J. Lopez, R. Roman, and C. Alcaraz, "Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Networks", *Foundations of Security Analysis and Design 2009*, LNCS vol. 5705, pp. 289-338, 2009.
http://doi.org/10.1007/978-3-642-03829-7_10
NICS Lab. Publications: https://www.nics.uma.es/publications

Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Networks

Javier Lopez, Rodrigo Roman, and Cristina Alcaraz

Computer Science Department
University of Malaga, Spain
{jlm, roman, alcaraz}@lcc.uma.es

Abstract. As sensor networks are more and more being implemented in real world settings, it is necessary to analyze how the different requirements of these real-world applications can influence the security mechanisms. This paper offers both an overview and an analysis of the relationship between the different security threats, requirements, applications, and security technologies. Besides, it also overviews some of the existing sensor network standards, analyzing their security mechanisms.

1 Introduction

Wireless sensor networks, or WSN, have evolved in the past years from a promising research field to a useful technology applicable to real-world scenarios (e.g. industrial monitoring in critical infrastructures [1]). Research on security for sensor networks have advanced as well, showing promising results such as efficient implementations of public key cryptography algorithms and lightweight self-healing mechanisms. Still there is one particular aspect of sensor network security that is commonly neglected or overlooked: the relationship between the security requirements, the features of the application and its context, and the security mechanisms. Indeed, the context and the requirements of a specific application have a great influence on the security mechanisms that should be used to protect the network. Besides, new standards for WSN are being developed, but there are some security aspects that seem to be overlooked as these standards focus mainly on securing the communications between nodes.

The purpose of this paper is twofold. Our first objective is to provide an analysis of the relationship between requirements, applications, and security mechanisms. Therefore, we will review the different security attacks that target WSN, and will explain the different security requirements that a sensor network application should fulfil in order to protect itself from those attacks. Afterwards, we will indicate explicitly how the different technologies, applications and network structures influence over the selection and provisioning of the security services. Finally, we will overview the state of the art of the security mechanisms for sensor networks, pointing out already available solutions and open issues. As for our second objective, we aim to describe the existing sensor network standards

and their security mechanisms. Consequently, we will provide an overview of these different standards, focusing on their security capabilities. Later, we will analyze not only how these standards compare with each other, but also if they offer support for implementing the previously mentioned security mechanisms.

2 Overview of Wireless Sensor Networks

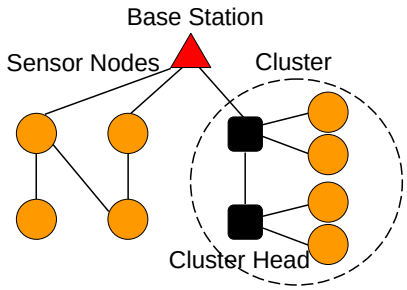

Fig. 1. An overview of the architecture of WSN

The ability to perceive the physical world is not inherent to the nature of computer systems: they are tightly tied to the realm of the abstract. The existence of sensor hardware tries to build a bridge between the abstract world and the physical world. These sensors are devices that can measure a physical quantity (e.g. temperature, humidity) and convert it into a digital signal. Using these sensors, computer systems ranging from the simplest washing machine to the Large Hadron Collider (a particle accelerator located at the European Organization for Nuclear Research (CERN) [2]) can acquire and process information coming from the physical world. This ability to "feel" the world is usually embedded in the design of a computer system, e.g. sensors in a washing machine are integrated within the system from the initial design. However, it would be particularly interesting to make this ability available as an off-the-shelf component. As a result, any computer system, regardless of its design, could be able to perceive the physical world. Such is the task of Wireless Sensor Networks, or WSN.

The structure of a wireless sensor network can be seen in Figure 1. A wireless sensor network is composed by two types of devices: sensor nodes, and base stations. The sensor nodes, also known as motes or simply nodes, are small and constrained devices that have the ability to "feel", "think", "talk", and "subsist". They can "feel", because they can sense the physical features of their surrounding (e.g. temperature, humidity, radiation, vibration) using hardware sensors. They can "think", because although they are highly constrained in both computational power and memory, they are capable of processing information on their own. They can "talk", because they are equipped with wireless transceivers,

and can collaborate towards a common goal. Finally, they can "subsist" because they are in most cases powered by batteries, and can survive in their deployment field for more than a year if their internal operations are optimized.

Regarding the base station, it is a more powerful device that usually behaves as an interface between the services provided by the sensor nodes (the "data acquisition network") and the users of the network (the "data dissemination network"). Normally, the base station collects all the information coming from the sensor nodes and stores it for later use. Also, it can issue control orders to the sensor nodes in order to change their behaviour. While it would seem that wireless sensor networks are highly dependent of the existence of this base station, the architecture of the network is not centralized. Sensor nodes can operate in a decentralized fashion, managing themselves without accessing to the base station.

A powerful simile that can be used to illustrate the structure of wireless sensor networks is to consider them as "living beings". The sensor nodes behaves as "cells", since they all belong to the same body (WSN), are usually loaded with the same "DNA" (program), and cooperate unselfishly towards a common goal. On the other hand, the base station could be considered as the "brain", since it receives and processes all the physical information coming from the "cells", and can also send control information to them. Note that, in terms of communications, the "cells" also behave as "nerves", since they transmit (wirelessly) the information sensed by other "cells" to the "brain".

The services offered by wireless sensor networks can be classified into four major categories: monitoring, alerting, provisioning of information "on-demand", and actuating. As for the first case, sensor nodes can continuously monitor certain features of their surroundings (e.g. measuring the ambient noise level) and timely send such information to the base station. Secondly, sensor nodes can check whether certain physical circumstances (e.g. a fire) are occurring, alerting the users of the system when an alarm is triggered. In the third case, the network can be queried about the actual levels of a certain feature, providing information "on-demand". Finally, sensor nodes can be able to change the behaviour of an external system (e.g. an irrigation system) according to the actual state of the context (e.g. humidity of the soil)[1]. Due to the computational capabilities of the sensor nodes, it is possible to reprogram the network during its lifetime, or even use it as a distributed computing platform under specific circumstances.

Finally, wireless sensor networks can be organized in a completely distributed way (flat configuration), but they can also implement levels of hierarchy (hierarchical configurations). In flat configurations, all the nodes contribute in the decision-making process and participate in the internal protocols, like routing. Conversely, in hierarchical configurations the network is divided into clusters or group of nodes. Inside a cluster all organizational decisions, like data aggregation, are made by a single entity called cluster head. It should be noticed that it is also possible to have a combination of the two previous configurations into

[1] These type of sensor networks are also known as Wireless Sensor and Actuator Networks (WSAN).

the same network; for instance, to avoid situations where the "spinal cord" of the network - the cluster heads - fail and the information must be routed to the base station.

Aside from their structure, services, and organization, there are some features that help to define what a wireless sensor networks is: specificity, autonomy, self-configurability, long lifetime, deployment location, and mobility. The embedded intelligence of sensor nodes allow them to perform various tasks, but their services, protocols and architectures are highly dependent on the functionality of the network, due to their inherent constraints. Also, unlike other more capable devices such as PDAs, there is no human user directly controlling the sensor node: motes are usually accessed through other motes or through the base station. In fact, sensor nodes can set up their services and function properly in situations where there is no central control available. Due to this autonomy, sensor nodes need to self-configure and maintain themselves during the lifetime of the network. Precisely, a wireless sensor network can function for long periods of time, ranging from several days to one or two years. Regarding the network deployment, sensor nodes are usually deployed near the physical source of the events, and the exact deployment location of these sensor nodes is usually not known in advance. Finally, sensor nodes are usually not mobile, although there might be scenarios where some sensor nodes or even the base station need to move (e.g. tracking a target).

3 Security Aspects of Wireless Sensor Networks

In any environment, either physical or logical, there exists the need of maintaining someone or something safe, away from harm. This is the role of security. On any computer-related environment, security can be considered as a non-functional requirement that maintains the overall system usable and reliable, protecting the information and information systems. In fact, in wireless sensor networks, security is of paramount importance: the network must be adequately protected against malicious threats that can affect its functionality. Due to the role of sensor networks as a "sensory system", any disturbance in a sensor network may have consequences in the real world. However, sensor networks are especially vulnerable against external and internal attacks due to their peculiar characteristics.

- The devices of the network (i.e. sensor nodes) are constrained in terms of computational capabilities, memory, communication bandwidth, and battery power. As a result, it is challenging to implement and use the cryptographic algorithms and protocols required for the creation of security services.
- In most cases, it is easy to physically access sensor nodes: they must be located near the physical source of the events. Since nodes are not tamper-resistant due to cost constraints, any human user or machine can reprogram them or simply destroy them.

– Any internal or external device can access to the information exchange because the communication channel is public. Besides, attacking the availability of the wireless channel is not a complex task.
– It is a difficult task to monitor and control the actual state of the elements of the network due to the inherent distributed nature of sensor networks. Any failure in any of its elements may remain unnoticed, or the actual cause of the malfunction may not be known. Besides, a sensor network can be attacked at any point.

3.1 Security Threats

Due to their previously shown inherent vulnerabilities, sensor networks have to face multiple passive and active attacks that may easily hinder its functionality and nullify the benefits of using its services. Passive attacks are able to retrieve data from the network, but do not influence over its behaviour. On the other hand, active attacks directly hinder the provisioning of services. The different threats that target sensor networks will be detailed in the following paragraphs, and they can be categorized as follows:

– *Common Attacks*. As the wireless medium is used as the main transmission channel in WSN, it is easily subject to various types of attacks, either passive (eavesdropping) or active (data injection).
– *Denial of Service Attacks* (DoS). These attacks prevent any part of WSN from functioning correctly or in a timely manner. Such attacks can target the communication channel (e.g. jamming) or the life of the nodes themselves (e.g. power exhaustion).
– *Node Compromise*. An embedded device is considered being compromised when an attacker, through various means, gains control or access to the node itself after it is being deployed. These attacks are usually utilized as a foundation for more powerful, damaging attacks.
– *Side-channel Attacks*. An adversary can monitor certain physical properties of the nodes, such as electromagnetic emanation, whenever it performs a cryptographic operation. If the recorded physical values are influenced by the secret key, then the adversary can extract information about that key.
– *Impersonation Attacks*. A malicious sensor node can create multiple fake identities (sybil attack), and also can create duplicates with the same identity (replication attack). These types of attacks are also the initial step which enables the attacker to conduct a wide range of malicious attacks.
– *Protocol-specific Attacks*. Some essential protocols used in WSN, such as routing, aggregation, and time synchronization, are targeted by specific attacks that aim to influence the internal services of the network.

By using the so-called *common attacks class*, a malicious adversary uses a device that does not belong to the sensor network in order to access to the contents of the communication channel. The simplest instance of common attack is eavesdropping. It can be defined as the interception of information or data by

an unintended party. Due to the broadcast nature of the communication channel, any adversary (using a mote or a more powerful device such as a PDA) can sniff out packets at a particular frequency, obtaining confidential information about the state of the network and the physical parameters sensed by the nodes. As the eavesdropping attack has an inherent passive nature, it does not directly influence over the behaviour of the network.

However, the acquired information from passive attacks can be used to perform active attacks. The effects of active attacks are far more destructive: adversaries can create fake events or hide problematic situations, and can even introduce bogus control information. One of these active attacks is message modification, where an adversary intercepts and modifies the packets' content meant for the base station or intermediate nodes. Another active attack is message replay. In this attack, the adversary reuses valid transaction messages or packets' content with malicious intent. The adversary performs a replay attack by first intercepting a valid critical transaction data packet and then re-transmitting at a later time. Lastly, attackers can use message injection to fabricate and send out false data into the network, maybe masquerading as one of the nodes.

One special class of active attack, known as *Denial of Service (DoS)*, deserves a category of its own. In this kind of attack, the objective of the malicious adversary is simple: to avoid the provisioning of services. As these services are published by the sensor nodes through a wireless channel, the most basic DoS attacks can target the nodes themselves (power exhaustion attack) or the communication channel (jamming attack). In the power exhaustion attack, an attacker imposes a particularly complex task to a sensor node in order to deplete its battery life. Sensor nodes usually have a limited supply of energy, thus this attack is particularly dangerous. Besides, as sensor nodes have limited computational power, this attack can also slow down their reaction time. An example of an expensive operation is the verification of a cryptographic signature using public key cryptography. An attacker can take advantage of the complexity of this operation by repeatedly sending fake signatures to force the receiver to check their correctness. Power exhaustion attacks are not limited to only CPU attacks: an attacker can target the MAC protocol of the WSN, effectively preventing nodes from entering their duty/sleep cycle an wasting their batteries [4].

