Exploiting Correlations to Detect False Data Injections in Low-Density Wireless Sensor Networks

Zhongyuan Hau
zy.hau17@imperial.ac.uk
Imperial College London
London, United Kingdom

Emil C. Lupu
e.c.lupu@imperial.ac.uk
Imperial College London
London, United Kingdom

ABSTRACT

We propose a novel framework to detect false data injections in a low-density sensor environment with heterogeneous sensor data. The proposed detection algorithm learns how each sensor's data correlates within the sensor network, and false data is identified by exploiting the anomalies in these correlations. When a large number of sensors measuring homogeneous data are deployed, data correlations in space at a fixed snapshot in time could be used as as basis to detect anomalies. Exploiting disruptions in correlations when false data is injected has been used in a high-density sensor setting and proven to be effective. With increasing adoption of sensor deployments in low-density setting, there is a need to develop detection techniques for these applications. However, with constraints on the number of sensors and different data types, we propose the use of temporal correlations across the heterogeneous data to determine the authenticity of the reported data. We also provide an adversarial model that utilizes a graphical method to devise complex attack strategies where an attacker injects coherent false data in multiple sensors to provide a false representation of the physical state of the system with the aim of subverting detection. This allows us to test the detection algorithm and assess its performance in improving the resilience of the sensor network against data integrity attacks.

CCS CONCEPTS

• **Security and privacy → Distributed systems security**; • **Computer systems organization → Sensor networks**.

KEYWORDS

False Data Injections, Sensor Networks, Anomaly Detection

ACM Reference Format:
Zhongyuan Hau and Emil C. Lupu. 2019. Exploiting Correlations to Detect False Data Injections in Low-Density Wireless Sensor Networks. In *5th ACM Cyber-Physical System Security Workshop (CPSS '19), July 8, 2019, Auckland, New Zealand.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3327961.3329530

1 INTRODUCTION

Wireless Sensor networks (WSNs) are an integral part of Cyber-Physical Systems (CPS) where devices are used as interfaces to sense and interact between the cyber and physical realm. Sensors are deployed to collect data from the physical environment and the information is represented in the cyber domain which could be used for further computational processing, analysis and decision making across many applications in public and private spaces. In public areas, WSNs are used to monitor infrastructure usage and surveillance such as smart energy meters for power grids and sensors for water distribution networks. In personal spaces, WSNs are used by individuals in smart homes for monitoring and control of smart appliances, or for health and well-being such as activity tracking and monitoring of physiological parameters. In addition, these devices are fielded in unprotected environment, rendering them vulnerable to physical tampering [9].

The data collected from deployed sensors in WSNs are critical in the various applications of industrial automation, surveillance and health analytics. Attacks that undermine data authenticity can result in financial losses or events that result in loss of lives. Malicious events can be instigated by tampering with sensor readings resulting in industrial process upset in an Industrial Control System or inaccurate administering of medicine dosage in an automated drug delivery system [2, 7]. Therefore, it is important to ensure the authenticity of the data collected by WSNs as the information is processed by sense-making and decision-making engines, where compromised data could potentially result in dire consequences.

The act of tampering (modifying or replacing) of sensor measurements by an attacker is known as **false data injections**. Data collected from WSNs are used to provide a depiction of the sensed physical environment. Any deviations from the expected environment parameters would usually trigger a system response. As such, the goals of an attacker when tampering with sensor measurements can be to:

1. ***Elicit* an undesired system response.** The goal of the attacker is to cause an undesired system response by spoofing a sensed system condition that the system would react to.
2. ***Mask* an undesired system condition.** The goal of the attacker is to hide an undesirable system condition to prevent a system response.

Sensor data can be tampered with at various stages, from data collection to data transmission phase. Thus, with the widespread adoption of WSNs in critical applications, it is important to develop detection mechanisms to secure and ensure these systems are resilient to data integrity attacks [13].

1.1 Background and Related Work

In the research area of securing WSNs and CPSs, many studies focus on ensuring the integrity of the data on the physical and network level [18, 21]. However, it is possible for the sensor measurements to be manipulated before or during transmission.

CPSS '19, July 8, 2019, Auckland, New Zealand
2019. ACM ISBN 978-1-4503-6787-5/19/07…$15.00
https://doi.org/10.1145/3327961.3329530

Recent studies on detecting malicious data injections in CPSs focus on identifying anomalous sensor measurements by leveraging the physical behavior of the systems [27]. Physics-based attack detection of sensor readings involves building a model of the physical system and using models such as Auto-Regressive (AR) or Linear Dynamical State-Space (LDS) to obtain a prediction of the sensor value which will be compared to the reported sensor value [10]. Process Invariants based detection in CPSs [1] were also used to detect attacks by inspecting deviations from the normal behavior of the system. However, this approach of physics-based detection requires knowledge and ability to model the physical behavior in the specific deployment environment, which might be difficult in some applications such as healthcare monitoring.

For anomaly detection in low density sensor networks, there are studies that explore techniques to detect anomalous data from the sensor measurements. In smart home sensor networks with heterogeneous sensors, [20] explored using principle component analysis (PCA) and canonical component analysis (CCA) to learn relationships between sensor data for fault detection by identifying anomalous data. Another approach taken by [25] employed machine learning techniques such as Support Vector Machine (SVM) to model a classifier for detecting anomalous events in sensor data and a linear regression model to learn to predict an event from sensor data. These works focus on detecting single anomalous data which are useful for fault detection but do not address the problem when more than one sensor readings are anomalous.

Few works cover detecting false data injections in a scenario where sensors collude, that is, they act in concert to provide a false representation of the state of the physical environment. The solutions proposed in these works exploit correlations between sensor measurements to detect false data from genuine ones and were able to identify sensors responsible for the anomalous events [14–16]. The work in [16] proposed a general framework using a combination of linear predictive model based on spatial correlations across homogeneous sensors and were able to detect false data injections with a collusion of more than half the sensors in a variety of WSN settings.

In [14]'s work, the authors successfully used Continuous Wavelet Transform (CWT) cross-scale analysis to distinguish between genuine and false data reported by sensors in a collusion attack setting applied on a wildfire data set. The approach was to use the parameter scale as the inter-sensor distance and the parameter translation as the centered position of the sensor. CWT on sensor readings for varying scales and translations provide measurements of similarity of a sensor data (centered sensor) relative to its neighbors. The low-scale coefficient characteristic was learnt from historical signals which were used for analysis on the test signal. This approach learns the spatial correlations at a fixed time, which was used for analysis to detect any deviations from expected behaviour. The approach of using correlations is useful in settings where the physical behavior is difficult to be explicitly modeled (i.e. how temperature varies and diffuse across space in a wildfire scenario) and/or specific to the deployment.

However, such an approach which uses only spatial correlations works well only in high-density sensor networks where a large number of deployed sensors are measuring the same physical parameter. In a low-density environment where the number of sensors in the deployment area is limited and non-homogeneous sensor measurements are collected, limited useful information can be extracted from spatial correlations alone. Moreover, there is an increasing adoption of deploying small numbers of sensors that measure heterogeneous parameters due to constraints in deployment environment and costs of deploying high-density sensor networks. Examples include body sensor networks for healthcare analytics and sensors for smart homes. Hence, there is a need to develop detection algorithms that would be able identify false data injections in low-density sensor environment with heterogeneous data.

1.2 Contributions

In a low density sensor setting, especially when sensor data is heterogeneous, the information redundancy is constrained and correlations in a single domain (i.e. fixing time and analyzing spatial correlation) might not yield much meaningful information. Hence, an alternative would be to **combine the analysis of correlations in multiple domains** such as *temporal-attribute* or *temporal-spatial*; a novel approach that has not been explored by existing literature. In this work we present a general framework that **exploits anomalies in temporal-attribute correlations** to detect false data injection in a low-sensor density environment with collusion in multiple heterogeneous sensors.

In order to systematically test the framework against possible attacks, we provided an adversarial model which was used to construct attack scenarios to evaluate its performance. The proposed **adversarial model utilizes a graphical method to represent the sensor network based on data correlation and generate attack strategies**. This guides the attacker's choice of subset of sensors to inject coherent false data with the goal of subverting detection through collusion. This would also solve the problem of lack of adversarial data sets by allowing us to construct adversarial data from genuine sensor data to test against the detection algorithm.

