C. Alcaraz, C. Fernandez-Gago, and J. Lopez, "An Early Warning System based on Reputation for Energy Control Systems", *IEEE Transactions on Smart Grid*, vol. 2, pp. 827-834, 2011.
http://doi.org/10.1109/TSG.2011.2161498
NICS Lab. Publications: https://www.nics.uma.es/publications

An Early Warning System based on Reputation for Energy Control Systems

Cristina Alcaraz, Carmen Fernandez-Gago, and Javier Lopez

Computer Science Department, University of Malaga,

Campus de Teatinos s/n, 29071, Malaga, Spain

{alcaraz,mcgago,jlm}@lcc.uma.es

October 29, 2015

Abstract

Most of energy control or SCADA (Supervisory Control and Data Acquisition) systems are very dependent on advanced technologies and on traditional security mechanisms for protecting the a system against anomalous events. Security mechanisms are not enough to be used in critical systems, since they can only detect anomalous events occurring at a certain moment in time. For this reason it becomes of paramount importance the usage of intelligent systems with capability for preventing anomalous situations and reacting against them on time. This type of systems are, for example, Early Warning Systems (EWS). In this paper, we propose an EWS based on Wireless Sensor Networks (WSNs) (under the ISA100.11a standard) and reputation for controling the network behaviour. The WSN are organized into clusters where a Cluster Head (CH) is designated. This CH will contain a *Reputation Manager Module*. The usability of this approach is also analyzed considering a Smart Grid scenario.

Keywords: Early Warning Systems, Wireless Sensor Networks, Reputation, SCADA Systems, Smart Grid

1 Introduction

According to the conceptual model described by NIST in [1], part of the functionality of a Smart Grid is associated to the remote control of power substations that ensure an efficient energy generation and distribution to residential complexes. The control is basically managed by a centralized system known as SCADA (Supervisory Control and Data Acquisition) system. Although this control form a unique entity, it has a special influence on the rest of domains by supervising operational activities and residential complexes.

However, this type of interaction among domains could trigger in serious consequences when part of the control and substations are disrupted. Then,

it is crucial to address problems of the current SCADA systems to strengthen its architecture, and thus to ensure a reliable and survivable Smart Grid. In fact, most of the security mechanisms applied in SCADA systems are not based so far on dynamic and autonomous procedures where different entities with different complexities are integrated in the same context. It is necessary to provide advanced security mechanisms to individually protect domains, and thus protect the rest of the system as a whole. We agree with [2] that part of this security must be focused on the prevention of anomalous events and the preview of any significant change that can cause a cascading effect over other domains of the Smart Grid or over other Critical Infrastructures (CIs) [3]. For this reason, we propose an intelligent Early Warning Systems (EWS) able to protect the substations as independent subdomains, but indirectly protecting the rest of domains of a Smart Grid when serious failures occur.

Unfortunately, up to date there are not specialized EWSs for critical environments and therefore researching in this new area is very much needed in order to protect the system against anomalous events, failures or threats. The approach we present in this paper tries to fill this lack. We introduce a model of an EWS that covers this security aspect for energy control systems. The approach is based on Wireless Sensor Networks (WSNs), able to perceive the real state of the infrastructure, on reputation to control the real behaviour of the control network and on the ISA100.11a standard [4] for an efficient alarm management. Moreover, the approach is able to detect specific anomalous behaviours from sensor nodes, and in particular behaviours associated to compromised situations, such as a replay or a delay attack.

The paper is organized as follows. Section 2 analyzes the need of EWSs in highly-critical energy scenarios and it identifies the main components of them. Section 3 provides a background about WSN and reputation systems in WSNs and critical infrastructures. Section 4 presents the general architecture of our model, which includes a set of components, together with their respective modules and functionalities. In Section 5 our model is analyzed in a critical application context, such as a Smart Grid and in Section 6 implementation details of the model are given. Finally, Section 7 concludes the paper and outlines the future work.

2 Early Warning Systems in the Cascading Effect Control

The protection of critical infrastructures in our society such as energy generation and distribution systems has been proven of paramount importance. Some national and international action plans and initiatives have already been proposed in order to discuss some security issues related to Critical Infrastructures Protection (CIP). Even though modelling and simulation techniques are feasible mechanisms to visualize the connectivity among systems and assess risks, they are not enough to ensure an effective control of a cascading effect. It

would be desirable to *predict and anticipate anomalous situations, even before they happen* in order to react against them, reducing as much as possible risks and efforts in recuperation processes. These can be solved with Early Warning Systems (EWSs), specialized security mechanisms that aid the protection of highly-critical environments in early stages.

Unfortunately, EWSs are not yet considered by today's control industry. Its security currently depends on security policies, access control mechanisms, security applications and specific detection systems, such as firewalls or Intrusion Detection Systems (IDSs). At this point, we may think that an EWS is apparently a security mechanism similar to an IDS, since both of them can detect anomalous situations through patterns/rules. However, there is an important difference between them. An IDS is only able to detect existing anomalous events, whereas an EWS is able to predict and warn about them. An EWS consists of an advanced monitoring component based on integrated techniques that analyze and interpret data streams from distributed sensors in remote energy substations. These techniques also include decision making procedures to avoid or reduce the propagation of a possible effect originated by an anomalous event [5]. To be more precise, the idea is not to precisely detect an existing failure/threat and subsequently correct it. The success of an EWS depends on the ability to anticipate an event sequence, as well as facing an anomalous situation to control the effect over the system or systems.

