Deliverable: Design of a service-oriented CII architecture
Workpackage 1, Task 1

Rodrigo Roman, Cristina Alcaraz
University of Malaga

March 15, 2010

1 Introduction

The main objectives of this task are twofold: first, to define at low level a sensor network (WSN) architecture that is suitable for later integration within critical infrastructures. Second, to define at high level a specification of a Service Oriented CII Architecture.

1) Low-Level WSN Architecture In these last years, numerous standards that are specially designed for providing services in critical environments have been created (e.g. WirelessHART, ISA100.11a). Examples of these services include functions for supervisory control, detection of anomalous situation and alerting. While the functionality of these protocols may seem good enough to allow a seamless integration with critical infrastructures, there are some security-related aspects that have been underdeveloped. The purpose of our research is to fill these gaps, and as a result of this task we provide both the definition of certain security mechanisms that are essential for critical infrastructures (public key cryptography and self-awareness) and a strategy for integrating these mechanisms within WSN platforms (through the use of a transversal layer).

2) Service-Oriented CII Architecture As we have achieved a secure WSN architecture for critical infrastructures in the previous subtask, we have focused this subtask in describing how that WSN could be integrated in present-day critical infrastructure elements, such as SCADA systems, to provide services. This way, the WSN will be able to fulfill high-level tasks such as Early Warning Systems, which will be studied at later deliverables. Note that there are some particular services that must be considered at a high level but cannot be managed by a WSN, such as alarm assignment to human operators. We also have used this subtask to show a solution to this particular problem.

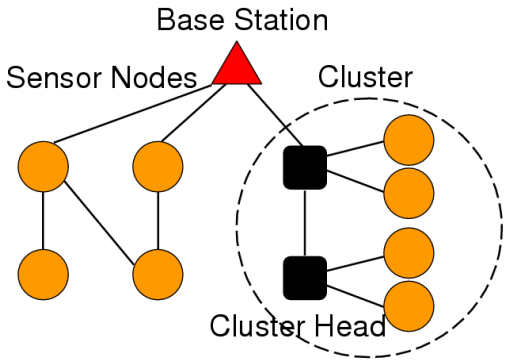

Figure 1: An overview of the architecture of WSN

The structure of this document is as follows. Section 2 introduces the WSN technology, providing a summary of both the most recent protocols developed for industrial environments and the latest capabilities of sensor network hardware. After that, section 3 describes how a transversal layer can hold the necessary security mechanisms that are needed in critical environments, and we also explain such mechanisms (public key cryptography, self-awareness). Section 4 introduces one of the most important critical infrastructure systems, the SCADA system, and shows how the previously explained WSN architecture can be integrated in order to provide services. Besides, a human-centric problem that cannot be solved using WSN, alarm assignment, is studied. Finally, section 5 concludes the deliverable. Note that there is one additional section, appendix A, that explains in detail certain public-key security primitives shown in section 3.

2 WSN: Introduction and Challenges

2.1 An introduction to WSN

A Wireless Sensor Network (WSN) [1] is composed of small and autonomous devices, deployed over a certain region, that cooperate with each other in order to achieve the same objective. These devices, called sensor nodes, make use of different types of sensors to monitor the physical state of a specific object or to examine the environmental conditions of its surroundings. Thanks to these attractive features, this technology is increasingly being applied in diverse scenarios and applications (from simple, complex to critical) of very different sectors (such as agricultural, business, environment, health care, homeland security, industry, and so on).

Sensor nodes can measure a wide range on environmental conditions, like temperature, humidity, lighting, radiation, noise, and others. Such information must be transmitted to the end user (a human being or computer) with the purpose of obtaining, evaluating, and studying relevant samples. However, there

is no direct link between the real world, where the sensor nodes are deployed, and the end user. Between both points, there should exist devices whose resources and capabilities have to be more powerful than sensor nodes. Those devices are known as Base Stations. Any device with enough capabilities to manage the services offered by the sensor network, such as a laptop or a PDA handed by a user, can become a base station.

Regarding the services offered by a WSN, sensor nodes not only can monitor the environment, but also can issue warnings and receive queries about the state of the network or a certain property. Indeed, all measurements perceived (e.g. radiation) and processed by the nodes must be sent to the closest base station, being later retransmitted to the end user. Besides, nodes must be able to detect any kind of anomalous activity of the environment (e.g. high levels of radiation) and alert the end users. Finally, the base stations can request to the nodes information about a specific feature of the network or environment, which is provided "on-demand". Note that base stations can also send control packets in order to reconfigure the network without using an additional infrastructure, since the nodes have the capability of self-configuring themselves. Therefore, the channel of communication between the sensor nodes and base station is totally bidirectional.

It must be noted that there are two types of architectures in WSN, which are represented in figure 1: hierarchical (HWSN) and flat / distributed (DWSN). In a hierarchical network, the sensor nodes are organized into groups, known as clusters. In every cluster there exists a special node, called "cluster head", entrusted to manage certain tasks in the cluster, as for example data aggregation. In contrast, in a distributed network the sensor nodes are completely independent, making their own decisions and determining which their next actions are by themselves. Note that it is possible to have both architectures in a sensor network at the same time (i.e. hybrid), thus improving the resilience and robustness of the network in case the "spinal cord" (i.e. the "cluster heads") fails.

2.2 WSN Protocols and Hardware

2.2.1 WSN Protocols

Since Wireless Sensor Networks are being demanded for industrial control and automation applications, diverse international organizations have joined efforts to standardize their communications. One of the first consortiums was the ZigBee Alliance [2], which produced the following ZigBee standards: 2004, 2006 and 2007/PRO. Previous releases of ZigBee Alliance were thought for home automation environments; however, and due to the critical nature of diverse systems, last versions contemplate diverse and specific services to improve the monitoring of complex systems, industrial systems and critical infrastructures [3]. After ZigBee, other standards that try to address the needs of industrial and automation systems have been developed. Examples of these standards are WirelessHart [4] and ISA100.11a [5], as shown in Figure 2. The remainder of

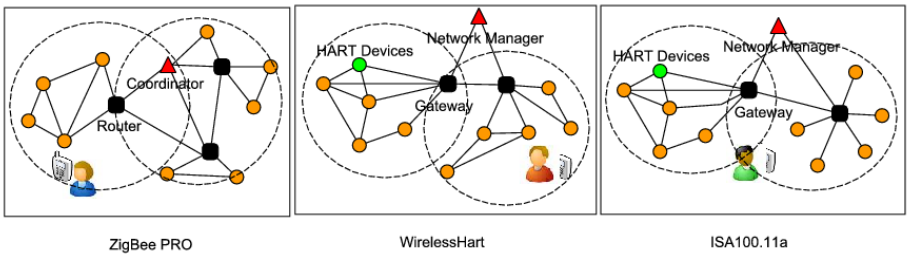

Figure 2: WSN Standards

this section will introduce these communication standards for WSN.

ZigBee. The ZigBee standard, created by the ZigBee alliance, is a wireless communication standard for constrained devices. At present, the newest version of the standard is known as ZigBee-2007. It is able to support mesh networks using different types of network devices: i) coordinator, ii) routers to help end-points to transmit data to the coordinator, and iii) end-points - sensor nodes. Regarding the stack architecture of ZigBee, it is based on four main layers: PHY, MAC, NWK (Network) and APS (Application) layers. This last layer also includes two important sub-layers: ZDO (ZigBee Device Object) and Application Framework. Basically, the two lowest layers (PHY and MAC layers) are specified by the IEEE 802.15.4 standard. The NWK layer is in charge of packet routing, network and security management, and joining/rejoining management. The APS layer defines the application domains, and provides data transmission, security and binding (matching of compatible devices such as switches and lamps) to the endpoints. The APS layer must also keep a table that stores the nodes or clusters deployed on the whole network. The ZDO sub-layer is the responsible for the local and over-the-air management on the network, security management, and node and service discovery. Lastly, the Application Framework sub-layer allows to add new applications to the network.

With respect to security, ZigBee supports symmetric keys with AES-128, providing authentication and confidentiality at NWK and APS levels through a transversal security service provider layer. The highest level of security is provided by a subset of the ZigBee specification, known as ZigBee PRO, which offers a "High Security" mode with better protection levels than the standard mode. Basically, ZigBee PRO introduces a new key, known as master key. Such key is preconfigured in the new devices in order to generate the link key (i.e. key used to protect communications between nodes at the APS layer) by means of a Symmetric-Key-Key-Exchange (SKKE) algorithm. At the moment that the link key is generated, the network key (i.e. key used to provide confidentiality at the NWK layer) is transmitted encrypted with it. The network key is frequently updated in a unicast mode and protected with the link key by the coordinator.