1.3 Organization of Paper

The rest of the paper is organized as follows. We first discuss the definition of sensor data and the various types of correlations in Section 2. Next, we describe the techniques and procedure used to learn correlations across the various sensor data in Section 3. In Section 4, we present our proposed detection framework for false data injection in low density sensor networks. Next, we described the adversarial model used to generate attack scenarios to test the detection framework in Section 5. We then described the experiment setup in Section 6 and present the analysis and evaluation of the results in Section 7. Finally, we provide the conclusion and discuss possible future works in Section 8.

2 SENSOR DATA MODELING

Sensor data refers to the measurements and readings of the sensed physical parameters in the deployed environment. The sensor measurement consists of the true value and an error term. Errors can be categorized into Systemic and Random. Systemic errors have non-zero mean and consistently shift the value away from the true value. These can be attributed to sensor faults or false data injections. Random errors are errors that fluctuates about the true value and can be attributed to noise and precision limitations of the sensing device. For *genuine sensor readings from dataset*, we assume no systemic error (i.e. no faults and false data injections) and model the sensor measurement (ϕ) with the true value (γ) and random error (ϵ) as follows:

$$\phi = \gamma + \epsilon \text{ , where } \epsilon \sim \mathcal{N}(0, \sigma^2) \tag{1}$$

The only observable quantity is the sensor measurements, and we are unable to observe the true values or errors. When systemic errors are introduced (either through faults or false data injections), the genuine sensor measurements are obscured. For a reliable sensor network, there is then a need to discern between genuine measurements and measurements with systemic errors. In order to detect false data by measurement inspections, there exist many anomalous data detection techniques such as outlier detection and statistical tests [12]. These techniques are unable to detect false data in the event of collusion, however, this could be achieved by exploiting relationships between observable sensor measurements.

2.1 Correlations between sensor data

Relationships between sensor measurements can be used to determine whether a particular sensor measurement is genuine. The relationship between sensor measurements are called *correlations* which can be derived on-line or modeled using historical data. In the presence of anomalous data, the correlations between measurements would be disrupted and through this observation we are able to identify anomalous sensor data. In the data collected by WSNs, the correlations of interest can be categorized into *temporal, spatial and attribute* correlations [24].

2.1.1 Temporal Correlation. Temporal correlations arise from the relationship of how a particular sensor's measurement changes with time. The changes of measurements with time are usually constrained by the physical laws that governs these changes. Measurements collected over a time-series that do not adhere to such constraint suggest that there are disruptions to the temporal correlations and potentially include maliciously injected data. Various change-point detection algorithms can be used to detect changes in time-series models, an example is the use of the CUSUM algorithm that detects variations from the expected mean of the data across time [4].

2.1.2 Spatial Correlation. Spatial correlations arise from the relationship between sensor measurements across space at a given point in time. At a fixed (interval) of time, the difference in sensor measurements due to an event's occurrence is the spatial correlation of the sensors. One approach proposed for the detection of false data injections in [13] assumes spatially homogeneous sensors and using an expected distribution of the spatial measurements to to detect anomalies.

2.1.3 Attribute Correlation. Attribute correlation explains how different types of sensor measurements are inter-dependent, modeling the relationships between different physical quantities in the sensed environment (e.g. temperature and humidity). Attribute correlations could be physical laws that explains how two physical quantities relate to each other and change according to a domain of analysis such as time or space. More complex attribute correlations between sensor measurements can arise from composite analysis of more than one correlation in the sensors' data.

3 LEARNING CORRELATIONS

A collection of sensor data over a period of time is a time-series signal. Analysis of time-series is a research area in which we can find various techniques to extract information [11]. In our case, we are interested in extracting relationships across various sensors that could possibly measure different physical quantities (heterogeneous sensor data) and we would like to extract correlations between two signals by performing pairwise signal cross-correlation. Similarities between signals can then be ascertained via cross-correlation functions and the result can be used to quantify the relationship between signals.

3.1 Cross Correlation Function

The cross correlation function (CCF) is defined as

$$R_{y,x}(k) = E[y(n)x(n+k)] = \sum_{n_1}^{n_2} y(n)x(n+k) \tag{2}$$

where $y(n)$ and $x(n)$ are signals and and k is the amount of time that signal $x(n)$ is delayed with respect to $y(n)$.

A more useful measure to work with is the normalized cross correlation function (NCCF) which is defined as

$$\rho_{y,x}(k) = \frac{C_{y,x}(k)}{\sigma_y \sigma_x} = C_{y,x}(k)\frac{1}{\sqrt{E(y)E(x)}} \tag{3}$$

where $\rho_{yx}(k)$ has a value between -1 and 1 for negatively correlated and positively correlated signals respectively and a value of 0 when the signals are uncorrelated [26].

3.2 Procedure for Cross-correlation Analysis

The following procedure outlines the steps required to estimate the cross-correlation function between two signals.

(1) **De-trending and Removal of Seasonality:** Signals are examined for trends and seasonality. The measurements in time series have seasonality removed and de-trended. Both seasonality and trends can be removed by differencing of measurements in the signal. For seasonality, by knowing the seasonality component (i.e. recurring cyclic period), the measurements can be differenced by the corresponding measurement of a previous period.

$$x'_p(t) = x_p(t) - x_{p-1}(t) \tag{4}$$

where p is the current period and $p-1$ is the previous period. For trend removal in datasets with non-stationary mean, we can perform first-order-differencing which is the difference between the values at time t and $t-1$:

$$x'(t) = x(t) - x(t-1) \tag{5}$$

(2) **Feature Normalization (for multi-variate signals):** For multi-variate signal analysis (i.e. analysis of time series with measurements of different measurements), we perform feature normalization to standardize the scales of the signals for analysis.

$$x'_t = \frac{x_t - \overline{x}}{\sigma} \tag{6}$$

where t is time, x' is the new value, x is the raw value , $\overline{x}$ is the signal mean and σ is the signal standard deviation.

(3) **Alignment:** Cross-correlations functions are computed for various lags (delay of one signal with respect to the other) and the dominant lag (if exists) that produces the greatest correlation should be used to align the signals.

(4) **De-noise:** Generate residual signals (de-noised), if necessary.

(5) **Cross-correlation:** Compute auto-correlation and cross-correlation functions.

(6) **Significance:** Test the magnitude of correlation functions.

In a multiple sensor setting, performing pair-wise cross-correlation analysis of N sensors, we would obtain an $N \times N$ matrix of correlation. The diagonal of the matrix provides the auto-correlation function.

3.3 Temporal Correlations

Pairwise cross-correlation is usually performed on the full signal to determine the correlations between the entire signal wave. However, our goal is to learn time variations of correlations between signals.

Our proposed approach to extract temporal variations in correlations therefore involves the following two methods:

(1) **Increasing Window Correlation:** Obtaining cross-correlations of signals from start to the current time. This will provide us with the information about long-time scale correlation changes.

(2) **Sliding Window Correlation:** Obtaining cross-correlations of signals within a fixed window time-frame that moves across time. This will provide us with information on short-time scale correlation changes.

The result of the above two methods is a **time-series of cross-correlation functions**, which is how the correlation between two signals changes with time. We are then able to achieve our goal to analyze correlations in the temporal domain.

4 FALSE DATA INJECTION DETECTION FRAMEWORK

In low-density sensor networks with heterogeneous data, we are constrained by data density in the spatial and and attribute domains at any single time snapshot. The use of correlations between sensor measurements provides a form of information redundancy that we can exploit. With limited data that are heterogeneous at any fixed time, we propose combining temporal correlations and attribute correlations as a time-series of changing attribute information to increase the data density for analysis.

The proposed detection framework ***exploits disruptions in temporal-attribute correlations*** when an attacker manipulates the the sensor measurements in the subset of sensors under his/her control. The detection is based on identifying changes in correlations across various different sensor data types over a time period.