Four main components constitute any EWS: (i) a *detection component* represented by sensorial nodes, (ii) a *reaction component*, (iii) an *information recollection component* to store evidences, and (iv) an *alarm management component*. The reaction component includes a process of decision making whose determination will depend on the type of threat, the criticality of the affected environment, the interaction with other involved elements, the associated risk and the damage-cost relationship. In any case, all of these components have to be active before, during and after a failure/threat appears in the system. 'Before' in order to anticipate and warn about a set of suspected actions, 'during' in order to avoid that an effect starts to propagate itself in the system, and ' after' in order to control that such effect can be propagated towards other systems.

3 Background and State of the Art

3.1 WSN, a Control, Reaction and Warning Component

As mentioned earlier, an EWS has four essential components associated to control/detection, reaction, recollection and warning tasks. A technology capable of offering all of these services is precisely a WSN. Its nodes (called sensor nodes) are able to monitor, detect, track and alert anomalous situations thanks to its adhered sensors capable of sensing physical events from their surroundings, as well as its computational capabilities. They can also collaborate among them in order to achieve a common goal (e.g., control of energy generators), in addition to being self-configurable, self-healing and smart devices. This type of

self-configurability allows sensor nodes to adapt by themselves in the network and react against failures, whereas self-healing provides them with capabilities for facing unexpected network events.

WSN is currently considered as one of the most demanded wireless technologies by the control industry and Smart Grid, since it guarantees the same control services as a RTU (Remote Terminal Unit) but to a low installation and maintenance cost [6]. Even this demand has implied the recent standarization of their comunications with ZigBee PRO [7], WirelessHART [8] and ISA100.11a. In this paper we will mainly focus on ISA100.11a, since it is an extended version of WirelessHART and it improves some of its services [6].

ISA100.11a allows both mesh and star topologies using: (i) sensor nodes (working at 26MHz, RAM 96KB, 128KB Flash Memory and 80KB ROM), (ii) routers, (iii) gateways (one or several working at 533MHz, 64MB RAM and 8MB Flash Memory) to establish redundant connection with the SCADA centre, (iv) backbone routers to provide connectivity to other networks, and (v) two special managers: a system manager and a security manager. The system manager is in charge of allocating resources and providing communication, whereas the security manager affords key management services. Moreover, ISA100.11a is based on the IEEE 802.15.4-2006 standard, which specifies the physical (PHY) and Media Access Control (MAC) layers for Wireless Personal Area Networks (WPANs), providing it with security mechanisms based on AES-128 bits, Message Authentication Codes (MAC) and an Access Control List (ACL) to authenticate any received message. In addition, the standard provides security at link and transport level using Message Integrity Codes, and unique symmetric keys of 128-bits for solving confidential issues.

Lastly, ISA100.11a offers a set of services for guaranteing reliability of communications, diagnosis, and alert and priority management. Specially, this priority management depends on four subcategories (a device diagnostic, a communication diagnostic, a security alert and a process alarm) and on five priority levels (journal [0-2], low [3-5], medium [6-8], high [9-11] and urgent [12-15])). With respect to the dissemination from sensors is managed through objects using DMAP (Device Management Application Process). DMAP is a class individually installed in each network device that includes a set of objects used for configuring, supervising and requesting parameters belonging to sensor nodes. More precisely, DMAP contemplates the ARMO (Alert Reporting Management Object) class for managing, at first level, alerts and generating reports through an AlertReport service to ARO (Alert Receiving Object). ARO is a class configured in a unique device in the network (the gateway in our case). All of these alert management objects will be discussed in more detail throughout this paper given that they play an important role in our proposal.

3.2 Reputation and Trust Management for WSNs

Reputation and trust are related concepts, however they have different meanings. Reputation is defined by the Concise Oxford Dictionary as 'what is generally said or believed about a person or the character or standing of a thing'

while trust is defined as 'the firm belief in the reliability or truth or strength of an entity'. From these definitions we can infer that the concept of reputation is more objective compared to the concept of trust.

For assuring a successful collaboration, a node should be able to discover which neighbouring nodes are more likely of accomplishing a certain task. If a node knows in advance how the different elements of the network will react in any situation, then it will be able to make a flawless decision. However, in a WSN the outcome of a certain situation cannot be clearly established or assured. That is, we need to take uncertainty into account. Trust and reputation systems aid dealing with the problem of uncertainty. In a collaborative environment such as WSN by determining which is the best node to collaborate with becomes very important. It is then when reputation plays a key role. By knowing the reputation of nodes in their neighbourhood and their actual behaviour, it is possible for the nodes to choose a suitable course of action when making operational decisions (knowing which is the best partner for starting a collaboration or in extreme situations, e.g. malfunctioning nodes).

