Imperial College London
Faculty of Engineering
Department of Computing

Ensuring the Resilience of Wireless Sensor Networks to Malicious Data Injections through Measurements Inspection

Vittorio Paolo Illiano

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of the Imperial College London,
September 2017

Declaration of Originality

I herewith certify that all material in this dissertation is my own work and that all else has been properly acknowledged.

Copyright Declaration

Abstract

Malicious data injections pose a severe threat to the systems based on *Wireless Sensor Networks* (WSNs) since they give the attacker control over the measurements, and on the system's status and response in turn. Malicious measurements are particularly threatening when used to spoof or mask events of interest, thus eliciting or preventing desirable responses. Spoofing and masking attacks are particularly difficult to detect since they depict plausible behaviours, especially if multiple sensors have been compromised and *collude* to inject a coherent set of malicious measurements.

Previous work has tackled the problem through *measurements inspection*, which analyses the inter-measurements correlations induced by the physical phenomena. However, these techniques consider simplistic attacks and are not robust to collusion. Moreover, they assume highly predictable patterns in the measurements distribution, which are invalidated by the unpredictability of events.

We design a set of techniques that effectively *detect* malicious data injections in the presence of sophisticated collusion strategies, when one or more events manifest. Moreover, we build a methodology to *characterise* the likely compromised sensors. We also design *diagnosis* criteria that allow us to distinguish anomalies arising from malicious interference and faults.

In contrast with previous work, we test the robustness of our methodology with automated and sophisticated attacks, where the attacker aims to evade detection. We conclude that our approach outperforms state-of-the-art approaches. Moreover, we estimate quantitatively the WSN degree of resilience and provide a methodology to give a WSN owner an assured degree of resilience by automatically designing the WSN deployment.

To deal also with the extreme scenario where the attacker has compromised most of the WSN, we propose a combination with *software attestation techniques*, which are more reliable when malicious data is originated by a compromised software, but also more expensive, and achieve an excellent trade-off between cost and resilience.

Acknowledgements

I am grateful to my supervisor Professor Emil Lupu, for believing that I was suited for this PhD position, for motivating me along the whole project, for working closely with high interest in the definition of the main steps, and for all his priceless support both in and out of the college.

I wish to thank the Resilient Information Systems Security Group (RISS) for their comments and suggestions, which have made a valuable contribution to this thesis.

Finally, I wish to thank my family: my son Leonardo, my life partner Noruena, and our parents, for filling me with joy and giving me the right mood to face the challenges of this work.

Dedication

In loving memory of my grandmother, Nonna Concetta.

'All things have intelligence and a share in thought.'

Empedocles

Publications

Journals:

- "Detecting malicious data injections in wireless sensor networks: A survey", V. P. Illiano and E. C. Lupu., ACM Computing Surveys, Oct. 2015 [IL15b].
- "Detecting malicious data injections in event detection wireless sensor networks", V. Illiano and E. Lupu, IEEE Transactions on Network and Service Management, Sept 2015 [IL15a].
- "Don' t fool me!: Detection, characterisation and diagnosis of spoofed and masked events in wireless sensor networks", V. P. Illiano, L. Munoz-Gonzalez, and E. Lupu, IEEE Transactions on Dependable and Secure Computing, 2016 [IMGL17].
- "Determining Resilience Gains from Anomaly Detection for Event Integrity in Wireless Sensor Networks", V. P. Illiano, A. Paudice, L. Munoz-Gonzalez, and E. Lupu . To appear in ACM Transactions on Sensor Networks.

Conferences:

- "Unity is strength! Combining Attestation and Measurements Inspection to handle Malicious Data Injections in WSNs" [ISL17], V. Illiano, R. Steiner, and E. Lupu, Proceedings of the 10th ACM Conference on Security & Privacy in Wireless and Mobile Networks. WiSec '17.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past decade, computer systems have experienced a radical change in the way they are conceived, as they automatically collect and elaborate data to monitor, optimise and customise a physical environment to our needs. In such contexts, frequent human interactions are expensive and possibly dangerous because of harsh environments, or impractical because of quick evolutions of the physical phenomena. So, it is usually more convenient to rely on *sensor networks*, i.e. networks made of sensor nodes that are able to sense a physical phenomenon and transmit the sensed values.

Wireless Sensor Networks are flexible, low-cost, and easy-to-deploy infrastructures of wirelessly connected sensor nodes, to collect data from physical spaces.

The very first deployments of WSNs were motivated by military applications [Đur+12], but they have now become popular also to monitor critical infrastructures, such as power grids [LLR10], water networks [Jia+09], and road transport [Boh+08]. An increasing interest in WSNs is also due to the widespreading Internet of Things (IoT) applications, such as smart cities [ALK10], smart homes [Sur+15], and monitoring of physiological parameters for healthcare through both wearable and implantable sensors [Gub+13].

To effectively deliver the applications, sensor nodes need to be cheap, physically small, communicate wirelessly, survive in harsh environmental conditions with minimal supervision, and

flexibly adapt to changes in the topology, such as sensors joining and leaving the network. These characteristics are also their main limitations. The sensor nodes have limited computational and power resources, whether to monitor a human body or a large flood plain. They are subject to wear, due to the effects of the environment, which cause failures. They need to be resource efficient. Their battery needs to be replaced when the nodes are battery-powered.

WSNs carry several vulnerabilities in the sensor nodes, the wireless medium, and the environment. The nodes are vulnerable to tampering on the field, since they are often unattended, physically accessible, and use of tamper-resistant hardware is often too expensive. The sensor nodes may also be controlled by an attacker at deployment time, since new nodes may join the network dynamically in some applications. The wireless medium is difficult to secure and can be compromised at all layers of the protocol stack since cryptographic operations and key management consume valuable computational and power resources. Finally, the cost of monitoring the deployments with external surveillance (e.g., CCTV) is generally prohibitive, hence an attacker may access the sensing environment without being noticed. Yet, despite this, WSNs are increasingly used to monitor critical infrastructures, and human health where malicious attacks can lead to significant damage and even loss of life.

Faced with the challenge of securing WSNs, new security solutions have been proposed for these platforms. The literature is rich and includes, among other, secure routing [KW05], authentication [KA10], cryptography [LN08], and key management [Du+06]. Most studies focus on proposing solutions against communications-layer and network-layer threats, such as jamming attacks, attacks against the routing protocols, confidentiality and integrity of the data in transit, etc. However, these solutions cannot prevent an attacker compromising the nodes themselves to impair the measurements' integrity at the time the measurements are taken, or before they are transmitted, e.g. connecting to their physical interfaces. The measurements collected are used to analyse the sensed phenomenon, react to it, and deliver a critical service, hence the correctness of the measurements is a requirement for the fulfilment of the WSN's function.

A compromised node could be subject to *malicious data injections*, consisting of manipulations

that enable an attacker to solicit an incorrect system's response, such as concealing the presence of problems, or raising false alarms. This could cause high economic losses and harm people's safety. For instance, in 1979 the Three Mile Island Nuclear Generating Station (Pennsylvania, USA) suffered a core meltdown [Com14]. The human machine interface was sending ambiguous signals to the operators, who were not able to immediately handle the accident. This led to one of the most serious nuclear accidents on US soil. Similar accidents may be deliberately triggered by attackers by injecting false readings in a nuclear power plant's radiation monitoring device, or in any other device advocated to monitor a critical physical phenomenon.

Several security mechanisms have been proposed so far to guarantee measurement integrity in WSNs, these can be divided into two main categories: proactive and reactive. Proactive mechanisms aim at preventing integrity violations by strengthening the data collection and transmission process. Reactive techniques, in contrast, seek to detect attacks after they have already occurred.

Proactive mechanisms work by securing: 1. The sensor nodes, e.g. with the use of tamper-proof hardware[Abe+16] 2. The measurements transmission [SLP11; Sim+10] with the use of more sophisticated devices that run, e.g., cryptographic hash functions. 3. The access to the sensed environment, e.g. by supporting the main WSN with a video surveillance WSN [Che+08] on the deployment field. There are two main problems with proactive solutions: 1) They are expensive and thus against one of the main characteristics that make WSNs attractive. 2) They do not cover all attacks: e.g., proactive mechanisms would not prevent a sensor node to be compromised before being introduced in the network, such as when the network is expanded with new nodes, or a faulty device is replaced.

The main subject of this thesis is the family of techniques known as *measurements inspection*, which acquired much interest in literature [RLP08; Lop+10; Jur+11; Xie+11; IL15b]. Measurements inspection is reactive since it operates when the measurements have been already altered by the attacker, with no assumption on the attack vector. Hence, it can address threats that are very different in nature, and even if they are unknown. The principle behind measurements inspection is to detect malicious interference by partitioning the measurements into genuine

and malicious. This task is done without any knowledge about the malicious measurements, as only the genuine measurements are characterised.

Rather than characterising the set of genuine measurements under each possible scenario, which would be impractical, measurements inspection aims to characterise genuine inter-measurements relationships, referred to as *correlations*. The idea behind measurements inspection is to test whether such correlations hold. Measurements inspection has been considered as a method for counteracting both faults and malicious interference [IL15b]. This has led to assume that malicious measurements have the same characteristics of randomness, typical of faults. Instead, in [TH06; CP07; Rez+13], it is shown how the problem gets more complex when the sensor nodes are assumed to *collude*, i.e. act in concert according to a common strategy. The colluding nodes may inject data which resemble genuine data to avoid detection, or to make genuine nodes appear responsible for the degradation in correlation. However, most existing work has been evaluated for cases not involving collusion.

The principle that allows one to detect malicious interference is that correlations may be disrupted when some of the measurements change under malicious interference, especially if the malicious measurements noticeably differ from their original counterparts. Unless the system is in an unstable state, malicious and original measurements generally need to significantly differ to cause high damage to the WSN system. Even if there is a large difference between the genuine and the malicious measurement that replaced it, there are three central problems involved: 1. The genuine value is not observable, hence cannot be compared against the malicious measurement. 2. Many malicious nodes can collude. 3. The measurements correlations can experience considerable variations due to changes in the physical phenomenon, referred to as *events*. Examples of events are earthquakes for seismic sensors which causes high vibrations, or floods for water level monitoring sensors which would register the surface rise. If the injection process is sophisticated enough, it can exploit such limitations to make malicious data indistinguishable from genuine data. It is our goal to develop the methodologies and methods that deal with such sophisticated attackers, and in this thesis we will show that we are able to either detect the attack or force it to minimise the damage that can be caused whilst staying undetected.

In particular, the effects of events on the data will be a central topic of this thesis since most critical WSN applications are event-based, and thus demand higher security. Related work does not address detection of malicious data under the possible presence of events, which distort inter-measurements correlations giving rise to more challenging problems. Moreover, when malicious data is used to elicit or prevent event detection, they produce two threatening and difficult to detect attacks, which we denote as event *spoofing* and *masking* respectively. Spoofing attacks may cause severe damage, such as evacuation from a building when no real emergency is present. Similarly, masking attacks may prevent triggering an alarm in dangerous conditions, such as the presence of a natural disaster, which would hinder rescue and emergency management. Although different event detection applications have different tasks, they usually collect sensor measurements and interpret them to carry out a remedial response. Such response may have significant consequences and cost. Therefore, the measurements leading to the event detection become a critical resource to secure.

In event detection WSNs, malicious data injections may be particularly sophisticated. For instance, the characteristics of spoofed events may resemble genuine ones, whilst masked events may resemble rest conditions. Thus, there is the need for a mechanism, denoted with *detection*, which identifies malicious measurements even when the attacker maximises their similarity with genuine ones. Moreover, if colluding nodes act in concert in the attack, the compromised sensors are difficult to identify since a spoofed event may look similar to the remainder of a masked event, or a masked attack may split the genuine event into multiple parts that appear as spoofed events. A *characterisation* step is needed to deal with this problem, i.e. characterising the measurements affected by malicious interference and identifying the sensors responsible for the attack by reasoning about the possible dynamics that led to the set of observed measurements. Finally, faulty sensors may be confused with malicious sensors, especially when subject to a common-mode failure, which produces correlated measurements just like in the case of collusion. The detection step is generally able to detect inconsistencies in the measurements, but cannot distinguish between genuine and malicious interference. Thus, we devise also a *diagnosis* step to deal also with this problem. We address *detection*, *characterisation* and *diagnosis* of malicious data injections through different analyses of the measurement data, by extracting

the information that is shared among multiple sensor nodes. Indeed, the observed phenomena have some correlation properties, which are connected to physical laws, and such properties are induced in the measurements collected by sensors.

1.1 Contribution and Outcomes

The contributions and outcomes reported in this thesis are summarised below.

Literature Survey The first achievement of this work has been the identification of benefits and shortcomings of the current techniques aimed at detecting malicious data injections. The survey has revealed a large number of algorithms proposed for measurements inspection in sensor measurements. However, these tackled malicious data injections together with faulty measurements, as both may introduce anomalies in the data, without considering also the attacker's effort to evade detection. Malicious measurements are, by and large, more difficult to detect than faulty measurements, which instead show evident inconsistencies with genuine data. This holds true especially when multiple malicious sensors collude and produce measurements that are consistent with each other. Moreover, this study has suggested that the measurements' variability in the spatial domain deteriorates the detection performance. More precisely, a substantial decrease in performance is seen when moving away from a homogeneous space model, where all sensors perceive similar measurements, to heterogeneous space models, where measurements are expected to show notable variations in space, such as in the case of events.

Detection of Malicious Data Injections when Single Events Occur We have developed a first technique to *detect* malicious data injections in the presence of sophisticated collusion strategies among a subset of sensor nodes, when a single event occurs at a time. The detection algorithm exploits the internal measurements correlation and selects only information that appears reliable by filtering out the individual contributions from the measurements that do not support the general behaviour. Colluding sensors are not allowed to compensate for each other in the detection metric whilst still injecting malicious data thanks to an aggregation operator

that is accurate in the presence of genuine measurements as well as resistant to malicious data. Since detecting anomalies in measurements is not sufficient to counteract them effectively, the *characterisation* of the malicious measurements is also tackled, with a particular focus on avoiding being deceived by colluding nodes in this task. Since, in different event detection WSNs, the requirements and the nature of the events is markedly different, we also developed a methodology to tailor a detection algorithm to a specific application. This is achieved through customisation of the parameters based on a collection of historical data and information about the application's goals and requirements. The methodology for customisation, on the other hand, can be provided with a generic but well defined procedure. Experiments have been carried out in three different applications: health-care monitoring, monitoring of volcanic activity and home fire alarms. The results validated the approach, showing that it achieves high detection rates without introducing significant overhead in the sensor nodes.

Detection of Malicious Data Injections when Multiple Events Occur Since the previous approach is based on the use of two models, one used in the presence of events and the other one in their absence, it cannot cope effectively with events that are highly different in nature, such as events that affect only a localised area, or manifest with multiple sources. In such scenarios, each delimited area of a WSN can be subject to zero or more events, hence the presence of events should be evaluated locally. Moreover, multiple events, provide the opportunity for new, more complex attack strategies, such as creating false events near legitimate ones, transforming a widespread event into many narrower events etc. So, the initial approach has been reviewed and re-developed to cope with such complex scenarios which, in addition to detecting malicious data injections, also characterises the responsible sensors and can *diagnose* anomalies, i.e. infer when the anomaly is most likely malicious or genuine (e.g., caused by faults). Indeed, detection may also be triggered by genuine faults, as the measurements from faulty sensors do not correlate with those of healthy ones. This may lead to the wrong conclusion that there was an attack, hence the diagnosis step distinguishes malicious interference from faulty behaviours.

Evaluation against Sophisticated Attacks The novel method for detecting malicious data, which is described above, has been validated by testing it with sophisticated collusion-enabled attacks, produced using novel approaches that we have also designed:

1) A *mimicry* approach, which achieves event spoofing or masking by mimicking the genuine characteristics of the sensed phenomenon to minimise the chance of detection. This approach is used to evaluate the performance of both *detection*, *characterisation*, and *diagnosis*. An automatic test suite has been designed and implemented which addresses the main problems, i.e. selection of the genuine data that will be mimicked from an organised historical dataset, creation of malicious data, and contextualisation to minimise disruptions in correlations with current genuine data.

2) A *worst-case* approach, which constrains the attacker to stay undetected and to spoof or mask events, whilst minimising the attack resources that are needed. The latter are expressed in terms of sensor nodes that need to be compromised by an attacker, as a function of: application, deployment area and size, and number of measurements in control of the attacker. This work required an accurate model for the attacker and a careful design of the solving algorithm, which deals with an $\mathcal{NP}$-hard problem. The optimisation algorithm has been used for three different objectives: 1. Estimating the gain in resilience, quantified by the number of malicious sensor nodes that can be tolerated, offered by a measurements inspection technique. 2. Automatically retrieving the main vulnerabilities that may be exploited and the attacker's strategies that represent the highest risk for the WSN. 3. Providing a security-driven design methodology, i.e. an automatic tool that designs the WSN deployment to give a guaranteed degree of resilience to a certain amount of compromised sensors.

Increasing the Integrity of the Measurements through the Integrity of the Sensor Nodes The vulnerabilities highlighted by the latter study concern especially scenarios where the compromised area is contiguous and large compared to the typical event spread. Thus, measurements inspection has been complemented with occasional recourse to a more expensive technique that is effective also with many colluding sensors, i.e. software attestation. However,

running many attestations would impair the measurements inspection's benefits in terms of power consumption. This has required us to study how to minimise the use of attestation whist keeping its benefits. The results have shown that malicious nodes can be identified with high resiliency and low power overhead, which are the main advantages of attestation and measurements inspection, respectively. The combined technique is suited for a practical deployment in a real WSN, as it allows a WSN to operate autonomously by healing itself without frequent in field reconfigurations.

1.2 Thesis Outline

The remainder of this thesis is organised as follows.

Chapter 2 introduces the problem of measurements collection in WSNs and the corresponding problem of malicious data injections. We introduce the principles behind exploiting correlation, and introduce a unified notation to facilitate the comparison among different state-of-the-art approaches and also with our novel methods.

Chapter 3 reviews the related work. We identify the two main families of techniques that are used for measurements inspection: *anomaly detection* and *trust management*. For each of them, we identify the main techniques that are available in literature, explain how they make use of measurements correlation, and compare the performance of different techniques, whilst emphasising the shortcomings and limitations that prevent the deployment in real applications.

Chapter 4 introduces a model for the attacker, in terms of capabilities and possible damage that they can cause to the system. We show that a genuine subset of sensors is required for any measurements inspection technique to be applicable, give a formal definition of spoofing and masking attacks, and study the characteristics and possible effects of collusion.

Chapter 5 presents a solution for the problem of malicious data injections where the attacker is highly sophisticated and events occur in the monitored phenomenon. In particular one event at a time is considered, and with predictable pattern. We show that our approach, based on

robust aggregation of measurements estimate is more reliable compared with techniques that identify malicious measurements with majority voting. Moreover, we present a novel and more general methodology to apply the algorithm in different application settings.

Chapter 6 presents a complementary solution, which applies to multiple events that manifest with unpredictable patterns. We deal with the detection problem with a method based on the wavelet transform, which can evaluate the events characteristics regardless of their number, location, and size. We also deal with the *characterisation problem*, identifying groups of sensors with common behaviour and reasoning about their conflicts to identify compromised sensors. Finally we deal with the *diagnosis* problem by proposing a fault model, which is also based on the wavelet transform. The measurements are compared against this model to infer whether they have been compromised by faulty or malicious interference.

Chapter 7 gives two automatic testing tools to validate and evaluate the robustness of measurements inspection to malicious data, with different approaches: mimicry, which is a best-effort approach, and evasion, which finds the worst-case scenario. The wavelet-based methodology is evaluated with both approaches to: 1. Evaluate reliability under different sets of attacker capabilities 2. Test the robustness to attacks whose ability to cause damage whilst staying undetected is maximised. 3. Study the main weaknesses of measurements inspection techniques. 4. Provide an assured degree of resilience to a WSN at design time.

Chapter 8 shows how the data-integrity protection given by measurements inspection can be complemented with the node-integrity protection given by software attestation to obtain a highly accurate and power-efficient multi-layer integrity verification technique.

Finally, **Chapter 9** concludes this thesis by summarising the achievements, the practical impact on real applications and the future work that may complement the results obtained.

Chapter 2

Problem Definition and Notation

In this chapter, our objectives are characterised in light of the related work. A unified mathematical formulation is given for the problem of malicious data injections, which will be used for the following chapters to:

1. Define the problem unambiguously.
2. Define the measurements' correlations based on the formulation.
3. Describe and compare the related work with a coherent terminology as the terms used often differ from one article to another.
4. Describe the novel proposed techniques with the same terminology.

Finally, the ways to extract expectations on the measurements based on internal correlations are illustrated, making use of the formulation introduced.

2.1 Wireless Sensor Networks and Sensed Measurements

Measurements inspection is a data analysis technique that can be used to detect malicious measurements. Which is the entity that runs measurements inspection, and which is the data

that is being analysed are two important questions that affect the detection of malicious measurements. The choices are restricted by the architecture of the WSN and the measurements collection procedure, which are analysed in this section.

The typical workflow of a WSN starts with measuring a physical phenomenon through sensing devices connected to a wireless node that propagates the measurements through the network. The measurements are collected and aggregated by data sinks (e.g., base stations), and can then be interpreted or transmitted to a remote station. However, data can also be aggregated in the network by the intermediate transmitting nodes, with many possible variations on where and how data is aggregated. The choice between the different schemes is based on criteria that optimise power efficiency, number of devices, coverage of the physical space etc. Finding the optimal architecture based on such criteria remains an important research challenge.

Early work considered that all raw measurements are collected at the base station, which performs data fusion and other computations [She02; SWR98]. Later on, especially after the introduction of the LEACH protocol [HCB02], architectures became increasingly hierarchical. LEACH applies a one-level hierarchy where sensors are organised in clusters and communicate with the cluster-head, which, in turn, communicates with the base station, as shown in Fig. 2.1. Cluster-based protocols, and especially those where the clusters change dynamically in time [HCB02], have proven to be more energy efficient when communication with the base station requires multi-hop transmissions [HCB02].

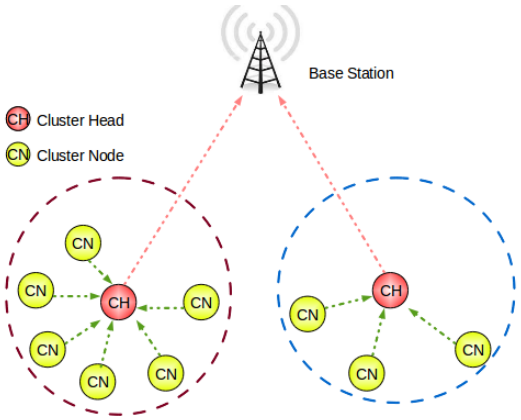


Figure 2.1: LEACH measurements collection architecture.

The one-level hierarchy introduced in LEACH can be generalised to tree-based structures as

described in [Fas+07]. Intermediate tree nodes may simply merge the packets generated by different sources into a single packet without processing the data, or after applying lossless compression. Alternatively, the sensor measurements can be processed with lossy aggregation operators, which prevent us to recover the original measurements. The aggregation can take place thanks to the cluster head, which could calculate, e.g., the mean, minimum, or maximum of the cluster's measurements. Another possibility is to take only a subset of measurements thanks to compressive sensing techniques [CW08].

Measurements inspection algorithms are often run by data sinks, since they can compare different measurements with each other. However, if the measurements are subject to a lossy compression whilst being collected, the data sink cannot analyse all the measurements. The information discarded in lossy compression is considered redundant; nevertheless, this is true only if all sensors are genuine. On the contrary, when some sensors are malicious, even perfectly correlated sets of measurements may be useful. Indeed, if perfect correlation is expected between the measurements of two sensors and one is malicious, this will be detected as soon as its measurements change.

Measurements inspection techniques are still applicable if in-network aggregation with size reduction is in place. For instance, a possible way of keeping lossy measurements compression without impairing detection of malicious data may be to ascertain part of the discarded data, e.g. by asking the cluster head for proofs about the individual measurements collected [PSP06]. However, as we will discuss in detail in Section 5.5, the computational advantages given by lossy aggregation are overcome by the disadvantages of requiring extra-communications and reducing the ability to detect malicious measurements. For these reasons, the following discussions will assume that the data sinks in charge of running the measurements inspection algorithm can access the raw measurements sent by the sensor nodes. Below, we will discuss which are the objectives that a measurements inspection algorithm seeks to achieve, and how this relates to the measurements sent by the sensor nodes and to their correlations.

2.2 The Objectives of Measurements Inspection

The main challenge in detecting malicious data injections is finding sufficient evidence of the attack. Measurements inspection techniques assume that the attacker aims to cause noticeable undesired effects and injects measurements that differ in some detectable way from the correct values that should be reported at that point in time and space. This is the assumption that enables the use of data analysis to detect malicious data. However, the real value that *should* be reported is not observable directly, but can only be characterised from indirect information which, under genuine circumstances, is correlated with the real value. However, there may be other different reasons for disruptions in such correlations, including faults, discontinuities in the sensed environment, or naturally occurring events. *Faults* refer to any kind of genuine errors, transient or not, and may be difficult to distinguish from a malicious injection. *Events* refer to substantial changes in the sensed phenomenon like a fire, an earthquake etc. We refer to the problem of distinguishing malicious data injections from faults as *diagnosis*. In general, also distinguishing events is a diagnosis task when events are detected through *anomaly detection*. However, in this thesis we will consider that events are identified by a separate *event-detection algorithm*.

Moreover, changes in correlations that are due to malicious interference are not necessarily obvious to identify, especially when more than one sensor are compromised. Indeed, if multiple compromised sensors *collude*, i.e. produce malicious values in a coordinated fashion, the information coming from genuine nodes may not be sufficient to detect the compromise as the malicious measurements could endorse each other. Then, we need to model the relationship between different measurements in a way that is robust to malicious colluding attacks. Below, we characterise the measurements collection problem, and how it is affected by malicious interference, to study the robustness of different measurement correlation models to malicious data injections.

2.2.1 A Characterisation of the Problem

To characterise the measurements data in WSNs, we consider N sensors, whose locations belong to the set Ω, which may be two dimensional if the sensors are placed approximately on the same plane, three-dimensional if they have, e.g., different altitudes, or even six-dimensional if both their location and their orientation are relevant to the application. Every sensor measures a physical attribute such as temperature, wind, water quality, power, gas flows etc., which can be modelled as an ideal function of time and space φ. We refer to this function as the *physical attribute function*. The sensors' goal is to sample this function in space and time, i.e. retrieve the function's values at a finite set of locations $\mathbf{u} \in \Omega$ and a finite set of time instants $i \in 1, ..., T$[1]. However, this operation introduces a measurement error, leading to the value $x_i(\mathbf{u})$.

$$x_i(\mathbf{u}) = \varphi(t_i, \mathbf{u}) + \epsilon_N(t_i, \mathbf{u}) + \epsilon_E(t_i, \mathbf{u}) \quad \mathbf{u} \in \Omega, i \in 1, ..., T \tag{2.1}$$

Where $\epsilon_N(t_i, \mathbf{u})$ represents the contribution of usual noise, while $\epsilon_E(t_i, \mathbf{u})$ is the contribution of *genuine errors*, i.e. those caused by non-malicious faults. Malicious interference, regardless of its nature, introduces a perturbation, which we refer to as *manipulation*, and is defined as the difference between $x_i(\mathbf{u})$ and the measurement that is *collected* at the i-th time instant from the sensor located at $\mathbf{u}$. The measurement collected is denoted with $\breve{x}_i(\mathbf{u})$ and is defined below.

$$\breve{x}_i(\mathbf{u}) = x_i(\mathbf{u}) + m_i(\mathbf{u}) \quad \mathbf{u} \in \Omega, i \in 1, ..., T \tag{2.2}$$

Where $m_i(\mathbf{u})$ is the effect of malicious interference. For instance, if malicious injections take place through malicious software, the malicious node may receive $x_i(\mathbf{u})$ in input from the sensor, but transmit $\breve{x}_i(\mathbf{u})$ to the sink.

The purpose of measurements inspection is to:

[1] We focus on interpreting the data and abstract from implementation-related issues such as synchronisation between sensors, and network related issues such as packet loss or delays.

- Infer if any collected measurement has been corrupted by malicious interference, i.e. there is a $m_i(\mathbf{u}) \neq 0$ component. (Detection)
- Identify the subset $\{\mathbf{v} \in \Omega : m_i(\mathbf{v}) \neq 0\}$. (Characterisation)

However, the collected sensors' readings $\breve{x}_i(\mathbf{u})$ are the only observable quantities. Moreover, assuming that errors and malicious interference are unpredictable, measurements inspection can extract only information about the expected behaviour of the physical attribute and noise, through a modelling process. However, when considering also genuine faults, it is not possible to state if the observed measurements do not comply with the model because of the manipulation or the error component. Gathering such information requires further analyses. Thus, the generic flow of measurements inspection is:

- Infer if any collected measurement has been corrupted by malicious interference, i.e. there is a $m_i(\mathbf{u}) \neq 0$ component. (Detection)
- Identify the subset $\mathcal{C} \coloneqq \{\mathbf{v} \in \Omega : m_i(\mathbf{v}) \neq 0 \vee \epsilon_E(t_i, \mathbf{v}) \neq 0\}$. (Characterisation)
- Distinguish $m_i(\mathbf{v}) \neq 0$ from $\epsilon_E(t_i, \mathbf{v}) \neq 0$ $\forall \mathbf{v} \in \mathcal{C}$. (Diagnosis)

The role of diagnosis and characterisation is discussed in detail below.

Diagnosis and Characterisation of Malicious Data Injections While detection consists mainly of identifying measurements that do not comply with a model, finding the cause for the divergence pertains to the problem of *diagnosis*. This task, needs to carefully distinguish among different causes with common effects.

In WSNs two main phenomena can produce similar deviations from expected behaviour: faults and events of interest. Faults represent generic unintentional errors introduced e.g., by obstacles in the environment, sensors' battery depletion, pollution, fouling etc. Events of interests represent environmental conditions that manifest under exceptional circumstances, but are interesting as they can reveal an alarm scenario e.g., heart attacks, fires, volcanic eruptions etc.

If the measurements model includes also events, i.e. events do not trigger the detection step, then diagnosis reduces to the distinction between faults and malicious compromise.

Information about the cause of an anomaly or of an untrustworthy sensor can be precious. With fine-grained knowledge about the nature of the problem, an appropriate response can be initiated to address it. Unfortunately, the measurements inspection literature lacks an exhaustive diagnosis phase [IL15b].

Additionally, even when the presence of an attack can be ascertained with confidence, further information is needed to determine the course of action to be taken. For example, there is the need to know the attack's effects and the system area (nodes) affected by the attack. We refer to this other task as the *characterisation* of the attack. If *detection* and *diagnosis* of malicious data injections answers the question "Is there an attack?", *characterisation* answers questions such as "Which are the compromised sensors?" and "How is the attack performed?" This information is valuable since it drives the reaction to the attacks, e.g. it tells the network operator which devices need to be reconfigured and which vulnerabilities should be covered with the highest priority.

Detection, characterisation and diagnosis all rely on a data model, which describes the expected behaviour of the measurements under different circumstances. Note that while we can model the measurements behaviour under the presence of faults or events, malicious behaviours should not be modelled to avoid failures of the measurements inspection technique in the presence of unforeseen attacks. We introduce below the techniques that allow the detection of malicious data injections without restricting the applicability to a well-known set of attacks.

2.3 Modelling Expected Behaviour

Malicious data injections have been addressed in literature with both *anomaly detection* techniques (e.g., [TH06; LCC07; Sun+13]) starting from about 2005 ([TH06]) and with *trust management* (e.g., [Ata+08; Bao+12; OHC12]) from about 2006 ([ZDL06]). While anomaly detection defines normal behaviours to infer the presence of anomalies, trust management evaluates

the confidence level (trustworthiness) that a sensor's behaviour is normal. Compromised sensors are expected to get low trust values when some *expected behaviour* in the data is not respected in the opinion of the sensors which hear the measurements reported. Although anomaly detection is also based on the definition of expected behaviour –"Anomaly detection refers to the problem of finding abnormalities in the data that do not conform to expected behaviour" [CBK09]– the two approaches differ in how deviations are interpreted. In trust management, the measurements data are analysed with the granularity of a sensor, and each sensor has a trust value that is incrementally updated in time. Anomaly detection approaches, instead, can be applied with no restrictions in granularity from the single measurement to the whole system, and generally work by defining a boundary for expected behaviour such that everything outside that boundary is abnormal.

In our context, expected behaviour refers to a set of properties characterising the measurements that are free of malicious injections. In general, describing the unobservable real measurement in terms of observable properties is a modelling process, that makes assumptions about how data can be described. The relation that links the problem to a model is a one-to-many relation. Different models of the same problem are not equivalent and choosing a good model is essential for good performance. In particular, a good model should be characterised by:

- **Accuracy** – No model is perfect and every model is in fact an approximation. An accurate model minimises the approximation error.
- **Adaptability** – Physical attributes measured by the sensors change in time. As a consequence, models should cope with dynamically changing environments.
- **Flexibility** – Good models should be applicable in a flexible way, regardless of the application. Such models should abstract as many details as possible and capture only those properties that are needed.

These desirable characteristics conflict with each other: accuracy may be better achieved with context-specific details, which limit flexibility and compromise adaptability. A particular adaptability requirement which significantly affects accuracy and flexibility is the sensors' *mobility*,

as when sensor nodes migrate to new locations, previous expectations are invalidated. Indeed sensor migrations may completely change the physical attribute function. Even though mobility is an aspect that is not directly addressed in the detection of malicious data injections, some techniques are more suited to support mobile sensors than others. In particular, anomaly detection techniques that compare the measurements within a neighbourhood without considering past behaviour (e.g. [Han+75; NLL06; LCC07; Wu+07; GZH09]), can generally accommodate mobility, since for every time instant, new expectations are extracted. However, such techniques also need to become aware of topology changes in the presence of mobility.

Trust-management techniques with exchanges of trust information (e.g., [Bao+12; HLT05; GBS08; MCA08]) are also suited for mobility, since a sensor A which migrates to a new area and becomes a neighbour of B, can benefit from recommendations from sensors which have been B's neighbours in the past [Zah+10]. So far, exchanges of trust information have been considered without investigating the effects of mobility, therefore sensor A will generally maintain indirect information about sensor B only if there is interaction between A and B, and A cannot observe B's behaviour (e.g., it is not in the wireless communication range). When sensors are mobile instead, even if A and B never interacted, they may interact in the future if they get closer. Only at that time, do recommendations for B become of interest to A. Nevertheless, a criterion to request such recommendations is still missing in literature.

Anomaly detection and trust management define the expected behaviour based on modelling inter-measurement correlations. There are different choices available in literature concerning the domain in which correlations are identified and how the are used to build expectations.

2.3.1 Exploiting Correlation

Since the original measurements substituted with fabricated ones cannot be observed directly, they need to be characterised indirectly with related information. The relationship between two pieces of information is a *correlation*, which can be calculated online, with historical data, or modelled a-priori. In either case, coexistence of genuine and compromised measurements

may cause disruptions in correlation, assuming that the correlations have not changed between the moment when they are calculated and the moment when they are used.

We refer here to *correlation* in a broad sense, meaning that there is some kind of continuous dependency, as opposed to Pearson's correlation coefficient, which is the most commonly used correlation metric. Referring to E, μ and σ as the expected value, the mean and the standard deviation respectively, the Pearson correlation coefficient ρ_{XY} between two random variables X and Y is given by:

$$\rho_{XY} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{2.3}$$

Note that this coefficient measures only linear dependency between two variables, while non-linear dependencies may be missed. Moreover, the expected Pearson correlation may vary in time as physical phenomena are generally non-stationary.

In Wireless Sensor Networks we can generally consider correlations across three different domains: *temporal*, *spatial* and *attribute* domain [RZM13].

- **Temporal correlation** is the dependency of a sensor's reading on its previous readings. It models the coherence in time of the sensed physical process.

- **Spatial correlation** is the dependency in readings from different sensors at the same time. It models the similarities in how the sensed phenomenon is perceived by different sensors.

- **Attribute correlation** is the dependency in readings that are related to different physical processes. It models physical dependencies among heterogeneous physical quantities such as temperature and relative humidity.

Usually a combination of these different kinds of correlation is used. We now analyse how they contribute to the definition of expected data.

2.3.2 Temporal Correlation

Variations in time of the sensed data can be modelled as a random process [Bou09], where the random variables at different times are correlated. In (2.1) we observe that the variation of a sensor's measurements in time depends on both the variations introduced by the physical attribute and the measurement error. The variation of the physical attribute in time is subject to constraints, such as the presence of gradual changes, or the alternation of some patterns, since the phenomenon observed usually follows the laws of physics. So, if the measurements are gathered with sufficiently high frequency, consecutive measurements would be subject to similar constraints. This simple observation justifies a procedure that identifies errors (including malicious injections) when temporal variations do not respect these constraints. However, there are two main difficulties in applying this observation to assess deviations: the time evolution of the process is subject to uncertainty and the measurements are subject to noise.

When using Kalman filters [Kal60], these two factors are known respectively as *process noise* and *measurement noise*. The measurement noise is typically modelled as a Gaussian process. The process noise, instead, comes from the imperfections of the model used to describe the process dynamics. In [Sun+13], the physical process is modelled as a discrete Markov process, and a Kalman filter is used to estimate the value at time t_1 as:

$$\varphi(t_1, \mathbf{u}) = F(\varphi(t_0, \mathbf{u})) + w(t_0) \tag{2.4}$$

where F models the expected evolution of the time process and w is the process noise. This approach is then used for change-point detection to detect attackers that abruptly modify the measurements time series of a sensor.

Although temporal correlation is useful to detect genuine anomalies, such as sensor failures, it is not reliable enough to detect malicious interference. Indeed, the temporal domain is completely under the attacker's control during the time when the sensor is compromised, so the attacker can keep temporal correlation consistent by introducing slow enough transitions from genuine to malicious measurements. Temporal attacks, i.e. on single sensors are well researched in the

literature [Sub+06; BHL07; Sun+13] and are successful when the attacker is constrained to a maximum time.

2.3.3 Spatial Correlation

In the presence of sudden events, the dynamics of a physical process can change rapidly. Often detecting such events, such as a forest fire, a volcanic eruption, a cardiac attack etc., is the very purpose of the WSN. The occurrence of events increases the complexity of detecting malicious measurements since events may disrupt temporal correlations, giving rise to false anomalies. Nevertheless, different sensor nodes generally are affected by the event and produce measurements that are spatially correlated to the event source: as a consequence, the measurements of different sensors are correlated during the manifestation of the event [Bou09]. This phenomenon is known as spatial correlation.

The main advantage of spatial correlation is that it requires the attacker to compromise more sensors to keep correlations consistent. Indeed, if the measurements of any genuine sensor are expected to be highly correlated with those of another sensor that has been compromised, significant data manipulations will be detected. Moreover, the compromised sensors are required to coordinate themselves, i.e. collude, to prevent inconsistencies among the malicious sensors themselves.

Spatial correlation can be modelled in different ways, with different amount of restrictions about the coherence of the physical process in space. Therefore we analyse the different models with a particular focus on their generality, which determines applicability, and on the correlation extraction procedures.

The most widespread spatial correlation model is also the simplest: it assumes that all sensors would produce the same measurements in the absence of errors and noise i.e., they measure the same value, and we refer to this model as *spatially homogeneous* [ZDL06; NLL06; Wu+07; LCC07; BHL07]. In terms of the physical attribute $\varphi(t_i, \mathbf{u})$, it is considered a function of time only. Under this assumption, detecting sensors with abnormal readings becomes then a simple

matter of detecting deviations from the spatial measurements' distribution and the accuracy of the distribution estimation increases with the number of sensors.

The homogeneous model is suitable only for regions of space which are small enough and free of obstacles. However, when the deployment topology and characteristics of the physical phenomena violate the homogeneity assumption, the spatial propagation rules can still induce spatial correlations. A first attempt to relax the monotonicity assumption is made in [GZH09], where spatial variation is considered but it is assumed monotonic. This implies that the values of the physical attribute at a point in space, should either increase or decrease as the distance from that point increases. To ascertain whether this property holds, Guo, Zhong, and He [GZH09] divide the deployment space into sections, called faces. For each face, the authors construct a "distance sequence", corresponding to the sequence of sensors ordered by the distance from that face. While sensing the phenomenon, the sensors readings are sorted to generate the *estimated sequence*, which is then compared to all possible distance sequences, as shown in Fig. 2.2. The

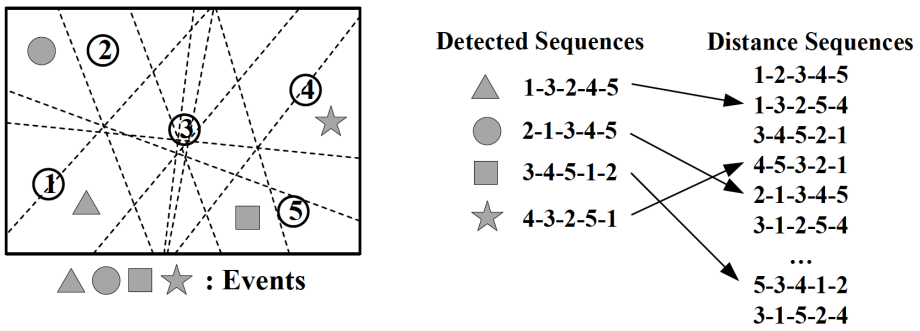


Figure 2.2: Detection of measurements which do not comply with the monotonicity assumption, from [GZH09].

sensors measurements are consistent with the expectation if the estimated sequence corresponds exactly to one of the distance sequences.

Instead of considering a strict assumption like the monotonicity of the measurements, it is possible to model correlations between the sensors' readings as a function of their spatial positions. An example of such a model is the *variogram*, defined as the variance of the difference between values of a physical phenomenon at two locations. In our notation, the variogram between two locations $\mathbf{u_1}$ and $\mathbf{u_2}$ is defined as $var(||\varphi(t_i, \mathbf{u_1}) - \varphi(t_i, \mathbf{u_2})||)$. When the physical phenomenon is assumed to be isotropic, the variogram is expressed as a function of the distance only. Zhang

et al. [Zha+12] have applied it to compute an expected measurement as a function of the measurements from other sensors. Note that in the presence of obstacles, the variogram is not only a function of the distance, but also depends on the absolute positions.

Rather than considering distances between sensors, spatial correlation can be calculated as a function of the sensor values themselves. This choice caters for sensors at the same distance, but subject to different noise or obstacles in space. However, it comes at the price of correlation updates when sensors are mobile. For example, Sharma, Golubchik, and Govindan [SGG10] express a sensor's measurement as a linear combination of the measurements from the other sensors, extract the function's parameters and derive expected sensor readings. Dereszynski and Dietterich [DD11] instead, derive expected readings by fitting the joint probability distribution of the measurements from N sensors, after assuming it is an N-variate Gaussian distribution. Note that this approach also implicitly assumes a linear model, as the covariance between two random variables captures linear dependencies (we have mentioned in Sect. 2.3.1 that this is true for the Pearson correlation coefficient, which is just a normalisation of the covariance index).

Not infrequently, spatial correlation is used in conjunction with temporal correlation, since they capture different kinds of deviations. For example, Bettencourt, Hagberg, and Larkey [BHL07], propose an outlier detection technique based on two kinds of differences: between a sensor's reading and its own previous reading (temporal correlation) and between readings of different sensors at the same time (spatial correlation). A distribution for both differences is used to check if data samples are statistically significant as related to the temporal domain as well as to the spatial domain.

2.3.4 Attribute Correlation

In the same WSN, sensors observing different physical attributes such as light, vibrations, temperature etc., may coexist. Some of these attributes may be correlated because of the physical relationship between them e.g., temperature and relative humidity. Commonly, at every deployment location, many sensors in charge of measuring different physical processes

could be connected to a single sensor node. As described in (2.5), for a fixed point in space and time we have a set of A physical attributes. We define attribute correlation as the correlation between them.

$$\varphi^a(t, \mathbf{u}) \quad a \in 1, ..., A \tag{2.5}$$

Attribute-based expectations are very useful when spatial redundancy is limited. For example, body sensor networks for healthcare have limited redundancy since it is impractical to cover the patient with several sensors. We can then still exploit correlation among different physiological values (the attributes) measured by different sensor nodes. However, in the context of malicious data injections, attribute correlation is useful only when combined with spatial correlation. Indeed, if the measurements of two attributes come from the same node (implying a combination with temporal correlation), the attacker controls both by controlling the node. Instead, if they come from different nodes (implying a combination with spatial correlation) the attacker needs to control both nodes to keep attribute correlation consistent.

2.3.5 Overview of Measurements Correlation Types

In the previous sections we have analysed different types of correlations, the information they capture, and variations in the exploitation of the same correlation type. In Table 2.1 we summarise this analysis.

2.4 Conclusions

In this chapter we have introduced a unified notation for describing the variables involved in the measurements collection process in WSNs, as well as the components introduced by faults and malicious data injections.

We used the notation introduced to describe the different kinds of measurements correlations.

Table 2.1: Correlation Types

Correlation Type	Information Captured	Variations
Temporal	$\text{corr}(\varphi^a(t_1, \mathbf{u}), \varphi^a(t_2, \mathbf{u}))$	• Time-series evolution model • Time memory (the maximum gap between two correlated time instants)
Spatial	$\text{corr}(\varphi^a(t, \mathbf{u_1}), \varphi^a(t, \mathbf{u_2}))$	• Spatial model, e.g. homogeneous, monotonic, variogram, linear dependency • Correlation variational model, e.g. distance-dependent, sensors-dependent, fixed • Neighbourhood selection criterion
Attribute	$\text{corr}(\varphi^{a_1}(t, \mathbf{u}), \varphi^{a_2}(t, \mathbf{u}))$	• Correlation extraction process, e.g. from physical laws, temporal/spatial analysis etc.

In the following chapters, the same notation will also serve to review the related work and to present novel techniques as well.

We have considered three different types of correlations: temporal, spatial, and attribute correlation. We have also identified correlation models with remarkably different levels of abstraction, especially for the spatial domain. As we will see in the next chapter, each model choice will affect significantly the techniques' performance and applicability.

We have discussed that spatial correlation is preferable when dealing with malicious interference because it requires the attacker to compromise more sensors. Temporal correlation can be kept consistent if the sensor is kept under the attacker's control for the duration of the attack, hence it is appropriate only for non-colluding attacks or for genuine anomalies. Attribute correlation, instead, is generally used to increase the reliability of either temporal or spatial correlation.

In the next chapter we look in detail at the related work, using the notation introduced here. We will compare different works in term of approach, performance, applicability, computational complexity and issues addressed.

Chapter 3

Related Work

The integrity of the sensed data can be verified through measurements inspection techniques, i.e. techniques that analyse the measurements and are able to infer the presence of malicious compromise. Existing studies advocate the use of techniques designed for detecting faulty sensors or faulty data also for malicious data injections. Comparatively, only a small proportion of papers focuses on the specific problems introduced by malicious data injections. Indeed, there is a significant difference between faults and maliciously injected data since the latter is *deliberately* created in sophisticated ways to be difficult to detect. Therefore, there is a need to:

1. Analyse the achievements and shortcomings of the work targeted to malicious data injections.

2. Review the state-of-the art techniques proposed for non-malicious data compromise and evaluate their suitability to this problem.

We do so in the following and compare studies according to their:

1. Adopted approach.

2. Scope: i.e detecting malicious data injections, faults, or events.

3. Results and performance.

This chapter is structured in this way: In Section 3.1 we individually review the techniques for detecting the presence of abnormal measurements, distinguishing the techniques that pertain to anomaly detection from those pertaining to trust management. In Sections 3.2 and 3.3 we describe the state-of-the-art for the techniques that complement the detection task, i.e. diagnosis and characterisation respectively. The different techniques are compared in Section 3.4 according to the approach, performance, and overhead. Finally, in Section 3.5, we summarise the conclusions.

3.1 Detecting Deviations from Expected Data

Measurements inspection techniques define some expected properties for the measurements and provide a method to evaluate the deviation of the reported measurements from expectation and detect abnormal behaviours. The detection task has been addressed with anomaly detection techniques and with trust management, which have a similar way to inspect the measurements, as both build an expectation of the measurement values, but use different criteria to cope with abnormal data. Specifically, anomaly detection uses the expectation to discriminate between anomalous and normal data. Trust management instead, uses a criterion to map the deviation from expected data to a trust value.

3.1.1 Anomaly Detection Techniques

Anomaly detection is a method to characterise data as normal or anomalous. In contrast to Rajasegarar, Leckie, and Palaniswami [RLP08] who consider outlier detection and anomaly detection as equivalent, in the following outlier detection is considered one of the techniques belonging to the anomaly detection category. The reason for this choice is that outlier detection identifies the samples that are unlikely to manifest. However, the measurements could be anomalous with respect to other criteria, that cannot be reduced to the problem of finding

outliers. To clarify this aspect, we present statistical tests for anomaly detection and highlight their differences with more traditional outlier detection techniques. Then we delve into techniques for outlier detection, which is still the most commonly adopted technique for anomaly detection.

Statistical Tests

Techniques based on statistical tests assume a probabilistic data distribution. Real data is then checked against this distribution to verify its compliance with it. Techniques based on statistical tests are more general than outlier detection because they check the compliance of both outliers and non-outliers with the distribution whereas outlier detection focuses on the *classification* of single data samples.

For example, Rezvani et al. [Rez+13] use a technique based on statistical tests to detect malicious colluding nodes. They assume spatial homogeneity, i.e. that all sensors should perceive the same value, and model sensor measurements as a ground-truth value plus some noise. The ground truth is estimated as a weighted average of measurements, where the weights are as high as the measurement is close to the weighted average calculated in the previous iteration. The difference between the estimated value and each measurement is assumed to be normally distributed. Compliance with the normal distribution is then assessed with the Kolmogorov-Smirnov test, which quantifies the distance between an empirical distribution (the errors distribution) and a reference distribution (the normal distribution).

Outlier Detection

Outlier detection methods consider as anomalous data that lies outside of the space where most data samples lie. This technique can identify malicious data injections reasonably effectively as long as maliciously injected values are a minority in the dataset and deviate significantly from the other data.

Historically, outlier detection has been proposed in WSNs for different purposes, sometimes with

opposing goals: in some cases the techniques aim to filter out outliers, in others the outliers represent the main interest. For example, outliers are filtered out to increase data accuracy [JAP06] and for energy saving [Raj+09]. Applications where outliers are the main interest include fault detection [PH07], event detection [Bah+09; Zha+12] and detection of malicious data. We describe below different approaches to the outlier detection problem independently of the application context, but we focus on those techniques that can be applied to detecting malicious data injections.

Nearest-Neighbour-Based Outlier Detection In nearest-neighbour based outlier detection, an outlier is a data sample with a narrow neighbourhood, where a neighbourhood comprises the data samples within a certain distance. Most nearest-neighbour based techniques in WSNs are inspired from the well-known LOCI method [Pap+03], which calculates for every sample, the number of neighbours in a data space characterised by the radius αr, where α is a parameter used to reduce computational complexity. The relative difference with the average number of neighbours, i.e. the samples within a radius r in the data space, constitutes the *Multi-Granularity Deviation Factor* (MDEF). The MDEF is compared to a threshold equal to 3 times the MDEF standard deviation to ensure that less than 1% values are above the threshold when the distances between data samples follow a Gaussian distribution (the percentage increases up to 10% for other distributions). Note that this method is applicable to malicious data injections by considering the sensors' measurements as the data samples. However, the research community seems to have somewhat lost interest in approaches based on nearest-neighbour since they have large computational overheads due to the calculation of the neighbours for each new data sample.

Clustering-Based Outlier Detection Clustering is another technique often used for outlier detection. Here the outliers are the elements distant from the others, after organising close elements into clusters. For example, Rajasegarar et al. [Raj+06] identify a cluster as anomalous if its distance to other clusters is more than one standard deviation of the distance of the cluster elements from the mean. The main drawback of this approach is that an attacker that controls

multiple colluding measurements can keep the inter-cluster distance low by introducing intermediate clusters, i.e. clusters of malicious measurements that have a low inter-cluster distance, and that decrease the inter-cluster distance for other clusters of malicious measurements.

PCA-Based Outlier Detection Principal component analysis (PCA) [Mar09] is a common data analysis technique, that has also been applied to find outliers [CP07]. PCA is based on a projection of the k-dimensional data space onto another k-dimensional data space, where the variables describing the data samples are linearly uncorrelated. This transformation is defined in such a way that the projected variables are sorted with descending variance. The first p out of k variables are defined as the *principal components* and can be projected back to the original data space to obtain a prediction vector y_{norm} [JM79], also referred to as *normal data* [CP07]. The difference between original and normal data constitutes the *residual vector* y_{res}. Residual vectors that are large in magnitude (i.e., when the squared prediction error $SPE = \|y_{res}\|^2$ of the residual vector is greater than a threshold) are interpreted as deviations from the predicted (normal) vector and considered as outliers [CP07]. PCA can be applied to k-dimensional datasets e.g., made up of the measurements time series of k sensors [CP07]. In this case y_{res} reflects changes in spatial correlation but the same idea can also be applied to the temporal or attribute domains.

Classification-Based Outlier Detection Traditional classification techniques learn how to recognise samples from different classes. Anomaly detection considers two classes: anomalous and normal; however, anomalous data samples are rarely observable compared to the normal ones. Therefore, classification for anomaly detection is generally reduced to a one-class classification problem, based on the observation of normal samples only.

Normal and anomalous samples can be viewed as points within two different regions of the data space. Finding the boundary that separates the two regions may be infeasible, because the regions overlap and, even when a boundary exists, it may have a complex shape. Support Vector Machines (SVM) are a classification technique that can overcome this limitation by projecting the data samples into a higher dimensional space. In the projected data-space, a

boundary that separates normal from anomalous points may exist even if it does not exist in the original space, or may have a simpler shape. For example, the normal samples could be contained within a sphere in the projected data space. When the data space contains only positive values, this problem reduces to a special type of SVM called *one-class quarter-sphere SVM* [Las+04], which is represented in Fig. 3.1. With this approach, the classification problem

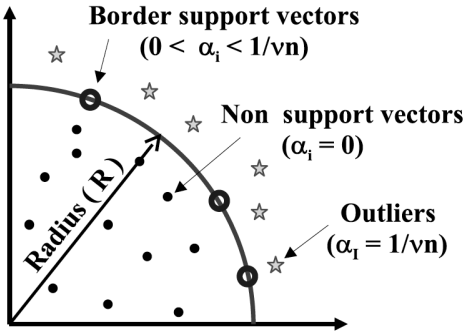


Figure 3.1: One-class quarter-sphere support vector machine, from [Raj+09].

reduces to finding the sphere's radius. Depending on how the WSN dataset is given in input to quarter-sphere SVM, the classification can be made across its time domain [Raj+09], attribute domain, or both [SNQ12] .

Bayesian networks have also been applied in WSNs to detect outliers with a classification-based approach. A Bayesian network defines the relations of conditional independence between random variables through a network graph. In WSNs, the random variables can be different values in space and time of the physical attributes.

An example of application of Bayesian networks to WSNs is given by Dereszynski and Dietterich [DD11]. The physical attribute $\varphi(t_i, \mathbf{u})$ is modelled as a random variable, which is correlated with the previous sample $\varphi(t_{i-1}, \mathbf{u})$ (1^{st}-order Markov relationship) and with values at different locations $\varphi(t_i, \mathbf{v}v \in \Omega)$. The aim is to find the state of a sensor, modelled by a random variable with two possible values: *working* and *broken*. The posterior probability of the measurements, which depends on both the physical attribute and on the sensor state variable, is maximised with respect to the state variables to identify faulty nodes. Dereszynski and Dietterich [DD11] evaluated their approach assuming that faulty sensors have a high increase their measurements'

variance (by 10^5), motivated by the observation that the measurements of faulty sensors appear more noisy. Though reasonable in the case of faults, this assumption does usually not hold for data injections, where an attacker can choose the measurements distribution arbitrarily and wishes, in most cases, to remain undetected.

Statistical Outlier Detection Statistical outlier detection identifies outlying data samples through statistical characterisation of the tail of the samples' probability distribution, as shown in Fig. 3.2.

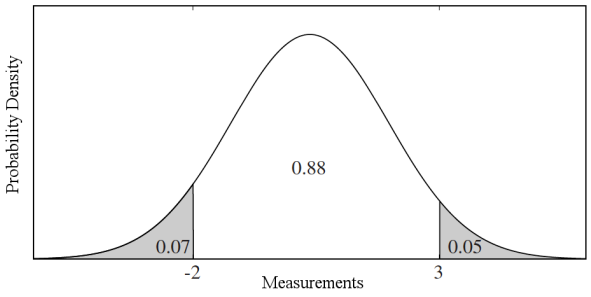


Figure 3.2: Statistical characterisation of the sensed data for outlier detection, from [BHL07].

Note that this approach differs from anomaly detection based on statistical tests, as it does not test the samples' compliance to their expected distribution, but only identifies the outliers that lie on the tails of the distribution. For example, outliers can be defined as samples far from the mean. Ngai, Liu, and Lyu [NLL06] have applied this idea to measurements from different sensors, thus exploiting spatial correlation. The *spatial* sample mean $\hat{\mu}_S$ of measurements from N different sensors at the i-th time instant is defined as:

$$\hat{\mu}_i^S = \frac{1}{N} \sum_{j=1}^{N} \hat{x}_i(\mathbf{u_j}) \tag{3.1}$$

Ngai, Liu, and Lyu [NLL06] use it to evaluate the deviation of sensor j from the spatial mean, compared to the magnitude of the mean itself with the metric: $f(j,t) = \sqrt{\frac{(\hat{x}_i(\mathbf{u_j}) - \hat{\mu}_i^S)^2}{\hat{\mu}_i^S}}$.

Similarly Tanachaiwiwat and Helmy [TH06], use the metric $t^* = \frac{\hat{x}_i(\mathbf{u_k} - (\mu_{i,j}^T \pm \delta)}{S_{i,j}^T / \sqrt{W}}$, where $\mu_{i,j}^T$ and $S_{i,j}^T$ are respectively j's temporal mean and sample standard deviation in the time window $[i - W + 1, i]$ and δ is a priorly known variation between sensor j and k due to the observed

phenomenon's spatial propagation. Considering the model in Sect. 2.2.1, a generic sensor j calculates its *temporal* sample mean in the W-wide time window $[i - W + 1, i]$ as:

$$\hat{\mu}_{i,j}^{T} = \frac{1}{W} \sum_{t=0}^{W-1} \hat{x}_{i-t}(\mathbf{u_j}) \tag{3.2}$$

The *temporal* standard deviation is instead calculated as:

$$S_{i,j}^{T} = \sqrt{\frac{1}{W-1} \sum_{t=0}^{W-1} (\hat{x}_{i-t}(\mathbf{u_j}) - \hat{\mu}_{i,j}^{T})^2} \tag{3.3}$$

The value of t^* is then compared with a threshold, that is set to 3 since, in normally distributed data, this accounts for approximately 99.7% of the population (the percentage decreases to 90% for other distributions).

In some cases the median is preferred to the mean, since the former has the advantage of being insensitive to outliers. Indeed, a problem in outlier detection is how to find the general (non-outlying) trend from data affected by outliers. The mean is sensitive to outliers, since it is proportional to the magnitude of each operand. The median takes instead one element to represent all the others. Wu et al. [Wu+07] use the median operator to aggregate sensors measurements in a neighbourhood. We can refer to it as a *spatial median*. If we order the N sensors measurement at the i-th time instant such that $\hat{x}_i(\mathbf{u_1}) \leq \hat{x}_i(\mathbf{u_2}) \leq ... \leq \hat{x}_i(\mathbf{u_N})$, the median in the spatial domain is calculated as:

$$\tilde{\mu}_S = \begin{cases} \hat{x}_i(\mathbf{u_{(N+1)/2}}) & \text{if N is odd} \\ \\ \hat{x}_i(\mathbf{u_{N/2}}) & \text{if N is even} \end{cases} \tag{3.4}$$

After calculating the difference between the median and each value, there are two possibilities: comparing each difference to the measurements magnitude, or comparing it to the general distribution of the differences. Yang et al. [Yan+06] and Wu et al. [Wu+07] detect outliers in the differences, assuming they are normally distributed. Instead of relying on the assumption of a Gaussian distribution, the probability distribution can also be estimated from the data [BHL07].

When sensing multiple physical attributes, the distribution of the measurements across all attributes can be considered, rather than a separate distribution for each one. This approach can potentially detect outliers that a separate approach would fail to detect. Liu, Cheng, and Chen [LCC07] combine different attributes using the Mahalanobis distance, which is based on the inter-attribute correlation and defines how the data is statistically distributed in the attribute space. This scheme is shown in Fig. 3.3.

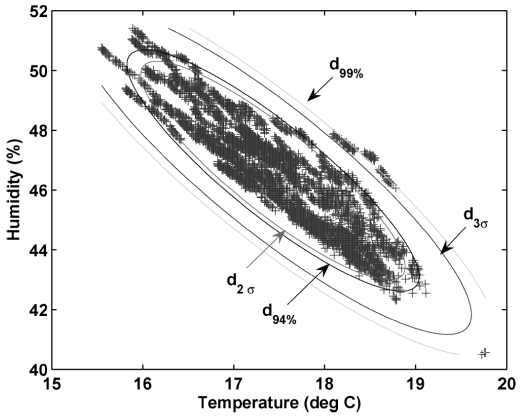


Figure 3.3: Statistical distribution in the attribute space made up by temperature and humidity. Points with Mahalanobis distance greater than d are treated as outliers, from [Raj+10].

3.1.2 Trust-Management Based Techniques

Trust-management considers the trustworthiness between two classes of entities: a trustor and a trustee. The trustor assigns each trustee a trustworthiness value, based on how much the trustee's behaviour matches an expectation. Trustworthiness values are usually in the range $[0, 1]$, decreasing when the trustee exhibits deviations from the expected behaviour and increasing when the trustee's behaviour matches the expectation.

