ can be computed efficiently using the Weil and Tate pairings on elliptic curves. The network operator selects a random element $g \in Z_P^*$ known as the master key, and computes the secret keys of the sensor nodes based on their identities. A node with identity ID_A receives the key $d_A = g \cdot H'(ID_A) \in G_a$, where $H':\{0, 1\}^* \rightarrow G_a$. According to *Discrete Logarithm Difficulty*, given P and $g \cdot P$, there is no algorithm running in expected polynomial time that can compute g, where $P \in G_a$. Using IBC, two sensor nodes with identity/private key pairs (ID_A, d_A) and (ID_B, d_B) can independently compute the shared key using one pairing operation and without exchanging messages as follows:

$$
\begin{aligned}
\mathbf{K_{AB}} &= \hat{e}(H'(ID_B), d_A) \\
&= \hat{e}(H'(ID_B), g \cdot H'(ID_A)) \\
&= \hat{e}(g \cdot H'(ID_B), H'(ID_A)) && \text{(\textit{bilinear property})} \\
&= \hat{e}(d_B, H'(ID_A)) \\
&= \hat{e}(H'(ID_A), d_B) = \mathbf{K_{BA}} && \text{(\textit{symmetric property})}
\end{aligned}
$$

5.2 Bootstrapping Phase

This phase is performed only one time in the lifetime of the network, after the network is deployed and before it starts data collection. This phase has three main purposes: (1) Informing the *Sink* about the nodes' locations to link an event to its location; (2) Assigning fake source nodes and discovering the shortest routes to the *Sink*; and (3) Forming groups that are necessary for creating clouds. After deploying the network, the *Sink* broadcasts a beacon packet and each sensor node adds its identity and broadcasts the pack-

et. Each node can know the shortest route to the *Sink* which includes the identities of the nodes in the first received beacon packet. Every sensor node determines its own location information using some localization methods such as those proposed in [30, 31] and notifies the *Sink* through the shortest route.

In order to assign fake source nodes, node A broadcasts *Fake Nodes Request Packet* (*FREQ*) that contains the maximum number of hops (h_{max}) the packet can be propagated. Each node adds its identity and broadcasts the packet if the number of hops is fewer than h_{max}; otherwise, it unicasts *Fake Nodes Request Reply* (*FREP*) packet to node A, containing the identities of the nodes in the route. Node A receives multiple *FREP* packets containing different routes with maximum number of hops of h_{max}. It chooses a group of nodes at different number of hops and unicasts the *Fake Node Assignment Packets* (*FASS*) to assign them as fake source nodes to its packets. For each *FASS* packet, node A adds the identities of the nodes in the route and a random value that will be used to generate pseudonyms shared between each two neighboring nodes in the route.

In order to reduce the number of *FASS* packets, one packet can assign multiple fake source nodes, i.e., the end node and some intermediate nodes. Moreover, the nodes can also use the *FREP* and *FASS* packets that they relay to assign fake source nodes. Actually, *FREQ* and *FREP* packets may not be needed if a node could obtain sufficient number of routes to probable fake source nodes from the beacon packets or from relaying the other nodes' *FREQ*, *FASS*, and *FREP* packets.

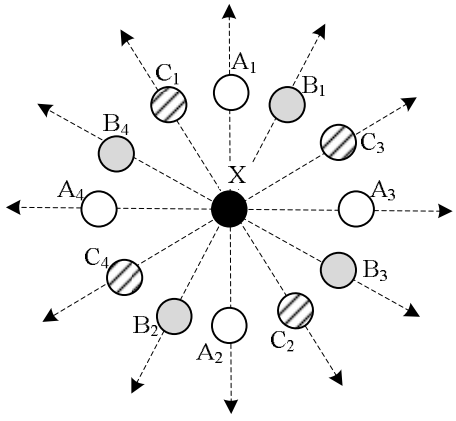

Fig. 7: Grouping the one-hop neighbors of node X.

Finally, each node groups its one-hop neighbors in such a way that each group can send packets in different directions, e.g., by choosing the nodes that are in opposite directions in one group. We note that one node may participate in multiple groups. An example for grouping a node's neighbors is shown in Fig. 7. Node X divides its neighbors into three groups with four nodes in each group ({{A_1, A_2, A_3, A_4}, {B_1, B_2, B_3, B_4}, {C_1, C_2, C_3, C_4}}), so that they can send packets in different directions. Node X has also to share a key with each group. A simple way to do this is by computing the shared keys with its neighboring nodes using bilinear pairing as discussed in Section 5.1, and sending the group key and random value encrypted with the shared key to each neighboring node. The random value will be used to create pseudonyms for the group.

5.3 Event Transmission Phase

Privacy is the guarantee that information in its general sense is observable or decipherable by only those who are intentionally meant to observe or decipher it. According to Pfitzmann and Kohntopp [32, 33], anonymity is defined as the state of being unidentifiable within a set of objects called the anonymity set. The essence of our scheme is based on the principle that one of the best ways to avoid being identified is to mix with the crowd. Our scheme conceals a source node within a group of nodes with an irregular shape, called "cloud". A source node is considered to have a complete anonymity if the adversary cannot identify it in the cloud, i.e., the adversary may be able to know that a node of a cloud sends an event packet, but he cannot identify this node.

As illustrated in Fig. 8, in this phase, a real source node sends an event packet anonymously to a fake source node to send to the *Sink*. Simultaneously, a cloud of fake packets is activated to protect the source node's location. In order to make it infeasible to infer a source node's location by analyzing the traffic-analysis information collected from the monitored areas, the nodes of the cloud send fake packets to add randomness to the traffic pattern to (1) make the transmission of the event packet from the real source node to the fake one indistinguishable; and (2) make the source node indistinguishable by analyzing the packet sending rates of the cloud's nodes. Instead of using a single path or a single fake source node, the real source node transmits packets through different paths to different fake sources to prevent the linkability between the real and fake source nodes and make packet back tracing infeasible.