WirelessHART. The WirelessHART protocol, developed by the HART Communication Foundation, has as goal to provide industrial solutions through wireless mesh networks composed of node groups. Its network architecture is based on four essential components: i) a gateway, ii) a network manager, iii) the sensor nodes, and iv) the existing industrial devices or equipments (such as a Remote Unit Terminal, RTU). The network manager deals with the routing tables, the synchronization schedule, the network configuration and the security. On the other hand, the gateway is the interface between the WSN world and the control system, as for example the control center of a SCADA (Supervisory Control and Data Acquisition) system [6]. With respect to the stack of WirelessHart, the PHY layer is based on the IEEE 802.15.4-2006 whereas the MAC layer is exclusive. Its MAC layer uses the TDMA (Time Division Multiple Access) protocol for collision control with an special mechanism known as superframe. Besides, it controls the noise and interferences in the communication channel by applying frequency hopping and blacklisting methods. On the upper layers, WirelessHart also offers other very suitable services for critical environment, such as: energy management, a diagnostic mechanism (embedding the path to follow into the message header) or message priority management.

In terms of enforcing security, it provides protection at both network-level and MAC-level, managing four types of security keys: public key, network key, join key and session key. The public key and the join key have to be preconfigured in every new network device in order to generate the MIC of the MAC layer and NWK layer, respectively. This process will allow any device to be later authenticated in the network manager. Whenever a new node is authenticated by the network manager, it will receive the session key and the network key. The session key is a unique key between two network devices to encrypt any interchanged messages, while network key is shared by all network devices to generate the MIC of the MAC layer. The MIC is generated with CCM* (counter with CBC-MAC) using the AES-128 algorithm. For its generation is necessary to include a 128-bit key whose value will depend on node state (a new node public key or an old node network key), a nonce of 13 bytes and the message header without encryption.

ISA100.11a. The ISA100.11 release one is an open standard approved by the ISA100 Standards Committee in April 2009. This standard is focused on providing diverse control services at automation and control systems. Examples of these services are functions for supervisory control, detection of anomalous situation and alerting in mesh and star networks. Other goals include the assurance of interoperability with other communication systems, compatibility with existing hardware and software systems, energy conservation, reliability and security.

The ISA100.11a architecture is focused on the OSI model, where the lowest layers (PHY and MAC layers) use the IEEE 802.15.4-2006 standard operating in the 2.4GHz frequency band. The Data Link Layer (DLL) layer implements the TDMA protocol and several functions to provide frequency hopping and mesh

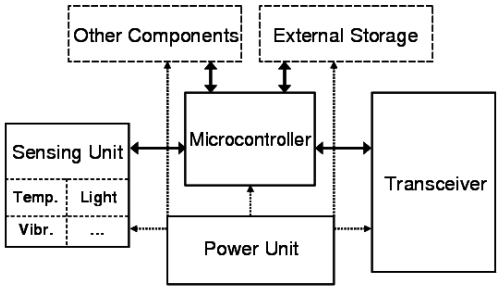

Figure 3: HW Elements of a Sensor Node

routing. The NWK layer is in charge of offering functions of inter-networking routing (i.e., mesh to mesh routing), such as addressing, routing, quality of service (QoS) and management functions. The transport layer includes a set of functions to transmit data between network devices in a reliable way, incorporating mechanisms such as flow control, reliable / unacknowledged service, enhanced-secure / basic-secure service, fragmentation and reassembly, and so on. Finally, the application layer include services that guarantee interoperability among diverse communication technologies and infrastructures with very low latencies. Moreover, This layer also provides a native protocol and a tunneling protocol. As for the security services, these are extended throughout the whole stack and are based on the security offered by IEEE 802.15.4-2006 with symmetrical and asymmetrical keys, configuration, operation and maintenance.

2.2.2 WSN Hardware

A sensor node is typically made up of four basic components: sensing unit, transceiver, processor unit, and power unit, as seen in Figure 3. The sensing unit consists of an array of sensors that can measure the physical characteristics of its environment, like temperature, light, vibration, and others. The processing unit is, in most cases, a microcontroller, which can be considered as a highly constrained computer that contains the memory and interfaces required to create simple applications. The transceiver is able to send and receive messages through a wireless channel. Finally, the power unit provides the energy required by all components, and such energy may come from either a battery or renewable sources. Most nodes have additional components, such as LEDs and buttons, which are used as user interfaces. There can be also other components depending on the needs of the application, like external data storage (e.g. flash memory), location devices (e.g. GPS chips), or cryptographic chips [7].

One of the most important components in a sensor node is the transceiver (i.e. transmitter-receiver). As one of the foundations of the sensor network paradigm is distributed collaboration through wireless communication, it is necessary for the sensor nodes to be able to "converse" with other nodes. Most sensor nodes have a limited energy supply, thus transceivers have to offer an ad-

	Speed	RAM	ROM	Energy
Class I	4 Mhz	1 KB	4-16 KB	1.5 mA
Class II	4-8 Mhz	4-10 KB	48-128 KB	2-8 mA
Class III	13-180 Mhz	256-512 KB	4-32 MB	40 mA

Table 1: Classes of Sensor Nodes.

equate balance between a low data rate (e.g. between 19.2 Kbps and 250 Kbps) and a small energy consumption in low-voltage environments (i.e. around 3V), allowing the node to live for an extended period of time. For most environments, Radio frequency (RF) communication is ideal because it is not limited by line of sight and current technology allows implementation of low-power radio transceivers [7].

As for the microcontroller, it has to provide enough computational capabilities and memory for executing simple tasks while consuming as less energy as possible. It is possible to classify the microcontrollers used in sensor nodes intro three types, as seen in table 1. The most constrained class of sensor nodes (class I) is very limited and its elements barely support the "de-facto" standard operating system for academic sensor nodes, TinyOS [8], while the most powerful sensor nodes (class III) have PDA-like capabilities and can host complex operating systems or Java-based virtual machines. Finally, there are devices that are resource-constrained but powerful enough to hold complex applications (class II). This is the most common type of device for sensor nodes, and there are many microcontrollers that fall into this category.

One open question that remains is how these constrained microcontrollers will evolve. Since 2004, the technical specifications of class II nodes range from 4 KB of RAM and 48 KB of instruction memory to 16 KB of RAM and 128 KB of instruction memory. It has been hinted (cf. [9]) that future versions of these nodes will achieve around 16 KB of RAM and 256 KB of instruction memory. In fact, these predictions have been confirmed by the specifications of the sensor nodes that can be found in the industrial market. For example, actual sensor nodes used in ZigBee PRO industrial applications (cf. [10, 11, 12]) have the following specs: 4 MHz - 32MHz microcontrollers, 8KB - 128KB RAM, and 128 KB - 192 KB of flash memory. Regarding ISA100.11a-ready sensor nodes [13], they provide 96kB RAM (containing both instructions and data), 26MHz microcontrollers, and 80KB ROM.

3 Low-Level WSN Architecture

All WSN protocols that have been developed for industrial environments can provide enough services to fulfill the needs of critical infrastructures. However, the underlying layer we need to create a low-level WSN architecture that can be completely integrated into these critical infrastructures is not complete. The main reason is security: there are some particular challenges that have not been

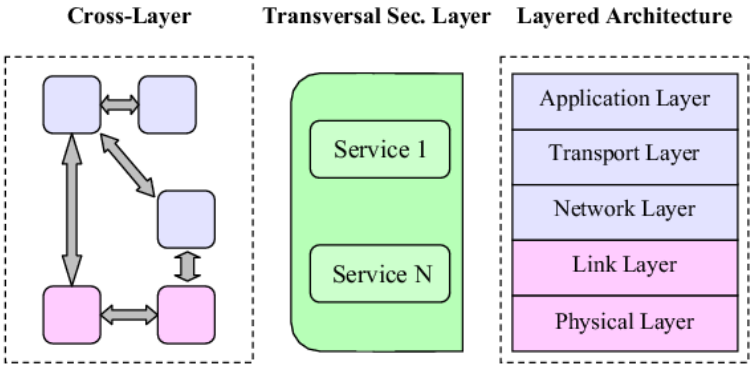

Figure 4: Security as a transversal layer

completely solved by these industrial protocols. For example, most attacks can be launched by malicious authorized insiders who know the inherent vulnerabilities of the system. Besides, WirelessHART and ISA100.11a are slightly more vulnerable to denial of services attacks due to fake messages and routing attacks, and part of the key negotiation of ZigBee PRO is done without any security at all.

As an output of this task, our first objective is to produce a security architecture that can integrate new security mechanisms into existing infrastructures without breaking the actual design. We have achieved this goal by defining and analyzing a transversal layer. Our second objective is to present certain security mechanisms that can provide some protection against the attacks that target industrial protocols: public key cryptography mechanisms, that can be used for secure broadcasting, key negotiation and authentication purposes, and self-awareness mechanisms, that can provide input to a broad range of high-level mechanisms such as self-configuration, intrusion detection, and accountability.

