Providing Source Location Privacy in Wireless Sensor Networks: A Survey

Article *in* IEEE Communications Surveys & Tutorials · January 2013

DOI: 10.1109/SURV.2013.011413.00118

3 authors:

Mauro Conti
University of Padova
349 PUBLICATIONS **6,034** CITATIONS
SEE PROFILE

Jeroen Willemsen
Xebia
1 PUBLICATION **101** CITATIONS
SEE PROFILE

Bruno Crispo
Università degli Studi di Trento and KU Leuven
261 PUBLICATIONS **3,976** CITATIONS
SEE PROFILE

Providing Source Location Privacy in Wireless Sensor Networks: A Survey

Mauro Conti, Jeroen Willemsen, and Bruno Crispo

***Abstract*—Wireless sensor networks (WSNs) consist of numerous small nodes that can sense, collect, and disseminate information for many different types of applications. One of these applications is subject tracking and monitoring, in which the monitored subjects often need protection. For instance, a WSN can be deployed to monitor the movement of a panda in a national park. The panda needs protection from different adversaries, such as hunters and poachers. An adversary might trace the messages in the WSN to find the source node that sensed the panda, with the final aim of killing the panda. Hence the question is: how do we hide the location of the source node from the adversary? This question is relevant in several of the scenarios related to this application, such as patient monitoring and battlefield surveillance. In other words, the problem is to provide privacy to the source node: source location privacy.**

In this paper, we provide a survey of the state of the art in source location privacy. We first discuss the key concepts in source location privacy, such as anonymity, unobservability, safety period, and capture likelihood. Then, we present an overview of the solutions that provide source location privacy within a WSN, in relation to the assumptions about the adversary's capabilities. In particular, we summarize the concepts and solutions, which are categorized based on the core techniques used to provide source location privacy. We mention the limitations of the algorithms as found in the literature, classify the solutions based on their approach, and provide an overview of the assumptions on the adversarial capabilities related to each solution.

***Index Terms*—WSN Privacy, Location privacy, Source location privacy, Context privacy**

I. Introduction

A WIRELESS sensor node is a low-cost autonomous device that consists of sensing, data processing, and communication components. Wireless sensor nodes have limited storage, computing power, and energy supply [1]–[4]. The nodes are organized into a wireless sensor network (WSN), and deployed into an area in which they need to inspect a certain phenomenon. After deployment, the nodes are left unattended in most of the possible applications [5], [6]. A WSN can consist of different types of sensors and serve different kind of applications [7]. In this paper, we are interested in tracking and monitoring applications, such as wildlife tracking and monitoring the movement of doctors and patients in a hospital. The architecture of a WSN in these applications is as follows: nodes monitor an area and look for the presence of a certain type of object of interest, which we call the subject. A subject can be anything, depending on the application, such as a human, an animal or a vehicle. When a node senses the subject, it informs one or more sinks [8], [9]. A sink is a node that has more storage capacity, computation power, and a better power supply. It is able to do a lot of computational tasks that normal nodes are mostly not capable of. The sink collects all the data of the WSN and either sends this to a server, or it allows for the manual extraction of the data. The moment that a node senses the subject is called an event [10], [11].

Let us take the panda hunter game [12]–[16] as an example: Nodes track the location of a panda in an area. A node that senses the panda informs the sink by sending a message that travels via intermediary nodes to the sink. In the meanwhile, a hunter is out to hunt the panda. The hunter traces the messages from the WSN all the way to the source node that sensed the panda, in order to kill the panda. How do we protect the panda from the hunter? A similar problem pops up in other applications, such as: monitoring the patients and doctors in a hospital [17], and tracking friendly soldiers on the battlefield [18], [19]. In order to protect a subject from the adversary, we must hide the location of the source that senses the subject. Hence, we aim at the *source location privacy* (SLP). SLP requires more than confidentiality of the messages exchanged between nodes: SLP requires that the flow of the messages does not give away the location of a source node. In fact, the confidentiality of a message is part of another privacy category, called content privacy. Content privacy focuses on, amongst others, providing integrity, freshness, non-repudiation, and confidentiality of the messages exchanged in a WSN. SLP is part of context privacy, which focuses on hiding the contextual information of a WSN [20]. Context privacy comprises, for instance, hiding the identity and the location of each node, and hiding the traffic flow between the nodes.

Hiding the location of a source node is complicated by the fact that there are many factors that influence the effectiveness of a solution. For instance: nodes that are mobile require different SLP solutions compared to scenarios in which nodes are static. Another example of an influencing factor is the type of adversary that a source node needs to be protected from: an adversary that can compromise nodes poses different requirements than an adversary that cannot. Another factor, related to the adversary, is whether the adversary has the capability to view all traffic within the WSN or just a part of it. The complexity and variety of the scenarios are the reason for a large set of studies towards solutions that provide SLP.

Manuscript received June 11, 2012; revised October 6, 2012.

M. Conti is with University of Padua, IT (e-mail: conti@math.unipd.it).

J. Willemsen is with Vrije Universiteit Amsterdam, NL (e-mail: jeroenwillemsen2001@gmail.com).

B. Crispo is with University of Trento, IT (e-mail: crispo@disi.unitn.it).

Digital Object Identifier 10.1109/SURV.2013.011413.00118

a) Contribution: In this paper, we provide a thorough survey of the state of the art in SLP. We first describe the different concepts used in SLP. Then, we look at the different adversarial capabilities considered in the literature. Next, we categorize and discuss the solutions that have been developed and published in the literature. Our objective is to provide an overview of the different solutions in relation to the type of adversary they should provide SLP against.

Our work is in that sense unique as, to the best of our knowledge, there are no surveys that focus entirely on SLP in WSNs. When we compare our work to the existing body of research, then we see the following. Rios et al. [21] provide a set of considerations regarding SLP in WSNs, yet do not aim to deliver a survey. Their work provides insight in some of the existing solutions that provide SLP and in some of the limitations of a WSN. Li et al. [12] provide a survey in which they first describe different types of privacy, such as data privacy, SLP, and sink location privacy. Then, they provide a discussion and comparison of a set of solutions, and categorize the solutions based on the type of privacy the solutions deliver. A part of their work focuses on SLP and provides insight in only a few of the existing solutions that provide SLP. Another work, similar to [12], comes from Aivaloglou et al. [22]. They provide an overview similar to the one of Li et al. [12], but do not cover half of the solutions that Li et al. [12] cover. Hence, there is no updated comparison of the different solutions provided in the literature. Our work fills this gap, by providing a classification and description of the solutions in the literature that provide SLP in a WSN. Therefore, the contributions of this paper are three-fold: First, we provide insight in how an adversary can be classified. Second, we discuss each of the solutions that provide SLP. Third, we provide insight in the fragmentation of the adversarial capabilities as assumed by the authors in the literature. More precisely, we present an overview in which we show for each solution what the authors of the solution assume the adversary is capable of.

Note that we do not mean to say that SLP makes content privacy obsolete. On the contrary: we still need data confidentiality to hide the contents of a message from an adversary as we do not want an intercepted packet to tell anything about the whereabouts of a subject. A WSN would also not function correctly if we do not have data freshness, data integrity, and an acceptable availability of the nodes. Therefore, we would like to recommend not to focus on SLP alone, but also on content privacy when designing solutions for a WSN. However, content privacy is out of scope for this paper. We would like to recommend the reader to look at the work of Chen et al. [23] for a survey in content privacy.

b) Organization: The remainder of this paper is organized as follows. Section II provides background information, such as the different concepts used in the literature to describe SLP, and a description of the adversarial capabilities found in the literature. Then, Section III discusses the different categories that we used to classify the solutions, after which we discuss each of the solutions within these categories. Then, Section IV provides a summary of all the different solutions that we discussed. Finally, in Section V we give some concluding remarks.

II. Background

In this section, we describe the different concepts used to describe source location privacy (Section II-A), and the different adversarial capabilities (Section II-B).

A. Source Location Privacy

The concept of source location privacy (SLP) in relation to *context privacy* was first described by Ozturk et al. [14] based on the panda hunter game, as we described in the introduction of our paper. The essence of the panda hunter game is that the hunter can only use the traffic flow to track the panda. Kamat et al. [20] were the first to formalize the source location problem based on the panda hunter game. Both Ozturk et al. [14] and Kamat et al. [20] argue that it is up to the routing strategy to hide the location of the subject, with respect to an adversary that only traces packets through the WSN. Kamat et al. [20] use two metrics to quantify the privacy delivered by the routing algorithms. The first metric, shared by Ozturk et al. [14], is the safety period of a routing protocol. The safety period is the number of messages a source node sends out before the adversary locates the subject, given the movement strategy of the adversary. The second metric is the capture likelihood of a routing protocol, which is the probability that the hunter can capture the subject within a specified time period, given the adversaries' moving strategy. Other authors, such as Spachos et al. [24], Wei-ping et al. [25], Zhang et al. [26], Yang et al. [27], Chen [28], Jhumka et al. [29], and Yao et al. [30] use similar metrics. Spachos et al. [24] claim that the safety period ends either when the adversary finds the subject or when the subject has moved away from the source node. SLP is breached when an adversary finds the subject, while it is at the source.

In essence, providing source location privacy means one has to counter traffic analysis. The problem of countering traffic analysis is much older than the problem of SLP in a WSN. Countering traffic analysis brings us to the context of the Internet. One of the first solutions to counter traffic analysis on the internet was based on *anonymity* and *untraceable routes*, as described by Chaum [31] as "*the solution to the traffic analysis problem*". Chaum focused on anonymous and untraceable electronic mail based on mixing messages, and using digital pseudonyms. This solution is now known as a MIX-net [32]. Seven years later, Chaum provided a stronger concept by introducing "*Unconditional sender and recipient untraceability*" [33]. This provides a much stronger and more verifiable version of anonymity as both sender and receiver were made completely anonymous within a group. Again, this concept was focused on the Internet. This solution is now known as a DC-Net [32]. MIX-nets and the underlying idea of anonymity were also discussed and evaluated in other works. We only mention three of them here. Pfitzmann et al. [34] studied the concepts of anonymity, evaluated MIX-nets, and proposed a few extensions to it. They discussed anonymity sets in which a sender and a receiver of a message are part of a larger group of anonymous participants that might have been the sender or the receiver of the message. The sets of potential senders and receivers are all anonymous and alike to the adversary,

so that the adversary cannot couple a sender and a receiver to an observed message. In a later work, Pfitzmann et al. [35] provided a version of a MIX-net that was more suitable for low band-width connections, such as an ISDN-network. Kesdogan et al. [36] provided a more in-depth analysis of anonymity. They said that a solution provides deterministic anonymity if the solution always provides an anonymity set with more than one member. They provided a probabilistic approach on creating anonymity sets and provided a solution based on this relation, called stop-and-go-MIX as an alternative to MIX-nets.

Another solution against traffic analysis came from Reed et al. [37]. Reed et al. [37] who made use of the principle of anonymous connections for the Internet, in which the traffic between sender and receiver would be obfuscated and routed through several different stations in between. This principle was realized by them with the concept of onion routing. Later on, Reiter et al. [38] revisited anonymity and provided a set of degrees of anonymity ranging from absolute privacy to provably exposed. They provide a solution called CROWDS. In CROWDS, a sender and a receiver blend into a crowd of other communicating parties. They argue that their CROWDS solution is the first solution that makes directly use of the notion of anonymity sets.

Concepts related to anonymity, *pseudonyms* and other concepts used by Chaum [31], [33], Kesdogan et al. [39], Reiter et al. [38], and Reet et al. [37] have been formalized by Pfitzmann et al. [40], [41], again with a focus on the Internet. Their formalization is actually an extension of their study regarding anonymity in [34]. Pfitzmann et al. [40], [41] define anonymity as "*the state of being not identifiable within a set of subjects, the anonymity set*" [40], [41]. Anonymity conceals the identity of a node as the identity could belong to any of the nodes within the set. Pfitzmann et al. [40], [41] show that anonymity gets stronger if the anonymity set grows. Next, they describe the concept of sender and receiver anonymity. In *sender anonymity*, a message is not linkable to the sender and a sender is not linkable to a message. In *receiver anonymity*, a receiver is not linkable to any message and a message not to a particular receiver. A weaker form of anonymity is found in *relationship anonymity*, in which a sender and receiver are not linkable. Von Ahn et al. [32] use a different approach towards anonymity. They give the notion of full anonymity and a weaker form of *K-anonymity*. *K*-anonymity refers to the strength of the anonymity given an anonymity set of *K* members.

Pfitzmann et al. [40], [41] also introduce *unlinkability*. Unlinkability means that two things cannot be related by an adversary based on a-priori knowledge or knowledge gained after a run of the system. Anonymity in terms of unlinkability means that a physical node cannot be related to a logical identifier.

Pfitzmann et al. [40], [41] describe *unobservability*. In contrast to anonymity, which only protects the identity of a subject, unobservability makes any communicating party indistinguishable from any other communication party within the unobservability set. Unobservability can be split into *sender unobservability*, *receiver unobsevability*, and *relationship unobservability*. Sender unobservability means that it is not noticeable whether a member of the unobservability set transmits a message. Receiver unobservability means that it is not noticeable whether a member of the unobservability set receives a message. Relationship unobservability means that it is not noticeable whether anything is sent out of any potential sender to any potential receiver. Hence, unobservability is much stronger than anonymity.

Serjantov et al. [42] show that relying on the size of the anonymity set as a metric can be a pitfall. This is mostly because some solutions still allow for attacks which reduce the anonymity-set based on information leaked by the technical realization of the solution. Instead, Serjantov et al. [42] recommend to use an approach based on *information entropy*. Information entropy is introduced by Claud Shanon [43], shows how much information is produced on the average for each letter of a text in a language. The concept of information entropy can be translated to anonymity in a WSN. The basic idea of this approach is that SLP can be evaluated as follows. Given a message *M*, we have a probability distribution of its possible senders and receivers, as perceived by the adversary. Some nodes may not be the sender at all, so they have a probability of 0 to be the sender. The same holds for the role of receiver. Nodes that have a different probability than 0 form the anonymity set. Serjantov et al. [42] introduce the *effective size of an anonymity probability*, based on the assigned probability distribution. The effective size of an anonymity probability can range from 0 to a maximum threshold based on the number of nodes. If the value is 0, then there is no anonymity. If the value is at the maximum threshold, then perfect anonymity is provided. The calculation-specific details are out of scope of our paper and we recommend the reader to read the paper [42] for further details.

The concepts from Pfitzmann et al. [40], [41], Serjantov et al. [42], and von Ahn et al. [32] found their way to WSNs as various authors applied them to the SLP problem in a WSN. As an example, Wadaa et al. [8] use anonymity as a basic evaluation of their solution to provide anonymity in a WSN with multiple sinks. Fan et al. [44] use the concept of anonymous networking, which is the same as the relationship anonymity of Pfitzmann et al. [40]. Fan et al. [45] use the metrics of the anonymity set and privacy entropy to calculate the quality of the source anonymity. There are also many works, in which various forms of information entropy are used as a measurement for SLP [10], [11], [45]–[53].

Information entropy is not always a good measurement for source and destination location privacy according to Doomun et al. [52]. In fact, they argue that information entropy does not capture the information that stems from path analysis, which provides hints about the location of source and destination nodes. Hoh et al. [54] measure SLP by using the expectation of *distance error* for a path, which measures how accurate an adversary can estimate a subjects position. Several authors, i.e. Ortolani et al. [55], [56] use a modern event-based version of unobservability and anonymity, in which they provide lower bounds for the unobservability set and the anonymity set.

Shao et al. [57] and Alomair et al. [58] use the concept of statistical source anonymity and *event unobservability* to provide SLP. Statistical source anonymity means that the source cannot be found with traffic analysis nor with statistical

tests. Event unobservability means that a real event is not distinguishable from a fake event. A fake event is basically a dummy message that is created by another node than a source node. The fake messages from anonymous nodes are indiscernible from real messages, so that an adversary does not know which message to follow.

Note that concepts as anonymity and unobservability might not be enough to protect a subject's location. In fact, SLP can be endangered if an adversary is able to find previous locations of a subject, which allows the adversary to create a trajectory and forecast the next position. Zhao et al. [59] call this the *continuous privacy threat*. A similar view is expressed by Mehta et al. [15], [16], as they argue that a global eavesdropper could be more interested in the trajectory of the monitored subjects.

There are also different views and definitions on how to name or position SLP inside the categories of privacy. SLP is mostly seen as part of context privacy [60] (or contextual privacy [13], [61], [62]). Context privacy often includes *identity privacy*, *location privacy*, *timing privacy*, and *route privacy*. Identity privacy ensures that the identity of a node remains hidden. Location privacy concerns hiding the location of a node. Timing privacy aims at hiding the temporal relation between incoming and outgoing traffic. Route privacy covers hiding the traffic flow. Kamat et al. [63] define hiding the traffic flow as spatio-temporal privacy. They also mention *temporal privacy*, which hides timing information related to packet creation. Li et al. [12] provide a slightly different taxonomy to describe context privacy. They mention SLP as *Data Source Location Privacy*. They divide context privacy in location privacy and temporal privacy. Location privacy is then further divided in *data source location privacy* and *base station location privacy*.

Another approach is given by Shaikh et al. [19]. They define SLP as part of *network level privacy* and call it *sender node location privacy*. In sender node location privacy no intermediary nodes, other than the direct neighbours of the source, know the location of the source. Shaikh et al. [19] argue that this is only achievable when *full network level privacy* is achieved. Full network level privacy comprises three elements. First, a node's identity is kept private, which they call *identity privacy*. Second, the route between source and sink must be kept private, which they call *route privacy*. Third, the data within the packets must stay private, which they named *data (packet) privacy*. Again, note that data (packet) privacy [23], [64], [65] is out of scope of our paper. Li et al. [12] use the concept of data source location privacy, which is similar to SLP and also part of context privacy. The last concept we mention, is given by Pai et al. [17], [66] as they use the concept of *transactional confidentiality*, which is essentially the same as context privacy [62].

One last note on fake and dummy messages: authors in the literature often mean the same thing when they mention fake or dummy messages. In general, dummy messages and fake messages obfuscate the traffic of actual events. We will clarify the exact use of a mentioned fake message or dummy message when we describe the solutions.

B. Adversarial Capabilities Based on Threat Models

In this section, we provide an overview of the adversarial capabilities that are listed in the several threat models considered in the literature for SLP. In particular, we classify the different threat models based on four distinctive properties: the behaviour of an adversary (Section II-B.1), the view of the network an adversary has (Section II-B.2), the resources and devices an adversary has (Section II-B.3), and the information a network exposes to an adversary (Section II-B.4). Let us discuss each of the properties.

1) Adversarial behaviour

The behaviour of the adversary is the criteria on our first classification. There are three distinctive classification criteria which can be used:

- Whether the adversary is internal or external (access level),
- Whether an adversary is active or passive (interference), and
- Whether the adversary is semi-honest or dishonest (compliance).

Let us first discuss the properties and then look at the relations between them.

The first classification criterion for the adversarial behaviour classification is whether an adversary is internal or external. Li et al. [12] describe the criterion related to the adversarial behaviour regarding the internal versus the external adversary. An internal adversary can compromise a node within the WSN whereas an external adversary cannot do this. We call this criterion the access level criterion as an internal adversary has access to components and an external adversary does not have access.

The second classification criterion for the adversarial behaviour classification is whether an adversary is active or passive. A passive adversary does not actively influence the nodes or the traffic between the nodes, whereas an active adversary does alter the traffic or the behaviour of the nodes. We call this criterion the interference criterion as an active adversary does interfere with the WSN and a passive adversary does not interfere. Some authors use different names for the interference criterion. For instance, Jiang et al. [67] refer to passive attacks as non-invasive attacks and to active attacks as invasive attacks. Note that being an active adversary does not exclude the adversary from executing passive attacks. However, being a passive adversary does exclude the adversary from active attacks.

Many threat models in the literature comprise a passive adversary. As an example, let us take the threat model of Yang et al. [27], as it covers most of the passive attacks. The first passive attack is eavesdropping [27], in which the adversary listens to the traffic and tries to inspect the content that is exchanged between the nodes. A little more advanced is the hop-by-hop-trace attack [27], in which an adversary follows the traffic between the nodes to get to the source node, which is the origin of the traffic. The hop-by-hop-trace attack is also called the path tracing attack [68]. Even more advanced are the rate monitoring attack and the time correlation attack [27]. In a rate monitoring attack, the adversary looks for nodes with a higher transmission rate, as these nodes are probably

closer to the source or the sink. In a time correlation attack, the adversary observes the correlation in transmission time between a node and its neighbour to find the route that a packet travels to the sink. Very similar to the time correlation attack, is the identity-analysis attack as described by Yang et al [27]. In the identity-analysis attack an adversary analyses the frequently used identities to see whether there is a relation between the nodes. Majeed et al. [69] describe the timing analysis attack. The timing analysis attack is used to monitor transmission patterns to discern sensitive information, such as the structure of the network. Hong et al. [70] show that timing analysis attacks can also be used to correlate incoming to outgoing traffic. There are also specific passive attacks to locate a node, such as the angle of arrival (AoA) indication, which shows the direction of a transmitting node [13]. Another example of a passive attack that is used to locate a node is the received signal strength (RSS) indication, which allows an adversary to estimate the distance between itself and the node [13].

There are a few threat models in the literature that comprise an active adversary. Two examples of an active adversary are described by Shaikh et al. [19] and Hong et al. [70]. Shaikh et al. [19] describe an adversary that is active, yet also uses eavesdropping and traffic analysis attacks. The active attacks of the adversary comprise injecting false packets into the network traffic and dropping real packets. Hong et al. [70] describe an adversary that is capable of compromising a node to block traffic and to monitor traffic surrounding the node. See the work of Cardenas et al. [5] and Chen et al. [23] for a more complete list of active attacks.

The third classification criterion for the adversarial behaviour classification is whether an adversary is semi-honest or dishonest (or malicious). This criterion is also mentioned by Mingjun et al. [6]. A semi-honest adversary follows the protocols within the WSN to ensure that it remains undetected as an adversary, while a dishonest adversary does not comply with the protocols within the WSN. We call this criterion the compliance criterion as a semi-honest adversary is compliant with the protocol and a dishonest adversary is not compliant. Note that the attackers that we found in the threat models that are relevant to this survey follow the protocols and are therefore semi-honest.

Figure 1 summarizes the relations between the different criteria. Figure 1 has three vertical blocks containing the three different criteria regarding to the adversarial behaviour: A: Access level, B: Interference, C: Compliance. Each of the criteria has two members as described above. Interference has the members A.1 Internal, depicted as an internal adversary, and B.1 External depicted as an external adversary. Access level has the members B.1 Active and B.2 Passive depicted as an active and a passive adversary. Compliance has two members C.1 dishonest and C2. semi-honest. An internal adversary (A.1) can be both active (B.1) and passive (B.2). Take for example an adversary that can compromise a node. The adversary can either passively read the contents of the node (A.1:B.2) or actively use the node to influence the traffic (A.2:B.1). An external adversary (A.2) can be both active (B.1) as passive as well (B.2). An external passive adversary (A.2:B.1) only eavesdrops the network, whereas

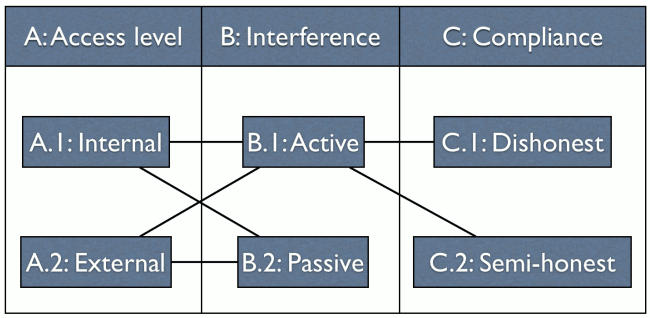


Fig. 1. Relations between adversarial behaviour classification criteria.

an external active adversary (A.2:B.2) could also inject fake packets into the network. In our opinion, only an active adversary (B.1) can either be dishonest (C.1) or semi-honest (C.2), as a passive adversary does not actively contribute to the protocols used within the WSN.

2) View of the Network

The view that an adversary has at the network is the criteria on the second classification. There are two types of adversaries when it comes to their view of the network: the global adversary and the local adversary.

A global adversary has a full view of the network [15], [16]. The adversary often uses its own network with sniffers or a powerful laptop [71] to eavesdrop all communications. Some authors, like Ouyang et al. [72] call a global adversary a laptop class adversary.

On the other hand, a local adversary has a local view of the network, based on eavesdropping nodes with sniffers and other techniques [19]. Authors like Ouyang et al. [72] call a local adversary a mote class adversary. A local adversary is sometimes not alone and might collaborate with others. For instance, Jhumka et al. [29] describe a threat model that includes multiple local adversaries which are collaborating by sharing information on the topology and the traffic within the WSN. The collaborating adversaries together have a multi-local view. We use the classification of a multi-local view not only for multiple collaborating local adversaries, but as well for adversaries that have a very powerful device to oversee a substantial part of the network as defined by Li et al. [73]. Li et al. [73] call such an adversary a semi-global adversary.

Note that solutions that defend against a local adversary can often not cope with a global adversary, whereas solutions designed against a global adversary can often cope with a local adversary [74].

3) Amount of Resources

A threat model often describes the amount of resources an adversary has in terms of available memory and computational power [19]. Threat models, such as the one of Kamat et al. [13] and the one of Shaikh et al. [19] also define whether an adversary is device rich or not. Device rich means that the adversary has antennas, spectrum analysers, and other additional equipment to listen to the signals from the nodes, and locate the position of the nodes. We do not use the adversaries' amount of resources nor its device richness as classification criteria as they have minor influence on the classification process. First,

most authors consider a similar bounded adversary if it comes to the resources of the adversary, so the amount of resources is not a key differentiator. Second, when two adversaries share the same attack capabilities and view of the network, then they often have the same state of device richness.

4) Information Exposed by the Network

The knowledge an adversary has about the network and the implemented privacy protocols is the criteria on the third classification. The knowledge of the adversary about the network varies in the literature, as authors make different assumptions on what an adversary knows about the WSN [19]. Let us mention a few examples: Kamat et al. [13] define an informed adversary, who knows the methods being used at the nodes within the WSN; Wei-Ping et al. [25] assume that the adversary knows the location of the sink. However, most of the knowledge an adversary could have is not explicitly formulated within the threat models, but can be found implicitly within the formulated solutions. We identified the following information that an adversary can know about the WSN and its nodes: the topology of the WSN, the distribution of the events, the location of a specific node, the specific identities of the nodes or the range of identities that are used within the WSN, the algorithms and protocols that are used in the WSN, the methods that are applied in each of the nodes and the keys used for encryption. Note that Pai et al. [17] describe a method that allows an adversary to understand what type of wireless sensor nodes are used within the WSN. However, we keep node recognition out of the scope of our paper as it does not directly influence the effectiveness of the solutions.

III. Solutions for Providing Source Location Privacy

In this section, we first introduce the categories that we used to classify the solutions (Section III-A), after which we discuss the solutions within the categories (Sections III-B - III-L). We end this section with a set of notes on flooding in Section III-M.

A. Categories for the Solutions

In order to classify the solutions, we need a set of categories for them as well. In this section, we define a set of categories based on the core techniques used within each of the solutions. Then, we show how these categories relate to the adversaries' view of the network. More specifically, we first discuss eleven different categories namely: the random walk, geographic routing, delay, using dummy/fake data sources, cyclic entrapment, location anonymization, cross-layer routing, separate path routing, network coding, limiting the node detectability, and others. We then map these eleven categories onto the adversaries' view of the network in Figure 2.