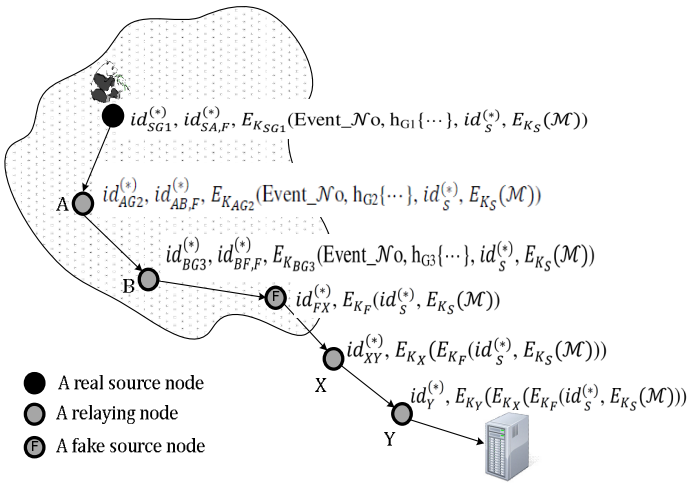

Fig. 8: Event transmission phase.

5.3.1 Pseudonyms

If two nodes share a key, they can create a sequence of pseudonyms using a one-way keyed hash function by iteratively hashing a random value. For example, the two nodes A and B which share the key K_{AB} can create the following

pseudonyms using the random value R and the hash function *H()*:

$$id_{AB}^{(1)}, id_{AB}^{(2)}, id_{AB}^{(3)}, \ldots., id_{AB}^{(n)},$$

Where, $id_{AB}^{(i)} = H(K_{AB}, id_{AB}^{(i-1)})$ and $id_{AB}^{(1)} = H(K_{AB}, R)$.

Obviously, with changing the random value, the two nodes can generate different pseudonyms using the same key. This means that *pseudonyms are not only used for identifying the sending and receiving nodes, but also for identifying routes by using different random values for different routes.*

However, if a node does not receive a pseudonym, e.g., due to packet drop, the two nodes may lose pseudonym synchronization. To avoid this, each node should use a sliding window to match a received pseudonym against a window of expected pseudonyms. From Fig. 9, the expected pseudonym is number *i*, but the node matches a received pseudonym with a window of n expected pseudonyms. If the node receives pseudonym number *i*+1 instead of *i*, the node can identify the pseudonym and does not lose synchronization. The window is shifted to update the starting point to the pseudonym number *i*+2. The length of the sliding window (n) is governed by the packet loss rate, with a larger n at higher packet loss rate.

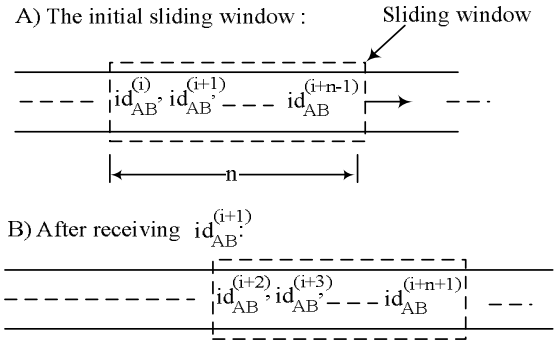

Fig. 9: Pseudonyms' sliding window.

5.3.2 Real - Fake Source Nodes' Route

When a source node (S) wants to send data to the *Sink*, it first picks up a fake source node (*F*) from its list of fake source nodes and a group (G_1) that contains *F*, and sends the following event packet:

$$id_{SG1}^{(*)}, id_{SA,F}^{(*)}, E_{K_{SG1}}(\text{Event\_}\mathcal{N}\text{o}, \text{h}_{\text{G1}}\{\cdots\}, id_S^{(*)}, E_{K_S}(\mathcal{M}))$$

Where:

$E_{K_{SG1}}$ is an encryption operation using the shared key between S and G_1.

$E_{K_S}(\mathcal{M})$ is the ciphertext of message $\mathcal{M}$ encrypted with K_S.

$h_{G1}\{\cdots\}$ is the number of hops the fake packets should be sent by the nodes of G_1.

$id_S^{(*)}$ is the pseudonym shared between node S and the *Sink*.

$id_{SG1}^{(*)}$ is the pseudonym shared between S and G_1.

$id_{SA,F}^{(*)}$ is the shared pseudonym between S and the relaying node A in the route to the fake source node *F*.

The source node encrypts the message $\mathcal{M}$ with the shared key with the *Sink* (K_S) to provide message confidentiality, authenticity, and integrity. In order to enable the *Sink* to know the location of the source node and the key it should use to decrypt the message, the source node attaches its pseudonym ($id_S^{(*)}$). Event_$\mathcal{N}$o is the event unique number. After receiving the packet, the neighbors of S first match the attached group pseudonym ($id_{SG1}^{(*)}$) to the expected ones. If a node cannot find the pseudonym in its table, it discards the packet, else it accepts the packet and updates the group pseudonyms' window to synchronize with node S. For the group nodes, if a node does not find the relaying node's pseudonym ($id_{SA,F}^{(*)}$) in its table, it has to generate a fake packet, else it relays the packet to the next node in the route to the fake source node.

The relaying node (A) selects the group that contains the next-hop relaying node (B) in the route to *F*, e.g., G_2. Then, it attaches the group pseudonym ($id_{AG2}^{(*)}$) and the pseudonym shared with node B ($id_{AB,F}^{(*)}$). The node computes the hop counts ($h_{G2}\{\cdots\}$) by reducing $h_{G1}\{\cdots\}$ to limit the propagation area of the fake packets that are generated as the event packet propagates. Node A encrypts the packet with the key shared with G_2 and sends the following packet:

$$id_{AG2}^{(*)}, id_{AB,F}^{(*)}, E_{K_{AG2}}(\text{Event\_}\mathcal{N}\text{o}, \text{h}_{G2}\{\cdots\}, id_S^{(*)}, E_{K_S}(\mathcal{M}))$$

These procedures are repeated at each hop until the fake source node receives the event packet anonymously as shown in Fig. 8. Since the packet is encrypted with different keys at each hop, it looks quite different as it is relayed from the real source node to the fake source node.

