Privacy of Contextual Information in Wireless Sensor Networks

by

Alejandro Adrián Proaño Lozada

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

Doctor of Philosophy

In the Graduate College

The University of Arizona

2 0 1 5

Get the official approval page
from the Graduate College
before your final defense.

Statement by Author

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Signed: _____

Approval by Dissertation Director

This dissertation has been approved on the date shown below:

_____ _____

Dr. Loukas Lazos
Associate Professor

Date

Acknowledgements

Dedication

Table of Contents

TABLE OF CONTENTS—*Continued*

Table of Contents—*Continued*

List of Tables

List of Figures

LIST OF FIGURES—*Continued*

Abstract

Continuous advances in electronics, wireless technologies, manufacturing processes, and software engineering have led to the proliferation of a plethora of mobile devices-mobile phones, tablets, wearables, sensors, smart consumer electronics, etc.- in our everyday lives. The interconnection of these devices into a single web of communication, information, and computation gives rise to a densely meshed wireless ecosystem that transforms the way users interact with their environment. However, ubiquitous interactions with devices that collect data about user activities pose challenging privacy and security problems. Without protection mechanisms, the systems we deploy breach user privacy, often without the users knowledge or consent. The collected information could reveal the user whereabouts, track his motion through space, infer his habits and personal preferences, record user relationships, acquaintances, and contacts, and compromise sensitive information. We investigate the leakage of the so-called contextual information in wireless communications. We focus on event-driven wireless sensor networks (WSNs), whereby wireless transmissions are triggered upon the detection of important events such as the detection of an object of interest, the recording of an abnormal physical parameter, etc. Privacy in event-driven WSNs is particularly important, because traffic patterns can be directly associated to events.

We devise general traffic analysis techniques for extracting contextual information from WSN communications. We further investigate the inference of contextual information when the WSN transmissions are protected by traffic normalization methods, which rely on statistical source anonymity (SSA). To counter traffic analysis, we develop resource-efficient communication and routing methods for reporting events over multi-hop routes, without revealing the event location and occurrence time, as well as the location of the sink. Our work explores the tradeoff between the communication overhead for normalizing traffic and the end-to-end real packet delay for delivering

the event report to the sink. This is achieved by limiting the number of fake transmissions for obfuscating traffic patterns. To do so, we map the problem of selecting fake sources to the problem of finding a minimum connected dominating set (MCDS) that covers the WSN deployment area. We then impose transmission schedules on the fake sources to accelerate the delivery of real event reports. Finally, we propose strong privacy traffic normalization techniques that reduce the number of fake transmissions without relying on the concept of SSA. In the proposed solution, the WSN is partitioned into connected dominating sets (CDSs) that are activated in a round-robin fashion. We show that our methods reduce the communication by several orders of magnitude, while maintaining privacy under strong adversary models.

Privacy of Contextual Information in Wireless Sensor Networks

Alejandro Adrián Proaño Lozada, Ph.D.
The University of Arizona, 2015

Director: Dr. Loukas Lazos

Continuous advances in electronics, wireless technologies, manufacturing processes, and software engineering have led to the proliferation of a plethora of mobile devices-mobile phones, tablets, wearables, sensors, smart consumer electronics, etc.- in our everyday lives. The interconnection of these devices into a single web of communication, information, and computation gives rise to a densely meshed wireless ecosystem that transforms the way users interact with their environment. However, ubiquitous interactions with devices that collect data about user activities pose challenging privacy and security problems. Without protection mechanisms, the systems we deploy breach user privacy, often without the users knowledge or consent. The collected information could reveal the user whereabouts, track his motion through space, infer his habits and personal preferences, record user relationships, acquaintances, and contacts, and compromise sensitive information. We investigate the leakage of the so-called contextual information in wireless communications. We focus on event-driven wireless sensor networks (WSNs), whereby wireless transmissions are triggered upon the detection of important events such as the detection of an object of interest, the recording of an abnormal physical parameter, etc. Privacy in event-driven WSNs is particularly important, because traffic patterns can be directly associated to events.

We devise general traffic analysis techniques for extracting contextual information from WSN communications. We further investigate the inference of contextual information when the WSN transmissions are protected by traffic normalization methods, which rely on statistical source anonymity (SSA). To counter traffic analysis, we develop resource-efficient communication and routing methods for reporting events over

multi-hop routes, without revealing the event location and occurrence time, as well as the location of the sink. Our work explores the tradeoff between the communication overhead for normalizing traffic and the end-to-end real packet delay for delivering the event report to the sink. This is achieved by limiting the number of fake transmissions for obfuscating traffic patterns. To do so, we map the problem of selecting fake sources to the problem of finding a minimum connected dominating set (MCDS) that covers the WSN deployment area. We then impose transmission schedules on the fake sources to accelerate the delivery of real event reports. Finally, we propose strong privacy traffic normalization techniques that reduce the number of fake transmissions without relying on the concept of SSA. In the proposed solution, the WSN is partitioned into connected dominating sets (CDSs) that are activated in a round-robin fashion. We show that our methods reduce the communication by several orders of magnitude, while maintaining privacy under strong adversary models.

Chapter 1
INTRODUCTION

Continuous advances in electronics, wireless technologies, manufacturing processes, and software engineering have led to the proliferation of a plethora of mobile devices– mobile phones, tablets, wearables, sensors, smart consumer electronics, etc.–in our every day lives. The interconnection of these devices into a single web of communication, information, and computation gives rise to a densely meshed wireless ecosystem that transforms the way users interact with their environment. Armed with a wealth of (near) real-time information, this wireless ecosystem is expected to elevate the human perception of the complex physical space. Promising applications are emerging on every facet of our daily lives, including communications, healthcare, smart spaces, commerce, tactical communications, transportation, resource management and monitoring, manufacturing, and agriculture.

However, ubiquitous interaction with devices that collect data about user activities pose challenging privacy and security problems. Without protection mechanisms, the systems we deploy breach user privacy, often without the user's knowledge or consent. The collected information could reveal the user whereabouts, track his motion through space, infer his habits and personal preferences, record user relationships, acquaintances, and contacts, and compromise sensitive information. As an example, a fitness activity tracker continuously records the user's heart rate and GPS coordinates. This data, which is uploaded to an enterprise server for storage and analysis, lies beyond the control of the user. The enterprise could track the user by extrapolating the GPS coordinates and could also infer medical conditions by correlating the navigation data with the heart rate data. This information could also be revealed to malicious third parties that infer private data by intercepting transmissions over the

unprotected wireless medium.

In the context of electronic communications, the problem of establishing privacy has been widely studied since the seminal work of Chaum in the early 1980's [8]. In his work, Chaum proposed cryptographic methods for protecting the privacy of the source and destination, as well as the message content in electronic mail applications. However, for the wireless sensory systems deployed today, cryptography alone is not sufficient to hide private data. This is because communication patterns could be used to reveal auxiliary information that is not included within the payload of the exchanged messages. We refer to any information that is inferred without accessing the contents of the communication as *contextual information.*

In this dissertation, we investigate the leakage of contextual information in wireless systems. We focus on event-driven wireless sensor networks (WSNs), whereby wireless communications are triggered upon the detection of an important event such as the detection of an object of interest, the recording of an abnormal physical parameter, etc. Privacy in event-driven WSNs is particularly important, because data transmission patterns can be directly associated to events.

Contextual information can be exposed by eavesdropping on over-the-air transmissions and obtaining *transmission attributes,* such as inter-packet times, packet source and destination IDs, and number and sizes of transmitted packets [50, 65]. These attributes can be correlated to obtain the location and time of occurrence of a sensed event, the sink's position, and the routing paths followed by data packets [45–48, 50, 65].

For instance, Figure 1.1 shows a tactical WSN deployed in a hostile environment for area surveillance. The objective of the WSN is to detect if an intruder E entered a restricted area. When E is detected by a sensor, this sensor reports the presence of E by sending a message to the sink. In our example, sensor v_1 detects the presence of E and reports this detection by transmitting a message to the sink via the multihop path $v_2 - v_3 - v_4 -$ sink. To be aware of his possible detection, the adversary

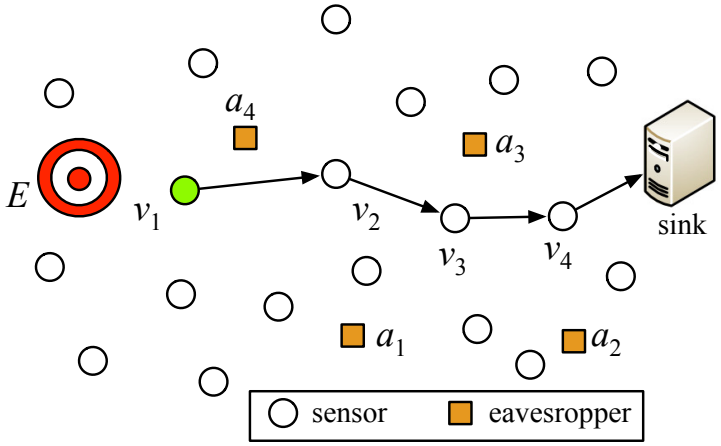


FIGURE 1.1. A tactical WSN deployed to surveil a restricted area.

deploys its own network of eavesdroppers a_1, a_2, a_3, and a_4. Eavesdropper a_4 (located in the vicinity of v_1) intercepts the transmission of the report sent by v_1. Even if the transmission is encrypted, the adversary can correlate the transmission of v_1 with the possible detection of E. Specifically, the time at which packets are first transmitted can be associated with E's detection time, and the origin of the transmissions can be associated with the location where E was detected. This information is inferred without examining the contents of the communication. Finally, the adversary may use the fact that E has been detected to change the location of E, or take other action of his choice to prevent its detection in the future.

The problem of preserving contextual information privacy in WSNs has been studied under various adversarial scenarios. Threat models can be classified based on the adversary's network view (local vs. global) or the capabilities of the eavesdropping devices (packet decoding, localization of the transmission source, etc.). Under a local eavesdropper model, eavesdroppers are assumed to intercept only a fraction of the WSN traffic [37, 45–48, 50]. Hiding methods include using random and directed random walks, adding of pseudo-sources and pseudo-destinations [43, 46–48, 76], creating of routing loops, and flooding [37]. These methods can only provide probabilistic

obfuscation guarantees, because eavesdroppers locations are unknown. Moreover, methods aiming at limiting local adversaries fail to provide privacy protection under a global eavesdropper intercepting all communications within the WSN [23, 50, 80].

A global threat model is a plausible scenario due to the relatively low sensor acquisition cost. For instance, 1,000 sensors valued at $9 each, can be obtained at an expense of $9,000 [6]. A common critique to the global adversary model is that if the adversary was able to deploy a global monitoring system, there would be little incentive to monitor the transmissions of the WSN. Instead, he could directly monitor the physical parameters of interest. However, in certain application scenarios, the adversary is interested in the communication patterns themselves rather than the events monitored by the WSN. For instance, in the example Figure 1.1 the adversary is interested in detecting when the surveillance network detects physical intrusion. Furthermore, we note that the global adversary model is relevant even when the adversary can only intercept a fraction of the WSN transmissions. As eavesdroppers are passive devices, their locations are unknown. Therefore, one cannot determine which of the transmissions were intercepted. In the absence of eavesdropper location information, one has to account for all possible eavesdropper location configurations, which is equivalent to a global adversary model.

1.1 Privacy Challenges in Event-driven WSNs

Defending against eavesdropping poses significant challenges. First, eavesdroppers are passive devices that cannot be detected without specialized hardware [52]. Second, the availability of low-cost commodity radio hardware makes it inexpensive to deploy a large number of colluding eavesdroppers. Third, even if encryption is applied to conceal the packet payload, some fields in the packet headers still need to be transmitted in the clear for correct protocol operation (e.g., PHY-layer headers used for frame detection, synchronization, etc.). From the eavesdropper's standpoint,

these unencrypted fields facilitate accurate estimation of transmission attributes.

One candidate solution is to hide contextual information by normalizing communication patterns. A simple way to achieve this goal is to transmit dummy packets according to a pre-determined probability distribution. Real transmissions take place by substituting scheduled dummy transmissions. This decorrelates the occurrence of an event from the eavesdropped traffic patterns. However, injection of dummy traffic comes at the expense of additional communication overhead, which is proportional to the number of nodes injecting dummy traffic and the number of dummy packets transmitted. Moreover, when traffic has to conform to a predetermined pattern, a real packet cannot be transmitted immediately after the event occurrence. It has to be delayed until the next scheduled transmission, thus introducing an extra delay overhead. The overhead is compounded on a per hop basis, until the packet reaches the sink. Note that the delay overhead can be reduced at the expense of reducing the inter-packet time of dummy transmissions (i.e., increasing the communication overhead). This reveals a tradeoff between the end-to-end reporting delay and the communication overhead for maintaining contextual information privacy.

1.2 Main Contributions

In this dissertation, we study the problem of protecting contextual information privacy in event-driven WSNs. We first develop methods for inferring contextual information through a network of colluding eavesdroppers that collectively process the intercepted transmissions. We propose communication- and delay-efficient traffic normalization techniques to protect the privacy of contextual information against global eavesdroppers. Our work is categorized to the following interrelated topics.

1.2.1 Contextual Information Inference in Unprotected WSNs

We develop contextual information inference techniques for an adversary that deploys a network of mote-level eavesdroppers in the WSN area. While a wealth of countermeasures have been proposed in the literature, the exact methods used to breach the WSN privacy are either tailored to specific solutions or overlooked. As a result, it becomes challenging to quantize and compare the privacy level offered by the various solutions. To remedy this limitation, we devise general traffic analysis techniques for extracting contextual information from WSN communications.

In our analysis, eavesdroppers are limited to recording the transmission time and a content hash, for each intercepted packet. We limit the recorded transmission attributes to make our analysis broadly applicable to several models including WSNs that apply packet encryption (including headers), identity anonymization (e.g., via rolling pseudonyms), and other kinds of privacy protection (location, temporal, routing path). The collected information is processed to infer the occurrence time of events, event location, and the routing path used to report the event. We apply our analysis to WSNs protected by different solution classes and evaluate the privacy level offered by each class.

1.2.2 Time Series Analysis for Breaching Statistical Source Anonymity

We further investigate the inference of contextual information when the WSN transmissions are protected by a traffic normalization method which relies on statistical source anonymity (SSA). State-of-the-art countermeasures against global eavesdroppers conceal traffic associated to real events by injecting dummy packets according to a predefined distribution. For instance, every sensor can be scheduled to transmit on average one packet per minute. To conceal real transmissions, sensors substitute scheduled dummy transmissions with real ones, thus eliminating any information leakage.

However, this theoretically perfect solution has two serious drawbacks. First, dummy transmissions create significant overhead for the energy-constrained sensors. To limit this overhead, dummy transmissions can be scheduled with the same frequency as the occurrence of real events (or even at lower rates). This amplifies the end-to-end delay for reporting an event to the sink. The source as well as every relay sensor have to delay the real transmission until the next scheduled dummy transmission. For time-critical applications this delay could be unacceptable.

Several researchers have devised methods to shorten the end-to-end report delay. The main idea is to accelerate the transmission of a real packet and delay the transmission of the following dummy packet such that the difference observed by the adversary is not statistically significant. Such methods achieve SSA. The primary assumption of SSA is that the adversary executes some statistical goodness of fit test to verify if the observed inter-packet times fit an a priori known distribution.

In our study, we investigate if SSA methods still leak information about the occurrence of real events, despite passing goodness of fit tests. The main premise of our investigation is the fact that the modification of inter-packet times for accelerating real transmissions and recovering the distribution mean can cause detectable outliers. These outliers can be used to distinguish between intervals containing real transmissions and intervals containing only dummy ones. Moreover, in an interval with real transmissions, the adversary can pinpoint with high confidence the exact time when a real transmission occurred by using the outlier position within a time series.

We show that SSA can be breached by applying well-known outlier detection methods for time series data. These methods are designed to identify specific data points that break the temporal data continuity by causing sudden changes or unusual patterns. We apply an autoregressive model that does not require a priori knowledge of the distribution used to schedule dummy transmissions. Using this model, the adversary can pinpoint intervals with real events with a probability far greater than a random guess. Moreover, the adversary can use the location of the outliers within

the time series to substantially limit the candidate event occurrence times. Prior methods for identifying events are limited to identifying intervals, which can have long durations. We note that our goal is not to develop an outlier detection method that maximizes the probability of event detection. The field of outlier detection is fairly broad, with extensive studies in a large number of application domains. We aim at demonstrating that such methods can breach SSA.

We extend the outlier detection method to time series generated by collectively analyzing the transmissions over multiple hops. The key insight here is that correlated transmission patterns can be observed, when each of the relay sensors accelerates the transmission of a relayed real packet. The appearance of short-long transmission patterns in sequence amplifies the outliers in the "collective" time series and reduces false alarms.

We show that existing countermeasures that call for the insertion of fake events which mimic real events are not effective when outlier detection is applied over multiple hops. This is because sensors independently insert fake events without coordinating over multiple hops. Therefore, these fake events do not appear is if the occur in sequence, as is the case during the relay of real packets.

1.2.3 Energy-efficient Traffic Normalization for SSA

We first address the problem of contextual information privacy from a resource point of view. As sensors are typically battery-operated devices limited by their finite energy, we attempt to conserve energy resources by reducing the number of sensors that transmit dummy packets for traffic normalization. Our work differs from previously proposed solutions in that we address the tradeoff between the communication and real-packet delay overhead by limiting the set of sensors required to inject dummy traffic. The number of dummy sources does not grow with the deployment density. To reduce the number of dummy transmissions, we map the problem of selecting dummy

sources to the problem of finding a minimum connected dominating set (MCDS) that covers the WSN deployment area. The MCDS guarantees that every sensor is at most one hop away from a MCDS sensor and that every eavesdropper observes dummy traffic patterns.

To prevent the global adversary from detecting real transmissions, we propose rate control techniques that regulate the transmission of dummy packets for nodes in the CDS. In the case of real packets, nodes in the CDS transmit them by simply replacing the next scheduled dummy packet by a real one. However, for nodes that are not part of the CDS, real packets are transmitted at a rate that has statistically insignificant impact on the traffic patterns observed by eavesdroppers in the neighborhood of the real source.

We consider the cases where eavesdroppers perform traffic analysis independently and when network traffic is collectively analyzed by a central server. Independent analysis yields faster detection and considerably lower communication overhead. Collusion yields higher accuracy in the information extracted. Moreover, while previous methods focus on protecting only source location, our methods are designed to preserve the privacy of the source location, event location, event occurrence time, and the sink location.

1.2.4 Traffic Normalization beyond SSA

As shown in Chapter 4, SSA methods can actually leak contextual information if the adversary applies specialized statistical test designed to detect anomalous data points. To eliminate this information leakage, we propose traffic normalization techniques that reduce the number of dummy transmissions without relying on statistical anonymity. In the proposed solution, the WSN is partitioned into connected dominating sets (CDSs) that are activated in a round-robin fashion. We aim at hiding the event location, its occurrence time, and the sink location from global eavesdroppers.

We partition the network in two types of CDSs; minimum connected dominating sets (MCDSs) and MCDSs with shortest paths to the sink (SS-MCDSs). The former one includes sets of minimum size, which aims at minimizing the amount of dummy traffic introduced. The latter one is introduced to include the shortest paths from any sensor in the CDS to the sink, designed the reduce the end-to-end delay of real packets. We characterize the algorithmic complexity for partitioning the network into MCDSs and SS-MCDSs, and develop efficient heuristics.

We note that even when sensors transmit dummy (or real) packets in an uncoordinated fashion, the end-to-end delay for reporting a real event can increase significantly. To reduce this delay, we propose a rate control scheme, that loosely coordinates sensor transmission over multi-hop paths without revealing real traffic patterns or the traffic directionality.

1.3 Dissertation Organization

The rest of this dissertation is organized as follows. Chapter 2 presents an extensive review of the previous work on the areas of anonymous communications in wired and wireless networks, as well as source privacy in wireless sensor networks. In Chapter 3, we introduce traffic analysis techniques to infer contextual information. Chapter 4 presents an extensive analysis on the limitations of statistical methods used to reduce the end-to-end delay of real packets. Chapter 5 focuses on the problem of reducing the energy overhead for hiding contextual information in a statistical sense. In Chapter 6, we propose two traffic normalization techniques based on the construction of connected-dominating-sets partitions of the network. These techniques are immune to any statistical analysis. Finally, in Chapter 7 we summarize our findings.

Chapter 2
Preliminaries and Related Work

In this chapter, we provide definitions of privacy and security relevant to the main technical contributions addressed in this dissertation. Furthermore, we state the network and adversary models. Finally, we give an overview of the state-of-the-art in privacy protection methods for WSNs.

2.1 Definitions of Privacy

Privacy in WSNs can be defined with respect to the types of information that must remain confidential from external observers. The information types include not only the contents of WSN transmissions, but also the contextual information that can be derived by observing communication patterns. Here, we provide formal definitions about the different privacy types.

Definition 1 (Content Privacy)**.** *The ability of a WSN to provide confidentiality, integrity, and freshness of the messages transmitted in a WSN [10].*

Protecting content privacy is a fundamental part of a secure WSNs, since it directly involves the content of the reports generated by sensors that observe/detect the occurrence of events. Satisfying content privacy prevents an adversary from accessing the content of an event report, altering the content of a packet before it is delivered at the destination, and replaying old event reports [69, 75]. Content privacy is typically achieved via cryptographic methods [10]. However, satisfying content privacy alone does not prevent an adversary from accessing information such as the location and occurrence time of an event. Hiding such information in addressed by contextual privacy.

Definition 2 (Contextual Privacy)**.** *The ability of a WSN to prevent the correlation of timing and communication attributes with confidential information related to an event as well as the location and identity of wireless sensors [16]. Typically contextual privacy in WSNs is categorized as follows.*

1. **Location privacy** *refers to the location of occurring events as well as the location and identity of the sensor reporting the event occurrence. It also includes the location of sensors in the multi-hop routing path used to report an event to the sink and the location of the sink.*

2. **Temporal privacy** *aims at protecting the temporal relation between time at which packets are eavesdropped and the occurrence time of events.*

2.2 System and Adversarial Models

2.2.1 System Model

We consider a set of sensors $\mathcal{V}$, deployed to sense physical events within a given area. When a sensor detects an event of interest, it sends a report to the sink via a single-hop or a multi-hop route (depending on the relative sensor-sink position). Sensors are aware of the locations of their one- and two-hop neighbors by using a neighbor discovery service [67]. The WSN is loosely synchronized to a common time reference [61].

Content confidentiality is protected using standard cryptographic methods. Packet transmissions are re-encrypted on a per-hop basis to prevent tracing of relayed packets. For implementation details on link-layer re-encryption, we refer the reader to [2, 46, 48]. Sensors can establish trust during a bootstrapping phase by using any suitable key establishment scheme [27, 54]. The sensors then share cryptographic secrets (pairwise keys, group keys, etc.) to execute cryptographic protocols. Moreover, sensors within a two-hop neighborhood share random seeds used for pseudo-random

number generation. Key establishment and management, as well as content privacy is beyond the scope of this work (interested readers are referred to [10, 26, 77] for a relevant literature review).

2.2.2 Adversarial Model

The adversary aims at passively inferring contextual information by analyzing the traffic intercepted by eavesdropping devices. We assume that the adversary deploys a large number of cheap eavesdropping devices $\mathcal{A}$ that passively monitor WSN transmissions. An eavesdropper $e \in \mathcal{A}$, located at ℓ_e, has a reception area C_e. The number and locations of the eavesdroppers remain unknown. Due to the scale of the attack, the eavesdropping devices cannot differentiate the packet sources within their eavesdropping range by employing radiometric hardware such as antenna arrays, spectrum analyzers, etc. Eavesdroppers are limited to recording communication attributes of interest such as packet sizes, inter-packet times, unencrypted header fields, etc. These attributes are used to passively inferring contextual information of interest. The contextual information of interest is:

a. the location of a physical event,

b. the occurrence time of that event, and

c. the sink location.

We consider two traffic analysis models. In the first model, eavesdroppers are assumed to independently perform statistical tests. Their observations sets do not propagate to a central server in order to reduce the delay between the event occurrence and detection and also reduce the energy overhead for operating the eavesdropper network. An eavesdropper notifies the central server only if it detects a statistical change in the observed traffic patterns in his vicinity. This model is suitable when the adversary is interested in near real time detection and wants to prolong the lifetime of

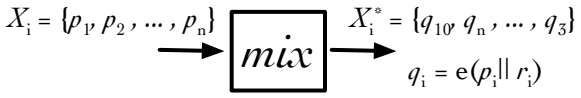


FIGURE 2.1. A mix node that uses encryption, padding and re-ordering to achieve anonymity

battery-operated sensors. Note that this still captures a global adversary capable of eavesdropping on all communications.

In the second model, eavesdroppers are assumed to collude by transferring their observations to a central server, similar to the adversary assumed in [19,50,65,80]. The server collectively performs traffic analysis on all observations. The latter adversarial model is suitable for contextual information that does not vary significantly with time such as the sink's location. Finally, we assume that the adversary does not launch active attacks (e.g., jamming, packet modification, packet injection attacks, etc.) against the WSN.

2.3 Related Work

2.4 Anonymous Communication Systems

The problem of anonymous communication systems (ACS') was initially studied by Chaum in the 1980s [8]. Chaum focused on anonymous and untraceable electronic mail based on mixing messages. This solution proposes the implementation of mix nodes, designed to transform an input set of packets X_i into a set X_i^*, where X_i and X_i^* are not correlated. That is, X_i cannot be obtained by simply observing X_i^*. For instance, in Figure 2.1 we present a mix node that changes the transmission order of packets, adds random padding to normalize packet sizes, and encrypts the packets to prevent the adversary from reading the content. In the example, set $X_i = \{p_1, p_2, \ldots, p_n\}$ is transformed into $X_i^* = \{e_K(p_{10}||r_{10}), e_K(p_n||r_n), \ldots, e_K(p_3||r_3)\}$, where p_i is a packet, $e_K()$ is an encryption function, K is an encryption, "||" denotes

the concatenation operation, and r_i is a random padding. Note that, it is assumed that the adversary cannot obtain the set of packets X_i before it is processed by the mix node. The use of mix nodes has been extended to other applications such as ISDN networks [57], web mix cascades [4], mix networks and peer-to-peer networks [30, 51, 59], and has been generalized to electronic communications [21, 22].

Onion routing implements ACS in the domain of context circuit-based routing [25, 58]. Such implementations work similarly to mix networks. In onion routing, all packets exchanged between two nodes follow the same path created by the first message according to message labeling techniques. That is, the first packet p_i is encapsulated in multiple layers of encryption (e.g., $e_{K_1}(e_{K_2}(e_{K_3}(p_i)))$), depending on the number of hops in the path between the source and destination. In this case, $e_{K_i}()$ represents an asymmetric key encryption function, where K_i is a public key. Each hop decodes one layer of encryption to obtain the id of the next hop where the packet is transmitted to. The identity of the source is protected since each intermediate node only knows the identity of the immediately preceding node in the path. These systems are effective even if the adversary

In [9], Chaum proposed an ACS that uses cryptographic methods to hide the identity of the communication parties within a set. Given a set of nodes S and the transmission of a message from a node $v \in S$, any node $u \in S$ (other than v) is aware of the transmission but not of the identity of its source. Von Ahn et al. [72] extend the work of Chaum by introducing the notion of full anonymity and a weaker version of anonymity, referred to as k-anonymity, where $k < |S|$. In this case, the identity of a sender v is narrow down to a set of k nodes. In [40], the authors propose a probabilistic approach based on anonymity sets. In [55, 56], the authors extend the concept of anonymity to unlinkability and unobservability. The former is defined as the inability of an adversary to correlate a node and its identity based on a-priori knowledge of the system. In the case of the latter, the authors define an observability set. Any communication party (sender of receiver) that belongs to this set is indistinguishable

to any other communication party from the same set. That is, an adversary does not notice that a member of the unobservability set has transmitted or received a message. In [62], the authors show that an adversary with knowledge of the system, might be able to significantly reduce the size of unobservability sets. This vulnerability is used to prove that the size of the unobservability sets is not a good measure of anonymity. They propose to quantify anonymity based on information theoretic definitions [18,64], which provides a more adequate measure of the anonymity offered by a given system.

In [24], the authors introduce the use of dummy traffic in mix networks. In this case, the mixes inject dummy traffic to prevent the adversary from performing traffic analysis at any part of the network. That is, each node transmits packets at a given rate, if the node does not have a real packet to send, it generates a dummy one. In [20], Dai proposes to set the traffic at any link between mix nodes to a constant value R_1. Dummy packets are introduced to "pad" a link when the traffic is below R_1.

The implementation of mix nodes in the wireless domain is challenging due to the open nature of the medium, as well as limitations of computational resources of some wireless devices [41]. However, concepts like anonymity, unlinkability and unobservability have been adapted to wireless applications. For instance, the authors of [73] propose to randomize transmissions to provide anonymity of system components, and hide the cluster structure and routing paths in a wireless network. The solutions proposed in [28, 29] are based on the unlinkability and unobservability properties. Also, several authors have proposed the injection of dummy traffic to provide anonymity in wireless networks [5, 19, 37, 50, 78–80]. Several of these techniques measure the effectiveness of anonymity methods using information-theoretic metrics.

2.5 Anonymous Communications in Wireless Networks

In the wireless domain, the problem of anonymous communications has become relevant since the open nature of the medium makes it easy to eavesdrop on private communications. In event-driven wireless communications, such as wireless sensor networks (WSNs), transmissions are triggered by the occurrence of an event. Even when traffic is encrypted, the mere transmission of packets reveals the occurrence of an event. Contextual information such as the transmission time and the packet's source location can be correlated to the time and location of the reported event. Contextual information privacy methods for WNSs can be categorized based on privacy type they offer, the capabilities of the adversary, and the methods employed to achieve privacy. Extensive literature reviews of the state-of-the-art in this domain can be found in recent surveys [12, 16]. Here, we present relevant work on different traffic analysis techniques employed by the adversary to obtain contextual information, as well as defense strategies against local and global adversaries.

2.5.1 Traffic Analysis

A passive adversary, local and global takes advantage of the traffic generated by a node (or the entire network) to obtain contextual information of interest. In [80], the adversary eavesdrops at wireless sensor's transmission and inspects the content exchanged between nodes. The headers content are used to infer the identity of the nodes. For the case when the contents are encrypted, the authors of [70, 80] propose an adversary that eavesdrops at the sink, and tracks back the direction where the traffic comes from in order to get to the source node. A rate monitoring and time correlation attack was also proposed in [80]. In the former, the adversary classifies nodes according to their packet rates. A node with a high packet rate is said to be close to the sink. For the latter, the adversary correlates the transmission times of neighboring nodes in order to determine the direction and the route that a packet

traverses to the sink.

The authors of [36, 49] propose a timing analysis attack. The study transmission times of neighboring nodes to infer sensitive information such as the topology of the network. In [39], signal processing techniques are used to obtain the angle-of-arrival and the received signal strength to infer the location of the nodes.

2.5.2 Local Adversary

A local adversary has a local view of the WSN [63]. The adversary eavesdrops a limited number of ongoing communications within the WSN. A local adversary deploys a single, or a small number of mobile devices (typically around the sink) that attempt to identify an information source by back tracing the intercepted transmissions [47, 48].

Mahmoud et al. considers a highly capable local adversary that can precisely localize the source of a packet transmission using radiometric hardware [48]. The eavesdropper is assumed to be initially located in the sink's vicinity. He attempts to breach the source's location privacy by back tracing packets from the sink to the source. To hide the source location, the authors propose the introduction of dummy traffic from node clouds that become active only during real data transmissions. They further propose efficient cryptographic schemes for making real traffic indistinguishable from dummy traffic. In [47], a novel privacy-breaching attack, referred to as the *hotspot-locating attack* is presented. In a hotspot-locating attack, a set of eavesdropping devices perform traffic analysis to locate areas with high transmission activity. By performing back-tracing from the sink, the authors show in an analytic manner that eavesdroppers are eventually capable of locating the traffic sources. In [46], the authors present a comprehensive solution to the hotspot-locating attack. They devise a cloud-based solution, whereby a set of fake sources generate bogus traffic to hide the source of a real event.

In [45], the authors proposed the use of multiple routing paths to prevent a local

adversary from tracing packets back to their source. In these techniques, each node divides its neighborhood into a closer set and a further set. The closer set includes nodes at a short hop count to the sink. The further set contains those nodes within a hop distance to the sink equal or longer than the source's distance. When a node has a real packet for transmission it randomly selects the next hop from the closer set. Any node that belongs to the further set overhearing a transmission broadcasts a dummy packet with some probability. The probability distributions used to select nodes from the closer set and to transmit dummy packets are adapted to maintain the same average communication overhead per node. The source privacy is preserved under the local adversary model.

In [42,43,66,74,76], the authors proposed routing techniques for hiding the sources' locations. In these techniques, packets are diverted to a fake source located several hops away from the real source using unicast transmissions. The fake source forwards packets to the sink. If an adversary cannot intercept the unicast path to the fake source, the original source is protected In [60], the authors proposed the homogenous injection for sink privacy (HISP) protocol, which is based on random routing and the injection of dummy traffic. In HISP, when a node has a real packet for transmission, it chooses two neighbors. The first neighbor receives the real packet, while the second one receive a fake one. Once a node receives a packet, it randomly forwards the packet to one of its neighbors.

2.5.3 Global Adversary

In WSNs, a global adversary is assumed to eavesdrop on all transmissions. This is usually achieved at a low cost by deployment of a network of mote-level eavesdroppers. In [50], the authors proposed traffic normalization techniques based on the injection of dummy traffic; periodic collection and source simulation schemes. In periodic collection, each sensor generates bogus packets at a fixed packet rate. To transmit real

data, sensors substitute dummy packets with real ones while maintaining the same packet rate. This method prevents colluding eavesdroppers from determining the source of real traffic, the path to the sink, and the sink's location, at the expense of significant communication overhead. In the source simulation method, the communication overhead is reduced by selecting only a subset of sensors as sources of dummy traffic. Bogus sources are chosen to simulate the expected distribution of real events. This distribution has to be known a priori.

In [37], the authors considered a threat model that uses multiple local adversaries that collaborate to share information on the topology and the traffic in the WSN. This technique uses fake sources to protect the privacy of real ones. To do so, each time a real or fake source reports an event, the packet is flooded in the network. To choose fake sources, the neighbors or the source transmit a message to the nodes located at a distance equal to their hop count to the sink. Once these nodes receive the message, they randomly decide on becoming a fake source. This scheme protects the source and sink location, however the overhead of flooding the network is high. Besides their overhead, traffic normalization techniques incur unavoidable delay. This is due to the fact that transmissions of real packets are delayed in order to conform to predefined transmission patterns. Moreover, transmissions over multiple hops are largely uncoordinated, thus making packet delay proportional to the number of hops required to reach the destination (sink).

In [80], the authors propose methods to reduce dummy traffic propagation. The network is divided into cells that are designed to cover the minimum area unit where events can occur. To normalize traffic, each cell generates encrypted bogus traffic. When a real event has to be reported, the bogus packets are replaced with real ones. In the *Proxy-based Filtering Scheme (PFS)*, a subset of cells are designated as proxies in different parts of the deployment area. Each cell transmits packets (real or dummy) to the closest proxy who filters dummy traffic and forwards real packets to the sink. At proxies, real traffic is transmitted along with dummy traffic to maintain the traffic

pattern uniformity. In the *Tree-based Filtering Scheme (TFS)*, proxies are organized as a tree rooted at the sink to expedite packet delivery and reduce the number of filtered dummy packets. However, TFS reveals the sink's location, which may be critical for some application scenarios. In [5], the authors argue that since TFS and PFS impose a hierarchy, each proxy performs only a single type of filtering. They proposed the *Optimal Filtering Scheme (OFS)*, in which proxies are organized as a general directed graph instead of a tree. This allows each proxy to filter packets from every other proxy as well as from other sensors. Using a linear programming, the authors prove that OFS achieves optimizes the network lifetime.