1) **Random Walk:** The aim of the "random walk" approach is to have packets follow a random route through the network. The random walk should make a packet's path look completely random to an adversary in order to counter the adversaries traffic analysis and hop-by-hop-traces. Solutions in this category use either a technique derived from the random walk, as described by Ozturk et al. [14], or a technique that results in a similar pattern, such as rumor routing from Braginsky et al. [75] and routing through randomly selected intermediary node from Li et al. [3], [76].
   The following solutions are part of this category: angle-based multi-intermediate nodes selection [76], the directed random walk [30], the greedy random walk [77], the location privacy support scheme [61], the mules-saving-source protocol [73], opportunistic routing [24], phantom routing [13], phantom routing with locational angle [25], phantom single-path routing [13], random routing [78], the random routing scheme [79], routing through randomly selected intermediary node [76], the self-adjusting directed random walk [26], and a combination of different solutions [26].
2) **Geographic Routing:** Solutions in this category use the physical location of the nodes together with geographic routing algorithms to route packets through the WSN [19]. Geographic routing algorithms take the position of a node, its neighbours, and the sink into account, in order to route a packet from the source to the sink. The solutions in this section make use of additional methods, such as the usage of synonyms, encryption, and random intermediary node selection to hide the flow of the traffic against a local adversary.
   In this category, we identified the following solutions: sink toroidal region routing [80], identity, route and location privacy [19], and reliable identity, route and location privacy [19].
3) **Delay:** This category consists of solutions that alter the flow of the traffic as follows. Each node buffers incoming packets and holds a packet for a random time before forwarding it. As a consequence, nodes alter the chronological order of the packets: they send the packets in a different order than how they received them. As the chronological order of the packets change, so does the traffic pattern. The change of the traffic pattern makes it hard for a local adversary to track the traffic to the actual source.
   In this category, we identified the following solutions: probabilistic reshaping [70], the extended version of probabilistic reshaping [70], and rate controlled adaptive delaying [10].
4) **Using Dummy Data Sources:** Solutions in this category introduce dummy traffic to obfuscate real traffic. The aim is that an adversary should no longer be able to see which part of the traffic is real, and which part is fake. In this category, we found the following solutions: aggregation-based source location protection scheme [81], A-real and A-fake [58], the cloud-based scheme for protecting source location privacy [60], constant rate [57], the dynamic bidirectional tree [28], distributed resource allocation algorithm [46], dummy wake-up scheme [68], fake sources 1 [29] and fake sources 2 [29], fitted probabilistic rate [57], the group algorithm for fake-traffic generation [82], globally optimal algorithm [72], the heuristic greedy algorithm [49], mixes [49], the optimal filtering scheme [62], periodic

collection [15], persistent fake source routing [13], the probabilistic algorithm [72], proxy-based filtering [27], SECLOUD [52], short-lived fake source routing [13], source simulation [15], the timed efficient source privacy preservation [83], the timing analysis resilient protocol [69], tree based filtering [27], the trusted computing enabled heterogeneous WSN [84], unobservable handoff trajectory [55], and the zigzag bidirectional tree [28].

5) **Cyclic Entrapment:** The solutions in this category aim at confusing the adversary by shaping the traffic between nodes in cyclic patterns. A local adversary, which tracks the traffic between the nodes, will travel in circles without finding the actual source [15], [16].
   This category consists of two solutions: cyclic entrapment method [85], and information hiding in distributing environments [86].
6) **In Network Location Anonymization:** Solutions in this category hide either the identity or the location of a node.
   The following solutions are part of this category: the anonymous communication scheme [79], anonymous path routing [87], the cryptographic anonymity scheme [88], destination controlled anonymous routing protocol for sensornets [50], hashing based ID randomization [89], max query aggregation [90], phantom ID [91], the probabilistic destination controlled anonymous routing protocol for sensornets [50], the reverse hashing ID randomization [89], and the simple anonymity scheme [88].
7) **Cross-layer Routing:** With cross-layer routing, the nodes use the beacon frames, which are normally only used for network maintenance, to share information on sensed events. Local adversaries, which normally only listen to the network level packets, miss part of the information exchange, and do not find the real source.
   This category has two solutions: the cross-layer solution [47] and the double cross-layer solution [47].
8) **Separate Path Routing:** A local adversary often needs multiple packets along the same route to track the actual source. The solutions based on separate path routing make sure that the packets travel via different non-intersecting paths from source to sink. Using separate paths leads to less packets per path, which delays the local adversary in its tracking, or even makes the adversary unable to track the actual source at all.
   This category consists of random parallel routing [92], weighted random stride routing [92], and weighted random stride routing towards a global viewing adversary [92] .
9) **Network Coding:** In network coding, each node cuts up its message and sends it out in smaller pieces. These pieces are then forwarded via different routes towards the sink.
   This category consists of the solutions from Fan et al. [44], [45], as they propose to use network coding to provide SLP.
10) **Limit Node Detectability:** This category consists of solutions that limit the transmission power of the nodes to make them harder to detect.

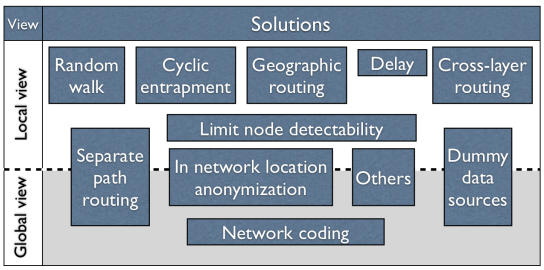


Fig. 2. The different categories mapped towards the adversaries' view of the network.

We have identified the following solutions in this category: anti localisation by silencing [93], context-aware location privacy [94], hidden anchor [95], hyberloc [51], lowering radio transmission power [96], and multi cooperator power control [97].

11) **Others:** This category consists of a collection of solutions which are designed for specific scenarios in which normally no SLP is provided. One of these solutions comes from Shao et al. [98] and is focused on providing privacy in a data-centric WSN, which is normally not designed to provide SLP. Another solution in this category is the preserving hop-distance computation protocol by Mingjun et al. [6]. This solution allows two nodes to compute the distance between them, without giving away their actual locations to an adversary. Usually, protocols that compute the hop-distance already disclose the locations of different nodes, which makes it very easy for an adversary to get an overview of the topology.

Figure 2 shows how the categories (for the solutions in this section) can be projected on the adversaries' view of the networks (discussed in Section III-A.2). In particular, the solutions of a category that is positioned in the upper white block only defend against local adversaries. The solutions of a category positioned in the lower grey block defend against global adversaries. The solutions, that overlap the border between the lower and the upper block, contain both solutions that defend against local adversaries and solutions that defend against global adversaries.

Solutions that use the random walk, cyclic entrapment, geographic routing, delay, cross-layer routing or that limit the detectability of the node are only able to defend against a local adversary. Network coding based solutions, on the other hand, are able to defend against a global adversary. Solutions that use dummy data sources, in network location anonymization, or separate path routing are not strictly defending against either a local adversary or a global adversary. The effectiveness of these solutions against a global or a local adversary highly depend on the implementation. That is why some of the solutions using these techniques can counter a global adversary, while others only defend against a local adversary.

Now that we have shortly introduced each of the categories, let us move on to discussing the solutions within each of the categories. Note that we only discuss specific details of a solution, such as relative power consumption and the effectiveness of the solution, when the literature provides them. We use a similar approach when it comes to the data

privacy aspects of a solution. We only mention methods for encryption, authentication, and message integrity, when they show how a solution provides better SLP.

B. Random Walk Based

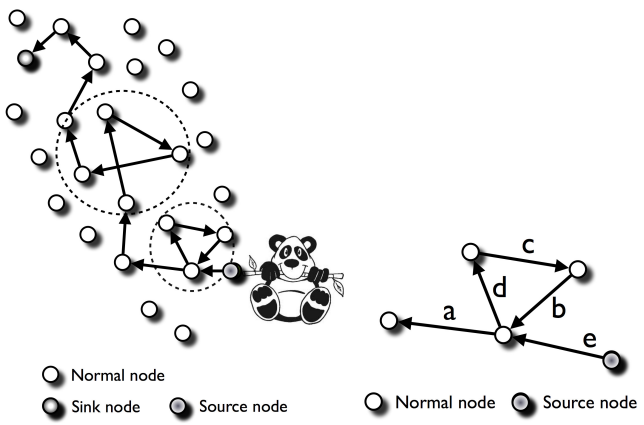
In this section, we focus on solutions that are based on the random walk approach. We first look at the history of the random walk in Section III-B.1. Then, we look at the solutions in the literature that use the random walk (Section III-B.2), and we summarize our findings in Section III-B.3.

1) History and Introduction

The random walk should make a packet's path look completely random to an adversary to counter traffic analysis and hop-by-hop-traces. The randomness of the route comes from the fact that nodes forward the packet to one of their neighbours that they selected randomly [14]. The randomness of selecting a neighbour is based on the forwarding probability of each of the neighbours, as described by Wei et al. [25]. The forward probability of a neighbour is the probability that a node forwards a packet to that specific neighbour. The higher the forward probability of a neighbour, the more likely a node will forward a message to that specific neighbour.

Servetto et al. [99] are one of the first that studied the random walk method for a WSN in depth. They use the random walk for traffic load-balancing within a WSN. Later, Ozturk et al. [14], [100] use the random walk as a technique to provide SLP. Ozturk et al. [14] show that a pure random walk cannot provide SLP effectively. In a pure random walk, the forward probability is uniformly distributed among the neighbours of the node. A node can therefore send a packet to any of its neighbours, including those that have already forwarded the packet. If a node forwards the packet to a neighbour that already forwarded the packet, then together they create a cycle as we show with two examples in Figure 3. Figure 3(a) shows an overview of a random walk with two cycles. The dashed circles show the locations of the cycles. Note that the route the packet has travelled in Figure 3(a) is already quite optimal for a pure random walk. In Figure 3(b) we see one instance of a cycle. Let us assume that multiple packets take the same path as shown by the arrows within Figure 3(b). The source sends the packets one after another, creating a stream of packets. Each packet starts at the source and then takes the following hops consecutively: *e*, *d*, *c*, *b*, and *a*. If an adversary uses the hop-by-hop-trace attack, then it tracks the packets one or more hops at the time, depending on its hearing range. Now, if a cycle is small enough to fit in the adversaries' hearing range, then the adversary might trace more hops at the same time. So in our example of Figure 3(b), the adversary does not have to travel via the path *a*, *b*, *c*, *d*, *e*, but it can go directly from *a* to *e*, which saves the adversary time. In fact, Kamat et al. [13], [20] show the following: on average, when a local adversary does a successful hop-by-hop-trace attack, then the adversary only has to travel one-fifth of the distance that the packets travel during a pure random walk.

Note that the cycles in a pure random walk do not only ease the tracing for an adversary, they also increase the energy consumption of the nodes within the cycle as these nodes need to receive and transmit the packets.



(a) An instance of a pure random walk with its cycles from source to sink. The dashed circles show the locations of the cycles.

(b) A cycle close to the source.

Fig. 3. Cycles introduced by a pure random walk.

The solutions described in Section III-B.2 hardly make use of the pure random walk. Instead, they include an enhanced version of the random walk that does not introduce cycles in the route of a packet. The enhanced versions of the random walk are often augmented with additional techniques to improve the safety period of the solution. Take the location privacy support scheme by Kang et al. [61] for example: it incorporates dummy messages and an enhanced random walk.

2) SLP via Random Walk Based Solutions in the Literature

Ozturk et al. [14], [100] introduce the phantom routing scheme (PRS), which is the first solution that uses the random walk to provide SLP. PRS consists of two phases. The first phase is the random walk phase and the second phase is the flooding phase. Authors like Wang et al. [92] also refer to PRS as random-walk-based phantom routing, due to the application of the random walk. Ozturk et al. [14], [100] do not use the pure random walk, but an improved version, called the directed walk. In a directed walk, each node divides its neighbours into two sets, which are in opposite directions of one another. For instance, one set contains neighbours that are positioned North and West of the node and the other set contains neighbours positioned South and East of the node. When a source needs to send a message, it chooses one set randomly and sends the message to a neighbour chosen at random out of that set. Each intermediary node that receives the packet from a neighbour out of one set, will randomly select a neighbour out of the opposite set to forward the packet to. For instance: if an intermediary node gets the packet from a neighbour out of its South-East set, then it forwards the packet to one of its neighbours in the North-West set.

The forwarding of the packet stops when the packet has travelled h hops or when the packet cannot be forwarded anymore into the same direction. Ozturk et al. [14], [100] call the node that has the packet at the end of the random walk the phantom source. After that, the phantom source starts the flooding phase of PRS. The phantom source starts to flood the packet towards the sink. When an adversary starts tracking the source of the flooding, it will find the phantom source instead of the real source. See Section III-M on the use of flooding.

Kamat et al. [13], [20] use a similar approach to PRS, but instead of using flooding for the second phase, they use single path routing algorithms, such as shortest path routing. The combination of the random walk together with single path routing is often referred to as the phantom single-path routing scheme (PSRS). Kamat et al. [13], [20] take the directed walk from Ozturk et al. [14], [100] and expand it as follows. First, they rename the original directed walk to sector-based directed random walk and provide a technique to divide the neighbours of a node into the required sets. Next, Kamat et al. [13], [20] introduce a second version of the directed walk, which they call the hop-based directed random walk. The hop-based directed random walk relies on the hop-distance between a node and the sink. The hop-distance consists of the smallest number of hops a message has to make to get from the source to the destination. One hop in this case is the successful transmission of the message from one node to another. In the hop-based directed random walk, each node checks the hop-distance between itself and the sink and between its neighbours and the sink. Then, every node divides its neighbours into two sets. One set contains neighbours which have an equal or smaller hop-distance to the sink than the node. The other set contains neighbours that have a larger hop-distance to the sink than the node. When a source node senses an event, it randomly selects one of the two sets and sends the packet to a randomly selected neighbour out of the set. For the next h hops, the message is forwarded by intermediary nodes using the corresponding set. The hop-based directed random walk ends after the h hops or when an intermediary node does not have any members in the corresponding set. Kamat et al. [13], [20] call the node that holds the packet at this point the phantom source. The phantom source uses single path routing algorithms, such as shortest path routing, to route the message towards the sink.

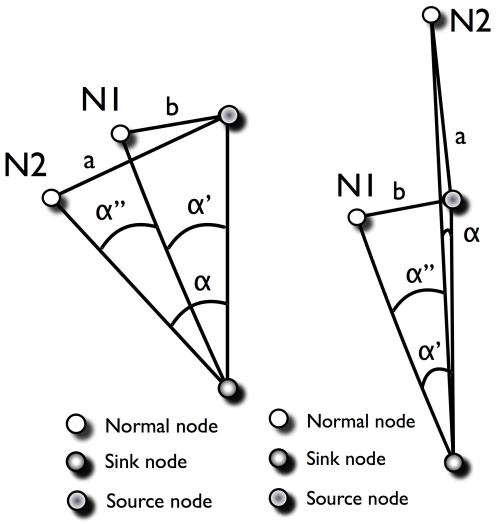
PRS and PSRS have received a lot of attention in the literature. On the one hand, Mehta et al. [15] consider PSRS to be quite effective against local adversaries, while on the other hand, authors find weaknesses in PRS and PSRS. Wang et al. [92] show that the directed random walk provides poor SLP, as it is easy for an adversary to trace back to the source. Shaikh et al. [19] mention that both PRS and PSRS do not provide identity privacy. They also mention that PSRS is slightly more memory consuming than PRS, but does save a lot of power as they do not use flooding. According to them, PSRS does not provide strong SLP due to re-usage of the routing paths and exposure of direction information. Lightfoot et al. [80] confirm the exposure of direction information of PSRS. Suarez et al. [68] argue that PSRS cannot work against a global adversary. Deng et al. [101] show that a random walk pattern does complicate a rate monitoring attack, but that it is still vulnerable to a time correlation attack. Park et al. [91] mention that PRS cannot be successful without hiding the *ID* of every node.

Zhang [26] shows that terminating a random walk prematurely has a negative impact on the safety period. A random walk terminates prematurely if it does not travel the minimum amount of hops. She [26] shows that the sector-based directed random walk provides a longer safety period than the hop-based directed random walk, even though both methods have their weaknesses. The hop-based directed random walk becomes less random towards the sink, as there are less alternative paths around the sink. The sector-based directed random walk is sensitive to the position where the subject is located. If the subject is sensed close to one of the borders of the WSN, and the walk is directed towards the border closest to the subject, then the random walk cannot complete its walk of h hops. Other works do not really review the methods, but just provide a summary of PRS and PSRS [12], [25], [30], [50], [52], [74], [76], [79].

Zhang [26] also introduces an improvement of the sector-based directed random walk, with the introduction of the self-adjusting directed random walk (SADRW). SADRW can deliver longer random walks than the sector-based direction random walk. The potential difference in length of the random walk comes from the fact that SADRW's random walk can continue where a sector-based directed random walk would end. In SADRW each node divides its neighbours into four sets depending on their cardinal direction: neighbours to the North, neighbours to the South, neighbours to the East and neighbours to the West. When a source starts a random walk, it randomly picks a neighbour out of one of the four directional sets and sends the packet to it. For the next h hops, each intermediary node forwards the packet to one of its neighbours in the same direction. If an intermediary node receives the packet and it cannot forward the packet into the same direction, then it chooses a new direction to forward the packet to. This new direction is from here on referred to as the second direction. The intermediary node chooses the second direction of the packet in such a way that the packet moves further away from the source, based on the information stored in V. From there on, the intermediary nodes forward the packet into the second direction until the packet has travelled a total of h hops. It can be the case that the packet arrives at the edge of the WSN without travelling the required h hops as the packet reaches an edge of the WSN. If the packet is at the edge, then the following happens. The last node that receives the packet should see whether the packet has travelled at least a minimum of i hops, where i is a fixed parameter to tune the minimum length of the random walk and $h \geq i$. The random walk terminates if the message has travelled at least i hops. Otherwise, the node selects one of the two remaining directions, to which the packet has not been forwarded yet, as the third and last direction to forward the packet to for the remaining hops.

Wei-Ping et al. [25] introduce phantom routing with a locational angle (PRLA) as an improvement of PSRS. In PRLA, the random walk is based on the inclination angle between a node and its neighbour towards the sink.

The inclination angle is shown in Figure 4, where nodes *N1* and *N2* are two neighbours of the source. The source can reach neighbour *N2* via hop a, and neighbour *N1* via hop b. The inclination angles between nodes *N1*, *N2*, and the source are depicted as follows. The inclination angle between a source and neighbour *N2* is α. The inclination angle between neighbour *N1* and the source is α', whereas the inclination angle between neighbour *N1* and neighbour N2 is α''. In Figure 4(a), it is clearly visible that $\alpha \geq \alpha'$ and that the length of hop $a \geq$ the length of hop b. In other words: the larger the



(a) Instance of neighbours *N1* and *N2*, with $\alpha \geq \alpha'$ (b) Instance of neighbours *N1* and *N2*, with $\alpha' \geq \alpha$

Fig. 4. The inclination angles between nodes towards the sink.

inclination angle, the longer the distance is between a node and its neighbour from the perspective of the sink. Note that the inclination angle can be deceiving for an actual measurement of the distance. Take Figure 4(b) for instance. In Figure 4(b) the inclination angle between *N2* and the source towards the sink (α), is smaller than the inclination angle between *N1* and the source towards the sink (α'). So from the perspective of the sink, *N1* should be further away from the source than *N2*, but *N2* is actually further away from the source than *N1*. This is easy to see by measuring the length of *a* and *b* as shown in Figure 4(b) where the length of $a \geq$ the length of *b*.

Wei-Ping et al. [25] assume that the local adversary remains close to the sink where it starts to track the source. They intent to confuse the adversary at the sink by choosing a random inclination angle for each packet routed from the source to the sink. Choosing a random inclination angle for each packet does not only complicate the local adversary in its attempt to correlate an intermediary node to a source. In fact, it also creates separate routing paths, which make the hop-by-hop-trace attack harder. PRLA works as follows. First, every node calculates the inclination angle between itself and its neighbours. Then, every node uses the inclination angle to calculate the forward probability of each of its neighbours. The higher the inclination angle of a neighbour, the higher the forward probability of that neighbour will be. When a source senses an event, it uses the forward probability of the neighbours to select a neighbour with a certain inclination angle and sends the packet to the selected neighbour. Then, each intermediary node needs to forward the packet to a neighbour with the same inclination angle for at least *h* hops. The last node that is not able to forward the packet with the same inclination angle or that gets the packet after *h* hops becomes the phantom source and forwards the packet to the sink, using a single path routing strategy. This solution has a longer safety period than PSRS and PRS and it is also studied by authors such as Chen et al. [28], Rios et al. [21], and Kazatzopoulos et al. [102].

Yao et al. [30] provide another improvement of the random walk by introducing the directed random walk (DROW). DROW relies on the hop-distance between a node and the sink. In DROW, each node checks its own hop-distance to the sink and the hop-distance of its neighbours to the sink. Each node chooses as its parents the neighbours with a lower hop-distance towards the sink than itself. When a node receives or generates a packet, it forwards the packet to a randomly selected parent. DROW works at its best when intermediate nodes have multiple parents and pick a different parent for each packet that they forward. DROW provides a longer safety period than PRS; DROW is more energy efficient than PRS, and it provides a lower latency than PRS. Spachos et al. [103] and Deng et al. [101] refer to DROW as multi-parent routing. Deng et al. [101] show that DROW does complicate the rate monitoring attack, but does not defend against a time correlation attack. Wang et al. [104] mention that the direction information retrieved from the packet headers helps the adversary to find the source of a message.

Xi et al. [77] introduce the greedy random walk (GROW), which comprises two improved random walks. Both random walks meet up at a receptor-node. A receptor-node can be any node within the WSN other than the source, and the sink. One random walk starts from the sink and goes to a randomly chosen receptor-node. The other random walk starts from the source and meets the first random walk at the receptor-node. Then, the receptor-node uses the path established by the random walk from the sink to the receptor-node to route the packet from the source to the sink. Note that the receptor-node is also often referred to as the phantom source [94].

Both the random walks in GROW are improved by using a Bloom filter [77] in the packets, to keep track of whether a node has forwarded the packet already. A node first adds its identity to the Bloom filter, before it forwards the packet. A Bloom filter is an efficient data structure that is used to test whether an element is a member of a set. In this case, a node can test whether one of its neighbours is a member of the set of nodes that have forwarded the packet. Bloom filters can give false positives, which means in this case that a node would have been visited by the packet, but it was not visited yet. Bloom filters cannot give any false negatives. A node only forwards the packet to a neighbour that did not get the packet yet.

Xi et al. [77] call this solution a greedy solution as it tries to cover as much of the WSN during the random walk, without creating any cycles. Note that the solution has some open ends: Xi et al. [77] mention that there should be a minimum random walk length, but do not provide any methods to implement this. Second, they note that additional information, such as the geographic location of the nodes could be used as well if it is available. However, they do not provide any methods to do so. The solution is partially based on the concept of rumor routing from Braginsky et al. [75], in which a sink sends out multiple queries that traverse the WSN with a random walk. The idea is that the query should hit one of the sources that sensed the event and then inform the sink. The greedy random walk is more energy efficient than PRS and has received quite some attention in the literature. Yao et al. [30] state that the random walk used in GROW is inefficient at creating a safe

distance between the receptor-node and the source. Wei-Ping et al. [25] point out that the latency is unstable due to the usage of two random walks. Shaikh et al. [19] points out that finding a trustworthy receptor-node is essential in the case of an internal adversary, yet there is no criteria for selecting a proper receptor-node. Rios et al. [94] point out that the receptor-node can still be too close to the sink or source. Li et al. [76] argue that GROW is not feasible for large scale networks. Lightfoot et al. [80] and Yao et al. [30] give a similar point of view. Lightfoot et al. [80] also mention that the messages leak too much information to an eavesdropping adversary. Other authors [12], [21], [74], [78], [81] just provide a summary of GROW.

Li et al. [76] introduce routing through a randomly selected intermediary node (RRIN) as an improvement over PRS. RRIN, in contrast to PRS, does not leak any directional information via its packets. RRIN works as follows. A source node first randomly selects an intermediate node and sends the packet to it. Next, the intermediate node sends the packet to the sink. There are two versions of RRIN. In the first version of RRIN, the location of the intermediary node must be at least a minimum distance away from the source and be normally distributed within the rest of the WSN. In the second version of RRIN, each node in the WSN has an equal probability to be the intermediate node of any given source node. The first version of RRIN has the same latency and power consumption as PRS, but a higher safety period. The second version of RRIN consumes much more energy and has a higher delay than PRS, but it does provide an even better safety period. Lightfoot et al. [80] confirm the high energy consumption of the second version as well. The second version of RRIN has one advantage over the first version of RRIN. The second version of RRIN does not give any basis for an adversary to correlate an intermediary node to a source, while the first version of RRIN does give away additional information to a local adversary.

Li et al. [3], [76] suggest two alternative routing strategies in the same work as where they discuss RRIN. The first strategy is called angle-based multi-intermediate nodes selection (AMNS). This routing strategy works very similar to PRLA and it is based on the inclination angle of the source and the intermediary node towards the sink. The only difference with PRLA is that AMNS is not based on random forwarding. In AMNS, the source node chooses the inclination angle and the distance between itself and the final intermediary node. The source node also preselects the intermediary nodes based on the chosen final intermediary node. In the second strategy, called quadrant-based multi-intermediate nodes selection (QMNS), the source chooses the intermediary node based on a geographical lay-out of the WSN. First, the source node *S* splits the WSN up in four geographical areas, which are numbered quadrant 1, 2, 3, and 4. The borders of the different quadrants meet at the sink *D*, as depicted in Figure 5. In this example, the source is located at quadrant 1. Next, the source node takes one of the adjacent quadrants, which are the grey areas in Figure 5, and selects a random node out of the chosen quadrant to be its intermediate node. Then, the source sends the message to the intermediary node, which routes the message to the sink. The different solutions from Li et al. [3], [76], RRIN, AMNS, and QMNS require the source to do some route planning and encapsulation of the routing information within the packet. All intermediary nodes update the routing information within the packet to make sure that an adversary is never able to see were the packet came from, but can only see were the packet is going to.

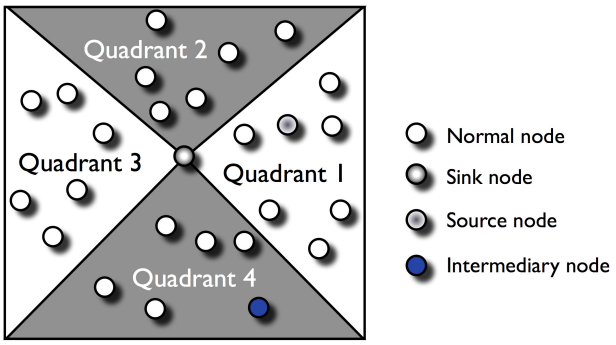


Fig. 5. Source and selected intermediary node in QMNS.

Kang [61] introduces the location privacy support scheme (LPSS). LPSS is inspired by the location privacy routing protocol from Deng et al. [101], which focuses on protecting the sink. LPSS extends the location privacy routing protocol by protecting both the location of the sink and the source. LPSS is based on the gradient of each node, which is a raw estimation of the hop-distance from the node to the sink. LPSS uses the following routing scheme. With probability p, a node sends a packet to a neighbour with the same gradient to the sink. With probability $1 - p$, a node sends a packet to a neighbour with a smaller gradient. Setting the value of p allows one to tune between stronger SLP and lower latency. When a node sends a real packet, it generates a dummy packet with a time to live (*TTL*) counter. A *TTL* counter holds the number of hops a packet is still allowed to make before it is dropped. Every node that forwards a packet with a *TTL* counter decrements the *TTL* counter before it forwards the packet. A node drops the packet when the *TTL* counter reaches zero. This dummy packet is then send to a randomly chosen neighbour, which is more distant to the sink. Nodes continue to forward the packet to a randomly chosen neighbour in a direction away from the sink until the *TTL* counter reaches zero. The dummy packets should equalize the traffic directed towards the sink and directed away from the sink, so that a local adversary cannot make up the location of the sink as well. Luo et al. [79] argue that LPSS will actually attract adversaries towards the real traffic as nodes that forward real traffic will make two transmissions instead of one.

Luo et al. [79] are inspired by several solutions, such as PRS and LPSS, and introduce a combination of three schemes to provide a stronger solution. Their first scheme is the random routing scheme (RRS), which is similar to the routing part of LPSS. In RRS a node divides its neighbours in two sets, which are called the *closer* and the *further* set. The *closer* set contains neighbours that have a shorter hop-distance to the sink. The *further* set contains neighbours that are at an equal or longer hop-distance to the sink. If a node needs to send or forward a packet, it randomly selects a node from the *closer* set. If a

node sends a packet to one of the members of its *closer* set, then the node updates the forward probability distribution of the *closer* set. The forward probability distribution is updated in such a way that each of the members of the *closer* set will get to forward an equal amount of packets. RRS has a much lower latency and energy consumption than the routing part of LPSS, as the messages are forwarded directly towards the sink.

The second scheme of Luo et al. [79] is the dummy packet injection scheme (DPIS), which provides dummy traffic. When a node overhears a transmission from a neighbour from its *further* set, it broadcasts a dummy packet based on probability. The probability of broadcasting a dummy packet is based on the remaining energy a node has. The more energy a node still has, the higher the probability that it broadcasts a dummy packet. A dummy packet contains a *TTL* counter, similar to the dummy packets in LPSS, which decrements at every hop. When a node overhears the transmission of the dummy packet of a neighbour of its *closer* set, then it does the following. The node uses a similar probability function, based on its energy level, to decide whether it accepts the dummy packet or not. If it accepts the dummy packet, it decrements the *TTL* counter and broadcasts the packet again. This process is then repeated until the *TTL* timer reaches zero and the dummy packet gets dropped.

The third scheme from Luo et al. [79] is the anonymous communications scheme (ACS), which hides the real identities of the nodes away from an adversary. This scheme is further explained in Section III-G.2.

Armenia et al. [78] mention random routing (RaRo) as a theoretical example of a solution. RaRo uses the random walk in its purest form to provide SLP. In RaRo each node selects the next intermediary node at random for each incoming packet. Unfortunately, RaRo provides very little SLP, very high latencies, and consumes a lot of energy.