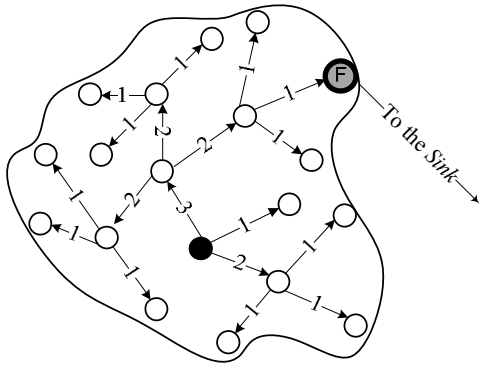

Fig. 10: Event/fake packets spreading.

5.3.3 Fake Packets

As an event packet is propagating from the real source node to the fake source, fake packets are sent to create a cloud of fake traffic. To send a fake packet, the node T chooses a group, e.g., G_4, and sends the following fake packet, where Rand is a random value:

$$id_{TG4}^{(*)}, \text{Rand}, E_{K_{TG4}}(\text{Event\_}\mathcal{N}\text{o}, \text{h}_{G4}\{\cdots\}, id_S^{(*)}, E_{K_S}(\mathcal{M}))$$

Since the nodes of G_4 cannot find Rand in their pseudonyms' tables, the nodes will generate fake packets. The nodes have also to decrease the hop counts $h_{G4}\{\cdots\}$ to limit the propagation area of the fake packets. Once the hop

count reaches zero, the nodes stop generating fake packets. For example, if a source node sends an event packet with the hop counts $h_G\{2, 4, 3\}$, where group G includes the three nodes A, B, and C, the fake packets will be relayed two, four, and three hops through the nodes A, B, and C, respectively. The nodes A, B, and C also reduce the hop counts to be {1, 1, 1}, {3, 3, 3}, and {2, 2, 2}, respectively. Fig. 10 illustrates how our scheme can restrict the propagation area of fake packets, where the numbers on the arrows are the hop counts. By this way, the source node can determine the hop counts so that its location can be anywhere in the cloud, and the cloud shape is an irregular and changeable each time the source node sends an event. To prevent generating multiple fake packets for one event when a node receives the same packet from different neighbors, all packets with the same Event_$\mathcal{N}$o are processed once. To prevent time correlation, when a node receives a packet, it should wait a random delay before relaying the packet or generating a fake packet.

5.3.4 Fake Source Node - Sink Route

From the pseudonym $id_{BF,F}^{(*)}$, the fake source node can know that the packet has to be sent to the *Sink*. As shown in Fig. 8, the fake source node sends the packet to the first node in the route to the *Sink*. The packet contains a pseudonym shared with the next-hop node ($id_{FX}^{(*)}$), and the message and the source node's pseudonym encrypted with the shared key with the *Sink*. Each relaying node re-encrypts the packet with the shared key with the *Sink* and replaces the pseudonym with the one shared with the next node in the route. The purpose of adding an encryption layer at each relaying node is to make the packet look different as it propagates from the fake source node to the *Sink* to prevent packet correlation and make back tracing packets to the fake source node infeasible. As the packet propagates to the *Sink*, the neighboring nodes do not send fake packets because they cannot find the packet's pseudonym in their tables. When the *Sink* receives the packet, it checks the pseudonym ($id_Y^{(*)}$) to determine the keys it should use to remove the encryption layers and recover the message. We note that the pseudonym of node Y is linkable to one route which is known to the *Sink*.

5.3.5 Merging Clouds

If a node receives multiple packets from multiple clouds during a short time interval τ, it sends only one fake packet, e.g., for the packet that has larger number of hop counts. If the adversary cannot distinguish the traffic belonging to the individual clouds, the clouds can be merged into a larger cloud because the adversary will see the nodes of the merged cloud send one packet in a time interval. From Fig. 11, if the clouds of source nodes S_1 and S_2 are intersected, the adversary can see a large merged cloud. This cloud merging is possible when S_1 and S_2 transmit events within a small time interval. Cloud merging has two main benefits: (1) *Low cost*: the nodes of the intersection areas do not send one fake packet for each cloud, e.g., if a node participates in n clouds, it sends only one fake packet instead of n, and thus the node can save n-1 fake packets; and (2) *Stronger privacy protection*: a merged cloud has a larger anonymity set because it has more nodes than the individual clouds. Cloud merging property is especially important for hotspots because clouds are very likely intersected which can both reduce the number of fake packets and boost privacy protection.

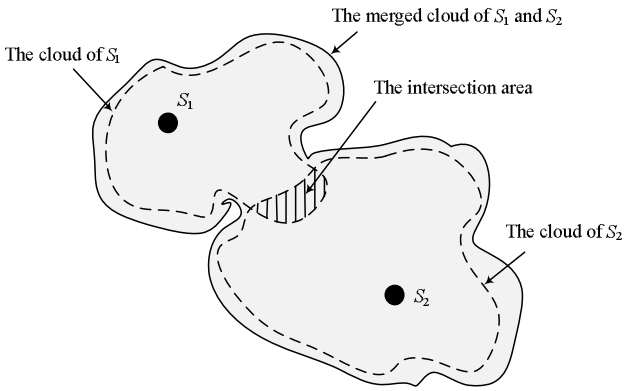

Fig. 11: Merging clouds.

6. Evaluations

6.1 Privacy Preservation

6.1.1 Analysis

For *Pseudonyms unlinkability,* the adversary cannot link the pseudonyms of one sequence. The importance of this property lies in the fact that if an adversary could link a pseudonym to a node, he will not benefit from this conclusion in the future. In our scheme, generating or correlating pseudonyms is infeasible without knowing the secret key used in generating them. Even if there is only one transmission, fake packets can make pseudonyms linkability infeasible because the adversary cannot distinguish between event and fake packets. Pseudonym collision means that more than one node have the same pseudonym, because the hash function may generate the same hash value from hashing two different inputs. Using birthday paradox, the pseudonym collision probability is $2^{-K/2}$, where K is the number of bits of a pseudonym. For example, if K = 64 bits, the pseudonym collision probability is $2.3e^{-10}$, which implies one pseudonym collision every $4.2e^{+9}$ pseudonyms. Although this probability can be negligible, a collision will be resolved automatically in our scheme because the nodes use different keys for generating pseudonyms, i.e., if multiple nodes have the same pseudonym at the same time, this will not continue because of using different keys.

