Real-world Deployment of a Smart and Green Wireless Sensor Network for Intrusion Detection

Maïssa Mbaye[1], Mohamed Aymen Chalouf[2], Francine Krief[3] and Martin Peres[3]

[1] LANI , Gaston Berger University
Saint-Louis, Senegal

[2] IRISA, University of Rennes 1
Rennes, France

[3] LaBRI, Bordeaux INP
Bordeaux, France

Abstract

Intrusion detection is one of the most challenging applications in this research field of Wireless Sensor Networks(WSN). It combines challenges of minimizing false positives and negatives as well as those optimizing WSN lifespan.

In this paper we present the design and the deployment of a smart and green wireless sensor network for intrusion detection. Our system considers surveillance zone as a set of detection areas. Each area contains a set of sensor nodes related to each other and collaborating with each other in order to reduce false positives and false negatives under the supervision of a correlation node. The correlation node orchestrates area events to send alarms only when a true intrusion occurs and detects faulty nodes thanks to reputations parameters.

We tested our proposition with FreeRTOS based simulator we developed and real-world nodes. Results of simulations as well as those of the real-world deployments show that our proposal improves energy consumption and intrusion detection efficiency.

Keywords: Wireless Sensor Networks, Collaborative intrusion detection, Real-world deployment, Green networking, Autonomic networking.

1. Introduction

Surveillance networks require multiple wireless sensors spread over a large area. It would be impractical for nodes to have the necessary power to be able to reach the computer that centralises the sensor's data directly. Surveillance networks thus usually use a multi-hop ad-hoc network topology.

Data transmission in a multi-hop wireless ad-hoc network plays a large role in nodes' power consumption [1] and it will comparatively grow larger and larger as computing costs go down thanks to the constant improvement in the processors' efficiency (Koomsley's law [2]). This is especially the case in Wireless Sensor Networks (WSNs) where computing power is needed because sensor nodes are expected to keep the network connected, acquire data from the connected sensors, create network frames containing the sensors' data and send them to a gateway.

The energy cost of transmitting 1 KB over a distance of 100 m is approximately the same as the cost of executing 3 million instructions by a 100 million instructions per second (MIPS)/W processor [3]. However, a transmission does not only drain the emitter's battery, it also increases the power consumption of surrounding nodes as they have to demodulate the incoming signal and process the decoded frame. It also increases the contention over the network which prevents radios from being asleep when they do not need to communicate because they need to check if the medium is free or not. WSN lifespan can be enhanced by reducing the number of the transmitted packets and/or their size even if it will need more processing power resources. This trade-of depends on the distance at which the message should be sent and the efficiency of the processor. Localizing data processing among the nodes of a network is the best way of reducing the size and the number of frames even if it creates other challenges as we will see.

WSN for intrusion detection combines the constraints of network lifespan enhancement with those of intrusion detection So, the challenge here is to reduce communications for energy savings thanks to the collaboration between the nodes. Our proposition addresses these challenges and is a continuous enhancement of our previous work [4]. The main principle is that we have two levels of decision making. The local decision process we propose is based on modality-agnostic collaborative detection of spatio-temporally correlated events and it has two levels. The first level of correlation aims at sending the minimum amount of information as possible to keep the current view of the correlation node up to date without streaming data continuously. The first level thus runs on all the nodes hosting one or more sensors and aims at detecting events using only values returned by the sensor. When an event happens, the node hosting the sensor sends an alert to the correlation node. The second level of decision-making aims at correlating the alerts sent by the sensors of the area before sending alarms to further reduce the number of messages sent in the network. Since the correlation node is filtering most of the messages of the area, it would be impossible for the operator of such a network to audit the cause for a false positive or a false negative. To enhance the auditing capabilities of the network, we proposed storing a history of the most important events that happened in the network for a defined period of time. We also proposed a reputation mechanism to detect faulty sensors automatically to alert the network operator. This mechanism can also be used inside the network to evict faulty sensors from the correlation process. Finally, we implemented the proposed algorithms and deployed them in various realistic scenarios.

In Section 2, we introduce related work. In Section 3, we present our green and self-optimized intrusion detection. The modality-agnostic collaborative detection of spatio-temporally correlated events in a WSN is evaluated in Section 4. Section 5 details how the proposed algorithms were implemented and deployed in a realistic scenario. We conclude and present future work in Section 6.

2. Related Work

In spatially-correlated sensor networks, collaborative intrusion detection builds on two concepts: (1) data aggregation where sensors first locally correlate sensing data to remove unnecessary information and then, (2) cluster-based data aggregation where the clusterhead correlates the information of all the sensors of its area [5]. This approach avoids false positives induced by defective or ill-calibrated sensors and can be used on heterogeneous sensor networks [6]. Moreover, it is known to produce better results than value-fusion when fault-tolerance is needed [7].

Distributed and collaborative detection reduces both the number and the average length of communications [4]. This is to our knowledge the best way to provide green WSNs.

Using reputation in the context of wireless sensor network is not new. Indeed, some research works used reputation in broad contexts such as enhancing node's authentication [8], detecting malicious behaviors [9][10], clustering [11] and data routing [12]. The main difference between our contribution and these works is the use of the reputation notion for designing an efficient WSN for intrusion detection. Our solution combines local correlation mechanism, area level collaboration for detection and reputation to reduce false positives and negatives while detecting faulty behaviors of sensor nodes.

In the field of WSN, only a limited number of researches achieve real world deployment, probably because of complexity of reproducibility. Indeed, most of the work, such as [13] and [14], is only theoretically evaluated. By the way, few scientific platforms have been deployed during this last decade such as historical Motelab that provides an indoor testbed [15] and Trio which enables a large-scale solar-powered sensor network [16]. More recent platforms have been developed like Vigil-Net for field surveillance [17], SensorScope for weather monitoring in the wild [18], and GreenOrbs [19] for measuring systems performance and scalability.

Concerning our application domain (i.e. WSN for intrusion detection), we note that most of the past and recent works ([20], [21], [22] and [12]) used simulations to evaluate their propositions. Despite the interesting ideas and approaches proposed in some of those works ([22], [13] and [12]), we think that a real world validation is required to confirm the theoretical results as in [23].

Using reputation in the context of wireless sensor network is not new. Indeed, some research works used reputation in broad contexts such as enhancing node's authentication [8], detecting malicious behaviors [9][10], clustering [11] and data routing [12]. The main difference between our contribution and these works is the use of the reputation notion for designing an efficient WSN for intrusion detection. Our solution combines local correlation mechanism, area level collaboration for detection and reputation to reduce false positives and negatives while detecting faulty behaviors of sensor nodes.

In this section, we tried to go around the work related to the main issues discussed in this paper. These lasts cover the approach (local, collaborative, etc.) on which the intrusion detection can be based, as well as the definition of some form of intelligence aiming at ensuring reliability and energy savings. Indeed, one original feature of our work consists in the provided reliability by basing the

detection system on parameters like reputation and confidence levels, which are generally used for security purpose. In addition, any type of new sensors can be added at runtime thanks to the criticality level. Finally, we made a point about the actual deployment level of some similar project. Considering this feature, we think that this work is characterised by the culmination of deployed solution.

3. Green and Self-optimized Intrusion Detection

Logical reasoning allows a system to make inferences using only logical deductions [24]. An automated logical reasoning can use prior knowledge and/or experience to improve its reasoning abilities [25].

We take advantage of the existing modal and spatial redundancy in sensor networks where sensing ranges overlap to achieve better accuracy, reliability and energy efficiency through smart cooperation. In-network reasoning and cooperation between autonomous sensors allow real-time surveillance, with reconfiguration flexibility.

3.1 Collaborative detection of spatio-temporally correlated events

Figure 1 presents an area of WSN for intrusion detection. In this WSN, 'D1', 'D2', 'S1' and 'S2' represent sensor nodes forming a cluster which is managed by the 'Cluster Head'. When an intruder is inside this covered area, it will be detected by all the sensor nodes and, ideally, each of them will send an alert to the 'Cluster Head'. This last node is responsible of correlation and transmission of alarms based on the received alerts and some other parameters described in section 3.1.3 . In this section, we describe the main aspects of our proposition.

