Determining Resilience Gains From Anomaly Detection for Event Integrity in Wireless Sensor Networks

VITTORIO P. ILLIANO, ANDREA PAUDICE, LUIS MUÑOZ-GONZÁLEZ, and EMIL C. LUPU, Imperial College London

Measurements collected in a wireless sensor network (WSN) can be maliciously compromised through several attacks, but anomaly detection algorithms may provide resilience by detecting inconsistencies in the data. Anomaly detection can identify severe threats to WSN applications, provided that there is a sufficient amount of genuine information. This article presents a novel method to calculate an assurance measure for the network by estimating the maximum number of malicious measurements that can be tolerated. In previous work, the resilience of anomaly detection to malicious measurements has been tested only against arbitrary attacks, which are not necessarily sophisticated. The novel method presented here is based on an optimization algorithm, which maximizes the attack's chance of staying undetected while causing damage to the application, thus seeking the worst-case scenario for the anomaly detection algorithm. The algorithm is tested on a wildfire monitoring WSN to estimate the benefits of anomaly detection on the system's resilience. The algorithm also returns the measurements that the attacker needs to synthesize, which are studied to highlight the weak spots of anomaly detection. Finally, this article presents a novel methodology that takes in input the degree of resilience required and automatically designs the deployment that satisfies such a requirement.

CCS Concepts: • **Security and privacy → Intrusion/anomaly detection and malware mitigation**;

Additional Key Words and Phrases: Integrity violation, tampering, malicious data, resilience, wireless sensor networks, event detection

ACM Reference format:
Vittorio P. Illiano, Andrea Paudice, Luis Muñoz-González, and Emil C. Lupu. 2018. Determining Resilience Gains From Anomaly Detection for Event Integrity in Wireless Sensor Networks. *ACM Trans. Sen. Netw.* 14, 1, Article 5 (February 2018), 35 pages.
https://doi.org/10.1145/3176621

1 INTRODUCTION

Wireless sensor networks (WSNs) offer economically sustainable solutions for applications such as health monitoring, critical infrastructure monitoring, and military surveillance. Due to the high economic and safety load, several security mechanisms have been proposed so far to guarantee the integrity of the measurements in WSNs [23, 33, 53].

This work was funded by the UK Engineering and Physical Sciences Research Council (EPSRC) under grant EP/N023242/1 for the PETRAS Internet of Things Research Hub.

Authors' address: V. P. Illiano, A. Paudice, L. Muñoz-González, and E. C. Lupu, Department of Computing, Imperial College London, 180 Queen's Gate, London, SW7 2AZ, United Kingdom; emails: {v.illiano13, a.paudice15, l.munoz, e.c.lupu}@imperial.ac.uk.

ACM 1550-4859/2018/02-ART5 $15.00
https://doi.org/10.1145/3176621

Integrity violations that compromise the measurements inevitably impair the delivery of all critical services that are spreading through the Internet of Things (IoT), such as smart cities, smart homes, and wearable healthcare systems. The number of IoT deployments is already wide, with at least 6 billion smart devices estimated in 2016, and is forecast to overcome 20 billion by 2020 [1].

The integrity of the WSN measurements is essential for such applications to correctly analyze the sensed phenomenon, react to it, and deliver a critical service. However, WSNs are vulnerable to physical tampering with sensor nodes, environment manipulations, and interception of network transmissions that give an attacker access to the measurements. When this happens, an attacker may introduce malicious measurements, which we refer to as malicious data injections.

Different kinds of techniques have been proposed to defend against malicious data injections, both proactive and reactive. In this article, we focus on a reactive technique, *measurements inspection*, which has received much interest in literature [23, 26, 34, 46, 61], as in contrast to proactive techniques, it does not require the use of more sophisticated devices, such as tamper-proof hardware [28, 29, 58].

Measurements inspection identifies measurements alterations through inspection of the measurements themselves. Generally, it makes use of anomaly detection (i.e., detecting unexpected behaviors by identifying inconsistencies in the data), which may indicate the presence of integrity violations to the measurements. The chance to identify anomalies in the data is highly related to the quality of the data itself, which is connected to the quality of the sensing devices, the degree of noise, the fault frequency, and the deployment's density.

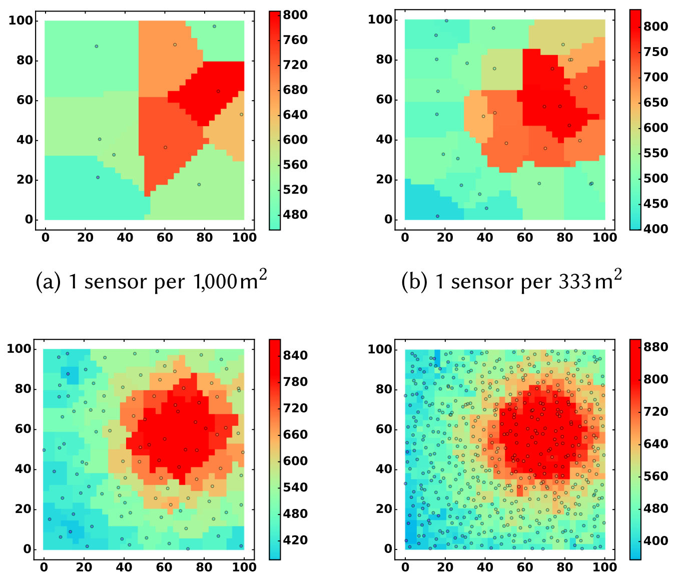
Even though it is intuitive that denser deployments increase the data quality, and hence the robustness of measurements inspection against malicious data injections, there is currently no means to quantify and adjust such a robustness, which is the main goal of this work.

For example, we show in Figure 1 the same physical phenomenon—that is, the temperatures in an area where a fire is present, observed by four different deployments. In particular, the deployments differ by density of sensor nodes. It is evident that with higher sensor density, there is a dramatic improvement in the quality of the information. For instance, Figure 1(d) shows that with one sensor per 20m^2, we can precisely identify the spread and the boundaries of the fire, as well as the presumable breakout point.

When the measurements' quality is irremediably impaired by noise, faults, and malicious data, relying on denser deployment is more and more necessary, as the propagation dynamics of the physical phenomena are complex. Indeed, the mutual information shared by different sensors can be used to overcome the data degradation issues. This problem is even more complex when considering malicious interference rather than genuine data corruption, because malicious measurements are not isolated nor random, in the general case. Indeed, although traditional anomaly detection usually assumes that malicious measurements manipulations are independent (local), a sophisticated attacker will generally manipulate the measurements in the malicious sensors to collude toward the same objective. In this way, malicious measurements can be, by and large, less distinguishable from genuine measurements.

Currently, there is no methodical means to link the deployment structure to malicious data injections. In this article, we provide a WSN owner with the set of techniques to do the following:

- Measure the degree of resilience of a WSN to malicious data injections in terms of maximum number of compromised sensors/measurements that can be tolerated
- Identify the most critical resources for the attacker and the most threatening strategy to inject malicious measurements
- Provide an assured degree of resilience at design time by operating on the WSN deployment.



(a) 1 sensor per 1,000 m^2 (b) 1 sensor per 333 m^2

(c) 1 sensor per 100 m^2 (d) 1 sensor per 20 m^2

Fig. 1. Example of data quality with different deployments with increasing density of sensors. The WSN area is a two-dimensional space. The circles identify the sensors, and colors map the temperatures in Kelvin. With higher density, it is more evident that a fire is present, and its characteristics (affected area, breakout point, etc.) can be better estimated.

Since measurements inspection is agnostic to the type of attack, our work is not restricted to a specific kind of attack vector. We consider violations to the measurements integrity that are conveyed through node tampering (both via hardware and software), environment manipulation, and network packets modification. However, regarding network-layer data integrity, we consider attacks that affect only the endpoints. The problem of network-layer integrity in forwarding nodes is better addressed with ad hoc techniques, such as cryptographic hash functions [9, 22, 63] and intrusion detection [11, 43, 48], which are beyond the scope of this work.

In this article, we focus on *event detection* WSNs. Event detection refers to the task of identifying conditions of interest in the monitored phenomenon that can be observed with noticeable changes in the measurements. For instance, an earthquake is an event for seismic sensors that causes high vibrations, or a flood is an event for water-level monitoring sensors that would register the surface rise. The main mission of such systems is to detect an *event condition* when there are events and not detect an event condition when the monitored phenomenon is at rest. Hence, the damage that an attacker can bring into such systems is eliciting event detection to detect false conditions of interest or masking real conditions of interest.

In this context, an attack is sophisticated if it subverts event detection and is not detected by the anomaly detection algorithm (i.e., it achieves evasion). Yet there is no standard way of testing anomaly detection against the most sophisticated attack that an attacker can cause with a certain number of compromised nodes. A possible approach is to use a best effort strategy that considers the subversion of event detection as a priority and does its best to avoid detection. The approach just mentioned belongs to the category of *mimicry attacks* [7] (i.e., the characteristic of genuine samples is mimicked). Even though mimicry attacks are useful to validate anomaly detection algorithms, they cannot consider exhaustively the algorithm's blind spots that an attacker may exploit to evade it. In other words, they are reliable when successful but do not give guarantees when unsuccessful.

Rather than mimicry, the approach pursued in this article is that of the *worst-case-attack* [7]. This consists of computing the measurements that cause the worst damage for the system (i.e., guarantee evasion and subvert event detection). This scenario corresponds to the case where the adversary has full knowledge of the anomaly detection algorithm. This study gives an assured lower bound to the system's degree of resiliency and constructs the strategy that enables the attacker to evade the anomaly detection algorithm. Based on the analysis of the worst-case attack, in this work we bring three novel contributions to the problem of determining resilience gains from anomaly detection in WSNs, which are detailed in the following:

(1) Our first contribution is a novel method to estimate the malicious measurements that cause the worst-case attack while minimizing the number of sensors the attacker needs to control to both evade anomaly detection and deceive the event detection algorithm. This is done with an optimization algorithm that applies to a generic WSN, characterized by sensed physical phenomenon, number of sensors, and sensors deployment locations. This algorithm is a tool for a systematic and automatic evaluation of the detectability performance for an already deployed WSN and gives information about the maximum number of malicious sensors that a WSN can tolerate. The algorithm is designed alongside a set of mechanisms that make the problem tractable. We evaluate and discuss the approximations that can be reasonably introduced. This is to ensure that the analysis runs in a reasonable time. Moreover, since the method is independent from the anomaly detection algorithm, it can also be exploited to evaluate different anomaly detection approaches and compare them to each other. This algorithm can be used by organizations managing WSNs to calculate the degree of exposure and sensor maintenance, given their risk tolerance.

(2) The second contribution is a study of the injections strategies that are more difficult to defend against, which is conducted by analyzing the attacks returned by the optimization algorithm. Through the help of graphical examples, we show what malicious data would look like in worst-case scenarios and where the compromised sensors would be located with respect to each other, and the WSN deployment area as well. This study, in turn, reveals the weak spots that make up most of the WSN risk and if this relates to the innate WSN data characteristics or to shortcomings of the anomaly detection algorithm. With such information, a WSN operator may decide to give better protection to sensors that are more attractive for the attacker, or to apply a different anomaly detection algorithm.