Each node should not store the initial random value that is used in generating pseudonyms, and store only the last used pseudonym. This is because if the node is compromised and its key is known, the adversary cannot compute the pseudonyms used before, taking advantage of the one-way property of the hash functions, i.e., it is infeasible to compute $id_{AB}^{(i-1)}$ from $id_{AB}^{(i)}$ even if the key K_{AB} is known.

For *packets content correlation*, if the packets of a flow (or a cloud) have a fixed or a distinguishable part that does not change at each hop, the adversary can identify the

packets' flow (or the cloud). Encrypting the packets cannot prevent this correlation because the ciphertexts look the same at each hop, but the packets' appearance has to change at each hop. In our scheme, the hop-by-hop decryption/encryption operations and changing pseudonyms can make a packet look quite different and uncorrelated as it is relayed. Actually, we make use of the diffusion property of the encryption scheme, i.e. encrypting a message $\mathcal{M}$ with different keys produces different ciphetexts, e.g., although the ciphertexts $E_{KA}(\mathcal{M})$ and $E_{KB}(\mathcal{M})$ are for the same message, they look completely different. Moreover, given $E_{KA}(\mathcal{M})$ and $E_{KB}(\mathcal{M})$, it is computationally infeasible to correlate the two ciphertexts without knowing the secret keys K_A and K_B and with using secure cryptosystems such as DES and AES.

For *packet length correlation*, the packets of one flow can be correlated if they are distinguishable from their lengths. To prevent this, all packets should have the same length, or random length by adding random-length padding bits at each relaying node. For ` *correlation*, an adversary may attempt to deduce the forwarding path by observing the transmission time of a node and its neighbors. The adversary makes use of the fact that the nodes usually relay packets after short delay and based on first-received-first-transmitted basis. Changing packets' appearance at each hop cannot prevent this correlation because it depends on the packets' sending times and not the content. To obfuscate the temporal relationship between the transmissions of consecutive hops, each node can delay relaying the packets for a random amount of time and buffer/reorder packets. Moreover, fake packets can make time correlation nearly impossible because the random nature of the channel access in MAC protocols introduces randomness to the transmissions' times. Sending fake and real packets also confuses the adversary and makes time correlation infeasible even if there is only one event transmission.

For *fake and real source nodes unlinkability*, if an adversary could locate a fake source node, he should not gain any information about the location of the corresponding real source node. This linkability is infeasible because each real source node sends its packets through multiple fake sources, each fake source node serves multiple real sources, and the distance between a fake source node and the real source is random. If the distance between a fake source node and the real one is fixed or has a minimum number of hops (d_{min}), the adversary can figure out the relative location of the real source node or conclude that it cannot be in the fake source node's d_{min}-hop neighbors. What also makes this linkability infeasible is that the adversary sees all the transmissions of a cloud random because he cannot distinguish between fake and real packets. Moreover, even if the adversary eavesdrops on both the real and fake source nodes, he cannot link them because the packets look different. The packets of a pair of real and fake source nodes sent at different times are uncorrelated because pseudonyms are unlinkable, and thus even if the adversary could correlate a real and fake source nodes pair in one occasion, he will not benefit from this conclusion in the future.

For *source node and Sink unlinkability*, if an adversary eavesdrops on a source node and the *Sink*, he cannot link the packets. This property is achievable in our scheme because the packets at the source and *Sink* look completely different for the adversary. For *identity and source node unlinkability*, the *Sink* has to know the real identity of a source node to know the location of the sensed data, but an adversary should not know the real identity or link a packet to its source node. In our scheme, the source node uses dynamic pseudonyms that can be linked to the real identity only by the *Sink*. Using one identity or a fixed group of pseudonyms is not secure because if the adversary could link the identity to its sensor node, he can infer the source node's location each time the node sends an event.

For *cloud shape and source node unlinkability*, if a strong adversary could trace a part of a cloud or all the cloud, he cannot infer any information about the source node's location. For example, if a cloud is circle shaped and the source node is located at the center, the adversary can gain some information about the source node's location by tracing a part of the cloud. In our scheme, this linkability is infeasible because clouds are irregular and changeable, and some nodes may belong to multiple clouds at the same time, which creates an overlapped and complex merged cloud.

For *merged-cloud splitting* attack, the adversary tries to reduce the size of a merged cloud, e.g., to reduce the anonymity set. In our scheme, the traffic of individual clouds is indistinguishable because a cloud's packets do not have any data that refer to the cloud, and thus the adversary cannot split a merged cloud or even identify the boundaries of the individual clouds. Cloud merging can increase the anonymity set without extra overhead, e.g., if two clouds each with n_c nodes are merged, the anonymity sets of the individual clouds are n_c but the anonymity set of the merged cloud is $2n_c - n_o$, where n_o is the number of nodes belonging to the two clouds.

For *packet back tracing attack*, it is unlikely that the adversary will continuously receive event packets from a source node because packets are sent through different fake source nodes which can be far from each other. What also complicates this attack is that event packets sent from a real or fake source node at different times are uncorrelated. Moreover, even if the adversary could capture the same packet at different relaying nodes, he cannot correlate the packets. Even if the adversary could trace back packets to a fake source node, he cannot locate the corresponding real source node due to the *fake and real source nodes unlinkability*. Actually, fake packets can make tracing packets from a fake source node to the real one nearly impossible. When a node sends an event packet, any neighboring node can be the receiver and it is infeasible to figure out the next-hop node. This is because it is impossible to know the owner of the packet's pseudonym and more than one neighbor send packets either fake or event. The adversary cannot also infer the direction to the real source node by following the

movement of the fake packets because the packets are sent after random delay.