3.1.1 Assumptions

Our solution is based on correlation mechanisms and assumes the following hypothesis:

- **Sensors are area organised:** each sensor node of the WSN belongs to an area as in clustering. An area is a cluster-like organisation where all sensor nodes can communicate and are correlated. In other words, an area is geographically limited by communication range of nodes inside and the fact that information they can provide are related. Indeed, in normal and optimal operation, when an intrusion occurs, each sensor node inside the area should detect it in maximum time that we call *minimum intrusion time*. These constraints must be respected during the WSN deployment in order to reduce energy consumption and communication range.
- **Sensors are not individually fully reliable:** this means that they may err in detecting intrusion and emit false positives. Sensors are supposed to be ill calibrated when they are deployed and they may sometimes not be reachable, even if the network is globally connected. Our solution is fault-tolerant using decision-making process at the sensor level and correlation processing at the area level. Indeed, a correlation node, at least one in each area, will monitor failing nodes and make decision with the aim of reducing false positives and negatives.
- **Sensors are ill calibrated in deployment time:** This means that thresholds used by sensor to detect an intrusion are not fully correct during deployment time. Our system will detect trends of its detections; try to self-adjust its calibration to enhance detection efficiency. Each node makes a local decision to send or not an intrusion alert. Based on the received alerts from the area's sensor nodes, the correlation node will make a decision (raise or not an alarm). Each node has a local threshold used to decide to send or not an alert. Using criticality level (see section 3.1.3) enables an abstraction of sensor type participating in intrusion detection. This abstraction has many benefits such as the ability to correlate sensors alerts (including ill calibrated ones) to decide whether or not something happened. Dedicated nodes are responsible of this task (correlation node described in section 3.1.3).

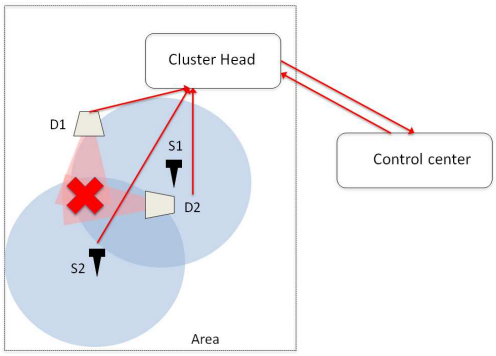


Figure 1: An example of area with heterogeneous sensors

3.1.2 Communication's system

Communication between sensor nodes and correlation node is done using topic-based Pub/Sub paradigm. This communication model is based on data-interest announcement [26]. Subscribers announce descriptions of data they want to receive (topics) and the system bring them by filters applied to messages in the network. In our context, sensor nodes in the area are publishers since they have individual alerts they want to send to correlation nodes. Correlation nodes are subscribers since they need to receive alerts to do their main task of correlation and alarm transmission.

Pub/Sub communication system brings two main benefits to our solution. Firstly, it facilitates the replacement of the correlation node since Publishers do not have to know directly who the subscribers are. Secondly, correlation node can make complex filtering of data they want to receive by creating topics. For instance, when a sensor is not working properly, correlation node can simply add a condition to exclude alert from this sensor temporally.

3.1.3 Network's nodes

- **Sensor nodes:**

Sensors detect intrusion using local threshold. If sensed value is out of normal threshold, sensor node raises an alert. Since at the beginning sensors' individual thresholds are ill calibrated, there is a calibration period in which each sensor tries to readjust its local threshold. During this period, sensors would generate false positive but correlation node should not consider any of them as faulty node. Local threshold readjustment is done using the following event-based algorithm and executed by all sensor nodes.

Algorithm 1 : Node's local threshold adjustment during calibration time

On Event 1 : Current node has detected an intrusion and correlation node confirms that it is a true Intrusion

| do not modify Current Threshold

On Event 2 : Current node has detected an intrusion and correlation node confirms that it is a false positive

$$| \quad threshold = cur\_threshold - \left(\frac{cur\_threshold - min\_sensor\_value}{2} \right)$$

On Event 3 : Current node missed a true intrusion (it received an alarm from correlation node without detecting it)

$$| threshold = cur\_threshold + \left(\frac{max\_sensor\_value - cur\_threshold}{2} \right)$$

This algorithm works with the principle that if a sensor raises a false positive it means that its local threshold is probably too low and should be increased and if a sensor missed a true intrusion it means that sensor's local threshold is too high and should be decreased. Node's thresholds are adapted using a modified dichotomy version and according to min and max thresholds. In this new process, dichotomy is between current value and maximum or minimum value instead of the center of two last value. The main argument not using pure dichotomy is its sensitivity to faulty events at the early network operation. Indeed, if first alarms are false positives, pure dichotomy cannot correct itself event with a large number of correct alarms after. Finally if the sensor node receives positive feedback from correlation node, it can consider that current threshold is good. Figure 2 shows an example of threshold adaptation.

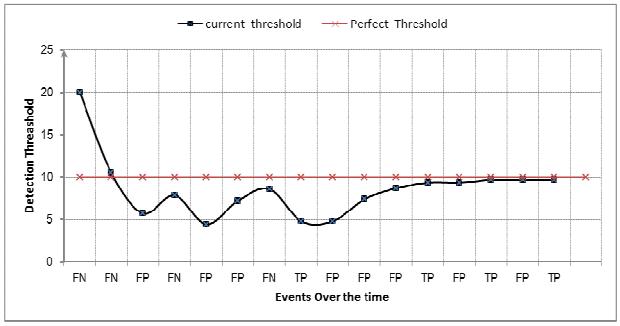


Figure 2: Example of the adaptation algorithm of a node's threshold (FP: False Positive, FN: False Negative, TP: True Positive)

Each sensor has to detect events by itself. This detection follows an algorithm which mainly depends on the modality of the sensor as well as the event to detect. For example, in the case of the use of a binary infrared optical barrier in the detection of a car driving down a road, the squared signal length will depend on the signal's length and car's speed. The polling frequency will also depend on the signal's length as well the car's speed. When a sensor detects something, it uses a correlation timer to confirm the event. The event is confirmed when the sensor keeps on detecting something, and a message ("alert") should be sent to the correlation node.

A minimum delay between alerts should be implemented in order to further reduce the number of messages. Doing so while keeping the correlation node up to date concerning the state of each sensor may require to break the re-emission rule if the node participates to more detection (i.e. its confidence level increased). For instance, in Figure 3, we see the evolution of the value returned by an analog sensor. An alert is first sent when the value increased over a threshold and then re-emitted after a period. If the value changes in a fast way, an alert can be sent earlier.

- **The correlation node**

Correlation node in one area is a smart node. When the correlation node receives an event from a sensor (alert), we propose that it stores the current timestamp into memory for future reference and computes the confidence level of that sensor. Then, it should compute the area's *criticality level* by summing the contribution of all the area's sensors. Being able to automatically trigger responses to an event allows the creation of a fully-autonomic collaborative WSN. A sensor's contribution to the criticality level is the confidence level of the sensor times the age factor. The confidence level is attributed to sensor detection and ranges from 1 to 3. The *confidence level* of each area's node will be computed and managed by the correlation node as detailed in Section 3.2. Its value is initialised to 1,

and should then be increased gradually to 3 as long as the sensor keeps on detecting intrusion. During the correlation time, the age factor decreases from 1 to 0 in a linear manner. An event has a minimum time to occur and the correlation time should have the same duration roughly. For instance in intrusion detection scenario, it is equal to the maximum time taken by a human walker to cross the monitored area. If a sensor becomes unavailable, its criticality level should be raised to the maximum for a few minutes to prevent attacks on the system that involve disabling sensors remotely by shooting them. When detection intrusion occurs, old alerts raised by individual sensor nodes should count less than newer alerts in correlation process. This is due to the fact that alerts are related to individual intrusion and then have a limited relevance that decreases with time. The age of alerts in correlation is controlled by the *age factor* parameter. For simplicity, we choose linear decreasing for this factor. However, depending on the application, the way it decreases can be more sophisticated. The criticality level of an area is computed based on age factor using the following equations (1).