In [78], the authors introduced an aggregation-based scheme that reduces the propagation of dummy traffic. The WSN is divided into clusters, each with its own clusterhead. The clusterheads are organized to a tree structure rooted at the sink. In this scheme, each sensor transmits dummy traffic to its respective cluster heads. Finally, cluster heads aggregate the received traffic and forward it towards the sink.

The authors in [65] proposed the FitProbRate scheme, in which dummy traffic is introduced according to a pre-determined distribution P_F with mean μ_F. This represents the distribution of the time imd_j between the j^{th} and $j+1^{\text{th}}$ dummy packet transmissions. The adversary is assumed to know the pre-determined distribution P_F. However, the initial seed value used by the nodes to generate imd_j is assumed to be secret, which prevents the adversary from obtaining the actual times used by nodes to transmit dummy packets. In order to detect the transmission of a real event, the adversary maintains and analyzes a sliding window of k inter-packet times observations $X_i = \{imd_{i-k+1}, \ldots, imd_i\}$, where imd_i is the most recent time obtained by the adversary. The adversary uses goodness of fit hypothesis testing to check whether set X_i is distributed according to P_F. The hypothesis test is applied each time a new value of imd_j is obtained. If the test fails, the adversary concludes that a real transmission has occurred and it is associated to imd_j.

The authors design the FitProbRate scheme such that each real packet can be

transmitted at an earlier time than the time dictated by distribution P_F. Moreover, FitProbRate ensures that data set X_i, observed by the adversary, satisfies the property of $(\alpha, \epsilon)-$unobservability. This is defined as follows.

Definition 3. *$(\alpha, \epsilon)-$unobservability ($\alpha, \epsilon > 0$) is satisfied when the empirical probability distribution P_{X_i} estimated from data set X_i (with mean μ_{X_i}) is statistically indistinguishable from a pre-determined probabilistic distribution P_F with mean μ_F under the following conditions:*

a. P_{X_i} and P_F are indistinguishable from each other, according to a statistical hypothesis test of significance level α.

b. $(1-\epsilon)\mu_F \leq \mu_{X_i} \leq (1+\epsilon)\mu_F$

As soon as an event has occurred, a sensor observing the event generates λ packets with the a central server (e.g., sink) set as destination. Each real packet is transmitted at an earlier time than the one corresponding to the next scheduled dummy transmission (i.e., $imd_j - \beta$ for $\beta > 0$). To prevent the adversary from detecting these early transmissions, the value of $imd_j - \beta$ is chosen such that a statistical hypothesis test determines that set $X_j = \{imd_{j-k+1}, \ldots, imd_j - \beta\}$ is distributed according to P_F. However, the authors note that the application of this technique on several real transmissions will affect the value of the mean of the sets observed by the adversary. To compensate for this mean reduction, each time the mean of the observation set X_i is under a certain threshold, (i.e., $|\mu_{X_i} - \mu_{P_F}| \leq \epsilon \times \mu_F$), FitProbRate schedules the next transmission (dummy or real) at a later time than one drawn from P_F (i.e., $imd_{i+1} + \delta$, for $\delta > 0$). The value of δ is chosen such that X_{i+1} satisfies the two conditions of $(\alpha, \epsilon)-$unobservability. The authors show that FitProbRate scheme can reduce the delay of real transmissions up to a 10% of the one achieved when real transmission times are drawn from a P_F distribution, while providing of $(\alpha, \epsilon)-$unobservability.

We note that ProbFitRate is designed to satisfy the distribution indistinguishability condition of $(\alpha, \epsilon)-$unobservability only on sets $(X_j, \ldots, X_{j+\lambda-1})$, where $(imd_j, \ldots, imd_{j+\lambda-1})$ are associated to the λ real packet transmissions. However, at least one of the inter-packet times $(imd_j, \ldots, imd_{j+\lambda})$ is also part of sets $(X_{j+\lambda}, \ldots, X_{j+\lambda+k-1})$. Because ProFitRate only check for deviation of the mean, there is no guarantee that the distribution indistinguishability property is satisfied for these sets.

In [19] and [79], the authors expose some security vulverabilities of the FitProbRate scheme. Since real transmissions are speeded up and likely the next dummy transmission is delayed (for mean recovery), this introduces certain data patterns that may lead the adversary to detect real packets. In their analysis, they show that the inter-packet time for real transmissions tends to be

$$imd_i < \mu_F,$$

and likely the following time used to recover the mean of P_F is

$$imd_{i+1} > \mu_F,$$

Inter-packet times imd_i, imd_{i+1} form a "short-long pattern". To facilitate the analysis of these short-long patterns, a simple binary coding technique has been proposed, in which each imd_i is assigned a "binary digit" C_i as follows,

$$C_i = \begin{cases} 0, & \text{if } imd_i < \mu_F \\ 1, & \text{if } imd_i \geq \mu_F. \end{cases}$$

Note that, a short-long pattern is mapped to a binary sequence of the form '01'.

In [79], the authors used '01' patterns to detect real transmissions. They measured the probability of detecting a real event simply by associating a '01' pattern to a real transmissions. Their experiment shows that the probability that a '01' corresponds to a real transmission is 0.0213. They also considered experiments in which an event is reported with the transmission of $\lambda \geq 1$ packets. In this case, the adversary searches

for a sequence of λ consecutive '0's and one '1'. For instance, if $\lambda = 5$, instead of targeting a '01' sequence, the adversary targets a '000001' sequence. Under this scenario, the probability that '000001' is associated with a real transmission is 0.2. However, this detection method is inefficient from the adversary's perspective, as the chances of detecting a real transmission are low. The authors reduce the detection rate, by introducing a dynamic mean scheme, in which the value of μ_F is randomly changed after a certain number of dummy transmissions. They showed that, when the mean is adjusted every 10 transmissions, the probability that '000001' is associated with a real transmission reduces to 0.15. This technique is not effective against an adversary that uses a seasonal-trend decomposition process on the observed data [14]. Since the mean is changed on a consistent basis (e.g., every 10 transmissions), this effect can be associated the trend component of the data. A seasonal-trend decomposition process separates the data into its seasonal, trend and residual component. The seasonal component is not important on this analysis, since it represents the cyclic part of the data. At the end, the adversary can perform the statistical analysis of its choice on the residual component.

In [19], the authors show that a sequence $S_R = \{C_1, \ldots, C_k\}$, obtained during a period of time where real transmissions occurred, can be distinguished from a sequence $S_F = \{C'_1, \ldots, C'_k\}$, obtained during a period of time where only dummy transmissions where observed. To do so, these sequences are compared with respect a reference one defined as follows,

$$ref = \{0101 \ldots 01\}.$$

They show that the correlation factor between the former and ref is higher than the correlation factor between the latter and ref. Specifically, they show that the correlation factor $\rho(S_R, ref)$ is greater than $\rho(S_F, ref)$ in more than 70% of the cases. Thus, the adversary is able to distinguish between S_R and S_F. However, the adversary is not able to pinpoint a short-long pattern associated to a real transmission.

To reduce the detection of real intervals, the authors proposed the random injection of fake '01' patterns in S_F, such that the correlation factor of both S_R and S_F is the same. This security measure reduces the detection rate to 50%, equivalent to a random guess. As the authors suggest, this method is not intended to be a complete solution, and its purpose is to illustrate the idea of fixing the correlation factor on each observed sequence.

Chapter 3

Traffic Analsysis methods for inferring contextual information

In this chapter, we devise general traffic analysis techniques for extracting contextual information from WSN communications. Eavesdroppers are limited to recording packet transmission times and a packet content hash of any transmission within their reception range. These limitations are imposed to make our methods broadly applicable to several models including WSNs that apply packet encryption (including headers), node identity anonymization (e.g., via rolling pseudonyms), and other kinds of privacy protection (location, temporal, routing path). The collected information is analyzed to infer the event location, the event occurrence time, the sink location, and the routing path used to report the event. This work is meant to serve as a baseline for comparing countermeasures under any assumptions. We apply our analysis to WSNs protected via path randomization methods [43, 76], probabilistic flooding [39], and dummy traffic methods [50]. We start by introducing fundamental definitions used by our methods.

3.1 Preliminaries

We adopt the unit disc graph communication model [13], in which sensors have identical transmission radii. Moreover, sensors are assumed to be capable of applying power control to set their transmission radius to a discrete number of ranges.

We introduce fundamental definitions of a *transmission set* and an *observation set* used to determine the information at the disposal of the adversary. The transmission set is a truthful representation of all WSN transmissions taking place over a period of time. This set reflects the *maximum* information that could be captured by the

adversary. The observation set represents the *actual* information that is captured by the adversary for a specific eavesdropper deployment and assumed capability.

Specifically, each packet p_i is associated with a unique signature

$$\sigma(p_i) = \{h(p_i), t(p_i), \ell(p_i)\},$$

where $h(p_i)$ is a hash digest of p_i, $t(p_i)$ is the transmission time of p_i, and $\ell(p_i)$ is the location of the originating sensor. The signature $\sigma(p_i)$ constitutes the *ground truth* for the transmission of p_i. This ground truth may differ from the observation of p_i by an eavesdropper e, who tags p_i with $tag_e(p_i) = \{h(p_i), t(p_i), \ell_e\}$. A $tag_e(p_i)$ differs from $\sigma(p_i)$ in the location attributed to the source of p_i (the time difference due to propagation delay between p_i's transmission time and its reception at e is assumed negligible). Instead of $\ell(p_i)$, an eavesdropper e could at least attribute p_i to its own location ℓ_e and approximate $\ell(p_i)$ with accuracy that equals to the reception area C_e. Using the packet signatures and tags, we define the transmission set and observation set as follows.

Definition 4 (Transmission Set)**.** *For a sensor $v \in \mathcal{V}$, the transmission set $\Theta_v(W)$, defined over an epoch W is:*

$$\Theta_v(W) = \{\sigma(p_i) : \ell(p_i) = \ell_v, t(p_i) \in W\}.$$

The transmission set for the entire network over W is:

$$\Theta(W) = \{\Theta_v(W) : v \in \mathcal{V}\}.$$

Definition 5 (Observation Set)**.** *For an eavesdropper e, the observation set $\mathcal{O}_e(W)$ over W, is:*

$$\mathcal{O}_e(W) = \{tag_e(p_i) : t(p_i) \in W\}.$$

The observation set captured by $\mathcal{A}$ over W is:

$$\mathcal{O}(W) = \{\mathcal{O}_e(W) : e \in \mathcal{A}\}.$$

The adversary's goal is to infer contextual information by analyzing $\mathcal{O}(W)$. We evaluate the privacy loss due to eavesdropping by computing the accuracy of the information inferred by the adversary. For instance, if the adversary is interested in the event location, we quantify privacy as the distance between the inferred location based on $\Delta(W)$ and the location of the sensor that reported the event. We call this measure *privacy distance* and formally define it as follows.

Definition 6 (Privacy Distance)**.** *Let $\epsilon \in \mathbb{R}^n$ be some private information of interest, estimated as $\Xi \in \mathbb{R}^n$ based on eavesdropping. The* privacy distance *of ϵ is*

$$\Pi = \int_{\xi \in \Xi} s(\xi)\, P(\xi)\, \mathrm{d}\xi$$

where $s(\xi)$ is the Euclidean distance between ϵ and $\xi \in \Xi$, and $P(\xi)$ a probability measure over the points in Ξ.

The privacy distance Π is a direct measure of the adversary's success in localizing physical events in space and time. For the source and sink location privacy, $\epsilon, \Xi \in \mathbb{R}^2$. For the event occurrence time, $\epsilon, \Xi \in \mathbb{R}^+$. For example, in the WSN of Figure 3.2, v_1 reports the occurrence of event Ψ during epoch W by transmitting $\Theta_{v_1}(W)$ to the sink. Eavesdroppers e_1 and e_4 capture $\mathcal{O}_{e_1}(W)$ and $\mathcal{O}_{e_4}(W)$ respectively. By jointly analyzing the collected observation sets, the adversary localizes the event source to $\Xi = C_{e_1} \bigcap C_{e_4}$. All points within Ξ are assumed equally likely event sources (there is no further information to bias the event location within Ξ). Therefore $P(\xi) = 1/\mathrm{area}(\Xi)$. For this case,

$$\Pi = \frac{1}{\mathrm{area}(\Xi)} \int_{\xi \in \Xi} s(\xi)\, \mathrm{d}\xi.$$

3.2 Traffic Analysis

In this section, we propose a traffic analysis method to obtain contextual information. Our method proceeds in the two stages: a traffic cleansing stage followed by a

contextual information inference stage. Since our method is applied on a per-epoch basis, we omit the W notation when it is redundant.

3.2.1 Traffic Cleansing

The observation sets recorded by the scattered eavesdroppers are likely to contain duplicate tags. This is because more than one eavesdropper may overhear the same packet transmission. In the traffic cleansing stage, the adversary uses duplicate tags in the observation set $\mathcal{O}$ to obtain a better estimation of transmission set Θ.

In Algorithm 1, we present a process for attributing tags to different sensors and eliminating duplicate tags. Specifically, for two eavesdroppers a and e with overlapping reception areas, we divide their respective observation sets to tags intercepted in $C_a \cap C_e$, $C_a \backslash C_e$, and $C_e \backslash C_a$. Each tag set is associated with a *sensor label* that represents the transmissions within the respective area. The location of each sensor label is approximated by the area intersection (difference) between C_a and C_e. Details are described in Algorithm 1.

Algorithm 1: Tag Cleansing

Step 1: For each eavesdropper e, set $\hat{\Theta}_v = \mathcal{O}_e$, $\hat{\ell}_v = C_e$, and $NS_e = \{v\}$. Here, v is a label for any sensor in C_e, $\hat{\ell}_v$ is the approximation area of v's location, and NS_e is the estimated sensor neighborhood of e.

Step 2: For each $\hat{\Theta}_v$ and $a \in \mathcal{A}, a \neq e$, if $\hat{\Theta}_v \cap \mathcal{O}_a \neq \emptyset$ **and** $\hat{\Theta}_v \backslash \mathcal{O}_a \neq \emptyset$, replace $\hat{\Theta}_v$ with,

$$\hat{\Theta}_u = \hat{\Theta}_v \cap \mathcal{O}_a, \quad \hat{\Theta}_w = \hat{\Theta}_v \backslash \mathcal{O}_a$$

Here, the intersection and complement set operations are defined based on the packet hash/timestamp dual contained in the tags. Labels u and w represent new sensor labels in e's reception range, i.e., $NS_e = \{u, w\}$.

Step 3: Approximate the locations of u and w by $\hat{\ell}_u = \hat{\ell}_v \cap C_a$ and $\hat{\ell}_w = \hat{\ell}_v \backslash C_a$, respectively.

Step 4: Compute $\mathcal{O}$ and an estimate $\hat{\mathcal{V}}$ of set $\mathcal{V}$ as:

$$\hat{\mathcal{V}} = \{v : v \in NS_e,\ \forall e \in \mathcal{A}\},\ \ \mathcal{O} = \{\hat{\Theta}_v : v \in \hat{\mathcal{V}}\}.$$

Step 5: To eliminate duplicates from $\mathcal{O}$ and $\hat{\mathcal{V}}$, find $\hat{\Theta}_v$, $\hat{\Theta}_u$, with $\hat{\Theta}_v = \hat{\Theta}_u$. Discard $\hat{\Theta}_u$ and update $\hat{\mathcal{V}} = \hat{\mathcal{V}} \backslash \{u\}$.

Consider the application of Algorithm 1 on the WSN of Figure 3.1. Sensor v_1 reports an event by sending packets p_1, p_2, and p_3, which are relayed by v_2 (as p_4, p_5, and p_6) and later by v_3 (as p_7, p_8, and p_9). Traffic is eavesdropped by e_1 and e_2, which create sets:

$$\begin{aligned}\mathcal{O}_{e_1} =& \{tag_{e_1}(p_1), \ldots, tag_{e_1}(p_6)\},\\ \mathcal{O}_{e_2} =& \{tag_{e_2}(p_4), \ldots, tag_{e_2}(p_9)\}.\end{aligned}$$

In Step 1, label u_1 is associated to $\hat{\Theta}_{u_1} = \mathcal{O}_{e_1}$ and u_2 to $\hat{\Theta}_{u_4} = \mathcal{O}_{e_2}$. Steps 2 and 3 define new labels for e_1, based on the tag intersection and difference of $\hat{\Theta}_{u_1}$ and $\mathcal{O}_{e_2}$.

$$\begin{aligned}\hat{\Theta}_{u_3} =& \{tag(p_1), tag(p_2), tag(p_3)\},\ \ \hat{\ell}_{u_3} = \hat{\ell}_{u_1} \cap C_{e_2},\\ \hat{\Theta}_{u_4} =& \{tag(p_4), tag(p_5), tag(p_6)\},\ \ \hat{\ell}_{u_4} = \hat{\ell}_{u_1} \backslash C_{e_2}.\end{aligned}$$

Similarly for e_2,

$$\begin{aligned}\hat{\Theta}_{u_5} =& \{tag(p_4), tag(p_5), tag(p_6)\},\ \ \hat{\ell}_{u_5} = \hat{\ell}_{u_2} \cap C_{e_1}\\ \hat{\Theta}_{u_6} =& \{tag(p_7), tag(p_8), tag(p_9)\},\ \ \hat{\ell}_{u_6} = \hat{\ell}_{u_2} \backslash C_{e_1}.\end{aligned}$$

Also, $NS(e_1) = \{u_3, u_4\}$ and $NS(e_2) = \{u_5, u_6\}$. In Step 4 the observation and sensor sets are consolidated as:

$$\mathcal{O} = \hat{\Theta}_{u_2} \cup \hat{\Theta}_{u_3} \cup \hat{\Theta}_{u_5} \cup \hat{\Theta} u_6,\ \text{and}\ \hat{\mathcal{V}} = \{u_3, u_4, u_5, u_6\}.$$

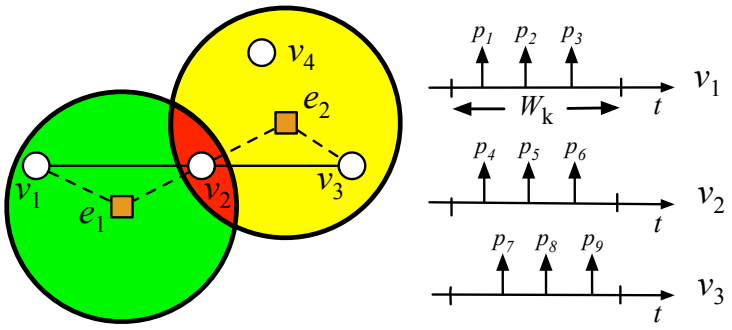


FIGURE 3.1. Tag cleansing using Algorithm 1.

Finally, in Step 5, $\hat{\Theta}_{u_5}$ and u_5 are eliminated from $\mathcal{O}$ and $\hat{\mathcal{V}}$, respectively, since they are duplicates of u_4. This reduces $\hat{\mathcal{V}}$ to $\{u_3, u_4, u_6\}$. Note that sets $\hat{\Theta}_{u_3}$, $\hat{\Theta}_{u_4}$, and $\hat{\Theta}_{u_6}$ form a partition of $\mathcal{O}$. Also in this scenario, e_2 cannot distinguish between v_3's and v_4's transmissions.

3.2.2 Contextual Information Inference

In the second stage, the adversary performs timing analysis on $\mathcal{O}$ to infer contextual information. The adversary takes advantage of the bursty nature of traffic in event-driven WSNs to link traffic streams with the occurrence of physical events. We organize tags in $\mathcal{O}$ into disjoint sets $\mathcal{Y}_1, \mathcal{Y}_2, \ldots$, where $\mathcal{Y}_j$ is attributed to event Ψ_j $(j = 1, 2, \ldots)$. The division depends on the temporal and spatial tag correlation. For instance, consider packets p_1 and p_2, from v and u in $\hat{\mathcal{V}}$. These packets are assigned to the same event if $|t(p_1) - t(p_2)|$ is between certain bounds dependent on the distance between u and v. Details are given in Algorithm 2.

Algorithm 2: Event Filtering

Step 1: Sort $\mathcal{O}$ in ascending order according to tag timestamps.

Step 2: Associate two consecutive packets p_1 and p_2 of $\mathcal{O}$, from sensor labels u and v, with the same set $\mathcal{Y}_j$ if

$$\beta_l(d_{\min}(\hat{\ell}_u, \hat{\ell}_v)) < t(p_2) - t(p_1) < \beta_h(d_{\max}(\hat{\ell}_u, \hat{\ell}_v)).$$

where $\beta_l(d_{\min}(\hat{\ell}_u, \hat{\ell}_v))$ and $\beta_h(d_{\max}(\hat{\ell}_u, \hat{\ell}_v))$ are lower and upper bounds, depending on the minimum and maximum distance between areas $\hat{\ell}_u$ and $\hat{\ell}_v$, respectively.

Step 3: Otherwise, associate p_1 with $\mathcal{Y}_j$ and p_2 with $\mathcal{Y}_{j+1}$.

Step 4: Associate tags in set $\mathcal{Y}_j$ to event Ψ_j.

Thresholds $\beta_l(d_{\min}(\hat{\ell}_u, \hat{\ell}_v))$ and $\beta_h(d_{\max}(\hat{\ell}_u, \hat{\ell}_v))$ reflect bounds on the minimum and maximum delays for relaying packets from $\hat{\ell}_u$ to $\hat{\ell}_v$. The bounds are calculated as a function of the minimum and maximum distance between two areas measured in hops and the per-hop relay delay. For instance, the lower bound is defined as

$$\beta_l(d_{\min}(\hat{\ell}_u, \hat{\ell}_v)) = T_h \left\lfloor \frac{d_{\min}(\hat{\ell}_u, \hat{\ell}_v)}{\gamma} \right\rfloor,$$

where γ is the sensor communication radius and T_h is an estimate of the packet transmission delay between two hops. The lower threshold prevents false association of tags recorded at distant parts of the WSN, due to event concurrence. Similarly, the upper bound is defined as

$$\beta_h(d_{\max}(\hat{\ell}_u, \hat{\ell}_v)) = T_h \left\lfloor \frac{d_{\max}(\hat{\ell}_u, \hat{\ell}_v)}{\gamma} \right\rfloor.$$

The upper bound associates transmissions that occur close in time and in space with the same event. Both thresholds were chosen assuming that the end-to-end path is geometrically linear, which is true for dense network deployments. For other deployments, the thresholds can be adjusted by a multiplying factor.

Algorithm 2 outputs sets of the type $\mathcal{Y}_j = \hat{\Theta}_{u_1} \cup \ldots \cup \hat{\Theta}_{u_y}$ for y labels in $\hat{\mathcal{V}}$. The set $\mathcal{Y}_j$ is used for the inference of event Ψ_j. Moreover, the number of tags in each set $\hat{\Theta}_{u_i} \in \mathcal{Y}_j$ identifies the number of packets transmitted by u_i. Sensors that relay an event report over W should transmit the same number of packets (or approximately the same, if packet retransmissions are present). Thus, the adversary can also obtain an estimate of the number of packets x triggered by event Ψ_j. The accuracy of this

estimation depends on the number of sensors associated with each label. For example, in Figure 3.1, after applying Algorithm 1, the adversary obtains $\hat{\mathcal{V}} = \{u_2, u_3, u_6\}$, and assigns the same label to v_3 and v_4.

The adversary can utilize the size of each $\hat{\Theta}_{u_i} \in \mathcal{Y}_j$ to estimate the number of sensors associated with each label and the number of packets x that report Ψ_λ. The latter is estimated as the size of the smallest transmission set in $\mathcal{Y}_j$. Once the number of sensors per label is found, a topology approximation is obtained by establishing links between labels. Details are given in Algorithm 3.

Algorithm 3: Topology Approximation

Step 1: Let $c_1, c_2, \ldots$ be counters associated with each label v in Y_j, where $c_v = |\hat{\Theta}_v|$. Estimate the number of packets sent by the source to report Ψ_λ as

$$\hat{x} = \min\left(c_1, c_2, \ldots\right).$$

Step 2: Estimate the number of sensors $\hat{n}_v$ associated with label v and counter c_v as,

$$\hat{n}_v = \left\lfloor \frac{c_v}{\hat{x}} \right\rfloor.$$

Step 3: Establish a link (u, v) between labels u and v if

$$d_{\min}(\hat{\ell}_u, \hat{\ell}_v) \leq \gamma.$$

The value of $\hat{n}_v$, as estimated in Step 2, is incorrect if there are inactive sensors in the label area (e.g., v_4 in Figure 3.1), or if a single sensor concurrently relays traffic of more than one event. Finally, based on sets $\mathcal{Y}_j$, the adversary can infer the source and sink location, the routing path to the sink, and the event's occurrence time associated with event Ψ_j, using Algorithm 4.

Algorithm 4: Contextual Information Inference

Step 1: Estimate the event location for Ψ_j as $\hat{\ell}_{v^*}$, where:

$$v^* = \arg\min_{v \in \hat{V}} \{t(p_i) : tag(p_i) \in \mathcal{Y}_j\}.$$

The event location of Ψ_j is estimated to be the area of the tag with the earliest transmission time in $\mathcal{Y}_j$.

Step 2: The occurrence time for $\mathcal{Y}_j$ is estimated as

$$\hat{t}(\mathcal{Y}_j) = \min_{v \in \hat{V}} \{t(p_i) : tag(p_i) \in \mathcal{Y}_j\},$$

i.e., the earliest transmission time recorded in $\mathcal{Y}_j$.

Step 3: The path $p(v, s)$ from the source v to the sink s is estimated as the label sequence of tags in $\mathcal{Y}_j$ (sorted in ascending order, based on transmission time).

Step 4: The sink location is estimated to the location area $\hat{\ell}_s$ of the last label in path $p(v, s)$.

The sink location estimation can be iteratively improved when multiple events are reported to the sink.

3.3 Performance Evaluation

In this section, we applied the traffic analysis algorithms outlined in Section 3.2 and measured the privacy distance achieved for different privacy types (source location, sink location, and event occurrence time). We compared the following representative methods (a) a base method that does not protect contextual information privacy, (b) the geographic routing method proposed in [44], referred to as "STaR", (c) the random walk/probabilistic flooding method proposed in [53], referred to as "phantom flooding", and (d) the traffic normalization method in [50], which normalizes traffic at all sensors via injection of dummy transmissions, referred to as "global norm." Local

adversary countermeasures were evaluated to show the effectiveness of our traffic analysis method. Schemes were implemented in MATLAB 2013.

3.3.1 Simulation Setup

We randomly deployed 250 sensors within a 450m×450m area. We also deployed an eavesdropping network on a square grid to achieve homogeneous coverage of the WSN. We varied the square grid size α and the corresponding number of deployed eavesdropers. The sensor transmission range and the eavesdropper reception range were set to 50m. We abstracted the PHY and MAC layers into a simple per-hop delay model, which consists of a fixed component representing the transmission and propagation delays at the PHY layer, and the contention delay at the MAC layer. This delay was set to 166ms for a packet transmission of 1280 bytes, according to the IEEE 802.15.4 protocol evaluation presented in [31]. No retransmissions due to collisions or impairments of the wireless medium were considered. This simple model was preferred to eliminate the randomness in different system realizations due to contention. Moreover, this model closely matches event-driven networks, which operate under low-contention conditions due to the sparsity of transmissions.

Events were generated at randomly selected locations in the WSNs. The inter-event time followed a uniform distribution in the $[0, 60]$sec interval. Events were reported by the closest active sensor to the event location. The event is reported by transmitting one packet of 1,280 bytes to the sink. Unless otherwise noted, simulation was terminated after the occurrence of ten events.

3.3.2 Source Location Privacy

In Figure 3.2(a), we show the privacy distance for the source location as a function of the grid square size, normalized to the eavesdropping reception range (50m). Confidence intervals of 95% are also shown. We observe that the base, STaR, and phantom

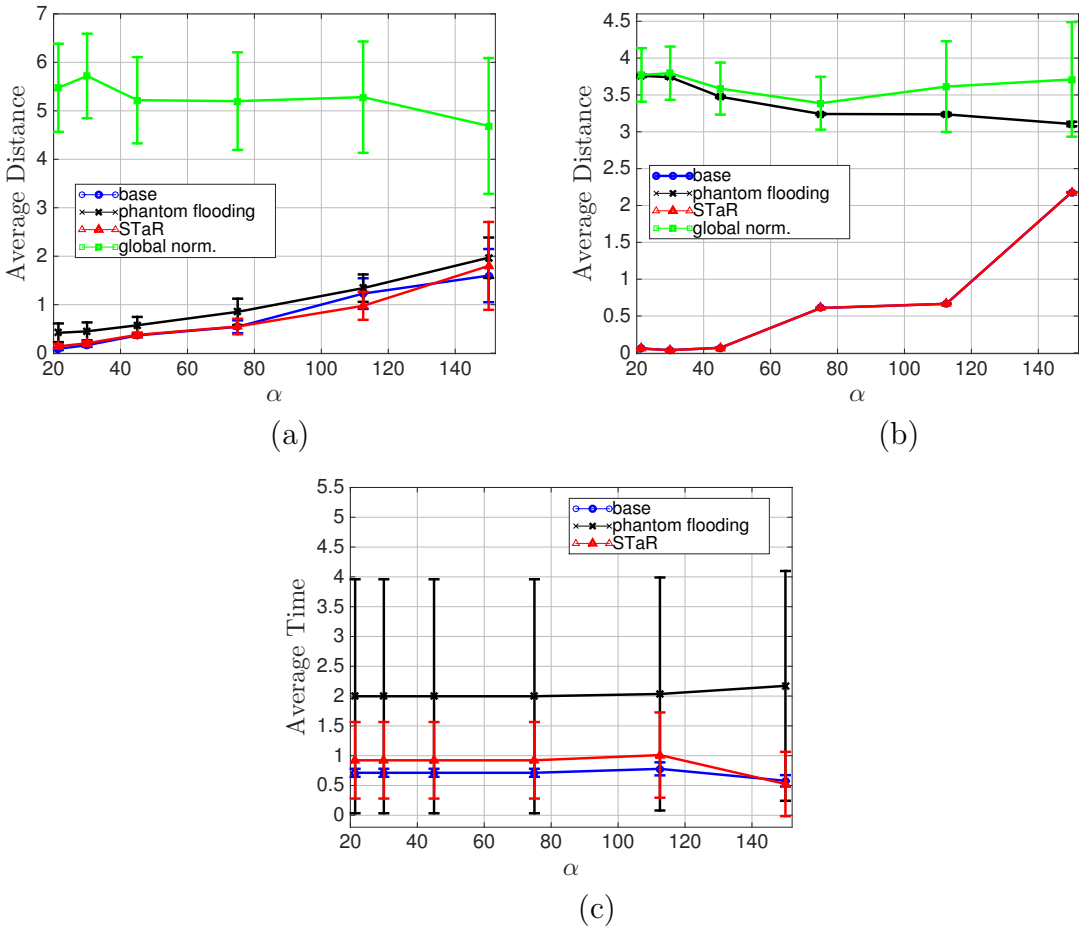


FIGURE 3.2. Privacy distance for: (a) source privacy, (b) sink privacy, and (c) temporal privacy.

flooding schemes offer similar source location privacy. The privacy distance obtains low values (less than 1) for sufficiently dense eavesdropper deployments ($\alpha \leq 80$). On the other hand, the global norm. scheme achieves a privacy distance that is 5-6 times larger than the eavesdropping reception area. The privacy distance remains relatively constant for all α (all values are in the 95% interval). The larger variance observed is due to the difference in the Euclidean distance between the random event locations and all dummy traffic sources.

3.3.3 Sink Location Privacy

In Figure 3.2(b), we show the privacy distance for the sink location. The base and STaR schemes are unable to hide the sink location for sufficiently dense eavesdropper deployments. However, for large α (small number of eavesdroppers), the sink privacy increases considerably. This is because the adversary loses trace of the path from the source to the sink, when some poorly eavesdropped area is traversed. Note that, phantom flooding and global norm. preserve the sink location's privacy. Also, the privacy in phantom flooding is slightly lower, as the path between the source and the intermediate decoy sensor before the application of probabilistic flooding is identifiable.

3.3.4 Temporal Privacy

In Figure 3.2(c), we show the temporal privacy distance for the base, phantom flooding, and STaR schemes. The event occurrence time was estimated using Algorithm 4. We normalized the temporal privacy with respect to the end-to-end delay of reporting the event of interest. For base, STaR, and phantom flooding, the privacy distance takes values between 0.8 and 2, which shows that the accuracy of the adversary's estimations is in the order of the end-to-end delay. Phantom flooding exhibits larger variance compared to the other schemes, due to the probabilistic nature of the flooding operation that causes significant fluctuations in the end-to-end delay. Finally, Figure 3.2(c) does not include results for global norm. because for these methods there is no correlation between the recorded transmission times and the events' occurrence times. Hence, the adversary cannot infer the event occurrence time.

3.3.5 Visual Representation of Privacy

Initially, we visually represented the privacy achieved by the different schemes for adversarial networks of 16 and 250 eavesdropping devices. In this representation,

the sensors true locations are depicted as dots. A shaded area around a sensor v corresponds to the location approximation $\hat{\ell}_v$ for a label v, obtained via the traffic analysis algorithms of Section 3.2. When multiple events are detected, transmission activity of the same event is shaded with the same color. Moreover, the positions of various critical nodes such as the source, the sink, and intermediate nodes are highlighted, if identified by the eavesdropper.

Figure 3.3(a) and Figure 3.3(b) show the inferred sensor activity for the base scheme for 16 and 250 eavesdroppers, respectively. A higher number of eavesdroppers leads to a highly accurate localization of the sources of the two events, and the paths to the sink. The sink location is approximated by the areas of the last two sensors known to transmit for each of the detected events.

Similarly, Figure 3.3(c) and Figure 3.3(d) present the inferred sensor activity for the STaR scheme. Despite the use of an intermediate node as a decoy, the global view of the adversary allows him to pinpoint the source and sink locations, and the path to the sink. Moreover, the two events are clearly distinguishable in the case of the higher number of eavesdroppers.

Figure 3.3(e) and Figure 3.3(f) show the transmission activity inferred for phantom flooding. For this scheme, the adversary pinpoints the source location based on the earliest transmission time. Also, the adversary can construct the path from the source to the intermediate sensor. However, the sink location is protected by the application of the probabilistic flooding.

Finally, Figure 3.3(g) and Figure 3.3(h) show the sensor activity inferred when the global norm. method is applied. This method successfully hides contextual information. In Figure 3.3(e), Figure 3.3(f), Figure 3.3(g), and Figure 3.3(h), we only show results for a single event to preserve visual clarity.

3.4 Summary of Contributions

We proposed a traffic analysis technique for extracting contextual information in event-driven WSNs. Our technique was designed to collectively process the transmission patterns intercepted by a network of eavesdroppers and inferring the event location, its occurrence time, and the sink location. The proposed method is based on the analysis of the packet interception times and the eavesdropper locations. We apply these techniques to WSNs protected by different countermeasures and evaluated the privacy level offered by such countermeasures.