Li et al. [105], [106] introduced a combination (which we abbreviate to C1) of RRIN together with a network mixing ring (NMR). A network mixing ring is composed of multiple nodes, named ring nodes, that form a ring around the sink. These ring nodes are divided into relay ring nodes and normal ring nodes. Relay nodes generate vehicle messages that travel through the mixing ring. These vehicle messages can contain multiple data units. Members of the ring share keys with their direct neighbours to decrypt and encrypt the vehicle messages, so that the vehicle messages look different at each retransmission. Nodes first route a message via RRIN to an intermediary node. This intermediary node then forwards the message to the ring, where the content of the message is loaded into a vehicle message. One example of this protocol is shown in Figure 6, where the source in the right-bottom corner sends two messages via different normal nodes onto the network mixing ring. One of these normal nodes is located to the North of the source and the other node is located to the West of the source. The network mixing ring is the circle of dashed arrows surrounding the sink. The ring relay nodes forward a vehicle message to the sink based on a pre-set probability. The forwarding rate at the ring, together with the probability at the ring relay nodes, balance the delivery rate, SLP, and the power consumption.

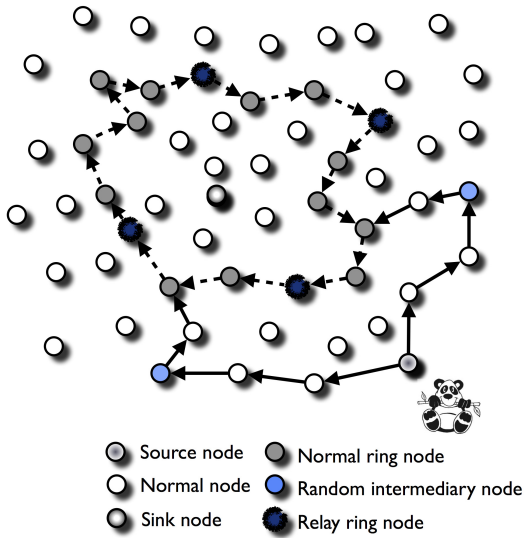


Fig. 6. The packets of a source routed via different intermediary nodes onto the mixing ring.

Some authors shortly refer to this solution or summarize it, such as Chen et al. [28], Spachos et al. [103], and Rios et al. [21].

Another solution in this category is opportunistic routing (OpRo), from Spachos et al. [24], [103]. In OpRo, nodes record their distance towards the sink, and use the distance as the "cost of delivery" (*cod*) metric for transmitting their reports to the sink. The actual route of a packet depends on both the *cod* of every intermediary node towards the sink and the packet error ratio. In particular, the longer the distance between a sender and a receiver and the more noise, the higher the packet error ratio of a transmission will be. When a node wants to send something, it checks for a free data channel and sends a request to send (*RTS*). Nodes that overhear *RTS* and are closer to the sink than the sender of *RTS*, set up a timer before they respond with a confirm to send a clear to send (*CTS*). The duration of the timer depends on the difference of *cod* between the sender and the potential receiver. If the potential receiving node is much closer to the sink than the sending node, then the receiving node uses a very short timer. If the potential receiving node is not that much closer to the sink than the sending node, then the receiving node uses a long timer. When a timer of a potential receiver reaches zero, then the potential receiver sends a *CTS*. If the sender receives the *CTS*, then it transmits the packet towards the receiver. Finally, note that receivers that have a much smaller *cod* then a sender might have a very short timer, but the distance between the sender and receiver will introduce a high packet error ratio. In other words: the noise might cause a corruption of the *CTS* or the *RTS* used between the nodes. The combination of *cod* and the packet error ratio gives way to random routes in case of noisy channels. Note that nodes can also be in a sleeping state, in which they do not respond to an *RTS*. We would also like to mention the work of Hsu et al. [107], providing more details on opportunistic routing within WSNs.

Spachos et al. [103] introduce three techniques to increase the quality of SLP coming from RaRo. The first technique is non-repeating opportunistic routing (NROR). In NROR, every

node keeps a flag to decide whether it is allowed to participate in a transmission. When the flag is false, then the node is allowed to participate. If the flag is set to true, then the node is not allowed to participate. Initially all nodes their flags are set to false. When a node transmits a packet in a time slot, then the flag of the node is set to true for the next time slot. A node will not participate in any transmissions during that time slot. The node sets the flag to false again when the next time slot ends. This forces streams of packets to use different nodes for every hop as packets cannot continuously use the same node as a relay. The second technique from Spachos et al. [103] is opportunistic routing with random delay (ORRD). In ORRD, a node uses a different *CTS* timer mechanism than in standard OpRo. Every potential receiving node adds a random amount of milliseconds to its *CTS* timer when it receives a *RTS*. The third technique from Spachos et al. [103] is opportunistic routing with random relay (ORRR). In ORRR, a sender has to wait for all *CTS*s from the potential relay nodes to come in. Finally, the sender chooses a relay node randomly out of the set of nodes were it got a *CTS* from.

The last solution in this category is the mules-saving-source protocol (MussP), from Li et al. [73]. They assume that the WSN can be divided into cells. They furthermore assume that the adversary oversees a large part of the WSN and that the adversary starts his tracking from the sink towards the source. The adversary starts estimating the location of the source at the beginning of the adversaries' tracking and will update this estimation as the adversary tracks the traffic. Li et al. [73] introduce α-angle anonymity as "*A protocol is α-angle anonymous if the real direction of data source is equally likely distributed in the angle range [β - α, $\beta + \alpha$], where β is the angle of the direction inferred by the attacker based on his observation.*" [73]. The goal of MussP is to provide α-angle anonymity, based on a pre-set α. Li et al. [73] realize this by trafficking the packets from a source to an intermediary node via mobile agents, which they call data mules. A mobile agent is a mobile autonomous program that can traverse a network and be executed on different machines within the network [108]. The data mules traverse the network, collect the packets from the sources and drop them off elsewhere. The solution consists of three phases. In phase 1, multiple sensor nodes sense the presence of a subject. The sensor nodes decide together which node is going to be the source that has to report to the sink. The source selects an inclination angle β similar as to PRLA, which ranges between [α ; -α]. In phase 2, a data mule moves into the cell where the source is part of. All sources that are part of the cell give their data to the mule and the mule continues to traverse the network. A mule can drop packets from a source at a dropping point. A dropping point can be any node that has an inclination angle β towards the sink. If there is no such node, a mule drops the data at a node that has an inclination line close to β. Lastly, in phase 3, the node that was the dropping point in phase 2 now takes the packets it received from the mule and routes them towards the sink.

3) Summary of the Random Walk Based Solutions

This section contains the solutions that use the random walk or an improvement of the random walk to provide SLP. Most of the solutions we discussed combine the random walk with another technique to improve the safety period. Despite the different augmentations and improvements, the random walk only defends against a local adversary. In fact, seven of the fifteen solutions that we found in the literature have documented weaknesses, which shows that the random walk is not always effective. The random walk is one of the early techniques used to provide SLP. The solutions and their adversarial models are summarized in Table I in Section IV.

Note that there are more solutions that provide a random walk pattern with similar results, yet they do not aim to provide SLP. We would like to mention them for completeness: the random data collection scheme form Ngai et al. [109], the random forwarding as introduced by Pai et al. [17], the randomized routing with hidden address from Ngai et al. [110], the secure real-time routing protocol from Ahmed et al. [111], the random direction query(+) as introduced by Dimitriou et al. [112], the random walk from Zeng et al. [113], and table-assisted random walk from Zeng et al. [113].

C. Geographic Routing

In this section, we discuss the solutions based on geographic routing. We first introduce this category and look at the history of geographic routing in Section III-C.1. Then, we discuss the solutions in the literature that use the geographic routing (Section III-C.2). We summarize our findings in Section III-C.3.

1) History and Introduction

Geographic routing makes use of geographic information available within the WSN to route packets from the source to the destination. The geographic information available can vary from the location of the sink to the location of a node or the locations of the neighbours surrounding a node. Geographic routing in a wireless network was first studied by Karp et al. [114]. They used geographic routing as a basis to design a protocol with a low routing protocol message cost, a high packet delivery success rate, and low memory requirements for each of the nodes. Shaikh et al. [19] were one of the first to use geographic routing with the aim to provide SLP. They [19] combine geographic routing together with other techniques, such as encryption, trust management, and dynamic pseudonyms to provide SLP. This category consists of three solutions as described in Section III-C.2: i) identity, route and location privacy [19], ii) reliable identity, route and location privacy [19], and iii) sink toroidal region routing [80].

2) SLP via Geographic Routing in the State of the Art

Shaikh et al. [19] propose identity, route and location privacy (IRL) to provide SLP. In IRL, every node divides its neighbours in four sets based on the position of the sink. Figure 7 shows the geographical spread of each of the sets in relation to the sink. The *front* set is directed towards the sink. The other sets: the *right-backward* set, the *middle-backward* set, and the *left-backward* set, are directed away from the sink.

Nodes calculate the trustworthiness of each of their neighbours based on the successful and unsuccessful interactions they had with them. A node uses the calculations to classify a neighbour as trustworthy, uncertain or untrustworthy. When

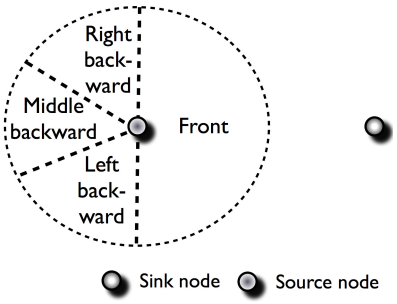


Fig. 7. Region related sets in IRL. Note that the source could also be a normal node.

a source wants to inform a sink, it first checks its *forward* set for trustworthy neighbours. If the *forward* set contains one or more trusted neighbours, then the source sends the packet to one of them. If there is no such neighbour in its forward set, then the source checks all the other sets and randomly picks a trusted neighbour out of one of the sets. If there is no trustworthy neighbour at all, then the source drops the packet. When an intermediate node gets the packet, it first checks whether it already received the packet once. The node drops the packet if it gets the packet for the third time. If a node does not receive the packet for three times, then the node first checks for any trustworthy neighbours in its *forward* set. If there are trusted neighbours in the node's *forward* set, then it sends the packet to one of them. If there is no trusted neighbour in the *forward* set, then the node checks the other sets for a trusted neighbour. The set that contains the previous sender of the packet is checked last. If there is no trusted neighbour in that set as well, other than the one where the node received the packet from, then the node drops the packet. Otherwise the node forwards the packet to a trusted neighbour in one of the sets.

The problem with this forwarding method is that it can introduce cycles. Cycles can lead to additional latency and packet drops. When a source sends a packet, it does not send its own identity in the clear as well. Instead, in order to provide identity privacy, the source adds a random number of equal length to its identity and encrypts the new number with the public key of the sink. The data payload in a packet is encrypted with a key that is shared between the source and the sink. Next, each packet only contains the identity of the sender and the identity of the receiver to make a hop-by-hop trace harder. IRL provides route privacy, location privacy, and identity privacy, this technique is also summarized by Rios et al. [21].

Shaikh et al. [19] introduce reliable identity, route and location privacy (R-IRL) as an extended version of IRL. R-IRL uses multicast, instead of unicast, in order to send a packet to at least r trusted neighbours in the *forward* set. If there are less than r trusted neighbours in the *forward* set, then the node includes trusted neighbours from the other sets in the multicast to get r destinations within the multicast. Using multicast reduces the chance of creating cycles and increases the reliability of the system.

Lightfoot et al. [80] introduce sink toroidal region routing (STaR) as an improvement over RRIN. The problem with RRIN is that it either consumes a lot of energy or that it leaks information on the location of the source. With STaR, both problems are fixed. The idea behind STaR is that the intermediary node will neither be too close to the source nor too far from the source. STaR gives an optimal balance in power consumption and source location privacy. Lightfoot et al. [80] assume a WSN that spans a large area and consists of multiple grids. A grid in this case refers to the topology of the sensor deployment and not to a form of resource sharing within a WSN [115]. STaR uses the notion of a random intermediary node, similar to RRIN. Only this time, the intermediate nodes can only be selected from an area surrounding the sink. This area is called the sink toroidal region or STaR area. Figure 8 shows how this region is distributed. The sink node is in the center of the region. The sink toroidal region begins at r meters from the sink and ends at R meters from the sink. The sink toroidal region spans multiple grids. Note that we zoom in on the sink toroidal region in Figure 8 and that the figure does not display the full WSN.

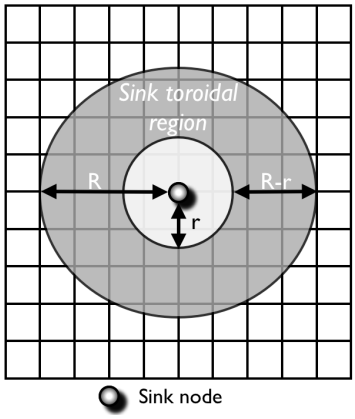


Fig. 8. The Sink Toroidal region.

Every node in the WSN knows the X,Y coordinates of the sink and the values of r and R. When a source wants to send a packet to the sink, it uses these parameters to calculate the X,Y coordinates of a random location within the sink toroidal region. Next, the node sends its packet towards the grid where the intermediary location is located. Once the packet is within the grid that contains the random intermediary location, it is sent to the header node of that grid. The header node of that grid becomes the random intermediary node. It can be the case that the grid does not contain any nodes at all. If the grid of the random intermediary location does not contain any nodes, then the node that currently holds the packet becomes the desired location and its header node becomes the random intermediary node. The random intermediary node then routes the packet to the sink. Each node has a dynamic *ID* which changes over time in order to provide identity privacy. Lightfoot et al. [80] assume that the sink is the only one that knows the location of a node, given its dynamic *ID*. STaR is more efficient than PSR and totally random RRIN. STaR has a similar latency to PSR, but also a higher packet drop rate than PSR.

3) Summary of Geographic Routing Based Solutions

In this section, we discussed three solutions: IRL, R-IRL, and STaR. All three of them use geographic routing as their

base technique. In IRL and R-IRL, to group its neighbours, a node uses the physical location of the sink. In STaR, to route the packets, a node uses the physical area of the sink toroidal region and the physical location of the random intermediary location. All three solutions provide SLP against a local adversary. See Table I in Section IV for an overview of the different solutions and their adversarial models.

Note that there are other solutions that use geographic routing and provide a little SLP against a local adversary, yet they do not aim to provide SLP: Geographic Random Forwarding from Zorzi et al. [116], [117], and the SIGF-family as introduced by Wood et al. [118].

D. Delay: Providing Timing and Temporal Privacy

This section contains the solutions that mainly provide timing and temporal privacy through delay. We introduce this category in Section III-D.1. Next, in Section III-D.2, we look at the solutions in the state of the art that use delay as a core technique. We summarize our findings in Section III-D.3.

1) History and Introduction

The solutions within this section aim to provide timing privacy [70] or temporal privacy [10], [12], [63]. When a solution provides temporal privacy, it provides SLP as temporal privacy makes a timing analysis attack or a hop-by-hop trace attack unfeasible. For instance, temporal privacy provides SLP in case a mobile subject is monitored. If an adversary at the sink uses a timing analysis attack based on the packets near the sink, then it will not be able to find the current location of the subject as the chronology of the reported events is mixed. Timing privacy provides SLP against an adversary that tries to do a hop-by-hop-trace attack. In this case, the adversary can no longer correlate the incoming traffic of a node to its outgoing traffic, as the packets are no longer forwarded in the same order as they were received.

The technique of delaying outgoing transmissions to counter timing analysis has been applied in older solutions, such as the mentioned MIX-nets [31], [34]–[36] and the DC-Nets that stem from Chaum [33]. Hong et al. [70] were one of the first to focus on delay in a WSN to counter timing analysis attacks from an adversary. We discuss their two solutions, probabilistic reshaping [70] and extended probabilistic reshaping [70], together with rate controlled adaptive delaying from Kamat et al. [10] in Section III-D.2 .

2) Solutions that use Delay in the Literature

Hong et al. [70] introduce probabilistic reshaping (PRESH) to counter timing analysis attacks. They assume that every node shares an individual key with its neighbours for encryption and that all packets are padded to have the same size. In PRESH, every node has to incorporate its own identity in an unencrypted header when it sends a packet, while a node can hide the identity of the receiver.

The actual solution works as follows. When a source generates a packet, it transmits the packet directly towards the first intermediary node (or to the sink, if the sink is located directly next to the source). When an intermediary node needs to forward a packet, it delays the transmission of the packet. The duration of the delay of each packet is random, and follows an exponential distribution, which can be set based on the protocol parameter μ. Packets are not sent in the same order as they are received due to the random delay. There can be many transmissions in a neighbourhood of a node during the delay of a packet. So, when a node transmits the delayed packet, the adversary cannot tell where the packet came from, as there are too many other transmissions to which it could have been related. The worst-case scenario is when no other nodes in the area have transmitted anything during the delay of a packet. In that case, the adversary has a 50% chance of making the right guess after the transmission: the node was either a source or an intermediary node. Note that Hong et al. [70] use the notion of an anonymity set in a different way: they state that the size of the anonymity set is equal to the amount of packets in a queue. The setup of the exponential distribution can be used to tune between timing privacy or SLP, and latency.

Hong et al. [70] introduce extended probabilistic reshaping (exPRESH) as an extension to PRESH to counter the worst-case scenario of PRESH. Let us assume that a node has only one packet in its buffer while there are no other transmissions in the neighbourhood, and the node does not generate packets by itself. Then the node is allowed to delay the packet in its buffer again. This process of repeated delaying can continue until a node has delayed the packet for a certain amount of time D. After this time, the node has to transmit the packet. Note that the exponential distribution used for additional delays differs from the exponential distribution used in PRESH, to further confuse the adversary. The parameter D together with the delaying distribution parameters allow to tune between privacy and latency. Note that Hong et al. [70] do not assume anything about the actual routing algorithm. PRESH and exPRESH only contain a delaying strategy and not any routing algorithm. It seems that PRESH and exPRESH have received little attention from other authors. Nezhada et al. [50] only mention PRESH and point to the lack of a routing algorithm.

Kamat et al. [10] introduce rate controlled adaptive delaying (RCAD) to provide temporal privacy. Kamat et al. [10] assume that the header of a packet is in plaintext, while the payload is encrypted. A header shows the *ID* of the source and the *ID* of the last intermediary node on the route. Kamat et al. [10] assume a local, passive, external adversary that starts its tracking at the sink. The adversary tries to overhear as many traffic streams as possible in order to estimate where the subjects are located, based on the estimated time of creation of a packet. The solution attempts to complicate the adversaries' estimation of when a packet was created. If the adversary cannot give a good estimation, then the adversary cannot locate the subject, nor find the trajectory of a mobile subject. Kamat et al. [10], [63] use the mean square error of the estimation of the adversary to measure the effectiveness of RCAD.

The solution works as follows. Every node delays its packets for a random amount of time. After this time, the node sends the packet to the next hop. The delay follows a to-be-chosen statistical distribution, which can vary per node. These delays stack up, hop after hop, providing a large random amount of delay at the sink. The random delay complicates the adversary

at the sink in getting a proper estimation of the temporal information and of the actual location of a source. The buffers of the intermediary nodes fill up as they have to buffer multiple packets for a variable amount of time. Nodes need to use a buffer pre-emption strategy to ensure that packets are not dropped due to the lack of empty buffers. When a node has full buffers it selects a packet according to its pre-emption strategy and forwards the packet immediately, so that the node has space for the incoming packets again. The buffer pre-emption strategy forces a maximum to the duration of the delay of a packet. In other words: the buffer pre-emption strategy automatically adapts the delaying of the packets towards the rate new packets are created within the WSN.

Kamat et al. add four pre-emption strategies in [63]. The interrupted delaying of the packets makes the actual delay of a packet vary even more. The increased variation of the delay of the packets complicates the adversary in finding the statistical distribution of the delays. The first strategy is the longest delay first (LDF): the packet that stayed in the buffer the longest is sent away. The second strategy is the longest remaining delay first (LRDF), in which the packet with the longest remaining delay is sent off first. The third strategy is the shortest delay time first (SDTF), in which the packet with the shortest delay is sent off first. The fourth strategy, is the shortest remaining delay first (SRDF), in which the packet with the shortest remaining delay time is sent first. LRDF provides the least pre-emptions and alters the original traffic flow the most, which results in the best mean square error of the packet creation time prediction. RCAD has received little attention from the literature: Li et al. [12] mentions and shortly summarizes RCAD.

3) Summary of Delay Based Solutions

In this section, we discussed the following solutions: PRESH, ex-PRESH, and RCAD. All the solutions are based on delaying packets at (intermediary) nodes. Where ex-PRESH and PRESH are more focused on the active local adversary trying to do a timing analysis attack on any node, while RCAD is more focused on confusing a local adversary that does a timing analysis attack from the position of the sink. In ex-PRESH and PRESH only the identity of the current sender is known, whereas in RCAD both the identity of the origin and the current sender are known. Lastly: PRESH and ex-PRESH are designed to work against an active, internal adversary, whereas RCAD was originally designed for a passive, external adversary. All three solutions aim at providing temporal privacy, hence providing SLP. See Table I in Section IV for an overview of the different solutions and their adversarial models.

E. Using Dummy Data Sources

In this section, we discuss the solutions based on dummy data sources. We first look at the history of dummy data sources in Section III-E.1. Next, in Section III-E.2 we look at the solutions in the state of the art that use dummy data sources as a core technique. We summarize our findings in Section III-E.3.

1) History and Introduction

Dummy data sources are nodes that send out dummy packets to other nodes within the network. The dummy packets do not contain information about real events, but are used to obfuscate the real traffic or divert the adversary by mimicking the presence of a fake subject. When an adversary cannot decrypt the traffic, it cannot distinguish a real packet from a dummy packet. Dummy traffic was not specifically designed to obfuscate real traffic. We find applications of dummy traffic far back, such as in the ALOHA packet broadcasting protocol that needs dummy traffic for maintenance related reasons [119]. An early example of how dummy data can obfuscate real traffic comes from Pfitzmann et al. [35]. They introduce dummy channels in ISDN-mixes. A dummy channel of Pfitzmann et al. [35] consists of a logical connection that conveys dummy traffic in order to make local traffic unobservable. Later on, authors like Ozturk et al. [14] and Shao et al. [57] use dummy traffic to divert an adversary and obfuscate real traffic.

This section comprises the following solutions, as described in Section III-E.2: aggregation-based source location protection scheme [81], A-real and A-fake [58], cloud-based scheme for protecting source location privacy [60], constant rate [57], the dynamic bidirectional tree [28], distributed resource allocation algorithm [46], dummy wake-up scheme [68], fake sources 1 [29] and fake sources 2 [29], fitted probabilistic rate [57], the group algorithm for fake-traffic generation [82], globally optimal algorithm [72], the heuristic greedy algorithm [49], mixes [49], the optimal filtering scheme [62], periodic collection [15], persistent fake source routing [13], the probabilistic algorithm [72], proxy-based filtering [27], SECLOUD [52], short-lived fake source routing [13], source simulation [15], the timed efficient source privacy preservation [83], the timing analysis resilient protocol [69], tree based filtering [27], the trusted computing enabled heterogeneous WSN [84], unobservable handoff trajectory [55], and the zigzag bidirectional tree [28].

2) SLP via Dummy Traffic in the State of the Art

One of the first solutions that is based on using dummy data sources comes from Ozturk et al. [14]. Their solution is called short-lived fake source routing (SLFSR), and it is mentioned as a theoretical example to introduce the concept of fake sources. The solution should cope with a local adversary and works as follows. If a node receives a real packet, it uses a probability p to decide whether to generate a dummy message. If p is below a threshold P, then the node broadcasts a dummy packet to its neighbours. The neighbours drop the dummy packet immediately when they receive it. The threshold P allows to tune between power consumption and SLP. The solution costs about one and a half more power in comparison to phantom flooding. However, it improves the safety period (see Section III-M for more details on phantom flooding).

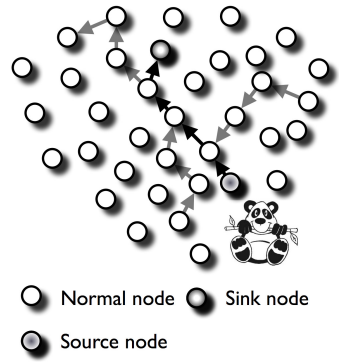
Both Kamat et al. [13] and Ozturk et al. [14] point out that having one dummy data source at the time for only one dummy packet is not enough to distract an adversary. They both recommend using persistent fake source routing (PFSR). The persistent fake source routing is similar to the short-lived fake source routing, with the difference that a node continues to broadcast dummy packets that are dropped immediately

when the neighbours receive the packets. Kamat et al. [13] and Ozturk et al. [14] point out that fake sources positioned wrongly might lead the adversary towards the actual source. Li et al. [12] argue that adversaries are not distracted long enough by the dummy data sources, due to the short-liveness.

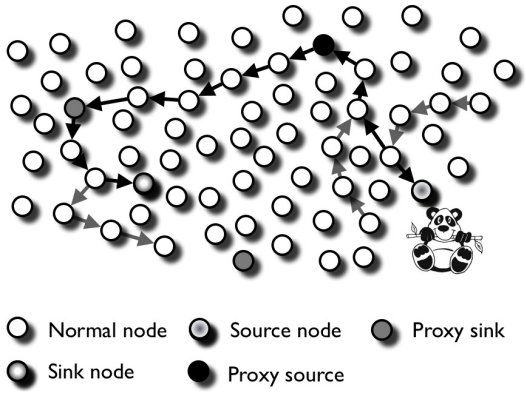
Chen et al. [28] provide two solutions to confuse a local adversary. Both solutions add dummy traffic, in such a way that the adversary is not sure if it is tracking real traffic from the source towards the sink, or if it is following dummy traffic. Their first solution, the dynamic bidirectional tree (DBT) works as follows. Each node knows its own hop count to the sink and of its neighbours to the sink. When a source generates a packet or when a node needs to forward a packet, it forwards the packet to a randomly selected neighbour with a shorter or equal hop-distance. This forwarding pattern creates a random walk. Intermediary nodes that are closer to the source than to the sink use a probability p to randomly select a neighbour to create a branch. When a neighbour creates a branch, it forwards a message with the order to create dummy traffic for at least h hops away. The last node that gets this message and is h hops away starts sending dummy packets, which are routed back to the intermediary node that started the branch. Intermediary nodes that are closer to the sink use probability p to randomly select a neighbour to start branches as well. If a branch is started close to the sink, then the intermediary node establishes a path of at least h hops and sends dummy packets over the path, to fake a separate path towards a fake sink. Figure 9(a) shows an instance of DBT. The direction of the arrows depicts the flow of the traffic. Black arrows depict real traffic, and grey arrows depict dummy traffic. Two intermediary nodes close to the source and one intermediary node close to the sink have created a separate branch, which should divert the adversary. The dummy traffic should divert the adversary away from the actual location of the sink and the source.

The second solution from Chen et al. [28] is the zigzag bidirectional tree (ZBT). The sink generates two proxy sinks, at each side of the sink. Both the real sink as well as the proxy sinks flood the network with a beacon so that all nodes know their hop count towards the proxy sinks and the real sink. A source node selects the proxy sink that is the furthest away from it as a destination. Next, the source randomly selects a node that is i hops away from the source, and marks it as its proxy source. From here onwards, the real traffic is routed from the source, to the proxy source, to the proxy sink, and then to the sink. ZBT follows the same dummy traffic generation scheme as DBT, but does not allow for any branches between the proxy source and the proxy sink. Figure 9(b) shows an instance of ZBT. The direction of the arrows show the flow of the traffic; black arrows depict real traffic and grey arrows depict dummy traffic. The source node selected a proxy source at the required distance and selected the proxy sink that is the furthest away from the source.

Jhumka et al. [29] introduce two different fake source based solutions. Their first solution is called fake source 1 (FS1) and works as follows. When a source reports to the sink, it broadcasts a message to its neighbours. The neighbours check if they have received the message before and if not, they increase the hop count and broadcast it again. If a node has received



(a) Instance of dynamic bidirectional tree



(b) Instance of zigzag bidirectional tree.

Fig. 9. Bidirectional tree approaches: the black arrows depict real traffic, the grey arrows depict dummy traffic.

the message before, it drops the message. This process is repeated until the message reaches the sink. Next, the sink floods an *away*-message with the hop count of the message that informed the sink of the event. Nodes that receive the *away*-message check if they have a message from the source with a hop count of one. If a node received such a message, then it takes the hop count from the *away*-message minus one and uses it as the value of a *TTL* counter for its *choose*-message. The node then broadcasts the *choose*-message. If a node receives a *choose*-message, it uses a probability p to decide whether to forward the *choose*-message. If $p \geq P$ (where P is a set threshold), then the node decrements the *TTL*-counter of the *choose*-message and forwards it to its neighbours. If the *TTL*-counter of the message equals to zero, then the node that has received the message uses probability p to decide whether it should become a fake source. If $p \geq P$, then the node becomes a fake source.

Fake source 2 (FS2) is the second solution of Jhumka et al. [29]. This solution differs from FS1 in the way the *choose*-message is forwarded. In FS2, each node that receives a *choose*-message decrements the *TTL* counter and forwards the message until the *TTL* counter reaches 0, without deciding based on p.