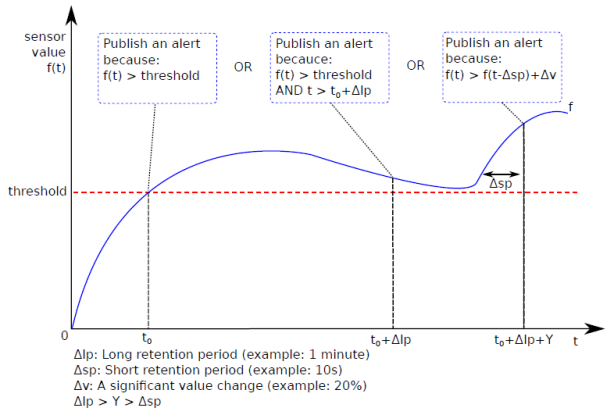


Figure 3: An example of alert re-emission policy depending on the evolution of an analog sensor's value

$$age\_factor(a) = \begin{cases} 0 \; if \; alert\_age(a) > correlation\_time \\ 1 - \dfrac{alert\_age(a)}{correlation\_time} \quad otherwise \end{cases}$$

$$alert\_age(a) = current\_time() - alert\_timespamp(a)$$

$$contrib\_alert(a) = confidence(a) \times age\_factor(a) \qquad (1)$$

$$criticality\_level = \sum_{a=0}^{alert\_count} contrib\_alert(a)$$

An area has a criticality level that increases according to the number of sensor nodes in the area that detect an event. The criticality level continues to grow until a threshold that depends on the minimal correlation factor targeted by the system designer and a latency mode indicating how early an alarm should be sent to notify the zone operator or to trigger an automatic response. The area's criticality threshold is computed using (2) when a new sensor node is added to the area or after a sensor node has been unreachable for some time, 10 minutes for instance.

$$criticality\_threshold = sensors\_count \times 1.5 \times latency\_mode$$

$$latency\_mode = \begin{cases} 0.25 \rightarrow Red\ Mode \\ 0.50 \rightarrow Orange\ Mode \\ 0.75 \rightarrow Yellow\ Mode \\ 1.00 \rightarrow Green\ Mode \\ 1.50 \rightarrow White\ Mode \end{cases} \qquad (2)$$

The correlation node is responsible of notifying the control center about intrusions. We call this out-of-area intrusion notification an 'Alarm'. Alarms are emitted when area criticality level goes over the area threshold and are destined to the network administrator. Sensor nodes inside an area use this alarm as a feedback to their individual alert from correlation node. They adjust or not their calibration on the basis of this feedback. The 1.5 constant from Equation 2 is the maximum confidence level (3) divided by 2. This means that when using the green latency mode, all the sensors should report an average of 1.5 before an alarm is sent. As we said before, when starting the intrusion detection system, all sensors have a minimal confidence level, which is equal to 1. Then, this level will increase progressively when the node participates in the detection of actual intrusions. This requires considering two important aspects. First, specification of the link between the confidence level of a node and its reputations (defined in Section 3.2). Then the start-up phase of our system. Indeed, if the system is started with "Green" mode, the latency_mode will be equal to 1 and the confidence of all the nodes will have a value of 1. The criticality threshold will then never be reached and the confidence level of nodes will not move. Thus, we propose to start the system in "Orange" mode to make possible the detection of actual intrusions. Then, the confidence level of the nodes will increase progressively, and we can switch to "Yellow" mode and the "Green" one. "Red" and "White" modes will be respectively used when the risk of intrusion is maximum or minimum. We note that these two modes may increase respectively the number of false alarms and the undetected intrusions. It is also important to note that when our intrusion detection system stabilises, the "sensors_count" parameter (Equation 2) is equal to the number of nodes having a confidence level above a certain threshold. Below this threshold, we can no longer trust the node.

An example of the evolution of an area's criticality level can be seen in Figure 4. At t0, a node sent an alert which raised the criticality level. This criticality level then

linearly decreased towards 0. At t4, the contributions of alerts received at t1, t2, t3 and t4 raised the criticality level enough to reach the criticality threshold, which led to the emission of an alarm. At t5, the correlation node received another alert which contributed more than the alert at t0 because the alert had a higher confidence level.

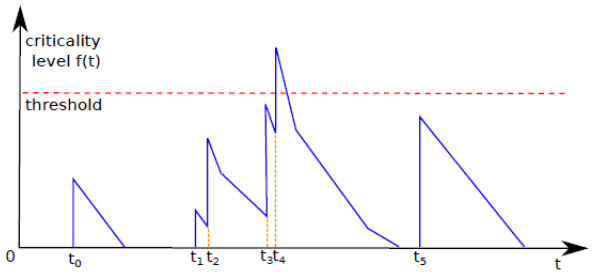


Figure 4: Example of the evolution of an area's criticality level

Our solution is organised around the concept of events correlation (node and area levels). An important issue is how correlation nodes are elected inside areas. This election should be done to avoid single point of failure and ensure fault tolerant operation. If a correlation node in not available for any reason (energy depletion, material fault, selective jamming of this node), the overall system should be able to handle this failure and continue to work.

To achieve this objective, we propose that each area should have two correlation nodes: one primary correlation node and one secondary correlation node. These nodes subscribe to Alert topics (Pub/Sub communication paradigm) to receive all the individual alerts transmitted by all the sensor nodes. Based on the received alerts, the primary and secondary nodes can decide to send or not an alarm to the control center. The secondary correlation node is responsible of sending alarm when the primary one is not available or has an incorrect behavior. In other words, the secondary node is the node that will replace the primary correlation node if this last is unavailable or faulty. When the primary node works perfectly, the secondary one behaves like a normal node regarding alarm. The difference between the primary and the secondary correlation nodes is the following. The primary correlation node must transmit an alarm as fast as possible. As for the secondary correlation node, it has to wait for a certain delay (e.g. some seconds) before the emission of the alarm unless the primary correlation node transmits the alarm before the delay's expiration.

The election of the two correlation nodes could be performed just after deploying the whole network (i.e. all the nodes). Since the election process takes into account some relevant parameters (e.g. distance to the other area's nodes), this solution will allow the system to have an optimal behaviour at starting time. However, for simplicity, we chose to designate the two first deployed nodes as the primary and secondary correlation nodes. Then we defined the events that will cause re-election of a new secondary node among other nodes, for example, when the primary correlation node has not enough energy to handle next sensor's events. Since the correlation node has to handle the area nodes reputations, we have two possible cases: the primary correlation node is able or not to transfer the knowledge about nodes' reputations to the secondary one which is automatically promoted primary node. In the first case, the system will rapidly retrieve its normal functioning. However, when the primary node is not able to transmit that knowledge, the secondary node will restart the computation of all the areas nodes' reputations. Another solution consists in computing and storing reputations by the two correlation nodes which will allow a rapid retrieval of the normal system's behaviour. This last solution will have a cost in terms of the used resources by the second correlation node. The secondary node is also responsible of noticing the administrator when primary node is not working properly. This misbehavior of primary node can be detected using parameters such as the number of untreated alarms or the number of false positives from administrator's feedback. The decision of downgrading a primary node is decided by administrator for security purpose.

The main challenge of the election in WSN is to define the set of criteria on which the election will be based. Many existing clustering algorithms for WSN tackle this problem [27]. In our proposition we use three criteria: node's absolute available energy (Joules); distance to the network's gateway (number of hops); node's connectivity (number of routes to the gateway). These criteria help to reduce energy consumption while being fault tolerant. By minimizing the distance to the gateway, less power will be consumed to send alarms to the control center. When a node maximizes energy level and the number of routes to the gateway this reduces risks of faults caused by energy lack or connectivity loss. Another aspect of fault-tolerant is redundancy of correlation nodes, two ones at any time in any area. In a more general context, we can consider the election of *n correlation* nodes to ensure the intrusion detection with a different delay of the alarm transmission. This redundancy of correlation nodes will cost in term of power consumption. By receiving all alerts from nodes in the area and correlating events, a correlation node consumes more power than a simple node. There is a trade-off network lifespan and detection quality enhancement with redundancy.