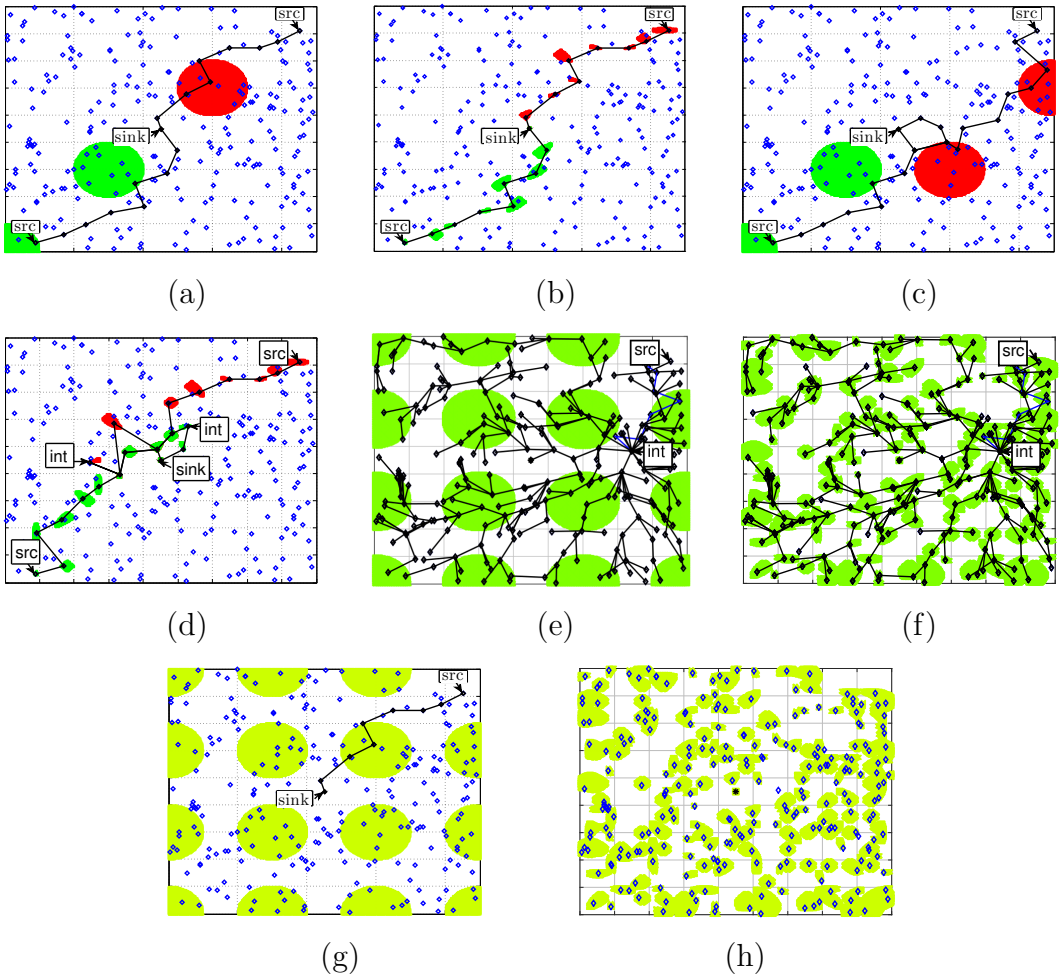


FIGURE 3.3. Inferred sensor activity areas for the following schemes: (a),(b) no protection, (c),(d) STaR, (e),(f) phantom flooding, (g),(h) global norm. for an adversarial network of 16 and 250 eavesdroppers, respectively.

Chapter 4

Time Series Analysis for Breaching Statistical Source Anonymity

State-of-the-art countermeasures against global eavesdroppers conceal traffic associated to real events by injecting dummy packets according to a predefined distribution [5, 50, 65, 78]. For instance, every sensor can be scheduled to transmit on average one packet per minute. To conceal real transmissions, sensors substitute scheduled dummy transmissions with real ones, thus eliminating any information leakage. This solution is demonstrated in Figure 4.1(a). A sensor is scheduled to transmit dummy packets p_i, p_{i+1}, and p_{i+2}, as shown in the top diagram of Figure 4.1(a). The inter-packet times are drawn from distribution P_F. In the lower diagram of Figure 4.1(a), an event E occurs before the transmission of p_{i+1}. Event E is reported by making p_{i+1} a real packet (represented by the dashed line).

However, this theoretically perfect solution has two serious drawbacks. First, dummy transmissions create significant overhead for the energy-constrained sensors. To limit this overhead, dummy transmissions can be scheduled with the same frequency as the occurrence of real events (or even at lower rates). This amplifies the end-to-end delay for reporting an event to the sink. The source as well as every relay sensor have to delay the real transmission until the next scheduled dummy transmission. For time-critical applications this delay could be unacceptable.

Several researchers have devised methods to shorten the end-to-end report delay [50, 65, 78]. The main idea is to accelerate the transmission of a real packet and delay the transmission of the following dummy packet such that the difference observed by the adversary is not statistically significant. Such methods achieve *statistical source anonymity* (SSA). The acceleration method is shown in Fig 4.1(b). The sensor

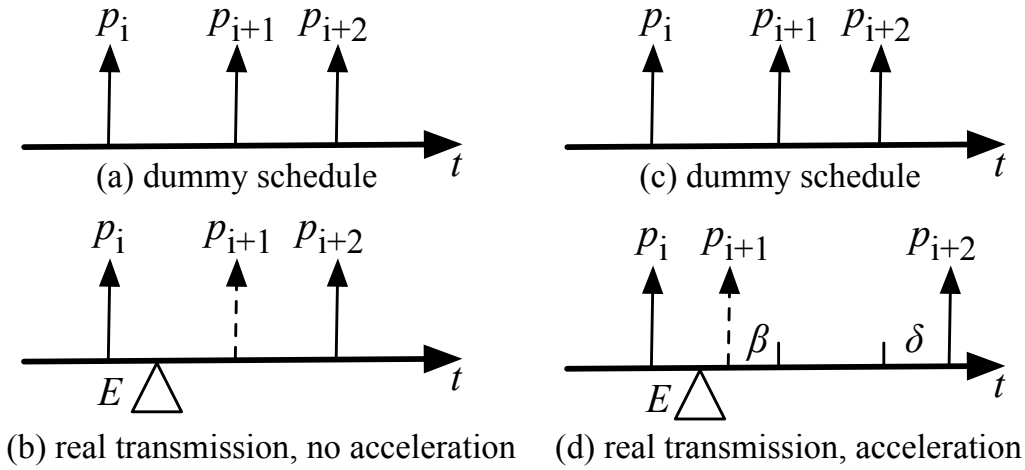


FIGURE 4.1. Tactical WSN deployed to detect the location of enemy entity E.

follows the schedule of dummy transmissions shown in the top diagram. In the bottom diagram, an event occurs after p_i. The sensor accelerates the transmission of p_{i+1}, which is substituted by a real packet, and delays the transmission of p_{i+2} such that the statistical moments observed by the adversary remain the same.

The primary assumption of SSA is that the adversary executes some goodness of fit test to verify if the observed inter-packet times fit the probability distribution used to generate the transmission schedule [19]. As long as the inter-packet times pass the statistical test, real transmissions remain undetectable. However, the modification of inter-packet times can cause data anomalies that are identifiable with methods specifically designed for detecting anomalous data. As an example, let the inter-packet time $\mathbf{x}_{i+1}$ between p_{i+1} and p_{i+2} be drawn from distribution P_F. In the bottom diagram of Figure4.1(b), p_{i+1} is accelerated by β and p_{i+2} is delayed by δ, thus enlarging $\mathbf{x}_{i+1}$ by $\beta+\delta$ and shortening $\mathbf{x}_i$ by β. While the mean of inter-packet times is maintained, $\mathbf{x}_i$ and x_{i+1} can become outliers in the time series representing the inter-packet times.

The adversary can further strengthen its capability of detecting real events by

co-analyzing the transmission patterns of neighboring sensors. Sensors on a path to the sink sequentially accelerate the transmission of a real packet and delay the transmission of the following dummy packet. Observing this phenomenon on a path can yield event detection with higher probability and reduce false alarms.

In our study, we investigate if SSA methods still leak information about the occurrence of real events despite passing goodness of fit tests. The main premise of our investigation is the fact that, the modification of inter-packet times for speeding up real transmissions and recovering the distribution mean can cause detectable outliers. These outliers can be used to distinguish between datasets containing real transmissions and datasets containing only dummy ones. Moreover, in a dataset with real transmissions, the adversary can pinpoint with high confidence the exact time when a real transmission occurred by using the outlier timing.

In our study, we show that SSA can be breached by applying well-known outlier detection methods for time series data. These methods are designed to identify specific data points that break the temporal data continuity by causing sudden changes or unusual patterns. We apply an autoregressive model that does not require a priori knowledge of the distribution used to schedule dummy transmissions. Using this model, the adversary can pinpoint intervals with real events with a probability far greater than a random guess. Moreover, the adversary can use the location of the outliers within the time series to substantially limit the candidate event occurrence times. Prior methods for identifying events are limited to identifying intervals, which can have long durations. We note that our goal is not to develop an outlier detection method that maximizes the probability of event detection. The field of outlier detection is fairly broad, with extensive studies in a large number of application domains. We aim at demonstrating that such methods can breach SSA.

We extend the outlier detection method to time series generated by collectively analyzing the transmissions over multiple hops. The key insight here is that correlated transmission patterns can be observed, when each of the relay sensors accelerates the

transmission of a relayed real packet. The appearance of short-long transmission patterns in sequence amplifies the outliers in the "collective" time series and reduces false alarms.

We show that existing countermeasures that call for the insertion of fake events which mimic real events are not effective when outlier detection is applied over multiple hops. This is because sensors independently insert fake events without coordinating over multiple hops. Therefore, these fake events do not appear is if the occur in sequence, as is the case during the relay of real packets.

The feasibility of improving source anonymity in the light of outlier detection analysis remains an open and challenging problem. It is particularly interesting to investigate by how much real transmissions can be accelerated before real sources become detectable. Our work demonstrates that there is a clear tradeoff between end-to-end report delay and source anonymity. This exposition is left as future work.

4.1 Preliminaries

The adversary is assumed to simultaneously monitor all transmissions of the WSN, which are collectively analyzed at a fusion center. For each intercepted packet, he can determine the origin and time of transmission. Specifically, for each sensor $v \in \mathcal{V}$, the adversary records a transmission set

$$\Theta_v = \{\mathbf{x}_i(v) : i = 1, 2, \ldots\},$$

which records the time $\mathbf{x}_i(v)$ between the i^{th} and $i{+}1^{st}$ transmission from v. The times $\mathbf{x}_i(v)$ represent random variables which are drawn from a known distribution P_F when the i^th and $i+1^st$ transmissions are fake or take values according to the scheme used to speed up the transmission of real events, when one of the transmissions corresponds to a real event. The distribution P_F is known to the adversary. By analyzing sets $\Theta_v, \forall v \in \mathcal{V}$, the adversary attempts to determine (a) the location where events occur

and (b) the occurrence time (or a time range) of events. The analysis techniques used by the adversary cannot break the security of cryptographic algorithms used to conceal the contents of packets.

4.2 SSA Model

To evaluate the anonymity of the WSN, we adopt the SSA framework proposed in [19]. In this framework, the authors introduce the notion of interval indistinguishability which is defined as follows:

Definition 7 (Interval indistinguishability)**.** *. Let I_F denote a time interval without any real event transmission (called the "fake interval"), and I_R denote a time interval with real event transmissions (called the "real interval). The two time intervals are said to be statistically indistinguishable if the distributions of inter transmission times during these two intervals cannot be distinguished with significant confidence.*

Interval indistinguishability is a strictly stronger notion than event indistinguishability. The latter pertains to the ability of the adversary to distinguish between real and fake transmissions (as opposed to intervals) by means of statistical tests. As stated in [19], interval indistinguishability implies event indistinguishability, but not conversely. However, we note that both notions are useful in privacy analysis. This is because breaching interval indistinguishability yields inferior "information quality" to the adversary. As an example, consider a statistical methods that can identify the occurrence of an event within an interval of several days. Though this method breaches interval indistinguishability, the WSN can still be considered secure due to the length of the interval. On the other hand, identifying the transmissions corresponding to a real event can prove detrimental to the WSN.

To quantify the anonymity of a WSN against an adversarial strategy σ, we use the anonymity metric proposed in [19], defined as follows.

Definition 8 (Interval Anonymity)**.**

$$\Lambda_\sigma := 1 - 2(P_d^I - 0.5),$$

where P_d^I, denotes the probability of identifying the I_R by applying σ, when I_R and I_F are presented to the adversary.

We emphasize that it is reasonable to assume that the adversary has access to I_F (i.e., it knows distribution P_F from which timings in I_F are drawn). This is because the adversary can always observe sensors over many intervals and categorize them into two groups. One group is likely the fake intervals and the other group is the real ones. In our approach, intervals with the fewest outliers correspond to fake intervals.

Besides Λ_σ, we will compute standard metrics in hypothesis testing to evaluate the effectiveness of outlier analysis on breaching SSA.

4.3 Outlier Detection For Breaching SSA

In this section, we develop an adversarial strategy σ for breaching SSA. The strategy σ relies on the detection of outliers caused by the acceleration of real transmissions and the delay of the following dummy transmissions (short-long patterns). The corresponding data points, while not skewing the mean when combined (or any other moment), are individually anomalous and unfit with the rest of the time series. The key insight to using outlier analysis is that SSA methods allow for any inter-packet time values as long as the goodness of fit test is satisfied. For instance, the inter-packet time corresponding to a real transmission could be as small as zero, while the value used for mean recovery may be considerably large. In both cases, the inter-packet times can represent outliers on the data observed by the adversary.

A short-long pattern generated due to a real transmission is of the form $(\mathbf{x}_j - \beta, \mathbf{x}_{j+1} + \delta)$, where $\mathbf{x}_j$ and $\mathbf{x}_{j+1}$ are i.i.d random variables drawn from P_F, and β and δ are positive values used to accelerate the real transmission and to recover the mean.

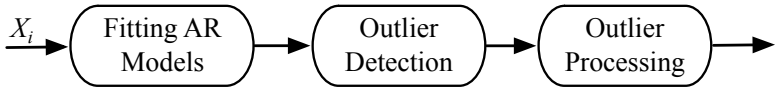


FIGURE 4.2. The three stages of the autoregressive time series analysis.

Although the pair $(\mathbf{x}_j - \beta, \mathbf{x}_{j+1} + \delta)$ is designed to yield the same mean (or other moments) as $(\mathbf{x}_j, \mathbf{x}_{j+1})$ given all other values in X_i, it is not distributed according to P_F.. The effects of such a design are shown in the analysis presented in [19]. A time series that contains several real transmissions has a higher correlation factor compared with a series which contains only dummy transmissions (when correlated with respect to a reference time series of consecutive short-long patterns). In our analysis, we show that outlier detection can be used to identify the intervals that contain real transmissions. Moreover, the location of those outliers within the time series serves as a good indicator for pinpointing the occurrence event time.

4.3.1 Time Series Analysis

Outlier detection in time series has been a subject of research for several decades. A large number of techniques have been specialized for different application domains with varying performance and complexity degrees [7, 34]. For our purposes, we adopt a statistical technique based on *autoregressive (AR) models* [35]. Note that are goal is not to develop the best technique, but to demonstrate that outlier analysis can breach schemes that satisfy SSA. The advantage of using AR models is that they do not require a priori knowledge of distribution P_F, used to generate the schedule of the dummy transmissions.

An AR technique consists of the three stages shown in Fig. 4.2. In the first stage, a regression model is fitted to the time series. This model is a representation of the data in the absence of outliers. In the second stage, the differences between the time series and the regression model are used to detect outliers. In the final stage, the

outliers are processed so that they do not affect the analysis of future points in the time series. We now describe each stage in detail.

4.3.2 Fitting Autoregressive Modeling

To generate the AR model, the adversary must be in possession of a time series

$$X_i = \{x_i - k + 1, x_i - k + 2, \ldots, x_i\}$$

of k consecutive data points (inter-packet times) that correspond to a fake interval I_F. Series X_i is a sample realization of an invite series of inter-packet times generated according to P_F, which is unknown to the adversary. The AR model takes as input X_i and outputs model parameters that are used to predict future series values. Suppose the following linear AR model of order m [35]:

$$x_{i+1} = \phi_1 x_i + \phi_2 x_{i-1} + \ldots + \phi_m x_{i-m+1} + e_{i+1},$$

where $\phi_1, \phi_2, \ldots \phi_m$ are the autoregressive parameters, and e_{i+1} is the accumulated error. The error is assumed to be i.i.d (for all i). If e_i is distributed according to a normal distribution, the AR process is said to be Gaussian, for other cases the central limit theorem indicates that e_{i+1} is approximately normally distributed. Now let,

$$z_{i+1}^T = (x_i, x_{i-1}, \ldots, x_{i-m+1}),$$

$$\phi^T = (\phi_1, \phi_2, \ldots, \phi_m).$$

The AR(m) model can be expressed as,

$$x_{i+1} = z_{i+1}^T \phi + e_{i+1}.$$

In a set of k points $x_1, \ldots, x_k$, the equations of the AR(m) models become,

$$X_k = \Gamma \phi + e,$$

where, $X_k = (x_1, \ldots, x_k)^T$ and $e^T = (e_1, e_2, \ldots, e_m)$, and

$$\Gamma = \begin{bmatrix} x_0 & x_{-1} & \ldots & x_{-m+1} \\ x_1 & x_0 & \ldots & x_{-m+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \ldots & x_{k-m} \end{bmatrix},$$

where any x_j for $j < 0$ is assumed to be zero. The least square (LS) estimate $\hat{\phi}$ of ϕ is given by,

$$\hat{\phi} = (\Gamma^T \Gamma)^{-1} \Gamma^T X_k.$$

The fitted values of the model are defined by,

$$\hat{X_k} = \Gamma \hat{\phi},$$

and the least square fitted residuals (LSR) $r = (r_1, r_2, \ldots, r_n)^T$ are,

$$r = X_k - \hat{X_k}.$$

Finally, the least square (LS) estimate s^2 of the variance of e is estimated as,

$$\hat{s}^2 = \frac{r^T r}{k}.$$

We use residuals and the LS approximation of the variance in the detection of outliers in the time series.

4.3.3 Outlier detection

After generating an AR model, the adversary uses the values of $\hat{\phi}$ to predict the next inter-packet time value. That is, assuming x_i is the last time that the adversary obtained, the value of $\hat{x}_{i+1}$ is given by,

$$\hat{x}_{i+1} = \hat{\phi}_1 x_i + \hat{\phi}_2 x_{i-1} + \ldots + \hat{\phi}_m x_{i-m+1}.$$

Once the actual value of the x_{i+1} is obtained, the adversary check if x_{i+1} is within the prediction interval (PI), defined as follows,

$$\hat{x}_{i+1} \pm t_{\alpha/2,k-1} \times s\sqrt{1+\frac{1}{k}},$$

where $t_{\alpha/2,k-1}$ is the $100(1-\alpha)$ percentile of a Student's $t-$distribution with $(k-1)$ degrees of freedom (k is the size of X_i), and s is the LS approximation of the standard deviation of the residuals (obtained from the AR model). If x_{i+1} does not fall within the bounds of the PI, x_{i+1} is classified as an outlier.

4.3.4 Outlier processing

Finally, when an inter-packet time x_{i+1} is identified as an outlier, the adversary processes it in a way that the detection of future outliers is consistent. For instance, if the adversary uses outlier x_{i+1} to estimate $\hat{x}_{i+2}$, the next PI value (used to determine whether x_{i+2} is an outlier) is compromised. To avoid this problem, the adversary flags of x_{i+1}, and uses $\hat{x}_{i+1}$ in the calculation of $\hat{x}_{i+2}, \hat{x}_{i+3}, \ldots, \hat{x}_{i+m+1}$. That is, the value of $\hat{x}_{i+2}$ is calculated as follows,

$$\hat{x}_{i+2} = \hat{\phi}_1\hat{x}_{i+1} + \hat{\phi}_2 x_i + \ldots + \hat{\phi}_m x_{i-m+2}.$$

4.4 Interval and Event Indistinguishability

In this section, we outline the steps for breaching interval and event indistinguishability by applying the outlier analysis method presented in Section 4.3. First, the adversary uses the AR model generated based on the time series of a fake interval I_F to distinguish if an interval I_R contains real transmissions. If I_R is identified to contain real transmissions, their occurrence times are associated with the timing locations of the outliers in I_R.

4.4.1 Interval Indistinguishability

To determine if interval indistinguishability can be breached via outlier analysis, we examine if for two intervals I_F and I_R, the adversary can identify I_R as the real one with probability higher than a random guess. To do so, the adversary applies outlier detection on both I_F and I_R and counts the number of outliers found in each interval. As I_R includes inter-packet times following the short-long timing pattern, this interval is likely to include more outliers than I_F. Let O_{I_R} and O_{I_F} be the sets of outliers found at I_R and I_F, respectively. The interval type for I_R is decided according to the following rule:

$$I_R = \begin{cases} \text{real}, & \text{if } |O_{I_R}| > \psi |O_{I_F}| \\ \text{fake}, & \text{otherwise.} \end{cases} \tag{4.1}$$

In (4.1), $\psi \geq 1$ denotes the *outlier expansion factor*. This factor measures the fraction of additional outliers expected in a real interval relative to a fake interval and is simply defined as

$$\psi = \frac{|O_{I_R}|}{|O_{I_F}|}.$$

The value of ψ is statistically evaluated by comparing many instances of I_F and I_R.

4.4.2 Event Distinguishability

The additional outliers found in real intervals are likely caused by the short inter-packet times used to accelerate real transmissions, or the long inter-packet times used to recover the mean. Based on this observation, we can exploit the outlier location within I_R to pinpoint the occurrence time of real events. For real intervals as identified by the method described in Section 4.4.1, we associate *all* outlier locations (i.e., set O_{I_R}) with real transmissions. This is done as follows.

Let the time series X_i corresponding to I_R be

$$X_i = \{x_i - k + 1, x_i - k + 2, \ldots, x_i\}.$$

An inter-packet time $x_j \in X_i$ is associated with the transmission times of packets p_{j-1} and p_j, i.e., $\mathbf{x}_j = t(p_j) - t(p_{j-1})$. If x_j is identified as an outlier in X_i, the real event occurrence time is determined by comparing x_i to the mean μ of P_F. Specifically, the event occurrence time t_e is set to

$$t_e = \begin{cases} t(p_j), & \text{if } x_j < \mu_F \\ t(p_{lj+1}), & \text{otherwise.} \end{cases} \tag{4.2}$$

Essentially, if the outlier is caused by a short inter-packet time relative to the mean of P_F (packet acceleration), the later transmission is mapped to a real transmission. If the outlier is caused due to a long inter-transmission time (mean recovery), the earlier transmission is selected to be the real transmission. For the calculation of the mean μ_F of P_F, all observations in X_i are averaged after the outliers have been removed, as dictated by the outlier processing stage (Section 4.3.4).

4.5 Exploiting Multihop Paths

So far, we have attempted to detect outliers by analyzing the traffic patterns of individual sensors. This strategy is inline with the statistical analysis assumed in state-of-the-art SSA methods. However, the adversary could attempt to correlate transmissions over multihop paths. The key advantage of collectively analyzing observations across a path is the possible correlation in the transmission timings, when real events are relayed over that path. When transmissions are fake, receiving sensors discard them and continue their dummy transmission schedule. On the other hand, when a transmission is real, every sensor along the path to the sink will attempt to accelerate it. This generates a sequence of short-long transmission patterns along the path.

As an example, consider the three-sensor path in Fig. 4.3. Suppose that an event E in the vicinity of v is reported by the transmission of packet p_{i+1}. The packet p_{i+1} is rushed by v, causing a short inter-packet time x_i. The following packet p_{i+2} is delayed

to recover the mean, causing a long inter packet time x_{i+1}. The real packet p_{i+1} is received by u, which is equivalent to the occurrence of an event in u's neighborhood. Similarly to v, sensor u accelerates the transmission of q_{j+1}, causing a short inter-packet time y_j and delays the transmission of q_{j+2}, causing a long inter-packet time y_{j+1}. If the respective inter-packet times are added, the adversary obtains a new time series containing

$$\{x_i + y_j, x_{i+1} + y_{j+1}, \ldots\}$$

If the individual inter-packet times were generated according to P_F, the values of the new time series would follow the distribution obtained by the summation of i.i.d random variables. This distribution is given by the convolution of the PMFs of the original distributions. However, values $x_i, x_{i+1}, y_j, y_{j+1}$ are highly dependent. Their summation is expected to far deviate from the sum of i.i.d. random variables. We exploit this key insight and apply outlier detection on time series generated by the addition of inter-packet times for sensors that belong to paths towards the sink. Let the time series containing the compounded inter-packet times across an interval I for a path $\gamma(v, s)$ be denoted by

$$X(\gamma(v, s)) = \{\delta_1, \delta_2, \ldots, \delta_k\}.$$

The process of creating $X(\gamma(v, s))$ is described in Algorithm 1 (the same process is followed for generating time series for both fake and real intervals).

Algorithm 1: Time series generation for a path $\gamma(v, s)$

Input: Path $\gamma(v, s)$, time series $X(u)$, $\forall uin\gamma(v, s)$, transmission times $T(u)$, $\forall u \in \gamma(v, s)$,

Output: $X(\gamma(v, s))$.

Step 1: For each inter-packet time $x_i(v) \in X(v)$, set $\delta_i = x_i(v)$. Identify the transmission time $t_{i+1}(v) \in T(v)$ of packet $p_{i+1}(v)$, where $x_i(v) = t_{i+1}(v) - t_i(v)$.

Step 2: Let u be the first sensor downstream of v. Let also $p_j(u)$ be the first packet transmission of $u \in \gamma(v, s)$, after time $t_{i+1}(v)$. Update $\delta_i = \delta_i + x_j(u)$. Repeat Step 2

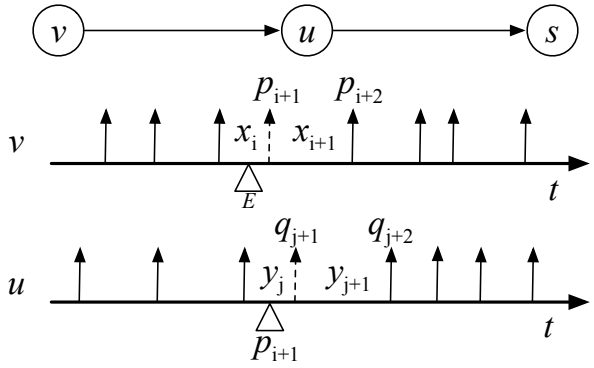


FIGURE 4.3. Outlier detection in multi-hop transmissions.

for all sensors in $\gamma(v, s)$ until s is reached.

Step 3: Repeat Steps 1 and 2 for all $x_i(v) \in X(v)$.

To demonstrate the application of Algorithm 1, consider the 3-hop path shown in Fig. 4.3. The adversary uses $X(v)$, $X(u)$, $T(v)$, and $T(u)$ to obtain $X(\gamma(v, s))$. Consider the calculation of δ_i. In Step 1, the adversary initializes $\delta_i = x_i(v)$. It also identifies $t_{i+1}(v)$ as the transmission time of packet p_{i+1}. In Step 2, the adversary identifies the transmission of $q_{j+1}(u)$ as the first packet transmitted by u, after time $t_{i+1}(v)$. It then updates $\delta_i = x_i + y_j$. In Step 3, the adversary repeats the Steps 1 and 2 for inter-packet time x_{i+1}.

Using Algorithm 1, the adversary constructs the two time series that are used in outlier detection, as it is outlined in Section 4.3. The adversary then applies the interval distinguishability and event distinguishability criteria stated in Section 4.4 to determine if the real interval contains real transmissions.

Note that the analysis presented for the multihop scenario makes two implicit assumptions. The first assumption is that a fake interval is available to the adversary. This interval does not contain real transmissions by any of the sensors in $\gamma(v, s)$. Access to such a fake interval is obtained by observing the network for long periods of time and applying outlier detection (the fake intervals are those with the fewest

outliers). The second assumption pertains to the knowledge of $\gamma(v, s)$. The path to the sink for a given node v can be known a priori, if the algorithm used for routing is deterministic (e.g., shortest path algorithm), or can be discovered on demand using outlier detection itself. In the latter case, the adversary can perform a search in the one-hop neighbor of the origin node v and use the multihop path outlier analysis to find the next hop that yields the highest number of outliers. He can then continue the same process to expand the path towards the sink. The details of such a discovery process are left for future work. In the present work, we focus on whether analysis over a known multihop path can improve the interval and event distinguishability.

4.6 Evaluation

In this section, we evaluate the effectiveness of outlier detection on breaching SSA anonymity. We simulate three experiments. In the first experiment, we evaluate the interval and event indistinguishability when the adversary analyzes the communication patterns of individual sensors. In the second experiment, we extend our analysis to observations collected from sensors that belong to multhop paths. In the last experiment, we consider the insertion of fake events during dummy intervals, as suggested in [19].

4.6.1 Simulation Setup and Metrics

Simulation setup: For the individual sensor scenario, we generated time series of inter-packet times for dummy and real intervals. For dummy intervals, the inter-packet times were drawn from an exponential distribution of rate $\lambda_F = 20$. For real intervals, real events occurred according to an an exponential distribution with rate λ_R, which was varied. The inter-packet times associated with real transmissions were generated according to the FitProbRate scheme [65]. Following the experiments

performed by the authors in [65], we used the Anderson-Darling goodness of fit test with significance $\alpha = 0.05$. For purposes of mean recovery, we set $\epsilon = 0.1$.

For two intervals I_F and I_R, the adversary applied outlier detection using an AR model of order $m = 3$. The value of ψ in criterion (4.1) was set to one. Testing of higher order AR models did not yield notable performance gains. To train the AR model we used time series with 200 observations. Each experiment trial was terminated after the occurrence of 50 real events. The results of our simulations were obtained based on $1,000$ trials.

Metrics: We used the following metrics to evaluate the performance of outlier analysis in detecting real events.

- Interval anonymity, as defined in Definition 8.

$$\Lambda := 1 - 2(P_d^I - 0.5),$$

  where P_d^I, denotes the probability of detecting real intervals, when I_R and I_F are presented to the adversary. This is given by

$$P_d^I := \frac{\eta_d^I}{\eta},$$

  where η_d^I denotes the number of real intervals identified by outlier analysis, when the adversary is presented with η pairs of real and fake intervals.

- Probability of false alarm for interval indistinguishability

$$P_{fa}^I := \frac{\eta_{fa}^I}{\eta},$$

  where η_{fa}^I denotes the number of fake intervals miscategorized as real, when the adversary is presented with η pairs of fake intervals.

- Probability of event detection

$$P_d^E := \frac{\eta_d^E}{\eta_E},$$

where η_d^E denotes events the number of detected real events and η_E is the total number of events.

- Probability of false alarm for event indistinguishability

$$P_{fa}^E := \frac{\eta_{fa}^E}{\eta_d^E}.$$

where η_{fa}^E is the total number of falsely detected events and η_d^E is the total number of detected events.

We also report the outlier expansion factor (ψ) for real intervals.

4.6.2 Individual Sensor Analysis

In the first set of experiments, we evaluated the breach of SSA when outlier analysis is applied to traffic patterns of individual sensors. Figure 4.4(a) shows the interval anonymity (Λ) as a function of the rate of real events (λ_R). Recall that $\Lambda = 1$ corresponds to a probability of real interval detection $P_d^I = 0.5$, which is equivalent to a random guess (perfect anonymity). Any $\Lambda < 1$ indicates that the adversary can distinguish real from fake intervals with higher probability than a random guess. In Fig. 4.4(c), we present the probability of false alarm for interval indistinguishability as a function of λ_R. From Fig. 4.4(a) and Fig. 4.4(b), we observe that Λ becomes as low as 0.2 and the probability of false positive 0.08 when $\lambda_R = 10$, respectively. When real transmissions are relatively sparse compared to fake transmissions, the former can be accelerated significantly without violating the A-D goodness of fit test. However, this causes the occurrence of more outliers that can be used to distinguish real from fake intervals. On the other hand, for $\lambda_R = 30$, the interval anonymity increases to 0.7, and the probability of false positive to 0.32. This is because most of the occurring transmissions correspond to real events, leaving small opportunities for the acceleration of real transmissions. The anonymity of real intervals significantly decreases because real transmissions simply replace the scheduled dummy ones.

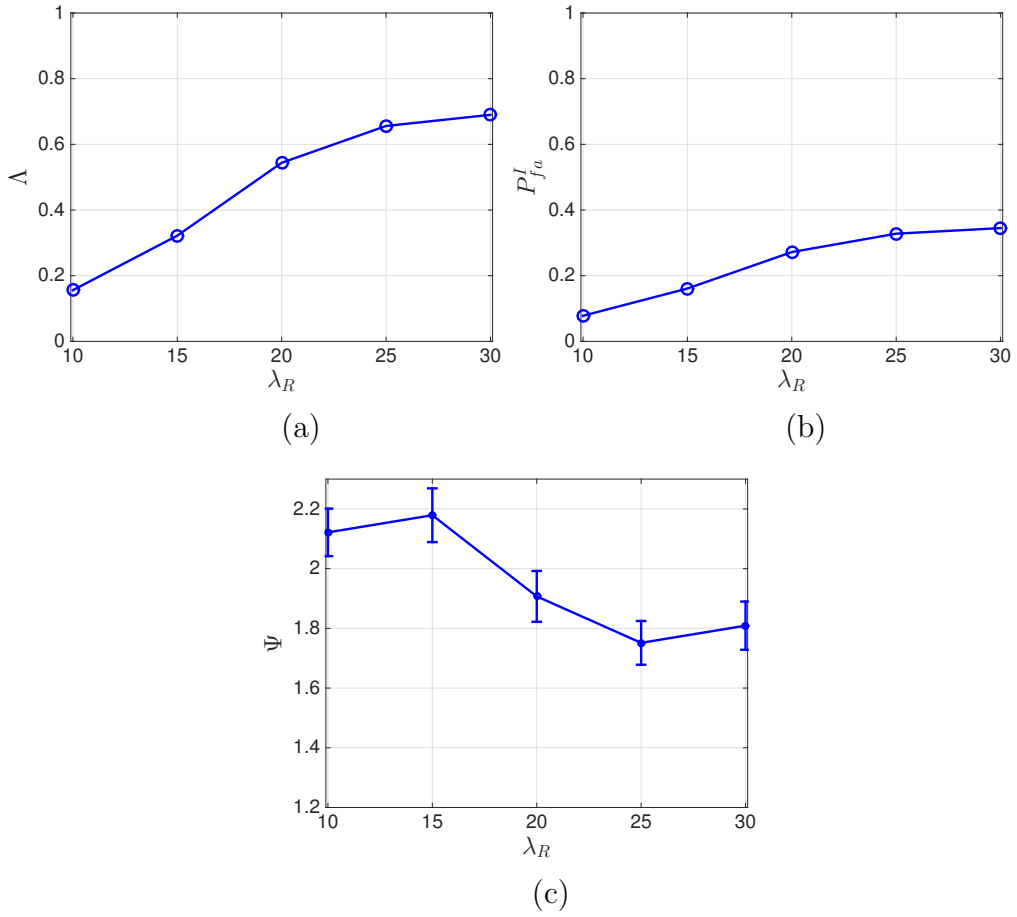


FIGURE 4.4. (a), (d) interval anonymity Λ (b), (e) probability of false alarm for interval indistinguishability, and (c), (f) outlier expansion factor ψ,for single-node and multi-hop scenario, respectively.