3.1 Security as a Transversal Layer

The major purpose of the security mechanisms is usually not to implement the logic of the application, but to offer support for the creation of secure applications. We can see then the security mechanisms as tools, used either by other elements of the architecture or by other security mechanisms. As they share the same goal, they can be considered as a single logical unit inside the architecture, thus we could group all the security mechanisms within a single layer. By using this approach, we can have a better management of all the mechanisms, delimiting their functionality and controlling their interdependencies. As for the relationship between the mechanisms and the rest of the architecture, the mechanisms should be able to access all the elements of the architecture in order to obtain information, and the architecture itself should be able to access any

of the security services when needed.

Therefore, security should be considered as a transversal layer (cf. Figure 4), that in both layered and cross-layer architectures will be able to provide information and services to the other elements of the architecture. Security services and primitives (e.g. protocols, algorithms) are located inside that transversal layer, and they interact between them and between the other elements of the architecture through well-defined interfaces. Besides, the transversal layer only provides security-related services, invokes the services of other layers in order to obtain information, and signals specific events to warn other layers about a specific situation. Therefore, any layer cannot access the functionality of other layers through the transversal layer in an indirect way.

From a technical point of view, the interfaces provided by the transversal layer must follow these two principles: independence and extensibility. The specific implementation details of the algorithms and protocols should not affect the definition of the interfaces, so the components that use the transversal layer do not need to change their implementation whenever a existing primitive is upgraded or changed. This can be achieved by defining a common interface for services and primitives of the same type (e.g. symmetric cryptography primitives), and also by using mechanisms such as the "factory" design pattern (in a similar fashion to the Java Security API [14]). As similar primitives and services will use the same type of interface, it is simple to add a new element whenever it is needed.

Precisely, the specific security elements that are contained within the transversal layer can be chosen according to the necessities of the applications and the capabilities of the devices. For example, if a specific application needs of digital signatures, the transversal layer can provide an interface for accessing a signature service. Such service can be implemented by using lightweight algorithms (e.g. signature algorithms based on Elliptic Curve Cryptography) if the devices are constrained in terms of resources. More powerful devices can implement more algorithms (e.g. identity-based signatures) inside the transversal layer. The transversal layer should also provide the services and primitives it contains whenever it is queried, so it is possible for two devices to negotiate security elements that they have in common.

Note that the definition of wrappers that allow any component of the transversal layer to access the functionality of elements located outside it is also important. Any change in the interface of the architecture will impact on the wrapper, but not on the implementation of the security elements located inside the transversal layer. In addition, if we do not define interfaces that access functional parts of the elements of the architecture (such as sending messages through a communication channel), then it will not be possible to bypass the layered structure of an architecture through the transversal layer.

3.1.1 Benefits and Applicability of Transversal Layers

By using the transversal layer approach, it can be possible to retain most of the benefits of a layered architecture. All the security mechanisms and services

are contained within the transversal layer, thus the modularity of the system is preserved. Besides, the transversal layer should only be accessed through well-defined interfaces, thus any component/layer of the system, located either inside or outside the transversal layer, will know how to interoperate with them in advance. In addition, since the transversal layer is isolated from the other layers, and even the internal components of the transversal layer can only access other components through their interfaces, it can be possible to either modify of upgrade the internal design of the layer without interfering with the other elements of the architecture. Note also that a layer cannot use the transversal layer as a bridge to access the services of unconnected layers, thus there is no risk of hidden dependencies.

The benefits of cross-layer architectures are also maintained by the transversal layer. The services contained within the transversal layer can access information published by the services of other layers, thus can have a holistic point of view of the state of the device. For example, a situation awareness service, in charge of monitoring the actual state of a node and its neighbourhood, can retrieve node status information from other layers. The external layers can also benefit from this transversal approach in many ways. First, since there is just one instance of a security mechanism (e.g. a cryptographic primitive), there is no need to replicate its functionality in order to implement different protection mechanisms (e.g. link layer protection and end to end protection). Second, all security mechanisms and services can be accessed by all the layers of the architecture. As a result, it can be possible to implement more effective protection procedures. For example, if an intrusion detection system is included in the architecture, both applications and communication protocols can use its outputs to limit any interaction with rogue nodes.

Finally, the possible disadvantages that may exist whenever cross-layer relationships are included in an architecture are reduced when using the transversal layer. The information flow between the transversal layer and other layers is delimited by well-defined interfaces, thus interactions amongst existing layers are no longer subtle. Any change inside the elements of the transversal layer must adhere to the definition of the interfaces, thus the chances of breaking the design of the whole architecture are slim. This level of isolation also limits the possible dependencies that may appear between layers after a change takes place.

Not only the transversal layer retains the benefits of both layered and cross-layer architectures, but also it can be integrated within both types. In layered communication architectures, the existence of a transversal layer containing the security services will allow the modification of those services according to the requirements of the applications (e.g. choose a different security primitive) without causing collateral effects that may negatively affect the other layers. In addition, it can be possible to add new security services derived from the necessities of the applications and / or long-term academic research.

On the other hand, in cross-layer architectures, the security layer behaves as another component of the architecture that can be accessed from any other component, proving security-related services and information. By centralizing

all security services inside one single component, we can improve the overall stability and maintainability of the architecture, and also we can provide a better control over the possible interactions and dependencies that may arise.

3.2 Extended WSN Security Services for Critical Infrastructures

3.2.1 Public Key Cryptography Services

Public Key Primitives. Public Key Cryptography (PKC), also known as asymmetric cryptography, is useful for secure broadcasting and authentication purposes. It requires of two keys: a key called secret key, which has to be kept private, and another key named public key, which is publicly known. Any operation done with the private key can only be reversed with the public key, and vice versa. Public Key Cryptography was considered to be unattainable for sensor node platforms, but that assumption was shattered a long time ago. The approach that made PKC possible and usable in sensor nodes was Elliptic Curve Cryptography (ECC), which is based on the algebraic structure of elliptic curves over finite fields. ECC has smaller requirements both in computation and memory storage, due to its small key sizes and its simpler primitives.

One of the most known software implementations of ECC for limited devices, TinyECC [15], implements ECC-based signature generation and verification (ECDSA), encryption and decryption (ECIES), and key agreement (ECDH). Note that the computational and memory requirements of these algorithms are not small (e.g. ECDSA requires 19308 ROM and 1510 RAM for the MICAz, generating a signature in 2s. and verifying it in 2.43s), although the implementation of these primitives is constantly evolving and improving. For example, one of the latest works in the area [24] implements a point multiplication (the foundational primitive of ECC) over a binary field in just 0.42s. Another example is the pairing primitive, which is the foundation of identity-based cryptography and other public key mechanisms. In [25] is reported that computing η_T on the supersingular elliptic curve $y^2 + y = x^3 + x$ over $GF(2^{271})$ (providing an equivalent RSA-1024 bit security) requires 62.73 mJ, which constitutes a major advancement in this area as earlier attempts, for instance [26], reported as much as 712.43mJ.

Another point that must be considered is the existence of certificates, which associate the identity of a node with its public/private key pair. While it could be possible to use the X.509v3 ITU-T [27] standard format for defining the contents of the digital certificates, this standard defines many fields that might not be necessary for the interactions between the members of a single sensor network. Fortunately, in the context of sensor networks, it is actually possible to simplify this certificate for local use, i.e. node authentication and secure broadcasting [28]. Nevertheless, there are some solutions that do not need certificates at all, and they will be discussed in the next paragraphs.

MICA2	Comp.	Comm.		**MICAz**	Comp.	Comm.	
ECMQV	59.33	29.5	88.83	ECMQV	59.33	23.97	83.3
SOK	62.73	8.04	70.77	SOK	62.73	6.53	69.26
SC-ECMQV	43.03	14.77	**57.8**	SC-ECMQV	43.03	12	**55.03**
UWM2000	Comp.	Comm.		**UWM4000**	Comp.	Comm.	
ECMQV	59.33	704.98	764.31	ECMQV	59.33	2291.23	2350.56
SOK	62.73	191.99	**254.72**	SOK	62.73	623.99	**686.72**
SC-ECMQV	43.03	352.99	396.02	SC-ECMQV	43.03	1147.24	1190.27

Table 2: Per node energy cost of authenticated key exchange (in mJ)

Authenticated Key Exchange. Certificates are needed to establish a trusted link between a public key and the identity of its owner (in our case a sensor node) in order to prevent man-in-the-middle attacks. In a sensor network, nodes are supposed to establish pair-wise keys with nodes that belong to the same network, and forbidden to do so with nodes or devices outside the network. Therefore, in key establishment protocols like ECMQV (Elliptic Curve Menezes-Qu-Vanstone [16, 17]), the nodes must at the beginning exchange their public keys and certificates. It is natural to assume these certificates take the form of a signature by the base station on the identity and public key of the node. In general, nodes public and secret keys are set up by the base station. Such a setting can be viewed as a key-escrowed system, that is, there exists a trusted party who computes the secret keys of the users. As a consequence one is tempted to use different forms of key-escrowed public key paradigms that have smaller bandwidth requirements, like identity-based cryptography [18, 19] (even if it does not provide certain properties such as forward secrecy) or self-certified cryptography [20, 21]. This is particularly important for certain critical networks such as Underwater Sensor Networks [23], where there are severe limitations in bandwidth and the transmission of one bit requires a huge amount of energy.