Trust-management can be usefully applied in WSNs to reduce the influence of the compromised sensor nodes that inject malicious data. Indeed, if the expected behaviour accurately characterises genuine nodes, compromised nodes would be assigned a low trustworthiness when deviating from it. Since trust values are a continuous metric defined inside an interval, there is no direct classification of compromised and genuine nodes. Instead, the trust values are used to apply a penalisation proportional to the confidence that the sensor is compromised. Note

that the influence of the compromised nodes become negligible only when the confidence is sufficiently high. Filtering all the sensors with a trustworthiness under a given threshold [SLD12], could help mitigate this drawback, but requires a method to set the appropriate threshold.

Event-based techniques

Trust-management in sensed data was originally introduced as a complement to network-level trust, i.e. the reliability to correctly perform network-level tasks [GBS08; Ray+08; MCA08] such as communicating routes, participating to the route discovery process, routing incoming packets etc. The behaviour with respect to each of these tasks can be of two kinds: cooperative and uncooperative.

The first examples of trust management specifically designed for sensed data use a similar binary evaluation to build the trustworthiness, defined with respect to an event detection process. Initially, a *decision logic* establishes the presence of the event by combining the sensed data and the trust values. Then, the sensed data is compared to the final decision to measure the sensor's cooperativeness and update the trust values. This criterion is based on the assumption that nearby sensors are expected to agree about the event presence, which is a form of spatial correlation (see Sect. 2.3.3).

One of the first techniques to adopt this approach is described in [Ata+08]. As shown in Fig. 3.4, the reading of a generic sensor i, $S_i(t)$, which can take the values 0 and 1 (absence/presence of an event), are relayed to a *forwarding node*. This node computes $\sum_{n=1}^{N} W_n \hat{x}_i(\mathbf{u_n})$, where $W_{n\,:\,n\in 1\ldots N}$ denote the trust weights.

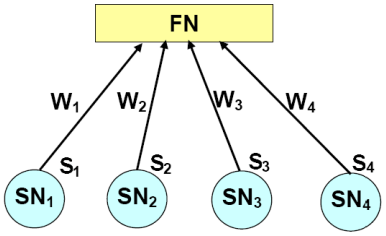


Figure 3.4: Trust-weighted aggregation for event detection. FN is a forwarding node, which collects reports from the sensor nodes SN, from [Ata+08].

The result is used to decide about the ground truth E. Afterwards, weights are updated with the following rule:

$$W_n = \begin{cases} W_n - \theta r_n, & \text{if } \hat{x}_i(\mathbf{u_n}) \neq E \\ W_n, & \text{otherwise} \end{cases} \tag{3.5}$$

where r_n is the ratio of sensors giving different output over the total number of sensors and θ is a penalty weight that determines a trade-off between the detection time and accuracy. In summary, the trustworthiness values, which coincide with the weights, are calculated based on the measurements consistency with the aggregated value. The latter is considered more reliable than the single readings, since sensors which exhibited inconsistent (e.g. malicious) readings in the past contribute less to the aggregation process. Finally, malicious nodes are detected by comparing the weights to a threshold, which the authors heuristically set to 0.4. Note that the algorithm is vulnerable to the *on-off attack*: a node that performs well for a time period, acquires high trustworthiness, then suddenly starts malfunctioning [Sun+07].

To counteract the on-off attack Oh, Hong, and Choi [OHC12] and Lim and Choi [LC13] propose to penalise $\hat{x}_i(\mathbf{u_n}) \neq E$ by a quantity α and reward $\hat{x}_i(\mathbf{u_n}) = E$ by a quantity β with $\beta < \alpha$. As $\frac{\alpha}{\beta}$ grows bigger, faulty and malicious nodes are filtered out faster. However, sensors with transient faults are also filtered out, even though they may report correct measurements later on. To avoid this, the ratio $\frac{\alpha}{\beta}$ needs to also consider the probability of transient faults and their duration distribution. Therefore, this operation just reduces the frequency with which an attacker can switch between "good" and "bad" behaviour in an on-off attack.

When the sum of all trust weights is equal to 1, the weighted sum of sensors reading corresponds to a weighted mean. As described in the previous section, the mean has the drawback of being directly proportional to extreme readings. So in trust-based aggregation as well, the median could be used as a more robust aggregation operator. A trust weighted median has been applied by Wang, Ding, and Bi [WDB10] in the context of acoustic target localisation, where the median allows them to filter out faulty measurements. The advantages of using the weighted median

increase when an element with high weight has an extreme value. Indeed, while the weighted mean would be biased towards that value, the weighted median would still filter it out, if the other values are not extreme and the sum of their weights is bigger than the weight of the extreme value. This property reduces the efficacy of an on-off attack.

Another aspect to take into account is the uncertainty in the event's presence. Raya et al. [Ray+08] deal with this aspect by using a decision logic based on Dempster-Shafer Theory (DST), which expresses the belief about the event presence as a combination of individual beliefs from sensor nodes. DST combines the sensors' information supporting the event with the information non-refuting the event (the uncertainty margin which may comply with the event presence).

Anomaly-based techniques

Rather than analysing the compliance with the output of on an event decision logic, other trust-management techniques look for anomalous behaviours with techniques similar to anomaly detection ones.

In fact, the output of anomaly detection itself can be used to define a cooperative/uncooperative behaviour [GBS08], but a more flexible approach, that does not restrict the observations to a binary value, is to update trust values based on an anomaly score. An example is given by Bankovic et al. [Ban+10], using self-organizing maps (SOM). SOM are a clustering and data representation technique, that maps the data space to a discrete 2D neuron lattice. Bankovic et al. [Ban+10] build two SOM lattices: one in the temporal domain and another in the spatial domain. The trust values are assigned based on two anomaly scores: the distance between the measurement and the SOM neuron, and the distance between the neuron to which the measurement has been assigned and other SOM neurons. The main disadvantage of this algorithm is its computational time. For better accuracy, SOM require many neurons, but the computational time increases noticeably [McH03].

Another example is given by Zhang, Das, and Liu [ZDL06], who use a statistical-test approach

(see Sect. 3.1.1) to assign reputation values to the sensors. The measurements gathered in time are assumed to approximately follow a normal distribution. The normal and actual measurements' distributions are compared with Kullback-Leibler divergence D_n, which evaluates the information lost when a probability distribution is used in lieu of another. The divergence is then used to update the trust values, with the following expression:

$$W_n = \frac{1}{1 + \sqrt{D_n}} \tag{3.6}$$

Using Second Hand Information

In the trust-management schemes previously analysed, each sensor's trust values are computed and updated by the device with the trustor role, typically a forwarding node. However, when the trustor is not in the transmission range of its trustee i, it may rely on information from its neighbours N_i to calculate its trustworthiness. Bao et al. [Bao+12] deal with this problem by introducing two different trust update criteria:

$$T_{ij}(t) = \begin{cases} (1-\alpha)T_{ij}(t-\delta t) + \alpha T_{ij}(t) & \text{if } j \in N_i \\ avg_{k \in N_i}\{(1-\gamma)T_{kj}(t-\delta t) + \gamma T_{kj}(t)\} & \text{otherwise} \end{cases} \tag{3.7}$$

The calculations of the second case represent node j's *recommendation*, i.e. the trustworthiness extracted from relayed information. Eventually, recommendations depend on trustworthiness from the viewpoint of direct neighbours. However, such trustworthiness can be manipulated by malicious nodes to bad-mouth or good-mouth other nodes. Bao et al. [Bao+12] mitigate this problem by controlling the impact of recommendations through parameter γ, set to $\frac{\beta T_{ik}(t)}{1+\beta T_{ik}(t)}$. Thus, if a sensor has little trust compared to the parameter β, the contribution of its recommendation will be small. However, sensors conducting an on-off attack can give false recommendations for a short while and then behave correctly again without being detected.

Even when direct information is available, recommendations can be used as second hand information and combined with direct information to obtain a *reputation*. Second hand information

speeds up the convergence of trust values but adds network traffic overhead and introduces new parameters, such as the weighting criterion for recommendations and the recommendation exchange frequency [HLT05]. Ganeriwal, Balzano, and Srivastava [GBS08] follow this approach and treat reputation as a probability distribution, updated as a combination of direct and indirect reputation. Direct reputation is updated based on a watchdog module, while indirect reputation is updated with recommendations, i.e. reputation from other nodes. The framework's scheme is shown in Fig.3.5. Note that such a definition of reputation introduces a loop: indirect reputations come from reputations given by other sensors, which in turn depend on indirect reputations. To avoid the information loop, recommendations need to be taken only from direct observers.

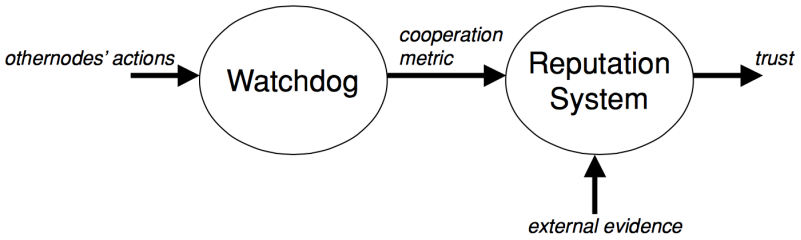


Figure 3.5: Combination of direct information and recommendations, from [GBS08].

Modelling reputation as a single value does not consider the uncertainty that a sensor has in trusting another sensor. This information is particularly useful with recommendations, as recommendations from sensors with high uncertainty should contribute less. To consider uncertainty, reputation can be modelled with a probability distribution whose choice is dictated mainly by the trust evaluation and update criteria. For example, Ganeriwal, Balzano, and Srivastava [GBS08] use the *beta distribution* since it is the posterior distribution when the binary interactions between nodes are modelled with a binomial distribution. Momani, Challa, and Alhmouz [MCA08] apply a normal distribution to model the differences between the measurements of two sensors (spatial homogeneity is assumed, see Sect. 2.3.3).

3.2 Diagnosis in Related Work

In the previous section we have analysed how state-of-the-art measurements inspection techniques cater for the detection of malicious data injections. We discuss below the state-of-the-art in the diagnosis task.

Distinguishing Events from Faults

From the analysis of the work available in literature, we have observed that most of the diagnosis efforts aim at diagnosing events as opposed to faults. The general assumption used to distinguish between them is that faults are likely to be stochastically unrelated, while event measurements are likely to be spatially correlated [LDH06; SNQ12]. Note that this assumption excludes common-mode failures from the analyses. Based on this assumption, after detecting deviations from expected data with temporal [BHL07; SNQ12] or attribute [SNQ12] correlations, it is possible to diagnose whether the deviation was caused by a fault or an event, by exploiting spatial correlation. When there is a consensus among a set of sensors about the presence of an event, discording sensors are considered faulty [LDH06; SNQ12; BHL07]. Similarly, some sensed attributes (e.g. human vital signs, such as glucose level, blood pressure, etc.) can be assumed heavily correlated in the absence of faults, which instead disrupt attribute correlations. Then, if we further assume that events would cause a minimum number of outlying attributes, faults can be identified when the minimum is not reached [SLM13].

Fewer advances have been made towards diagnosing malicious interference as opposed to faults and events – we summarise them in the following sections.

Distinguishing Malicious Interference from Events

In the literature, malicious interference is distinguished from events through an agreement-based strategy [LCC07; Ata+08; WDB10; OHC12; LC13; Sun+13], i.e. the sensor's information is first used to decide about the presence of an event and then sensors which did not support the

final decision are identified as malicious. This approach is based on the assumption that sensors are sufficiently spatially correlated to correctly detect events. However, multiple compromised nodes can also collude in the attack to keep the spatial correlations consistent between themselves. This complicates discriminating between genuine events and malicious data injections, and allows an attacker to fabricate false events or to mask genuine ones. This aspect is discussed in more detail in Sect 3.3.

Distinguishing Malicious Interference from Faults

Criteria to distinguish malicious data injections from faults are not broadly investigated. Two main approaches can be identified: delegating the diagnosis to intrusion-detection techniques and leveraging fault statistics.

Intrusion Detection One of the main challenges in detecting attacks with anomaly-based techniques, is that such techniques abstract the means through which an attack is conducted. This choice comes from the objective to detect new attacks with unknown patterns, as opposed to intrusion detection techniques which are based on recognising known attack signatures. The framework proposed by Ngai, Liu, and Lyu [NLL06] is a trade-off between an anomaly-detection technique and an intrusion detection system, since the detection is carried out through anomaly detection achieving high detection rate, while the diagnosis is carried out with intrusion detection. Clearly this approach provides diagnosis only for known attacks and cannot distinguish between an unknown attack and a fault.

Fault Statistics The statistical characterisation of faults can also be used to distinguish them from malicious interference. Oh, Hong, and Choi [OHC12] and Lim and Choi [LC13] use the expected frequency of transient faults to avoid excluding from the system sensors subject to transient faults. Indeed, their trust management algorithm allows such sensors to recover trustworthiness, by allowing temporary misbehaviour. Only sensors misbehaving with higher frequency, including malicious sensors and sensors with permanent faults will then be excluded.

3.3 Characterisation in Related Work

In related work, characterisation is often addressed simultaneously with detection, since the information characterising the attack can be precious to improve the detection. In particular, information about spatial correlations can help identifying a malicious sensor with measurements that are not consistent with those in the neighbourhood.

However, when multiple sensors have been compromised and *collude* in the attack, they can act in concert to change the measurements whilst evading, if possible, any anomaly detection applied. Therefore, identifying which sensors are more likely to be genuine and which sensors are more likely to be compromised becomes an integral part of detecting the attack itself.

In collusion attacks, the compromised sensors follow a joint strategy that reduces the advantages of spatial correlation, since the compromised nodes co-operate to form credible spatially correlated data [TH06]. In the presence of collusion, diagnosis is also significantly more complex. Tanachaiwiwat and Helmy [TH06] point out that when a genuine outlier (for example related to an event) occurs, extreme readings from the colluding nodes could be hidden. The problem becomes increasingly difficult as the percentage of (colluding) compromised sensors increases. Ultimately, when the number of colluding sensors increases to the point of exceeding genuine sensors, the attack may still be detectable, but it may be impossible to distinguish which nodes are genuine and which nodes are compromised. Tanachaiwiwat and Helmy [TH06] evaluate their anomaly detection algorithm against colluding nodes and find that performance noticeably decreases when more than 30% nodes are colluding. A similar result is reported by Chatzigiannakis and Papavassiliou [CP07].

Bertino, Ignatovic, and Jha [BIJ14] describe a new attack scenario applicable when the trustworthiness is calculated through an *iterative filtering* algorithm. While in generic (non-iterative) trust-evaluation techniques, trust weights are updated based on data from the current time instant and the weights calculated at the previous time instant, in iterative filtering the weights are iteratively updated with data of the same time instant until a convergence criterion is satisfied. In this context the authors introduce a new attack scenario where all colluding nodes

but one, produce noticeable deviations in their readings. The remaining compromised node reports, instead, values close to the aggregated value of all the readings (including malicious ones). Eventually, this node acquires a high trust value, while all the others acquire low trust values. The aggregated value, in turn, quickly converges to a value far from that of the genuine nodes. The authors show that this attack is successful when the sensors are assigned equal initial trustworthiness. They therefore propose to calculate the initial trustworthiness as a function that decreases as the error variance increases. The error is defined as the distance from an estimated physical attribute value $\varphi(t)$, and is the same for all the sensors.

In [Rez+13] the same authors proposed another technique that detects collusion rather than counteracting it. This technique is based on the assumption that deviations from the aggregated values are normally distributed for genuine nodes. This assumption comes from the observation that the deviations of non-compromised nodes, even if large, come from a large number of independent factors, and thus must roughly have a Gaussian distribution. For colluding nodes instead, they assume that this condition does not hold. Then, by running the Kolmogorov-Smirnov test to check compliance to the normal distribution, they discriminate colluding nodes from genuine nodes.

In summary, while many studies propose propose new anomaly detection algorithms to cater for a broad range of scenarios, comparatively fewer address specifically malicious data injections in a way that can cater for more sophisticated attacks involving collusion between sensors. Such scenarios need to be explored further and constitute one of the main problems covered in this thesis.

3.4 Discussion

In the previous sections we have seen how different techniques can be applied to detect malicious data injections, how they leverage measurements' correlations, and the assumptions on which such correlations are based. We have examined the different detection techniques and how they find deviations from the expected behaviour. We have highlighted the importance of

distinguishing between different sources of deviations and presented the main directions of work towards this objective so far.

We now combine these analyses by building direct comparison tables, which summarise their main characteristics. A summary of the results reported by each of the techniques mentioned is provided in the following section.

3.4.1 Comparison of Approaches

We divide our comparison of the approaches analysed so far into Tables 3.1 and 3.2, containing the anomaly detection and trust management techniques respectively. The content of the columns from left to right is: technique name and reference; correlation used to define expected data; assumptions about the spatial model if any; detection criterion used; possible sources of anomalies (as mentioned in their paper) and for which of them diagnosis criteria are given, e.g. {Event},{Malicious or Faulty} means that the authors give a criterion to discern between anomalies arising from events and from malicious or faulty sensors.

We observe that spatial correlation is most often exploited, and frequently under the assumption of a homogeneous space. The situation is particularly evident for papers considering the presence of malicious data injections and is probably a consequence of the fact that only a minor subset of sensors is assumed to be compromised. Therefore, in the spatial domain there is always a significant set of genuine measurements that can be exploited to detect the malicious ones.

Assuming spatial homogeneity makes the calculations significantly simpler, since the sensors are considered to measure the same value. However, it also significantly restricts the applicability of the techniques in real cases. When the physical phenomenon is observed with low precision, e.g. overall temperature across a large open space area, this assumption is still valid if the spatial variations are small enough to treat them as noise. However, this allows an attacker to introduce malicious data that are within the noise bounds yet still deviate significantly from the real values. While this assumption is generally appropriate in small areas, small areas also

typically include fewer sensors which have higher risk of an attacker compromising them all.

When multiple types of correlation are considered, temporal correlations are generally exploited along with spatial ones. Use of attribute correlations is rather infrequent, probably due to the fact that understanding them requires knowledge about their physical significance and this is application specific. The tables highlight even more the lack of diagnosis and characterisation (see Sect. 6.7). Few papers consider specifically malicious injections with collusion and even fewer papers deal with the problem of distinguishing them from other causes of deviations. While distinguishing events from faults is the kind of diagnosis more frequently considered, distinguishing attacks from faults is undoubtedly more challenging and still rather rare.

Table 3.1: Anomaly detection techniques

Work	Correlation exploited	Spatial model	Detection method	Classes considered	Inter-class discrimination
EKF, CUSUM GLR [Sun+13]	Temporal	None	Change in the distribution of error from estimate	Event, Malicious, Faulty	{Event}, {Malicious or faulty}
MGDD [Sub+06]	Temporal	None	Measurement probability	Event, Fault	None
[NLL06]	Spatial	Homogeneous	Difference with neighbours	Suspicious of Sinkhole attack	None
[Wu+07]	Spatial	Homogeneous	Difference with neighbours	Event	None
FIND [GZH09]	Spatial	Monotonic WRT event source	Spatial monotonicity disruptions	Fault	None
[SLM13]	Attribute-temporal	None	Energy of fluctuations	Event, Fault	{Event} {Faulty}
STIOD [Zha+12]	Spatio-temporal	Variogram	Difference with estimate	Event, Error	{Event} {Error}

MAP+HBST [NP12]	Spatio-temporal	Linear spatial trend	Difference with estimate	Fault	None
[LCC07]	Spatial	Homogeneous	Difference with neighbours	Malicious, Event	{Malicious}, {Event}
ART [TH06]	Spatial	Homogeneous	Difference with neighbours	Compromised, Uncalibrated Sybil	{Compromised or Faulty}, {Uncalibrated}, {Sybil}
[Raj+09]	Spatio-temporal	Homogeneous	Values outside a quarter-sphere	None	None
STA-QS-SVM [SNQ12]	Spatio-temporal and Spatio-attribute	Homogeneous	Values outside a quarter-sphere	None	None
[CP07]	Spatial	High Pearson correlation	Changes in correlation	Fault, Malicious	{Point failure or malicious node}, {Group failure or Collusion}
[BHL07]	Spatio-temporal	Homogeneous	Distribution of temporal and spatial differences	Event, Fault	{Event}, {Point failure}
[Han+75]	Spatial	Linear combination of state variables	Difference with estimate	Fault	None
Robust IF [Rez+13]	Spatial	Homogeneous	Distribution of distance from estimation	Fault, Malicious	None

3.4.2 Comparing Reported Evaluation Results

In the previous sections, we have considered techniques that could be applied to the problem of detecting, diagnosing and characterising malicious data injections. For those techniques that

Table 3.2: Trust based detection techniques

Work	Correlation exploited	Spatial model	Detection method	Classes considered	Inter-class discrimination
[ZDL06]	Spatio-temporal	Homogeneous	Distance from mean of top-trust sensors	Malicious	None
WTE [Ata+08]	Spatial	Homogeneous	Trust under a threshold	Malicious	None
[MCA08]	Spatial	Homogeneous	Trust under a threshold	Faulty, Malicious	None
[WDB10]	Spatial	Homogeneous	Difference with aggregated value	Faulty, Event	{Faulty}, {Event}
[Ban+10]	Spatio-Temporal	Heterogeneous	Difference with learnt pattern	Malicious	None
Trust-based IDS [Bao+12]	Spatial	Homogeneous	Trust under a threshold	Malicious, Event	{Malicious}, {Event}
DWE [OHC12]	Spatial	Homogeneous	Trust under a threshold	Malicious, Permanent Fault, Transient Fault, Event	{Malicious or Permanent Fault}, {Event}
Dual threshold [LC13]	Spatial	Homogeneous	Trust under a threshold	Malicious, Permanent Fault, Transient Fault, Event	{Malicious or Permanent Fault}, {Event}

focus specifically on malicious data injections we now present the experimental evaluation set-up used by the authors and compare the reported results. None of these techniques has been tested on real attack scenarios. This is not surprising as finding real attack data in existing WSN deployments is difficult. In fact, two approaches have been broadly adopted to evaluate the algorithms for detection of malicious data injections: *simulation* [Sun+13; LCC07; Rez+13; Ata+08; Ban+10; OHC12; Bao+12; LC13] and *injection of attacks* in real datasets [TH06; CP07].

Table 3.3 summarises all the results achieved, together with all the relevant simulation parameters. The last three columns express the False Positive Rate (FPR) when the True Positive Rate (TPR) is respectively 0.90, 0.95 and 0.99. TPR is, by definition, the number of attack instances that are correctly detected, divided by the total number of attack instances. We want

Table 3.3: Detection performances, independent attacks

Work	Dataset size	Dataset malicious percentage	Input size for each algorithm execution	FPR for TPR=0.90	FPR for TPR=0.95	FPR for TPR=0.99
EKF [Sun+13]	10000 samples	50% samples, same node	6	0.22	0.42	0.7
[LCC07]	4096 nodes	10-25% nodes	10	0.01	0.01	0.07
ART [TH06]	100 nodes	30-50% samples, random selection of malicious nodes	100	0.25	0.22	0.21
[CP07]	40 nodes	10% nodes	40	0.67	0.69	0.7
[CP07]	40 nodes	40% nodes	40	0.48	0.5	0.6
WTE [Ata+08]	100 nodes * 200 samples	0-25% nodes	100	0.03	0.41	0.78
WTE [Ata+08]	400 nodes * 200 samples	0-25% nodes	400	0.10	0.44	0.78
[Ban+10]	2000 nodes * 2500 samples (1000 are used for training)	5% nodes	2000	0.5	0.5	0.5
Trust-based IDS [Bao+12]	900 nodes	N/A	N/A	0.001	0.05	N/A
DWE [OHC12]	200 samples	20% nodes	20	0.01	0.01	0.02
Dual threshold [LC13]	100 samples	10% nodes	12	N/A	N/A	0.001
Dual threshold [LC13]	100 samples	20% nodes	12	0.18	0.14	0.10

Table 3.4: Detection performances, colluding attacks

Work	Dataset size	Colluding percentage	Input size for each algorithm execution	FPR for TPR=0.90	FPR for TPR=0.95	FPR for TPR=0.99
ART [TH06]	100 nodes	30-50% samples	100	0.25	0.22	0.21
[CP07]	40 nodes	10% nodes	40	0.67	0.69	0.7
[CP07]	40 nodes	40% nodes	40	0.76	0.78	0.8
Robust IF [Rez+13]	20 nodes per 400 samples	40% nodes	20	N/A	0.021	0.021

this value to be high as undetected attacks may cause severe damage, thus we report the results for TPR greater than 0.90. FPR is, by definition, the number of times normal data instances are misclassified as attacks, divided by the total number of normal data instances. This value needs to be close to 0, as a high frequency of false alarms makes the method impractical. The actual value that can be tolerated depends on the application as well as on the time between two executions of the measurements inspection algorithm. Nevertheless, values above 0.10 are generally impractical in most cases.

The relationship between TPR and FPR is known as the Receiver Operating Characteristic (ROC). Column 2 reports information about the size of the dataset used in the experiments. Column 3 reports the percentage of either malicious nodes or malicious measurements. Column 4 reports the input size for the algorithm; for example in an experiment with 100 nodes, where the nodes are clustered in groups of 10 and the algorithm is run on clusters, the algorithm input size is 10.

Generally, in each paper, the tests are conducted in scenarios with different assumptions. For instance, Liu, Cheng, and Chen [LCC07] generate data with a normal distribution for normal sensors and another normal distribution for malicious sensors. The results are excellent, but depend a lot on the difference between the two distributions. Another important assumption, which has noticeable impact on the results, is the spatial model. As pointed out in Sect. 2.3.3, most papers assume that the sensors' readings are homogeneous in the space; in other words

the measurements are expected to be equal to each other, apart from noise and errors. The consequence of this assumption is that, by increasing the number of sensors, the information redundancy also increases and the number of sensors taken into account is decisive. Recall from Sect. 2.3.3 that the sensing space can be approximately homogeneous only if we consider a small portion of space where there are no obstacles. In works like [CP07] and [Ban+10], where this assumption is not present, the FPR is higher, but the algorithm has wider applicability. Tanachaiwiwat and Helmy [TH06] rely on the spatial homogeneity assumption, and apply their technique to a large neighbourhood (100 nodes). The FPR is better but still not negligible (more than 20%). Atakli et al. [Ata+08] also rely on this assumption and apply their algorithm on very large neighbourhoods. With 100 nodes the FPR is 0.03 for TPR=0.90, but it increases by an order of magnitude for TPR=0.95 and TPR=0.99. In contrast, [OHC12; Bao+12; LC13], are successful in keeping the FPR low even for high TPR. Note that with a larger number of nodes the FPR of the technique described in [Ata+08] increases. This result contrasts with the consideration that we made about the spatial homogeneity assumption. The reason behind that, lies probably in the inaccuracy of the empirical ROC curve calculation. Another possible cause is that the algorithm is sensitive to the absolute number of compromised nodes rather than to its ratio to total nodes. For example 80 out of 400 compromised nodes may be harder to detect than 20 out of 100, event though the percentage of malicious nodes is 20% in both cases.

In Table 3.4, we report the results for the cases considering collusion. The results reported in [CP07] show non negligible FPR values (above 60%). The results reported in [TH06] have a better FPR (around 20%). Rezvani et al. [Rez+13] instead, achieve FPR as low as 5%. Nevertheless, recall that this technique is applicable only when the spatial homogeneity assumption among the 20 sensors is reasonable. In scenarios where the sensors readings cannot be assumed to share the same physical attribute function, the results may degrade substantially. This is the case for physical attributes like vibration, light, wind etc., where the correlation of the attribute measured at different locations rapidly decreases with the event propagation.

Table 3.5: Techniques overhead

Class	Work	Computational Overhead	Communication Overhead
Anomaly Detection	ART [TH06]	$O(W * N_n)$	$O(1)$
	[LCC07]	$O(N_n^2)$	$O(N_n)$
	[CP07]	$O(WN_n^2 + N_n^3)$	0
	EKF [Sun+13]	$O(1)$	$O(N_n)$
	Robust IF [Rez+13]	$O(WN^2)$	0
Trust management	WTE [Ata+08]	$O(N_n)$	0
	[Ban+10]	$O(N_n^2) + O(W^2)$	0
	Trust-based IDS [Bao+12]	$O(N_n)$	$O(N_n)$
	DWE [OHC12]	$O(N_n)$	0
	Dual threshold [LC13]	$O(N_n)$	0

3.4.3 Comparing Techniques Overhead

The applicability of a technique to a real WSN does not only depend on the relationship between true positive rate and false positive rate, but also on the overhead introduced. We analyse computational and communication overhead for the techniques discussed in the previous section, and summarise their asymptotic complexity in table 3.5. As usual, N is the number of sensors, while N_n is the average number of neighbours and W is the temporal memory, i.e. the number of past samples used.

From table 3.5, we note that anomaly detection techniques generally introduce more computational overhead than trust management techniques. The reason behind this result is that trust management iteratively refines its confidence about a sensor's trustworthiness, whereas anomaly detection builds such confidence from scratch at each iteration. On the other hand, this is also the main reason why trust-management algorithms are vulnerable to on-off attacks (see Sect. 3.1.2).

Another noticeable result is that communication overhead is always kept lower than computational overhead – this result is to be expected since network communication is more expensive in terms of energy and leads to faster battery depletion. In anomaly detection techniques, the

communication overhead comes from the execution of consensus-like protocols which decide about the maliciousness of nodes after anomalies are detected. Trust management techniques instead, delegate such decisions to the nodes that are higher in a WSN hierarchy (e.g. the forwarding nodes, cluster heads, base station). Thus communication overhead is introduced in trust management techniques only when recommendations are enabled (such as in [Bao+12]).

3.5 Conclusions

Malicious data injections are a considerable threat for WSNs. We reviewed state-of-the-art techniques that can detect malicious data injections by defining an expected behaviour and then detecting deviations from it. We classified these approaches into two main families: *anomaly detection* and *trust management*. They differ in the assessment of an anomalous condition, but both rely on the definition of an expected behaviour. We analysed and compared the techniques by their definition of expected behaviour and noted that expectations can come from correlations: *a*) *in time:* different time, same sensor, same attribute; *b*) *in space:* same time, different sensors, same attribute; *c*) *across different physical attributes:* same time, same sensor, different attributes; or *d*) their combination.

While many techniques can be applied, comparatively few studies target explicitly malicious data injections, especially when collusion between compromised sensors is considered. Most techniques aim to detect erroneous measurements, either to improve the quality of the measuring process (e.g. [Sub+06; BHL07]), or to reduce the power associated with the transmission of the measurements (e.g. [WDB10; SLM13].

Work aimed at detecting malicious data injections, generally uses spatial correlation in constructing the expectations (e.g. [ZDL06; LCC07; CP07]), in keeping with a general assumption that only a subset of sensors has been compromised. In this case, a non-void set of genuine measurements is always present in the spatial domain. In the rest of this thesis, we will always take this result in consideration, and tie the discrimination of genuine and malicious measurements to the spatial domain.

We discussed the different assumptions that characterise the spatial domain, and analysed how they impact the performance of the detection algorithms. More precisely, we observed a substantial decrease in performance when moving away from a homogeneous space model, where all sensors perceive similar measurements, to heterogeneous space models, where different measurements are expected at different locations. This result is visible, for example, in the difference between the results achieved in [TH06] and [Rez+13], who assume a homogeneous space, and those achieved by [CP07], who only assume some degree of correlation between the sensors. The results, in the latter case, show noticeable higher false positive rates. We conclude that **when the spatial domain is heterogeneous, the state-of-the-art techniques do not provide a practicable balance between TPR and FPR**. In fact, improving the results in such scenario is one of the first objectives pursued in this thesis.

We explored different approaches to the detection phase, where the deviation from the expected behaviour is assessed, and noted a clear preference in the literature for outlier-detection techniques (e.g. [NLL06; LCC07; Sun+13]). In this case, the expectation of a measurement is compliant with a generalisation of the measurements behaviour. This approach is independent of the context and is preferred to more context-specific techniques based on model checking (e.g. [Han+75]). This advantage is valid also for other anomaly detection techniques which differ from outlier-detection. This is the reason why we will develop anomaly detection techniques, but we have clarified that these do not need to look necessarily for outliers.

Finally, to complete the detection of malicious data injections, we identified two main aspects that need to be addressed: *diagnosis* and *characterisation*.

Diagnosis consists of identifying the cause of the detected anomaly which, besides malicious data injections, may lie in faults or events of interest. Both these phenomena may produce deviations from expected behaviour similar to malicious injections. In particular, events will cause deviations from expected behaviours if the event-related measurements are not properly modelled. Whilst partial diagnosis is investigated in, e.g., [TH06; BHL07; CP07; OHC12], an exhaustive diagnosis phase is still lacking. Fault-related anomalies may be handled separately from malicious data injections, as fault models are relatively well categorised and understood.

However, event-related anomalies cannot be considered separately (like in [LCC07]), since an attacker may inject malicious measurements that depict a fabricated event or conceal a real event. Therefore, in WSNs that monitor the occurrence of events, malicious injections and events should be addressed together, to produce a compromise-resistant detection and characterisation of events. Diagnosis has been insufficiently studied in the literature, so we cater for it in Chapter 6. We discern between malicious and faulty interference by learning the events patterns and incorporating them in the model. Instead, we distinguish faults by looking for typical fault characteristics, such as singular measurements that are clearly unrelated to the rest.

Similarly, further investigation of the attack characterisation, is needed, in particular to identify the compromised sensors. This has been done within homogeneous spatial models (e.g. [Rez+13]) or when the spatial model is heterogeneous but the malicious sensors do not collude (e.g. [CP07]). The problem of detecting colluding sensors when their inconsistencies are not evident because of natural variations in the spatial domain is challenging and has not been dealt with before.

In fact, neither has the detection part been addressed when malicious nodes collude in spaces that need to be modelled as heterogeneous. This is one goal of this thesis and is tackled by identifying and characterising the sources of heterogeneousness, referred to as spatial *events*, and contextualising the measurements in such events. Furthermore, we first focus our attention on the scenarios where there is one source of heterogeneousness, i.e. one event, in Chapter 5, and then generalise to multiple events in Chapter 6.

Before dwelling on the techniques that deal with the problem, we present in the next chapter the adversarial model, which can vary highly in the degree of control over the single measurements and in the way collusion is enabled. We also introduce the information that we aim to preserve, namely the ability to correctly detect any event of interest, and the challenges involved.

Chapter 4

Adversarial Model

When detection of malicious data injections is carried out, the attacker must strike a trade off between the potential damage it can introduce and the risk of being detected: higher potential damage is more likely to cause evident disruptions and, in turn, trigger detection.

The previous chapter has shown that collusion attacks are more difficult to detect, yet not all collusion attacks are equally effective because the attacker's capabilities need to be considered as well. Moreover, the concept of attack effectiveness is ambiguous unless the impact of alterations to the WSN measurements is well defined.

In this chapter, we first discuss the attacker's capabilities in terms of control on the WSN measurements. Then, we select a set of resources that enable the attacker to cause high damage without introducing obvious disruptions in correlation that are easy to identify. We define the integrity violations that need to be defended against, which indirectly define the integrity requirements. Finally, we convey how the attacker can leverage its capabilities, including complete control over sensor nodes that can collude between themselves to construct attacks on the measurements' integrity.

4.1 Attacker's Capabilities

Understanding the most important resources that can be exploited by an attacker, in terms of control on the measurements, is a necessary preliminary step to define the adversarial model.

An important resource which may affect the attacker's capability is the attack time. Indeed, because of the trade off between staying undetected and causing damage, the measurements need to be manipulated with a sophisticated strategy, which may require significant time to define (e.g. for executing a software that builds the attack). So, if the attack needs to be completed in a limited time, the strategy's search time may be constrained to satisfy that limit. However, relying on the attacker's time constraints on the defence side is a risky approach, since the latter can be reduced with improvements in the technology of the devices and algorithms. Instead, we will assume that **the attacker does not have time constraints in running the attack**. This assumption means that the attacker is free to stay inactive for an indefinite time whilst deciding about the malicious measurements to inject, but that does not mean that time constraints, e.g. imposed by the protocols, can be overlooked.

Another important resource for the attacker is information about the measurements, since the detection algorithm discerns genuine from malicious measurements by contextualising them in a measurements subset. To maximise the potential damage, we assume that **the attacker has full knowledge of the past measurements transmitted by all sensors**. Anomaly detection considers normal a measurement that is coherent with the most recent measurements and with those of neighbouring sensors, so this assumption implies that the attacker can access the same data processed by the anomaly detection algorithm. For the sake of simplicity, we assume that the measurements of all sensors are known, even if they are not neighbours of a compromised sensor; however, they will not be used for the attack.

The attacker's control on the measurements needs also to be defined quantitatively and qualitatively. We assume that **the attacker controls the measurements time series of a subset of sensors of its choice, and has full control over them**. In other words, the attacker can freely choose which sensors are compromised and, for each compromised sensor, can freely

decide the values of the measurements.

In summary, we assume attackers that:

1) Have infinite time for the attack.

2) Observe the measurements of all sensors.

3) Have full control of the measurements of C out of N sensors of their choice.

Assumption 3 implies that the spatial domain is not completely under the control of the attacker. We have already witnessed a preference for applying measurements inspection in the spatial domain in Chapter 3. Assuming that some sensors are not compromised is reasonable as long as the attacker's cost, which may be both a financial cost and the cost for risking detection, increases with the number of sensors that need to be compromised. This is the case if the attacker exploits physical access to tamper with sensor nodes, where the longer duration of the physical presence, the higher risk of being noticed.

If the measurements' integrity is compromised through the network layer, the attacker may compromise more measurements by controlling a sensor on a multi-hop path. However, here we consider attacks that affect only the end-points. The problem of network-layer integrity in forwarding nodes is better addressed with ad-hoc techniques, such as cryptographic hash functions [CPS06; Zhu+07; HST10] and intrusion detection [OM05; Sil+05; RZL06], which are beyond the scope of this thesis.

Other attack vectors, such as malware, may exploit some properties that are common to more sensors, thus gaining control over multiple sensors in a row. However, in practice sensors are not all identical (e.g., different generations) even when they measure the same values; some sensors may be physically hard to reach whilst others are easier; some sensors will be subject to maintenance e.g. because of fouling. A scenario where all the nodes are compromised is therefore extreme. If that is the case, the attacker can depict any arbitrary state of the physical phenomenon whilst staying undetected. This aspect is formally shown below. It is

proved that without assumption 3, the capabilities 1 and 2 prevent genuine and compromised sets of measurements from being distinguished.

4.1.1 Need for Genuine Subset of Measurements

Before the time when malicious measurements start to be injected, the WSN system is assumed to be in a genuine state, i.e. all the measurements are genuine and so are their correlations. Hence, we need to clarify why an attacker who starts to control all the measurements after that time is always able to move the system into another arbitrary state, entirely made of malicious measurements and correlations, without any chance of being detected.

We address this problem by defining which data is under the control of the attacker, which data is valid and which is not. So, based on such definitions, we aim to prove that an attacker with the capabilities listed in the previous Section is able to inject undetected any set of measurements that represent a plausible scenario (but different from the real one). Since control over all the sensors corresponds to a control on the spatial domain, we start by refining the notation introduced in Section 2.2.1 to group the measurements in such domain. In this way, the measurements are organised as a time series of *process snapshots*, which are defined below.

Definition 4.1. *A **process snapshot** is a set of spatial samples of the measured process at the same time instant, denoted with* $\mathbf{X}_{t_i} = \{x_{t_i}(\mathbf{u}) : \mathbf{u} \in \Omega\}$.

Measurements inspection may detect compromised process snapshots, i.e. corrupted by faults or malicious interference. So, we introduce the definition of valid process snapshot.

Definition 4.2. $\mathbf{X}_{t_i}$ *is a **valid** process snapshot if a time instant* t^* *exists such that* $\mathbf{X}_{t^*} \approx \mathbf{X}_{t_i}$, $\mathbf{X}_{t^*}$ *is observed at time* t^**, and neither faults nor malicious interference occur at* t^*.

Validity is defined on the snapshot's value, so $\mathbf{X}_{t_i}$ is valid even if it cannot be observed at time t_i, as long as a close snapshot $\mathbf{X}_{t^*}$ can be observed at time t^*. In general, a snapshot can be approximated with another one if the difference between them is within the noise range. Note that a snapshot that is not corrupted by faults and malicious interference is valid, but

a valid snapshot is not necessarily free of faults and malicious interference. By definition, as long as a snapshot that is corrupted by faults or malicious interference coincides with another one which can be observed under genuine circumstances, validity holds. This choice reflects the limitations of a measurements inspection algorithm: it can learn the structure of genuine data, but there is no way to detect malicious data when it is close to genuine data.

Since the measurements are contextualised in a time evolution of the physical process, validity needs to hold also for other snapshots at contiguous time instants, which constitute a *snapshot sequence.*

Definition 4.3. *A **snapshot sequence** is a set of time-ordered process snapshots* $[\mathbf{X}_{t_0}, \mathbf{X}_{t_1}, ..., \mathbf{X}_{t_l}]$*, denoted with* $\mathbf{X}_{t_0}^{t_l}$ *with* $t_l \geq t_0$*.*

Definition 4.4. *A snapshot sequence* $\mathbf{X}_{t_0}^{t_l}$ *is valid if a sequence* $\mathbf{X}_{t_m}^{t_{m+l}} = \mathbf{X}_{t_0}^{t_l}$ *exists such that all the* $\mathbf{X}_{t_i} : i \in [m, m+l])$ *are valid.*

With such definitions we are able to introduce the *returning property of physical processes.*

Definition 4.5. *A physical process is **returning** if, given any pair of valid snapshot sequences* $\mathbf{X}_{t_1}^{t_{1+l}}$ *and* $\mathbf{X}_{t_2}^{t_{2+n}}$*, a valid snapshot sequence* $\mathbf{X}_{t_{1+l}}^{t_2}$ *exists such that the concatenation* $\mathbf{X}_{t_1}^{t_{1+l}} \cup \mathbf{X}_{t_{1+l}}^{t_2} \cup \mathbf{X}_{t_2}^{t_{2+n}}$ *is a valid snapshot sequence.*

Dealing with returning physical processes implies the following theorem, which is just the property that we aimed at proving.

Theorem 4.1. *If the attacker controls all the spatial samples of a returning physical process since time* t_c*, any valid snapshot sequence can be injected at a time* $t_{a^*} \geq t_c$ *without compromising validity.*

Proof. The control of the attacker over all the spatial samples starting from time t_c translates into control over all the measurements, then the process snapshots $\mathbf{X}_{t_i} : t_i \geq t_c$ are under full control of the attacker. In particular, the attacker can inject any valid snapshot sequence $\mathbf{X}_{t_a}^{t_l}$ with $t_a \geq t_c$. Moreover, if a sequence $\mathbf{X}_{t_i}^{t_a}$ is valid, since the physical process is returning, there always exists a time $t_{a^*} \geq t_a$ such that $\mathbf{X}_{t_{a^*}}^{t_{a^*+l-a}} = \mathbf{X}_{t_a}^{t_l}$ and $\mathbf{X}_{t_i}^{t_a} \cup \mathbf{X}_{t_a}^{t_{a^*}} \cup \mathbf{X}_{t_{a^*}}^{t_{a^*+l-a}}$ is valid. □

When all the snapshot sequences are valid, there is no distinction between genuine and compromised snapshot sequences. Note that injecting measurements that create valid snapshot sequences may require highly sophisticated attackers, which are the main concern of this work.

In the following, we will assume that:

Hypothesis 1. *The WSN monitors returning phenomena.*

This assumption fits well all the applications where the physical phenomenon can manifest in similar ways multiple times. If this assumption does not hold, e.g. if the physical phenomenon compromises the environment broadly and for a long time (e.g. an entire forest is burnt by a wildfire), some valid snapshots become invalid. Yet Hypothesis 1 is not restrictive as long as the measurements inspection algorithm is not able to invalidate these snapshots or the attacker does not inject invalidated snapshots.

4.2 Event Spoofing and Masking

The goal of measurements inspection is to detect malicious measurements that could cause some damage to the system. Yet the definition of damage is ambiguous, as long as the WSN's mission is not well defined.

In event detection WSNs, the main mission is to identify conditions of interest of the monitored phenomenon that can be observed with noticeable changes in the measurements. For instance, an earthquake is an event for seismic sensors which causes high vibrations, or a flood is an event for water level monitoring sensors which would register the surface rise. The mission of such systems fails either if an event condition is not detected when there are events or if an event condition is detected when no such event occurs. Hence, the damage that an attacker can bring into such systems is *spoofing* event detection to detect false conditions of interest or *masking* real conditions of interest.

In general, the event detection algorithm is a function applied to W-long snapshot sequences to verify the presence of typical event properties, such as changes in time of the measurements'

average or variance, changes in the spatial distribution etc. So, we model the goal of the attacker as the will to transition from a non-event state to an event state or vice versa. We formalise this in reference to the notation given so far. To do that, we preliminary define the event detection function.

Definition 4.6. *A **event detection function** is a function* $\{\mathbb{R}^W \times \mathbb{R}^N\} \mapsto \{0, 1\}$ *whose input is a* W*-long snapshot sequence. The* 0 *value indicates no event while the* 1 *value indicates one or more events.*

Now the spoofing and masking can be unambiguously defined.

Definition 4.7. *A **spoofing attack** is successful if the snapshot sequence* $\mathbf{X}_{t_a}^{t_{a^*+W}}$*, is replaced with* $\breve{\mathbf{X}}_{t_a}^{t_{a^*+W}}$*, and both* $e(\breve{\mathbf{X}}_{t_{a^*}}^{t_{a^*+W}}) = 1$ *and* $e(\mathbf{X}_{t_{a^*}}^{t_{a^*+W}}) = 0$ *hold, where* e *is the event detection function.*

And dually:

Definition 4.8. *A **masking attack** is successful if the snapshot sequence* $\mathbf{X}_{t_a}^{t_{a^*+W}}$*, is replaced with* $\breve{\mathbf{X}}_{t_a}^{t_{a^*+W}}$*, and both* $e(\breve{\mathbf{X}}_{t_{a^*}}^{t_{a^*+W}}) = 0$ *and* $e(\mathbf{X}_{t_{a^*}}^{t_{a^*+W}}) = 1$ *hold, where* e *is the event detection function.*

We have defined spoofing and masking attacks successful independently from the detection carried out by the measurements inspection algorithm. This is to separate the concept of subverting the event detection algorithm from that of staying undetected. In that case we refer to the attack as an undetected successful (spoofing or masking) attack.

Note that distinguishing spoofing from masking attacks is generally a more complex task than detection. For example, in the scenario shown in Fig. 4.1, two main behaviours can be identified: some sensors indicate the presence of an event (light blue/green), while some others do not perceive it (dark blue/violet). In particular, the event measurements are located in the centre-left and the centre-right. Assuming that malicious data injections have been detected, we may be in a masking scenario, where the two events have been masked (originally it may have been even one single event that has been split by the masking sensors); we may also be

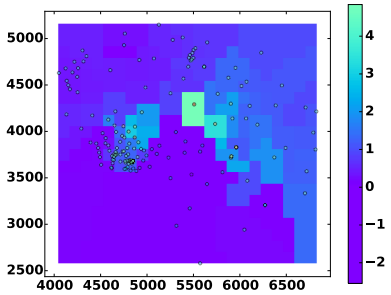


Figure 4.1: Seismic measurements with malicious data. The attack is difficult to characterise as spoofing or masking.

in an spoofing scenario where one, or even both events may have been spoofed. The problem becomes increasingly complex as the number of malicious sensors increases. When the attacker's capabilities are sufficiently high, the attacker may correctly reproduce genuine events without triggering detection or make malicious sensors be identified as genuine, and genuine sensors as malicious. In the next section we introduce our methodology designed to cope with such sophisticated malicious data injections.

4.3 Collusion

In Section 4.1.1 we concluded that an attacker with full control on the measurements from all sensors can eventually inject any valid sequence of measurements without compromising validity, and in Section 4.2 we have introduced the concept of event spoofing and masking. Since event-detection WSNs, by definition should detect events if and only if they are present, there are always at least two valid behaviours: one that detect events and the other that does not. In virtue of Theorem 4.1, we deduce that an attacker controlling the measurements from all sensors can eventually spoof or mask an event without compromising validity. This is summarised in the Corollary below.

Corollary 1. *If the attacker controls all the spatial samples of a returning physical process since time t_c, any event can be spoofed or masked at a time $t_{a^*} \geq t_c$ without compromising validity.*

In practice an attacker can achieve such degree of control only if the modifications to the process

snapshots are coordinated. Otherwise, since the underlying physical process is correlated, the attack is likely to generate invalid snapshots, or snapshot sequences. We refer to the principle of orchestrating the malicious measurements from different compromised sensors according to a unified strategy as *collusion*.

There are are different ways to introduce collusion, which depend on the particular attack vectors used. For instance, if the nodes' software is tampered with, it may contain a routine that identifies a unique snapshot sequence to inject, based on previous snapshot sequences that have been observed (in virtue of the capability 2 reported in Section 4.1).

If the measurements injections occur through environment manipulation, instead, the attacker may trigger its actions simultaneously to obtain a collusion effect. For instance, multiple smoke pot widgets may be simultaneously activated to obscures some light sensors.

Since there are potentially highly sophisticated ways to run collusion attacks, in this thesis we will not limit the attacker's collusion methods. Indeed, we will assume an attacker that is able to control all the compromised measurement in a coordinated way. Instead, we will focus on the point 3 of the adversarial model in Section 4.1, which limits the number of sensors that are in control of the attackers, to build a method that detects malicious measurements under partial compromise.

The collusion strategy, from the attacker perspective, needs to be optimised considering that there is only a partial control over the spatial domain. This constraint, if the measurements inspection algorithm is well designed, will establish a trade-off between the alteration of the event detection algorithm output and avoiding detection from the measurements inspection algorithm. If the measurements inspection algorithm is accurate enough in separating valid and invalid sequences, and if the density of genuine sensors is high enough, spoofing and masking may not be achieved whilst staying undetected because there is no valid snapshot sequence that introduces a spoofed event or hides the presence of genuine events. Nevertheless, an invalid snapshot sequence may exist such that distinguishing which measurements are malicious inside the snapshots is impossible, i.e. collusion may also be exploited to make the characterisation task fail.

4.4 Conclusions

In this chapter, we have introduced the adversarial model, by identifying the knowledge and control the attacker has on the malicious data, and how this affects the chance of success for measurements inspection. We have concluded that we can assume an attacker with full knowledge and control, but we need to impose the presence of a subset of sensors that produce genuine measurements.

We have formally defined the attacker's threats in a generic way that is pertinent to the main mission of an event-detection WSN, i.e. spoof events if there are none, and mask events when there are one or more. We have also defined a threat that affects the characterisation when a detection triggers, i.e. making genuine measurements be classified as malicious and malicious measurements be classified as genuine.

We have defined collusion as the capability for the attacker to coordinate the malicious measurements and we have anticipated that for the rest of this thesis we will assume the attacker capable of adopting any collusion-based strategy, aimed at disrupting event-detection whilst staying undetected. The high sophistication of the attacker, together with the powerful set of attack resources and capabilities that we are assuming, make the problem difficult to solve.

In the following, the measurements inspection algorithm will be designed considering that the attacker may exploit collusion to impair the algorithm's output. Protecting the event detection task in such circumstances is challenging since it reduces to the capability of accurately delimiting the domain of the valid measurements sets. The measurements domain is a high-dimensional one, spreading across time and space, and is partially unknown because it is not possible to know all the possible ways in which the physical process can evolve, and how that would affect the sensed measurements.

These challenges have not been tackled together before, as discussed in Chapter 3, as the attacks are either unsophisticated, e.g. random biases are added to the measurements, or the measurements distribution is over-simplified and, in fact, finds no application in event detection WSNs, where the distribution can change significantly with the manifestation of events. The

problem is more and more complex as such changes are considerable. Thus, rather than tackling the problem directly in its worst-case scenario, i.e. where the variability is unpredictable and irregular in space, we first solve the problem where one event can manifest at a time, and induces a predictable pattern in the measurements' spatial correlations. This assumption is also a reasonable one for some applications, such as volcanic eruptions and healthcare, and in fact offers higher performance compared with a general approach that is suitable for all kinds of measurements variability.

Chapter 5

Securing Single Events with Regular Patterns

In this chapter, we aim to give a first solution to the problem of malicious data injections under the assumption of highly capable attackers given in the previous chapter. In summary, such capabilities give the attacker full freedom to control C out of N sensors, from a certain point in time and for an unlimited time, plus the knowledge of all the measurements that were transmitted in the past. We also assume that the attacker is able to exploit sophisticated collusion-enabled strategies, i.e. the malicious measurements can be produced according to a unique plan that maximises the damage and minimises the chance of being detected.

Collusion attacks have been considered in a few works, [TH06; CP07; Rez+13]; however the context has been restricted to those in which the measurements distribution does not exhibit significant changes. This assumption prevents the application to WSNs where events manifest.

Event-detection applications include military surveillance [He+04], health [Ott+05], and environment (e.g., volcano) monitoring [WA+06a]. For such applications the security requirements are particularly critical and so is the integrity of the event detection task.

When events occur, the measurements distribution can abruptly change, hence common techniques such as outlier detection trigger because of the mismatch in the distributions before and

after the event. In contrast, we assume here that the measurement distribution can change in the presence of events, but only one event at a time manifests. This requires us to understand if the measurements are still genuine when they may be affected by events.

To detect malicious data injections in event detection WSNs, we aim to reconstruct the measurements that would be observed in genuine circumstances. Each non-compromised measurement is an unknown variable, so it needs to be characterised through observable properties, which in turn are not compromised. In particular, we will consider different measurements other than those being analysed. When no attacks occur, all measurements are correlated, in different degrees, since they are affected by the same set of physical phenomena that manifest in the WSN space. Compromised measurements, instead, may disrupt such correlations, especially when they incorrectly represent the physical phenomena, which is the case where detection is most important.

The presence of correlations indicates that the measurements contain redundant information. In some cases, redundancy is present by design. Since the main goal of a WSN is to sample a process to run analyses or react to it, the density of the samples should be high enough to gather sufficient information under all possible scenarios. For instance, an access restricted area may be equipped with a number of cameras which cover each path to its entrance at least once and that take pictures with a period less than the time needed to overpass the cameras.

However, since the attacker may seek to reproduce the scenarios where correlation is minimum, the degree of redundancy that is present by design may need to be increased to detect malicious measurements. This holds true especially when the attacker controls more measurements: if the subset of genuine sensors is too small with respect to the measurements correlation, the attack may be undetectable or, even if detectable, there may be insufficient information to determine which sensors are compromised and which are genuine.

The algorithm presented in this chapter characterises the relationships between the sensors' measurements arising from the *spatial correlations* present in the physical phenomenon. Even though correlation-based analyses may easily spot a single malicious measurement, the problem becomes more difficult in the presence of multiple malicious measurements, originating from

colluding sensors. Indeed, the attacker can judiciously select the sensors to compromise to maximise the correlation among compromised sensors and keep consistent with genuine sensors. To ensure the detection is resistant to collusion, novel ways of aggregating measurements are introduced that are aimed to discard malicious contributions under attack and minimise the false positives under genuine circumstances as well. Even though the approach is based on simple and solid techniques (e.g., linear regression, weighted median, Pearson correlation) it can detect sophisticated collusion attacks that have not been considered before. Furthermore, the low computational complexity of such techniques makes the overall approach suitable for extensive analysis of online data.

The algorithm is presented together with a novel more general methodology to apply the algorithm in different application settings. In particular, the algorithm parameters can be derived from tests and knowledge that are pertinent to the deployment and application of the WSN, such as the event detection criterion used. Indeed, many prior studies (e.g., [TH06; LCC07; Rez+13; Sun+13]), propose algorithms that are evaluated in a single deployment or application setting, or even on a single simulated dataset. Would the same algorithm work in a different setting? For many reasons, this is not likely. In contrast, this work is applicable across three different application domains: health-care monitoring, monitoring of volcanic activity, and home fire alarms, each with different challenges. The approach is tested against realistic attacks that undermine the core objective of each application and minimise the chance of detection.

5.1 Methodology

To cope with the presence of events, we first extend the assumption made in [GBS08; TH06; LCC07; Raj+09; Ray+08; Rez+13; Sun+13] where the value of the sensed phenomenon is required to be the same within a neighbourhood and measurements differ only because of noise. This assumption allows to easily estimate the ground truth and label the measurements as outlying through e.g., one-class quarter sphere support vector machines (SVM) [Raj+09]. However, this assumption is generally valid only for very small neighbourhoods, where collusion

attacks can be successful by compromising all the sensors. To resist collusion, it is necessary to broaden neighbourhoods and cope with measurements that are significantly different. But in the absence of a common ground truth, what should a sensor's collected measurement be compared against? Previous studies, such as [CP07; Raj+10; SGG10], have proposed to detect inconsistencies in the correlation within a neighbourhood by extracting a unique overall consistency metric, to which every neighbour contributes. This, however, allows colluding sensors to compensate for each other and reduce the overall inconsistency, whilst still disrupting the collected values [LNR11].

In our approach, each sensor exploits spatial correlation to produce an *estimate* for the measurements of other sensors. Hence, this method is applicable even if the measurements follow different – but still correlated – distributions. Then, we aggregate the estimates from different sensors with a collusion-resistant operator that produces a final reliable estimate, which is compared with the actual reported measurement. An approach similarly based on aggregation of individual sensors' information is majority voting [LCC07; ZYN08; Hin09; Sun+13], where each sensor votes for a neighbour's maliciousness and the votes are aggregated by majority. The vote is normally the output of one of the anomaly detection algorithms, which we have analysed in Section 3.1.1. Similarly, trust-management frameworks aggregate individual beliefs about a sensor's behaviour [GBS08; Ray+08; ZDL06; Bao+12]. As we have described in Section 3.1.2, a sensor's behaviour is mapped to a trust value by all its neighbours, and then the sensor's trustworthiness is obtained, e.g., by averaging the trust values [Bao+12]. However, the main drawback of both majority voting and trust management is that they introduce an additional variable - the vote, or trust value - about which an attacker can lie with or without lying about the measurements at the same time. Detecting such attacks incurs additional computation and communication costs. In contrast, we extend voting-based and trust-based frameworks by *aggregating measurements estimates rather than votes or trust values*. Such a choice does not introduce additional variables, since the estimates are directly calculated from the raw measurements.

The potential of estimation-based frameworks are analysed by studying the limitations that arise in the voting scenarios among the sensors in Fig.5.1. Consider at first nodes A, B, and C

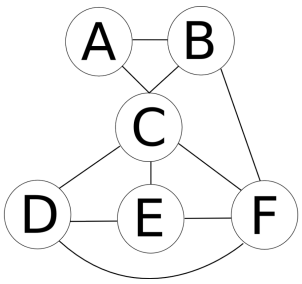


Figure 5.1: Example WSN topology. Nodes represent sensors and edges indicate a neighbour relationship.

to be compromised. In this case A is free to inject arbitrary malicious data if B and C collude to not report on it and act genuinely to avoid reports from D, E, F. If estimates were available, the measurements of B and C would look consistent with those of D, E, F (the reason why they don't report), but not consistent with those of A. Alternatively, consider nodes D, E, F to be compromised. Here nodes D and E can inject any kind of measurements, although C may report on them. Indeed, node F can avoid reporting on them and report on C instead. Then, with a simple majority voting approach [Hin09], node C would appear as the compromised node. Through the estimates instead, it is possible to detect that there is no valid reason for sensor F to endorse D and E and report on C, uncovering the collusion attempt. A majority voting approach will always fail when more than 50% of sensors are compromised. However, such upper bound considers the best case scenario, where all the genuine nodes report correct votes without uncertainty. The estimation-based framework instead, considers the *degree of uncertainty*, which becomes the only bounding factor. Indeed, the experiments in Sect. 5.6 show tolerance against up to 88% compromised nodes.

Our novel *estimation*-based framework, which iteratively extracts and aggregates measurements estimates, is at the core of the detection mechanism. For each new measurement we iteratively compute an estimate and then run a *similarity check*, which compares the estimate and the actual value as shown in Fig. 5.2. When the similarity check fails, we also run the *characterisation* step – an extensive analysis that identifies the likely compromised sensors. The models used to build the estimates are learnt during an *initialisation*, which serves to customise the techniques to the specific WSN deployment, and whose output is the *estimation models*.

During the *initialisation*, the network is assumed to be free of compromise. The estimation

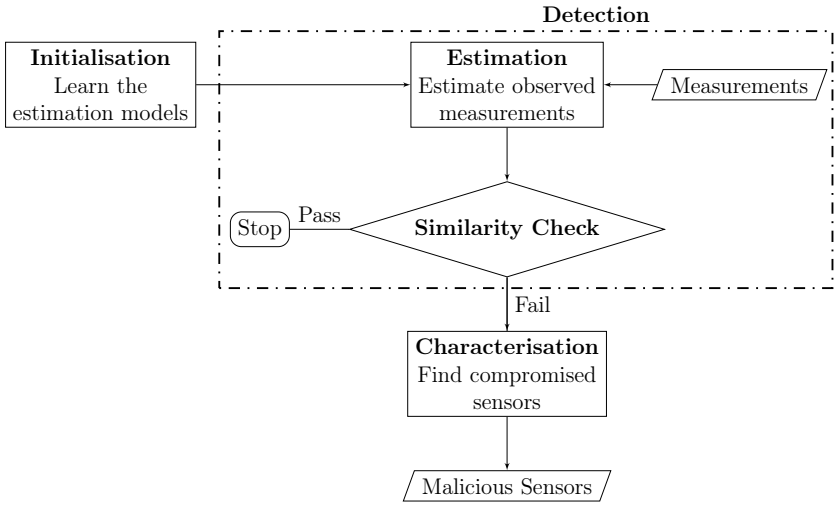
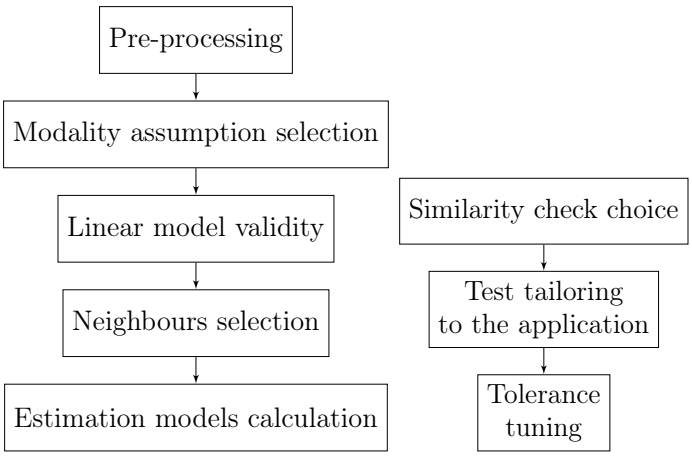


Figure 5.2: Measurements Inspection Workflow.

models are extracted with a *learning* step, which is shown in Fig. 5.3a: *a*) We first pre-process the data to eliminate faulty readings, alleviate noise and transform the data to extract the parameters of interest. *b*) We check whether the estimation models show significant changes in the presence of events, and if so we build a separate set of estimation models for each *modality*, i.e. distribution of measurements, of the physical phenomenon. *c*) We analyse the data to test if the correlation detected allows us to build a linear model to perform the estimation. Linear models describe accurately the correlation among different sensors, even when the physical phenomenon does not propagate with a linear law. Moreover it is lightweight as we need to store only two parameters for each model. *d*) We quantify the validity of a linear relationship for each pair of sensors and, based on that, we identify the neighbourhood of a sensor as the set of sensors with which there is a strong linear relationship, *e*) Finally we calculate the parameters that fit the estimation models.