The framework (Fig. 1) consists of a learning phase and the testing phase. During the learning phase, we assume that the sensor measurements are genuine and faultless, where we learn the signal statistics (signal mean and variance) and signal correlations under this "normal state". The learnt correlation metrics are then used to build a distribution to perform statistical anomaly detection. With our built detection algorithm from the learnt distribution,

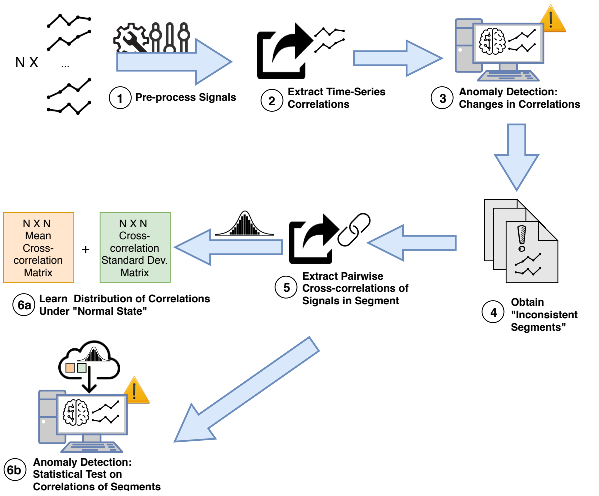

N X
...
1 Pre-process Signals
2 Extract Time-Series Correlations
3 Anomaly Detection: Changes in Correlations
N X N Mean Cross-correlation Matrix
+
N X N Cross-correlation Standard Dev. Matrix
6a Learn Distribution of Correlations Under "Normal State"
5 Extract Pairwise Cross-correlations of Signals in Segment
4 Obtain "Inconsistent Segments"
6b Anomaly Detection: Statistical Test on Correlations of Segments

Figure 1: Workflow of Proposed Detection Algorithm

the anomalous segments identified are tested against the statistical anomaly detection algorithm and each of the sensors votes on the degree of abnormality of the correlations within the time segment. The votes are then aggregated to a final anomaly score for classification of the segment. In the following sections, we explain the implementation of the various components in the proposed detection framework.

4.1 Pre-processing Signals

Before extracting time-series cross-correlations between the signals, there is need for a pre-processing step for the following reasons:

(1) **Heterogeneous Data:** The sensors measures different physical parameters of the environment and hence, have signals of different scales in magnitude and sampling frequency. Therefore, there is a need to standardise the signals to uniform magnitude and time scale.

(2) **Trends and Seasonality:** The presence of trends and seasonality results in non-stationary signals which may introduce non-linearity. Consequently, the true signal cross-correlations might not be obtained as the cross-correlation function we adopt assumes linear systems. We employ differencing methods on the time-series sensor measurements to remove trends and seasonality.

(3) **Noise:** The presence of noise in the signals would corrupt the information we would want to extract as the disturbances introduced alter the true signal to varying extents. Noise in signals can be reduced through application of filters and de-noising techniques.

4.2 Extracting Time-series Correlations

We are not only interested in how the full signal waveforms are correlated, but also in the temporal characteristics such as how the correlations between signals changes with time. With a time-series of correlations, we are able to detect temporal changes in cross-correlations that suggests a possibility of an attack. In order to extract the temporal changes in correlations, we perform both

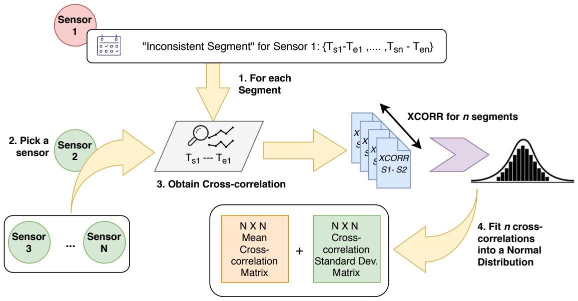

Sensor 1
"Inconsistent Segment" for Sensor 1: {T_{s1}-T_{e1} ,.... ,T_{sn} - T_{en}}
1. For each Segment
2. Pick a sensor
Sensor 2
T_{s1} --- T_{e1}
3. Obtain Cross-correlation
XCORR for *n* segments
XCORR S1- S2
Sensor 3
...
Sensor N
N X N Mean Cross-correlation Matrix
+
N X N Cross-correlation Standard Dev. Matrix
4. Fit *n* cross-correlations into a Normal Distribution

Figure 3: Learning signal cross-correlations distribution under genuine sensor measurements

sliding window and increasing window sampling to extract short-term and long-term correlation changes in the signal respectively.

With every iteration of the increasing-window or sliding-window correlation extraction, we obtain an $N{\times}N$ matrix of cross-correlations. At the end of iterating through the training set, we will get T $(N{\times}N)$ matrices, where T is the length of the training set and N is the number of sensors in the network. We then construct time-series of each element in the $N \times N$ matrix from $t = 0$ to $t = T$. (See Fig. 2)

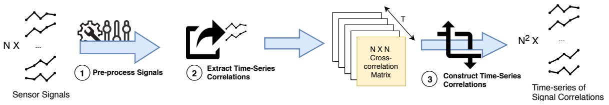

N X
...
Sensor Signals
1 Pre-process Signals
2 Extract Time-Series Correlations
N X N Cross-correlation Matrix
T
3 Construct Time-Series Correlations
N^2 X
...
Time-series of Signal Correlations

Figure 2: Extracting Time-series of signal cross-correlations

4.3 Variations in Temporal Correlations

After obtaining time-series of signals cross-correlations, we use a change-point detection algorithm on these time-series to identify abrupt changes in the cross-correlations, indicating a possibility of an attack. Change point detection allows us to identify when the distribution of the correlation changes. They can be tuned to trade off accuracy for sensitivity in identification of the change point.

We adopted the use of CUSUM algorithm to detect changes in the time-series cross-correlations obtained as it was found to produce best trade-off between accuracy and computational complexity [3], and is also widely adopted in attack detection algorithms. The change-points detected for each cross-correlated signal were then sequentially picked out in pairs to form time segments with inconsistent correlations for further analysis (we call these segments ***"inconsistent segments"***). A **minimum run-length** for the time segment was set to prevent extremely short segments that would result in high uncertainty in the cross-correlation obtained. The minimum run-length is a parameter to be tuned to a desired sensitivity.

This procedure of extracting "inconsistent segments" was applied to both training and testing phases. For training phase, these segments were used to learn variations in correlations under "normal state". During testing phase, the segments were used for anomaly detection to identify attacks.

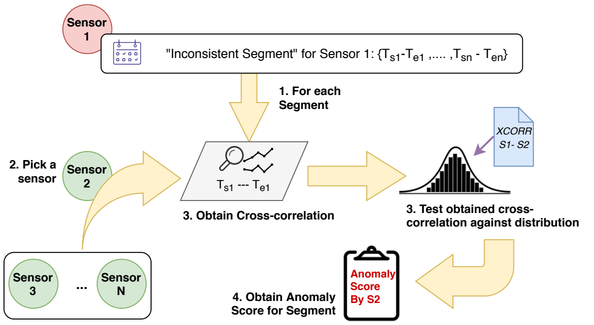

Sensor 1
"Inconsistent Segment" for Sensor 1: {T_{s1}-T_{e1} ,.... ,T_{sn} - T_{en}}
1. For each Segment
2. Pick a sensor
Sensor 2
T_{s1} --- T_{e1}
3. Obtain Cross-correlation
XCORR S1- S2
3. Test obtained cross-correlation against distribution
Sensor 3
...
Sensor N
4. Obtain Anomaly Score for Segment
Anomaly Score By S2

Figure 4: Obtaining anomaly score for each "inconsistent segment" identified for every sensor signal

4.4 Learning "Normal" Temporal Correlations

We use the set of "inconsistent segments" extracted from the training data set (sensor measurements free from malicious data) to learn the natural distribution of variations in the cross-correlations of sensor measurements under genuine conditions (Fig. 3). These variations of pairwise cross-correlation are assumed to be independent and identically distributed, which is reasonable as these variations in correlations are expected to be caused by external independent events. Therefore, we can fit correlation distribution of $C_{n,n}$ from the "inconsistent segments" into a Normal Distribution.

$$C_{n,n} \sim \mathcal{N}(\mu_{n,n}, \sigma^2_{n,n}) \quad \forall n \in \{1, 2, ..., N\}$$

The set of normal distributions are characterized by the mean $\mu_{n,n}$ and standard deviation $\sigma^2_{n,n}$ which we can store in an $(N \times N)$ matrix each. The normal distribution allows us to perform statistical outlier anomaly detection by testing observations against the distribution to obtain an *anomaly score* (i.e. Probability of Anomaly).