By analyzing the algorithm definitions (contained in appendix A), we can derive that the overall energy cost and transmission cost of ECMQV for one node amounts to:

$$2\text{mexp}(2) + 1\text{exp} + 2\text{sqrt}(+\text{trans. } 1410 \text{ bits} + \text{recep. } 1410 \text{ bits}) \quad (1)$$

whereas the energy cost and transmission cost of SOK for one node amounts to:

$$1\text{hash}_{\mathbb{G}} + 1\text{pairing}(+\text{trans. } 384 \text{ bits} + \text{recep. } 384 \text{ bits}) \quad (2)$$

and the the energy cost and transmission cost of SC-ECMQV for one node amounts to:

$$1\text{mexp}(3) + 1\text{exp} + 2\text{sqrt}(+\text{trans. } 706 \text{ bits} + \text{recep. } 706 \text{ bits}) \quad (3)$$

considering that i) one packet containing nodes identities, protocol ID, message ID, checksum, and low-level headers and footers, amounts to a total of 384 bits,

ii) public keys have 161 bits, iii) each ECDSA certificate has 320 bits, and iv) each ephemeral key contributes with 320 bits.

Using these equations and the energy figures from [25] (pairings and public key primitives) and [29, 30] (energy consumption of "normal" and underwater sensor networks, respectively), we can obtain the results described in Table 2. This table shows the energy consumption of a sensor node engaged in authenticated key exchange protocols in "normal" and underwater sensor networks, in terms of mJ. From the table, we discover that *for both UWSN platforms we considered, SOK is much better than the other protocols*, due to the high transmission cost. Moreover, *SC-ECMQV beats ECMQV even when the motes use radio frequency transceivers.* This is not surprising, since both the computation and communication costs of using SC-ECMQV are smaller.

3.2.2 Self-Awareness Services

In a critical environment, it is essential for the nodes to be aware of their environment. A sensor node that does not know its own situation and the situation of its environment cannot react to possible events that may influence its functionality. Therefore, it is essential to have certain self-awareness services that provide this information (e.g. whether a certain node has disappeared from a neighbourhood) to the protocols of the sensor node. In fact, this information is also vital for the user of the network, because he/she should be able to know at all times the state of the network. Existing industrial protocols, such as WirelessHART, provides partial support for self-awareness services, but do not provide tools for analyzing all possible situations that might happen in the network.

Due to the distributed nature of the network, most sensor nodes are equally important for offering the services of the network (e.g. monitoring the infrastructure). Also, any occurring event will be detected mostly by the neighbourhood in which the event takes place. Therefore, it is necessary to adopt a decentralized solution, where total coverage is assured by making all nodes and base stations participant in the analysis of the state of the network. This leads to a issue that must be considered: the limiting resources of a typical sensor node. A sensor node is very constrained in terms of battery life and processing power, so any detection mechanism crafted for sensor nodes should consist only on simple tasks.

One solution is to use the simile of "a sensor network as a living body", where a sensor node is considered the "cell" of the system, and the base station is the "brain", as seen in Figure 3. Having in mind this simile, it is possible to think that the presence of certain symptoms (i.e. collateral effects) will be indicative of the existence of a disease (i.e. abnormal event).

One of the difficulties associated with the diagnosis of a disease consists on separating the existing symptoms from the normal behavior of the body. However, the functionality of sensor networks is usually fixed, with sensor nodes providing the same services during all the lifetime of the network. Therefore, any deviation of the behavioral pattern of the network, or the existence of a well-

established set of unusual patterns, can be considered as a potential effect of an abnormal event. Another issue that can affect the diagnosis is to distinguish one disease from another one according to the existing symptoms. Nevertheless, in a sensor network context, the mere possibility of detecting and warning about the existence of a problem can be useful enough for the user of the network. Even more, most abnormal events do not share the same effects.

Abnormal event (disease)	**Collateral effect (symptom)**
Jamming	Wide data unavailability
Hw. failure ("unavailable" node)	Data unavailability
Node subversion	Node temporarily unavailable
Tampered, Malfunctioning sensor	Deviations, Inconsistences
Packet Replaying	Packet too old
Impersonation Attacks	New neighbours, Packets per node
Message creation	Changes in packet density, Inconsistent alerts
Packet alteration	Changes in packet (only for broadcasted)
Feature advertising	Inconsistent feature with neighborhood
Time-Related attacks	Long delays, Traffic imbalance

Table 3: Relationship between WSN attacks and their symptoms

As a result from our analysis, it is possible to link the different attacks and node malfunctions that can affect the functionality of a sensor network with the "symptoms" that they produce, as shown in table 3. Note that, in most cases, the detection mechanisms that infer the existence of abnormal events from these "symptoms" are not complex, and such events can be detected just by storing and analyzing simple statistics generated by the network. As a result, these mechanisms can be lightweight enough for constrained environments.

Such mechanisms can be integrated with full-fledged intrusion detection systems, like [31]. Note that a intrusion detection system must comply with other important properties, besides simplicity (support for lightweight detection mechanisms): audit data management (works with very application-specific partial audit data), secure cooperation (no node should be completely trusted in cooperative algorithms), full network coverage (all the elements of the network must be considered as potential entry points), support for extensibility (distinguish the incorporation of new nodes from other attacks), flexibility (possibility to include new detection mechanisms), and robustness (the system should be able to withstand an attack against itself).

As for the architecture of the IDS client, there is a "de-facto" agreement on its basic elements: a local packet monitoring entity that receives the packets from the neighbourhood, a statistics module that stores the information derived from the packets, a local detection engine that detects the existence of the different attacks, an alert database that stores information about possible attacks, and a cooperative detection engine that collaborate with agents located within the neighbourhood. In fact, in order to achieve full network coverage, all nodes should have a IDS client installed. Note that it has been proved in real world

settings that this architecture is lightweight enough to work in class II nodes, although it is possible to further optimize this distribution by running the detection mechanisms at regular intervals and by selecting (manually or statistically) the nodes that will be in charge of monitoring the messages coming from the neighbourhood.

4 Essential Services for a Service Oriented CII Architecture

Before starting this section, some concepts related to critical infrastructure and Service-oriented Architecture should be considered and clarified, considering application, challenges and problems in the literature.

CII / CIIP The well-being of the national and international economy, security and quality of life, is becoming increasingly dependent on the safety and the robustness of Critical Infrastructures (CI), such as energy, banking, transport, and others. According to the European Commission, Critical Infrastructures consist of *those physical and information technology facilities, networks, services and assets which, if disrupted or destroyed, would have a serious impact on the health, safety, security or economic well-being of citizens or the effective functioning of governments in the Member States. Critical Infrastructures extend across many sectors of the economy, including banking and finance, transport and distribution, energy, utilities, health, food supply and communications, as well as key government services* [32]. These infrastructures depend on a spectrum of highly interconnected national (and international) software-based control systems for their smooth, reliable, and continuous operation. This information infrastructure underpins many elements of the aforementioned Critical Infrastructures, and is hence called *Critical Information Infrastructures* (CII). As a result, key sectors of modern society that are vital to the national security and the essential functioning of industrialized economies are dependent of the well-being of these CII, making Critical Information Infrastructure Protection (CIIP) a priority.

An important issue to discuss is the connectivity of CIs. A CI is based on a set of collaborative and adaptive components [33], with capability to learn of past experiences. These components communicate with each other in a certain context, and receive as inputs the outputs corresponding from other components. Moreover, a specific input could produce a certain effect on the state of a component. This way of establishing connections between components can also be applied to the relationships between complex other infrastructures. Such connectivity can be done through dependent or interdependent connections. When the relation between two infrastructures is individual and unidirectional, it is considered a simple dependency connection. This can be seen as a linkage between two points i, j, where i depends on j, but j does not depend on i, and any

problem in j affects on i, but not on the contrary. However, in the real life the connections of infrastructures are much more complex. Every infrastructure is connected to other by means of bidirectional links (j depends on i, and i depends on j), known as interdependency connection. This relation implicates that any state of each infrastructure influences on the behaviour of other and vice versa, involving an interblock between them [34]. This concept is quite relevant within CIIP since a failure o threat could affect to other infrastructures. Therefore, any kind of accidental or provoked failure can cascade through and between infrastructures, with unpredictable and extremely damaging consequences.

As special note, most of these CIs belonging to the industrial sector. These critical infrastructures are controlled by other specialized and complex control systems known as *Supervisory Control and Data Acquisition Systems* (SCADA) - the Section 4.1.1 is basically dedicated to this infrastructure since this is considered one of the most important of the CII literature by the government, industry and scientific community. In particular, this complex and critical system helps the operators know the state of the infrastructure in real-time, by simply observing those data come from RTUs (Remote Terminal Units) and sensors deployed in all the development area. Even though the main requirement of a SCADA system is to ensure the performance and availability of the controlled system, security issues should be taken into consideration since a failure in the controlling process or a threat could mean a harmful cascade effect in the whole system [35]. Hence, SCADA systems are considered critical infrastructures by themselves as well.