For *packet-replay* attack, the adversary tries to replay old packets repeatedly in order to observe the traffic patterns of packet forwarding, e.g., to figure out the network topology to locate source nodes. This is infeasible because the adversary cannot compute fresh pseudonyms and the nodes drop packets if they cannot recognize their pseudonyms. For *packet sending rate* analysis, the adversary attempts to make use of the fact that the nodes near of hotspots send more packets than the nodes far away to locate hotspots. Even with changing the packets' appearance at each hop, the adversary can still analyze the packet sending rate. Our scheme uses fake packets to camouflage the nodes that are close to pandas with the other nodes in the cloud in such a way that makes this spot indistinguishable.

For *event packets flow recognition* attack, the adversary attempts to recognize the flow of real packets to identify the source node or at least a small area around it. For example, from Fig. 8, if the adversary could recognize the flow of the real packets from node B to F, he can deduce that the panda cannot be in the region between B and F and reduce the anonymity set. In our scheme, the event and the fake packets are indistinguishable and the adversary cannot correlate an event packet as it is relayed from the real source node to the fake one. Actually, the adversary can only observe transmissions but the logical interpretation of these transmissions is infeasible. Each relaying node in the route between a real source node and the *Sink* can correlate the packets of two hops but it has no idea about the flow in the other hops, i.e., to trace an event packet, the adversary has to compromise all the relaying nodes between the real source node and the *Sink*.

Event unobservability means that the adversary cannot know whether pandas are sensed or not. This property is more important in other applications such as military applications because the adversary can know whether the network operator (enemy) could observe his soldiers. However, this property is not important in habitat monitoring application especially when the network is large and exhaustive search for pandas is infeasible. Moreover, achieving this property requires extreme energy cost due to sending dummy packets periodically. In our scheme, the adversary may know that pandas are detected, but he cannot know the exact locations of the pandas or at least a small area where he can search for them.

Routing-based privacy preserving schemes use privacy metric called safety period which is the number of packets the adversary has to capture in order to move from the *Sink* to a source node. Stronger privacy protection can be achieved with increasing the safety period. This metric is not accurate because it measures the best case when the adversary starts from the *Sink*, but if the adversary captures a packet at any relaying node, the safety period decreases. For example, from Fig. 2, if the adversary starts from the *Sink*, the safety period is five packets, but if he captures the transmission at R_3, he can reduce the safe period to only two packets. Moreover, if pandas spend some time in one location, the adversary can capture enough packets to locate pandas even if the safety period is large.

We use information-theoretic metric, called entropy [34], to measure the privacy protection provided by our scheme. The entropy of identifying the real source node in a cloud is given in Eq. 1, where P_i is the probability that node i is the source node, n_c is the number of nodes in the cloud, and $\sum_{i=1}^{n_c} P_i = 1$. The entropy characterizes the adversary's uncertainty about the location of the source node in a cloud. The maximum entropy (or the maximum anonymity level) can be achieved when the probabilities P_i pursue uniform distribution, i.e., when the adversary believes that all the nodes in the cloud have the same probability to be the real source node or $P_i = 1/n_c$. In this case, the source node is perfectly hidden in the cloud and the adversary cannot reduce the anonymity set. The maximum entropy (En_M) for individual and merged clouds is given in Eq. 2, where n_S is the number of source nodes in the cloud.

$$En = -\sum_{i=1}^{n_c} P_i \cdot \log_2(P_i) \quad (1)$$

$$En_M = -\sum_{i=1}^{n_c} \frac{n_s}{n_c} \cdot \log_2\left(\frac{n_s}{n_c}\right) = \log_2\left(\frac{n_c}{n_s}\right)^{n_s} \quad (2)$$

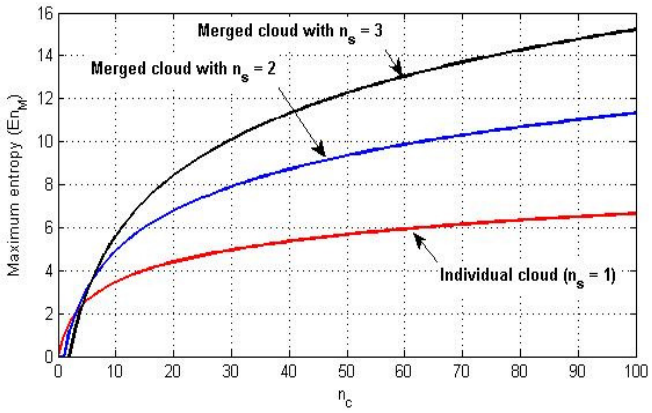

Fig. 12: The maximum entropy versus n_c.

Fig. 12 shows the relation between the maximum entropy and the number of nodes in an individual or merged cloud at different number of source nodes. The figure demonstrates that the increase of n_c increases the maximum entropy of individual clouds, but increasing n_c above 50 has little impact on the maximum anonymity but certainly increases the number of fake packets. It can also be seen that when n_c is above eight, the maximum anonymity of the merged clouds outperforms the individual clouds, which indicates that locating the n_s source nodes of the merged cloud is more difficult than locating the one source of the individual cloud.