4.5 Testing Phase - Anomaly Detection

During testing, after identifying "inconsistent segments" for each sensor, we proceed to extract pairwise cross-correlations with all the other sensors. A statistical outlier test by calculating the Z-score of the observed correlation using the Cumulative Distribution Function to obtain an *anomaly score* (Fig. 4).

4.5.1 Voting on Anomaly Score. For each time segment, there would be $N - 1$ anomaly scores (pair-wise cross-correlation , $AS_{n,i}(t_s, t_e)$) reported by each of the other sensor on the extent of observed correlation being an outlier from learnt correlation distribution.

The final anomaly scores attributed to the particular time segment can be obtained by aggregating the individual anomaly scores by majority voting. The following weighting scheme provides equal weighting to all anomaly scores obtained for sensor n between time t_s and t_e :

$$AS_n(t_s, t_e) = \sum_{i}^{N} \frac{AS_{n,i}(t_s, t_e)}{N - 1} \quad \forall i \in \{1, 2, ..., N\}, i \neq n \qquad (7)$$

Similar to the work in [16], where majority voting was used to aggregate sensor value prediction for comparison against reported sensor measurement to detect anomaly, we used majority voting to aggregate anomaly scores instead. Aggregation of anomaly scores would allow us to use information from all sensors in the network to rate the authenticity of the sensor data based on their correlation.

4.5.2 Temporal Anomaly Score. With a time period associated with the aggregated anomaly score, we are able to rationalize how the anomaly score changes with time by building a time-series of anomaly score for each sensor. For overlapping segments, the anomaly score for the overlap region is averaged. Having a time-series of anomaly scores provides resolution in time on the variation of anomaly score which would be helpful in determining the time of attack (if any).

4.5.3 Decision Threshold. A decision threshold value was used to trigger an alert for anomalous data when the time-variation of anomaly score for a particular sensor crosses a threshold value for more than a continuous run-length of n time samples, where n is the *minimal run-length of extracted segments*.

The decision threshold value is obtained by taking the anomaly score that separates the top $x\%$ of anomaly score obtained from running the anomaly detection on the training set sensor measurements (where x is a tuning parameter for the model). Since the training set data is assumed to be free from malicious data, the anomaly scores obtained from the data is the result of model uncertainty. Any anomaly score value that is greater than $(100-x)\%$ of the anomaly score from genuine data would be used as a sentencing criteria for detection trigger. This method of setting a threshold would be more robust to noise from the learnt correlation as compared to a threshold set at 50%, typically used for majority voting.

4.6 Summary

In all, the proposed detection algorithm first extract pairwise temporal cross-correlations over all the sensors. It then learns the distribution of the variations of correlations under the "normal state" without attack. During testing, the segments with inconsistent correlations are similarly extracted and inspected against all the learnt pairwise correlation distribution to obtain anomaly scores. The anomaly scores are then aggregated using majority voting, the final score that crosses above the threshold value indicates an attack.

5 ADVERSARIAL MODELING

Given the lack of adversarial data sets and the need to test anomaly detection algorithms, an adversarial model was proposed to construct attack strategies. The attack strategies were subsequently used to generate adversarial data set from the genuine sensor data, which allowed us to test the detection algorithm and evaluate its performance against attacks of varying complexity.

The goal of an attacker in the context of false data injections is to elicit or mask events without triggering detection. Sensor data used for ***event detection*** can be manipulated by the attacker in order to either ***spoof events*** or ***mask events***. These attacks can be easily achieved without any anomaly detection algorithm, however in the presence of an anomaly detection, **collusion attacks** where more than one sensor measurements are tampered with to report coordinated and coherent measurements can subvert these anomaly detection algorithms.

5.1 Assumptions on the attacker

Our assumptions for the attacker's resources and knowledge follow closely with those described by Illiano [12]. The assumptions made are the following:

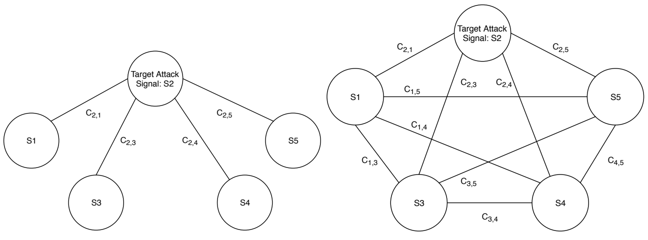
1. **No time constraints to conduct attacks.**
   Having time constraints on attacks could possibly have adverse impact on the quality of attacks conducted. Complex attack strategies (such as those that exploit historical data or requires chaining of multiple software / infrastructure vulnerabilities) usually takes time, having time constraints would severely affect an attacker's capabilities to conduct attacks. As such, we assume that the attacker has no time constraints in strategising and conducting an effective attack that maximizes the damage while staying undetected.
2. **Complete knowledge and access to historical measurements of all sensors.**
   Anomaly detection algorithms usually distinguish false data from genuine ones by comparing the data against past behaviour of the system and its coherence with the other sensor measurements. The assumption of complete access to all historical measurements means that the attacker would have access to the same data as the anomaly detection algorithms. With this assumption, the attackers are not disadvantaged and would be able to build attack signals that are coherent where reported measurements support each other in a bid to fool the anomaly detection algorithms.
3. **Has control of a subset of sensors in the network**
   The assumption that the attacker has full control of a subset of sensors of his/her choice according to the attacker's strategy and is able to control the reported measurements of these sensors. This means that the attacker has the ability to conduct complex attacks across multiple sensors to ***collude*** against the detection algorithm by reporting coherent sensor measurements.

In all, the assumptions made about the attacker would provide us with the worst-case attacker scenario where the attacker has full capabilities and resources to conduct complex false data injection attacks in any subset of sensors in network to subvert detection.

5.2 Multi-sensor Attacks

An attack on a single sensor would be easy to detect as it would not be reporting measurements that are coherent with other sensors. The discrepancies between the reported measurement by a compromised sensor with other sensor measurements would be more apparent with the presence of strongly correlated signals. In order for the attacker to conduct a more convincing attack, signals that are correlated should have coordinated changes and report coherent values accordingly to their strength of correlation.

With access to historical and current sensor measurements, the attacker would be able to obtain knowledge of the sensor behaviors which would be helpful to orchestrate an attack where sensors report coherent measurements that aims to subvert the detection algorithm. The knowledge can be used to construct attack signals off-line, and the prepared sensor measurements were then used to replace online genuine sensor measurements.



(a) Naive Attacker (b) Full Cross-Correlations

Figure 5: Naive Attacker's Assumption of Signals Cross-correlations

5.3 Learning Signal Cross-correlations

With historical sensor measurements of all the sensors in the network, the attacker is able to obtain information of how various sensor measurements correlate with each other over time. The attacker learns the *Normalized Cross-correlation Function* of the sensor historical data which provides a useful measure of relative signal correlation strength in order to construct correlated attack signals for collusion attacks.

5.4 Signal Construction

Correlated signals x and y can be constructed using learnt cross-correlation from the parent node (signal y) by reversing the order of normalizing, de-trending and/or seaonality removal :

$$x_n(t) = y_n(t) \times C_{y,x} \quad \text{to construct normalized signal} \tag{8}$$

$$x'(t) = x_n(t) \times \sigma + \overline{x} \quad \text{to reverse normalization} \tag{9}$$

$$x(t) = x'(t) + x'(t-1) \quad \text{to reverse first-order differencing} \tag{10}$$

$$x_p(t) = x'_p(t) + x'_{p-1}(t) \quad \text{to reverse trend removal} \tag{11}$$

With a target attack signal (complete signal to achieve attacker's goal), the attacker is able to use the above procedure to construct a suite of attack signals in the sensor network that correlates with the target signal.