(3) The third contribution is to introduce the first security-driven design methodology for WSN deployments. Given a maximum number of sensors that can be compromised in a WSN, we calculate at design time how many sensors need to be deployed to guarantee that evading the anomaly detection and causing damage to the WSN are mutually exclusive goals for the attacker. This study provides a way to obtain guarantees about the degree of the gain in resilience introduced by anomaly detection. Indeed, the performance of anomaly detection depends on the density of genuine sensors compared to malicious sensors. The guaranteed degree of resilience to integrity violations can be applied for two complementary objectives. The first is the design of critical WSNs with strict security requirements: given a required degree of resilience for a WSN application, we provide an algorithm that automatically designs the WSN deployment to achieve that resilience. The second is a feasibility analysis: given a maximum budget in sensor devices, the designed

WSN given in output by such an algorithm enables one to decide whether it is worth applying the anomaly detection algorithm.

The rest of this article is organized as follows. In Section 2, we review the related work. In Section 3, we introduce a state-of-the-art measurements inspection algorithm based on wavelet transform [24] that is used as a reference. In Section 4, we present the optimization problem that constitutes the attack model. In Section 5, we propose an algorithm that solves the attacker's problem and deals with its high computational burden. In Section 6, the algorithm is applied to the measurements inspection algorithm introduced in Section 3 and to another state-of-the-art algorithm based on principal component analysis (PCA) [10], in the context of a wildfire-monitoring WSN. This analysis enables us to discuss and analyze the maximum number of malicious sensors that can be tolerated and the characteristics of the calculated attack. In Section 7, we analyze the algorithm's output in detail to gather information about the threats that the WSN is subject to in worst-case scenarios. In Section 8, we present an algorithm that automatically designs a WSN deployment given the number of malicious sensors to tolerate, apply it to wildfire-monitoring WSNs, and analyze the results. Finally, Section 9 presents our conclusions and illustrates future work directions.

2 RELATED WORK

Many techniques have been proposed for counteracting malicious data injections, even though few deal explicitly with sophisticated attackers [10, 23, 47, 59]. In general, this family of techniques is difficult to validate, as malicious data injections are not common yet, or at least they are not easy to retrieve. Even if they were available, malicious data can be injected into several different ways, with different degrees of sophistication and different results on the detection algorithms.

For these reasons, most of work available in the literature is tested against simulated attacks in either simulated WSNs [2–4, 30, 32, 42, 47, 56] or real WSN datasets [10, 59]. The simulated attacks aim to reproduce some of the characteristics that are expected in real ones, such as introducing bias or high variance in the measurements. In this article, we seek to identify the worst-case attacks rather than simulating possible attackers, as there are two main problems with attack simulation:

- Simulations define the attack steps a priori. However, the attacker 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—that is, having defined the steps to build an attack, the specific attack is chosen at random. Instead, the attacker generally seeks to optimize the attack among all feasible ones.

In summary, the simulations produce attacks that do not offer assurance or guarantees, as they do not model the attacker as an active entity that seeks to stay undetected, and do not optimize the attack from the attacker's perspective. Problems of this nature have been considered by *adversarial machine learning*, which is concerned with the robustness and security properties of machine learning algorithms and anomaly detection techniques. As described in Huang et al. [21], 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 behavior of the system by injecting malicious points at learning time. In exploratory attacks, the attacker cannot alter the behavior 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 work, we consider a particular case of exploratory attack, where the goal of the attacker is to compromise the smallest subset of sensors in a WSN that allows the crafting of malicious measurements capable of triggering or masking events without being detected by the anomaly detection system.

In the context of security, exploratory attacks have been extensively studied for intrusion detection systems (IDS) [15, 17, 37] and spam filtering applications [31, 36, 50, 60]. Despite the differences between the two applications, the approaches to evade the anomaly detector rely on the same principle: exploiting flaws in the selection of the features used by the system by crafting malicious samples that add or delete features or by encrypting or obfuscating malicious functionality so that the system does not recognize them as malicious features.

A different approach is considered in Lowd and Meek [35], where the authors propose an algorithm to reverse engineer 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 Nelson et al. [40], the authors propose an extension of this framework to the family of convex-inducing classifiers—that is, classifiers that partition their sample space into two sets, one of which is convex. The model proposed in Nelson et al. [40] also models the optimization 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 Lowd and Meek [35]. However, both models are limited to the family of classifiers to which they can be applied, whereas in many real applications nonlinear detection systems are used. Evasion attacks have also targeted deep neural networks in computer vision problems, demonstrating the existence of *adversarial examples*—that is, those images that can be misclassified by deep learning algorithms while being only imperceptibly distorted [18, 45, 57]. A more general framework for modeling evasion attacks is described in Biggio et al. [8], where a gradient-based strategy is applied to evade a broader range of differentiable machine learning classifiers.

Attacks against machine learning systems have also been shown to have some transferability properties—that is, the attack points generated targeting one specific algorithm are often effective against other learning systems. Thus, in Papernot et al. [44], the authors show the effectiveness of evasion attacks crafted with surrogate datasets and models when tested on different algorithms. More recently, it was shown in Muñoz-González et al. [39] that poisoning attacks are also transferable across learning algorithms.

However, transferability properties can be applied only when the attacker's problem is modeled as in Biggio et al. [8], Lowd and Meek [35], Nelson et al. [40], and Papernot et al. [44], which is a search for the lowest-cost sample that evades the detection. Instead, in this work, we consider that the attacker's cost to minimize is the number of sensors to be compromised 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 Biggio et al. [8], Lowd and Meek [35], Nelson et al. [40], and Papernot et al. [44]. A second difference is that we do not attempt to reverse engineer the anomaly detector, but we look for the blind spots that allow the crafting of 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 nonlinearities increase the time required to solve the problem.

The algorithm is first run on a real measurements inspection technique: the wavelet-based detection algorithm described in Illiano et al. [24], 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 to be 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 that is based on PCA [10]. This shows that the novel approach described in this article can also be used to compare the robustness of different anomaly detection algorithms, as we do in Section 6.3.

3 MEASUREMENTS INSPECTION ALGORITHM: WAVELET-BASED DETECTION

In this section, we summarize the wavelet-based detection algorithm described in Illiano et al. [24]. We will use this algorithm because it has been designed for event detection WSNs and to our knowledge provides the highest performance (in terms of true positives vs. false positives) for detecting both elicited and masked events, even when many sensors collude. We analyze in detail how wavelet-based detection can be evaded through the optimization algorithm. However, the optimization is not tailored specifically to this approach; indeed, we analyze also the resilience of another state-of-the-art anomaly detection algorithm in Section 6.3. Nevertheless, the optimization algorithm runs in feasible time provided a simple formulation of the anomaly detection algorithms. For algorithms as complex as the wavelet-based detection, a linearization is required, which can be carried out only by knowing the algorithm's mechanisms that are reported in the following.

Wavelet-based anomaly detection is designed to inspect the variation of the measurements across different sensors and infer if that is compatible with the diffusion pattern of genuine events/rest conditions or due to the introduction of malicious measurements. This is achieved through a multiscale analysis, which compares the variation between neighboring sensors (low scale) with the variation observed with a broadest view (high scale).

Thanks to the multiscale analysis, the intermeasurements variation can be evaluated in context as well as the duration of the changed trend, or the spread since we are reasoning in spatial dimensions. Using a cross-scale comparison, this algorithm can quantify short-spread steep changes and link them back to the changes introduced by events, to infer if the former can be explained by the latter.

Wavelet-based detection is based on second-generation continuous wavelet transform [27]. We consider a spatial measurements signal x, defined at the sensors locations $\mathbf{u} \in \Omega$, and denote the respective transform as $(T^{wav}x)$. This is a function of two variables s and τ, corresponding to scale and translation, defined as follows:

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

where

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

The scales s are monodimensional measures of space, which determine the granularity of the analysis: higher scales give analyses with coarser granularity. The translations τ represent the location in the space being analyzed and thus have the same dimensionality as the WSN space. This allows one to circumscribe the outcome of the analyses and is particularly useful when the measurements distribution changes in space, a condition that is likely to occur with the manifestation of events.

The principle behind (1) is to isolate scale-specific trends of the measurements signal x through multiplication with the signal $\Psi_{s,\tau}$, which acts like a filter with a size depending on s. In particular, higher scales filter out high frequencies, and lower scales filter out low frequencies. For each τ, the function $\Psi_{s,\tau}$ works also as a spatial filter. Therefore, each wavelet coefficient $(T^{wav}x)(s,\tau)$ is an indicator of the signal's spectral content for a fixed frequency and spatial area.

The actual filtering output depends on the wavelet function $\Psi_{s,\tau}$, which in Illiano et al. [24] is the second-generation variant of the "difference-of-Gaussians" wavelet. The wavelet function operates in the domain of the analyzed signal—that is, the spatial domain in the case of WSN measurements. At a generic location $\mathbf{u}$, the function's value is the following:

$$\Psi_{s,\tau}(\mathbf{u}) = \frac{e^{-(\frac{\mathbf{u}-\tau}{s})^2/2}}{\sum_{\mathbf{v}\in\Omega} e^{-(\frac{\mathbf{v}-\tau}{s})^2/2}} - \frac{e^{-(\frac{\mathbf{u}-\tau}{\beta s})^2/2}}{\sum_{\mathbf{v}\in\Omega} e^{-(\frac{\mathbf{v}-\tau}{\beta s})^2/2}}. \tag{3}$$

For $\beta > 1$, the second Gaussian is wider and more flat than the first, or in other words, the first emphasizes more τ's neighborhood. Their subtraction emphasizes changes in the measurements that happen around τ and stabilize thereafter. Since a Gaussian transforms to a Gaussian with the Fourier transform, the same happens also in frequency—that is, $\Psi_{s,\tau}$ acts like a band-pass filter. The parameter β determines the wavelet's band-pass characteristics. In accordance with the authors of this wavelet transform [27], we use the value 1.87, which gives the transform the desired band-pass characteristics and also simplifies the interpretation. Indeed, the main frequency of the wavelet function is just $1/s$ (i.e., one cycle every s spatial units).