In Fig. 4.4(c), we show the outlier expansion factor (ψ) as a function of λ_R. Confidence intervals of 95% are also shown. We observe that ψ reduces from 2.1 when $\lambda_R = 10$ to 1.8 when $\lambda_R = 30$. The range of ψ justifies setting $\psi = 1$ for the criterion in (4.1). The decrease of ψ with λ_R justifies the corresponding increase in interval anonymity, as fewer real intervals contain more outliers than compared to the outliers in fake intervals. Finally, we note that for $\lambda_R = \lambda_F = 20$ (the dummy packet rate matching the event rate), $\Lambda = 0.57$ which corresponds to a real event detection probability of 73% (the same value as the one obtained using correlation

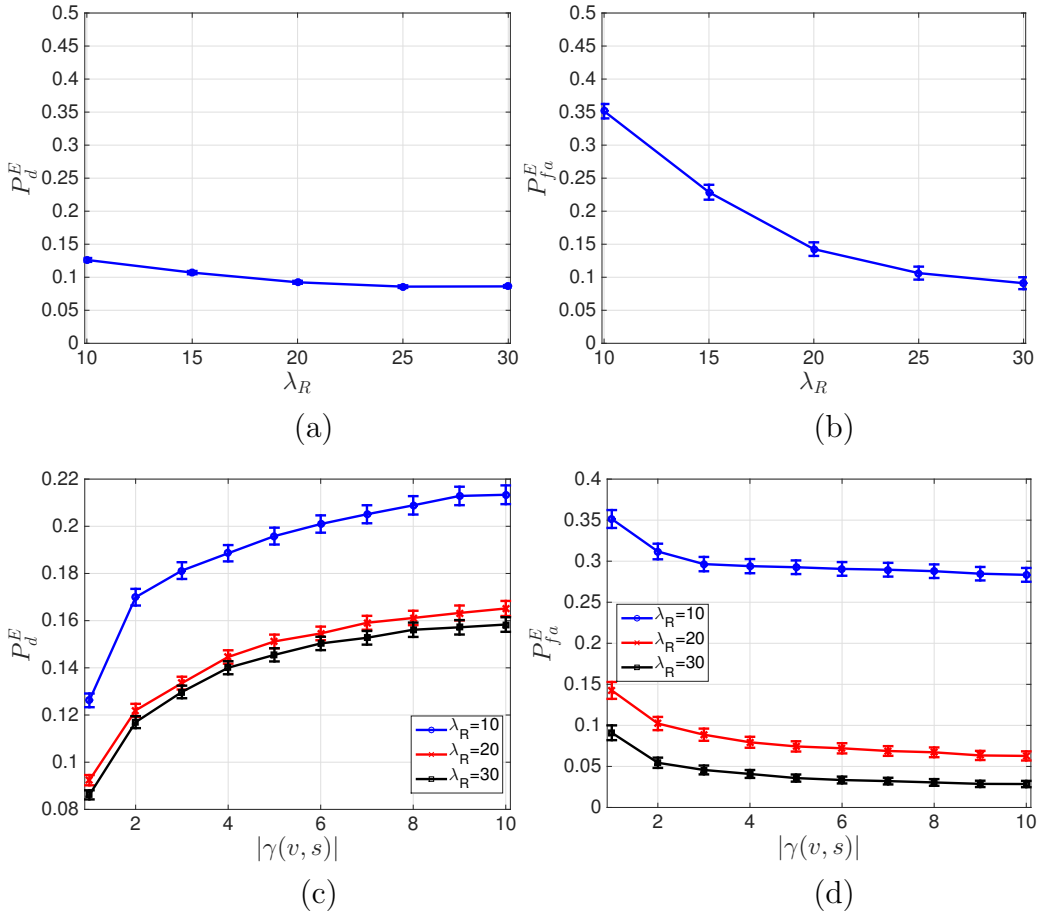


FIGURE 4.5. (a), (c) Probability of event detection, and (b), (d) probability of false alarm for event indistinguishability, for single-node and multi-hop scenario, respectively.

analysis in [19]). This probability is far greater than a random guess, indicating that SSA is breached via outlier analysis.

In Fig. 4.5(a), we show the probability of event detection (P_d^E) as a function of λ_R. The value of P_d^E decreases from 0.13 for λ_R to 0.07 for λ_R. Although by comparison to expected performance values in detection theory, an average P_d^E of 0.1 seems quite poor, the interpretation of P_d^E in our setup is quite different. It demonstrates the fraction of real events for which the adversary can *exactly pinpoint the event occurrence time*. This is a fairly strong inference, given the application of

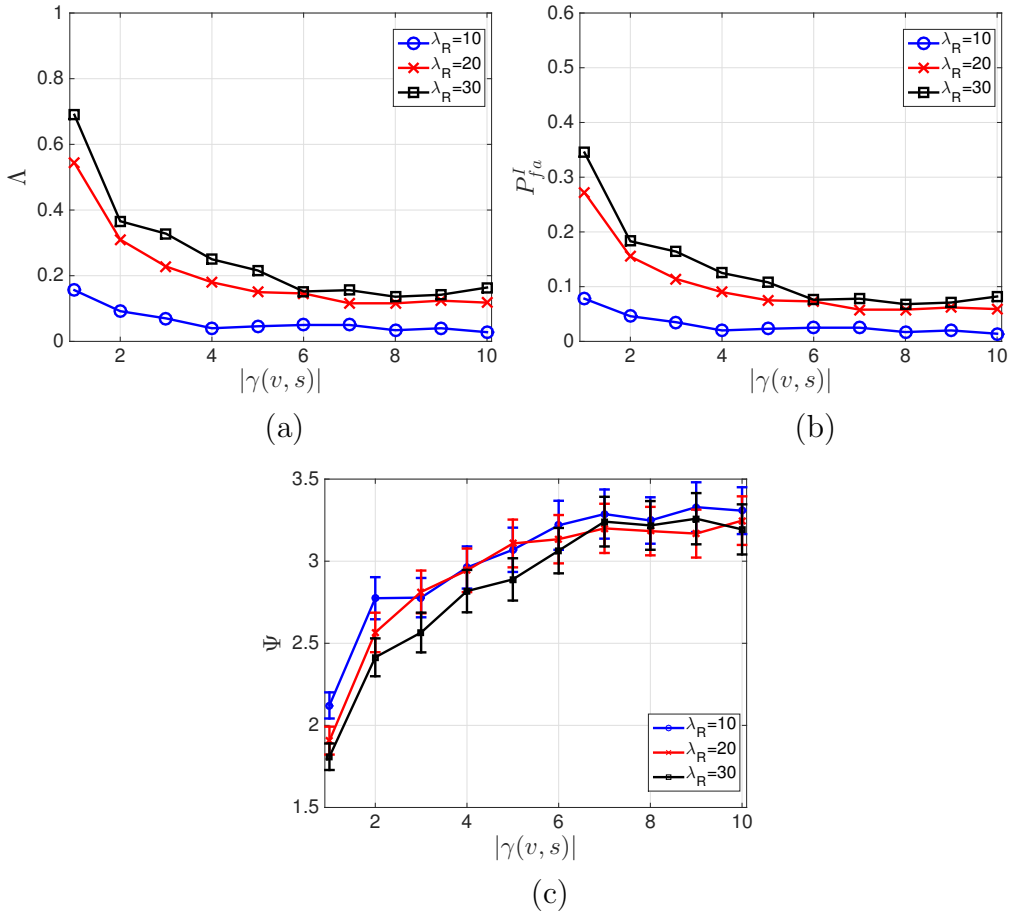


FIGURE 4.6. (a) interval anonymity Λ (b) probability of false alarm for interval indistinguishability, and (c) outlier expansion factor ψ for multi-hop scenario, respectively.

SSA, specifically designed to prevent event distinguishability.

Finally, in Fig. 4.5(b) we present the probability of false alarm for event indistinguishability as a function of λ_R. The value of P_{fa}^E decreases from 0.35 for $\lambda_R = 10$ to 0.07 for $\lambda_R = 30$. The false alarm probability drops significantly in high λ_R because most transmissions causing outliers are real transmissions (the majority of transmissions within an interval correspond to real transmissions).

4.6.3 Multihop Path Analysis

In this set of experiments, we evaluated the breach of SSA when outlier detection is applied to a set of observations collected over a multihop path. Sensors along the path interpreted the reception of a real transmission as an occurrence of a real event and accelerated the relay operation without violating the A-D goodness of fit test. We varied the path length and computed the metrics of interest.

In Fig 4.6(a), we show Λ as a function of the path length from the first sensor that detected the event (v) to the sink s. The path length is denoted by $|\gamma(v,s)|$. We observe that Λ decreases with $|\gamma(v,s)|$ for all values of λ_R. This verifies that the collective analysis of observations over multiple hops further reduces the effectiveness of SSA methods applied to individual sensors. The aggregation of inter-packet times across a path create more outliers that allow the clear distinction between real and fake intervals. A similar effect is captured by the probability of false positive for interval indistinguishability. In Fig 4.6(b), we observed that $P_{fa}^{I} < 0.1$ for values of $\lambda_R \geq 5$. This is also verified by the increase in the outlier expansion factor, as shown in Fig. 4.6(c). Of particular interest is the case of $\lambda_R = 30$, for which single-hop outlier analysis yielded a $\Lambda = 0.97$. The interval anonymity drops to 0.38 for a path of length two and further decreases to less than 0.2 for paths longer than six hops. This indicates that multihop analysis breaches interval indistinguishability even when the latter property is retained on a sensor by sensor basis.

In Fig. 4.5(c), we show the probability of real event detection as a function of the path length. The multihop analysis improves the adversary's ability to pinpoint the exact time that real events occurred. Furthermore, the probability of false alarm for event indistinguishability shown in Fig. 4.5(d) drops with $|\gamma(v,s)|$. This is because fewer outliers are caused by the summation of random i.i.d. values.

4.7 Summary of Contributions

We proposed traffic analysis techniques to breach SSA. Our methods are based on outlier detection in time series. We showed that the number of outliers in a time series that contains real transmissions is greater than those found in a time series corresponding to a fake interval. We demonstrated that the adversary is able to pinpoint the transmission time of event reports. Moreover, our analysis showed that the adversary may take advantage of its knowledge of the network topology to increase the probability of detection.

Chapter 5

Energy-efficient Traffic Normalization for SSA

We first address the problem of contextual information privacy from a resource point of view. As sensors are typically constrained by their limited battery life, we attempt to conserve energy resources by reducing the number of sensors that transmit dummy packets for traffic normalization. Our method differs from prior-art in that we address the tradeoff between the communication and real-packet delay overhead by limiting the set of sensor required to inject dummy traffic. The set of dummy packets sources is independently chosen from the sensor deployment density. To do so, we map the problem of selecting the set of dummy packets sources to the problem of finding a minimum connected dominating set (MCDS) that covers the WSN deployment area. The MCDS guarantees that every sensor is at most one hop away from an MCDS sensor and that every eavesdropper observes random traffic patterns.

To prevent the global adversary from identifying real transmissions, we propose rate control techniques that regulate the transmission of dummy packets for nodes in the CDS. In the case of real packets, nodes in the CDS transmit them by simply replacing the next scheduled dummy packet by a real one. However, for nodes that do not belong to the CDS, real packets are transmitted at a rate that has statistically insignificant impact on the traffic patterns observed by eavesdroppers located in the vicinity of the real source.

We consider the cases when eavesdroppers perform traffic analysis independently and when network traffic is collectively analyzed by a central server. Independent analysis yields faster detection and considerably lower communication overhead. Collusion yields higher accuracy in the information extracted. Moreover, while previous

methods focus on protecting only source location, our methods are designed to preserve the privacy of the source location, event location, event occurrence time, and the sink location.

5.1 Preliminaries

Let P_F denote the probability distribution of a communication attribute observed by eavesdropper a in the absence of real traffic and P_X denote the probability distribution of the same attribute when real traffic is transmitted by any sensor in the vicinity of a. The adversary performs a statistical test on the distributions P_F and P_X to determine if the two distributions are statistically different. Distribution P_F is assumed to be empirically known based on long-term observations. Real traffic is indistinguishable from bogus traffic if Definition 9 is satisfied.

Definition 9. *(β, ϵ)-unobservability–Given a candidate set of events E, an observation set $\mathcal{O}$ ($E \subseteq \mathcal{O}$), and probability distributions P_F and P_X with parameters (moments) $\theta_1, \ldots, \theta_\zeta$, distributions P_F and P_X are indistinguishable under the conditions:*

(a) $f(P_F, P_X) \leq g(\beta)$,

(b) $(1-\epsilon)\theta_i \leq \hat{\theta}_i \leq (1+\epsilon)\theta_i$, *for* $i = 1, \ldots, \zeta$,

where function $f(P_F, P_X)$ is the distance between probability distributions P_F and P_X. This distance which is obtained using statistical tests such as the test χ^2, the Kolmogorov-Smirnov, or the Anderson-Darling goodness of fit test [68]. Parameter β is the significance level, which represents the false alarm rate due to condition (a). Function g yields the deviation tolerance between P_F and P_X, and ϵ is the deviation tolerance for the moments of the distribution. Moreover, the adversary selects the desirable number of observations n for the statistical tests. A larger n lowers the

false alarm rate and reduces the deviation tolerance between P_F and P_X. However, it increases the uncertainty with respect to the precise time that real events occurred.

We consider two traffic analysis models. In the first model, eavesdroppers are assumed to independently perform statistical tests. Their observations sets do not propagate to a central server in order to reduce the delay between the event occurrence and detection and also reduce the energy overhead for operating the eavesdropper network. An eavesdropper notifies the central server only if it detects a statistical change in the observed traffic patterns in his vicinity. This model is suitable when the adversary is interested in near real time detection and wants to prolong the lifetime of battery-operated sensors. Note that this still captures a global adversary capable of eavesdropping on all communications.

In the second model, eavesdroppers are assumed to collude by transferring their observations to a central server. The server collectively performs traffic analysis on all observations. The latter adversarial model is suitable for contextual information that does not vary significantly with time such as the sink's location. Finally, we assume that the adversary does not launch active attacks (e.g., jamming, packet modification, packet injection attacks, etc.) against the WSN.

5.2 Resource-Efficient Traffic Normalization

5.2.1 Design Motivation

Traffic normalization techniques incur high communication overhead for normalizing traffic patterns due to the passive nature of eavesdropping attacks. When the eavesdropper locations are unknown[1], traffic patterns must be normalized for all candidate eavesdropping locations. This is typically achieved by introducing bogus traffic to all sensors of the WSN [50, 65, 78]. Our design goal is to reduce the bogus traffic sources,

[1]Passive receivers can be detected by the radiation leakage of their antennas. However, sophisticated hardware is required for such detection.

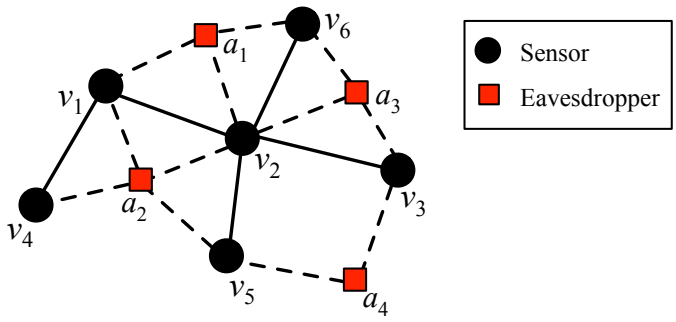


FIGURE 5.1. A WSN operating in the presence of four eavesdroppers $a_1 - a_4$.

while normalizing traffic in the entire WSN deployment area.

We first apply link-layer re-encryption to all packet identifiers (headers, payload, etc.)[2] The encryption operation randomizes the packets transmitted by the same source (even when the payload is static), as well as retransmissions of the same packet over multiple hops. The operation of link-layer re-encryption with encrypted headers was demonstrated in [2]. Alternatively, the anonymous communication methods presented in [46–48] can be used to obfuscate the link-layer identities of the communicating parties.

If the eavesdropper cannot identify the packet originator beyond its reception range, any sensor within the communication range of the eavesdropper can be used to normalize traffic. This is illustrated in the network of Figure 5.1. Eavesdroppers a_1-a_4 intercept the transmissions of sensors v_1-v_5. The solid lines indicate the WSN topology, while the dashed lines indicate the sensors overheard at each eavesdropping location. Instead of transmitting bogus traffic from all sensors, it is sufficient to choose a subset that covers the locations of a_1-a_4. One such subset consists of sensors $\{v_1, v_2, v_3\}$.

To further preserve $(\beta, \epsilon)-$unobservability, bogus traffic rates must be assigned

[2]Some PHY-layer header fields such as the preamble must remain unencrypted for correct protocol operation. However, these fields do not associate a packet transmission with its originator.

such that the transmission of real packets does not alter the eavesdropped transmission patterns at the eavesdropping location. For instance, a_1 and a_2 observe a rate of

$$r_{a_1} = r_{a_2} = r_{v_1} + r_{v_2},$$

while a_3 and a_4 observe a rate of

$$r_{a_3} = r_{v_2} + r_{v_3}$$

and

$$r_{a_4} = r_{v_3},$$

respectively. When a bogus traffic source needs to transmit real packets, it simply substitutes bogus packets with real ones. For inactive sensors (e.g., v_4, v_5, and v_6), transmissions of real packets must not statistically change the packet rate recorded by any of the eavesdroppers.

5.2.2 Traffic Analysis by Independent Eavesdroppers

Independent eavesdroppers may attempt to infer contextual information by observing various communication attributes such as transmission duration, packet type, traffic volume, and inter-packet times. To prevent information leakage, we first normalize packet sizes to a single size. Moreover, packets are encrypted on a per-hop basis to prevent their correlation based on static ciphertexts. To ensure that ciphertexts are random even if the same plaintext is transmitted multiple times, all packets are prefixed with a short random nonce before encryption. In the absence of any other information, the eavesdropper may attempt to identify real traffic by detecting statistically significant changes in the observed packet rate. We prevent the detection of such changes by developing a scheme that consists of two phases: (a) a bogus source selection phase, and (b) a packet rate assignment phase. We now describe the two phases in detail.

Phase I: Bogus Source Selection In the first phase, we select the subset of bogus sources responsible for normalizing traffic patterns. We model the WSN as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of sensors and $\mathcal{E}$ is the set of links, computed based on the unit disc graph model [13]. Let $\mathcal{D} \subseteq \mathcal{V}$ be the set of bogus sources. Set $\mathcal{D}$ is designed to satisfy the following principles:

(a) Bogus traffic is overheard at all candidate eavesdropping locations.

(b) Set $\mathcal{D}$ is of minimum size.

(c) Transmissions of sensors in $\mathcal{V}\backslash\mathcal{D}$ are minimized.

(d) Sensors in $\mathcal{D}$ form a connected network.

Principle (a) is necessary to guarantee contextual information privacy. If eavesdropper a does not intercept any bogus traffic, a will always identify any traffic activity in its vicinity as real traffic. Since the locations of the eavesdroppers are unknown, principle (a) is satisfied when $\mathcal{D}$ covers the WSN deployment area. The second principle minimizes the number of sensors transmitting bogus traffic to reduce the communication overhead. Principle (c), on the other hand, minimizes the number of sensors in $\mathcal{V}\backslash\mathcal{D}$ relaying real traffic to limit the exposure of real sources to detection. Finally, principle (d) is a consequence of principle (c). To keep the traffic transmitted by sensors in $\mathcal{V}\backslash\mathcal{D}$ at a minimum, set $\mathcal{D}$ must form a connected graph for routing real packets to the sink (or any other sensor). This also further improves the communication efficiency by allowing sensors in $\mathcal{D}$ to replace dummy packets with real ones. To satisfy principles (a)-(d), we reduce the problem of obtaining $\mathcal{D}$ to the problem of *finding a minimum connected dominating set that covers the WSN deployment area*. To show this mapping, we first provide the definition of an MCDS [33].

Definition 10. Minimum Connected Dominating Set: *For a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a subset $\mathcal{D} \subseteq \mathcal{V}$ is a dominating set (DS) if any vertex $u \in \mathcal{V}$ either belongs to $\mathcal{D}$, or is adjacent*

(within one hop) to some vertex in $\mathcal{D}$. If $\mathcal{D}$ induces a connected subgraph on $\mathcal{G}$, then $\mathcal{D}$ is a connected dominating set (CDS). The CDS $\mathcal{D}$ with the smallest cardinality is called the minimum connected dominating set (MCDS).

It is straightforward to verify that an MCDS that covers the WSN satisfies properties (a)-(d). Properties (a) and (b) are satisfied by definition. Moreover, any sensor in $\mathcal{V}\backslash\mathcal{D}$ requires only a single transmission to reach a sensor in the MCDS due to the DS property. Once a packet is received by a sensor in $\mathcal{D}$, it can be relayed to the sink (or any other destination) via paths in $\mathcal{D}$. Hence, the number of transmissions from nodes in $\mathcal{V}\backslash\mathcal{D}$ are minimized (property (c)). Finally, the MCDS forms a connected network. Finding an MCDS for arbitrary topologies is known to be an NP-complete problem [32]. We develop a distributed heuristic solution inspired by the heuristic algorithm developed in [11].

Algorithm 5: MCDS approximation–We generate $\mathcal{D}$ in three stages. In Stage 1, we generate a DS and in Stage 2, we approximate the MCDS by expanding the DS to form a connected graph. In Stage 3, we further expand the MCDS to cover the WSN deployment area.

Stage 1: For a sensor $v \in \mathcal{V}$, let $m(v)$ be a marker, which can take the values WHITE, BLACK, or GRAY. Let $\mathcal{N}_v$ be the one-hop neighbors of v, $\delta(v) = |\mathcal{N}_v|$ be the degree of v, and $\delta^*(v)$ be the *effective degree* of v. Parameter $\delta^*(v)$ is the number of WHITE neighbors of v. Let also $\eta(v)$ be the rank of v, defined as the order that v changed its marker relative to a leader node. Finally, let $b(v)$ denote the number of higher-ranked BLACK neighbors of v. All nodes are initialized to $m(v) =$ WHITE, $\delta^*(v) = \delta(v)$, $b(v) = 0$, and $\eta(v) = 0$. The marking process that outputs a DS is as follows.

Stage 1: DS generation

Step 1: A randomly chosen leader v^* starts the process by changing $m(v^*)$ to BLACK. Node v^* becomes a "dominator" and broadcasts $m(v^*) =$ BLACK and

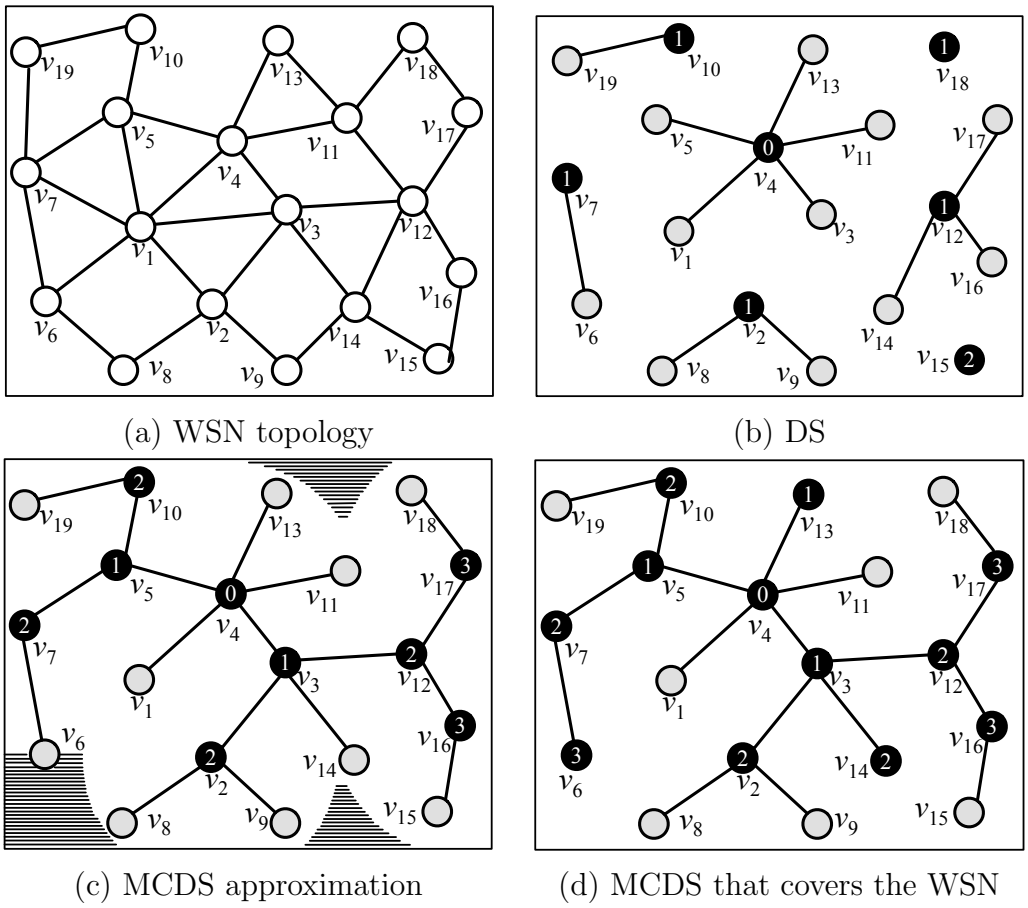


FIGURE 5.2. Stages of the MCDS approximation algorithm. (a) WSN topology, (b) DS topology, (c) approximated MCDS topology, and (d) approximated MCDS that covers the WSN deployment area.

$\eta(v^*) = 0$.

Step 2: Any WHITE sensor u receiving $m(v) =$ BLACK changes its marker to GRAY, sets $\eta(u) = \eta(v)$, and broadcasts $m(u)$ and $\eta(u)$.

Step 3: Any WHITE sensor u that receives $m(v) =$ GRAY, decreases $\delta^*(u)$ by one, updates its rank to $\eta(u) = \eta(v) + 1$ if $\eta(u) \leq \eta(v)$, and broadcasts $\delta^*(u)$ and $\eta(u)$.

Step 4: A WHITE sensor u changes $m(u) =$ BLACK, if

$$u = \arg\max_{v \in \mathcal{N}_u \cup \{u\}} \{\delta^*(v)\}.$$

Ties are broken arbitrarily. Sensor u becomes a "dominator" and broadcasts its new marker value and rank.

Step 5: Any GRAY sensor u receiving $m(v)$ = BLACK and $\eta(u) < \eta(v)$ updates $b(u)$ to $b(u) + 1$.

Step 6: Steps 2 - 5 are repeated until no sensors are marked as WHITE (i.e., $\delta^*(v) = 0, \forall v \in \mathcal{V}$).

Algorithm 5 terminates when there are no sensors marked as WHITE. The set of BLACK sensors dominates all remaining GRAY sensors and forms a DS.

In Figure 5.2(b), we show the DS generated by the application of Algorithm 5 on the WSN of Figure 5.2(a). In Step 1, sensor v_4 declares itself as the leader by broadcasting $m(v)$ = BLACK and $\eta(v) = 0$. In Step 2, sensors v_1, v_3, v_5, v_{11}, and v_{13} change their markers to GRAY and set their ranks to zero. The GRAY sensors broadcast their own markers and ranks. In Step 3, sensors v_2, v_6, v_7, v_{10}, v_{12}, v_{14}, and v_{18} decrease $\delta^*(v)$ by one and increase their ranks by one, after receiving $m(u)$ = GRAY from their GRAY neighbors. In Step 4, sensor v_{12} changes its marker to BLACK since it has the highest effective degree in its neighborhood ($d^*(v_{12}) = 3$). Sensor v_{12} broadcasts its marker and rank. Similarly, v_2, v_7, v_{10}, and v_{18} become BLACK with a rank equal to one. In Step 5, sensors v_1, v_3, v_5, and v_{11} set $b(v)$ to two. The process is repeated for one more iteration which marks v_{15} as BLACK and sets $\eta(v_{15}) = 2$. The WSN is now partitioned to a set of star subgraphs. Each subgraph consists of a set of GRAY sensors dominated by a BLACK sensor. All sensors of a star have the same rank, and the rank of each star increases with its "distance" from the leader.

Stage 2: In Stage 2, GRAY nodes connect two or more star subgraphs by dominating their BLACK neighbors. We select the GRAY sensors that maximize the number of interconnected stars for becoming BLACK in a greedy fashion. The MCDS approxi-

mation stage is as follows.

Stage 2: Approximation of the MCDS

Step 1: All BLACK sensors are assumed not to be dominated. All GRAY sensors v broadcast $b(v)$.

Step 2: The leader becomes dominated by itself.

Step 3: A dominated BLACK sensor v selects GRAY sensor $u \in \mathcal{N}_v$ with

$$u = \arg\max_{\{\mathcal{N}_v, \eta(v)=\eta(u)\}} \{b(u)\}.$$

Sensor u changes its marker to BLACK, its rank to $\eta(u) = \eta(u) + 1$, $b(u) = 0$, and broadcasts $m(u)$, $\eta(u)$, and $b(u)$. Ties are broken arbitrarily.

Step 4: A non-dominated BLACK sensor v receiving $m(u) =$ BLACK and $\eta(u) = \eta(v)$ from $u \in \mathcal{N}_v$, becomes dominated by u. Dominated BLACK sensors change their rank to $\eta(v) = \eta(u) + 1$ and broadcast their new rank.

Step 5: A GRAY sensor w overhearing a rank update from a BLACK sensor $v \in \mathcal{N}_w$ with rank $\eta(v) > \eta(w) + 1$ updates $b(v) = b(v) - 1$.

Step 6: A GRAY sensor w overhearing a rank update from a BLACK sensor v with rank $\eta(v) < \eta(w)$ changes its dominating sensor to v and broadcasts its new $\eta(v)$.

Step 7: Steps 3-6 are repeated until all GRAY sensors have $b(v) = 0$.

Step 8: If a BLACK sensor does not dominate at least one other sensor it changes its marker to GRAY.

From Step 7, we observe that Stage 2 terminates when all GRAY sensors have $b(v) = 0$. That is, all BLACK sensors of Stage 1 are dominated by GRAY sensors of lower rank. Since the algorithm is initiated by the leader, every time a BLACK sensor u is dominated by a GRAY sensor v that turns BLACK, sensor u gets connected to the leader. The process iteratively continues until all BLACK sensors are dominated by GRAY sensors. Therefore, Stage 2 culminates in a connected graph. Note that

in Step 3, the GRAY sensor that can dominate the highest number of BLACK sensors (interconnect the maximum star subgraphs) is selected to change its marker to BLACK. This implements a greedy heuristic approach to minimize the CDS size.

Figure 5.2(c) shows the CDS generated after Stage 2 for the DS of Figure 5.2(b). In Step 1, all GRAY sensors broadcast $b(v)$. In Step 2, leader v_4 becomes dominated by itself. In Step 3, v_4 selects GRAY node v_5 with $b(v_5) = 2$ to connect to the star subgraphs led by v_7 and v_{10}. Sensor v_5 changes its marker to BLACK and updates $b(v_5) = 0$ and $\eta(v_5) = 1$. In Step 4, v_7 and v_{10} change their rank to two. In Step 5, v_1 overhears the rank update from v_7 and changes $b(v_1) = 1$ (it can only dominate v_2). In Step 7, no GRAY sensor overhears a rank update from a BLACK sensor of lower rank. Sensors v_3, v_{16}, and v_{17} turn BLACK in subsequent iterations. Finally, in Step 8, sensors v_{15} and v_{18} turn GRAY since they do not dominate any GRAY sensors. BLACK sensors form a CDS. However, the CDS does not cover the WSN deployment area. The greyed area of Figure 5.2(c) is not covered by any BLACK sensor. At the last stage, we extend the CDS to cover the WSN.

Stage 3: Verifying coverage

Step 1: Sensor v marked GRAY changes its marker to BLACK if its communication area C_v is not covered by $u \in \mathcal{D}$. It also sets its rank to $\eta(v) + 1$.

Step 2: Sensor v broadcasts its new marker and rank.

Note that a GRAY sensor must be aware of the locations of all CDS sensors in its two-hop neighborhood. Therefore, it can compute if its communication area is covered by CDS transmissions. Figure 5.2(d) shows the extended CDS that covers the WSN deployment area. Sensors v_6, v_{13} and v_{11} have changed their markers to BLACK to cover the greyed areas of Figure 5.2(c).

Phase II: Packet Rate Assignment In this phase, we perform the packet rate assignment that satisfies $(\beta, \epsilon)-$ *unobservability*. We separately consider the rate assign-

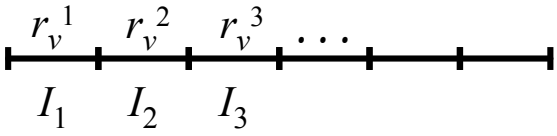


FIGURE 5.3. Packet rate assignment for different intervals.

ment for sensors in $\mathcal{D}$ and in $\mathcal{V}\backslash\mathcal{D}$.

Packet rate assignment in $\mathcal{D}$: We divide time into intervals $I_1, I_2, \ldots$ of length T. At every interval I_k, the packet rate of a sensor $v \in \mathcal{D}$, denoted by r_v^k, is selected from a probability distribution $\mathcal{Y}(\mu_\mathcal{Y}, \sigma_\mathcal{Y})$, where $\mu_\mathcal{Y}$ and $\sigma_\mathcal{Y}$ are the mean and the standard deviation of $\mathcal{Y}$, respectively. The sample space for $\mathcal{Y}$ is set to $[r_{min}, r_{max}]$. The assignment is shown in Figure 5.3.

In the absence of real traffic, $v \in \mathcal{D}$ transmits dummy packets at rate r_v^k. To transmit real traffic, v substitutes dummy packets with real ones, while maintaining the same rate. Real packets propagate to the sink by following the shortest path on the CDS. Since all sensors on the path to the sink are part of the CDS, they can relay real packets without changing their assigned rates. Transmissions from sensors in $\mathcal{D}$ preserve $(\beta, \epsilon)-$*unobservability*, as they follow distribution $\mathcal{Y}$.

Packet rate assignment in $\mathcal{V}\backslash\mathcal{D}$: Assume now that a sensor $u \in \mathcal{V}\backslash\mathcal{D}$ wants to transmit real traffic. Sensor u must relay its traffic to the CDS via a one-hop transmission. To do so, sensor u transmits at interval I_k at a rate r_u^k. The interval I_k and rate r_u^k are selected to conform to the packet rate distribution observed by any eavesdropper. This is achieved as follows. Consider eavesdropper a in the vicinity of u. Let $\mathcal{N}_a$ be the set of sensors in $\mathcal{D}$ overheard by a (at least one such sensors is guaranteed to exist since $\mathcal{D}$ covers the WSN deployment area). In the absence of real traffic, node a observes a packet rate of

$$r_a^k = \sum_{u \in \mathcal{N}_a} r_u^k.$$

The rate observed by a is distributed according to Z_a, which is the sum of $|\mathcal{N}_a|$

independent and identically distributed (i.i.d.) random variables, with each following distribution $\mathcal{Y}$. If u transmits during interval I_k with rate r_u^k, adversary a observes a rate,

$$r_a'^k = r_u^k + r_a^k,$$

which follows distribution Z_a'.

To detect u's transmission, a employs a statistical test that uses a $n-$sample set $\{r_a^{k-n}, \ldots, r_a'^k\}$, collected over n time intervals. We note that sample $r_a'^k$ (that includes r_u^k) is part of n consecutive sets of samples, that are used by a to perform the statistical test in n consecutive intervals (from I_k to I_{k+n}). Since the adversary does not know the interval when u transmitted, a failure in the statistical test during interval $I_{k'}$ $(k \leq k' \leq k+n)$ leads the adversary to conclude that a real transmission has occurred in its neighborhood, and at some interval in the window of time from $I_{k'-n}$ to $I_{k'+n}$.

In order to avoid the detection, sensor u must choose r_u^k in a way that the statistical test performed by a satisfies $(\beta, \epsilon)-$*unobservability* for each of the observation sets that include r_u^k. Sensor u runs the tests on each observation set $\{r_a^{k-n}, \ldots, r_a'^k\}$, $\{r_a^{k-n+1}, \ldots, r_a^{k+1}\}$, $\ldots$, $\{r_a'^k, \ldots, r_a^{k+n}\}$, and chooses the maximum rate r_u^k that passes all of them. If such rate cannot be found, sensor u postpones its transmission for a future interval. To compute the appropriate rates r_a^i of each observation set, sensor u must be aware of the rates individually selected by sensors in $\mathcal{N}_a$. Since the rates are randomly chosen, u calculates them using the random seed value that each sensor in $\mathcal{N}_a$ uses. This information can be made available during the CDS setup phase.

Also, note that $\mathcal{N}_a$ depends on the eavesdropper's location. Since the location of a is unknown, the set of observations must be computed for all possible locations of a in the communication area of u. For each location, a unique subset of CDS nodes that can be overheard by a is defined. We denote such unique sets as $\mathcal{N}_a^1, \mathcal{N}_a^2, \ldots, \mathcal{N}_a^m$.

For each $\mathcal{N}_a^j$, sensor u computes the set,

$$\mathcal{O}_j = \{\mathcal{O}_j^1, \ldots, \mathcal{O}_j^n\},$$

where $\mathcal{O}_j^i = \left\{r_a^{k-n+i}, \ldots, r_a'^k, \ldots, r_a^{k+i}\right\}$. We now describe Algorithm 6, that is used for determining sets $\mathcal{N}_a^1, \mathcal{N}_a^2, \ldots, \mathcal{N}_a^m$.