The *similarity check* also needs to be customised to the sensor deployment during the initialisation, according to the steps shown in Fig. 5.3b. Since the final goal of the detection scheme is securing the event detection, the similarity metric should check the integrity of observable properties that characterise the event. Such properties can be derived from the *event detection criterion*, i.e. the algorithm that is run on the measurements to detect the presence of events. Two main properties characterise most event detection criteria: the *magnitude* and the *shape* of the measurements signal, based on which, respective tests are built. Therefore, during

the customisation step we: *a*) choose a similarity check based on such tests, *b*) tailor the test to the application according to the event detection criterion, *c*) tune a *dissimilarity tolerance* parameter to reach an objective false positive rate (FPR).



(a) Learning estimation models (b) Similarity check

Figure 5.3: Methodology Scheme for Application Tailoring

The tasks pertinent to each step can be assigned to different devices. However, in estimation-based frameworks, as in voting-based or trust-based frameworks, the final detection decision should be taken by an entity that is not compromised, typically the base station. The decision is taken based on the values (votes, trust, measurements) received from the sensors. In some cases, the deciding entity cannot perform such aggregation alone e.g., when measurements are aggregated while being collected in the network. In such cases the base station can delegate the detection tasks to the aggregating nodes but must also, additionally, verify the validity of the aggregation. Techniques to detect injections in aggregated reports have been proposed in [PSP06; GBS08; Roy+14] among others and are complementary to the work presented here, which focuses on the analysis of the raw sensor data.

5.2 Estimation

In Sect. 2.2.1 we introduced a unified notation for the sensing problem in WSNs, which is briefly summarised below to introduce also the estimation problem. We consider a set of N deployment locations Ω. The measurement *observed* by sensor j ($j \in 1, ..., N$) placed at the

locations $\mathbf{u}_j \in \Omega$, at time instant t_i is referred to as $x_{t_i}(\mathbf{u}_j)$, while the measurement *collected* to the base station (or data sink), is denoted by $\breve{x}_{t_i}(\mathbf{u}_j)$. We assume that each measurement can be traced back to the sensor that collected it, e.g. thanks to an *authentication* scheme.

We now introduce the estimate of each measurement, denoted with $\hat{x}_{t_i}(\mathbf{u}_j)$, which is a function of the measurements given by $N(j)$, i.e. the set of j's neighbours. For the moment, all the other sensors are considered as *neighbours* but we will review the selection of neighbours in Sect. 5.2.1. This aggregated estimate allows us to detect if the sensor has reported a measurement that differs significantly from it. Note that compromised neighbours could collude, as mentioned in Sect. 5.1, to bias the estimate and make it more consistent with the collected measurement. To avoid this problem, separate (pairwise) estimates are obtained from each neighbour. In a second step, the estimates are aggregated by an operator that is resistant to compromised estimates.

5.2.1 Pairwise Estimation

The measurements of two sensors are related, and in particular spatially correlated, because the sensed physical phenomena affect and propagate across the environment in which the sensors are placed. Ideally, the relationship could be characterised in a mathematically precise way, given by the laws of the physical phenomenon and its propagation. In reality, environmental changes, noise, interactions with other phenomena etc., constrain us to work with approximations and more specifically inferred correlations that can be established with a certain margin of error.

Though exploiting spatial correlations, in event detection WSNs a temporal parameter needs also to be accounted for, which is the propagation delay of the event. The latter introduces the *inter-sensor delay* δ_{lm}, which is defined as the time it takes for the event to propagate from $\mathbf{u}_l$ to $\mathbf{u}_m$. This quantity is estimated as the value that maximises the *Pearson correlation coefficient* and use it to align the measurements in time. δ_{lm} is assumed small enough to perform the alignment within the timeliness requirements of the event detection. Note that the calculation of the inter-sensor delay absorbs any synchronisation errors, so the method does not require clock synchronisation. δ_{lm} can change with the value of the measurements, so ideally, would

need to be estimated for each new measurement. This choice is cumbersome (calculation of the Pearson correlation is expensive), and an attacker can also try to subvert this calculation. Thus, the calculation is made only once for each new estimation model, by optimising the average Pearson correlation coefficient. For the remainder of this chapter, the notations $x(\mathbf{u}_i)$ and $x(\mathbf{u}_j)$ refer to two temporal samples collected at the respective location, aligned by the calculated inter-sensor delay, for instance $x_l(\mathbf{u}_i)$ and $x_{l+\delta_{lm}}(\mathbf{u}_j)$.

At this point, we calculate $\hat{x}^j(\mathbf{u}_i)$, which is an estimate of $x(\mathbf{u}_i)$ based on $\breve{x}(\mathbf{u}_j)$. This quantity, is approximated with a linear combination of $\breve{x}(\mathbf{u}_j)$ plus a residual noise random variable $\epsilon^j_{\mathbf{u}_i}$. Noise is modelled with a normal distribution, after removing obvious outliers; when this condition cannot be satisfied, robust regression should be preferred [GH06]. A linear relationship makes the estimation lightweight both in time and space complexity (see Sect. 5.5). Its validity can also be tested statistically, and the neighbourhood of a sensor is defined as the set of sensors for which this relationship holds. When this linear relationship holds less, the estimate is weighted to have a smaller influence on the final result (as shown in Sect. 5.2.2). The equation that approximates x with a function of $\breve{x}(\mathbf{u}_j)$ is referred to as the *estimation model* described below.

Each estimation model is extracted through *linear regression*, which calculates the coefficients $a^j_{\mathbf{u}_i}$ and $b^j_{\mathbf{u}_i}$ in:

$$\hat{x}^j(\mathbf{u}_i) = a^j_{\mathbf{u}_i}\breve{x}(\mathbf{u}_j) + b^j_{\mathbf{u}_i} \tag{5.1}$$

Thus, an estimation model is defined just by the pair $(a^j_{\mathbf{u}_i}, b^j_{\mathbf{u}_i})$, which is calculated as shown in Algorithm 1, so this approach is not burdensome in terms of memory requirements and enables a sensor to store the models for many of its neighbours. In the absence of an a-priori characterisation of the random variables $x(\mathbf{u}_i)$ and $\breve{x}(\mathbf{u}_j)$, sample mean, sample variance and sample covariance are used respectively, estimated on real data.

Regression schemes for anomaly detection in WSN have been previously proposed in [SLD12; SGG10; Zha+12]. However, [SLD12] and [Zha+12] apply regression in the temporal rather than the spatial domain i.e., they estimate correlations between measurements of a single sensor. In [SGG10] the spatial domain is also considered with multiple linear regression; however, this technique is sensitive to outliers, especially when they are correlated because of collusion.

Algorithm 1 Estimation models calculation

Require: $\breve{x}(\mathbf{u})\ \forall \mathbf{u} \in \Omega$
Ensure: $(a^j_{\mathbf{u}_i}, b^j_{\mathbf{u}_i})\ \forall i \neq j$
1: {Initialisation: align the measurements with the inter-sensor delay}
2: **for all** $i \in S$ **do**
3: **for all** $j \in N(i)$ **do** {$N(i)$ indicates i's neighbours}
4: $a^j_{\mathbf{u}_i} = \frac{cov(\breve{x}(\mathbf{u}_i),\breve{x}(\mathbf{u}_j))}{var(\breve{x}(\mathbf{u}_i))}$
5: $b^j_{\mathbf{u}_i} = E[\breve{x}(\mathbf{u}_i)] - a^j_{\mathbf{u}_i} E[\breve{x}(\mathbf{u}_j)]$
6: store $(a^j_{\mathbf{u}_i}, b^j_{\mathbf{u}_i})$
7: **end for**
8: **end for**

Since an attacker can always introduce measurement deviations gradually enough to make them appear consistent, the spatial domain is the only one which has an uncorrupted portion thanks to the presence of genuine sensors.

The Validity of the Estimation Models

In event detection applications, the estimation models need to be valid also when events manifest, otherwise it is likely to detect false anomalies when genuine events manifest. However, an event may induce different relationships between sensors than those present in non-event conditions. This is confirmed by the data sets that have been examined. We address this problem by considering different *modalities* in which the network operates, each corresponding to a set of estimation models. We consider three possible *modality assumptions*:

1. *Unique modality.* The same relationships between sensors hold in event or non-event conditions.

2. *One modality in event conditions, a different one at rest.* This occurs when events induce a different spatial pattern, but all events have similar patterns.

3. *Infinite or unknown number of modalities.* In the most general case.

This chapter caters for assumption 1) and also for 2) since it is a straightforward extension of 1). Two different sets of estimation models are computed: one in event conditions, the other

in non-event conditions. The estimation models learnt in event conditions apply when the detection criterion is satisfied while those learnt in restful conditions are used when no event is detected. Considering different modalities complicates detection, so different modalities should be used only when different behaviours in the relationships between sensors can be identified.

A higher number of modalities can similarly be catered for under assumption 3). This is necessary when multiple events can manifest at once. This problem is dealt with in the next chapter since it requires a different approach, which still relies on measurements estimation, but based on the current context rather than pre-learnt models.

Building Estimations from Sources with Different Accuracy

The accuracy of the estimates is influenced by several factors including the accuracy of the linear relationships, noise, faults, and the informative content of the measurements. Noise is a common source of inaccuracy, especially for sensors in harsh environments. Generally, noise signals can be removed or reduced with standard filters. Another common cause for wrong estimates is the presence of faults. The algorithm alone cannot discern if the wrong estimate was intentional (malicious) or unintentional (faulty), but both will be detected to enable a proper reaction. We assume that a dedicated fault-detection module runs in parallel, and when an anomalous measurements are detected we infer malicious interference if fault-detection does not trigger.

To increase accuracy, the analysis must be applied to the data containing the information of interest, which may not be the raw measurements but information derived from them. For example, the heart voltage measured by electrocardiography is less informative than its frequency. Another example, regarding the use of infrasound sensors for monitoring volcanic eruptions, is shown in Section 5.6.3. In such cases pre-processing the raw signals improves the accuracy significantly.

The accuracy of the estimation models also depends on the existence of a linear relationship between the sensors measurements. Previous work (e.g., [GBS08; TH06; LCC07; Rez+13;

Sun+13]) has considered a stronger assumption that does not allow spatial variations between the sensors, i.e. the measurements are assumed to be so close that their expected value is the same. The technique we present here can be applied to a broader set of applications since it assumes a linear relationship between the measurements. For some WSN applications such as target tracking, linear relationships can be seen only at very close points, and if the WSN deployment is not dense enough there may be no pair of sensors for which such a relationship holds. Therefore, we check the validity of the linearity assumption beforehand with the *squared Pearson correlation*, which is a *goodness of fit* metric for linear regression. In some cases, the approach can also be applied when linear relationships cannot be found, by applying a non-linear transformation to achieve linearity [Lar06]. After the transformation, an increasing Pearson correlation coefficient indicates whether the transformation is convenient.

The goodness of fit allows us also to measure the goodness of the neighbours contribution and weigh it accordingly. Thus, even when considering all the other sensors as neighbours, uncorrelated neighbours will not degrade the result. However, the complexity of the detection algorithm increases with the neighbourhood size (see Section 5.5). Then, a network-wide neighbourhood is realistic only in small scale deployments while in large deployments, the neighbourhoods should be restricted. The principle for selecting the best neighbourhoods is selecting the first sensors in a list sorted by descending goodness of fit: this choice is more robust than distance based criteria since, e.g., in the presence of obstacles, two sensors may be very close but show poor correlation.

Restricting the neighbourhoods does not ensure that all the neighbours are equally correlated, hence the goodness of fit should still be used to weigh the neighbours' contributions. This is done through the *prior weight* ($w^{j-}_{\mathbf{u}_i}$), which is the goodness of fit normalised across a neighbourhood, and captures the relative a-priori confidence in the pairwise estimation model between a sensor and one of its neighbours. By weighting the neighbours appropriately, the accuracy increases with the neighbourhood size.

5.2.2 Robust Aggregation of Pairwise Information

From the previous steps, we have a set of pairwise estimates, calculated with linear regression models. For every new measurement collected by a sensor, we now need to aggregate the pairwise estimates into a final estimate $\hat{x}(\mathbf{u}_i)$ that approximates $x(\mathbf{u}_i)$ and allows us to detect the presence of malicious data injections. To achieve this, $\hat{x}(\mathbf{u}_i)$ must aggregate estimates in a way that is both accurate and minimally corrupted by malicious estimates. In particular, the second requirement demands us not to trust the relationships between different estimates. Indeed, different estimates for the same measurements share some mutual information, or in other words the information brought in by an estimate is reduced by knowledge of other estimates. Nevertheless, such property holds only in the absence of malicious interference. With respect to malicious data injections instead, even two estimates that are expected to be perfectly correlated bring in independent information, since independent probabilities of compromise are assumed for different nodes. For this reason, we will not consider inter-estimate correlation.

Two candidates to aggregate pairwise estimates are *weighted mean* and the *weighted median*: both take as input a set of estimates and their prior weights and return an aggregated value. The *weighted mean* can achieve a smaller error than those of the single estimates. However, it is highly sensitive to compromise, since the final result is proportional to the input values: even one compromised (outlier) estimate can introduce an arbitrary deviation in the result. In contrast, the *weighted median* [WDB10] is more resistant to compromise. It first sorts the values ascendingly, then arranges the weights with the same order, transforms them into substrings with a length proportional to the weight and picks the element at the half-length of the resulting string. Its drawback is that by picking one among all estimates, the error cannot be reduced further.

Since there is a trade-off between accuracy and compromise resistance, the two operators are combined with the following heuristic: first, the weighted median operator is applied; then the weighted mean is calculated with new weights, the *posterior weights* ($w^{j+}_{\mathbf{u}_i}$), obtained as the prior weights times a function which penalises values distant from the result of the first step. Such function is the complementary cumulative distribution function of the estimation error,

where the latter is calculated as the difference between the pairwise estimates and the result of the weighted median.

$$\begin{aligned} p^j_{\mathbf{u}_i}(\hat{x}'(\mathbf{u}_i), \hat{x}^j(\mathbf{u}_i)) &= P(|\epsilon^j_{\mathbf{u}_i}| > |\hat{x}'(\mathbf{u}_i) - \hat{x}^j(\mathbf{u}_i)|) \\ &= 1 - erf(\tfrac{|\hat{x}'(\mathbf{u}_i) - \hat{x}^j(\mathbf{u}_i)|}{\sqrt{2} std(\epsilon^j_{\mathbf{u}_i})}) \end{aligned} \tag{5.2}$$

Where $\hat{x}'(\mathbf{u}_i)$ is the result of the weighted median for sensor i, $\hat{x}^j(\mathbf{u}_i)$ is the estimate given by sensor j, a generic neighbour of sensor i, erf is the error function and $std(\epsilon^j_{\mathbf{u}_i})$ is the residual standard deviation [GH06], calculated together with the respective estimation model. The overall procedure is detailed in Algorithm 2 below, where $\boldsymbol{\hat{x}(\mathbf{u}_i)_{N(i)}}$ are the estimates for i's observed measurement from its neighbours and $\boldsymbol{w^-_{iN(i)}}$ are their respective prior weights.

Algorithm 2 Calculation of the aggregated estimation

Require: $\boldsymbol{w^-_{iN(i)}}$, $\boldsymbol{\hat{x}(\mathbf{u}_i)_{N(i)}}$
Ensure: $\hat{x}(\mathbf{u}_i)$
1: $\hat{x}'(\mathbf{u}_i) = \text{weightedMedian}(\boldsymbol{w^-_{iN(i)}}, \boldsymbol{\hat{x}(\mathbf{u}_i)_{N(i)}})$
2: **for all** $j \in N(i)$ **do** {Calculate the posterior weights}
3: $\quad w^{j+}_{\mathbf{u}_i} = w^{j-}_{\mathbf{u}_i} \cdot p^j_{\mathbf{u}_i}(\hat{x}'(\mathbf{u}_i), \hat{x}^j(\mathbf{u}_i))$
4: $\quad \boldsymbol{w^+_{iN(i)}}.\text{append}(w^{j+}_{\mathbf{u}_i})$
5: **end for**
6: $\boldsymbol{w^+_{iN(i)}} = \frac{\boldsymbol{w^+_{iN(i)}}}{\sum_{j \in N(i)} w^{j+}_{\mathbf{u}_i}}$
7: $\hat{x}(\mathbf{u}_i) = \text{weightedMean}(\boldsymbol{w^+_{iN(i)}}, \boldsymbol{\hat{x}(\mathbf{u}_i)_{N(i)}})$
8: **return** $\hat{x}(\mathbf{u}_i)$

The novel algorithm gives a collusion-resistant and accurate aggregation. It differs from robust aggregators, such as the Orthogonalized Gnanadesikan-Kettenring (OGK) operator [LCC07], since it takes also the prior weights as input to cater for the general case where the values are not equally pertinent to the aggregate. As a consequence, the accuracy of the aggregate always increases when adding more values if the prior weights are correct, while the accuracy of OGK decreases with the introduction of less accurate values. Moreover, the operator exploits the residual standard deviation $std(\epsilon^j_{\mathbf{u}_i})$ to quantify how much a weight should be penalised; OGK instead penalises all values equally, without considering their specific uncertainty. The penalty applied by OGK depends on the data variability measured with the median absolute deviation. This is disadvantageous for collusion resistance, as an attacker may seek to increase

the estimates' variability to be penalised less.

To control the aggregation result, an attacker must ensure that the weighted median is one of the compromised estimates and thus that the sum of the weights of compromised estimates is > 0.5. This condition enables non-detectable injections into a single sensor but is not sufficient to keep the attack undetected. The attacker also needs to control the estimations for the other compromised sensors. The total number of sensors needed to keep all compromised sensors undetected depends on the strength of the pairwise correlations. Instead, the number of sensors needed to mask or elicit an event depends on the event detection criterion. The empirical evaluations show that although a few sensors are generally required to subvert the event detection, a substantial additional number of sensors is required to avoid detection.

5.3 Testing the Deviation from Measurements Estimates

From the estimate aggregation step, introduced in Section 5.2.2, each collected measurement $x(\mathbf{u}_i)$ has an estimate $\hat{x}(\mathbf{u}_i)$ of the observed value. To detect data injections in $x(\mathbf{u}_i)$, the two are compared using a *similarity metric* that must be consistent with the event detection criterion. So, two signals that are similar according to the metric must also have similar effects on the event detection and vice-versa. We therefore introduce two different tests:

1. The *magnitude test*, which verifies that collected measurements are close in magnitude to their estimates.

2. The *shape test*, which verifies that the estimate and collected signal have a similar shape.

The choice of the most appropriate test, or a combination of the two should be made at design time based on the event detection criterion.

5.3.1 Similarity Test 1: Magnitude

In some WSNs, events are triggered when measurements are higher or lower than a reference value. For example, fire alarms trigger when the temperature is above a threshold. An attacker must therefore inject measurements, which differ in magnitude with the observed ones. In such cases we use $M(\mathbf{u}_i) = (\hat{x}(\mathbf{u}_i) - x(\mathbf{u}_i))$ – the difference between the collected measurement and its estimate – to build a *magnitude test*, which checks that the difference is small enough.

The regression *residual*, i.e. the error between a value and its estimate, is assumed zero-mean and normally distributed. Even if $\hat{x}(\mathbf{u}_i)$ is the result of the aggregation described in Section 5.2.2, the error $\epsilon_i = (\hat{x}(\mathbf{u}_i) - x(\mathbf{u}_i))$ can still be assumed to be normally distributed. Indeed, the aggregate is a weighted mean of pairwise estimates, so it equals the true value plus the weighted mean of the pairwise residuals as shown below, where $\epsilon^j_{\mathbf{u}_i}$ denotes the residual in the regression of sensor i's measurement based on sensor j's.

$$\begin{aligned} \hat{x}(\mathbf{u}_i) = \textstyle\sum_{j\in N(i)} w^{j+}_{\mathbf{u}_i} \hat{x}^j(\mathbf{u}_i) = \sum_{j\in N(i)} w^{j+}_{\mathbf{u}_i}(x(\mathbf{u}_i) + \epsilon^j_{\mathbf{u}_i}) \\ = x(\mathbf{u}_i) + \textstyle\sum_{j\in N(i)} w^{j+}_{\mathbf{u}_i} \epsilon^j_{\mathbf{u}_i} \end{aligned} \tag{5.3}$$

Assuming that neighbours have independent residuals (e.g., because of independent noise), ϵ_i is a linear combination of independent normally distributed samples, and is thus normally distributed too [Bry95]. Its mean is still zero, and its variance is:

$$var(\epsilon_i) = \sum_{j\in N(i)} w^{j+2}_{\mathbf{u}_i} var(\epsilon^j_{\mathbf{u}_i}) \tag{5.4}$$

This equation has an important characteristic: the variance of the estimate is a combination of the variances given by each neighbour. Therefore, if a sensor joins or leaves the network, it is sufficient that all its new/old neighbours recompute the variance instead of learning a new one. Since $\frac{\epsilon_i}{std(\epsilon_i)}$ follows the standard normal distribution, also $\widetilde{M}_i = \frac{M_i}{std(\epsilon_i)}$ does when the measurements are genuine. $\widetilde{M}_i$ is referred to as the *magnitude deviation*.

Related studies (e.g., [Pap+03; TH06]) have defined the normal samples with a confidence

interval, characterised by the condition $|\widetilde{M}_i| < \theta$. The threshold θ determines the trade-off between false positives and false negatives, and has been usually set to $\theta = 3$: in this case only 0.3% samples are expected to be beyond the interval. Though 0.3% may seem a small percentage, it needs to be compared with the measurements' sampling period. For instance, with a sampling period of 1 second, each genuine sensor will generate a false positive about every 5.5 minutes. This may trigger the shutdown of the sensor, reconfiguration and/or restart, which can be so expensive that the cost of detection may be prohibitive.

Increasing the threshold reduces false positives, but decreases the detection rate. However, in event detection WSNs the false positives can be partly reduced without losing the detection rate by elaborating magnitude deviations in the same way as the event detection criterion elaborates the measurements. Specifically, if events are detected by applying a function to the measurements in a W_E-long time window, the same function can be applied to the magnitude deviations in the same time window. Consecutive magnitude deviations are unlikely to cause genuine anomalies with a long duration, unless there is a permanent fault that the fault-detection module should detect. Anomalies due to compromise, instead, have a longer duration as the attacker aims to subvert the event detection result. The final step consists of comparing the elaborated magnitude deviation to the *threshold* T_M. In the presented experiments, the algorithm was run on genuine historical data with different values of T_M and selected the lowest value of T_M that achieves a reasonable false alarm rate (calculated as sampling frequency over FPR) for that application.

5.3.2 Similarity Test 2: Shape

Some event detection algorithms trigger based on changes in the time evolution of measurements such as changes in trend or of frequency. These are characteristics of the shape of the signal rather than its magnitude.

A metric that measures similarity in the shapes of two signals is the Pearson correlation coefficient. Since the purpose is to check the shape of the measurements used for event detection, this coefficient is calculated within a moving time window of size W_E: the event

detection time window. Calculating Pearson correlation for all sensor pairs in a neighbourhood would have a computational complexity of $\mathcal{O}(N_N^2 W_E)$, with N_N being the neighbourhood size. In contrast, we evaluate the Pearson correlation coefficient of a sensor's measurements with its estimates, achieving a complexity of $\mathcal{O}(N_N^2 + W_E N_N)$. Indeed, we calculate the coefficient $R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)}$, between W_E consecutive values of $x(\mathbf{u}_i)$ and $\hat{x}(\mathbf{u}_i)$, and compare it against the distribution of $R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_j)}$. Specifically, if the coefficient is below the median[1], the test checks if at least $100 - C_R\%$ samples are expected to be so low by testing $\widetilde{R}_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)} = \frac{R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)} - MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})}{D_{R_i}} > 1$, where D_{R_i} is the C_R-th percentile of $R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)}$.

To eliminate the need for the distribution of $R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)}$, we approximate the quantities $MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})$ and D_{R_i} with $\widehat{MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})}$ and $\hat{D}_{R_i}$ respectively. These are calculated with a heuristic described in Algorithm 3 for a generic sensor i. The best neighbour j^*, for which the median Pearson correlation coefficient is maximum, is chosen. Then $MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})$ is approximated with its median and D_{R_i} with its respective distance to the C_R-th percentile.

Algorithm 3 Characterisation of the distribution of $R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)}$

Require: $R_{ij:j\in N(i)}(r)$, C_R
Ensure: $\widehat{MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})}, \hat{D}_{R_i}$
1: **for all** $j \in N(i)$ **do**
2: $MED_{R^j_{\mathbf{u}_i}} = MED(R^j_{\mathbf{u}_i}(r))$
3: $\boldsymbol{MED}_{\boldsymbol{R^j_{\mathbf{u}_i}}}$.append$(MED_{R^j_{\mathbf{u}_i}})$
4: $r^{LOW} = \{r : r < MED_{R^j_{\mathbf{u}_i}}\}$
5: $D_{R^j_{\mathbf{u}_i}} = \text{percentile}(MED_{R^j_{\mathbf{u}_i}} - R^j_{\mathbf{u}_i}(r^{LOW}), C_R)$
6: $\boldsymbol{D}_{\boldsymbol{R^j_{\mathbf{u}_i}}}$.append$(D_{R^j_{\mathbf{u}_i}})$
7: **end for**
8: $j^* = \text{argmax}_{j\in N(i)}(\boldsymbol{MED}_{\boldsymbol{R^j_{\mathbf{u}_i}}})$
9: $\widehat{MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})} = \boldsymbol{MED}_{\boldsymbol{R^j_{\mathbf{u}_i}}}[j^*]$
10: $\hat{D}_{R_i} = \boldsymbol{D}_{\boldsymbol{R^j_{\mathbf{u}_i}}}[j^*]$
11: **return** $(\widehat{MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})}, \hat{D}_{R_i})$

In the absence of the distributions, $R_{ij\in N(i)}(r)$, $MED_{R^j_{\mathbf{u}_i}}$ and $D_{R^j_{\mathbf{u}_i}}$ are estimated on historical data. Note that $MED(R_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)})$ and D_{R_i} could be estimated empirically, by running the whole algorithm on an ad-hoc dataset. However, the latter must be disjoint from the data used

[1]The samples below the median are characterised since the injected measurements are supposed to have a low correlation with the real values.

to learn the estimation models, otherwise the results could be biased by overfitting [Bis06]. Moreover, the parameters would only be valid for the configuration of sensors in the ad-hoc dataset. Instead, the heuristic removes the need for an ad-hoc dataset and automatically adapts when sensors join or leave the network.

For genuine sensors $\breve{x}(\mathbf{u}_i) = x(\mathbf{u}_i)$, then $\widetilde{R}_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)} \leq 1$ for $C_R\%$ genuine samples. $\widetilde{R}_{x(\mathbf{u}_i),\hat{x}(\mathbf{u}_i)}$ is defined as the *shape deviation*. C_R is calculated as the lowest value that achieves a reasonable false-alarms frequency. The false positives due to short term anomalies can be reduced in a similar way to that used in the magnitude test i.e., by computing the median of W_{Sm} consecutive correlation coefficients calculated on overlapping time windows. W_{Sm} should never exceed W_E, otherwise the information from disjoint time windows would be merged.

5.4 Characterisation

When the similarity check fails for a measurement, the sensor that reported it may have been compromised. However, in some cases the similarity check could also fail on the measurements from genuine sensors, because the wrong modality was chosen (e.g., a non-event modality rather than an event modality) or because the estimation was disturbed by compromised sensors.

The latter scenario may occur when several nearby sensors collude in providing malicious estimates. However, to bias the estimates for genuine sensors by a certain quantity and increase their deviation, compromised nodes need to inject measurements that have even larger deviations (indeed their contribution has weight that cannot be higher than 1). Therefore, we implement the characterisation step by removing the sensors with the highest deviation, one by one, and recomputing the similarity check on the remaining sensors in the neighbourhood. Each time a sensor is removed, presumably compromised, the genuine sensors gain in consistency with their estimate whereas colluding sensors lose the benefits of the removed sensor's estimates. The procedure stops when all the remaining sensors pass the similarity check. The overall characterisation algorithm is shown in Algorithm 4, where SCheck is the similarity check and D_i is the generic deviation (coming from the magnitude/shape tests) calculated for

the similarity check.

Algorithm 4 Characterisation algorithm

Require: $D_i \; \forall i \in S$
Ensure: compromisedSet
1: compromisedSet={}
2: residualSet=S
3: **while** OR$_{i \in \text{residualSet}}$(SCheck($D_i$)$fails$) **do**
4: $s^* = argmax_{i \in residualSet} D_i$
5: compromisedSet.append(s^*)
6: residualSet.remove(s^*)
7: **for all** $j \in S : s^* \in N(j)$ **do**
8: $N(j) = N(j) \setminus s^*$
9: recompute D_j
10: **end for**
11: **end while**
12: **return** compromisedSet

Another factor to consider when the similarity check fails is the *modality assumption* (Sect. 5.2.1). When different modalities are used in event conditions and non-event conditions, there is some uncertainty about which modality to use because malicious data may have compromised the event detection output. In this case, the wrong estimation model may be used and genuine sensors may fail it. Since the event detection algorithm is designed to detect events in genuine scenarios, it may be particularly sensitive to malicious interference. So, the best solution is to not rely on the event detection output, but evaluate both hypotheses, i.e. run Algorithm 4 in both modalities when the similarity check fails. Then, the most likely scenario can be selected.

The likelihood of each scenario can be evaluated by considering that the attack costs increase with the number of sensors that need to be controlled. Hence, the maximum-likelihood criterion chooses the modality in which the smallest compromised set is returned. Note that this is different from event detection with majority voting, since the measurements are not required to trigger in majority, but the majority of them is required to correctly describe an event or a rest condition, i.e. without disrupting correlations.

5.5 Complexity analysis

Our approach relies on computations on raw measurements, which are carried out by a trustworthy node. When the base station collects all the measurements, it is a good candidate for this role as it has a global view, which is useful to deal with collusion of compromised nodes [ZYN08]: this scenario is referred to as *centralised*. This is also the case when the measurements are collected through intermediate *aggregators* but the aggregation is lossless. When the intermediate aggregators perform a lossy aggregation, the base station can instead delegate the detection task to the aggregators rather than forcing delivery of all raw data. In this case, which is referred to as *distributed*, the base station must also assess the integrity of the processing at the aggregators. The centralised and distributed solutions have different computational and communication overheads, analysed below.

For the WSN nodes that do not act as aggregators, the estimation-based framework adds no overhead, because no additional software is run on the sensor nodes to manage votes or trust values like in [GBS08]. For the base station and aggregators, the most computationally expensive operation of the approach is the calculation of the estimation models. When this operation is done one-off, powerful devices may be used offline, but when this is not possible, for instance because there is not enough historical data, the models need to be estimated in real time. In this case, using external devices may be infeasible, and an efficient calculation is required to estimate the models with the sensor nodes. This problem can be dealt with using incremental regression, which updates the model through D new samples with a complexity $\mathcal{O}(D)$ [SL08]. The overall complexity in the distributed case is thus $\mathcal{O}(DN_AN_N)$ for each aggregator, where N_A is the number of sensors managed by the aggregator and N_N is the average number of neighbours used for the estimations. Similarly, in the centralised approach, the complexity is $\mathcal{O}(DNN_N)$. This complexity is generally low for a base station, but may be high for an aggregator: in this case, random sampling and neighbourhood size reduction reduce D and N_N respectively. Note that new sensors joining the network require the computation of further regression problems, $2N_N$ on average, so the total overhead for a new node joining is $\mathcal{O}(DN_N)$.

Besides the calculation of the estimation models, the estimations calculation and the similarity check have a computational overhead. Each aggregated estimation requires N_N multiplications and sums, while the similarity check has a complexity $\mathcal{O}(W_E)$ (for the shape test it is the complexity of Pearson correlation and for the magnitude test it is the complexity of the operators commonly used, such as mean, median, etc.). Since these operations need to be repeated for each sensor, the overall time complexity is then $\mathcal{O}(N_N^2 + W_E N_N)$ and $\mathcal{O}(N_N N + W_E N_N)$ for the centralised and distributed case respectively. Note that the neighbourhood size is the parameter to reduce in large WSNs, e.g., if neighbourhoods are structured into a tree, the complexity can be reduced to $\mathcal{O}(Nlog(N) + W_E N_N)$ and $\mathcal{O}(log^2(N) + W_E N_N)$ in the centralised and distributed cases respectively.

Regarding the communication overheads, since in the centralised solution the data needs to be conveyed to the base station anyway, the framework does not introduce additional overhead as no additional packets are transmitted, in contrast with [ZYN08]. In the distributed scenario, instead, the base station needs to assess the integrity of the aggregation at each aggregator. For this task, solutions have been proposed with a communication overhead sublinear in N, such as the *aggregate-commit-prove* approach [PSP06]. This technique is based on three steps. First the aggregator collects the measurements and aggregates them. Then, it reports the result to the base station together with a commitment value. The base station challenges the aggregator by requiring some of the measurements and additional information to verify the commitment value. Based on such information, the base station proves with a certain probability that the committed measurements are authentic and have not been changed during the challenge. Note that the overhead of verifying the integrity of the processing done by the aggregators nullifies the advantages of distributing the processing on the measurements. Additionally, since the detection is only performed within a cluster, an adversary may be able to compromise a significant number, or even all, of the sensors in a cluster. In Chapter 7.3, we show that the minimum number of sensors that an attacker needs to compromise to make any undetected and dangerous attack decreases when fewer measurements are available. In conclusion, lossy aggregation may be a convenient approach when the integrity of the measurements can be violated by the aggregators only. Instead, when the measurements are compromised before being

transmitted to the aggregators, e.g. because of compromised sensor nodes, the advantages of distributed computations are overcome by the disadvantages of requiring extra-communications and reducing the ability of detecting malicious measurements.

5.6 Experiments

Different WSN applications sense different processes, have different semantics and different deployments. It would be cumbersome to design new data injection detection algorithms for each new set of circumstances. Yet few, if any, algorithms proposed in the literature have been shown to work in different scenarios or identify how they can be tailored to them.

We evaluated the algorithms and methodology in three very different contexts: *Monitoring health parameters* (Sect. 5.6.1), *detecting volcanic eruptions* (Sect. 5.6.2) and *home fire alarms* (Sect. 5.6.3). In each case, the data is divided into two datasets: A historical dataset, from which the estimation models and other parameters are learnt, and a test set, which represents the online measurements and is used for evaluation.

Ideally, the datasets should include both genuine measurements and malicious data injections in order to calculate the False Positive Rate (FPR) and the False Negative Rate (FNR). However, real malicious data injections are not common yet, so the attacks need to be simulated in each case to study the algorithm's behaviour. Since it is reasonable to assume that the test sets used in the experiments are free from malicious injections, the FPR estimates are still reliable. However, FNR estimates are not, so they are omitted.

Attacks are considered where non-existent events are elicited and where real events are masked. The attacker model is the one presented in Chapter 4. In summary we assume attackers that:

1. Can target the best C out N sensors (i.e., those that can mislead the event detection algorithm with maximum chance of remaining undetected).

2. Have infinite time for the attack.

3. Can overhear the measurements of the other sensors.

The attacks are simulated by injecting measurements describing normal circumstances (i.e. absence of faults and compromise) but that subvert the event detection result, i.e. elicit a non existent event, or mask a real event. In some cases, the attacker may need to inject measurements substantially different from the observed ones, but this will not be easily noticeable because the data describes wrong but still normal circumstances. Moreover, the measurements injected from different compromised sensors collude to be perfectly adherent to the expected correlation, thus estimates from two compromised sensors will always support each other. Note that such sophisticated attacks require a sound and well-planned strategy, and thus are difficult to automate. The purpose of the following experiments is indeed to show that, even with a sophisticated collusion strategy, the algorithm is capable of detecting the attack and, under certain conditions, to correctly characterise the compromised sensors. Even though experiments based on naive principles, such as random increases in the measurements magnitude or amplification of noise power, lead to experiments that are easier to automate and to results that are easier to interpret, they do not give any information about the ability to detect attacks in real scenarios.

All the experiments are compared with a *majority voting* framework, since it includes a large amount of related work, e.g. [LCC07; ZYN08; Hin09; Sun+13]. In addition, majority-voting is a generic framework, so we can study the benefits of our estimation-based framework by using the same algorithms proposed in this chapter, to make the comparison as fair as possible. So, we will use a local version of the magnitude test for majority voting, where the votes are the pairwise magnitude tests below:

$$\frac{\epsilon^j_{\mathbf{u}_i}}{std(\epsilon^j_{\mathbf{u}_i})} < T_M \tag{5.5}$$

Moreover, we will use a local version of the shape test, where the votes are the pairwise shape tests below:

$$\frac{R_{x(\mathbf{u}_i),x(\mathbf{u}_i)} - MED(R_{x(\mathbf{u}_i),x(\mathbf{u}_j)})}{D^j_{\mathbf{u}_i}} < T_S \tag{5.6}$$

The threshold T_M or T_S are set to achieve the same FPR achieved by the algorithm and

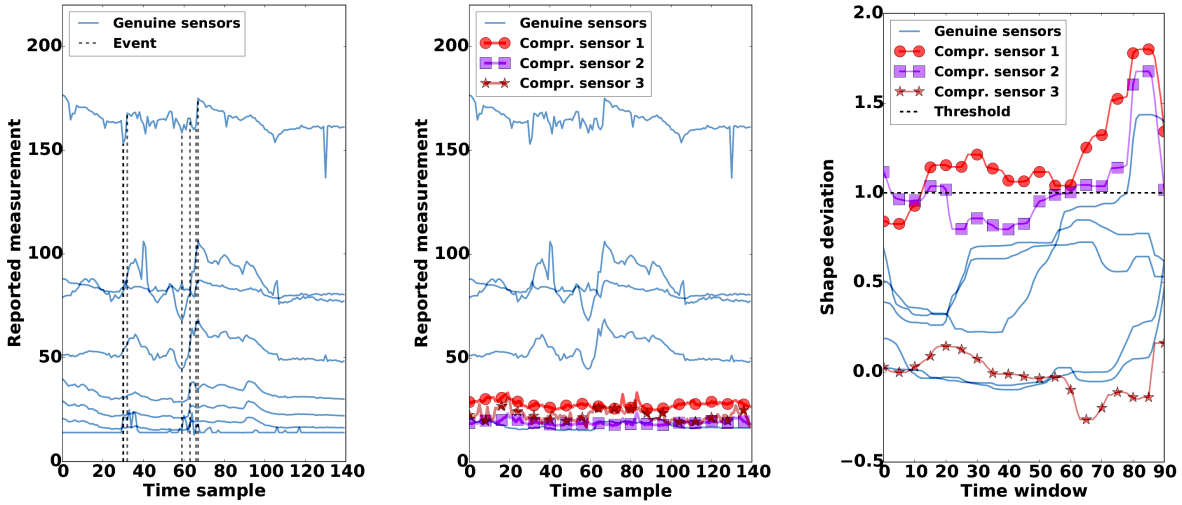
classify as compromised the nodes for which at least 50% votes are collected. When introducing collusion, the compromised nodes are assumed to not report on themselves, while they report on the genuine nodes.

5.6.1 PhysioNet Dataset

The *PhysioNet MIMIC II Waveform Database* [Phy] contains thousands of recordings collected from bedside patient monitors in adult and neonatal intensive care units. The parameters considered are: blood and pulmonary arterial pressure (mean, systolic, and diastolic), heart rate, pulse, and respiration. Note that the sensors measuring different physical phenomena are considered; however, the algorithm is still applicable since it abstracts from the physical meaning of the measurements and only requires that there is correlation between them. The event detection algorithm that is used is the one described in [Sal+14] and tested on the *PhysioNet MIMIC Database*, a predecessor of *PhysioNet MIMIC II Waveform Database*. An event is triggered when there are high temporal variations in the measurements of the sensors within a time window, which in the experiments is 40 samples long. Table 5.1 summarises the experimental setup.

Table 5.1: PhysioNet experiment setup

PhysioNet		
Data	Content	Health-related parameters
	Sampling period	1 minute
	Number of sensors	8
	Historical data size	6282 samples per sensor
	Test set size	3412 samples per sensor
Event Detection	Criterion	Described in [Sal+14], based on temporal variations
	Time window size	40
Algorithm	Similarity check	Shape test
	Similarity check parameters	$W_{Sm} = 10$, $C_R = 99.7$
	Modality assumption	Unique modality for event presence and absence
	FPR	0.002
	Expected false positive frequency	1 in 8 hours



(a) Alarms with genuine data (b) Alarms with compromised data (c) Shape deviation

Figure 5.4: PhysioNet Dataset: masking attack. Alarm conditions are present and the compromised sensors mask them all. The shape test fails on 2 compromised sensors and also on a genuine one, because of collusion.

Fig. 5.4a shows the original measurements from the dataset and the events detected. To simulate malicious data injections, the attacker is simulated to control 3 sensors and inject measurements registered by the same sensors while monitoring healthy patients. If the attacker chose uncorrelated sensors, well correlated neighbours would detect the attack. Instead, 3 correlated sensors are chosen to inject correlated values from the same sensors in non-event conditions that also show the best correlation with the remaining sensors: the sensors for mean, systolic, and diastolic blood pressure. As shown in Fig. 5.4b, no event is detected after the attack, although the patient is subject to a life-threatening condition.

To apply the algorithm, the similarity check needs to be specified first. Since the event detection triggers on fast increasing or decreasing measurements, regardless of their magnitude, the shape test is more appropriate. The shape deviation for the data after the malicious data injections is shown in Fig. 5.4c. Note that one of the genuine sensors also fails the shape test in the last part; this is due to the collusion effects explained in Sect. 5.4. Nevertheless, the characterisation algorithm described in Sect. 5.4 correctly returns the set of compromised sensors, and does not include any genuine sensor.

The experiments were run with other sets of compromised nodes. The attacker needs to compromise at least one more sensor to become undetected – the one closest to those already

compromised (Fig. 5.4b). When this sensor is also compromised, each compromised sensor has 4 genuine neighbours and 3 compromised ones, but there is not enough information to determine if the estimates of the genuine neighbours disagree because of errors or because the other sensors are malicious. Nevertheless, in this scenario an attacker needs to compromise 50% (4 out of 8) sensors to remain undetected.

With majority voting, no detection was achieved at all when 1 or 2 compromised sensors were injecting malicious measurements. In the case of Fig. 5.4b, with 3 compromised sensors, it did not identify any malicious sensor but also misclassified 2 genuine sensors as compromised.

5.6.2 Home Fire Alarm Dataset

The second dataset originates from a WSN conceived for monitoring homes to generate fire alarms. It is available from the NIST website [Nis] as part of the Home Smoke Alarm Tests project (see NIST Technical Note 1455 [Bukowski]). The data has been collected by three groups of temperature sensors in three adjacent rooms, each group made up of 5 sensors placed on a wall at different distances from the ceiling. Table 5.2 summarises the experiment setup.

In this case, the second modality assumption (Sec. 5.2.1) was used, where different relations between sensors are present in event and non-event conditions. Intuitively, in the absence of a fire, temperatures are broadly uniform but a fire introduces spatial patterns across the rooms that are significantly different. Linear relationships are therefore derived for both modalities, the event detection algorithm defines which modality applies, and finally the estimates are calculated according to that modality.

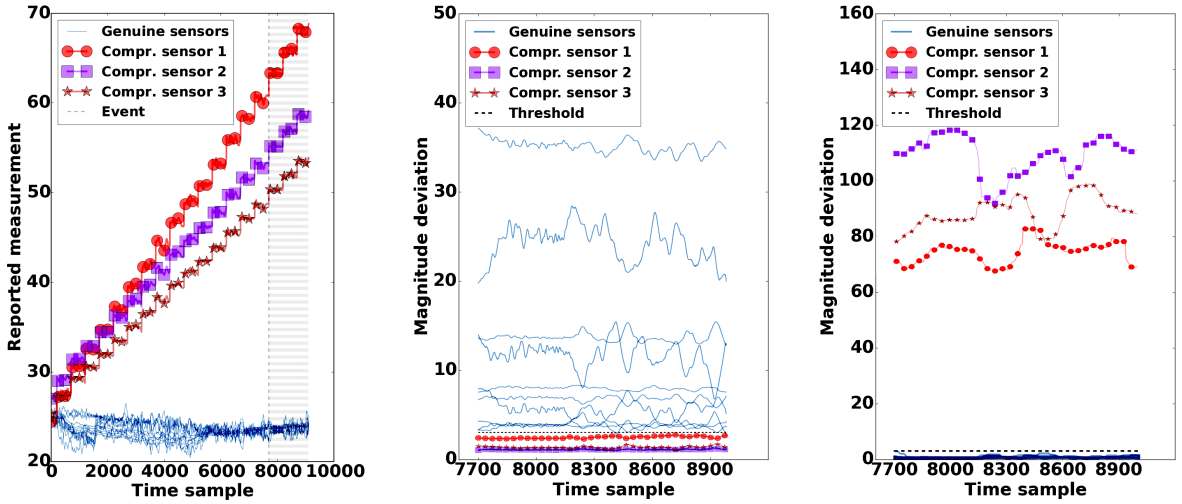
The synthetic attack consists of eliciting a false fire alarm. Here, an event is detected when a fixed temperature is reached, in compliance with the detection algorithm for fixed temperature heat detectors [Bukowski]. Such devices generally trigger at temperatures between 47 °C and 58 °C, so 50 °C is used in the experiment. To make the signals more credible 3 compromised sensors that collude in reporting high temperatures have been simulated. These progressively increase the measurements of 3 sensors as shown in Fig. 5.5a, eventually triggering a false fire

Table 5.2: Home Fire Alarm experiment setup

Home Fire Alarm		
Data	Content	Temperature readings in a room at different distance from the ceiling
	Sampling period	2 seconds
	Number of sensors	15
	Historical data size	14769 samples per sens. at rest, 800 in event mod.
	Test set size	9210 samples per sens. at rest, 312 in event mod.
Event detection	Criterion	One sensor observes 3 consecutive measurements above 50 °C
	Time window size	3
Algorithm	Similarity check	Magnitude test
	Similarity check parameters	$T_M = 3$
	Modality assumption	One modality for event-presence, one modality for event-absence
	FPR	0.003 in event mod., 0 at rest
	Expected false positive frequency	1 every 3000 fires

alarm.

Since fire events are detected based on the magnitude of measured values, the magnitude test is more appropriate for this application. As explained in Sect. 5.3.1, the magnitude test reflects the event detection criterion, which triggers alarms when the temperature is high. As shown in Fig. 5.5a, the event is detected around sample 7700, hence, in the first place, the estimation models learnt under event-conditions apply here. Note that the measurements of the genuine sensors are not consistent with the presence of the fabricated event, hence the similarity check fails for most of them as shown in Fig. 5.5b and the characterisation algorithm identifies 12 nodes (10 of which are genuine). Since the check failed, the characterisation algorithm (see Section 5.4) needs to decide if the event modality is the correct one and it does so by running the detection again on the same measurements but with the estimation models for the non-event modality. This time, the characterisation algorithm returns the 3 compromised sensors, a smaller set than the 12 nodes identified with the event modality, so the non-event modality is



(a) Alarms with compromised data, represented with vertical lines

(b) Magnitude deviation in alarm modality

(c) Magnitude deviation in non-alarm modality

Figure 5.5: Home Fire Alarm Dataset: eliciting attack. The 3 compromised sensors trigger the fire alarm. Many sensors fail the check in the wrong modality. In the correct modality, the compromised sensors can be easily identified.

correctly chosen and the compromised sensors are correctly detected. The magnitude deviation under the non-event modality is shown in Fig. 5.5c.

The experiments were run with other sets of compromised nodes. The attacker needs to compromise at least 7 sensors to remain undetected. In this scenario, 11 sensors are identified in the non-event modality and 8 sensors (all the genuine ones) in the event modality, therefore the event modality is wrongly chosen and the genuine sensors are classified as malicious. In this scenario the attacker needs to compromise at least 47% sensors for a successful undetected attack.

For this experiment majority voting detected correctly the compromised nodes when 1 sensor was compromised. Detection succeeds with 2 compromised nodes but misclassifies 2 genuine sensors. In the case shown in Fig. 5.5a with 3 compromised sensors, detection fails and 4 genuine nodes are misclassified. Note that the real limit of majority voting here is 13% (2 sensors), which is far below the 50% theoretical limit. Such limit requires correct votes from the genuine nodes, which are guaranteed only with perfect correlation.

5.6.3 Reventador Volcano Dataset

Finally a dataset gathered from a WSN of infrasound sensors is considered. The WSN was deployed at the Reventador volcano by the Harvard Sensor Networks Lab [WA+06b]. The network consists of 16 sensor nodes deployed every 200-400 metres along a side of the volcano. The sensors are connected to a remote base station, which waits for the sensors to trigger the presence of an event (which may reflect earthquakes or eruptions): if at least 30% sensors trigger, the base station collects the last 60 seconds of data from all the sensors to analyse the event. The event detection algorithm, given in [WA+06b] is based on temporal changes in the measurements' average. Here, the measurements shown in Fig. 5.6a are considered, which trigger the event detection[2]. Table 5.3 summarises the experiment setup.

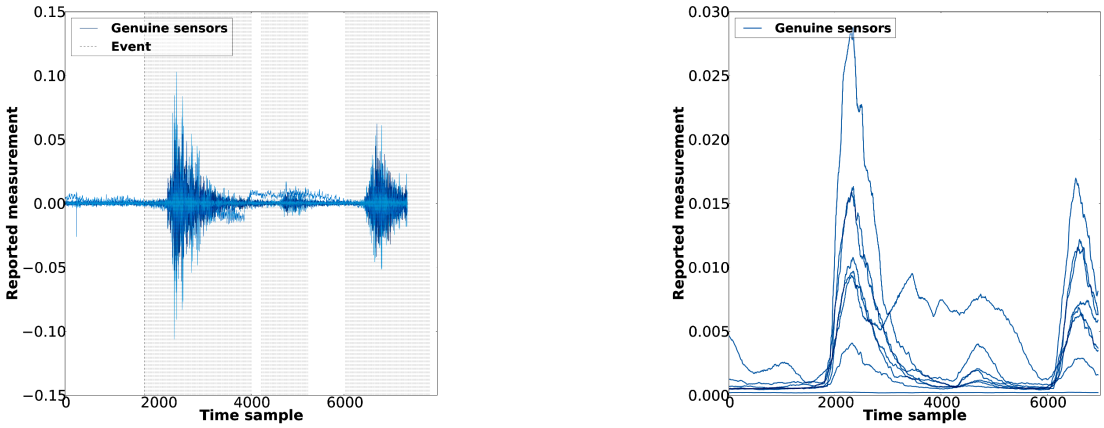
Table 5.3: Reventador experiment setup

Reventador Volcano		
Data	Content	Infrasound readings along a side of the volcano
	Sampling period	0.01 seconds
	Number of sensors	8
	Historical data size	28471 samples per sensor
	Test set size	8398 samples per sensor
Event detection	Criterion	EWMA [WA+06b], based on changes in temporal mean
	Time window size	6000 samples
Algorithm	Similarity check	Shape test
	Similarity check parameters	$W_{Sm} = 1$, $C_R = 99.7$
	Modality assumption	Unique modality for event presence and absence
	FPR	0
	Expected false positive frequency	0

Since the infra-sound data is made up of oscillations around zero, applying the algorithm to the raw data would be inappropriate since the measurements are mostly uncorrelated and uninformative. In infrasound measurements, the valuable information is mostly contained in

[2]Note that only 8 sensors are analysed since the remaining 7 sensors were not working during the observed time.

the trend of the peak values, which can be captured with a pre-processing step that averages the measurements' absolute value in a short time window. From the graphs in 5.6a the peak values appear generally consistent for about 400 data samples, so a good pre-processing time window is of 400 samples. The pre-processed measurements are shown in Fig. 5.6b.



(a) Alarms with genuine data

(b) Pre-processed genuine data

Figure 5.6: Reventador Dataset. Alarming conditions are present. The measurements are noticeably more correlated after the transformation.

Table 5.4: Reventador shape deviation. The values that failed the shape test are in boldface

Compromised Sensors	Event masked	S_1 dev.	S_2 dev.	S_3 dev.	S_4 dev.	S_5 dev.	S_6 dev.	S_7 dev.	S_8 dev.	Characterisation output
2	NO	0.03	**25.13**	0.32	-0.04	0.05	-0.04	-0.07	0.03	2
2,6	NO	0.07	**25.27**	0.23	-0.05	0.06	**8.48**	0.03	0.08	2,6
1,2,6	NO	**8.12**	**25.15**	**4.47**	0.01	0.07	**8.66**	**2.52**	**1.40**	2,6,1
5,6,7	NO	-0.01	0.31	0.09	-0.05	0.89	**8.55**	**13.36**	0.08	7,6
5,6,7,8	NO	**2.04**	**6.50**	**7.22**	-0.05	0.89	**8.41**	**13.17**	**6.75**	7,6,8
1,2,3,5,6,7	YES	**4.28**	**11.72**	**15.14**	0.04	0.48	**5.63**	**8.07**	**6.94**	3,2,7,1,6
1,2,3,5,6,7,8	YES	-0.05	-0.07	-0.05	0.40	-0.98	-0.05	-0.14	-0.10	N/A

For this experiment, masking attacks are simulated to silence the event detection by injecting measurements taken from the same sensors, but in restful conditions, i.e. when there is no volcanic activity. In the pre-processed data, injected measurements appear as roughly constant data, i.e. without increasing/decreasing trends. According to the event detection algorithm described in [WA+06b], an event occurs when 2 or more sensors trigger (30% of 8). The event is triggered by all the sensors except sensor 4, which is the one with a roughly flat signal in Fig. 5.6b. So, to mask the event, an attacker needs to compromise at least 6 sensors that do not include sensor 4. As conveyed in Section 5.3, the similarity check should be tailored to the event detection algorithm, that in this case is a exponentially-weighted moving average (EWMA). Since EWMA triggers with variations in the time series, including sudden but small

variations, the shape test is more appropriate than the magnitude test for the similarity check.

A first experiment simulates a scenario where only sensor 2 is injecting malicious measurements. The results are shown in Table 5.4. The shape test unequivocally recognises the inconsistency of the measurements from sensor 2. Sensor 4 is also not triggering, however the similarity check does not fail on it since its measurements are consistent with the estimation models, which indirectly captured the characteristic that the sensor may not trigger even if all the other sensors are triggering.

The number of sensors injecting malicious measurements is progressively increased. Table 5.4 summarises the output from the characterisation algorithm in the order in which sensors are found. Note that whenever the similarity check fails, the genuine nodes that failed the test because of the collusion effect are finally classified as genuine. With the exception of sensor 5, all the compromised sensors are correctly detected by the characterisation algorithm. Sensor 5 is the sensor in Fig. 5.6b, whose measurements mostly dissociate themselves from the others. The behaviour of this sensor is therefore less predictable, and a considerable deviation is required to make the similarity check fail.

When sensors {1,2,3,5,6,7} are all injecting malicious measurements, the masking attack eventually succeeds. However, the algorithm still detects the attack, even though 75% of sensors are reporting malicious measurements, as the colluding sensors did not succeed in making the measurements credible. The attacker needs to compromise all the triggering sensors (88% of the total) to carry out an undetected masking attack as shown in the last row of Table 5.4.

In the same settings, majority voting fails with 1 compromised node (13%). With 2 or 3 compromised nodes (25%, 37%), the detection succeeds but 2 genuine nodes are classified as malicious. With 4 (50%) nodes, it reaches its theoretical limit, so detection clearly fails. Instead, the algorithm detects the attack without false positives when even 75% sensors are compromised and reaches its limit at 88%.

5.6.4 Discussion

The current approach has shown to detect malicious interference also with sophisticated attacks, based on injection of credible measurements. The characterisation algorithm correctly detects the set of compromised sensors when the number of genuine sensors is low compared to the expected correlation. Note that, in the approach, the number of compromised sensors that can be tolerated is correlation-dependent. In one of the experiments, attacks could be detected whenever fewer than 88% sensors were compromised. Voting-based frameworks instead, cannot tolerate more than 50% compromised sensors and, when the algorithm tolerated less than 50% compromised sensors, majority voting tolerated a substantially lower percentage. The reason behind this result is that the correlation between the sensors used in the experiments is not high enough to guarantee correct votes from all the genuine sensors and votes become inaccurate. The approach deals with such inaccuracy by merging the contributions from each sensor, weighting the contributions according to their expected accuracy, and discarding potentially unreliable contributions.

The attack detection has proved not always sufficient to correctly identify the compromised nodes, especially when the correlations change dramatically between restful and event conditions. In this case, the presence of a problem can be easily detected but it is not easy to infer whether the restful or event-related measurements are correct. The presented approach chooses the most likely condition, but it may be possible to reject the detection when there is not enough confidence in it. The final decision may be taken otherwise and may require human intervention.

5.7 Conclusions

The proposed solution improves upon state-of-the-art approaches and provides:

1. A method to run the detection on broad neighbourhoods where the measurements of two neighbours can significantly differ (but are still correlated).

2. An extension of voting-based and trust-based frameworks to *estimation*-based frameworks.

3. A general methodology to flexibly tailor the technique to WSN applications that detect different kinds of events.

Contribution 1) consists of extending the assumption of having a unique monitored value, which is generally valid only for very small neighbourhoods, where collusion attacks can be successful by compromising all the sensors. To resist collusion, it is necessary to broaden neighbourhoods and cope with measurements that are significantly different. But in the absence of a common ground truth, it is not obvious what a sensor's collected measurement should be compared against. Previous studies, such as [CP07; Raj+10; SGG10], have proposed to detect inconsistencies in the correlation within a neighbourhood by extracting a unique overall consistency metric, to which every neighbour contributes. This, however, allows colluding sensors to compensate for each other and reduce the overall inconsistency, whilst still disrupting the collected values [LNR11].

An approach similarly based on aggregation of individual sensors' information is used in majority voting and trust-management frameworks. However, the main drawback of these techniques is that they introduce an additional variable - the vote, or trust value - about which an attacker can lie with or without lying about the measurements at the same time. Detecting such attacks incurs additional computation and communication costs. In contrast, the contribution 2) extends voting-based and trust-based frameworks by *aggregating measurements estimates rather than votes or trust values.* Such choice does not introduce additional variables, since the estimates are directly calculated from the raw measurements.

Since no general methodology has been proposed for systematically tailored to different deployments and different applications, contribution 3) deals with this aspect. In addition, sophisticated collusion attacks are described that are application-agnostic and therefore can be used as a generic testing criterion for assessing the robustness of different algorithms.

The proposed algorithm provides detection of malicious data injections in event detection

WSNs, in particular when collusion between compromised sensors occurs, in a way that can be customised and used in different applications, and for different kinds of events.

Dealing with collusion and the occurrence of events makes the problem of detecting malicious data injections significantly more complex because both affect the dynamics of the system and comparisons between measurements. Furthermore, they interact with each other as collusion may leverage deviations in sensed values introduced by the event.

Addressing this challenge has exposed several trade-offs in the design of the algorithm. Firstly, resistance to collusion requires comparison to measurements over a broader set of sensors and thus introduces additional complexity and computational cost. This trade-off is particularly visible in the selection of neighbourhoods, which becomes a simple ranking-based choice when using the pairwise estimation models. Another trade-off arises when merging information with potentially malicious sources. While information coming from genuine sensors increases the estimates accuracy, it is important to select only information that appears reliable. Colluding sensors should not be allowed to compensate for each other in the detection metric whilst still injecting malicious data. This requires the use of pairwise comparisons and an aggregation operator that is accurate in the presence of genuine measurements as well as resistant to malicious data.

The methodology applied in three applications, where requirements and the nature of the events is markedly different, has proved that the development of a general framework to cope with malicious data injection in event detection WSNs is possible. However, it requires customisation of the parameters based on a collection of historical data and information about the application's goals and requirements. The methodology for customisation, on the other hand, can be provided with a generic but well defined procedure.

Experimental results validated the choice of structuring the detection on top of simple techniques that, without introducing significant overhead in the sensor nodes, achieve high detection rates.

Having designed the methodology and algorithms specifically for the scenarios where one event

manifests, and with a pattern that can be learnt from historical data, we have optimised the problem solution for a number of applications, such as volcano monitoring, home fire alarms, and health related parameters. On the other hand, extending the approach to other scenarios, where events can be multiple and there are so many spatial patterns that it is difficult to learn them all, requires a certain effort. In particular, the estimates need to be contextualised in the unpredictable event's characteristics, and a detection method is needed that copes also with estimates that are highly affected by malicious interference. These problems are tackled in the next chapter.

Chapter 6

Securing Multiple Events with Variable Patterns

In the previous chapter, we have considered events that manifest with recurring patterns, such as seismic activities on volcanoes, home fires, and physiological conditions. For such applications, two events differ mainly in their intensity and the way they evolve in time, hence pairwise linear models can model with sufficient accuracy all events. However, in other applications such as hazardous gas detection, smart grids, and traffic monitoring among others, multiple events occur simultaneously, and the number, location, and intensity of these events affect drastically the event manifestation.