The term $h_\tau(s)$ in (2) coincides with the standard deviation of the wavelet coefficient $(T^{wav}x)(s,\tau)$ when x is *white Gaussian noise* and thus is referred to as the standardization factor. However, for energy-related considerations, the nonstandardized transform is considered—that is,

$$\left(T^{wav}_{ns}x\right)(s,\tau) = \sum_{\mathbf{u}\in\Omega}[\Psi_{s,\tau}(\mathbf{u})S(\mathbf{u})] = (T^{wav}x)(s,\tau)h_\tau(s). \tag{4}$$

The wavelet transform is used to learn the relationship between coefficients at low scales (which identify the measurements' behavior in the sensors' immediate neighborhood) and coefficients at higher scales (which characterize the measurements' evolution at intermediate granularities, up to the granularity of the whole WSN deployment). To make such a comparison, the low-scale coefficients are seen as a function of the higher-scale coefficients. The dependent variable of such a relationship, denoted with $D(\tau)$, is the raw low-scale coefficient indicated next:

$$D(\tau) = (T^{wav}x)(s_0,\tau), \tag{5}$$

where s_0 is the lowest scale, coinciding with the average distance between a sensor and its closest neighbor. Low-scale information is highly affected by malicious measurements especially at the boundary between genuine and malicious measurements.

The independent variable instead, denoted with $I(\tau)$, is a function of the higher-scale coefficients as defined next:

$$I(\tau) = \sum_t \sum_{s\in hs,\ldots,Hs} \left(T^{wav}_{ns}x\right)^2(s,\tau)m(s,t-\tau), \tag{6}$$

where $m(s,t-\tau)$ is a disc mask centered in $t-\tau$ and radius s, and $[hs,Hs]$ is the range of higher scales [24]. The purpose of Bettencourt et al. (6) is to quantify the energy of the changes that are happening at higher scales. For instance, a large energy change at high scales can depict the presence of an event. Note that the independent variable is a sum of squared coefficients. In the interest of computational efficiency, this function will be linearized as discussed in Section 5.2.

To change the values of some measurements while evading detection, the attacker needs to manipulate a larger set of measurements so that modifications to the low-scale and high-scale coefficients bring a new consistent cross-scale relationship.

The cross-scale relationship between low- and high-scale coefficients is extracted by learning a function q, which returns the maximum magnitude of $D(\tau)$ that can be accepted for the value of

$I(\tau)$ at a given location τ. Hence, the following condition should hold:

$$|D(\tau)| \leq q(I(\tau)). \tag{7}$$

The q function is learned by dividing the space of the I values into intervals, filtering out obvious outliers, and returning the maximum value of D in such an interval. The intervals are individuated with a simple and effective criterion, which sorts the data by the values of I, and iteratively produces a new interval as soon as a new maximum is found. The resulting q function is a step function, which can be easily linearized through interpolation, as discussed in Section 5.2.

The low-scale coefficient is more likely anomalous when the ratio between the value returned by q and the low-scale coefficient at the same location is high. The maximum ratio across all WSN locations is referred to as the anomaly score, which is denoted as $a(S)$ and defined next:

$$a(S) = \max_{\tau} q(I(\tau))/|D(\tau)|). \tag{8}$$

The presence of malicious data is inferred when the anomaly score is higher than a detection threshold T_d. The correct value of T_d depends on how representative the data used to learn the cross-scale relationship is, with respect to the generic data distribution. For well-representative historical data, the correct value for T_d is one, which means that the cross-scale relationship is also valid for new data. Otherwise, T_d will be higher than one, and the most suitable value can be found experimentally on a separate test set by selecting the value that achieves the most appropriate trade-off for the application between false positives and false negatives.

4 ATTACK OPTIMIZATION MODEL

To study the robustness and weaknesses of anomaly detection for an event detection WSN, we need to consider the worst-case scenario. The latter refers to 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 the following:

(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.

Our goal is to compute the worst-case scenario. Without loss of generality, we introduce a model for the problem of malicious data injections in WSNs. The notation is summarized in Table 1.

We consider a set of N sensors and their respective measurements time series. The sensors are placed in a set of locations Ω and periodically observe the measurements $x_i(\mathbf{u}_j) \ : j \in 1, \ldots, N \ \mathbf{u}_j \in \Omega$, $i \in 1, \ldots, T$, where $\mathbf{u}_{j:j \in 1,\ldots N}$ is an arbitrary ordering of all spatial locations $\mathbf{u} \in \Omega$.

A number of C sensors are malicious ($C \leq N$) and are allowed to modify the measurements $x_i(\mathbf{u}_j)$ into $\check{x}_i(\mathbf{u}_j) = x_i(\mathbf{u}_j) + m_i(\mathbf{u}_j)$, where $m_i(\mathbf{u}_j)$ is a value in control of the attacker and constitutes the *manipulations signal*.

The first objective of the attacker is to select the sensors to target for compromise. For this task, we introduce the design variable $y(\mathbf{u}_j)$, which equals one if the sensor located at $\mathbf{u}_j$ is malicious and zero otherwise.

Table 1. Notation Summary

Term	Description
N	Number of sensors
C	Number of malicious sensor nodes
Ω	Set of sensors locations
$x_i(\mathbf{u}_j)\ :\mathbf{u}_j \in \Omega,\ i \in 1, \ldots, T$	Measurement observed by sensor located at $\mathbf{u}_j$ at *i-th* time instant
$m_i(\mathbf{u}_j)\ :\mathbf{u}_j \in \Omega,\ i \in 1, \ldots, T$	Difference between malicious and genuine measurement for sensor located at $\mathbf{u}_j$ at *i-th* time instant
$\check{\mathbf{x}}_i(\mathbf{u}_j)$	Measurements after potential attack (i.e. $x_i(\mathbf{u}_j) + m_i(\mathbf{u}_j)$)
W	Size of the event detection window
$\mathbf{X}_{w_t}$	$W \times N$ matrix of $\mathbf{x}$
$\mathbf{M}_{w_t}$	$W \times N$ matrix of $\mathbf{m}$
$\check{\mathbf{X}}_{w_t}$	$W \times N$ matrix of $\check{\mathbf{x}}$
$y(\mathbf{u}_j)$	Decision variable; equal to one if the *j-th* sensor is malicious and zero otherwise
$\mathbf{y}$	N-long vector of all y variables
$a(\mathbb{R}^N)$	Anomaly score (i.e., degree of anomalousness)
$e(\mathbf{X}_{w_t})$	Event score (i.e., confidence in the presence of events)
T_d	Anomaly detection threshold
T_e	Event threshold
N_{min}	Minimum number of deployed sensors for a desired degree of resiliency

The second objective is to find the manipulations signals at time instant i, $m_i(\mathbf{u}_j)$ such that the resulting signal $x_i(\mathbf{u}) + y(\mathbf{u})m_i(\mathbf{u})$ (which is the observed signal at time instant i, i.e., including malicious data injections) evades detection and changes the output of the *event detection function*.

For convenience, we introduce the row vector $\mathbf{y}$, where $\mathbf{y}_j = y(\mathbf{u}_j)$. Likewise, we introduce the $T \times N$ matrices $\mathbf{X}$ and $\mathbf{M}$, where $\mathbf{X}_{ij} = x_i(\mathbf{u}_j)$ and $\mathbf{M}_{ij} = m_i(\mathbf{u}_j)$.

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, we then introduce the following:

$$\begin{aligned} \mathbf{X}_{w_t} &= \{x_i(u_j)\} \quad i \in t, \ldots, t+W-1 \quad j \in 1, \ldots N, \\ \mathbf{M}_{w_t} &= \{m_i(u_j)\} \quad i \in t, \ldots, t+W-1 \quad j \in 1, \ldots N. \end{aligned} \tag{9}$$

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 [23]. Even though there is a body of work that exploits the temporal domain (i.e., analyzes the measurements time series on single sensors [6, 54, 56]), 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 that gradually modifies the measurements time series (e.g., to depict the presence of a false event with the temporal dynamics of a real event). However, 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 while evading anomaly detection can be modeled as the optimization problem defined in Chatzigiannakis and Papavassiliou (10), which minimizes the number of malicious sensors required by the attacker to subvert the event detection output while staying undetected.

$$\min_{y} \sum_{j=1}^{N} (\mathbf{y}_j) \tag{10}$$

$$s.t.$$

$$\mathbf{y}_j \in \{0,1\} \, \forall i \in 1,\ldots,N \tag{10a}$$

$$e(\mathbf{X}_{w_t}) \neq e(\mathbf{X}_{w_t} + \mathbf{1}^T\mathbf{y} \odot \mathbf{M}_{w_t}) \, \forall t \in 1,\ldots,T-W+1 \tag{10b}$$

$$a(\mathbf{X}_i + \mathbf{y} \odot \mathbf{M}_i) \leq T_d \, \forall i \in 1,\ldots,T \tag{10c}$$

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, and hence it is a $W \times N$ matrix. For convenience, we give an alias to this matrix:

$$\check{\mathbf{X}}_{w_t} = \mathbf{X}_{w_t} + \mathbf{1}^T\mathbf{y} \odot \mathbf{M}_{w_t}. \tag{11}$$

Constraint (10c) 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 wavelet-based detection, it coincides with the anomaly score, defined in Biggio et al. (8). The anomaly score is constrained to be less than the constant T_d for the resulting signal in input. Although in this work we will focus on analyzing the specific detection algorithm described in Section 3, this choice suits most detection algorithms well, as eventually there is a quantity that is compared to a threshold to decide if the data is anomalous or not.

Constraint (10b) 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 and zero otherwise.[1] The presence of binary inequalities would force us to choose more complex and burdensome optimization algorithms, so we first evaluate the value of $e(\mathbb{R}^W \times \mathbb{R}^N)$ 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 summarized by the following expression:

$$\begin{cases} e(\check{\mathbf{X}}_{w_t}) = 1 & \text{if} \quad e(\mathbf{X}_{w_t}) = 0 \\ e(\check{\mathbf{X}}_{w_t}) = 0 & \text{if} \quad e(\mathbf{X}_{w_t}) = 1. \end{cases} \tag{12}$$

This formulation is still a problem for common optimization algorithms because the event detection function has a binary output. However, we can consider $e(\mathbf{X}_{w_t})$ as the *event confidence*—that is, 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

$$\begin{cases} e(\check{\mathbf{X}}_{w_t}) \geq T_e & \text{if} \quad e(\mathbf{X}_{w_t}) < T_e \\ e(\check{\mathbf{X}}_{w_t}) < T_e & \text{if} \quad e(\mathbf{X}_{w_t}) \geq T_e. \end{cases} \tag{13}$$

[1]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 characterization of the events (e.g., incorrect estimation of spread and intensity). This issue can be dealt with by introducing further constraints on the terms $\check{\mathbf{X}}_{w_t}$, which require the event to have specific characteristics. Thus, this aspect requires just a refinement of problem (10), and we leave this for future work.

Constraints 10c and 10b 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 optimization is to find, for a fixed vector of compromised sensors $\mathbf{y}$, any combination of $\mathbf{M}_{w_t}$ that satisfies the constraints.

The model introduced previously 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 optimization of the event detection performance. Similarly, we introduced the threshold T_d for the detection of malicious interference, which we assume to be already optimized to guarantee a maximum number of false positives. The frequency of retraining/learning of the thresholds depends both on the quality of the data and the algorithm's generalization capabilities. In Illiano et al. [24], it is shown that the wavelet-based algorithm achieves good results even with as low as nine training samples, provided that they depict varied event scenarios.