The development of trust and reputation management systems for WSN is in a very early stage of research although growing rapidly in the past few years. Most of the existing works are proper of Ad-Hoc and P2P networks [9]. However, these systems do not fit all the requirements and features required by WSN. As mentioned, this research area is becoming very active and several surveys have been produced [10]. Still, many of the solutions are designed with the purpose of solving very specific problems and most of them do not deal with all the features that a trust management system for WSN should provide although they have been outlined in [11]. When designing a trust or reputation system for WSN important aspects that should be taken into account are the sources of information and the way of computing and modelling reputation or trust.

Reputation provides some interesting benefits to trust management systems (e.g. better management of aspects such as aging). Some authors have proposed reputation-based frameworks where nodes maintain reputation for other nodes and use it in order to evaluate their trustworthiness. Probabilistic functions are widely used as it is the case for [12].

The approach we are going to consider for the design of a reputation system for EWS is the use of clusters. In fact, some sensor networks group their nodes into clusters for various reasons (e.g. better energy management or use of more powerful nodes to execute complex tasks). The cluster head (CH) is in charge of gathering and analysing reputation values of the nodes in its cluster and making event decisions. This is the case, for example, of TIBFIT [13], a protocol that aims at detecting arbitrary node failures in an event-driven WSN where the nodes are organized into clusters. Still, not all cluster-based systems place the trust entity only in the CH, such as for example [14].

4 An Early Warning System Model based on Reputation

4.1 The Architecture of the Model

EWS could provide a good support for SCADA systems, as the actions of the latter might be influenced by the information provided by the EWS. Therefore, it is desirable to provide EWS with techniques that lead towards a good performance of the SCADA systems. Our intention is to provide a reputation mechanism within the EWS in such a way that facilitates the decision making process once a warning (i.e., a SCADA alert) has reached the SCADA system.

The general architecture of our model can be seen in Figure 1. Due to the energy and resource constraints of sensors in general, organizing sensor networks into clusters is a good approach for managing the existing resources. This hierarchical configuration will allow us to select a *cluster head* (*CH*) in every cluster, which will be a trustworthy node with enough resources to take up essential functionalities of the EWS. In particular, it is going to include the *Reputation Manager* (RM in the following, see Figure 2). The RM will provide the gateway with information about the nodes in the EWS. The gateway contains an *Alarm Manager* (AM in the following). This manager is in charge of dealing with the alerts received from the *CH* about nodes in the EWS in order to determine the level of criticality of such an alarms. The AM sends to the SCADA center a SCADA alert in order to be properly treated and registered (accounting), while the most suitable operator in the affected area is also being located. When the incidence has been finally treated by the selected operator, he/she has to send a report to the gateway to be later on re-sent to the corresponding *CH*. Depending on the report, this last one will update the reputation of the sensor node accordingly.

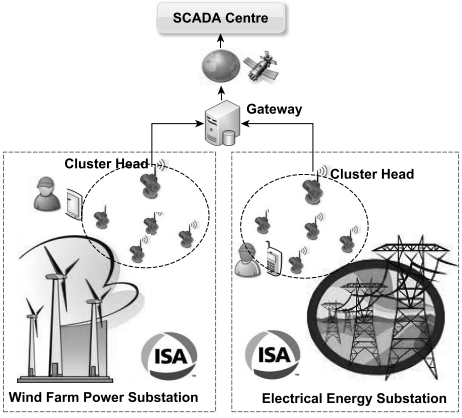

Figure 1: General Architecture of the Model

4.2 The Cluster and its Components

Figure 2 represents the components integrated into the *CH* needed for implementing our EWS. At first, nodes, s_i, in the cluster send their messages (r_i (e.g. voltage streams) and a_i (e.g. anomalous events)) to the *CH* which first operates the *Message Normalization* component to combine combine and represent different data inputs in a same generic format. The result of such a normalization is then sent to the *Pattern Association* component in order to verify the nature of such a message. This component must be based on simple patterns due to high constraints of the networks devices, such as for instance values (readings) out/in of a specific threshold (e.g. minimum and maximum values for voltage, $[V_{min}, V_{max}]$), as well as delayed messages circulating in the network and/or replay attacks. Note that lost messages that come from the same node can also mean a possible delay attack. All of these situations and any alert generated by a sensor node (such as circuit break, stresses, strong fluctuations, routing, etc.) have to go through the RM in order to aid the whole system to verify the real state of the affected area.