In these contexts, we refer to events as events with unpredictable pattern. When events with unpredictable patterns occur, more sophisticated attacks are possible. For instance, a real event can be elicited to appear larger or masked to appear smaller, a spoofed event can be introduced close to a genuine event to gain credibility, a real event can be split into two or more events, etc. A prototypical application with such characteristics is that of wildfire monitoring, where the fire can be modelled as a collection of points where the fire area is wider and propagates to other locations, with a pattern that is not known a-priori. An example of a spoofed fire next to a genuine fire is shown in Fig. 6.1. Distinguishing the two fires is a complex task, so the spoofed event gains credibility.

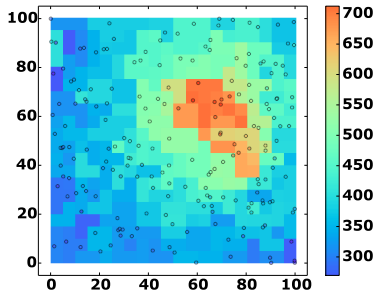


Figure 6.1: Temperatures (in Kelvin) in a wildfire monitoring WSN: axes represent the spatial location, circles identify the sensors and colours map the measurements space. The spoofed event centred at (80,40) is difficult to isolate from the genuine event centred at (65,60).

With variable patterns the attacker has more freedom given by the uncertainty regarding the event-related information. This is the reason why applying techniques designed for events with variable patterns when events show a regular pattern is not an optimal choice.

Events with variable patterns invalidate the technique presented in the previous chapter, since the estimation of a sensor's measurement based on the measurement of just one other sensor has too much uncertainty to provide meaningful information.

The approach pursued in this chapter is to revert to estimations based on multiple contributions. Nevertheless, as discussed in the previous chapter, when the estimations are extracted from multiple contributions, collusion attacks can bias the estimate. Hence, the estimates are not used to approximate the true value, but rather to identify change trends, which can then be judged as collusion attempts or genuine changes.

When the event's pattern can significantly change, the inter-measurements correlations depend on the event itself. Hence, the relevance of each measurement to the estimate may not be calculated from historical data as it could change drastically from one event to another. However, since the monitored phenomena generally have a spatial coherence that decreases with distance, it can be assumed that close sensors produce measurements that are more likely to be correlated.

These considerations lead to the construction of a new estimation model, which rather than learning the correlation between the measurements of two different sensors, evaluates the variation between them in context. For instance, if the event is located close to one of the two

sensors but further from the other, we may expect a large variation between the two sensors. The basis of this new approach is producing measurement estimates as a weighted average of other measurements, with weights that decrease with distance. This is the idea behind local regression ([Loa06]), where the expected value of the data to estimate is calculated through a regression model, with locally defined regression weights. The crucial ability of the new model is to contextualise the information with respect to the information in a neighbouring area.

6.1 Local Regression of Scale-Specific Trends

Local regression is able to give an estimate for a value, given a subset of neighbouring values.

In particular, a regression model that makes a locally weighted average of the data is the *Nadaraya-Watson estimator*. Such an estimator will be used in the following to detect changes in the measurements' trends. The weights of this estimator are calculated with a *kernel* function, i.e. a symmetric function which integrates (or sums up, if discrete) to one over its domain.

$$\eta_{s,\boldsymbol{\tau}} = \frac{k(\frac{|\mathbf{v}-\boldsymbol{\tau}|}{s})}{\sum_{\mathbf{u}\in\Omega} k(\frac{|\mathbf{u}-\boldsymbol{\tau}|}{s})} \tag{6.1}$$

The Nadaraya-Watson estimator works as a low-pass filter, thanks to the average operators, where the bandwidth of the filter depends on the variable s. The kernel k determines how the weights decay as the distance from the centre $\boldsymbol{\tau}$ increases. A typical kernel that is used is the Gaussian kernel $e^{-|\mathbf{v}|^2}$. This kernel smoothly decreases with distance, and after a certain distance it can be approximated to zero. Moreover, since a Gaussian transforms to a Gaussian with the Fourier transform, the function applied in the spatial domain is not only space-limited, but also approximately band-limited, so it gives a precise selection of both spatial and frequency contributions.

The low-pass filtering of the estimate filters out the dynamic component, i.e. the high frequencies. In such frequencies we find information that is precious to identify malicious compromise: group-wise (mid-frequencies) and individual (high frequencies) behaviours. To study also higher

frequencies, a set of band-pass filters is required. In the following section, a mathematical tool that is equivalent to the application of a set of band-pass filters is presented: the *wavelet transform*. The wavelet theory that is important to measurements inspection is first presented. Then, we show that the Nadaraya-Watson estimator can be used as the basis for a special kind of wavelet transform.

For the following discussions, we will assume that the original domain of analysis is the temporal domain, as the wavelet theory was originally developed for temporal signals. Afterwards, we will transition to the spatial domain, where wavelets still find many applications such as image processing [CS05]. Similarly to image processing, we will consider the measurements as a spatial signal with correlated samples. However, whilst images are uniformly spaced, measurements signals generally are not. Therefore, we will briefly introduce the temporal wavelet transform, move to the spatial domain, and finally introduce a more recently developed type of wavelet transform that copes also with non-uniformly spaced samples. Afterwards, we will introduce a novel data analysis method, which makes use of a state-of-the-art wavelet transform to analyse the cross-scale information contained in the WSN measurements. This will be the basis for our new algorithm to tackle malicious data injections in WSNs where multiple events manifest with unpredictable patterns.

6.2 Wavelet Transform: A Tool for Extracting Changes in Scale-Specific Trends

The wavelet transform provides analyses at multiple scales (which are linked to the signal's frequencies) and also at different translations, i.e. points in time/space (according to the domain of analysis). Data processing applications involving the wavelet transform include *noise removal*, *pattern recognition*, and *data compression*. When noise can be modelled as a random process which is independent and identically distributed in time and space, its contribution is mainly observable in the lower scales, corresponding to the high frequencies, which may be filtered out. Pattern recognition is a process that classifies the data, based on recurrent

characteristics, known as *features*. Wavelet decomposition is able to separate singularities in the data from the background. This can be used, for instance, to identify the edges of an object inside an image, and exploit information about the object's shape for the pattern recognition task. Data compression with wavelets consists of discarding some of the lower scales, similarly to noise filtering. However, some signals are particularly interesting when there are spikes or discontinuities. In this case, the lower scales cannot be discarded, but thanks to the multi-scale and multi-location decomposition they can be represented compactly.

Wavelets are a powerful tool for data analysis tasks, since they provide information which may be more comprehensible than raw data thanks to the multi-scale analysis. For instance, when analysing the most recent samples of a time series, we may observe a decreasing trend at high scales and an increasing trend at low scales. In practice, this means that the values of the time series have decreased in the long-term, but a more recent increase is observable, which may either be a short-term increase or the change point for a new long-term increase. In summary, scale isolation gives different views of the same phenomenon at different degrees of granularity.

The characteristics of wavelet transforms are particularly advantageous for detecting malicious measurements in WSNs, especially when there is no particular source of knowledge to compare against, such as a set of measurement estimates. Indeed, the multi-scale analysis allows one to contextualise the low scale information, which includes the information coming from one sensor or a small group of sensors, within the high scale information, which includes the information coming from many sensors. The latter includes also the effects of events, so we exploit the high scale information to characterise the event, and check that it is consistent with the information provided by the individual sensors. Since the scales are related to the signal's frequency, the same problem may be tackled also with the Fourier transform. In fact, the wavelet transform is often seen as a competitor of Fourier transform. However, wavelets are more suitable for our problem. In brief, this is because wavelets can localise the spectral information in different areas of the WSN deployment space, and also because the low scales, i.e. the high frequencies can be reconstructed more compactly with wavelets. Below we analyse the main differences between wavelet and Fourier transform in more detail.

Wavelet VS Fourier Transform There are several analogies between wavelet and Fourier transform. Both approaches decompose the input signal into different components which make a selection in the frequency domain. While the selection is extremely accurate with the Fourier transform, the wavelet coefficients act like a band-pass filter, i.e. they select a *range* of frequencies. In particular, the latter depends on the scale variable s: the higher s, the lower frequencies.

However, the perfect separation of the frequency components achieved by the Fourier transform has a downside in the time domain, which is dominated by uncertainty. In other words, given the Fourier transform of a time signal, whose frequency spectrum evolves in time, it is impossible to say which frequencies belong to which time.

In our problem, this limitation would prevent us from inferring which sensors generated the anomaly if any is detected. A possible way to reduce the limitation is to revert to the short Fourier transform. The idea is to run the Fourier transform in limited time windows to make sure that the spectral information can be localised into such window. However, the *Gabor limit* states that "One cannot simultaneously sharply localize a signal in both the time domain and frequency domain" [Gab46]. As a consequence, a shorter time window reduces the frequency resolution and vice versa.

Clearly, the wavelet transform cannot overcome the Gabor limit. Rather, it uses a different approach which gives higher importance to the time localisation of high frequency and lower importance to the time localisation of low frequencies. Since high frequencies correspond mainly to the singularities in the signal, i.e. transitory elements, the resolution error is made proportional to the duration of the component. As a consequence, each value of the original signal has an influence on a range of wavelet translations which is proportional to the scale of analysis. This effect is known as *Cone Of Influence*, since in a graph where the translations are on one axis and the scales on another axis, each point's influence area is cone-shaped.

A similar reasoning holds also for spatial signals, where the duration becomes the spatial extension. With the spatial wavelet transform, we are able to analyse the high scale information to detect the effects of events on the measurements, but we cannot localise an event with high

accuracy. For instance, we have an approximated location for the event's core, which may be, e.g., the epicentre of an earthquake. On the other hand, we can localise with high accuracy the low scale information, which is related to small subsets of sensors. So, for instance, we can well localise spikes in the measurements caused by faults, the edges of an event, singularities introduced by malicious sensors, or discontinuities between genuine and malicious sensors. We can conclude that wavelet transform is more suitable than Fourier transform in tackling malicious data injections. However, not all wavelet transforms are equivalent, especially in terms of applicability to the specific problem. Thus, we analyse the evolution of the wavelet transform theory, from the first-generation wavelet transform in the time domain, up to the continuous second-generation wavelet transform, which will be used for our purposes.

First-Generation Wavelet Transform The wavelet transform was first introduced for continuous-time signal.

Given a generic continuous-time signal $x(t)$, its wavelet transform is defined as:

$$C(s,\tau) = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} \psi(\frac{t-\tau}{s}) x(t) dt \tag{6.2}$$

Where s is the scale and τ is the translation. The function $\psi(\omega)$ is known as the *mother wavelet*, and in (6.2) is present in a dilated and translated form. Indeed, the scale factor s dilates the mother wavelet when its value increases, i.e. it makes the function $\psi(t)$ wider, and hence with lower frequency. The translation term τ shifts the mother wavelet in time: a positive τ shifts it to future time instants, while a negative τ shifts it to past time instants.

The wavelet transform corresponds to a convolution of the original signal with a dilated (because of s) and flipped version (in a proper convolution $\tau - t$ would be in place of $t - \tau$) of the mother wavelet. Therefore, the wavelet transform is equivalent to the repeated application of the same type of filter with varying bandwidth. The Fourier transform can also be interpreted as the filtering with varying frequency, but the filters have a bandwidth that tends to zero. Wavelets, on the contrary, have a range of frequencies whose modulus tends to zero as the frequency tends

to infinity. More precisely, the wavelet spectrum needs to satisfy the admissibility condition:

$$\int_{-\infty}^{+\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < +\infty \tag{6.3}$$

Which implies also that the wavelet is zero-mean in the time domain.

The wavelet transform can also be applied to uniformly sampled discrete time signals. In this case the transform is known as discrete-time wavelet transform. Since the sensors in a WSN perform a sampling of the physical phenomenon, the discrete-time wavelet transform will be assumed in the following.

If the discrete-time signal is modelled as a continuous-time signal convoluted with an impulse train $\delta(t - nT_s)$ (where T_s is the sampling period), the extension of 6.2 to uniformly sampled signals is straightforward and becomes:

$$C(a, m) = \frac{1}{\sqrt{a}} \sum_{n=0}^{M-1} \psi(\frac{n-m}{a}) x_n \tag{6.4}$$

Where M is the length of the signal, $x_n = x(nT_s)$, and a and m are the scale and translation (note that the translation has become and index, the effective temporal translation is mT_s).

Second-Generation Wavelet Transform When the signal under analysis is a non-uniformly sampled discrete signal, which is the case for sensor deployments in the spatial domain, the translation and dilation of a mother wavelet function is not a suitable approach. Furthermore, since the samples' locations are not known a-priori, it may be impossible to fully specify the transformation.

The second-generation wavelet transform can overcome these limitations, thanks to their approach of defining the wavelet filters *while* performing the wavelet transform. Originally, this approach was exploited to improve the implementation of the discrete wavelet transform and found its first realisation with the *Lifting scheme*.

The Lifting scheme is made of three steps: 1) **Deinterleave** the input digital stream into *even* and *odd* stream. 2) **Predict** the odd stream through the even stream. 3) **Update** the even sample through the prediction. This step would serve for the next prediction, if any.

The lifting scheme has several advantages compared to the classical implementation of wavelet transform (i.e., with filters designed in the frequency domain). However, the interest towards second-generation wavelet transform is mainly in its extension to non-uniformly sampled discrete-time signal. This is possible since the prediction step is compatible with techniques such as interpolation and regression which do not require the input signal to be uniformly-sampled.

In [JNS01], a nearest-neighbour prediction is used to realise a second-generation discrete wavelet transform for non-uniformly sampled discrete-time signal. The approach of making predictions based on a set of nearest neighbours makes the technique also applicable to signals with multi-dimensional domains.

The wavelet transform in [JNS01] is a discrete wavelet transform[1], since the scales and translations are fixed. For the goal of measurements inspection, however, it is important to analyse arbitrary scales, e.g. to isolate phenomena that occur at specific scales such as events, and arbitrary translations, e.g. to identify edges. Hence, a more suitable approach is a *second-generation continuous wavelet transform* (SGCWT), which was introduced in [KF06].

Second-Generation Continuous Wavelet Transform In Section 6.1 the Nadaraya-Watson estimator has been presented as a regression tool that adapts to scenarios where the measurements distribution can highly change, which is the case when events with variable pattern occur.

It is just this tool that forms the basis of the SGCWT, working similarly to the *predict* step in the lifting scheme. Nevertheless, the SGCWT, is not defined through the lifting scheme, which is intended for the discrete transform, but through a data-dependent wavelet basis function:

[1]Not to be confused with discrete-time wavelet transform. A discrete-time wavelet transform can be either discrete or continuous.

$$\Psi_{s,\tau}(t) = \frac{e^{-(\frac{t-\tau}{s})^2/2}}{\sum_{t\in\Omega_t} e^{-(\frac{t-\tau}{s})^2/2}} - \frac{e^{-(\frac{t-\tau}{\beta s})^2/2}}{\sum_{t\in\Omega_t} e^{-(\frac{t-\tau}{\beta s})^2/2}} \tag{6.5}$$

The result corresponds to a second-generation variant of the "Difference-of-Gaussians" *wavelet*, where s acquires the meaning of the wavelet *scale* and τ becomes the *translation* term.

The SGCWT is finally obtained with the formula:

$$(T^{wav}x)(s,\tau) = \frac{1}{h_\tau(s)} \sum_{t\in\Omega_t} [\Psi_{s,\tau}(t)x(t)] \tag{6.6}$$

where

$$h_{\boldsymbol{\tau}}(s) = \sqrt{\sum_{t\in\Omega_t} [\Psi_{s,\boldsymbol{\tau}}(t)]^2} \tag{6.7}$$

Compared to the first-generation CWT in 6.4, the SGCWT has an explicit dependence on the samples' locations in time, which compose the set Ω_t. Moreover, the result of the wavelet-based filtering is divided by $h_\tau(s)$ rather than by $\sqrt{s}$. Since $h_\tau(s)$ varies with each scale and translation, the SGCWT coincides with the application of a set of band-pass filters, which is divided by a signal depending on the set of scales and translations.

From this consideration, the asymptotic computational complexity can be quickly derived. If the number of scales and translations of interest are N_s and N_τ respectively, the computational complexity is that of applying N_s N_τ-long filters to N-long non-uniform time series. Tackling the problem with Non-Uniform Fast Fourier Transform, the asymptotic complexity for each desired scale can be as low as $\mathcal{O}(N_\tau log(N_\tau)) - Nln(\epsilon)$, where ϵ is the accuracy of the transform [DS99].

Transition to Space Domains The wavelet transform also has several applications in the spatial domain. The most widespread application is probably image processing and steganalysis [CS05; LF06; Sut+07], but more recently it has been applied in other fields, such as geogra-

phy [Bis14] and biology [KF06]. In the field of measurements analysis in WSNs, the wavelet transform has been previously applied only in the temporal domain and in its discrete form [SLM13]. Rather than detecting malicious data injections, the objective in [SLM13] is to build a fault-resistant event-detection algorithm, based on the observation that the wavelet coefficients at the lowest scale fluctuate more in the presence of events. Nevertheless, this technique would not prevent an attacker from injecting malicious data whilst staying undetected. Instead, we run the wavelet analysis in the spatial domain, that cannot be completely taken over with a limited number of malicious sensors. In the spatial domain, we learn the relationship between wavelet coefficients at different spatial scales, which characterises genuine events.

The equation of the mother wavelet, which in time corresponds to (6.5), in the spatial domain becomes:

$$\Psi_{s,\boldsymbol{\tau}} = \frac{e^{-(\frac{|\mathbf{v}-\boldsymbol{\tau}|}{s})^2/2}}{\sum_{\mathbf{u}\in\Omega} e^{-(\frac{|\mathbf{u}-\boldsymbol{\tau}|}{s})^2/2}} - \frac{e^{-(\frac{|\mathbf{v}-\boldsymbol{\tau}|}{\beta s})^2/2}}{\sum_{\mathbf{u}\in\Omega} e^{-(\frac{|\mathbf{u}-\boldsymbol{\tau}|}{\beta s})^2/2}} \tag{6.8}$$

And the equation of the wavelet transform (6.6) becomes:

$$(T^{wav}x_t)(s,\boldsymbol{\tau}) = \frac{1}{h_{\boldsymbol{\tau}}(s)} \sum_{\mathbf{v}\in\Omega} [\Psi_{s,\boldsymbol{\tau}}(\mathbf{v}) x_t(\mathbf{v})] \tag{6.9}$$

where

$$h_{\boldsymbol{\tau}}(s) = \sqrt{\sum_{\mathbf{u}\in\Omega} [\Psi_{s,\boldsymbol{\tau}}(\mathbf{u})]^2} \tag{6.10}$$

For $\beta > 1$, the second Gaussian is wider and more flat than the first, or in other words, the first emphasises more $\boldsymbol{\tau}$'s neighbourhood. Their subtraction emphasises changes in the measurements that happen around $\boldsymbol{\tau}$ and stabilise thereafter. The parameter β controls the wavelet's band-pass characteristics and is fixed to 1.87, i.e. the value that makes $1/s$ the dominant scale of analysis [KF06].

The term $h_{\boldsymbol{\tau}}(s)$ in (6.10) coincides with the standard deviation of the wavelet coefficient

$(T^{wav}x_t)(s, \boldsymbol{\tau})$ when x_t is *white Gaussian noise*, hence we refer to this term as *standardisation factor*. For convenience, we also introduce the non-standardised version of the wavelet transform, i.e.

$$(T^{wav}_{ns}x_t)(s, \boldsymbol{\tau}) = \sum_{\mathbf{v}\in\Omega}[\Psi_{s,\boldsymbol{\tau}}(x)x_t(\mathbf{v})] = (T^{wav}x_t)(s, \boldsymbol{\tau})h_{\boldsymbol{\tau}}(s) \tag{6.11}$$

This is to estimate a signal's energy at a certain scale. Indeed, division by the standardisation factor transforms a wavelet coefficient into a Signal to Noise Ratio (SNR), which is scale and translation independent, but cannot be used for estimating the signal's energy. The relationship between the low scale coefficients and the energy of the higher scale coefficients of the measurements signal is the cross-scale relationship that enables the detection of malicious data injections. Different signals, deployments and environments introduce different relationships, hence we learn this relationship with the method described in the next section.

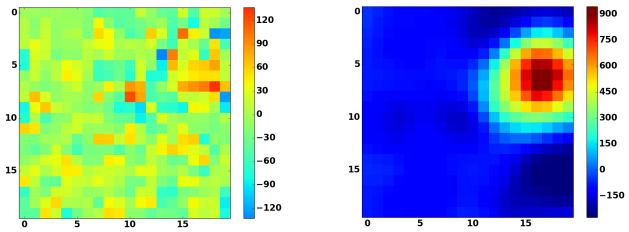
6.3 From Pairwise to Cross-Scale Measurements Correlation

The estimate given by the Nadaraya-Watson estimator is not reliable when malicious interference can occur, which can highly bias it. However, the wavelet transform defined above, uses such an estimator to extract the changes in the measurements, which may also be due to malicious interference, and at multiple granularities. *Our observation is that the attacker cannot introduce arbitrary changes in the measurements at two scales at the same time because there are some cross-scale relationships to respect.* In particular considering a low scale and a set of higher scales, the low scale coefficients, which capture the behaviour of the individual sensors or small groups of them, need to be acceptable when contextualised in the behaviour of larger sets of sensors, which is summarised by the higher scale coefficients.

The events' are the main variable that governs the higher scale coefficients, as the spatial propagation of the event induces a spatial trend in the measurements. Such a trend is best

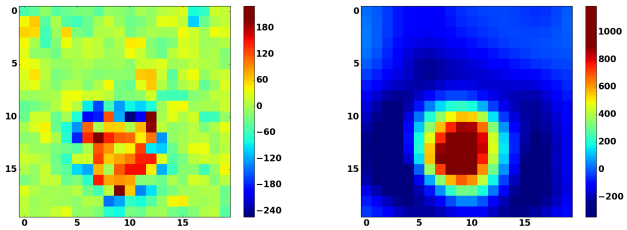
observable at scale s_e, the scale where $(T^{wav}x_t)(s, \boldsymbol{\tau}_e)$ is maximum, with $\boldsymbol{\tau}_e$ being the event's central location. The events' effect is also observable in transitions with small granularity, corresponding to a scale comparable to the average inter-sensor distance: s_l. For instance, in the case of wildfires, a large high scale coefficient will be observable in the area of the fire's breakout point, whereas a set of large low scale coefficients will be observed around the transitioning points, such as the boundaries of the fire. We observe that the relationship between coefficients at scales s_e and s_l is altered by malicious data injections, in consequence of the trade-off between two goals that a sophisticated attacker has: *1*) noticeably change the measurements and *2*) staying undetected.

If the attacker puts more effort into the first goal, the signal changes will be more abrupt and less supported by the malicious sensors' neighbours. The coefficients at low scale are thus large compared to higher scales. For example, the high scale coefficients of the spoofed event in Fig. 6.3 are comparable to those of the genuine event in Fig. 6.2, but the coefficients at low scale are larger in the malicious scenario, because there is a gap between genuine rest measurements and false event measurements.



(a) Coefficients at scale s_l (b) Coefficients at scale $4s_l$

Figure 6.2: CWT of a genuine event.

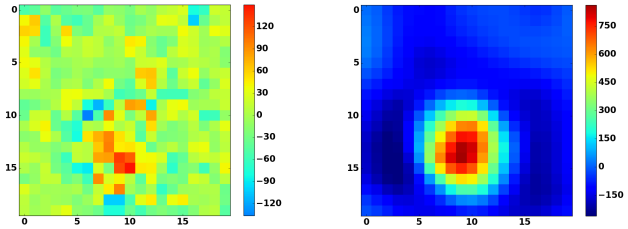


(a) Coefficients at scale s_l (b) Coefficients at scale $4s_l$

Figure 6.3: CWT coefficients of a spoofed event with larger low scale coefficients than the genuine event.

If the attacker puts more effort into the second goal, the gap between genuine and malicious

measurements is reduced, and so are the low scale coefficients in turn. However, as shown in Fig. 6.4, a spoofed event with low scale coefficients as small as the genuine event in Fig. 6.2, also has smaller high scale coefficients than the same genuine event.



(a) Coefficients at scale s_l (b) Coefficients at scale $4s_l$

Figure 6.4: CWT coefficients of a spoofed event with smaller high scale coefficients than the genuine event.

Ultimately, the chance of staying undetected needs to be traded with significant changes in the measurements if the cross-scale relationship is guaranteed.

6.4 Methodology

The methodology of the previous chapter was focused at making a set of application-specific choices which tailored the main algorithms to the specific context, e.g., calculation of the expected degree of correlation, choice of how the events affect correlations, adjustments in the anomaly detection test etc. In the new methodology, whose full flow chart is summarised in Fig. 6.5, it is the algorithms themselves which contextualise the information with the application, thus the initialisation part now only calculates the models that quantify the ranges of low scale coefficients that can be accepted for a given set of high scale coefficients. Moreover, the new methodology adds a group-wise analysis for characterisation, and the new diagnosis step. In the remainder of this section we analyse each step in detail.

The *first step* is the *detection* of anomalies in the measurements. We divide the measurements into groups, characterised by a common behaviour (e.g. caused by the same underlying event). Therefore, we compute for each group an *anomaly score* based on the deviation from the cross-scale relationship that was learnt. This step is based on application of the wavelet transform

and analysis of the relationship between the coefficients at different scales (Sect. 6.4.1).

Detecting an anomaly is generally not sufficient to identify the anomalous sensors. The *second step*, which we refer to as *characterisation*, identifies the anomalous sensors and determines if they have ELICITED or MASKED events. Identification of anomalous sensors is complex in the presence of collusion because malicious measurements can correlate well between themselves. Hence, we first identify groups of sensors with highly correlated measurements (Sect. 6.6.1). Then, measurement anomalies are translated into group anomaly scores, which are used to identify anomalous groups (Sect. 6.6.2). The general behaviour of the measurements in the anomalous group unveils if the anomalous measurements have elicited or masked events (Sect. 6.6.3). This, in turn, enables us to link back the group-wise anomaly to the individual sensors (Sect. 6.6.4).

The presence of anomalies does not imply the presence of malicious data injections since anomalies may be caused by other phenomena, such as genuine faults. For this reason, we further evaluate the properties of the measurements to distinguish, among other, faulty and maliciously compromised sensors. We refer to the task of ascribing the detected anomaly to a particular category as *diagnosis* (Sect. 6.7).

In the remaining of this chapter, we present the novel methods that deal with the tasks described above. We start from the preliminary analysis that enables the detection task, i.e. learning the cross-scale correlation models, then we move to the detection task itself, to the characterisation of compromised sensors, and finally to the diagnosis of faulty or malicious interference.

6.4.1 Learning The Events' Spatial Cross-Scale Relationships

To detect anomalous cross-scale relationships in the measurements, we learn their normal characteristics from historical data, which corresponds to learning the diffusion patterns of the underlying physical phenomenon. A possible alternative is to modelled the cross-scale relationships a priori, but this would be hindered by the effects of noise and environment, which are difficult to quantify. Moreover, the requirement of a model for each WSN application compli-

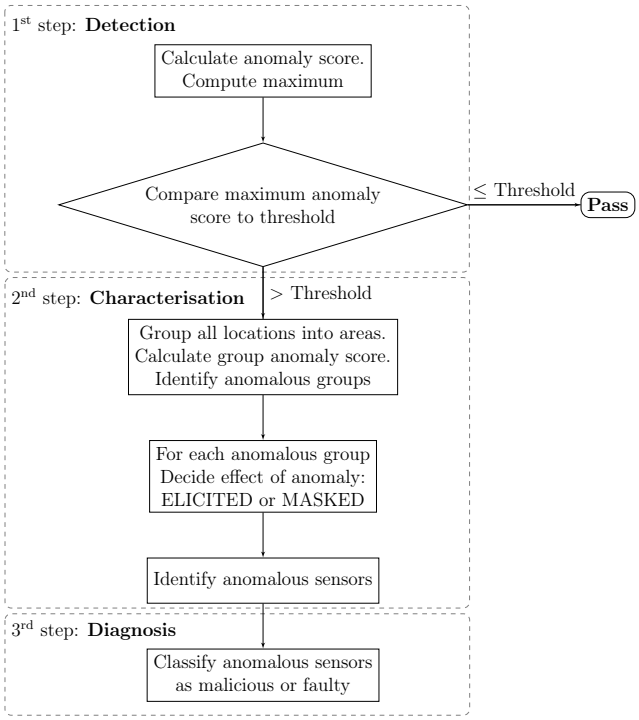


Figure 6.5: Methodology scheme.

cates the technique's deployment and introduces an additional point of failure: the accuracy of the model.

A requirement for learning the cross-scale relationship is that the historical data is a good representation of the measurement distribution. In particular the spatial frequencies that can be observed in the physical phenomenon need to be present in the historical data, especially under the presence of events. In some cases historical data may be difficult to observe, e.g. when events are particularly rare, so we may revert to simulations of the effect of events on the measurements to build a synthetic historical dataset. Moreover, the cross-scale relationship varies in time, under the most general scenario. In this case the learning phase may be addressed with incremental updates. However, a high confidence should be be guaranteed in the genuineness of the data used for the updates (e.g. by manually inspecting the data), otherwise an attacker may seek to bias the cross-scale correlation model to take advantage of it.

Since cross-scale relationships arise from spatial correlation, the historical data should consist of snapshots in time of the measurements reported by each sensor. Such snapshots should arise from different event scenarios, hence we split the measurements time series into time windows

where there are no events, one event, two events, etc. Afterwards, we take one representative time sample from that window, which is the sample where events are most noticeable if any, or a random sample if there is no event. We refer to a specific time snapshot selected with this criterion as *historical signal*. Historical signals are transformed through the CWT and the relationship between low scale and high scale coefficients is learnt with Algorithm 5.

Algorithm 5 Cross-scale relationship fitting

Require: historicalSignals, sensors
Ensure: xScaleModel
1: lst=[] {low scale term}
2: hst=[] {high scale term}
3: **for all** $\mathbf{X}_t \in$ historicalSignals **do**
4: $(T^{wav}\mathbf{X}_t) = \mathrm{CWT}(\mathbf{X}_t)$
5: **for all** $\boldsymbol{\tau} \in \mathcal{T}$ **do**
6: $E_{\boldsymbol{\tau}} = \sum_{s \in \mathcal{H}_s} \sum_{\boldsymbol{\tau}_{COI} \in COI_{s,\boldsymbol{\tau}}} [(T^{wav}_{ns}\mathbf{X}_t)(s, \boldsymbol{\tau}_{COI})]^2$
7: hst.append($E_{\boldsymbol{\tau}}$)
8: lst.append($(T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau})$)
9: **end for**
10: **end for**
11: maxLs=modelFit(hst,lst)
12: **return** maxLs

Each historical signal is transformed with the CWT (line 4), and the contribution of higher scale coefficients is calculated by the quantity $E_{\boldsymbol{\tau}}$ (line 6), reported below.

$$E_{\boldsymbol{\tau}} = \sum_{s \in \mathcal{H}_s} \sum_{\boldsymbol{\tau}_{COI} \in COI_{s,\boldsymbol{\tau}}} [(T^{wav}_{ns}\mathbf{X}_t)(s, \boldsymbol{\tau}_{COI})]^2 \tag{6.12}$$

We refer to such term as *energy index*, since it is related to the energy of the signal after isolating it in both space and frequency.

Spatial isolation is achieved by considering a set of spatial locations $\boldsymbol{\tau} \in \mathcal{T}$, where $\mathcal{T}$ is a multi-dimensional grid. The distance between two consecutive translations equals the average distance between a sensor and its closest neighbour, in order to approximate the non-uniformly spaced WSN with its closest uniformly spaced correspondent. The choice of a grid for the translations allows us to make the energy index spatially consistent. Indeed, the energy index is calculated at line 6 as the sum of the contributions at the higher scale within the wavelet coefficient's

Cone Of Influence (COI), defined as the area with a distance from $\boldsymbol{\tau}$ less than the one where the boundary effects of the wavelet basis function start to be noticeable. Specifically for the wavelet transform defined in (6.9), the COI is centered in $\boldsymbol{\tau}$ and has a radius of approximately s spatial units [KF06].

Frequency isolation of the energy index is achieved by summing only a set of higher scales $\mathcal{H}_s$, which are higher than s_l. The value chosen for s_l is half of the average inter-sensor distance, corresponding to the highest observable frequency in the space of translations $\mathcal{T}$. The higher scales, instead, are selected to have little overlap with the low scale s_l, i.e. the function in (6.8) for $s = s_l$ should have a frequency spectrum which is approximately disjoint from that of the functions with $s \in \mathcal{H}_s$.

The function $maxLs$ is learnt with a model fitting technique. We divide the energy indices into consecutive intervals, filter out obvious outliers, and return the maximum among the low scale coefficients in such interval. The intervals are individuated with a simple and effective criterion, which sorts the data by ascending energy index, and iteratively produces a new interval as soon as a new maximum is found. Note that different models are learnt for positive and negative low scale coefficients, since they generally have different distributions as the wavelet basis function is not symmetric with respect to the origin. With the models available, the measurements can be tested for anomalies, as explained in the next section.

6.5 Detection

The models calculated with the steps described enable the comparison of low scale coefficients with high scale coefficients. We define the ratio between a low scale coefficient and its maximum acceptable value as *anomaly score* (A), calculated as:

$$A(\boldsymbol{\tau}) = |(T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau})|/maxLs(E) \tag{6.13}$$

Where $\boldsymbol{\tau}$ is the location of sensor x. Algorithm 6.13 shows that the model relating low scale

and high scale coefficients needs to return the maximum acceptable low scale coefficient, given the input high scale coefficients' energy. The final step of detection consists in checking:

$$\mathrm{MAX}_{\boldsymbol{\tau}\in\mathcal{T}} A(\boldsymbol{\tau}) > T_d \tag{6.14}$$

If this condition holds, anomalies are present. The threshold T_d should be set according to the acceptable false positives. If the measurements used to learn the cross-scale relationship are a good representation of the whole measurements distribution, $T_d = 1$ ensures a false positive rate equal to zero (if outliers were filtered out during model fitting, it will be equal to the outlier rate). The real false positive rate may be higher if the dataset does not represent the data distribution well. In this case, a reliable false positive rate can be estimated empirically as shown in Sect. 7.2.5 to select the threshold that gives the most appropriate TPR/FPR combination. Such procedure may be repeated when new data, that does not comply with the current data distribution, is collected. At this point, the anomalous sensors need to be identified through the characterisation step shown in Sect. 6.6.

6.5.1 Computational complexity

In WSNs, the computational complexity is important as the sensor nodes have low computational capabilities and highly demanding calculations cause battery depletion.

Nevertheless, the algorithm that we introduced may be run from the system that analyses the measurements and takes appropriate reactions, e.g. the base station or a remote server. Such devices generally have less computational constraints as they are in charge of complex tasks.

Beyond the computational burden, reacting promptly to malicious data is also important. Note that the detection test in (6.14) is always run, while the characterisation and diagnosis comes into play only when detection triggers. Hence, we report below the analysis of the asymptotic computational complexity for the detection step:

- Merging the measurements from N sensors in a time window with size W: $\mathcal{O}(WN)$

- Calculating the wavelet coefficients at N_s scales and N_τ translations: $\mathcal{O}(SN_\tau log(N_\tau) - SNln^2(\epsilon))$, where ϵ is the desired accuracy for the wavelet coefficients [DS99].
- Estimating the energy of the high scale coefficients: $\mathcal{O}(SN_\tau \log(N))$, i.e. the complexity of a convolution of the coefficients at a specific scale and the COI mask.
- Calculating the maximum admissible low scale coefficients: $\mathcal{O}(N_\tau)$
- Testing the admissibility of each low scale coefficient: $\mathcal{O}(N_\tau)$

The total cost of detection is then $\mathcal{O}(WN + SN_\tau log(N_\tau) - SNln^2(\epsilon))$.

6.6 Characterisation of Anomalies: Effects and Responsible Sensors

When detection triggers, we know that there are anomalies in the measurements but, at this stage, the responsible sensors are unknown. Without an accurate analysis, it is likely that we will confuse malicious sensors with genuine sensors and vice versa. For instance, if a set of malicious sensors collude and surround a genuine sensor, they can make it look responsible for the anomaly introduced.

Moreover, the points where the cross-scale relationship is not respected do not necessarily characterise the area where the malicious sensors lie. Indeed, these points testify only of a disruption in the cross-scale relationship, but either the low scale, or the high scale coefficients, or both, may have been altered. Tracing back the anomalous measurements requires further analyses, which constitute the characterisation phase.

Our approach to this problem is to make a group-wise analysis of conflicts between sensors, where sensors belong to the same group if their measurements are correlated. In this way, a genuine sensor is not judged individually and making it look responsible for the anomaly is harder for colluding sensors. The first requirement is thus to identify groups of sensors with correlated measurements, and then move from sensor-wise to group-wise anomaly analysis.

6.6.1 Sensor Grouping

When malicious measurements are correlated because of collusion, an anomaly may be evident only where malicious sensors are close to genuine ones. However, when a sensor's measurement disrupts correlations, the responsibility is shared among the sensors with the same behaviour, which is defined with respect to the events being sensed.

Events can be identified thanks to the wavelet coefficients: wherever there is an event, there is a peak in the wavelet coefficients at scale s_e and translation $\boldsymbol{\tau}$, which respectively characterise the spread of the event, and its centre (akin to the centre of mass and not necessarily its geometrical centre). Algorithm 6 identifies events peaks with a *ridge lines*-based peak detection algorithm (lines 1-3). It is a variant of the algorithm described in [DKL06], which in addition considers 2-dimensional spaces and extracts the peak scale, defined as the scale with the maximum coefficient in the ridge line. At such a scale the main characteristics of the event can be captured, for instance if it has an increasing or decreasing trend.

Ridge lines are sets of n-dimensional points that are local maxima with respect to at least one dimension. In [DKL06] the dimensions considered are the wavelet scale and translation and ridge lines connect local maxima translations across consecutive scales. Local maxima are, by nature, not well defined since they depend on the degree of "locality". In our case, the smallest degree of locality is the distance between neighbouring sensors. To consider also higher degrees of locality, at line 7 we merge together two of such maxima that are within a certain distance (max_d) and instead assign the points to different ridge lines if such distance is overcome.

If a point is associated with a long ridge line, i.e. it is a local maximum at many scales, it is more likely to be a true peak compared with a point within a short ridge line. Indeed, being a local maximum with respect to more scales indicates that the increase around that point is not transitory. For this reason, ridge lines shorter than min_l are filtered out by the function *filterRidgeLines* at line 14.

The parameters max_d and min_l respectively represent the maximum distance between local maxima and the minimum length of ridge lines. There are well known heuristics for choosing

Algorithm 6 getEventsPeaks

Require: $(T^{wav}\mathbf{X}_t)(s, \boldsymbol{\tau})$, max_d
Ensure: eventsPeaks
1: **for all** $s \in s_l, ..., s_m$ **do** {s_m is the maximum distance between two sensors}
2: relExtrema[s]={ $\boldsymbol{\tau} : (T^{wav}\mathbf{X}_t)(s, \boldsymbol{\tau}) > (T^{wav}\mathbf{X}_t)(s, \boldsymbol{\tau}_i)$ $\forall \boldsymbol{\tau}_i : ||\boldsymbol{\tau} - \boldsymbol{\tau}_i|| = \min_j(||\boldsymbol{\tau}_j - \boldsymbol{\tau}_i||)$ }
3: **end for**
ridgeLines=relExtrema[s_{max}]
4: **for all** $s \in s_m - 1, ..., s_l$ **do**
5: **for all** rE $\in$ relExtrema[s] **do**
6: closestRidgeLine=findClosest(ridgeLines, rE)
7: **if** distance(closestRidgeLine,rE)$< max_d$ **then**
8: closestRidgeLine.append(rE)
9: **else**
10: ridgeLines.append(rE)
11: **end if**
12: **end for**
13: **end for**
14: filteredRidgeLines = filterRidgeLines(ridgeLines,min_l)
15: **for all** rL $\in$ filteredRidgeLines **do**
16: peakIndex = argmax(rL)
17: eventPeak.s=s(rL(peakIndex))
18: eventPeak.loc=$\boldsymbol{\tau}$(rL(peakIndex))
19: eventsPeaks.append(eventPeak)
20: **end for**
21: **return** eventsPeaks

such parameters [DKL06], which have proven to give satisfactory results in our experiments.

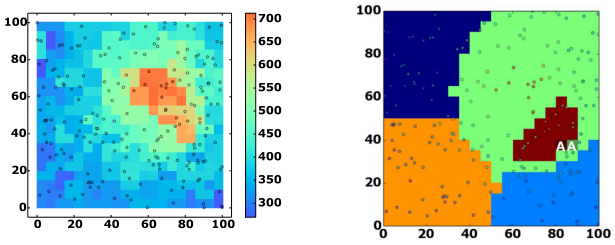
After detecting peaks, each sensor is assigned to a group with Algorithm 7. For each identified peak, a sensor is assigned to the closest group where the sign of the coefficients at the peak and sensor's location coincide (line 8). Indeed, if the sign is discordant, the sensor's measurements do not follow the event trend. Finally, the groups are labelled (line 15): in this phase, contiguous sets of sensors belonging to the same group are given a common label. An example of the result of the grouping procedure is shown in Fig. 6.6.

6.6.2 Identification of Conflicts Between Groups

The locations where the anomaly score is high indicate the presence of a conflict between sensors in that area. A conflict arises when the measurements from different sensors originate from

Algorithm 7 getSensorsGroups

Require: $(T^{wav}\mathbf{X}_t)(s, \boldsymbol{\tau})$, eventsPeaks
Ensure: sensorsGroups
1: GroupsMap($\boldsymbol{\tau}$) = 0 $\forall \boldsymbol{\tau} \in \mathcal{T}$
2: **for all** $\boldsymbol{\tau} : \boldsymbol{\tau} \in \mathcal{T}$ **do**
3: sortedEPindices=argsort(distance(eventsPeaks,$\boldsymbol{\tau}$))
4: grouped = False
5: peakIndex=1
6: **while** not(grouped) and peakIndex $\leq$ length(eventsPeaks) **do**
7: eP=eventsPeaks(sortedEPindices)(peakIndex)
8: **if** $(T^{wav}\mathbf{X}_t)(\text{eP.s}, \text{eP.loc}) \cdot (T^{wav}\mathbf{X}_t)(\text{eP.s}, \boldsymbol{\tau}) > 0$ **then**
9: GroupsMap[$\boldsymbol{\tau}$]=peakIndex
10: grouped=True
11: **end if**
12: peakIndex= peakIndex+1
13: **end while**
14: **end for**
15: sensorsGroups=label(GroupsMap)
16: **return** sensorsGroups



(a) Sensed Temperatures (b) Grouping Result

Figure 6.6: Grouping Effect Example. Sensors are sorted into five groups; in particular the green group contains a genuine event, while the brown group a spoofed event. A white "A" represents the condition A($\boldsymbol{\tau}$)> T_c.

incompatible factors, for instance a real event and a spoofed event.

The remaining task for characterisation is to identify conflicting groups, and thereby the sensors that caused the conflicts. Since high anomaly scores indicate the presence of disruptions in correlation, a conflict is registered for the two areas that are the closest to each anomalous location. This step is summarised in lines 2-9 of Algorithm 8.

During characterisation, the anomaly score is compared to the threshold T_c, which differs from the detection threshold T_d to find the trade off between the costs of acquiring malicious data and of losing genuine sensors. The value of T_c should decrease with the former and increase

Algorithm 8 findAnomalousGroups

Require: groups, $A(x)$,T_c
Ensure: Ag {Anomalous groups}
1: Conflicts(g1,g2) = 0 $\forall$ g1,g2 $\in$ groups {Initialise group conflicts}
2: **for all** $\boldsymbol{\tau} : \boldsymbol{\tau} \in$ sensors **do**
3: **if** A($\boldsymbol{\tau}$)> T_c **then**
4: g1= argmin$_{\text{g}\in\text{groups}}$(dist($\boldsymbol{\tau}$, g))
5: g2= argmin$_{\text{g}\in\text{groups},\text{g}\neq\text{g1}}$(dist($\boldsymbol{\tau}$, g))
6: Conflicts(g1,g2) = Conflicts(g1,g2)+1
7: Conflicts(g2,g1) = Conflicts(g2,g1)+1
8: **end if**
9: **end for**
10: Ag={}
11: GA(g)=0 $\forall g \in$ groups {Initialise group anomaly score}
12: **while** $\sum_{g_i\in\text{groups}} \sum_{g_j\in\text{groups}} \text{Conflicts}(g_i, g_j) > 0$ **do**
13: **for all** $g_i \in$ groups **do**
14: GA(g_i) = $\sum_{g_j\in\text{groups}} |g_j|\text{Conflicts}(g_i, g_j)$
15: **end for**
16: ag = g: GA(g) = MAX(GA)
17: Ag.append(ag)
18: Conflicts(ag,g) = Conflicts(g,ag) = 0 $\forall g \in$ groups
19: **end while**
20: **return** Ag

with the latter and its choice, like for T_d, can be done experimentally as shown in section Sect. 7.2.5. In Figure 6.6b, the locations where A($\boldsymbol{\tau}$)> T_c are marked with an "A". Since in both cases the closest groups are the green and the brown group, two conflicts are registered between these two groups.

The next step, corresponding to lines 13-15, makes a transition from location-wise to group-wise anomalies. Each group is assigned an anomaly score equal to the sum of conflicts multiplied by the size of the conflicting groups. The sum reflects the confidence increase in the incompatibility between the two groups when more conflicts are observed. Instead, the multiplication by the conflicting group size ensures the selection of the most likely scenario, under the assumption that the prior probability of having C anomalous sensors decreases as C increases. This is a consequence of the increase in attack's cost with the number of sensors to compromise. In the example of Fig. 6.6b, the brown group would correctly be blamed for the anomalies since it has fewer sensors than the green group with which is conflicting.

Finally, the group with the maximum anomaly score is added to the collection of anomalous groups and the procedure is reiterated until all conflicts disappear. Indeed, the malicious sensors may be divided into several groups, since they could act according to different behaviours.

6.6.3 Identification of the Effects of Anomalies

Having collected the anomalous groups, the first characterisation task can be completed, i.e. identification of the effects of anomalies. We characterise each anomalous group as an eliciting or a masking group if the measurements have been likely increased or decreased respectively.

This information can be extracted through the wavelet transform. Indeed, the transformed coefficients are positive or negative corresponding to increases and decreases of the measurements values respectively. Note that this information is scale-specific. So, for instance, transitory decreases within general increases produce negative coefficients at lower scales and positive coefficients at higher scales.

Since the sign of the local maximum associated with the group indicates the generic trend of the sensors within the group, the measurements were elicited or masked if the sign of the local maximum is positive or negative respectively.

6.6.4 Identification of anomalous sensors

The last step of characterisation is moving from anomalous groups to anomalous sensors with the steps described in Algorithm 9. This consists of a filtering operation that keeps only the group locations where the low scale coefficients are positive/negative if the group was marked as ELICITED, or MASKED respectively.

Indeed, given an anomalous group of sensors that is marked as ELICITED, the sensors that are most likely responsible for the anomaly are those whose measurements increase is more remarkable, i.e. those where the low scale coefficients are positive and high. Instead, the sensors where the low scale coefficients are negative and low may be genuine sensors that at

Algorithm 9 characteriseAnomalousSensors

Require: Ag
Ensure: As
1: $As = \{\}$
2: **for all** $ag \in Ag$ **do**
3: **for all** $\boldsymbol{\tau} \in ag$ **do**
4: **if** group.effect = ELICITED **then**
5: $As_g = \{\boldsymbol{\tau} : \boldsymbol{\tau} \in ag, (T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau}) > 0\}$
6: **else** {group.effect = MASKED}
7: $As_g = \{\boldsymbol{\tau} : \boldsymbol{\tau} \in ag, (T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau}) < 0\}$
8: **end if**
9: $As = As \cup As_g$
10: **end for**
11: **end for**

higher scale appear as eliciting just because of the effect of their anomalous neighbours. A dual reasoning holds when a group is marked as MASKED.

6.7 Diagnosis of Anomalies: Faulty or Malicious?

After the characterisation step, each sensor is classified as anomalous or normal. However, anomalies could be explained by either: 1. *Single-point failures* 2. *Common-mode failures* 3. *Malicious data injections.*

Since the characteristics of malicious data injections are arbitrary and cannot be modelled, the means by which anomalies can be linked to malicious interference is by exclusion: the characteristics of both single-point failures and common-mode failures are modelled and inspected as described below. If they match the data being analysed, then a diagnosis of single fault or group fault is inferred.

If both can be excluded, then we are most likely in the case of malicious interference. In the following, we describe the procedures to diagnose faulty behaviours.

6.7.1 Single faults

By single faults, we mean the presence of genuine alterations in the measurements that trigger a single-point failure, i.e. impair the measurements of one particular sensor.

Single-point failures are characterised by large low-scale coefficients, since the measurements of a faulty sensor and its neighbours broadly differ. However, just as for malicious data, this is not matched by adequate high scale trends. Differently from colluding malicious measurements[2], faulty measurements are independent from those of other sensors. For instance, in the presence of an event, the measurements of a faulty sensor may not be affected.

To detect measurements arising from a different distribution, we use the median absolute deviation (MAD) outlier detection, which is known as an outlier-resistant outlier detection technique [AH15], since the centre of the distribution is estimated with the median (which is not affected by extreme values) and the variation of the distribution is estimated with the median distance from the median (which is not affected by extreme variations). The single fault diagnosis procedure is summarised in Algorithm 10.

The method is based on the *modified z-score* [Cro94]: a robust variant of the z-score, which instead quantifies the distance from the mean, normalised by the standard deviation. Similarly, the modified z-score identifies as outliers the samples with a low probability of occurrence. The threshold T_{sf} regulates the cutoff probability. In [Cro94], the suggested threshold is 3.5, but a higher value may be preferred to reduce the false positives further.

Single faults are identified when the threshold is crossed. Afterwards, the characterisation phase should be repeated to check the presence of also non-faulty anomalies and cope with the case where faulty sensors coexist with malicious sensors in the same group.

[2]Note that non-colluding malicious data injections may show such characteristics. In this case, single faults require more specific analyses, such as fault statistics or fault characterisation (as described in Sect 6.8).

Algorithm 10 diagnoseSingleFaults

Require: group, T_{sf}
Ensure: singleFaultsDiagnosis
$lc = (T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau})$
for all $x \in$ group.sensorsLocations **do**
  if $A(\boldsymbol{\tau}) > T_d$ **then**
    $lc_{\boldsymbol{\tau}}$=$(T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau})$
    $lc_{COI(\boldsymbol{\tau})}$= $(T^{wav}\mathbf{X}_t)(s_l, \boldsymbol{\tau}_y) \forall \boldsymbol{\tau}_y \in COI(s_e, \boldsymbol{\tau})$
    $\widetilde{lc}_{COI(\boldsymbol{\tau})}$=median($lc_{COI(\boldsymbol{\tau})}$)
    $\Delta lc_{COI(\boldsymbol{\tau})}$=$|lc_{COI(\boldsymbol{\tau})} - \widetilde{lc}_{COI(\boldsymbol{\tau})}|$
    $MAD(lc_{COI(\boldsymbol{\tau})})$=median($\Delta lc_{COI(\boldsymbol{\tau})}$)
    $\tilde{z}(\boldsymbol{\tau})$=$0.6745\frac{\Delta lc_{COI(\boldsymbol{\tau})}}{MAD(lc_{COI(\boldsymbol{\tau})})}$ {The factor 0.6745 converts from MAD to standard deviation when the data are normally distributed.}
    if $\tilde{z}(\boldsymbol{\tau}) > T_{sf}$ **then**
      return True
    end if
  end if
end for
return False

6.7.2 Group Faults

Group faults include alterations of the measurements that originate from non-malicious faults, involve more than one sensor, and introduce a *common-mode failure*, i.e. the fault's effect is similar among all the sensors. The key characteristic for diagnosing common mode failures relies just in this observation: the anomalous sensors produce similar measurements, without the typical gradual changes of real events, and of eliciting and masking attempts in turn.

Algorithm 11 summarises the diagnosis of group faults. The similarity of anomalous measurements is evaluated with a statistical operator, known as *coefficient of variation* (CV), which is defined as standard deviation divided by sample mean. The act of normalising by the mean enables us to compute a relative variability, that is not affected by the samples' magnitude.

Algorithm 11 diagnoseGroupFaults

Require: As_g, $S(\boldsymbol{\tau})\boldsymbol{\tau} \in As_g$, T_{gf}
Ensure: groupFaultDiagnosis
$\hat{c_v} = \frac{\text{std}(S(\boldsymbol{\tau})\boldsymbol{\tau} \in As_g)}{\text{mean}(S(\boldsymbol{\tau})\boldsymbol{\tau} \in As_g)}$
return $|\hat{c_v}| < T_{gf}$

The CV is compared to a threshold T_{gf} which should be set according to the model of common

mode failures. For instance, if the measurements of sensors subject to common mode failures are expected to differ only by noise, the threshold should be equal to the relative variation brought in by noise, i.e. the noise to signal ratio (inverse of SNR).

6.8 Conclusions

In this chapter, we have focused on detecting malicious data injections in WSNs when one or more events can occur, causing a loss in correlation that is exploited by colluding compromised sensors. We have proposed a novel methodology to detect malicious data injections, based on the measurements cross-scale relationship analysed through the wavelet transform.

Since detecting anomalies in the measurements is not sufficient to effectively counteract them, we have also provided an approach to characterise malicious colluding nodes, by partitioning the sensor nodes based on the correlation between their measurements. This approach considers the effects of events, hence it is able to detect groups of sensors that elicit or mask events. Finally, we provided a novel measurements-based diagnosis technique to distinguish fault-induced anomalies from malicious anomalies. Indeed, genuine faults may also introduce anomalies, as the measurements from faulty sensors do not correlate with those of healthy ones. This may lead to the wrong conclusion that there was an attack, but by classifying the main characteristics of genuine faults we were able to infer when the anomaly is most likely malicious.

To validate the effectiveness of the new techniques we will follow systematic approaches which are the state-of-the-art in the field of adversarial machine learning. Indeed, rather than constructing by hand ad-hoc attacks, which are not necessarily sophisticated and so cannot be used as a reliable indicator for the system's robustness, we build automatic evaluations that seek to maximise the attacker's exploitation of its resources towards the goals of subverting event detection and staying undetected. In the next chapter we introduce the field of adversarial machine learning and two families of attacks belonging to this field, i.e. mimicry and evasion attacks. Then, we evaluate the techniques presented in this chapter with both approaches and obtain information about the security level offered, and how this is influenced by

the application, environment, and deployment.

Chapter 7

Robustness of Measurements Inspection

Detection of malicious data injections in Wireless Sensor Networks (WSNs) is difficult to validate since they are not common yet or, at least, they are not easy to retrieve. Even if they were common and easy to gather, malicious data can be injected in several different ways, with different degrees of sophistication and different results on the detection algorithm.

For these reasons, most of work available in literature is tested against simulated attacks [Sun+13; LCC07; Rez+13; Ata+08; Ban+10; OHC12; Bao+12; LC13] and *injection of attacks* [TH06; CP07], which aim to reproduce some of the expected characteristics of the attack, e.g. bias or high variance in the measurements. There are two main problems with this kind of approach:

- Simulations define the attack steps a-priori. However, the attacker is an *intelligent agent*, i.e. can observe parts of the WSN system, which may include the measurements inspection algorithm, and adapt his strategy accordingly.
- Simulations define the attack steps probabilistically, i.e. having defined the steps to build an attack, the specific attack is chosen at random. Instead the attacker generally seeks to optimise the attack among all the feasible ones.

In summary, simulations produce attacks that are not significant as they do not model the attacker as an active entity that seeks to stay undetected, and do not optimise the attack

from the attacker's perspective. These problems have been dealt with in *adversarial machine learning*, which is briefly introduced below. After that, in this chapter we will evaluate the wavelet-based algorithms using adversarial machine learning methods. In particular, we will consider two different approaches which give the attacker a different trade-off between staying undetected and causing damage to the system. We will refer to these approaches as *mimicry* and *worst-case* attacks, which will be designed, implemented and used to quantitatively and qualitatively estimate the reliability of measurements inspection against malicious data injections.

7.1 Adversarial Machine Learning and Exploratory Attacks

The emerging field of adversarial machine learning is concerned with the robustness and security properties of machine learning algorithms and anomaly detection techniques. As described in [Hua+11], according to the influence of the attacker, we can classify the attacks against machine learning systems as *causative* and *exploratory*. In the first case, the attacker can modify the behaviour of the system by injecting malicious points. In exploratory attacks, the attacker cannot alter the behaviour of the system but attempts to circumvent the learning algorithm by exploiting weaknesses or blind spots that allow him to craft malicious samples evading detection.

In this chapter we aim to evaluate the approach described in the previous chapter with exploratory attacks. The reason for this choice is that the attacker's influence on the learning phases of the measurements inspection can be prevented by using only data that has been thoroughly validated, e.g. with manual inspection. As we will see in the next sections, with the wavelet-based algorithm it is sufficient to collect a few measurements snapshots under different scenarios. In particular, Section 7.2.5 shows that with the order of 10 training snapshots the algorithm is already operational, and with the order of 100 snapshots the results are highly reliable.

In the context of security, exploratory attacks have been extensively studied for Intrusion

Detection Systems (IDS) [FL06; GR06; MC03] and spam filtering applications [LM05a; WW04; LS07; SWB06]. Despite the differences between the two applications, the approaches to evade the anomaly detector rely on the same principle: exploiting the flaws in the selection of the features used by the system by crafting malicious samples that add or delete features or by encrypting malicious functionality so that the system does not recognize them as malicious features (e.g. by substituting *viagra* by *v1agra* to craft a spam email).

In addition to distinguishing causative and exploratory attacks, Huang et al. also distinguish targeted from indiscriminate attacks, i.e. attacks that seek to introduce an attack vector with specific characteristics, or from a generic class respectively [Hua+11]. It is at this point that our approach necessarily diverges from classical adversarial machine learning and extends it. Indeed, even if we consider attacks that are indiscriminate with respect to the measurements inspection algorithm, i.e. it is sufficient to stay undetected by any means, there are some application-specific requirements, i.e. the event detection task needs to be impaired.

As staying undetected and impairing event detection are two conflicting goals, we identify two different kinds of exploratory attacks: *mimicry* and *evasion* attacks. These prioritise respectively the event impairment and being undetected. Each approach demands different resources from the attacker and the results offer different kinds of information, as detailed below.

The goal of mimicry attacks is to consider only significant impairments to the event detection task, and then minimise the chance of being detected with a *best-effort* strategy, i.e. the attacker will leverage all the resources in his control to avoid the detection from the measurements inspection algorithm. This family of attacks is compatible with an attacker who does not know, or has partial knowledge about the measurements inspection algorithm. Instead, he knows how to cause damage to the event detection task, i.e. he knows, or has an estimate, of the effect of the attack on the WSN system. Mimicry attacks consist of replaying data that has been observed in the past, possibly after processing it to adapt it to the current context. Studying the reaction of the system to mimicry attacks gives a measure of performance of the measurements inspection algorithm. In particular, the results are good if the inter-measurements correlations

are correctly captured, which is the only discriminant between genuine data collected in the past and malicious data replayed in the future.

Evasion attacks operate from a dual perspective: they constrain the malicious data to pass the measurements inspection detection, and optimise the malicious input. In this case, the attacker needs to know the measurements inspection algorithm. The optimisation of the malicious data can be intended with respect to both the attacker's resources and the damage to the event detection. Indeed, different eliciting or masking attacks cause different damage. However the degree of damage connected with the spoofed and masked event is difficult to estimate and is subject to arbitrary considerations. So, in this thesis, we constrain evasion attacks to make a successful spoofing or masking, independently from their nature, and minimise the resources needed by the attacker. This analysis evaluates the degree of resilience given by the measurements inspection algorithm as a function of the application and deployment configuration.

In the next section, we analyse in detail what is needed to implement mimicry and evasion attacks. Then, we design the attacks accordingly and implement automatic attack generation tools. Finally, we analyse the result and obtain a reliable robustness evaluation for both the wavelet-based measurements inspection algorithm and other state-of-the-art techniques for comparison.

7.2 Robustness of Measurements Inspection Against Mimicry Attacks

By mimicking data previously classified as genuine, mimicry attacks aim at threatening the very structure of the anomaly detection algorithm. Clearly, not all mimicry attacks are threatening, even if they keep the attack undetected. For instance, keeping the measurements under control of the attacker unchanged is an edge case of mimicry attack: it stays undetected, unless there is a false positive, but the measurements are not changed at all. Hence, when constructing

meaningful mimicry attacks, it is necessary also to consider the relative risk in terms of potential damage to the system.

In fact, it is likely that some data that was classified as genuine in the past, may cause significant damage in the future, because of a change of context. In event detection WSNs, the most relevant change of context occurs with the manifestation of events. Thus, injecting event measurements (i.e. measurements that trigger event detection) in a rest condition and injecting rest measurements (i.e. that do not trigger event detection) when an event is present are the most threatening scenarios. However, as discussed in the previous chapter, other attacks are possible such as eliciting an already present event, or partially masking an event to split it into more events. Although the degree of damage of similar attacks may be lower, they are still worthy of consideration since they could be easier to achieve. In summary, an attacker may exploit the mimicry approach to get a wide range of possible attack vectors, which are more or less difficult to contextualise in the current scenarios.

Since this approach does not imply knowledge about the anomaly detection algorithm, it is generally not known in advance whether the constructed attack will be detected; however, the chance of staying undetected will likely increase when reducing the context mismatch between mimicked and current data. On the other hand, mimicry attacks are generic, i.e. valid for different algorithms. Furthermore, since they are not tied to the vulnerabilities of the specific algorithms, they can also be used as a fair benchmark tool.

To implement mimicry attacks, there are different problems to solve, which define how the attacker may exploit his resources. Consistently with the attacker model introduced in Chapter 4, we assume that the attacker controls the measurements from a subset of sensors of his choice. So, the choice of such sensors needs to be optimised. We also need to define how the data that will be mimicked is selected, how it is adapted to the current context, and how the malicious sensors coordinate themselves in the injection of the malicious data. These problems are covered in the following sections. In addition, we provide an implementation of a test suite for the automatic generation of mimicry attacks, apply it on WSN measurements, and conduct tests and benchmarks on the wavelet-based algorithm.