Jamming is the primary physical layer DoS attack against WSN. In a jamming attack, the attacker constantly emits radio frequency signals that do not follow an underlying MAC protocol, thus any member of the network in the affected area will not be able to send or receive any packet. The energy requirements of this attack are very high, as the attacker must flood the communication channel with noise. There are some optimizations to the basic jamming attack, such as the random jamming, where the attacker alternates between sleep and jamming to save energy, or the reactive jamming, where the jam signal is transmitted only when the attacker senses traffic [3]. Finally, another clever optimization that reduces the energy consumption of the attacker is to target MAC protocols on the link layer [4][5], e.g. by jamming only request-to-send (RTS) packets. As a side note, it should be mentioned that DoS attacks can be performed by using

some of the attacks explained in other categories [6], although those attacks are usually more complex and can be used to disrupt other functional elements of the network (e.g. the authenticity of the physical/control data).

Most of the previously shown attacks can be performed by outsiders: attackers that do not have access to the network elements and services. However, if an attacker have access to the network as one of its elements, i.e. as an insider, it is possible to perform attacks that are more subtle and devastating. The first step to become an insider is to compromise a node, usually by performing *node compromise attacks*. A sensor node can be considered compromised when an attacker, through various means, can either read or modify its internal memory. The ultimate goal of this attack is, in most cases, to obtain the secret keys stored within a trusted node in order to infiltrate a mole inside the network. Attacks that can lead to a node compromise are invasive or non-invasive. In an invasive attack, the attacker physically breaks into the hardware by modifying its hardware structure (e.g. using focused ion beam, or drilling a hole in the storage media). On the other hand, in non-invasive attacks the data is taken from the hardware device without any form of structural modification done to the device itself. Invasive attacks usually fall under the category of side channel attacks, as these attacks obtain confidential data directly from the chips of the nodes.

Regarding non-invasive attacks, they usually take advantage of the hardware interfaces of the nodes. One example is the JTAG interface [8]. This interface enables accessing and controlling of the signal levels on the processor chip, and is also used for debugging purposes. Through the use of an AVR ICE JTAG programming tool, an attacker can dump all the information from the program flash, the EEPROM and also the SRAM. As a result, the attacker can replicate the functionality of the node to facilitate the integration of the malicious node. While most of these non-invasive attacks simply aim to obtain information from the node, there exist more advanced attacks that are capable of injecting code inside a working sensor node. For example, it is possible to exploit the serial bootstrap loader (BSL) of certain models of the Texas Instruments MSP430 low-power microcontrollers with the aim of extracting or replacing the firmware [9]. Even more, it is possible to inject malicious code remotely in AVR-based nodes by exploiting buffer overflow vulnerabilities [10].

In order to compromise a node, it is also possible to attack its hardware through *side-channel attacks*. The main objective of side channel attacks is to obtain confidential data stored within the node. Most attackers focus on obtaining security credentials such as secret keys, since these credentials will provide the attacker with a powerful tool capable of crafting more powerful attacks. Side channel attacks can be classified in the following categories: passive vs. active and non-invasive vs. semi-invasive vs. invasive. Passive attacks extract information from the device merely by observing physical properties of the devices, while active attacks involve the manipulation (tampering) of the device itself. In contrast, non-invasive attacks do not manipulate the device substantially, while semi-invasive attacks depackage the device but do not make direct electrical contact with the chip's surface, and invasive attacks have practically no limits to

the measures which can be taken to extract the information of the device (e.g. probing station, focused ion beam). Note that not all semi-invasive or invasive attacks are active attacks: passive semi-invasive attacks may try to just read sensitive data from memory components, and passive invasive attacks can use a probe station to sense valuable data signals.

Specific examples of side-channel attacks are power analysis attacks, electromagnetic attacks, and timing attacks [7]. In power analysis attacks, the adversary studies the power consumption of the devices, focusing mainly on the energy used by cryptographic operations. For performing these attacks, it is possible to either use single power traces to look for distinguishing features (Simple power analysis, SPA) or use larger numbers of power traces alongside with powerful statistical methods (Differential power analysis, DPA). Electromagnetic attacks (or EM attacks) are similar to power analysis attacks, since they also analyze power traces with simple (SEMA) and differential (DEMA) methods. However, they derive the power traces from electromagnetic emanations, collected by EM probes. Beyond simple and differential analysis, EM attacks can employ more advanced techniques, such as adding spatial information to the measurement data, or analysing the frequency domain rather than the time domain. Finally, as the execution time of cryptographic algorithms often shows slight differences dependent on the input of the algorithm, timing attacks exploits the variance in execution time for different branches in the cryptosystem.

Once an attacker becomes an insider, it is easier to perform *impersonation attacks*. For this particular class of attack, the goal of the adversary is to make the victim believe that it is communicating with an impersonated entity. As a result, a malicious node will interact with other nodes as one trusted member, but at the same time it can manipulate the internal behaviour of the network whenever the adversary needs it. Impersonation attacks can either replicate and insert duplicate nodes back into selected regions of the network (node replication attack or clone attack) or use multiple identities to deceive other sensor nodes (sybil attack). In node replication attacks, the attacker only needs to subvert one node in order to create an army of clones following his orders. These clones can not only manipulate the internal operations of the network, but also exert a strong influence over those processes that require of a majority vote. As for sybil attacks, a sybil node can either fabricate new identities or steal them from legitimate nodes [11]. Sybil nodes can be able to execute powerful attacks, disrupting several of the functions that may be conducted on a WSN including data aggregation, voting, routing and fair resource allocation.

Beyond impersonation, an insider can perform *protocol-specific attacks*, attacking those "core" protocols needed by the network such as routing protocols, aggregation protocols, and time synchronization protocols. Attacks against routing protocols in a WSN fall into one of the following categories [12]: corruption of the internal control information such as the routing tables (Spoofed Routing Information), selective forwarding of the packets that traverse a malicious node depending on some criteria (Selective Forwarding), creation of a 'wormhole' that captures the information at one location and replays them in another

location either unchanged (Wormhole attack) or tampered (Sinkhole attack), creation of false control packets during the deployment of the network (HELLO Flood Attack), and creation of false acknowledge information (Acknowledgment Spoofing).

Data aggregation protocols combine information coming from the same area in order to reduce the overall communication overhead. As these protocols need to use routing protocols in order to fuse information and forward it to the base station, every attack that target the routing infrastructure can also be used to hinder the aggregation process. Most of these attacks try to discard data, either selectively or indiscriminately. Though losing data is a problematic situation for the network, this is not the primary type of attack against aggregation: most attacks focus on falsifying information. If an aggregator node is being controlled by an adversary, it can easily ignore the data received from its neighbours and create false reports. Moreover, trusted aggregators can still receive false data from faulty nodes or from nodes being controlled by an adversary.

Regarding time synchronization, it is needed because as the time obtained from clocks of different nodes may differ due to different starting times (offset), inaccurate quartz crystals (skew), or ambient influence (drift), it is necessary to synchronize these clocks in order to maintain a global notion of time [13]. Most time synchronization protocols rely on two neighbouring nodes adjusting their local clocks by means of sender-receiver (mutual synchronization) and receiver-receiver (beacon signals) protocols. In these scenarios, the main objective of an attacker is to deceive other nodes into thinking that an incorrect time is accurate. Besides internal attacks, where the attacker can outrightly lie about the value of its internal clock, an attacker can use the following external attacks: manipulation of the contents of the negotiation messages through message forging and replay, and delaying the messages exchanged in the negotiation process by means of a pulse-delay attack.

3.2 Security Requirements

As we have previously seen, sensor networks are vulnerable to external and internal attacks. The effects of those attacks in the network are not trivial, since they can render the services of the network useless. It is clear that there is the need of using security mechanisms either to prevent the attacks from influencing over the functionality of the network or to minimize the adverse effects of such attacks. By using the security mechanisms, it can be possible to enforce in sensor networks the following security properties:

- *Confidentiality.* This property tries to fulfil the following principle: A given message must not be understood by anyone other than the desired recipients. While confidentiality is an important security property, it may be not mandatory in certain scenarios where the data is public by itself (i.e. the temperature of a street) and no other information can be derived from it. However, there are particular situations and scenarios where the physical data obtained by the network can be deemed as sensitive, and should not be

read by external entities. Data can be considered sensitive due to its inherent nature (e.g. patient data such as temperature), the nature of the context (e.g. a private household, a military setting), or the nature of the sensed entities (e.g. a protected animal like a panda). Besides, certain control data exchanged by the nodes, such as security credentials and secret keys, must be hidden from unauthorized entities.

- *Integrity.* This property states that the data produced and consumed by the sensor network must not be maliciously altered. Unlike confidentiality, integrity is, in most cases, a mandatory property. The wireless channel can be accessed by anyone, thus any peer (outsiders and insiders) can manipulate the contents of the messages that traverse the network. Even more, data loss or damage may occur due to the harsh communication environment, and in the worst case the network will accept corrupted data. As the main objective of a sensor network is to provide services to its users, the sensor network will fail in its purpose if the reliability of those services can not assured due to inconsistencies in the information.
- *Authentication.* Informally, data authentication allows a receiver to verify that the data is really sent by the claimed sender. This security property is quite important in sensor networks. In fact, without authentication the barrier between external and internal members of the network would not exist, as any outsider could claim that it is a registered member of the network. Moreover, even existing network members could easily pose as their neighbours. This situation would encourage many problematic situations, such as adversaries forging the whole packet stream by injecting additional packets, and nodes accepting false administrative tasks (e.g. network reprogramming).
- *Authorization.* As for this property, it states that only authorized entities (sensor nodes and base station) can be able to perform certain operations in the network (e.g. information providing, controlling the system). Since a sensor network can be considered as one single entity, where all nodes perform the same tasks and acknowledge the role of the base station as manager and supervisor, it could be supposed that any authenticated device is inherently authorized to perform its tasks. Nevertheless, there might be situations (e.g. when nodes actuate over physical systems) where some members of the network need to have a proper authorization in order to perform certain tasks. Is in these situations where authorization must be taken into account.
- *Availability.* The users of a sensor network must be capable of accessing its services whenever they need them. As a result, the different hardware and software elements of the network must be robust enough to be able to provide services even in the presence of malicious entities or adverse situations. Nevertheless, this property is related not only to the protection of the services, but also to the security mechanisms themselves: all protection mechanisms should be as energy efficient as possible in order not to quickly drain the batteries of the nodes.
- *Freshness.* Sensor networks are very data-centric: they exist due to the physical data they have to collect from an environment. One important property

that arises from this fact is freshness: the data produced by the sensor network must be recent. Consequently, the messages of the network should aim to reduce the network delay to the smallest possible value, even in unfavourable situations where the network is under attack. While in some networks is admissible to have a certain delay, in certain scenarios the information must be received as soon as possible (e.g. alerts in nuclear power plants). Freshness is not only linked to delay but also linked to forgery: if an adversary success on replaying an old message inside the network, the data not only will be useless, but also harmful (e.g. it may inform of a non-existent alarm).