Majeed et al. [69] introduce the timing analysis resilient protocol (TARP) to counter a local adversary's timing analysis attack. The solution ensures that all nodes send an equal amount of traffic at the same rate, so that the adversary

cannot learn anything from the transmissions. Nodes do not send individual packets, but frames of exactly n slots to equalize the traffic among the nodes. Each slot of a frame contains one packet. Every packet within a frame has its own destination. So, when a neighbour receives the frame, it selects the packets addressed to it and buffers them. Every node keeps on buffering the incoming packets until the node's broadcast timer reaches zero. Then, the node puts the buffered packets with new destinations into a frame and broadcasts the frame, after which the node resets the broadcast timer again. If a node does not have n packets to fill the frame with, then it adds dummy packets to fill the empty slots of the frame. An adversary cannot discern the different packets within a frame, as nodes can encrypt the whole frame or encrypt each packet.

TARP has two enhancements. The first one is multipath-TARP (M-TARP), which uses multipath routing instead of single path routing. In M-TARP, nodes use the unfilled slots within their frames to send real data via multiple paths to the sink. M-TARP has a better utilization of the network capacity. Another improvement is adaptive-TARP (A-TARP). In A-TARP, nodes use adaptive rate timers, instead of timers with a constant period. Nodes alter their sending rate depending on the amount of packets that they receive, in order to provide a better throughput. The problem with A-TARP is that it could lure adversaries towards places within the WSN where there are more transmissions.

All the solutions based on dummy data mentioned so far only defend against a local adversary. Mehta et al. [15] were one of the first to defend against a global adversary with a solution based on dummy traffic. Their solution is called periodic collection (PeCo). In PeCo, every node shares an individual key with each of its neighbours for encryption. When a node receives a packet, it decrypts the packet and places it in a first in first out (FIFO) ordered queue. Every node has a decrementing transmission timer. When the transmission timer expires, a node selects the message that is at the head of the queue, encrypts the message with the shared key of the next hop, and forwards the message. If there is no packet in the queue while the transmission timer expires, then the node sends a dummy packet. Dummy packets are dropped upon arrival. Note that nodes do need to have enough buffers to achieve a good delivery rate. PeCo has received a considerable amount of attention in the literature. Aomair et al. [74] note that PeCo suffers from latency and computational overhead. Bicakci et al. [62] say that PeCo could be combined with solutions such as PFS and TFS or FitProbRate. Rios et al. [21] explain that PeCo consumes too much energy for a WSN. Other authors, such as Wang et al. [92], Yang et al. [81], Ouyang et al. [72], Yang et al. [27], Chen et al. [28], Lu et al. [83], and Abbasi et al. [46] only refer to PeCo.

Shao et al. [57] introduce Constant Rate (ConstRate) as a fist step towards event unobservability via dummy traffic. In order to reach unobservability, nodes use slotted transmission and send a packet at each interval. If a node does not have a real packet to send at a slot, it sends a dummy packet. Real traffic cannot be discerned from dummy traffic as all traffic is encrypted. The fixed timeslots introduce latency and nodes produce a lot of dummy messages, which increases power consumption. The solution has been mentioned by multiple authors in the literature. In fact, some of the solutions presented here are introduced as an improvement of ConstRate as we show later on. Li et al. [76] mention that ConstRate can lead to: packet collisions, a decrease in network throughput, and a high consumption of energy. Other authors, such as Fan et al. [45], Yang et al [120], and Ngai et al. [109] only refer to ConstRate. Note that Ouyang et al. [72] call ConstRate the "naive" algorithm and call the dummy messages "maintenance messages".

Shao et al. [57] introduce probabilistic rate (ProbRate) as an improvement of ConstRate. The slotted transmission scheme from ConstRate is replaced in ProbRate with a probabilistic scheme that follows an exponential distribution. The exponential distribution of the transmissions should be similar to the distribution of the events. The probabilistic transmission scheme decreases the amount of necessary dummy messages and decreases latency. We note that this solution fails if the adversary finds the parameters of the used exponential distribution. Furthermore, the solution will not perform optimally in the face of bursts of events.

Shao et al. [57] provide an improvement of ConstRate and ProbRate with fitted probabilistic rate (FitProbRate). In FitProbRate, nodes generate dummy messages according to a statistical distribution, which is similar to the statistical distribution of the real events. Every source embeds its real packets in the stream of dummy messages, so that an adversary should not be able to tell which part of the traffic is real and which part is fake. The embedding happens in such a way, that the statistical distribution of the dummy traffic itself is not altered. Instead, a statistical test is applied to make sure that the real event reporting distribution is indiscernible from the dummy traffic rate distribution.

Shao et al. [57] call the indiscernability of the interval distribution of fake and real traffic interval indistinguishability. Shao et al. [57] show that FitProbRate's latency and power consumption is much lower than the latency and power consumption of ConstRate. The authors in the literature show that the decrease of latency and power consumption comes at the cost of privacy. Alomair et al. [121] mention that an adversary can detect 74 percent of the real messages in some cases. They propose to encrypt all the packets and replace dummy messages with real messages whenever they are available. Yang et al. [120] show that if there is a lot of real traffic, then it becomes easier for a global adversary to detect the sources. Kokalj et al. [82] argue that FitProbRate is not designed for WSNs in which the source is more than one hop away from the sink. When a source is more than one hop away, then the latency will stack up over multiple hops. Kokalj et al. [82] refer to this latency as the publishing route latency. The actual publishing route latency depends on the rate of event reporting. If the rate is on a second basis, then the delay is very little. However, if it is on a daily basis, then the delay might become unacceptable. Fan et al. [45] note that FitProbRate cannot defend against attacks that correlate the content of incoming and outgoing messages. Wang et al. [92] state that FitProbRate sacrifices location privacy for short message delivery delay(Other authors, such as Lightfoot et al. [80], Chen et al. [28], Alomair et al. [74], and Abbassi et al. [46] only quickly mention or summarize the solution).

Alomair et al. [58] take the idea of FitProbRate a step further. They present two algorithms: A-real and A-fake. The first algorithm is used when there are real events within the WSN. In this case, nodes embed real traffic within dummy traffic, based on statistical tests similar to the ones used in FitProbRate. The second algorithm is used when the nodes within the WSN do not detect any real events, meaning that there are no subjects present in the monitored area. In this case, the nodes fake the presence of subjects by introducing fake events, which are then embedded in the dummy traffic. An adversary that observes the traffic will see no difference between A-fake and A-real, so that it does not know whether there is a subject present.

Ouyang et al. [72] introduce three different solutions as an improvement over ConstRate.

One of the solutions that they introduce, is the globally optimal algorithm (GOA). In GOA, each node has a pseudo random number generator that defines the time a node should count down before it can transmit again. When the timer reaches zero, the node needs to send a packet. If it has no real packet to send, then it sends out a dummy packet. All nodes use the same pseudo random generator and know the seeds of each other. A node can use this information to calculate a path with the shortest delay towards the sink. When comparing GOA to ConstRate, then the following can be said. On the positive side: GOA has a better throughput and lower energy consumption than ConstRate. However, in GOA, a node needs to know the complete topology, whereas a node does not have to know the topology in ConstRate.

Another solution from Ouyang et al. [72] is the heuristic greedy algorithm (HGA) as an improvement of GOA. Nodes follow the same approach as in GOA, with one difference: a node does not know the complete topology, but it only has the information on the location and the seeds of its neighbours. A node uses this information to choose the best neighbour to forward a real packet to.

Lastly, Ouyang et al. [72] introduce the probabilistic algorithm (PBA) as an improvement of GOA and HGA with the goal to reduce the overhead of the dummy traffic. Nodes still follow the process of HGA, but now they do not send a dummy message every time their counter reaches zero. When the counter of a node reaches zero and the node does not have any real messages to send, then it uses a probability p to decide whether it should send a dummy message or not. Setting the threshold of p allows to tune between SLP and communication overhead.

Another algorithm based on the idea of ConstRate, is the distributed resource allocation algorithm (DRAA), as introduced by Abassi et al. [46]. In DRAA, each node calculates iteratively the best rate of their dummy message generation. The goal is to find the minimal necessary amount of dummy traffic to obfuscate the real traffic , without having to apply the approach of ConstRate. This solution provides reasonable SLP and can save a substantial amount of energy compared to ConstRate, depending on the amount of resources and nodes in an area.

Solutions similar to ConstRate generate a lot of dummy traffic, which has a negative influence on the lifetime of a WSN. There are multiple solutions within the literature that aim at filtering the dummy traffic in such a way that the dummy traffic still obfuscates the real traffic, without shortening the lifetime of the WSN too much.

Yang et al. [27] provide two solutions that filter out the dummy traffic. Their first solution is the proxy-based filtering scheme (PFS). This solution introduces proxies, which filter out dummy traffic. Each proxy is connected to a set of nodes, called a cell, and shares a key with each of the members of the cell. The nodes send their output to the proxy in a similar fashion to ProbRate and the proxy filters out the dummy traffic and queues the real data. The proxy sends the queued data after a certain amount of time towards the sink. If a proxy receives a message from another proxy, then it forwards the message directly. If a proxy does not have any queued data over time, it sends a set of dummy messages to the sink. Each proxy shares a key with the sink and uses a different transmission rate than the other members of its cell in order to confuse the adversary. Using the wrong buffer size at the proxy, the wrong proxy positions or transmission rates causes unfiltered traffic, high latency or packet-loss. The second solution of Yang et al. [27] is the tree based filtering scheme (TFS). The proxies from PFS are now ordered in a tree-based structure. The normal nodes are still grouped in cells and form the leafs of the tree. The proxies within the tree aggregate the data, filter out dummy traffic, and forward real data towards the sink at a constant interval. TFS reduces dummy traffic and power consumption compared to PFS. However, it doubles the overall latency.

Both PFS and TFS received quite some attention in the literature. Bicakci et al. [62] have done an in-depth analysis and simulation of TFS and PFS and provide optimal placing algorithms for the proxies. They recommend to look at using homomorphic encryption instead of symmetric key encryption, so that the fake traffic can be filtered and real traffic re-encrypted with a few algebraic operations. Suarez et al. [68] note that these solutions can be quite powerful, but finding the optimal locations of the proxies is still an NP-hard problem. Fan et al. [45] point out that the proxies themselves become the weaker points in both solutions in the case of an internal adversary. Yang et al. [81] point out that the traffic at a proxy node can be heavily congested if many events happen in the area of the sensors that send their findings to the proxy. The congestion leads to large message delays and message drops. Other authors, such as Li et al. [12], Alomair et al. [74], and Lightfoot et al. [80] only provide a summary of the solutions or just quickly mention them.

Another filter-based solution comes from Bicakci et al. [62] based on their analysis of TFS and PFS. This solution is called the optimal filtering scheme (OFS). In OFS every node can become a proxy, which allows for the optimal routing and filtering, giving the WSN an optimal lifetime. Every proxy within OFS can follow its own filtering rules to further optimize the balance between SLP the lifetime of the WSN. Proxies can also filter out dummy traffic of other proxies. Proxies can either work asynchronously to increase latency and SLP or work synchronously to reduce latency and SLP. Positioning the proxies correctly is again a challenging task.

Yang et al. [81] introduce the aggregation-based source location protection scheme (ASLP), which looks similar to the filtering solutions mentioned above. The solution consists of

three phases and combines dummy packets, packet encryption, and data aggregation to provide SLP. First, the WSN is divided into separate clusters, each with its own cluster head. The cluster-heads form a tree structure with the sink as the root of the tree. Every member of the cluster shares a key with the cluster head. The nodes report to their cluster-heads at a rate that follows an exponential distribution, which can be regulated via the distribution parameter λ. The cluster nodes report every τ time units to their parent cluster head, which could be the sink. In the second phase, the sink distributes the values of λ and τ through the WSN. In the third phase, the actual sensing and reporting begins. Nodes keep on sending packets to their cluster head following the exponential distribution. Nodes send a real packet if they sense an event, otherwise they send a dummy packet. The cluster-heads aggregate the incoming information and send the aggregate encrypted towards the sink. The variables λ and τ can be used to tune between latency and power consumption.

Yang et al. [84] provide a trusted computing based solution. Their solution consists of a trusted computing enabled heterogeneous WSN architecture that makes use of dummy data. We name this solution TCH-WSN (Trusted Computing enabled Heterogeneous WSN). In TCH-WSN, the WSN is divided into clusters. Each cluster has a very powerful node with a trusted platform module (TPM) or a mobile trusted module (MTM) to provide data security and node integrity checks via attestation. See [122], [123] for more details on the MTM and TPM. Both the sink and the cluster-heads have their own power supply which does not deplete. The sink can communicate directly to all the nodes in the WSN, whereas sensor nodes can only send data to the sink via their cluster-head. The cluster-heads ask a sensor node within their cluster to be a fake source for a while, after which they request another node within their cluster to take over the role of fake source. The fake messages from the fake sources are then transmitted to the sink.

Lu et al. [83] introduce the timed efficient privacy preservation (TESP^2). TESP^2 provides SLP. The solution uses cluster-heads that filter out the dummy traffic as well. The solution consists of three phases. First, the sink creates a pair of public and private keys for each node. The sink bases its keying on elliptic curve cryptography, which is less computationally intensive than conventional asymmetric key algorithms, such as RSA. Second, all the nodes are deployed and organized in clusters. Each cluster has a cluster-head, that communicates with the other cluster-heads and forms a tree with them, with the sink at the root of the tree. Each node computes a symmetric key for each neighbour so that they can exchange encrypted messages. Third, each sensor checks on an interval whether it has sensed something. If not, it generates a fake message which is encoded by the cluster-heads public key. Otherwise it generates a message encoded by the sink's public key. The message travels via other nodes to the cluster-head, using the earlier exchanged symmetric keys for encryption and decryption. When a cluster-head receives the message, it deletes the fake message upon successful decryption. The cluster collects all the real messages and aggregates them. Next, the cluster-head waits for a data collection signal from a cluster-head higher in the tree or from the sink. When the cluster-head receives the data collection signal, it sends its aggregated re-encrypted data towards an upstream cluster-head. The cluster-head stores the aggregated data and sends it together with its own aggregate at the next data collection phase. This process is repeated until the aggregate reaches the sink.

Another solution in this category is called source and destination seclusion using clouds (SECLOUD). Doomun et al. [52] introduce SECLOUD as a solution that should defend against global adversaries. The solution works in three phases. First, all nodes periodically flood the network with a hello message that has a *TTL* counter of h hops in order to find the neighbours of each node up to h hops. The neighbours of a node up to h hops form together the cloud for that specific node. When a node becomes a source and wants to send something, it selects a few members of its cloud and marks them as pseudosources. The pseudosources broadcast dummy messages at the same rate as the real source sends real traffic. The dummy messages are dropped directly by all receivers. The sink has its own cloud of size i hops, which also contains pseudosources.

Doomun et al. [52] leave room for different routing methods to let a source communicate to the sink. One option is to use single path routing algorithms. Another option is to use delegate sources and delegate sinks. A source selects a few nodes within its cloud and marks them as delegate sources. The sink selects a few nodes within its cloud and marks them as delegate sinks. Messages from the source are then routed to the sink via the delegate sources and delegate sinks.

Doomun et al. [52] add the usage of fake clouds to further obfuscate the real traffic. A fake cloud can be a fake source with a cloud or a fake sink with a cloud. The fake source and the fake sink follow the same methods as a real source and sink, which make it hard for a global or a local adversary to locate the actual source. However, Doomun et al. [52] do not fully describe how these fake clouds are located and constructed. The parameters i and h together with the number of pseudo sources and the number of fake clouds allow to tune energy consumption and SLP.

A solution that looks similar to SECLOUD is the cloud-based scheme for protecting source-location privacy (CSPSLP) from Mahmoud et al. [60]. They assume that multiple sensors can pick up a subject within an area. When multiple sensors detect a subject, then they all start to inform the sink, which creates a so called hotspot of local traffic. Multi-local adversaries are looking for hot-spots to find the source. The aim of CSPSLP is to let a source node blend into a cloud of multiple nodes, so that the normally formed hot-spot blends into a large crowd. The solution works in three phases. In the pre-deployment phase, every node gets its unique *ID*, a shared key with the sink, and a secret key which it can use to compute pairwise shared keys with its neighbours. The pairwise shared keys, the identities, and a deployed hashing function are used by both the nodes to create outgoing and incoming pseudonyms that are similar to the hidden identities of the Anonymous Path Routing. The second phase is the bootstrapping phase, which is only executed at deployment. In this phase, nodes first report their location to the sink and find the shortest route to the sink. Next, every node requests a few randomly chosen nodes that are at most h hops away to

become its fake source and they exchange data so that they can generate pseudonyms of each other. Lastly, every node groups its neighbours in several separate groups. Every group consists of a set of members that are directed in opposite directions of one another. A node shares a key with every group as well. The next phase, is the event transmission phase. When a source senses the subject, it selects a fake source and broadcasts a packet to the group of neighbours that has a member on the route towards the fake source. The packet contains the actual event information encrypted with a pairwise shared key owned by the source and the sink, the pseudonym shared by the fake and the real source and the pseudonym shared with the intermediary node between the fake source and the fake sink and the source. The intermediary node only accepts the packet if its own pseudonym is in there. If its pseudonym is not within the packet, then it has to send a fake packet with a *TTL* counter, which has to be forwarded till the *TTL* counter reaches zero. If the node's pseudonym is in the packet, then it changes the pseudonym to the next intermediary node and forwards the packet to the group that is connected to it. Again, if a member of the group receives the packet, while its pseudonym is not within the packet, then it needs to send a fake packet, otherwise it accepts the packet and repeats the procedure. This procedure is repeated until a fake source gets the packet. When the fake source receives the packet, it routes the packet towards the sink, using a similar approach to APR. The only difference with APR is that in CSPSLP every intermediary node between the fake source and the sink re-encrypts the event information with the key it shares with the sink. When the sink gets the packet, it checks for the pseudonym of the source and the fake source, so that it knows which keys it needs to use to decrypt the event information. Note that the fake traffic coming from the same cloud can be filtered by intermediary forwarding nodes: if a node gets a set of fake packets from one cloud, then it is allowed to filter the fake packets and only forward one of them.

Another dummy data source based solution comes from Suarez et al. [68] as they introduce the dummy wake-up scheme (DWUS). The aim of DWUS is to confuse both local and global adversaries by creating several dummy data streams towards the sink. The dummy data streams are organized in such a way that they look a lot like real event streams. The solutions consists of three phases. First, the WSN is divided into d squads of dummy populations. Each squad periodically selects a new squad header. A squad header is responsible for selecting a dummy source. Second, the wake-up phase: each squad header appoints a dummy source close to itself and sends a wake-up message to it. Third, the dummy source receives the wake-up message and sends a dummy event-message to the corresponding relaying node towards the sink. The last two phases are repeated at a certain rate to fake the presence of a subject as multiple nodes might sense the same (fake) subject. Note that the solution is still not completed as Suarez et al. [68] show in their recommendations for future work.

A solution that looks similar to DWUS is the group algorithm for fake-traffic generation (GAFG), from Kokalj et al. [82]. In GAFG, each node sends its traffic on a pre-defined route to the sink. Nodes forward messages with a higher rate than at which the sources can generate a new report. When a source has sensed a subject, it directly sends a message to the sink. The event reporting follows an exponential distribution and GAFG aims at creating a dummy data report rate that is close to this distribution. The solution ensures that a several nodes throughout the WSN will send a dummy packet with a timing based on the event distribution.

Mehta et al. [15], [16] take a different approach with their solution called source simulation (SoSi). They believe that a global adversary is only interested in a trace of events through a WSN that indicates the presence of a mobile subject. The adversary observes every trace, whether fake or real. Every observed trace is a candidate trace to be a real trace. The goal of Mehta et al. [15], [16] is to confuse the adversary with multiple traces. They argue that the optimal privacy is provided when an adversary overhears a set of transmissions, which indicates that there could be a subject near any of the sensors within the WSN. Fake candidate traces are implemented using virtual objects, which mimic the behaviour of real objects in order to divert the global eavesdropper. The virtual objects are implemented as follows. A set of randomly selected nodes within the WSN gets a token at deployment time. These nodes are called token-nodes. From there on, the WSN functions in rounds that take a certain amount of time. In each round, a token-node triggers its neighbours to report an event to the sink, which creates a stream of reporting messages. At the end of the round, each token-node selects a next node to become the token node based on the virtual object model. The solution does not introduce additional latency as the virtual objects do not delay the actual event reporting.

Yang et al. [84], claim that the combination of a random walk together with SoSi can be very powerful, even though SoSi is not as power efficient as their solution (TCH-WSN). Yang et al. [81] argue that SoSi saves a lot of energy compared to other dummy traffic solutions, but does sacrifice a little SLP(SoSi is mentioned in [21], [28], [46], [83]).

The last solution in this category is unobservable handoff trajectory (UHT) by Ortolani et al. [55]. This solution is built for a specific scenario in which subjects enter the WSN from the border (or perimeter) and move towards a random point within the WSN. The perimeter nodes know the poisson distribution of the frequency of subjects entering the WSN. The distance traveled by the subjects within the WSN is uniformly distributed. The first distribution is learned by the nodes at the edge of the WSN, called the perimeter nodes, as subjects come by. The second distribution is known a priori by the perimeter nodes. Perimeter nodes generate fake traffic based on the poisson distribution when there are no real subjects entering the WSN in the sensing range of the perimeter node. When a border node generates a fake message, it includes a *length* variable that works similarly to a *TTL* counter. Next it XORs the fake message with the identity of the next node that has to generate a fake event as well. Then, the border node sends the fake message to another intermediary node that routes the fake message to the sink. Note that nodes use broadcasting to send a message. When a node overhears the message, it tries to XOR the message with its own identity to see whether it should generate a fake event. If a readable message comes from the XOR-ing operation,

then it decrements the *length* variable. If the *length* variable is still above zero, then the node generates a fake message, XOR-ed with the identity of the next node that should create a fake event. If the *length* variable is equal to zero, then the node XORs the fake message with its own identity. Next, the node sends the fake message to an intermediary node to send it to the sink. The messages from the fake events are forwarded to the sink. This solution confuses the adversary: a global adversary that observes the traffic can no longer tell whether the candidate trace is a real trace or a fake one. Note that all messages are encrypted and decrypted with the same key which is shared by all nodes within the WSN.

3) Summary of Dummy Source Based Solutions

In this section, we discussed the solutions based on dummy traffic. Most of the solutions, except for SLFSR, DBT, ZBT, and TARP, defend against a global adversary. Some solutions, such as A-real, A-fake, ConstRate, ProbRate, FitProbRate, HGA, GOA, PBA, and DRAA rely intensive on dummy traffic that spans the complete WSN to obfuscate the real traffic against a global adversary. It is because of the heavy usage of dummy traffic that filter solutions, such as PFS, TFS, and OFS, were introduced to filter out the dummy traffic. Other solutions, such as ASLP, TCH-WSN, and PeCo, combine dummy sources and the filtering in a single solution.We found solutions, such as UHT, DWUS, SoSi, and GAFG, that mimic the presence of a real subject as they provide fake readings and even complete fake event traces that look similar to the movement of a real subject. See Table I in Section IV for an overview of the different solutions and their adversarial models. Note that there are also other solutions in the literature that use dummy data and provide SLP. We did not incorporate them in the discussion as they do not aim to provide SLP. We quickly mention them here for completeness: the usage of buses from Beimel et al. [124], a confidential and lifetime-aware routing protocol from Wang et al. [125], the location privacy routing protocol from Jian et al. [126], finding the optimal cover mode by Jiang et al. [127], the fractal propagation approaches from Deng et al. [101], and the approaches fake sinks and balanced flows by Bicakci et al. [128].

F. Cyclic Entrapment Based

In this section, we discuss the solutions based on the concept of cyclic entrapment. We first introduce the concept in Section III-F.1. Next, we look at the solutions in the state of the art that use the cyclic entrapment concept in Section III-F.2, and we summarize our findings in Section III-F.3.

1) History and Introduction

Ouyang et al. [85] introduce the cyclic entrapment concept as a special case of fake data source routing. In cyclic entrapment, multiple nodes act as fake data sources , connect to each other, and form a loop. The members of the loop start to send messages to one another as soon as a member starts the message exchange within the loop. The member that starts the message exchange starts the activity within the loop: it activates the loop. All the fake traffic within the loop follows the same cyclic route with no beginning or end. The aim of cyclic entrapment is to confuse a local adversary with these loops during a hop-by-hop-trace attack. If the loops are large enough, then it should take a while before a local adversary realizes that it is moving in cycles while tracing the traffic. As Ouyang et al. [85] therefore note that the local adversary is trapped within the cyclic traffic. This category consists of two solutions as described in Section III-F.2: the cyclic entrapment method [85] and information hiding in distributing environments [86].

2) Cyclic Entrapment Based Solutions in the State of the Art

Ouyang et al. [85] introduce the cyclic entrapment method (CEM) as an addition to other routing algorithms. CEM itself does not provide any routing algorithm. CEM only provides methods to setup the loops, to activate the loops and to de-activate the loops. The loop-creation process starts right after the deployment of the WSN. Every node decides, based on a certain probability distribution, whether it will create a loop or not. If a node A creates a loop, then it takes two randomly picked neighbours B and C, and it sends a packet to B with neighbour C as the destination of the packet. Next, the packet travels through the network with any type of routing protocol until it arrives at node C. Every node that gets the packet, adds its identity to the packet. Node C orders the identities found in the packet and creates a cyclic loop. Then, node C informs all the members of the loop so that each member can update its routing table. Note that nodes can be part of multiple loops at once and that the shape and coverage of the loop highly depends on the used routing protocol during the initialization of the loop. A loop gets activated as soon as one of its members forwards a real packet from the source to the sink. The member that forwarded the real packet and activated the loop is called the activation node. Nodes that are members of multiple loops can also activate multiple loops at once. The more loops a node activates, the more traffic an adversary has to analyse. Once the loop is activated, its members start to create and forward dummy messages within the loop until the loop is deactivated. The loop is deactivated when the source stops sending packets via the routing path.

Ouyang et al. [85] introduce probabilistic forwarding within the loop to save energy. In the case of probabilistic forwarding, every member of the loop only forwards a dummy message to the next member of the loop based on a certain probability. CEM provides SLP against local adversaries if the loops are distributed correctly. Otherwise, a loop could guide an adversary towards the actual subject. Shaikh et al. [19] note that CEM cannot provide identity privacy, but it does provide SLP. Shaikh et al. [19] and Mehta et al. [15], [16] mention that the energy consumption and the effectiveness of the solution increases as the size of the loops and the number of loops increases. Alomair [74] points out that CEM cannot protect against a global adversary.

Figure 10 shows an instance of CEM with a couple of activated loops. The grey arrows show the traffic flow of a loop. The source in the South-east corner of the WSN has detected a subject and now sends a packet towards the sink. The packet is routed to the sink using shortest path routing as depicted by the black arrows. When a member of a

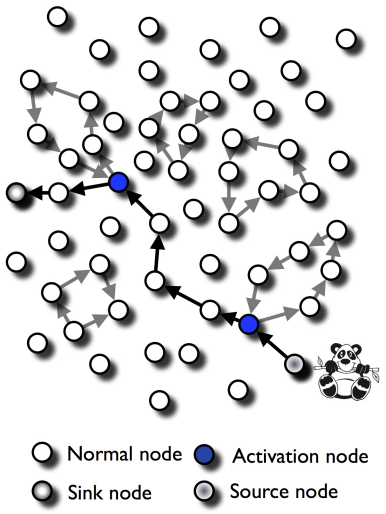


Fig. 10. Shortest path routing together with the application of CEM.

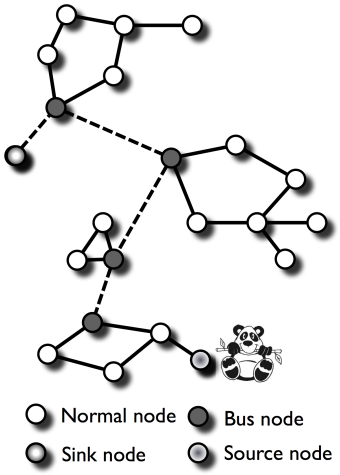


Fig. 11. An instance of iHIDE in a WSN.

loop forwards the packet, it becomes an activation node that activates the loop. Members of an activated loop then start to exchange messages, following the grey arrows.

The second solution in this category is information hiding in distributing environments (iHIDE), as introduced by Kazatzopoulos et al. [86], [102]. iHIDE uses virtual cyclic loop structures, called rings, that consist of a number of interconnected nodes. Each ring has a main node, called the bus nodes (BUNs) of the ring. The BUNs are connected to each other, forming a logical connection that spans the WSN that ends at the sink. This logical connection is called the bus.

Figure 11 shows an instance of the virtual overlay of iHIDE. The dark grey nodes are the BUNs. The dashed line that interconnects the BUNs is the bus. Each BUN has multiple normal nodes connected to it, which together form a ring. Note that some nodes are only connected to one member of a ring, a fact to which we will get back later on again.