5.5 Naive Correlated Attack

For collusion attacks, the attacker has to a chose a subset of sensors to manipulate in order to spoof coherent sensor measurements with the target sensor. The choice of subset of signals to spoof is the strategy of the attacker and there are various approaches that the attacker can use to guide its choice.

5.5.1 Naive Assumption. This approach assumes that all other signals are correlated to the target signal and are independent from one another, disregarding any inter-dependencies between the non-target signals (Figure 5(a)). The attacker would focus on construction of attack sensor measurements based solely on the sensor's correlation strength to the target sensor, neglecting any other cross-sensor correlations.

5.5.2 Using Naive Assumption to Guide Attack Strategy. A possible strategy the attacker can employ is to choose accomplice sensor(s) in priority of descending pairwise cross-correlation strengths and manipulate the subset of sensor measurements to construct a suite of coherent attack signals.

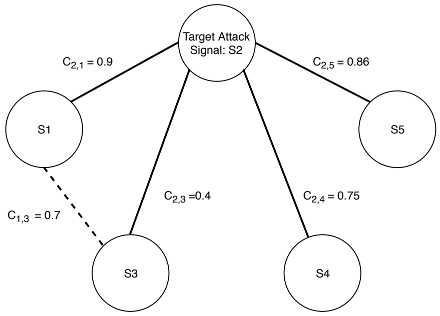


Figure 6: Example of Naive Assumption

	S2 (attack Signal)	S1	S3	S4	S5
Naive	1	0.9	$1 * 0.4 = 0.4$	0.75	0.86
Naive with $C_{1,3}$ **ignoring** $C_{2,3}$	1	0.9	$(1 * 0.9) * 0.7 = 0.63$	0.75	0.86

Table 1: Example of updating sensor values based on correlations considered

5.5.3 Disadvantages of Naive Approach. Cross-correlation functions provides a measure of the linear relationship between two signals. The naive approach describes how the target signal is related to the other signals in the network. The naive approach can be extended to all the signals in the network to obtain a *complete graph* that provides the inter-dependencies of all signals in the network (Fig. 5(b)).

The naive approach approximation of the signal relationship in a network works well when all the correlations to the target signal are the maximal for the associated sensor. However, when the correlations to the target signal are dominated by correlations between other sensors, this approach to approximate the relationship fails.

As we are considering *normalized cross-correlations*, the values of $C_{x,y}$ are bounded such that $-1 \leq C_{x,y} \leq 1$. Thus as we travel along any path starting from any vertice, the magnitude of the propagated correlation can only remain the same or decrease. Therefore, it would be theoretically ideal to use the naive attacker assumption if all correlations with the target signal are of comparable magnitude as any other paths from the would decrease the correlation.

In the toy example above (Fig. 6, Table 1), we choose the two accomplice sensors to be $S1$ and $S3$. The correlation between target signal S2 and signal S3 is low with value $C_{2,3} = 0.4$ and hence, the update to the value of S3 will be proportionally low. The Naive Attacker Assumption does not take into account dependencies between other signals, disregarding the correlation $C_{1,3}$. As shown, we would expect the updated value of $S3$ from path $S2 - S1 - S3$ to be larger than an update directly from $S2$ via $S2 - S3$. This could possibly be detected as an anomaly in the correlation between signals $S1$ and $S3$.

In all, the Naive Assumption may work well under some circumstances, but to devise a more effective attack strategy, the attacker would require a better representation of the signal correlations.

5.6 Using MWST to Orchestrate Attacks

The modeling of linear relationships can also be extended to other nodes in the network, resulting in a complete graph where edges are weighted by their pair-wise signal cross-correlations.

While we strive to use all the inter-signal dependencies, considering all the signal cross-correlations (Fig. 5(b)) results in a complete graph and any changes we decide to perform on the target attack node and corresponding updates to the other nodes would result in loopy propagation due to cyclic paths. Some of the problems in loopy propagation are: 1) Failure to converge or convergence errors 2) Cycling errors where new information is treated as old information [5]. While there are techniques for exact inference from loopy propagation, these are usually computationally expensive.

Hence, we propose to use a **Maximally Weighted Spanning Tree (MWST)** to approximate the signal dependencies in terms of the strength of their cross-correlations (Fig. 7). A maximally weighted spanning tree is a subgraph in the complete graph where minimal number of edges are used to connect the edges and the sum of the total edge-weights is maximal. The use of MWST is a good trade-off between model accuracy and computational efficiency as the tree structure averts the problem of message propagation found in cyclic graphs and provides a model that maximizes the total correlations strength in the network.

Constructing the maximally weighted spanning tree is a greedy algorithm (Prim and Kruskal [19, 23]) that ensure that vertices are connected such that the sum of the total correlations of in the network is the maximal. The MWST can be used to guide the attacker on which nodes to compromise and the order with which the nodes should be updated accordingly against the reference sensor node's measurements. We can derive a hierarchical structure when the attack target is chosen as the root node of tree, and traversing down the tree provides a sequence to update the children nodes. The tree structure can also be utilized by the attacker to perform attack cost analysis, where the cost is the number of sensor nodes to compromise. With the MWST and the attack signal chosen as the root node, the updating of the values is simply traversing down the tree and updating the nodes based on the edge values (i.e. cross-correlation strength). The attacker is able to construct a full set of attack signals where the target attack signal is first constructed and the other signals are sequentially updated.

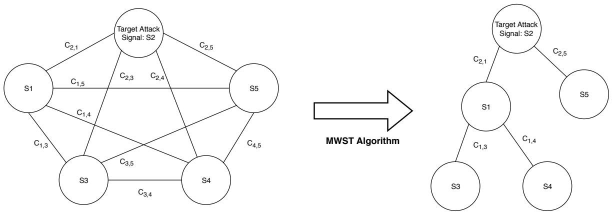


Figure 7: Full Cross-correlation to a MWST

5.7 Using MWST for Optimising Attacks

With the tree-structure, the attacker is able to visualise the relationship of signals in terms of strength of cross-correlations. There exists two extreme cases when:

(1) **All signals are not correlated.** This would result in a spanning forest where all the nodes in the graph are not connected. For an attacker, this is the ideal case where changes in any single signal would not have any correlated effects.
(2) **All signals are equally correlated.** For the assumed worst-case attacker where the attacker is able to control the full set of sensor signals, this would not pose a problem. However, if the attacker has constraints on the subset of sensor he/she controls, the attacker would not be able to build a complete set of coherent signals and risk detection.

5.7.1 *Optimising Cost of Attack: Lowest Cost of Attack.*

The MWST of sensor cross-correlation strengths can be used by attackers to determine vulnerable sensor nodes. We define vulnerable sensor nodes as sensors whose signal is weakly correlated to the other rest of the sensor signals, relative to the sensitivity of the correlation detection algorithm (i.e. the correlations of the vulnerable sensor nodes with the other sensors are too weak to provide useful information to the detection algorithm).

The following is a proposed procedure to search for the minimal subset of sensors to compromise to minimize cost:

(1) Begin from the singly connected node with the smallest edge weight (root node: target sensor whose attack signal will be the reference for accomplice signals)
(2) Traverse down the tree along the path of largest edge weights. This will ensure that we use sensor measurements that are the most correlated.
(3) Terminate when the detection algorithm fails to detect the attacks.
(4) The nodes visited will form the subset of nodes to compromise that minimizes attack cost based on the MWST.

5.7.2 *Optimising Cost of Attack Cost: Tree Pruning.*

An attacker can also consider removing edges from MWST for the following scenarios: 1) the original tree is large (in high-density sensor environment) and becomes computationally expensive to update all nodes or 2) attacker does not have full control over the network and wishes to exclude the uncontrolled nodes. The result will be a spanning forest with trees that are not connected to the root node (chosen target attack signal) and nodes in these trees would not be updated. However, the attacker has to consider the detection risk trade-off when pruning the tree.

6 EXPERIMENTAL SETUP

We applied the detection framework on the PhysioNet MIMIC 221n (Medical Information Mart for Intensive Care) Dataset, an intensive care clinical dataset which contains various vital-signs measurements of patients [22]. This dataset was chosen for its multivariate parameters and low sensor density limitations to evaluate the effectiveness of our proposed framework under such settings.