- *Forward and Backward Secrecy.* As new sensor nodes can be deployed whenever other sensor nodes fails, there are two properties that need to be considered: forward secrecy, where a sensor should not be able to read any future messages after it leaves the network, and backward secrecy, where a joining sensor should not be able to read any previously transmitted message. These properties may not be important in certain scenarios, where there is no need to hide the contents of the network from old nodes and new nodes authorized to perform the same tasks as their partners. However there are other scenarios where these properties must be taken into account, such as in networks with nodes that must be authorized to perform certain tasks.
- *Self-Organization.* One specific property related to the autonomous nature of sensor networks is self-organization: sensor nodes must be independent and flexible enough to autonomously react against problematic situations, organizing and healing themselves. These problematic situations can be caused either by external or internal attackers trying to influence over the behaviour of the elements of the system or by extraordinary circumstances in the environment or in the network itself. This is an essential property to the functioning of a sensor network and optimal resource use during its lifetime. It is desirable that all possible problems that may occur can be detected and prevented without any margin of error. However, as the previous statement may not be realistic, nodes should be able to at least adapt their activities to assure the continuity of the services.
- *Auditing.* The elements of a sensor network must be able to store any significant events that occur inside the network. This property is necessary due to the autonomous nature of the nodes. As users do not operate the sensor nodes directly, but through the base station, they may not be able to know about the existence of a certain event unless the nodes store it. Besides, if the whole sensor network fails, auditing information can be used to analyze the behaviour of the system prior to the failure. This property is also closely related to self-organization: in order to adjust their behaviour, sensor nodes must be able to know the state of their surroundings. Note that the inherent memory constraints of sensor nodes complicate the task of storing audit data.
- *Non-repudiation.* While non-repudiation is not considered in the existing literature as an important security property for most sensor networks, it may be necessary to at least consider its applicability in certain contexts where

sensor nodes monitor critical components, as acknowledging the reception and processing of serious alarms is of key importance. This property is described as follows: a node cannot deny sending a message it has previously sent. Note that non-repudiation can also consider repudiation of receipt, where the recipient tries to deny the reception of the message. For achieving non-repudiation, it is necessary to produce certain 'evidence' in case a dispute arises. Using the evidence, it is possible to prove that a device of the network performed a task.

– *Privacy and Anonymity.* These security properties are very important in those scenarios where the location and identities of the base station and the nodes that generated information should be hidden or protected. For example, any network that monitors endangered species should provide no clues on their physical location. Also, in a battlefield, it would be important to not be able to distinguish whether a certain signals belongs to a soldier or a vehicle. In contrast, there are situations where this property should not be enforced: in an earthquake rescue situation locating the source nodes (if the nodes are worn by, for example, dogs) is an absolute must. Note that this property can transcend beyond the technological dimension and affect its social environment, since sensor networks could be used as a surveillance tool to collect data about the behaviour of human beings.

4 The Influence of Technologies and Applications in Security

Although it would seem that every security property needs to be completely enforced in order to have an entirely secure sensor network, it is usually not the case [15][16][17]. The role of security is to protect the network against the existing threats that may affect it, and not all sensor networks are equally affected by all threats. As a result, it is necessary to analyze how a specific sensor network could be affected by possible attackers, and if its elements are suitable enough to implement the security countermeasures that are needed. In this section we will review the different technologies associated to sensor networks and the context of the applications, and how these technologies and applications influence over the selection of the security mechanisms.

4.1 Technologies: Hardware and Software

A sensor node is typically made up of four basic components: sensing unit, transceiver, processor unit, and power unit, as seen in Figure 2. 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

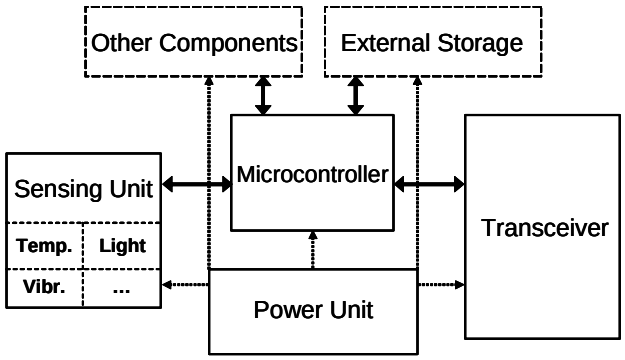

Fig. 2. HW Elements of a Sensor Node

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 [18].

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 adequate 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 [18].

As for the evolution of transceivers used in sensor nodes, the first prototype platforms tended to use transceivers that conformed to proprietary standards or to no standards at all. Most of these early transceivers used narrowband communication, which typically operate at lower frequencies (433Mhz - 915Mhz) and provide low data rates (10 Kbps - 76.8 Kbps), but have less power consumption and faster wakeup times. After the appearance in 2003 of the IEEE 802.15.4 standard [78] for low-rate wireless personal area networks (PANs), most sensor nodes started to use transceivers that complied with this standard. This standard uses wideband communication (faster, more robust, more power-demanding), operating in the 2.4 Ghz frequency band with a maximum (theoretical) throughput of 250 Kbps. Other mote platforms chose to integrate transceivers that implemented the Bluetooth standard. While this standard is widely adopted in mobile phones and other electronic devices, its power consumption is significantly higher than the IEEE 802.15.4 standard. Notice that, in all cases, the energy consumption of the transceiver is far greater than the energy consumption of

the microcontroller, thus sensor nodes are encouraged to do as much in-network processing as possible [14].

Another indispensable component in sensor nodes is the processor unit. Actually, sensor nodes use microcontrollers instead of microprocessors. They are especially suitable for the wireless sensor networks environment due to their cost-effectiveness: a microcontroller used in a sensor node has enough computational capabilities and memory for executing simple tasks while consuming as less energy as possible. The selection of a microcontroller depends on what services has to provide to the node in terms of energy consumption, instruction memory and RAM memory, storage, speed, and external IO ports. 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 sensor nodes, TinyOS [19], 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.

	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.

As of 2009, 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. [20]) that future versions of these nodes will achieve around 16 KB of RAM and 256 KB of instruction memory. It should be noted that the CPU speed and the amount of memory available to the node are not the only factors that must be taken into account when selecting a microcontroller: other factors include a low active current, a wide operating voltage range, a 16-bit sleep timer, fast wakeup from sleep, and the existence of direct memory access (DMA) channels that can operate while the CPU sleeps [20]. Class III nodes are not linked to these requirements: they already provide the computer horsepower required by more complex applications at the expense of higher energy consumption, reduced lifetime, and higher costs.

One of the components of sensor nodes that deserve an analysis of its own is the power unit. This subsystem supplies power to the node, allowing it to survive for weeks, months or even years without any maintenance. Most class II sensor nodes are powered by AA batteries, while class III sensor nodes are

usually powered by high energy density batteries (e.g. based on lithium-ion chemistry). As nodes will stop functioning once the battery voltage drops below the recommended operation voltage range of the node and its components, the protocols and services of a sensor network have to take energy consumption into consideration.

It is also possible to harvest energy from the environment to power the electronic circuits of a node. The main sources of ambient energy considered suitable for use with WSNs are solar (generated by sunlight or artificial light), mechanical (generated by the movements of objects) and thermal energy (generated by temperature differences between two objects) [21]. Although there are many challenges in the design of energy harvesters for sensor nodes, such as reducing the size of the harvesters while storing the energy in an efficient way, the state of the art is advanced enough to offer off-the-shelf components that can provide some power to the nodes [22]. Besides, there are also commercial nodes like eKo [23] that use renewable energy (i.e. solar energy). Note that in most cases ambient energy is used only to recharge the batteries of a node, but there are also some research lines that pursue the implementation of long-lived sensor networks that rely only on harvested energy in order to operate [21].

Security Analysis. The different hardware components used in a sensor node have a great influence on both the nature and the capabilities of the different security primitives and security protocols that can be used in a sensor network. For transceivers, the main influence factors are bandwidth, energy consumption, and channel error rate. The speed of the wireless channel will influence on the completion time of the security protocols, and also will determine the overhead produced by any security mechanism that protects the confidentiality, integrity, and authentication of messages. As for the energy consumption, if the transceiver spends too much energy on sending and receiving messages, it is necessary to reduce both the message size and the number of steps of the security protocols. Moreover, the reliability of the wireless channel will affect the design of the security protocols, as they must be robust against failures in the communication.

	MICA2	MICAz	UWM2000	UWM4000
Working range	150 m	100 m	1500 m	4000 m
Throughput	19.2 Kbit/s	250 Kbit/s	9600 bit/s	4800 bit/s
Tx. consumption	81mW	52.2mW	4000 mW	7000 mW
Rx. consumption	30mW	59.1mW	800 mW	800 mW
μJ per bit (Tx)	4.12 μJ	0.204 μJ	416.66 μJ	1458.33 μJ
μJ per bit (Rx)	16.8 μJ	16.8 μJ	83.33 μJ	166.66 μJ

Table 2. Analysis of the energy consumption of acoustic modems.

A specific example of the influence of the transceiver in the security protocols can be found in underwater environments [24]. Here, it is unpractical to use radio frequency transceivers, because of the severe attenuation factor presented by

water. It is then necessary to use specific underwater acoustic modems, which have different features than RF transceivers: they are highly unreliable, their bandwidth is very limited, and sending or receiving one bit of information carries a high energy penalty. A table comparing radio transceivers and acoustic modems can be found at table 2. Due to these differences, certain CPU-intensive identity-based key establishment protocols such as Sakai, Ohgishi and Kasahara (SOK) are more optimal than other public key-based protocols like Elliptic Curve Menezes-Qu-Vanstone (ECMQV). The reason is simple: in this underwater environment, where the energy consumption and the channel error rate of the channel are very high, the number of messages exchanged in a security protocol should be reduced to a minimum. Identity-based protocols need to exchange only one message: the ID of the node is used as its public key.

The processor unit (its memory and CPU power) also has an influence over the security mechanisms and protocols. The amount of memory available to the node dictates how many mechanisms, both security-related and application-related, can be included inside it. In sensor nodes with limited memory, it is necessary to achieve a balance between the different components. If the application is too complex, there will be little room for security mechanisms. Conversely, if the security mechanisms occupy too much space, it will be very difficult to implement the application logic. Besides, it should be necessary to optimize the use of the security primitives. For example, by using a symmetric cryptography algorithm like AES, it is possible to obtain message authentication codes through the CMAC mode of operation. On the other hand, nodes with more available memory can implement more security mechanisms if needed. Memory is also important for holding important security data such as credentials. Precisely, the low amount of memory available to the nodes is one of the reasons that have made the field of key management systems one of the most active fields of sensor networking research [25].

The CPU power of the microprocessor dictates how much time is needed to execute a security primitive. If the microcontroller is too slow, any security protocol that is based on complex security primitives won't be practical for specific situations. For example, in scenarios where sensor nodes are mobile, two nodes that meet for a short period of time will need to use a fast protocol to perform a handshake and exchange information. However, those nodes won't be able to communicate successfully if they use protocols based on CPU-intensive primitives such as identity-based encryption and signatures. Note that there are other lightweight primitives (such as stream ciphers [28]) that do not require of a fast microcontroller, or applications with low bandwidth requirements where the overhead imposed by the security primitives is negligible. On these cases, there is no need to have a class III microcontroller where a class II microcontroller can provide the necessary services.

As for the power unit, the energy contained within the battery dictates how many operations can be performed before the sensor node disappears from the network. As a result, the lifetime of the network will be increased if energy-consuming operations, such as certain security primitives and protocols, are

scarcely used. However, it can be possible to use batteries with higher capacities. For example, D cells have a typical capacity of 12000mAh, while AA cells have a typical capacity of 2400mAh [2]. As a result, for typical class II sensor nodes, the lifetime of the network will increase by a factor of 5. In scenarios where the size of the sensor nodes is not a problem and applications can benefit from expensive security operations, it is possible to use high capacity batteries. Note that the opposite also holds: in certain scenarios it is mandatory to use batteries of lower capacity, mainly due to size requirements. Here, the execution of complex security primitives and protocols must be severely limited.

4.2 Applications

The evolution of sensor networks into a generic wireless "sensing layer" for computer systems has opened a wide range of application possibilities. These kind of networks can not be considered as the "panacea" for all the sensing needs of computer systems, since they are not especially suitable for very complex applications (such as the Large Hadron Collider [29]) or applications with hard Quality of Service (QoS) requirements. Nevertheless, there are many applications that can take advantage of the inherent characteristics of WSN (survivability, independence, reactivity, low cost, easy to maintain). According to Culler et. al. [30], those applications can be classified into the following categories:

- *Monitoring space.* The sensor network simply monitors the physical features of a certain environment. This category includes applications such as environmental and habitat monitoring, precision agriculture, indoor climate control, surveillance, treaty verification, and intelligent alarms.
- *Monitoring things.* The sensor network controls the status of a physical entity. This category includes applications such as structural monitoring, ecophysiology, condition-based equipment maintenance, medical diagnostics, and urban terrain mapping.
- *Monitoring interactions.* The sensor network monitors the interactions of things (both inanimate and animate) with each other and the encompassing space. This category includes applications such as wildlife habitats, disaster management, critical (information) infrastructure systems, emergency response, asset tracking, healthcare, and manufacturing process flow.