SOA As was already described a CIIs are complex networks that provide the most fundamental requirements of the society. Their importance in the smooth conduct of the society has made their role more and more prominent. A failure in any of these important components of today's industrial society can well affect the lives of millions of people. It is not only their individual break down that raises serious concerns, but their mutual reliance (interdependency) is even more threatening. Although interdependency in these infrastructure systems provides many benefits for their operation, a failure in one can ripple down to the others and cause a catastrophic irremunerable event. One way of controlling such interdependences and guaranteing a suitable performance is designing a set of appropiated services using a service-oriented architecture (SOA).

A service-oriented architecture is essentially a collection of basic services, which communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity [36]. Some means of connecting services to each other is needed. An example of service in a critical environment could be the prevention and control of anomalous behaviors, thus the system will be able to autonomously respond against a possible incident/threat and in other cases to alert human operators so they can take action instead.

Thus the propagation of an anomalous event is avoided, and hence protecting our critical and national infrastructures. Other service could be associated to the alarm management, since multiple alarms in a critical system could not be properly attended by operators. An error or failure in attending an alert could mean a serious cascade effect.

The field of protecting Critical Information Infrastructures (CIIP), faces numerous challenges, such as the design of a Service Oriented CII Architecture able to manage secure interaction between peers, assure the resilience and robustness of the overall system and deploy a set of basic services of prevention and control. In this deliverable is possible to identify two main services: (i) a service of prevention, control and monitoring, and (ii) a smart service of alarm assignment in order to assure a suitable and fast respond. Both services could add new improvements, reliability, security and performance.

4.1 Prevention, Control and Monitoring Services

CIIP faces numerous challenges, for example managing the secure interaction between peers, assuring the resilience and robustness of the overall system, or deploying warning and alert systems. For carrying out such proposals, suitable and smart sensors belonging to a Wireless Sensor Networks (**see Section 2**) are required for providing support to such protection. Sensor nodes can provide a robust and self-reactive network that is able to continuously monitor any kind of physical event of an infrastructure, such as vibration, humidity, radiation, or others. Also, in case the infrastructure starts to fail, the sensor network can provide the exact location and extent of the problem, helping to solve the situation in a short period of time.

A WSN can be easily set up in a physical context where it is needed, being extremely useful for controlling and diagnosing any previously existent equipment. For example, in case a control system is faced with a serious disruption that renders the operation of its subsystems unusable, a sensor network can be deployed "on the spot" for providing reliable and robust information about the physical infrastructure or the status of any component. Such WSN can also be used for diagnosis purposes, comparing the actual values returned by a fully functional control system with the values acquired in real time by the network.

The data provided by the network that monitors both the infrastructure and the systems that control the infrastructure can also be used for providing an accurate diagnosis of a certain context, detecting the events previous to a dangerous situation by feeding systems such as *Early Warning Systems*. Not only that, but it is also possible to use the events generated by the EWS as an input for *Dynamic Reconfiguration Systems* (DRS), which are capable of reconfiguring the different components of the CII in an automatic way. Surely, the redundant and resilient information provided by the sensor networks will help a system to react accurately against serious stresses or disruptions, avoiding for example a serious cascade effect among CIs.

A particular case of Critical Infrastructure is precisely those industrial systems, among them: critical control systems, known also as SCADA.

4.1.1 Highly Critical Control Systems: State of the Art

A SCADA system (or control system) is a complex system capable of controlling and managing other complex system whose resources are considered critical (such as water, gas, oil or electricity). In general, these control systems have evolved over time and are, at present, based on distributed environments. Generally, they are composed by very varied (hardware and software) components, being most of their logical components COTS (Commercial-Off-The-Shelf) so as to reduce cost of implementation and maintenance. However, both the interaction among different components and the new connection towards external networks, such as Internet, involve multiples and diverse problems of security.

In particular, a SCADA network architecture is composed by two types of foundation networks (both are depicted in the figure 1): the corporative network and the control network. In the corporative network, the operations are more related to the general supervision of the system and the contractors/employees require of strong authentication procedures to interact with the databases (historical, alarms, etc.) and critical servers. On the other hand, the control tasks (as for example, to open/close a pump or to retrieve a measurement) are carried out in the control network. All these tasks are managed by a HMI (Human Machine Interface) localized in the principal SCADA control centre or remote substations, and transmitted to certain field devices which are usually located in the industrial plants or substations.

A field device (such as a RTU - Remote Terminal Unit) is a device with constrained capabilities but autonomous and independent enough to be able to process data and to identify which sensor or actuator is the responsible of executing an order in a substation. Moreover, they are able to establish connections with other substations, other RTUs and other field devices such as PLCs (Programmable Logic Controllers). Furthermore, they can simultaneously process and respond to several messages transmitted by multiple sources since they can support multiples sessions with TCP/IP. Some RTUs can even support Linux/Unix or Microsoft Windows to provide Web applications with graphical interfaces to generate the reports.

Nowadays, numerous industrial and proprietary protocols coexist and work in a same system. Most of them work with the TCP/IP standard: Modbus/TCP [37], DNP3.0 [38], IEC 104 [39] or ICCP [40]. Alternatively, there are other protocols, such as the protocols corresponding to the Common Industrial Protocol (CIP) family supported by Open DeviceNet Vendors Association (ODVA) [41]: Ethernet/IP, DeviceNet, CompoNet and ControlNet. These protocols are useful for the control process, but they lack of protection mechanisms, hence they could open new and important security holes that can affect the security of the system. Regarding remote controlling from any geographic localization point, it is necessary that diverse communication infrastructures interact with each other, such as Ethernet, dial-up, Satellite, microwave, optical fiber, WiFi,

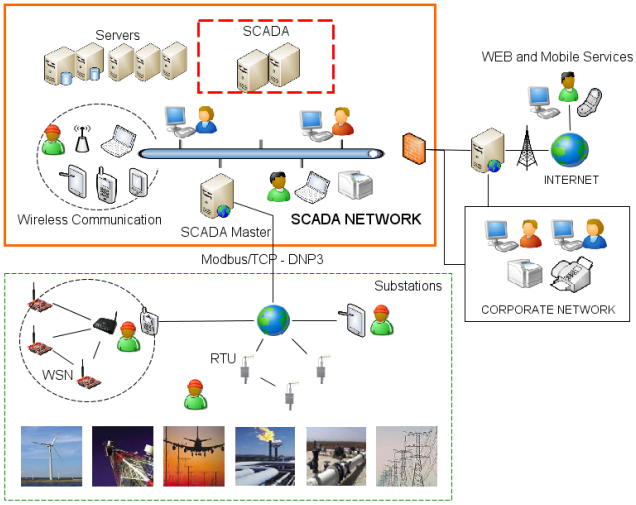

Figure 5: A SCADA network architecture

WiMAX, etc. Some SCADA systems could also provide Web and mobile (GSM or TETRA) services in order to reduce maintenance tasks and increase performance and availability of the system.

In general, a SCADA network, which is depicted in the figure 1, has multiple potential security holes, since internal and external attacks could appear in any point of the system. Internal attacks are associated to (intentioned or not intentioned) human actions, while external attacks are more related to the vulnerabilities corresponding to the standard TCP/IP, as well as the use of new technologies (for example, RFID or Wireless Sensor Network) and COTS components [42]. At present, many of these vulnerabilities are registered in public databases, such as CERT [43] or BICT (British Columbia Institute of Technology) [44]. CERT has approximately 2.500 vulnerabilities identified and 150 technical reports published since 1998. Similarly, BICT has the database ISID (Industrial Security Incidents Data), which was utilized by Byres et.al [45] to make a statistical study about the type of security problems in critical environment. They concluded that the external vulnerabilities had just started to emerge since 2001, rising every year.