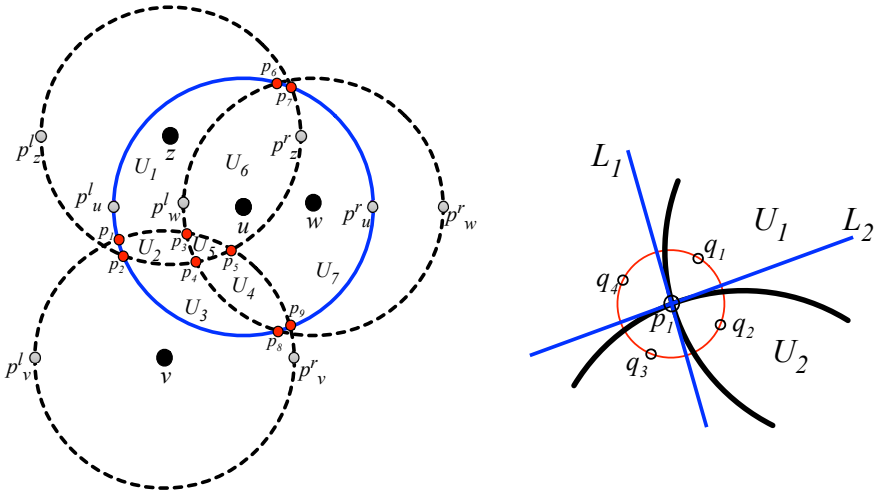
Algorithm 6: Consider the area R_u^ψ, where sensor u is eavesdropped, centered at ℓ_u. Here, ψ denotes the reception radius of an eavesdropper. The unique sets $\mathcal{N}_a^j$ overheard by an eavesdropper in R_u^ψ can be determined by partitioning R_u^ψ to areas $U_1, U_2, \ldots, U_m$, defined by the intersection of R_u^ψ with the communication areas of sensors in $\mathcal{D}$. For two areas U_i and $U_j, i \neq j$ it holds that $\mathcal{N}_a^i \neq \mathcal{N}_a^j$. Hence, determining $\mathcal{N}_a^j$ at each area U_j yields all unique sets of sensors overheard by an adversary a that can be arbitrarily located in R_u^ψ. In Figure 5.4, we show R_u^ψ partitioned to seven areas due to the transmissions by CDS sensors z, v, and w.

To calculate $\mathcal{N}_a^1, \mathcal{N}_a^2, \ldots, \mathcal{N}_a^m$, sensor positions in the area $R_u^{2\psi}$ are assumed to be known by u. Sensor u computes set $S = \{v | v \in \mathcal{D}, R_u^\psi \cap R_v^\psi \neq \emptyset\}$ using Algorithm 6. Set S contains all CDS sensors whose communication areas intersect with R_u^ψ. Algorithm 6 proceeds in two stages. In Stage 1, u computes all points within R_u^ψ where at least two communication area boundary circles of sensors in $S \cup \{u\}$ intersect. In Stage 2, u computes $\mathcal{N}_a^j$ by determining the sensors overheard at each area U_j.

Stage 1: We first provide a relevant definition.

Definition 11 (Endpoint)**.** *Points $p^l = (\ell^x - \psi, \ell^y)$ and $p^r = (\ell^x + \psi, \ell^y)$, of a circle $R(\ell, \psi)$ of radius ψ, centered at $\ell = (\ell^x, \ell^y)$, represent the left and right circle endpoint, respectively.*

To compute all intersection points between R_u^ψ and any other circle, we use the "plane sweep" algorithm in [3]. Let $\mathcal{P}$ be the list of endpoints of all communication areas of sensors in S that fall in R_u^ψ, sorted in ascending order (from left to right). A vertical line L sweeps the plane from p_u^l to p_u^r. We use set $\mathcal{Q}$ to store the current



(a) Stage 1 (b) Stage 2

FIGURE 5.4. Execution of Algorithm 6. (a) Computation of circle intersection points, and (b) computation of $\mathcal{N}_a^j$, for p_1.

list of sensors whose communication range intersects with the current position of L. Finally, let $\mathcal{M}$ be the list of intersecting points between any circles of sensors in $S \cup \{u\}$. Set $\mathcal{M}$ also stores the ids of the circles intersecting at any point. The plane sweep algorithm proceeds as follows.

Stage 1: Plane Sweep

Step 1: Initialize S, $\mathcal{P}$, $L = \mathcal{P}(1)$, and index $i = 1$.

Step 2: Obtain $\mathcal{Q}(i) = \{v \mid R_v^{\psi} \cap L \neq \emptyset\}$

Step 3: Compute the intersection points between any two circles of sensors in $\mathcal{Q}$, and add them to $\mathcal{M}$.

Step 4: Move L to the next point $\mathcal{P}(i+1)$. If $\mathcal{P}(i+1)$ is a left endpoint of a sensor v, $\mathcal{Q}(i+1) = \mathcal{Q}(i) \cup \{v\}$. Otherwise, if $\mathcal{P}(i+1)$ is the right endpoint of v, $\mathcal{Q}(i+1) = \mathcal{Q}(i) \backslash \{v\}$.

Step 5: Update $i = i + 1$, and repeat Steps 2-4 while $i \leq |\mathcal{P}|$.

An example of the plane sweep algorithm is shown in Figure 5.4(a). Initially, $S = \{u, v, w, z\}$, $\mathcal{P} = \{p_u^l, p_w^l, p_v^r, p_z^r, p_u^r\}$ and $\mathcal{M} = \emptyset$. The vertical line L is initialized at point p_u^l. In Step 2, we set $\mathcal{Q}(1) = \{u, v, z\}$. In Step 3, points p_1, p_2, p_5, p_7, and p_9 are added to set $\mathcal{M}$. These points correspond to the intersection of circles from pairs of sensors (u, v), (u, z) and (v, z). In Step 4, L is moved to p_w^l and $\mathcal{Q}(2) = \{u, v, z, w\}$. Repeating Step 3 yields the intersection points between circles from pairs of sensors (u, w), (v, w) and (z, w). Set $\mathcal{M}$ becomes $\{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9\}$. This stage terminates when the vertical line L is placed at point p_u^r.

Stage 2: In Stage 2, we use the points in $\mathcal{M}$ to determine the unique sets $\mathcal{N}_a^j$ that can be overheard at each U_j. Stage 2 proceed as follows.

Stage 2: $\mathcal{N}_a^j$ Determination

Step 1: Initialize $p = \mathcal{M}(1)$

Step 2: For each circle that intersects at p, draw a tangent line at p. For x intersecting circles, the tangent lines at p partition the plane to $2x$ sections.

Step 3: For each section, select a point q on the internal angle bisector of that section, with $|p - q| \leq \epsilon$, where ϵ is some small value. Then $\mathcal{N}_a^j = \{v \mid v \in \mathcal{D}, |\ell_v - q_j| \leq \gamma\}$. Advance to next intersection point in $\mathcal{M}$.

Step 4: Repeat Steps 2-3 for all points in $\mathcal{M}$.

Step 5: Prune sets $\mathcal{N}_a^j$, that appear more than once.

In Figure 5.4(b), we show the application of Stage 2 for two circles intersecting at p_1. The tangent lines at p_1 partition the plane to four sections. Point q_1 is selected on the bisector of section U_1. Based on the distance of q_1 to sensors in $\mathcal{D}$, set $\mathcal{N}_a^1 = \{z\}$. Similarly, we obtain q_2 and $\mathcal{N}_a^2 = \{v, z\}$. Note that q_3 and q_4 are not within the communication area of u, and therefore are not taken into account.

With the termination of Algorithm 6, sensor u is aware of all possible sets $\mathcal{N}_a^j$ that can be overheard by a. For each $\mathcal{N}_a^j$, sensor u computes the corresponding set $\mathcal{O}_j$.

To do so, sensor u must know the packet rates of all sensors in $\mathcal{N}_a^j$. This information is made available during the CDS setup phase. Each sensor $v \in \mathcal{D}$ generates the sequence of rates for each interval using a random seed. This seed is shared with all sensors in the two-hop neighborhood of v. Hence, sensor $u \notin \mathcal{D}$ is aware of all the seeds of its two-hop neighbors that belong to $\mathcal{D}$ (only the two-hop neighborhood of u can be part of $\mathcal{N}_a^j$). Moreover, to account for co-located real data sources, when u chooses rate r_u^k for interval I_k, it announces this information to its two-hop neighborhood (using the selected rate). Hence, other sensors that are affected by this real traffic transmission can obtain the correct sample values. Note that this announcement is disseminated by sensors of the CDS, after it is initially transmitted by u.

Sensor u performs statistical tests on every set $\mathcal{O}_j$ obtained for each unique $\mathcal{N}_a^j$, and determines the highest rate r_u^k that is statistically undetectable. The computation overhead for performing statistical tests can be reduced by considering only the observation set that is likely to introduce the highest deviation between Z_a and Z'_a. This set corresponds to the $\mathcal{N}_a^j$ with the smallest cardinality. In the following proposition we show that as the cardinality of $\mathcal{N}_a^j$ increases, the deviation between Z_a and Z'_a reduces.

Proposition 1. *Let Z_a and Z'_a be the sum of N and $(N+1)$ i.i.d. random variables of type $\mathcal{Y}$, respectively. The difference between distributions Z_a and Z'_a restricted to the sample space of Z_a, decreases as N increases.*

Proof. Let f_{Z_a} and $f_{Z'_a}$ represent the probability density function of Z_a and Z'_a, respectively. Since both distributions are the sum of random variables $\mathcal{Y}$, the sample space of Z_a is $(0, Nr_{\max})$ and the one of Z'_a is $(0, (N+1)r_{\max})$. Let the difference

between these two distributions, restricted to the sample space of Z_a, be given by,

$$\begin{aligned}
d &= \int_0^{Nr_{\max}} f_{Z_a}(r) - f_{Z'_a}(r)dr \\
&= \int_0^{Nr_{\max}} f_{Z_a}(r)dr - \int_0^{Nr_{\max}} f_{Z'_a}(r)dr \\
&= 1 - \int_0^{Nr_{\max}} f_{Z'_a}(r)dr \\
&= 1 - \int_0^{(N+1)r_{\max}} f_{Z'_a}(r)dr + \int_{Nr_{\max}}^{(N+1)r_{\max}} f_{Z'_a}(r)dr \\
&= 1 - 1 + \int_{Nr_{\max}}^{(N+1)r_{\max}} f_{Z'_a}(r)dr \\
&= \int_{Nr_{\max}}^{(N+1)r_{\max}} f_{Z'_a}(r)dr.
\end{aligned}$$

Since Z'_a is distributed according to the sum of i.i.d. random variables, we apply the Central Limit Theorem (CLT) to approximate Z'_a as a normal distribution. Therefore, we obtain,

$$\begin{aligned}
d &= \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{(N+1)r_{\max} - (N+1)\mu_{\mathcal{Y}}}{\sqrt{2(N+1)\sigma_{\mathcal{Y}}^2}}\right)\right] - \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{Nr_{\max} - (N+1)\mu_{\mathcal{Y}}}{\sqrt{2(N+1)\sigma_{\mathcal{Y}}^2}}\right)\right] \\
&= \frac{1}{2}\left[\operatorname{erf}\left(\frac{(N+1)r_{\max} - (N+1)\mu_{\mathcal{Y}}}{\sqrt{2(N+1)\sigma_{\mathcal{Y}}^2}}\right) - \operatorname{erf}\left(\frac{Nr_{\max} - (N+1)\mu_{\mathcal{Y}}}{\sqrt{2(N+1)\sigma_{\mathcal{Y}}^2}}\right)\right].
\end{aligned}$$

Without loss of generality, we assume that $\mathcal{Y}$ is a uniform distribution in the interval $(0, r_{\max})$. Then we have $\mu_{\mathcal{Y}} = \frac{r_{\max}}{2}$ and $\sigma^2 = \frac{r_{\max}^2}{12}$. Then by replacing this values, we have,

$$\begin{aligned}
d &= \frac{1}{2}\left[\operatorname{erf}\left(\frac{(N+1)r_{\max} - (N+1)\frac{r_{\max}}{2}}{\sqrt{2(N+1)\frac{r_{\max}^2}{12}}}\right) - \operatorname{erf}\left(\frac{Nr_{\max} - (N+1)\frac{r_{\max}}{2}}{\sqrt{2(N+1)\frac{r_{\max}^2}{12}}}\right)\right] \\
&= \frac{1}{2}\left[\operatorname{erf}\left(\sqrt{\frac{3(N+1)}{2}}\right) - \operatorname{erf}\left(\sqrt{\frac{3(N+1)}{2}} - \sqrt{\frac{3}{2(N+1)}}\right)\right].
\end{aligned}$$

In order to study the monotonic behavior of d with respect to N, we obtain its derivative,

$$\frac{\partial d}{\partial N} = \frac{1}{\sqrt{\pi}}\left[e^{-\frac{3(N+1)}{2}}\left(\sqrt{\frac{1}{6(N+1)}}\right) - e^{-\frac{3(N+1)}{2}+\frac{9N+6}{2(N+1)}}\left(\sqrt{\frac{1}{6(N+1)}}+\sqrt{\frac{27}{32(N+1)^3}}\right)\right]$$
$$= \frac{1}{\sqrt{\pi}}e^{-\frac{3(N+1)}{2}}\sqrt{\frac{1}{6(N+1)}}\left[1 - e^{\frac{9N+6}{2(N+1)}}\left(1+\sqrt{\frac{27}{32(N+1)^3}}\right)\right].$$

Note that, for any $N \geq 0$ we have the following,

$$\sqrt{\frac{1}{6(N+1)}} > 0, e^{-\frac{3(N+1)}{2}} > 0.$$

Then the sign of $\frac{\partial d}{\partial N}$ is determined by the sign following expression,

$$1 - e^{\frac{9N+6}{2(N+1)}}\left(1+\sqrt{\frac{27}{32(N+1)^3}}\right).$$

Since,

$$1+\sqrt{\frac{27}{32(N+1)^3}} > 1, e^{\frac{9N+6}{2(N+1)}} > 1,$$

we have,

$$1 - e^{\frac{9N+6}{2(N+1)}}\left(1+\sqrt{\frac{27}{32(N+1)^3}}\right) < 0,$$

and therefore,

$$\frac{\partial d}{\partial N} < 0,$$

which shows that d is a decreasing monotonic function with respect to N. Thus, the difference between Z_a and Z'_a decreases, as the value of N increases. □

Statistical Testing Process–Sensor u uses the n sample sets $\mathcal{O}_j = \left\{\mathcal{O}_j^1, \ldots, \mathcal{O}_j^n\right\}$ affected by u's transmission, in order to estimate r_u^k such that (β, ϵ)-*unobservability* is satisfied. Recall from Definition 9 that distributions Z_a and Z'_a are indistinguishable if,

(a) $f(Z_a, Z'_a) \leq g(\beta)$,

(b) $(1-\epsilon)\theta_i \leq \hat{\theta}_i \leq (1+\epsilon)\theta_i$, for $i = 1, \ldots, \zeta$.

We first obtain the maximum value of r_u^k that satisfies condition (b). For example, consider the first moment μ_{Z_a} of Z_a.

$$(1-\epsilon)\mu_{Z_a} \leq \mu_{Z'_a} \leq (1+\epsilon)\mu_{Z_a}.$$

Applying the CLT we can reduce the right hand side of this equation to,

$$\frac{1}{\left(\left|\mathcal{N}_a^j\right|+1\right)}\left(r_u^k + \sum_{v \in \mathcal{N}_a^j} r_v^k\right) \leq (1+\epsilon)\mu_{Z_a}.$$

We consider only the right hand side of the inequality since we want $r_u^k > 0$. Taking into account conditions (a) and (b), the rate r_u^k is calculated as the maximum value that satisfies,

$$f(Z_a, Z'_a) \leq g(\beta),$$

$$r_u^k \leq \left(\left|\mathcal{N}_a^j\right|+1\right)(1+\epsilon)\mu_{Z_a} - \sum_{v \in \mathcal{N}_a^j} r_v^k,$$

for an statistical test f such as the χ^2, the Kolmogorov-Smirnov, or the Anderson-Darling goodness of fit tests. This process is applied for the set,

$$\mathcal{N}_a^j = \arg\min_{\mathcal{N}_a^i \in \{\mathcal{N}_a^1, \mathcal{N}_a^2, \ldots, \mathcal{N}_a^m\}} \left|\mathcal{N}_a^i\right|.$$

Similar computations can be made for other the moments θ_i, for $i = 2, \ldots, \zeta$.

5.3 Centralized Traffic Analysis

In this section, we analyze a collusion scenario, in which the eavesdropped attributes are collectively analyzed at a central location. This scenario is relevant when the delay of inferring contextual information is not critical. We assume that eavesdroppers record a packet hash $h(pk_i)$ and the transmission time t_i, for each packet pk_i they

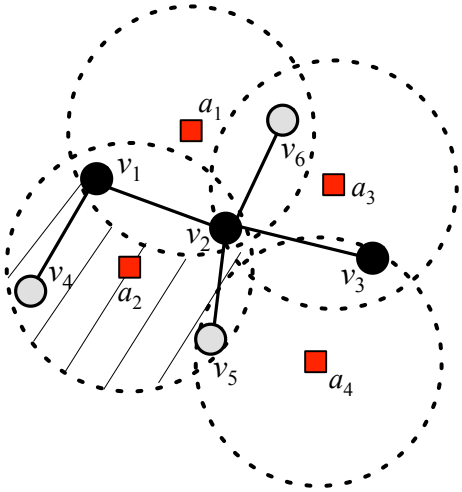


FIGURE 5.5. An eavesdropper collusion scenario.

intercept. At the end of an interval I_k, each eavesdropper forwards to the central server the set of hashes $\mathcal{H}_a^k = \{h(pk_1), h(pk_2), \ldots\}$, the associated transmission times $\mathcal{T}_a^k = \{t_1, t_2, \ldots\}$ and his location ℓ_a.

Joint processing of the packet hashes and of timing information can lead to the identification of real data sources, even when traffic is normalized using the method presented in Section 5.2. This vulnerability is illustrated in Figure 5.5 which depicts five sensors coexisting with four eavesdroppers. The reception ranges of the eavesdroppers are marked with dashed circles. In this example, traffic is normalized by CDS sensors $\mathcal{D} = \{v_1, v_2, v_3\}$. In the absence of real traffic, eavesdroppers a_1 and a_2 intercept the traffic generated by sensors v_1 and v_2. It follows that $\mathcal{H}_{a_1}^k = \mathcal{H}_{a_2}^k$ and $\mathcal{T}_{a_1}^k = \mathcal{T}_{a_2}^k$ for all intervals I_k that real transmissions do not occur. Assume that v_4 introduces real traffic at rate r_4^j during some interval I_j.

Sensor v_4 is in the reception range of a_2, but not that of a_1. Therefore, for interval I_j, it follows that $\mathcal{H}_{a_1}^j \neq \mathcal{H}_{a_2}^j$ and $\mathcal{T}_{a_1}^j \neq \mathcal{T}_{a_2}^j$. Using this discrepancy, the central server can conclude that real data transmissions occurred within the communication range of a_2 but not of a_1 (grayed area of Figure 5.5). In fact, the central server can identify

the packets and transmission times of the real traffic sent by v_4.

A similar analysis can be made for cases where two eavesdroppers do not overhear a common sensor that belongs to the CDS, but overhear common sensors that do not belong to the CDS. In the example of Figure 5.5, sets $\mathcal{H}_{a_2}^k$, $\mathcal{H}_{a_4}^k$, and $\mathcal{T}_{a_2}^k$, $\mathcal{T}_{a_4}^k$ report common packets only when v_5 is active. To address this attack, we propose a solution that employs power control to randomize the sensor transmission radii. Sensors in $\mathcal{D}$ vary their transmission power levels to randomize the transmission patterns intercepted by colluding eavesdroppers.

5.3.1 Traffic Randomization based on Power Control

To randomize the transmission patterns observed by colluding eavesdroppers, each sensor $v \in \mathcal{D}$ individually selects at random its transmission power ρ_v^k for interval I_k from a set of discrete power levels

$$\{\rho_1, \rho_2, \ldots, \rho_\lambda\} = \left\{ \sqrt{\frac{1}{\lambda}}\gamma, \sqrt{\frac{2}{\lambda}}\gamma, \ldots, \gamma \right\},$$

where γ is the maximum power level allowed at each sensor.

The discrete power levels are selected to partition the sensor communication area to smaller areas of equal size, assuming line-of-sight signal propagation (attenuation factor equal to two). Sensors generate the sequence of power levels used at every interval based on a unique random seed. This random seed is shared with all two-hop neighbors of every CDS sensor, so that the graph topology is known during the packet rate assignment phase. The two-hop neighborhood of a sensor is defined based on the maximum power level ρ_λ. We now describe the necessary modifications to the two phases of the traffic normalization technique presented in Section 5.2, for accommodating varying power levels.

5.3.2 Phase I: Selection of Bogus Traffic Sources

In Phase I, we select the bogus traffic sources by constructing an MCDS approximation using Algorithm 5, presented in Section 5.2. We note that under randomly selected power levels, the underlying graph for executing Algorithm 5 could take an arbitrary form, that does not necessarily follow the unit disc model. To execute Algorithm 1 on a fixed graph, we consider the graph that is constructed when the lowest power level ρ_1 is selected by *every* sensor. This results to a graph $\mathcal{G}_1(\mathcal{V}, \mathcal{E}_1)$, where $\mathcal{E}_1$ denotes the edge set obtained when all sensors set their power level to ρ_1. Graph $\mathcal{G}_1$ follows the unit graph model and hence, we can apply Algorithm 5 on $\mathcal{G}_1$ to obtain an MCDS. It is straightforward to show that the MCDS approximation constructed for $\mathcal{G}_1$ is also a CDS for any other graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ that is a result of random power level selection. This is because $\mathcal{E}_1 \subseteq \mathcal{E}$ for all possible $\mathcal{E}$ realizations. That is, any edge set $\mathcal{E}$ due to random power selection contains all edges of E_1. After applying Algorithm 5 on graph $\mathcal{G}_1$, the resulting CDS $\mathcal{D}$ satisfies the following properties.

- $\mathcal{D}$ is an MCDS approximation when all sensors select the lowest power level ρ_1.
- $\mathcal{D}$ forms a CDS when sensors select their transmission arbitrarily from discrete power levels $\{\rho_1, \ldots, \rho_\lambda\}$.

5.3.3 Phase II: Power and Packet Rate Assignment

In Phase II, we provide $(\beta, \epsilon)-$*unobservability* under colluding eavesdroppers by assigning appropriate packet rates and power levels to sources of real and bogus traffic. We separately treat CDS sensors and sensors that belong to $\mathcal{V} \backslash \mathcal{D}$.

Power and Rate Assignment in $\mathcal{D}$ At every interval I_k, sensors $v \in \mathcal{D}$ generate bogus traffic at a packet rate r_v^k randomly selected from $\mathcal{Y}(\mu_\mathcal{Y}, \sigma_\mathcal{Y})$. Packets are transmitted at power level ρ_v^k selected at random from $\{\rho_1, \ldots, \rho_\lambda\}$. Transmissions of real packets occur by substituting dummy packets with real ones, while maintaining the same

packet rate and power level. The bogus traffic rate observed by an eavesdropper during interval I_k is given by,

$$r_a^k = \sum_{v \in \mathcal{N}_a^k} r_v^k,$$

where $\mathcal{N}_a^k$ is the set of sensors overheard by a during I_k. Note that $|\mathcal{N}_a^k|$ is a random variable which is independent of the individual rate selection at each sensor.

Let $|\mathcal{N}_a^k|$ be distributed according to $\mathcal{W}(\mu_{\mathcal{W}}, \sigma_{\mathcal{W}}^2)$. Then the packet rate distribution Z_a observed by a is a random sum of independent and identically distributed (i.i.d.) random variables with

$$\mu_{Z_a} = \mu_{\mathcal{W}} \mu_{\mathcal{Y}},$$

and

$$\sigma_{Z_a}^2 = \mu_{\mathcal{W}} \sigma_{\mathcal{Y}}^2 + \sigma_{\mathcal{W}}^2 \mu_{\mathcal{Y}}^2.$$

The adversary uses the values of μ_{Z_a} and $\sigma_{Z_a}^2$ to perform statistical tests for detecting real traffic.

Power and Rate Assignment in $\mathcal{V} \backslash \mathcal{D}$ A sensor $u \in \mathcal{V} \backslash \mathcal{D}$ always selects the minimum power level ρ_1 for transmitting real traffic. This selection minimizes the exposure of u to eavesdropping. Note that u is guaranteed to be within one hop from $v \in \mathcal{D}$ when transmitting at power level ρ_1 since $\mathcal{D}$ is constructed based on $\mathcal{G}_1$.

The rate r_u^k is selected following a similar process to Algorithm 6. For a given interval I_k, sensor u applies the plane sweep algorithm to compute all the intersection points between the communication area boundary circles of its active one- and two-hop neighbors. The circle radii are set to the transmission radii of the u's neighbors sensors, as computed based on the known random seed of each sensor. With the completion of Algorithm 6, sensor u has identified the observation sets of an adversary within its communication range. Sensor u proceeds to find the maximum allowable packet rate that preserves $(\beta, \epsilon)-$*unobservability*, by applying the testing process outlined in Section 5.2. We note that Algorithm 6 is based on the execution of the

"plane sweep" process proposed in [3]. This process does not depend on the unit disc model and is applicable on graphs formed by nodes with varying communication ranges.

5.4 Performance Evaluation

In this section, we evaluate the privacy, communication overhead and computational complexity of our algorithms.

5.4.1 Privacy Analysis

The privacy provided by our methods relies on the indistinguishability between the transmission of real and bogus packets. This is achieved by generating bogus traffic by a subset $\mathcal{D}$ of nodes. For sensors in $\mathcal{D}$, real information is transmitted by simply replacing scheduled transmission of dummy packets by real ones. By design $\mathcal{D}$ is chosen to be a connected subgraph, which is used to route packets to the sink. Hence, the routing of real packets does not alter the traffic patterns observed by any eavesdropper, individually, or collectively.

For sources in $\mathcal{V}\backslash\mathcal{D}$, our schemes are designed to satisfy the (β, ϵ)-unobservability. We note that, the level of privacy provided by our methods is equivalent to the one provided by the methods proposed in [1, 50, 65]. In these methods, all sensors generate bogus packets, and real transmissions are scheduled by replacing bogus ones. In our methods and those in [1,50,65], the statistical properties of the traffic patterns observed by the eavesdropper do not change. Therefore, the eavesdroppers are limited to randomly guessing whether a packet is real or bogus.

As our schemes generate traffic using set $\mathcal{D}$, it is critical that the adversary is not able to identify the nodes in $\mathcal{D}$. We first note that the eavesdroppers are assumed to intercept all packets transmitted within their communication range without being able to differentiate between the packet sources (no localization capabilities). Recall that

explicit packet identifiers have been removed to prevent the packet association with its source using link-layer re-encryption [2]. Under this eavesdropping model, eavesdroppers observe the aggregate packet rate of all the sensors within their transmission range. This aggregate rate exhibits the same statistical properties independent of the transmission of real traffic. Hence, eavesdroppers cannot employ traffic analysis to differentiate between nodes in $\mathcal{D}$ and $\mathcal{V}\backslash\mathcal{D}$. Moreover, by construction $\mathcal{D}$ is connected and covers the deployment area, which guarantees that at least one dummy traffic source would be overheard at every possible eavesdropping location.

5.4.2 Communication Overhead

We first evaluate the communication overhead of Algorithm 5, which is presented in Section 5.2. This is given by the following proposition.

Proposition 2. *The communication overhead of Algorithm 5 is upper-bounded by* $|\mathcal{V}|\,(2\Delta+3)$, *where* Δ *is the maximum node degree in the WSN.*

Proof. In Stage 1 of Algorithm 5, a sensor changes its marker to either BLACK or GRAY. In Stage 2, a subset of GRAY sensors change their markers to BLACK, while a subset of BLACK sensors (CDS tree leaves) change their markers to GRAY. Finally, in Stage 3, GRAY sensors change their markers to BLACK to achieve coverage. Therefore, one sensor can transmit up to three packets to indicate a marker change. Moreover, in Stage 1, the value of $\delta^*(v)$ is transmitted by a sensor v each time a neighboring sensor becomes either BLACK or GRAY. Given the maximum node degree Δ, each $\delta^*(v)$ can be transmitted up to Δ times. Finally, in the worst case, a sensor v would have all its neighbors marked as BLACK, which forces v to update $b(v)$ up to Δ times. Summing all possible transmissions yields the upper bound of Proposition 2. □

The communication overhead for traffic normalization depends on the distribution $\mathcal{Y}$ used for selecting packet rates. In the absence of real traffic and assuming a rate selection from a uniform distribution over $[r_{\min}, r_{\max}]$, the overhead is given by the following proposition.

Proposition 3. *The average number of bogus packets transmitted during one interval I_k, assuming uniformly selected packet rates over $[r_{\min}, r_{\max}]$ is*

$$\frac{1}{2} |\mathcal{D}| T (r_{\max} - r_{\min}) . \tag{5.1}$$

Proof. The proof follows immediately by observing that Equation (5.1) is the mean value of the uniform distribution multiplied by the number of bogus traffic sources (size of $\mathcal{D}$). This is true because each sensor selects its packet rate independently. □

5.4.3 Computational Complexity

We now evaluate the computational complexity of Algorithm 6 and the complexity of the necessary statistical tests for determining an acceptable packet rate at a real source that does not belong to the CDS. The complexity of Algorithm 6 is given by the following proposition.

Proposition 4. *Algorithm 6 has computational complexity of*

$$O((\Delta^2 + X) \log \Delta^2),$$

where Δ is the maximum node degree in the WSN, and X is the total number of intersection points between the circle C_u of sensor u and the circles C_v of u's neighbors.

Proof. Consider the application of Algorithm 6 at sensor u. The process of obtaining sets $\mathcal{N}_a^1, \mathcal{N}_a^2, \ldots, \mathcal{N}_a^m$ consists of two stages. In Stage 1, we use the algorithm presented in [3] to obtain the intersection points between the communication area boundaries of sensors in $\mathcal{D}$, that belong to the one-hop neighborhood of a sensor u. As shown in [3],

this algorithm has complexity $O((\Delta^2 + X)\log\Delta^2)$, where X is the total number of intersection points between the area C_u of sensor u and the areas C_v of its neighbors $v \in \mathcal{N}_u \cap \mathcal{D}$, that belong to the CDS.

In Stage 2, we execute a constant number of operations to obtain the tangents and the points in U_j. Therefore, Stage 2 has complexity $O(X)$. Adding the respective complexities yields the result of Proposition 4. □

We now consider the complexity of performing n statistical tests over n observation sets. For our analysis, we use the Kolmogorov-Smirnov (K-S) test which is known to have good performance when the observed distribution is unknown [71]. Similar results can be obtained for other goodness-of-fit tests, such as the Anderson-Darling test.

Proposition 5. *The complexity of performing n K-S tests on sets $\mathcal{O}_j^1, \ldots, \mathcal{O}_j^n$ is $O(n^2)$.*

Proof. To perform a K-S test, the observation set $\mathcal{O}_j^1$ is first sorted to obtain the value of the K-S statistic

$$\Phi = \max_{1\leq k\leq n}\left(F_{Z_a}(r_a^k) - \tfrac{k-1}{n}, \tfrac{k}{n} - F_{Z_a}(r_a^k)\right),$$

where F_{Z_a} is the cumulative distribution function of Z_a, and n is the size of each observation set. Φ is used to compare the theoretical and sample distributions. Sorting a sample set $\mathcal{O}_j$ can be completed in $O(n\log n)$ operations, using well known sorting algorithms. On the other hand, obtaining the value of Φ requires $O(n)$ operations. Therefore, the total complexity of performing one K-S test is $O(n\log n)$. For performing n tests, note that consecutive observation sets (e.g., $\mathcal{O}_j^1$ and $\mathcal{O}_j^2$) differ only in the first and last sample. Therefore, consecutive tests require $O(\log n)$ operations to be sorted (by placing the new sample in the right order). Moreover, the computation of Φ requires $O(n)$ operations, making the complexity of each subsequent test

$O(\log n + n) = O(n)$. Accounting for all $(n-1)$ remaining tests yields a complexity of $O(n^2)$. □

5.5 Simulation Experiments

In this section, we use simulations to evaluate the performance of our traffic normalization scheme. We performed our simulations on Matlab 2012. All results were averaged over 10 independent runs. In our evaluation, we considered the following performance metrics.

1. The average fraction of sensors that generates bogus traffic (normalized CDS size).
2. The average packet rate of sensors in $\mathcal{V}\backslash\mathcal{D}$ for transmitting real packets while satisfying the statistical tests.
3. The average end-to-end delay for propagating real packets to the sink under a fixed communication budget.
4. The average communication overhead for propagating real packets to the sink under a constant dummy packet rate generation.
5. The fraction of eavesdroppers that are able to detect a real transmission under a collusion scenario.

In our experiments, we uniformly deployed a WSN within an area of 450×450 meters and fixed the communication range of each sensor to 50 meters. The sensor deployment density was controlled to ensure a connected network and satisfy varying connectivity degree requirements. For the average end-to-end delay computation, we measured the delay due to the packet rate control applied at each sensor and ignored any physical-layer delays (e.g., transmission and propagation delay). This is because

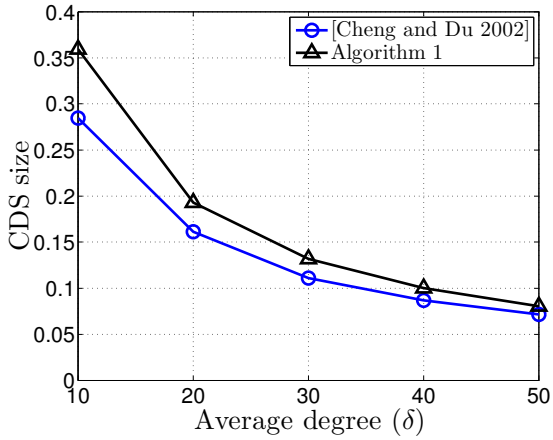


FIGURE 5.6. Average CDS size normalized over the WSN size.

the delay due to packet rate control is expected to be several orders of magnitude larger compared with physical-layer delays, and hence is the dominant delay factor. Moreover, physical-layer delays occur even if packet rate control is not applied. Our goal is to measure the additional delay overhead due to controlling the transmissions at each sensor.

5.5.1 CDS Size

In the first set of experiments, we varied the average sensor degree δ by adjusting the sensor density. We performed 10 WSN deployments for each δ and executed Algorithm 5 for obtained an MCDS approximation $\mathcal{D}$. We compared the CDS size obtained by Algorithm 5 with the CDS obtained by executing the heuristic algorithm presented in [11]. The latter is not designed to guarantee coverage of the deployment area.

In Figure 5.6, we show the average fraction of $\mathcal{V}$ that belongs to $\mathcal{D}$, as a function of the average sensor degree δ. We observe that as the average sensor degree increases, the fraction of sensors that belong to the CDS decreases. This is expected since the number of sensors dominated by each sensor in the CDS increases with the deployment

density. The CDS size reflects the energy savings compared with previous approaches that require all sensors in the WSN to be sources of dummy traffic [50, 65]. In our approach, only 35% of the sensors act as bogus sources when $\delta = 10$. This fraction drops to less than 10% when $\delta = 50$, indicating significant energy savings. Moreover, we observe that Algorithm 5 generated CDSs that have similar size to the heuristic in [11], while guaranteeing coverage of the WSN deployment area.