While all the application areas presented in the previous classification are mere ideas of where wireless sensor networks could be applied, the research community has already proven the usefulness of wireless sensor networks in real-world settings. Some examples of prototypes and research areas include: Monitoring of ageing infrastructures, critical (information) infrastructures, surveillance, detecting equipment vibration, control of vineyards, water quality analysis, control of glacier behaviour, monitoring of an active volcano, habitat monitoring, firefighters assistance, monitoring of assisted-living residents, healthcare, smart

[2] Note that D cell batteries are approximately 5 times bigger than AA batteries.

environments, checking availability of washing machines, and optimization of HVAC systems. In fact, wireless sensor networks have already jumped from the research laboratories to the commercial world, with applications such as precision agriculture, pipeline and freighter monitoring, management of critical infrastructures, and many others.

The characteristics of the applications and their context have a great influence on the structure and the elements of a sensor network. This is quite normal, as the requirements of an application will dictate how software and hardware elements should work in order to achieve the desired functionality. Examples of these influencing factors are lifetime, complexity, size, mobility, network deployment, and environment. Short-lived networks can provide its services without taking energy consumption as a primary factor in the design of the network. Complex applications may need of powerful sensor nodes that can handle the required standards and services. Smaller networks usually do not have scalability problems, while big networks need to consider not only scalability but also other issues like maintaining and monitoring the overall structure. If some devices are mobile, the network needs of specific mobility-aware protocols that can handle neighbourhoods where nodes and base stations appear and disappear. If the deployment location of nodes is known in advance, it is possible to preload information within the nodes that will optimize the behaviour of the internal protocols (e.g. provide default routing tables that can be used to offer the services of the network as soon as possible). Finally, in harsh environments it is necessary to include extra mechanisms (e.g. more redundancy) in order to preserve the reliability and robustness of the network.

Security Analysis. The aforementioned examples are only a small subset of the influence that the characteristics of the applications have on sensor networks. In fact, security is also influenced by the characteristics of the applications and their context. As a first step, it is necessary to obtain the security requirements of the application and to quantify the risks and consequences of a failure in the network security. Starting from this point, it is possible to know not only what security mechanisms are needed, but also the actual importance of every mechanism. However, this is not enough. There are different approaches for implementing the security mechanisms (cf. [25] for examples in Key Management Systems), but not all approaches can be optimally applied to a certain scenario. It is therefore necessary to choose the implementations of the security mechanisms that are more suitable for the context of a specific application.

An example of application requirement analysis and risk assessment for sensor networks can be found in [26]. This paper provides an analysis of the security requirements of sensor networks that monitors large suspension bridges and underground tunnels, and it shows how the importance of the security mechanisms can be derived from the application and security requirements. For example, the existence of node actuators is not considered in both scenarios, as operators can cross-check the sensor readings with other clues. Therefore, there is no need to include actuator-related security mechanisms. On the other hand, the integrity of the sensed data is very important for both applications, as false positives

can force maintenance operators to waste time trying to locate and repair non-existent faults. Finally, some requirements like confidentiality and availability do not have the same importance. In the suspension bridge scenario, sensor readings are not expected to show anything extraordinary or embarrassing, and it is expected that the bridges will not depend exclusively on the sensor readings. In contrast, the information from underground tunnels should be accessed only by the operators before anyone else, and any issue that affects the availability of the sensor readings will hinder the operator capabilities of reacting in real time to possible problems.

As for the suitability of the security mechanisms, an application designer must select those security algorithms and protocols that provide advantageous properties for the application. Therefore, it is necessary to discover those properties (e.g. low memory footprint, scalability, high energy consumption), and link them afterwards to the different application requirements. The protocols and algorithms that fulfil most requirements will be considered as suitable for the application. An example of this type of analysis can be found in the area of Key Management Systems (KMS) [27]. The following list details the different properties of KMS, and explicitly shows how the application requirements directly influence over choosing a certain security mechanism:

- Memory Footprint (ROM and RAM used for the protocol). The importance of this property will be linked to the complexity of the application. If the application is complex, a KMS with a high memory footprint will not be useful in most cases.
- Communication Overhead (Number of messages exchanged between peers). The following scenarios require of KMS with a small communication overhead: scenarios where the communication channel is unreliable or prone to errors, scenarios where the network must advertise itself as less as possible.
- Processing Speed (Computational cost of the protocol). The processing speed is a critical property whenever it is crucial to set up a secure channel between two previously unknown nodes as fast as possible.
- Network Bootstrapping (Confidentiality of the bootstrap process). For applications where the deployment environment is secure enough, confidentiality is not an issue. Problems may arise whenever the deployment area is public or when the information managed by the sensor nodes is important.
- Network Resilience (Resistance against stolen credentials). The need for network resilience increases as the chances of a node being subverted by an adversary becomes higher.
- Global Connectivity (Existence of a "key path" between any node of the network). This is, in most cases, an essential property, because in many scenarios all nodes are equally important for providing the network services.
- Local Connectivity (Existence of a shared secret between neighbour nodes). This will assure that most nodes will be able to set up a pairwise key with their neighborhood with a negligible overhead.
- Node Connectivity (Existence of a shared secret between nodes regardless of their location). This is indispensable in scenarios where nodes are mobile, as they will need to open a secure channel with any node in their vicinity.

– Scalability (Support for big networks). In applications that require of a large number of nodes, KMS should be scalable.
– Extensibility (Capability of adding new nodes). This property is important for those scenarios that have dangerous external conditions, or that have to provide certain services for long periods of time.
– Energy (Optimization of the energy usage). There are some scenarios (short-lived networks) where saving energy is not crucial.

4.3 Network Type and Context

One remarkable effect of the influence of the applications is the configuration of the network architecture and its context. In fact, the "Wireless Sensor Networks" concept can be seen as a generic term that encompasses many different types of sensing architectures. The most "classic" WSN architecture is known as terrestrial sensor networks, where sensor nodes are distributed over a reasonably-sized geographical area, and use the air as a transmission medium. Examples of applications that use these terrestrial sensor networks are healthcare monitoring and precision agriculture systems. However, there are other types of sensor networks, which are especially suitable for different classes of applications. Every one of these networks has a particular name and some inherent features that differentiate them from other types of WSN.

There are some factors that can determine the existence of new types of sensor networks. One of these factors is the physical location of the network. Sensor nodes can be deployed underwater (underwater sensor networks or UWSN) or underground (underground sensor networks or WUSN), although there are also very small networks that are located on, near, or within a human body (body sensor networks or BSN). Another factor is the content of the network: while most sensor networks provide only physical information of the environment, other networks manage more complex information like video and audio streams (multimedia sensor networks or MWSN). Finally, there are some WSN that have either an unorthodox structure or specific goals: some networks do not require the permanent existence of a base station (unattended sensor networks or USN), while other networks collaborate in a heterogeneous environment similar to an Ad-hoc network (embedded peer to peer systems or EP2P).

One of the most important WSN paradigms whose features are completely influenced by the location of the nodes is known as underwater sensor networks, or UWSN. In these networks, sensor nodes are deployed below the ocean surface. These sensors can be used to measure seismic activity in order to provide tsunami warnings (disaster prevention), to detect the location of underwater oilfields (undersea explorations), and to monitor sea-based structures such as oil platforms and undersea cables (structural monitoring), amongst other things [31]. It would seem that the only difference between underwater and terrestrial sensor networks is the water that surrounds the nodes. However, this particular situation imposes additional constraints to the network [32]. For example, underwater sensor nodes must use an acoustic communication channel in order to exchange information wirelessly. This channel has a limited capacity, and the energy consumption and

the propagation delay of the channel are very high. Sensor nodes are also especially prone to failure, due to specific underwater threats such as fishing trawlers, underwater life, failure of waterproofing, or mere corrosion. Moreover, underwater nodes are equipped with a limited battery that cannot be recharged due to their location.

There are other special features of UWSN that must be taken into account: cost, coverage, hardware capabilities, and mobility. Unlike terrestrial sensor networks, the deployment of underwater sensor nodes is more costly, due to the complex components of the nodes (e.g. water-proof housing, acoustic modems) and the equipment that must be used to deploy the nodes (e.g. ships). As the deployment of underwater sensor nodes is usually sparser, and because sensor nodes can move due to anchor drift or other external effects, the coverage of the network is reduced. However, the capabilities of underwater sensor nodes are usually greater: they have more memory to perform data caching in case of intermittent connections, and they also have a battery with a higher capacity due to the cost of using the communication channel and the complexity of replacing a drained battery. Finally, some UWSN can have mobile underwater robots that either support an existing architecture of fixed nodes or collaborate on their own to monitor chemical leaks and other biological phenomena.

A paradigm that is structurally similar to underwater sensor networks is wireless underground sensor networks (WUSN) [33]. In these types of networks, sensor nodes are buried or deployed in a cave or mine, monitoring the state of the soil or the air quality of an underground environment. The major design challenges of this paradigm are the communication channel, topology planning, power conservation, and environmental threats. As electromagnetic (EM) waves encounter a high attenuation in soil, it is necessary to optimize factors such as the packet size [34] and the location of the nodes in order to achieve reasonable connectivity. Besides, some sensor nodes may not be located close to the surface, thus battery replacement may not be a viable option. Moreover, the node can be affected by various threats, such as wildlife and environmental extremes. Note that WUSN do not have some of the constraints that affect its underwater counterpart: network deployment is easier and cheaper, sensor nodes are usually not mobile, and there exist scavenging opportunities for WUSN devices (such as seismic vibration and thermal gradients).

While most UWSN and WUSN consider that their nodes are distributed over a reasonably-sized geographical area, there is one specific paradigm where the network is located within a very small area: body sensor networks, or BSN [35]. The reason is simple: In these BSN nodes are located on, near, or within a human body. Such nodes monitor the physical state of its wearer, offering real-time measurements that can be integrated in complex healthcare scenarios such as telemedicine. These networks are simpler than other sensor networks, as there are a limited number of sensor nodes surrounding the patient. Besides, the architecture of the network is inherently hierarchical, where all data is, in most cases, directly retrieved and managed by a central device known as body area aggregator. Nodes are only affected by a limited number of physical threats, as

they are worn by a human user. Moreover, these nodes and their batteries (excluding nodes located within the body of the patient) can be easily replaced in case of hardware problems or energy depletion.

Nevertheless, BSN have some particular challenges that must be considered. Different devices (e.g. blood pressure and electrocardiography (ECG) sensors) have different energy, QoS and bandwidth requirements, thus it is necessary to prioritize certain traffic according to the state of the network and the patient. However, communication is hindered by factors such as body shadowing (body absorption of RF energy) and movement. In fact, device and service heterogeneity suggest the coexistence of diverse communication technologies, such as inductive coupling, in-body RF-communication, and Ultra-Wideband [36]. Another factor that must be taken into account is the size of the nodes: as nodes must be small, unobtrusive, and ergonomic, the capacity of the batteries of certain BSN devices will be extremely limited due to size restrictions. Besides, it is not possible to use protocols that take advantage of the redundancy of the network due to the low number of nodes. Finally, the QoS requirements of BSN applications are very high, as the devices are managing physical data from a human being: the failure of one device could threaten life.

Location is not the only important factor that determines the existence of new types of sensor networks: content is another of those factors. An example of content-oriented sensor networks is the multimedia sensor networks paradigm, or MWSN. In these networks [37], the devices are able to retrieve, process, and distribute video and audio streams, still images, and scalar sensor data, through the use of embedded cameras. These specific sensor networks can be used to enhance and complement existing surveillance systems (surveillance), to monitor car traffic and to analyze accident scenes (traffic monitoring), to monitor the behaviour of elderly people, and so on. The main characteristics of this type of WSN are the specific bandwidth and QoS requirements of the applications, as well as the need of including cross-layer optimizations and in-network processing of raw data. Actually, even for low-resolution images, it is suggested that class II nodes are more energy-consuming than class III nodes, mainly due to the time needed to perform operations in the image stream. Besides, it is necessary to analyze the tradeoffs between physical layer protocols that provide high data rates (e.g. IEEE 802.11) and protocols that do not have a high bandwidth but are lightweight in terms of energy consumption (e.g. IEEE 802.15.4). There are many other specific challenges in this kind of networks, such as achieving a balance between reliability and congestion control, describe optional routing metrics, and inter-media synchronization.