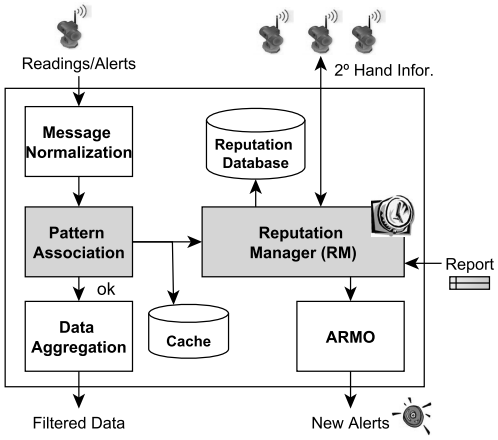

Figure 2: Architecture of the Cluster Head

These components have been integrated into the *CH* because of three main reasons. Firstly, the control in these types of subnetworks is generally simplified to a small number of nodes in order to reduce the memory storage with information associated to the subnetwork. Secondly, we believe that if traditional sensors (working at 4-8MHz, 4-16KB RAM and 48-128KB flash memory) are able to work as *CHs*, our *CHs* with higher capabilities (working at 26MHz, 96kB RAM, 128KB serial flash memory, and 80KB ROM) are equally feasible to process, calculate and store information. Finally, part of the processing is straightforward, since the approach has been designed for specific situations using basic behaviour patterns and the calculation of the reputations values is

simple. Lastly, and given the dimension of our approach, we assume that the entire process of key management and agreement between peers properly work using the security mechanisms offered by the ISA100.11a. as well as authentication and integrity services.

In the following we will concentrate on the functionalities of two of the main components: the Pattern Association and the Reputation Manager.

4.2.1 Pattern Association Component

The validation process of a normalized message basically consists of checking if the received message is really unique and recent. In order to achieve this, a "cache memory" is needed (see Figure 2). This memory allows the system to check if a new message received from a sensor node, s_i was already received recently. In order to carry out an efficient search in the cache the entries have to be ordered by a specific and unique data, such as for instance the timestamp corresponding to the received messages. It would also be desirable that the data stored in the cache memory are relevant and simple information, at least, the identification of the node s_i, IDs_i, the timestamp of the message and the value associated to the message (i.e., a r_i, or a_i).

Another important task of the Pattern Association component is also to verify that the messages were received in a valid time period, i.e. in an *expected time* according to their timestamp. This time is calculated as the difference between the real time of the *CH* (T_system) and the timestamp of the message, such that the result (T_time) <= (T_{MAX}) a configurable value according to its security policies. In case where T_{time} is within the expected time interval, the component has to check in the cache memory if other recent message was already received from the same node s_i. The cache memory is ordered by the timestamps what makes more efficient the search as it can be achieved by tracking it using as primary identifiers the timestamp and the IDs_i. For example, when the timestamp of the message is greater than the timestamp of an entry in memory, the search can be stopped in order to reduce computational overhead.

The next step is to verify the message nature or suspect actions such as lost of messages, delay attacks or replay attacks. For the analysis, the new alert will have to include, at least, the IDs_i, the type of the alert and its priority, as well as the type of detected event, which could be categorized as follows: *event_alert* (any a_i received from a s_i), *event_reading_out_threshold* (a $r_i \notin [V_{min}, V_{max}]$), *event_lost_message*, *event_delay_attack*, *event_replay_attack* and *event_discard_node*.

In order to control delay attacks, for each entry in the cache memory a new column is also included storing in it two possible labels: *event_lost_message* / *not_event_lost_message*. When the component receives a lost message from an IDs_i, it checks in cache if such a node did not send another lost message recently by using the label *event_lost_message*. This label is included in the memory for the first time when a specific message is detected as a lost message. On the contrary, when the component receives an a_i alert or a valid r_i (i.e., $r_i \in [V_{min}, V_{max}]$) the component must also store them in the cache using the label

not_event_lost_message for future queries. In the particular case where a reading is valid, it must be filtered and aggregated by the *Data Aggregation* component to be sent it to the gateway later on. This last action is the main task of a *CH*. In the following, we will formally describe by means of a pseudocode our model, previously defining the functions used.

- $Obtain\_Normalized\_Message$: it obtains normalized messages from the Message Normalization component.
- $Obtain\_ID\_Node$: it obtains the IDs_i.
- $Exist\_Memory$: it tracks the cache in order to find an existing entry.
- $Exist\_Memory\_LostMessage$: it tracks the cache to check if an IDs_i corresponding to a node s_i already sent an *event_lost_message*.
- $Send\_RM$: it sends the normalized message to the RM in order to validate the state of the node and the type of detected event.
- $TReading$: it verifies if the message is a r_i.
- $TAlert$: it verifies if the message is an a_i.
- $Store\_Cache$: it stores the messages in the cache and their corresponding label (i.e., either *not_event_lost_message* or *event_lost_message*).
- $Send\_Message\_DataAggregation$: it filters and aggregates data streams to be sent to the gateway.
- $Confirmation\_ACK(ID_{s_i})$: it confirms the reception of the message to the IDs_i with an ACK (acknowledge packet).