5.5.2 Average Packet Rate for Sensors in $\mathcal{V}\backslash\mathcal{D}$

In the second set of experiments, we evaluated the average packet rate achieved by sensors in $\mathcal{D}\backslash\mathcal{V}$, with and without power control. To detect real traffic sources based on the observed packet rates, eavesdroppers were assumed to perform the K-S goodness of fit test on observation sets of 50 rate samples. Time was divided to intervals $I_1, I_2, \ldots$ of length one second. At each interval I_k, sensors of the CDS selected their rates uniformly in the range $(0, 5]$ packets per second. Each sensor in $\mathcal{D}\backslash\mathcal{V}$ was assumed to generate a real packet with probability p. The packet transmission rate of sensors in $\mathcal{D}\backslash\mathcal{V}$ was calculated using the methods described in Section 5.2 and Section 5.3. The rate of sensors in $\mathcal{V}\backslash\mathcal{D}$ was selected so as not violate the statistical tests conducted by possible eavesdroppers, independent of the eavesdropper location. This was achieved by considering all eavesdropper locations for which an eavesdropper can observe a distinct rate (based on areas $U_1, U_2, \ldots, U_m$).

Figure 5.7(a) shows the average achievable rate at a sensor $u \in \mathcal{D}\backslash\mathcal{V}$ as a function of $|\mathcal{N}_a|$. Here, $|\mathcal{N}_a|$ denotes the minimum number of CDS sensors that can be overheard within the communication area C_u of sensor u. For each sensor u, the minimum value of $|\mathcal{N}_a|$ was calculated based on the areas $U_1, U_2, \ldots, U_m$ that partition C_u, based on the communication areas of the one-hop neighbors of u that belong to $\mathcal{D}$ (see Section 5.2). We observe that the lowest rate is achieved for $|\mathcal{N}_a| = 1$. A considerable rate increase of around 3 packets per second occurred when $|\mathcal{N}_a|$ increases from 1 to 2.

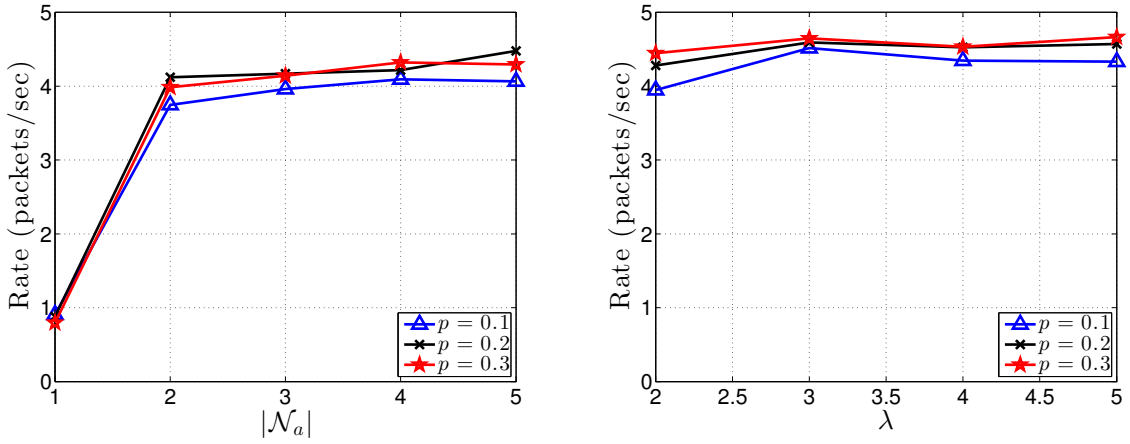


FIGURE 5.7. (a) Average achievable rate in the absence of power control, (b) average achievable rate with power control.

This is because the perceivable distribution change caused by the introduction of r_u^k is smaller when the rate distribution Z_a observed by the eavesdropper under normal conditions (no real traffic from u) is the sum of bogus transmission activities of a larger number of sensors (see Proposition 1). For higher values of $|\mathcal{N}_a|$ we note that the achievable rate does not change significantly, which is expected since the variation of the traffic patterns is greater than the allowed transmission rates. For instance, the variance of distribution Z_a is 4.16, 6.24, 8.32 and 10.4 for $|\mathcal{N}_a| = 2, 3, 4$ and 5, respectively. Moreover, this variation allows us to assign similar rates to co-located sensors transmitting real traffic simultaneously, which is shown when the value of p increases.

To evaluate the achievable rate of sensors that do not belong to the CDS when power control is employed, the sensors of the CDS were assumed to select their power levels at random from set $\{\rho_1, \rho_2, \ldots, \rho_\lambda\}$. The rate selection varied per interval I_k. In Figure 5.7(b), we show the achievable rate as a function of the number of power quantization levels λ. We observe that the average packet rate of a sensor in $\mathcal{D}\backslash\mathcal{V}$

is slightly higher compared with the case of no power control. This is due to the following factors. First, with the introduction of power control, the distribution Z_a of the rate observed by an eavesdropper becomes a function of two variables; the WSN topology and the transmission power adopted by each sensor. This significantly increases the variation of the traffic patterns observed by eavesdroppers, and allows sensors to achieve a higher transmission rate without being detected. Second, since the CDS generation algorithm for the case of power control uses the lowest power level ρ_1 to determine the CDS topology, the CDS size and consequently $|\mathcal{N}_a|$, increase with λ, thus allowing higher rates.

5.5.3 Average Delay

In this set of experiments, we evaluated the average delay of sensors in $\mathcal{D}\backslash\mathcal{V}$. We selected the sensors in $\mathcal{D}\backslash\mathcal{V}$ because they do not transmit bogus traffic, and have to adjust their rate per interval in order to avoid detection. Therefore, they are likely to incur higher delay than CDS sensors. As in the previous scenario, we divided time into one-second intervals. At each interval, CDS sensors selected bogus traffic rates according to an uniform distribution in the interval $(0, 5]$. Sensors in $\mathcal{D}\backslash\mathcal{V}$ selected their rates using the algorithms presented in Section 5.2 and Section 5.3. Once a packet from a sensor in $\mathcal{D}\backslash\mathcal{V}$ was relayed to one of the CDS sensors, it was forwarded to the sink using the shortest path in the CDS and by substituting dummy packets.

We compared the performance of our schemes with the delay of the schemes presented in [50,65], which we refer to as "Basic". In Basic, all sensors transmit bogus traffic at the same average packet rate. To provide a fair comparison, we equated the communication overhead of Basic with that of our schemes by appropriately adjusting the average packet rate of the sensors, when operating under the Basic scheme.

Note that, in our approach, the end-to-end delay has two components. The first component is the number of intervals that a sensor $u \in \mathcal{D}\backslash\mathcal{V}$ has to wait until a

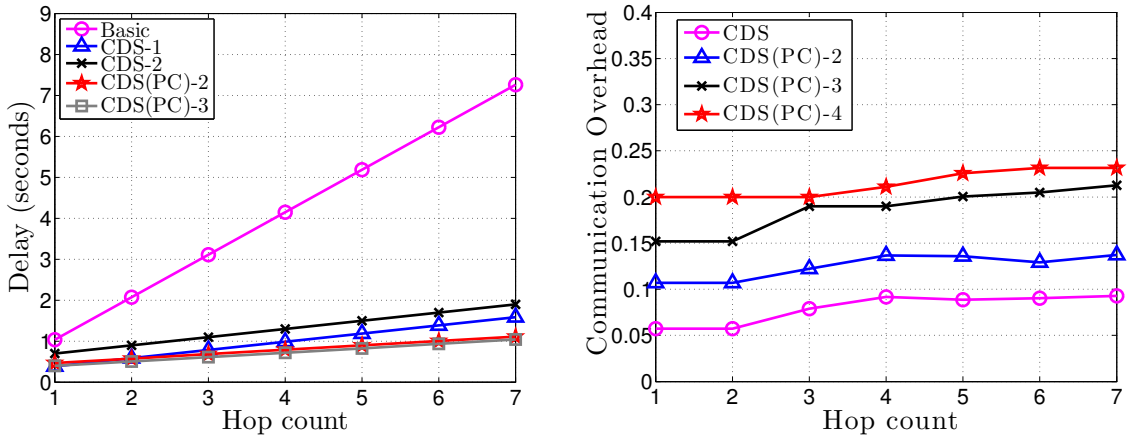


FIGURE 5.8. (a) Average end-to-end delay as a function of the hop count to the sink, (b) average communication overhead normalized over the overhead introduced by the Basic scheme.

rate $r_u^k > 0$ satisfies the (β, ϵ)-*unobservability* criterion. The second component is the waiting time incurred at each hop in the communication path to the sink, due to the application of packet rate control (recall that transmission and propagation delays are ignored by our simulations).

In Figure 5.8(a), we show the end-to-end delay for sensors in $\mathcal{V}\backslash\mathcal{D}$, as a function of the hop count to the sink. In this figure, we labeled as CDS-$|\mathcal{N}_a|$ the delay incurred by nodes that do not apply power control and have a minimum value of $|\mathcal{N}_a| = i$. We further labeled CDS(PC)-λ, the delay incurred by nodes that apply power control with λ power levels. We observe that for the same communication overhead, our scheme requires significantly less time to deliver a packet to the sink. This is because once a packet is received by the CDS, it is relayed at a faster rate compared to the Basic scheme, in which all sensors operate at a lower rate in order to meet the same communication budget. We further observe that the schemes that use power control achieve the lowest delay. This is expected based on the results of Figure 5.7(a) and Figure 5.7(b) that show that the power control scheme achieves a higher packet rate

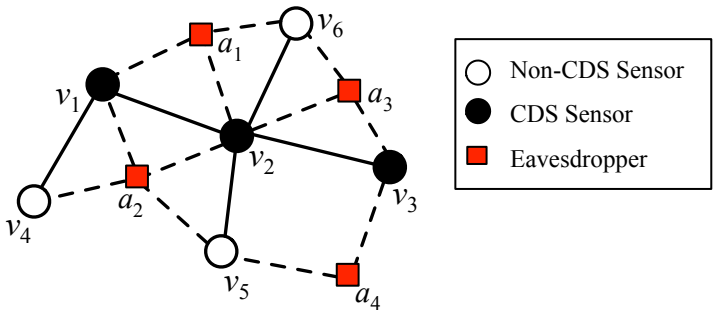


FIGURE 5.9. A WSN operating in the presence of four eavesdroppers.

for nodes that do not belong to the CDS.

5.5.4 Communication Overhead

In this set of experiments, we compared the communication overhead of our schemes with the overhead of the Basic scheme [1, 50, 65]. We fixed the packet rate of every dummy source to 1 packet\sec. We then generated events at random WSN locations. Each event triggered the transmission of one real packet from the sensor closest to the event. For the Basic scheme, the shortest paths from the source to the sink were followed. For our construction, real packets had to first be relayed to one of the CDS nodes, before they were routed to the sink. The route to the sink includes only CDS nodes and can be considerable longer than the shortest path. For each scheme, we measured the overhead as the number of bogus packets that are transmitted by any bogus traffic source, until the real packet is delivered to the sink.

In Figure 5.8(b), we show the average communication overhead normalized over the overhead of the Basic scheme as a function of the hop count of the shortest path to the sink. Similarly to Figure 5.8(a), we label as CDS and CDS(PC)-λ the overhead introduced when nodes do not apply power control, and when nodes apply power control with λ power levels, respectively. We observe that our scheme requires considerably lower overhead to deliver a real packet to the sink compared to the Basic

scheme. For instance, when no power control is applied, the communication overhead introduced is reduced to as low as 5.5% relative to the Basic scheme. This reduction is due to the size of the CDS used to inject bogus packets, relative to the network size. Also, we observe that overhead increases with the number of power levels used to apply power control. This is because the CDS size increases with λ.

Figure 5.8(b) shows that the overhead increases with the hop count from the sink. This increase is due to the route inflation caused by the use of the CDS as a routing backbone. For shorter hop counts, the CDS path to the sink has a similar length to the shortest path. As we move further away from the sink, the route expansion factor (ratio of CDS path vs. the shortest path) increases leading to less efficient routes that prolong the real packet relay until it reaches the sink. The minimization of the CDS size due to Algorithm 5, provides no guarantees about the optimal path length from a source to the sink. The overhead increase is more substantial for the scheme without power control, due to the minimal nature of the CDS produced by Algorithm 5. Finally, the overhead is almost doubled when the hop count increases from 1 to 7. This is due to the increase of the deviation between the CDS paths and the shortest paths when the distance between the source and sink increases.

5.5.5 Detection of Real Transmissions under Eavesdropper Collusion

In the last set of experiments, we evaluated the resistance of our schemes to collusion. Specifically, we measured the fraction of sensors in $\mathcal{V}\backslash\mathcal{D}$ that are detected while transmitting real traffic, when traffic observations are collectively analyzed at a central server. We uniformly deployed a number of eavesdroppers within the WSN deployment area. The average node degree for the WSN was set to $\delta = 50$, while we varied the average node degree for the eavesdropping network form 10 to 50.

We considered two detection scenarios. In the first scenario, any eavesdroppers that observe the same traffic pattern for the majority of time overhear the same

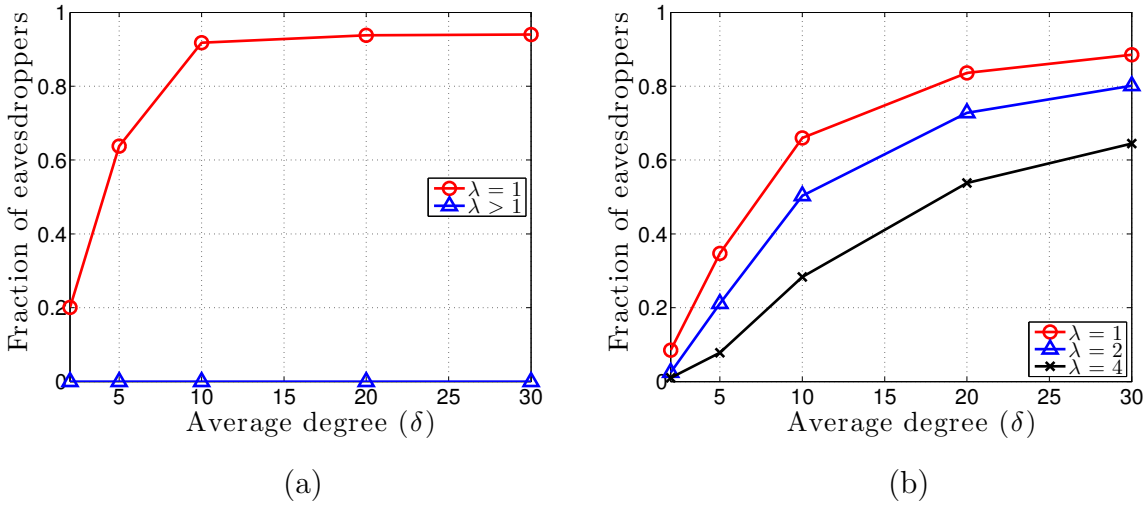


FIGURE 5.10. (a) Fraction of eavesdroppers detecting real transmissions using $\mathcal{H}^k_{e_1} = \mathcal{H}^k_{e_2}$, (b) fraction of eavesdroppers detecting real transmissions using $\mathcal{H}^k_{e_1} \cap \mathcal{H}^k_{e_2} = \emptyset$.

subset of CDS sensors. As a result, they forward the same set of packet hashes $\mathcal{H}^k_{a_i}$ to the central server. When a sensor that does not belong to the CDS performs a real transmission overheard only be one of the eavesdroppers, say a_1, this transmission is immediately detectable because the traffic patterns forwarded by a_1 to the central server are no longer identical to those coming from the rest of the eavesdroppers. This scenario is illustrated in the topology of Figure 5.9, in which a_1 and a_2 observe the bogus traffic from CDS sensors v_1 and v_2. When v_4 (not in the CDS) transmits a real packet to the CRS, the central server can detect it as a real one because its hash will be included in the report from a_2, but not a_1. Similar detections can occur for the real transmissions of sensors v_5 and v_6. In Figure 5.10(a), we show the fraction of sensors in $\mathcal{V}\backslash\mathcal{D}$ that can be detected while transmitting real traffic, using this analysis. The fraction is plotted as a function of the average degree of the adversarial network. We show that in the absence of power control, 20% of real transmissions can be identified for $\delta = 2$. This number rapidly increases as the degree of the adversarial network increases as well, which is expected since more observations are obtained by

the adversary. Also, this fraction drops to zero for any value of $\lambda > 1$, which we expect as the CDS nodes observed by the eavesdroppers change with the variation of the power level.

In the second detection scenario, any eavesdroppers that report disjoint traffic patterns, observe bogus traffic from disjoint subsets of CDS sensors. As a result, these eavesdroppers forward distinct set of packet hashes to the central server. When a sensor that does not belong to the CDS performs a real transmission and is overheard by two or more eavesdroppers that normally report disjoint sets, the packet hash of the real transmission will appear on the packet hash reports of those eavesdroppers. Therefore, the central server can identify the corresponding packet as a real transmission. For instance, eavesdroppers a_2 and a_4 in Figure 5.9, overhear sensors $\{v_1, v_2\}$ and $\{v_3\}$, respectively. When v_5 transmits a real packet, both a_2 and a_4 overhear that transmission. The central server can then identify the transmission of v_5 by finding a common packet hash on the reports of a_2 and a_4. Figure 5.10(b) plots the fraction of sensors that can be detected in this scenario. We observe that more than 8% of the deployed network can be detected when power control is not applied. However, with the introduction of power control, this fraction is reduced in to 2% for $\lambda = 2$ and 1% for $\lambda = 4$, for $\delta = 2$. This fraction increases with the increase of the number of deployed eavesdroppers, as the adversary obtains more observations that can be used in the centralized analysis.

5.6 Summary of Contributions

We addressed the problem of energy-efficient contextual information privacy in WSNs. Our traffic normalization techniques are based on the generation of bogus traffic from a subset of dummy sources. Our methods rely on the computation of a minimum connected dominating set (MCDS) that covers the WSN deployment area. This set is responsible for randomizing the traffic patterns intercepted by arbitrarily located

eavesdroppers. We further applied power control to randomize the traffic patterns observed by colluding eavesdroppers. We showed that event unobservability can be satisfied by randomizing the packet rates of the bogus traffic sources and regulating the rates of real ones. Our simulations verified that a significantly smaller number of fake sources is necessary to achieve event unobservability.

Chapter 6

Traffic Normalization beyond SSA

As shown in Chapter 4, SSA methods can actually leak contextual information if the adversary applies specialized statistical tests designed to detect anomalous data points. To eliminate this information leakage, we propose traffic normalization techniques that reduce the number of dummy transmissions without relying on statistical anonymity. Our traffic normalization techniques protect the event location, its occurrence time, and the sink location privacy.

In the proposed solution, the WSN is partitioned into connected dominating sets (CDSs) that are activated in a round-robin fashion. Compared to prior art, our method considerably reduces the communication overhead and end-to-end delay of real packets by limiting the injection of dummy traffic. We partition the network in two types of CDSs; minimum connected dominating sets (MCDSs) and MCDSs with shortest paths to the sink (SS-MCDSs). The former one includes sets of minimum size, which aims at minimizing the amount of dummy traffic introduced. The latter one is introduced to include the shortest paths from any sensor in the CDS to the sink, designed the reduce the end-to-end delay of real packets. We characterize the algorithmic complexity for partitioning the network into MCDSs and SS-MCDSs, and develop efficient heuristics.

We note that even when sensors transmit dummy (or real) packets in an uncoordinated fashion, the end-to-end delay for reporting a real event can increase significantly. To reduce this delay, we propose a rate control scheme, that loosely coordinates sensor transmission over multi-hop paths without revealing real traffic patterns or the traffic directionality.

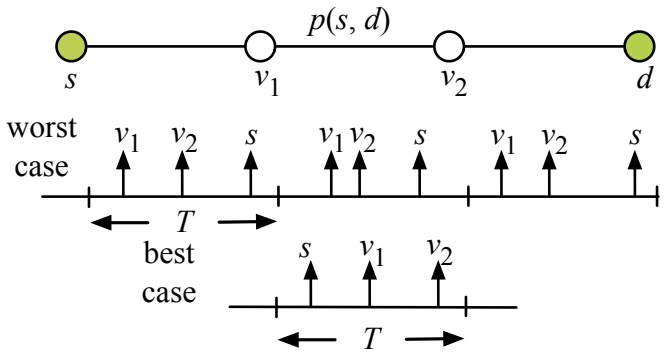


FIGURE 6.1. Randomization of traffic patterns.

6.1 Efficient Traffic Normalization

To counter a global eavesdropper that uses traffic analysis techniques such those presented in Chapter 3, most existing solutions introduce bogus traffic at every sensor [5, 50, 65, 80]. This is because the eavesdroppers' locations are unknown and any transmission can be potentially intercepted by the adversary. Hence, the traffic patterns of each sensor that can report an event are normalized. This approach significantly increases the communication overhead.

Moreover, the normalized traffic patterns can lead to the accumulation of packet delay on a per-hop basis. For instance, consider the path $p(s, d)$ shown in Figure 6.1. Assume that the traffic rate of every node is normalized to one packet per T units of time. The worst-case end-to-end delay for a real packet is equal to $|p(s, d)|T$, where $|p(s, d)|$ is the path length in hops. This delay occurs when downstream nodes relative to d transmit earlier than upstream ones. In the best case, the end-to-end delay is equal to T. The best-case delay occurs when upstream nodes transmit earlier than downstream ones. Formally, for randomly selected transmission times within T, Proposition 6 shows that a packet will be forwarded over less than two hops per T, on average.

Proposition 6. *When sensors transmit one packet at a random time within an in-*

terval of T, the average number of hops that a packet can traverse per T is 1.7183.

Proof. Consider a source v with a hop-count $\eta = |p(v,u)|$ to the destination u. Nodes in $p(v,u)$ randomly select their transmission times within an interval T. Let the randomly selected transmission times be denoted by $t_1, t_2, \ldots, t_\eta$. We compute the average number of hops (H) traversed over T by a packet m originating at v_1, using a combinatorial approach. The value of H depends on the arrangement order of $t_1, t_2, \ldots, t_\eta$. All possible arrangements $n_\eta!$ occur with equal probability, as the transmission times of each sensor are randomly and independently selected within T.. A packet traverses η hops during T, only if $t_1 < t_2 < \ldots < t_\eta$. This events occurs with probability,

$$\Pr(H = \eta) = \frac{1}{\eta!}.$$

Similarly, to find the probability of traversing $\eta - 1$ hops, we consider the number of arrangements that satisfy $t_1 < t_2 < \ldots < t_{\eta-1}$, but not $t_{\eta-1} < t_\eta$. Thus, we obtain,

$$\Pr(H = \eta - 1) = \frac{1}{\eta!}\left(\frac{\eta!}{(\eta-1)!} - 1\right).$$

For an arbitrary number of hops $x \leq \eta - 1$ we get,

$$\begin{aligned}\Pr(H = x) &= \frac{1}{\eta!}\left(\frac{\eta!}{x!} - \frac{\eta!}{(x+1)!}\right) \\ &= \frac{x}{(x+1)!}.\end{aligned}$$

Computing the expectation of H

$$\begin{aligned}E(H) &= \frac{\eta}{\eta!} + \sum_{x=1}^{\eta-1} \frac{x^2}{(x+1)!} \\ &= \frac{1}{(\eta-1)!} + \sum_{x=1}^{\eta-1} \frac{x^2}{(x+1)!}.\end{aligned}$$

By applying the ratio test of series on the second term of $E(H)$, we obtain

$$\begin{aligned}\lim_{x\to\infty} \frac{\frac{(x+1)^2}{(x+2)!}}{\frac{x^2}{(x+1)!}} &= \lim_{x\to\infty} \frac{(x+1)^2(x+1)!}{x^2(x+2)!} \\ &= \lim_{x\to\infty} \frac{x^2+2x+1}{x^3+2} \\ &\leq 1,\end{aligned}$$

which proves that $E(H)$ is a converging series. Finally, we compute the convergence value of $E(H)$ via numerical analysis for all $\eta \geq 1$,

$$E(H) = 1.7183.$$

□

To address the inefficiencies of prior traffic normalization methods, we first reduce the number of bogus traffic sources required to normalize the traffic patterns observed in the network. To do so, we divide time into epochs and partition the set of sensors $\mathcal{V}$ into subsets. Only one subset is active at a given epoch, and subsets are periodically rotated in a round-robin fashion. A node is allowed to send traffic (bogus or real) only if a subset it belongs to is active. Each subset forms a subgraph designed to satisfy the following properties: (a) each subgraph is connected, (b) each subgraph can deliver packets to any vertex of the original graph, and (c) the sizes of the subgraph are minimized.

Properties (a) and (b) guarantee that an active subgraph can deliver a real packet to the sink. Moreover, the sink location remains hidden because all sensors can be reached by the subgraph. Finally, property (c) minimizes the number of active bogus traffic sources per epoch required to satisfy properties (a) and (b), thus reducing the communication overhead for hiding traffic patterns. For instance, Figure 6.2 shows the

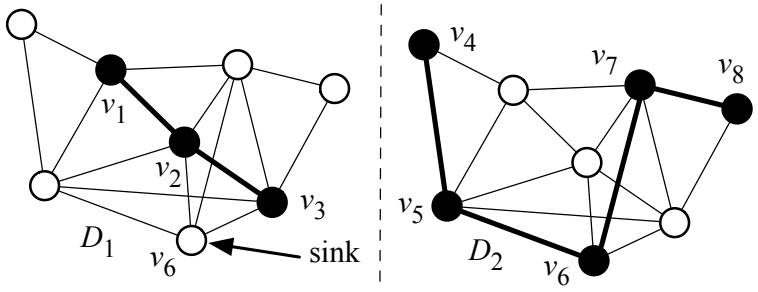


FIGURE 6.2. Partition of $\mathcal{V}$ to two subgraphs $\mathcal{D}_1$ and $\mathcal{D}_2$.

partition of a small WSN into two subsets $\mathcal{D}_1 = \{v_1, v_2, v_3\}$ and $\mathcal{D}_2 = \{v_4, v_5, v_6, v_7, v_8\}$ that satisfy properties (a)-(c). Nodes in both $\mathcal{D}_1$ and $\mathcal{D}_2$ can deliver a packet to any node in $\mathcal{V}$. When $\mathcal{D}_1$ becomes active, v_1, v_2 and v_3 have a routing path to the sink v_6.

To further reduce the end-to-end delay, we loosely coordinate sensor transmissions based on tree structures. Node coordination expedites the relaying of real packets over multiple hops while hiding contextual attributes. Our traffic normalization scheme consist of a network partition and a transmission coordination phase.

6.1.1 Network Partition Phase

In the network partition phase, we partition $\mathcal{V}$ into subsets $\{\mathcal{D}_1, \ldots, \mathcal{D}_z\}$. Each subset remains active for one epoch. Subsets are periodically rotated in a round-robin fashion. Sensors of an active subset transmit dummy packets at a fixed packet rate, following a pre-assigned schedule. The dummy packets are replaced with real ones, when a sensor of an active subset needs to report an event. To satisfy design properties (a)-(c), we reduce the problem of partitioning $\mathcal{V}$ to the problem of finding a partition of connected dominating sets (CDSs).

The partition of $\mathcal{V}$ to disjoint MCDSs satisfies properties (a)-(c). Property (a) is satisfied, as the set of MCDSs spans $\mathcal{V}$. Hence, each sensor belongs to at least one D_j, and can transmit real traffic when D_j becomes active. By design, the traffic pattern

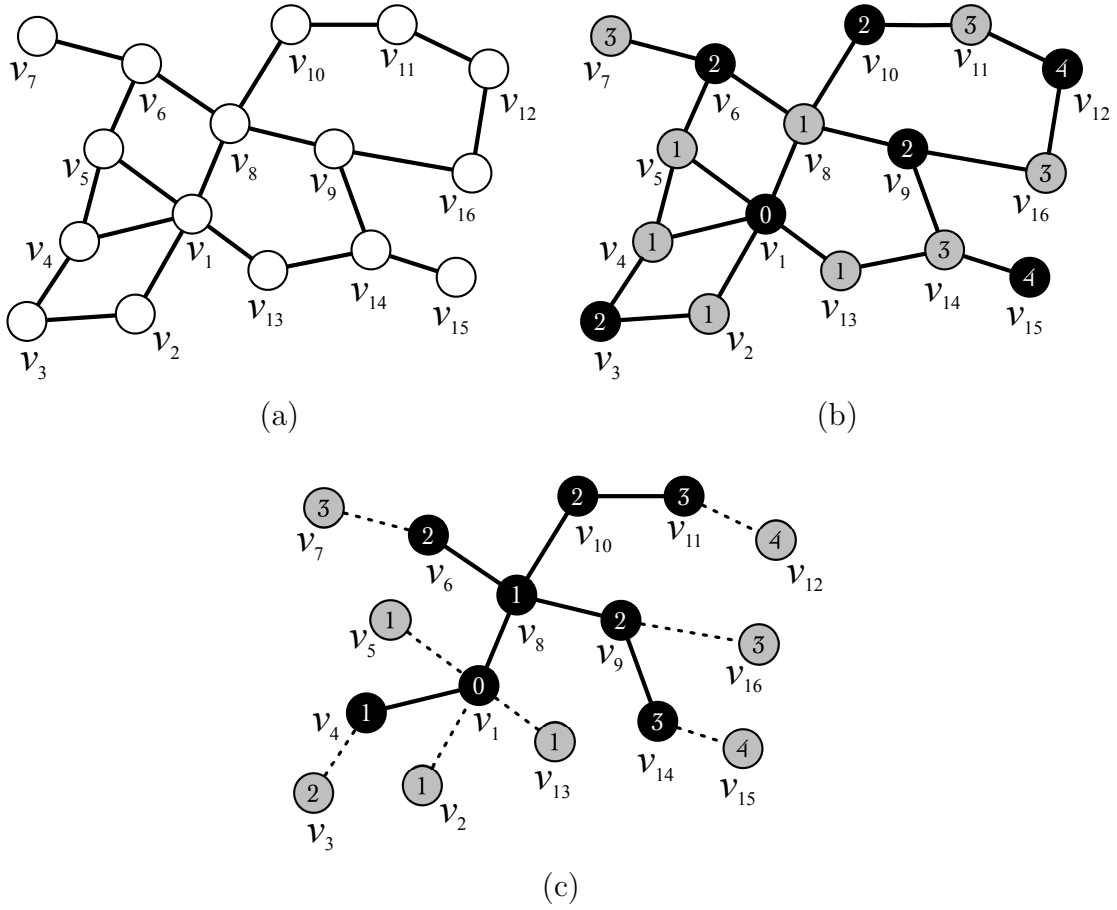


FIGURE 6.3. (a) Original WSN graph, (b) a DS generated in Stage 1, (c) a MCDS approximation generated in Stage 2.

of an active sensor is not altered when dummy packets are replaced by real ones. For property (b), a CDS ensures that any sensor in $\mathcal{V}$ is either part of D_j or within one hop from a sensor in D_j. Moreover D_j forms a connected graph. Hence, a real packet transmitted by a sensor in D_j can be forwarded to any sensor in $\mathcal{V}$ using only D_j. Finally by definition, an MCDS minimizes the size of each subgraph.

However, we note that MCDS graphs do not necessarily include shortest paths to the sink. This could lead to an increase in the end-to-end packet delay. For time-critical sensor applications, the construction of CDSs that contain the shortest paths from any CDS sensor to the sink may be desirable. In the following section, we

develop algorithms for both CDS designs.

6.1.2 Network Partition–Sets of Minimum Size

We first consider the partition of $\mathcal{V}$ into MCDSs. Such a partition is not guaranteed to exist for arbitrary graph topologies (this can be easily shown for a topology in which the minimum vertex cut is equal to one). Moreover, the problem of computing a single MCDS is NP-complete [32]. In the absence of a guaranteed MCDSs partition and a polynomial-time algorithm to construct MCDSs, we relax the partition requirement and allow nodes to be part of more than one MCDSs. We denote the *appearance frequency* of node v to MCDSs as $f(v)$.

We propose a heuristic algorithm that computes an approximation of $\mathcal{V}$'s partition. Our heuristic to balances between the appearance frequency, the number of MCDSs that span $\mathcal{V}$, and the MCDS size. These parameters are used to control the tradeoff between the end-to-end delay and the communication overhead. Note that, if $\mathcal{V}$ is partitioned to a large number of MCDSs, the number of epochs until each MCDS becomes active increases, thus increasing the end-to-end delay. However, a small number of MCDSs results in larger communication overhead because more sensors are active per epoch.

Algorithm 7: MCDS approximation–We generate a CDS partition in three stages. In stage 1, we construct a minimum DS graph based on a well-known DS approximation (the problem of computing a minimum DS is also NP-complete [32]). In stage 2, we connect the DS to generate a CDS. The nodes selected to connect the DS maximize the number of DS nodes that are interconnected in a greedy fashion to reduce the CDS size. In stage 3, we repeat the algorithms of stage 1 and 2 to obtain a partition of $\mathcal{V}$ to CDSs.

For each $v \in \mathcal{V}$, let $m(v)$ be a marker, which can take the values *white*, *black*, or *gray*. $\mathcal{N}_v^k$ and $[\mathcal{N}_v^k] = \mathcal{N}_v^k \cup v$ are v's open and closed k-hop neighborhoods,

respectively. Let $\delta(v) = |\mathcal{N}_v|$ be the degree of v, and $\delta^*(v)$ the effective degree of v. The latter is defined as the number of v's neighbors marked as *white*. Let $r(v)$ be the rank of v, defined as the order that v changed its marker relative to a leader node. Finally, $\rho(v)$ is the dominator node of v. Initially, the appearance frequency of every node is set to $f(v) = 0$. We base the marking process for generating a DS to the algorithm presented in [38].

Stage 1: DS generation

Step 1: Each $v \in \mathcal{V}$ initializes and broadcasts the values of $m(v) = \textit{white}$, $\delta^*(v) = \delta(v)$, and $r(v) = 0$.

Step 2: A randomly chosen leader s sets $m(s) = \textit{black}$ and broadcasts $m(s), r(s)$, and $f(s)$ to $\mathcal{N}_s$.

Step 3: A *white* node u receiving $m(v) = \textit{black}$ is dominated by v, sets $m(u) = \textit{gray}$, $\rho(u) = v$, and $r(u) = r(v) + 1$. It then broadcasts $m(u)$ and $r(u)$ to $\mathcal{N}_u$.

Step 4: A *white* node v receiving $m(u) = \textit{gray}$ from $u \in \mathcal{N}_v$, decreases $\delta^*(v)$ by one, updates $r(v) = r(u) + 1$ if $r(v) \leq r(u)$, and broadcasts $\delta^*(v)$ and $r(v)$ to $\mathcal{N}_v$.

Step 5: A *white* node v changes $m(v)$ to *black*, if

$$v = \underset{u \in [\mathcal{N}_v]}{\arg\max} \left\{ \frac{\delta^*(u)}{\delta^*_{\max}(v)} \times \frac{1}{f(u)+1} \right\}, \tag{6.1}$$

where $\delta^*_{\max}(v) = \max_{u \in [\mathcal{N}_v]} \delta^*(u)$. Ties are broken arbitrarily. Node v becomes a "dominator" and broadcasts $m(v) = \textit{black}$ and $r(v)$ to $\mathcal{N}_u$.

Step 6: Repeat Steps 3-5 until all nodes are marked as *black* (dominator) or *gray* (dominated).

With the termination of stage 1, the set of *black* nodes forms a DS. Note that, the domination metric in Eq. (6.1) tradeoffs between two competing factors: (a) the CDS size and (b) the number of CDSs in the partition of $\mathcal{V}$. By maximizing $\frac{\delta^*(v)}{\delta^*_{\max}(v)}$, we include in the DS nodes that dominate the largest fraction of their neighbors within

their closed neighborhood. Therefore, the size of the DS is reduced in a greedy fashion. On the other hand, the factor $\frac{1}{f(v)+1}$ favors the selection of nodes with the lowest appearance frequency. Because $\frac{\delta(v)}{\delta_{\max}(v)} \leq 1$, the factor $\frac{1}{f(v)+1}$ guarantees that every node will be part of a CDS, when the Algorithm 4 is iteratively applied to obtain a partition, and thus our algorithm will terminate.