6.1 Event Alarms

Threshold alarm values were referenced from medical literature [8, 17, 28] research in healthcare monitoring for patients in ICUs. The threshold alarm values used for the respective health parameters are summarized in Table 2.

Sensor Parameter	Lower Threshold	Upper Threshold
ABP_{mean} (ABPm)	195 mmHg	60 mmHg
$ABP_{systolic}$ (ABPs)	195 mmHg	90 mmHg
$ABP_{diastolic}$ (ABPd)	110 mmHg	0 mmHg
$HeartRate$ (HR)	120 Bpm	50 Bpm
$PulseRate$ (PR)	120 Bpm	50 Bpm
$RespiratoryRate$ (RESP)	28 Bpm	8 Bpm
$Oxygenation$ (SPO2)	100 %	90 %

Table 2: Threshold Alarm for Parameters in Healthcare Monitoring Dataset. *ABP is "Arterial Blood Pressure", Bpm for heart rate and pulse rate is "Beats per minute" and Bpm for respiratory rate is "Breaths per minute"*

6.2 Model Parameters

The model parameters used for the Healthcare Monitoring Dataset were summarized in the Table 3. These parameters were obtained by manual grid search to achieve the required detection sensitivity.

Parameters	Value
Sliding Window Size	35s
CUSUM Window for sliding window correlations	30s
CUSUM Window for increasing window correlations	15s
Minimum Run Length	20s
Decision Threshold (x)	10%

Table 3: Model Parameters for detection algorithm used on Healthcare Monitoring Dataset

7 EVALUATION AND ANALYSIS

The MWST for the dataset was constructed using the attacker's learnt signal cross-correlations which were subsequently used to devise attack strategies to perform evaluation and analysis of the proposed detection framework.

From the MWST constructed (Fig. 8), we identified that the sensor that measures the parameter ABPd having the most children, suggesting that it would be the most difficult to attack. Hence, we performed our analysis of the detection framework by conducting various attack strategies with ABPd as the target signal to spoof. In addition, we also performed the ***lowest cost of attack*** analysis and conducted attacks on sensor nodes that are vulnerable. We observed that there are 3 singly connected nodes in the MWST namely: ABPs, RESP and PR. However, the edge weights for ABPs and RESP are relatively high in the context of the whole network, hence we would conduct attacks on HR to determine the smallest subset of nodes to compromise.

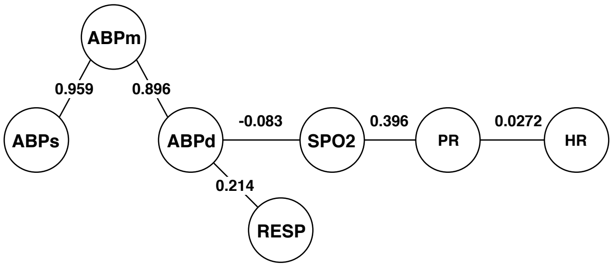


Figure 8: MWST for correlations of sensors

7.1 Attack on Single Sensor : ABPd

As the sensor that measures ABPd was identified using the MWST as the most robust to attacks, we have chosen to conduct analysis of our detection framework with various attack strategies (Single Sensor and Collusion Attacks). The performance of our detection framework would be evaluated based on how well it is able to detect various complexity of attacks by systematically increasing the number of accomplice sensors that spoof coherent measurements.

We first investigated how the detection algorithm performs under the scenario where only the target sensor measurements (measurements of ABPd) are manipulated. We perform a linear slow increase of sensor measurements to exceed the threshold of 110 mmHg from $t = 400s$ (boiling frog attack [6]). Figure 9 shows the sensor measurements of the patient's vital signs for $2000s$ and the attack signal is in the first row where measurements were manipulated to cross the upper threshold values. All other signals are genuine sensor measurements.

The proposed detection algorithm was able to accurately detect malicious data injection on a single sensor without collusion (Fig. 10; row 1 anomaly score crosses threshold). A separate experiment using the CUSUM algorithm was performed to detect abrupt changes in the sensor measurements, however, no anomalies were detected as the manipulated sensor measurement was incremented slowly to subvert detection.

We can conclude that the proposed detection algorithm is minimally able to detect false data injection in a single sensor without collusion in an attack scenario where the attacker avoids anomaly based detection. In addition it showed that the proposed detection algorithm is superior compared to the CUSUM anomaly detection method that performs sensor measurements inspection.

7.2 Collusion Attack: ABPd & ABPm

From the MWST (Fig. 8), with ABPd as the root node, the child node that has the highest cross-correlation is ABPm, suggesting that performing an attack on this sensor would be the next logical step as a more strongly correlated signal would be expected to behave more similarly to the target signal. If a weakly correlated signal is chosen to be the accomplice, other strongly correlated signals would be able to detect the anomaly. With ABPm chosen as an accomplice signal, the attack signal for ABPm was constructed using the attacker's learnt cross-correlation and the proposed procedure. Figure 11 shows the sensor signals with the signals manipulated in rows 1 and 2 respectively for ABPd and ABPm.

The detection algorithm managed to detect the attacks on both ABPd and ABPm, although the detection time for both sensors was longer and the attack period detected was considerably shorter than the actual attack. We noticed that the time-series anomaly scores during the period of attack were threading close to the threshold but seldom crossed it to raise alarms. This could possibly be due to the robustness of the learnt correlations and sensitivity of the detection algorithm (threshold level). In addition to the two manipulated signals, the detection algorithm also raised alarms for anomalies in ABPs, HR and RESP. The false positives can be attributed to the effect of collusion during the voting. Every sensor would have 6 sensors voting on the anomaly score, thus with 2 compromised sensors, the voting would be skewed.

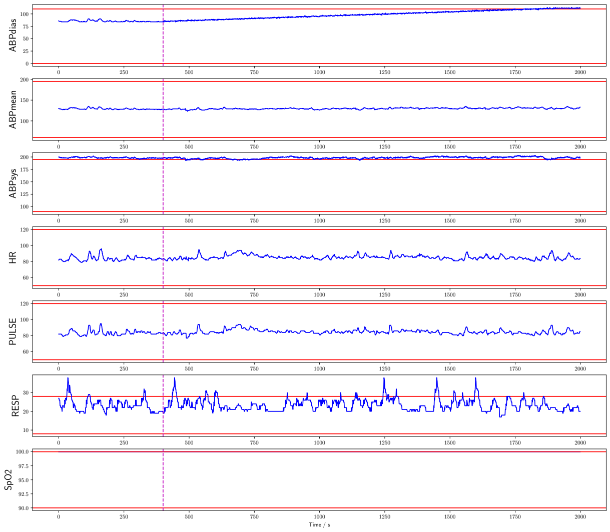


Figure 9: Sensor Measurements with malicious data injected for sensor ABPd (row 1) to increase sensor measurements above its upper threshold level marked by the red line. Attack start time ($t = 400s$) is marked by magenta vertical dotted line.

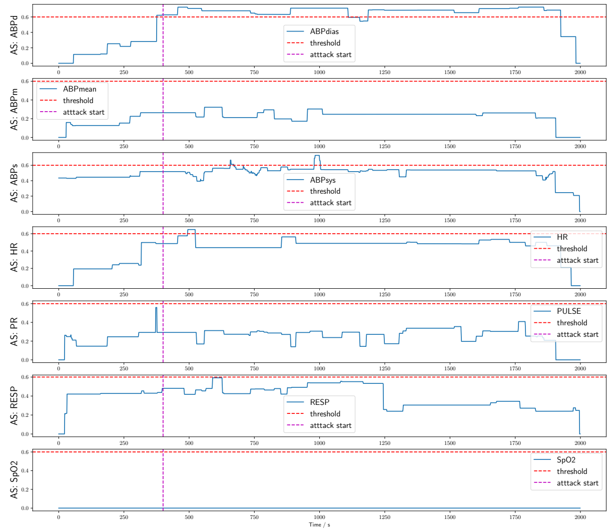


Figure 10: Anomaly score (AS) of ABPd (row 1) is above the threshold value after the start of attack, indicating that the attack was detected. AS crossed the threshold slightly before attack as it was a look-ahead sliding window.