Then,

```
message = Obtain_Normalized_Message();
T_time = T_system − TSTAMP(message);
if (T_time <= T_MAX) then
    if (Exist_Memory(message)) then //An expected time, T_time ≤ T_MAX
        Send_RM(message, event_replay_attack); //a possible replay attack
    else
        if ((TReading(message) AND (Obtain_Data(message) ∉ [V_min, V_max]))) then
            //a r_i ∉ [V_min, V_max]
            Send_RM(message, event_reading_out_threshold);
        else
            Store_Cache(message, not_event_lost_message);
            ID_{s_i} = Obtain_ID_s_i(message);
            Confirmation_ACK(ID_nodo);
            if (TAlert(message)) then
                //storage of alert, ACK and its transfer to the RM
                Send_RM(message, event_alert);
            else
                //storage of a reading, ACK and aggregation
                Send_DataAggregation(message);
            end
        end
    end
else
    //The message was received in an unexpected time.
    //This may mean : (i) a lost message or (ii) a possible delay.
    ID_{s_i} = Obtain_ID_Nodo(message);
    if (Exist_Memory_LostMessage(ID_{s_i})) then
        Send_RM(message, event_delay_attack);
```

```
    else
        Store_Cache(message, event_lost_message);
        Send_RM(message, event_lost_message);
    end
end
```

The effectiveness of this approach for detecting delay and replay attacks will depend on the size of the cache. In addition, the control of delayed or resent messages from a compromised node simplifies the computational cost inverted in the aggregation tasks. This means that the *CH* is able to filter and aggregate data streams from different nodes, but not repeated packets from a unique node.

4.2.2 Reputation Manager

In this section we will introduce how reputation can be included as an input for the decision making process for a WSN, part of an EWS. Our intention is to design a *Reputation Manager* (RM) for aiding the *CH* determining which nodes in its cluster are not functioning properly or are acting in a compromised way in order to leave them out of the network for certain actions. The *CH* is responsible of maintaining the RM.

After the information has gone through the Message Normalization and the Pattern Association components (see Figure 2), the following parameters are required by the RM: $(IDsi, r_i, a_i, rep_{s_i}^{t_j}, type\_of\_event\_detected)$. Part of this information (such as $IDsi$, r_i or a_i and type of event) is obtained from the information passed by the Pattern Association component. $rep_{s_i}^{t_j}$ is acquired from a simple reputation database. Concerning the parameter a_i the most interesting information contained in it (e.g., alert type, alert category and alert priority) for reputation purposes is the alert priority (low, medium, high, urgent, etc.). In contrast, $rep_{s_i}^{t_j}$ is the reputation parameter of a node s_i at time t_j, and it is calculated by a reputation engine inside the RM. The entries that this engine takes into account in order to calculate reputation values are initial reputation, observations gathered by neighbour nodes and second hand information obtained from different iterations. The parameter t_j is a timestamp representing the cycle of time that passes by in between the node s_i emitted an alert (and reported it to the RM) until the reputation of the node is updated (see Section 4.4).

Initially, all the nodes in a cluster behave in a trustworthy way and therefore their initial value of reputation is the highest possible value. After the first iteration, when nodes come into play, the RM can update the reputation values for each of them. The RM generates a new alert using the ARMO class defined by the ISA100.11a standard in order to send to the gateway the IDs_i, the type of occurred event, its priority and the timestamp t_j. It is important to highlight that depending on the type of event, the RM has to determine the criticality of it. In particular, events associated to an *undesirable minimum reputation value* (it corresponds to an *event_discard_node*, see Section 4.4), a r_i *out of the threshold* (an *event_reading_out_threshold*) and a *compromised node* in the network (an *event_replay_attack* or an *event_delay_attack*) must be labelled as an "urgent" alert, since these cases represent serious situations that have to be attended on

time. Furthermore, as the events *event_replay_attack*, *event_delay_attack* and *event_discard_node* are associated to compromised behaviours of sensor nodes, a replace/reconfiguration procedure of affected nodes is required in the network. Only two of them are really managed by the RM. They are *event_alert* and *event_reading_out_threshold*. A register or warning of anomalous situations is needed for the rest of the events.

Four situations may arise in our critical context: 1) *false positive* if the analyzed event by a node is innocuous, but it is classified as a threat (failure/-compromised node); 2) *true positive*, if the analyzed event is properly classified as a threat (failure/compromised node); *false negative*, if the analyzed event is a threat (failure/compromised node) but the node classified it as normal/innocuous; and *true negative*, if the analyzed event is correctly classified as normal/innocuous. The levels of importance of these cases will serve as a basis for updating reputation values by the RM (see Section 4.4).

4.3 The Gateway and its Components

The next step of our model is to analyze any type of alert received from *CHs*. The analysis must be managed by the ARO class which is configured in the gateway and uses an organized queue ordered by priorities [4], and for each priority is used a buffer with a maximum size (see Figure 3). Depending on the queue and its priorities, two tasks can concurrently be executed: (i) to send the alert to the SCADA central system, (ii) to activate the *Operator Location* component when an alert is really critical. The former task involves registering r_i and a_i occurred in the system, since they must be stored by the SCADA central system. As the information has to be transmitted to the SCADA system through specific SCADA protocols under TCP/IP (e.g. Modbus/TCP, DNP3 or IEC-104), the gateway must also act as a special interface with capability for understanding and translating messages between different systems. For instance, IEEE 802.15.4 messages to Modbus/TCP messages and viceversa. In contrast, the latter task involves treating any type of critical incidence (i.e. high/urgen a_i) and react against them on time. It also involves location and warning to the closest field operator so that he/she can approach the affected area and check the real nature of the situation as soon as possible. To this end, the operator must previously know the IDs_i and its localization, Loc_i, in the area as well as the occurred event by using a handheld device.