7.2.1 Collusion and Sensor Targeting Strategies

One important attacker's resource, which makes a significant difference in the ability to stay undetected, is the capacity of coordinating the compromised sensors in the injection of malicious measurements, which has been denoted with collusion. Without collusion, the resulting measurements would likely appear anomalous. For example, if malicious seismic sensors are trying to spoof an earthquake, they need to agree about the kind of earthquake that is being mimicked, otherwise they could produce remarkably different information about the earthquake's characteristics, e.g. epicentre and magnitude, and an anomaly may be be detected.

Moreover, since some malicious sensors are expected to share higher correlations with the genuine sensors (e.g., they are closer), their injected measurements can diverge less from the genuine counterpart. If this information is shared among all malicious sensors, then they could all collaborate in keeping correlations consistent. Otherwise, if a malicious sensor needs to solve the problem locally, e.g. keep correlations consistent with neighbouring sensors, even the expected correlations between malicious sensors might not be respected.

The generic collusion strategy that we seek to implement consists of:

1. Making malicious measurements agree about the scenario to mimic.

2. Use some malicious measurements to mediate with genuine measurements.

Once the mimicked scenario has been agreed on, the malicious sensors will try to replicate it. This task needs to be addressed with a subset of sensors. Since different subsets have different expected inter-sensor correlations, a relevant impact on the chances of avoiding detection is involved. Hence, we propose a sensor targeting strategy that gives priority to the sensors with high expected correlations between themselves, and with low expected correlations with genuine sensors. In practice, when genuine sensors have low expected correlations with malicious ones it means that they do not have enough shared information to state that the scenarios depicted by the malicious sensors diverge from reality.

To transform the strategies discussed above into real attacks, there is the need to design and implement an automatic tool, which we refer to as a *test suite*. In the next section we give an outline of the challenges this tools is addressing and present its high-level architecture.

7.2.2 Test Suite Design

Essentially, mimicry attacks involve two problems: the selection of the data to mimic, and the usage of the mimicked data. Likewise, the test suite will have two main components: one that facilitates the selection, and another one that helps building the malicious data.

In similar measurements synthesis problems, the selection is uniformly random [ACS17]. Nevertheless, a random selection would not guarantee the introduction of damage to the event detection task. This problem arises in our particular context because the data distribution can be seen as a mixture of different distributions, which characterise the measurements in the presence of events. So it is important to separate the mixtures before applying a random selection criterion.

So, the first step of our test suite is to classify historical data, as shown in the architecture scheme in Figure 7.1. The historical data is analysed, organised into time batches, and labelled as a member of one of the available collections. The labelling criterion reflects the presence and properties of the events. Even though our architecture can cope with any event property that is discriminative for the successful outcome of the attacks, we focus on the events' cardinality. In general, collection i will refer to measurements that were collected in the presence of i events. This would be the correct choice when the WSN risk can be modelled as the number of events that are incorrectly identified. Nevertheless, if other properties determine the risk, such as the spread of the events, it is sufficient to classify the historical data also with respect to this property. This step concludes the *data structuring* of the data to be mimicked. At this point, the *data synthesis* phase can draw the basic data from structured historical data, modify it, and use it to build the malicious data injections.

As we anticipated, the selection criterion for the measurements to mimic is based on the at-

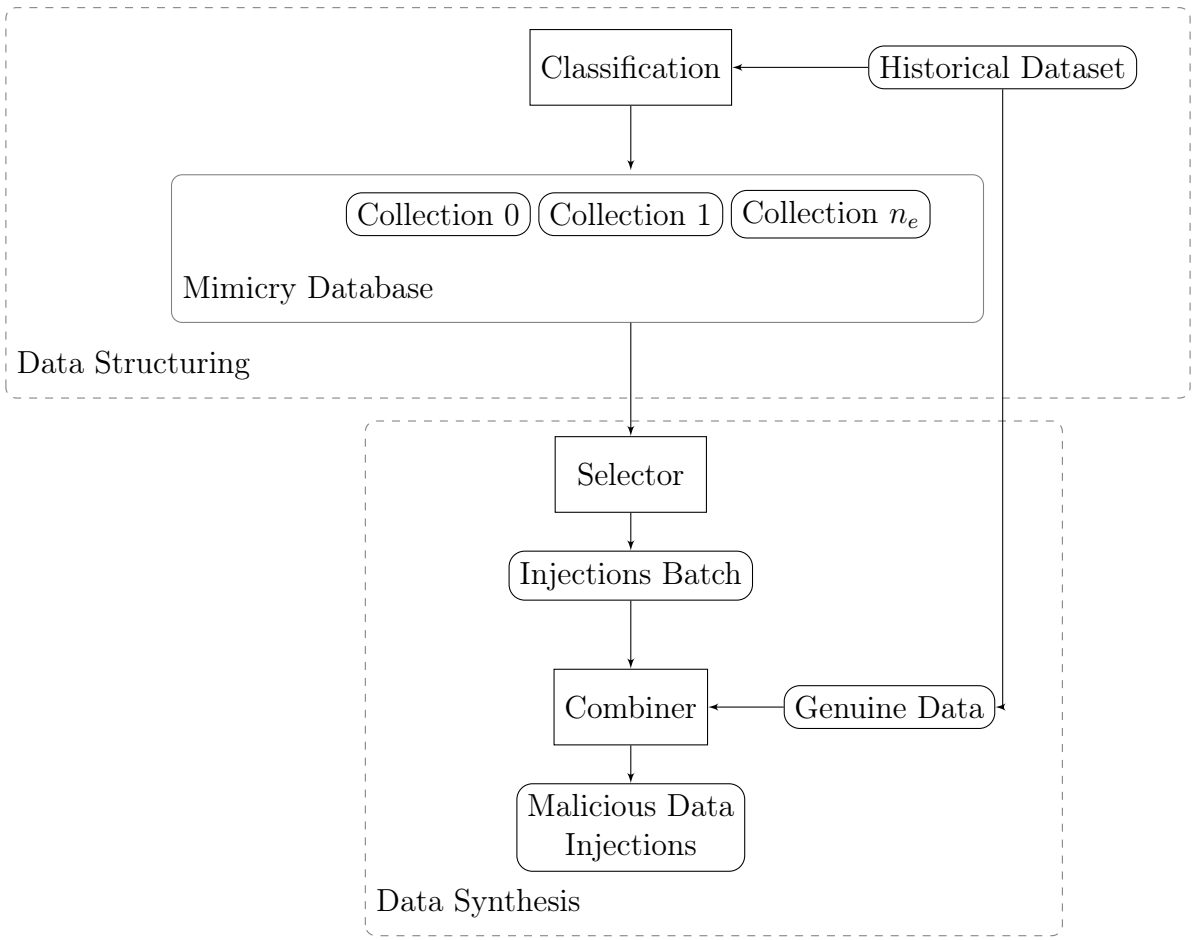


Figure 7.1: Architecture of the algorithm

tacker's goal, i.e. the number of events that need to be elicited or masked. Hence, the selection step will draw a random test batch from Collection i where i is the number of events that the attacker wishes to make detectable (e.g., it may be i=0 if the attack consists of silencing all alarms, or $i = 1$ if the attack is to trigger a false event detection). The selected batch of data cannot be used immediately to produce malicious data injections as multiple problems are involved. One is the problem that the attacker controls only a subset of sensors, hence the mapping can only be partial. Moreover, it is likely that the data needs to be relocated in space, e.g. because producing the same event in the same location where it manifested in the past may be suspicious, or because sensors are mobile. Finally, the most challenging task of mimicry attacks is to take data which was collected under some circumstances and contextualise them in the new circumstances. For example, if the application is that of CCTV, replacing the nightly video stream of a subset of cameras with a daylight stream would create an evident context mismatch. The combiner is the component of the synthesis step which addresses these central

issues.

In the following sections we analyse in detail the problems involved in the data structuring and synthesis phases and address them to make the test suite ready to be implemented.

7.2.3 Data Structuring

In the previous section, we concluded that, unlike other data mimicry problems, malicious data injections have specific restrictions about the properties of the data to be mimicked, including event-related information.

In general, we cannot assume that the attacker knows the event detection algorithm, and even if that is the case, the event detection algorithm may not give information about the number and location of the events but only about their presence. A generic algorithm event detection algorithm will look for gravity points, e.g. areas in which there is an increase in vibrations, in heart rate, in traffic etc., depending on the sensed phenomenon. This property holds true, provided a measurements representation that makes the values' magnitude an indicator of the event, which is possible through a pre-processing stage, as shown in Section 5.6.3. The practical implication of this reasoning, is that a generic event detection algorithm can look for local maxima in space.

In this previous chapter, we concluded that the problem of finding local maxima is similar to the identification of multi-scale trends. Indeed, a local maximum is a point where an increase of measurements is observable and is persistent when looking at the data with different degrees of granularity (otherwise the local maximum may just be the effect of noise). So, we introduced Algorithm 6, a peak identification algorithm denoted with *getEventsPeaks*, which operates in the domain of the wavelet coefficients and gives an array of events peaks as output. The latter are data structures with two members: the event's location and the scale where the peak is most noticeable (related to the main frequency of the event signal).

The event-based classification of the historical data can make use of the *getEventsPeaks* function to identify the number of events and their location. However, this algorithm runs on an

individual measurements snapshots, whereas the purpose of the classification is to collect the whole event's time evolution. In other words, we wish to classify a sequence of measurements snapshots, which were defined as a snapshot sequence in Chapter 4. The event's lifespan can be characterised as the time between the point where at least one event manifests, and the point where at least one event disappears. Hence, there is the need for an algorithm that keeps track of the events' evolution in time, detects changes in the events' presence, and stores the start and ending time instants of an event, together with the events' location. Algorithm 12 deals with these issues.

Algorithm 12 identifyEventBatches

Require: $\mathbf{X}(t_i), \forall t_i$
Ensure: $\mathcal{E}$,$\mathcal{E}_{\text{meta}}$ {Mimicry Database: classified snapshot sequences plus meta-data}
1: oldNEvents = $-\infty$
2: **for all** t_i **do**
3: $(T^{wav}\mathbf{X}_t)(s,\boldsymbol{\tau})$= CWT($\mathbf{X}_t$) {Continuous Wavelet Transform}
4: ePs = getEventsPeaks($(T^{wav}\mathbf{X}_t)(s,\boldsymbol{\tau})$)
5: nEvents=|ePs|
6: **if** oldNEvents $\neq$ nEvents **then**
7: **if** oldNEvents $> -\infty$ **then**
8: $\mathcal{E}$(oldNEvents).append($\mathbf{X}_{t_s}^{t_i}$)
9: **for all** eP $\in$ ePs **do**
10: $\mathcal{E}_{meta}$(oldNEvents).append(eP.loc) {eP.loc contains the peak's location}
11: **end for**
12: **end if**
13: $t_s = t_i$ {Initialise the starting instant of the next snapshot sequence}
14: oldNEvents = nEvents
15: **end if**
16: **end for**
17: **return** collections

The algorithm processes each time sample of historical data individually and sequentially. This assumes that the historical data is made of consecutive time samples. Nevertheless, if this is not the case it is sufficient to split the data into datasets where the time samples are consecutive.

For each time sample, the *getEventsPeaks* function applied to the data's wavelet transform (lines 3-4) retrieves the event's related data, in particular their number and location. The number of events is compared against the previous time instant, and if a change occurred the current classification is terminated (lines 8-10) and a new event snapshot sequence is initiated at the current time instant (line 13). The termination of the current classification consists of

storing both the snapshot sequence – which contains the data relative to all the time instants where the number of events has kept consistent – and its meta data. We refer to the union of all classified snapshot sequences and their meta-data as the *mimicry database*.

In our case, the meta-data coincides with the events' locations, which will be used during the data synthesis phase. Algorithm 12 keeps the last location of the event but, in some applications, the events' locations may change significantly, so it may be more appropriate to track them at multiple time instants. Similarly, in the synthesis phase we will use only the events' locations for selection of the data to mimic, but other properties may be of interest, such as the events' scales (available as an output of the *getEventsPeaks* function), which enables the attacker to select only the events with a spread that is presumably adequate to the attacker's resources.

With the application of algorithm 12, the data structuring phase is complete, and the mimicry attacks are ready to be built with the procedures that are presented in the next section.

7.2.4 Data Synthesis

The data structuring has enabled the automation of the mimicry attacks, since the data has been organised from the perspective of the attack's goals. In this way, it is sufficient to query the mimicry database to retrieve the data to mimic. However, how the data needs to be manipulated depends not only on the goal but also on the real data. Thus the selector block in Fig. 7.1, is in charge of combing both the desired outcome and the current context to select the basic data for the synthesis phase.

Selector

We have considered that the number of elicited and masked events is a fundamental metric of the attack's impact. So it is desirable to design mimicry attacks that satisfy a desired number of elicited and masked attacks, which are denoted with the parameters n_e and n_m, respectively. The starting point of the data synthesis is using such parameters to access the mimicry database.

In particular, if $n_m > 0$, i.e. there are events to mask, the selector draws a random uniform sample in the first collection, i.e. the collection where the measurements batches are free of events. This holds true even if not all events are to be masked, since the measurements of the events to keep will be left unchanged. If $n_e > 0$, the selector draws a random uniform sample from the n_e-th collection, i.e. the collection where the measurements batches contain exactly n_e events. In some cases, such a collection may be empty, especially if it is unlikely that n_e simultaneous events will be observed. In that case, we run the whole synthesis procedure n_e times and draw the samples from the 1^{st} collection.

In the cases where the attack includes both eliciting and masking, the whole synthesis procedure will be repeated, giving priority to the masking, to avoid masking an elicited event.

The parameter n_g is used to select the test batch from the historical dataset. In particular, the batch is randomly selected within the collection n_g to guarantee that the real number of events is n_g.

The parameters n_e and n_g operate on the historical dataset to select the injections batch from a specific collection. If $n_m > 0$, then a random batch of rest measurements is selected. If $n_e = 1$, a random batch of measurements from collection 1 is selected. If $n_e > 1$ there are two different possibilities instead: selecting one batch from collection n_e or selecting n_e batches from collection 1.

The first solution has the advantage of producing more credible measurements: indeed when multiple events are present they may interact with each other and produce measurements that are not the simple juxtaposition of the measurements generated by each event. Compared to the other solution, it has the constraint that the malicious sensors need to be arranged to match the relative positioning of events.

Combiner

To validate the robustness of measurements inspection to malicious data injections, we wish to implement the test suite in Figure 7.1, which reuses historical data. So far, we have addressed

the problem of gathering the data to reuse, and now the next step is to transform it to make it adapt for the current context and also maximise the effort in making the data credible for measurements inspection.

To achieve this goal we need to:

- Select the sensors that will be under the attacker's control.
- Construct the mimicry measurements data from the data given in output by the Selector.
- Contextualise the mimicked data into the current context.

The combiner is the component which is in charge of solving these central problems. We deal with each problem below; in particular, we separate the sensor selection problem for eliciting and masking attacks, which have remarkable differences, while the processes to construct the mimicry data and the contextualisation are conceptually the same. Since many variables are involved in this process, we summarise them in Table 7.1.

Sensors Selection For Elicited Events To minimise the chance of detection, eliciting mimicry attacks will select the best sensors to compromise based on the location at which it is desirable to elicit the events. For this task, we envisage the need for three different criteria:

1. Attack-related: i.e. an attacker may wish to elicit a false event in a specific area, such as triggering a false emergency in a critical area.
2. Hiding the elicited events close to genuine events, when there are one or more genuine events.
3. Random selection, when the other two criteria cannot apply.

The sensors selection problem needs to select the sensors which both elicit the desired event and reduce the chances of detection. These two goals can be pursued at the same time by selecting the sensors that are closest to the location of the elicited events. Indeed, the samples

Table 7.1: Mimicry Combiner Notation Summary

Term	Description
n_e	Number of events to elicit
n_m	Number of events to mask
C	Number of malicious sensor nodes
t_c	Current time
t_m	Starting time of the mimicry batch
T_a	Attack's duration
$\mathbf{X}_{t_m}^{t_m+T_a}$	Genuine measurements to mimic, originally collected between times t_m and $t_m + T_a$
$\breve{\mathbf{X}}_{t_c}^{t_c+T_a}$	Measurements after mimicry attacks, they differ from $\mathbf{X}_{t_c}^{t_c+T_a}$ in maximum C locations.
Ω_{t_i}	Set of sensors locations at time t_i
μ_N	Mean value of measurements noise
σ_N	Standard deviation of measurements noise
$\mathcal{L}_c$	Vector of events locations at current time.
$\mathcal{L}_m$	Vector of events locations at the time of the mimicked data.
$\breve{\mathcal{L}}_c$	Vector of desired events locations for elicited events.

that are closer to the event's location are usually the most representative of the main event's characteristics, which are the basis for the event detection criterion. At the same time, if the elicited event is detected and there are samples close to its location which do not support the event because they are genuine, they will most likely introduce detectable anomalies. So, when the budget of malicious sensors is limited, it is important to prioritise them based on the closeness to the elicited event's location.

In Algorithm 13 we implement the strategies discussed above to address the problem of selecting the locations of the elicited events and the malicious sensors. Line 4 implements the first event location selection criterion, i.e. when it is an attack's input. Line 13 implements the third criterion, i.e. the random uniform selection across the deployment range.

Lines 7-11 implement the second and most complex criterion, which applies when there are genuine events and the location is not given in input. Line 6 selects the genuine event next to which the eliciting attack will take place, while lines 7-11 define the exact location, which

Algorithm 13 Selection of eliciting sensors

Require: C, n_e Ω_{t_c}, $\mathcal{L}_c$,$\mathcal{L}_m$,$\breve{\mathcal{L}}_c$
Ensure: malIndices {Indices of Malicious Sensors}
1: **for all** e in $1, ..., n_e$ **do**
2: $\mathbf{l}_m$=$\mathcal{L}_m(e)$ {Get the location of the event to mimic in its original context}
3: **if** $|\breve{\mathcal{L}}_c| \geq e$ **then** {If a preference has been give for the location where to elicit the e-th event, use it}
4: $\breve{\mathbf{l}}_\mathbf{c} = \breve{\mathcal{L}}_c(e)$
5: **else if** $|\mathcal{L}_c| > 0$ **then** {If there are genuine events, elicit next to them}
6: $\mathbf{l_c} = \mathcal{L}_c(\text{MIN}(e, |\mathcal{L}_c|))$
7: **for all** $d \in 1, |\mathbf{l_c}|$ **do**
8: $b_d^{low} = \text{MAX}_{\omega_{t_c} \in \Omega_{t_c}}(\mathbf{l_c}(d) - \omega_{t_c}(d))$
9: $b_d^{high} = \text{MAX}_{\omega_{t_c} \in \Omega_{t_c}}(\omega_{t_c}(d) - \mathbf{l_c}(d))$
10: $\breve{\mathbf{l}}_\mathbf{c}(d) = \text{SkewNormal}(\mathbf{l_c}(d), b_d^{low}, b_d^{high})$
11: **end for**
12: **else** {No location is particularly convenient for eliciting}
13: $\breve{\mathbf{l}}_\mathbf{c} = \text{Uniform}(\text{range}(\Omega_{t_c}))$
14: **end if**
15: closestToElicited = $argsort||\Omega_{t_c} - \breve{\mathbf{l}}_\mathbf{c}||$ {Sort the sensors by closeness to the event to elicit}
16: **if** $e < n_e$ **then**
17: C_e=$\lfloor C/n_e \rfloor$
18: **else**
19: C_e=$\lfloor C/n_e \rfloor + (C - n_e \lfloor C/n_e \rfloor)$
20: **end if**
21: malIndices.append(closestToElicited$(1, .., C_e)$) {Keep the C_e closest sensors}
22: **end for**
23: **return** malIndices

is built with random samples of the *Skew Normal distribution* [DV04]. This distribution is chosen because its peaked shape allows us to give high probability to the locations close to the elicited event. Moreover, since it is skewed, we can spread the locations probability across the deployment area, whose boundaries have different distances from the genuine event's location. Indeed, for each spatial dimension d of the deployment, we calculate b_d^{low} and b_d^{high}, i.e. the distance from the furthest sensor in one direction at line 8, and in the other direction at line 9. Then, we can sample from a Skew Normal distribution that: is centred in the genuine event $\mathbf{l_c}(d)$, and is approximately within the deployment range, i.e. $[\mathbf{l_c}(d) - b_d^{low}, \mathbf{l_c}(d) + b_d^{high}]$ (namely, with probability 0.994).

The role of the SkewNormal function at line 10 is to use these three values to calculate the three parameters of the Skew Normal Distribution. To make $\mathbf{l_c}(d)$ the centre of the distribution

we make it equal to the 50% percentile. To keep the values in the deployment range, we make b_d^{low} and b_d^{high} equal to the 0.3% and 99.7% percentile respectively. The result is a system of three equations with three unknowns that are solved numerically[1].

At this point, with the location of the elicited event well defined, the malicious sensors are selected at lines 15-21 as the closest sensors to such location. In particular, we use C/n_e for each event to be elicited (except for the last elicited event which takes all the remaining sensors if C is not divisible by n_e), i.e. we partition the resources equally for each event since the objective is to make all elicited events undetected. In general, the minimum number of sensors that make an elicited event undetectable is unknown, so with a fair partitioning we seek to equally minimise the chance of detection for all of them.

Sensors Selection for Masked Events Similarly to the eliciting scenario, a requirement for mimicking rest conditions is the selection of the masking sensors. However, since there is no event to elicit, the sensors selection criterion is just based on the genuine event that needs to be masked. Algorithm 14 implements this criterion, which is analogous to the last part of Algorithm 13, with the difference of selecting the closest sensors to the event to mask, rather than elicit, at line 3. The idea is to remove the most meaningful samples for the event in order to prevent its detection. At the same time, leaving some event measurements close to the genuine event unchanged (i.e. with their genuine value) would likely introduce anomalies.

Construct the mimicry measurements At this point, the event to mimic can be mapped to the malicious measurements, whose locations have been selected with Algorithm 13 or 14, so they may not match the locations of the mimicked data. Algorithm 15 addresses this problem by transforming the absolute sensor locations into locations relative to two *focus points*, one

[1]The system of equations is

$$\Phi(\frac{q_i-\xi}{\omega}) - 2T(\frac{q_i-\xi}{\omega}, \alpha) = \begin{cases} 0.003 & \text{if} \quad i=1 \\ 0.5 & \text{if} \quad i=2 \\ 0.997 & \text{if} \quad i=3 \end{cases} \tag{7.1}$$

Where ξ, ω and α are the three unknowns. q_1, q_2, and q_3 coincide with $\mathbf{l_c}(d)$, b_d^{low} and b_d^{high} respectively. Φ is the cumulative distribution function of the standard normal distribution. T is the *Owen's T Function* [Owe56].

Algorithm 14 Selection of masking sensors

Require: C, n_e Ω_{t_c}, $\mathcal{L}_c$
Ensure: malIndices {Indices of Malicious Sensors}
1: **for all** e in $1, ..., n_m$ **do**
2: $\breve{\mathbf{l}}_\mathbf{c} = \mathcal{L}_c(e)$
3: closestToEvent = argsort($||\Omega_{t_c} - \mathbf{l}_c|$) {Sort the sensors by closeness to the event to mask}
4: **if** $e < n_m$ **then**
5: $C_m = \lfloor C/n_m \rfloor$
6: **else**
7: $C_m = \lfloor C/n_m \rfloor + (C - n_m \lfloor C/n_m \rfloor)$
8: **end if**
9: malIndices.append(closestToEvent$(1, .., C_m)$) {Keep the C_m closest sensors}
10: **end for**
11: **return** malIndices

for the mimicked and one for the current data.

In particular, for eliciting attack, the focus point for the current data is the elicited event's location, which is subtracted from the current sensors' locations at line 6. Similarly, the focus point for the mimicry data is the original event's location, which is subtracted from the original sensors' locations to make them event-relative at line 7.

Instead, for masking attacks, the current locations' focus point is the event to mask, which is used at line 9. Since in a rest condition there is no particular point of interest, the mimicked data uses the centre of the WSN deployment as a focus point, as this choices generally maximises the density of known data points in the area where the interpolation points lie. When this is not the case, i.e. the centre of the WSN deployment is not densely populated with sensors, it is more appropriate to use another point when the density of sensors is higher.

Once the absolute locations are transformed into relative ones, they acquire the same meaning. Yet there is no obvious match between the sensors' locations in the original and current context, so we need a cross-context interpolation (line 15), which becomes a classical interpolation in the relative locations thanks to the previous step. For this task, an interpolation that works on non-uniformly spaced samples is needed, such as the Inverse-distance-weighted interpolation [She68], which will be used in the following experiments.

The interpolation produces the event measurements that could be injected by the malicious

Algorithm 15 Construct mimicry measurements

Require: elicit, n_a, Ω_{t_c}, Ω_{t_m}, $\mathcal{L}_c$ $\breve{\mathcal{L}}_c$, $\mathcal{L}_m$, T_a, $\mathbf{X}_{t_c}^{t_{c+T_a}}$, $\mathbf{X}_{t_m}^{t_{m+T_a}}$, μ_N, σ_N
Ensure: $\breve{\mathbf{X}}_{t_c}^{t_{c+T_a}}$

1: **for all** e in $1, ..., n_a$ **do**
2: $\mathbf{l}_m$=$\mathcal{L}_m(e)$
3: $\breve{\mathbf{l}}_\mathbf{c} = \breve{\mathcal{L}}_c(e)$
4: $\mathbf{l}_c = \Omega_{t_c}$
5: **if** elicit **then**
6: relMalLocs = $\{\boldsymbol{\omega}_{t_c} - \breve{\mathbf{l}}_\mathbf{c} \, \forall \boldsymbol{\omega}_{t_c} \in \Omega_{t_c}$(malIndices) {Locations of malicious sensors relative to elicited event}
7: relOrigLocs = $\{\boldsymbol{\omega}_{t_m} - \mathbf{l}_m \, \forall \boldsymbol{\omega}_{t_m} \in \Omega_{t_m}\}$ {Locations of original sensors relative to original event}
8: **else**
9: relMalLocs = $\{\boldsymbol{\omega}_{t_c} - \mathbf{l}_c \, \forall \boldsymbol{\omega}_{t_c} \in \Omega_{t_c}$(malIndices) {Locations of malicious sensors relative to masked event}
10: relOrigLocs = $\{\boldsymbol{\omega}_{t_m} - \text{centre}(\Omega_m) \, \forall \boldsymbol{\omega}_{t_m} \in \Omega_{t_m}\}$ {Locations of original sensors relative to deployment's centre}
11: **end if**
12: **for all** $i \in 1, ..., T_a$ **do**
13: **for all** malIndex $\in$ malIndices **do**
14: relMalLoc = relMalLocs(malIndex)
15: interp = interpolate(relMalLoc ,relOrigLocs,$\mathbf{X}_{t_{c+i}}$, $\mathbf{X}_{t_{m+i}}$)
16: $\breve{\mathbf{X}}_{t_{m+i}}$(malIndex) = timeSmooth(interp,i,T_a,$\mathbf{X}_{t_{m+i}}$(malIndex))
17: **end for**
18: spaceSmooth($\breve{\mathbf{X}}_{t_{m+i}}$, $\boldsymbol{\omega}_{t_c}$, malIndices, b)
19: addNoise($\breve{\mathbf{X}}_{t_{m+i}}$,$\mu_N$, σ_N)
20: **end for**
21: **end for**
22: **return** $\breve{\mathbf{X}}_{t_c}^{t_{c+T_a}}$

sensors. The sensors have to cover a contiguous area around the elicited or masked event, to maximise the chance of passing measurements inspection. Nevertheless, the eliciting measurements should be adapted to the current context to avoid obvious anomalies. This task is handled with temporal smoothing at line 16, which is done individually for each malicious sensor, a spatial smoothing at line 18, which considers the relationships between genuine and malicious sensors, and the addition of noise at line 19. These three steps solve the contextualisation problem, and are described below.

Contextualising the Mimicked Data The injection of malicious data causes a mismatch in the temporal and spatial domain due to the presence of malicious measurements, which

were following the temporal evolution of the physical phenomenon before the attack time, and respected all its spatial propagation characteristics. The chance of disrupting both the temporal and spatial characteristics of the physical phenomenon can be minimised by variations that are sufficiently small.

In the case of the time domain, the genuine temporal dynamics of the physical phenomenon can be respected by setting the malicious measurements at an intermediate point between the genuine value and the target value, which has been calculated by the Combiner at the previous step. If such an intermediate point shifts towards the target value with the dynamic of the physical phenomenon, we achieved both the target of injecting the desired measurements and that of making temporal transitions indistinguishable from genuine ones. In fact, this goal can be achieved with a weighted average of the real and malicious measurement, where the weight of the latter starts at $\frac{i}{T_a}$ and becomes equal to one at the end of the attack, i.e. after T_a time steps.

$$\breve{\mathbf{X}}_{t_{m+i}}(\text{malIndex}) = (\frac{i}{T_a})\text{interp} + (1 - \frac{i}{T_a})\mathbf{X}_{t_{m+i}}(\text{malIndex}) \tag{7.2}$$

Note that our attack model, described in Chapter 4, assumes that the attacker has infinite time for the attack and that this allows them to avoid any kind of anomaly given by temporal relationships. However, the use of a limited time T_a, which is the time imposed by the length historical time series, may not be sufficient to achieve this goal; indeed, the role of (7.2) is to minimise the chance of introducing anomalies in the temporal domain.

Since the temporal domain is completely under the attacker's control, the malicious contribution to the measurements time series can be smoothed arbitrarily to appear normal. In the spatial domain instead, the problem is more complex as the smoothing needs to consider the presence of genuine measurements, which cannot be controlled. This constraint leads us to an *N-nearest neighbour average* where the neighbours also include genuine sensors.

If we let all malicious sensors replace their target measurements with a neighbourhood average, we would likely miss the goal of causing damage to the event detection task. Thus, we use

instead only a subset of malicious sensors for these task, and let them mediate between the genuine sensors and the malicious sensors that are focusing on eliciting or masking events. The mediating sensors are referred to as *blending-in* sensors and are selected as the furthest sensors from the elicited/masked event. Indeed, for the sensor selection criterion, which is by closeness to the elicited/masked event's location, the sensors that are closest to the genuine events are just the furthest from such a location. This operation is similar to the blurring trick used in image forgery when inserting an extraneous object inside an image [HP05], which smooths the edges of the extraneous object.

Algorithm 16 spaceSmooth

Require: ($\breve{\mathbf{X}}_{t_{m+i}}$, $\mathbf{\Omega}_{t_c}$, malIndices, b, N_n)
Ensure: $\breve{\mathbf{X}}_{t_{m+i}}$ (with spatially smoothed injections)
1: $C_a = |\text{malIndices}|$
2: $\mathbf{\Omega}^C_{t_c} = \mathbf{\Omega}_{t_c}(j)\, \forall j \in \text{malIndices}$
3: $\mathbf{\Omega}^G_{t_c} = \mathbf{\Omega}_{t_c} - \mathbf{\Omega}^C_{t_c}$
4: $\breve{\mathbf{X}}^b_{t_{m+i}} = \breve{\mathbf{X}}_{t_{m+i}}$ {Initialise vector of blended injections}
5: blendingInIndices = $\text{argsort}_{\omega^C_{t_c} \in \mathbf{\Omega}^C_{t_c}} MIN_{\omega^G_{t_c} in \Omega^G_{t_c}} ||\omega^C_{t_c} - \omega^G_{t_c}||$
6: **for all** $b_j \in$ malIndices(blendingInIndices) **do**
7: $\boldsymbol{\omega}^{b_j}_{t_c} = \mathbf{\Omega}_{t_c}(b_i)$
8: $N(\boldsymbol{\omega}^{b_i}_{t_c}) = \text{argsort}(||\mathbf{\Omega}_{t_c} - \boldsymbol{\omega}^{b_i}_{t_c}||)$ {Nearest neighbours of $\boldsymbol{\omega}^{b_i}_{t_c}$}
9: $N_n(\boldsymbol{\omega}^{b_i}_{t_c}) = N(\boldsymbol{\omega}^{b_i}_{t_c})(1, ..., N_n)$ {N_n Nearest neighbours of $\boldsymbol{\omega}^{b_i}_{t_c}$}
10: $\breve{\mathbf{X}}^b_{t_{m+i}}(b_i) = \frac{1}{N_n} \sum_{j \in N_n(\boldsymbol{\omega}^{b_i}_{t_c})} \breve{\mathbf{X}}_{t_{m+i}}(j)$
11: **end for**
12: $\breve{\mathbf{X}}_{t_{m+i}}(b_j) = \breve{\mathbf{X}}^b_{t_{m+i}}(b_j) \quad \forall b_j \in$ malIndices(blendingInIndices)
13: **return** $\breve{\mathbf{X}}_{t_{m+i}}$

Algorithm 16 describes the steps: bC_a malicious sensors are used as blending-in sensors, where C_a is the number of malicious sensors used for the mimicry attack and b is a value between 0 and 1 indicating the fraction of blending-in sensors. For each blending-in sensor , the average among its N_n nearest neighbours is calculated and substituted to the original value. This choice ensures that with more malicious sensor the value of the blending in sensors still brings in a high degree of damage. Instead, when many genuine sensors surround the blending in sensor, the mediation of the blending in sensors is more remarkable.

The value of N_n should be high enough to include the data of the malicious sensors that are not blending-in, and not too high to avoid including the data of genuine sensors which are not

expected to be correlated. For this reason, it will be picked in the interval $[(1-b)C_a, (2-2b)C_a]$, with a preference towards the left bound to increase the chance of causing damage and a preference towards the right bound to decrease the chance of detection.

A final contextualisation step is needed to reduce the chance that the reuse of historical data is detected by an ad-hoc system. This task has been partially addressed with the interpolation of the data into new locations and is completed by the summation of Gaussian noise to the malicious data obtained so far. The noise component is:

$$\mathcal{N}(\mu_N, \sigma_N) \tag{7.3}$$

That is a Gaussian independent and identically distributed noise component. The average is an estimation of the sensor's bias, while the standard deviation is an estimation of the device sensing precision, for which an estimate is generally given by the sensor's vendor, and alternatively can be estimated from the historical data (e.g., through a Kalman filter [Kal60]).

7.2.5 Experiments

The test suite presented in the previous Section for the production of mimicry attacks is an automatic tool for testing the robustness of generic measurements inspection algorithms for generic WSNs. In this section we build mimicry attacks which we submit to the inspection of the wavelet-based approach. Moreover, we feed a state-of-the-art approach with the same data to compare the results.

As pointed out in Sect. 4.1, we consider an attacker with full capabilities in sensor selection, attack time, and overhearing measurements. As a consequence, the parameter that mostly determines the attacker's capabilities in our model is the number of sensors injecting compromised measurements, denoted with C. We study the impact of such parameters by evaluating the results of our experiments with different values of C. Note that, on the other hand, detection is generally easier with a higher number of total sensors N. However, the detection performance

depends also on the deployment and on the events' spatial propagation patterns: for instance, if the attacker compromised all the sensors within the boundary of a generic event, such sensors are enough to mask the event without even being detected, regardless of N. For this reason, we fix the deployment distribution to a *random uniform* distribution of N sensors in a well-defined space and study the results with different values of C.

Another testing problem arises from events. As they are generally infrequent, measurements collected in the presence of one or especially multiple events are difficult to retrieve. A possible solution is to simulate the measurements under event conditions when they are particularly rare. Despite that, one still wishes to have confidence that the technique is suitable for real measurements.

In addressing the problems described above, we tested our technique both on synthetic and real measurements. We prove the applicability of the technique on a few real sets of measurements, while we evaluate the detection, characterisation and diagnosis performance on several sets of synthetic measurements.

Synthetic wildfire temperatures dataset

The data measurements being used below are model-generated temperatures perceived by N sensors randomly scattered in an area of 100×100 m^2. The random function that generates the deployment is the "continuous uniform" distribution over the interval $[\mathbf{u}_j - \sqrt{\mathbf{N}}, \mathbf{u}_j + \sqrt{\mathbf{N}}]$, where $\mathbf{u}_j$ is the location where sensor j would be placed if all sensors were placed in a grid. This choice simulates the effort to put the sensor nodes in a grid, which maximises the WSN coverage, and also falling back on a pseudo-grid due to the presence of deployment constraints. We generate 200 samples from the deployment distribution to reproduce the measurements of as many temperature sensors.

The temperature sensors may observe temperature increases due to the outbreak of fires within the WSN area and anywhere next to it (hence only part of fire may be observed by the sensors). The fires are modelled as the result of the radiated power from one or more heat sources placed at

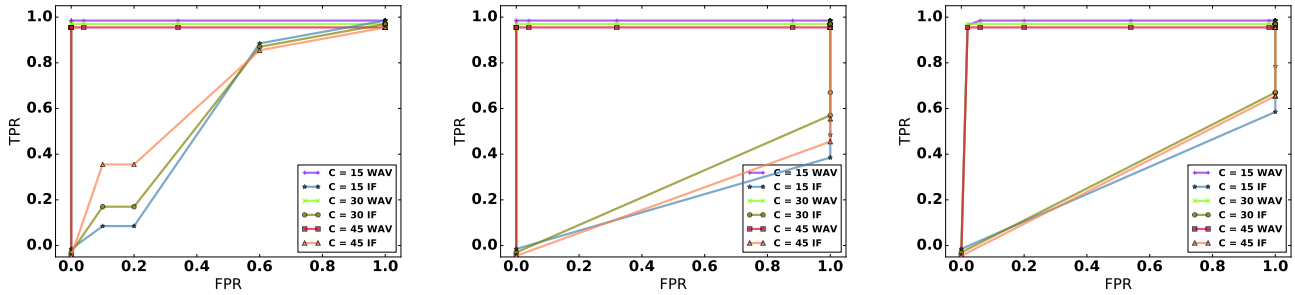
different locations, which are modelled as spherical black bodies with temperature K_i (between 573.15 and 1473.15 Kelvin [SEK03; SC92]). Such black bodies have radius r_i, centred at $\mathbf{c_i}$, and their power is radiated to each location $\mathbf{u} \in \Omega$ at distance $d(\mathbf{c_i}, \mathbf{u})$, producing the temperature:

$$K(\mathbf{u}) = \sum_i (\frac{1}{4} {K_i}^4 (\frac{r_i}{d(\mathbf{c_i}, \mathbf{u})})^2)^{\frac{1}{4}} + \mathcal{N}(0, 3) \tag{7.4}$$

Where $\mathcal{N}$ is additive zero-mean Gaussian noise with $3K$ standard deviation, which simulates the interference arising from the sensing devices.

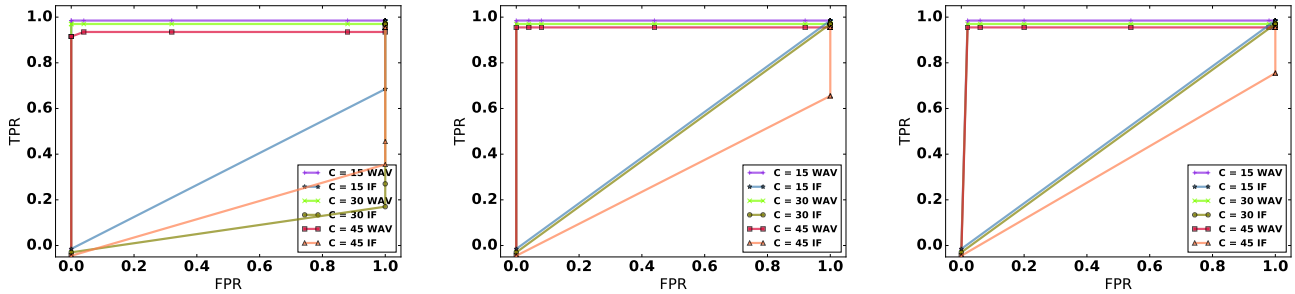
Detection We first evaluate the detection of malicious data injections with the receiver operating characteristic (ROC), which shows the relationship between True Positive Rate (TPR) and False Positive Rate (FPR). To obtain the FPR, we simulated 50 sets of genuine measurements and gathered the frequency of positive detection. To obtain the TPR, we first simulated 50 sets of genuine measurements, then we simulated malicious data injections with the method described in Sect. 7.2.2, and finally collected the frequency of positive detection. Events are generated with random locations and peak temperatures (corresponding to parameters c_i and T_i in (7.4)) and with radius $r_i = 22$ metres both for elicited and masked events.

We made an analogous analysis for the algorithm described in [Rez+13], which is an *Iterative filtering*-based algorithm for detecting malicious colluding sensor nodes. The steps of the algorithm are: 1. For each time instant, the measurements of all sensors are aggregated with iterative filtering. 2. The error time series of each sensor is calculated, i.e. the difference between a sensor's measurements time series and the aggregates time series. 3. Errors are collected in batches, normalised with subtraction of mean and division by standard deviation, and tested for normality. In the last step, the sensor is classified as malicious if its measurements are not normally distributed. The normality test is run with the Kolmogorov-Smirnov test, which quantifies the distance between the errors distribution and the normal distribution. In particular errors are not normally distributed if such distance is bigger than a threshold T_{IF}. Since this algorithm performs detection with the granularity of sensors, we assume that the detection of malicious data injections is positive if at least one sensor fails the test.



(a) No Event (b) 1 Event (c) 3 Events

Figure 7.2: Eliciting Detection ROC curves VS number of events. Performance decrease with more events.



(a) 1 Event (b) 2 Events (c) 3 Events

Figure 7.3: Masking Detection ROC curves VS number of events. Performance increase with more events.

We split the analysis into eliciting and masking scenarios, shown respectively in Fig. 7.2 and Fig. 7.3. For both we present the ROC for an increasing number of events. We denoted our wavelet-based method as *WAV*, while the iterative-filtering based method is denoted with *IF*. The values of the ROC curves from left to right have been obtained with thresholds values between 0.55 and 1.2 for WAV, and between 0.05 and 0.15 for IF.

As shown in Fig. 7.2a, with no genuine events WAV can achieve TPR=1 with FPR = 0. When there are many genuine events, the performance slightly decreases. IF, in comparison, cannot achieve WAV's TPR without a significant increase in false positives. Furthermore, with one or more events FPR=1 for any TPR higher than zero: this means that regardless of the IF parameters, every time there is an event this triggers detection. Overall, IF's performance with one or more events is worse than the random classifier. This is due to the lack of a learning phase in IF, which is present in WAV instead and enables it to extract the properties

of genuine events. This consideration is confirmed by the fact that IF's detection improves with more malicious sensors: indeed the latter can spoof larger events, which look more anomalous for IF.

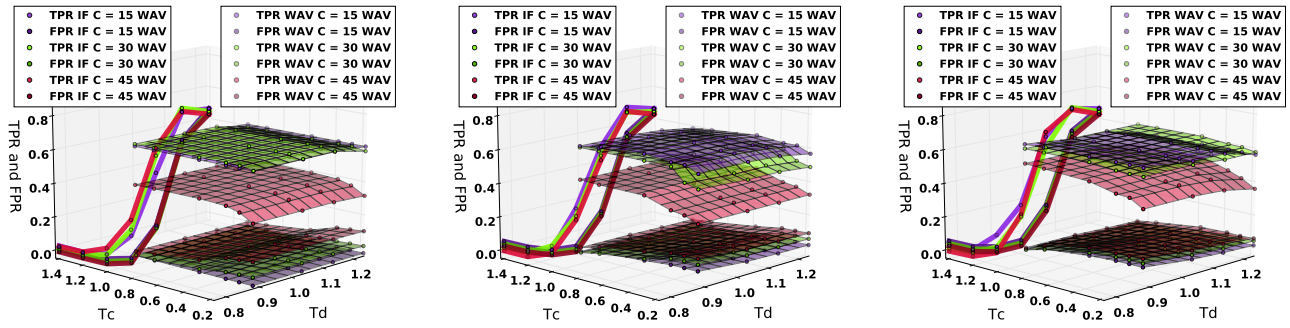
Both with WAV and IF, the detection performance generally decreases as the number of events increases. The reason is that the continuous changes introduced by genuine events sometimes become valid explanations for the changes brought in by the eliciting sensors.

With masking scenarios there are opposite results compared to eliciting scenarios: the detection performance increases with the number of genuine events. Indeed the shortage of rest sensors with more events makes the masking sensors contradict the measurements of more sensors. Fig. 7.3 shows that WAV's ROC curve are nearly ideal. As in eliciting scenarios, IF is not able to avoid false positives in masking scenarios when the TPR is higher than zero.

Characterisation We present below the results for the characterisation task, whose performance is measured by TPR vs FPR averaged across the 50 measurements time snapshots. This is done in WAV for values of T_c between 0.1 and 1.0 and in IF for different values of T_{IF} between 0.05 and 0.15. Note that the characterisation algorithm in WAV depends both on the characterisation threshold T_c and on the detection threshold T_d, hence we show 3D curves for the characterisation TPR and FPR for WAV. IF instead, has a unique threshold, hence it produces 2D curves. Note that characterisation is not a classification problem, hence the TPR is not in a monotonic relationship with the FPR. Indeed a lower threshold T_c may shift the blame from a malicious group to a genuine group, causing both a decrease in TPR and an increase in FPR. The result is not a proper ROC curve, therefore we present the TPR and FPR separately.

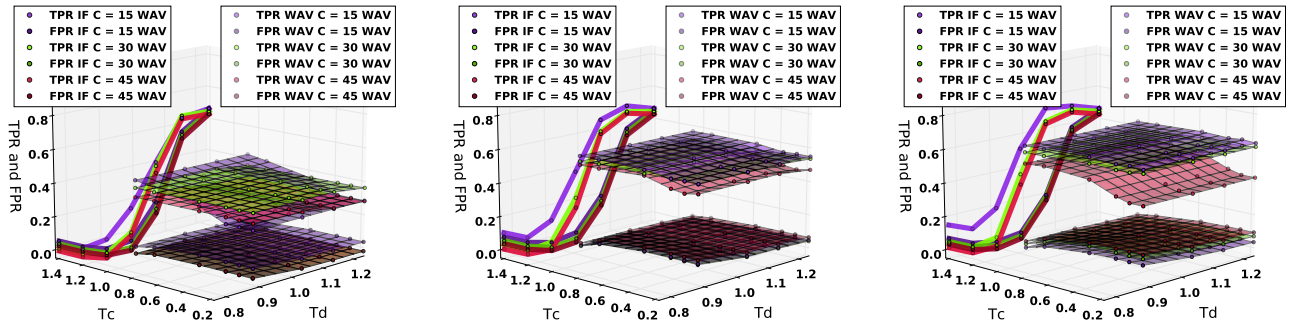
In Fig. 7.4 and 7.5 we show the characterisation results for the eliciting and masking scenario respectively, for increasing number of genuine events. Note that the stability of TPR and FPR for different values of the detection threshold implies that a wrong selection of the detection threshold has a reduced impact on the characterisation performance.

Like for detection, the characterisation performance with increasing number of events shows a decrease in eliciting scenarios and an increase for masking scenarios. WAV's characterisation



(a) No Event (b) 1 Event (c) 3 Events

Figure 7.4: Eliciting Characterisation results. WAV achieves characterisation with TPR=0.4 to 0.7 when FPR=0.02.



(a) 1 Event (b) 2 Events (c) 3 Events

Figure 7.5: Masking Characterisation results. WAV achieves characterisation with TPR=0.4 to 0.7 when FPR=0.02.

TPR is quite steady in the area $0.4 - 0.6$ and the FPR is steady as well in the area $0 - 0.1$. IF instead, has a continuous increase of the TPR which is matched with a similar increase in the FPR. However, keeping a reduced amount of malicious sensors in the network is not a major concern since they will be likely discovered in the future, consequently to the reduction of their collusion power. For this reason keeping the FPR to a minimum may be preferred, and in the areas where the FPR is low, the WAV's TPR is amply above IF's.

The reason why WAV's TPR seems upper bounded around 0.5 is that, as explained in Sect. 7.2.4, the Combiner block of the test suite contextualises the malicious data by using malicious blending in sensors, i.e. sensors not actively participating in the eliciting/masking attempt, but acting as accomplices that mediate between genuine and malicious sensors to evade detection. In our experiments, we used 50% of the C malicious sensors as blending-in sensors, which WAV

avoids judging as malicious to reduce false positives, as a similar behaviour is observable also under genuine circumstances.

Diagnosis Diagnosis is the most complex step in our methodology, not only because distinguishing faults and malicious interference is a complex problem in nature, but also since the correctness of diagnosis is conditioned to the correctness of both detection and characterisation. This characteristic introduces an error propagation problem that is hard to solve without additional information.

Table 7.2 presents the diagnosis confusion table for the following categories: {1 Genuine Event}, {1 Event Elicited}, {1 Genuine Event Masked}, {1 Single Fault Within Genuine Event}, {Group Fault of 10 Sensors Within Rest Condition}. The categories on the rows represent the ground truth, while the categories on the columns represent the output of diagnosis. Each table cell describes the frequency of times the category on the row is classified as the category on the column. The values of T_d and T_c are fixed for this analysis and equal to 1 and 0.85 respectively, since they provided a good trade off of TPR vs FPR.

Table 7.2: Diagnosis Confusion Table

	C	Genuine	Elicited	Masked	Single Fault	Group Fault
Genuine		0.98	0.02	0.00	0.00	0.00
Elicited	15	0.00	0.82	0.12	0.00	0.06
	30	0.00	0.76	0.12	0.00	0.12
	45	0.00	0.76	0.14	0.00	0.10
Masked	15	0.00	0.10	0.86	0.00	0.04
	30	0.02	0.22	0.74	0.00	0.02
	45	0.04	0.24	0.62	0.00	0.10
Single Fault	1	0.02	0.12	0.16	0.70	0.00
Group Fault	10	0.00	0.08	0.04	0.00	0.88

We consider also the genuine event category to analyse also which categories evade detection most. As we observe in the first row, the main responsibility for a wrong detection are elicited events, which resemble the genuine events most.

As the second row summarises, eliciting scenarios are mainly confused with group faults and masking scenarios, especially when C increases. Indeed, with more malicious sensors, inferring if the incomplete events have been elicited or are the remainder of a genuine event is more difficult.

In the third row, we note that masking scenarios are mainly confused with eliciting scenarios, since masking injections that remove most part of a genuine event make the event appear incomplete and it is difficult to say if it has been incompletely elicited or incompletely masked.

In the fourth row we note that single faults are equally confused with the eliciting and masking categories, but not with group faults. Indeed, single faults introduce higher measurements variation, whereas group faults are identified with a low coefficient of variation.

Group faults are more likely to be confused with malicious data injections, as evident at the fifth row. Among them, confusion with eliciting injections prevails because the sensors are simulated to report corrupted measurements that are larger than the real ones.

In general, our diagnosis criteria make a correct classification for a significant portion of samples (around 75%). This result has been achieved without any hypothesis about the nature of faults, which would probably improve performance despite reducing the method generality.

Real worldwide seismic vibrations dataset

The dataset retrieved from a real WSN is made up of measurements collected by seismic networks from around the world [Iric; Iria], and kindly provided by the Incorporated Research Institutions of Seismology (IRIS) Data Management Center (DMC) [Irib].

The events considered are earthquakes perceived by seismic sensors in: Alaska, Arizona, California, Michigan, Oklahoma, Oregon, Texas, Utah, and Dominican Republic. The application to different sensors and different environments highlights that our technique focuses on properties of the physical phenomenon and therefore is particularly suited to WSNs where the sensors enter, leave or migrate across the deployment area.

To learn the cross-scale relationship we exploit as few as nine earthquakes. We use other two real earthquakes for testing: one is the Arizona's earthquake of 1st December 2014, the second is the California's earthquake of 30th May 2015, both with a magnitude of 4.7 Richter.

Before testing our techniques on this dataset with the mimicry test-suite, we pre-process the measurements to satisfy our requirement that the measurements magnitude is proportional to the event perceived. In particular, we use a time domain processing known as delayed STA/LTA (DSTALTA) [Wit+98]. The STA/LTA algorithm evaluates the ratio of short to long-term energy density, and a delay is added between the two to increase the independence between the two [RH92]. Finally we apply a logarithm for better visualisation.

Through the help of figures, we present a set of measurements with the respective anomaly score, the sensors given in output by the characterisation step (identified by white crosses), and the diagnosis output (written on top of the anomaly score map). Since we have few samples to learn the cross-scale relationships from this case, it is safer to use higher thresholds. Hence, we used $T_d = 1.4$, and $T_c = 1.2$.

In the experiments presented below, the number of sensors considered is not fixed but it is defined such that the earthquakes' boundaries are well visible. There are about 800 sensors in the American IRIS networks, but the earthquake is always perceived only by a subset of them. Fig. 7.6 shows the detection result on the genuine Arizona's earthquake, which is identified as genuine as the anomaly score is lower than T_d. In this scenario about 25 sensors perceive mostly the earthquake, while for other 150 the vibrations are close to the noise range $[0, 10^{1.5}]$. As shown in Fig. 7.7, detection is neither triggered on the genuine California's earthquake: this is a rare case where two earthquakes are simultaneous (4 p.m. local time) and with close epicentres (northern and southern California). Fig. 7.8, shows a sophisticated spoofing attack that is introduced close to a genuine event. The latter is perceived by nearly 30 sensors and the spoofed event, which is introduced by 20 malicious sensors, looks indistinguishable from it. Nevertheless, our detection algorithm triggers, and characterisation assigns different events to different groups (Sect. 6.6.1), and is able to return four out of 20 malicious sensors without false positives.

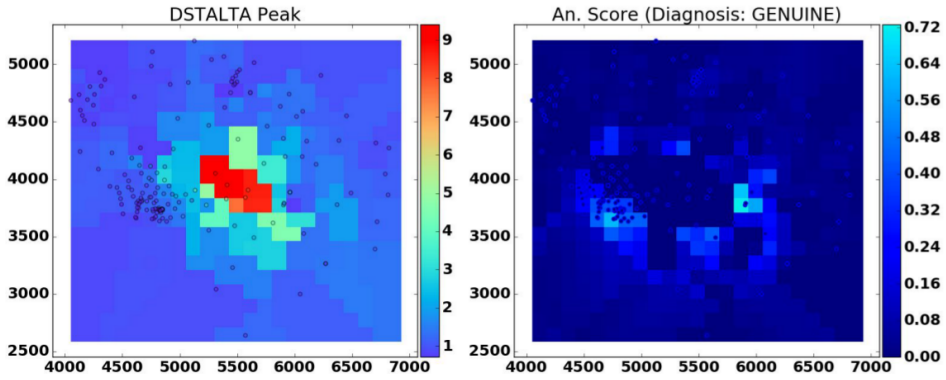


Figure 7.6: A genuine event does not produce anomaly score bigger than T_d

.

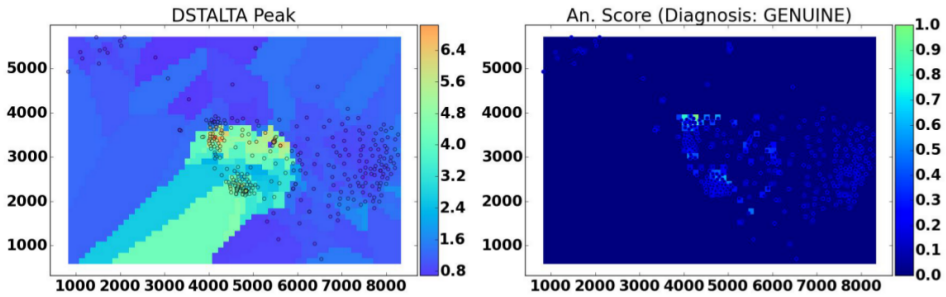


Figure 7.7: The cross-scale relationship induced by two close genuine events has been correctly learnt.

Fig. 7.9 simulates an attacker that is trying to replicate the Arizona's earthquake with the 70 closest sensors to the event peak. Despite the measurements that perceived the event most are correctly replicated, the genuine sensors, which have lower measurements, raise a conflict with the malicious measurements. Here, 14 out of 70 malicious sensors are correctly identified without false positives.

Finally, Fig. 7.10 shows an attempt to mask the Arizona's earthquake with as many as 120 malicious sensors, whose measurements depict the absence of any event. Note that, as a result, the original event is split into two smaller events, which may be confused with elicited events. Nevertheless, our characterisation algorithm successfully identifies the presence of a masking attack and returns 26 out of 120 malicious sensors without false positives.

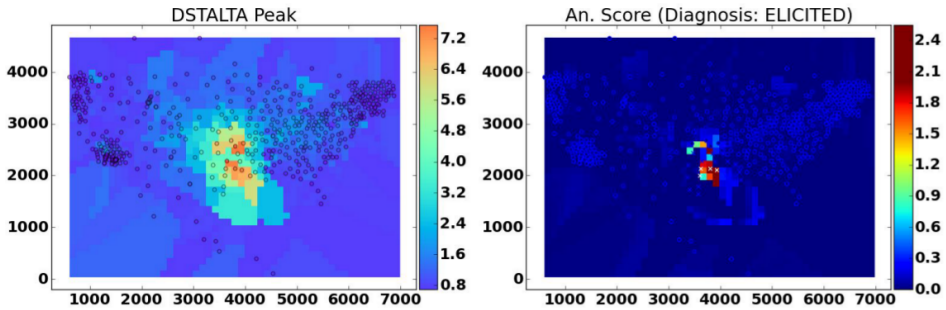


Figure 7.8: An elicited event close to a genuine event can be correctly characterised since the characterisation step assigns the two events to different conflicting groups.

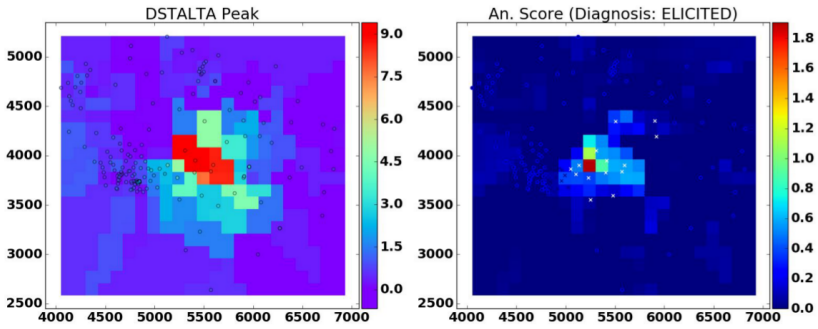


Figure 7.9: A spoofed event, made as a partial genuine event. The cross-scale relationship is not respected.

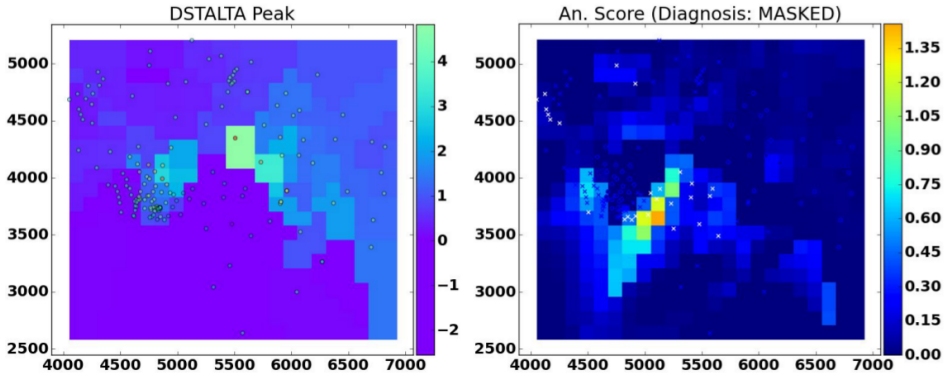


Figure 7.10: A genuine event that is mostly masked introduces a characterisation problem. The characterisation algorithm is not fooled by the presence of event-like peaks and is able to infer that the measurements were masked.

7.3 Robustness of Measurements Inspection Against Worst-Case Attacks

The role of measurements inspection is to force the attacker to a trade-off between the degree of damage and the chance of staying undetected. In other words the goal is to make high-damage attacks detectable and to make undetected attacks cause low damage. While mimicry attacks focus on causing high-damage with a best effort to stay undetected, worst-case attacks explore the whole measurements space to find the best choice both in terms of selected sensors to compromise and of malicious measurements that replace the genuine ones. Given an attacker with fixed capabilities, we seek to infer if at least one attack exists that subverts event detection and misleads anomaly detection, using the attacker's capabilities, which include:

1. Controlling an arbitrary subset of C sensors.
2. At time T, having full knowledge of the genuine measurements at $t < T$.
3. Knowing the event detection function.
4. Knowing the anomaly detection algorithm.

If anomaly detection is successful under these conditions, we are guaranteed that no other attacker with C or less compromised sensors can perform an attack that both triggers or masks an event and stays undetected. Thanks to this knowledge, worst-case attacks provide information about the measurements inspection algorithm's blind spots that an attacker may exploit to evade it. Thus, worst-case attacks seek the worst-case scenarios, as opposed to the best-effort approach of mimicry attacks. The worst-case approach is certainly more complex than the mimicry one. Moreover, as we will see in detail, the time needed to solve a worst-case attack problem can be in the order of hours or days with a general purpose CPU. These times are mostly incompatible with the time dynamics of the measured phenomenon, so we will run offline analyses rather than simulating attackers that seek to craft the measurements in real-time. Still, we will need to approximate the measurements inspection algorithm with a simple

function to achieve a feasible solution time that allows us to run many experiments and extract meaningful statistics about the resilience gains offered by measurements inspection. While this is practicable for the detection algorithm, the algorithms for characterisation and diagnosis are difficult to represent because of their decision process. So, we remark that whilst the worst-case approach is valuable to evaluate the resilience to the most sophisticated attackers, and to uncover the main vulnerabilities of the measurements inspection algorithm, it cannot replace the mimicry approach in the performance evaluation, especially for the characterisation and diagnosis step.