3.2 Sensor's reputations and confidence

During its operation, a node can emit some false positives and it can also miss some intrusions that occur in its area. This history of node's activity can be used to

evaluate its reliability and correctness of the alerts it emits. We call the result of this evaluation reputation. We define two reputations by two separated parameters: *False Positive reputation* and *False Negative reputation*. The first measures how often the current node detects an intrusion while nothing happened, what is the usefulness of alerts it emits. The second measures how often the current node missed a real event.

We consider a false positive happened when the network emitted an alarm when no real event happened. Likewise, a false negative happened when the network did not emit an alarm when a real event happened. At a sensor's level, a false positive is when the node sent an alert when no meaningful event happened and a false negative when it did not send one when it should have.

A false negative can happen when the sensor did not detect the event (wrong polling frequency on the sensor or wrong calibration). A false positive can happen when the sensor's sensitivity is too high or if detected an event that was not meaningful to the network (a dog instead of a pedestrian for instance).

For simplicity, reputations are represented by real numbers varying from 0 to 1. The best reputation has value 0 while the worst value is 1.

Correlation node manages a history window of each sensor alerts. From this history it maintains statistics needed to compute correlation. The first statistic it maintains is the Total number of alerts from a sensor. This is just a counter incremented each time the related sensor emits an alert. This counter is the base for calculating reputation. The second statistic is the Total number of true positive involving a sensor. Correlation node raises an alarm when enough sensor nodes emit alerts for the same event to ensure that is a true positive. These alerts are summed up using criticality level variable to know when there are enough alerts. In this case correlation node will increase the total number of true positive for all sensors that contribute to alarm. This counter helps to evaluate how much a sensor has contributed in raising an alarm. The false positive reputation of a sensor computed using this counter with the equation detailed in (3).

$$reputation\_\ fp(s) = 1 - \frac{area\_\det ection\_count\_involving(s)}{sensor\_events\_count(s)} \quad (3)$$

The third counter is the number of alarms a sensor misses to detect. This counter is relevant to know the ratio of missed alarms by current sensor. The False Negative Reputation is computation is detailed by the equation (4).

$$reputation\_\ fn(s) = 1 - \frac{sensor\_correlated\_count(s)}{area\_\det ection\_count()} \quad (4)$$

Reputation is used to weight sensors contribution in the criticality level of an area during the correlation process. In particular, the false positive reputation is very relevant. When a sensor node has a good false positive reputation (approaches 0), it means that its alerts are reliable and then its contribution should be highly weighted. In reverse when a sensor node has a bad false positive reputation (approaches 1) its contribution to criticality level should be low. In Addition, correlation node should alert the operator when one sensor gets one of its reputations lower than a certain threshold. Finally, we propose that sensors producing too many false positives should be evicted from the correlation process to avoid polluting the history of the correlation node and reduce the number of false alarms. Likewise, we propose that if a node has too many false negatives, they should be evicted or simply not taken into account when calculating the threshold. We propose those binary decisions because the area's threshold computation should be stable which prevents using the false positives/negatives values for the criticality and/or the threshold computation.

As introduced before, there must be a link between the confidence level of a node and its reputations (false positive and false negative). Thus, we defined a formula (Equation 5) to express the confidence level based on reputations.

$$confidence(s) = 3 - (\alpha \times reputation\_\ fp(s) + \beta \times reputation\_\ fn(s)) \quad (5)$$
$$where\ \alpha + \beta = 2$$

From Equation 5, we note that the initial confidence level of a node is equal to the minimum value (1), because its reputations are initialised to the worst value (1). If the node performs well in detecting all intrusions, its reputations will be optimal (0 or near) and its confidence level will then quickly reach or approach its maximum value (3). Otherwise, the confidence level will decline and stabilise at its minimum value (1). Thanks to the two coefficients (α and β) we can give more importance to one of the two reputations. If the two reputations have the same relevance, we can fix α = β = 1.

3.3 Auditing

Due to the drastic message reduction found at the gateway when using in-network reasoning, auditing the system becomes difficult. Auditing is needed by the network operator to understand what is going wrong with the system in case of false positives or false negative detections.

In our solution, statistics may be sufficient to reason and decide for any alarm event. However, these are not the only important information for the network administrator. This last may want to see into details what happens; if

there is an intruder, where it comes from, where he is, where he seems to go, etc. To meet this constraint, the network will store history of important events inside correlation nodes. Since all events could not be stored, we propose to only store events in a FIFO queue using window sliding of time (a day for example). A historic event record is composed of the local timestamp, a confidence level and the list of sensors contributing to this event and their relative contribution to it. When more space is needed to store a new event, the oldest event with the lowest usefulness is deleted from queue and the new event is added at the end of the queue. The usefulness of an event is a score attributed by correlation node according to the confidence of the event in a way that it decreases with the time.

The event confidence is computed by the following formula (Equation 6).

$$score(e) = \frac{confidence(s) \times local\_\max\_time(e)}{1 - \frac{age(e)}{\max\_storage\_time}} \quad (6)$$

An event is not stored in the history unless it reaches the history threshold. An example of the evolution of an area's criticality level can be seen in Figure 5.

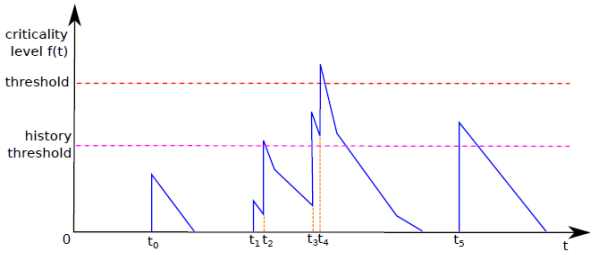


Figure 5: Example of the evolution of an area's criticality level with regards to the the history threshold

Figure 5 features the same criticality level evolution as in Figure 4 with the addition of the history threshold. In this example, only the events at t2, t3, t4 and t5 will be stored to the history because the others did not increase the criticality level enough to reach the history threshold. The most significant entry would be the one found at t4 because it is the one with the highest criticality level and it has been the local maximum for a long time. The least significant event in this example is the one from t3, despite its relatively-high criticality level. This is because it has been the local maximum for a very short period. This event will thus be the first one to get deleted if space was becoming a problem.

The history along with all the other parameters and context can then be queried on-demand by a network operator using a REST-like protocol such as CoAP [28]. This improves the network's debugging abilities without needing an external radio to spy on the exchanged messages.

4. Evaluation

This research was conducted during the DIstributed Applications and Functions Over Redundant Unattended Sensors (DIAFORUS) ANR project. The project's goal was to develop an energy-efficient framework for distributed applications and functions over redundant unattended sensors. As a demonstration of the framework, an intrusion detection application was written using redundant and heterogeneous sensors.

For evaluation purpose we use a physical intrusion detection scenario. Testbed's parameters are the following:

Parameter	Testbed value
Correlation time	<= 1s (less than 1s)
Detection latency	<=10s
Maximum detection time	5s

The evaluation has been implemented in a custom emulated environment based on FreeRTOS operating system. In this environment, virtualised instances of FreeRTOS represent sensor nodes. Communications between nodes is managed by a Dispatcher and are implemented using TCP sockets. Architecture is described in Figure 6.

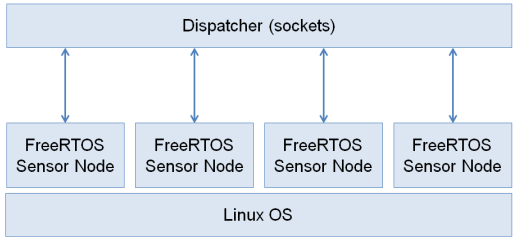


Figure 6: Architecture of our environment

The compilation process is controlled by a custom-built python script called "build network.py" which can also be used to flash the firmware of all the nodes. For the network layer, we decided to use IPv6 Low Power Wireless Personal Area Networks (6LoWPAN) instead of IPv6 because of its reduced header size.