In Figure 6.3(b), we show the DS generated during Stage 1 for the graph of Figure 6.3(a). In Step 1, we initialize $m(v) = \textit{white}$, $f(v) = 0, \forall v \in \mathcal{V}$, and $\delta^*(v_1) = 6, \delta^*(v_2) = 3$, etc. In Step 2, v_1 is chosen as the leader, sets $m(v_1) = \textit{black}$, and broadcasts $m(v_1), r(v_1)$, and $f(v_1)$. In Step 3, nodes v_2, v_4, v_5, v_8, and v_{13} change their markers to *gray*, set $r(v) = 1$ and $\rho(v) = v_1$, and send $m(v)$ and $r(v)$ to their neighbors. In Step 4, v_3 updates and broadcasts $r(v_3) = 2$ and $\delta^*(v_3) = 0$. A similar process is applied at v_6, v_9, v_{10} and v_{14}. In Step 5, v_3, v_6, v_9, and v_{10} are marked as *black*. The process is repeated until v_{12} and v_{15} become *black*. At the end, nodes $v_1, v_3, v_6, v_9, v_{10}, v_{12}$ and v_{15} form a DS. The color and rank of the nodes is shown in Figure 6.3(b).

In stage 2, we approximate the MCDS by adding *gray* nodes that "bridge" the most *black* nodes. Let the number of v's higher ranked *black* neighbors be $b(v)$.

Stage 2: Approximation of the MCDS

Step 1: Each *gray* node $v \in \mathcal{V}$ broadcasts $b(v)$ to $\mathcal{N}_v^2$

Step 2: Starting with the leader's neighborhood, a *gray* node v becomes *black* if,

$$v = \arg\max_{u \in Z} \left\{ \frac{b(u)}{b_{\max}(v)} \times \frac{1}{f(u)+1} \right\}, \tag{6.2}$$

where $Z = \{u : u \in [\mathcal{N}_v^2], r(u) = r(v)\}$, $b_{\max}(v) = \max_{\{u \in [\mathcal{N}_v^2], m(u) = \textit{gray}\}} b(v)$, and $b(u), b_{max}(v) > 0$. Node v broadcasts $m(v)$ in $\mathcal{N}_v^2$. Ties are broken arbitrarily.

Step 3: A node $w \in \mathcal{N}_u$ overhearing the change of u's marker from *gray* to *black*, with $m(w) = \textit{black}$ and $r(w) = r(u) + 1$ sets and broadcasts $\rho(w) = u$ to $\mathcal{N}_w$.

Step 4: A *gray* node u overhearing $\rho(w)$ from a *black* node w with $r(u) = r(w) + 1$

broadcasts $b(u) = b(u) - 1$ to $\mathcal{N}_u^2$.

Step 5: Steps 2-4 are iterated for all *black* nodes in the DS, until all *gray* nodes have a $b(v)$ value equal to zero.

Step 6 (Pruning): If a *black* node v with $f(v) > 0$ does not dominate at least one *gray*, it changes $m(v) = gray$.

After the execution of Step 5, each *gray* node has a $b(v) = 0$, thus all *black* nodes of Stage 1 are dominated. Moreover, these nodes are dominated by a *gray* node of lower rank, that turns *black* in Step 3. Since the process is initiated in the leader's neighborhood, with the change of a *gray* node into *black,* each dominated *black* node is connected to the leader. Therefore, with the termination of stage 2, all *black* nodes have a path to the leader, forming a CDS. Similarly to Stage 1, the metric used in Eq. 6.2 tradeoffs between the CDS size and the number of CDS in the partition. Maximizing the fraction $\frac{b(v)}{b_{\max}(v)}$ benefits the nodes that connect the highest number of *black* nodes in the DS, while $\frac{1}{f(v)+1}$ favors the selection of nodes with the lowest appearance frequency. Finally, in the pruning step, we eliminate all black nodes that do not dominate any gray nodes, provided that these black nodes have appeared at least at one CDS ($f(v) > 0$).

Figure 6.3(c) shows the CDS generated in stage 2 from the DS of Figure 6.3(b). In Step 1, *gray* nodes broadcast $b(v_2) = 1, b(v_4) = 1$ and so on. In Step 2, node v_8 broadcasts $m(v_8) = black$, because it has the highest value of $b(v)$ in Z. In Step 3, nodes v_6, v_9, and v_{10} broadcast $\rho(v) = v_8$. In Step 4, node v_5 receives $\rho(v_6)$ broadcasts $b(v_5) = 0$. In Step 5, the process is repeated until v_4, v_{11}, and v_{14} become *black* to dominate v_3, v_{12}, and v_{15}, respectively. In Step 6, nodes v_{12} and v_{15} can be pruned (become *gray*, as they do not dominate any node. In the final stage, the MCDS generation process is iteratively applied until all sensors become part of one MCDS.

Stage 3: MCDS Update

Step 1: Increment $f(v)$ by one unit for all nodes in $\mathcal{D}_j$.

Step 2: Repeat Stages 1 and 2 until $f(v) > 0,\ \forall v \in \mathcal{V}$.

Proposition 7. *Algorithm 7 terminates in at most* $\delta_{\max} + 1$ *iterations, where* $\delta_{\max} = \max\{\delta(v) : \forall v \in \mathcal{V}\}$.

Proof. By contradiction. Assume that Algorithm 7 does not terminate. That is, there exist a $v \in \mathcal{V}$ that is *gray* for each iteration of Algorithm 7. Based on Step 3 of Stage 1, a node u becomes *black* if

$$v = \arg\max_{u \in [\mathcal{N}_v^2]} \left\{ \frac{\delta^*(u)}{\delta^*_{\max}(v)} \times \frac{1}{f(u)+1} \right\}.$$

In the worst-case scenario, v is a leaf node that cannot become *black* during Stage 2. Each time Stage 1 is executed, v becomes *gray* and is dominated by a node $u \in \mathcal{N}_v$ if,

$$\frac{\delta^*(v)}{\delta^*_{\max}(v)} \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{1}{f(u)+1},$$

or,

$$\delta^*(v) \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{\delta^*_{\max}(v)}{f(u)+1},$$

where $\delta^*_{\max}(v)$ is the maximum degree in $\mathcal{N}_v^2$ and $\delta^*_{\max}(u)$ is the maximum degree in $\mathcal{N}_u^2$. Note that because $\mathcal{G}$ is connected, $\delta^*(v) \geq 1$, and

$$1 \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)}.$$

Hence,

$$1 \leq \frac{\delta^*_{\max}(v)}{f(u)+1}.$$

Because we assumed Algorithm 7 does not terminate and $\delta^*_{\max}(v)$ is finite, at some iteration n' of Stage 3, $f(u) = \delta^*_{\max}(v)$ for all $u \in \mathcal{N}_v$. Therefore, during iteration $n'+1$ we have $\frac{\delta^*_{\max}(v)}{f(u)+1} < 1$ and,

$$\delta^*(v) \geq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{\delta^*_{\max}(v)}{f(u)+1}.$$

In this case, v becomes *black* during Stage 1, contradicting the assumption that v is always *gray*. Moreover, $n' = \delta^*(v) \times \delta^*_{\max}(v)$. Also, note that in the worst case $\delta^*(v) = 1$ and $\delta^*_{\max}(v) = \delta_{\max}$, where $\delta_{\max}$ is the maximum degree in the network. Thus, $n' = \delta_{\max}$ and the maximum number of iterations of Stage 3 is $\delta_{\max} + 1$. □

6.1.3 Network Partition–CDSs with shortest paths

We now consider the problem of partitioning $\mathcal{V}$ to CDSs that contain the shortest paths from any CDS node to the sink. We call this CDS type as a Single-destination Shortest-path CDS (SS-CDS), defined it as follows.

Definition 12 (Single-destination Shortest-path CDS)**.** *Let $p(s, \mu)$ denote the shortest path between s and μ in $\mathcal{G}$. Let also μ be a single destination (sink). Set $\mathcal{D} \subseteq \mathcal{V}$ is a single-destination shortest-path CDS if for each $s \in \mathcal{D}$, $p(s, \mu)$ belongs to $\mathcal{D}$. The set $\mathcal{D}$ with the smallest cardinality is called a single-destination shortest-path minimum connected dominating set (SS-MCDS).*

A partition of $\mathcal{V}$ to SS-MCDSs is not guaranteed to exist for arbitrary graph topologies. Moreover, we show that the problem of constructing SS-MCDSs is NP-complete by reducing it to the Minimum Shortest-path Steiner arborescence problem.

Proposition 8. *The problem of finding an SS-MCDS in arbitrary graphs is NP-complete.*

Proof. We first show that the problem of computing a single SS-MCDS is NP-complete. To do so, we prove that SS-MCDS is both in NP and it is at least as hard as the Minimum Shortest-path Steiner arborescence (MSPSA) problem, that is known to be NP-complete [15]. We first define the MSPSA.

Definition 13 (Minimum Shortest-path Steiner arborescence)**.** *Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$ with positive edge weights, a target set $\mathcal{T} \subseteq \mathcal{V}$ and a unique root node $\mu \in \mathcal{T}$, a* shortest path Steiner arborescence *S is a Steiner tree rooted at μ, spanning all vertices in*

$\mathcal{T}$ such that each path $p(u,\mu)$ is a shortest path in $\mathcal{G}(S,\mathcal{E}(S))$, for all $u \in \mathcal{T}$. The arborescence of smallest cardinality is called minimum shortest-path Steiner arborescence (MSPSA) [17].

The following verifier for the SS-MCDS problem runs in polynomial time in the size of the input SS-MCDS ($\mathcal{D}$). In the verifier, $p(u,\mu)$ denotes the shortest path between u and the root node μ in $\mathcal{G}(\mathcal{V},\mathcal{E}(\mathcal{V}))$, while $p_{\mathcal{D}}(u,\mu)$ denotes the shortest path between u and μ in $\mathcal{G}(\mathcal{D},\mathcal{E}(\mathcal{D}))$.

SS-MCDS Verifier

Input: $\mathcal{G}(\mathcal{V},\mathcal{E}(\mathcal{V})), \mathcal{D}$, and $p(v,\mu)$ for all $v \in \mathcal{V}$.

Execution: If the following are true accept, else reject:

1. For all $v \in \mathcal{V}$, either v is in $\mathcal{D}$, or there exist a $u \in \mathcal{D}$ for which $u \in \mathcal{N}(v)$
2. Nodes in $\mathcal{D}$ form a connected subgraph, if so obtain $p_{\mathcal{D}}(u,\mu)$ for all u in $\mathcal{D}$
3. For all $u \in \mathcal{D}$, $|p(u,\mu)| = |p_{\mathcal{D}}(u,\mu)|$

In Step 1, the SS-MCDS verifier checks whether $\mathcal{D}$ satisfies the DS property. If v belongs to $\mathcal{D}$, the check requires only one computation. However, in the worst case, v is not in $\mathcal{D}$ and the algorithm goes through v's 1-hop neighborhood looking for a $u \in \mathcal{D}$. Since the maximum size of $\mathcal{N}(v)$ is $|\mathcal{V}|-1$, the maximum cost of this operation is $O(|\mathcal{V}|)$. Therefore, the cost of Step 1 is $O(|\mathcal{V}|^2)$ or $O(|\mathcal{D}|^2)$ (since $|V| = O(|\mathcal{D}|)$). In Step 2, connectivity is tested by the Floyd and Warshall's Algorithm, which is of complexity $O(|\mathcal{D}|^3)$ [17]. This algorithm also outputs the shortest paths, $p_{\mathcal{D}}(u,\mu)$, between $u \in \mathcal{D}$ and μ, used in Step 3. Finally, in Step 3, each node v compares the lengths of $p(v,\mu)$ and $p_{\mathcal{D}}(v,\mu)$. The cost of this operation is $O(|\mathcal{D}|)$. Thus, the total cost of the the SS-MCDS Verifier is $O(|\mathcal{D}|^3)$. As the SS-MCDS Verifier runs in polynomial time, the SS-MCDS problem is in NP.

We now show that the SS-MCDS problem is NP-Hard. We first prove that MSPSA $\leq_P$ SS-MCDS. For the MSPSA problem, we define the target set as,

$$\mathcal{T} = \{v : |p(v,\mu)| \geq |p(u,\mu)|, \forall u \in \mathcal{N}_v\} \cup \{\mu\}.$$

Set $\mathcal{T}$ contains all leaf vertices of $\mathcal{G}$ (i.e., the set of nodes for which all their neighbors are closer to the origin μ) plus μ. Let function f take input graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$, with an edge cost $w(u,v) = 1, \forall (u,v) \in \mathcal{E}(\mathcal{V})$, and an origin vertex μ. Function f constructs $\mathcal{G}'(\mathcal{V}', \mathcal{E}(V'))$ as follows:

- For each $v \in \mathcal{T}$, define a virtual node v'. Assign all nodes v' to set $\mathcal{X}$, and make

$$\mathcal{V}' = \mathcal{V} \cup \mathcal{X},$$

- Connect each v to the corresponding v' and set the edge cost $c(v', v) = 0$. Edges $(v, v'), \forall v \in \mathcal{V}$ form edge set $\mathcal{E}_1$.

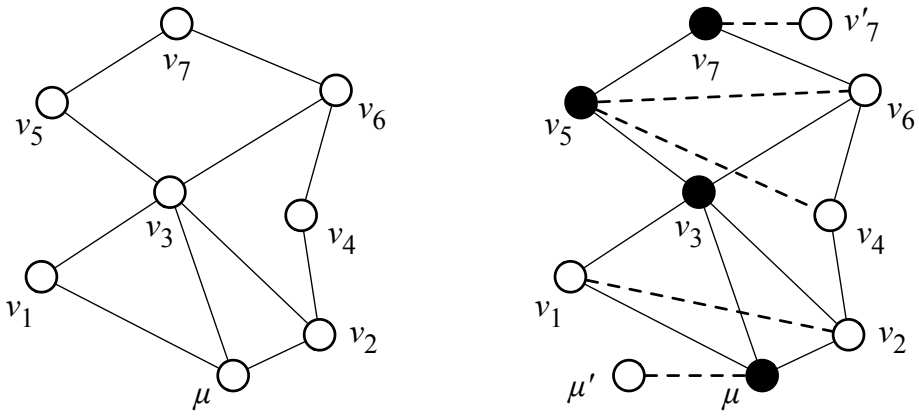
- Connect each u and v in $\mathcal{V}$ with $|p(u,\mu)| = |p(v,\mu)|$ and $(u,v) \notin \mathcal{E}(\mathcal{V})$, and set the edge cost to $c(u,v) = 1$. These edges form set $\mathcal{E}_2$.

- Define $\mathcal{E}(\mathcal{V}')$ as,

$$\mathcal{E}(\mathcal{V}') = \mathcal{E}(\mathcal{V}) \cup \mathcal{E}_1 \cup \mathcal{E}_2.$$

To demonstrate the construction of $\mathcal{G}'$, consider the graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$ presented in Figure 6.4(a). Note that, the only leaf node is v_7, which makes $\mathcal{T} = \{\mu, v_7\}$. To generate $\mathcal{G}'(\mathcal{V}', \mathcal{E}(V'))$, we first define the set of virtual nodes $\mathcal{X} = \{\mu', v_7'\}$, and their respective 0-cost links $\mathcal{E}_1 = \{(\mu, \mu'), (v_7, v_7')\}$. Finally, we add links $\mathcal{E}_2 = \{(v_1, v_2), (v_4, v_5), (v_5, v_6)\}$ of unitary cost, to connect nodes with no links in $\mathcal{E}(\mathcal{V})$ that have the same hop-count to μ. The resulting graph $\mathcal{G}(\mathcal{V}', \mathcal{E}(\mathcal{V}'))$ is shown in Figure 6.4(b).

We first prove that if $(\mathcal{D}, K)$ is an instance of the SS-MCDS in $\mathcal{G}'$, where $K = |\mathcal{D}|$, it is also an instance of the MSPSA in $\mathcal{G}$.



(a) (b)

FIGURE 6.4. (a) $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$, (b) $\mathcal{G}'(\mathcal{V}, \mathcal{E}(\mathcal{V}'))$ and (S, K) formed by black nodes.

1. Each vertex $v' \in \mathcal{X}$ is dominated by its respective vertex $v \in \mathcal{T}$.

2. The shortest path property of the SS-MCDS guarantees that nodes in $\mathcal{T}$ have a shortest path with respect to $\mu \in \mathcal{D}$. Therefore, it follows that $(\mathcal{D}, K)$ is an instance of the MSPSA problem.

Conversely, if $(\mathcal{D}, K)$ is an instance of the MSPSA problem in G, we show that $(\mathcal{D}, K)$ is an instance of the SS-MCDS problem in $\mathcal{G}'$. First, note that since $\mathcal{E} \subset \mathcal{E}'$, set $\mathcal{D}$ also induces a tree rooted at μ in $\mathcal{G}'$ that includes the shortest between μ and vertices in $\mathcal{T}$. The rest of the proof is as follows,

1. As $\mathcal{D}$ is rooted at μ, it follows that the shortest path property (with respect to μ) is satisfied for all vertices in $\mathcal{D}$.

2. By the Steiner arborescence definition $\mathcal{T} \subseteq \mathcal{D}$. Thus, vertices in $\mathcal{X}$ are dominated by their respective vertices in $\mathcal{T}$.

3. Note that set $\mathcal{T}$ includes the node with the highest hop count ($h_{\max}$) to μ. It follows that $\mathcal{D}$ contains vertices with hop counts of $1, 2, \ldots, h_{\max}$, due to the

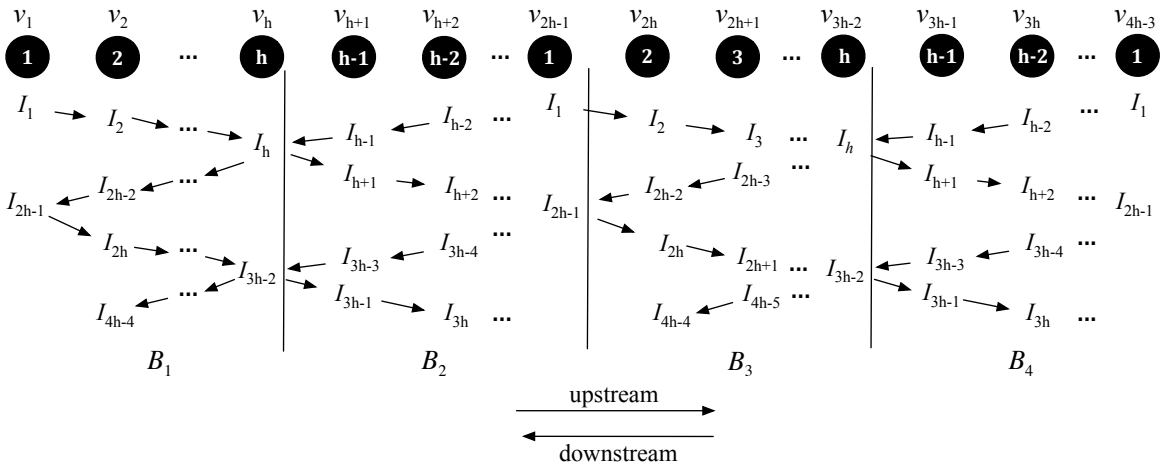


FIGURE 6.5. Transmission schedule for a path of length $4h - 3$ according to the DFAS algorithm.

shortest path property. Since edges in $\mathcal{E}_2$ connect all vertices with the same distance to μ, all vertices in $\mathcal{V}' \backslash \mathcal{X}$ are dominated as well.

4. Therefore, $\mathcal{D}$ of size K is a CDS that contains the shortest paths of all vertices in $\mathcal{D}$ with respect to μ. This makes $(\mathcal{D}, K)$ an instance of the SS-MCDS problem, which concludes both proof directions.

□

We propose a heuristic algorithm that: (a) approximates the SS-MCDS in a greedy fashion and, (b) balances between the appearance frequency of every node in an SS-MCDS, and the number of SS-MCDSs that partition $\mathcal{V}$. We now describe our algorithm in detail.

Algorithm 8: SS-MCDS approximation–We modify Algorithm 7 to generate an SS-MCDS partition of $\mathcal{V}$. In Algorithm 8, we run stage 1 of Algorithm 7 to generate a DS, forcing the sink to be the leader. Then, we modify Stage 2 of Algorithm 7 to restrict the selection of "bridge nodes" (*gray* nodes turning *black* to connect DS nodes) to those found in the shortest path to the sink. Initially, for each $v \in \mathcal{V}$ we set the appearance frequency to $f(v) = 0$.

Algorithm 8: SS-MCDS approximation

Step 1: Set the leader node s to the sink μ and execute stage 1 of Algorithm 7 to generate a DS.

Step 2: Each $v \in \mathcal{V}$ with $m(v) = gray$ broadcasts $b(v)$ to $\mathcal{N}_v^2$, where $b(v)$ is the number of *black* neighbors with hop count $|p(v,s)| + 1$ (at least one such neighbor exists).

Step 3: A *gray* node v becomes *black* if,

$$v = \arg\max_{u \in Z} \left\{ \frac{b(u)}{b_{\max}(v)} \times \frac{1}{f(u)+1} \right\}, \tag{6.3}$$

where,

$$Z = \{u : u \in \mathcal{N}_v, |p(u,s)| = |p(v,s)|, m(u) = gray\},$$

$$b_{\max}(v) = \max_{\{u \in [\mathcal{N}_v^2], m(u) = gray\}} b(u),$$

and $b(v), b_{max}(v) > 0$. Node v broadcast $m(v)$ in $\mathcal{N}_v^2$. Ties are broken arbitrarily. If $|p(w,s)| \geq |p(v,s)|$ for $w = \rho(v)$, node v sets $\rho(v) = nill$ and broadcasts it.

Step 4: A node $w \in \mathcal{N}_u$ overhearing the change of marker of u, with $m(w) = black$ and $r(w) = r(u) + 1$ sets and broadcasts $\rho(w) = u$ to $\mathcal{N}_w$. If $m(w) = gray$, $r(w) = r(u) - 1$ and $\rho(u) = nill$, w increases $b(w)$ by one.

Step 5: A *gray* node u overhearing $\rho(w)$ from a *black* node w with $r(u) = r(w) + 1$ broadcast $b(u) = b(u) - 1$ to $\mathcal{N}_v^2$.

Step 6: Steps 3-5 are iterated for all *black* nodes in the DS, until all *gray* nodes set $b(v) = 0$.

Step 7 (Pruning): If a *black* node v with $f(v) > 0$ does not dominate at least one *gray*, it sets $m(v) = gray$.

Step 8: Run Stage 3 of Algorithm 7 until $\mathcal{V}$ is partitioned.

With the termination of Algorithm 8, the set of black nodes forms a SS-CDS. To ensure that the shortest paths to the sink are included, the sink is chosen as the leader

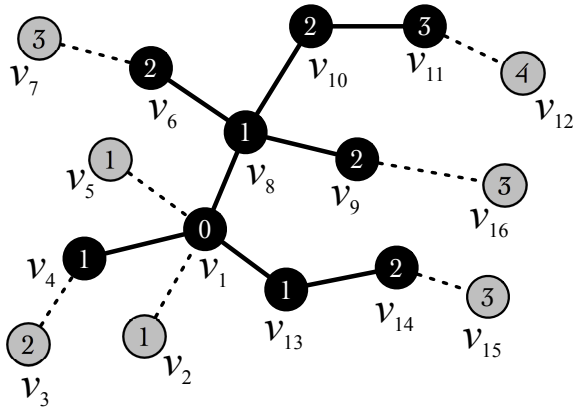


FIGURE 6.6. SS-MCDS obtained by Algorithm 8.

in the DS generation. Moreover, a *black* node v with hop-count to the sink $|p(v,s)|$ can only be dominated by a *gray* node with hop-count of $|p(v,s)|-1$ (Steps 2 and 3 of Algorithm 8). After Step 6, all *black* nodes are dominated, and are connected to the sink via the shortest path.

Consider the application of Algorithm 8 on the topology of Figure 6.3(a) and assume that v_1 is the sink. In Step 1, node v_1 is chosen as leader to generate the DS shown in Figure 6.3(b). In Steps 1 - 6, the SS-CDS is generated by adding node v_8 to dominate v_6, v_9, and v_{10}; node v_{11} to dominate v_{12}; node v_{14} to dominate v_{15}; and node v_{13} to dominate v_{14}. We observe that to satisfy the shortest-path condition, both v_{14} and its dominator v_{13} are added during this connecting stage. In Step 7, nodes v_{12} and v_{15} are pruned, if $f(v_{12})$ and $f(v_{15})$ are greater than one, respectively. The resulting SS-MCDS is shown in Figure 6.6.

Privacy Analysis In this section, we analyze the contextual information privacy achieved by the MCDS partition scheme. Assume the occurrence of an event Ψ at time $t(\Psi) \in W$ that is sensed by a sensor v located at ℓ_v. Let v belong to CDS $\mathcal{D}_i$ that is currently active (the alternative case is treated similarly, as v would have to wait until its CDS becomes active before reporting the event).

Source location and occurrence time privacy: To report Ψ, sensor v replaces the transmission of dummy packets with real ones and forwards of the event report to the sink. Note that real packets are indistinguishable from dummy ones due to the application of per-hop packet re-encryption. Downstream sensors receiving v's report continue to forward it by substitutting dummy packets with real ones. By applying the techniques described in Chapter 3, the eavesdropper can reduce the locations of the dummy transmissions to location approximation areas of the sensors in $\mathcal{D}_i$. However, events cannot be meaningfully distinguished.

The partition of the observation set $\mathcal{O}(W)$ to disjoint sets $\mathcal{Y}_1, \mathcal{Y}_2, \ldots$ depends on the selection of the bound functions β_ℓ and β_h. Selecting a small upper bound to model the relay of the immediate relay of real packets practically divides $\mathcal{O}(W)$ to disjoint sets containing one or very few tags. Every sensor transmission in $\mathcal{O}(W)$ is assumed to be a distinct event, thus hiding the source location and occurrence time of the real event Ψ. If the adversary loosens the upper bound to account for the traffic normalization applied by the CDS sensors, $\mathcal{O}(W)$ is divided to few sets $\mathcal{Y}_1, \mathcal{Y}_2, \ldots$ with large cardinality, which are mapped to events $\Psi_1, \Psi_2, \ldots$. The adversary cannot identify which set contains the real event. Moreover, the source location and time occurrence of the first tag in the set that contains $t(\Psi)$, is uncorrelated to $t(\Psi)$, but depends on the random transmission times of each sensor. We note that the adversary could apply other statistical analysis methods, such as those reported in [50, 65]. These methods fail to detect Ψ, since the transmission patterns of sensors in $\mathcal{D}_i$ do not change when real traffic is introduced.

Sink location privacy: After all CDSs have been active, the adversary can approximate the topology of each $\mathcal{D}_i$ and obtain an estimate $\hat{\mathcal{V}}$ of $\mathcal{V}$, as described in Chapter 3. When $\mathcal{V}$ is partitioned to MCDSs, the MCDS structure is unrelated to the location of the sink and hence, the sink location privacy is protected. When $\mathcal{V}$ is partitioned to SS-MCDSs, the adversary can take advantage of the shortest-path property and localize the sink. This can be achieved by Algorithm 9. Let, $p^*(u, v)$

and $p(u, v)$ be the shortest path between u and v in $\mathcal{D}_i$ and $\hat{\mathcal{V}}$, respectively.

Algorithm 9: Sink location inference

Step 1: For each $u \in \mathcal{D}_i$, obtain paths $p(u, v)$ and $p^*(u, v)$ to each $v \in \mathcal{D}_i$. Obtain the cumulative path difference as

$$\Delta_u^i = \sum_{v \in \mathcal{D}_i} |p^*(u, v)| - |p(u, v)|$$

Step 2: Repeat Step 1 for each $\mathcal{D}_i$, and calculate the average cumulative path difference as,

$$\Delta_u = \frac{1}{f(u)} \sum_{u \in D_i} \Delta_u^i$$

Step 3: Identify the sink as $\hat{\mu} = \arg\min_{u \in \hat{\mathcal{V}}} \Delta_u$.

The intuition behind Algorithm 9 is to exhaustively test every $u \in \hat{\mathcal{V}}$ as a candidate sink by computing the average cumulative path difference Δ_u. A low difference indicates that u can be reached by all nodes in the $\mathcal{D}_i$'s it belongs to via shortest paths. Since each SS-MCDS is constructed to contain the shortest paths to the sink, the sensor $\hat{\mu}$ with the lowest score over all the sets in the partition is considered to be the sink. Thus, the sink location privacy distance is the average distance between $\ell(\hat{\mu})$ and $\ell(\mu)$. The identification of the sink in the SS-MCDS design introduces a tradeoff between privacy and delay. The end-to-end delay is reduced by traversing shortest paths, while revealing the sink location.

6.1.4 Transmission Coordination

As we showed in Proposition 6, when sensors transmit in an uncoordinated fashion, the end-to-end delay for reporting a real event can increase significantly. This delay can be reduced if nodes transmit in order, relative to their depth in the CDS tree,

when the tree is assumed to be rooted at the sink. For an interval T, if downstream nodes are scheduled to transmit after upstream ones, a real transmission is guaranteed to reach the sink within T. However, this simple scheme reveals the sink location due to the use of the sink as the tree root. To preserve the sink location privacy, we propose the direction-free assignment scheme (DFAS) that guarantees the traversal of a minimum number of hops per T, while hiding the traffic directionality and the sink location.

Direction-Free Assignment Scheme (DFAS) Consider a CDS $\mathcal{D}_i$. We first divide $\mathcal{D}_i$ into several subpaths. Let h be a control parameter of the subpath length and κ a control factor of the packet rate of each node. Our algorithm uses h and κ to compute the transmission intervals for each node in $\mathcal{D}_i$.

Algorithm 10: Direction-Free Assignment Scheme (DFAS)

Step 1: Epoch W that $\mathcal{D}_i$ remains active is divided into sub-epochs $I_1, I_2, \ldots, I_{\kappa \times S}$, where $S = 4h - 4$.

Step 2: Randomly select a node μ' as the pseudo-sink.

Step 3: A node v is labeled with,

$$id_v = \begin{cases} q + 1, & \text{if } q < h \\ 2h - q + 1, & \text{if } q \geq h \end{cases}$$

where $q = \mod (|p(v, \mu')|, 2h - 2)$.

Step 4: If $id_v = 1, h$, node v with id_v transmits at random in sub-epochs

$$I_{id_v + qS}, I_{2h + id_v - 2 + qS},$$

if $2 \leq id_v \leq h - 1$ in transmits in subepochs

$$I_{id_v + qS}, I_{2h - id_v + qS}, I_{2h + id_v - 2 + qS}, I_{4h - id_v - 2 + qS}.$$

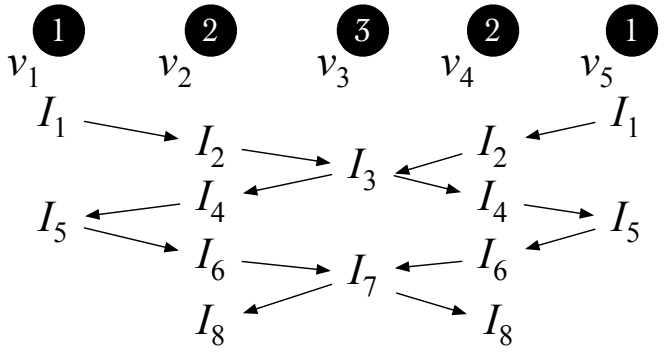


FIGURE 6.7. DFAS assignment for two subpaths ($h = 3, \kappa = 1$).

In DFAS, we have assumed that $h > 1$. If $h = 1$, the transmission assignment degenerates to uncoordinated transmissions within W. To demonstrate the operation of DFAS, consider the example of Figure 6.7 where $h = 3$ and $\kappa = 1$. In Step 1, epoch W is divided into 8 sub-epochs. In Step 2, v_1 is selected as the pseudo-sink. In Step 3, we label the network as a tree rooted at v_1, and obtain $id_{v_1} = 1$, $id_{v_2} = 2$, etc. In Step 4, sensors are assigned to transmit at random within the designated sub-epochs, according to their ids. Sensors with ids 1 and 3 transmit two packets per epoch, while all other sensors transmit four packets per epoch. As shown in the privacy analysis, the symmetry in this assignment hides the traffic direction. We now show that DFAS guarantees $2h$ relay operations per any $(4h - 4)$ sub-epochs.

Proposition 9. *In DFAS, a packet is guaranteed to be forwarded* $2h$ *hops per* $(4h-4)$ *sub-epochs, irrespective of the flow direction and the origin sensor.*

Proof. To prove Proposition 9, we apply DFAS on a path of length $4h - 3$ nodes (Figure 6.5). The path length is selected to accommodate four subpaths B_1, B_2, B_3 and B_4, so that the proposition can be proved, irrespective of the sensor that originates a packet transmission and the traffic direction. Without loss of generality, we select v_1 to be the pseudo-sink and label each node in the path according to Step 3 of Algorithm 8. We then allocate the transmission times for each node according to Step

4 of Algorithm 8. The sub-epoch assignments for each node are shown in Figure 6.5. The arrows between sub-epochs show the possible flow of traffic in two consecutive sub-epochs, in either direction.

First, consider nodes with ids 1 or h. From Figure 6.5, it is straightforward to verify that a packet m originating from a node with id 1 or h can be forwarded in the upstream or the downstream direction over

$$H_1 = 4h - 4 \text{ hops}$$

in $4h-4$ sub-epochs. This is because the neighbors of nodes with id 1 or h are assigned to transmit in successive sub-epochs in either direction. Consider now a node from B_1 or B_3 with id other than 1 or h, say j, which is assigned to transmit during sub-epochs I_j, I_{2h-j}, I_{2h+j-2}, and I_{4h-j-2}. For the upstream direction, the worst-case delay for node with id j is incurred when it initiates the packet forwarding in sub-epochs I_{2h-j} or I_{4h-j-2}. This is because the upstream neighboring node of j is scheduled to transmit $2j-2$ sub-epochs after j (as opposed to being scheduled in the successive sub-epoch when j initiates the upstream forwarding in I_j or I_{2h+j-2}). Once the packet reaches the upstream neighboring node of j, it can be relayed $4h-4-(2j-2) = 4h-2j-2$ hops in the remaining sub-epochs of the $4h-4$ period. Thus, the total number of hop relays for node with id j is equal to

$$H_2 = 4h - 2j - 2 \text{ hops},$$

which obtains the minimum value of $H_2 = 2h$ hops for $id = h-1$. In a similar way, for the downstream direction, the worst-case delay for node with id j occurs when it starts forwarding m in sub-epochs I_j or I_{2h+j-2}. The downstream node of j is now scheduled to transmit $2h-2j$ sub-epochs after j. After the packet reaches the downstream neighboring node of j, it can be relayed $4h-4-(2h-2j) = 2h+2j-4$ hops in the rest of the $4h-4$ period. The total number of hops for node with id j is

$$H_3 = 2h + 2j - 4 \text{ hops},$$

which obtains its minimum value of $H_3 = 2h$ hops for $id = 2$. Note that, if the node was chosen from B_2 or B_4, the same analysis applies with the difference that H_2 would correspond to the downstream case, and H_3 to the upstream one.

Finally, the minimum number of hops irrespective of the flow direction, in a period of $(4h - 4)$ sub-epochs is equal to

$$H_{\min} = \min\{H_1, H_2, H_3\} = 2h \text{ hops},$$

for $h \geq 2$. □

Privacy Analysis Assume event Ψ's occurrence at $t(\Psi) \in W$, while CDS $\mathcal{D}_i$ is active. Let Ψ be sensed by $v \in \mathcal{D}_i$ located at ℓ_v.

Source location and occurrence time privacy: The DFAS coordination is applied irrespective of the occurrence of an event. Sensor v and all sensors downstream to the sink, will replace the dummy transmissions scheduled at different sub-epochs with real packets. Hence, the occurrence time of Ψ is concealed. Moreover, DFAS maintains the same source location privacy as the uncoordinated case. Without an estimate of $t(\Psi)$, the adversary cannot localize v.