As for the WSN that have an unorthodox structure or specific goals, we can mention the unattended sensor networks (USN) and embedded peer-to-peer systems (EP2P). The main characteristic of USN is the location of the base station [38]. Base stations are not present at all times in the network, and only appear near the network whenever there is data to retrieve. As a result, sensor nodes must store the physical information retrieved from the environment within themselves or in collector nodes. These nodes should have enough storage to store

this temporal physical information. In fact, data should be moved or replicated in order to assure its survivability against possible attackers. Moreover, it is not possible to perform any kind of self-healing or self-management protocol that depends on the participation of the base station, as the base station is not always present. Since nodes have to proactively collaborate with each other in order to store the data and maintain the integrity of the network, it is necessary to support a communication model where nodes can open a channel with any other node in the network.

Finally, in the EP2P paradigm [39], sensor nodes and other devices collaborate in the processing and management of information in a P2P fashion. This paradigm has some resemblance with the unattended sensor networks paradigm: all nodes need to collaborate in order to perform a particular service. However, there are some crucial differences between these paradigms. For instance, a base station can be constantly available, and PDA-like devices can act as both temporal base stations and data retrieval devices due to the network heterogeneity. Besides, the network is more dynamic: its elements can enter the system and exit in an independent way. As a result, connections and disconnections may happen in an unpredictable and frequent manner, causing frequent reorganizations. From all the paradigms that are described in this section, this paradigm is the one that is more similar to an Ad Hoc network.

Security Analysis. All the inherent features of the previously mentioned types of sensor networks not only have influence on the importance of the security requirements and the characteristics of the security mechanisms, but also limits the number of security mechanisms that could be used to protect the network.

Regarding the security requirements, their importance can be strengthened (i.e. the property must be enforced no matter how) by the features of the network. For example, the self-organization property is crucial for the survivability of unattended sensor networks: the network is truly decentralized, and if there is a problem within it, there is neither base station that can be notified nor users that can provide feedback about the incident. Also, availability and freshness are essential for body sensor networks, because there are human lives at risk if the physiological data is not received on time. Other examples include forward and backward secrecy in embedded P2P systems (due to the dynamic nature of the network) and auditing for underwater sensor networks (as it is very difficult to physically reach the nodes after their deployment). Note that the characteristics of an application also have an influence over the security requirements (e.g. authorization is crucial whenever various users try to directly access the contents of the nodes). However, the influence that exerts the type of sensor network (UWSN, BSN, ...) remains constant, regardless of the nature of the application.

The features of the network will also help to dictate what security algorithms and protocols are more suitable for a certain context. For example, protocols with poor scalability are surprisingly suitable for body sensor networks, because the network size is small and most, if not all interactions with the external world are performed through a centralized device. Also, it is necessary to use mobility-aware protocols in embedded P2P systems and in certain types

of underwater sensor networks due to their dynamic nature. There are more examples: unattended sensor networks will greatly benefit from protocols that can test whether a certain node has been subverted or not (i.e. code attestation protocols), and both underwater and multimedia sensor networks can be able to use computationally-intensive primitives if needed due to the extra power of their sensor nodes.

Note that the sensor network type also limits the kind of security mechanisms that can be used to protect the network. One of the most clear examples is underwater sensor networks, whose security protocols must send as few packets as possible due to the high cost of sending and receiving messages through the acoustic modem. Another clear example is the unattended sensor network, which is not able to incorporate any security mechanism that depend on the existence of a base station. Multimedia sensor networks also have to take care with the overhead imposed by the cryptographic primitives, as it needs to reduce the latency while sending video and audio streams. Finally, body sensor networks need to severely limit the use of energy-consuming primitives and protocols due to the size of its batteries, and embedded P2P systems cannot depend on key management systems that do not provide node connectivity.

5 Overview of Existing Security Mechanisms

5.1 Security Primitives

Most security protocols and mechanisms need of cryptographic primitives in order to integrate the security properties into their operations. These cryptographic primitives are Symmetric Key Cryptography (SKC), Public Key Cryptography (PKC), and Hash functions [40]. Symmetric Key Cryptography (SKC) can provide confidentiality and integrity to the communication channel, and require that both the origin and destination share the same security credential (i.e. secret key), which is utilized for both encryption and decryption. As a result, any third-party that does not have such secret key cannot access the information exchange. 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. As for (cryptographic) hash functions, they are used to create "digital fingerprints" of data. This property can be used to build other cryptographic primitives like the Message Authentication Code (MAC), which provides authenticity and integrity in the messages. These primitives alone are not enough to protect a system, since they just provide the confidentiality, integrity, authentication, and non-repudiation properties. Nevertheless, without these primitives, it would be nearly impossible to create secure and functional protocols.

The development of optimal implementations of security primitives for sensor nodes is a very advanced research field, with solutions that can be easily used in new sensor deployments. In the area of Symmetric Key Cryptography, There are

two types of primitives: Block Ciphers and Stream Ciphers. Block Ciphers are more flexible and powerful, while Stream Ciphers are simpler and faster. One of the most important block ciphers is the Advanced Encryption Standard or AES, which is the encryption standard used by all U.S. government organizations for the protection of sensitive information. While this encryption primitive is not one of the fastest primitives, it is usable in sensor nodes: one of the most optimized software implementation of AES-128 achieves an encryption speed of 286.35 Kbps, a RAM requirement of 260 bytes, and a code size of 5160 bytes running on a 8 Mhz Texas Instruments' MSP430 microcontroller with no operative system [41]. There are even other block ciphers that, when implemented on software, offer an adequate balance between resource consumption and security. For example, the Skipjack cipher is slightly less secure than AES-128 due to its key size (80 bits), but some implementations have achieved a reasonably low encryption overhead per byte (25 μs) and a low memory overhead (code size of 2600 bytes).

Regarding stream ciphers, one of the most known ciphers is RC4, which is very simple and has an impressive speed. Although it is possible to implement it in a sensor node with just 428 bytes of code size [42], its inherent weaknesses (which are mostly concentrated on the initialization phase [43]) make advisable the use of other stream ciphers in new applications. Precisely, the eSTREAM project (organized by the EU ECRYPT network [28]) aimed to identify new stream ciphers that could be used even in constrained devices. Some of the ciphers of the resulting portfolio provide good results [44] in sensor nodes: the Salsa 20/12 algorithm requires 1412 bytes of code size in AVR platforms and it provides a throughput of 43700 bytes per second, and the Sosemanuk algorithm requires more memory (9092 bytes of code size in AVR platforms) but provides a higher throughput (67660 bytes per second)[3].

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, TinyECC [45], 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 [46].

In fact, the improvements on the implementations of ECC primitives have allowed the existence of more complex PKC primitives in sensor nodes, such as identity-based cryptography (IBC). In IBC systems, only the identity of the

[3] Note that, in the reference paper [44], the throughput of AES-128 is 43671 bytes per second.

sensors must be exchanged, and as a result there is no need to send either public keys or certificates. This saves energy as there is less data to be sent through the communication channel, although IBC is also very costly in terms of memory and CPU usage. One of the most optimal implementation of pairings executes the η_T(P,Q) pairing on 1.71 seconds, requiring 4.17 KB of RAM and 23.66 KB of code size running on a 8 Mhz Texas Instruments' MSP430 microcontroller [47]. While it would seem that this primitive is not useful in sensor nodes, there may be certain contexts where it could be useful, such as underwater sensor networks (cf. Section 4.3).

As for hash functions, some standards like SHA-1 can be easily included in sensor nodes: an unoptimized implementation needs of 122 μs for digesting one byte [48]. Note that, as practical collision attacks can be found against SHA-1 [49], NIST is currently working on the selection of a new hash standard [50]. The work on this new standard is focused on PC-like platforms, although performance on embedded systems will not be overlooked. As a result, it is possible that new hash functions applicable to sensor nodes will appear soon. Nevertheless, it is not necessary to use hash functions to assure the integrity of a message if special modes of operation (such as CMAC) are used, although they require of specific block ciphers that could implement that functionality.

5.2 Key Management and Secure Channels

All devices that want to open a secure channel with other nodes must share some security credentials, i.e. secret keys. Key management systems (KMS) aim to solve the problem of creating, distributing, and maintaining those secret keys. The design of a KMS for sensor networks is not a trivial task, though: it is not advisable to rely on centralized entities due to the distributed and self-configurable nature of the network. Also, the existing constraints of sensor nodes (memory, computational capabilities) may discourage the use of resource-intensive algorithms for most scenarios. Finally, there are other factors, such as the potential size of the network, the connectivity of its nodes, the energy spent in the key setup processes, etc, that influence over the design of a KMS as well.

Due to their importance, the Key Management Systems for Wireless Sensor Networks have received increasing attention on the scientific literature, spanning many different types of protocols [25]. In fact, since one of the most important link-layer standards in sensor networks, IEEE 802.15.4, does not specify how secret keys should be exchanged, it is essential to utilize one of these protocols. These protocols can be classified into four major frameworks. Although the major purpose of all these frameworks is to bootstrap the secret keys that are needed by the link layer, their underlying mechanisms and design goals are different.

- *"Key Pool" Framework.* This is one of the first and most important KMS frameworks. The basic scheme behind this framework is quite simple [51]: the network designer creates a "key pool", a large set of precalculated secret keys, and before the network deployment every node in the network is assigned with a unique "key chain", i.e. a small subset of keys from the "key

pool" (Key pre-distribution phase). After the deployment, the nodes can interchange the identification numbers of the keys from their "key chains", trying to find a common shared secret key (shared-key discovery phase). If two nodes do not share any key, they will try to find a "key path" between them in order to negotiate a pairwise key (path-key establishment phase). The major design goal of the protocols that belong to this framework is to assure a limited secure connectivity between nodes, regardless of the size of the network.

- *Mathematical Framework.* Certain KMS protocols use mathematical concepts (Linear Algebra, Combinatorics, and Algebraic Geometry) for calculating the pairwise keys of the nodes. The foundation of the Linear Algebra schemes is the Blom scheme [52]. In this scheme, every node i can calculate the pairwise key it shares with another node j by solving $A(i) \cdot G(j)$, whereas G is a public Vandermonde matrix and A is calculated using a symmetric random secret matrix D. On the field of Combinatorics, the Generalized Quadrangle and Symmetric Design models [53] are the most important. Using Generalized Quadrangles $GQ(s,t)$ or Finite Projective Planes $FPP(q)$, a network designer can construct a "key chain" of size $s+1$ or $q+1$, respectively. Finally, on the field of Algebraic Geometry, the basic primitive is the Bivariate Polynomial [54]. By using a bivariate polynomial f, every node A in the network is able to obtain a pairwise key with another node y by solving $f(A,y)$. All these protocols allow the creation of pairwise keys between nodes without major communication overhead. On the other hand, these designs are often difficult to apply, and they are not very scalable.
- *Negotiation Framework.* All protocols that generate their keys through mutual agreement, negotiating keys with their closer neighbours just after the deployment of the network, can be considered part of this framework. They are usually applied under the assumption that there is little or no treat against the integrity of the network in its first moments of life [55]. Nevertheless, it is possible to use other mechanisms and protocols (such as the Guy Fawkes protocol [56]) in order to assure the authenticity of the peers in any step of the network deployment. Other protocols that can be included inside this framework are those protocols that organize the network into dynamic or static clusters [57].
- *Public Key Framework.* Most of the previous frameworks rely only on Symmetric Key Cryptography. However, Public Key Cryptography can also be used to securely bootstrap the pairwise key of two nodes over a public communication channel. In these protocols, two nodes just need to interchange their public keys and some information (through protocols such as ECDH and ECMQV) to effectively create their pairwise secret keys. While constrained sensor nodes can be able to use PKC through Elliptic Curve Cryptography, the amount of memory required for the implementation and the time and energy needed to complete the negotiation is, in most cases, substantially higher than other KMS frameworks. However, PKC-based KMS usually have better properties than the systems of other frameworks.