5 SOLVING THE OPTIMIZATION PROBLEM

Regardless of the specific characteristics of the WSN under analysis, solving exactly Problem (10) is intractable unless the WSN deployment size is in the order of 10 sensors. This is a consequence of many factors that we discuss next.

5.1 Elements of Complexity

5.1.1 Binary Variables. In Problem (10), 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 optimization 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. Thus, even if Problem (10) were efficiently solved, finding a solution would be practical only for small deployment sizes.

5.1.2 Event Score Function. The event score function introduces two main problems. (1) First, it may be arbitrarily complex. This results in irregularities (e.g., nonlinearities, discontinuities, nonsmoothness) that require the use of expensive optimization algorithms, which make several evaluations of the event score function on its domain. (2) Second, t input size of the event score function is WN, whereas the other terms in Chatzigiannakis and Papavassiliou (10) have an input size of N. Consequently, the event score function needs to be simple enough to enable the application of fast optimization 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, and so forth. 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 optimization algorithm.

5.1.3 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, and so forth. Unlike event detection, simplifications of the anomaly detection algorithms require a more careful analysis, as they are likely to bias the optimization problem. Indeed, although 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 optimization algorithm, whose goal is to find the attack that requires less resources.

ALGORITHM 1: Binary Constraint Relaxation

```
Input: $\mathbf{y}', \mathbf{M}'_i, C$
Output: $\mathbf{y}$
1: sortInd = argsort($\mathbf{y}' \odot \mathbf{M}'_i$)
2: $\mathbf{y}_j = 0 \quad \forall j \in$ sortInd(1), ..., sortInd($N - C$)
3: $\mathbf{y}_j = 1 \quad \forall j \in$ sortInd($N - C + 1$), ..., sortInd($N$)
```

Nevertheless, simplifications are necessary, as a complex anomaly detection not only complicates the optimization but also causes an 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 Bettencourt et al. (6). Although evaluating each of them on a known input has an asymptotic complexity $O(N)$, evaluating them on unknowns requires the calculation of $\binom{N}{2}$ products, which is $O(N^2)$, with an increase in complexity of a factor N. Thus, even if anomaly detection is generally not particularly time consuming, optimizing its input may have execution times that are higher by a few orders of magnitude.

5.2 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. Afterward, the solution obtained with the simplified models is tested and adapted for the original problem.

5.2.1 Relaxing Binary Variables. The presence of binary variables makes the problem intractable even for low values of N, as 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 [5], which start with the relaxation of the binary variables into continuous variables between zero and one. Thus, 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, as 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 mainly exploited by the attacker are those with a 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 summarized next.

This can be done when the event detection algorithm has a memory equal to one, which allows one 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 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 one 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

$$\mathbf{y}_j = \begin{cases} 1 & \text{if } \mathbf{y}'_j > 0 \\ 0 & \text{if } \mathbf{y}'_j = 0. \end{cases} \tag{14}$$

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, particularly 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 we check if they enable the adversary to change the output of event detection while evading anomaly detection. This is an unknown value, but we are guaranteed that the problem is well posed because (14) 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 [12], as a binary variable is zero if its correspondent in the relaxed solution is zero. However, the RINS ultimately runs a simplified binary optimization while we simplify the problem until all variables are continuous. Thus, we do not run any binary optimization.

5.2.2 Linearizing 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 spatial domains. 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 6.2, we show that an event may be detected when at least one sensor reading is above a threshold. Another example is given in Section 6.3, where an event is detected if the variability in at least one neighborhood 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 domains, and a constant threshold:

$$\begin{aligned} e(\check{\mathbf{X}}_{w_t}) \geq T_e \approx \\ A^{(1)}\check{\mathbf{X}}_{w_t}B^{(1)} \geq T_{e_1} \vee A^{(2)}\check{\mathbf{X}}_{w_t}B^{(2)} \geq T_{e_2} \vee \cdots \vee A^{(Z)}\check{\mathbf{X}}_{w_t}B^{(Z)} \geq T_{e_Z}, \end{aligned} \tag{15}$$

where the terms $A^{(1)}, \ldots, A^{(Z)}$ and $B^{(1)}, \ldots, B^{(Z)}$ 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 analyze the spatial domain (e.g., event diffusion pattern). Their multiplication with the $W \times N$ matrix $\check{\mathbf{X}}_{w_t}$ returns a scalar, which is compared to the thresholds $T_{e_1}, \ldots, T_{e_Z}$.

In eliciting scenarios, (10) becomes a series of Z optimization 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 (10b) of optimization problem (10) 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.

5.2.3 *Linearizing the Anomaly Score Function.* The anomaly score function in (8) involves two nonlinear terms: the high scales contribution $I(\tau)$ and the function q that takes it as input and returns the maximum accepted low-scale coefficient.

The term $I(\tau)$ involves the computation of $n_s n_\tau^2$ squares of N-order polynomials, where n_s and n_τ respectively are the number of scales and translations. This has an asymptotic complexity $O(n_s n_\tau^2 N^2)$. Computing $I(\tau)$ exactly becomes soon 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 (10) more scalable with the number of sensors.

The constant term includes the wavelet transform of the original measurements signal, whereas the variables contain the manipulations (i.e., the difference between the corrupted and original signal). Thus, 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 linearized to obtain an entirely linear approximation for the anomaly score. Since the input and output of function q in (8) are in a monotonic relationship (larger high-scale coefficients are usually matched by larger low-scale coefficients), standard linearization techniques, such as linear regression [16], give accurate results.

Note that the evasion requirement is not guaranteed with the linearized anomaly score, as it may also give lower estimates. This requires us to validate a solution found with the linearized anomaly score against the actual anomaly detection algorithm before accepting it.

5.2.4 *Transforming Bilinear Terms Into Linear Terms.* After linearizing the anomaly score function, the only term that makes the optimization problem nonlinear 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:

$$\mathbf{W}_{ij} = \mathbf{y}_j \mathbf{M}_{ij} \quad i \in 1, \ldots, T \quad j \in 1, \ldots N. \tag{16}$$

Rather than adding the precediing equation as a constraint, which would still be bilinear, we use the *McCormick envelopes* [38], which are known to well approximate (16).

$$\begin{aligned}
\mathbf{W}_{ij} &\geq \mathbf{M}_{ij}^{inf} \mathbf{y}_j \\
\mathbf{W}_{ij} &\geq \mathbf{M}_{ij}^{sup} \mathbf{y}_j + \mathbf{M}_{ij} - \mathbf{M}_{ij}^{sup} \\
\mathbf{W}_{ij} &\leq \mathbf{M}_{ij}^{sup} \mathbf{y}_j \\
\mathbf{W}_{ij} &\leq \mathbf{M}_{ij} + \mathbf{M}_{ij}^{inf} \mathbf{y}_j - \mathbf{M}_{ij}^{inf} \\
\mathbf{M}_{ij}^{inf} &\leq \mathbf{M}_{ij} \leq \mathbf{M}_{ij}^{sup} \\
\mathbf{M}_{ij}^{inf} &\leq \mathbf{W}_{ij} \leq \mathbf{M}_{ij}^{sup}
\end{aligned} \tag{17}$$

The McCormick constraints are introduced for all $j \in 1, \ldots, N$, giving a total of $8N$ additional constraints per each time instant. The result is the surrogate problem (18), which replaces the variables as in (16), and whose minimum is an overestimation for the original problem.

$$\min_{y} \sum_{j=1}^{N} (\mathbf{y}_j) \tag{18}$$

$$s.t.$$

$$\mathbf{y}_j \in \{0,1\}\ \forall j \in 1,\ldots,N \tag{18a}$$

$$e(\mathbf{X}_{w_t}) \neq e(\mathbf{X}_{w_t} \pm \mathbf{W}_{w_t})\ \forall t \in 1,\ldots,T-W+1 \tag{18b}$$

$$a(\mathbf{X}_i \pm \mathbf{W}_i) \leq T_d\ \forall i \in 1,\ldots,T \tag{18c}$$

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, (18) introduces a slight modification to (10), 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, and therefore $\mathbf{M}_{ij}^{inf}$ can always be set to zero (which corresponds to no malicious manipulation), whereas $\mathbf{M}_{ij}^{sup}$ 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 $N = C$, as a genuine rest condition should always be allowed to transition to a genuine event and vice versa.

5.3 Solving Algorithm

To solve Problem (18), we use Algorithm 2: *findAttackVector*. This builds the optimization problems that are involved in the eliciting/masking attack and collects the results.

In a first step, it builds the optimization 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. Thus, the problem will be optimized with respect to the dummy variables $\mathbf{W}_{ij}$, whereas 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 optimization problem that it has built to Algorithm 3: *solveRelaxedAndCheck*. Since eliciting problems are solved as the best solution among the Z optimization problems that arise by enabling the sufficient event conditions one by one, *solveRelaxedAndCheck* is run Z times between lines 22 and 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 Z sufficient event conditions. Thus, masking scenarios require only one call to Algorithm 3, which solves the problem built between lines 30 and 37. This problem contains Z event detection constraints that all require the event conditions not to be satisfied.

Specifically, *solveRelaxedAndCheck* first 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 one 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

ALGORITHM 2: *findAttackVector*

```
Input: $X_{w_t}$
Output: $C_{min}$, $\mathbf{M}^*$, $\mathbf{y}^*$
 1: elicit = $e(X_{w_t}) < T_e$
 2: $\mathcal{P}^r$ = new optimization problem {Initialize a new optimization problem.}
 3: for $j \in 1, \ldots, N$ do
 4:     Add constraint to $\mathcal{P}^r$: $y_j \in [0, 1]$ {Binary constraints relaxation}
 5:     for $i \in 1, \ldots, W$ do
 6:         Add constraint to $\mathcal{P}^r$: McCormick constraints (17). Replace all occurrences of $\mathbf{y}_j\mathbf{M}_{ij}$ in $\mathcal{P}^r$ with
            $\mathbf{W}_{ij}$ {Bilinear constraints relaxation}
 7:     end for
 8: end for
 9: if elicit then
10:     $\check{\mathbf{X}}_{w_t} = X_{w_t} + \mathbf{W}$
11: else
12:     $\check{\mathbf{X}}_{w_t} = X_{w_t} - \mathbf{W}$
13: end if
14: for $\boldsymbol{\tau} \in \Omega$ do
15:     $D(\boldsymbol{\tau}) = (T^{wav}S)(s_0, \boldsymbol{\tau})$
16:     $I(\boldsymbol{\tau}) = \sum_{\boldsymbol{t}} \sum_{s \in hs, \ldots, Hs} (T^{wav}_{ns} \check{\mathbf{X}}_{w_t})^2 (s, \boldsymbol{\tau}) m(s, \boldsymbol{t} - \boldsymbol{\tau})$
17:     $a(\check{\mathbf{X}}_{w_t}, \boldsymbol{\tau}) = q(I(\boldsymbol{\tau}))/D(\boldsymbol{\tau})$
18:     $a^l(\check{\mathbf{X}}_{w_t}, \boldsymbol{\tau})$ = Linearise $(a(\check{\mathbf{X}}_{w_t}, \boldsymbol{\tau}))$
19:     Add constraint to $\mathcal{P}^r$: $a^l(\check{\mathbf{X}}_{w_t}, \boldsymbol{\tau}) < T_d$ {Linearized evasion constraints}
20: end for
21: $C_{min} = N$, $\mathbf{M}^* = \{0, \ldots, 0\}$, $\mathbf{y}^* = \{0, \ldots, 0\}$
22: if elicit then
23:     for $z \in 1, \ldots, Z$ do
24:         Add constraint to $\mathcal{P}^r$: $A^{(z)}\check{\mathbf{X}}_{w_t}B^{(z)} \geq T_e$ {Eliciting constraint}
25:         $C'$,$\mathbf{M}'$,$\mathbf{y}'$ = solveRelaxedAndCheck($\mathcal{P}^r$,$C_{min}$) {Eliciting problem built. Now solve it.}
26:         if $C' < C_{min}$ then
27:             $C_{min} = C'$, $\mathbf{M}^* = \mathbf{M}'$, $\mathbf{y}^* = \mathbf{y}'$
28:         end if
29:     end for
30: else {Masking Scenario}
31:     for $z \in 1, \ldots, Z$ do
32:         Add constraint to $\mathcal{P}^r$: $A^{(z)}\check{\mathbf{X}}_{w_t}B^{(z)} \leq T_e$ {Masking constraints}
33:     end for
34:     $C'$,$\mathbf{M}'$,$\mathbf{y}'$ = solveRelaxedAndCheck($\mathcal{P}^r$,$C_{min}$) {Masking problem built. Now solve it.}
35:     if $C' < C_{min}$ then
36:         $C_{min} = C'$, $\mathbf{M}^* = \mathbf{M}'$, $\mathbf{y}^* = \mathbf{y}'$
37:     end if
38: end if
39: return $C_{min}$, $\mathbf{M}^*$, $\mathbf{y}^*$
```