When a source senses an event, it creates a packet with a certain time to live (*TTL*) counter and routes the packet via intermediate members of the ring to the BUN. The packet's *TTL* counter is decremented at every hop. The packet is dropped when the *TTL* counter reaches zero. A BUN stores the packet in its cache upon receiving the packet and forwards the packet to the next member of the ring if the *TTL* counter is not yet zero. A BUN forwards the packet that it cached over the bus to the next BUN, while the members of its own ring might still be exchanging the packet. When the next BUN gets the packet, it uses probability p to either directly forward the packet to the next BUN or first route a copy into its own ring. A BUN adds a new *TTL* counter within the copy of the packet before a BUN sends the copy into its own ring. If a BUN puts it in the ring first, it will forward the message a little later to the next BUN on the bus. Note that traffic that is routed within the ring is only forwarded to members of the ring that are connected to two other members of the ring. Nodes that are connected to just one member of the ring do not actively participate in the forwarding. The rings of iHIDE are set up using a routing plan, which is distributed in the form of forward probability tables to each of the nodes. Nodes use the forward probabilities to correctly send the packets to the next member of the ring or to a BUN. In iHIDE, nodes use a set of identities to counter identity-based attacks. Only a node's direct neighbour in the ring, a node's BUN, and the sink can relate a node to its identity.

iHIDE is also designed to counter node capturing by a local adversary. Every node shares a unique key with the sink and all nodes forward the encrypted payload of a source without decrypting or encrypting it. So, in order to get the payload, an adversary has to compromise the sink or the source of the payload. There are three variables to tune between a higher power consumption and a longer safety period: the threshold for probability p to send a packet into the ring of a BUN, the number of rings, and the length of the rings. iHIDE received little attention in the literature.

3) Summary of Cyclic Entrapment Based Solutions

The concept of cyclic entrapment is applied in two solutions: iHIDE and CEM. Both the solutions use loops, where packets are sent along to confuse a local adversary. In CEM, any node can activate the loop and nodes within the loop can use probabilistic forwarding to regulate the traffic within the loops. In iHide on the other hand, only BUNs can activate the rings and nodes do not use probabilistic forwarding, but use *TTL* counters and probability to regulate the ring traffic. See Table I in Section IV for an overview of the different solutions and their adversarial models.

G. In Network Location Anonymization

In this section, we focus on solutions that mainly use anonymity to provide SLP. We shortly mention the history of this category in Section III-G.1. Next, we discuss the solutions found in the literature in Section III-G.2, and we summarize our findings in Section III-G.3.

1) History and Introduction

Anonymity has been one of the basic building bricks of providing SLP as we discussed in Section II. Works from Chaum [31], [33] and Pfitzmann et al. [40], [41] show the importance of anonymity to make communication untraceable and, thereby, provide SLP. Solutions in this section use

anonymity as the central concept to provide SLP. The solutions in this section either use aggregation or pseudonyms to provide anonymity and SLP.

This section consists of the following solutions as described in Section III-G.2: the anonymous communication scheme [79], anonymous path routing [87], the cryptographic anonymity scheme [88], the destination controlled anonymous routing protocol for sensornets [50], hashing based ID randomization [89], max query aggregation [90], phantom ID [91], the probabilistic destination controlled anonymous routing protocol for sensornets [50], the reverse hashing ID randomization [89], and the simple anonymity scheme [88].

2) Anonymization Based Solutions in the State of the Art

The first two solutions in this category come from Misra et al. [88]. It was one of the first attempts to concentrate on anonymity through pseudonyms in order to counter the traffic analysis of a global adversary within a WSN. Misra et al. [88] make the following assumptions: every node has a pairwise shared key with each of its neighbours, nodes are organized in clusters, every cluster has a periodically selected cluster-head, every cluster has its own cluster key, the cluster-head is the intermediary hop between the source and the sink, and the WSN has node compromise detection and node revocation techniques to cope with compromised nodes.

The first solution in this category is the simple anonymity scheme (SAS) in which nodes communicate with each other using pseudonyms. In SAS, all nodes share a pseudonym space that consists of a large range of pseudonyms. Every node gets a subspace of the pseudonym space assigned to it before deployment. The sink has a pseudonyms table that holds the pseudonyms per node, so that the sink knows which pseudonyms belong to a specific node within the WSN. Next, the sink organizes the WSN into a breadth first tree using beacon messages. The tree has the sink as the root, the cluster-heads at the next level, and the other cluster members as the leafs of the tree. Every sensor has a pseudonyms table in which it stores the pseudonym ranges used for communications with the other nodes within its neighbourhood. A node stores two ranges per neighbour: one range of pseudonyms for traffic directed from the node to the neighbour and one range for traffic directed from the neighbour to the node. A node includes an index i within its table to denote which range is used for that specific neighbour. The pseudonyms table maps the used pseudonyms to the correct shared key for encryption and decryption of exchanged messages. When a node wants to communicate with another node in its cluster, it selects an outgoing pseudonym concatenated with the index i as the source of the message, an incoming pseudonym of the neighbour concatenated with index i as the destination of the message, and uses the shared key for encryption and decryption of the message.

The second solution from Misra et al. [88] is called the cryptographic anonymity scheme (CAS). It is designed to reduce the amount of memory used for pseudonym management at the expense of a higher computational complexity. In CAS, nodes do not store the ranges of pseudonyms, instead, nodes use a keyed hashed function to generate the necessary pseudonyms. Every node uses their shared keys, together with a seed known by both the node and its neighbours to generate the necessary pseudonyms.

Both solutions have received some attention from the literature. Shaikh et al. [19] claim that SAS and CAS do not have a routing strategy and do not provide SLP. The problem with this claim is that Misra et al. [88] do show in their paper that they use a cluster based approach in which packets are exchanged from a source to a sink via a cluster-head. Park et al. [91] compare CAS to the hashing based ID randomization solution from Ouyang et al. [129], which we discuss below. Park et al. [91] argue that CAS is less resilient to node compromise than the solution of Ouyang et al. [129]. Chen et al. [130] show how SAS and CAS cannot withstand an internal adversary.

The next two solutions in this category come from Ouyang et al. [129]. Their first solution is the earlier mentioned hashing based ID randomization (HIR), which is created as an alternative to CAS in case of a node compromise. In HIR, a node communicates with its neighbour using the hash value *HV* of that neighbour. A node creates an *HV* by hashing an *ID* of a node multiple times with a keyed hash function. A node creates multiple *HV*s. It creates an *HV* to communicate with the sink, based on the node's *ID*, a pairwise shared key shared with the sink, and the keyed hash function. The other *HV*s are created to communicate with the neighbours of a node. A node creates one *HV* for each neighbour, using the pairwise shared key shared with that neighbour, its *ID*, and the keyed hash function. Whenever the *HV* needs to be renewed, it uses the currently used *HV* together with the shared key used for that specific *HV* to generate a new *HV*.

The solution consists of two phases. The first phase is the deployment phase. In this phase, nodes check their location and report their location to the sink. The sink records the locations together with the identities of the nodes. Then, every node exchanges its *ID* with its neighbours and computes a shared key with each of its neighbours. Every node checks whether a neighbour is an uplink or a downlink neighbour. Uplink neighbours are closer to the sink than the node and downlink neighbours are further away from the sink than the node. Then, nodes set up a table in which they record three things per neighbour. The first column contains a hashing index that indicates the number of times the stored *HV* in the next column has been hashed. The second column contains the *HV* of the uplink node in relation to the neighbour (in other words: the *HV* of the node itself is stored if the neighbour is a downlink neighbour and the *HV* of the neighbour is stored if the neighbour is an uplink neighbour). The third column records whether the neighbour is an uplink or a downlink neighbour.

The next phase is the runtime phase, in which nodes start sensing and reporting. A source can generate a message consisting of the *HV* of the receiver, the *HV* that identifies the source towards the sink, the index of the *HV* used to communicate with the sink, and the actual data. A source updates both the used *HV*s in its table after a successful transmission. When an intermediary node receives the message, it checks in its own table whether it has a match of the *HV* used to identify the destination. If it does, then it means that the intermediary node is the next hop. It accepts the message and updates the *HV* in its table. Next, the intermediary node

selects an uplink neighbour and sends the message to that uplink node after changing the destination *HV* of the message into the *HV*. The node updates the *HV* of the uplink neighbour after a successful transmission and waits for a new message or event. This process is repeated until the message reaches the sink. The sink can use the index together with the source *HV* of the message to calculate from which node the message came. When an adversary captures a node, it cannot see the *HV*s used prior to the capture of the node, as each node only stores the shared keys, the hash function, and the currently used *HV*.

The second solution from Ouyang et al. [129] is called reverse hashing based ID randomization (RHIR). In RHIR, nodes first generate all the *HV*s and store them as a hash chain. Nodes then traverse the hash chain in reversed order and use the *HV*s to communicate in the same way as in HIR. A node drops a hash out of the hash chain when it is used. Both RHIR and HIR provide anonymity. Both the solutions only provide SLP in a dense network with a lot of traffic as messages can still be traced using timing analysis. Park et al. [91] argue that the hashing based ID randomization solution from Ouyang et al. [129] is more resilient against node capturing than CAS. When an adversary captures a node, then the adversary knows all the used, current, and future *ID*s. In the case of CAS, whereas the adversary only knows the current and future *ID*s used in the case of HIR.

Another solution against a global adversary comes from Sheu et al. [87]. They introduce Anonymous Path Routing (APR)[1] as a solution that blocks traffic analysis attacks since it hides the identities of the nodes and changes the encryption of each message at every hop. This solution consists of three techniques: anonymous one-hop communication, anonymous multi-hop path routing, and anonymous data forwarding. Anonymous one-hop communication works as follows. Nodes have a link table, in which they save the following information per neighbour: the identity *ID* of a neighbour, a sequence number, a hidden identity for inbound communication with that specific neighbour, a hidden identity for outbound communication with that specific neighbour, a key for message encryption, and a key for message authentication. The sequence number indicates how many times the hidden identities have been updated using a hash operation. When a source *S* has something to send to a neighbour *N1*, it uses the outbound identity for the communication from itself towards *N1*, encrypts the packet using the shared key, and includes a message authentication code using the shared message authentication code. When *N1* gets the packet, it sends an acknowledgement using the inbound hidden identity of *S*. Both nodes increment the sequence number, and update the used hidden identities, and keys using XOR as well as hash operations. Changing the keys and the hidden identities per transmission will make it hard for an adversary to link a message to a recipient. Even if an adversary captures a node, then the adversary can only link the transmissions to specific neighbours and the adversary will have to compromise more nodes to get an overview of the route. Note that nodes need to store older identities of their neighbours to synchronise with a neighbour in case a neighbour is reset or a transmission has failed. We have to mention that the work does not include anything about delaying transmissions or acknowledgements in the case of sparse traffic. In case of sparse traffic, the acknowledgement of a packet could give away the next hop of a transmission.

The second technique of APR is anonymous multi-hop path routing, which allows a source to find a route to the sink. Every node stores a routing table next to the previously mentioned linktable. The routing table contains the following information for every route that uses the node: the *pathID* of the route, the previous hop of the route, the next hop of the route, and the hidden identity that the source shares with the destination. Every node shares a pre-distributed key with possible destinations, such as the sink. A source uses this key together with the identity of the destination node and its own identity to create a shared hidden identity with the destination, which we call *HIP*. When a source wants to set up a routing path to a destination, then it creates a routing request with its own identity, *HIP*, and a sequence number attached to it. The source then broadcasts this packet locally. When an intermediary node receives such a routing request, it first checks the combination of the sequence number and *HIP* to see whether this is a new route request or an old one. If the route request is an older one, then the node drops the packet. Next, the intermediary node checks whether it is the destination by checking *HIP*. If it is not the destination, it keeps the packet, changes the source identity into its own, and rebroadcasts the packet within the network. This process is repeated by all intermediary nodes within the network, until the packet hits the destination. The destination alters its routing table, as it adds a *pathID* together with the *HIP* shared with the source. Next, it also sets the previous hop field in its routing table to the node identity where the destination got the routing request from. Then, the destination node sets up a routing reply with a *PathID* and sends it as a reply to the node from which it got the routing request. Every node that gets the routing reply and it is not the source updates its routing table as follows: it stores the *pathID* in its routing table, sets the previous hop field to the identity of the node where it got the request from, and the next hop field to the identity of the node where it got the routing reply from. The node then forwards the routing reply towards the node where it got the routing request from. When the source gets the routing reply, it updates its routing table by adding the following values to the table: the *pathID*, *HIP*, and the identity of the node from which it got the routing reply as the next node.

The third technique is called anonymous data forwarding, uses the path established by the multi-hop routing technique and the techniques from anonymous one-hop communication to encrypt the data per hop. Jiang et al. [67] argue that the computational overhead of APR is not as power consuming as the transmission overheads in many other solutions. Chen et al. [130] say that without a proper anonymous broadcast in the path discovery process, APR cannot achieve sender anonymity.

Di Pietro et al. [90] introduce max query aggregation (MQA) as a solution that provides SLP, without considerable computational overhead. The solution is based on data aggre-

[1]Note that APR should not be confused with alternate path routing solutions.

gation in which the maximum value is reported to the sink. They assume that the WSN is organized in an hierarchical tree, with the sink as the root of the tree, and the sensors as the leafs of the tree. The authors also assume that every sensor node has an unique symmetric key shared with the sink. The solution works as follows. Every node has sensed a certain value that is captured by an integer of i bits. Now, a WSN goes through i rounds to aggregate all the sensed values of each node, using one bit at the time. In each round i, the sink sends out a random value v to all the nodes in the WSN. Every leaf node in the tree uses the random value, together with the shared key and the i-th bit of a sensed value, to generate a reply to the sink. The leaf nodes send this reply to their parent within the tree. When a parent node received all the values of its children, it calculates the sum of all the messages and forwards this message to its parent node. This process is repeated all the way till the sink. The sink uses the shared keys together with its random value to calculate how many leaves send a bit with value 1. If there are more than zero nodes that send a bit with a value of 1, then the sink asks all the nodes that send a 0 to report as an inactive sensor. An inactive sensor reports a zero at every round, until all the bits of the sensed values have been accumulated. The nodes that send a 1 as a value, on the other hand, are marked as active and continue to send their next bit in the next round. In the case that all the sensor nodes send a 0 to the sink, then the sink does not mark any active nodes as inactive and it continues to the next round. The authors included a probabilistic approach of MQA as well in order to reduce the size of the messages reported by intermediate nodes. In the probabilistic version, an intermediate node does not calculate the sum of the reported values, but it uses an XOR operation.

Another solution against a global adversary comes from Chen et al. [130]. The solution is called efficient anonymous communication (EAC) and provides sender, communication relationship, and base station anonymity. Base station anonymity covers hiding the sink from the adversary: the adversary can no longer see which node within the WSN is a sink. This solution consists of three phases. The first is the network pre-deployment phase, in which each node i gets two random numbers a and b, together with two hash functions *H1* and *H2*, and an *ID*. The network initialization phase follows. First, every node learns its position in the network and the distance to the sink in terms of the number of hops. Then, every node creates an global anonymous identity *AI* based on the hash function *H1*, a, and the node's *ID*. Then, every node creates an anonymous broadcast identity *BAI* based on the hash function *H1*, the node's identity, b, and the node's *ID*. Next, nodes set up an anonymous one hop identity *OHAI* and an anonymous acknowledgement identity *AAI*. A node has one *OHAI* and one *AAI* per neighbour for incoming traffic of that neighbour and one *OHAI* and one *AAI* for outgoing traffic per neighbour. In the next phase, the actual sensing and reporting starts. When a source wants to send data to the sink, it chooses a neighbour based on its forward probability, and sends a message addressed to the *OHAI* of that neighbour. The payload of the message is encrypted with the key shared between the source and its neighbour. Part of the payload is the data towards the sink, which is encrypted with a key shared between the source and the sink. Another part of the data is the *AI* of the source, so that the sink knows which key to use. Every neighbour checks the used *OHAI* to see whether the message is addressed to it. If not, it drops the message, otherwise it knows that it is the next intermediary node. The intermediary node acknowledges the transmission using its *AAI*. When a node participates in a successful transmission, it has to update its *OHAI*, as a source its *AI*, and as a receiver its *AAI*. The intermediary node takes the incoming message, decrypts the payload, replaces the *OHAI* with the *OHAI* of the next hop, and encrypts the payload with the shared key between the intermediary node and the next hop. Each intermediary node goes through the same process, until the message reaches the sink. This solution allows a node to anonymously broadcast as well, using the node's *BAI*.

The next two solutions that come from Nezhada et al. [50] provide two versions of their destination controlled anonymous routing protocol for sensornets (DCARPS). The standard DCARPS works against global adversaries if there is dense traffic. The second version of DCARPS is called probabilistic DCARPS and works against a local adversary if there is no dense traffic. Let us discuss both versions.

Standard DCARPS consists of six parts, of which we discuss the five relevant parts or phases. First, every node gets deployed with a preprogrammed unique *ID*, a random nounce *DL* shared with the sink, and a secret shared key K which is shared between only that node and the sink.

In the second phase, the sink learns the complete topology of the WSN, while in the third phase, the sink calculates a tree that covers the WSN, with the sink as the root of the tree. Every node within the WSN is part of the tree. The sink enforces the tree like organisation by means of labels. The sink assigns labels for uplink communications to each of the nodes. A node gets one outgoing and one incoming label. The outgoing label of a node is the same as the incoming label of a neighbour closer to the sink. The labels are used, later on, as follows: if a node needs to send a packet towards the sink, it uses its outgoing label and embeds it into the packet. When its neighbour hears the transmission and finds a match in its incoming label and the label used within the packet, then it receives the transmission, replaces the label of the packet with its own outgoing label, and retransmits the packet again. The sink needs to calculate the label assignments in such a way that the traffic gets balanced over the WSN and that labels can be reused within the WSN, without confusing a node when it transmits a packet or it overhears a transmission.

In the fourth phase, the sink sets up the routes. The sink sends out a message, which has the form of a cryptographic onion. A cryptographic onion is a structure that consists of multiple encrypted layers. Each layer of the encryption might contain some additional information and the core of the onion contains the final message. In this case, the onion contains at each layer the labels for a node that can decrypt that layer and the *DL* for the next recipient of the onion. The sink sends these onions over the network. When a node overhears a broadcast of an onion, it checks whether the onion contains the node's *DL* as a destination. If that is the case, it accepts the onion, decrypts a layer of the onion, and takes the labels out of the onion. Then, it puts the new *DL* as the label of the onion

and broadcasts it again. This decrypting and forwarding is continued until all nodes have their labels after which the onions are completely decrypted.

In the next phase, the sensing starts. When a source senses something, it encrypts the data via the key it shares with the sink, and appends its own outgoing label to the packet. The neighbour with the correct incoming label receives the packet and re-encrypts the data again using the key it shares with the sink, and adds its own outgoing label and transmits the packet. This process is repeated until the packet arrives at the sink. Note that the size of the packet does not change throughout the procedure. When the sink receives the packet, it decrypts the different layers and finds the actual message. The continuous re-encryption of the message makes it harder for an eavesdropper to correlate incoming and outgoing traffic.

The sixth and last part of DCARPS defines how the sink can send commands using downlink labels. Each node uses a random back-off period before it tries to transmit the packet. This back-off period is mostly used to prevent collisions, but it also helps in decorrelating outgoing and incoming traffic at a node. Phase two till four are repeated over time to change the labels of all the nodes in order to confuse an adversary.

Probabilistic DCARPS works similarly to standard DCARPS. In probabilistic DCARPS, nodes can have multiple outgoing labels and multiple paths towards the sink. Each path has its own *pathID*. There are two versions of probabilistic DCARPS. In one version, every intermediary node can select a random path with a *PathID* to forward the packet along, whereas in the other version only the source selects a certain *PathID*. In the latter case, every intermediary node needs to use the same path corresponding to the *PathID* chosen by the source.

The work from Nezhada et al. [50] got some attention from other authors. For instance, Chen et al. [130] show that both versions of DCARPS cannot provide base station nor communication relationship anonymity as the broadcast tree structure within DCARPS will give away part of the structure of the network due to the difference in anonymous identities.

Luo et al. [79] provide a solution against a local adversary, called anonymous communications scheme (ACS). This solution is part of a larger solution, that consists of RRS, DPIS, and ACS. RRS and DPIS are described in section III-B.2. The aim of ACS is to hide the real identity of every node from an adversary. In ACS, every node shares an individual key with each of its neighbours. A node and its neighbour use the shared key together with their identities to make outgoing and incoming hidden identities based on a hash function. Both the node and its neighbour update the hidden identities periodically using the shared key and the hash function. Each node keeps a link table in which it records the following entries in columns. The first column contains the identity of each neighbour. The second column contains the sequence of the hashing that indicates how many times the hash function is applied to the identity. The third column contains the current outgoing hidden identity of that neighbour. The fourth column contains the current hidden incoming identity of that neighbour. When a node wants to send a packet to its neighbour, it first encrypts the payload with the shared key, and then it addresses the packet to the incoming hidden identity of the neighbour. Note that Park et al. [91] call the approach of ACS a naive solution as it would be too computationally intensive.

Park et al. [91] provide a solution that is less computationally intensive than ACS. This solution is called Phantom ID (PhID). In PhID, sensors have a real *ID*, which is referred to as the source *ID* or *SID* and a phantom *ID*, called *PID*. Nodes know their own locations and the locations of their neighbours. Nodes also share pairwise shared keys with their neighbours for the encryption, and decryption of messages. Lastly, nodes use an adapted version of the simple message authentication code (SMAC) to verify the integrity of the message and the identity of the sender of the message.

Phantom ID works as follows. First, when a sensor is deployed, it uses two pre-distributed parameters q and p to calculate its *PID* as follows: $PID = q^{SID} \bmod p$. Then, every node shares its *PID* with its neighbours encrypted with the pairwise shared keys. Each node stores the following information regarding its neighbours in a table: the neighbour's *PID*, the neighbour's *SID*, and whether the neighbour is an uplink node or a downlink node. Next, a node exchanges its table with its neighbours. If a node receives a table that contains a similar *PID* entry, then it marks the sender of the table with that *PID* within its own table as an uplink or as a downlink neighbour, depending on the position of the neighbour. From here on, a node does not communicate with its actual *SID*, but it communicates with its *PID*, together with a hidden vector *HV*. The *HV* consists of the *SID* of the sender XOR-ed with the shared key of sender and receiver, and XOR-ed with a random number, which both sender and receiver can generate. *HV* is updated regularly as both sender and receiver generate a new random number that is used to refresh the *HV*. When a source sends a message, it combines its *PID*, the *HV*, a timestamp, the destinations *ID*, the encrypted payload, and a modified SMAC code in a message and sends it towards the sink. A receiver of a message uses the *PID* together with the *HV* to see from which node the message came and forwards it to the sink. Note that there is no routing algorithm included in PhID. When the sink gets the message, it uses *PID*, *HV*, and the adapted *SMAC* to check the integrity and authenticity of the message. It then decrypts the payload. Nodes regularly update their *PID* and *HV* to further confuse the adversary.

3) Summary of Anonymization Based Solutions

This section comprised the solutions that focus on using anonymization techniques. The following solutions use pseudonyms to anonymize the traffic: ACS, APR, DCARPS, probabilistic DCARPS, EAC, HIR, RHIR, PhID, SAS, and CAS. Only MQA uses aggregation to provide SLP without using pseudonyms. Solutions like EAC, SAS, and RHIR save the to be used synonyms in the memory of the nodes, while solutions such as HIR and CAS use hash functions to compute a new pseudonym per transmission. We see that only PhID uses a synonym for only the source of a message, while the other pseudonym based solutions use a pseudonym for both source and destination. See Table I in Section IV for an overview of the different solutions and their adversarial models.

Note that there are more solutions that use anonymization techniques and provide SLP, yet they do not aim to provide SLP. Let us mention them shortly for completeness: the quality aware algorithm and resource aware algorithm from Chow et al. [18], the secure hop-by-hop data aggregation protocol from Yang et al. [131], the privacy-preserving robust data aggregation from Conti et al. [132], the privacy preserving data gathering protocol from Koasar et al. [133], general cloaking from Gruteser et al. [134], the homomorphic secret sharing based scheme from Ren et al. [135], the fully-reporting secret pertubation-based scheme from Feng et al. [136], the improved privacy-preserving data aggregation from Bista et al. [137], the integrity protecting hierarchical concealed data aggregation from Ozdemir et al. [138], the privacy preserving based on anonymously shared keys and omniscient sink from Zhang et al. [139], privacy preserving based on anonymously shared keys and ignorant sink from Zhang et al. [139], the data aggregation protocol based on additive homomorphic encrypion from Di Pietro et al. [140], the multidimensional privacy-preserving data aggregation from Lin et al. [141], the integrity-protecting private data aggregation scheme from He et al. [142], the pertubated histogram-based aggregation family from Zhang et al. [143], the data concealment in histogram-based aggregation from Wang et al. [144], the reliable private data aggregation scheme from Rahman et al. [145], the concealed data aggregation family as first described by Castelluccia et al. [146]–[150], [150], [151], and the privacy-preserving data aggregation family as described in [137], [141], [150], [152], [153]. We should note though that the attestation processes described in [153], [154] could compromise SLP.

H. Cross-layer Routing

In this section, we discuss the solutions that use cross-layer routing to provide SLP. We shortly introduce the concept in Section III-H.1. Next, we discuss the solutions in Section III-H.2 and summarize our findings in Section III-H.3.

1) History and Introduction

Nodes normally use the network layer to exchange packets that contain information on sensed events. The solutions we discussed thus far mostly concentrate on activities at the network level. The fake or dummy packets that we often refer to are network level packets as well. The solutions in this section differ from the solutions discussed so far, as the solutions in this section use the underlying layers of the communication protocol stack in a different way. Solutions that use cross-layer routing do not only utilize the network level layer to exchange messages about events, they use control messages from the medium access control (MAC) layer as well. Deviating from the standard network protocol stack is not new for a WSN. In fact, there are many cross-layer optimizations known for WSNs in order to prolong the battery life of the nodes and enhance the network throughput [155], [156].

Both solutions in this category defend against a local adversary, come from Shao et al. [47], and are called the cross-layer solution and the double cross-layer solution.

2) SLP via Cross-layer Routing in the State of the Art

Shao et al. [47] introduce two solutions: the cross-layer solution (CLS) and the double cross-layer solution (DCLS). Shao et al. [47] assume, for both the solutions that the WSN uses a beacon-enabled version of IEEE 802.15.4 [157]. This means that the nodes need to exchange a beacon frame once in a while. Nodes can sleep between beacons to increase battery life, whereas the beacon interval can range from 15,36 ms to 786,432 seconds. The beacon frame contains the superframe specification, MAC control fields, and one optional field: the beacon payload field. The beacon payload field can be filled by protocols of a higher layer. IEEE 802.15.4 provides multiple link layer security services: symmetric data encryption, frame integrity, access control, and data freshness. These link layer security services secure the beacon payload field and make it an interesting medium for information exchange.

In CLS, nodes use the payload field of the beacon frames to exchange data. The beacon frames might go unnoticed if the adversary does not listen to them and only checks the network layer for traffic. The only problem is that a beacon frame interval of at most 786,432 seconds might introduce too much latency. That is where the pivot nodes come into play. A pivot node is an ordinary node within the WSN, which is selected by the source. The pivot node takes the event information from the beacon frame and routes it to the sink via the network layer. Let us discuss the first part in a little more detail. If a source detects a subject, then it marks a node that is at least h hops away as the pivot node. Then, the source encodes its node *ID*, an event *ID*, and a timestamp into a beacon frame together with the *ID* of a pivot node and broadcasts its beacon. All the neighbours receive the beacon, decrypt the beacon to extract the event information from the payload field, and add the event information into their own beacon frame. They all send their beacon frame at the next interval, which is then processed by all their neighbours. This process is repeated for at least h hops and at least one node transmits its beacon to the designated pivot node. Note that nodes need to record event information from the beacon frames for a certain amount of time in order to recognise old information of passing beacons. When a node receives a beacon with old event information, then it overwrites the information with dummy data. Note that if h is too low, then the adversary might still spot the subject and the source.

DCLS works similar to CLS. A source first sends the event via the beacon to a randomly chosen pivot node that is h hops away. The pivot node routes the information to a random intermediary node via the network layer. This random intermediary node starts the normal CLS procedure in the same way as a source does. The intermediary node sends the event information, using the real source's *ID* and puts the information in a beacon. The beacon is then flooded through the WSN where it hits another pivot node. This second pivot node routes the event information to the sink.

The cross-layer routing approach received some attention from the literature. Rios et al. [94] mention CLS quickly and in [21] they provide a larger summary of CLS. Reindl et al. [158] and Kazatzopoulos et al. [102] both quickly mention CLS.

3) Summary of Cross-layer routing Based Solutions

In this section, we described the two cross-layer routing based solutions from Shao et al. [47]. Cross-layer routing

solutions only work against a local adversary, given the assumption that the adversary does not inspect beacon frames or other low level messages. Cross-layer routing can introduce a lot of latency, depending on the distance between the source and the pivot node, and depending on the beacon interval time. Double cross-layer routing can even further confuse an adversary.