As for the actual state of the art, every one of these frameworks contains many different protocols, and these protocols can implement specific optimizations (e.g. use of deployment knowledge, optimize the message exchange, tweak the behaviour of the protocols, combine protocols from different frameworks) in order to improve their features and be more useful for certain contexts. In fact, there exists certain methods that are able to select the most adequate KMS for a certain context by using the application requirements as an input [27]. By using these tools, it is possible to conclude that the actual state of the art for KMS in sensor networks is good enough for protecting certain applications. However, there are some issues that remain to be solved, such as the creation of protocols with better resilience (resistance against stolen credentials) and extensibility (capability of adding new nodes) properties.

One interesting detail is that, for most applications, simple KMS protocols such the basic polynomial-based and Blom schemes provide most of the properties needed by the applications. The reason is simple: most real-world applications have a number of nodes ranging from 50 to 1000. And for this size, simpler protocols are good enough. Another interesting point is the use of PKC. For every single situation PKC seems to be the ideal protocol, and in fact PKC-based protocols such as EDH and ECMQV provide good properties like excellent resilience and extensibility. Nevertheless, there might be situations where simpler KMS protocols can provide the properties needed by the applications.

5.3 Network core protocols

When speaking about "core" protocols, we refer to those network protocols that a sensor network needs in order to function properly. These protocols are: routing (transmitting a packet from one sensor node to another sensor node), data aggregation (briefing many sensor readings into one single piece of data), and time synchronization (synchronizing the clocks of the network). The behaviour and the properties of these protocols are highly dependent on the characteristics of the sensor network application where they are running, because they must be adapted to the requirements of the scenario. As a result, there are many core protocols from where an application designer can choose the most optimal for his/her scenario.

One of the factors that should be considered by every one of these protocols is security. Due to the vulnerable nature of sensor networks, the core protocols should be able to withstand attacks from malicious adversaries, either external or internal. However, it is very difficult to define security mechanisms that could be seamlessly adapted to every type of protocol. They must be prepared to analyze their internal processes in order to detect failure or misbehaviour, and to react to these events in order to maintain the functionality of the network.

Routing is one of the most important protocols of sensor networks. As sensor networks have an inherent distributed nature, the nodes must be able to forward the information to those devices that need it. Many, if not most sensor network protocols depend on the availability of a routing infrastructure. Its importance

makes it a potential target for attackers: most of the attacks presented in section 3.1 can be crafted to hinder the routing processes. As there are many types of routing strategies (e.g. flat routing, data-centric routing, hierarchical routing, location-based routing), it is necessary to find suitable security approaches for every strategy that take into account their specific properties. Nevertheless, it can be possible to define certain generic countermeasures that can provide some security properties to all strategies.

Some of the existing countermeasures against routing protocol attacks are analyzed by both existing surveys [58] and by drafts of routing standards for sensor networks like ROLL (Routing over Low power and Lossy Networks) [59]. Attackers trying to manipulate the routing discovery mechanisms (using HELLO flood and acknowledgement spoofing attacks) transmit their packets with a higher transmission power. Therefore, nodes can defend themselves by verifying that the link is truly bidirectional, using extra protection mechanisms such as one-time keys if needed. An adversary can also try to overload a sensor node with irrelevant messages, as the lifetime of sensor networks is highly tied to the number of exchanged messages. Nodes can lessen the effects of this attack by introducing traffic quotas if the network seems overloaded.

There are also some countermeasures that defend against more complex attacks, such as selective forwarding, sinkholes, and wormholes. Selective forwarding attacks can be mitigated by using multipath routing, either by sending replicated packets through different paths or by choosing a different path for every packet that is being sent. Sinkholes and wormholes can be detected by estimating the distance between neighbours, as neighbours joined through a wormhole will be physically distant from each other. Other solutions for the wormhole problem include using the base station to identify distortions in the distribution of the number of neighbours, use packet leashes (e.g. timestamps included within the packet) to detect if the sender is further away than the nodes' communication range, or performing various distance-bounding protocols. Note that all these protection techniques could be further improved, although routing protocols may use other protection mechanisms included within the node to detect and react against possible problems (e.g. using an intrusion detection system to sense a possible selective forwarder). Moreover, it can be possible to take advantage in the specific features of the context to implement extra protection mechanisms. For example, if the nodes were aware of their physical location, the detection of attacks such as sinkholes and wormholes could be done in an easier way.

There is a specific case of routing that must be considered separately due to its importance: authenticated broadcast. Not only it is important to avoid a malicious flooding that would drain the energy of battery-dependant nodes, but also it is essential to assure that the received data comes from its intended source and is not manipulated. In nodes with enough resources, it is possible to use public key cryptography to apply a digital signature scheme to the broadcasted message. Another solution can use one-way hash chains (a collections of values where c_{n-1} is a one-way function of the next value c_n) in conjunction with

symmetric-based message authentication codes (MAC) to "sign" a broadcasted message. Knowing cn in advance, nodes cannot "sign" messages that use c_{n-1} because they do now know it, but after receiving c_{n-1}, they can verify its validity by using c_n (as $H(c_{i+1}) = c_i$). There are many mechanisms that take advantage of this particular property of hash chains, such as μTesla and its variants, and the protocols that use one-way signatures.

Aggregation may not be mandatory for all sensor networks, but it provides a very useful service: the combination of data coming from nodes deployed at the same area. This is very important for most sensor network contexts, as it is necessary to optimize the usage of the communication channel. The aggregation process is closely related to the routing service, since aggregation capabilities are usually embedded inside routing protocols. There are also different strategies for aggregating data: routing-driven algorithms (opportunistic aggregation), coding-driven algorithms (compression at the source), and fusion-driven algorithms (full-fledged aggregation along the routing path). Both this heterogeneity and the dependence on the routing protocols add an extra complexity to the development of protection mechanisms, although some of the mechanisms that were used to protect the routing protocols can also be used to protect the aggregation messages. Nevertheless, there are specific aggregation attacks, targeting the data sources and the aggregation nodes, which must be lessened somehow [60][61].

One simple countermeasure to limit the effect of a compromised node trying to inject false data is to improve the resilience of the aggregation functions, as some functions (e.g. minimum, maximum, sum, average) are more insecure than others. The redundancy of some networks can be also used as a tool to detect fake or faulty readings that are too deviated from the average of the neighbourhood (e.g. due to the physical properties of heat, a node cannot sense that the environment is extremely hot when its neighbours located in the same area feel much colder values). While detecting problematic data is important, the main challenge of secure data aggregation is to detect a misbehaving aggregator. One solution is to use interactive proofs, where the user can verify that the aggregated data provided by the aggregator is a good approximation of the true value. Another solution uses witnesses, that is, redundant sensor nodes that aggregate the data and create a proof (e.g. a MAC) of the validity of the aggregation value.

If the network aims to provide end-to-end confidentiality, that is, to make the data known only by the original server and the receiver, it is necessary to use specific mechanisms such as homomorphic encryption. This primitive aims to perform specific algebraic operations in the ciphertext without decrypting it. Some aggregation protocols implement this feature by using elliptic curves for the sake of efficiency, as homomorphic encryption is expensive for sensor nodes in terms of computational resources. However, these protocols can only work with specific query-based aggregation functions such as sum and average. End-to-end aggregation, and aggregation in general, is an open field with research issues such as aggregation in dynamic environments, detection of compromised

nodes injecting fake data, design of efficient homomorphic cryptography able to work with all types of functions, development of multilayer hierarchical data aggregation protocols, and so on.

Time synchronization does not provide any direct functionality to the users, but it is extremely important for the consistency of the network. It is essential not only to know when certain data was retrieved, but also to know the exact sequence in which events such as alarms occurred. Therefore, all nodes must have a similar time stored in their internal clocks regardless of hardware problems such as offset, skew and drift problems. Both sender-receiver and receiver-receiver time synchronization protocols (cf. Section 3.1) need of two nodes exchanging timing information. As a result, it is possible to use a secure communication channel in order to avoid attacks (e.g. message replay) perpetrated by outsider attackers. However, insider attackers have more chances to break these protocols by providing a fake time [62].

A simple protection against malicious insiders is to use delay thresholds. The typical drift rate of the nodes is not very high (clocks of sensor nodes usually accumulate several seconds of drift error per day), thus it is possible to specify the maximum amount of time offsets they will tolerate. Another method uses statistic techniques such as GESD (generalized extreme studentized deviate) to identify messages coming from compromised nodes. Note that these techniques only limit the effect of attacks, although some fine-tuned algorithms can decrease the threshold to very low values. A logical evolution is to have more than one partner for performing the clock synchronization. As for the time aggregation strategy, some protocols use the median (instead of the mean) to calculate the clock drift, sacrificing the ability to improve precision through multiple independent observations in order to have a high protection against malicious nodes.

While these protection mechanisms are focused on assuring the security of the information exchange between two nodes, it is also necessary to protect protocols that broadcast synchronization information in order to adjust the clocks of all the nodes in the network. This can be possible by using mechanisms such as secure synchronization trees [63]. Besides, other mechanisms even provide support for group synchronization, and are able to deal with inconsistent behaviour from a subset of nodes in the group [64]. The precision of the most advanced protection mechanisms is quite good, as they restrict the maximum impact of the attacker on the synchronization precision to under a few microseconds. Note, however, that most of these protocols have been tested in terrestrial sensor networks, with no experimental results on radio unfriendly environments. Still, some applications simply require the clocks of their nodes to be loosely synchronized.

5.4 Self-Healing and Self-Management Protocols

All protocols, regardless of their design and functionality, should be aware of their environment: which nodes are on a certain neighbourhood, whether a node seems to be alive or dead, the actual state of the communication channel, and so on. Therefore, it is essential to have certain self-awareness mechanisms that

provide this information (e.g. whether a certain node has disappeared from a neighbourhood) to the protocols of the sensor node. This information is also vital for allowing the existence of self-healing mechanisms, as 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. By knowing the context where it is located and the situation of its surroundings and its neighbourhood, a sensor node can be able to tweak its functionality to be more robust and work in a more optimal way.

Besides, those self-healing mechanisms can facilitate the creation of security services such as intrusion detection systems and trust management systems. By using intrusion detection systems, it is possible to know if a certain node is suspicious of misbehaviour or malicious. This knowledge can be used by any of the protocols of the network to ignore any interactions with these malicious entities. As for trust management systems, they can indicate if a certain node can be trusted for a particular task (e.g. route a packet to the base station), improving the overall intelligence of the protocols of the network.

An intrusion detection system (IDS) for WSN must have certain specific capabilities in order to be useful and functional [65][66]: audit data management (works with very application-specific partial audit data), simplicity (IDS should not take much resources from the nodes), 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). There are many challenges that must be considered in order to create a IDS that fulfils these features, such as the development of efficient detection mechanisms, the definition of a IDS architecture, the location of the detection entities (i.e. IDS clients), and so on.

There already exist various lightweight detection mechanisms that can detect anomalous events in the network, like jamming, malfunctioning sensors, node subversion, and time-related attacks (e.g. delay). Besides, although there is still room for improvement, the research community has also developed other detection mechanisms that analyze the state of sensor-specific protocols and derive the existence of an attack. For example, sinkhole attacks can be detected by using anomaly detection techniques, and some specific aggregation attacks can be detected with more accuracy if integrated with other IDS mechanisms. All these alerts can be processed by the base station, which can derive the identity of the nodes performing the attacks or distinguish errors and network failures from attacks.

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.

Regarding trust, it aids the members of a WSN (trustors) to deal with uncertainty about the future actions of other participants (trustees): nodes with high trust values are expected to provide the services they have been asked for. Most of the existing work on trust management systems for sensor nodes has focused on solving specific problems, that is, on developing trust solutions that only solve one problem in particular [68][69]. Reputation ('What is generally said or believed about an entity') is often used as one of the inputs for calculating the trust values. Most systems represent it through a Bayesian formulation, more specifically, a beta reputation system (a,b), where a denotes good behaviour and b denotes bad behaviour. A common optimization in trust management systems for sensor networks is to make use of clusters to calculate and store the trust of the network entities. Other systems consider the use of extra parameters such as risk and entropy.