remainder are set to zero and the new problem, where the binary variables have fixed values, is solved (lines 4 through 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 (18). However, the latter includes a linear approximation of the real anomaly detection algorithm, which is the one that

Table 2. 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

ALGORITHM 3: *solveRelaxedAndCheck*

```
Input: $\mathcal{P}^r$,$C_{min}$
Output: $C'$, $\mathbf{M}'$, $\mathbf{y}'$
 1: $C' = C_{min}$, $\mathbf{M}^* = \{0, \dots, 0\}$, $\mathbf{y}' = \{0, \dots, 0\}$
 2: Solve $\mathcal{P}^r$ and get the relaxed solution $\mathbf{M}^r$, $\mathbf{y}^r$
 3: $\text{obj}^r = \sum_{j \in 1,\dots,N} \mathbf{y}_j^r$
 4: for $C \in \text{obj}^r, \dots, C_{min}$ do
 5:     $\mathcal{P}' =$ copy of $\mathcal{P}^r$
 6:     $\mathbf{y}' = \mathbf{y}^r$
 7:     sortInd = argsort($\mathbf{y} \odot \mathbf{M}_i$)
 8:     $\mathbf{y}_i' = 0 \quad \forall i \in \text{sortInd}(1), \dots, \text{sortInd}(N - C)$
 9:     $\mathbf{y}_i' = 1 \quad \forall i \in \text{sortInd}(N - C + 1), \dots, \text{sortInd}(N)$
10:     Solve $\mathcal{P}'$ and get the solution $\mathbf{M}'$
11:     if $\mathbf{M}'$ and $\mathbf{y}'$ respect the nonlinearized undetectability constraint $a(\check{\mathbf{X}}_{w_t}) < T_d$ then
12:         $C' = C$, $\mathbf{M}' = \mathbf{M}^r$, $\mathbf{y}' = \mathbf{y}^r$
13:     end if
14: end for
15: return $C'$, $\mathbf{M}'$, $\mathbf{y}'$
```

should be evaded. For this reason, the solution is considered only if it achieves evasion on the anomaly detector (lines 11 and 12).

This algorithm was implemented using the *Gurobi* solver [19] as a back-end for the optimization of the basic problems. Gurobi is an efficient solver for linear and (convex) quadratic optimization. However, *findAttackVector* requires the optimization of several problems and the construction of complex constraints that become especially burdensome with high values of N. To point out the problem's complexity, we report in Table 2 the execution times in minutes of the *findAttackVector* function, executed by an Intel® Core$^{\text{TM}}$ 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 6.1. 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.

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 optimization problems, there are Z event detection constraints, but Algorithm *solveRelaxedAndCheck* is run only once per experiment.

The times in Table 2 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 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 period of 2 weeks. This confirms that the approximations made in

Section 5.2 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, as the constraint of respecting the intersensor correlations would require the different parts to frequently communicate with each other, which results in a loss of the main advantages of parallelism.

6 ATTACK VECTOR FOR EVASION

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 in 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 the 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—that is, 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 intersensor correlation that 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 optimization algorithm for each configuration of the aforementioned elements, we take the average of all minimum numbers of malicious sensors that are needed to subvert event detection across the 20 different scenarios. In this way, we have reliable and contextualized 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 next the model-generated measurements for wildfire-monitoring WSNs as a case study.

6.1 Case Study: Wildfire-Monitoring Wireless Sensor Networks

Wildfire-monitoring WSNs are well suited for the evaluation of measurements inspection techniques. Indeed, they observe events (i.e. fires) that have nontrivial 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 for the analyses are temperatures perceived by N sensors randomly scattered in an area of $100 \times 100\text{m}^2$. 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 (e.g., in the order of 10 hectares), we need to complement measurements inspection with other techniques, such as software attestation, as shown in Illiano et al. [25].

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 maximizes the WSN coverage, and also falling back on a pseudogrid due to the presence of deployment constraints. Note that having defined the deployment area size (10,000m^2) and the deployment criterion, the average sensor density is well defined and is equal to

$$\rho_N = D_A / N \frac{1}{\mathrm{m}^2}, \tag{19}$$

where D_A is the deployment area (i.e., D_A = 10,000m^2 in our analyses). The quantity ρ_N, rather than N, is what ultimately matters for the performance of event detection and anomaly detection, as it tells how fine-grain the information gathered about the physical phenomenon is. Nevertheless, since the deployment area is well defined, we will refer to the number of sensors rather than their density, as it is easier to visualize and interpret.

The temperature sensors may observe temperature increases due to the outbreak of fires within the WSN area and anywhere next to it (hence, a fire may be only partially observable for the sensors). The fires are modeled as the result of the radiated power from one or more heat sources placed at different locations, which are modeled as spherical black bodies with temperature K_i (between 573.15 and 1,473.15 Kelvin [52, 55]). Such black bodies have radius r_i, centered in $\mathbf{c_i}$, and their power is radiated to each location $\mathbf{u} \in \Omega$ at distance $d(\mathbf{c_i}, \mathbf{u})$, producing the temperature:

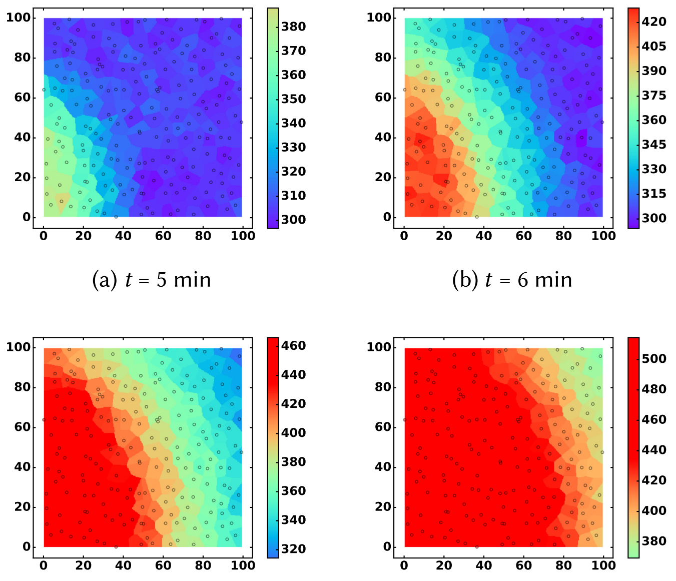
$$K(\mathbf{u}) = \sum_i \left(\frac{1}{4} {K_i}^4 \left(\frac{r_i}{d(\mathbf{c_i}, \mathbf{u})} \right)^2 \right)^{\frac{1}{4}} + \mathcal{N}(0, 3), \tag{20}$$

where $\mathcal{N}$ is additive zero-mean Gaussian noise, with a standard deviation equal to 3 simulating the interference that arises from the sensing devices.

The time evolution of the fires is modeled 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 (e.g., slope of the terrain, wind speed, and grass type) [41]. Existing simulation environments such as *FARSITE*—developed by the U.S. Forest Service [14]—are more accurate and consider different combinations of the variables involved, including wind, fuel, terrain slope, and so forth. However, these simulators focus on the wildfire's spreading behavior and do not provide the spatiotemporal distribution of temperatures.

An example of the resulting temperature evolution in time is shown in Figure 2. 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 Figure 2(d).

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.



(a) t = 5 min (b) t = 6 min

(c) t = 7 min (d) t = 8 min

Fig. 2. Example of fire evolution in time. The WSN area is a two-dimensional space. The circles identify the sensors, and colors map the temperatures in Kelvin. Note that the sensors are not placed in a grid but are randomly scattered.

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 Figure 2(a), the fire can already be well spread.

6.2 Simplistic Event Detection Algorithm

The performance of measurements inspection is dependent on the event detection algorithm, as 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—that is,

$$(\check{\mathbf{X}}_{i,1} \geq T_e) \vee (\check{\mathbf{X}}_{i,2} \geq T_e), \ldots, \vee(\check{\mathbf{X}}_{i,N} \geq T_e). \tag{21}$$

That corresponds to triggering a fire alarm if any of the N temperatures are above T_e, usually set to 60°C (corresponding to 333.15 Kelvin). This choice makes it particularly easy to spoof an event, as 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 3.

The results with eliciting attacks shown earlier are obtained by averaging the output of Algorithm 2 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 2 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 (e.g., as in

Table 3. Minimum Number of Malicious Sensors for Both Eliciting and Masking Attacks With Different Sensor Numbers (and Density in Turn)

Attack \ N	10	30	60	90	100	200
Eliciting	3	3	5	7	7	7
Masking	4	11	21	33	34	81

Note: The eliciting attacks require a small set of sensors to be bompromised due to the event detection algorithm, which requires just one sensor to trigger.

Figure 2(d)). 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 Figure 2(b)). Generally, at this stage, the fire is above the 60°C for about 25% of the sensors, whereas the measurements inspection algorithm requires the attacker to compromise around 37% of the sensors to mask the event while 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, and hence the extra two to four sensors that are needed to avoid detection by the measurements inspection algorithm are actually a 200% to 400% gain. The second reason is that it is not unlikely for temperatures to jump from 20 to 60°C in a few meters, as the temperatures may show a large variation in the presence of a fire, and 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 characterising 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 to all possible scenarios introduce uncertainty in the characterization 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 3, where the number of sensors needed to mask the event defined in (21) is almost one order of magnitude bigger than in the eliciting case.