As ISA100.11a defines statical networks, the Loc_i can be obtained from a simple and local database configured in the gateway. Note that this database must store, at least, the IDs_i and the Loc_i. This type of design does not only allow locating the node in the affected area in a short time, but also reducing computational and memory overhead in *CHs* as well as communication overhead if Loc_i was maintained by clusters. The location of operators is outside of the scope of this paper. However, some attempts have been made in order to find the most suitable operator in [15] based on reputation.

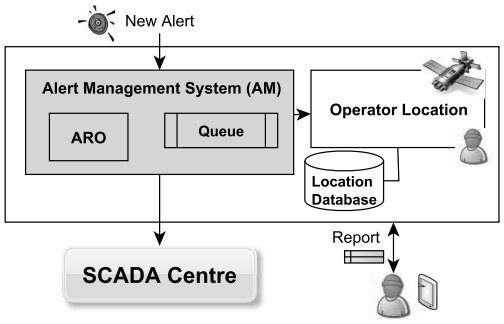

Figure 3: Architecture of the ISA100.11a Gateway

4.4 Updating Reputation

After the alerts with types *event_alert* and *event_reading_out_threshold* are treated by the AM, a report is sent back to the RM regarding how accurate the estimation of the type of alert priority given by a node was. This report will be used in order to update reputation. When an alert reaches the SCADA center and an operator deals with it, he can deliver a report about the behaviour of the node. The operator is who determines the real priority of the alert. Let the priority of the alert given by the operator, $al_{prio(op)}$. Then, two cases arise:

- The operator considers that the type of the alert priority was as critical as the node determined. This means the node behaved in the right way and therefore, the operator will report this fact to the gateway. The gateway will inform the RM which then will increase the reputation of the node.

- The operator determines that the type of alert priority that the node initially reported is not as high as the node estimated or it was even higher than the node estimated. Then, this report is sent back to the RM which will decrease the reputation of the node.

An important issue is how to decrease or increase the reputation values of a certain node, s_i. This will depend on the alert type priority the node reported and what the operator, who is treating the alerts, determined about this alert.

When the reputation of a node reaches a certain threshold set by the system as the minimum possible reputation value, the node is discarded as useless for the cluster purposes. Let this minimum threshold be denoted by rep_{min}. If for any node s_i in the cluster, $rep_{s_i}^{t_j} \leq rep_{min}$ then the node is discarded from the cluster and no readings coming from it will be taken into consideration any longer.

The algorithm for updating reputation values for nodes is as follows.

Let us assume the number of nodes in the cluster is n and s_i is a sensor node in it, where $1 \leq i \leq n$. Let the alert priority reported by the node be denoted

by $al_{prio(s_i)}$ and the alert priority reported by the operated be $al_{prio(op)}$ (see Section 3.1 for the priority levels schema).

- At the initial time, t_1, $rep_{s_i}^{t_1}$ is the maximum possible reputation to be given by the RM.
- For $j = 2, ...n$ at time t_j the RM has received a report back from the gateway stating how accurate the alert estimated by the node was. Then,
  - If $al_{prio(s_i)} \in [12-15]$ and $al_{prio(op)} \notin [12-15]$ then a false positive is produced and the RM decreases the $rep_{s_i}^{t_{j+1}}$ as $w_{ial} rep_{s_i}^{t_j}$, where w_{ial} is a weight that depends on the $al_{prio(op)}$. If $al_{prio(op)} \in [9-11]$ then the used weight should produce a reputation value that is decreased with respect to the previous value. In case that the $al_{prio(op)} \in [6-8]$, i.e., it is medium, the reputation value should decrease more than in the case where $al_{prio(op)} \in [9-11]$. For the other two cases where $al_{prio(op)} \in [3-5] or [0-2]$ then the reputation value is decreased more respectively.
  - If $al_{prio(s_i)} \in [9-11]$ but $al_{prio(op)} \notin [9-11]$ then as it happened in the previous case the reputation of the node is decreased by applying a weight w_{ial}. In this case if $al_{prio(op)} \in [12-15]$ a false negative is produced and therefore the reputation of the node should decrease more by using an appropriate weight than the one used when $al_{prio(op)} \in [6-8]$ where a false positive is produced.
  - If $al_{prio(s_i)}$ is in the same interval of level priority than $al_{prio(op)}$ then the reputation is increased by choosing an appthe weight w_{ial} should be chosen in such a way that the resulting reputation value increases.
  - If the $al_{prio(s_i)}$ is journal, low or medium at time t_j no action is taken but the RM keeps this record in a temporal buffer. If at consecutive instances in time, i.e., t_{j+1}, $t_{j+2}, ..., t_{j+k}$, the same node keeps producing the same type of alerts then a new message is generated by the RM that informs about this situation and it is sent to ARMO. The kind of message sent will inform the operator that the alert produced by node s_i should be checked. The operator then will determine whether the alert is a false negative or a false positive and then the RM will act as in the previous cases.
- The process continues until $rep_{s_i}^{t_j} \leq rep_{min}$. Then, the node is discarded. In order to warn about this situation, the RM has to generate another new alert of type *event_discard_node* with "urgent" priority.