Initially, the adversary does not have any information about the source node's location, and thus the probability that a node is the source is uniformly distributed, or $P_i = 1/n$ for all the nodes and n is the number of nodes in the network. In order to locate a source node, the adversary can calculate the probability that a node i is the source using Eq. 3, where γ_i is the average event-packet sending rate of node i. Obviously, without fake packets, the adversary

can locate hotspots because P_i of the nodes at hotspots are much larger than those of the other nodes. Fake packets can be used to prevent the adversary from reaching this conclusion. With fake packets, P_i is given in Eq. 4, where x_i is the average fake-packet sending rate of node i. P_i is nearly the same for all the nodes in the cloud if $(\gamma_i + x_i)$ is nearly the same, and thus the maximum entropy can be achieved and the adversary cannot gain any information from the inconsistency in the nodes' event packet sending rates, which can preserve the hotspot location. In this way, fake packets are used to add noise to the nodes' packet sending rates to conceal the hotspot in a larger area. Therefore, the only way to find pandas may be exhaustive search, which is infeasible if the cloud size is adequate.

$$P_i = \frac{\gamma_i}{\sum_{i=1}^{n} \gamma_i} \tag{3}$$

$$P_i = \frac{\gamma_i + x_i}{\sum_{i=1}^{n_c} (\gamma_i + x_i)} \tag{4}$$

Since not all the nodes in the geographic area of a cloud send fake packets, the adversary may overestimate the anonymity set if he does not have full view to the traffic in the cloud. For example, if a cloud has n_c nodes that send fake and real packets and the cloud's geographic area has n_g nodes, where $n_c \leq n_g$, the actual anonymity set is n_c but the anonymity set seen by the adversary is n_g-n_d, where n_d is the number of nodes the adversary knows that they do not send fake packets and $n_d \leq n_g$-n_c. The amount of information the adversary can gain from knowing that n_d nodes do not send fake packets can be expressed by the degree of anonymity (D) given in Eq. 5. The minimum degree of anonymity is given in Eq. 6 when the adversary can identify all the nodes that do not send fake packets. Fig. 13 shows the degree of anonymity versus n_d at different values of n_g and n_c. We can see that the increase of n_d decreases the degree of anonymity, and the increase of n_g regardless of n_c increases the degree of anonymity.

$$D = 1\text{-}\frac{En_M - En(X)}{En_M} = \frac{En(X)}{En_M} = \frac{\log_2 (n_g - n_d)}{\log_2 n_g} \tag{5}$$

$$D_{min} = \frac{\log_2 n_c}{\log_2 n_g} \tag{6}$$

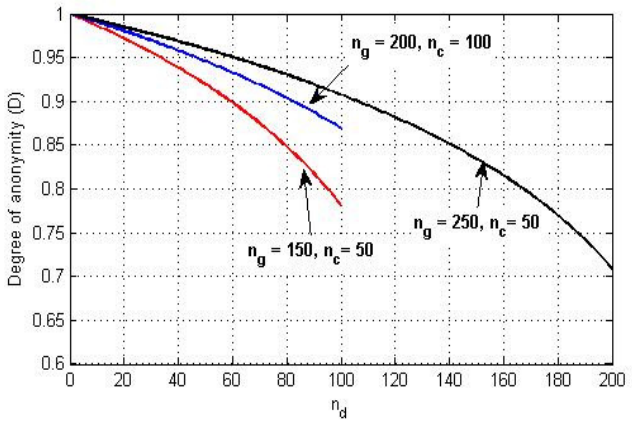

Fig. 13: The degree of anonymity versus n_d.

6.1.2 Simulation Results

We have built up a discrete and event-based simulator to evaluate the effectiveness of the *Hotspot-Locating* attack and the privacy protection of our scheme and *routing-based* schemes. 4000 nodes are uniformly randomly deployed over 3500 m × 3500 m network field, and the *Sink* is located at the center. The nodes' radio transmission radius is 50 m, and the monitoring devices' overhearing radius is (ξ × 50) m. The network has one hotspot that is randomly located and fixed during each simulation run, and the number of source nodes in the hotspot is 30. The number of monitoring devices is $\mathcal{N}_m$, the event transmission rate is 1/30 seconds, and h{⋯} is six at the source node.

Using these network parameters, the average number of neighbors is 9.83 and less than 1.1% of the nodes are weakly connected with fewer than five neighbors and less than 0.03% of the nodes are not connected to the *Sink*. For phantom routing, the number of hops of the random walk (h_w) is four and eight which correspond to 248.76 and 503.69 meters on average. The adversary calculates the nodes' packet sending rates over periods of eight minutes. The results are averaged over 100 runs and each run is performed for five simulation hours. Each run ends when the adversary locates the hotspot or the simulation time expires. The hotspot is located when the adversary becomes at 50 m or less. The simulation parameters are summarized in Table 1.

Table 1: Simulation parameters.

Parameter	Value
Number of nodes	4000
Network size	3500 m × 3500 m
Number of hotspot	1
Number of sensor nodes in the hotspot	30
A sensor node's transmission range	50 m
Adversary's hearing range	ξ × 50 m
Sink location	Center
Sensor nodes and the hotspot	Uniformly randomly distributed
The number of monitoring devices	$\mathcal{N}_m$
Event transmission rate	1/30 seconds

We consider two metrics called the *detection probability* and the *false positive probability*. The detection probability is the probability that the adversary can locate the hotspot during the simulation time. It is measured by the number of times the adversary could locate the hotspot to the total number of runs. The false positive probability is the probability that the adversary falsely identifies an area as a hotspot. It is measured by the number of times the adversary falsely identifies an area as a hotspot to the total number of times the adversary suspects that an area is a hotspot. The decrease of the detection probability and the increase of the false positive probability are indicators for providing high privacy protection for the hotspot. We are also interested in evaluating the impact of the number and overhearing radius of the monitoring devices on the detection and false positive probabilities.

The simulation results given in Tables 2 and 3 demonstrate that the false positive probability decreases and the detection probability increases when the monitoring devices' overhearing radius increases. This is because the adversary can monitor more nodes and collect more accurate traffic information. This is also true when the number of monitoring devices increases. It can also be seen that the weak adversary who has few monitoring devices with small overhearing radius will very likely locate the hotspots in the shortest path and Phantom schemes. This is because the shortest-path scheme does not preserve location privacy and the Phantom scheme cannot prevent packet correlation and conceal traffic analysis information. The slight improvement in the location privacy protection with increasing h_w is because of adding little randomness to the network traffic.

In our scheme, the powerful adversary who has a large number of monitoring devices with large overhearing radius will not locate hotspots. We found that in the runs that the adversary could be close to the cloud, he could not get information about the location or the direction of the hotspot in the cloud. The few times the adversary could locate the hotspot were random. Therefore, what an adversary can do is to exhaustive search the cloud.