Routing is handled using RPL [29], an IPv6 Routing Protocol for Low-Power and Lossy Networks, because it is becoming a standard in wireless sensor networks. We use to query nodes via the REST protocol CoAP [28] and route Pub/Sub messages from/to the broker. RPL's DIO mechanism is used as a heartbeat indicating to surrounding node that the node is still available. This heartbeat is used to detect faulty nodes.

Diase (DIAforus Simulation Environment) is a simulation environment developed to allow us to simulate our proposal. It enables the simulation of many intrusion scenarios (deployment, monitoring, etc.). Diase is meant to be used on a tablet. The main GUI of Diase can be seen in Figure 7 with an intrusion detection scenario. It showcases 2 areas containing 8 nodes, 3 seismic sensors, 2 infrared directional barriers sensors, a multi-directional barrier sensors and 1 actuator (an alarm).

The Command & Control mode (C2 mode) can be seen on Figure 7. In this Figure, we can see that the area 2 is reporting an intrusion thanks to its red colour. The sensors that contributed to the intrusion are nodes 21 and 23. The past intrusions can be found in the alarms log. This file contains the time at which an alarm happened, the area concerned and the list of sensors involved. It also contains the intrusion time which is computed as the difference between the first sensor that detected the intrusion and the last one that did. This enables the operator to get a sense of the speed of the intrusion and allow him/her to react appropriately. For instance, in this example, the alarms log shows that multiple alarms have been received for the same intrusion because the intruder(s) that is/are currently in the area move(s) very slowly.

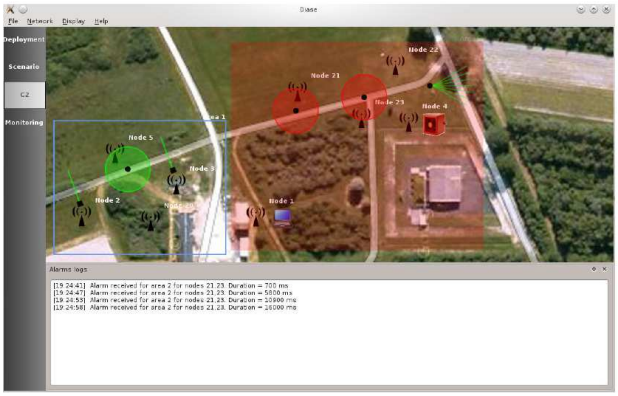


Figure 7: Command & Control mode: Visualization of an alarm in an area along with which sensors participated in it

Diase also allows the network operator to inspect the state of every node in the network. To do so, Diase uses a REST protocol for Constrained Applications called CoAP. It then displays the gathered data into a textual or a graph form. Examples of such graphs can be seen in Figure 9.

- **Modality-agnostic Collaborative Detection of Spatio-temporally Correlated Events**

The aim of collaborative intrusion detection we developed is to reduce both the number and the average length of messages. We use these parameters as metrics for our simulations.

Figure 8: Screenshot of Diase running an intrusion scenario in real time and injecting simulated events to the emulated wireless nodes.

As we assumed, sensors may be ill calibrated at deployment time. In other words, the probability they emit false positives is not null. We model alert decision of sensors with a Bernoulli distribution. This distribution is used as follows: considering the parameter p, a sensor node has the probability p to generate a false positive and a probability 1-p to choose the right decision not to emit an alert. Every second, each node has to take this decision.

Considering an area composed of 3 nodes reading sensed values every second during 30 minutes, we compared the number of messages in different kinds of WSN topologies:

- The number of messages exchanged in short term range (one hop) and long-distance (towards the control center),
- The number of sensed (read) values by sensor nodes compared to the number of messages.

The results can be found in Table 1.

Table 1: Comparing WSN data management on a 3-nodes area with a sensor false positive probability (p=0.1, f=1Hz)

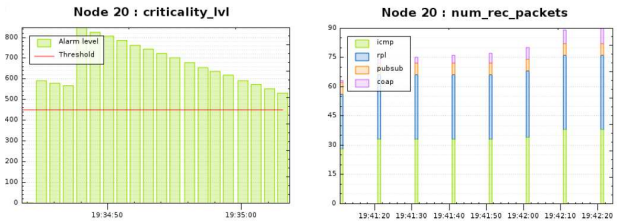
WSN type	readings	short-distance	long-distance
Sink	5400	0 (0%)	5400 (100%)
Cluster aggregation	5400	3600 (67%)	1800 (33%)
Local detection	5400	0 (0%)	540 (10%)
Collaborative detection	5400	$\leq$ 540 (10%)	$<$ 180 (3.33%)

In the tree topology WSN, to achieve the one second correlation time constraint, the network forwards a bulk message to the gateway every second. This message contains all readings data from sensor nodes. This operation mode is the worst case possible since it generates only long-distance communications toward the gateway.

In cluster data aggregation WSN, simple sensor nodes (actually 2 nodes of 3) send their data to a special node called aggregation node. This node aggregates (transforms 2 values to 1 value) data it receives before forwarding it to the gateway. This topology reduces long term distance in comparison to tree topology. In cluster data aggregation

only 1/3 (33%) of communications are long distance. All remaining traffics are local to the cluster (67%).

Local detection event consist in the case where each sensor node is responsible of detecting intrusion locally. They only communicate with the gateway to transmit value of a suspicious event. Since the testbed parameters set sensor's probability to detect an even to 0.1, they send to the gateway only 1/10th of values they read. In other words only 10% of communications in the network are long-distance while there is no local occurs in the cluster (short distance traffic).



(a) Criticality level of an area (b) Types of packets received

Figure 9: Examples of the real-time visualisation capabilities of Diase

Communications in collaborative detection will include intra-area and extra area communication (to the control center). Since correlation is done at area level, most of communications are of short range type. In theory, network should have much more short distance communications than long distance because long-range communications occur only when alarms are raised by the correlation node.

Another interesting question is how the correlation time and sensor ill calibration (noise) can influence number of intra and extra area communications. To answer to this question, we simulated a scenario with false positives only in the worst case, when the link to the correlation node is broken. Table 2 summarizes the results we got and from what we can learn two main things:

- The number of local detections is around 10%, which is the sensor noise.
- The longer the correlation time, the higher the number of messages

These results can be explained by the fact that the longer the correlation time is, the higher the probability of correlation between sensors is. To reduce the number of false positive, areas size should be minimised and this should be a very important optimisation.

We studied in Table 3 the influence relation between sensor noise in one side, and, the number and length of packets in the WSN on the other side. We can observe that the number of local detections is a linear function of sensor noise. This result is correct with our expectations. On the other hand, when the sensor noise increases, the number of communications toward the gateway increases too. In other words, the more sensor nodes are noisy, the more they generate long-distance communications.

With the result in Table 2 and 3, we can conclude that number of long-distance communication is not linear with the correlation time and noise probability. This fact can be probably explained by the no-alarm-re-emission policy that limits the number of packets in the network. We also have an average frequency of alarm of one alarm every 8.5 second when p=1 while the minimum intrusion detection time is set to 10s.