One important point that must be considered when developing a trust management system for WSN is how it should be influenced by the inherent features of sensor networks. In fact, it is possible to link those features with the state of the art in order to obtain a set of "best practices" that any trust management system for WSN should take into account. Note that existing trust protocols do not consider all these principles at the same time, although they fulfil some of them. These "best practices" are described as follows:

- Considering Trust and Reputation. A trust management system for WSN should have the following two elements: A reputation manager (to store the behaviour of nodes) and a trust manager (to calculate the trust values using reputation as one of its inputs). By not calculating the trust directly from the behaviour of a node, it is possible to better handle aspects such as the evolution of the node, aging, etc.
- Trust and the Base Station. The base stations should be able to participate on the trust management process. A base station can use the information produced by the sensor network to observe and analyze the behaviour of its nodes, storing their reputation and making global trust decisions.
- Information Gathering. The events that occur during the lifetime of a WSN can be used as inputs for a trust management system, as they model the behaviour of a certain sensor node. Besides, the reputation information about other nodes should be distributed, as neglecting the use of such second-hand information may result on decisions that are not fully consistent with the actual state of the network.

- Initial Values. At the beginning of the lifetime of the WSN, all its nodes must have a good reputation and be equally trusted. Nodes are usually programmed in a controlled environment, and at the very beginning any malicious adversary had neither the time nor the chance to influence or subvert a node.
- Granularity. A node needs to maintain separate opinions about the existing actions of their peers, thus it needs a different set of reputation values. Besides, different trust values should also exist: A specific trust value (e.g. routing) cannot be used in most cases to deduce what the peer could do in a different task (e.g. sensing).
- Updating and Aging. The internal state of the trust management system must be updated with information received during the lifetime of the network. Regarding aging, in the updating process trust entities should use aging mechanisms as a way to incorporate new information in a smooth way. Besides, different events should not have the same impact on the reputation of a node, and the evolution of the reputation and trust values should not be ignored.
- Risk and Importance. Two factors that should influence over the calculation of the trust values are the risk of the interaction between the trustor and the trustee, and the importance of the reputation value and that specific interaction. Risk and importance also should have influence when selecting a threshold.

5.5 Privacy and Anonymity in Sensor Networks

As mentioned in the overview of the security requirements, there are certain scenarios where the privacy of the elements of the network needs to be taken into account. There are basically three types of threats against privacy [67]: content, location, and identity. If an adversary can determine the meaning of a communication exchange because of the existence of a message and the context and timing of the situation, there is a content privacy threat. If the adversary is able to infer the physical location of a communication entity or to approximate the relative distance to that entity, there is a location privacy threat. And if an adversary is able to deduce the identities of the nodes involved in a communication, there is an identity privacy threat. It should be noted that these privacy threats are closely linked with the anonymity property. However, privacy techniques may also aim for unobservability (messages cannot be distinguished from random noise), whereas anonymity do not try to hide the existence of a certain event but just tries to make it undistinguishable from events of the same kind.

Existing privacy-preserving solutions for WSN can be classified in data privacy protection, which aims to protect the privacy of the data content, and context privacy protection, which aims to protect contextual information such as location and timing [70]. For data protection, there are mechanisms that try to preserve the privacy of data from malicious adversaries during the aggregation process. One idea is to add specially crafted noise to the raw data sensed by the WSN, thus the aggregator will not be able to know the individual raw data

items but can obtain precise aggregated values. There are other approaches that protect aggregation privacy, such as slicing original data into pieces in order to recombine the randomly, and forcing the aggregator to obtain only an accurate estimate of the histogram of the data distribution instead of raw data. Other data protection mechanisms try to protect the privacy of queries, as it is possible to infer some information from the existence of these queries (e.g. a patient staying at home being monitored by a BSN). One possible solution is to fuzzy the target region of the query according to pre-defined transformation functions.

For context privacy protection, existing techniques focus on protecting the location of nodes and the timing of the generation of the data. On the field of location privacy, it is possible to attain perfect unobservability by constantly sending bogus data, thus an attacker will not be able to discover the existence of traffic coming from a real target (i.e. the location of pandas or elephants). As this technique is quite energy-consuming, it is necessary to create more optimal mechanisms, although there is some preliminary work on this matter based on random routing techniques. Note that these mechanisms protect the location of a node, but not the location of a base station. For protecting the base station, it is necessary to thwart local attackers by hiding the parent-child relationship between nodes, while global attackers may require of more complex mechanisms. The techniques that aim to protect temporal privacy are actually simpler, based on random delays. Certainly, there is a need to make a tradeoff between the protection of timing privacy, the efficiency of buffer space, and the maximum delay allowed.

5.6 Software-based Protection and Testing

By reviewing the different attacks presented in section 3.1, it can be deduced that one of the most dangerous class of attacks that an adversary can perform is node compromise. Whenever a node is compromised there is no immediate effect in the network, but starting from that point an external attacker becomes an internal attacker, and the amount and scope of the attacks it can be able to perform is simply devastating. Therefore, one of the priorities for protecting a sensor network should be to incorporate hardware and software mechanisms that either delay or avoid these compromise attacks. There are some hardware strategies that may protect the nodes against attackers with limited resources, such as deactivating the JTAG debugging interface and randomizing the password to access the bootstrap loader in certain microprocessors [9]. Unfortunately, most sensor nodes do not have any kind of tamper resistant package, and any attacker with enough resources will surely success on retrieving information directly from the nodes.

Fortunately, there exist some software-based techniques that can check if a node has been subverted or replicated by an adversary. Remote Attestation is one of these tools [71], where any node (verifier) can send a challenge to another node (target). As nodes are "cells" that are usually loaded with the same code, the challenge usually consist on a "pseudorandom memory traversal", where the target needs to randomly access and store some positions of its own memory. If

the target has the same code as the verifier, it should provide a correct answer to the challenge within acceptable time bounds. There are some enhancements that improve the reliability of this basic scheme: the challenge routine can be obfuscated or use a keyed hash, and some important state information such as the program counter can be included as part of the verification process.

Another remote identification method is radio fingerprinting. This technique make use of the Received Signal Strength Indicator (RSSI) values and the Link Quality Indicator (LQI) values of radio signals. These values can be used for estimating the distance between a pair of nodes, thus they can be used to detect certain types of attacks (e.g. HELLO floods). However, it can be possible to extract an unique fingerprint from these values due to the physical characterization of the signal. As a result, one node can identify the sender of a message analyzing its RSSI and LQI values [72]. The actual state of the art is very promising, as the identification method yields no false negatives and the false positives rate is small (1.05%). It is also possible to improve the signal capturing method by using extra hardware [73]. In this HW solution, sensor nodes can be recognized from large distances (up to 40 m indoors) with a high accuracy (achieving a Equal Error Rate (EER) of 0.24%.)

5.7 Other protocols and mechanisms

Beyond the "core" protocols and the information retrieval mechanisms of sensor networks, there are other services and protocols that can be useful to improve the functionality of the network. For example, it is possible to upload new code to the motes through the wireless channel. This way, there is no need to physically obtain every node in the network in order to reprogram it. It is also possible to include small virtual machines inside the motes, greatly enhancing the ability of the base station to issue orders to the network. Another interesting service is group management: in certain WSN the nodes must group themselves in dynamic clusters that try to solve a particular problem. Not surprisingly, all these services and protocols need to be protected against possible external and internal attacks.

Regarding code dissemination, it is necessary to protect the code fragments that traverse the WSN. Signing all fragments is a valid solution, but it imposes a considerable overhead over the sensor nodes. Therefore, there are many techniques that try to simplify the process (using hash chains, hash trees, and others) while providing protection against attacks such as Denial of Service due to battery exhaustion. As for group management, the basic mechanisms that need to be provided are group key management and cluster head election. Note that there are many challenges to solve in these areas: forward and backward secrecy properties should be provided in the maintenance of group keys and node revocation processes, and the election of a cluster head should not be affected by the presence of malicious adversaries.

Furthermore, there are plenty of security mechanisms for sensor networks that need to be further investigated. For example, random number generation is an important primitive for sensor nodes, as many cryptographic primitives depend on it. The creation of a pseudorandom number generator for WSN that

complies with certain quality metrics is still underdeveloped. Other mechanisms are related to authorization, as the base station may delegate some of its privileges to other devices in order to perform maintenance and management tasks "on-site". At last, other aspects of the security in WSN that are in need of more research are: the relationship between the security and the QoS requirements of the network, methodologies such as attacks trees applied to WSN in order to quantify their risks, development of secure location algorithms, analysis of name and addressing vulnerabilities, creation of secure architectures and middlewares that use cross-layer optimizations, distributed computing, data redundancy and survivability, mechanisms for the protection of the MAC layer, and so on.

6 Existing Standards and their Security Mechanisms

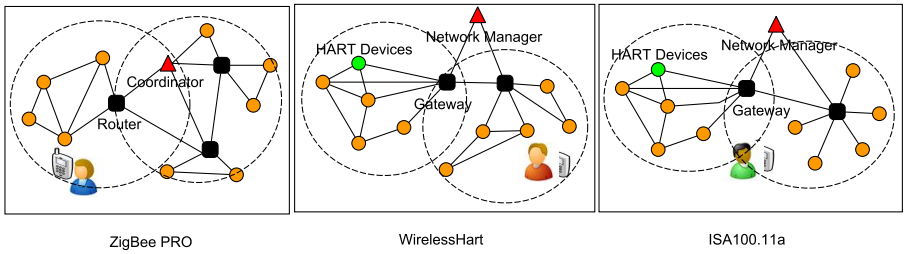

Fig. 3. WSN Standards

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 [75], 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 [74]. After ZigBee, other standards that try to address the needs of industrial and automation systems have been developed. Examples of these standards are WirelessHart [76] and ISA100.11a [77], as shown in Figure 3. The remainder of this section will study these communication standards for WSN, focusing on their network architectures and their security mechanisms.

6.1 IEEE 802.15.4-2006

Most of the wireless communication standards applicable to WSN are based on the IEEE 802.15.4-2006 standard [78], that specifies the physical (PHY) and

Media Access Control layer (MAC) layers of a Wireless Personal Area Network (WPAN). The PHY layer is in charge of activating the transmission radio to send/receive packets through the most suitable radio frequency (RF), protecting the channel from other unauthenticated devices and managing signal functions. In contrast, the MAC layer is in charge of generating beacons (frames for synchronizing the clock of all the networks devices belonging to a same network) and providing association services.

This standard also offers diverse mechanisms for secure deployment, network and device discovery, energy control (through low-duty cycles, i.e., cycles where a device can be in a sleeping state), and RF channel change to control interferences and noise in the communication channels. Such mechanisms work over networks based on a star and peer-to-peer topology with low complexity and transmission rate. Furthermore, it can operate in the following unlicensed frequency bands: 2.4GHz (worldwide use) using up to sixteen channels at 250Kbps, 868-868.8MHz (Europe) using one channel at 20/100/250Kbps, and 902-928MHz (North America) using up to thirty channels at 40/250Kbps.

Noise control is performed by using the Direct Sequence Spread Spectrum (DSSS) method. This method is responsible for modulating the information before its transmission using a lower spectral power density in order to assure an interference reduction in the frequency channels. The standard also defines three types of modulations that preserve the DSSS approach: Binary Phase Shift Keying (BPSK), Offset Quadrature Phase Shift Keying (O-QPSK) and Parallel Sequence Spread Spectrum (PSSS). Regarding interferences, they are controlled through the CSMA-CA (Carrier-Sense Multiple-Access with Collision Avoidance) collision control protocol, which listens to the medium before starting a transmission. If the medium is being used by another network device, then the node must wait a random time again. Another way of controlling the interferences is to use Guarantee Time Slots (GTS) protocol with centralized medium access times.

From a security standpoint, this standard provides HW support for AES-128 (Advanced Encryption Standard with key length of 128 bits) to assure confidentiality in the messages. Integrity is achieved with the use of a Message Integrity Code (MIC) or Message Authentication Code (MAC) with 32/64/128bits. The MIC/MAC is composed of three main fields: frame control, auxiliary security control and data payload. The frame control field is composed of a security control value (it specifies the type of security to use, e.g. AES-CBC-MAC, AES-CTR or AES-CCM), a frame counter value (avoiding message replay attacks) and a key identifier value (additional information related to the key). Also, IEEE 802.15.4-2006 allows network devices to authenticate any message received by using an Access Control List (ACL). This list includes the address of trustworthy neighbours, a key with 128bits length, a last initial vector (IV), a replay counter (making sure the freshness of the messages) and a security policy, as for instance AES-CTR. In case where a device is not included on the list, the message will have to be refused or go through another authentication mechanism.