Table 2: False positive probability.

Scheme	$\mathcal{N}_m$	4			8		
	ξ	1	2	4	1	2	4
Shortest path		0.21	0.17	0.09	0.13	0.08	0
Phantom	$h_w = 4$	0.25	0.16	0.1	0.11	0.06	0.02
	$h_w = 8$	0.31	0.22	0.17	0.2	0.1	0.05
Our scheme		1	0.97	0.92	0.96	0.91	0.89

Table 3: Hotspot detection probability.

Scheme	$\mathcal{N}_m$	4			8		
	ξ	1	2	4	1	2	4
Shortest path		0.7	0.79	0.91	0.83	0.92	1
Phantom	$h_w = 4$	0.4	0.46	0.6	0.49	0.72	0.8
	$h_w = 8$	0.31	0.42	0.58	0.44	0.69	0.79
Our scheme		0	0.043	0.1	0.05	0.13	0.21

6.2 Energy Cost

As we have discussed earlier, using cryptosystems is necessary to prevent packet correlation, and using fake packets can boost source nodes' location privacy preservation. To reduce the overhead, our scheme uses energy-efficient cryptosystems, including hash function and symmetric key cryptography, and avoids the extensively energy-consuming asymmetric-key cryptography. From [35, 36, 37], Table 4 gives the consumed energy for sending/receiving one bit and computing the cryptographic operations required for our scheme. We can see that the hashing and symmetric key encryption/decryption operations consume low energy comparing to pairing operations. However, pairing operations are used only one time in the network lifetime because keys can be permanently stored after they are computed due to the static nature of the network topology.

Since the *Sink* has more computational and energy capabilities than the sensor nodes, the nodes in the route between a fake source node and the *Sink* encrypt the packets but the *Sink* removes the encryption layers instead of using encryption and decryption operations at each node. The overhead can be further reduced by encrypting the packets at some nodes instead of all the nodes in the route. Unlike [38] where pseudonyms are pre-computed and stored, pseudonyms do not require large storage space or computational power in our scheme, because they can be computed by the efficient hashing operations.

Comparing to *global-adversary-based* schemes, our scheme uses fake packets much more efficiently by sending them only if there is an event instead of periodically. Moreover, fake packets are sent only in the active cloud instead of flooding the entire network, and cloud merging can reduce the number of fake packets. Although our scheme requires more cryptographic operations than *global-adversary-based* schemes, these operations consume much less energy than transmitting/receiving packets, as indicated in Table 4. From [39], the required energy for transmitting 1KB of data over 100 m consumes as much energy as executing three million instructions using a processor with100 MIPS.

Table 4: The energy cost.

Cryptosystem		Consumed energy
Hash functions (per byte)	SHA-1	0.76 μJ
	HMAC	1.16 μJ
	MD5	0.302 μJ
Symmetric-key encryption and decryption operations (per byte)	AES	1.21 μJ
	IDEA	1.47 μJ
	DES	2.08 μJ
Pairing operation		25.5 mJ
Transmit/receive one bit		0.72 /0.81 μJ

7. Conclusion and Future Work

In this paper, we have introduced a novel attack to locate source nodes in WSNs, called *Hotspot-Locating*, which uses a realistic adversary model. We have also proposed a source location privacy-preserving scheme that creates a cloud of fake packets around the source node, varies traffic routes, and changes the packets' appearance at each hop. We have shown that even if the adversary does not have global view to the network traffic, he can locate hotspots using a limited number of monitoring devices and simple traffic analysis techniques. Our simulation and analytical results have demonstrated that *routing-based* schemes cannot preserve the location privacy of hotspots because they are not designed to conceal traffic-analysis information. Moreover, our scheme can provide strong protection against *Hotspot-Locating* attack with much less energy cost comparing to *global-adversary-based* schemes.

In our future work, we will try to locate hotspots with low false-positive probability using computer-based image recognition algorithms in addition to employing traffic-analysis techniques. The existing algorithms usually recognize an image's objects by identifying their boundaries. We will use these algorithms to locate hotspots in the traffic-pattern image created by the traffic analysis techniques.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", IEEE Communications Magazine, vol. 40, no. 8, pp. 102-116, 2002.

[2] I. Akyildiz, W. Su, Y. Sankarasubramanian, and E. Cayirci, "Wireless sensor networks: a survey, computer networks", Computer Networks, vol. 38, pp. 393-422, 2002.

[3] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, and et al., "A line in the sand: A wireless sensor network for target detection, classification, and tracking", Computer Networks, vol. 46, pp. 605-634, 2004.

[4] "WWWF-the conservation organization", http://www.panda.org/.

[5] Star News, Panda poaching gang arrested, Shanghai Star Telegram, April 2003.

[6] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source location privacy in sensor network routing", Proc. of IEEE International Conference on Distributed Computing Systems (ICDCS'05), pp. 599-608, Columbus, Ohio, USA, 6-10 June 2005.

[7] A. Perrig, R. Szewczyk, D. Tygar, V. Wen, and D. Culler, "Spins: security protocols for sensor networks", Wireless Networks, vol. 8, no. 5, pp. 521-534, 2002.

[8] M. Shao, Y. Yang, S. Zhu and G. Cao, "Towards statistically strong source anonymity for sensor networks", Proc. of IEEE INFOCOM'08, pp. 51–59, Phoenix, Az, USA, April 2008.

[9] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao, "Towards event source unobservability with minimum network traffic in sensor networks", Proc. of ACM WiSec, pp. 77–88, Alexandria, Virginia, USA, April 2008.

[10] B. Hoh and M. Gruteser, "Protecting location privacy through path confusion", Proc. of IEEE/CreateNet International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm), pp. 194–205, Athens, Greece, September 5 - 9, 2005.

[11] M. Reed, P. Syverson, and D. Goldschlag, "Anonymous connections and onion routing", IEEE Journal on Selected Areas of Communications, vol. 16, no. 4, pp 482-494, May 1998.