5 An Application Case Scenario: Smart Grid

Given that an energy control system is part of a Smart Grid, we can apply our approach to such a scenario. The idea is to anticipate a response against

a possible anomalous event in order to reduce as much as possible its negative effect over the other parts of the system as a whole. To this end, let us assume that the model presented in Section 4 is integrated into an electrical remote substation, which is the main unit responsible for supervising, at first level, the energy generation and its distribution to large residential complexes. The cluster heads, CH_i, are able to identify five different cases:

1. *Case 1*: A CH_i receives an alert from a sensor, s_i. For example, a neighbour s_j is not forwarding packages interrupting thus the communication within the network.
2. *Case 2*: A CH_i receives a r_i of voltage with value v, from s_i and $v \in [V_{min}, V_{max}]$.
3. *Case 3*: A CH_i receives a r_i with value v, from a s_i and $v \notin [V_{min}, V_{max}]$.
4. *Case 4*: A CH_i receives the same message from a s_j several times given that this node is a compromised node trying to carry out a replay attack. Note that a delay attack is very similar to this particular case.
5. *Case 5*: A CH_i receives a message after T_{MAX} from a s_i. This means that the message was lost in the network and it was received in an unexpected time.

We will next analyze the above cases and the behaviour of the components involved in the EWS.

1. *Cases 1 and 2*. These cases correspond to an alert and a normal situation, respectively. Here, the Pattern Association component analyzes if the message delivered by s_i is a unique and recent message. Otherwise, the component will have to validate if this unique message is an alert or a valid reading. We will see the differences in between these two cases next.

   If the message is an alert then the Pattern Association component performs as described in Section 4.2.1 and send it to the RM, which realizes the operations detailed in Section 4.2.2.

   If the message is a correct measure of voltage then the Association Pattern component has to (i) store it in the cache memory for recent and future analysis of events and (ii) send such a reading to the Aggregation component in order to aggregate a set of readings to the gateway.

2. *Case 3*. An anomalous situation happens: the Association Pattern component analyzes if the voltage reading is a unique and recent information from a s_i by using the information stored in the cache memory. In this case as $v \notin [V_{min}, V_{max}]$ then the system determines that the anomalous situation is possibly a circuit break. All these information is forwarded to the RM in order to validate the alert in an appropriate way. Then, a new alert of type *event_reading_out_threshold* is generated by the RM and set as "urgent". Again, as in the previous cases, the reputation of nodes is updated as described in Section 4.4.

3. *Cases 4 and 5.* These two cases correspond to replay attacks and lost of messages in the network, respectively. The Pattern Association component analyzes that the messages are unique and recent (Case 4) and that the timestamp of the message is not over T_{MAX} (Case 5). If the message is not unique then it is forwarded to the RM in order to generate a new alert of type *event_replay_attack* with "urgent" priority. In contrast, Case 5 is associated to finding an entry in the cache memory with the IDs_i and a label *event_lost_message*. If it is not found, it must be stored in cache with label *event_lost_message* by the Pattern Association component. Then, it is transferred to the RM in such a way that a new alert of type *event_lost_message* with "low" priority is generated. As this case is not really critical, this type of situation does not require a reputation update process, but a register process in the SCADA central system for future auditing or maintenance procedures. In case that this action is repeated several times, a field operator will have to replace/reconfigure the node.

6 Implementation Details and Expected Results

At present, we are implementing our model using the "de facto" standard Operative System for sensor nodes called TinyOS, which provides limited support for network and protocol simulations. For this reason, we are going to extend the simulations to a proprietary testbed architecture with support for NesC (component-based C-dialect [16]) and Java. This will allow us to provide several experiments considering diverse (either extreme or non-extreme) situations.

Although, we expect to obtain tangible results in the very near future, we believe that a first approximation of the results will be as follow:

1. *Fast response and protection* to the rest of domains of a Smart Grid. A hierarchical configuration allows the system to quickly locate an affected area and respond on time.

2. *Safety and Security.* Safety in the control of the cascading effect. This control is based on a detection and reputation mechanism and on the use of a smart AM. With respect to security, the system is able to identify compromised nodes by means of reputation values.

3. *Performance.* The EWS ensures performance since part of its logic is configured in the cluster heads and the gateway, both of them with enough resources to carry out their tasks of detection and alarm management, respectively.

4. *Adaptability.* Our model can equally work in an ISA10011.a, a ZigBee PRO and a WirelessHART network, since all of them share certain topological, structural and functional characteristics.

5. *Auditing and maintenance.* The system is able to ensure maintenance by identifying compromised nodes, in addition to offering relevant information to carry out efficient auditing procedures.