4.1.2 Prevention, Control and Monitoring Service in Highly Critical Control Systems

In an industrial context, a WSN is possible to use too. In particular in substations. These industrial sensor networks composed of industrial sensor nodes whose hardware capabilities significantly differ from conventional sensor nodes. In particular, as shown in section 2.2.2, they are equipped with a 4MHz-32MHz micro-processor, 8KB-128KB RAM, and 128KB-192KB ROM, and with sensors

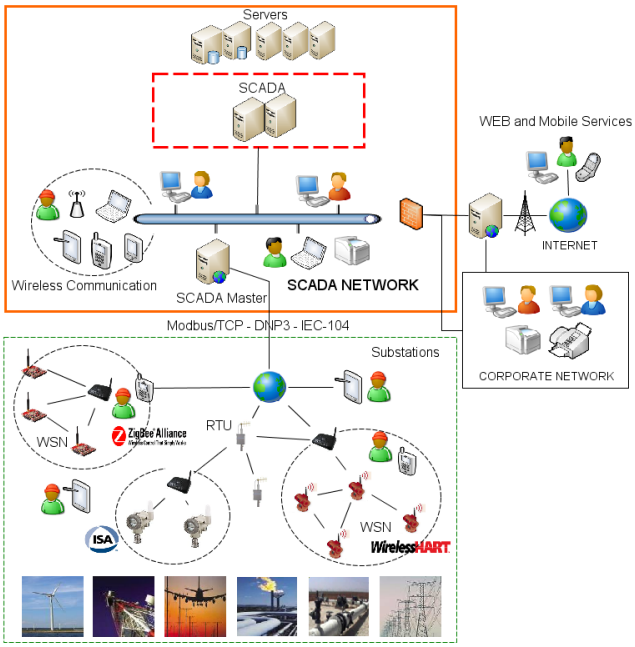

Figure 6: A Complete SCADA network architecture with different WSNs

to measure environmental data, such as temperature, pressure, vibration, light intensity, etc. Generally, and depending on the application context, the nodes are linked to an energy supplier or industrial equipment in order to maximize their lifetime (by between 5 and 10 years). It is possible to note in the figure 6 that different industrial WSNs can coexist in a same environment, whose communications and security are managed by coordinators or managers. This will depend on if the WSN is implemented under a ZigBee, WirelessHart or ISA100.11a communication (cf. section 2.2.1).

These industrial sensor nodes are smart and autonomous devices capable of processing any information acquired from their sensors and transmitting it to a central system with considerable hardware and software resources, such as for example an RTU working as a data collection device. In addition, they can offer auto-configuration, self-monitoring and self-healing capabilities, as well as detection/tracking of anomalous situations, alarm generation and reporting of any life-threatening situation, such as was described in section 4.1. The measurements sensed and the alarms generated will allow system to detect anomalous events, to react against them and to help system to execute more specialized audit and maintenance services based on real-time evidences. Moreover, these evidences will also be the base input to future early warning industrial systems and forensics techniques/applications. Hence, WSNs (with industrial sensor nodes or not) are considered a key technology for the protection and control of

many of our CIs.

Some real experiments have been carried out in industrial environments in order to validate their functionality in an industrial context. For example, Krishnamurthy et al. [46] designed a WSN to predict failures at a semiconductor plant and on an oil platform located in the north sea. Carlsen et al. [47] also deployed a WSN at an oil and gas platform off the coast of Norway to improve productivity from the wells. Bai et al. [48] installed a WSN at a wind farm to control the energy level obtained. Likewise, several international companies are offering nowadays real industrial solutions based on WSNs, such as the Emerson Process [49]. Examples are the control of railcars at the Mill Hall plant [50], the control of calcining at the Lime Kiln [51] and the control of wastewater discharge during steam injection at Chevron [52].

4.2 Adaptive Assignment and Control Services

Although SCADA systems are controlled by traditional policies, security services and protection mechanisms, it is very important to provide other and specialized security services for controlling the human operators' activities. These services could work in parallel with all the conventional policies and mechanisms and can provide a more accurate control. As a result, an automated adaptive response mechanism have been designed. This mechanisms offers control and prediction services since this is able to estimate the most suitable human operator to effectively respond to incidents and alarms in a SCADA system. Thus, reducing the possibility that an alert could remain untreated.

4.2.1 Adaptive Assignment based on Reputation

This new service based on a specific mechanism will make use of a *Reputation* module where all operators are assigned a certain value according to their behaviour and to their reaction when dealing with incidences. This new service will have to guarantee task control and monitoring without reducing the overall performance of the system, reliability (to identify the human operator that is more suitable for performing a certain task), security (a reputation module will allow managing the behaviour of the elements of the system and it can be used for detecting some malicious activities coming from internal attackers), and availability. The availability is achieved since this mechanism works in parallel with the other elements of the system and it is decoupled from the reputation module.

The reputation module is useful for storing the overall behaviour of the human operators, but it does not hold any decision-making capabilities. For that reason, a new automated adaptive response mechanism was designed, which needs an "Incidence Manager" component known as the *Adaptive Assignment Manager* (AAM) (see Figure 7). This component takes an alarm as an input, and it determines which operator and supervisor are the most appropriate to take it up. The AAM is also in charge of updating the reputation of the operators in the reputation module by using the feedback of the supervisors.

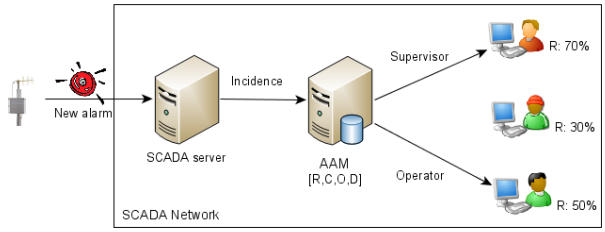

Figure 7: Functionality of an Adaptative Assignement Manager (AAM)

The AAM component does not pretend to completely replace the response and alert management capabilities of human operators and supervisors. Instead, it facilitates their work by selecting, in the first instance, the most skilled staff that could provide an early and effective response to the incidence, offering all the relevant information to supervisors in a way that they can do their job in an assisted manner. Also, it will allow system improve its functionality and performance, since all of this information will be part of the input for other future tools, improving some services related to audit and maintenance, as well as forensic techniques.

In order to determine which operator or supervisor are the most suitable for taking care of an incidence, the AAM considers the following set of parameters:

- ***C*riticality of the alarm**. The alarms are categorized by the type of criticality of an event occurred in the system. Such alarms are received by a SCADA server, which generates the associated incidences.
- ***R*eputation** of the operator and supervisor, obtained from the reputation module.
- ***A*vailability** of the operator and supervisor according to their contracts. They should be authorized in the system and being available in their work place.
- ***L*oad of work** of the operator and supervisor. This parameter is related to the overload of critical incidences that an operator might be dealing with at a certain time. If an operator is attending a number of non very critical incidences he could be still available for taking up a more critical one. However, if the operator is dealing with other critical incidences (even if it is only one) the system should identify another operator who could deal with it. The process is analogous for the supervisor.

Another task of the AAM is to serve as an interface to the values stored inside the reputation module. This way, the managers can determine the knowledge of the operators and even the level or mistakes made by them during the life time of a system. For example, an operator can i) reach the minimal reputation value, or ii) reach the maximal reputation value. In the first case the system should

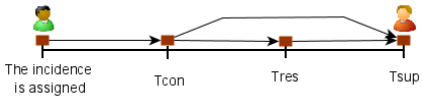

Figure 8: Counter to be used in this schema

notify it to the responsible managers of the organization owning the SCADA system, thus that they are aware of the situation. However, in the second case the organization should reward these employees in order to maintain this high threshold of reputation.

4.2.2 Analyzing Specific Scenarios

Different situations can be found in the management of an incidence by an operator assigned by the AAM system. At first, the order of processing C, R, A, L employed by the AAM system could be crucial for quickly reducing the set of candidates to be chosen as operators and supervisors. This is a key point when critical alarms need to be dispatched as soon as possible. Availability (A) seems to be the first parameter to be processed since it can reduce the group of operators to be evaluated in a speedy way as it puts aside those employees that are not actually at work. The rest of variables can be sorted in different ways depending on each scenario but a logical sequence that can be used is to take into account criticality (C) of alarms. This can be used in a third step to select those personnel that are less busy (L), and from them the one with a higher reputation (R).

As for incidence management, after selecting a human operator to manage an incidence received by the AAM system, a supervisor is chosen for monitoring the way it is going to be resolved by him/her. The operator must confirm the acceptance of the assignment before a defined time ($Tcon$). At that moment the resolution of the incidence starts. The supervisor is informed of the assignment done by the AAM system and a time counter ($Tres$) for determining how long it has been spent for resolving the incidence is started. This counter will warn the supervisor when an incidence remains unresolved for longer than it should. Thus, this counter could also help to calculate the efficiency of the operator in the resolution of incidences. Finally, a third counter must be used ($Tsup$) to check that the maximum time spent by a supervisor for managing an incidence not resolved by an operator is reached. These three counters are shown in Figure 8.

At this point, three situations can happen (see Figure 9).

- The incidence is successfully resolved by the operator assigned before *Tres* is reached. Nonetheless, the supervisor checks his/her resulting action after this counter is reached. The operator's reputation must be increased. (Figure 9-A).

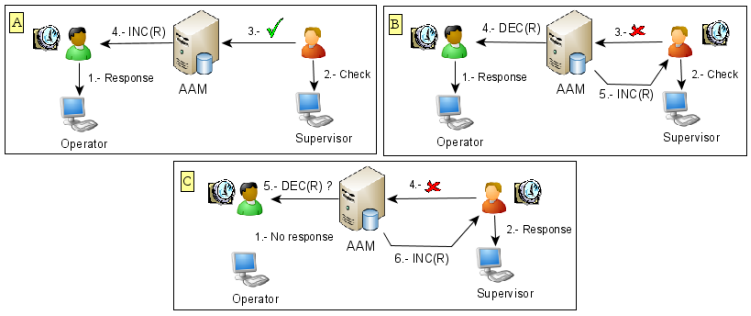

Figure 9: Three main situations in the management of an incidence in a SCADA system

- The incidence is not successfully resolved by the operator and *Tres* is reached. The supervisor checks the operator's action to be in charge of resolving it again. Finally, the operator and supervisor's reputation are changed (Figure 9-B).