I. Separate Path Routing Schemes

In this section, we discuss the solutions that use separate path routing schemes. We first look at the history and introduction of separate path routing in Section III-I.1. Then, we look at the solutions that use separate path routing in Section III-I.2 and we summarize our findings in Section III-I.3.

1) History and Introduction

A local adversary often uses a hop-by-hop-trace attack to trace a stream of packets all the way to the source. The number of packets that an adversary needs to find the source depends on his hearing range and the distance between the adversary and the source. If the source is rather far away and the adversary has a limited hearing range, then the adversary often needs multiple packets before it can find the source. Separate path routing complicates the hop-by-hop-trace attack as packets no longer follow the same path. In separate path routing, a source randomly chooses a path out of the available set of paths and routes the packet via that path towards the source. Every path has a certain probability to be chosen by the source, which is called the message probability. If an adversary cannot overhear all the separate paths, then the adversary does not overhear all the packets anymore, which means that the adversary has to wait until a packet is again forwarded on the path that it can overhear and correlate it to the source. We will see in Section III-I.2 that this approach can have its use against a global adversary as well.

Separate path routing is a special form of multipath routing that was first used to improve the resiliency and throughput of a WSN. Authors, like Dulman et al. [159] and Lou et al. [160] refer to separate path routing as disjoint multipath routing. Their work was focused on improving the throughput and resiliency of the network, while they did not took SLP in consideration. Later on, Wang et al. [92] started to use separate path routing as a core technique to provide SLP.

We discuss the following three solutions from Wang et al. [92] in Section III-I.2: random parallel routing, weighted random stride routing, and weighted random stride routing adapted towards a global viewing adversary.

2) SLP via Separate Path Routing in the State of the Art

The first solution from Wang et al. [92] is random parallel routing (RP). With RP, every node is assigned n parallel non-intersecting routing paths that lead from that source towards the sink at deployment time. These routing paths are dispersed over the WSN in such a way that these paths do not cover a considerably large geographical area. If the paths would cover a considerably large geographical area, then the packets can easily attract local adversaries and routing the packets via these paths would have a large cost in terms of energy. The message probability of the paths is distributed in relation to the length of the path: the message probability increases as the path length increases. The strength of RP lies in its path separation. The path separation makes it harder for a local adversary to do a fast hop-by-hop-trace attack. The problem with RP is that the separate paths leak direction information to a local adversary, as the adversary can see where the packets are coming from when the adversary is close to the sink. Another problem with RP is that every intermediary node needs to know the path for each source that uses this intermediary node. This solution got some attention from other authors in the literature. Authors such as Rios et al. [21], [94] summarize RP.

The next solution in this category from Wang et al. [92] is called weighted random stride (WRS). This solution works similar to PRLA as described in Section III-B.2. This solution uses a different terminology: the inclination angle from PRLA is now called the forwarding angle. Like in PRLA, every node knows its location and the location of the sink. A node uses this information to split its radio reach in different degree-based sectors, based on the forwarding angle towards the sink. When a source needs to send a packet, it randomly selects one of the sectors and sends the packet to a neighbour within that sector. Then, intermediary nodes need to forward the packet in the same direction for at least *stride* amount of hops. The next node to hold the packet needs to select a new sector and the intermediary nodes then need to forward the packet for a preset amount of hops in that direction. This process is repeated until the packet reaches sink. The message probability of a sector is related to the actual forwarding angle of that sector. The larger the forwarding angle, the higher the message probability of that sector. This solution got some attention from other authors in the literature. Authors such as Rios et al. [21] summarize WRS.

The last solution from Wang et al. [92] is called weighted random stride routing adapted towards a global viewing adversary (WRS extended). Wang et al. [92] follow an approach that is similar to ConstRate. This solution uses slotted transmission and fake traffic to hide the actual transmissions against an adversary. If a node does not have a real packet to send during a transmission slot, then it sends a dummy packet. Wang et al. [92] assume that real traffic can be generated in bursts, with a higher burst rate than the slotted transmission rate. Therefore, they provide an algorithm with WRS extended to to find N disjoint paths that allow a source to route the event information through several disjoint paths as it cannot send the burst through one path only. The paths are chosen in such a way that the total length of all the paths is minimized, which means that the used energy for the reporting is minimized.

3) Summary of Separate Path Routing Based Solutions

This section described three solutions that are based on separate path routing. Both RP and WRS extended the use of pre-defined paths to route information from the source to the sink, whereas WRS does not need pre-defined paths. We see that separate path based solutions only work against a local adversary unless they are combined with additional measures such as dummy traffic.

Note that there are other works that use disjoint multipath routing in such a way that they provide SLP as a byproduct against a local adversary. We mention a few for completeness: Lou et al. [160] and Dulman et al. [159].

J. Using Network Coding

In this section, we discuss the solutions that use network coding to counter traffic analysis. We briefly look at the history of network coding in Section III-J.1. Then, we discuss the solutions that use network coding to provide SLP in Section III-J.2, and summarize our findings in Section III-J.3.

1) History and Introduction

In most traditional networks, nodes can route or forward a message. Each message on an output link of a node is either generated by that node or is a copy of a message of an input link of the node. Network coding differs from the traditional routing as each node is now allowed to perform computations on the messages. The main type of computations include mixing (encoding) or re-mixing (re-encoding) of the messages. The destinations of the messages can unmix (or decode) the messages again. With network coding, nodes can use multiple paths to route (parts of) the messages and combine messages to send them together in one message towards the sink. Network coding delivers many benefits: a potential throughput improvement, minimization of energy used during transmission and a minimization of the delay [45], [161]. There are several applications for network coding, such as: multimedia streaming, P2P networks, tactical communications in military networks and sensor networks [45]. One of the earliest notions of network coding comes from Ahlswede et al. [162]. They show in their work how the application of network coding can save bandwidth in a network. Later on, authors as Rabbat et al. [163] show how network coding can be used to efficiently distribute data through a WSN. Chou et al. [161] deliver a clear in-depth explanation of how network coding works.

Fan et al. [44], [45], [164] look at how network coding can provide SLP. One of the benefits of network coding in this context is that the incoming and the outgoing packets are unlinkable when they are mixed and buffered, which makes it harder for an adversary to do a successful traffic analysis attack. The only problem is that decoding by the adversary is still possible and the linear dependence can still be analyzed. In fact, the attached global encoding vectors (GEV or tags) provide insight in the encoding mechanisms and allow attackers to break SLP. If an attacker gathers enough packets, then it can analyse the information inside the packets and find the original messages. That is why Fan et al. [44], [45], [164] introduce two different solutions that provide SLP in a WSN as described in Section III-J.2. The first solution is the privacy-preserving scheme for network coding [44], and the second solution is the source unobservability by network coding [45].

2) SLP via Network Coding

The first solution in this category is the privacy-preserving scheme for network coding (PPSNC) by Fan et al. [44], [164]. The aim of the solution is to hide the tags or *GEV*s attached to every message, so that an adversary cannot find the original messages and it can not correlate the incoming traffic of a node to the outgoing traffic of a node. Fan et al. [44], [164] propose to encode the *GEV*s with homomorphic encryption functions (HEFs). Homomorphic encryption allows sources and intermediary nodes to execute mathematical operations on the ciphertext, which can be decoded by the sink after the operations. The application of *HEF*s, therefore, allows nodes to re-encode packets with an encrypted *GEV*. Fan et al. [44], [164] recommend using the Paillier cryptosystem as described in [165]. The Paillier cryptosystem is a public key cryptosystem that satisfies the homomorphic properties of additivity and scalar multiplicativety. Fan et al. [44], [164] assume that there are multiple sinks, sources, and intermediary nodes. Every sink has a public encryption key and a private decryption key from a trusted authority. Furthermore, the public encryption key and the generation number of a packet are hidden by the routing scheme itself. When a source needs to send out a set of messages, then it prefixes the unit vectors to the messages, chooses a random local encoding vector (*LEV*), and performs a normal linear encoding operation on the messages. The tags attached to the message are encrypted using the HEFs. Intermediary nodes buffer the incoming packets, until they have a number of packets that belong to the same generation. Then, an intermediary node applies a random *LEV* and computes a new linear recombination of the packets with which the node practically mixes the incoming packets and creates new packets out of them. The intermediary node cannot easily decrypt the ciphertext, as it misses the decryption key, but it can apply recoding operations on the ciphertext. The sink decodes the encoded tags using its private decoding key, and creates the global encoding matrix to decode the messages. Note that PPSNC only works well if there is dense traffic. If the traffic is rather sparse due to the lack of multiple sources, then the traffic becomes easier to analyse as the flow is more visible.

The second solution from Fan et al. [45] is source unobservability by network coding (SUNC). This solution uses the mechanisms from PPSNC to hide the *GEV*s. Fan et al. [45] add special dummy messages to the solution. The dummy messages consist of homomorphic ciphertexts and have the property of dummy nutility (i.e. that a dummy message itself has no value when decoded). In other words: a dummy message is absorbed in the re-encoding process when it is mixed with real traffic. The intermediary node does not notice whether it is handling a dummy or a real message, even though it filters out the dummy message. The property of not seeing whether the message is a dummy or a real message is called forwarder blindness. Forwarder blindness also prohibit an adversary that compromises a node to distinguish real from fake messages. Nodes use a similar approach to ProbRate: a node sends a real message when it has something to report or forward, otherwise it sends a dummy message. All nodes send their messages on an interval with a specified probability distribution. The combination of network coding and dummy traffic make incoming packets unlinkable to outgoing packets, which prevents traffic analysis attacks. Fan et al. [45] argue that SUNC delivers source unobservability [44], [45], [164].

3) Summary of Network Coding Based Solutions

In this section, we discussed two solutions from Fan et al. [44], [45], [164]. The first solution, PPSNC, only works in situations with dense traffic and will confuse an adversary. The second solution, SUNC, will generate enough traffic by itself to hide the real sources. Both the solutions show that network coding together with homomorphic encryption creates a feasible basis for countering traffic analysis and node compromise of a global adversary.

There are other works in the literature that deliver a similar result to the works mentioned in this section. These works are not discussed in Section III-J.2, as they do not aim to provide SLP. We still mention them here for completeness: concealed network coding from Hessler et al. [166], the network coding for gradient based routing from Miao et al. [167], and the location-aware network-coding security from Ayday et al. [168]. Note that the last two approaches might deliver only limited SLP as they do not encrypt the GEVs.

K. Limiting Node Detectability

In this section, we discuss solutions that aim to provide SLP through limiting the detectability of the nodes. We introduce this category in Section III-K.1. Next, we describe the solutions that belong to this category in Section III-K.2, and summarize our findings in Section III-K.3.

1) History and Introduction

Solutions in this category use the physical layer of the WSN's network protocol stack to provide SLP. The solutions aim to hide either the location of a node or the presence of a transmission. There are two general approaches within this category. The first approach is to silence a node, so that it is harder for an adversary to locate the node. If the node remains silenced during an event, then the event itself will go unnoticed to the adversary as well. The second approach is to lower the radio transmission power, so that a local adversary might not detect the transmission. The second approach can be extended to varying the radio transmission power. If a node varies its transmission power, then it is harder for an adversary to find the exact location of the node that is transmitting. The techniques behind the approaches mentioned above are not specifically designed for SLP. Take the work by Correia et al. [169] for instance. They provide a protocol in which the radio transmission power is more controlled to enhance the throughput of the network and to reduce the energy used by the sensors. Later on, Dutta et al. [93] and Tavli et al. [96] use these approaches to limit the detectability of the nodes.

This section comprises the following solutions as described in Section III-K.2: anti localisation by silencing [93], context-aware location privacy [94], hidden anchor [95], hyberloc [51], lowering radio transmission power [96], and multi cooperator power control [97].

2) SLP via Limiting Node Detectability in the State of the Art

The first solution in this category is lowering radio transmission power (LRTP) by Tavli et al. [96]. They argue that it is hard for a local adversary to get into the WSN without being detected. They assume that a local adversary wants to remain undetected, which is why they expect an adversary to eavesdrop the WSN from a position outside of the WSN. If the nodes within the WSN limit their radio transmission power and thereby their radio range, then it will be harder for a local adversary to detect the transmissions from outside the WSN. Tavli et al. [96] point out that reducing the radio range does mean that nodes have to relay more messages. If a node has a radio range that spans multiple nodes towards the sink, then it can send its messages to a node that is closest to the sink in which case the message skips a few intermediary nodes. In case of a limited radio range, the node can no longer skip the other intermediary nodes and must route the packet via these intermediary nodes. Tavli et al. [96] present a linear programming framework to compute the optimal radio transmission power for each node given the topology of the WSN. The calculated optimal radio transmission power provides a balance between enhanced privacy and optimal lifetime of the network (Oh et al. [97] quickly mention the solution and note that this solution will not work if the adversary installs enough of its own sensors).

The next solution comes from Dutta et al. [93] and is called anti-localisation by silencing (ALbS). The goal of this solution is to locate a local adversary within the WSN, while the nodes within the WSN must remain undetected by the adversary. Dutta et al. [93] assume that the WSN is organized in grids and that nodes know their own position. They furthermore assume that nodes are able to detect adversaries using vibration or infrared sensors. A grid of sensors can detect adversaries within the grid and near the borders of the grid. The adversarial localisation precision is at the grid level: if a grid can find the adversary, then the adversary is localized, otherwise the adversary is lost. Nodes at the border of a grid detect whether an adversary enters their grid and nodes at the center of a grid can detect whether the adversary is leaving. The detecting nodes inform the other nodes in case of a grid switch of the adversary. Nodes maintain silence when the adversary is inside their grid for as much as possible. The only nodes that communicate at that point, are the nodes that need to inform others about the grid-switches of the adversary.

Rios et al. [94] introduce with their context-aware location privacy (CALP) a similar approach to ALbS. Rios et al. [94] position CALP as an additional software module that can be installed on a node and that only works against a local adversary. CALP relies on the assumption that nodes have sensors with which they can detect the presence of both subjects and adversaries within the area. The subjects are able to identify themselves with an authentication technique, whereas the adversaries cannot authenticate themselves. When a node detects an adversary, it broadcasts a warning message, which includes the distance of the adversary in terms of the number of hops that the adversary is away from the node. When a node receives such a warning message, it increases the hop counter of the message and forwards the message to its neighbours. The warning message is sent across the WSN, informing each node of the presence of the adversary. Nodes can also use the MAC level beacon message [157] to exchange the warning message, which might hide the information exchange on the cost of additional delay. Rios et al. [94] assume that the

detection range of the adversary is known. They incorporate a minimum safety distance, which is the minimal distance, in number of hops, that a node needs to be away from the adversary to safely transmit a packet without being noticed. They define two different methods to cope with the presence of the adversary. The first method is strict CALP routing. In strict CALP routing, a node that is within the minimum safety distance is not allowed to forward a packet to a neighbour. The node needs to buffer the packet until the adversary has moved far enough. If there is an adversary within the WSN and the node is not within the minimum safety distance, then the node should route the packet on a path that does not hit the minimum safety zone. Strict CALP hides the transmissions from a local adversary, if the minimum safety distance is chosen correctly. The problem with strict CALP is that it can also create cycles, high consumption of energy or even packet drops if the adversary is too close to the sink. The second method is permissive CALP. In permissive CALP, a node does not alter the path of the packet, unless the node is within the minimum safety distance. If a node is within the minimum safety distance of the adversary, then it routes the packet away from the adversary. Permissive CALP is applicable for non-delay-tolerant applications, but does provide less SLP as an adversary can still trace the packets.

Oh et al. [97] provide a jamming based approach with their multi cooperator power control (MCPC) solution. They describe how adversaries can use localisation attacks based on the received signal strength (RSS) and the time of arrival (TOA) to locate a node. See [97] for an explanation of both methods. When nodes start to collaboratively jam transmissions, then both the adversarial localisation methods will become imprecise. The source of the imprecision stems from the transmission power used by the jamming nodes. This transmission power complicates the location calculations as it becomes harder to find the actual signal strength of the source that is sending a message and the signal to interference and noise ratio.

The last two solutions in this section aim at confusing an internal adversary that compromises a node. The first solution is Hyberloc from El-Badry et al. [51]. El-Badry et al. [51] assume that the WSN uses a localization technique that relies on the anchor beacons of anchor nodes. In Hyberloc, anchor nodes vary their transmission power per beacon frame transmission according to a certain statistical distribution. The beacon frames are encrypted and contain the information necessary to correct the location calculation, such as the used distribution and the actual used transmission power. Trusted nodes can decode the beacon frames and use the information to correct their location calculation, whereas untrusted nodes cannot. If a node cannot locate itself correctly while compromised, then the node becomes less usable for eavesdropping in certain routing scenarios. El-Badry et al. [51] aim to provide physical layer location privacy with this solution, which could result in SLP in case of a local internal adversary. According to Rania et al. [95], [170], Hyberloc is not fully feasible given the current state of the sensors. They argue that the Hyberloc solution only obfuscates the location and the beacon related data, so a compromised node can still identify it as an anchor node. It is because of the current state of the hardware sensors that they introduced their solution called hidden anchor (HA). In HA, anchor nodes no longer use variable transmission power, but they use the *ID* of one of the trusted nodes within their neighbourhood, which hides the anchor beacon into the noise of the WSN.

3) Summary of Limiting Node Detectability Based Solutions

This section presented several solutions that either attempt to hide the location of a node or the presence of a transmission. We saw that LRTP and CALP both try to hide transmissions. Nodes limit their radio range in LRTP, whereas nodes adapt their routing in case of a detected local adversary in CALP. The ALbS solution takes the principle from CALP one step further and hides the node's location by silencing the nodes in case of a near adversary. Solutions such as HA and Hyberloc aim to confuse the local active adversary that tries to understand the WSNs topology while eavesdropping via compromised nodes.

L. Others

In this section, we discuss solutions that do not do not fit in the other categories. We first introduce the category in Section III-L.1. Next, we discuss the solutions that we found in Section III-L.2 and we summarize our findings in Section III-L.3.

1) Introduction

In this category, we look at specific solutions for specific problems and classes of a WSN that normally do not provide SLP. We briefly discuss two solutions in Section III-L.2. The first solution is the privacy-enhanced data centric sensor [98], which is designed for a subclass of WSNs, called data-centric WSNs or data-centric sensor networks. Data-centric WSNs are normally not designed to provide SLP and use different hardware than normal WSNs. The second solution is privacy preserving hop-distance computations [6] in which two nodes can calculate the hop-distance between each other, while hiding their actual locations from an eavesdropping adversary. Normally, protocols that compute the hop-distance already give away the locations of different nodes, which makes it very easy for an adversary to get an overview of the topology.

2) Other Solutions in the Literature

Shao et al. [98] introduce the privacy-enhanced data centric sensor (pDCS) as a solution to provide SLP to data-centric sensor networks. A data centric sensor (DCS) based WSN differs from a standard WSN as it does not have a sink. In a DCS, the sensor itself is not as important as the data. The data is identified based on attributes such as event-type and the geographic location of the event. The data is spread among storage cells in the network based on properties of the data. Let us take the wildlife tracking application as an example. In a normal WSN, all event data would flow from the sensors to the sink. In a data centric sensor network, each sensor sends data about a specific animal to a specific node within the WSN. This specific node or group of nodes is called the storage cell of that data. If a user wants to know something about a certain type of animal, then the user, or a mobile sink, travels to the storage cell that keeps record of

the data of that specific animal after which the user queries the storage cell. The solution does not incorporate its own traffic analysis countermeasures. Shao et al. [98] recommend to look at other techniques to resolve this, such as mixes, delay, fake traffic or other techniques. The solution has three different types of implementation. All three implementations follow the following steps. A cell first determines the location of its storage cell through a keyed hash function. Next, the node encrypts the recorded information with its cell key. The node forwards the message towards the storage cell. When the storage cell receives the message, it stores it locally. If an authorized mobile sink or user is interested in an event in a specific area it will query the storage cells related to it. After the sink retrieves the data it will decrypt it using the cell-key of the cell that sensed the data. The combination of different techniques together with pDCS does provide SLP in a DCS network.

Many of the solutions in this paper require the node to know the hop-distance between itself and the sink or between itself and an intermediary node. Mingjun et al. [6] provide a secure solution to calculate the hop-distance between two nodes, without giving away the locations of the node. This solution is called the privacy preserving hop-distance computation (PPHC). The solution consists of five steps in which the two nodes exchange a set of vectors and calculate their hop distance based on the hop count of the exchanged messages and the location of the nodes.

3) Summary of Other Solutions

In this section, we briefly discussed two very different solutions that provide (source) location privacy or assist in providing SLP. The pDCS solution provides contextual privacy and SLP for a data centric network and PPHC provides location privacy when two nodes need to compute the hop-distance between them.

Note that there are a few papers in this area, that provide SLP in very specific scenarios even though they were not designed to do so. For instance, the authors of [171] describe what methods provide network level privacy in IEEE 802.11b based networks. Another example is how privacy is delivered to multimedia WSNs in [172].

M. Notes on Flooding

This section is dedicated to flooding, as it has received quite a lot of attention from the literature. We would like to start this section by quoting Ozturk et al. [14]: *"...flooding provides the least possible privacy protection since it allows the adversary to track and reach the source location within the minimum safety period."*. Still, we find many commentaries on flooding in relevant works that focus on providing SLP, such as [12]–[14], [19], [30], [50], [68], [74], [76], [77], [79], [80]. We basically find two techniques that constantly appear in the literature.

The first technique is baseline flooding (BF), as described by Ozturk et al. [14]. In BF, a node broadcasts a packet to its neighbours, its neighbours then broadcast the packet to their neighbours. This process continues until all nodes within the WSN have received the packet. When the number of nodes in a WSN increases, so does the number of broadcasts and the power consumption [68]. Flooding does not stop an adversary from finding the source as flooding often translates in flood-based shortest-path routing and often reuses the same paths for the next transmission. There are several power consumption reducing improvements proposed, such as: forward a packet only once or use accumulative broadcasting [13], [68], [173].

The next technique is probabilistic flooding (PF), which is described by Ozturk et al. [14] as well. Kamat et al. [13] call this technique flooding with forward probability. In PF, nodes use probability p to randomly select a number of nodes where it multicasts the message to. This technique provides a better safety period than BF, but it is still not usable as it consumes a lot of energy and provides a relatively low safety period.

Some techniques that are called flooding, do not flood the WSN at all. Take flooding with fake messages from Ozturk et al. [14] for instance. Flooding with fake messages consists of short-lived fake source routing and persistent fake source routing, which both do not flood the network, they just let nodes broadcast a dummy message. Another example is phantom flooding from Ozturk et al. [14]: phantom flooding is what we call PRS. We renamed phantom flooding to PRS because the routing strategy in itself is not a flooding strategy, but a combination of one of the two mentioned flooding techniques together with a random walk.

IV. Summary and Overview: Solutions and the Related Adversarial Models

In this section, we compare the solutions discussed in Section III and present an overview of all the solutions in Table I. We grouped the solutions in Table I per category and then ordered the solutions based on their acronym. For each solution, we provide an overview of the threat model related to the solution, and mention whether we found issues in the literature regarding the solution. The threat model of each solution is summarized based on the adversarial capabilities that we described in Section II-B.

Table I has the following horizontal structure: The first column mentions the category. The second column names the acronym of the solution and its primary literature reference. The third column names the active attacks of the adversary, which the solution needs to cope with. The fourth column contains the passive attacks where the solution needs to defend against. The fifth column depicts the view of the network an adversary has. The sixth column summarizes the information that an adversary has about the WSN. The last column depicts whether we found issues related to this solution in the literature. Furthermore, we use the following syntax within Table I.

- *Active attacks column (see Section II-B.1 for more details):*
  - **Denial of service:** This active attack allows the adversary to block all further communication of one or more nodes by a denial of service attack.
  - **NA:** (Not Available:) There is no information available on this subject.
  - **Node compromise:** There are two versions of node compromise:

  - ∗ *Active node compromise.* If a node compromise is mentioned in the "active attacks" column, then the adversary uses a compromised node to influence the protocol or to detect the presence of other nodes. An adversary can also destroy a node in this case.
  - ∗ *Passive node compromise.* If a node compromise is mentioned in the "passive attacks" column, then the adversary uses a compromised node to get information such as the identity of a node, the information received and sent by a node, and the encryption keys of a node.
- **None:** It is explicitly stated in the literature that there's no instance of this type of attack.
- **Packet alteration:** The adversary can intercept a packet and forward the packet with altered content.
- **Packet drops:** The adversary can drop packets.
- **Packet injection:** The adversary can insert its own packets into the network.

- *Passive attacks column (see Section II-B.1 for more details):*
  - **AoA:** Angle of Arrival: This passive attack allows an adversary to see in which direction a transmitting node lies. This attack is a specific subclass of "locating a node" and requires special hardware.
  - **Eavesdropping:** The adversary can overhear or intercept messages with this passive attack, but the adversary cannot decode them.
  - **Hob-by-hop-trace:** The adversary can track a stream of messages from one intermediary node to another by overhearing the transmissions. A successful hop-by-hop-trace attack ends at the source of the stream of messages.
  - **Identity-analysis attack:** The adversary can link one or more overheard identities to one or more nodes.
  - **Locating a node:** An adversary can find a node's physical location, but the exact method is not defined. See AoA and RSS as subclasses of this attack.
  - **Mapping attack:** This attack is only applicable to data centric sensor networks as described by Shao et al. [98]. An adversary can map the data type to a cell.
  - **Rate monitoring:** This passive attack is a subclass of "traffic analysis". The adversary looks for nodes that have a higher transmission rate as they might be closer to the sink or source.
  - **RSS:** This passive attack attack is a subclass of "locating a node". The adversary uses special equipment to measure the signal strength that the adversary receives from the transmission of a node. This signal strength is then used to calculate the distance from the adversary to the transmitting node.
  - **Size correlation:** This attack is a subclass of "traffic analysis", in which an adversary is capable of understanding the relation between incoming and outgoing traffic of a node based on the size of the packets.
  - **Time correlation:** This attack is a subclass of "traffic analysis" in which an adversary can use timing information to relate traffic from one node to another node and find a path from source to sink.
  - **Timing analysis:** An adversary is able to execute the timing analysis attack to understand the structure of the WSN. This attack is a subclass of "traffic analysis".
  - **Traffic analysis:** The adversary is capable of doing traffic analysis, but the exact method is not defined. The "rate monitoring", "timing analysis", and "size correlation" attacks are subclasses of this attack.

- *View of network column (see Section II-B.2 for more details):*
  - **Global:** The adversary has a global view of the network.
  - **Local:** The adversary has a local view of the network.
  - **Multi-local:** There are multiple adversaries with a local view collaborating and exchanging information. Other types of a multi-local adversary include a semi-global adversary or a very powerful local adversary.

- *Exposed information column (see Section II-B.4 for more details):*
  - **Identity / identities of....:** The adversary knows the identity of the mentioned object or objects.
  - **Location of...:** The adversary knows the location of the specified object.
  - **Methods used at the local nodes**: The adversary knows which methods are used within a node for communication and reporting.
  - **NA:** (Not Available:) There is no information available on this subject.
  - **Parts of the routing algorithm:** The adversary knows parts of the routing algorithm that is used within the network.
  - **Protocols:** The adversary knows the protocols used by the nodes within the WSN.
  - **Range of identities:** The adversary knows the range of identities used within the network.
  - **The topology:** The adversary knows the topology of the WSN.

- *Issue column:*
  - **X:** There are issues with this solution that compromise SLP, which are described in Section III.
  - **V:** There are issues with this solution that might compromise SLP, which are described in Section III.
  - **-:** We did not find any issues with this solution that compromise or might compromise SLP.

Note that we do not explicitly include the access level and compliance as defined in Section II-B.1. We did not include the access level, as this would make the table too big to fit the page. Instead, we group both the internal and external attacks in the categories active and passive. Node compromise for instance, is an internal attack, while eavesdropping is an external attack. We did not include the compliance level, as all attacks follow a semi-honest pattern.

TABLE I: Solutions and their adversary models.