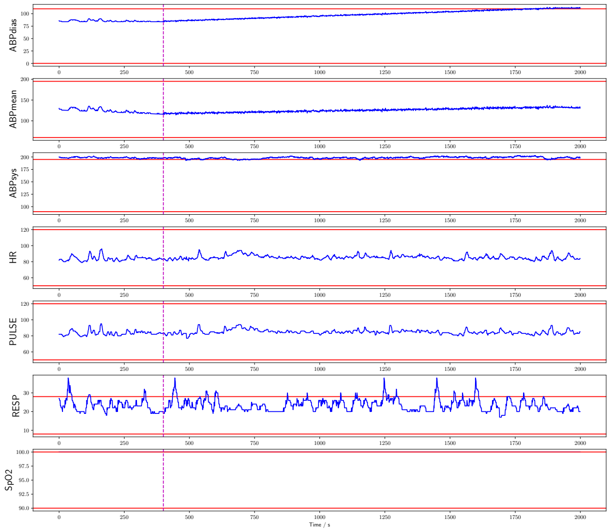


Figure 11: Sensor Measurements with malicious data injected for sensor ABPd (row 1) to increase measurements to cross the upper threshold and ABPm (row 2) measurements updated to reflect its correlation to ABPd. Attack start time ($t = 400s$) is marked by magenta vertical dotted line.

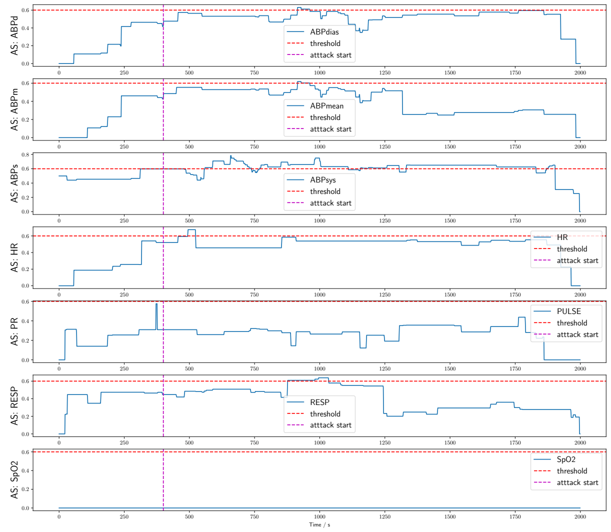


Figure 12: Anomaly scores of ABPd and ABPm (rows 1 and 2) both crosses the threshold at around $t = 900s$

7.3 Collusion Attacks : ABPd, ABPm & ABPs

With the attacker's failure to subvert detection with an additional compromised sensor, the MWST was used to determine the next node to compromise. By traversing down the tree and choosing the next node with the highest edge weight as the next accomplice sensor to work in concert and fool the detection algorithm. The three sensors used in this attack were: ABPd, ABPm and ABPs. It was found that with two additional compromised sensors providing supporting spoofed measurements, a total of 3 out of 7 sensors compromised was sufficient to spoof an event while subverting detection from our proposed algorithm. This result was expected as under equal weighting for majority vote where the correlations of signals are perfect (no noise in signals), the ideal majority of 50% (i.e. 3.5 sensors out of 6) would be required to "flip" the classification. This highlights one of the shortcomings of the approach of using majority voting as an aggregation technique. The detection performance was comparable to that of [16]'s work on the same data set.

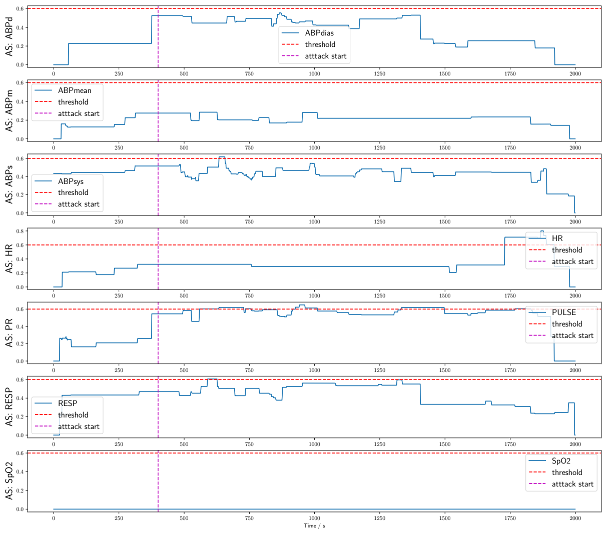


Figure 13: Anomaly Score (AS) for attack on single sensor node: HR (row 4). AS crosses the threshold at $t = 1700s$, indicating a delayed detection. Attack start time ($t = 400s$) is marked by magenta vertical dotted line.

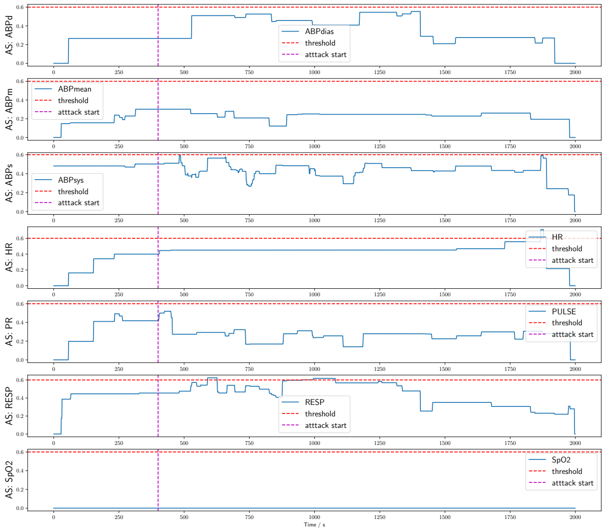


Figure 14: Anomaly Score (AS) for attack on sensor nodes: HR & PR (rows 3 & 4). AS did not cross threshold value for a time longer than Minimum Run Length, indicating that the detection algorithm failed to detect the attacks. Attack start time ($t = 400s$) is marked by magenta vertical dotted line.

7.4 Lowest Cost of Attack

With the sensor node HR identified as the node with the smallest correlation strength, a single sensor attack was conducted by manipulating the sensor measurements to spoof an event by crossing the upper threshold.

It was observed that attacking the single node HR (row 4 in Fig. 13) was insufficient to subvert detection, although the detection time was severely delayed, crossing the threshold at ($t = 1700s$). The detection algorithm identified that the reported sensor measurements of Pulse Rate were anomalous, possibly due to disagreements of sensors (with 1 out of 6 sensors compromised) during the voting for aggregated anomaly score.

With the failure to subvert detection by just attacking a single sensor node, we proceed to traverse down the tree on the path of the largest edge weight (maximal correlation strength). With HR as the root node, it only has one child (PR) which was used as the accomplice sensor node for the next iteration of attack. It was observed from the temporal anomaly scores (Fig. 14, rows 3 and 4 were below the threshold red dotted line), the detection algorithm failed to detect attacks on both HR and PR sensors.

With only one compromised sensor, the attacker was unable to hide from detection. Using the proposed graphical procedure to determine the lowest cost of attack, the next additional sensor node to compromise was identified. The use of the lowest cost of attack procedure based on the graphical method was effective as only a single extra sensor node was required to be compromised as compared to the theoretical value of 2 more nodes to achieve a majority to spoof the detection algorithm.

7.5 Performance Evaluation

A study of the performance of the preliminary detection model obtained by manual search of model parameters yielded reasonable results with fairly high false positive rates. Future work could be done to tune the model, which we believe would improve the performance.

In all, the detection framework was relatively effective in detecting collusion attacks with the best performance in a single sensor scenario (no collusion) and with increasing number of colluding sensors, we observe degrading performance (i.e. accuracy in detection and identifying attack time).

8 CONCLUSION AND FUTURE WORK

Detecting false data injections in low-density WSNs is essential for ensuring data integrity, especially with its increasing adoption in many critical applications such as healthcare and infrastructure monitoring. In low-density environments with heterogeneous sensor measurements, detection is a challenge due to the limitations in useful information that can be extracted. Moreover, presence of collusion where attacker injects coherent sensor measurements in more than one sensor exacerbates the problem.