7 Conclusions and Future Work

We have proposed an Early Warning System based on WSNs using the ISA100.11a standard for alarm management and reputation for controlling behaviours. This model consists of different components that have been presented and their functionalities have also been analyzed being the main ones the Pattern Association component and the Reputation Manager.

Although different cases have been analyzed for a Smart Grid scenario, the main idea of the proposed solution has been to protect the operational control domains to indirectly protect the entire Grid as a whole. Thanks to these analysis we have concluded that WSNs playing as EWS are perfect candidates for offering future solutions of EWSs. They provide the required ingredients for detecting, tracking and alerting about evidences that can produce a negative effect on the performance of a domain with an impact on the entire Grid, if they are not treated properly in advance.

We are at an initial development phase but our intention is to provide a complete implementation that shows its feasibility in a real and critical context. Lastly and as future work, we intend to research how to directly include all the logic of this approach within sensor nodes. However, this will obviously depend on the computational capabilities and resources offered by them, which are still constrained. Likewise, a security analysis of the approach will have to be performed in order to evaluate its integrity against existing and future faults.

Acknowledgments

This work has been partially supported by the ARES project (CSD2007-00004), the SPRINT (TIN2009-09237) project, being the last one also co-funded by FEDER and the EU under the Information and Communication Technologies (ICT) theme of the 7th Framework Programme for R& D(FP7) project NESSoS (FP7 256890). The first author has been funded by the Spanish FPI (Formacion de Personal Investigador) Research Programme.

References

[1] NIST, "NIST Framework and Roadmap for Smart Grid Interoperability Standards, Realease 1.0.", NIST Special Publication 1108, January, 2010.

[2] NIST, "Introduction to NISTIR 7628 Guidelines for Smart Grid Cyber Security", SIGiP, the Smart Grid Interoperability Panel, Cyber Se-

curity Working Group, September, 2010, http://csrc.nist.gov/publications/PubsNISTIRs.html, Retrieved on March 2011.

[3] J. Peerenboom and R. Fisher, "Analyzing Cross-Sector Interdependencies", IEEE Computer Society, HICSS, IEEE Computer Society, pp. 112–119, 2007.

[4] The International Society of Automation, "ISA-100.11a-2009. Wireless systems for Industrial Automation: Process Control and Related Applications", ISA100.11a, 2011.

[5] K. Walter and E. Nash, "Coupling Wireless Sensor Networks and the Sensor Observation Service - Bridging the Interoperability Gap", 12th Agile International Conference on Geographic Information Science, 2009.

[6] C. Alcaraz and J. Lopez, "A Security Analysis for Wireless Sensor Mesh Networks in Highly Critical Systems", IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, num. 4, vol. 40, pp. 419–428, ISSN:1094-6977, July 2010.

[7] ZigBee Alliance, ZigBee Technology, http://www.zigbee.org/, September, 2010.

[8] WirelessHART, HART Communication Foundation, http://WirelessHART.hartcomm.org/, September, 2010.

[9] M. Mejia and N. Pea and J. L. Muoz and O. Esparza, "A Review of Trust Modeling in Ad Hoc Networks", Internet Research, vol. 19, pp. 88–104, 2009.

[10] R. Roman and M. C. Fernandez-Gago and J. Lopez and H.-H. Chen, "Trust and Reputation Systems for Wireless Sensor Networks", On Security and Privacy in Mobile and Wireless Networking, Troubador Publishing Ltd, 2009.

[11] R. Roman and C. Fernandez-Gago and J. Lopez, "Featuring Trust and Reputation Management Systems for Constrained Hardware Devices", 1st International Conference on Autonomic Computing and Communication Systems (Autonomics 2007), vol. 302, ACM, Rome(Italy), October, 2007.

[12] A. Srinivasan, F. Li, and J. Wu, "A Novel CDS-Based Reputation Monitoring System for Wireless Sensor Networks", 28th International Conference on Distributed Computing Systems Workshops (ICDCS 2008), Beijing, China, June, 2008.

[13] M. Krasniewski and B. Rabeler, "TIBFIT: Trust Index Based Fault Tolerance for Arbitrary Data Faults in Sensor Networks", DSN '05: Proceedings of the 2005 International Conference on Dependable Systems and Networks, pp. 672–681, Washington, DC, USA, 2005.

[14] T. A. Zia, "Reputation-based Trust Management in Wireless Sensor Networks", Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP 2008), pp. 163–166, December, 2008.

[15] C. Alcaraz and I. Agudo and C. Fernandez-Gago and R. Roman and G. Fernandez and J. Lopez, "Adaptive Dispatching of Incidences Based on Reputation for SCADA Systems", TrustBus '09: Proceedings of the 6th International Conference on Trust, Privacy and Security in Digital Business, 5695, pp. 86–94, Springer-Verlag, Berlin, Heidelberg, September, 2009.

[16] E. Brewer and D. Culler and D. Gay and P. Levis and R. Behren and M. Welsh, "nesC: A Programming Language for Deeply Networked Systems", `http://nescc.sourceforge.net/`, 2004, Retrieved on March 2011.