Type	Solution	Active attacks	Passive attacks	View of network	Exposed information	Issues
Random Walk	AMNS [76]	None	Eavesdropping & hop-by-hop-trace & locating a node	Multi-local	NA	-
	C1, NMR [105]	NA	Eavesdropping & hop-by-hop-trace	Multi-local	NA	-
	DROW [30]	None	Eavesdropping & hop-by-hop-trace & locating a node	Local	Location of the sink & the methods used at the local nodes	X
	GROW [77]	NA	Hop-by-hop-trace & locating a node	Local	NA	X
	LPSS [61]	Compromise of source & sink	Eavesdropping & hop-by-hop-trace & AoA-identification	Local	Parts of the routing algorithms	X
	MussP [73]	None	Eavesdropping & hop-by-hop-trace & locating a node	Multi-local	The location of the sink	-
	OpRo [24]	None	Traffic analysis & locating a node & hop-by-hop-trace	Local	The location of the sink	-
	PRS [13]	NA	Eavesdropping & hop-by-hop-trace & RSS & AoA	Local	The location of the sink	X
	PRLA [25]	NA	Eavesdropping & hop-by-hop-trace & locating a node	Local	The location of the sink	-
	PSRS [13]	NA	Eavesdropping & hop-by-hop-trace& RSS & AoA	Local	The location of the sink	X
	QMNS [76]	None	Eavesdropping & hop-by-hop-trace & locating a node	Multi-local	NA	-
	RaRo [78]	None	Eavesdropping & hop-by-hop-trace	Local	The topology & the protocols	X
	RRIN [76]	None	Eavesdropping & hop-by-hop-trace & locating a node	Multi-local	NA	V
	RRS & DPIS (& ACS) [79]	NA	Rate-monitoring & time correlation & locating a node & hop-by-hop-trace & identity-analysis attack	Local	NA	-
	SADRW [26]	NA	Eavesdropping & locating a node & hop-by-hop-trace	Local	NA	-
Geographic Routing	(R)IRL [19]	Packet injection & packet drops	Eavesdropping & hop-by-hop-trace & AoA-identification & RSS	Local	Range of identities & identity of the sink & public key of the sink	-
	STaR [80]	NA	Compromise node & Traffic analysis & locating a node	Local	NA	V
Delay	(ex-) PRESH [70]	Node compromise & packet injection & packet drops	Eavesdropping & rate monitoring attack[2]	Local	NA	V
	RCAD [63]	None	Eavesdropping	Local	Identities of the nodes & the topology & the routing protocol involved	-

Continued on next page

[2]The adversary's capability of doing the rate monitoring attack is implicitly defined in [70].

TABLE I – continued from previous page

Type	Solution	Active attacks	Passive attacks	View of network	Exposed information	Issues
Dummy Data Sources	A-real & A-fake [58]	NA	Eavesdropping, rate monitoring & time correlation	Global	The topology & parts of the routing algorithms[3]	-
	ASLP [81]	NA	Traffic analysis & rate monitoring & time correlation	Global	The topology	-
	CSPSLP [60]	None	Traffic analysis & locating a node	Multi-local	The protocols & location of sending nodes & location of the sink	-
	Constrate [57]	None	Eavesdropping, rate monitoring & time correlation	Global	The topology	-
	DBT & ZBT [28]	NA	Eavesdropping & hop-by-hop-trace & RSS & AoA	Local	The protocols	-
	DRAA [46]	None	Eavesdropping & traffic analysis	Global	The topology	-
	DWUS [68]	None	Eavesdropping	Local & global	The topology	V
	FS1 & FS2 [29]	NA	Eavesdropping & locating a node	Multi-local	None	V
	GAFG [82]	NA	Eavesdropping & traffic analysis	Global	The topology & distribution of events	-
	GOA [72]	None	Eavesdropping & traffic analysis	Global	The topology	V
	HGA [72]	None	Eavesdropping & traffic analysis	Global	The topology	-
	OFS [62]	None	Eavesdropping & traffic analysis	Global	The topology	-
	PBA [72]	None	Eavesdropping & traffic analysis	Global	The topology	-
	PFSR [13]	NA	Eavesdropping & hop-by-hop-trace & RSS & AoA	Local	The location of the sink	X
	PeCo [15]	NA	Eavesdropping & traffic analysis	Global	The topology	-
	PFS & TFS [27]	None	Eavesdropping & rate monitoring & time correlation	Global	The topology & identities of the nodes	V
	ProbRate & FitProbRate [57]	None	Eavesdropping, rate monitoring & time correlation	Global	The topology	X
	SECLOUD [52]	None	Traffic analysis & rate monitoring & time correlation	Global	The topology	V
	SLFSR [13]	NA	Eavesdropping & hop-by-hop-trace & RSS & AoA	Local	The location of the sink	X
	SoSi [15]	NA	Eavesdropping & traffic analysis	Global	The topology	-
	TARP [69]	NA	Timing analysis	Local	NA	V
	TCH-WSN [84]	Node compromise	Traffic analysis	Global	The topology	-
	TESP^2 [83]	None[4]	Eavesdropping & traffic analysis	Global	The topology	-
	UHT [55]	None	Eavesdropping & hop-by-hop-trace[5] & rate monitoring & time correlation	Global	The topology & identities of the nodes	-
Cyclic Entrapment	CEM [85]	NA	Eavesdropping & hop-by-hop-trace	Local	NA	-
	iHIDE [86]	Node compromise	Eavesdropping & hop-by-hop-trace	Local	The topology	-

Continued on next page

[3]The knowledge of the adversary regarding the routing algorithms is indirectly defined in [58].

[4]Note that Lu et al. [83] mention active attacks and say that they do not want to discuss them.

[5]Note that Ortolani et al. [55] assume that the adversary can only track a restricted number of traffic streams in parallel if the location of the sink is unknown.

TABLE I – continued from previous page

Type	Solution	Active attacks	Passive attacks	View of network	Exposed information	Issues
In network location anonymization	ACS [79]	NA	Rate-monitoring & time correlation & locating a node & hop-by-hop-trace & identity-analysis attack	Local	NA	V
	APR [87]	Node compromise[6]	Eavesdropping & hop-by-hop-trace	Local[7]	NA	V
	DCARPS [50]	None	Eavesdropping & hop-by-hop-trace[8]	Global	The topology	X
	EAC [130]	Compromise nodes & inject traffic & denial of service	Eavesdropping & traffic analysis & locate a node	Global	The topology	-
	Probabilistic DCARPS [50]	None	Eavesdropping & hop-by-hop-trace[9]	Global	The topology	X
	HIR & RHIR [129]	NA	Node compromise & eavesdropping	Global[10]	NA	V
	MQA [90]	Packet injection	Eavesdropping & hop-by-hop-trace	Global	The topology & the used aggregation protocol	-
	PhID [91]	NA	Eavesdropping & traffic analysis	Local[11]	Topology	-
	SAS & CAS [88]	NA	Node compromise & eavesdropping & limited traffic analysis	Global	The topology	X
Cross-Layer Routing	CLS & DCLS [47]	None	Eavesdropping & traffic analysis & rate monitoring attack & time correlation attack	Multi-local[12]	The topology	V
Separate Path Routing	RP [92]	NA	Eavesdropping & hop-by-hop-trace	Local	The topology	V
	WRS [92]	NA	Eavesdropping & hop-by-hop-trace	Local	The topology	-
	WRS extended [92]	NA	Eavesdropping & traffic analysis & timing analysis	Global	The topology	-
Network Coding	PPSNC [44]	Node compromise	Node compromise & eavesdropping & traffic analysis & size correlation & timing analysis & message content correlation	Global	The topology	-
	SUNC [45]	Node compromise	Node compromise & eavesdropping & traffic analysis & size correlation & time-order correlation	Global	The topology	-
Limiting node detectability	ALbS [93]	NA	Eavesdropping, RSS & AoA	Local	NA	-
	CALP [94]	NA	Locating a node & AoA & hop-by-hop-trace & eavesdropping	Local	The topology	-
	HA [95]	Node compromise	RSS	Local	NA	-
	Hyberloc [51]	Node compromise	Eavesdropping & RSS & AoA	Local	NA	V
	LRTP [96]	NA	Eavesdropping	Local[13]	NA	V
	MCPC [97]	NA	Eavesdropping, RSS & AoA	Multi-local[14]	NA	-
Others	pDCS [98]	Node compromise & mapping attack	Eavesdropping	Global	The topology & the protocols	-
	PPHC [6]	Packet injection & packet alteration & packet drops	Eavesdropping & hop-by-hop-trace	Local	the protocols	-

[6]Note that Jiang et al. [67] assume that the node's hardware make a node compromise nearly impossible and ineffective.
[7]The local view is implicitly defined in [67], [87].
[8]Note that Nezhada et al. [50] assume that the hop-by-hop-trace is only executable for one source at the time.
[9]Note that Nezhada et al. [50] assume that the hop-by-hop-trace is only executable for one source at the time.
[10]The global view is implicitly defined in [129] based on how they discuss SAS and CAS.
[11]The local view of the adversary is implicitly defined in [91].
[12]The multi-local adversary model is implicitly defined in [47] as the authors refer to attackers.
[13]Note that the adversary is outside of the WSN.
[14]The multi-local view is implicitly defined in [97].

V. Concluding Remarks

In this work, we have provided a survey of the literature in source location privacy (SLP) for wireless sensor networks (WSNs). We first introduced the SLP problem based on the panda hunter game. Then, we discussed some of the works that have a high influence on the state of the art today, together with the concepts that they introduced. These concepts included anonymity, unobservability, safety period, capture likelihood, unlinkability, contextual priavcy, identity privacy, location privacy, timing privacy, and route privacy. Next, we included a classification of the adversary based on its behaviour, view of the network, and the information exposed by the network to the adversary. Then, we discussed, explained and grouped over 60 solutions that we found in the literature that provide SLP. We grouped these solutions in eleven different categories, based on the core techniques of the solutions and we augmented them with references to works that indirectly provide SLP, as it is not the core focus of these works to provide SLP. Augmenting the solutions with these works should give the reader a solid starting point with a good coverage within the literature. Lastly, we presented an overview of the solutions, against which type of adversary they should defend, and whether the solution is actually able to properly defend against that type of adversary.

In the literature, there are surveys that cover a part of what we cover, such as the works of Rios et al. [21], Li et al. [12], and Aivaloglou et al. [22]. Yet, none of these works covers the broad set of studies in the state of the art regarding SLP in WSNs as we do. We deliver an updated survey that can serve as a good starting point for researchers and professionals to see which solutions have been designed to provide SLP in WSNs and whether they are effective in doing so.

Based on our survey, we have the following concluding remarks. First of all, we see that many of the current random walk based solutions only work against a local adversary. This automatically means that the technique is not good enough against a global adversary or against a very powerful multi-local adversary. Solutions that use techniques such as cross-layer routing, cyclic entrapment, delay, and limiting the detectability of a node have the same issue: they only defend against a local adversary. Secondly, we observed that most of the dummy data source solutions that try to obfuscate real traffic often consume a lot more energy than solutions that try to emulate the presence of a subject. Solutions, such as UHT and SoSi require nodes to consume less power than solutions that follow the idea of Constrate and FitProbRate. The main reason behind this is, that solutions like UHT and SoSi do not require nodes to continuously send dummy packets. Of course, implementing proxy filters, as we see in OFS and TFS, can help, but placing these proxies is a challenging task. Another observation is that anonimization techniques alone are not enough to prevent traffic analysis. Hiding an identity of a node alone will not provide SLP by itself. Additional measures, such as delay, dummy traffic, re-encryption, and other means have to be used to counter traffic analysis.

We see that there are also a few open problems that should be mentioned:

- **More powerful adversaries.** The current literature often focuses on a local adversary or a global adversary that cannot do a full trace of all the traffic. One challenge that is still open for research is to provide solutions against more powerful adversaries that have a global view, have enough memory and computing power to trace every stream of messages, and analyse the timing and distributions of every node.
- **Dynamic environments.** One problem that is not fully covered yet in detail is making solutions more adaptive towards the environment. Solutions, such as FitProbRate and UHT are currently able to adapt towards a certain event distribution, given the nature of that distribution. Ortolani et al. [55] already identified the lack of following other statistical distributions and recommended it for future work, but we would like to see this future work described in detail and implemented in various solutions that have been described in this survey.
- **Real boundaries of proposed solutions.** As one can see in Table I, we found a set of issues with older solutions. However, that does not mean that newer solutions do not have SLP compromising issues. We would like to see these solutions tested and investigated in order to find their weaknesses.
- **Missing a framework.** The authors of the presented papers take very different approaches in proving the correctness of their solution. Some authors use simulation results, while others use a statistical approach or a formal set of definitions. We would like to see the works described here to be compared based on similar formal methods or simulation results for an in-depth comparison, which might help researchers come up with new solutions.
- **Powerful nodes, mobile ad-hoc networks, and vehicular ad-hoc networks.** We see that the computational power of power efficient microchips increases over time, which should give way to more powerful sensor nodes. The more powerful nodes make way for using computational heavier solutions, such as those currently used in mobile ad-hoc networks (MANET) and vehicle ad-hoc networks (VANET). Therefore, it would be very interesting to see which of the solutions currently used in MANETs and VANETs are actually applicable within a WSN given the more powerful nodes.
- **New solutions based on new technologies.** We have seen some solutions based on trusted computing and on network coding. New technologies like these still look promising for new solutions to provide SLP.

Finally, we have one recommendation for future work to extend our survey. Our survey did not cover all the works that provide SLP indirectly, as we did not cover all the currently existing literature. It might be interesting to see the routing algorithms that provide SLP indirectly, and create an overview of these works.

Acknowledgment

The authors would like to thank the anonymous reviewers who provided very useful feedback on this work.

Mauro Conti is supported by a Marie Curie Fellowship funded by the European Commission for the PRISM-CODE project (Privacy and Security for Mobile Cooperative Devices) under the agreement n. PCIG11-GA-2012-321980.

REFERENCES

[1] D. Noh and J. Hur, "Using a dynamic backbone for efficient data delivery in solar-powered wsns," *J. Network and Computer Applications*, vol. 35, no. 4, pp. 1277 – 1284, 7 2012.

[2] Z. Eu, H. Tan, and W. Seah, "Design and performance analysis of mac schemes for wireless sensor networks powered by ambient energy harvesting," *Ad Hoc Networks*, vol. 9, no. 3, pp. 300–323, 5 2011.

[3] Y. Li and J. Ren, "Providing source-location privacy in wireless sensor networks," in *Proc. International Conference on Wireless Algorithms, Systems and Applications*, ser. WASA 2009, NSFC. Berlin, Heidelberg, Germany: Springer-Verlag, 8 2009, pp. 338–347.

[4] E. Ekici, S. Vural, J. McNair, and D. Al-Abri, "Secure probabilistic location verification in randomly deployed wireless sensor networks," *Ad Hoc Networks*, vol. 6, no. 2, pp. 195–209, 4 2008.

[5] A. Cardenas, T. Roosta, and S. S.S., "Rethinking security properties, threat models, and the design space in sensor networks: A case study in scada systems," *Ad Hoc Networks*, vol. 7, no. 8, pp. 1434–1447, 11 2009.

[6] X. Mingjun, H. Liusheng, X. Hongli, W. Yang, and P. Zegen, "Privacy preserving hop-distance computation in wireless sensor networks," *Chinese J. Electronics*, vol. 19, no. 1, pp. 191–194, 1 2010.

[7] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 3 2002.

[8] A. Wadaa, S. Olariu, L. Wilson, M. Eltoweissy, and K. Jones, "On providing anonymity in wireless sensor networks," in *Parallel and Distributed Systems, 2004. Proceedings. Tenth International Conference on*, ser. ICPADS 2004, IEEE. Piscataway, USA: IEEE, 7 2004, pp. 411–418.

[9] H. Bahsi and A. Levi, "Energy efficient privacy preserved data gathering in wireless sensor networks having multiple sinks," in *Computer Science and its Applications, 2009. 2nd International Conference on*, ser. CSA '09, IEEE. Piscataway, NJ, USA: IEEE, 12 2009, pp. 1–8.

[10] P. Kamat, W. Xu, W. Trappe, and Y. Zhang, "Temporal privacy in wireless sensor networks," in *27th International Conference on Distributed Computing Systems*, ser. ICDCS 2007, IEEE. Piscataway, USA: IEEE, 6 2007, pp. 23–23.

[11] B. Sheng and Q. Li, "Verifiable privacy-preserving range query in two-tiered sensor networks," in *The 27th Conference on Computer Communications*, ser. INFOCOM 2008, IEEE. Piscataway, USA: IEEE, 5 2008, pp. 46–50.

[12] N. Li, N. Zhang, S. Das, and B. Thuraisingham, "Privacy preservation in wireless sensor networks: A state-of-the-art survey," *Ad Hoc Networks*, vol. 7, no. 8, pp. 1501–1514, 11 2009.

[13] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source-location privacy in sensor network routing," in *Proc. 25th IEEE International Conference on Distributed Computing Systems*, ser. ICDCS 2005, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 06 2005, pp. 599–608.

[14] C. Ozturk, Y. Zhang, and W. Trappe, "Source-location privacy in energy-constrained sensor network routing," in *Proc. 2nd ACM workshop on Security of Ad hoc and Sensor Networks*, ser. SASN '04, ACM. New York, NY, USA: ACM, 10 2004, pp. 88–93.

[15] K. Mehta, D. Liu, and M. Wright, "Location privacy in sensor networks against a global eavesdropper," in *IEEE International Conference on Network Protocols*, ser. ICNP 2007, IEEE. Piscataway, USA: IEEE, 10 2007, pp. 314–323.

[16] ——, "Protecting location privacy in sensor networks against a global eavesdropper," *IEEE Trans. Mobile Computing*, vol. 11, no. 2, pp. 320–336, 2 2012.

[17] S. Pai, M. Meingast, T. Roosta, S. Bermudez, S. Wicker, D. Mulligan, and S. Sastry, "Transactional confidentiality in sensor networks," *IEEE Security and Privacy*, vol. 6, no. 4, pp. 28–35, 7 2008.

[18] C. Chow, M. Mokbel, and T. He, "A privacy-preserving location monitoring system for wireless sensor networks," *IEEE Trans. Mobile Computing*, vol. 10, no. 1, pp. 94–107, 1 2011.

[19] I. Shaikh, H. Jameel, B. dAuriol, H. Lee, S. Lee, and Y.-J. Song, "Achieving network level privacy in wireless sensor networks," *Sensors*, vol. 10, no. 3, pp. 1447–1472, 8 2010.

[20] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source-location privacy in sensor network routing," in *Distributed Computing Systems, 2005. Proceedings. 25th IEEE International Conference on*, ser. ICDCS 2005, IEEE. Piscataway, USA: IEEE, 6 2005, pp. 599–608.

[21] R. Rios and J. Lopez, "Source location privacy considerations in wireless sensor networks," in *4th International Symposium of Ubiquitous Computing and Ambient Intelligence*, ser. UCAmI'10. CEDI, 9 2010, pp. 29–38.

[22] J. Z. Javier Lopez, Ed., *Wireless Sensor Networks*, ser. Cryptology & Information Security Series. Amsterdam, The Netherands: IOS Press, 2008, ch. Privacy protection mechanisms for sensor networks, pp. 223–250.

[23] X. Chen, K. Makki, K. Yen, and N. Pissinou, "Sensor network security: a survey," *IEEE Commun. Surveys & Tutorials*, vol. 11, no. 2, pp. 52–73, 4–6 2009.

[24] P. Spachos, L. Song, and D. Hatzinakos, "Opportunistic routing for enhanced source-location privacy in wireless sensor networks," in *25th Biennial Symposium on Communications*, ser. QBSC 2010, IEEE. Stoughton, WI, USA: The Printing House, Inc., 5 2010, pp. 315–318.

[25] W. Wei-ping, C. Liang, and W. Jian-xin, "A source-location privacy protocol in wsn based on locational angle," in *Communications, 2008. IEEE International Conference on*, ser. ICC'08, IEEE. Piscataway, USA: IEEE, 5 2008, pp. 1630–1634.

[26] L. Zhang, "A self-adjusting directed random walk approach for enhancing source-location privacy in sensor network routing," in *Proc. 2006 international conference on Wireless communications and mobile computing*, ser. IWCMC '06, ACM. New York, NY, USA: ACM, 7 2006, pp. 33–38.

[27] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao, "Towards event source unobservability with minimum network traffic in sensor networks," in *Proc. first ACM conference on Wireless network security*, ser. WiSec '08, ACM. New York, NY, USA: ACM, 4 2008, pp. 77–88.

[28] H. Chen and W. Lou, "From nowhere to somewhere: protecting end-to-end location privacy in wireless sensor networks," in *Performance Computing and Communications Conference, IEEE 29th International*, ser. IPCCC 2010. Piscataway, USA: IEEE, 12 2010, pp. 1–8.

[29] A. Jhumka, M. Leeke, and S. Shrestha, "On the use of fake sources for source location privacy: trade-offs between energy and privacy," *The Computer Journal*, vol. 54, no. 6, pp. 860–874, 2 2011.

[30] J. Yao and G. Wen, "Preserving source-location privacy in energy-constrained wireless sensor networks," in *Distributed Computing Systems Workshops, 2008. 28th International Conference on*, ser. ICDCS'08, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 6 2008, pp. 412–416.

[31] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 2 1981.

[32] L. Von Ahn, A. Bortz, and N. Hopper, "K-anonymous message transmission," in *Proc. 10th ACM conference on Computer and Communications Security*, ser. CCS '03, ACM. New York, NY, USA: ACM, 10 2003, pp. 122–130.

[33] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *J. cryptology*, vol. 1, no. 1, pp. 65–75, 3 1988.

[34] A. Pfitzmann and M. Waidner, "Networks without user observability," *Computers & Security*, vol. 6, no. 2, pp. 158–166, 4 1987.

[35] A. Pfitzmann, B. Pfitzmann, and M. Waidner, "Isdn-mixes: Untraceable communication with very small bandwidth overhead," in *Kommunikation in Verteilten Systemen, Grundlagen, Anwendungen, Betrieb, GI/ITG-Fachtagung*. London, UK, UK: Springer-Verlag, 2 1991, pp. 451–463.

[36] D. Kesdogan, P. Reichl, and K. Junghärtchen, "Distributed temporary pseudonyms: A new approach for protecting location information in mobile communication networks," in *Proc. 5th European Symposium on Research in Computer Security*, ser. ESORICS '98. London, UK, UK: Springer-Verlag, 10 1998, pp. 295–312.

[37] M. Reed, P. Syverson, and D. Goldschlag, "Proxies for anonymous routing," in *Proc. 12th Annual Computer Security Applications Conference*, ser. ACSAC '96, IEEE. Washington, DC, USA: IEEE Computer Society, 12 1996, pp. 95–104.

[38] M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Information and System Security (TISSEC)*, vol. 1, no. 1, pp. 66–92, 11 1998.

[39] D. Kesdogan, J. Egner, and R. Bschkes, "Stop- and- go-mixes providing probabilistic anonymity in an open system," in *Information Hiding*, ser. Lecture Notes in Computer Science. Berlin / Heidelberg, Germany: Springer, 1 1998, vol. 1525, pp. 83–98.

[40] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymitya proposal for terminology," in *Designing privacy enhancing technologies*, Springer. New York, NY, USA: Springer-Verlag New York, Inc., 7 2001, pp. 1–9.

[41] A. Pfitzmann and M. Hansen, "Anonymity, unlinkability, unobservability, pseudonymity, and identity management-a consolidated proposal for terminology," 8 2005.

[42] A. Serjantov and G. Danezis, "Towards an information theoretic metric for anonymity," in *Proc. 2nd international conference on Privacy enhancing technologies*, ser. PET'02, Springer-Verlag. Berlin, Heidelberg, Germany: Springer-Verlag, 4 2002, pp. 41–53.

[43] C. Shannon, "Prediction and entropy of printed english," *Bell System Technical Journal*, vol. 30, no. 1, pp. 50–64, 9 1951.

[44] Y. Fan, Y. Jiang, H. Zhu, and X. Shen, "An efficient privacy-preserving scheme against traffic analysis attacks in network coding," in *The 28th Conference on Computer Communications*, ser. INFOCOM 2009, IEEE. Piscataway, USA: IEEE, 4 2009, pp. 2213–2221.

[45] Y. Fan, J. Chen, X. Lin, and X. Shen, "Preventing traffic explosion and achieving source unobservability in multi-hop wireless networks using network coding," in *2010 IEEE Global Telecommunications Conference*, ser. GLOBECOM 2010, IEEE. Piscataway, USA: IEEE communications society, 12 2010, pp. 1–5.

[46] A. Abbasi, A. Khonsari, and M. Talebi, "Source location anonymity for sensor networks," in *Consumer Communications and Networking Conference, 2009. 6th IEEE*, ser. CCNC 2009, IEEE. Piscataway, USA: IEEE, 1 2009, pp. 1–5.

[47] M. Shao, W. Hu, S. Zhu, G. Cao, S. Krishnamurthy, and T. La Porta, "Cross-layer enhanced source location privacy in sensor networks," in *Proc. 6th Annual IEEE communications society conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON'09, IEEE. Piscataway, NJ, USA: IEEE Press, 6 2009, pp. 324–332.

[48] J. Song, V. Wong, and V. Leung, "Wireless location privacy protection in vehicular ad-hoc networks," *Mobile Networks and Applications*, vol. 15, no. 1, pp. 160–171, 5 2010.

[49] P. Venkitasubramaniam and L. Tong, "Anonymous networking with minimum latency in multihop networks," in *IEEE Symposium on Security and Privacy*, ser. ISSP 2008, IEEE. Piscataway, USA: IEEE, 5 2008, pp. 18–32.

[50] A. Nezhada and A. Dimitris Makrakis, "Location privacy and anonymity preserving routing for wireless sensor networks," *Computer Networks*, vol. 52, no. 18, pp. 3433–3452, 12 2008.

[51] R. El-Badry, A. Sultan, and M. Youssef, "Hyberloc: providing physical layer location privacy in hybrid sensor networks," in *IEEE International Conference on Communications*, ser. ICC 2010, IEEE. Piscataway, USA: IEEE, 6 2010, pp. 1–5.

[52] R. Doomun, T. Hayajneh, P. Krishnamurthy, and D. Tipper, "Secloud: Source and destination seclusion using clouds for wireless ad hoc networks," in *Computers and Communications, 2009. IEEE Symposium on*, ser. ISCC 2009, IEEE. Piscataway, USA: IEEE, 7 2009, pp. 361–367.

[53] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Proc. 3rd international conference on Privacy Enhancing Technologies*, Springer. Berlin / Heidelberg, Germany: Springer, 3 2003, pp. 184–188.

[54] B. Hoh and M. Gruteser, "Protecting location privacy through path confusion," in *Security and Privacy for Emerging Areas in Communications Networks, 2005. First International Conference on*, ser. SecureComm 2005, IEEE. Washington, DC, USA: IEEE Computer Society, 9 2005, pp. 194–205.

[55] S. Ortolani, M. Conti, B. Crispo, and R. Di Pietro, "Events privacy in wsns: a new model and its application," in *Proc. 12th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, ser. WoWMoM 2011, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 6 2011, pp. 1–9.

[56] ——, "Event handoff unobservability in wsn," in *Proceedings of the 2010 IFIP WG 11.4 international conference on Open research problems in network security*, ser. iNetSec'10, Springer. Berlin, Heidelberg: Springer-Verlag, 1 2011, pp. 20–28.

[57] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards statistically strong source anonymity for sensor networks," in *The 27th Conference on Computer Communications*, ser. INFOCOM 2008, IEEE. Piscataway, USA: IEEE, 4 2008, pp. 51–55.

[58] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran, "Towards a statistical framework for source anonymity in sensor networks," *IEEE Trans. Mobile Computing*, vol. 10, no. 12, pp. 1–1, 12 2011.

[59] B. Zhao, X. Su, Y. Sun, J. Su, and S. Li, "A distributed query protocol for continuous privacy preserving in wireless sensor networks," in *Computer and Information Technology. IEEE 10th International Conference on*, ser. CIT 2010, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 6–7 2010, pp. 2837–2842.

[60] M. Mahmoud and X. Shen, "A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 10, pp. 1805–1818, 10 2012.

[61] L. Kang, "Protecting location privacy in large-scale wireless sensor networks," in *Communications, 2009. IEEE International Conference on*, ser. ICC'09, IEEE. Red Hook, NY, USA: IEEE, 6 2009, pp. 1–6.

[62] K. Bicakci, H. Gultekin, B. Tavli, and I. Bagci, "Maximizing lifetime of event-unobservable wireless sensor networks," *Computer Standards & Interfaces*, vol. 33, no. 4, pp. 401–410, 5 2011.

[63] P. Kamat, W. Xu, W. Trappe, and Y. Zhang, "Temporal privacy in wireless sensor networks: Theory and practice," *ACM Trans. Sensor Networks (TOSN)*, vol. 5, no. 4, p. art.no.:28, 11 2009.

[64] T. Kavitha and D. Sridharan, "Security vulnerabilities in wireless sensor networks: a survey," *J. information Assurance and Security*, vol. 5, no. 1, pp. 31–44, 1 2010.

[65] M. Conti, R. Di Pietro, and L. Mancini, "Ecce: Enhanced cooperative channel establishment for secure pair-wise communication in wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 49–62, 1 2007.

[66] S. Pai, M. Meingast, T. Roosta, S. Bermudez, S. Wicker, D. Mulligan, and S. Sastry, "Confidentiality in sensor networks: Transactional information," *IEEE Security Privacy Mag.*, vol. 6, no. 4, pp. 28–35, 7 2008.

[67] J. Jiang, J. Sheu, C. Tu, and J. Wu, "An anonymous path routing (apr) protocol for wireless sensor networks," *J. information science and engineering*, vol. 27, no. 2, pp. 657–680, 4 2011.

[68] G. Suarez-Tangil, E. Palomar, B. Ramos, and A. Ribagorda, "An experimental comparison of source location privacy methods for power optimization in wsns," in *Proc. 3rd WSEAS international conference on Advances in sensors, signals and materials*, ser. SENSIG 2010 / MATERIALS 2010, World Scientific and Engineering Academy and Society (WSEAS). Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2 2010, pp. 79–84.