In this section, we consider worst-case attacks for multiple purposes. Firstly, we will retrieve information about the maximum number of malicious sensors that a WSN can tolerate, so that we can give an assurance of the robustness of measurements inspection against malicious data injections, and even offer a fair method to compare different approaches. Moreover, we will use the worst-case approach to study the injection strategies that are more difficult to defend against, by analysing the measurements that optimise the misleading of event-detection whilst staying undetected. This study, in turn, reveals the vulnerabilities that make up most of the WSN risk and if they can be associated with the innate WSN data characteristics or with shortcomings of the anomaly detection algorithm. Finally, we will introduce the first security-driven design methodology in the literature for WSN deployments. Given a maximum number of malicious sensors that can be present in a WSN, we are able to provide at design time how many sensors need to be deployed to guarantee that the anomaly detection algorithm is not evaded. This provides a way to obtain guarantees about the gain in security introduced by anomaly detection. Hence, this method can be used both for designing critical WSNs with strict security requirements and to make a feasibility analysis that states whether it is worth applying anomaly detection given a maximum budget in sensor devices.

We consider the problem where an attacker seeks to compromise the smallest subset of sensors in a WSN that allows to craft malicious measurements capable of triggering or masking events, without being detected by the anomaly detection system. A different approach is considered in [LM05b], where the authors propose an algorithm to reverse engineering linear classifiers, formulating the difficulty of evading the classifier as a complexity problem. The attacker searches

the lowest cost sample (for the attacker) that evades the detection by the linear classifier. In [Nel+10], the authors propose an extension of this framework to the family of convex-inducing classifiers, i.e. classifiers that partition their sample space into two sets, one of which is convex. The model proposed in [Nel+10] also models the optimisation problem considering that the attacker has a limited number of attempts to probe the system, which, depending on the application, is more realistic than the model in [LM05b]. However, both models are limited to the family of classifiers to which they can be applied, whereas in many real applications non-linear detection systems are used. A more general approach is proposed in [Big+13], where a gradient-based strategy can be applied to evade a broad range of machine learning algorithms. More recently, evasion attacks have targeted deep neural networks in computer vision problems, demonstrating the existence of *adversarial examples*, i.e. those images that can be misclassified by deep learning algorithms while being only imperceptibly distorted [Sze+13; GSS15; Pap+16].

In this chapter, the attacker model differs from [LM05b; Nel+10; Big+13] in that we consider that the attacker's cost to minimise is the number of sensors to be compromised in order to evade the anomaly detector. As a consequence, although the values of the measurements for the compromised sensors that evade the system are needed, they do not affect the attacker's cost, as in [LM05b; Nel+10; Big+13]. A second difference is that in our approach, it is not necessary to reverse engineer the anomaly detector, but we look for the blind spots that allow us to craft a malicious sample able to evade the system given a set of compromised sensors and the information about the other measurements in the WSN. A third difference is that we do not restrict the applicability of our work to linear classifiers, even though non-linearities increase the time required to solve the problem.

The algorithm is first run on the novel *wavelet-based* detection algorithm introduced in Chapter 6, which, to our knowledge, gives the best results against sophisticated attacks. Indeed, such an algorithm is able to detect both eliciting and masking attacks in event detection WSNs, even when many sensors act jointly to mimic real event or rest conditions.

Although we assume this algorithm as the target for the evasion task, the same methodology

holds also for other algorithms. In fact, we apply the same method also to another state-of-the-art anomaly detection algorithm which is based on *principal component analysis* [CP07]. This shows that our novel study of worst-case adversaries can also be used to compare the robustness of different anomaly detection algorithms, as we do in Section 7.3.3.

7.3.1 The Worst-Case Attack Problem

From the attacker perspective, the objective is to find the manipulations signals at time instant i, $m_{t_i}(\mathbf{u})$, such that the resulting signal $x_{t_i}(\mathbf{u})+y(\mathbf{u})m_{t_i}(\mathbf{u})$ (which is the observed signal at time instant t_i, i.e. including malicious data injections) evades detection and changes the output of the event detection function.

For convenience of notation, we introduce the row vector $\mathbf{y}$, where $y_i = y(\mathbf{u}_j)$, and $\mathbf{u}_{j:j\in 1,...N}$ is an arbitrary ordering of all the spatial locations $\mathbf{u} \in \Omega$. Likewise, we introduce the $T \times N$ matrices $\mathbf{X}$ and $\mathbf{M}$, where $\mathbf{X}_{ij} = x_{t_i}(\mathbf{u}_j)$ and $\mathbf{M}_{ij} = m_{t_i}(\mathbf{u}_j)$.

The event detection function is assumed to have memory equal to W, meaning that the presence of an event can be determined with the last W collections of measurements. For convenience of notation, we then introduce:

$$\begin{aligned} \mathbf{X}_{w_t} &= \{x_{t_i}(u_j)\} \quad i \in 1,...,W-1 \quad j \in 1,...N \\ \mathbf{M}_{w_t} &= \{m_{t_i}(u_j)\} \quad i \in 1,...,W-1 \quad j \in 1,...N \end{aligned} \tag{7.5}$$

We further assume that the anomaly detection algorithm operates on time snapshots, since the most effective anomaly detection approaches that counteract malicious data injections generally make use of spatial correlation [IL15b]. Even though there is a body of work that exploits the temporal domain, i.e. analyses the measurements time series on single sensors [Sub+06; BHL07; Sun+13], the temporal domain is completely under the attacker's control during the time when the sensor is compromised. As a consequence, unless the attacker is constrained to

a maximum time, there always exists an attack in the time domain which gradually modifies the measurements time series, e.g. to depict the presence of a false event, with the temporal dynamics of a real event. On the contrary, we will focus on the spatial domain, which contains reliable uncorrupted information when a subset of sensors is still genuine.

Since eliciting and masking events requires considerable changes in the measurements, which are likely to disrupt correlations, misleading event detection whilst evading anomaly detection can be modelled as the optimisation problem defined in (7.6), which minimises the number of malicious sensors required by the attacker to subvert the event detection output whilst staying undetected.

$$\min_{y} \sum_{j=1}^{N} (\mathbf{y}_j) \tag{7.6a}$$

$$s.t.$$

$$\mathbf{y}_j \in \{0,1\} \, \forall i \in 1, ..., N \tag{7.6b}$$

$$e(\mathbf{X}_{w_t}) \neq e(\mathbf{X}_{w_t} + \mathbf{1}^T \mathbf{y} \odot \mathbf{M}_{w_t}) \, \forall t \in 1, ..., T - W + 1 \tag{7.6c}$$

$$a(\mathbf{X}_i + \mathbf{y} \odot \mathbf{M}_i) \leq T_d \, \forall i \in 1, ..., T \tag{7.6d}$$

Note that $\mathbf{1}$ is the $1 \times W$ row vector of all ones and $\odot$ indicates the element-wise product. The matrix $\mathbf{X}_{w_t} + \mathbf{1}^T \mathbf{y} \odot \mathbf{M}_{w_t}$ denotes the observed measurements by all sensors (including the malicious ones), at each time instant inside the window w_t, hence it is a $W \times N$ matrix. For convenience, we give an alias to this matrix:

$$\breve{\mathbf{X}}_{w_t} = \mathbf{X}_{w_t} + \mathbf{1}^T \mathbf{y} \odot \mathbf{M}_{w_t} \tag{7.7}$$

Constraint (7.6d) requires this signal to be undetectable at each time instant. This is expressed with the function $a(\mathbb{R}^N)$, which is the output of the detection algorithm and, for the wavelet-based detection, it coincides with the anomaly score, defined in (6.13). The anomaly score is

constrained to be less than the constant T_d for the resulting signal in input. Although in this thesis we will focus on analysing the specific detection algorithm introduced in Chapter 6, this choice suits most detection algorithms, since eventually there is a quantity that is compared with a threshold to decide if the data is anomalous or not.

Constraint (7.6c) requires the resulting signal and the original signal to have a different output for the event detection function $e(\mathbb{R}^W \times \mathbb{R}^N)$, which is supposed to be one if there is an event, zero otherwise[2]. The presence of binary inequalities would force us to choose more complex and burdensome optimisation algorithms, so we first evaluate the value of $e(\mathbf{X}_{w_t})$, to check if the event condition is satisfied or not, and then constrain the absence of events if there is an event, or constrain the presence of events if there is a rest condition. This is summarised by the expression below:

$$\begin{cases} e(\breve{\mathbf{X}}_{w_t}) = 1 \quad \text{if} \quad e(\mathbf{X}_{w_t}) = 0 \\ e(\breve{\mathbf{X}}_{w_t}) = 0 \quad \text{if} \quad e(\mathbf{X}_{w_t}) = 1 \end{cases} \tag{7.8}$$

This formulation is still a problem for common optimisation algorithms because the event detection function has a binary output. However, we can consider $e(\mathbb{R}^W \times \mathbb{R}^N)$ as the *event confidence*, i.e. the confidence whether an event is present, and that an event is detected when this confidence is at least above a threshold T_e, to obtain:

$$\begin{cases} e(\breve{\mathbf{X}}_{w_t}) \geq T_e \quad \text{if} \quad e(\mathbf{X}_{w_t}) < T_e \\ e(\breve{\mathbf{X}}_{w_t}) < T_e \quad \text{if} \quad e(\mathbf{X}_{w_t}) \geq T_e \end{cases} \tag{7.9}$$

Constraints 7.6d and 7.6c will affect the values of the manipulations $\mathbf{M}_{w_t}$. Since these values do not appear in the objective function, the goal of the optimisation is to find, for a fixed vector of compromised sensors $\mathbf{y}$, any combination of $\mathbf{M}_{w_t}$ that satisfies the constraints.

[2]We assume that the potential risk of malicious data injections is just the incorrect event detection. In general, there is also a second risk, which is a wrong characterisation of the events, e.g. incorrect estimation of spread and intensity. This issue can be dealt with by introducing further constraints on the terms $\breve{\mathbf{X}}_{w_t}$, which require the event to have specific characteristics. So, this aspect requires just a refinement of problem (7.6) and we leave this for future work.

The model that we have introduced above describes the worst-case scenario where the attacker is able to keep the degree of anomaly low while introducing a significant change in the event detection function. We have introduced two functions that measure such quantities: the anomaly score and the event score. We have also introduced two thresholds for them: the event score is compared to T_e, which indicates the presence of events, and we assume it to be known thanks to a prior optimisation of the event detection performance. Similarly, we introduced the threshold T_d for the detection of malicious interference, which we assume to be already optimised to guarantee a maximum number of false positives. At this point we can solve the optimisation problem, but, as we will see below, this will require us to make some simplifications to make the problem tractable.

7.3.2 Optimising the Attacker's Problem

Solving Problem (7.6) exactly is intractable, unless the WSN deployment size is in the order of ten sensors. This is a consequence of many factors that we discuss below.

Elements of Complexity

Binary Variables In Problem (7.6), N binary variables appear, i.e. the targeted sensor variables $\mathbf{y}_j$. This is a major problem since the choice of their values is a binary optimisation problem, with $\binom{N}{C}$ possible solutions. It is well known that $\binom{N}{C} \geq (\frac{N}{C})^C$, and since we are interested in values of C that are comparable to N, the number of solutions is generally exponential in N. For example, $C = N/4$ gives $\sqrt{2}^N$ combinations. So, even if Problem (7.6) was efficiently solved, finding a solution would be practical only for small deployment sizes.

Event Score Function The event score function introduces two main challenges:

1. It may be arbitrarily complex. This results in irregularities (e.g. non-linearities, discontinuities, non-smoothness) that require the use of expensive optimisation algorithms, which make several evaluations of the event score function on its domain.

2. The input size of the event score function is WN, whereas the other terms in (7.6) have an input size of N. Consequently, the event score function needs to be simple enough to enable the application of fast optimisation algorithms that can cope also with high values of W and N. In general, event detection algorithms are not expressed in a simple way, as they include decision trees, statistical operations etc. In the following we introduce a simplification method to express the event detection function in terms of sufficient conditions, which enables us to run an efficient optimisation algorithm.

Anomaly Score Function Similarly to the event score function, the anomaly score function may be complex as the result of: thresholds, random choices, differential equations, etc. Unlike event detection, simplifications of the anomaly detection algorithms require a more careful analysis, since they are likely to bias the optimisation problem. Indeed, whilst a simplification of the event detection function generates a proportional inaccuracy in the problem solution, even a small modification to the anomaly detection function may introduce weaknesses that have a considerable impact on the solution. Indeed, the newly introduced weaknesses will likely be exploited by the optimisation algorithm, whose goal is to find the attack that requires less resource.

Nevertheless, simplifications are necessary as complex anomaly detection not only complicates the optimisation, but also causes exploding computational complexity, as its input is composed of unknown variables. In the specific case of the wavelet-based algorithm, the anomaly score includes the evaluation of squared P-order polynomials for the calculation of the higher scales contribution in (6.12). Although evaluating each of them on a known input has an asymptotic complexity $\mathcal{O}(N)$, evaluating them on unknowns requires the calculation of $\binom{N}{2}$ products, which is $\mathcal{O}(N^2)$, with an increase in complexity of a factor N. So, even if anomaly detection is generally not particularly time-consuming, optimising its input may have execution times that are higher by a few orders of magnitude.

Tearing Down the Problem Complexity

For each of the identified issues, we simplify the original problem by using surrogate models (which are computationally cheaper) to approximate the original problem. Afterwards, the solution obtained with the simplified models is tested and adapted for the original problem.

Relaxing Binary Variables The presence of binary variables makes the problem intractable even for low values of N, since it necessitates the exploration of a solution space whose size increases exponentially with N. To explore the solution space of the targeted sensors in polynomial time, there are well-known heuristics [Ber14], which start with the relaxation of the binary variables into continuous variables between zero and one. So, we will follow this approach.

In a second phase, the continuous values are converted into binary, paying attention to the constraints, which will not necessarily be satisfied after the conversion. In this phase, our approach slightly differs from well-known heuristics. Indeed, in our specific problem the objective function includes binary variables only. This feature simplifies finding a solution to the original problem, since we can evaluate a choice of the binary variables by directly checking the value of the objective function.

The idea is to make the conversion by turning C out of N values of $\mathbf{y}_j$ into one and the remainder into zero, considering that the sensors that are mainly exploited by the attacker are those with higher difference between genuine and malicious values. Denoting with $\mathbf{y}'$ and $\mathbf{M}'_i$ the targeted sensors and manipulations that are the solution of the relaxed problem, the criterion that identifies a possible subset of sensors to compromise is summarised below.

Algorithm 17 Binary Constraint Relaxation

Require: $\mathbf{y}'$, $\mathbf{M}'_i$, C
Ensure: $\mathbf{y}$
1: $\text{sortInd} = \text{argsort}(\mathbf{y}' \odot \mathbf{M}'_i)$
2: $\mathbf{y}_j = 0 \quad \forall j \in \text{sortInd}(1), ..., \text{sortInd}(N - C)$
3: $\mathbf{y}_j = 1 \quad \forall j \in \text{sortInd}(N - C + 1), ..., \text{sortInd}(N)$

This can be done when the event detection algorithm has a memory equal to one, which allows

to consider each time instant i separately. To extend this procedure to the case where the event detection memory is greater than one, we use the average of $\mathbf{y}' \odot \mathbf{M}'_i$ for all the values of i within the time window. This value identifies how much each sensor is exploited by the attacker on average.

The value of C is the minimum number that allows to respect the constraints, which is unknown. Hence, we will test all values of C in the interval $[\sum_{j=1}^{N}(y'_j), \sum_{\mathbf{y}'_j>0}]$. The left bound comes from the observation that the solution for the original problem cannot be better than the solution for the relaxed one. The right bound comes from the observation that with $C = \sum_{\mathbf{y}'_j>0}$, there is certainly a feasible solution, which is:

$$\mathbf{y}_j = \begin{cases} 1 & \text{if } \mathbf{y}'_j > 0 \\ 0 & \text{if } \mathbf{y}'_j = 0 \end{cases} \tag{7.10}$$

It is then sufficient to set the values of $\mathbf{M}_i$ to $\mathbf{y}' \odot \mathbf{M}'_i$, to get a solution for the original problem, in particular with $C = \sum_{\mathbf{y}'_j>0}$.

In summary, we select a subset of sensors for which $\mathbf{y}'_j$ is not zero, we set them equal to one, and check if they enable the adversary to change the output of event detection whilst evading anomaly detection. This is an unknown value, but we are guaranteed that the problem is well-posed because (7.10) is a feasible solution. To test if a combination of $\mathbf{y}_j$ is valid, we check if a combination of $\mathbf{M}_i$ exists that satisfies the constraints. In that case, $\mathbf{y}_j$ and $\mathbf{M}_i$ are a solution for the original problem, with objective function equal to $\sum_{j=1}^{N} \mathbf{y}_j$. The basic idea is similar to that of the *Relaxation Induced Neighborhood Search* (RINS) heuristic [DRP05], since a binary variable is zero if its correspondent in the relaxed solution is zero. However, RINS ultimately runs a simplified binary optimisation, while we simplify the problem until all variables are continuous. Thus, we do not run any binary optimisation.

Linearising the Event Score Function An event score function evaluates the compliance of the measurements with a set of features that describe the common event patterns, which are

defined both in the temporal and in the spatial domain. Generally, there may be many criteria that are sufficient to infer that there is an event, e.g. relative change of the measurements in time and absolute values. This makes the event detection a logical OR of a function applied to the measurements. For instance, in Section 7.3.3 we show that an event may be detected when at least one sensor reading is above a threshold. Another example is given in Section 7.3.3, where an event is detected if the variability in at least one neighbourhood of sensors is above the values that normally apply to rest conditions.

Hence, to reduce the computational burden introduced by the event detection function and still cover a variety of event detection criteria, we transform the event detection constraint into the logical OR of Z inequalities (with Z equal to the number of sufficient conditions for identifying an event) between a linear combination of the measurements, in both the temporal and spatial domain, and a constant threshold:

$$\begin{aligned} e(\breve{\mathbf{X}}_{w_t}) \geq T_e \approx \\ A^{(1)}\breve{\mathbf{X}}_{w_t}B^{(1)} \geq T_{e_1} \vee A^{(2)}\breve{\mathbf{X}}_{w_t}B^{(2)} \geq T_{e_2} \vee ... \vee A^{(Z)}\breve{\mathbf{X}}_{w_t}B^{(Z)} \geq T_{e_Z} \end{aligned} \tag{7.11}$$

Where the terms $A^{(1)}, ..., A^{(Z)}$ and $B^{(1)}, ..., B^{(Z)}$ are respectively $1 \times W$ and $N \times 1$ matrices of real coefficients. The A matrices represent the temporal analysis of the event detection function (e.g. change points), whereas the B matrices analyse the spatial domain (e.g. event diffusion pattern). Their multiplication with the $W \times N$ matrix $\breve{\mathbf{X}}_{w_t}$ returns a scalar, which is compared with the thresholds $T_{e_1}, ..., T_{e_Z}$.

In eliciting scenarios, (7.6) becomes a series of Z optimisation problems because each problem will try to satisfy one of the Z sufficient event conditions. The best solution among all of them is the best general solution. On the contrary, in masking scenarios constraint (7.6c) of optimisation problem (7.6) becomes a set of Z constraints, and all of them need to be satisfied as usual, leading to a logical AND. Indeed, none of the sufficient event conditions should be satisfied to infer that there is no event.

Linearising the Anomaly Score Function The anomaly score function in (6.13) involves two nonlinear terms: the high scales contribution $I(\boldsymbol{\tau})$ and the function q that takes it as input and returns the maximum accepted low scale coefficient.

The term $I(\boldsymbol{\tau})$ involves the computation of $n_s n_{\boldsymbol{\tau}}^2$ squares of N-order polynomials, where n_s and $n_{\boldsymbol{\tau}}$ are, respectively, the number of scales and translations. This has an asymptotic complexity $\mathcal{O}(n_s n_{\boldsymbol{\tau}}^2 N^2)$. Computing $I(\boldsymbol{\tau})$ exactly becomes intractable for values of N in the order of hundreds. However, the squares of the N-order polynomials can be replaced with the linear approximation given by the product of the polynomial with its constant term, making problem (7.6) more scalable with the number of sensors.

The constant term includes the wavelet transform of the original measurements signal, while the variables contain the manipulations, i.e the difference between the corrupted and original signal. So the energy of the manipulations signal is the difference between the energies of the signal corrupted by malicious interference and of the original signal. Because of the nature of the problem, the energy of such a difference is smaller than the energy of the original signal: otherwise anomalies would be evident. Hence, the most relevant terms in the square of the corrupted signal are those that are multiplied by the constant term.

The function q can also be linearised to obtain an entirely linear approximation for the anomaly score. Since the input and output of function q in (6.13) are in a monotonic relationship (larger high scales coefficients are usually matched by larger low scale coefficients), standard linearisation techniques, such as linear regression [GH06], give accurate results.

Note that the evasion requirement is not guaranteed with the linearised anomaly score, since it may also give lower estimates. This requires us to validate a solution found with the linearised anomaly score against the actual anomaly detection algorithm, before accepting it.

Transforming Bilinear into Linear Terms After linearising the anomaly score function, the only term that makes the optimisation problem non-linear is the presence of the bilinear terms $\mathbf{y}_j \mathbf{M}_{ij}$. Bilinear problems can be transformed into linear problems with some loss in optimality.

This is achieved by introducing the dummy variables:

$$\mathbf{W}_{ij} = \mathbf{y}_j \mathbf{M}_{ij} \quad i \in 1, ..., T \quad j \in 1, ...N \tag{7.12}$$

Rather than adding the equation above as a constraint, which would still be bilinear, we use the *McCormick envelopes* [McC76], which are known to well approximate (7.12).

$$\begin{aligned}
\mathbf{W}_{ij} &\geq \mathbf{M}_{ij}^{inf} \mathbf{y}_j \\
\mathbf{W}_{ij} &\geq \mathbf{M}_{ij}^{sup} \mathbf{y}_j + \mathbf{M}_{ij} - \mathbf{M}_{ij}^{sup} \\
\mathbf{W}_{ij} &\leq \mathbf{M}_{ij}^{sup} \mathbf{y}_j \\
\mathbf{W}_{ij} &\leq \mathbf{M}_{ij} + \mathbf{M}_{ij}^{inf} \mathbf{y}_j - \mathbf{M}_{ij}^{inf} \\
\mathbf{M}_{ij}^{inf} &\leq \mathbf{M}_{ij} \leq \mathbf{M}_{ij}^{sup} \\
\mathbf{M}_{ij}^{inf} &\leq \mathbf{W}_{ij} \leq \mathbf{M}_{ij}^{sup}
\end{aligned} \tag{7.13}$$

The McCormick constraints are introduced for all $j \in 1, ..., N$, giving a total of $8N$ additional constraints per each time instant. The result is the surrogate problem (7.14), which replaces the variables as in (7.12), and whose minimum is an overestimation for the original problem.

$$\min_{y} \sum_{j=1}^{N} (\mathbf{y}_j) \tag{7.14a}$$

$$s.t.$$

$$\mathbf{y}_j \in \{0, 1\} \, \forall j \in 1, ..., N \tag{7.14b}$$

$$e(\mathbf{X}_{w_t}) \neq e(\mathbf{X}_{w_t} \pm \mathbf{W}_{w_t}) \, \forall t \in 1, ..., T - W + 1 \tag{7.14c}$$

$$a(\mathbf{X}_i \pm \mathbf{W}_i) \leq T_d \, \forall i \in 1, ..., T \tag{7.14d}$$

Note that the McCormick constraints require the variable $\mathbf{M}_{ij}$ to be within $[\mathbf{M}_{ij}^{inf}, \mathbf{M}_{ij}^{sup}]$, and

$\mathbf{M}_{ij}^{inf}$ is required to be equal or greater than zero. For this reason (7.14) introduces a slight modification to (7.6), which considers that the manipulations can also be negative, i.e. the attacker can also decrease the measurements. In particular, since events are assumed to cause the measurements to increase, the manipulation will be summed for eliciting attacks and subtracted for masking attacks.

With such a modification, $\mathbf{M}_{ij}$ becomes the magnitude of the manipulations, therefore $\mathbf{M}_{ij}^{inf}$ can always be set to 0 (which corresponds to no malicious manipulation), while $\mathbf{M}_{ij}^{sup}$ can be set to a positive value. It is also possible to set it to the measurements range size. However, since the time needed to solve the problem increases with the variables range size, it may be worthwhile to find a reasonable value for $\mathbf{M}_{ij}^{sup}$ by starting with a small value and increasing it until the problem has a feasible solution. Note that there is always at least one feasible solution, which is $N = C$, since a genuine rest condition should always be allowed to transition to a genuine event and vice versa.

Solving Algorithm

To solve problem (7.14), we use Algorithm 18: *findAttackVector*. This builds the optimisation problems that are involved in the eliciting/masking attack and collects the results.

In a first step, it builds the optimisation problem with the constraints described in the previous sections. These include the relaxed domain constraints for the targeted sensors variables at line 4, the McCormick constraints at line 6, and the undetectability constraints at line 19. Note that after replacing the bilinear variables $\mathbf{y}_j\mathbf{M}_{ij}$ with the dummy variables $\mathbf{W}_{ij}$, the variables $\mathbf{y}_j$ and $\mathbf{M}_{ij}$ are present only in McCormick constraints. So, the problem will be optimised with respect to the dummy variables $\mathbf{W}_{ij}$, while the original variables will be set to satisfy the McCormick constraints. This part is common for both eliciting and masking scenarios, which are considered separately at lines 22 and 30 respectively.

At lines 25 and 34, *findAttackVector* passes the optimisation problem that it has built to Algorithm 19: *solveRelaxedAndCheck*. Since eliciting problems are solved as the best solution

Algorithm 18 *findAttackVector*

Require: X_{w_t}
Ensure: C_{min}, $\mathbf{M}^*$, $\mathbf{y}^*$
elicit = $e(X_{w_t}) < T_e$
$\mathcal{P}^r$ = new optimisation problem {Initialise a new optimisation problem.}
for $j \in 1, ..., N$ **do**
  Add constraint to $\mathcal{P}^r$: $y_j \in [0, 1]$ {Binary constraints relaxation}
  for $i \in 1, ..., W$ **do**
    Add constraint to $\mathcal{P}^r$: McCormick constraints (7.13). Replace all occurrences of $\mathbf{y}_j\mathbf{M}_{ij}$ in $\mathcal{P}^r$ with $\mathbf{W}_{ij}$ {Bilinear constraints relaxation}
  end for
end for
if elicit **then**
  $\breve{\mathbf{X}}_{w_t} = X_{w_t} + \mathbf{W}$
else
  $\breve{\mathbf{X}}_{w_t} = X_{w_t} - \mathbf{W}$
end if
for $\boldsymbol{\tau} \in \Omega$ **do**
  $D(\boldsymbol{\tau}) = (T^{wav}S)(s_0, \boldsymbol{\tau})$
  $I(\boldsymbol{\tau}) = \sum_{\boldsymbol{t}} \sum_{s \in hs,...,Hs} (T^{wav}_{ns} \breve{\mathbf{X}}_{w_t})^2(s, \boldsymbol{\tau}) m(s, \boldsymbol{t} - \boldsymbol{\tau})$
  $a(\breve{\mathbf{X}}_{w_t}, \boldsymbol{\tau}) = q(I(\boldsymbol{\tau}))/D(\boldsymbol{\tau})$
  $a^l(\breve{\mathbf{X}}_{w_t}, \boldsymbol{\tau})$ = Linearise $(a(\breve{\mathbf{X}}_{w_t}, \boldsymbol{\tau}))$
  Add constraint to $\mathcal{P}^r$: $a^l(\breve{\mathbf{X}}_{w_t}, \boldsymbol{\tau}) < T_d$ {Linearised Evasion constraints}
end for
$C_{min} = N$, $\mathbf{M}^* = \{0, ..., 0\}$, $\mathbf{y}^* = \{0, ..., 0\}$
if elicit **then**
  for $z \in 1, ..., Z$ **do**
    Add constraint to $\mathcal{P}^r$: $A^{(z)} \breve{\mathbf{X}}_{w_t} B^{(z)} \geq T_e$ {Eliciting constraint}
    C',$\mathbf{M}'$,$\mathbf{y}'$ = *solveRelaxedAndCheck*($\mathcal{P}^r$,C_{min}) {Eliciting problem built. Now solve it.}
    if $C' < C_{min}$ **then**
      $C_{min} = C'$, $\mathbf{M}^* = \mathbf{M}'$, $\mathbf{y}^* = \mathbf{y}'$
    end if
  end for
else {Masking Scenario}
  for $z \in 1, ..., Z$ **do**
    Add constraint to $\mathcal{P}^r$: $A^{(z)} \breve{\mathbf{X}}_{w_t} B^{(z)} \leq T_e$ {Masking constraints}
  end for
  C',$\mathbf{M}'$,$\mathbf{y}'$ = *solveRelaxedAndCheck*($\mathcal{P}^r$,C_{min}) {Maskng problem built. Now solve it.}
  if $C' < C_{min}$ **then**
    $C_{min} = C'$, $\mathbf{M}^* = \mathbf{M}'$, $\mathbf{y}^* = \mathbf{y}'$
  end if
end if
return C_{min}, $\mathbf{M}^*$, $\mathbf{y}^*$

Algorithm 19 *solveRelaxedAndCheck*

Require: $\mathcal{P}^r$,C_{min}
Ensure: C', $\mathbf{M}'$, $\mathbf{y}'$
1: $C' = C_{min}$, $\mathbf{M}^* = \{0, ..., 0\}$, $\mathbf{y}' = \{0, ..., 0\}$
2: Solve $\mathcal{P}^r$ and get the relaxed solution $\mathbf{M}^r$, $\mathbf{y}^r$
3: $\text{obj}^r = \sum_{j \in 1,...,N} \mathbf{y}^r_j$
4: **for** $C \in \text{obj}^r, ..., C_{min}$ **do**
5: $\mathcal{P}' =$ copy of $\mathcal{P}^r$
6: $\mathbf{y}' = \mathbf{y}^r$
7: sortInd = $\text{argsort}(\mathbf{y} \odot \mathbf{M}_i)$
8: $\mathbf{y}'_i = 0 \quad \forall i \in \text{sortInd}(1), ..., \text{sortInd}(N - C)$
9: $\mathbf{y}'_i = 1 \quad \forall i \in \text{sortInd}(N - C + 1), ..., \text{sortInd}(N)$
10: Solve $\mathcal{P}'$ and get the solution $\mathbf{M}'$
11: **if** $\mathbf{M}'$ and $\mathbf{y}'$ respect the non-linearised undetectability constraint $a(\breve{\mathbf{X}}_{w_t}) < T_d$ **then**
12: $C' = C$, $\mathbf{M}' = \mathbf{M}^r$, $\mathbf{y}' = \mathbf{y}^r$
13: **end if**
14: **end for**
15: **return** C', $\mathbf{M}'$, $\mathbf{y}'$

among the Z optimisation problems that arise by enabling the sufficient event conditions one by one, *solveRelaxedAndCheck* is run Z times between lines 22-29, and the eliciting solution is the best among all of them. Masking problems, instead, are solved as the solution to the problem that rejects all the Z sufficient event conditions. So, masking scenarios require only one call to Algorithm 19, which solves the problem built between lines 30-37. This problem contains Z event-detection constraints that all require the event conditions not to be satisfied.

Specifically, *solveRelaxedAndCheck* firstly solves the relaxed problem with continuous values for $\mathbf{y}_j$ (line 3). Because of the relaxation, the solution gives a lower bound to the minimum number of malicious sensors needed. Hence, in a second phase, it sets to 1 the $\mathbf{y}_j$ that produce the highest C values of $\mathbf{y}_j\mathbf{M}_{ij}$, which corresponds to the C compromised sensors that are mostly used. The remainder are set to 0 and the new problem, where the binary variables have fixed values, is solved (lines 4-10) to find the values of $\mathbf{M}_{ij}$ that satisfy the constraints, if any.

If a solution is found, it is a solution also for the original problem (7.14). However, the latter includes a linear approximation of the real anomaly detection algorithm, which is the one that should be evaded. For this reason, the solution is considered only if it achieves evasion on the anomaly detector (lines 11 -12).

This algorithm was implemented using the *Gurobi* solver [GO15] as a back-end for the optimisation of the basic problems. Gurobi is an efficient solver for linear and (convex) quadratic optimisation. However, *findAttackVector* requires the optimisation of several problems and the construction of complex constraints which become especially burdensome with high values of N. To point out the problem's complexity, we report in Table 7.3 the execution times in minutes of the *findAttackVector* function, executed by an Intel®Core™ i7-2600 CPU @ 3.40GHz. Such times are distinguished according to the type of attack (eliciting/masking) and by number of sensors N. Each time is the average obtained for 20 measurements sets, generated randomly with the method described in Section 7.3.3. The times refer to a single experiment, which is a particular eliciting or masking scenario, within a specific WSN deployment, and with a specific set of original measurements.

Table 7.3: Execution times for a single run of *findAttackVector*, in minutes.

Attack \ N	10	30	60	90	100	200
Eliciting	6.5	15	52.9	50.6	75	99.6
Masking	3.5	4.5	5.4	6.6	7.66	14.8

Note that the eliciting problem is more time-consuming because it requires us to run Algorithm *solveRelaxedAndCheck* Z times and return the best result. The masking problem is less burdensome since, instead of Z optimisation problems, there are Z event detection constraints, but Algorithm *solveRelaxedAndCheck* is run only once per experiment.

The times in Table 7.3 start to be high when we run each experiment several times, and consider more experiment scenarios. With 20 samples per experiment, which is the case for our evaluations, solving all the problems took 114 hours. For comparison, we have also run the experiments by removing, one-by-one, the simplifications that we have introduced, and observed that the algorithm could not find any solution within a two-weeks period. This confirms that the approximations made in Section 7.3.2 are necessary to make the problem tractable, even though the calculations are run offline at deployment time. Note that parallel computing would not bring significant benefits into the problem, since the constraint of respecting the inter-sensor correlations would require the different parts to frequently communicate with each other, which results into a loss of the main advantages of parallelism.

7.3.3 Evaluating the Resilience of Wireless Sensor Networks to Worst-Case Attacks

With the worst-case attack problem well defined and made tractable, we can now find the attack vector for a specific WSN, defined as the sensors to compromise and the measurements to be injected into those compromised sensors that achieve evasion and change the event detection output. In the following, we present the results of the application of the solving algorithm and discuss them to understand risks and advantages of measurements inspection algorithms, as well as of the event detection algorithm and of WSNs in general.

The analysis requires a WSN with well-defined deployment and the measurements from all sensors collected at different times. However, the results for a specific deployment and for fixed characteristics of the physical phenomenon do not enable us to draw generic conclusions about the damage that can be caused by an attack, and the weak spots that may be exploited.

Hence, our approach is to fix only a WSN deployment area, i.e. the area where sensor nodes can be installed. Within such an area, we identify multiple deployment configurations, with different sensor densities, to evaluate the impact of the increase in inter-sensor correlation which follows from increases in the sensor density. To also take into account the variability that exists between different rest conditions and events, we generate 20 rest conditions, for which the eliciting attack vector is calculated and 20 event scenarios, for which the masking attack vector is calculated. Each scenario is made of 100 temporal samples to study the response in time of both the event detection and the measurements inspection algorithm. Scenarios differ from each other in the shape of the deployment and in the original measurements, which are not under the control of the attacker.

After running the optimisation algorithm for each configuration of the aforementioned elements, we take the average of all the minimum numbers of malicious sensors that are needed to subvert event detection across the 20 different scenarios. In this way, we have reliable and contextualised information about the degree of resilience of a WSN.

In principle, different configurations of sensors might be deployed to compare the different

degrees of resilience that can be obtained. However, one can exploit the data collected with a specific deployment and interpolate to approximate what the measurement perceived by a sensor at a generic location would be.

Other than by interpolation, the measurements can be produced by a model, i.e. a physical model of the monitored phenomenon, which works as a simulator of measurements that may be perceived. This approach also makes the data collection step easier as it can generate an arbitrary set of event scenarios, which may be rare. We introduce below the model-generated measurements for wildfire monitoring WSNs as a case study.

Case Study: Wildfire Monitoring WSNs

Wildfire monitoring WSNs are well suited for the evaluation of measurements inspection techniques. Indeed, they observe events, i.e. fires, which have non-trivial diffusion patterns in space as a fire affects the sensors in a different and unpredictable way. Moreover, the deployments can cover wide areas, so we can run a comprehensive analysis about the impact of the deployment size on the performance of anomaly detection.

The data measurements being used below are model-generated temperatures produced with the wildfire model described in Section 7.2.5. We consider a set of N sensors randomly scattered in an area of 100×100 m^2, hence the deployment area's size is 10000m^2. Consequently, the average sensor density is well defined and it is equal to:

$$\rho_N = D_A/N \frac{1}{\mathrm{m}^2} \tag{7.15}$$

where D_A is the deployment area, i.e. $D_A = 10000\mathrm{m}^2$ in our analyses. Although real deployments may be considerably larger, e.g. in the case of forests, within the chosen deployment size we can reasonably rely on measurements inspection only. Within larger deployment areas, in the order of 10 hectares, we need to complement measurements inspection with other techniques, such as software attestation, as discussed in Chapter 8.

The quantity ρ_N, rather than N, is what ultimately matters for the performance of event

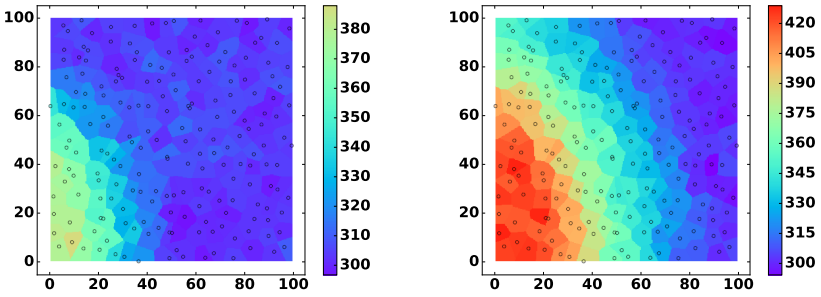
detection and anomaly detection, since it tells how fine-grained is the information gathered about the physical phenomenon. Nevertheless, since the deployment area is well defined, we will refer to the number of sensors rather than their density, since it is easier to visualise and to interpret.

The time evolution of the fires is modelled with a linear interpolation of both the black bodies temperatures and their radius. An average spread ratio of $0.4m/s$ is reasonable for typical environmental parameters (such as slope of the terrain, wind speed and grass type) [NHS08]. Existing simulation environments such as *FARSITE* – developed by the U.S. Forest Service [Fin+98] – are more accurate and consider different combinations of the variables involved, including wind, fuel, terrain slope etc. However, these simulators focus on the wildfire's spreading behaviour and do not provide the spatio-temporal distribution of temperatures.

An example of the resulting temperature evolution in time is shown in Fig. 7.11. The fire originates at the bottom left corner of the WSN space and spreads across the deployment area. If there is no intervention, the fire is supposed to proceed undisturbed and may also cover the whole WSN deployment space, as shown in Fig. 7.11d.

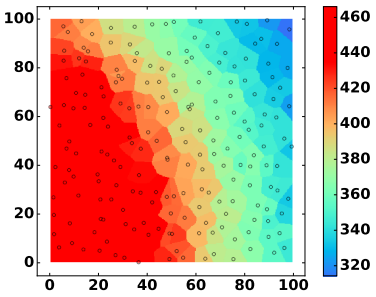
Note that the event evolution has both a temporal and a spatial effect. The temporal effect is the measurements increase in time, which is linked to the fact that the fire is getting more powerful and temperatures increase as a consequence (after a certain point they saturate). The spatial effect is in the spreading of the fire area: the temperatures of more and more sensors are affected by the fire. This effect has an important impact on the feasibility of masking attacks: indeed, if the event can eventually be perceived throughout the whole WSN, the attacker needs to compromise a number of sensors very close to N to prevent event detection. For this reason, rather than masking an event for its whole duration, an attacker may seek to highly delay the detection time instead.

Hence, in the following experiments, the constraints that impose the change of the event detection output are considered only for the first 6 minutes since the fire outbreak when, as shown in Fig. 7.11b, the fire can already be well spread. Firstly, we will analyse the resilience gains offered by measurements inspection when considering a simplistic event detection algorithm

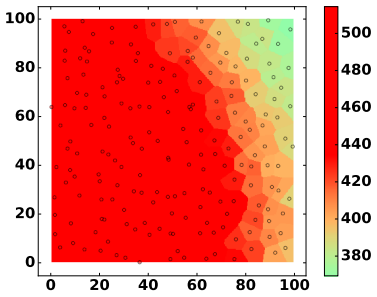


(a) t = 5 min

(b) t = 6 min



(c) t = 7 min



(d) t = 8 min

Figure 7.11: Example of fire evolution in time. The WSN area is a two-dimensional space. The circles identify the sensors and colours map the temperatures, in Kelvin. Note that the sensors are not placed in a grid but randomly scattered.

that does not exploit spatial correlation. After that, we will introduce a more suitable event detection algorithm and show that the resilience gains can be dramatically improved if the event detection algorithm is well designed.

Simplistic Event Detection Algorithm

The performance of measurements inspection is dependent on the event detection algorithm since it determines how much effort is needed by the malicious sensors to subvert the event detection result. If this effort is minimal, then even the best measurements inspection algorithm would have poor performance.

To clarify this point, we consider an oversimple event detection algorithm for the wildfire monitoring WSN, which is also the same algorithm usually adopted by temperature sensors for home fire alarms, i.e. :

$$(\breve{\mathbf{X}}_{i,1} \geq T_e) \vee (\breve{\mathbf{X}}_{i,2} \geq T_e), , ..., \vee (\breve{\mathbf{X}}_{i,N} \geq T_e) \tag{7.16}$$

That corresponds to triggering a fire alarm if any of the N temperatures is above T_e, usually set to $60°C$ (corresponding to 333.15 Kelvin). This choice makes it particularly easy to spoof an event, since only one malicious sensor needs to focus on subverting event detection. The remainder of malicious sensors can collude in endorsing its measurements and making them undetectable. This is confirmed by the results obtained and shown in Table 7.4.

Table 7.4: Minimum number of malicious sensors for both eliciting and masking attacks, with different sensors number (and density in turn). The eliciting attacks require a small set of sensors to be compromised due to the event detection algorithm, which requires just one sensor to trigger.

Attack \ N	10	30	60	90	100	200
Eliciting	3	3	5	7	7	7
Masking	4	11	21	33	34	81

The results with eliciting attacks above are obtained by averaging the output of Algorithm 18 using 20 different scenarios with no fire as input. Here the real measurements are around $293.15K$ ($20°C$). The results with masking attacks are obtained by averaging the output of Algorithm 18 using 20 randomly generated event scenarios as input. Here, a fire starts immediately spreading until it saturates and can be perceived throughout the whole WSN (such as in Fig. 7.11d). However, the attack is judged successful if no fire is detected for the first 6 minutes (an example of fire spread at 6 minutes is the one in Fig. 7.11b). Generally, at this stage the fire is above the $60°C$ for about 25% sensors, while the measurements inspection algorithm requires the attacker to compromise around 37% sensors to mask the event whilst staying undetected, with a gain of about 50%.

There are two main reasons behind the performance against eliciting attacks, both connected to the event detection algorithm. The first is that one single sensor is needed to spoof an event, hence the extra 2-4 sensors that are needed to avoid detection by the measurements inspection algorithm are actually a 200%-400% gain. The second reason is that it is not unlikely for temperatures to jump from 20 to $60°C$ in a few metres, since the temperatures may show a

large variation in the presence of a fire, thus the attacker needs to compromise a small area, i.e. a few sensors.

The goal of the algorithm that detects malicious data is analogous to charactering the admissible ranges, as a function of the measurements collected. This must be done under all possible genuine circumstances, including the presence of any event of interest. The inaccuracy of anomaly detection, and the need to cater for all the possible scenarios introduce uncertainty in the characterisation of admissible measurements. Therefore, even with 200 sensors, 7 malicious sensors are enough to introduce a variation of $40°C$ and change a $20°C$ measurement into $60°C$.

The detection of malicious data is an ill-posed problem in this case, because we are giving to a single sensor the power to decide on the event's presence, but, in practice, more sensors share information about the same event. This is also shown in the masking results in Table 7.4, where the number of sensors needed to mask the event defined in (7.16) is almost one order of magnitude bigger than in the eliciting case.

This happens because, with such event detection, all the sensors with temperature greater than T_e need to be compromised and report a temperature less than T_e. This implies both that C must be greater than the number of sensors with temperature higher than T_e, and that the constraints introduced by spatial correlation are respected by more sensors.

From this simple experiment, we conclude that **the event detection algorithm highly affects the chance of detection**. In particular, if it involves few sensors, spoofing and eliciting events become simple. The detection of spoofed events would benefit from an event detection algorithm that is as strict as possible: i.e. that capture as many features of the event as possible. However, this choice simplifies the masking task, since it is sufficient to mask just one of the features required to prevent event detection.

Ultimately, the best choice of event detection algorithm, from a security perspective, is a trade-off between the thorough characterisation of an event's characteristic and the characterisation of its most relevant features. This trade-off is governed by the eliciting and masking risks, which differ especially because they cause different types of damage.

Event Detection Algorithm Improvements

In this section, we consider a more sophisticated event detection algorithm that makes use of both temporal and spatial correlations to identify changes in the measurements distribution and verify that the change is experienced by multiple sensors. To improve the accuracy of event detection, some techniques also make a comparison between different *attributes*, e.g. temperature and humidity [SNQ12; SLM13], in conjunction with analyses in the spatial and/or temporal domain. Even though our approach is compatible with these event detection algorithms, we do not analyse them in detail since it is the event information captured in the spatial domain that ultimately matters in discriminating genuine from malicious sensors, as conveyed in Section 4.

The event detection criterion presented here is based on the observation that the sensed measurements follow different distributions in the absence and in the presence of events. This holds true especially in the temporal domain as the physical phenomena induce higher measurements variability in the presence of events. Changes in variability are likely perceived by more sensors, as they are all perceiving the same phenomenon. Hence, the presence of events can be verified by testing for increases in the average variability within a neighbourhood:

$$\frac{1}{N(j)} \sum_{N(j)} \delta^{W}_{\breve{\mathbf{X}}_{ij}} > T_e \tag{7.17}$$

In (7.17) $\delta^{W}_{\breve{\mathbf{X}}_{ij}}$ is the variability of the measurements within a W-wide time window $[t_{i-W}, t_{i-1}]$, perceived by a generic sensor j. $N(j)$ indicates its neighbourhood. The variability of the measurements is averaged across the neighbourhood to exploit the spatial correlation properties of events.

The variability of a signal is generally measured with the *variance*, which would be calculated in a W-wide time window as expressed below.

$$\delta^{W}_{\breve{\mathbf{X}}_{ij}} = \mathrm{Var}(\{\breve{\mathbf{X}}_{kj}\}_{k=i-W}^{i-1}) \tag{7.18}$$

The non-trivial problem with the use of variance in the event detection algorithm is the calculation of the variance itself. The most widely adopted algorithm is the *unbiased sample variance*, which is defined as:

$$\frac{1}{W-1}\sum_{i=1}^{W}(\breve{\mathbf{X}}_{ij}-\overline{\breve{\mathbf{X}}_j})^2 \tag{7.19}$$

where:

$$\overline{\breve{\mathbf{X}}_j}=\frac{1}{W}\sum_{i=1}^{W}(\breve{\mathbf{X}}_{ij}) \tag{7.20}$$

When the values of $\breve{\mathbf{X}}_{ij}$ are known, evaluating this formula has a complexity $\mathcal{O}(W)$, but when they are unknown, the complexity becomes $\mathcal{O}(W^3)$ because it involves W squares of the sum of $W+1$ monomials. For large values of W, problems with such event detection algorithm have an infeasible computational complexity just for defining the event detection equations.

For this reason, rather than (7.19), we use an approximation [ZWC12], which is:

$$\frac{1}{W^2}\sum_{i=1}^{W}\sum_{k=i+1}^{W}(\breve{\mathbf{X}}_{ij}-\breve{\mathbf{X}}_{kj})^2 \tag{7.21}$$

The advantage of this approximation is that it changes the number of monomials from $W+1$ to 2. Hence, the complexity for evaluating this formula when $\mathbf{y}_j$ are unknowns is $\mathcal{O}(W^2)$ rather than $\mathcal{O}(W^3)$.

To further reduce the complexity introduced by the variance-based event detection equations, another approximation of the variance can be used, which is made up of just W polynomials with 2 monomials, reaching a complexity of evaluating the formula of $\mathcal{O}(W)$. Such an approximation is:

$$\frac{1}{2W}\sum_{i=1}^{W}(\breve{\mathbf{X}}_{i+1\,j}-\breve{\mathbf{X}}_{ij})^2 \tag{7.22}$$

Note that constraints of the kind $\frac{1}{2W}\sum_{i=1}^{W}(\breve{\mathbf{X}}_{i+1\,j} - \breve{\mathbf{X}}_{ij})^2 \geq T_e$, which would be introduced for eliciting problems, make the problem non-convex. While convex problems can be solved in polynomial time, non-convex problems are known to be $\mathcal{NP}$-hard. To avoid this problem, we can remove the square and obtain:

$$\frac{1}{2W}\sum_{i=1}^{W}\breve{\mathbf{X}}_{i+W\,j} - \breve{\mathbf{X}}_{ij} \tag{7.23}$$

The value returned by (7.23) may sensibly differ from the real variance, but still evaluates the data variability in time. Note that since the differences are not squared, the value may be either positive or negative. In wildfire monitoring applications we expect this value to be positive when an event manifests, since the measurements will generally increase as a fire breaks out, and become negative when it starts to be extinguished. Hence, we build the new event detection algorithm by comparing the variability measure (7.23), averaged across a neighbourhood, to a positive threshold. Ultimately, our new event detection criterion is:

$$\frac{1}{N(j)}\sum_{N(j)}\frac{1}{2W}\sum_{i=1}^{W}\breve{\mathbf{X}}_{i+W\,j} - \breve{\mathbf{X}}_{ij} > T_e \tag{7.24}$$

The results with the new event detection algorithm are shown in Table 7.5. Note that the masking results have increased by 3.5 times on average, while the eliciting results have increased by 8.7 times on average. As a consequence, the gap between eliciting and masking results has been reduced too. Masking attacks still find it harder to evade anomaly detection because the gap between the events used for masking and rest conditions is higher than the gap between rest conditions used for eliciting and event conditions.

Table 7.5: Minimum number of malicious sensors. The improvement in the event detection algorithm increased the resiliency, especially against eliciting attacks.

Attack \ N	10	30	60	90	100	200
Eliciting	6	14	25	33	33	66
Masking	6	20	53	75	80	162

Note that since we defined the neighbourhood of a sensor as the sensors whose distance is less

than 80 m, with a maximum of 3 sensors an attacker can influence all the neighbourhoods in the network. Indeed, this is the minimum number of circles with radius equal to 80 needed to cover a square area with size 100. This means that with just 3 sensors an attacker can spoof or mask any event if no anomaly detection is in place[3], and that if we subtract 3 from the results in Table 7.5 we obtain the gain in resilience brought in by the wavelet-based measurements inspection algorithm. This is measured as the number of additional sensors that an attacker needs to compromise thanks to the measurements inspection algorithm.

To compare with another state-of-the-art technique, we have also run the same optimisation algorithm using the anomaly detection algorithm described in [CP07], based on Principal Component Analysis (PCA). Being based on linear transformations, this algorithm did not require a linearisation step and could be immediately used in the optimisation Algorithm 18. In this case we have run each experiment 20 times. The results showed that even with the highest number of sensors, i.e. 200, 1 malicious sensor was enough to spoof an event most of the times, and in a few cases, the number of sensors to compromise increased only up to 3. For the masking scenario the average was 1, with a maximum of 4 across the 20 experiments. Even though the PCA-based anomaly detection outperforms many similar techniques and is able to cope also with correlated anomalies, including collusion scenarios, in the worst-case scenario it does not introduce a significant gain in resilience.

In conclusion, we have discussed that **by controlling 3 sensors or fewer, an attacker can subvert the event detection algorithm in (7.24). To achieve that also whilst staying undetected, the PCA-based anomaly detection algorithm forces the attacker to compromise one to three further sensors, whereas the wavelet-based anomaly detection algorithm demands a substantially higher number of compromised sensors. Indeed, wavelet-based anomaly detection forces the average of the temporal variation in the neighbourhood, which is the criterion that identifies events in (7.24), to be spread across the neighbourhood. In this way, the subsets of colluding sensors need to be large, otherwise there will be some genuine sensor that has information**

[3]This is because the event detection algorithm makes an average of the temporal variation of each sensor in the neighbourhood. Hence, if a sensor produces an extremely low/high temporal variation, it can mask/elicit an event on its own.

about the event presence/absence and whose measurements are not consistent with those of the malicious ones. The analysis shows that the mimicry-based test suite can be used also for comparing different anomaly detection techniques.

7.3.4 Study of the Worst-Case Adversary

In the previous section we have shown how many compromised sensors are needed to change the event detection output whilst staying undetected. However, beside the minimum number of compromised sensors, the optimisation algorithm also returns the specific choices that an adversary would take in the process of impairing event detection whilst evading anomaly detection, with a budget of compromised sensors.

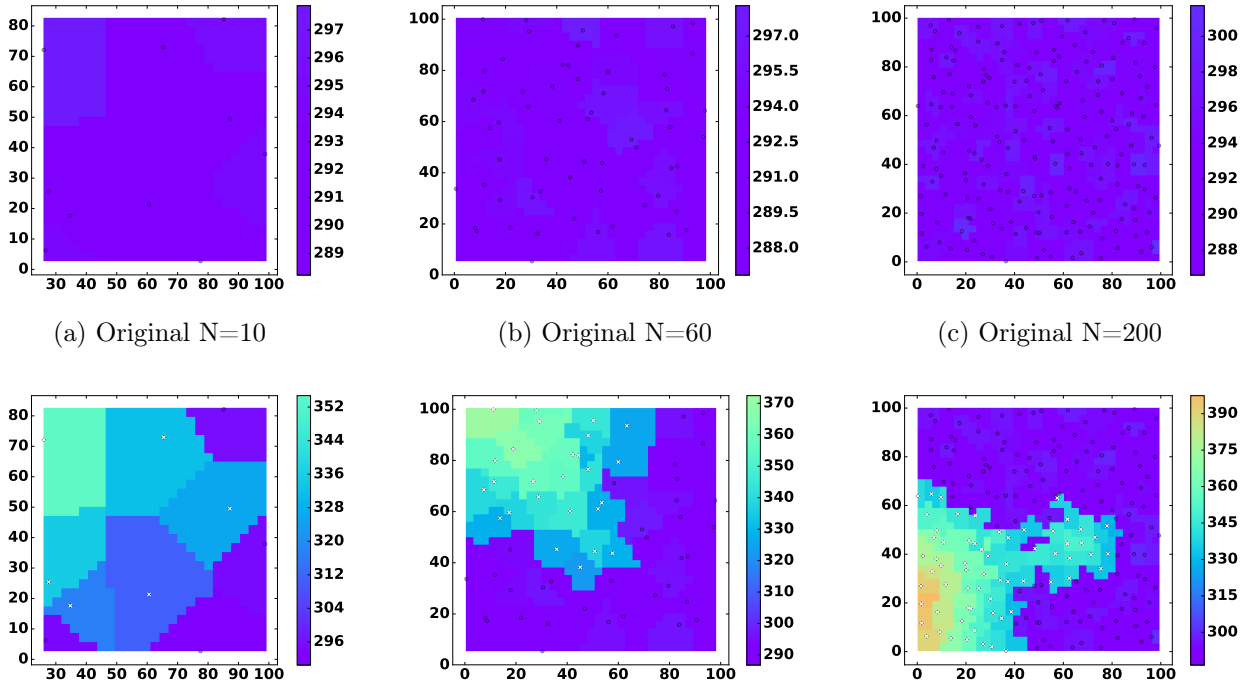
With such information, we are able to learn the most sophisticated strategies to violate event integrity. This can be useful in identifying the main threats in the WSN, as well as the main weaknesses of the anomaly-detection algorithm.

With the help of graphical tools, we analyse some examples of worst-case attacks and extract this kind of attack-related information. The analysis is first done for eliciting attacks and then for masking attacks, in the sections below.

Discussion of Eliciting Results

In Fig. 7.12, we show the effects of the malicious data injection attack produced in output by the optimisation algorithm. These measurements are able to spoof an event, i.e. making the event detection condition (7.24) true, and preventing detection by the wavelet-based anomaly detection algorithm.

A time snapshot of the original measurements (without the attack) is shown in Fig. 7.12a, 7.12b and 7.12c, for deployment sizes of 10, 60 and 200 sensors respectively. The violet colors indicate that the temperatures are not high, so there is no fire in the WSN area. Here the temperature measurements are pretty homogeneous and mainly perturbed by noise.



(a) Original N=10 (b) Original N=60 (c) Original N=200

(d) C=6 (e) C=26 (f) C=61

Figure 7.12: Examples of Elicited scenarios with increasing number of sensors. The circles represent the sensors. The white crosses indicate which sensors are compromised. Colours map the measurements space: bluish colours are for low temperatures, reddish colours are for high temperatures.

The subset of malicious measurements returned by the optimisation algorithm, manages to spoof an event. In particular, the compromised sensors transform the scenarios shown in Fig. 7.12a, 7.12b and 7.12c respectively into the scenarios shown 7.12d, 7.12e and 7.12f. The resulting measurements, shown at a fixed time instant, which is the time where event detection starts to trigger, are noticeably higher than the originals. Moreover, from these figures we can learn how one optimal attacker would act or, from the opposite perspective, the main risks of the WSN. These observations are summarised below.

Contiguity of Targeted Sensors In every eliciting scenario, the targeted sensors, i.e. the sensors that the attacker would choose to compromise and to replace their measurements, are contiguous. This means that a sophisticated attacker would target the sensors in the same area, and that all of them are likely needed to spoof an event. Indeed, if even one sensor was not compromised, it would keep reporting rest-condition measurements. This instead is not a

problem for the sensors that are far apart from the set of compromised sensors because, after a certain distance, spatial correlation fades away and it is plausible that they do not perceive the fire at all.

The main implication of this observation is that attacks to measurements integrity in WSNs would be better counteracted if the measurements inspection algorithm was supported by other techniques which trade a higher cost for better reliability. For instance, it may be the case of manual inspection of the sensor nodes, or of the use of tamper-proof hardware. If this is done for a subset of sensor nodes that are judiciously chosen, there may be significant improvements for a small increase in cost. In particular, the sensor nodes that undergo this special treatment should be: 1. Randomly chosen, to avoid that the attacker uses special treatment for them. 2. Well-spaced, to avoid introducing less protected areas in the WSN.

Optimal Location for Spoofing Events Fig. 7.12d, 7.12e and 7.12f share the characteristic that the spoofed event starts from one of the WSN's corners. In all likelihood, this is because sensors at the WSN's corner have fewer neighbours, and hence there are fewer sources of information that need to be compromised.

This result has relevant implications on the design of both the event detection algorithm and the WSN itself. Event detection should indeed treat the information coming from sensors that are more isolated with more suspiciousness, both for the detection and the rejection of events. The WSN instead, may be deployed in such a way that events at the boundary of the WSN are rare. For instance, a wildfire monitoring WSN may have the outermost sensors bordering on fire-resistant areas.

Real Event Mimicry If we compare Fig. 7.12f with Fig. 7.11a or 7.11b, we note a striking similarity between the characteristics of real and spoofed events. Just like real events, spoofed events represent the propagation of heat from one or more sources, where the temperature is maximum, towards the outskirts where the temperatures progressively decay. **These characteristics were emulated by the evasion optimisation algorithm without domain-**

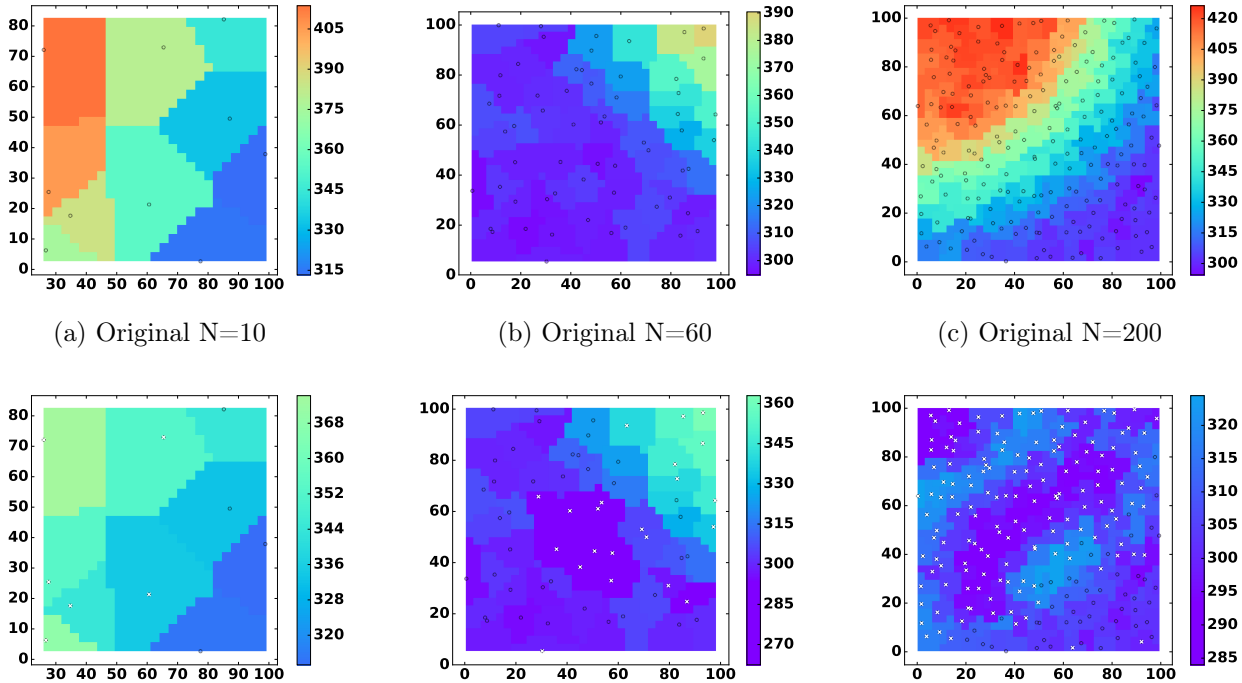
specific knowledge about how events should look like. Of course, the optimisation problem includes the event detection algorithm in its constraints, but this just requires the measurements variation to increase in time, while there is no information related to the events' spatial patterns. This means that it is the measurements inspection algorithm which has learnt the events pattern and has forced the spoofing action to mimic real events. The anomaly detection algorithm learns only one characteristic of spatial correlation, which is the cross-scale relationship. On the contrary, it learns no specific information about the propagation pattern of a fire. We conclude that the cross-scale relationship indirectly captures information about how events manifest, so the attacker is forced to mimic real events and cannot trigger fire detection with unrealistic measurements.