- The operator could not confirm the acceptance of the assignment done by the AAM system. This situation is detected because *Tcon* is reached. The supervisor will be in charge of the incidence if this counter is overtaken increasing his final reputation. (Figure 9-C).

When a supervisor is in charge of managing the incidence, the AAM system must offer him all the information generated for the assignment. Thus, the supervisor can use this report in order to evaluate the reason why the incidence was not successfully resolved in such a way that he can deal with it in a more accurate way. Besides this, the supervisor must make a decision about how to proceed with the resolution of the incidence before *Tsup* is reached, otherwise his reputation must be modified by the AAM system conveniently.

Finally, an AAM system could not find any operator with enough reputation and the load of work needed to be selected for the assignment of an incidence. Also, supervisors could have a parameter showing their load of work that could drive to a similar situation. These states must be evaluated for each scenario as in some cases it could be solved by re-sorting actual incidences and the staff assigned to them. Some other times these incidences can be queued waiting for an operator and a supervisor to deal with them. Lastly, all of this information will allow the system to define and develop new (or better) audit and maintenance applications, procedures or services, rules or patterns to improve the knowledge of the Intrusion Detection Systems and complete forensic techniques to identify both the cause, the origin of a problem and the main responsible.

5 Conclusions

In this report we have pursued two goals: i) to provide a low-level WSN architecture that can be used in critical infrastructures, and ii) to analyze how to integrate the previously mentioned WSN into a present-day critical infrastructure such as a SCADA. It should be noted that the strict definition of the Service-Oriented architecture on top of the WSN architecture will be considered in later deliverables, because we need to consider the security mechanisms that will be developed in those future deliverables. Nevertheless, as there is one service that does not depend on the functionality provided by the WSN (alarm assignment), we have developed it in this deliverable.

A Descriptions of Authenticated Key Agreement Protocols

A.1 ECMQV - Elliptic Curve Menezes-Qu-Vanstone

We recall the elliptic curve version of the Menezes-Qu-Vanstone authenticated key exchange protocol [16, 17], which is one of the most standardized key exchange protocols using public key cryptography. Here public keys are authenticated by using traditional certificates.

Algorithm A.1 ECMQV key derivation for entity A

Input: Elliptic curve domain parameters G, g, n, secret keys x_A, y_A, public keys $\mathrm{pk}_A, \mathrm{pk}_B$, and ephemeral keys E_A, E_B

Output: A secret key K_{AB} shared with entity with public key pk_B

1: $m \leftarrow \lceil \log_2(n) \rceil / 2$
   $\{m$ is the half bitlength of $n\}$
2: $u_A \leftarrow (u_x \bmod 2^m) + 2^m$
   $\{u_x$ is the x-coordinate of $E_A\}$
3: $s_A \leftarrow (y_A + u_A x_A) \bmod n$
4: $v_A \leftarrow (v_x \bmod 2^m) + 2^m$
   $\{v_x$ is the x-coordinate of $E_B\}$
5: $z_A \leftarrow s_A v_A \bmod n$
6: $K_{AB} \leftarrow KDF(E_B^{s_A} \cdot \mathrm{pk}_B^{z_A} \bmod n)$

In Algorithm A.1, KDF is a key derivation function, which can be implemented with SHA-160 for example. Node A's public key is $\mathrm{pk}_A = g^{x_A}$, where x_A is A's secret key. Similarly for node B. In the first stage, the nodes exchange and verify certificates vouching for the fact that pk_A and pk_B are public keys from nodes belonging to the network. In a second stage, they exchange their ephemeral keys $E_A = g^{y_A}$ and $E_B = g^{y_B}$, where y_A, y_B are taken at random from the finite field GF(p). We assume certificates are minimalist and take the form of ECDSA [22] signatures (r_A, s_A) and (r_B, s_B) by the owner/manufacturer

of the network on the messages $id_A||\mathrm{pk}_A$ and $id_B||\mathrm{pk}_B$ respectively, where $||$ denotes concatenation.

Entity B runs the same algorithm by simply swapping the values $(x_A, y_A, \mathrm{pk}_B, E_A, E_B)$ in Algorithm A.1 with $(x_B, y_B, \mathrm{pk}_A, E_A, E_B)$ and finally obtains the same key K_{AB} (cf. [17]).

A.2 SOK - Sakai, Ohgishi and Kasahara

In this section we recall a non-interactive authenticated identity-based key establishment scheme. Due to the lack of any standardized identity-based key exchange protocol, we describe a non-interactive scheme due to Sakai, Ohgishi and Kasahara [18, 19], which is the first identity-based authenticated key agreement protocol proposed in the literature.

In the SOK protocol, a hash function $H : \{0,1\}^* \rightarrow \mathbb{G}$ is included in the domain parameters of the system, together with $\mathbf{g}^z$, where the master secret key z is only known to the base station. Node A's secret key is $sk_A = H(id_A)^z$, while node B's secret key is defined as $sk_B = H(id_B)^z$. Notice that A's identity is id_A and B's identity is id_B.

Algorithm A.2 SOK non-interactive ID-based key derivation for entity A

Input: Bilinear map domain parameters $\mathbb{G}, \mathbb{G}_1, e, \mathbf{g}^z, n$, the identity id_B and the secret key sk_A

Output: A secret key K_{AB} shared with entity with identity id_B

1: $K_{AB} \leftarrow KDF\big(e(H(id_B), sk_A)\big)$

Entity B runs the same algorithm by simply swapping the values (id_B, sk_A) in Algorithm A.2 with (id_A, sk_B) and finally obtains the same key K_{AB} thanks to the bilinearity of the pairing,

$$\begin{aligned} e(H(id_B), sk_A) &= e(H(id_B), H(id_A)^z) = \\ e(H(id_B), H(id_A))^z &= e(H(id_A)^z, H(id_B)) \\ &= e(sk_B, H(id_A)) \end{aligned}$$

A.3 SC-ECMQV - Self-certified ECMQV

In this section we explore the energy performance of a variant of ECMQV where implicit public key certificates are used. The base station has a public key $g_0 = g^z$, where $z \in \mathbb{Z}_n$ is secret, and $h : \{0,1\}^* \rightarrow \mathbb{Z}_n$ is a public hash function. The self-certified public keys of the sensors are $w_A = g^{r_A}$ and $w_B = g^{r_B}$, with r_A, r_B only known to the base station, while the corresponding secret keys known to the nodes are $x_A = z \cdot h(id_A, w_B) + r_A$ and $x_B = z \cdot h(id_B, w_B) + r_B$ with $x_A, x_B \in \mathbb{Z}_n$. The ECMQV-like public keys $\mathrm{pk}_A, \mathrm{pk}_B$ can be obtained by combining the public information together with self-certified keys, i.e. $\mathrm{pk}_A = g^{x_A} = g_0^{h(id_A, w_A)} \cdot w_A$

and $\mathrm{pk}_B = g^{x_B} = g_0^{h(id_B, w_B)} \cdot w_B$. This allows to build a self-certified version of ECMQV.

In this version, sensors only exchange their self-certified public keys and the ephemeral keys E_A and E_B, where y_A, y_B are taken at random from the corresponding finite field. It is important to note that they do not exchange nor verify *any certificates*.

Algorithm A.3 Self-certified ECMQV key derivation for entity A

Input: Elliptic curve domain parameters G, g, n, g_0, h, secret keys x_A, y_A, self-certified keys w_A, w_B and ephemeral keys E_A, E_B

Output: A secret key K_{AB} shared with entity with self-certified public key w_B

1: $m \leftarrow \lceil \log_2(n) \rceil / 2$
   $\{m$ is the half bitlength of $n\}$
2: $u_A \leftarrow (u_x \bmod 2^m) + 2^m$
   $\{u_x$ is the x-coordinate of $E_A\}$
3: $s_A \leftarrow (y_A + u_A x_A) \bmod n$
4: $v_A \leftarrow (v_x \bmod 2^m) + 2^m$
   $\{v_x$ is the x-coordinate of $E_B\}$
5: $z_A \leftarrow s_A v_A \bmod n$
6: $K_{AB} \leftarrow KDF\big(E_B^{s_A} \cdot (g_0^{h(id_B, w_B)} \cdot w_B)^{z_A} \bmod n\big)$

Like ECMQV, entity B runs the same algorithm by simply swapping the values $(x_A, y_A, w_B, E_A, E_B)$ in Algorithm A.3 with $(x_B, y_B, w_A, E_A, E_B)$ and finally obtains the same key K_{AB}.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci. *Wireless sensor networks: a survey*. Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 38, no. 4, pp. 393-422, March 2002.

[2] ZigBee Alliance, `http://www.zigbee.org/`, Retrieved on 2010.