[12] M. Mahmoud and X. Shen, "Lightweight privacy-preserving routing and incentive protocol for hybrid ad hoc wireless networks", Proc. of IEEE INFOCOM, International Workshop on Security in Computers, Networking and Communications (SCNC), Shanghi, China, April 10-15, 2011.

[13] M. Mahmoud and X. Shen, "Anonymous and authenticated routing in multi-hop cellular networks", Proc. of IEEE International Conference on Communications (IEEE ICC'09), pp. 839-844, Dresden, Germany, June 14-18, 2009.

[14] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Mask: Anonymous on-demand routing in mobile ad hoc networks", IEEE Transactions on Wireless Communications, vol. 5, no. 9, pp. 2376-2385, September 2006.

[15] Y. Jian, S. Chen, Z. Zhang, and L. Zhang, "A novel scheme for protecting receiver's location privacy in wireless sensor networks", IEEE Transactions on Wireless Communications, vol. 7, no. 10, pp. 3769-3779, October 2008.

[16] Y. Jian, S. Chen, Z. Zhang, and L. Zhang., "Protecting receiver-location privacy in wireless sensor networks", Proc. of IEEE INFOCOM, pp. 1955-1963, Anchorage, Alaska, 6-12 May, 2007.

[17] J. Deng, R. Han, and S. Mishra, "Countermeasures against traffic analysis attacks in wireless sensor networks", Proc. of IEEE/CreateNet International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm), pp. 113–126, Athens, Greece, September 5 - 9, 2005.

[18] C. Ozturk, Y. Zhang, and W. Trappe, "Source-location privacy in energy constrained sensor network routing", Proc. of ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN) in conjunction with ACM Conference on Computer and Communications Security, pp. 88–93, New York, NY, USA, 2004.

[19] H. Wang, B. Sheng, and Q. Li, "Privacy-aware routing in sensor networks", Computer Networks, vol. 53, no. 9, pp. 1512–1529, 2009.

[20] K. Pongaliur and L. Xiao, "Maintaining source privacy under eavesdropping and node compromise attacks", Proc. of IEEE INFOCOM, Shanghi, China, April 10-15, 2011.

[21] Y. Fan, Y. Jiang, H. Zhu, and X. Shen, "An efficient privacy-preserving scheme against traffic analysis attacks in network coding", Proc. of IEEE INFOCOM'09, Rio de Janeiro, Brazil, April 19-25, 2009.

[22] R. Lu, X. Lin, H. Zhu, and X. Shen, "TESP2: Timed efficient source privacy preservation scheme for wireless sensor networks", Proc. of IEEE ICC'10, Cape Town, South Africa, May 23-27, 2010.

[23] K. Mehta, D. Liu, and M. Wright, "Protecting location privacy in sensor networks against a global eavesdropper", IEEE Transactions on Mobile Computing, to appear, 2011.

[24] B. Alomair, A. Clark, and J. Cuellar, "Statistical framework for source anonymity in sensor networks", Proc. of IEEE GLOBECOM, Miami, Florida, USA, 6-10 December, 2010.

[25] K. Bicakci, H. Gultekin, B. Tavli, and I. E. Bagci, "Maximizing lifetime of event-unobservable wireless sensor networks", Computer Standards & Interfaces, vol. 33, issue 4, pp. 401-410, June 2011.

[26] X. Hong, P. Wang, J. Kong, Q. Zheng, and J. Liu, "Effective probabilistic approach protecting sensor traffic", Proc. of IEEE Military Communication Conference (MILCOM), vol. 1, pp. 169–175, Atlantic City, NJ, USA, October 2005.

[27] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models", ACM SIGMOBILE Mobile Computing and Communication Review, vol. 6, no. 2, pp. 28–36, April 2002.

[28] W. Trappe and L. C. Washington, "Introduction to cryptography with coding theory", Prentice Hall, New Jersey, 2002.

[29] D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing", Proc. of Crypto'01, LNCS, Springer-Verlag, vol. 2139, pp. 213-229, 2001.

[30] Y. Zhang, W. Liu, Y. Fang, and D. Wu, "Secure localization and authentication in ultra-wideband sensor networks", IEEE Journal on Selected Areas in Communications, vol. 24, pp. 829–835, April 2006.

[31] X. Cheng, A. Thaeler, G. Xue, and D. Chen, "TPS: a time-based positioning scheme for outdoor wireless sensor networks", Proc. of IEEE INFOCOM, vol. 4, pp. 2685–2696, Hong Kong March 7-11, 2004.

[32] A. Pfitzmann and M. Kohntopp, "Anonymity, unobservability and pseudonymity – a proposal for terminology", Hannes Federath (Ed.), Designing Privacy Enhancing Technologies, Lecture Notes in Computer Science (LNCS), vol. 2009, pp. 1–9, Springer-Verlag, 2001.

[33] S. Jiang, N. Vaidya, and W. Zhao, "Prevent traffic analysis in packet radio networks", Proc. of DARPA Information Survivability Conference and Exposition (DISCEX II'01), pp.1153-1158, June 2001.

[34] C. Diaz, S. Seys, J. Claessens and B. Preneel, "Towards measuring anonymity", Privacy Enhancing Technologies (PET), Springer-Verlag LNCS 2482, pp. 54-68, 2002.

[35] N. Potlapally, S. Ravi, A. Raghunathan, and N. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security protocols", IEEE Transactions on Mobile Computing, vol. 5, no. 2, pp. 128-143, March-April 2006.

[36] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks", IEEE Journal on Selected Areas Communication, vol. 24, no.2, pp. 247-260, 2006.

[37] G. de Meulenaer, F. Gosset, F.-X. Standaert, and O. Pereira, "On the energy cost of communication and cryptography in wireless sensor networks", Proc. of IEEE International Conference on Wireless and Mobile Computing, pp. 580–585, 2008.

[38] Y. Li and J. Ren, "Preserving source-location privacy in wireless sensor networks", Proc. of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, (SECON), pp. 493–501, Piscataway, NJ, USA, June 2009.

[39] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors", Communications of ACM, vol. 43, no. 5, pp. 51–58, 2000.