Table 2: Message count in DIAFORUS with noisy sensors (f=1Hz, p=0.1) and a correlation time c. Experiment time of 30 minutes

Correlation	readings	short-distance	long-distance	$\frac{long}{short}$ ratio
c=10s	5400	545 (10%)	8 (0.15%)	1.5%
c=20s	5400	558 (10.3%)	23 (0.43%)	4.1%
c=40s	5400	541 (10%)	32 (0.59%)	6%
c=60s	5400	516 (9.5%)	33 (0.61%)	6.3%
c=90s	5400	525 (9.7%)	40 (0.74%)	7.7%
c=120s	5400	518 (9.5%)	41 (0.76%)	7.8%
c=150s	5400	552 (10.2%)	50 (0.93%)	9%
c=180s	5400	520 (9.6%)	56 (1.03%)	10.7%

Table 3: Message count in DIAFORUS with noisy sensors (f=1Hz, p) and a correlation time of 180s. Experiment time of 30 minutes

Sensor noise	readings	short-distance	long-distance
p=0	5400	0 (0%)	0 (0%)
p=0.002	5400	11 (0.2%)	0 (0%)
p=0.02	5400	94 (1.7%)	0 (0%)
p=0.1	5400	545 (10%)	56 (1.03%)
p=1	5400	5400 (100%)	210 (3.89%)

The number of communications and their amount have been lowered by our collaborative approach. Indeed, no alarms have been raised by correlation node with a small correlation time (60 seconds) and low false positive probability. Then, we evaluated the link between the sensors' average noise and the minimum number of the transmitted messages. The worst studied scenario is characterised by a maximum noise probability (p=1). In this scenario, the increase in the number of messages in comparison with a 'Tree-topologyWSN' architecture is evaluated to 3.89%. In the same time, the average communication distance approached one hop. This could be considered as an important improvement in terms of latency and power consumption in such kind of networks (large scale networks).

- **Sensors Reputation Management**

Previously in this paper we studied effects of the average sensor's noise and the maximum intrusion duration on the average communication length and the number of messages. We showed that, assuming that sensor nodes are not too noisy; the correlation node will be able to filter events and identify false positives. A drawback of this filtering is that the operator will never get any information from faulty sensors. We propose using our reputation contribution to make sure the operator gets this

information by sending a "faulty-sensor" Pub/Sub message containing the ID of the faulty sensor when the false positive or the false negative reputation drops below 0.5 and there at least 3 detections.

We created a testbed scenario composed of two areas. Sensors in these two areas are not correlated and we use one area for false positive reputation and the other for false negative reputation computation. Figure 8 describes our testbed.

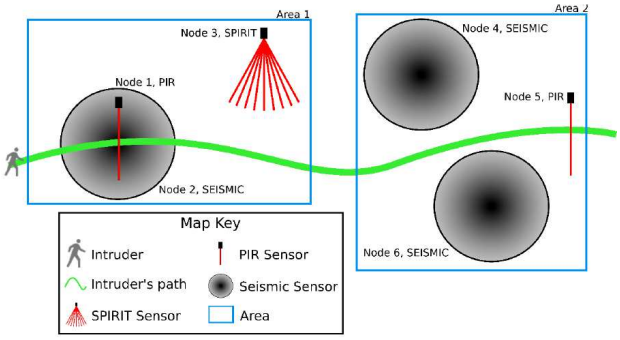


Figure 10: Scenario validating the reputation. 2 areas, 6 sensors.

In the testbed, sensors 1, 2 and 3 belong to area 1. This area is where we evaluate false negative events. Intrusion event in this area are generated in such a way that S1 and S2 always detect it and S3 never do it. Expected result is S1 and S2 should have perfect values for false positive and negative reputation while S3 has perfect value for false positive reputation and the worst one for false negative reputation. No sensor noise has been added to simplify validation.

We test false positives in area 2 which contains sensor nodes 4, 5, 6. A set of false intrusions are generated so as to be detected by the sensor 5, but never by the sensors 4 and 6. Once again, no sensor noise has been added to facilitate the validation. After a while, the nodes 4 and 6 should have both perfect false negative reputation and false positive since they don't detect any of false intrusions. However, node 5 is expected to have a perfect false negative reputation with the lowest false positive reputation because none of its alerts was useful to send an alarm.

Results described by Table 4 validated reputation computations we expected. Our proposal detects quite perfectly false positive and avoid false negative cases.

We then simulated the impact of noise (p=0.1, f=1Hz) on the sensors' reputation. No simulated intrusion is running during this experiment.

The results presented in Table 5 are obtained with the following characteristics of noise: p=0.1 and f=1Hz. They show that sensors transmit on average an alert each 10.7s. When the correlation time is equal to 20 seconds, there was a high probability that all sensors are involved in all the alarms transmitted over the 30 minutes of simulation. This explains that the false negative reputations of the three nodes take the value of 1.

Table 4: Results of the reputation experiment found on Figure 10. *NAN* is the value of False positive reputation (resp. False negative reputation) when *sensor_events_count(s)* (resp. *area_detection_count()*) equal to 0.

N_ID	False Positive Reputation	False Negative Reputation
1 & 2	$0=\left(1-\frac{18}{18}\right)$	$0=1-\frac{283}{283}$
3	$NAN=(0/0)$	$1=\left(1-\frac{0}{283}\right)$
4 & 6	$NAN=(0/0)$	$NAN=(0/0)$
5	$1=\left(1-\frac{0}{9}\right)$	$NAN=(0/0)$

Table 5: Reputation of noisy sensors (p=0.1), f=1Hz) after 30 minutes and correlation time of 20 seconds

N_ID	False Positive Reputation	False Negative Reputation
1	$0.660=\left(1-\frac{57}{168}\right)$	$0=\left(1-\frac{119}{119}\right)$
2	$0.658=\left(1-\frac{57}{167}\right)$	$0=\left(1-\frac{119}{119}\right)$
3	$0.662=\left(1-\frac{57}{169}\right)$	$0=\left(1-\frac{119}{119}\right)$

Sensors 1, 2 and 3 are relatively low but could be improved by less faulty sensors.

5. Real-world Deployment

The results presented earlier have been obtained using a simulated network. However, we ported this network on real nodes and validated the simulation results. The algorithms used for the simulation are the ones used on the real nodes. This means our proposal is feasible on standard sensor nodes. The hardware used and the tested scenarios are presented respectively in sections 5.1 and 5.2.

5.1 Hardware configuration

- **Sensors and actuators**

In our real-world testing, we used 4 types of sensors:

- An infrared unidirectional barrier from Thales (Figure11a);
- A 60° infrared barrier called SPIRIT from TechNext (Figure 11c);
- A seismic sensor from Thales (Figure 11b);
- A manual switch, for simulating a seismic sensor on tabletop demos.

The only actuator used during this demonstration was a buzzer to simulate an alarm when the correlation node would detect an intrusion.

- **Sensor nodes**

For our real-life test, we used the sensors nodes developed by Coronis. These nodes contain an APS3 core from Cortus [30], a 32 bit micro-controller with a Harvard architecture oriented towards power efficiency and small code size. We had 4 kB of RAM and 48 kB left for the code which could also hold some data although with a performance hit. The node is visible in Figures 11d and 11e.

(a) Infrared barrier

(b) Seismic sensor

(c) 16-beam infrared sensor

d) Four sensor nodes with an uni-directional infrared barrier attached

e) A close-up on the sensor node, when attached to its demonstration board

Figure 11: Deployment of some sensors for protecting a farm in a military field

The node's transceiver implements the Wavenis/wave2m standard [31]. It can operate on the 433, 868, 915 and 2400 MHz bands. Its bitrate ranges from 4.8 to 100 kbps. Its modulation is a GFSK and it also supports Frequency Hopping Spread Spectrum (FHSS) to increase the link reliability.

5.2 Real-world scenarios

We tested the versatility of our proposition using 6 operational scenarios proposed by Thales Telecommunications. These scenarios are presented in this section.

- **Scenario 1: Short-loop response**

This scenario aims at detecting an intrusion and triggering an automatic response without the intervention of the operator. The operator is however warned about the intrusion. To test short-loop response two unidirectional infrared barriers are deployed along a path leading to a restricted area. Another node receives the information about the intrusion and automatically sounds an alarm. The result shows that when a simulated pedestrian walks on the restricted path and crosses the first infrared barrier, nothing happens. When he/she continues and crosses the second barrier, an alarm sounds a few seconds later without the intervention of the operator. The operator however received an alarm on Diase, in the C2 mode. The result can be seen in Figure 12.