Discussion of Masking Results

In Fig. 7.13, we show the effects of some of the worst-case masking attacks, calculated by the optimisation algorithm. The malicious measurements are able to mask an event, i.e. making the event detection condition (7.24) false, and prevent detection from the wavelet-based anomaly detection algorithm.

Fig. 7.13a, 7.13b and 7.13c show the measurements perceived under real events, after 6 minutes from the fire outbreak, in deployments of 10, 60 and 200 sensors respectively. Fig. 7.13d, 7.13e and 7.13f show optimal masking attacks against these scenarios, i.e. the measurements after the masking attacks that transform the event conditions in Fig. 7.13a, 7.13b and 7.13c into rest conditions. A generic temperature decrease is observable in the measurements after the attack, especially in larger deployments. The observations about the worst-case attacks in masking scenarios are reported below.

Minimum Effort The resulting measurements in Fig. 7.13d, 7.13e, and 7.13f are rest measurements, because the event detection algorithm does not trigger. However, we note that only Fig. 7.13f is a pure rest scenario, whilst Fig. 7.13e and 7.13f are closer to event conditions than to rest conditions. In other words, the event score that is compared to the threshold in



(a) Original N=10 (b) Original N=60 (c) Original N=200

(d) C=6 (e) C=22 (f) C=150

Figure 7.13: Examples of Masked scenarios with increasing number of sensors. The circles represent the sensors. The white crosses indicate which sensors are compromised. Colours map the measurements space: bluish colours are for low temperatures, reddish colours are for high temperatures.

(7.24) to infer the presence of events, has been brought just below the threshold (here we are considering measurements variability for detecting events, so high temperatures are not sufficient to identify a fire). In practice, these measurements correspond to the scenario that one would observe immediately before the breakout of a fire, i.e. when measurements are increasing, but not enough to infer the clear presence of a fire. For the scenario with 200 sensors in Fig. 7.13f, this does not occur because the best attack strategy is to keep the measurements in the bottom right corner unchanged, as they are already compatible with a pure rest condition. However, since the fire to mask reaches high temperatures even at its boundary, the malicious sensors are forced to considerably decrease the measurements to prevent inconsistencies with the neighbouring genuine sensors, i.e. those not perceiving the fire.

Making the minimum effort in subverting event detection is generally more convenient for the attacker, since having reached a good-enough condition in compromising the event's integrity, all the remaining resources can be concentrated on staying undetected. Something similar

happens also in the eliciting scenario, since the spoofed events are also at the boundary of event detection, i.e. they are the smallest event that can trigger event detection, yet for the masking case this is more evident.

The practical implication of this observation is that the anomaly detection algorithm could be improved by analysing the temporal evolution of the event score. Indeed, the malicious measurements will likely aim at making the event score close to the threshold for the attack duration; nonetheless, the typical event evolution causes a more transitory behaviour around the threshold in general. Thus, it may be worthwhile checking that the time evolution of the event score does not show anomalous behaviours such as saturation just below/above the event detection threshold. Even though this is an analysis in the time domain, it refers to an index –the event score– which is defined also in the space domain. As we highlighted in Section 7.3.1, anomaly detection based only on the time domain can be circumvented by an attacker by mimicking the temporal dynamics of the monitored phenomena.

Smoother Transitions In Fig. 7.13d, we observe that the southernmost malicious sensors make up smooth transitions between the malicious measurements in the north, which have been significantly decreased, and the genuine measurements at south-east, which corresponds to low temperatures.

This may seem counter-intuitive because in the original genuine scenario, depicted in Fig. 7.13a the transitions are less smooth and they raise no anomaly. However, we should note that the masking attack causes a change of context, i.e. from a fire scenario to a scenario with no fire. In the presence of fires, the zones on fire can show a higher gradient with the zones that have not been reached by the fire yet. In the absence of fires, instead, the measurements are more homogeneous since the sensors are subject to similar environmental conditions. This is a physical characteristic of the way events manifest, that the wavelet-based anomaly detection algorithm has captured. As a consequence, the masking sensors are forced to reduce the measurements gradient whilst decreasing the measurements for masking the fire.

C explosion in Time The purpose of masking is to prevent event detection and, in our case it may mean silencing fire alarms. With no fire alarm, fires may not be dealt with, so they could keep spreading and covering wider areas of the WSN, as mentioned in Section 7.3.3. As a consequence, the sensors that are not perceiving the event at all at a certain time, may perceive it in the future. So, if C sensors are needed to mask an event at time t, $C + \Delta C$ are needed at time $t + \Delta t$. For instance, in Fig. 7.13c we show a fire that has covered most of the WSN deployment, and from Fig. 7.13f we deduce that the attacker needs to compromise 75% sensors to mask the event.

This effect depends on the nature of the events and in particular on the area they cover. If the average area covered by an event is smaller than the WSN deployment, then masking attacks are more threatening. The same consideration holds if the timeliness of intervention is crucial. For instance, delaying fire detection even by a few minutes may have dangerous consequences.

7.3.5 Design of Secure Wireless Sensor Networks

Evaluating the risk of a system makes sense only if such risk can be mitigated when high. Measurements inspection is generally able to highly reduce this risk but, when there is poor cross-sensor correlation in the measurements, its benefits are substantially reduced. Nevertheless, measurements inspection has considerable advantages compared to other integrity verification techniques, including a very little overhead on the system and negligible installation cost. So, since sensor nodes are not expensive, it is worthwhile studying the performance improvements that are obtained by increasing the deployment density and the cross-sensor correlation in turn.

Understanding how the number of sensors reduces the risk is not trivial, so this operation would highly benefit from a numerical analysis similar to the problem of calculating the attack vector that achieves evasion, but from the opposite perspective. Rather than calculating the minimum number of malicious sensors that are needed to change the output of event detection, the quantity of interest is the minimum number of deployed sensors that are needed to achieve resiliency to a required number of malicious compromises.

This problem shares some similarities to that of *k-coverage* deployment in sensor networks [HT05], which deals with the problem of guaranteeing a minimum of k sensors in the sensing range for a set of WSN spatial locations. The goal is to ensure that if an event is present at a certain location, at least k sensors are able to detect it. This approach is based on the assumption that each sensor can detect an event on its own, which holds for applications such as target detection and localisation. However, the events' presence can be determined only with statistical analyses on many sensors in other applications, e.g. detection of natural disasters, pathological conditions, and nuclear threats. Hence, a point in space may need to be in the sensory range of many sensors to be covered. Moreover, k-coverage does not deal directly with the problem of malicious nodes, but rather considers genuine faults and uncertainty [DCI02; ZC04]. In the case of malicious interference, the k-coverage model can be applied only if malicious sensor nodes are independently distributed. Instead, an attacker may compromise contiguous sensors to gain full control over a WSN area.

Our approach is to guarantee a resiliency requirement, which is the condition that anomaly detection triggers if the event detection is subverted with less than C_{max} malicious sensors. Because of such an *if-then* condition, it is not trivial to map the requirement to a set of constraints. Thus, we rather transform the design problem into multiple attack vector calculation problems with the function *designDeployment* shown in Algorithm 20.

This algorithm is a binary search in the space of all possible values N, that adds a tolerance for the solution *tol*, such that the returned solution overestimates the real solution by maximum *tol* times. This is to reduce the optimisation time, since the algorithm *designDeployment* can include many iterations of *findAttackVector*, which is computationally expensive, especially when high values of N need to be tested during the binary search.

Moreover, Algorithm 20 gives assurance about the number of compromised sensors that can be tolerated only for a specific scenario, i.e. a set of original genuine measurements which will be partially compromised. To obtain an assurance that applies to generic scenarios, we run Algorithm 20 under multiple scenarios, generated at line 6, and extract the maximum.

In our experiments, we have run each experiment under 20 different scenarios and calculated

Algorithm 20 *designDeployment*

Require: C_{max}, tol
Ensure: N_{min}
1: $N = 2C_{max}$
2: $N_{min}^{inf} = C_{max}$
3: $N_{min}^{sup} = \infty$
4: $N' = 0$
5: **while** $(N - N')/(N) > tol$ and $N > C_{max}$ **do**
6: Generate $W \times N$ measurements matrix X_{w_t}
7: C_{min}, $\mathbf{M}^*$, $\mathbf{y}^* = \mathit{findAttackVector}(X_{w_t})$
8: **if** $C_{min} < C_{max}$ **then**
9: $N_{min}^{inf} = N$
10: **else**
11: $N_{min}^{sup} = N$
12: **end if**
13: $N' = N$
14: **if** $N_{min}^{sup} < \infty$ **then**
15: $N = \left\lfloor \frac{N_{min}^{inf} + N_{min}^{sup}}{2} \right\rfloor$
16: **else**
17: $N = 2N$
18: **end if**
19: **end while**
20: **return** N_{min}^{sup}

the deployment size that guarantees resilience to a varying number of compromised sensors. We used a tolerance tol=0.10, which gives a maximum overestimation of N by 10%. The results are summarised in Table 7.6.

Table 7.6: N_{min}: Minimum number of deployed sensors that guarantee resilience to a maximum of C compromised nodes.

Attack \ C	10	30	60	90	100	200
Eliciting	20	75	180	400	>400	>400
Masking	22	45	93	119	124	398

These results show that for the masking case, the required number of sensors is definitely lower and has a lower rate of increase. This is not surprising as masking has proved more difficult to achieve from the attacker's perspective.

To mitigate this gap, the event detection algorithm may be set to trigger when more confident about the fire, but this inevitably delays the event detection. Instead, a more effective solution may be to define different levels of alarms. For instance low-risk, medium-risk and high-risk

alarms, depending on the spread of the fire. Indeed, the cost of spoofing an event would be proportional to the risk level, since the spoofed fire's spread is constrained by the number of malicious sensors. Clearly, if the event reaction is delayed until the risk of the alarm gets high, the system still gets some relevant damage, but if the counteraction is proportional to the risk of the alarm[4], the damage is minimised. This aspect should be taken into consideration when designing an event-detection WSN.

Thanks to Algorithm 20, the resilience of a WSN can be calculated and guaranteed at design time. The result may also be used for a feasibility analysis that checks if the cost savings given by measurements inspection with respect to other techniques for measurements integrity, such as manual intervention, are effectively worth the cost increase given by the enlargement in the deployment size. However, while the resilience of the WSN obtained through measurements inspection can be estimated through the methodology introduced above, there is currently no means to make analogous analyses for other techniques, or to combine them.

7.3.6 Conclusions

Analysing the potential damage of malicious data injections in WSNs requires a well defined problem for the attacker. In this section, we have considered the case of event detection WSNs, where the attacker has two conflicting goals. The first is to mislead event detection, by either triggering event detection when no event occurs or masking real events. The second is to evade the anomaly detection algorithm that may unveil the measurements manipulation.

We have first evaluated the robustness of measurements inspection through the creation of mimicry attacks, which build malicious data from an accurate selection and moulding of data observed under normal circumstances. With this approach, mimicry attacks obtain data that can represent accurately a false event or a false rest condition, which can cause high damage whilst achieving high chance of staying undetected.

We have designed and implemented an automatic test suite, which is able to autonomously

[4]E.g., with low risk: activate sprinkler systems, with medium risk send firemen, with high risk dispatch firefighting helicopters.

select the data to mimic, use it to build malicious measurements and adapt the latter to the current context. The test suite abstracts from the details of both the measurements inspection algorithm and the event detection algorithm, so it is context-independent and requires only an historical dataset of measurements as a knowledge base.

We have applied the test suite to the wavelet-based approach, with a synthetic and a real dataset and we conclude that the detection gives highly reliable results. Good results have been achieved for the characterisation and diagnosis phase, even though there is a substantial increase of complexity compared to detection. These are due to error propagation effects as well as to the complexity of the problem itself.

Then, we have introduced worst-case attacks, which identify the worst case scenarios by analysing the measurements space and identifying a set of malicious measurements that guarantees both the ability to stay undetected and to change the main output of the event detection algorithm (i.e. whether there is any event). However, we have faced several challenges that prevent to resolve the problem exactly in a reasonable time. Therefore, we have introduced an algorithm that approximates the desired result and can be solved computationally. With such an algorithm, a network operator can be informed of how the WSN can tolerate worst case scenario attack.

We have tested the algorithm on a wildfire monitoring WSN and extracted an assurance for the minimum number of compromised sensors that can be tolerated against eliciting and masking attacks. Moreover, we have shown that the algorithm can be applied to different anomaly detection algorithms. Effectively, the algorithm proved also as a methodology to compare the resilience provided by different approaches.

Moreover, we have gained valuable knowledge from the application of this algorithm, such as the impact of the event detection algorithm, which can considerably reduce the benefits of measurements inspection if it does not characterise events properly. In particular, a loose characterisation of events favours spoofing, while a strict characterisation makes the masking task easier. A trade-off is required in the number of event conditions that are evaluated to maximise the benefits of measurements inspection against both eliciting and masking attacks.

When a good event detection algorithm is in place, measurements inspection constrains the attacker to closely mimic event or rest conditions for achieving evasion. Our results have allowed us to study the attacker's strategies that can achieve this goal. The analysis of the worst-case attacks has revealed important information that may improve the design of a WSN and of the anomaly detection algorithm. For instance, we have observed that it is more advantageous for the attacker to compromise neighbouring sensors, and that compromising the sensors at the boundaries of the WSN deployment is the easiest way to spoof events.

Finally, we have presented the first technique that assures resilience to malicious data injections in WSNs at design time. Such an algorithm provides the minimum number of sensors that need to be deployed to guarantee detection with a maximum number of malicious sensors that can be tolerated. This not only gives security guarantees, which are generally missing in the anomaly detection field, but also evaluates the applicability of anomaly detection itself.

Chapter 8

Combining Measurement Integrity with Sensor Integrity

The techniques presented so far enable us to infer if the measurements sensed by a WSN can be trusted or not, and the robustness evaluation methodology presented in the previous chapter allows a network operator to quantify the reliability of such techniques and to design a WSN with an assured degree of reliability. The measurements inspection techniques presented in Chapters 5 and 6 build a degree of trust in an untrusted environment, as they verify the integrity of the measurements through the measurements themselves, which may be compromised.

In contrast, other techniques that differ from measurements inspection allow us to increase the WSN's trustworthiness by ascertaining the devices' trustworthiness. This could be achieved through manual verification; however, it would expensive for large deployments and impractical for WSNs that operate in harsh environments. The body of work concerned with *software attestation* instead verifies the integrity of the software running on the sensor nodes by exploiting the capabilities of the WSN devices themselves. If the software is proved to be uncorrupted, we expect that the measurements cannot be compromised through software-driven malicious data injections. In other words, the trustworthiness of the software running on a device can be transferred, to some extent, to the measurements observed by that device.

This characteristic can be exploited to alleviate a strict trade-off between the degree of confi-

dence in the measurements' integrity and the cost to obtain it. In fact, current solutions for the integrity, and the security in general, of WSNs are highly biased towards one of the two. In particular, the security level of attestation and the low cost of measurements inspection are their most attractive characteristics and we seek to extract the benefits of both, while keeping drawbacks to a minimum, by combining the two approaches. Attestation gives reliable information about the software integrity of a sensor node, but it introduces an ad-hoc communication protocol and additional computations, which become burdensome when run on many sensors and multiple times. Measurements inspection instead, has the ability to detect integrity violations with no extra communications (when run by the base station or a remote server) and low computational overhead. Moreover, it is more and more reliable when a subset of sensors is genuine, hence once attestation verifies the integrity of a few nodes, measurements inspection can extend the attestation's result to many neighbouring sensors.

The work presented in this Chapter was done in collaboration with with Rodrigo Vieira Steiner, as a colleague PhD Student at Imperial College London. Since software attestation is within his areas of expertise, Mr Steiner took part of the attestation-specific issues, while I took care of the issues regarding measurements inspection. Together, we designed the methods and protocols presented below, and built simulations and analyses.

This chapter presents a study of how these two techniques can be best combined, since there is a wide range of possibilities, and it is not obvious which combinations are better beforehand. In many cases, the way in which both mechanisms are combined ends up favouring the characteristics of only one of them, and some combinations might even degrade the overall performance when compared with each of the individual approaches. Instead, a judicious combination, which depends also on the application's integrity requirements, enables us to optimise the usage of the integrity verification techniques across the node layer and at the measurements layer. Indeed, we will be able to determine how many attestations are needed to trust all the WSN nodes, including those that are not attested, by exploiting the correlation analysis done by measurements inspection.

8.1 Software Attestation: a Generic Signature-Based Approach

Attestation is an integrity verification mechanism that allows a trusted device, named *verifier*, to validate the memory contents of an untrusted device, called a *prover* [SL16]. In particular, software-based attestation mechanisms are capable of attesting untrusted devices without the use of tamper-resistant hardware or any other particular piece of hardware that could be used to restrict an adversary's control over the prover, such as ROM or a Memory Protection Unit (MPU).

These mechanisms follow a challenge-response protocol as illustrated in Figure 8.1. The process starts with the verifier generating a challenge, which is essentially a random number, and sending it to the prover. The prover then uses the challenge to traverse its own memory in a pseudo-random fashion computing a checksum of the memory addresses accessed and sends the result to the verifier. It is assumed that the verifier knows in advance the expected memory contents of the prover, so it can perform the same computation and validate the response. However, the prover not only must come back with the correct response, it has to do so within a time limit after the challenge has been sent. This time limit is imposed to prevent an adversary from performing additional operations to masquerade any possible modifications it may have done to the prover's original memory contents.

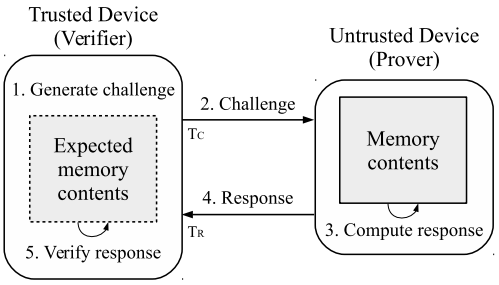


Figure 8.1: Attestation overview.

In practice, besides the time taken to execute the attestation routine, the timeout also has to incorporate the time necessary to send the challenge and receive the response. While it is possible to specify a time limit for sending the packets over the network, it is not possible to

make this time constant. Therefore, an adversary could attempt to reduce its network delay, earning time to execute more operations.

In theory, the best adversary has zero delay, meaning its time advantage would be the maximum time allowed to send and receive the messages over the network. To prevent this from happening the attestation routine has to be implemented in a time-optimal way. The routine comprises a main loop, where it handles the memory accesses, which must be executed a number of times.

To hide any modifications, an adversary would need to insert new instructions into the loop, introducing a fixed overhead per iteration. Consequently, the number of iterations is defined according to the maximum network Round-Trip Time (RTT) allowed, such that the adversary's overhead is greater than its advantage.

8.2 Attack Model Refinement

Since in this chapter we are considering software attestation beside measurements inspection, we need to refine the attack model presented in Chapter 4 to include also the attacker's capabilities with respect to the ability to tamper with the sensor nodes and software.

Even though measurements inspection can address also malicious data conveyed through e.g., environment manipulations, network integrity violations, or hardware tampering, we will focus on malicious data injection attacks conveyed through malicious software run by compromised nodes. In particular, we assume that the adversary cannot modify the node's hardware, otherwise software-based attestation would not be applicable. Hardware modification attacks include also expanding the node's memory with a flash memory or an SD card which contain, e.g., a copy of the authentic software. On the contrary, we do not restrict how an adversary may compromise a node's software. Measurements inspection is capable of identifying sensor nodes with compromised hardware, as long as a sufficient subset of sensors keeps its hardware unchanged, which is generally the case since physical attacks do not scale. However, in this chapter, we focus on the cooperation between both schemes and leave the cases where only individual techniques are effective for future work.

We also assume the WSN sink cannot be compromised, for instance being equipped with tamper-proof hardware, as it plays the role of the verifier. Fundamentally, if the sink gets compromised, then the whole WSN is compromised. We further assume that provers are within direct communication range and can only communicate with the verifier during attestation. This prevents a prover from relaying the challenge to another, more powerful, device to run attestation on its behalf. In practice this requirement can be met by distributing multiple tamper-resistant verifiers across the network.

Finally, we assume the network maximum RTT can be known a priori and that an authentication system is in place. All of these assumptions are requirements well-known in literature for the application of software-based attestation [SL16], and while we could relax them, by using hardware-based or hybrid attestation, that would mean equipping each and every sensor in the network with the specific hardware demanded by such techniques.

8.3 Anomaly-Based Measurements Inspection vs Software Attestation

While attestation is effective to determine if a specific node has been compromised, it is expensive since it adds the network overhead of the challenge-response protocol and prevents a node from doing any task other than the response computation. When attestation is used to ascertain the integrity of the measurements by inferring it from the integrity of the node's software, its cost increases even more as the challenge-response protocol needs to be executed with high frequency. This is a consequence of the time lapse between the time a sensor node is attested and the time when the measurement is taken, in which the sensor node could potentially be compromised. This problem has not been addressed yet in the literature, and we solve it by combining attestation with measurements inspection, which decreases the attestation frequency by two orders of magnitude, regardless of the compromise rate.

Indeed, while most works focus on the integrity test of attestation [Ses+04; PS05; Ses+08;

ZL10], the impact of performing attestation, especially in terms of energy consumption, are not well covered. Chen *et al.* [CWW10] investigate how often attestation should be triggered in order to optimize the network lifetime without degrading the detection rate of compromised nodes. They reach the conclusion that higher compromise rates require higher attestation frequencies; however, it is difficult to know how fast an adversary can compromise network nodes.

In Section 3.4.3, we have reported the overhead in computations and communications introduced by state-of-the-art measurements inspection algorithms, and we can observe that, especially in communications, the overhead is generally kept low. However, measurements inspection techniques have significantly poorer detection performance when most sensors nodes are malicious and collude in the injection of malicious data. For this reason, Tanachaiwiwat and Helmy [TH06] propose the deployment of tamper-resistant sensor nodes that authenticate suspicious sensors. This chapter is concerned with integrity problems rather than authentication, and detects malicious activity in highly compromised networks by assuming that only the data sink is tamper-resistant.

Although compromised nodes may perform attacks to undermine the network confidentiality and availability, which could be detected by attestation but not by measurements inspection, it is in attacks to data integrity that both schemes overlap. Moreover, while attestation is good at telling if a node has been compromised or not, it has no way of determining which nodes should be attested, whereas measurements inspection is good at pointing the finger at suspicious nodes but less effective in determining maliciousness. In this sense, the techniques complement each other. In real applications, we envision the coexistence of both mechanisms and based on a comparative study, which to our knowledge has not been done before, we propose novel methods to maximise the benefits of their complementary capabilities. Our goal is to combine them while keeping the high security level of attestation and the low cost of measurements inspection. Nevertheless, *there are many ways to integrate the two approaches, and it is not obvious which combinations are better beforehand.* To the best of our knowledge, no work in the literature presents a direct comparison of measurements inspection and software-based attestation in WSNs. In this chapter, we analyse and compare the two approaches in detail,

focusing on the aspects that make them complementary.

We start by comparing the performance of measurements inspection and software attestation in this Section. Then, we identify the benefits and drawbacks of these two techniques, examine how their parameters may be tuned to overcome these limitations and analyse the cost involved, and finally move to the design of the combination schemes, which manage to overcome such limitations and obtain a more convenient trade off between security and cost, compared with the individual techniques.

8.4 Combination Potentials and Issues

Attestation and measurements inspection are very different in nature. Besides the difference in performance, they also vary significantly with respect to: constraints introduced, test frequency (how often the integrity of the WSN is tested), and power overhead. We analyse these aspects below.

Constraints Introduced. Measurements inspection introduces the need for a trusted *inspector*, i.e. an entity which can run the anomaly detection algorithm and be trusted for its output. To address this, it is often cheaper to make a special device in charge of running the algorithm, such as the base station or a tamper-proof sensor node. Good performance is achieved when the data of many sensors is available, so the best candidate for running the detection algorithm is a centralised sink node that receives all the measurements. When this is not practicable, e.g. because the WSN is particularly large, the local sink, e.g. the cluster head, should be in charge of the analyses.

Similarly, software-based attestation introduces the need for a trusted *verifier* to attest untrusted nodes. Furthermore, attestation requires the network maximum RTT to be a known constant and works under the assumptions that the adversary cannot modify the prover's hardware.

Test Frequency. Measurements inspection should be run on every data batch used by the

application. If the application layer makes aggregations that collapse multiple time samples into one, then it is convenient to run anomaly detection on such aggregates. This results in a better aimed protection of the application task and in a reduced detection frequency. The latter can also be reduced when the application layer uses the measurements of each time instant separately, provided a mechanism that extends the validity of a detection check to future samples. An example is a temporal correlation check that ascertains the physical process is coherent in time [IMGL17].

Software attestation has analogous requirements. As soon as the software integrity is verified, the measurements produced during the time of a successful attestation are genuine. However, the sensor node may misbehave before or after the verification is complete. The larger the time gap between attestation and measurements transmission, the smaller the reliability. This requirement forces the verifier to be close, from a network-layer perspective, to the prover, because more transmission hops increase the overall delay and reduce the attestation's reliability. Note that this contrasts with the requirement of measurements inspection to have the uppermost sink as the inspector, since there may be many hops between it and another sensor node.

Power Overhead. When the anomaly detection algorithm is run by the sink, there is no communication overhead. On the contrary, such entity is subject to a computational overhead that is generally polynomial in the number of measurements [IL15b]. These characteristics lead to a low overall overhead.

Software attestation instead, introduces a communication overhead, due to the attestation protocol, as well as a computational overhead, especially for the verifier, which needs to validate every prover's response. This step is deliberately computational intensive, otherwise a malicious node would be able to restore the original software and reply to the challenge in time. Both in computation and communication, the overhead of attestation is noticeably higher than measurements inspection.

8.4.1 Trade-off Tuning

Individually, measurements inspection provides reasonable security confidence with low-power overhead while software-based attestation achieves high security confidence with high-power overhead. However, different choices in the usage and parameters of the individual techniques give a different balance in the cost/security trade-off. We analyse this aspect below.

Highly Reliable Software Attestation. Software attestation is mostly reliable when the Attestation Frequency (AF) is high. The attestation frequency is defined with respect to a single sensor node, i.e. the number of attestation challenges received by a single sensor node, divided by the number of measurements sent by the same sensor. Hence, we refer to AF as the attestation frequency averaged across all sensor nodes. For instance, if only one sensor is attested each time it sends a measurement, then $AF = 1/N$.

Assuming all sensors are equally important, they should all be attested with the same frequency. In such a case, the reliability of attestation is high when $AF \approx 1$, i.e. when an attestation response is sent together with each measurement report. However, with such a choice, the communication overhead is three times as high on average (because of the challenge and response messages). A noticeable computational overhead is also present for the calculation of the attestation response, which, in this scenario, becomes the main task of the sensor nodes processors. In conclusion, using $AF \approx 1$ reduces the network lifetime, requiring a high maintenance cost either to replace nodes' batteries or to insert new nodes.

Low-Power Software Attestation. The cost of software-based attestation can be reduced by decreasing AF. This can be done by either reducing the time between two attestations or the number of attested sensors. However, the reliability of attestation deteriorates when the time between two attestations, i.e. $\frac{T}{AF}$, is close to the time needed to swap the genuine software with a malicious one, where T is the time between two measurement transmissions.

A possible way to reduce the power of software-based attestation, is then to reduce the number of attested sensor nodes. This can be done by selecting a random subset of nodes to attest which is small compared to the total N nodes. Similarly to the detection step in measurements

inspection, when attestation fails for at least one sensor, a more thorough analysis can be triggered, i.e. attesting all other nodes.

Such "attestation-based detection" is reliable only when the number of attested nodes is comparable to $N - C$, i.e. the number of genuine sensors. Therefore, unless almost all sensor nodes are compromised, the power overhead cannot be reduced significantly without a significant loss in reliability.

Low-Power Measurements Inspection. Measurements inspection makes a reliable measurements integrity check during the detection step: when the false information introduced by malicious sensors is far from reality and there are genuine sensors whose correlation with malicious sensors is disrupted, detection unveils the presence of malicious data.

This is generally the case when there are at least G genuine sensors, where G depends on the WSN deployment, on the monitored physical phenomenon, and on the kind of malicious measurements. Measurements inspection is also able to identify malicious sensor nodes in the characterisation step. However, for this step to be as successful as detection, a higher value for G is required.

To keep power consumption at a minimum, measurements inspection needs to accept the presence of potentially malicious sensors and identify only the most likely compromised. Thereafter, if the remaining malicious nodes keep injecting malicious data, the attack becomes less efficient, and they need either to make the false measurements closer to reality or to become more detectable.

Highly Reliable Measurements Inspection. To increase the number of discovered malicious nodes, measurements inspection needs to rely more on the detection and less on characterisation. For instance, characterisation can just produce a set of possible scenarios, i.e. mutually exclusive hypotheses to the detected anomaly with the corresponding malicious sensors.

A further investigation, which is conducted in a reliable way, such as with surveys in the field, would then reveal which hypothesis is correct and which sensors are malicious in turn. Checking all the sensor nodes in the field, however, is generally expensive and prevents a prompt reaction

to the attack.

In conclusion, attestation can ascertain the measurements' integrity, provided that it is run close to the measurement transmission time. Running attestation for all measurements is too expensive, but running it for an arbitrary subset decreases reliability. Measurements inspection is able to detect the presence of malicious measurements and identify suspicious sensors, but the final decision about a sensor's maliciousness should be taken with a more reliable approach. The two techniques complement each other as measurements inspection can trigger attestation in anomalous scenarios only, reducing the attestation frequency and keeping reliability high. In the next section, we will design novel methods to combine the two techniques based on such observations.

8.5 Design and Analysis of Combinations of the Two Approaches

In the previous section, we have observed that, by relying on either measurements inspection or attestation, it is not possible to improve significantly in power overhead without impairing the degree of security or vice versa. The objective of the combination is to achieve a more convenient trade-off, i.e. with much higher security confidence than measurements inspection, and lower power overhead compared with software-based attestation.

While the power overhead is well defined, the security confidence may be measured with different metrics. In the following, we consider that confidence in the integrity of the system is determined by two factors: identifying as malicious only malicious scenarios (true positives) and not genuine scenarios (false positives). These two metrics will be used to evaluate the performance of measurements inspection, attestation, and their combination.

To compare the different techniques with each other we express the TPR and FPR as a function of the variables involved in the problem, i.e. the number of compromised sensors and the algorithms' parameters. This *analytical approach* allows us to analyse the influence of each

variable on the security confidence and compare the techniques under different circumstances without the need to run ad-hoc experiments for each scenario. Besides being more lightweight, the analytical approach enables us to express the TPR and FPR of the combination schemes that we will present as a function of the TPR and FPR of attestation and measurements inspection. So, rather than evaluating the performance of a combination scheme in a set of arbitrary scenarios, we obtain general context-independent performance information. To do that, we initially show the TPR and FPR of measurements inspection and attestation in the following sections.

8.5.1 Measurements Inspection Performance

In measurements inspection, detection, characterisation and diagnosis are different tasks, where the TPR and FPR in each step depends on the TPR and FPR in the previous steps. Moreover, the performance of detection is different from the performance of characterisation and diagnosis. Indeed, detection operates at the granularity of the network, or of a cluster of nodes, while characterisation and diagnosis operate at the granularity of sensors.

As we discussed in the previous chapter, we can link the performance of detection to the random variable C, representing the number of malicious nodes in the network. Instead, the performance of characterisation and diagnosis are linked to the random variable S_M, indicating that a generic sensor S is malicious (or dually S_G, the event that a generic sensor S is genuine, where $P(S_G) = 1 - P(S_M)$). The notation used in this chapter is summarized in Table 8.1.

If considering the whole measurements inspection process, a true positive is a malicious sensor that is diagnosed as malicious and characterised as anomalous, after anomaly detection triggered. A false positive, instead, is a genuine sensor diagnosed as malicious, after being characterised as anomalous. This can occur, e.g., because the anomaly is falsely detected or the characterisation falsely blames that sensor. Considering the three steps separately, the probability of having a true positive from measurements inspection is:

Table 8.1: Notation Summary.

Term	Description
N	Number of sensor nodes
C	Number of malicious sensor nodes
A_{D_T}	Anomaly Detection triggers
$A_{C_F^S}$	Sensor S fails Anomaly-based Characterisation
$A_{D_M^S}$	Anomaly-based Diagnosis output for sensor S is "Malicious"
MI_F^S	Sensor S fails Measurements Inspection
$A_{T_F^S}$	Sensor S fails attestation
$A_{T_P^S}$	Sensor S passes attestation
g	Group size
S_G	Sensor S is genuine
S_M	Sensor S is malicious
T_E	Total number of examined sensors
T_G	Total number of groups
T_M	Total number of malicious sensors
T_{M_F}	Total number of malicious sensors that fail a given test
TPR	True Positive Rate
FPR	False Positive Rate

$$P(MI_F^S|S_M) = P(A_{D_T}A_{C_F}A_{D_M^S}|S_M) = \\ P(A_{D_M^S}|A_{C_F}A_{D_T}S_M)P(A_{C_F}|A_{D_T}S_M)P(A_{D_T}|C>0) \tag{8.1}$$

Where MI_F^S is the event that sensor S fails measurements inspection, A_{D_T} denotes the event that the anomaly detection algorithm triggers, $A_{C_F^S}$ the event that characterisation fails, and $A_{D_M^S}$ the events that the sensor is diagnosed as malicious.

The probability of a false positive is:

$$P(MI_F^S|S_G) = P(A_{D_T}A_{C_F}A_{D_M^S}|S_G) = \\ P(A_{D_M^S}|A_{C_F}A_{D_T}S_G)P(A_{C_F}|A_{D_T}S_G) \\ \Big(P(C=0)P(A_{D_T}|C=0\cancel{S_G}) + P(C>0)P(A_{D_T}|C>0\cancel{S_G})\Big) \tag{8.2}$$

In the absence of a probabilistic model for both approaches, the probabilities can be approximated with experimental frequencies. In particular, the probabilities listed above can be characterised with True Positive Rates (TPR) and False Positive Rates (FPR). Indeed, the frequency of $(MI_F^S|S_M)$ and $(MI_F^S|S_G)$ correspond to the measurements inspection TPR and FPR, which we refer to as TPR_{MI} and FPR_{MI}. We give a proof for the former, while an analogous proof holds for the latter. Let T_E denote the total number of examined sensors, T_M denote the total number of malicious sensors, and T_{M_F} denote the total number of malicious sensors that fail measurements inspection, then:

$$P(MI_F^S|S_M) = \frac{P(MI_F^S \cap S_M)}{P(S_M)} \approx \frac{\frac{T_{M_F}}{T_E}}{\frac{T_M}{T_E}} = \frac{T_{M_F}}{T_M} = TPR_{MI} \tag{8.3}$$

With a similar reasoning, the $P(MI_F^S|S_G)$ can be approximated with the measurements inspection FPR (FPR_{MI}).

8.5.2 Attestation Performance

Let c be the prover's CPU clock speed, a be the adversary time advantage, and o be the adversary overhead per iteration of the attestation routine, then the number of iterations i can be calculated as [Ses+05]:

$$i \geq \frac{(c \cdot a)}{o} \geq m \ln(m) \tag{8.4}$$

Where m is the size of the memory being attested. Since the memory is traversed in a pseudo-random fashion, a minimum number of iterations is necessary to probabilistically assure (based on the Coupon Collector's Problem [MU05]) that each memory address is accessed at least once.

Under the assumptions made by software-based attestation mechanisms, that both the maximum network RTT and the minimum adversary overhead per iteration are known a priori, it is possible to define the number of iterations of the attestation routines to be executed such that a malicious sensor S_M has only a negligible chance ϵ to pass attestation. This chance is the possibility of a collision — when the genuine and malicious memory contents output the same checksum value. Thus:

$$P(A_{T_F^S}|S_M) \geq 1 - \epsilon \tag{8.5}$$

Whereas, a genuine node will always pass attestation:

$$P(A_{T_F^S}|S_G) = 0 \tag{8.6}$$

We can characterise the attestation TPR, which we refer to as TPR_{AT}, by analysing the fre-

quency of the event $(A_{T_F^S}|S_M)$. Let T_E denote the total number of examined sensors, T_M denote the total number of malicious sensors, and T_{M_F} denote the total number of malicious sensors that fail attestation, then:

$$P(A_{T_F^S}|S_M) = \frac{P(A_{T_F^S} \cap S_M)}{P(S_M)} \approx \frac{\frac{T_{M_F}}{T_E}}{\frac{T_M}{T_E}} = \frac{T_{M_F}}{T_M} = TPR_{A_T} \tag{8.7}$$

By definition $\text{FPR}_{A_T} = 0$, so the security confidence provided by attestation is nearly ideal. This is the main advantage of attestation compared with measurements inspection. However, as we discussed in Section 8.4.1, using such confidence to the guarantee the measurements integrity requires ideally all sensors to be attested, and for each new measurement ($AF = 1$) that is collected, which is most likely impractical. Our goal is to use measurements inspection to achieve a degree of security which is as close as possible to the approach just described, but for a much lower attestation frequency.

The combination of the two techniques gives rise to a full spectrum of solutions in the trade-off between power efficiency and security, depending on how and when measurements inspection hands over to attestation. Predicting the resulting performance for a given choice is complex, as it depends on the measured performance of each single approach and on their inter-dependence. For this reason, we present three different combination approaches and express their performance as a function of the variables involved.

8.5.3 Detect and Attest

The architecture of our first proposed combination, denoted with *Detect and Attest* (D&A), is shown in Figure 8.2. This scheme is designed to exploit only the most reliable step in measurements inspection, which is detection, and relay the characterisation task to attestation.

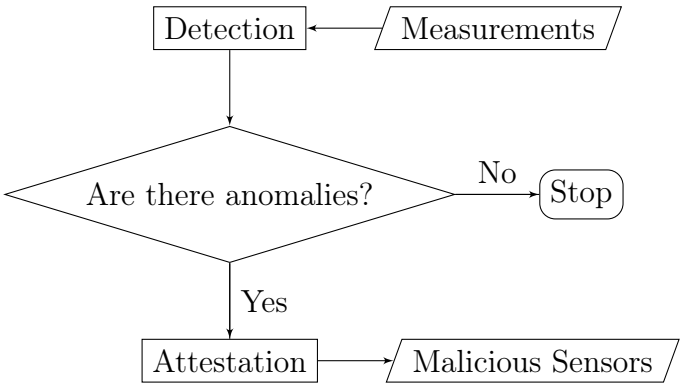


Figure 8.2: D&A Scheme.

Performance

Since D&A is the series of the detection step from measurements inspection and the result of attestation, the output TPR is the product of the TPRs of both:

$$TPR_{D\&A} = TPR_{AD}TPR_{AT} \tag{8.8}$$

A false positive occurs when measurements inspection detects an anomaly, regardless of the presence of malicious data, and attestation fails on a genuine node. Thus, the FPR of D&A is:

$$FPR_{D\&A} = \Big(P(C=0)FPR_{AD} + P(C>0)TPR_{AD}\Big)FPR_{AT} \tag{8.9}$$

Where the term in brackets is the probability that anomaly detection triggers. This coincides with the expected attestation frequency, since all sensor nodes are attested when anomaly detection triggers:

$$AF_{D\&A} = P(C=0)FPR_{AD} + P(C>0)TPR_{AD} \tag{8.10}$$

8.5.4 Group Subset Attestation

In our second proposed combination, *Group Subset Attestation* (GSA), measurements inspection is used in the detection step and produces the groups of possible malicious nodes, while attestation acts as a judge to take the final decision. Its architecture is depicted in Figure 8.3.

We observe that GSA uses measurements inspection for detection of anomalies and the grouping part of characterisation. Then it iteratively selects a random member for each group and attests it. When the ratio of genuine sensors in a group outweighs the ratio of malicious sensors or vice versa, the attestation for that group stops and the result for the majority of sensors is applied to the whole group. The guard condition that determines whether the number of attestations is high enough for the group is:

$$\frac{|||S \in g : S_G| - |S \in g : S_M|||}{|g|} > \delta_{GSA} \tag{8.11}$$

This condition makes sure that the ratio of genuine sensors in a group outweighs the ratio of malicious sensors by δ_{GSA} or vice versa. GSA keeps attesting all nodes in a group until the condition is met. When $\delta_{GSA} = 1$, the condition is never met and GSA coincides with $D\&A$, but with lower values, the number of attestation is lower compared to $D\&A$ and GSA may be more convenient.

To maximise energy efficiency, GSA may attest only one node per group. However, this choice is only accurate if the nodes in a group are all genuine or all malicious, which in general cannot be guaranteed. Indeed, measurements inspection is highly accurate in the grouping from the measurements perspective, i.e. the measurements in one group are either genuine or malicious. However, integrity violations at the node layer cannot be identified if the measurements are genuine. When used as a standalone technique, the identification of such malicious sensors is just delayed to the moment when the measurements become malicious. With GSA, if a node runs malicious software but reports genuine data at the time it is attested, we would infer that all nodes in the same group are malicious. This, in turn, causes wrong detection of non-compromised nodes whose measurements correlate with that of the malicious node.

The optimal value of δ_{GSA} makes an exhaustive attestation of groups that are mixtures of genuine and malicious nodes, and only one for groups where nodes are either all genuine or all malicious. The value used in the experiments is 0.25, which gives a percentage of attested nodes around 25%.

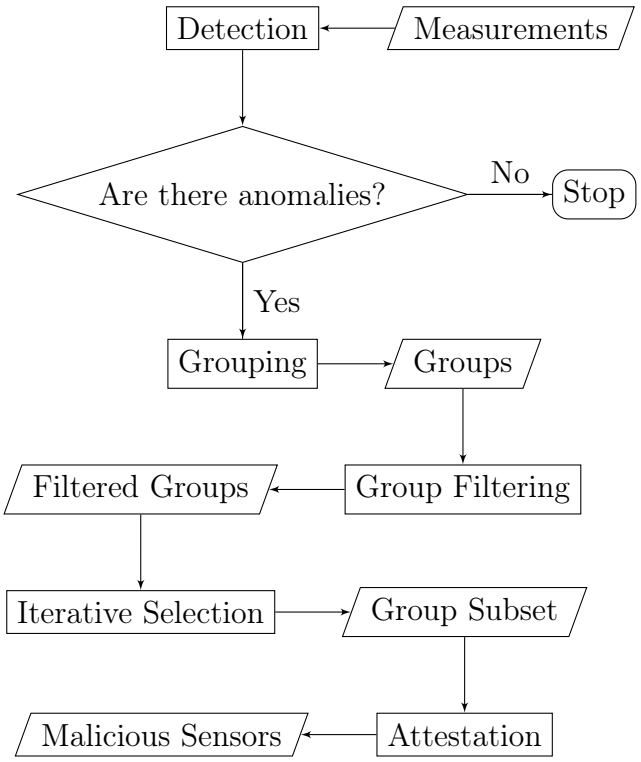


Figure 8.3: GSA Scheme.

Performance

The performance of GSA is a mixture of the performance of measurements inspection in detecting anomalies and correctly grouping sensor nodes, and the attestation performance. Note that for each group only a subset of sensors is tested, hence the TPR is:

$$\begin{aligned} TPR_{GSA} = \delta_{GSA} TPR_{AD} TPR_{AT} + \\ (1-\delta_{GSA}) \Big(P(S'_M | g(S) = g(S') S_M) P(A^{S'}_{T_F} | S'_M) + \\ P(S'_G | g(S) = g(S') S_M) P(A^{S'}_{T_F} | S'_G) \Big) TPR_{AD} \end{aligned} \tag{8.12}$$

Where $P(A^{S'}_{T_F}|S'_M) \approx TPR_{AT}$ and $P(A^{S'}_{T_F}|S'_G) \approx FPR_{AT}$.

We denoted with TPR_{AD} the TPR of the anomaly detection part of measurements inspection, and with $P(S'_M|g(S) = g(S')S_M)$ the probability that the sensor chosen for attestation S' is malicious given that the considered sensor S is malicious and belongs to the same group as S'. Namely, if the malicious sensor node is an attested node (they are δ_{GSA} of the total on average), then detection is correct if anomaly detection triggers and the node fails attestation.

Otherwise, the characterisation is correct only if the correct group is selected. Thus, the FPR becomes:

$$\begin{aligned} FPR_{GSA} = \Big(P(C=0)FPR_{AD} + P(C>0)TPR_{AD}\Big) \\ \Big(FPR_{AT}\delta_{GSA} + (1-\delta_{GSA})\Big(P(S'_M|g(S)=g(S')S_G)TPR_{AT} \\ +P(S'_G|g(S)=g(S')S_G)FPR_{AT}\Big)\Big) \end{aligned} \tag{8.13}$$

The attestation frequency here is equal to:

$$AF_{GSA} = \Big(P(C=0)FPR_{AD} + P(C>0)TPR_{AD}\Big)\delta_{GSA} \tag{8.14}$$

Compared to D&A, the attestation frequency is always lower, while the TPR and FPR are as good or better if the anomaly-based grouping is correct, and worse otherwise. In the next combination scheme we aim to achieve even lower attestation frequency by attesting only the sensors marked as malicious by measurements inspection.

8.5.5 Cascade

In large WSNs, the number of groups may be high, so even though GSA would bring remarkable benefits, the overhead introduced by attestation may still be non-negligible. To reduce even further the attestation frequency, we observe that the main cause of failure for measurements inspection is that anomalies may arise as a consequence of an unforeseen or unprecedented scenario that caused a wrong estimate of the measurements probability. Instead, software-based attestation produces undisputed proofs of a sensor's maliciousness if the strict time constraints are always guaranteed by genuine sensor nodes. So, it may be convenient to use software attestation to confirm a sensor node's maliciousness after it has been identified as such by measurements inspection, as shown in Figure 8.4. We refer to this approach as *Cascade*.

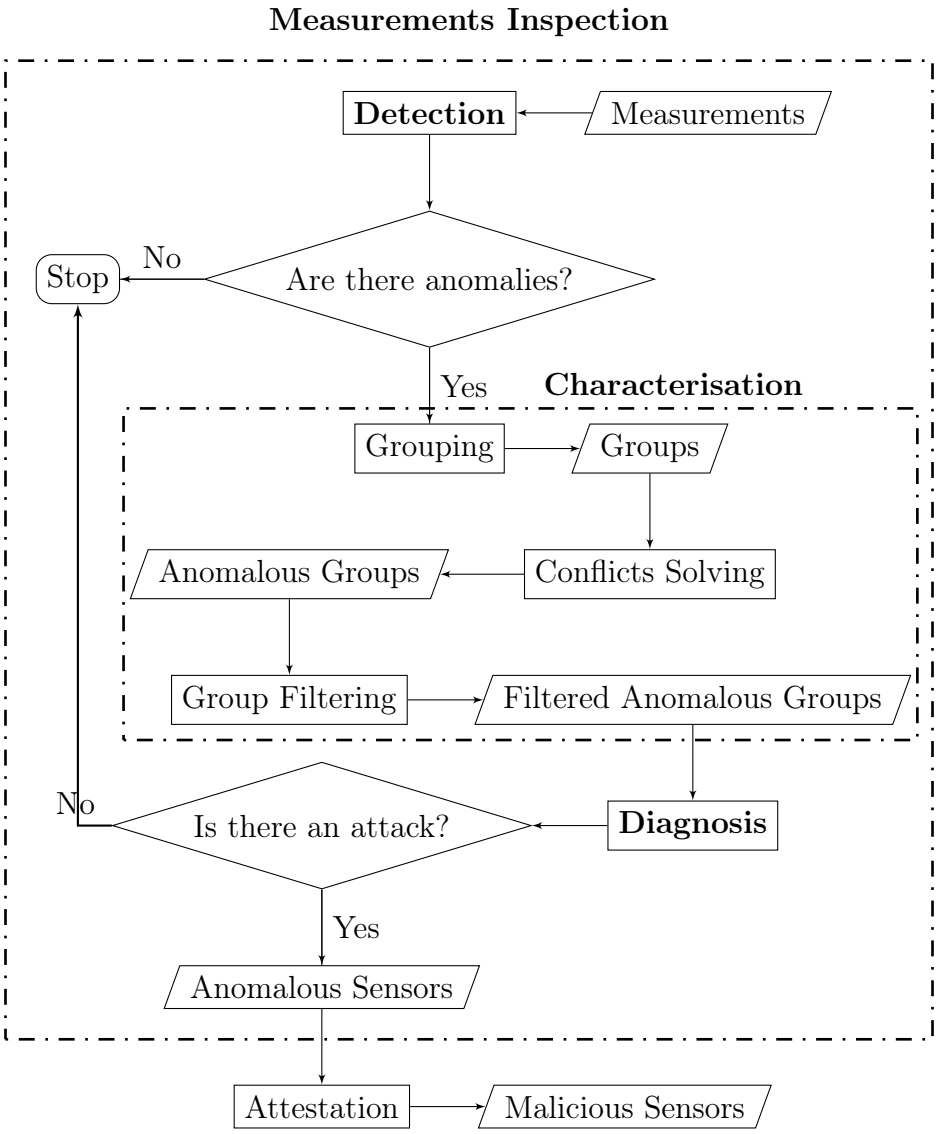


Figure 8.4: Cascade Scheme.

Performance

The TPR and FPR of the scheme are:

$$TPR_{Cascade} = TPR_{MI}TPR_{AT} \tag{8.15}$$

$$FPR_{Cascade} = FPR_{MI}FPR_{AT} \tag{8.16}$$

The Cascade scheme power overhead is lower than attestation since the attestation frequency

is:

$$AF_{Cascade} = \mathrm{E}[C|C > 0]P(C > 0)TPR_{MI} + P(C = 0)FPR_{MI} \tag{8.17}$$

This is also lower than the attestation frequency of GSA, when the measurements inspection FPR and the number of malicious nodes are low. Instead, with higher FPR, the Cascade scheme makes many attestations, while the number of attestations for the GSA scheme is upper bounded by the number of groups given in output by characterisation.

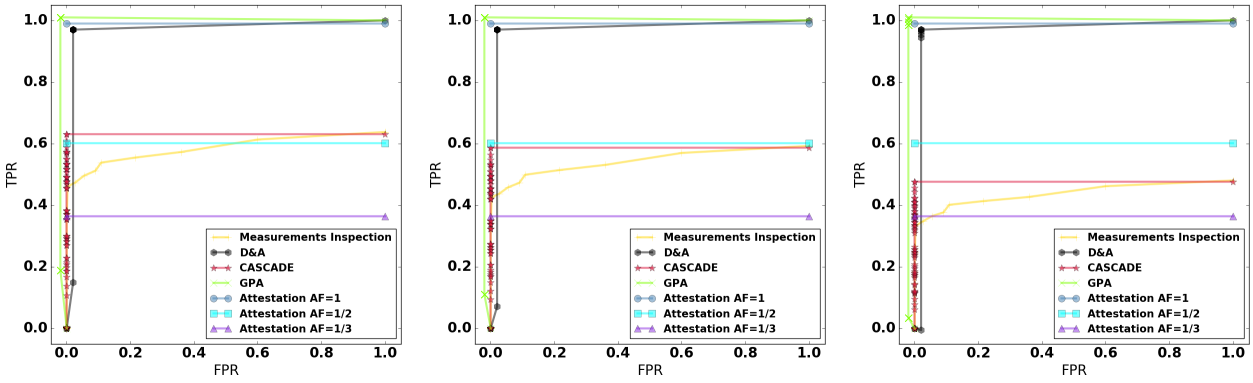
Note that if the software attestation time constraints are reliable then $\mathrm{TPR}_{AT} = 1$ and $\mathrm{FPR}_{AT} = 0$, so, compared with attestation, the Cascade solution trades a reduced power overhead for a lower TPR, which coincides with that of measurements inspection.

8.6 Analytical Evaluation

In this section, we evaluate the D&A, GSA and Cascade techniques in a WSN of 200 sensor nodes, of which a varying number C are compromised. Sensors collect a measurement about every 4 minutes. The probability of attack is as high as 10^{-2}, implying that, on average, there is an attack about every 7 hours. This value is in the same order of FPR_{AD}, so the analytical results remain substantially unchanged also for lower attack probabilities. When an attack occurs, the probability that a sensor is malicious has a uniform prior distribution across all sensors (i.e. each sensor is malicious with probability $1/C$).

The evaluation is done by calculating: 1) The *Receiver Operating Characteristic* (ROC) curves, i.e. the relationship between TPR and FPR in the identification of malicious nodes. 2) The attestation frequency, i.e. the time between two attestations divided by measurements transmission period, and averaged across all nodes.

The ROC curves of the combination schemes are obtained through the results of Section 8.5. The ROC curves for measurements inspection are obtained by interpolating the experimental curves obtained in Section 7.2.5. The ROC curves for attestation are obtained by modelling



(a) $C = 50$ (b) $C = 100$ (c) $C = 150$

Figure 8.5: ROC curves comparison varying the number of malicious sensors C[1].

the degradation in the reliability of attestation, when the time between two attestations is close to the time needed to swap the genuine software with a malicious one, and vice versa. Thus, the probability that an attacker manages to compromise a measurement and pass attestation behaves like an exponential probability distribution:

$$1 - e^{-\lambda(\frac{1}{\mathrm{AF}}-1)} \tag{8.18}$$

The parameter λ models the time needed to replace the genuine software and restore it. In particular, we show the performance of attestation when $\lambda = 0.5$, which is the case where the probability that a malicious node passes attestation is close to 0 when $AF = 1$, and quickly increases to about 0.4 and 0.6 for $AF = 1/2$ and $AF = 1/3$, respectively.

8.6.1 ROC curves comparison

Figure 8.5 shows the ROC curves for both individual and combined techniques. In particular, the TPR of software-based attestation is shown for $AF = \{1, 1/2, 1/3\}$, corresponding to one attestation run as soon as a new measurement is collected, or once every two or three measurements are collected. The TPR decreases with the attestation frequency since a quick attacker may substitute the original software with a malicious version, inject malicious data, and replace the original software before attestation is run again.

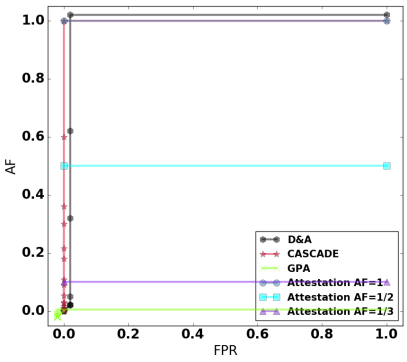


Figure 8.6: AF curves comparison[1].

We observe that the highest TPR of measurements inspection, achieved at the rightmost point of the ROC curve, is an upper bound on the TPR of Cascade. Nevertheless, while measurements inspection achieves such TPR with FPR close to 1, Cascade achieves it with FPR close to 0. The performance of D&A and GSA is not limited by measurements inspection since they mainly exploit the detection step, whose performance is comparable to attestation's. Indeed, their ROC curves nearly overlap with that of attestation.

8.6.2 Attestation Frequency Comparison

To make a fair comparison, we also consider the attestation frequency of each scheme, since with $AF \approx 1$ there is no advantage in using a combination scheme in place of simple attestation.

Figure 8.6 shows that the points where Cascade performs at its best are expensive, as the attestation frequency is close to 1. Instead, when the attestation frequency of D&A is as low as 0.02, the TPR in Figure 8.5 is close to attestation. Finally, GSA has a small attestation frequency which saturates and holds even for the highest values of TPR. In conclusion, Cascade is generally not convenient, since it covers a point in the reliability-cost space where reliability is close to measurements inspection and cost is close to attestation. For D&A and GSA, the reliability is almost as high as attestation. The cost for GSA is always comparable to measurements inspection. For D&A, the cost can be kept low for a low decrease in reliability.

[1] A perturbation of ± 0.02 was introduced in GSA and D&A to better distinguish the curves.

8.7 Numerical Simulations

In the previous section, we abstracted from the computations and network protocols that enable the application of each combination scheme, so we address them below. Preliminarily, we address the process of tailoring software-based attestation for measurements reliability, which has not yet been analysed in the literature.

We validate the analytical ROC curves and complete them with the time needed to achieve a certain TPR, which corresponds to the latency in the reaction to malicious data. Finally, we make an accurate estimation of the spent energy thanks to a fine-grained analysis of the sensor nodes' activities, including the transmission of each network packet. This allows us to calculate the impact of each approach on nodes' battery life.

To investigate these parameters accurately, we have set up simulations of a realistic application scenario in the open-source *Castalia* [NIC07] simulator. Simulations allow running both the individual and combined schemes under exactly the same scenario, giving accurate comparisons that highlight the gains in performance and energy consumption, and allow evaluation with more devices than would be practical with real nodes.

8.7.1 Simulation Settings

The simulations consider 200 nodes in a star-topology network, where the base station is located at the centre performing the role of network coordinator, measurements sink, and attestation verifier. Sensors measure the temperature and send the observed value to the sink about every 4 minutes. Nodes are equipped with the CC2420 RF transceiver [Ins13], which is common for its low-power transmissions. Namely, this device spends 57.42 mW while transmitting (at 0dBm), 62 mW while receiving and 1.4 mW while in sleep mode [Ins13].

For C out of 200 sensor nodes, the temperatures sent to the sink are replaced with higher values to trigger the detection of a wildfire, thanks to the mimicry-based test suite described in Section 7.2.2. The simulations are run for all individual and combination schemes, with three different

values of C: 50, 100, and 150. The parameters relevant to the simulations are summarised in Table 8.2. After 5 hours the simulation is stopped and we calculate the TPR and FPR for the detection of malicious nodes, and the energy consumption.

Table 8.2: Simulation Parameters.

Parameter	Value
CPU clock	500 kHz
Transmission Output Power	0 dBm
CAP duration	125.83 s
CFP duration	0 ms
Superframe Duration	125.83 s
Inactive Portion	125.83 s
Beacon Interval = Sampling Period	251.66 s
Maximum RTT	0.06 s

For the transmission of measurements and attestation-related data, we use the 802.15.4 MAC protocol [GCB03] because it provides us low energy consumption and a hierarchical architecture. 802.15.4 defines both the physical and data-link layer of the ISO OSI stack, including a CSMA-CA protocol for handling collisions by sensing the channel before transmissions and using the random backoff technique to make sensor nodes wait before retransmitting.

The data-link layer of 802.15.4 defines a frame structure which includes, among other, bit strings for ACK management, sequencing number, and source/destination addresses. The protocol makes use of a coordinator node that dictates how and when nodes can communicate through the use of a superframe structure, illustrated in Figure 8.7, which constitutes an active and an inactive period, where nodes are allowed to transmit or switch off their communication devices to save energy, respectively.

Furthermore, at the application level we define a transmission schedule so that nodes transmit their measurements one after the other, thus avoiding collisions. The coordinator informs nodes about the duration of the next superframe sections with a *beacon frame*. The latter may be

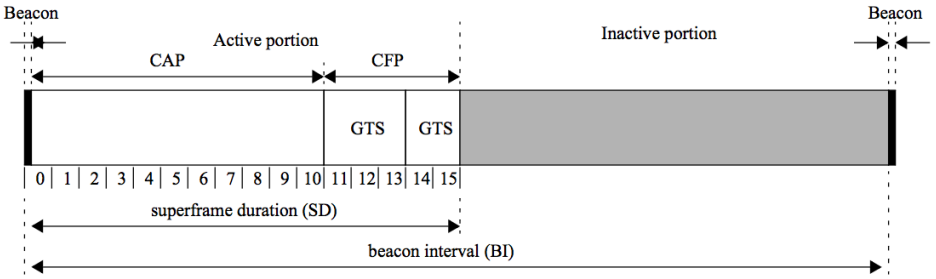


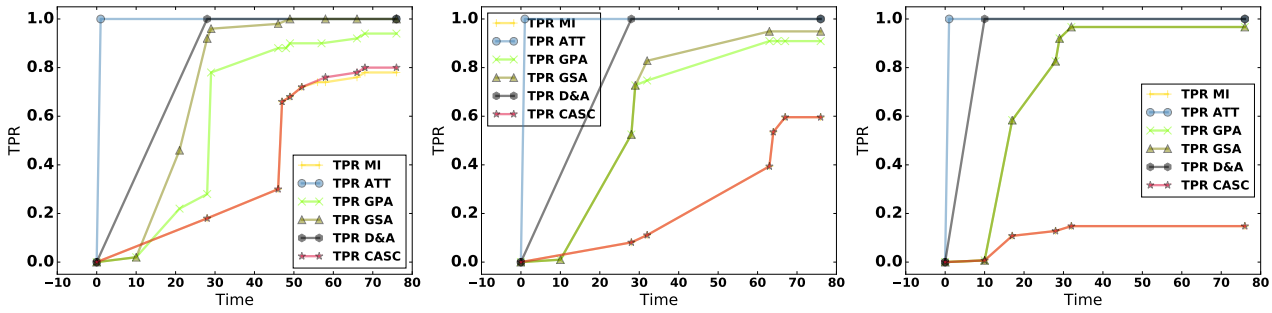
Figure 8.7: 802.15.4 MAC superframe.

followed by a Contention Access Period (CAP), where all nodes communicate with CSMA-CA. Then a Contention Free Period (CFP) may follow, which is divided into Guaranteed Time Slots (GTS), where only one node is allowed to transmit. Finally, an inactive period ends the superframe, where no communication occurs and hence the sensor nodes can switch off their communication devices to save energy. The durations of these periods are shown in Table 8.2.

In 802.15.4, when only one node is the coordinator, it gets the name of PAN coordinator. Once it receives all measurements it can trigger the measurements inspection, attestation, or a combined scheme. Note that when just measurements inspection or no scheme is in use, the nodes can go to sleep right after they transmit their data. Whereas, if a scheme with attestation is in place, the nodes must keep awake as they do not know in advance whether they will be attested. Since the communication pattern is not fixed we cannot use the canonical 802.15.4 guaranteed time slots. Also to avoid collisions, so that the network round-trip time can be reliably estimated, attestation is performed one node at a time. When the sink is done attesting all nodes for a collection round, it sends a message signalling to nodes that they can go immediately to sleep until the next active period.