[3] J.P. Peerenboom, and R. E. Fisher. *Analyzing Cross-Sector Interdependencies*. Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS 2007), pp. 112-119, Hawaii (USA), 2007.

[4] HART Communication, `http://www.hartcomm.org/`, Retrieved on 2010.

[5] ISA100.11a, Wireless Systems for Industrial Automation: Process Control and Related Applications, `http://www.isa.org/isa100`, Retrieved on 2010.

[6] C. Alcaraz, G. Fernandez, R. Roman, A. Balastegui, and J. Lopez. *Secure Management of SCADA Networks*. New Trends in Network Management, Cepis UPGRADE. vol. 9, no. 6, pp 22-28, December 2008.

[7] M. Healy, T. Newe, and E. Lewis. *Wireless Sensor Node Hardware: A Review*. Proceedings of IEEE SENSORS 2008, pp. 621-624, Lecce (Italy), October 2008.

[8] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler. *TinyOS: An Operating System for Sensor Networks*. On Ambient Intelligence, Springer-Verlag, ISBN 978-3-540-23867-6, 2005.

[9] P. Dutta, J. Taneja, J. Jeong, X. Jiang, and D. Culler. *A Building Block Approach to Sensornet Systems*. Proceedings of the Sixth ACM Conference on Embedded Networked Sensor Systems (SenSys'08), pp. 267-280, Raleigh (USA), November 2008.

[10] Meshnetics' ZigBit 900 Module with balanced RF output, `http://www.meshnetics.com/zigbee-modules/zigbit900/`, accessed on 2010.

[11] Ember's EM357 product datasheet, `http://www.ember.com/pdf/ember-EM300.pdf`, accessed on 2010.

[12] Jennic's JN5148 product datasheet, `http://www.jennic.com/download_file.php?brief=JN-DS-JN5148-1v2.pdf`, accessed on 2010.

[13] Nivis' VN210 sensor node datasheet, `http://www.nivis.com/Docs/Nivis_VersaNode_VN210.pdf`, accessed on 2010.

[14] Java Cryptography Architecture (JCA) Reference Guide. `http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html`. Accessed on 2010.

[15] A. Liu, and P. Ning. *TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks*. Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pp. 245-256, St. Louis (USA), April 2008.

[16] A. Menezes, M. Qu, S. Vanstone. *Some new key agreement protocols providing mutual implicit authentication*. Second Workshop on Selected Areas in Cryptography (SAC 95) 1995.

[17] L. Law, A. Menezes, M. Qu, J. Solinas, S. Vanstone. *An efficient protocol for authenticated key agreement*. Des. Codes Cryptography 2003, vol. 28, no. 2, pp. 119134.

[18] R. Sakai, K. Ohgishi, M. Kasahara. *Cryptosystems based on pairing over elliptic curve (in japanese)*. The 2001 Symposium on Cryptography and Information Security, Oiso (Japan), 2001.

[19] R. Dupont, A. Enge. *Provably secure non-interactive key distribution based on pairings*. Discrete Applied Mathematics 2006, vol. 154, no. 2, pp. 270276.

[20] M. Girault. *Self-certified public keys*. EUROCRYPT, vol. 574, pp. 490497, 1991.

[21] H. Petersen, P. Horster. *Self-certified keysconcepts and applications*. Communications and Multimedia Security 97, pp. 102116, 1997.

[22] X9 ASC. *American national standard X9.62-2005, public key cryptography for the financial services industry, the elliptic curve digital signature algorithm (ECDSA)* 2005.

[23] J. Heidemann, Y. Wei, J. Wills, A. Syed, and L. Yuan. *Research Challenges and Applications for Underwater Sensor Networking*. Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC 2006), vol. 1, pp. 228-235, Las Vegas (USA), April 2006.

[24] D. Aranha, L.B. Oliveira, J. Lopez, R. Dahab. *NanoPBC: Implementaing cryptographic pairings on an 8-bit platform*. CHILE2009.

[25] P. Szczechowiak, A. Kargl, M. Scott, M. Collier. *On the application of pairing based cryptography to wireless sensor networks*. WISEC, ACM, pp. 112, 2009.

[26] L.B. Oliveira, D.F. Aranha, E. Morais, F. Daguano, J. Lopez, R. Dahab. *Tinytate: Computing the tate pairing in resource-constrained sensor nodes*. NCA, IEEE Computer Society, pp. 318323, 2007.

[27] R. Housley, W. Polk, W. Ford, D. Solo. *RFC 3280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*, Request for Comments, April 2002.

[28] R. Roman, C. Alcaraz. *Applicability of Public Key Infrastructures in Wireless Sensor Networks*. Proceedings of the 2007 European PKI Workshop: Theory and Practice (EuroPKI'07), pp. 313-320, Mallorca (Spain), June 2007.

[29] Crossbow Technology, Inc. `http://www.xbow.com`, Retrieved on 2010.

[30] LinkQuest. Underwater acoustic modems. `http://www.link-quest.com`, Retrieved on 2010.

[31] T. Giannetsos, I. Krontiris, T. Dimitriou, and F.C. Freiling. *Intrusion Detection in Wireless Sensor Networks*. On Security in RFID and Sensor Networks, Auerbach Publications, CRC Press, ISBN: 978-1420068-399, 2009.

[32] Critical Information Infrastructure Research Co-ordination (CI2RCO). *Deliverable D12, "ICT R&D for CIIP: Towards a European Research Agenda*, 2007.

[33] S. Rinaldi, J. Peerenboom, T. Kelly. *Identifying, understanding, and analyzing Critical infrastructure interdependencies*. In IEEE Control Systems Magazine, vol. 21, pp. 11-25, 2001.

[34] S.M. Rinaldi. *Modeling and Simulating Critical Infrastructures and Their Interdependences*. In 37th Hawaiian International Conference on system Sciences, 2004.

[35] J.P. Peerenboom, R.E. Fisher. *Analyzing Cross-Sector Interdependencies*. In IEEE Computer Society, HICSS '07, IEEE Computer Society, pp. 112-119, 2007.

[36] M.P. Papazoglou. *Service-Oriented Computing: Concepts, Characteristics and Directions*. In Web Information Systems Engineering, International Conference on, Fourth International Conference on Web Information Systems Engineering (WISE'03), Roma (Italy), 2003.

[37] Modbus-IDA the architecture for distributed automation, `http://www.modbus.org/`, Retrieved on 2010.

[38] DNP3, DNP Users Group, `http://www.dnp.org`, Retrieved on 2010.

[39] IEC 60870-5-104, International Electrotechnical Commission, Retrieved on 2010.

[40] IEC 60870-6, ICCP/TASE2, International Electrotechnical Commission, Retrieved on 2010.

[41] ODVA, Open DeviceNet Vendors Association, `http://www.odva.org/`, Retrieved on 2010.

[42] A. Cardenas, S. Amin, S. Sastry. *Research Challenges for the Security of Control Systems*. In 3rd USENIX Workshop on Hot Topics in Security (HotSec'08), San Jose (USA), 2008.

[43] CERT, Carnegie Mellon Software Engineering Institute, *CERT/CC Statistics 1988-2010*, `http://www.cert.org/stats/vulnerability_remediation.html`, Retrieved on 2010.

[44] BCIT, British Columbia Institute of Technology, `http://www.bcit.ca/`, Retrieved on 2010.

[45] E. Byres, J. Lowe. *The myths and facts behind cyber security risks for industrial control systems*. In 'VDE Congress, VDE Association For Electrical, Electronic Information Technologies, British Columbia Institute of Technology and PA Consulting Group, 2004.

[46] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, M. Yarvis. *Design and Deployment of Industrial Sensor Networks: Experiences from a Semiconductor Plant and the North Sea*, In SenSys, pp. 64-75, 2005.

[47] S. Carlsen, A. Skavhaug, S. Petersen, P. Doyle. *Using wireless sensor networks to enable increased oil recovery*, Emerging Technologies and Factory Automation, pp. 1039-1048, 2008.

[48] X. Bai, X. Meng, Z. Du, M. Gong, Z. Hu. *Design of Wireless Sensor Network in SCADA system for wind power plant*. Automation and Logistics (ICAL), pp. 3023-3027, 2008.

[49] Emerson Process Management, `www.EmersonProcess.es/SmartWireless`, Retrieved on 2010.

[50] Emerson Process Management, *Chemical Manufacturer Improves Safety and Reduces Costs with Smart Wireless Solution*, `http://www.wina.org/WireSol/Documents/CaseStudy-TankCarApplication.pdf`, Retrieved on 2010.

[51] Emerson Process Management, *Lime Kiln Throughput Improves with Smart Wireless Solution*, `http://www.wina.org/WireSol/Documents/CaseStudy-LimeKiln.pdf`, Retrieved on 2010.

[52] Emerson Process Management, *Emerson's Wireless Technology Helps Chevron Improve Oil Field Personnel Safety and Increase Production*, `http://news.thomasnet.com/companystory/823530`, ThomasNet News, Retrieved on 2010.