Figure 12: Testing the short-loop scenario on real sensor nodes

- **Scenario 2: Inter-area communication**

Some areas may not be of immediate interest to an operator because they are often used by civilians. The situation of this area is however useful to increase the vigilance-level of an adjacent area that has a restricted access. The situation of adjacent zone thus allows an area to detect an intrusion faster by requiring less intra-area correlation while still limiting false positives. To test the inter-area communication to decrease of the intrusion detection latency we deployed two areas, an area with a restricted access and an area of interest with no access control. The restricted-access area has two sensors, a SPIRIT and simulated seismic sensors. The area of interest has two infrared barriers. At first, a pedestrian needs to cross both sensors in the area of interest to trigger an alarm. But if the pedestrian tripped both sensors of the area of interest, then the result shows that he/she will be detected after tripping only one sensor in the restricted-access area. This result can be seen in Figure 13.

- **Scenario 3: Per-area sensitivity settings**

Areas further from the zone to protect are not as latency-sensitive as areas closer to the zone to protect. The further an area is from the zone to protect, the more correlation it can operate before sending an alarm to the operator to limit false positives. On the contrary, areas closer to the protected zone should require less correlation to lower the detection latency at the expense of false positives. To test different area sensitivities we deployed two areas, an area in the protected zone and an area of interest. The protected area has three sensors: two infrared barriers and a simulated seismic sensor. The area of interest has two simulated seismic sensor and a SPIRIT. A pedestrian crossing the area of interest needs to trip the

two seismic sensors and the SPIRIT before an alarm is sent. In the protected area, the result shows that only two sensors are necessary before an alarm is sent to the operator. This result can be seen in Figure 14.

Figure 13: Testing the inter-area communication to decrease the intrusion detection latency

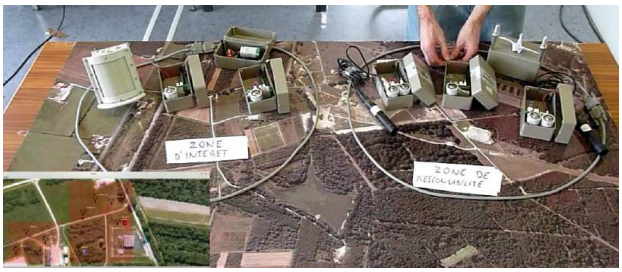


Figure 14: Testing different area sensitivities

- **Scenario 4: In-network network notification**

An operator may be in an area and would like to configure it, read its history or receive alarms when they arrive using a laptop or a tablet. This operator would however not be interested in getting alarms from other areas. To test in-network network notification we deployed two areas both containing multiple sensors. The main operator can access alarms and monitor nodes from both areas using CoAP. The mobile operator only receives alarms from the zone he/she is currently located in. The result shows that when a simulated pedestrian penetrates the first area and trips enough sensors to trigger an alarm only the main operator receives the alarm. But when the pedestrian enters the second area and trips enough sensors, both operators receive the alarm. This result can be seen in Figure 15.

Figure 15: Testing In-network network notifications

- **Scenario 5: Fault tolerance**

Wireless nodes can fail at any time. When they do, the network should also autonomously react to this loss and reconfigure itself to operate properly. This includes finding other communication routes and changing the criticality threshold of the area to react to the loss of the sensors attached to the failing node. Finally, the operator needs to be warned about this failure. To test fault tolerance we only deployed two infrared barriers. One of the barriers (node 3) is connected to the other one (node 2) to reach node 1, the gateway and correlation node. Node 3 cannot reach node 1 and has to go through node 1 or node 7 first. In this scenario an alarm is generated when both sensors are tripped by a simulated pedestrian intrusion. Node 2 is then disabled and another intrusion is simulated. An alarm is triggered after tripping node 3's barrier. This means node 3 reconfigured its route to reach the correlating node by going through node 7 instead of node 2. It also means that the correlation node reconfigured itself to take into account the loss of one sensor. The operator is advertised about the loss of node 2 by turning the node's symbol from black to red. The result can be seen in Figure 16.

(a) Before node 2 failed (b) After node 2 failed

Figure 16: Result of an intrusion before and after node 2 failed

- **Scenario 6: Protecting a civilian farm**

The deployment of the wireless sensor network should be as easily as possible. The final demonstration scenario aims at protecting a farm to alert the police in case of an intrusion. We installed two areas in the vicinity of the farm to be protected. One area containing two infrared barriers is located on the pathway leading to the farm. The second area, in front of the farm, contains one seismic sensor and a SPIRIT. In this scenario we see an operator installing the network by taking advantage of the terrain. The configuration and flashing of the nodes were done off-camera using the Diase and the build network firmware-generation script. Then, a pedestrian runs to the farm along the pathway leading to the farm, tripping both barriers. He then is picked up by the SPIRIT of the second area before being detected by the seismic sensor. By the time he reaches the farm, an alarm is sounded, deterring the intruder. The result can be seen in Figure 17.

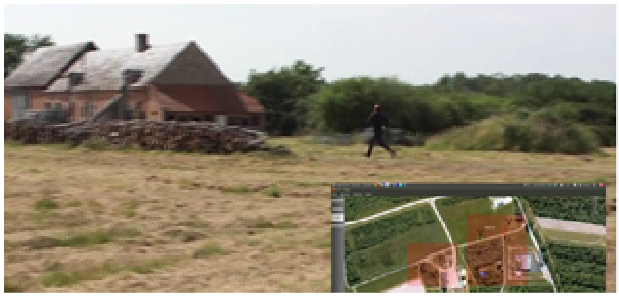


Figure 17: Final deployment of the DIAFORUS system to protect a farm

6. Conclusion and Future Work

In this paper, we proposed a modality-agnostic collaborative detection of spatio-temporally correlated events. The proposed correlation of sensors' values inside the network instead of forwarding sensing data to the gateway allows lowering and evenly-spreading power consumption by localising communication and reducing its amount. Our contribution is an improvement of the already existent collaborative detection approaches. Indeed, our proposition has enabled the reasoning node to correlate heterogeneous sensors while taking into account a possible ill calibration of some nodes. It also allows short-loop automatic responses to events detected in the network, and eases the definition of good auditing capabilities which allows overcoming the debugging problems encountered in decentralised data processing by providing introspection inside the network and nodes. These capabilities could be exploited by both the operator and the network. For example, they allow reaction to faulty sensors, in a true autonomic fashion.

Implementation results show that our proposal is feasible on standard sensor nodes. In addition this is to our knowledge the first real-world deployment which takes into account different realistic scenarios.

In this paper we do not treat the target tracking problem that could be very interesting in this specific context. Indeed, by addressing this issue in the future, we will provide our intrusion detection system with new features, such as the prediction of the intruder final target and the remote actuation of an extra security mechanism (e.g. actuation of locks, destruction of sensitive data, etc.). Future work will focus on improving the reputation management by allowing the WSN operator to report false positives and false negatives. This would increase the accuracy of the sensors' reputation by letting the operator specify when an intrusion happened and the criticality level did not raise enough to reach the threshold. The possibility of improving the criticality threshold computation by taking into account the reputation of each node to make sure the threshold can be reached will also be investigated. A study should also be carried out to evaluate the power consumption cost of adding redundancy for the correlation node. Then, the influence of the sensor density over the number of communications will be studied.

Acknowledgments

This work was supported by DIAFORUS project, funded in part by the National Agency for Research in France - ANR (Agence Nationale de la Recherche).

References

[1] Q. Wang, M. Hempstead and W. Yang, "A realistic power consumption model for wireless sensor network devices," in 2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks, 2006. SECON '06, vol. 1, Sep. 2006, pp. 286–295. 12, 22

[2] J. Koomey, S. Berard, M. Sanchez, and H. Wong, "Implications of historical trends in the electrical efficiency of computing," Annals of the History of Computing, IEEE, vol. 33, no. 3, pp. 46 –54, Mar. 2011. 17, 22, 25 [66] A. Name, "Dissertation Title", M.S. (or Ph.D.) thesis, Department, University, City, Country, Year.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," Computer Networks, vol. 38, pp. 393–422, 2002.