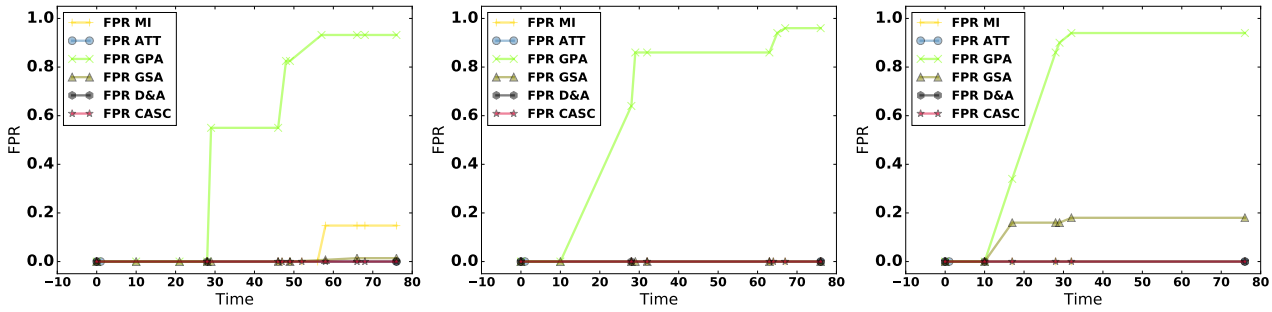
8.7.2 True Positives / False Positives Results

We calculate the TPR, defined as the number of malicious nodes that are detected at least once during the simulated attack. We do not assume the presence of *intrusion reaction systems*, therefore a detected node keeps carrying out the attack. This constitutes an upper bound on the system's performance since both the energy consumption and the detection of further nodes



Cumulative TPR in time.

(a) $C = 50$ (b) $C = 100$ (c) $C = 150$



Cumulative FPR in time.

(d) $C = 50$ (e) $C = 100$ (f) $C = 150$

Figure 8.8: TPR/FPR curves comparison with different numbers of malicious sensors C.

improves with the correct characterisation of malicious nodes. Indeed, a malicious node that is detected does not need to be evaluated again by either attestation or measurements inspection. The performance of the latter is also likely to increase, since fewer malicious nodes gain from collusion.

Figure 8.8 shows the TPR and FPR for each scheme with different numbers of malicious sensors. Rather than their final values, we show the cumulative TPR and FPR time series. Indeed, the time needed to achieve a certain TPR is decisive to minimise the attack's damage. Analogous considerations hold for the FPR. The combination technique with the best performance is certainly D&A, whose curves are the closest to those of attestation. Since D&A uses the detection step of measurements inspection, the TPR curves, in Figures 8.8a, 8.8b, and 8.8c, jump from 0 to 0.99 after 28, 28, and 10 collection rounds respectively, which are the points where anomalies are detected and attestation is triggered.

The curves of the Cascade scheme are close to measurements inspection, but an improvement is brought in the FPR curve in Figure 8.8d, which is decreased from 0.15 to 0. Higher TPR can be achieved by increasing the measurements inspection FPR and letting attestation take care of the false positives by attesting them. However, this choice increases the attestation frequency, as discussed in Section 8.6. The simulations also confirmed that D&A can achieve an FPR close to 0 and a TPR around 0.99. In the case of GSA the TPR is around 0.96 and the FPR is close to 0, especially when 50 and 100 sensor nodes out of 200 are malicious. When 150 sensors are malicious the FPR is evidently higher than D&A, but the TPR that eventually gets close to 1 makes GSA still a valid choice. Indeed, when the system detects that 150 out 200 sensor nodes are malicious, it means that a severe attack has taken place. In such scenario, generating false positives is not the main concern since the system needs a thorough recovery and reconfiguration process anyway.

8.7.3 Energy Consumption Comparison

In Table 8.3, the energy consumption during simulation time, averaged across all nodes, is reported for each scheme. First, we note that attestation has an energy consumption which is between 33 and 58% higher than measurements inspection, and that the latter does not introduce a perceptible increase with respect to the case where no security scheme is applied (NONE in Table 8.3). Note that, to have a complete estimate of the techniques' costs, we need to also consider that measurements inspection may require a higher density of sensors to achieve good detection performance.

Compared with measurements inspection, GSA and Cascade generally demand less than 1% extra energy. The increase for D&A, instead, is between 3 and 10%. To understand practical implications of such differences in energy consumption, we used the results in Table 8.3 to retrieve the expected number of days until the batteries would drain. As a reference, we assumed a typical power source of two alkaline long-life AA batteries, which store an energy of 18720 Joules [Ene]. As reported in Table 8.4, we see that the average duration of the batteries without measurements inspection nor attestation is about 48 days. Note that for

Table 8.3: Average Energy Consumption (Joules) after 5 hours.

C \ Scheme	NONE	MI	ATT	GSA	D&A	CASC
50	79.54	82.86	109.84	83.65	91.14	83.59
100	81.51	80.96	116.65	80.91	83.73	80.04
150	82.01	79.67	125.81	80.45	83.35	80.07

Table 8.4: Days To Battery Depletion.

C \ Scheme	NONE	MI	ATT	GSA	D&A	CASC
50	49.03	47.06	35.51	46.62	42.79	46.66
100	47.85	48.17	33.43	48.20	46.58	48.73
150	47.55	48.95	31.00	48.48	46.79	48.71

some applications such a battery life may not be enough. The main reason behind it is the power consumption of the CC2420 RF transceiver. A more advanced device may give better battery life; however, our main interest is the comparison of the energy spent when using different techniques rather than its absolute value.

When running measurements inspection, there is no significant change in battery life. With attestation instead, the batteries last 10 to 15 fewer days. GSA and Cascade generally cause the batteries to last 1 fewer day, while with D&A battery life diminishes by 2 to 4 days.

8.8 Conclusions

Combining measurements inspection with attestation achieves high accuracy in identifying malicious nodes whilst significantly reducing power consumption. We proposed three combination schemes: *Detect and Attest*, *Group Subset Attestation*, and *Cascade*. The first gives most relevance to the attestation step, the third stresses the measurements inspection steps, while the second is at a point in between. In this way, the spectrum of combinations is well covered.

Measurements inspection detects malicious measurements by inspecting the measurements themselves to find changes in their internal correlation structures. When performed in a centralized way, for instance by the base station, this approach requires no additional computations

from the sensor nodes themselves. From this perspective, it can be considered very lightweight. However, the accuracy in distinguishing genuine from compromised nodes is limited by the unpredictability of the sensed phenomenon, which introduces uncertainty in the measurements correlations. Attestation, on the other hand, ascertains the integrity of a node by verifying its memory contents through a challenge-response protocol. This approach has a considerable overhead on sensor nodes, which have to exchange additional messages and for a period cannot do anything other than calculating a computationally intensive response. Nonetheless, attestation can be very reliable if the challenge can only be responded to in time by the genuine nodes.

In this Chapter we:

1. Presented different ways of combining the two aforementioned integrity verification mechanisms.

2. Made a performance evaluation of each combination both analytically, at the techniques' components abstraction level, and numerically, through in-depth simulations.

Contribution 1 has been achieved by studying how to join advantages and neutralise drawbacks. We have conceived three different approaches: *Detect and Attest*, *Group Subset Attestation*, and *Cascade*. The first gives most relevance to the attestation step, the third stresses the measurements inspection steps, while the second is at a point in between. In this way, the spectrum of combinations is well covered.

Contribution 2 consists of evaluating all the combination schemes both analytically and through simulation. The Cascade scheme has shown to be limited by the measurements inspection's maximum TPR, which is achieved only with frequent attestations. Instead, both D&A and GSA offer a considerable gain in performance, which resulted very close to attestation's, but for significant less energy. This is confirmed by the energy results from the simulations and is due to the dramatic reduction in the number of attestations, which is observable in the analytical evaluation. A good trade-off between energy and performance is achieved by both D&A and GSA schemes. The former should be preferred when a couple of fewer days in battery

life do not have a considerable impact on costs. The latter, instead, is the best solution when the cost for the maintenance and materials involved in the battery replacement process is critical. While the individual techniques forced to choose between accuracy close to 100% with power overhead of 33-58%, or accuracy close to 50% with power overhead close to 0, the combination schemes allow us to choose accuracy in the range 96-99%, with a power overhead in the range 1-10%.

This work considered software-driven malicious data injections where the WSN is partially compromised. However, it can be extended to consider more general attack models, such as an adversary that physically tampered with the sensor nodes hardware to increase their clock speed, or that tampers with almost all sensor nodes. Attestation cannot be applied in the first scenario, while measurements inspection has poor detection accuracy in the second. In such extreme cases there is the need to give to the two techniques the possibility of raising alarms on their own.

Chapter 9

Conclusion

9.1 Summary of Thesis Achievements

This thesis provides progress in the field of security in Wireless Sensor Networks (WSNs), which are ubiquitously used in emerging fields such as Internet of Things (IoT) applications, but they are vulnerable to compromise and the integrity of the measurements collected by sensors needs to be ensured. This is essential as wrong measurements could lead the system to a wrong reaction, which may have considerable detrimental effects in applications such as critical infrastructure monitoring, disaster management, and human health.

To cause such kind of damage, an attacker can exploit *malicious data injections*, consisting of manipulations that enable an attacker to gain control over the measurements collected in the WSNs, which give control over the system's functionalities in turn. An attacker may, for instance, conceal the presence of problems, or raise false alarms through the injection of well-designed malicious measurements. To deal with this problem, some techniques belonging to the family of *measurements inspection*, have been proposed [RLP08; Lop+10; Jur+11; Xie+11; IL15b]. However, we have analysed these techniques in detail (Chapter 3) and concluded that they have been evaluated under simplistic assumptions, such as independent attacks from different compromised sources and the introduction of a random bias in the measurements. In this thesis instead, we have considered *collusion*-enabled attacks, where multiple compromised

measurements are injected according to a common plan. Moreover, the compromised measurements are not considered to be random, which is rather the case for faulty behaviours, but they are designed to reproduce a plausibly genuine behaviour. A genuine behaviour under wrong circumstances may have disastrous effects, such as evacuating a building when there is no emergency, or not dispensing insulin when the level of glucose in a patient is high.

We have advanced the current state-of-the-art to consider collusion attacks in WSN applications where the physical phenomenon may be subject to significant variations in space, especially in the presence of events of interest, such as fires, volcanic eruptions, and physiological conditions (Chapter 5). We have proposed a novel anomaly detection approach based on pairwise linear regression, to produce measurements estimates that are reliable in the presence of collusion attacks. Indeed, this method is based on robust aggregates of the pairwise estimates that have been designed specifically to resist collusion. We have shown that our algorithm can detect violations to the measurements integrity also under collusion attacks that spoof false events or mask real events. This technique is lightweight as it is based on simple statistical operators (weighted mean and median, Pearson correlation etc.) and gives good results when a single event occurs. Moreover we have given a methodology to apply the same technique to applications that are much different in nature, which makes our approach valid regardless of whether the measurements indicate blood pressure, temperature, vibrations, etc.

We have proposed a novel approach that ascertains the measurements integrity in the most general case, i.e. where many events can manifest at once in unpredictable ways (Chapter 6). The novel method is based on a test on the cross-scale relationship among the measurements, achieved by means of the *wavelet transform*. The new anomaly detection algorithm checks that the measurements variation observable between neighbouring sensors can be explained by the general status of the physical phenomenon, i.e. the events that are occurring and that can be observed with a higher granularity of analysis. We have shown that thanks to this analysis we can detect measurements integrity violations even when the attacker exploits the changes in correlations brought in by events, such as by spoofing an event next to a genuine one, or partially masking an event to divide it into multiple events. Moreover we have introduced a new algorithm to identify the sensors that are likely responsible for the manipulations (*charac-*

terisation) and to distinguish between malicious interference and genuine faults (*diagnosis*).

We have tested the robustness of our methodology by modelling the problem as an *adversarial machine learning* problem (Chapter 7). Indeed, we have observed that the attacker's goal is to classify malicious data, but at the same time the attacker will try to *evade* such classification and modify the injected data accordingly. This characteristic introduces many differences in the problem compared with the classification of, e.g. faults or events. We have explored in detail two different approaches to the problem of evading anomaly detection:

1) The *mimicry* approach, based on building malicious data from genuine data to spoof and mask events whilst maximising the effort to stay undetected. This approach has proven particularly useful to analyse the trade-off between True Positive Rate (TPR) (malicious data correctly classified) and False Positive Rate (FPR) (genuine data classified as malicious). We have applied the mimicry approach on both a synthetic and a real dataset, and run both the wavelet-based approach and the algorithm described in [Rez+13], which is based on *Iterative filtering* (IF). We observed that our algorithm achieved TPR ≈ 1 with FPR ≈ 0 whereas IF cannot achieve high TPR without a significant increase in FPR, especially when events manifest. Indeed, our algorithm has the advantage of learning the properties of genuine events beforehand, whilst IF judges all the measurements without such knowledge.

2) The *worst-case* approach, where the damage to event detection and the evasion of anomaly detection are guaranteed by solving an optimisation problem. With this approach we are able to provide an assurance to a network operator about the degree of resilience offered by measurements inspection in a deployed WSN, or even drive the design of the WSN itself by indicating which is the density of sensors that guarantees the resilience to a certain number of compromised devices. We have tested this approach on a wildfire monitoring WSN and gained some valuable knowledge about the problem tackled in this thesis. In particular, we have concluded that spoofing attacks are more successful when a few conditions are sufficient to trigger event detection, whilst the masking task can be achieved with less effort when many conditions are necessary to detect events. We have also observed that malicious data injections are more difficult to detect when neighbouring sensors are compromised and

when located at the boundaries of the WSN deployment. Moreover, we have shown that the *worst-case* approach can be applied to compare the assurance in terms of resilience to compromised sensors that is given by different measurements inspection algorithms. In particular, we have shown that our wavelet-based detection algorithm provides substantially higher resilience gains compared with the anomaly detection algorithm described in [CP07], based on *principal component analysis*.

Having concluded that the attacker needs to control large-enough sets of contiguous sensors to achieve a successful and undetected spoofing or masking attack, we have studied the impact of sparse trusted devices on the performance of measurements inspection. The presence of trusted devices may be guaranteed through maintenance operations, such as manual inspection and in field reconfiguration. However, this is an expensive approach which is against the WSN principles of self-monitoring and self-healing, thus we have devised a combination with *software attestation* (Chapter 8), which automatically ascertains the integrity of the software running on sensor nodes, but is also more expensive compared to measurements inspection. Indeed, to keep high confidence in the measurements integrity, many attestations are needed, which introduce the overhead of a challenge-response protocol and that of calculating the response. The required number of attestations increases significantly with the number of collected measurements; however, thanks to the integration with measurements inspection, we were able to significantly reduce them. We designed different combination schemes that achieve an excellent trade-off between cost and resilience, whereas the single techniques forced to choose between resilience and cost. The integration of measurements inspection with attestation enables a WSN operator to choose the most appropriate combination scheme, and customise it according to the application requirements.

9.2 Final Remarks

Throughout the presentation of this thesis, we have had the chance to make some significant contributions towards the solution of practical problems that are faced in real WSN applications.

Wireless sensor networks have become pervasive in both business, personal, and public applications, so they are a critical resource to secure. Unfortunately, the high importance given to such systems is not matched by an adequate attention to their security, despite the degree of exposure that is suffered because of the nature of the problem itself.

This thesis has raised awareness of the several threats that may lead an attacker to get in control of parts of a WSN system, which enable him to modify the sensed measurements. Moreover, we have shown how such control enables to cause some disastrous effect such as loss of life and shutdown of critical infrastructure.

This is possible if the attacker is able not only to alter the measurement data, but also to remain undetected. So, we have proposed a set of methods and methodologies, belonging to the body of work known as measurements inspection, to enable the detection of malicious interference and remain resilient even when many sensors have been compromised. Our tools have proven particularly effective, and provided significant improvements over the state-of-the-art in terms of detection rate against false positive rate. We have always constrained our algorithms to keep false positives reasonably low for the practical needs of the application, as we have witnessed that false positive rates that are considered acceptable in literature, often lead to frequent shutdown of the sensor, reconfiguration and/or restart, which can be so expensive that the cost of detection may be prohibitive.

We have designed our techniques to exploit the fraction of genuine devices that survive in a compromised WSN. To reduce also the risk in extreme scenarios, however, it may be desirable to resort to maintenance, which ascertains the integrity of the devices through manual inspections, as well as the integrity of the software through reconfigurations.

As an alternative to maintenance, we offered a combination of measurements inspection with software attestation, which produces new highly reliable and cost-effective techniques to ascertain the integrity of the WSN measurements. With this approach, the need for maintenance is minimised, and given a budget for defending against cyber-security threats, we enabled to select the optimal balance of technologies that guarantee the system's resilience and minimise maintenance cost.

We have devised that resilience-enabling techniques, even though extremely effective, have limited applicability if it is not possible to quantify the resilience gains offered. Indeed, there is a need for assurance, i.e. the quantification of the maximum risk that a WSN is subject to. We have introduced novel algorithms and numerical tools for the automatic creation of the most threatening attack scenarios, as well as for the calculation of the attacker's cost, which translates into the system's risk of being significantly damaged. So, we have transformed the abstract concept of "degree of resilience" into real measurable costs. Moreover, we have also enabled a WSN owner to tune such risk depending on the application requirements, thanks to an automated adjustment of the WSN deployment, which assures a maximum risk at design time.

9.3 Future Work

Measurements inspection techniques exploit the inter-measurements correlations that hold under genuine circumstances to detect malicious data injections. The probability of success depends on many factors which we have covered in this thesis: the time available for the attack, the knowledge of other measurements transmitted in the past and, above all, the number of compromised sensors. In this thesis, we have considered WSN applications such as fire monitoring, volcano monitoring, earthquake monitoring, and wearable and implantable health devices. Their inter-measurements correlations enabled our novel measurements inspections techniques to detect inconsistencies in the data, even with a large number of compromised sensors.

However, in other applications, the measurements coming from different sensors may be less correlated because of irregularities in the environment that cause the information contained in the spatial distance to become unreliable. It is the case, for instance, of water monitoring in large areas involving different rivers and basins, or areas characterised by many obstacles. In this case, if a single event manifests at a time, the estimation-based framework described in Chapter 5 is still applicable, since it calculates the correlations offline, thus learning implicitly discontinuities in the environment. The main limitation is with our wavelet-based approach,

which is based on Euclidean distance. Indeed, obstacles and irregular environments can cause low degrees of correlations even for two close sensors. In the future, there is the need to cater for this condition with more general tools, such as the *wavelet transform on graphs*, where spatial distance should be replaced by an estimate of the correlations, which constitute the graph's edges. The accuracy of detection will probably be lower compared with an application where the environment is mostly regular, so it may be worthwhile relying on the advantages obtained from the combination with software attestation (Chapter 8) to mitigate this problem.

We also need to consider that the deployment size has a significant impact on the measurements inspection accuracy. In Chapter 5, we have considered small-scale WSNs such as home fire alarms and healthcare monitoring, and observed that the attacker needed to compromise respectively 8 out of 15 and 7 out of 8 sensors to run a successful undetected attack. Instead, for wildfire monitoring WSNs the attacker needed to compromise more than 150 out of 200 sensors. Even though the percentage of compromised sensors is comparable, there is a significant difference in their absolute values. So, if we assumed that the cost for compromising a sensor does not vary with the deployment size, our techniques would be suitable mainly for large-scale deployments. Nevertheless, the attacker's cost for compromising the sensor nodes may be increased through proactive security mechanisms. For instance, the devices may be protected with tamper-proof hardware, and the risk of malware propagation may be reduced with software diversity. In small-scale deployments the cost for these mechanisms needs to be faced only for a few sensors, so it is more viable compared with large-scale deployments. The trade-offs involved would benefit from a more detailed investigation which we plan to address in our future work.

Finally, we wish to continue our studies on the coexistence between anomaly detection and attestation. In particular, we wish to cover also the scenarios where only one out of the two techniques can identify an integrity violation, whereas in Chapter 8 we have considered only the scenarios where both techniques can detect it. Among others, we wish to consider an adversary that physically tampered with the sensor nodes hardware to increase their clock speed, or that tampers with almost all sensor nodes. Attestation cannot be applied in the first scenario, while measurements inspection has poor detection accuracy in the second. In such extreme

cases there is the need to give to the two techniques the possibility of raising alarms on their own. So, we plan to develop a complete tool that can address all the possible attacks to the measurements integrity by coordinating the contribution of heterogeneous techniques.

Bibliography

[Abe+16] Tigist Abera et al. "C-FLAT: Control-FLow ATtestation for Embedded Systems Software". In: *CoRR* abs/1605.07763 (2016). URL: http://arxiv.org/abs/1605.07763.

[ACS17] Moustafa Alzantot, Supriyo Chakraborty, and Mani B Srivastava. "SenseGen: A Deep Learning Architecture for Synthetic Sensor Data Generation". In: *arXiv preprint arXiv:1701.08886* (2017).

[AH15] David F Andrews and Frank R Hampel. *Robust estimates of location: survey and advances*. Princeton University Press, 2015.

[ALK10] Moshaddique Ameen, Jingwei Liu, and Kyungsup Kwak. "Security and Privacy Issues in Wireless Sensor Networks for Healthcare Applications". In: *Journal of Medical Systems* 36.1 (2010), pp. 93–101. ISSN: 1573-689X. DOI: 10.1007/s10916-010-9449-4. URL: http://dx.doi.org/10.1007/s10916-010-9449-4.

[Ata+08] Idris M. Atakli et al. "Malicious node detection in wireless sensor networks using weighted trust evaluation." In: *SpringSim*. Ed. by Hassan Rajaei, Gabriel A. Wainer, and Michael J. Chinni. SCS/ACM, 2008, pp. 836–843. ISBN: 1-56555-319-5. URL: http://dblp.uni-trier.de/db/conf/springsim/springsim2008.html\#AtakliHCKS08.

[Bah+09] M. Bahrepour et al. "Use of event detection approaches for outlier detection in wireless sensor networks". In: *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*. 2009, pp. 439–444. DOI: 10.1109/ISSNIP.2009.5416749.

[Ban+10] Zorana Bankovic et al. "Distributed intrusion detection system for wireless sensor networks based on a reputation system coupled with kernel self-organizing maps." In: *Integrated Computer-Aided Engineering* 17.2 (2010), pp. 87–102. URL: http://dblp.uni-trier.de/db/journals/icae/icae17.html\#BankovicMAFVG10.

[Bao+12] Fenye Bao et al. "Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection." In: *IEEE Transactions on Network and Service Management* 9.2 (2012), pp. 169–183.

[Ber14] Timo Berthold. *Heuristic algorithms in global MINLP solvers*. Verlag Dr. Hut, 2014.

[BHL07] Luís M. A. Bettencourt, Aric A. Hagberg, and Levi B. Larkey. "Separating the Wheat from the Chaff: Practical Anomaly Detection Schemes in Ecological Applications of Distributed Sensor Networks." In: *DCOSS*. Ed. by James Aspnes et al. Vol. 4549. Lecture Notes in Computer Science. Springer, July 9, 2007, pp. 223–239. ISBN: 978-3-540-73089-7. URL: http://dblp.uni-trier.de/db/conf/dcoss/dcoss2007.html\#BettencourtHL07.

[Big+13] Battista Biggio et al. "Evasion Attacks against Machine Learning at Test Time". In: *Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Part III*. Vol. 8190. 2013, pp. 387–402.

[BIJ14] Elisa Bertino, Aleksandar Ignatovic, and Sanjay Jha. "Secure Data Aggregation Technique for Wireless Sensor Networks in the Presence of Collusion Attacks". In: *IEEE Transactions on Dependable and Secure Computing* 99.PrePrints (2014), p. 1. ISSN: 1545-5971. DOI: 10.1109/TDSC.2014.2316816.

[Bis06] C. Bishop. *Pattern Recognition and Machine Learning*. Springer Verlag, 2006.

[Bis14] A Biswas. "Scaling analysis of soil water storage with missing measurements using the second-generation continuous wavelet transform". In: *European journal of soil science* 65.4 (2014), pp. 594–604.

[Boh+08] Jens-Matthias Bohli et al. "A secure and resilient WSN roadside architecture for intelligent transport systems". In: *Proceedings of the first ACM conference on Wireless network security*. ACM. 2008, pp. 161–171.

[Bou09] Azzedine Boukerche. *Algorithms and Protocols for Wireless Sensor Networks*. Wiley series on parallel and distributed computing. Wiley-IEEE Press, 2009. ISBN: 9780470396360.

[Bry95] W. Bryc. *The normal distribution: characterizations with applications*. Lecture notes in statistics. Springer-Verlag, 1995. ISBN: 9780387979908. URL: http://books.google.co.uk/books?id=BQ7vAAAAMAAJ.

[Bukowski] Richard W Bukowski et al. *Performance of home smoke alarms: analysis of the response of several available technologies in residential fire settings*. National Institute of Standards, Technology. Fire Research Division. Building, and Fire Research Laboratory, 2007.

[CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey." In: *ACM Comput. Surv.* 41.3 (July 28, 2009). URL: http://dblp.uni-trier.de/db/journals/csur/csur41.html\#ChandolaBK09.

[Che+08] Wen-Tsuen Chen et al. "Design and implementation of a real time video surveillance system with wireless sensor networks". In: *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*. IEEE. 2008, pp. 218–222.

[Com14] U.S. Nuclear Regulatory Commission. *Backgrounder on the Three Mile Island Accident*. 2014. URL: https://www.nrc.gov/reading-rm/doc-collections/fact-sheets/3mile-isle.html.

[CP07] V. Chatzigiannakis and S. Papavassiliou. "Diagnosing Anomalies and Identifying Faulty Nodes in Sensor Networks". In: *Sensors Journal, IEEE* 7.5 (2007), pp. 637–645. ISSN: 1530-437X. DOI: 10.1109/JSEN.2007.894147.

[CPS06] Haowen Chan, Adrian Perrig, and Dawn Song. "Secure hierarchical in-network aggregation in sensor networks". In: *Proceedings of the 13th ACM conference on Computer and communications security*. ACM. 2006, pp. 278–287.

[Cro94] Teri Crosby. "How to detect and handle outliers". In: *Technometrics* 36.3 (1994), pp. 315–316.

[CS05] Tony F Chan and Jianhong Shen. *Image processing and analysis: variational, PDE, wavelet, and stochastic methods*. SIAM, 2005.

[CW08] Emmanuel J Candès and Michael B Wakin. "An introduction to compressive sampling". In: *IEEE signal processing magazine* 25.2 (2008), pp. 21–30.

[CWW10] Ing-Ray Chen, Yating Wang, and Ding-Chau Wang. "Reliability of wireless sensors with code attestation for intrusion detection". In: *Information Processing Letters* 110.17 (2010), pp. 778–786. DOI: 10.1016/j.ipl.2010.06.007.

[DCI02] Santpal S Dhillon, Krishnendu Chakrabarty, and S Sitharama Iyengar. "Sensor placement for grid coverage under imprecise detections". In: *Information Fusion, 2002. Proceedings of the Fifth International Conference on*. Vol. 2. IEEE. 2002, pp. 1581–1587.

[DD11] Ethan W. Dereszynski and Thomas G. Dietterich. "Spatiotemporal Models for Data-Anomaly Detection in Dynamic Environmental Monitoring Campaigns." In: *TOSN* 8.1 (2011), p. 3. URL: http://dblp.uni-trier.de/db/journals/tosn/tosn8.html\#DereszynskiD11.

[DKL06] Pan Du, Warren A. Kibbe, and Simon M. Lin. "Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching". In: *Bioinformatics* 22.17 (2006), pp. 2059–2065. DOI: 10.1093/bioinformatics/btl355. eprint: http://bioinformatics.oxfordjournals.org/content/22/17/2059.full.pdf+html. URL: http://bioinformatics.oxfordjournals.org/content/22/17/2059.abstract.

[DRP05] Emilie Danna, Edward Rothberg, and Claude Le Pape. "Exploring relaxation induced neighborhoods to improve MIP solutions". In: *Mathematical Programming* 102.1 (2005), pp. 71–90. ISSN: 1436-4646. DOI: 10.1007/s10107-004-0518-7. URL: http://dx.doi.org/10.1007/s10107-004-0518-7.

[DS99] AJW Duijndam and MA Schonewille. "Nonuniform fast Fourier transform". In: *Geophysics* 64.2 (1999), pp. 539–551.

[Du+06] Wenliang Du et al. "A pairwise key predistribution scheme for wireless sensor networks." In: *ACM Trans. Inf. Syst. Secur.* 8.2 (Feb. 9, 2006), pp. 228–258. URL: http://dblp.uni-trier.de/db/journals/tissec/tissec8.\#DuDHVKK05.

[DV04] Alessandra Dalla Valle. "The skew-normal distribution". In: *Skew-elliptical distributions and their applications: A journey beyond normality*. Chapman and Hall/CRC, 2004.

[Ene] *AllAboutBatteries.com*. 2011. URL: http://www.allaboutbatteries.com/Energy-tables.html.

[Fas+07] Elena Fasolo et al. "In-network aggregation techniques for wireless sensor networks: a survey." In: *IEEE Wireless Commun.* 14.2 (2007), pp. 70–87. URL: http://dblp.uni-trier.de/db/journals/wc/wc14.html\#FasoloRWZ07.

[Fin+98] Mark A Finney et al. *FARSITE, Fire Area Simulator–model development and evaluation*. Vol. 3. US Department of Agriculture, Forest Service, Rocky Mountain Research Station Ogden, 1998.

[FL06] Prahlad Fogla and Wenke Lee. "Evading Network Anomaly Detection Systems: Formal Reasoning and Practical Techniques". In: *Conf. on Computer and Communications Security*. 2006, pp. 59–68.

[Gab46] Dennis Gabor. "Theory of communication. Part 1: The analysis of information". In: *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering* 93.26 (1946), pp. 429–441.

[GBS08] Saurabh Ganeriwal, Laura Balzano, and Mani B. Srivastava. "Reputation-based framework for high integrity sensor networks." In: *TOSN* 4.3 (June 4, 2008). URL: http://dblp.uni-trier.de/db/journals/tosn/tosn4.html\#GaneriwalBS03.

[GCB03] Jose A. Gutierrez, Edgar H. Callaway, and Raymond Barrett. *IEEE 802.15.4 Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensor Networks*. New York, NY, USA: IEEE Standards Office, 2003. ISBN: 0738135577.

[GH06] Andrew Gelman and Jennifer Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. 1st ed. Cambridge University Press, 2006. ISBN: 052168689X. URL: http://www.amazon.com/Analysis-Regression-Multilevel-Hierarchical-Models/dp/052168689X/ref=sr_1_1?s=books&ie=UTF8&qid=1313405184&sr=1-1.

[GO15] Inc. Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2015. URL: http://www.gurobi.com.

[GR06] Amir Globerson and Sam Roweis. "Nightmare at Test Time: Robust Learning by Feature Deletion". In: *Int. Conf. on Machine Learning*. 2006, pp. 353–360.

[GSS15] Ian J Goodfellow, Jonathon Shlens, and Christian. Szegedy. "Explaining and Harnessing adversarial examples". In: *Int. Conf. on Learning Representations* (2015).

[Gub+13] Jayavardhana Gubbi et al. "Internet of Things (IoT): A vision, architectural elements, and future directions". In: *Future Generation Computer Systems* 29.7 (2013), pp. 1645 –1660. ISSN: 0167-739X. DOI: http://dx.doi.org/10.1016/j.future.2013.01.010. URL: http://www.sciencedirect.com/science/article/pii/S0167739X13000241.

[GZH09] Shuo Guo, Ziguo Zhong, and Tian He. "FIND: faulty node detection for wireless sensor networks." In: *SenSys*. Ed. by David E. Culler, Jie Liu, and Matt Welsh. ACM, Nov. 18, 2009, pp. 253–266. ISBN: 978-1-60558-519-2. URL: http://dblp.uni-trier.de/db/conf/sensys/sensys2009.html\#GuoZH09.

[Han+75] E Handschin et al. "Bad data analysis for power system state estimation". In: *Power Apparatus and Systems, IEEE Transactions on* 94.2 (1975), pp. 329–337.

[HCB02] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. "Energy-Efficient Communication Protocol for Wireless Microsensor Networks." In: *HICSS*. Jan. 3, 2002. URL: http://dblp.uni-trier.de/db/conf/hicss/hicss2000-8.html\#HeinzelmanCB00.

[He+04] Tian He et al. “Energy-Efficient Surveillance System Using Wireless Sensor Networks.” In: *MobiSys*. USENIX, Nov. 30, 2004. URL: http://dblp.uni-trier.de/db/conf/mobisys/mobisys2004.html#HeKSALSYGHK04.

[Hin09] Cheryl V. Hinds. “Efficient detection of compromised nodes in a wireless sensor network.” In: *SpringSim*. Ed. by Gabriel A. Wainer et al. SCS/ACM, 2009.

[HLT05] Lei Huang, Lei Li, and Qiang Tan. “Behavior-Based Trust in Wireless Sensor Network.” In: *APWeb Workshops*. Ed. by Heng Tao Shen et al. Vol. 3842. Lecture Notes in Computer Science. Springer, Dec. 23, 2005, pp. 214–223. ISBN: 3-540-31158-0.

[HP05] Dun-Yu Hsiao and Soo-Chang Pei. “Detecting digital tampering by blur estimation”. In: *Systematic Approaches to Digital Forensic Engineering, 2005. First International Workshop on*. IEEE. 2005, pp. 264–278.

[HST10] Shih-I Huang, Shiuhpyng Shieh, and JD Tygar. “Secure encrypted-data aggregation for wireless sensor networks”. In: *Wireless Networks* 16.4 (2010), pp. 915–927.

[HT05] Chi-Fu Huang and Yu-Chee Tseng. “The coverage problem in a wireless sensor network”. In: *Mobile Networks and Applications* 10.4 (2005), pp. 519–528.

[Hua+11] Ling Huang et al. “Adversarial Machine Learning”. In: *Workshop on Security and Artificial Intelligence*. 2011, pp. 43–58.

[IL15a] Vittorio Illiano and Emil Lupu. “Detecting Malicious Data Injections in Event Detection Wireless Sensor Networks”. In: *Network and Service Management, IEEE Transactions on* 12.3 (2015), pp. 496–510. ISSN: 1932-4537. DOI: 10.1109/TNSM.2015.2448656.

[IL15b] Vittorio P. Illiano and Emil C. Lupu. “Detecting Malicious Data Injections in Wireless Sensor Networks: A Survey”. In: *ACM Comput. Surv.* 48.2 (Oct. 2015), 24:1–24:33. ISSN: 0360-0300. DOI: 10.1145/2818184. URL: http://doi.acm.org/10.1145/2818184.

[IMGL17] V. P. Illiano, L. Munoz-Gonzalez, and E. C. Lupu. "Don't fool Me!: Detection, Characterisation and Diagnosis of Spoofed and Masked Events in Wireless Sensor Networks". In: *IEEE Transactions on Dependable and Secure Computing* 14.3 (2017), pp. 279–293. ISSN: 1545-5971. DOI: 10.1109/TDSC.2016.2614505.

[Ins13] Texas Instruments. "ChipconâĂŤCC2420". In: *TEXAS INSTRUMENTS,[Online]. Available: http://www. ti. com/product/cc2420.[Accessed 31 1 2015]* (2013).

[Iria] *Albuquerque Seismological Laboratory (ASL)/USGS (1988): Global Seismograph Network (GSN - IRIS/USGS). International Federation of Digital Seismograph Networks. Other/Seismic Network.* DOI: 10.7914/SN/IU.

[Irib] *Incorporated Research Institutions of Seismology.* http://www.iris.edu/.

[Iric] *Scripps Institution of Oceanography (1986): IRIS/IDA Seismic Network. International Federation of Digital Seismograph Networks. Other/Seismic Network.* DOI: 10.7914/SN/II.

[ISL17] Vittorio P. Illiano, Rodrigo V. Steiner, and Emil C. Lupu. "Unity is Strength!: Combining Attestation and Measurements Inspection to Handle Malicious Data Injections in WSNs". In: *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. WiSec '17. Boston, Massachusetts: ACM, 2017, pp. 134–144. ISBN: 978-1-4503-5084-6. DOI: 10.1145/3098243.3098249. URL: http://doi.acm.org/10.1145/3098243.3098249.

[JAP06] D. Janakiram, V. Adi Mallikarjuna Reddy, and A.V.U. Phani Kumar. "Outlier Detection in Wireless Sensor Networks using Bayesian Belief Networks". In: *Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on*. 2006, pp. 1–6. DOI: 10.1109/COMSWA.2006.1665221.

[Jia+09] Peng Jiang et al. "Design of a water environment monitoring system based on wireless sensor networks". In: *Sensors* 9.8 (2009), pp. 6411–6434.

[JM79] J. Edward Jackson and Govind S. Mudholkar. "Control Procedures for Residuals Associated with Principal Component Analysis". English. In: *Technometrics* 21.3

(1979), pp. 341–349. ISSN: 00401706. URL: http://www.jstor.org/stable/1267757.

[JNS01] Maarten Jansen, Guy P Nason, and Bernard W Silverman. "Scattered data smoothing by empirical Bayesian shrinkage of second-generation wavelet coefficients". In: *International Symposium on Optical Science and Technology*. International Society for Optics and Photonics. 2001, pp. 87–97.

[Jur+11] Raja Jurdak et al. "Wireless Sensor Network Anomalies: Diagnosis and Detection Strategies". In: *Intelligence-Based Systems Engineering*. Ed. by Andreas Tolk and LakhmiC. Jain. Vol. 10. Intelligent Systems Reference Library. Springer Berlin Heidelberg, 2011, pp. 309–325. ISBN: 978-3-642-17930-3. DOI: 10.1007/978-3-642-17931-0_12. URL: http://dx.doi.org/10.1007/978-3-642-17931-0_12.

[KA10] Muhammad Khurram Khan and Khaled Alghathbar. "Cryptanalysis and Security Improvements of 'Two-Factor User Authentication in Wireless Sensor Networks'". In: *Sensors* 10.3 (2010), pp. 2450–2459. ISSN: 1424-8220. DOI: 10.3390/s100302450. URL: http://www.mdpi.com/1424-8220/10/3/2450.

[Kal60] Rudolph Emil Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Transactions of the ASME-Journal of Basic Engineering* 82.Series D (1960), pp. 35–45.

[KF06] Timothy H. Keitt and Janet Fischer. "Detection of Scale-Specific Community Dynamics Using Wavelets". English. In: *Ecology* 87.11 (2006), pp. 2895–2904. ISSN: 00129658. URL: http://www.jstor.org/stable/20069308.

[KW05] Chris Karlof and David Wagner. "Secure routing in wireless sensor networks: attacks and countermeasures." In: *Ad Hoc Networks* 1.2-3 (July 14, 2005), pp. 293–315. URL: http://dblp.uni-trier.de/db/journals/adhoc/adhoc1.\#KarlofW03.

[Lar06] Daniel T. Larose. *Data mining methods and models.* Wiley, 2006, pp. I–XVI, 1–322. ISBN: 978-0-471-66656-1.

[Las+04] Pavel Laskov et al. "Intrusion Detection in Unlabeled Data with Quarter-sphere Support Vector Machines." In: *Praxis der Informationsverarbeitung und Kommunikation* 27.4 (2004), pp. 228–236. URL: http://dblp.uni-trier.de/db/journals/pik/pik27.html\#LaskovSKM04.

[LC13] Sung Yul Lim and Yoon-Hwa Choi. "Malicious Node Detection Using a Dual Threshold in Wireless Sensor Networks." In: *J. Sensor and Actuator Networks* 2.1 (2013), pp. 70–84.

[LCC07] Fang Liu, Xiuzhen Cheng, and Dechang Chen. "Insider Attacker Detection in Wireless Sensor Networks." In: *INFOCOM*. IEEE, June 27, 2007, pp. 1937–1945. URL: http://dblp.uni-trier.de/db/conf/infocom/infocom2007.html\#LiuCC07.

[LDH06] Xuanwen Luo, Ming Dong, and Yinlun Huang. "On Distributed Fault-Tolerant Detection in Wireless Sensor Networks." In: *IEEE Trans. Computers* 55.1 (May 10, 2006), pp. 58–70. URL: http://dblp.uni-trier.de/db/journals/tc/tc55.html\#LuoDH06.

[LF06] Siwei Lyu and Hany Farid. "Steganalysis using higher-order image statistics". In: *Information Forensics and Security, IEEE Transactions on* 1.1 (2006), pp. 111–119.

[LLR10] Ming Li, Wenjing Lou, and Kui Ren. "Data security and privacy in wireless body area networks". In: *Wireless Communications, IEEE* 17.1 (2010), pp. 51–58. ISSN: 1536-1284. DOI: 10.1109/MWC.2010.5416350.

[LM05a] D Lowd and C Meek. "Good Word Attacks on Statistical Spam Filters". In: *Conf. on Email and Anti-Spam*. 2005.

[LM05b] Daniel Lowd and Christopher Meek. "Adversarial Learning". In: *Conf. on Knowledge Discovery in Data Mining*. 2005, pp. 641–647.

[LN08] An Liu and Peng Ning. "TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks." In: *IPSN*. IEEE Computer Society,

May 7, 2008, pp. 245–256. URL: http://dblp.uni-trier.de/db/conf/ipsn/ipsn2008.html\#LiuN08.

[LNR11] Yao Liu, Peng Ning, and Michael K. Reiter. "False data injection attacks against state estimation in electric power grids." In: *ACM Trans. Inf. Syst. Secur.* 14.1 (2011), p. 13. URL: http://dblp.uni-trier.de/db/journals/tissec/tissec14.html#LiuNR11.

[Loa06] Clive Loader. *Local regression and likelihood.* Springer Science & Business Media, 2006.

[Lop+10] Javier Lopez et al. "Trust management systems for wireless sensor networks: Best practices." In: *Computer Communications* 33.9 (May 14, 2010), pp. 1086–1093. URL: http://dblp.uni-trier.de/db/journals/comcom/comcom33.\#LopezRAG10.

[LS07] Changwei Liu and Sid Stamm. "Fighting Unicode-Obfuscated Spam". In: *Procs. of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit.* 2007, pp. 45–59.

[Mar09] Stephen Marsland. *Machine Learning - An Algorithmic Perspective.* Chapman and Hall / CRC machine learning and pattern recognition series. CRC Press, 2009, pp. I–XVI, 1–390. ISBN: 978-1-4200-6718-7.

[MC03] Matthew V Mahoney and Philip K Chan. "An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection". In: *Workshop on Recent Advances in Intrusion Detection.* 2003, pp. 220–237.

[MCA08] M. Momani, S. Challa, and R. Alhmouz. "Can we trust trusted nodes in wireless sensor networks?" In: *Computer and Communication Engineering, 2008. ICCCE 2008. International Conference on.* 2008, pp. 1227–1232. DOI: 10.1109/ICCCE.2008.4580801.

[McC76] Garth P McCormick. "Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems". In: *Mathematical programming* 10.1 (1976), pp. 147–175.

[McH03] John McHugh. "Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory." In: *ACM Trans. Inf. Syst. Secur.* 3.4 (Nov. 28, 2003), pp. 262–294. URL: http://dblp.uni-trier.de/db/journals/tissec/tissec3.\#McHugh00.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.

[Nel+10] Blaine Nelson et al. "Near-Optimal Evasion of Convex-Inducing Classifiers". In: *AISTATS*. 2010, pp. 549–556.

[NHS08] Lewis Ntaimo, Xiaolin Hu, and Yi Sun. "DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment". In: *Simulation* 84.4 (2008), pp. 137–155.

[NIC07] NICTA. *Castalia Simulator*. 2007. URL: https://github.com/boulis/Castalia.

[Nis] *NIST*. http://www.nist.gov/.

[NLL06] E.C.-H. Ngai, Jiangchuan Liu, and M.R. Lyu. "On the Intruder Detection for Sinkhole Attack in Wireless Sensor Networks". In: *Communications, 2006. ICC '06. IEEE International Conference on*. Vol. 8. 2006, pp. 3383–3389. DOI: 10.1109/ICC.2006.255595.

[NP12] Kevin Ni and Gregory J. Pottie. "Sensor network data fault detection with maximum a posteriori selection and bayesian modeling." In: *TOSN* 8.3 (2012), p. 23. URL: http://dblp.uni-trier.de/db/journals/tosn/tosn8.html\#NiP12.

[OHC12] Seo Hyun Oh, Chan O. Hong, and Yoon-Hwa Choi. "A Malicious and Malfunctioning Node Detection Scheme for Wireless Sensor Networks." In: *Wireless Sensor Network* 4.3 (2012), pp. 84–90.

[OM05] Ilker Onat and Ali Miri. "An intrusion detection system for wireless sensor networks". In: *Wireless And Mobile Computing, Networking And Communications, 2005.(WiMob'2005), IEEE International Conference on*. Vol. 3. IEEE. 2005, pp. 253–259.

[Ott+05] Chris Otto et al. "System Architecture of a Wireless Body Area Sensor Network for Ubiquitous Health Monitoring". In: *J. Mob. Multimed.* 1.4 (2005), pp. 307–326. ISSN: 1550-4646. URL: http://dl.acm.org/citation.cfm?id=2010498.2010502.

[Owe56] Donald B Owen. "Tables for computing bivariate normal probabilities". In: *The Annals of Mathematical Statistics* 27.4 (1956), pp. 1075–1090.

[Pap+03] Spiros Papadimitriou et al. "LOCI: Fast Outlier Detection Using the Local Correlation Integral." In: *ICDE*. Ed. by Umeshwar Dayal, Krithi Ramamritham, and T. M. Vijayaraman. IEEE Computer Society, 2003, pp. 315–326. ISBN: 0-7803-7665-X. URL: http://dblp.uni-trier.de/db/conf/icde/icde2003.html\#PapadimitriouKGF03.

[Pap+16] Nicolas Papernot et al. "The Limitations of Deep Learning in Adversarial Settings". In: *IEEE European Symposium on Security and Privacy*. 2016, pp. 372–387.

[PH07] Lilia Paradis and Qi Han. "A Survey of Fault Management in Wireless Sensor Networks." In: *J. Network Syst. Manage.* 15.2 (Sept. 18, 2007), pp. 171–190. URL: http://dblp.uni-trier.de/db/journals/jnsm/jnsm15.html\#ParadisH07.

[Phy] *PhysioNet*. http://physionet.org.

[PS05] Taejoon Park and Kang G. Shin. "Soft Tamper-Proofing via Program Integrity Verification in Wireless Sensor Networks." In: *IEEE Trans. Mob. Comput.* 4.3 (July 26, 2005), pp. 297–309.

[PSP06] Bartosz Przydatek, Dawn Xiaodong Song, and Adrian Perrig. "SIA: secure information aggregation in sensor networks." In: *SenSys*. Ed. by Ian F. Akyildiz et al. ACM, Feb. 15, 2006, pp. 255–265. ISBN: 1-58113-707-9.

[Raj+06] S. Rajasegarar et al. "Distributed Anomaly Detection in Wireless Sensor Networks". In: *Communication systems, 2006. ICCS 2006. 10th IEEE Singapore International Conference on*. 2006, pp. 1–5. DOI: 10.1109/ICCS.2006.301508.

[Raj+09] Sutharshan Rajasegarar et al. "Quarter Sphere Based Distributed Anomaly Detection in Wireless Sensor Networks." In: *ICC*. IEEE, Apr. 15, 2009, pp. 3864–3869. URL: http://dblp.uni-trier.de/db/conf/icc/icc2007.html\#RajasegararLPB07.

[Raj+10] Sutharshan Rajasegarar et al. "Elliptical anomalies in wireless sensor networks." In: *TOSN* 6.1 (Mar. 15, 2010). URL: http://dblp.uni-trier.de/db/journals/tosn/tosn6.html\#RajasegararBLP09.

[Ray+08] M. Raya et al. "On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks". In: *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*. 2008, pp. –. DOI: 10.1109/INFOCOM.2008.180.

[Rez+13] Mohsen Rezvani et al. "A robust iterative filtering technique for wireless sensor networks in the presence of malicious attacks." In: *SenSys*. Ed. by Chiara Petrioli, Landon P. Cox, and Kamin Whitehouse. ACM, 2013, p. 30. ISBN: 978-1-4503-2027-6. URL: http://dblp.uni-trier.de/db/conf/sensys/sensys2013.html\#RezvaniIBJ13.

[RH92] BO Ruud and ES Husebye. "A new three-component detector and automatic single-station bulletin production". In: *Bulletin of the seismological society of America* 82.1 (1992), pp. 221–237.

[RLP08] Sutharshan Rajasegarar, Christopher Leckie, and Marimuthu Palaniswami. "Anomaly detection in wireless sensor networks." In: *IEEE Wireless Commun.* 15.4 (2008), pp. 34–40. URL: http://dblp.uni-trier.de/db/journals/wc/wc15.html\#RajasegararLP08.

[Roy+14] S. Roy et al. "Secure Data Aggregation in Wireless Sensor Networks: Filtering out the Attacker's Impact". In: 9.4 (2014), pp. 681–694. DOI: 10.1109/TIFS.2014.2307197. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6746043.

[RZL06] Rodrigo Roman, Jianying Zhou, and Javier Lopez. "Applying intrusion detection systems to wireless sensor networks". In: *IEEE Consumer Communications & Networking Conference (CCNC 2006)*. 2006.

[RZM13] Murad A. Rassam, Anazida Zainal, and Mohd Aizaini Maarof. "Advancements of Data Anomaly Detection Research in Wireless Sensor Networks: A Survey and Open Issues". In: *Sensors* 13.8 (2013), pp. 10087–10122. ISSN: 1424-8220. DOI: 10.3390/s130810087. URL: http://www.mdpi.com/1424-8220/13/8/10087.

[Sal+14] O. Salem et al. "Online Anomaly Detection in Wireless Body Area Networks for Reliable Healthcare Monitoring". In: *Biomedical and Health Informatics, IEEE Journal of* PP.99 (2014), pp. 1–1. ISSN: 2168-2194. DOI: 10.1109/JBHI.2014.2312214.

[SC92] D.A. Smith and G. Cox. "Major chemical species in buoyant turbulent diffusion flames". In: *Combustion and Flame* 91.3–4 (1992), pp. 226 –238. ISSN: 0010-2180. DOI: http://dx.doi.org/10.1016/0010-2180(92)90055-T. URL: http://www.sciencedirect.com/science/article/pii/001021809290055T.

[SEK03] AL Sullivan, PF Ellis, and IK Knight. "A review of radiant heat flux models used in bushfire applications". In: *International Journal of Wildland Fire* 12.1 (2003), pp. 101–110.

[Ses+04] Arvind Seshadri et al. "SWATT: SoftWare-based ATTestation for Embedded Devices." In: *IEEE Symposium on Security and Privacy*. IEEE Computer Society, June 21, 2004, pp. 272–282. ISBN: 0-7695-2136-3.

[Ses+05] Arvind Seshadri et al. "Pioneer: Verifying Code Integrity and Enforcing Untampered Code Execution on Legacy Systems". In: *ACM SIGOPS Operating Systems Review* 39.5 (2005), pp. 1–16. DOI: 10.1145/1095809.1095812.

[Ses+08] Arvind Seshadri et al. "SCUBA: Secure Code Update By Attestation in sensor networks." In: *Workshop on Wireless Security*. Ed. by Radha Poovendran and Ari Juels. ACM, Feb. 7, 2008, pp. 85–94. ISBN: 1-59593-557-6.

[SGG10] Abhishek B. Sharma, Leana Golubchik, and Ramesh Govindan. "Sensor faults: Detection methods and prevalence in real-world datasets." In: *TOSN* 6.3 (2010). URL: http://dblp.uni-trier.de/db/journals/tosn/tosn6.html\#SharmaGG10.

[She02] Timothy J. Shepard. "A Channel Access Scheme for Large Dense Packet Radio Networks." In: *SIGCOMM*. Dec. 9, 2002, pp. 219–230. URL: http://dblp.uni-trier.de/db/conf/sigcomm/sigcomm1996.html\#Shepard96.

[She68] Donald Shepard. "A two-dimensional interpolation function for irregularly-spaced data". In: *Proceedings of the 1968 23rd ACM national conference*. ACM. 1968, pp. 517–524.

[Sil+05] Ana Paula R da Silva et al. "Decentralized intrusion detection in wireless sensor networks". In: *Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*. ACM. 2005, pp. 16–23.

[Sim+10] Marcos A Simplício et al. "A survey on key management mechanisms for distributed wireless sensor networks". In: *Computer networks* 54.15 (2010), pp. 2591–2612.

[SL08] Alexander L Strehl and Michael L Littman. "Online linear regression and its application to model-based reinforcement learning". In: *Advances in Neural Information Processing Systems*. 2008, pp. 1417–1424.

[SL16] Rodrigo Vieira Steiner and Emil Lupu. "Attestation in Wireless Sensor Networks: A Survey". In: *ACM Comput. Surv.* 49.3 (Sept. 2016), 51:1–51:31. ISSN: 0360-0300. DOI: 10.1145/2988546.

[SLD12] Yan Sun, Hong Luo, and Sajal K. Das. "A Trust-Based Framework for Fault-Tolerant Data Aggregation in Wireless Multimedia Sensor Networks." In: *IEEE Trans. Dependable Sec. Comput.* 9.6 (2012), pp. 785–797. URL: http://dblp.uni-trier.de/db/journals/tdsc/tdsc9.html\#SunLD12.

[SLM13] Osman Salem, Yaning Liu, and Ahmed Mehaoua. "A lightweight anomaly detection framework for medical wireless sensor networks." In: *WCNC*. IEEE, 2013,

pp. 4358–4363. ISBN: 978-1-4673-5938-2. URL: http://dblp.uni-trier.de/db/conf/wcnc/wcnc2013.html\#SalemLM13.

[SLP11] Arvind Seshadri, Mark Luk, and Adrian Perrig. "SAKE: Software attestation for key establishment in sensor networks." In: *Ad Hoc Networks* 9.6 (2011), pp. 1059–1067. URL: http://dblp.uni-trier.de/db/journals/adhoc/adhoc9.html\#SeshadriLP11.

[SNQ12] Nauman Shahid, Ijaz Haider Naqvi, and Saad B. Qaisar. "Quarter-Sphere SVM: Attribute and Spatio-Temporal correlations based Outlier & Event Detection in wireless sensor networks." In: *WCNC*. IEEE, 2012, pp. 2048–2053. ISBN: 978-1-4673-0436-8. URL: http://dblp.uni-trier.de/db/conf/wcnc/wcnc2012.html\#ShahidNQ12.

[Sub+06] Sharmila Subramaniam et al. "Online Outlier Detection in Sensor Data Using Non-Parametric Models." In: *VLDB*. Ed. by Umeshwar Dayal et al. ACM, Sept. 27, 2006, pp. 187–198. ISBN: 1-59593-385-9. URL: http://dblp.uni-trier.de/db/conf/vldb/vldb2006.html\#SubramaniamPPKG06.

[Sun+07] Yan Lindsay Sun et al. "A Trust Evaluation Framework in Distributed Networks: Vulnerability Analysis and Defense Against Attacks." In: *INFOCOM*. IEEE, Dec. 6, 2007.

[Sun+13] Bo Sun et al. "Anomaly Detection Based Secure In-Network Aggregation for Wireless Sensor Networks." In: *IEEE Systems Journal* 7.1 (2013), pp. 13–25. URL: http://dblp.uni-trier.de/db/journals/sj/sj7.html\#SunSWX13.

[Sur+15] Nagender Kumar Suryadevara et al. "WSN-based smart sensors and actuator for power management in intelligent buildings". In: *IEEE/ASME Transactions On Mechatronics* 20.2 (2015), pp. 564–571.

[Sut+07] Yagiz Sutcu et al. "Tamper detection based on regularity of wavelet transform coefficients". In: *Image Processing, 2007. ICIP 2007. IEEE International Conference on*. Vol. 1. IEEE. 2007, pp. I–397.

[SWB06] David Sculley, Gabriel Wachman, and Carla E Brodley. "Spam Filtering Using Inexact String Matching in Explicit Feature Space with On-Line Linear Classifiers". In: *TREC*. 2006.

[SWR98] Suresh Singh, Mike Woo, and C. S. Raghavendra. "Power-Aware Routing in Mobile Ad Hoc Networks." In: *MOBICOM*. Ed. by William P. Osborne and Dhawal B. Moghe. ACM, 1998, pp. 181–190. ISBN: 1-58113-035-X. URL: http://dblp.uni-trier.de/db/conf/mobicom/mobicom1998.html\#SinghWR98.

[Sze+13] Christian Szegedy et al. "Intriguing Properties of Neural Networks". In: *arXiv preprint arXiv:1312.6199* (2013).

[TH06] Sapon Tanachaiwiwat and Ahmed Helmy. "Correlation Analysis for Alleviating Effects of Inserted Data in Wireless Sensor Networks." In: *MobiQuitous*. IEEE Computer Society, Jan. 25, 2006, pp. 97–108. ISBN: 0-7695-2375-7. URL: http://dblp.uni-trier.de/db/conf/mobiquitous/mobiquitous2005.html\#TanachaiwiwatH05.

[WA+06a] Geoffrey Werner-Allen et al. "Deploying a Wireless Sensor Network on an Active Volcano." In: *IEEE Internet Computing* 10.2 (July 24, 2006), pp. 18–25. URL: http://dblp.uni-trier.de/db/journals/internet/internet10.html#Werner-AllenLWMJRL06.

[WA+06b] Geoffrey Werner-Allen et al. "Fidelity and Yield in a Volcano Monitoring Sensor Network." In: *OSDI*. Ed. by Brian N. Bershad and Jeffrey C. Mogul. USENIX Association, 2006, pp. 381–396. ISBN: 1-931971-47-1. URL: http://dblp.uni-trier.de/db/conf/osdi/osdi2006.html#Werner-AllenLJLW06.

[WDB10] Xue Wang, Liang Ding, and Daowei Bi. "Reputation-Enabled Self-Modification for Target Sensing in Wireless Sensor Networks." In: *IEEE T. Instrumentation and Measurement* 59.1 (2010), pp. 171–179. URL: http://dblp.uni-trier.de/db/journals/tim/tim59.html\#WangDB10.

[Wit+98] Mitchell Withers et al. “A comparison of select trigger algorithms for automated global seismic phase and event detection”. In: *Bulletin of the Seismological Society of America* 88.1 (1998), pp. 95–106.

[Wu+07] Weili Wu et al. “Localized Outlying and Boundary Data Detection in Sensor Networks.” In: *IEEE Trans. Knowl. Data Eng.* 19.8 (2007), pp. 1145–1157. URL: http://dblp.uni-trier.de/db/journals/tkde/tkde19.html\#WuCDXLD07.

[WW04] Gregory L Wittel and Shyhtsun Felix Wu. “On Attacking Statistical Spam Filters”. In: *Conf. on Email and Anti-Spam*. 2004.

[Xie+11] Miao Xie et al. “Anomaly detection in wireless sensor networks: A survey.” In: *J. Network and Computer Applications* 34.4 (2011), pp. 1302–1325. URL: http://dblp.uni-trier.de/db/journals/jnca/jnca34.html\#XieHTP11.

[Yan+06] Yi Yang et al. “SDAP: : a secure hop-by-Hop data aggregation protocol for sensor networks.” In: *MobiHoc*. Ed. by Sergio Palazzo, Marco Conti, and Raghupathy Sivakumar. ACM, Dec. 22, 2006, pp. 356–367. ISBN: 1-59593-368-9. URL: http://dblp.uni-trier.de/db/conf/mobihoc/mobihoc2006.html\#YangWZ06.

[Zah+10] Theodore Zahariadis et al. “Trust management in wireless sensor networks”. In: *European Transactions on Telecommunications* 21.4 (2010), pp. 386–395. ISSN: 1541-8251. DOI: 10.1002/ett.1413.

[ZC04] Yi Zou and Krishnendu Chakrabarty. “Uncertainty-aware and coverage-oriented deployment for sensor networks”. In: *Journal of Parallel and Distributed Computing* 64.7 (2004), pp. 788–798.

[ZDL06] Wei Zhang, Sajal K. Das, and Yonghe Liu. “A Trust Based Framework for Secure Data Aggregation in Wireless Sensor Networks.” In: *SECON*. IEEE, 2006, pp. 60–69. ISBN: 1-4244-0626-9. URL: http://dblp.uni-trier.de/db/conf/secon/secon2006.html\#ZhangDL06.

[Zha+12] Y. Zhang et al. “Statistics-based outlier detection for wireless sensor networks.” In: *International Journal of Geographical Information Science* 26.8 (2012), pp. 1373–

1392. URL: http://dblp.uni-trier.de/db/journals/gis/gis26.html\#ZhangHMSVH12.

[Zhu+07] Sencun Zhu et al. "Interleaved hop-by-hop authentication against false data injection attacks in sensor networks". In: *ACM Transactions on Sensor Networks (TOSN)* 3.3 (2007), p. 14.

[ZL10] Dazhi Zhang and Donggang Liu. "DataGuard: Dynamic data attestation in wireless sensor networks." In: *DSN*. IEEE, 2010, pp. 261–270. ISBN: 978-1-4244-7501-8.

[ZWC12] Yuli Zhang, Huaiyu Wu, and Lei Cheng. "Some new deformation formulas about variance and covariance". In: *Modelling, Identification & Control (ICMIC), 2012 Proceedings of International Conference on*. IEEE. 2012, pp. 987–992.

[ZYN08] Qing Zhang, Ting Yu, and Peng Ning. "A Framework for Identifying Compromised Nodes in Wireless Sensor Networks." In: *ACM Trans. Inf. Syst. Secur.* 11.3 (Apr. 1, 2008). URL: http://dblp.uni-trier.de/db/journals/tissec/tissec11.html#ZhangYN08.

[Đur+12] Milica Pejanović Đurišić et al. "A survey of military applications of wireless sensor networks". In: *Embedded Computing (MECO), 2012 Mediterranean Conference on*. IEEE. 2012, pp. 196–199.