This happens because, with such an event detection, all 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, as 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 tradeoff between the thorough characterization of an event's characteristic and the characterization 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.

6.3 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*, such as temperature and humidity [49, 51], 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 analyze 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 both the absence and 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 neighborhood:

$$\frac{1}{N(j)} \sum_{N(j)} \delta^{W}_{\check{\mathbf{X}}_{ij}} > T_e. \tag{22}$$

In (20), $\delta^{W}_{\check{\mathbf{X}}_{ij}}$ 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 neighborhood. The variability of the measurements is averaged across the neighborhood 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 next:

$$\delta^{W}_{\check{\mathbf{X}}_{ij}} = \mathrm{Var}(\{\check{\mathbf{X}}_{kj}\}_{k=i-W}^{i-1}). \tag{23}$$

The nontrivial 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 follows:

$$\frac{1}{W-1} \sum_{i=1}^{W} (\check{\mathbf{X}}_{ij} - \overline{\check{\mathbf{X}}_{j}})^2, \tag{24}$$

where

$$\overline{\check{\mathbf{X}}_{j}} = \frac{1}{W} \sum_{i=1}^{W} (\check{\mathbf{X}}_{ij}). \tag{25}$$

When the values of $\check{\mathbf{X}}_{ij}$ are known, evaluating this formula has a complexity $O(W)$, but when they are unknown, the complexity becomes $O(W^3)$ because it involves W squares of the sum of $W+1$ monomials. For large values of W, problems with such an event detection algorithm have an infeasible computational complexity just for defining the event detection equations.

For this reason, rather than (24), we use an approximation [62], which is

$$\frac{1}{W^2} \sum_{i=1}^{W} \sum_{k=i+1}^{W} (\check{\mathbf{X}}_{ij} - \check{\mathbf{X}}_{kj})^2. \tag{26}$$

Table 4. Minimum Number of Malicious Sensors

Attack \ N	10	30	60	90	100	200
Eliciting	6	14	25	33	33	66
Masking	6	20	53	75	80	162

Note: The improvement in the event detection algorithm increased the resiliency, especially against eliciting attacks.

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 $O(W^2)$ rather than $O(W^3)$.

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 two monomials, reaching a complexity of evaluating the formula of $O(W)$. Such an approximation is as follows:

$$\frac{1}{2W}\sum_{i=1}^{W}(\check{\mathbf{X}}_{i+1\,j} - \check{\mathbf{X}}_{ij})^2. \tag{27}$$

Note that constraints of the kind $\frac{1}{2W}\sum_{i=1}^{W}(\check{\mathbf{X}}_{i+1\,j} - \check{\mathbf{X}}_{ij})^2 \geq T_e$, which would be introduced for eliciting problems, make the problem nonconvex. Although convex problems can be solved in polynomial time, nonconvex problems are known to be $\mathcal{NP}$-hard. To avoid this problem, we can remove the square and obtain the following:

$$\frac{1}{2W}\sum_{i=1}^{W}\check{\mathbf{X}}_{i+W\,j} - \check{\mathbf{X}}_{ij}. \tag{28}$$

The value returned by (28) 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, as 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 (28), averaged across a neighborhood, to a positive threshold. Ultimately, our new event detection criterion is as follows:

$$\frac{1}{N(j)}\sum_{N(j)}\frac{1}{2W}\sum_{i=1}^{W}\check{\mathbf{X}}_{i+W\,j} - \check{\mathbf{X}}_{ij} > T_e. \tag{29}$$

The results with the new event detection algorithm are shown in Table 4. Note that the masking results have increased by 3.5 times on average, whereas the eliciting results have increased by 8.7 times on average. As a consequence, the gap between eliciting and masking results has been reduced as well. 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.

Note that since we defined the neighborhood of a sensor as the sensors whose distance is less than 80m, with a maximum of three sensors an attacker can influence all neighborhoods in the network. Indeed, this is the minimum number of circles with a radius equal to 80 needed to cover a square area with size 100. This means that with just three sensors, an attacker can spoof or mask any event if no anomaly detection is in place,[2] and that if we subtract 3 from the results in Table 4, 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 to another state-of-the-art technique, we have also run the same optimization algorithm using the anomaly detection algorithm described in Chatzigiannakis and Papavassiliou [10], based on PCA. Being based on linear transformations, this algorithm did not require a linearization step and could be immediately used in the optimization Algorithm 2. In this case, we have run each experiment 20 times. The results showed that even with the highest number of sensors (i.e., 200), one malicious sensor was enough to spoof an event most of the time, and in a few cases, the number of sensors to compromise increased only up to three. For the masking scenario, the average was one, with a maximum of four across the 20 experiments. Even though PCA-based anomaly detection outperforms many similar techniques and is able to cope with correlated anomalies as well, 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 three or fewer sensors, an attacker can subvert the event detection algorithm in (29). To achieve that also while 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 neighborhood, which is the criterion that identifies events in (29), to be spread across the neighborhood. In this way, the subsets of colluding sensors need to be large, as otherwise there will be some genuine sensor that has information about the event presence/absence and whose measurements are not consistent with those of the malicious ones. The analysis shows that the technique presented in this article can be used for comparing different anomaly detection techniques as well.

7 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 while staying undetected. However, besides the minimum number of compromised sensors, the optimization algorithm also returns the specific choices that an adversary would take in the process of impairing event detection while evading anomaly detection, with a budget of compromised sensors.

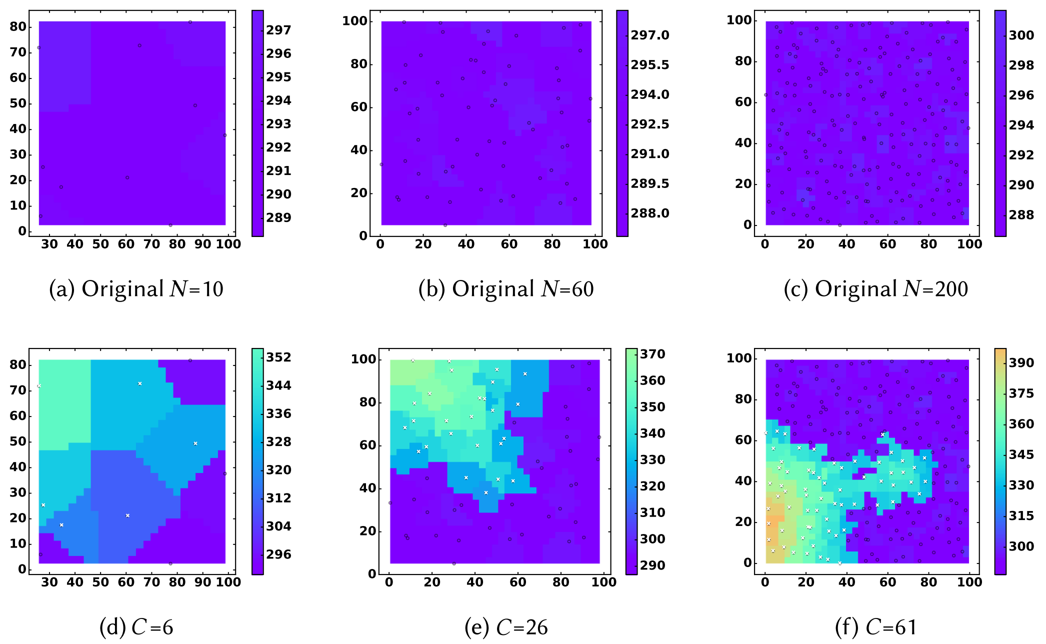
With such information, we are able to learn the most sophisticated strategies to violate event integrity. This can be useful in identifying the main threats in the WSN, as well as the main weaknesses of the anomaly detection algorithm.

With the help of graphical tools, we analyze 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 that follow.

7.1 Discussion of Eliciting Results

In Figure 3, we show the effects of the malicious data injection attack produced in output by the optimization algorithm. These measurements are able to spoof an event—that is, making the event detection condition (29) true and preventing detection by the wavelet-based anomaly detection algorithm.

[2]This is because the event detection algorithm makes an average of the temporal variation of each sensor in the neighborhood. Hence, if a sensor produces an extremely low/high temporal variation, it can mask/elicit an event on its own.



(a) Original N=10 (b) Original N=60 (c) Original N=200

(d) C=6 (e) C=26 (f) C=61

Fig. 3. Examples of elicited scenarios with increasing numbers of sensors. The circles represent the sensors. The white crosses indicate which sensors are compromised. Colors map the measurements space: bluish colors are for low temperatures, and reddish colors are for high temperatures.

A time snapshot of the original measurements (without the attack) is shown in Figure 3(a) through (c) 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.

The subset of malicious measurements returned by the optimization algorithm manages to spoof an event. In particular, the compromised sensors transform the scenarios shown in Figure 3(a) through (c) respectively into the scenarios shown Figure 3(d) through (f). 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 summarized next.

7.1.1 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 were 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 that 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 and (2) well spaced to avoid introducing less protected areas in the WSN.

7.1.2 Optimal Location for Spoofing Events. Figure 3(d) through (f) 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 neighbors, 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 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.

7.1.3 Real Event Mimicry. If we compare Figure 3(f) to Figure 2(a) or (b), 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, toward the outskirts where the temperatures progressively decay. **These characteristics were emulated by the evasion optimization algorithm without domain-specific knowledge about what events should look like**. Of course, the optimization problem includes the event detection algorithm in its constraints, but this just requires the measurements variation to increase in time, although there is no information related to the events' spatial patterns. This means that it is the measurements inspection algorithm that has learned 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. However, 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.

This is certainly a desirable outcome, as the best result that a measurements inspection algorithm can achieve is to force the attacker to create measurements that are exactly as they would manifest in genuine scenarios, which usually implies that the attacker needs to compromise more sensors.

7.2 Discussion of Masking Results

In Figure 4, we show the effects of some of the worst-case masking attacks, calculated by the optimization algorithm. The malicious measurements are able to mask an event (i.e., making the event detection condition (29) false) and prevent detection from the wavelet-based anomaly detection algorithm.

Figure 4(a) through (c) show the measurements perceived under real events, after 6 minutes from the fire outbreak, in deployments of 10, 60, and 200 sensors, respectively. Figure 4(d) through (f) show optimal masking attacks against these scenarios—that is, the measurements after the masking attacks that transform the event conditions in Figure 4(a) through (c) 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 next.