Sink location privacy: Assume that the adversary observes traffic during $4h-4$ sub-epochs of a single interval ($\kappa = 1$). According to the transmission assignment of DFAS, nodes transmit either 2 or 4 packets within this interval. The packet rate of each node could by used to identify the node positions within subpaths and consequently, the labeling of DFAS. For instance, in Figure 6.7, the adversary intercepts two transmissions from v_1, four transmissions from v_2, two from v_3, and so on. He can infer that $h = 3$ and discover the subpath labeling. The subpath labeling reduces the candidate nodes that could have been used as the pseudo-sink to nodes with $id = 1$ or $id = h$. However, the sink location remains hidden due to the random selection of the pseudo-sink.

We emphasize that the coordination imposed by DFAS conceals the traffic direction. Consider the transmission assignment shown in Figure 6.7. Without loss of

generality, assume that a packet m is relayed by v_1 in sub-epoch I_5, v_2 in I_6, and v_3 in I_7. However, the transmissions intercepted during these sub-epochs could be associated with packet relays in the opposite direction. The transmission in I_5 could be due to the relay of an m' by v_3, v_2, and v_1 in I_3, I_4, and I_5, respectively. Similarly, the transmission in I_7 could be due to the relay of an m' by v_5, v_4, and v_3, in I_5, I_6, and I_7, respectively. The symmetry in the assignment of the transmission sub-epochs over subpaths hides the traffic direction.

6.1.5 Routing Over Multiple CDSs

CDSs are rotated periodically per epoch to allow all sensors report events to the sink. A real packet m that originated from $v \in \mathcal{D}_i$, may be in transit while another CDS $\mathcal{D}_j$ becomes active. The CDS property guarantees that at least one node in $\mathcal{D}_j$ would overhear the last relay of m by a node in $\mathcal{D}_i$. We develop a simple routing scheme to forward packets over multiple CDSs. Here we assume that $\mathcal{D}_i$ is active during epoch W_k, and $\mathcal{D}_j$ in the next epoch W_{k+1}. The steps of our scheme are as follows.

Algorithm 11: Multiple CDS Forwarding Scheme (MCFS)

Step 1: A real packet m originating from $v \in \mathcal{D}_i$ at epoch W_k is forwarded to μ via the shortest path $p(v, \mu)$ in $\mathcal{D}_i$.

Step 2: Any $u \in \mathcal{D}_j$ (next active CDS) overhearing m's transmission during W_k's last sub-epoch (i.e., sub-epoch $I_{\kappa \times S}$), forwards m to the sink when $\mathcal{D}_j$ becomes active in W_{k+1}. Nodes in $\mathcal{D}_j$ discard any duplicates of m.

The MCFS operation is depicted in Figure 6.8. Node $v_1 \in \mathcal{D}_1$ sends a packet m to the sink during epoch W_k, using $\mathcal{D}_1$. The CDS is rotated to $\mathcal{D}_2$ while m is in transit. Node v_3 is the last one to transmit m in W_k. Nodes v_7 and v_8 overhear v_3's transmission. The relay of m is continued by sensors v_7 and v_8 during W_{k+1}, using

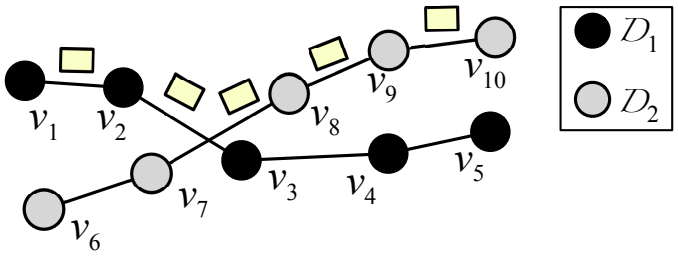


FIGURE 6.8. The MCFS operation.

$\mathcal{D}_2$. The duplicate packet forwarded by v_7 is discarded at v_8. Packet m is delivered to the sink during $\mathcal{D}_2$.

We now show the effects of CDS rotation on the number of hops traversed by a real packet.

Proposition 10. *The number of hops traversed by a real packet m originating from v until it reaches the sink μ is upper-bounded by $|p(v,\mu)| + 2rot$, where $|p(v,\mu)|$ is the shortest path length between v and μ and rot is the number of CDS rotations until m reaches μ.*

Proof. Without loss of generality, let $\mathcal{D}_1$ be active during epoch W_k. Node $v \in \mathcal{D}_1$ initiates the relay of a real packet m to the sink μ. Assume that $w \in \mathcal{D}_1$ is the last node to forward m, before rotating to a CDS $\mathcal{D}_2$ in W_{k+1}. The transmission of w is overheard by a node $u \in \mathcal{D}_2 \cap \mathcal{N}_w$. First, we show via contradiction that paths $p(v,w)$ and $p(w,\mu)$ are also shortest paths between the respective sources and destination. We break the shortest path $p(v,\mu)$ into subpaths $p(v,w)$ and $p(w,\mu)$ and assume that $p(v,w)$ is not a shortest path between v and w. That is, there is a path $p^*(v,w)$ such that $|p^*(v,w)| < |p(v,w)|$. Hence, there exist a path $p^*(v,\mu) = p^*(v,w) \cup p(w,\mu)$ with $|p^*(v,\mu)| < |p(v,\mu)|$, which contradicts the assumption that $p(v,\mu)$ is the shortest path.

With the termination of the epoch W_k, packet m has traversed $|p(v,w)| + 1$ hops to reach $u \in \mathcal{D}_2$. ($|p(v,w)|$ to reach w and one hop to reach u). The one-hop neighbor-

hood of w can be divided to three possible subsets with respect to their hop distance to the sink. One subset at a distance of $|p(w,\mu)|$, one subset at a distance $|p(w,\mu)|+1$, and one subset at a distance $|p(w,\mu)|-1$, where $|p(w,\mu)|$ is the length of the shortest path between w and μ. Again, this claim can be shown via contradiction. If there was a neighbor $k \in N_w$ that could reach μ at fewer hops than $|p(w,\mu)|-1$, then w has a shorter path to μ via k, which contradicts the assumption that $p(w,\mu)$ is the shortest path. If there was a neighbor $k \in N_w$ with a shortest path to μ longer than $|p(w,\mu)|+1$ hops, then k could reach μ in fewer hops via w, which contradicts the fact that every CDS contains shortest paths to μ.

We therefore conclude that in Step 2 of Algorithm 9, when node $u \in \mathcal{D}_2 \cap \mathcal{N}_w$ continues to forward m to μ via the shortest path, $p(u,\mu)$ has one of the following sizes:

a. $|p(u,\mu)| = |p(w,\mu)| - 1$

b. $|p(u,\mu)| = |p(w,\mu)|$

c. $|p(u,\mu)| = |p(w,\mu)| + 1$.

The worst case occurs when $|p(u,\mu)| = |p(w,\mu)| + 1$. If m is delivered during $\mathcal{D}_2$, packet m traverses in the worst case

$$H = |p(v,w)| + 1 + |p(u,\mu)| + 1 = |p(v,\mu)| + 2$$

Therefore, the number of hops traversed by m due to the rotation of CDSs, increases up to 2 hops. The same process is repeated each time that a CDS rotation occurs. So in the worst case, the total number of hops traversed by m is upper-bounded by $|p(v,\mu)| + 2rot$, where rot is the total number of rotations required to delivered the packet. □

6.2 Performance Evaluation

In this section, we evaluate the performance of partitioning WSN into connected dominated sets.

6.2.1 Simulation Setup

We randomly deployed 250 sensors within a 450m×450m area. The sensor transmission range and the eavesdropper reception range were set to 50m. We abstracted the PHY and MAC layers into a simple per-hop delay model, which consists of a fixed component representing the transmission and propagation delays at the PHY layer, and the contention delay at the MAC layer. This delay was set to 166ms for a packet transmission of 1280 bytes, according to the IEEE 802.15.4 protocol evaluation presented in [31]. No retransmissions due to collisions or impairments of the wireless medium were considered. This simple model was preferred to eliminate the randomness in different system realizations due to contention. Moreover, this model closely matches event-driven networks, which operate under low-contention conditions due to the sparsity of transmissions. Unless otherwise noted, simulation was terminated after the occurrence of ten events.

6.2.2 Generation of a CDS Partition

In these experiments, we studied the performance of Algorithms 7 and 8 in partitioning the sensors set $\mathcal{V}$ to multiple CDSs. We deployed several WSNs and varied the average node degree δ by increasing the number of sensors. Sensor locations were randomly drawn from a uniform distribution to generate random network topologies. We ran Algorithms 7 and 8 to obtain the MCDS and SS-MCDS partition. We evaluated the following metrics: (a) the average fraction of sensors that belong to a CDS (CDS size), (b) the number of CDSs needed to span $\mathcal{V}$, and (c) the probability mass function (pmf) of the appearance frequency $f(v)$ to a CDS.

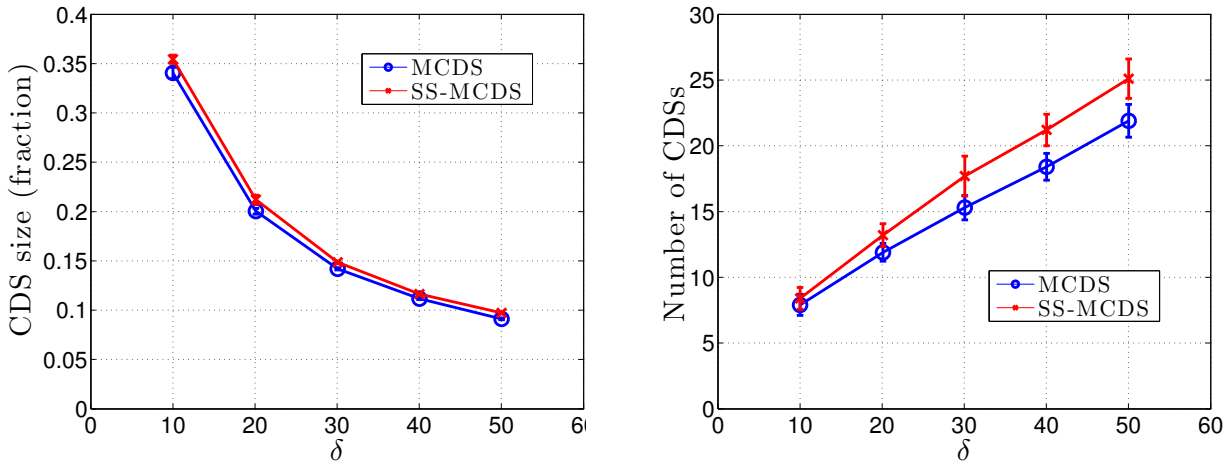


FIGURE 6.9. (a) Average CDS size normalized over $|\mathcal{V}|$, as a function of δ, (b) average number of CDSs that span $\mathcal{V}$ as a function of δ.

Figure 6.9(a) shows the average size of the CDSs that span $\mathcal{V}$, as a function of δ. Confidence intervals of 95% are also shown. The CDS size directly relates to the energy savings compared to the global norm. method, which requires all sensors to be dummy traffic sources [50]. We observe that fraction of active sensors decreases with δ, which indicates higher energy savings. Furthermore, the size difference between MCDSs and SS-MCDSs is close to 2%, for all values of δ. This indicates that the SS-MDSS partition optimizes the paths to the sink without a significant penalty on the CDS size. Figure 6.9(b) shows the average number of CDSs needed to span $\mathcal{V}$, as a function of δ. This value is related the delay until a CDS containing a sensor with a real packet for transmission becomes active. We observe an almost linear increase in the number of CDSs with δ. Also note that the total number of SS-MCDSs required to span $\mathcal{V}$ is slightly higher than the number of MCDSs. This difference is attributed to the requirement of including shortest paths to a fixed node (the sink) for the SS-MCDS case. In the MCDS case, the algorithm spans $\mathcal{V}$ faster, as the leader node changes in every iteration.

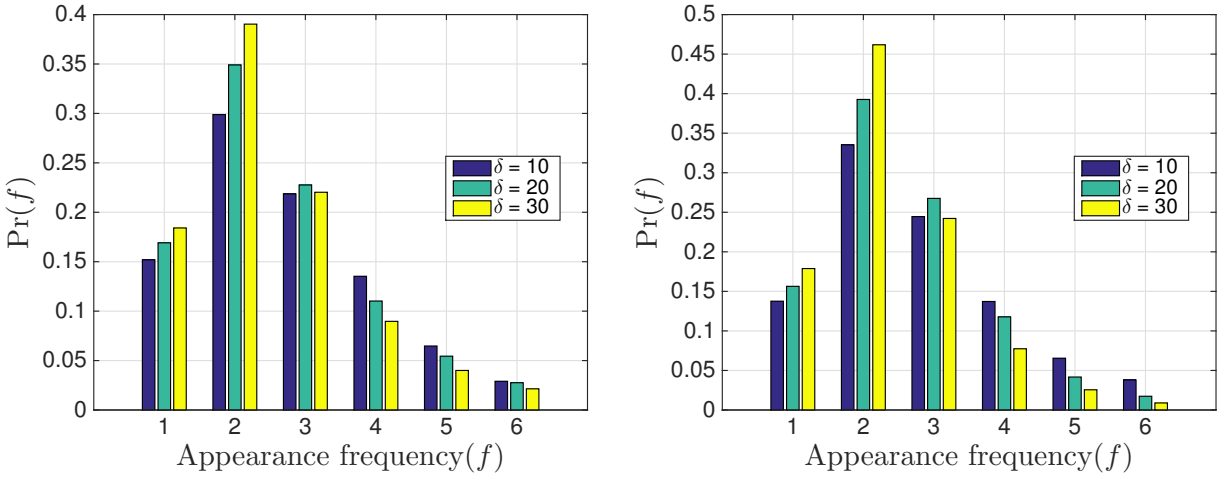


FIGURE 6.10. Empirical probability mass function of f for the (a) MCDS partition, and (b) SS-MCDS partition.

In Figs. 6.10(a) and 6.10(b), we show the empirical pmf for the appearance frequency $f(v)$ when constructing MCDSs and SS-MCDSs, respectively. The $f(v)$ represents the "quality" of the partition (ideally, $f(v) = 1, \forall v \in \mathcal{V}$). For both partitions types, more that 50% of sensors are part of one or two CDSs, while for 95% of the sensors, $f(v) < 5$. This indicates that Algorithm 8 favors the creation of disjoint CDSs to a large extend, thus reducing the per-sensor dummy traffic overhead.

6.2.3 Communication and Delay Overheads

In the last set of experiments, we studied the communication overhead and end-to-end delay for delivering real packets to the sink. We only compared our method to the global norm. method, which achieves the same privacy distance as MCDS. In our experiments, each CDS remained active for one epoch. Epochs were divided to $(4h - 4)$ sub-epochs, to compute the transmission schedule of sensors according to DFAS. An event was reported by all sensors that sensed the event to reduce the delay

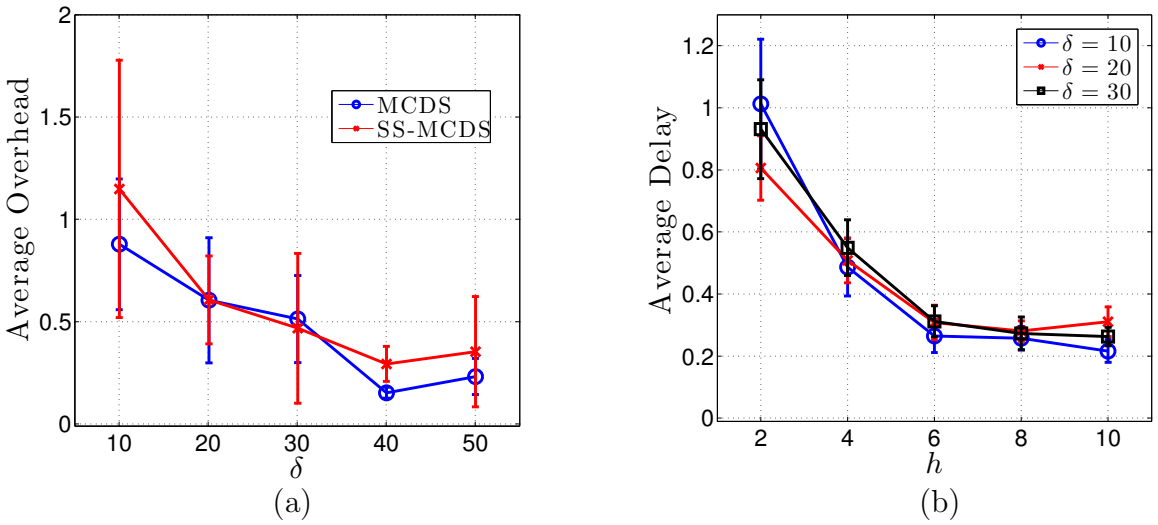


FIGURE 6.11. (a) Average delay as a function of the hop count to the sink, (b) average delay as a function of the subpath size.

due to CDS rotation. To provide a fair comparison, we set the dummy packet rate in the global norm. method that achieves the same end-to-end delay as in MCDS.

Figure 6.11 (a) shows the average communication overhead as a function of the sensor density. The overhead is normalized to the traffic volume introduced by the global norm. method. We observe that the SS-MCDS introduces an overhead of 120% when $\delta = 10$, while for the MCDS, the overhead drops to 90%. This difference is primarily due to the delay introduced by the CDS rotation. As a partition to SS-MCDSs requires a larger number of SS-MCDSs to span $\mathcal{V}$, a sensor that has detected an event has to wait longer until its CDS becomes active. The communication overhead is drastically reduced with the increase of δ. This is due to the fact that under denser sensor deployments, more sensors can detect an event. These sensors likely belong to different CDSs because they are within the same neighborhood. Therefore, the time until a CDS with a sensor that detected the event becomes active is lower.

In Figure 6.11(b), we show the average end-to-end delay achieved by DFAS as a function of the subpath size h. The delay is normalized to the delay incurred when

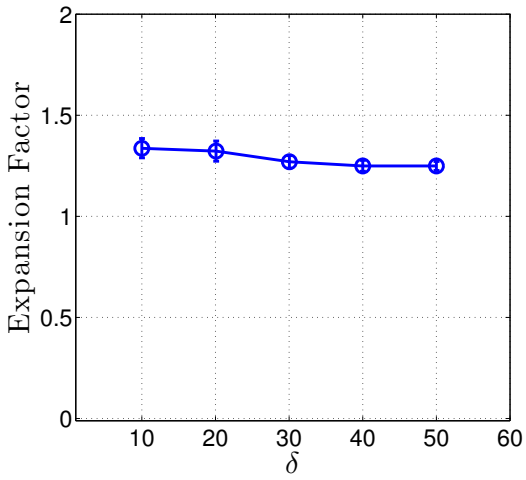


FIGURE 6.12. Expansion factor between MCDS and SS-MCDS partition as a function of δ.

transmissions are uncoordinated (sensors transmit at random times within an epoch). To provide a fair comparison, each scheme was restricted to transmit the same number of dummy packets. We considered events reported by sensors located 7 hops away from the sink (which is the highest hop-count in the network). As expected, the delay reduces with the increase of h, as a larger number of hops is traversed per epoch. Finally, in Figure 6.12 we present the average path expansion factor for the MCDS method relative to the SS-MCDS method, as a function of δ. The expansion factor is defined as the ratio between the path length to the sink, when MCDSs are constructed, relative to the shortest path. We observe that MCDS produces slightly longer routing paths to the sink (in the order of 30% when $\delta = 10$.) However, this difference reduces with δ due to the availability of more routes with short paths to the sink.

6.3 Summary of Contributions

We considered the problem of protecting the inference of contextual information privacy beyond SSA. We addressed the tradeoff between communication and delay over-

head by partitioning the network into MCDSs and SS-MCDSs. The former is designed to be of minimum size, while the latter includes the shortest paths between any source to the sink. By partitioning the network, we reduced the number of dummy packets that each sensor injects. Compared to prior methods capable of protecting against a global adversary model, we showed that limiting the dummy traffic transmissions to MCDS nodes, reduces the communication overhead due to traffic normalization. We also characterized the complexity of generating such partitions, and provided of heuristic algorithms that approximates the optimal partition. Finally, we note that when sensors transmit dummy (or real) packets in an uncoordinated fashion, the end-to-end delay for reporting real events can increase significantly. We proposed a loose coordination scheme that reduces the end-to-end delay without revealing real traffic patterns or the traffic directionality.

Chapter 7
Summary of Main Contributions

In this dissertation, we investigated the problem of protecting contextual information privacy in event-driven in WSNs. In this chapter, we summarize our main contributions.

7.1 Main Contributions

We investigated the exposure of contextual information in event-driven WSNs. We considered a global adversary model, whereby the adversary is capable of eavesdropping all communications within the WSN. We developed a suite of traffic analysis techniques designed to extract contextual information from intercepted communication attributes such as transmission time and packet content hashes. We limited the information available to the adversary to make our methods broadly applicable to several models including WSNs that apply security measures such as, link-layer re-encryption, identity anonymization, injection of dummy packets, etc. We showed that by analyzing the collected information, the adversary is able to obtain the event location, event occurrence time, the routing path traversed by event reports and the sink location.

We extended our analysis to the inference of contextual information when the WSN transmissions are protected by a traffic normalization method which relies on statistical source anonymity (SSA). State-of-the-art countermeasures against global eavesdroppers conceal traffic associated to real events by injecting dummy packets according to a predefined distribution. Even though such schemes provide perfect privacy, they introduce performance challenges in the form of either high communication overheads or high end-to-end delay for event reports. To address that tradeoff,

several authors have proposed to accelerate the transmission of a real packet and delay the transmission of the following dummy packet such that the difference observed by the adversary is not statistically significant. Such methods achieve statistical source anonymity (SSA).

In our study, we noted that the modification of inter-packet times for speeding up real transmissions and recovering the distribution mean can cause detectable outliers. These outliers can be used to distinguish between datasets containing real transmissions and datasets containing only dummy ones. We showed that SSA can be breached by applying well-known outlier detection methods for time series data. We applied an autoregressive model that does not require a priori knowledge of the distribution used to schedule dummy transmissions. By applying this model, the adversary is able to pinpoint intervals of time when real events were reported with probability far greater than a random guess. Moreover, we showed that the adversary can use the location of the outliers within the time series to significantly limit the candidate event occurrence time.

We extended the outlier detection method to time series generated by collectively analyzing the transmissions over multiple hops. The key insight here is that correlated transmission patterns can be observed, when each of the relay sensors accelerates the transmission of a relayed real packet. The appearance of short-long transmission patterns in sequence amplifies the outliers in the "collective" time series and reduces false alarms. We showed that existing countermeasures that call for the insertion of fake events which mimic real events are not effective when outlier detection is applied over multiple hops. This is because sensors independently insert fake events without coordinating over multiple hops. Therefore, these fake events do not appear is if the occur in sequence, as is the case during the relay of real packets.

We addressed the problem of contextual information privacy from a resource point of view. We considered the tradeoff between communication and end-to-end delay overhead by limiting the set of sensors required to inject dummy traffic. We mapped

the problem of selecting dummy sources to the problem of finding a minimum connected dominating set (MCDS) that covers the WSN deployment area. The MCDS guarantees that every sensor is at most one hop away from a MCDS sensor an that every eavesdropper observes dummy traffic patterns.

To prevent the leakage of private information, we proposed rate control techniques that regulate the transmission of dummy packets for nodes in the CDS. For real packets, nodes in the CDS transmit them by simply replacing the next scheduled dummy packet by a real one. However, for nodes that are not part of the CDS, real packets are transmitted at a rate that has statistically insignificant impact on the traffic patterns observed by eavesdroppers in the neighborhood of the real source. We showed that our methods are effective against eavesdroppers that perform traffic analysis independently and when network traffic is collectively analyzed by a central server. Moreover, we showed that the number of dummy sources reduces to 7% in comparison to schemes that use the whole WSN to introduce dummy traffic.

Finally, we proposed traffic normalization techniques that reduce both the communication and delay overhead, while preventing the leakage of private information. Our methods partitioned the network into connected dominating sets (CDSs) that are activated in a round-robin fashion. We provided event location, its occurrence time, and the sink location privacy. We showed that compared to prior art, our method considerably reduces the communication overhead to a 25% and the end-to-end delay to a 20%. We considered two types of partitions, minimum connected dominating sets (MCDSs) and MCDSs with shortest paths to the sink (SS-MCDSs). The former one includes sets of minimum size, which aims at minimizing the amount of dummy traffic introduced. The latter one is introduced to include the shortest paths from any sensor in the CDS to the sink, designed the reduce the end-to-end delay of real packets. We show that the routing paths in the MCDS are approximately 1.3 times higher than the paths in the SS-MCDS.. This is reflected in a lower end-to-end delay for SS-MCDS at the cost of a slightly higher communication overhead.

References

[1] B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Statistical framework for source anonymity in sensor networks. In *Proceedings of the GLOBECOM Conference*, pages 1–6, 2010.

[2] F. Armknecht, J. Girao, A. Matos, and R. Aguiar. Who said that? privacy at the link layer. In *Proceedings of the INFOCOM Conference*, pages 2521–2525, 2007.

[3] J. Bentley and T. Ottmann. Algorithms for reporting and counting geometric intersections. *Transactions on Computers*, 100(9):643–647, 1979.

[4] O. Berthold, H. Federrath, and S. Köpsell. Web mixes: A system for anonymous and unobservable internet access. In *Designing Privacy Enhancing Technologies*, pages 115–129. Springer, 2001.

[5] K. Bicakci, H. Gultekin, B. Tavli, and I. Bagci. Maximizing lifetime of event-unobservable wireless sensor networks. *Computer Standards & Interfaces*, 33(4):401–410, 2011.

[6] W. Blue Radios. Order & price info. `http://www.blueradios.com`, 2014.

[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

[8] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[9] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of cryptology*, 1(1):65–75, 1988.

[10] X. Chen, K. Makki, K. Yen, and N. Pissinou. Sensor network security: a survey. *Communications Surveys & Tutorials, IEEE*, 11(2):52–73, 2009.

[11] X. Cheng and D. Du. Virtual backbone-based routing in multihop ad hoc wireless networks. Technical report, University of Minnesota, 2002.

[12] G. Chinnu and N. Dhinakaran. Protecting location privacy in wireless sensor networks against a local eavesdropper–a survey. *International Journal of Computer Applications*, 56(5):25–47, 2012.

[13] B. Clark, C. Colbourn, and D. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1):165–177, 1990.

[14] R. Cleveland, W. Cleveland, J. McRae, and I. Terpenning. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.

[15] J. Cong, A. Kahng, and K. Leung. Efficient algorithms for the minimum shortest path steiner arborescence problem with applications to vlsi physical design. *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(1):24–39, 1998.

[16] M. Conti, J. Willemsen, and B. Crispo. Providing source location privacy in wireless sensor networks: A survey. *Communications Surveys Tutorials*, 15(3):1238–1280, 2013.

[17] T. Cormen, C. Leiserson, R. Rivest, C. Stein, et al. *Introduction to algorithms*, volume 2. MIT Press Cambridge, 2001.

[18] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[19] J. Cuellar and R. Poovendran. Toward a statistical framework for source anonymity in sensor networks. *IEEE TRANSACTIONS ON MOBILE COMPUTING*, 12(2), 2013.

[20] W. Dai. Pipenet 1.1. *Usenet post, August*, 1996.

[21] G. Danezis and J. Clulow. Compulsion resistant anonymous communications. In *Information Hiding*, pages 11–25, 2005.

[22] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *Proceedings of the Security and Privacy Symposium*, pages 2–15, 2003.

[23] J. Deng, R. Han, and S. Mishra. Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks. *Pervasive and Mobile Computing*, 2(2):159–186, 2006.

[24] C. Diaz and B. Preneel. Taxonomy of mixes and dummy traffic. In *Information Security Management, Education and Privacy*, pages 217–232. Springer, 2004.

[25] R. Dingledine, N. Mathewson, and P. Syverson. TOR: The second-generation onion router. Technical report, DTIC Document, 2004.

[26] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of the INFOCOM Conference*, volume 1, 2004.

[27] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the CCS Conference*, pages 41–47, 2002.

[28] 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 *Proceedings of the Globecom Conference*, pages 1–5. IEEE, 2010.

[29] Y. Fan, Y. Jiang, H. Zhu, and X. S. Shen. An efficient privacy-preserving scheme against traffic analysis attacks in network coding. In *Proceedings of the Infocom Conference*, pages 2213–2221. IEEE, 2009.

[30] M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 193–206. ACM, 2002.

[31] M. Fruth. Probabilistic model checking of contention resolution in the IEEE 802.15.4 low-rate wireless personal area network protocol. In *Proc. of the Symp. on Leveraging Applications of Formal Methods, Verification and Validation*, pages 290–297, 2006.

[32] M. Garey and D. Johnson. *Computers and Intractability*, volume 174. Freeman, 1979.

[33] J. Gross and J. Yellen. *Handbook of Graph Theory*. CRC, 2004.

[34] M. Gupta, J. Gao, C. Aggarwal, and J. Han. Outlier detection for temporal data. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 5(1):1–129, 2014.

[35] M. Hau and H. Tong. A practical method for outlier detection in autoregressive time series modelling. *Stochastic Hydrology and Hydraulics*, 3(4):241–260, 1989.

[36] X. Hong, P. Wang, J. Kong, Q. Zheng, and J. Liu. Effective probabilistic approach protecting sensor traffic. In *Military Communications Conference, 2005. MILCOM 2005. IEEE*, pages 169–175. IEEE, 2005.

[37] 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*, 54(6):860–874, 2011.

[38] L. Jia, R. Rajaraman, and T. Suel. An efficient distributed algorithm for constructing small dominating sets. *Distributed Computing*, 15(4):193–205, 2002.

[39] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk. Enhancing source-location privacy in sensor network routing. In *Proceedings of the ICDCS Conference*, pages 599–608, 2005.

[40] D. Kesdogan, P. Reichl, and K. Junghärtchen. Distributed temporary pseudonyms: A new approach for protecting location information in mobile communication networks. In *Computer SecurityESORICS 98*, pages 295–312. Springer, 1998.

[41] S. Labs. Wireless sensor node. `https://www.silabs.com/`, 2015.

[42] Y. Li, L. Lightfoot, and J. Ren. Routing-based source-location privacy protection in wireless sensor networks. In *Electro/Information Technology, 2009. eit'09. IEEE International Conference on*, pages 29–34. IEEE, 2009.

[43] Y. Li and J. Ren. Source-location privacy through dynamic routing in wireless sensor networks. In *Proceedings of the INFOCOM Conference*, pages 1–9, 2010.

[44] L. Lightfoot, Y. Li, and J. Ren. Preserving source-location privacy in wireless sensor network using STaR routing. In *Proc. of the IEEE GLOBECOM conference*, pages 1–5, 2010.

[45] X. Luo, X. Ji, and M. Park. Location privacy against traffic analysis attacks in wireless sensor networks. In *Proceedings of the ICISA Conference*, pages 1–6, 2010.

[46] M. Mahmoud and X. Shen. A cloud-based scheme for protecting source-location privacy against hotspot-locating attack in wireless sensor networks. *Transactions on Parallel and Distributed Systems*, 23(10):1805–1818, 2012.

[47] M. Mahmoud and X. Shen. A novel traffic-analysis back tracing attack for locating source nodes in wireless sensor networks. In *Proceedings of the ICC Conference*, pages 939–943, 2012.

[48] M. Mahmoud and X. Shen. Secure and efficient source location privacy-preserving scheme for wireless sensor networks. In *Proceedings of the ICC Conference*, pages 1123–1127, 2012.

[49] A. Majeed, K. Liu, and N. Abu-Ghazaleh. Tarp: Timing analysis resilient protocol for wireless sensor networks. In *Proceedings of the WIMOB conference*, pages 85–90. IEEE, 2009.

[50] K. Mehta, D. Liu, and M. Wright. Protecting location privacy in sensor networks against a global eavesdropper. *Transactions on Mobile Computing*, 11(2):320–336, 2012.

[51] U. Möller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster protocolversion 2. *Draft, July*, 2003.

[52] A. Mukherjee and A. L. Swindlehurst. Detecting passive eavesdroppers in the MIMO wiretap channel. In *Proc. of the Conf. on Acoustics, Speech and Signal Processing*, pages 2809–2812, 2012.

[53] C. Ozturk, Y. Zhang, and W. Trappe. Source-location privacy in energy-constrained sensor network routing. In *Proc. of the ACM workshop on Security of ad hoc and sensor networks*, pages 88–93, 2004.

[54] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. Culler. Spins: Security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, 2002.

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

[56] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymitya proposal for terminology. In *Designing privacy enhancing technologies*, pages 1–9. Springer, 2001.

[57] A. Pfitzmann and M. Waidner. Networks without user observability. *Computers & Security*, 6(2):158–166, 1987.

[58] M. Reed, P. Syverson, and D. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, 1998.

[59] M. Rennhard and B. Plattner. Introducing morphmix: peer-to-peer based anonymous internet usage with collusion detection. In *Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 91–102. ACM, 2002.

[60] R. Rios, J. Cuellar, and J. Lopez. Robust probabilistic fake packet injection for receiver-location privacy in wsn. In *Proceedings of the ESORICS Symposium*, pages 163–180, 2012.

[61] K. Römer, P. Blum, and L. Meier. Time synchronization and calibration in wireless sensor networks. *Handbook of Sensor Networks: Algorithms and Architectures*, pages 199–237, 2005.

[62] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Privacy Enhancing Technologies*, pages 41–53. Springer, 2003.

[63] R. A. Shaikh, H. Jameel, B. J. dAuriol, H. Lee, S. Lee, and Y.-J. Song. Achieving network level privacy in wireless sensor networks. *Sensors*, 10(3):1447–1472, 2010.

[64] C. E. Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

[65] M. Shao, Y. Yang, S. Zhu, and G. Cao. Towards statistically strong source anonymity for sensor networks. In *Proceedings of the INFOCOM Conference*, pages 51–55, 2008.

[66] T. Shu, M. Krunz, and S. Liu. Secure data collection in wireless sensor networks using randomized dispersive routes. *Mobile Computing, IEEE Transactions on*, 9(7):941–954, 2010.

[67] K. Sohrabi, J. Gao, V. Ailawadhi, and G. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7(5):16–27, 2000.

[68] M. Stephens. Edf statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, 69(347):730–737, 1974.

[69] D. R. Stinson. *Cryptography: theory and practice*. CRC press, 2005.

[70] G. Suarez-Tangil, E. Palomar, B. Ramos, and A. Ribagorda. An experimental comparison of source location privacy methods for power optimization in wsns. In *Proceedings of the WSEAS conference*, pages 79–84, 2010.

[71] W. Venables and B. Ripley. *Modern Applied Statistics with S*. Springer, 2002.

[72] L. von Ahn, A. Bortz, and N. J. Hopper. K-anonymous message transmission. In *Proceedings of the 10th ACM conference on Computer and Communications Security*, pages 122–130. ACM, 2003.

[73] A. Wadaa, S. Olariu, L. Wilson, M. Eltoweissy, and K. Jones. On providing anonymity in wireless sensor networks. In *Proceedings of the ICPADS*, pages 411–418. IEEE, 2004.

[74] W. Wei-Ping, C. Liang, and W. Jian-xin. A source-location privacy protocol in wsn based on locational angle. In *Proceedings of the ICC conference*, pages 1630–1634. IEEE, 2008.

[75] S. William. *Cryptography and Network Security*. Pearson Education, 2006.

[76] Y. Xi, L. Schwiebert, and W. Shi. Preserving source location privacy in monitoring-based wireless sensor networks. In *Proceedings of the IPDPS Symposium*, pages 1–8, 2006.

[77] Y. Xiao, V. Rayi, B. Sun, X. Du, F. Hu, and M. Galloway. A survey of key management schemes in wireless sensor networks. *Computer Communications*, 30(11):2314–2341, 2007.

[78] W. Yang and W. Zhu. Protecting source location privacy in wireless sensor networks with data aggregation. In *Proceedings of the ICUIC Conference*, pages 252–266, 2010.

[79] Y. Yang, M. Shao, S. Zhu, and G. Cao. Towards statistically strong source anonymity for sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 9(3):34, 2013.

[80] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao. Towards event source unobservability with minimum network traffic in sensor networks. In *Proceedings of the WiSec Conference*, pages 77–88, 2008.