[69] A. Majeed, K. Liu, and N. Abu-Ghazaleh, "Tarp: Timing analysis resilient protocol for wireless sensor networks," in *Proc. 2009 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, ser. WIMOB '09, IEEE. Washington, DC, USA: IEEE Computer Society, 10 2009, pp. 85–90.

[70] X. Hong, P. Wang, J. Kong, Q. Zheng *et al.*, "Effective probabilistic approach protecting sensor traffic," in *Military Communications Conference, 2005. IEEE*, ser. MILCOM 2005, IEEE. Piscataway, NJ, USA: IEEE, 10 2005, pp. 169–175.

[71] J. Cote, B. Wang, W. Zeng, and Z. Shi, "Capability and fidelity of mote-class wireless sniffers," in *2010 IEEE Global Telecommunications Conference*, ser. GLOBECOM 2010, IEEE. Piscataway, NJ, USA: IEEE, 12 2010, pp. 1–6.

[72] Y. Ouyang, Z. Le, D. Liu, J. Ford, and F. Makedon, "Source location privacy against laptop-class attacks in sensor networks," in *SECURECOM: Proc. 4th international conference on Security and privacy in communication netowrks*, ser. SecureComm '08, ACM. New York, NY, USA: ACM, 9 2008, pp. 5:1–5:10.

[73] N. Li, M. Raj, D. Liu, M. Wright, and S. Das, "Using data mules to preserve source location privacy in wireless sensor networks," in *13th International Conference on Distributed Computing Computing and Networking*, ser. ICDCN 2012, The Hong Kong Polytechnic University. Berlin/Heidelberg, Germany: Springer, 1 2012, pp. 309–324.

[74] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran, "Statistical framework for source anonymity in sensor networks," in *2010 IEEE Global Telecommunications Conference*, ser. GLOBECOM 2010, IEEE. Piscataway, USA: IEEE communications society, 12 2010, pp. 1–6.

[75] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proc. 1st ACM international workshop on Wireless sensor networks and applications*, ser. WSNA '02, ACM. New York, NY, USA: ACM, 9 2002, pp. 22–31.

[76] Y. Li, L. Lightfoot, and J. Ren, "Routing-based source-location privacy protection in wireless sensor networks," in *Electro/Information Technology, 2009. IEEE International Conference on*, ser. EIT '09, IEEE. Piscataway, NJ, USA: IEEE, 6 2009, pp. 29–34.

[77] Y. Xi, L. Schwiebert, and W. Shi, "Preserving source location privacy in monitoring-based wireless sensor networks," in *20th International Parallel and Distributed Processing Symposium*, ser. IPDPS 2006, IEEE. Piscataway, USA: IEEE, 4 2006, p. 8 pp.

[78] S. Armenia, G. Morabito, and S. Palazzo, "Analysis of location privacy/energy efficiency tradeoffs in wireless sensor networks," in *Proceedings of the 6th international IFIP-TC6 conference on Ad Hoc and sensor networks, wireless networks, next generation internet*, ser.

NETWORKING'07, Springer-Verlag. Berlin, Heidelberg: Springer-Verlag, 5 2007, pp. 215–226.

[79] X. Luo, X. Ji, and M. Park, "Location privacy against traffic analysis attacks in wireless sensor networks," in *Information Science and Applications International Conference on*, ser. ICISA 2010, IEEE. Piscataway, NJ, USA: IEEE, 4 2010, pp. 1–6.

[80] L. Lightfoot, Y. Li, and J. Ren, "Preserving source-location privacy in wireless sensor network using star routing," in *2010 IEEE Global Telecommunications Conference*, ser. GLOBECOM 2010, IEEE. Piscataway, USA: IEEE communications society, 12 2010, pp. 1–5.

[81] W. Yang and W. Zhu, "Protecting source location privacy in wireless sensor networks with data aggregation," in *Proc. 7th international conference on Ubiquitous intelligence and computing*, ser. UIC'10. Berlin, Heidelberg: Springer-Verlag, 10 2010, pp. 252–266.

[82] S. Kokalj-Filipovic, F. Le Fessant, and P. Spasojevic, "The quality of source location protection in globally attacked sensor networks," in *IEEE International Conference on Pervasive Computing and Communications Workshops*, ser. PERCOM 2011, IEEE. Piscataway, USA: IEEE, 3 2011, pp. 44–49.

[83] R. Lu, X. Lin, H. Zhu, and X. Shen, "Tesp2: Timed efficient source privacy preservation scheme for wireless sensor networks," in *IEEE International Conference on Communications*, ser. ICC 2010, IEEE. Piscataway, USA: IEEE, 5 2010, pp. 1–6.

[84] Y. Yang, J. Zhou, R. Deng, and F. Bao, "Better security enforcement in trusted computing enabled heterogeneous wireless sensor networks," *Security and Communication Networks*, vol. 4, no. 1, pp. 11–22, 1 2011.

[85] Y. Ouyang, Z. Le, G. Chen, J. Ford, and F. Makedon, "Entrapping adversaries for source protection in sensor networks," in *Proc. 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, ser. WOWMOM '06, IEEE Computer Society. Washington, DC, USA: IEEE Computer Society, 6 2006, pp. 23–34.

[86] L. Kazatzopoulos, C. Delakouridis, G. Marias, and P. Georgiadis, "Ihide: Hiding sources of information in wsns," in *Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006. Second International Workshop on*, ser. SECPERU '06, IEEE. Washington, DC, USA: IEEE Computer Society, 6 2006, pp. 41–48.

[87] J. Sheu, J. Jiang, and C. Tu, "Anonymous path routing in wireless sensor networks," in *Communications, 2008. IEEE International Conference on*, ser. ICC'08, IEEE. Piscataway, USA: IEEE, 5 2008, pp. 2728–2734.

[88] S. Misra and G. Xue, "Efficient anonymity schemes for clustered wireless sensor networks," *International J. Sensor Networks*, vol. 1, no. 1, pp. 50–63, 1 2006.

[89] L. Grieco, G. Boggia, S. Sicari, and P. Colombo, "Secure wireless multimedia sensor networks: a survey," in *Mobile Ubiquitous Computing, Systems, Services and Technologies, 2009. Third International Conference on*, ser. UBICOMM'09, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 10 2009, pp. 194–201.

[90] R. Di Pietro and A. Viejo, "Location privacy and resilience in wireless sensor networks querying," *Computer Communications*, vol. 34, no. 3, pp. 515–523, 3 2011.

[91] J. Park, Y. Jung, H. Ko, J. Kim, and M. Jun, "A privacy technique for providing anonymity to sensor nodes in a sensor network," in *Ubiquitous Computing and Multimedia Applications, Second International Conference*, ser. UCMA 2011, SERSC. Berlin Heidelberg, Germany: Springer, 4 2011, pp. 327–335.

[92] H. Wang, B. Sheng, and Q. Li, "Privacy-aware routing in sensor networks," *Computer Networks*, vol. 53, no. 9, pp. 1512–1529, 6 2009.

[93] N. Dutta, A. Saxena, and S. Chellappan, "Defending wireless sensor networks against adversarial localization," in *Proc. 2010 Eleventh International Conference on Mobile Data Management*, ser. MDM 2010, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 336–341.

[94] R. Rios and J. Lopez, "Exploiting context-awareness to enhance source-location privacy in wireless sensor networks," *The Computer Journal*, vol. 54, no. 10, pp. 1603–1615, 6 2011.

[95] R. El-Badry, M. Youssef, and A. Sultan, "Hidden anchor: Providing physical layer location privacy in hybrid wireless sensor networks," in *Proc. 3rd international conference on New technologies, mobility and security*, ser. NTMS'09. Piscataway, NJ, USA: IEEE Press, 5 2009, pp. 254–258.

[96] B. Tavli, M. Ozciloglu, and K. Bicakci, "Mitigation of compromising privacy by transmission range control in wireless sensor networks," *IEEE Commun. Lett.*, vol. 14, no. 12, pp. 1104–1106, 10 2010.

[97] S. Oh and M. Gruteser, "Multi-node coordinated jamming for location privacy protection," in *Military Communications Conference, 2011*, ser. MILCOM 2011, IEEE. Piscataway, NJ, USA: IEEE, 11 2011, pp. 1243–1249.

[98] M. Shao, S. Zhu, and W. Zhang, "pdcs: Security and privacy support for data-centric sensor networks," in *26th IEEE International Conference on Computer Communications*, ser. INFOCOM 2007, IEEE. Piscataway, USA: IEEE, 5 2007, pp. 1298–1306.

[99] S. Servetto and G. Barrenechea, "Constrained random walks on random graphs: routing algorithms for large scale wireless sensor networks," in *Proc. 1st ACM international workshop on Wireless sensor networks and applications*, ser. WSNA '02. New York, NY, USA: ACM, 2002, pp. 12–21.

[100] C. Ozturk, Y. Zhang, W. Trappe, and M. Ott, "Source-location privacy for networks of energy-constrained sensors," in *Software Technologies for Future Embedded and Ubiquitous Systems, 2004. Proceedings. Second IEEE Workshop on*, ser. WSTFEUS'04, IEEE. Piscataway, USA: IEEE, 5 2004, pp. 68–72.

[101] J. Deng, R. Han, and S. Mishra, "Countermeasures against traffic analysis attacks in wireless sensor networks," in *Proc. First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, ser. SECURECOMM '05, IEEE. Washington, DC, USA: IEEE Computer Society, 9 2005, pp. 113–126.

[102] L. Kazatzopoulos, K. Delakouridis, and G. Marias, "A privacy-aware overlay routing scheme in wsns," *Security and Communication Networks*, vol. 4, no. 7, pp. 729–743, 2 2011.

[103] P. Spachos, L. Song, F. Bui, and D. Hatzinakos, "Improving source-location privacy through opportunistic routing in wireless sensor networks," in *Computers and Communications. IEEE Symposium on*, ser. ISCC 2011, IEEE. Piscataway, USA: IEEE, 6 2011, pp. 815–820.

[104] H. Wang and T. Hsiang, "Defending traffic analysis with communication cycles in wireless sensor networks," in *Proc. 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, ser. ISPAN '09, IEEE. Washington, DC, USA: IEEE Computer Society, 12 2009, pp. 166–171.

[105] Y. Li and J. Ren, "Mixing ring-based source-location privacy in wireless sensor networks," in *Computer Communications and Networks, 2009. Proceedings of 18th Internatonal Conference on*, ser. ICCCN 2009, IEEE. Washington, DC, USA: IEEE Computer Society, 8 2009, pp. 1–6.

[106] ——, "Preserving source-location privacy in wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. 6th Annual IEEE Communications Society Conference on*, ser. SECON'09, IEEE. Piscataway, NJ, USA: IEEE Press, 6 2009, pp. 1–9.

[107] C. Hsu, H. Liu, and W. Seah, "Opportunistic routing-a review and the challenges ahead," *Computer Networks*, vol. 55, no. 15, pp. 3592–3603, 6 2011.

[108] R. Gray, "Agent tcl: A flexible and secure mobile-agent system," in *Proc. 4th conference on USENIX Tcl/Tk Workshop, 1996-Volume 4*, ser. TCLTK '96, USENIX Association. Berkeley, CA, USA: USENIX Association, 6 1996, p. pp 16.

[109] E. Ngai and I. Rodhe, "On providing location privacy for mobile sinks in wireless sensor networks," in *Proc. 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '09, ACM. New York, NY, USA: ACM, 10 2009, pp. 116–123.

[110] E. Ngai, "On providing sink anonymity for sensor networks," in *Proc. 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, ser. IWCMC '09, ACM. New York, NY, USA: ACM, 6 2009, pp. 269–273.

[111] A. Ahmed and N. Fisal, "Secure real-time routing protocol with load distribution in wireless sensor networks," *Security and Communication Networks*, vol. 4, no. 8, pp. 839–869, 5 2011.

[112] T. Dimitriou and A. Sabouri, "Privacy preservation schemes for querying wireless sensor networks," in *Pervasive Computing and Communications Workshops, IEEE International Conference on*, ser. PERCOM Workshops, 2011. Piscataway, USA: IEEE, 3 2011, pp. 178–183.

[113] Y. Zeng, J. Cao, S. Zhang, S. Guo, and L. Xie, "Random-walk based approach to detect clone attacks in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 5, pp. 677–691, 5 2010.

[114] B. Karp and H. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," in *Proc. 6th annual international conference on Mobile computing and networking*, ser. MobiCom '00, ACM. New York, NY, USA: ACM, 8 2000, pp. 243–254.

[115] H. Lim, Y. Teo, P. Mukherjee, W. Wong, S. See *et al.*, "Sensor grid: Integration of wireless sensor networks and the grid," in *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*, ser. LCN 2005, IEEE. Piscataway, USA: IEEE computer society, 11 2005, pp. 91–99.

[116] M. Zorzi and R. Rao, "Geographic random forwarding (geraf) for ad hoc and sensor networks: multihop performance," *IEEE Trans. Mobile Computing*, vol. 2, no. 4, pp. 337–348, 10-12 2003.

[117] ——, "Geographic random forwarding (geraf) for ad hoc and sensor networks: energy and latency performance," *IEEE Trans. Mobile Computing*, vol. 2, no. 4, pp. 349–365, 10-12 2003.

[118] A. Wood, L. Fang, J. Stankovic, and T. He, "Sigf: a family of configurable, secure routing protocols for wireless sensor networks," in *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, ser. SASN '06, ACM. New York, NY, USA: ACM, 10/11 2006, pp. 35–48.

[119] R. Binder, N. Abramson, F. Kuo, A. Okinaka, and D. Wax, "Aloha packet broadcasting: a retrospect," in *Proc. May 19-22, 1975, national computer conference and exposition*, ser. AFIPS '75, ACM. New York, NY, USA: ACM, 1975, pp. 203–215.

[120] Y. Yang, S. Zhu, G. Cao, and T. LaPorta, "An active global attack model for sensor source location privacy: Analysis and countermeasures," in *Security and Privacy in Communication Networks*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Berlin/Heidelberg: Springer, 2009, vol. 19, pp. 373–393.

[121] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran, "On source anonymity in wireless sensor networks," in *Proc. 40th IEEE/IFIP International Conference on Dependable Systems and Networks*, ser. DSN 2010, IEEE/IFIP. Piscataway, USA: IEEE, 6 2010, pp. 1–6.

[122] TCG, "The tcg mobile trusted module specification - version 1, revision 7.02," MPWG, TCG, Tech. Rep., 4 2010. [Online]. Available: http://www.trustedcomputinggroup.org/files/static_page_files/3D843B67-1A4B-B294-D0B5B407C36F4B1D/Revision_7.02-_29April2010-tcg-mobile-trusted-module-1.0.pdf

[123] ——, "The tcg mobile reference architecture- specification version 1.0," MPWG, TCG, Tech. Rep., 6 2007. [Online]. Available: http://hackipedia.org/Digital%20Rights%20Management/Trusted%20Computing/hardware/pdf/Revision_5-tcg-mobile-reference-architecture-1_0.pdf

[124] A. Beimel and S. Dolev, "Buses for anonymous message delivery," *J. Cryptology*, vol. 16, no. 1, pp. 25–39, 12 2003.

[125] X. Wang, X. Li, Z. Wan, and M. Gu, "Clear: A confidential and lifetime-aware routing protocol for wireless sensor network," in *Personal, Indoor and Mobile Radio Communications, 2009 IEEE 20th International Symposium on*, ser. PIMC 2009, IEEE. Piscataway, NJ, USA: IEEE Press, 9 2009, pp. 2265–2269.

[126] Y. Jian, S. Chen, Z. Zhang, and L. Zhang, "Protecting receiver-location privacy in wireless sensor networks," in *26th IEEE International Conference on Computer Communications*, ser. INFOCOM 2007, IEEE. Piscataway, USA: IEEE, 5 2007, pp. 1955–1963.

[127] S. Jiang, N. Vaidya, and W. Zhao, "Routing in packet radio networks to prevent traffic analysis," in *DARPA Information Survivability Conference & Exposition II, 2001. Proceedings*, ser. DISCEX '01, vol. 2, IEEE. Piscataway, NJ, USA: IEEE, 6 2001, pp. 153–158.

[128] K. Bicakci, I. Bagci, and B. Tavli, "Lifetime bounds of wireless sensor networks preserving perfect sink unobservability," *IEEE Commun. Lett.*, vol. 15, no. 2, pp. 205 –207, february 2011.

[129] Y. Ouyang, Z. Le, Y. Xu, N. Triandopoulos, S. Zhang, J. Ford, and F. Makedon, "Providing anonymity in wireless sensor networks," in *Pervasive Services, IEEE International Conference on*, ser. ICPS 2007, IEEE. Washington, DC, USA: IEEE Computer Society Press, 7 2007, pp. 145–148.

[130] J. Chen, X. Du, and B. Fang, "An efficient anonymous communication protocol for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 12, no. 14, pp. 1302–1312, 10 2011.

[131] Y. Yang, X. Wang, S. Zhu, and G. Cao, "Sdap: A secure hop-by-hop data aggregation protocol for sensor networks," *ACM Trans. Information and System Security (TISSEC)*, vol. 11, no. 4, p. 18, 6 2008.

[132] M. Conti, L. Zhang, S. Roy, R. Di Pietro, S. Jajodia, and L. Mancini, "Privacy-preserving robust data aggregation in wireless sensor networks," *Security and Communication Networks*, vol. 2, no. 2, pp. 195–213, 1 2009.

[133] M. Kaosar and X. Yi, *Wireless Technologies: Concepts, Methodologies, Tools and Applications*. Hershey: IGI Global, 2012, ch. Privacy Preserving Data Gathering in Wireless Sensor Network, pp. 239–253.

[134] M. Gruteser, G. Schelle, A. Jain, R. Han, and D. Grunwald, "Privacy-aware location sensor networks," in *Proc. 9th conference on Hot Topics in Operating Systems-Volume 9*, ser. HotOS IX '03, USENIX Association. Berkeley, CA, USA: USENIX Association, 6 2003, pp. 28–28.

[135] W. Ren, Y. Ren, and H. Zhang, "H2s: A secure and efficient data aggregative retrieval scheme in unattended wireless sensor networks," in *Information Assurance and Security, 2009. Fifth International Conference on*, ser. IAS '09, vol. 2, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 8 2009, pp. 450–453.

[136] T. Feng, C. Wang, W. Zhang, and L. Ruan, "Confidentiality protection for distributed sensor data aggregation," in *The 27th Conference on Computer Communications. IEEE*, ser. INFOCOM 2008, IEEE. Piscataway, USA: IEEE, 4 2008, pp. 56–60.

[137] R. Bista, K. Jo, and J. Chang, "A new approach to secure aggregation of private data in wireless sensor networks," in *Dependable, Autonomic and Secure Computing, 2009. Eighth IEEE International Conference on*, ser. DASC'09, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 12 2009, pp. 394–399.

[138] S. Ozdemir and Y. Xiao, "Integrity protecting hierarchical concealed data aggregation for wireless sensor networks," *Computer Networks*, vol. 55, no. 8, pp. 1735–1746, 7 2011.

[139] L. Zhang, H. Zhang, M. Conti, R. Di Pietro, S. Jajodia, and L. Mancini, "Preserving privacy against external and internal threats in wsn data aggregation," *Telecommunication Systems*, vol. 46, pp. 1–14, 2011.

[140] R. Di Pietro, P. Michiardi, and R. Molva, "Confidentiality and integrity for data aggregation in wsn using peer monitoring," *Security and Communication Networks*, vol. 2, no. 2, pp. 181–194, 4 2009.

[141] X. Lin, R. Lu, and X. Shen, "Mdpa: multidimensional privacy-preserving aggregation scheme for wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 10, no. 6, pp. 843–856, 6 2010.

[142] W. He, H. Nguyen, X. Liuy, K. Nahrstedt, and T. Abdelzaher, "ipda: An integrity-protecting private data aggregation scheme for wireless sensor networks," in *Military Communications Conference, 2008. IEEE*, ser. MILCOM 2008, IEEE. Piscataway, NJ, USA: IEEE, 10 2008, pp. 1–7.

[143] W. Zhang, C. Wang, and T. Feng, "Gpˆ 2s: Generic privacy-preservation solutions for approximate aggregation of sensor data (concise contribution)," in *Pervasive Computing and Communications, 2008. Sixth Annual IEEE International Conference on*, ser. PerCom 2008, IEEE. Piscataway, USA: IEEE, 3 2008, pp. 179–184.

[144] C. Wang, G. Wang, W. Zhang, and T. Feng, "Reconciling privacy preservation and intrusion detection in sensory data aggregation," in *INFOCOM, 2011 Proceedings IEEE*, ser. INFOCOM, 2011, IEEE. Piscataway, USA: IEEE, 4 2011, pp. 336–340.

[145] F. Rahman, E. Hoque, and S. Ahamed, "Preserving privacy in wireless sensor networks using reliable data aggregation," *ACM SIGAPP Applied Computing Review*, vol. 11, no. 3, pp. 52–62, 8 2011.

[146] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *Mobile and Ubiquitous Systems: Networking and Services, 2005. The Second Annual International Conference on*, ser. MobiQuitous 2005, IEEE. Piscataway, USA: IEEE communications society, 7 2005, pp. 109–117.

[147] C. Aldar and C. Castelluccia, "On the privacy of concealed data aggregation," in *Computer Security — ESORICS 2007*, ser. Lecture Notes in Computer Science, J. Biskup and J. Lopez, Eds. Springer Berlin / Heidelberg, 2007, vol. 4734, pp. 390–405.

[148] J. Girao, D. Westhoff, and M. Schneider, "Cda: Concealed data aggregation for reverse multicast traffic in wireless sensor networks," in *Communications, 2005 IEEE International Conference on*, ser. ICC 2005, vol. 5, IEEE. Piscataway, USA: IEEE computer society, 5 2005, pp. 3044–3049.

[149] D. Westhoff, J. Girao, and M. Acharya, "Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation," *IEEE Trans. Mobile Computing*, vol. 5, no. 10, pp. 1417–1431, 9 2006.

[150] A. Chan and C. Castelluccia, "A security framework for privacy-preserving data aggregation in wireless sensor networks," *ACM Trans. Sensor Networks (TOSN)*, vol. 7, no. 4, pp. 29:1–29:45, 2 2011.

[151] S. Peter, D. Westhoff, and C. Castelluccia, "A survey on the encryption of convergecast traffic with in-network processing," *IEE Trans. Dependable Secure Computing*, vol. 7, no. 1, pp. 20–34, 6 2010.

[152] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *26th IEEE International Conference on Computer Communications. IEEE*, ser. INFOCOM 2007, IEEE. Piscataway, USA: IEEE, 5 2007, pp. 2045–2053.

[153] W. He, X. Liu, H. Nguyen, and K. Nahrstedt, "A cluster-based protocol to enforce integrity and preserve privacy in data aggregation," in *Distributed Computing Systems Workshops, 2009. 29th IEEE International Conference on*, ser. ICDCS Workshops' 09, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 6 2009, pp. 14–19.

[154] B. Przydatek, D. Song, and A. Perrig, "Sia: Secure information aggregation in sensor networks," in *Proc. 1st international conference on Embedded networked sensor systems*, ser. SenSys '03, ACM. New York, NY, USA: ACM, 11 2003, pp. 255–265.

[155] L. Van Hoesel, T. Nieberg, J. Wu, and P. Havinga, "Prolonging the lifetime of wireless sensor networks by cross-layer interaction," *IEEE Wireless Commun.*, vol. 11, no. 6, pp. 78–86, 12 2004.

[156] G. Miao, N. Himayat, Y. Li, and A. Swami, "Cross-layer optimization for energy-efficient wireless communications: a survey," *Wireless Communications and Mobile Computing*, vol. 9, no. 4, pp. 529–542, 9 2009.

[157] J. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile, "Ieee 802.15.4: a developing standard for low-power low-cost wireless personal area networks," *IEEE Network*, vol. 15, no. 5, pp. 12–19, 9/10 2001.

[158] X. Phillip, R. Du, K. E. Nygard, and H. Zhang, "Lightweight source anonymity in wireless sensor networks," in *Global Telecommunications Conference, 2011 IEEE*, ser. GLOBECOM 2010, IEEE. Piscataway, NJ, USA: IEEE, 12 2011, pp. 1–5.

[159] S. Dulman, T. Nieberg, and J. Wu, "Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks," in *2003 IEEE Wireless Communications and Networking*, ser. WCNC 2003, IEEE. Los Alamitos, CA, USA: IEEE Computer Society, 3 2003, pp. 1918–1922.

[160] W. Lou, "An efficient n-to-1 multipath routing protocol in wireless sensor networks," in *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, ser. MAHSS 2005, IEEE. Piscataway, NJ, USA: IEEE, 11 2005, pp. 672–680.

[161] P. Chou and Y. Wu, "Network coding for the internet and wireless networks," *IEEE Signal Processing Mag.*, vol. 24, no. 5, pp. 77–85, 9 2007.

[162] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, 7 2000.

[163] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, "Decentralized compression and predistribution via randomized gossiping," in *Proc. 5th international conference on Information processing in sensor networks*, ser. IPSN '06, ACM. New York, NY, USA: ACM, 4 2006, pp. 51–59.

[164] Y. Fan, Y. Jiang, H. Zhu, J. Chen, and X. Shen, "Network coding based privacy preservation against traffic analysis in multi-hop wireless networks," *IEEE Trans. Wireless Commun.*, vol. 10, no. 3, pp. 834–843, 3 2011.

[165] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology — EUROCRYPT 99*, Springer. New York, NY, USA: Springer, 5 1999, pp. 223–238.

[166] A. Hessler, T. Kakumaru, H. Perrey, and D. Westhoff, "Data obfuscation with network coding," *Computer Communications*, vol. 35, no. 1, pp. 48–61, 1 2012.

[167] L. Miao, K. Djouani, A. Kurien, and G. Noel, "Network coding and competitive approach for gradient based routing in wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 6, 1 2012.

[168] E. Ayday, F. Delgosha, and F. Fekri, "Location-aware security services for wireless sensor networks using network coding," in *26th IEEE International Conference on Computer Communications. IEEE*, ser. INFOCOM 2007, IEEE. Piscataway, USA: IEEE, 5 2007, pp. 1226–1234.

[169] L. Correia, D. Macedo, D. Silva, A. Dos Santos, A. Loureiro, and J. Nogueira, "Transmission power control in mac protocols for wireless sensor networks," in *Proc. 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '05, ACM. New York, NY, USA: ACM, 10 2005, pp. 282–289.

[170] R. El-Badry, M. Youssef, and A. Sultan, "Hidden anchor: a lightweight approach for physical layer location privacy," *J. Comp. Sys., Netw., and Comm.*, vol. 2010, pp. 2:1–2:7, 1 2010.

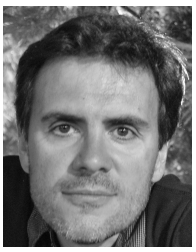
[171] D. Huang, "Traffic analysis-based unlinkability measure for ieee 802.11 b-based communication systems," in *Proc. 5th ACM workshop on Wireless security*, ser. WiSe '06, ACM. New York, NY, USA: ACM, 9 2006, pp. 65–74.

[172] D. Kundur, W. Luh, U. Okorafor, and T. Zourntos, "Security and privacy for distributed multimedia sensor networks," *Proc. IEEE*, vol. 96, no. 1, pp. 112–130, 1 2008.

[173] A. Hassanzadeh, R. Stoleru, and J. Chen, "Efficient flooding in wireless sensor networks secured with neighborhood keys," in *Wireless and Mobile Computing, Networking and Communications, IEEE 7th International Conference on*, ser. WiMob 2011, IEEE. Piscataway, NJ, USA: IEEE, 10 2011, pp. 119–126.

Mauro Conti received the Ph.D. degree from Sapienza University of Rome, Italy, in 2009. In 2008, he was a Visiting Researcher at the Center for Secure Information Systems, George Mason University, Fairfax, VA, USA. After earning his Ph.D. degree, he was a Postdoctoral Researcher at Vrije Universiteit Amsterdam, The Netherlands. From 2011, he is an Assistant Professor at the University of Padua, Italy. In 2012, he was a Visiting Assistant Professor at the University of California, Irvine, CA, USA. His main research interest is in the area of security and privacy. In this area, he has published more than 40 papers in international peer-reviewed journals and conferences. Dr. Conti was a Panelist at ACM CODASPY 2011. He served as program committee member of several conferences, and he is General Chair for SecureComm 2012 and ACM SACMAT 2013. In 2012, he has been awarded by the European Commission with a Marie Curie Fellowship.

Jeroen Willemsen has received his MSc degree in Computer Science at the Vrije Universiteit at Amsterdam in 2012. His research interests include: risk management, architectures, security, wireless sensor networks and mobile security.



Bruno Crispo holds a PhD in Computer Security from the University of Cambridge, UK. He is Associate Professor at the Department of Computer Science and Information Engineering at the University of Trento, Italy. His research interests include networks and distributed systems security, privacy, access control and policy enforcement, applied cryptography and security protocols. He is a senior member of IEEE.