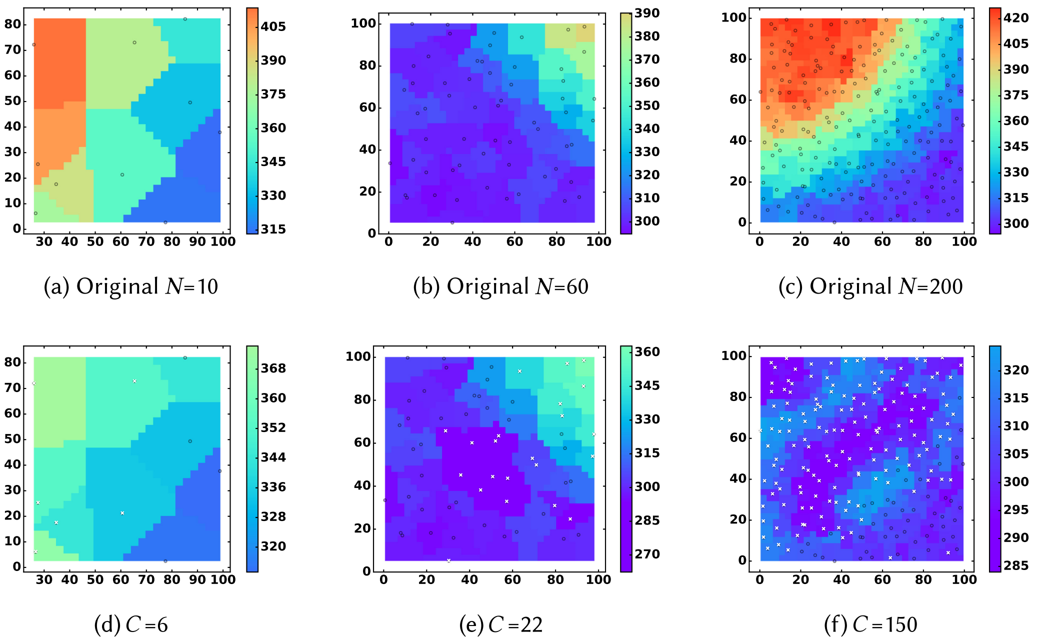


Fig. 4. Examples of masked scenarios with increasing numbers of sensors. The circles represent the sensors. The white crosses indicate which sensors are compromised. Colors map the measurements space: bluish colors are for low temperatures, and reddish colors are for high temperatures.

7.2.1 Minimum Effort. The resulting measurements in Figure 4(d) through (f) are rest measurements because the event detection algorithm does not trigger. However, we note that only Figure 4(f) is a pure rest scenario, whereas Figure 4(d) and (f) are closer to event conditions than to rest conditions. In other words, the event score that is compared to the threshold in (29) 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 Figure 4(f), 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 remaining resources can be concentrated on staying undetected. Something similar happens in the eliciting scenario as well, as the spoofed events are also at the boundary of event detection—that is, 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 analyzing the temporal evolution of the event score as well. 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 behavior around the threshold in general. Thus, it may be worthwhile checking that the time evolution of the event score does not show anomalous behaviors 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 in the space domain as well. As we highlighted in Section 4, anomaly detection based only on the time domain can be circumvented by an attacker by mimicking the temporal dynamics of the monitored phenomena.

7.2.2 Smoother Transitions. In Figure 4(d), 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 southeast, which correspond to low temperatures.

This may seem counterintuitive, because in the original genuine scenario, depicted in Figure 4(a), 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, which the wavelet-based anomaly detection algorithm has captured. As a consequence, the masking sensors are forced to reduce the measurements gradient while decreasing the measurements for masking the fire.

7.2.3 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 6.1. As a consequence, the sensors that are not perceiving the event at all at a certain time may perceive it in the future. Thus, 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 Figure 4(c), we show a fire that has covered most of the WSN deployment, and from Figure 4(f), we deduce that the attacker needs to compromise 75% of the sensors to mask the event.

This effect depends on the nature of the events and particularly 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.

8 DESIGN OF SECURE WIRELESS SENSOR NETWORKS

Evaluating the risk of a system makes sense only if such a 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. Thus, 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.

ALGORITHM 4: *designDeployment*

```
Input: $C_{max}$, $tol$
Output: $N_{min}$
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:     $C_{min}, \mathbf{M}^*, \mathbf{y}^* = findAttackVector(X_{w_t})$
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 $N_{min}^{sup}$
```

This problem shares some similarities to that of *k-coverage* deployment in sensor networks [20], 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 localization. However, the events' presence can be determined only with statistical analyses on many sensors in other applications, such as 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 [13, 64]. 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 fewer 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 4.

This algorithm is a binary search in the space of all possible values N, which 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 optimization time, as 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.

Table 5. *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

Moreover, Algorithm 4 gives assurance about the number of compromised sensors that can be tolerated only for a specific scenario—that is, a set of original genuine measurements that will be partially compromised. To obtain an assurance that applies to generic scenarios, we run Algorithm 4 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 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 summarized in Table 5.

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 to be 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, such as 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, as 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,[3] the damage is minimized. This aspect should be taken into consideration when designing an event detection WSN.

Thanks to Algorithm 4, 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, although the resilience of the WSN obtained through measurements inspection can be estimated through the methodology introduced in this article, there is currently no means to make analogous analyses for other techniques or to combine them.

9 CONCLUSIONS AND FUTURE WORK

Analyzing the potential damage of malicious data injections in WSNs requires a well-defined problem for the attacker. In this article, 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.

[3]For example, with low risk, activate sprinkler systems; with medium risk, send firemen; and with high risk, dispatch firefighting helicopters.

We have proposed a novel formulation of this as an optimization problem and introduced several approximations to solve it computationally. Therefore, we have proposed an algorithm that linearizes the optimization problem, solves it, and maps the solutions back to the original problem.

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.

We have highlighted that the event detection algorithm can reduce considerably the benefits of measurements inspection if it does not characterize events properly. In particular, a loose characterization of events favors spoofing, whereas a strict characterization makes the masking task easier. A trade-off is required in the number of event conditions that are evaluated to maximize 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 the anomaly detection algorithm. For instance, we have observed that for the attacker, it is more advantageous to compromise neighboring 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 prevents 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. Such an algorithm not only gives security guarantees, which are generally missing in the anomaly detection field, but also evaluates the applicability of anomaly detection itself.

In the future, we aim to study how to improve measurements inspection techniques, starting from the weak spots that have emerged in this work. For instance, we may occasionally have recourse to more expensive techniques, such as software attestation or tamper-proof hardware, for a small subset of sensors. Thanks to the analyses on intersensor correlation carried out by measurements inspection, a large set of sensors would benefit from it.

ACKNOWLEDGMENTS

We are grateful to the anonymous reviewers for their contribution in improving the quality of this article. We thank our colleagues of the RISS research group (http://www.rissgroup.org), who have contributed to this work through their valuable comments.

REFERENCES

[1] Gartner. 2015. Gartner Says 6.4 Billion Connected "Things" Will Be in Use in 2016, Up 30 Percent From 2015. Retrieved January 16, 2018, from http://www.gartner.com/newsroom/id/3165317.

[2] Idris M. Atakli, Hongbing Hu, Yu Chen, Wei-Shinn Ku, and Zhou Su. 2008. Malicious node detection in wireless sensor networks using weighted trust evaluation. In *SpringSim*, H. Rajaei, G. A. Wainer, and M. J. Chinni (Eds.). SCS/ACM, 836–843. http://dblp.uni-trier.de/db/conf/springsim/springsim2008.html#AtakliHCKS08.

[3] Zorana Bankovic, José Manuel Moya, Álvaro Araujo, David Fraga, Juan Carlos Vallejo, and Juan-Mariano de Goyeneche. 2010. Distributed intrusion detection system for wireless sensor networks based on a reputation system coupled with kernel self-organizing maps. *Integrated Computer-Aided Engineering* 17, 2, 87–102. http://dblp.uni-trier.de/db/journals/icae/icae17.html#BankovicMAFVG10.

[4] Fenye Bao, Ing-Ray Chen, Moonjeong Chang, and Jin-Hee Cho. 2012. Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection. *IEEE Transactions on Network and Service Management* 9, 2, 169–183.

[5] Timo Berthold. 2014. *Heuristic Algorithms in Global MINLP Solvers*. Verlag Dr. Hut.

[6] Luís M. A. Bettencourt, Aric A. Hagberg, and Levi B. Larkey. 2007. Separating the wheat from the chaff: Practical anomaly detection schemes in ecological applications of distributed sensor networks. In *Distributed Computing in Sensor Systems*. Lecture Notes in Computer Science, Vol. 4549. Springer, 223–239. http://dblp.uni-trier.de/db/conf/dcoss/dcoss2007.html#BettencourtHL07.

[7] Battista Biggio. 2010. *Adversarial Pattern Classification*. Ph.D. Dissertation. University of Cagliari, Cagliari, Italy.

[8] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. 2013. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases*. Lecture Notes in Computer Science, Vol. 8190. Springer, 387–402.

[9] Haowen Chan, Adrian Perrig, and Dawn Song. 2006. Secure hierarchical in-network aggregation in sensor networks. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*. ACM, New York, NY, 278–287.

[10] V. Chatzigiannakis and S. Papavassiliou. 2007. Diagnosing anomalies and identifying faulty nodes in sensor networks. *IEEE Sensors Journal* 7, 5, 637–645. DOI: http://dx.doi.org/10.1109/JSEN.2007.894147

[11] Ana Paula R. da Silva, Marcelo H. T. Martins, Bruno P. S. Rocha, Antonio A. F. Loureiro, Linnyer B. Ruiz, and Hao Chi Wong. 2005. Decentralized intrusion detection in wireless sensor networks. In *Proceedings of the 1st ACM International Workshop on Quality of Service and Security in Wireless and Mobile Networks*. ACM, New York, NY, 16–23.

[12] Emilie Danna, Edward Rothberg, and Claude Le Pape. 2005. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming* 102, 1, 71–90. DOI: http://dx.doi.org/10.1007/s10107-004-0518-7

[13] Santpal S. Dhillon, Krishnendu Chakrabarty, and S. Sitharama Iyengar. 2002. Sensor placement for grid coverage under imprecise detections. In *Proceedings of the 5th International Conference on Information Fusion*, Vol. 2. IEEE, Los Alamitos, CA, 1581–1587.

[14] Mark A. Finney. 1998. *FARSITE: Fire Area Simulator—Model Development and Evaluation*. Vol. 3. U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station Ogden.

[15] Prahlad Fogla and Wenke Lee. 2006. Evading network anomaly detection systems: Formal reasoning and practical techniques. In *Proceedings of the Conference on Computer and Communications Security*. 59–68.

[16] Andrew Gelman and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press. http://www.amazon.com/Analysis-Regression-Multilevel-Hierarchical-Models/dp/052168689X/ref=sr_1_1?s=books&ie=UTF8&qid=1313405184&sr=1-1.

[17] Amir Globerson and Sam Roweis. 2006. Nightmare at test time: Robust learning by feature deletion. In *Proceedings of the International Conference on Machine Learning*. 353–360.

[18] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *Proceedings of the International Conference on Learning Representations*.

[19] Gurobi. 2015. Gurobi Optimizer Reference Manual. Available at http://www.gurobi.com.