We have proposed a detection framework that exploits anomalies in temporal-attribute correlations between sensor measurements to detect false data. This approach is effective in identifying attacks where more than one sensor colludes to report coherent measurements. However, with increasing number of colluding sensors, the detection performance degrades and is limited to half the number of total sensors due to the use of majority voting to aggregate anomaly scores. With majority of the sensors colluding, the detection fails as the sensor measurements which are spoofed are the majority and would be classified as genuine data. Hence, there is merit to further research in collusion-tolerant anomaly score aggregation. Future work on the detection framework could also look into optimizing the framework by exploring use of other anomaly detection techniques (i.e. clustering-based anomaly detection) as well as applying this framework on other combinations of correlation domains (i.e. temporal-spatial correlations) for other deployment settings.

With a need to test the proposed detection algorithm and a lack of adversarial data, we presented an adversarial model to devise strategies and generate attack data. The adversarial model utilizes a graphical based representation to devise attack strategies that guide the attacker's choice of subset of sensors to compromise for collusion against the detection algorithm. A maximally weighted spanning tree was constructed from pairwise cross-correlation strengths across all the sensor measurements. A procedure to determine the lowest cost of attack, which is the smallest subset of sensors to compromise to subvert detection by collusion, was proposed and tested. The procedure produced a strategy that required only a single additional sensor to be compromised, which was lesser than the maximal number of sensors to required be compromised. The maximum number of sensors to compromised was derived from the sensor identified using the graphical model as the most difficult to spoof (therefore requiring the most number of colluding sensors). In all, the adversarial model was effective in producing targeted strategies to build collusion attacks based on sensor data correlations, it also proves to be a useful in performing threat assessment for a sensor network in context of correlation-based data integrity attacks.

REFERENCES

[1] Sridhar Adepu and Aditya Mathur. 2016. Using Process Invariants to Detect Cyber Attacks on a Water Treatment System. In *ICT Systems Security and Privacy Protection*, Jaap-Henk Hoepman and Stefan Katzenbeisser (Eds.). Springer International Publishing, Cham, 91–104.

[2] Riham AlTawy and Amr M Youssef. 2016. Security tradeoffs in cyber physical systems: A case study survey on implantable medical devices. *IEEE Access* 4 (2016), 959–979.

[3] Samaneh Aminikhanghahi and Diane J Cook. 2017. A survey of methods for time series change point detection. *Knowledge and information systems* 51, 2 (2017), 339–367.

[4] Michle Basseville, Igor V Nikiforov, et al. 1993. *Detection of abrupt changes: theory and application*. Vol. 104. Prentice Hall Englewood Cliffs.

[5] Janneke H Bolt and Linda C van der Gaag. 2004. The convergence error in loopy propagation. In *International Conference on Advances in Intelligent Systems: Theory and Applications*.

[6] Eric Chan-Tin, Daniel Feldman, Nicholas Hopper, and Yongdae Kim. 2009. The Frog-Boiling Attack: Limitations of Anomaly Detection for Secure Network Coordinate Systems. In *Security and Privacy in Communication Networks*, Yan Chen, Tassos D. Dimitriou, and Jianying Zhou (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 448–458.

[7] United States Nuclear Regulation Commision. 2018. Backgrounder on the Three Mile Island Accident. https://www.nrc.gov/reading-rm/doc-collections/fact-sheets/3mile-isle.html

[8] Kate Fagan, Allison Sabel, Philip S Mehler, and Thomas D MacKenzie. 2012. Vital sign abnormalities, rapid response, and adverse outcomes in hospitalized patients. *American Journal of Medical Quality* 27, 6 (2012), 480–486.

[9] A. Faquih, P. Kadam, and Z. Saquib. 2015. Cryptographic techniques for wireless sensor networks: A survey. In *2015 IEEE Bombay Section Symposium (IBSS)*. 1–6. https://doi.org/10.1109/IBSS.2015.7456652

[10] Jairo Giraldo, David Urbina, Alvaro Cardenas, Junia Valente, Mustafa Faisal, Justin Ruths, Nils Ole Tippenhauer, Henrik Sandberg, and Richard Candell. 2018. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 76.

[11] 1954 Hamilton, James D. (James Douglas). 1994. *Time series analysis*. Princeton University Press, Princeton, N.J. ; Chichester.

[12] Vittorio Illiano. 2018. Ensuring the resilience of wireless sensor networks to malicious data injections through measurements inspection.

[13] Vittorio Illiano and Emil Lupu. 2015. Detecting Malicious Data Injections in Wireless Sensor Networks: A Survey. *ACM Computing Surveys (CSUR)* 48, 2 (November 2015), 1–33.

[14] V.P. Illiano, L. MuÃśoz-GonzÃąlez, and E.C. Lupu. 2017. Don't fool me!: Detection, characterisation and diagnosis of spoofed and masked events in wireless sensor networks. *IEEE Transactions on Dependable and Secure Computing* 14, 3 (2017), 279–293.

[15] Vittorio Illiano, Andrea Paudice, Luis MuÃśoz-GonzÃąlez, and Emil Lupu. 2018. Determining Resilience Gains From Anomaly Detection for Event Integrity in Wireless Sensor Networks. *ACM Transactions on Sensor Networks (TOSN)* 14, 1 (March 2018), 1–35.

[16] Vittorio P. Illiano and Emil C. Lupu. 2015. Detecting Malicious Data Injections in Event Detection Wireless Sensor Networks. *Network and Service Management, IEEE Transactions on* 12, 3 (September 2015), 496–510.

[17] Jason Imperato, Daniel J Henning, Patrick J McBee, Leon D Sanchez, et al. 2017. Using markedly abnormal vital signs in the emergency department to anticipate needs for intensive care unit admission. *Journal of Acute Disease* 6, 6 (2017), 268.

[18] Chris Karlof and David Wagner. 2003. Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks* 1, 2 (2003), 293–315.

[19] Joseph B Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society* 7, 1 (1956), 48–50.

[20] Dorothy N. Monekosso and Paolo Remagnino. 2013. Data reconciliation in a smart home sensor network. *Expert Systems with Applications* 40, 8 (June 2013), 3248–3255.

[21] Adrian Perrig, John Stankovic, and David Wagner. 2004. Security in wireless sensor networks. *Commun. ACM* 47, 6 (June 2004), 53–57.

[22] Physiobank. 2000. physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov P, Mark RG, Mietus JE, Moody GB, Peng C, and Stanley HE. Circulation* 101, 23 (2000).

[23] Robert Clay Prim. 1957. Shortest connection networks and some generalizations. *Bell system technical journal* 36, 6 (1957), 1389–1401.

[24] Murad Rassam, Anazida Zainal, and Mohd Maarof. 2013. Advancements of Data Anomaly Detection Research in Wireless Sensor Networks: A Survey and Open Issues. *Sensors* 13, 8 (August 2013), 10087–10122.

[25] Osman Salem, Alexey Guerassimov, Ahmed Mehaoua, Anthony Marcus, and Borko Furht. 2014. Anomaly detection in medical wireless sensor networks using SVM and linear regression models. *International Journal of E-Health and Medical Communications (IJEHMC)* 5, 1 (2014), 20–45.

[26] Richard Shiavi. 2007. *Introduction to applied statistical signal analysis : guide to biomedical and electrical engineering applications* (3rd ed. ed.). Academic, Amsterdam : London.

[27] David I Urbina, David I Urbina, Jairo Giraldo, Alvaro A Cardenas, Junia Valente, Mustafa Faisal, Nils Ole Tippenhauer, Justin Ruths, Richard Candell, and Henrik Sandberg. 2016. *Survey and new directions for physics-based attack detection in control systems*. US Department of Commerce, National Institute of Standards and Technology.

[28] James Welch, Benjamin Kanter, Brooke Skora, Scott McCombie, Isaac Henry, Devin McCombie, Rosemary Kennedy, and Babs Soller. 2016. Multi-parameter vital sign database to assist in alarm optimization for general care units. *Journal of clinical monitoring and computing* 30, 6 (2016), 895–900.