[4] M. Peres, , R. Perier and F. Krief, , Overcoming the Deficiencies of Collaborative Detection of Spatially-Correlated Events in WSN, Advanced Infocomm Technology , Lecture Notes in Computer Science, 2013

[5] K. Phani, A.V.U. Reddy, A.M. D. Janakiram, Distributed collaboration for event detection in wireless sensor networks. In: Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing. p. 1{8. MPAC '05, ACM, New York, NY, USA (2005), http://doi.acm.org/10.1145/1101480.1101491

[6] Y. He, M. Li, and Y. Liu,: Collaborative query processing among heterogeneous sensor networks. In: Proceedings of the 1st ACM international workshop on Heterogeneous sensor and actor networks. p. 25{30. HeterSanet'08, ACM, New York, NY, USA (2008), http://doi.acm.org/10.1145/1374699.1374705

[7] E. Ould-Ahmed-Vall, B. Heck Ferri and G. Riley, Distributed Fault-Tolerance for event detection using heterogeneous wireless sensor networks. Mobile Computing, IEEE Transactions on PP(99), 1 (2011)

[8] C. Zhu; Nicanfar, H. ; Leung, V.C.M. ; Yang, L.T.; An Authenticated Trust and Reputation Calculation and Management System for Cloud and Sensor Networks Integration; IEEE Transactions on Information Forensics and Security, (Volume:10 , Issue: 1); 27 octobre 2014

[9] A. Ukil,; Trust and Reputation Based Collaborating Computing in Wireless Sensor Networks; Second International Conference on Computational Intelligence, Modelling and Simulation (CIMSiM), 2010 ; 28-30 Sept. 2010

[10] C. Alcaraz,; C. Fernandez-Gago, J. and Lopez, An Early Warning System Based on Reputation for Energy Control Systems, Smart Grid, IEEE Transactions on (Volume:2 , Issue: 4), 09 novembre 2011
[11] P.J. Balaji and S. Anandamurugan; Mobility aware reputation node ranking (MARNR) for efficient clustering at hot spots in wireless sensor networks (WSN);International Conference on Information Communication and Embedded Systems (ICICES), 2013 ; 21-22 Feb. 2013
[12] N. Katneni, , V. Pandit, ; L. Hailong and D. P. Agrawal, Hybrid Gaussian-Ring Deployment for intrusion detection in wireless sensor networks; IEEE International Conference on Communications (ICC), 2012
[12] N. Lewis and N. Foukia, An Efficient Reputation-Based Routing Mechanism for Wireless Sensor Networks: Testing the Impact of Mobility and Hostile Nodes; 1-3 Oct. 2008
[13] R. Zheng, C.-H Edith. Ngai and J. Liu, Wireless Sensor Network Deployment in Mobile Phones Assisted Environment, 18th IEEE International Workshop on Quality of Service (IWQoS), 2010
[14] L. Zhang, X. Zhang, J. Yang, G. Chen; Practical Node Deployment for Unique Localization in Large Scale Wireless Sensor Networks, IEEE International Conference on Wireless Communications and Signal Processing (WCSP), 2010
[15] G. Werner-Allen, P. Swieskowski, and M. Welsh, "MoteLab: A Wireless Sensor Network Testbed," Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN), 2005.
[16] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, and D. Culler, "Trio: Enabling Sustainable and Scalable Outdoor Wireless Sensor Network Deployments," Proc. Fifth Int'l Conf. Information Processing in Sensor Networks (IPSN), 2006.
[17] T. He, P. Vicaire, T. Yan, Q. Cao, G. Zhou, L. Gu, L. Luo, R. Stoleru, J.A. Stankovic, and T.F. Abdelzaher, "Achieving Long-Term Surveillance in VigilNet" ACM Trans. Sensor Networks (TOSN), vol. 5, no. 1, pp. 1-39, 2009.
[18] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: Out-of-the-Box Environmental Monitoring," Proc. Int'l Conf. Information Processing in Sensor Networks (IPSN), 2008.
[19] L Yunhao and Al. Does Wireless Sensor Network Scale? A Measurement Study on GreenOrbs, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 10, OCTOBER 2013
[20] P.K. Biswas; Avaya Labs., Lincroft, NJ ; Phoha, S.; Self-organizing sensor networks for integrated target surveillance; Computers, IEEE Transactions on (Volume:55, Issue: 8); Aug. 2006
[21] Y. Wang, Y. K. Leow, J. Yin; Is Straight-line Path Always the Best for Intrusion Detection in Wireless Sensor Networks; 15th IEEE International Conference on Parallel and Distributed Systems (ICPADS), 2009
[22] Y. Wang, W. Fu and D. P. Agrawal; Intrusion Detection in Gaussian Distributed Wireless Sensor Networks; IEEE 6th International Conference on Mobile Adhoc and Sensor Systems, 2009. MASS '09.
[23] D. Brunelli, Minakov, I. ; Passerone, R. ; Rossi, M., POVOMON: An Ad-hoc Wireless Sensor Network for indoor environmental monitoring, IEEE Workshop on Environmental Energy and Structural Monitoring Systems (EESMS), 2014
[24] J. Harrison, "A short survey of automated reasoning," in Proceedings of the 2Nd International Conference on Algebraic Biology, ser. AB'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 334–349. [Online]. Available: http://dl.acm.org/citation.cfm?id=1769026.1769050 26
[25] F. Portoraro, "Automated reasoning," in The Stanford Encyclopedia of Philosophy, summer 2014 ed., E. N. Zalta, Ed., 2010. [Online]. Available: http://plato.stanford.edu/archives/sum2014/entries/reasoning-automated/ 27
[26] R. Baldoni, , L. Querzoni,., and A. Virgillito, (2005). Distributed event routing in Publish/Subscribe communication systems: a survey.
[27] A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," Computer Communications, vol. 30, no. 14–15, pp. 2826–2841, Oct. 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366407002162 31.
[28] B. Frank, Z. Shelby, K. Hartke, and C. Bormann, "Constrained application protocol (CoAP)," Oct. 2010. [Online]. Available: https://tools.ietf.org/html/ draft-ietf-core-coap-03 16, 33, 36
[29] T. Winter, "RPL: IPv6 routing protocol for low-power and lossy networks," Mar. 2012. [Online]. Available: https://tools.ietf.org/html/rfc6550 15, 23, 36
[30] Cortus, "Cortus - APS3r," 2010. [Online]. Available: http://www.cortus.com/ aps3r.html 187
[31] Wave2m community, "Wave2m - the platform in-depth," 2008. [Online]. Available:http://www.wave2m.org/the-specification/the-platform-in-depth?view=item 11, 12, 188

Maïssa Mbaye received a Master Degrees (Distributed Systems, Computer Network and Parallelism) at 2006 from University of Bordeaux 1 (France). In 2009 he obtained his PhD degree from University Bordeaux 1. Currently, Mr. Mbaye is Assistant Professor at University Gaston Berger (Senegal) and member of LANI (Laboratoire d'Analyse Numérique et Informatique). His research interests include distributed systems, cloud computing, Internet of Things, and security. He is co-founding member of the Senegalese Association of researchers in Computer Science. He is also the Vice Chairman of the Scientific Committee of the National Conference on Research in Computer Science and its Applications.

Mohamed Aymen Chalouf has obtained his Computer Science Master (University of Bordeaux 1) and his Telecommunications Engineering Technology Diploma (ENSEIRB: Ecole Nationale Supérieure d'Electronique, Informatique et Radiocommunications de Bordeaux) in 2006. In December 2009, he received his Ph.D. from the University of Bordeaux 1. Currently, M. A. Chalouf is Asssociate Professor at the University of Rennes 1 (IUT of Lannion) and member of the IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires) Laboratory, «Networks, Telecommunications and Services» department (D2). His research interests include the management of quality, security and energy in new network architectures, systems and services. His research has produced a number of publications in international journals and conferences.

Francine KRIEF obtained the HDR degree (Habilitation à Diriger des Recherches) at University of Paris 6 on Context-aware Management, in December 2003. Currently, she is Professor at Bordeaux-INP and member of CNRS LaBRI Laboratory, UMR 5800, “Programming, Networks and Systems” team. Her main research activities concern self-management for wired and wireless networks, end-to-end signaling protocols and green networking. Her work on network and service management has led to many publications in journals and at conferences.