[20] Chi-Fu Huang and Yu-Chee Tseng. 2005. The coverage problem in a wireless sensor network. *Mobile Networks and Applications* 10, 4, 519–528.

[21] L. Huang, A. D. Joseph, B. Nelson, B. I. P. Rubinstein, and J. D. Tygar. 2011. Adversarial machine learning. In *Proceedings of the Workshop on Security and Artificial Intelligence*. 43–58.

[22] S.-I. Huang, S. Shieh, and J. D. Tygar. 2010. Secure encrypted-data aggregation for wireless sensor networks. *Wireless Networks* 16, 4, 915–927.

[23] Vittorio P. Illiano and Emil C. Lupu. 2015. Detecting malicious data injections in wireless sensor networks: A survey. *ACM Computing Surveys* 48, 2, Article 24, 33 pages. DOI: http://dx.doi.org/10.1145/2818184

[24] V. P. Illiano, L. Munoz-Gonzalez, and E. Lupu. 2016. Don't fool me! Detection, characterisation and diagnosis of spoofed and masked events in wireless sensor networks. *IEEE Transactions on Dependable and Secure Computing* PP, 99, 1. DOI: http://dx.doi.org/10.1109/TDSC.2016.2614505

[25] Vittorio P. Illiano, Rodrigo V. Steiner, and Emil C. Lupu. 2017. 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)*. ACM, New York, NY, 134–144. DOI: http://dx.doi.org/10.1145/3098243.3098249

[26] Raja Jurdak, X. Rosalind Wang, Oliver Obst, and Philip Valencia. 2011. Wireless sensor network anomalies: Diagnosis and detection strategies. In *Intelligence-Based Systems Engineering*, A. Tolk and L. C. Jain (Eds.). Intelligent Systems Reference Library, Vol. 10. Springer, Berlin, Germany, 309–325. DOI: http://dx.doi.org/10.1007/978-3-642-17931-0_12

[27] Timothy H. Keitt and Janet Fischer. 2006. Detection of scale-specific community dynamics using wavelets. *Ecology* 87, 11, 2895–2904. http://www.jstor.org/stable/20069308.

[28] Chongkyung Kil, Emre C. Sezer, Ahmed M. Azab, Peng Ning, and Xiaolan Zhang. 2009. Remote attestation to dynamic system properties: Towards providing complete system integrity evidence. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'09)*. IEEE, Los Alamitos, CA, 115–124.

[29] Joonho Kong, Farinaz Koushanfar, Praveen K. Pendyala, Ahmad-Reza Sadeghi, and Christian Wachsmann. 2014. PUFatt: Embedded platform attestation based on novel processor-based PUFs. In *Proceedings of the 51st Annual Design Automation Conference*. ACM, New York, NY, 1–6.

[30] Sung Yul Lim and Yoon-Hwa Choi. 2013. Malicious node detection using a dual threshold in wireless sensor networks.*Journal of Sensor and Actuator Networks* 2, 1, 70–84.

[31] Changwei Liu and Sid Stamm. 2007. Fighting unicode-obfuscated spam. In *Proceedings of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit.* 45–59.

[32] Fang Liu, Xiuzhen Cheng, and Dechang Chen. 2007. Insider attacker detection in wireless sensor networks. In *Proceedings of the 26th IEEE International Conference on Computer Communications (IEEE INFOCOM'07).* IEEE, Los Alamitos, CA, 1937–1945. http://dblp.uni-trier.de/db/conf/infocom/infocom2007.html#LiuCC07.

[33] Yao Liu, Peng Ning, and Michael K. Reiter. 2011. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security* 14, 1, 13. http://dblp.uni-trier.de/db/journals/tissec/tissec14.html#LiuNR11.

[34] Javier Lopez, Rodrigo Roman, Isaac Agudo, and M. Carmen Fernández Gago. 2010. Trust management systems for wireless sensor networks: Best practices. *Computer Communications* 33, 9, 1086–1093. http://dblp.uni-trier.de/db/journals/comcom/comcom33.#LopezRAG10.

[35] Daniel Lowd and Christopher Meek. 2005. Adversarial learning. In *Proceedings of the Conference on Knowledge Discovery in Data Mining.* 641–647.

[36] D. Lowd and C. Meek. 2005. Good word attacks on statistical spam filters. In *Proceedings of the Conference on Email and Anti-Spam.*

[37] Matthew V. Mahoney and Philip K. Chan. 2003. An analysis of the 1999 DARPA/lincoln laboratory evaluation data for network anomaly detection. In *Proceedings of the Workshop on Recent Advances in Intrusion Detection.* 220–237.

[38] Garth P. McCormick. 1976. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming* 10, 1, 147–175.

[39] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the Workshop on Artificial Intelligence and Security.* 1–11.

[40] Blaine Nelson, Benjamin I. P. Rubinstein, Ling Huang, Anthony D. Joseph, Shing-Hon Lau, Steven J. Lee, Satish Rao, Anthony Tran, and J. Doug Tygar. 2010. Near-optimal evasion of convex-inducing classifiers. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS'10).* 549–556.

[41] Lewis Ntaimo, Xiaolin Hu, and Yi Sun. 2008. DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment. *Simulation* 84, 4, 137–155.

[42] Seo Hyun Oh, Chan O. Hong, and Yoon-Hwa Choi. 2012. A malicious and malfunctioning node detection scheme for wireless sensor networks.*Wireless Sensor Network* 4, 3, 84–90.

[43] Ilker Onat and Ali Miri. 2005. An intrusion detection system for wireless sensor networks. In *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking, and Communications*, Vol. 3. IEEE, Los Alamitos, CA, 253–259.

[44] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: From phenomena to black-box attacks using adversarial samples. arXiv:1605.07277.

[45] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2016. The limitations of deep learning in adversarial settings. In *Proceedings of the IEEE European Symposium on Security and Privacy.* 372–387.

[46] Sutharshan Rajasegarar, Christopher Leckie, and Marimuthu Palaniswami. 2008. Anomaly detection in wireless sensor networks.*IEEE Wireless Communications* 15, 4, 34–40. http://dblp.uni-trier.de/db/journals/wc/wc15.html#RajasegararLP08.

[47] Mohsen Rezvani, Aleksandar Ignjatovic, Elisa Bertino, and Sanjay Jha. 2013. A robust iterative filtering technique for wireless sensor networks in the presence of malicious attacks. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys'13).* ACM, New York, NY, Article 30. http://dblp.uni-trier.de/db/conf/sensys/sensys2013.html#RezvaniIBJ13.

[48] Rodrigo Roman, Jianying Zhou, and Javier Lopez. 2006. Applying intrusion detection systems to wireless sensor networks. In *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC'06).*

[49] Osman Salem, Yaning Liu, and Ahmed Mehaoua. 2013. A lightweight anomaly detection framework for medical wireless sensor networks. In *Proceedings of the 2013 IEEE Wireless Communications and Networking Conference (WCNC'13).* IEEE, Los Alamitos, CA, 4358–4363. http://dblp.uni-trier.de/db/conf/wcnc/wcnc2013.html#SalemLM13.

[50] David Sculley, Gabriel Wachman, and Carla E. Brodley. 2006. Spam filtering using inexact string matching in explicit feature space with on-line linear classifiers. In *Proceedings of the 15th Text Retrieval Conference (TREC'06).* 1–10.

[51] Nauman Shahid, Ijaz Haider Naqvi, and Saad B. Qaisar. 2012. Quarter-sphere SVM: Attribute and spatio-temporal correlations based outlier and event detection in wireless sensor networks. In *Proceedings of the 2012 IEEE Wireless Communications and Networking Conference (WCNC'12).* IEEE, Los Alamitos, CA, 2048–2053. http://dblp.uni-trier.de/db/conf/wcnc/wcnc2012.html#ShahidNQ12.

[52] D. A. Smith and G. Cox. 1992. Major chemical species in buoyant turbulent diffusion flames. *Combustion and Flame* 91, 3-4, 226–238. DOI: http://dx.doi.org/10.1016/0010-2180(92)90055-T

[53] Rodrigo Vieira Steiner and Emil Lupu. 2016. Attestation in wireless sensor networks: A survey. *ACM Computing Surveys* 49, 3, Article 51, 31 pages. DOI: http://dx.doi.org/10.1145/2988546

[54] Sharmila Subramaniam, Themis Palpanas, Dimitris Papadopoulos, Vana Kalogeraki, and Dimitrios Gunopulos. 2006. Online outlier detection in sensor data using non-parametric models. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06).* ACM, New York, NY, 187–198. http://dblp.uni-trier.de/db/conf/vldb/vldb2006.html#SubramaniamPPKG06.

[55] A. L. Sullivan, P. F. Ellis, and I. K. Knight. 2003. A review of radiant heat flux models used in bushfire applications. *International Journal of Wildland Fire* 12, 1, 101–110.

[56] Bo Sun, Xuemei Shan, Kui Wu, and Yang Xiao. 2013. Anomaly detection based secure in-network aggregation for wireless sensor networks. *IEEE Systems Journal* 7, 1, 13–25. http://dblp.uni-trier.de/db/journals/sj/sj7.html#SunSWX13.

[57] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. arXiv:1312.6199.

[58] Hailun Tan, Wen Hu, and Sanjay Jha. 2015. A remote attestation protocol with trusted platform modules (TPMs) in wireless sensor networks. *Security and Communication Networks* 8, 13, 2171–2188.

[59] Sapon Tanachaiwiwat and Ahmed Helmy. 2005. Correlation analysis for alleviating effects of inserted data in wireless sensor networks. In *Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05).* IEEE, Los Alamitos, CA, 97–108. http://dblp.uni-trier.de/db/conf/mobiquitous/mobiquitous2005.html#TanachaiwiwatH05.

[60] Gregory L. Wittel and Shyhtsun Felix Wu. 2004. On attacking statistical spam filters. In *Proceedings of the Conference on Email and Anti-Spam*.

[61] Miao Xie, Song Han, Biming Tian, and Sazia Parvin. 2011. Anomaly detection in wireless sensor networks: A survey. *Journal of Network and Computer Applications* 34, 4, 1302–1325. http://dblp.uni-trier.de/db/journals/jnca/jnca34.html#XieHTP11.

[62] Yuli Zhang, Huaiyu Wu, and Lei Cheng. 2012. Some new deformation formulas about variance and covariance. In *Proceedings of the 2012 International Conference on Modelling, Identification, and Control (ICMIC'12).* IEEE, Los Alamitos, CA, 987–992.

[63] Sencun Zhu, Sanjeev Setia, Sushil Jajodia, and Peng Ning. 2007. Interleaved hop-by-hop authentication against false data injection attacks in sensor networks. *ACM Transactions on Sensor Networks* 3, 3, 14.

[64] Yi Zou and Krishnendu Chakrabarty. 2004. Uncertainty-aware and coverage-oriented deployment for sensor networks. *Journal of Parallel and Distributed Computing* 64, 7, 788–798.

Received June 2017; revised October 2017; accepted November 2017