6.2 ZigBee 2006, 2007 and PRO

The ZigBee standard, created by the ZigBee alliance, is a wireless communication standard for constrained devices. At present, there are three versions of the standard: ZigBee-2006, ZigBee-2007 (it is also known as ZigBee) and ZigBee PRO (Zigbee-2004 is considered as an obsolete release). These three ZigBee versions have certain features in common. For example, they are capable of supporting 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. Generally, the coordinator behaves like a trust center, which is able to manage the deployment, maintenance and control processes on the network. A mesh network simplifies the coexistence with other communication systems, such as Bluetooth or WiFi. Furthermore, this topology provides communication reliability since alternatives paths can be dynamically chosen through the Ad Hoc On Demand Distance Vector Routing (AODV) protocol.

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.

ZigBee 2006, 2007 and PRO are being applied to diverse commercial systems, consumer applications and industrial systems. Table 3 represents a summary about the services and mechanisms provided by these three standards. Particularly, ZigBee-2007 and Zigbee provide self-forming and self-configurable mesh and cluster tree networks with 31,101 nodes. These specifications offers diverse application services and mechanisms, such as device and service discovery, acknowledge service, fragmentation and reassembly of packets, a PAN ID conflict resolution mechanism, a commissioning cluster to configure the devices based on additional information related to the network or the nodes, and a RF channel change service with the frequency agility method.

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 security mode utilized in this case is officially known as "Standard Security" mode and is compatible with residential security of ZigBee-2006. In this mode, it is considered that "all nodes on the network trust each other" and the coordinator is the trust center responsible for managing, distributing and updating the symmetric keys on the whole network. In addition, the standard mode manages two important keys: link key, which is

Feature Set	ZigBee 2006	ZigBee 2007	ZigBee PRO
Mesh networks	✓	✓	✓
Cluster tree networks	✓	✓	
Many-to-one networks			✓
Scalability	✓	✓	✓
Commissioning cluster		✓	✓
Fragmentation and reassembly		✓	✓
Frequency agility		✓	✓
Source routing			✓
Symmetric link			✓
Stochastic addressing			✓
Multicasting			✓
Broadcasting	✓	✓	✓
Security standard mode	✓	✓	✓
Security high mode			✓
Application context	Residential	Residential	Commercial

Table 3. A brief comparative of the ZigBee specifications.

shared between two network devices and used for the confidentiality at the APS layer, and network key, which is shared by all the nodes on the network and is used for the confidentiality at the NWK layer. The network key can be obtained through two ways: either transmitting it without protection from coordinator or encrypting it with the link key, which must be previously preconfigured in the new device. Note that updates of the network key are frequently broadcasted by the coordinator.

In contrast, ZigBee PRO, included in the ZigBee-2007 specification, was designed for large mesh and many-to-one networks with a maximum of 65,540 nodes. At present, this standard shares some services with ZigBee-2007, such as frequency agility, commissioning cluster or fragmentation. However, other and new attractive services are only offered by ZigBee PRO, such as multicasting, symmetric link, source routing or stochastic addressing. The multicasting service allows a network device to transmit packets to many devices at the same time. The symmetric link is a method able to choose a path with better link quality. The source routing allows gateway to return traffic to a source node, embedding the path from the source node to the gateway into packet header. Moreover, this mechanism assures an entry reduction in the routing tables and minimizes the broadcast traffic in the network. The stochastic addressing consists of randomly assigning an address to the new network devices. In case where such address is in conflict with another one in the network, a conflict resolution mechanism is automatically activated along with the IEEE MAC physical addresses.

As regards security in ZigBee PRO, it offers a "High Security" mode with better protection levels than the standard mode. Basically, this introduces a new key, known as master key. Such key is preconfigured in the new devices in order to generate the link key by means of a Symmetric-Key-Key-Exchange

(SKKE) algorithm. At the moment that the link key is generated, the network key is transmitted encrypted with it. The network key is frequently updated in a unicast mode and protected with the link key by the coordinator. This set of security keys assures that any application under ZigBee PRO is robust enough to face threatening situations. Moreover, it is considered a suitable standard for the control of critical and complex systems, as for example an energy generation/distribution plant.

6.3 WirelessHart™

Nowadays, the vast majority of industrial leaders are demanding state of the art technologies and infrastructures so that the control processes can take advantages of new and attractive control services. In fact, their installations are continually changing in order to provide reliability, security, and an improved system functionality. In fact, one of the most demanded technologies is wireless communication, since it provides the same advantages than a wired infrastructure but with a low installation and maintenance cost. However, the integration of multiple wireless technologies at the same critical system supposes to take into consideration several important issues. Predominantly, interoperability among wireless communication systems (Bluetooth (IEEE 802.15.1), RFID, Wireless Sensor Networks (using IEEE 802.15.4), WiMAX (IEEE 802.16), WiFi (IEEE 802.11), etc.), compatibility with existing hardware and software components, security, and reliability in the communications. All of these requirements were recently considered by HART [76] to define a specific protocol as part of the HART Field Communication Protocol Revision 7. This new protocol, known as WirelessHart, has as goal to provide industrial solutions through wireless mesh networks composed of node groups.

The WirelessHart 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 is considered as a trusted node in the whole network, and it could be integrated in the gateway. Generally, this device possesses enough resources to manage the routing tables, the synchronization schedule, the network configuration and the security. In fact, the network manager updates the routing tables and the communication schedule as new nodes join the network. The tables are based on information associated to a routing graph with redundant paths for each node. 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 [79]. Such interface should be able to interpret any type of protocol, as may be: Modbus/TCP [80], DNP3 [81] or IEC-104 [82].

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 a fixed time-slot (i.e., 10 ms). In addition, the concept of superframe is introduced in order to group a sequence of time-slots whose value starts with an absolution slot

number (ASN) equal zero. Such superframes are periodically repeated by time periods. On the other hand, WirelessHart controls the noise and interferences in the communication channel by applying frequency hopping and blacklisting methods. The frequency hopping method allows network devices to change of RF channel. The blacklisting method consists of including on a blacklist those RF channels with interferences or noise.

In terms of enforcing security, it provides protection at both NWK-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 nounce of 13 bytes and the message header without encryption.

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. This last service identifies four priority levels and classified by: commands, measurements, normal messages and alarms (which includes both the event occurred and the alarm).

6.4 ISA100.11a

ISA100.11 release one is an open standard approved by the ISA100 Standards Committee in April 2009 [77]. This standard is focused on providing diverse control services at automation and control systems. In fact, its main goal is to assure interoperability with other communication systems, compatibility with existing hardware and software systems, energy conservation, reliability and security. Moreover, the standard contemplates a set of functions and operations for the monitoring of non-critical and critical systems, among them the industrial and control systems (e.g. a SCADA system). Furthermore, this first version offers specific functions for supervisory control, detection of anomalous situation and alerting in mesh and star networks.

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 routing. On the other hand, 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. The native protocol is composed of specific functions that manage the bandwidth and energy of the network, while the tunneling protocol allows interoperability with other standards such as Modbus, Profibus [83], Fieldbus [84], etc. Regarding 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.

6.5 Discussion

Feature Set	ZigBee PRO	WirelessHART	ISA100.11a
Mesh networks	✓	✓	✓
Many-to-one networks	✓	✓	
Star networks			✓
Scalability	✓	✓	✓
RF channel change	✓	✓	✓
High security	✓	✓	✓
Noise/interference control	✓	✓	✓
Priority management		✓	✓
Energy saving	✓	✓	✓
Interoperability with other systems	✓	✓	✓
Application context	Commercial	Industrial	Industrial

Table 4. A brief comparative among wireless communication standards.

Table 4 presents a brief summary and comparative among the different wireless communication standards. It is possible to observe that the three standards follow common objectives, such as: interoperability with other communication systems, scalability, energy saving, communication reliability, compatibility with existing industrial devices, and security. Precisely, all the standards protect the communication channel against external attackers, and provide some mechanisms to refresh the keys used in the network.

Still, there are some WSN-specific aspects that are not considered by the current standards. This does not mean that the standards have neglected security in their design. Most standards include protection against jamming and Denial of Service attacks through the use of frequency hopping techniques. Also, all protocols provide secure communication channels, assuring the confidentiality, integrity, and authentication of data. As a result, it can be possible to avoid or mitigate the effects of attacks perpetrated by malicious outsiders. However,

it is fairly easy to include a malicious node inside the network by using node compromise attacks (cf. section 3.1). As a result, the protocols used by these standards can be attacked by malicious insiders, effectively hindering the provisioning of services. It should be noted that sensor nodes working on industrial environments may have tamper-resistant packages due to the criticality of the environment.

As insider attacks can affect the functionality of the network, the core protocols defined by the standards should incorporate some lightweight security mechanisms in order to be robust against attacks. Besides, these standards should provide support for self-healing and intrusion detection mechanisms. Note that some standards, like WirelessHART, already provide support for self-healing. However, these mechanisms are oriented to apply corrections in the event of routing faults such as nodes disappearing from the network, not to react against attacks such as blackholes. Another challenge is the use of Public Key Cryptography: all standards use symmetric cryptography to establish the security infrastructure of the network, and the properties of PKC (e.g. network resilience, authenticated broadcasts) could be very useful for an industrial environment. Finally, all these standards have been only tested in terrestrial sensor networks, and may not work correctly or efficiently in other network types such as underwater sensor networks and body sensor networks. Nevertheless, this is comprehensible, as these standards aim to provide a specific service for terrestrial environments.

6.6 Addendum: 6LoWPAN and the Internet

6LoWPAN is an open standard for low power WPANs under IPv6 [85], whose name comes from the working group on the Internet area of IETF (Internet Engineering Task Force). This working group was established in 2004 to address the challenges of enabling wireless IPv6 communication over the IEEE 802.15.4-2006 standard with low-power radio network devices and with limited resources. At present, this standard is considered a suitable element to introduce the concept "Internet of Things" [86].

The 6LoWPAN stack architecture is slightly similar to the OSI model, although its session and presentation layer are not explicitly used. The two lowest layers are based on the IEEE 802.15.4-2006 standard, where the PHY layer transmits packets with a payload of 127 bytes and an offered load of 250Kbps. Likewise, the MAC layer provides diverse services to guarantee single-hop communication links between network devices using the CSMA/CA protocol. On the other hand, the NWK layer is the responsible for providing internetworking capabilities, addressing IPv6, network management with SNMP (Simple Network Management Protocol) and security services. Furthermore, the NWK layer has integrated a special sub-layer for routing, known as Adaptation layer. This sublayer provides a set of services and functions, such as: TCP/IP header compression, fragmentation and reassembly, network device discovery, multicasting and routing. In fact, besides ROLL [59], there are several routing protocols defined, such as LOAD (6LoWPAN Ad hoc Routing Protocol), DYMO Low (Dynamic MANET On-demand) or Hi-Low.

Therefore, 6LoWPAN offers essential services for routing in the personal area network (PAN) space, providing packet routing between the IPv6 domain and the PAN domain, IP adaptation and interoperability, addressing schemes and address management, device and service discovery, and security considering set-up, deployment and maintenance. As for security, 6LoWPAN depends on the security offered by IEEE 802.15.4 with AES-128, and it lacks of strong security mechanisms. However, even if 6LoWPAN were able to provide a stronger foundation for security, there are plenty of previously unidentified security challenges when integrating WSN and the Internet. Those challenges must be taken into account by both sides. Some of the challenges include: communication security, device and user authentication, availability, accountability, WSN-specific optimizations, and robustness against attacks [87]. Note that these challenges are mostly protocol-independent: future IP extensions of protocols like ZigBee should also take these challenges into account.

7 Conclusions

As sensor networks become increasingly used in real-world settings, it is necessary to provide efficient and usable security mechanisms that could protect the network against attacks. While it is possible to deploy a secure sensor network for certain applications, there are some challenges that need to be considered. However, one of those challenges is not related to the technology but to the network designers and users: they must become aware of the existing connections between the context of the application, its security requirements, and the security mechanisms. The purpose of this paper was to raise the awareness on this particular subject, and to show how existing standards should also consider this factor and the specific needs of sensor networks, such as support for self-healing mechanisms that can mitigate the effect of internal attacks.

Acknowledgements

This work has been partially supported by the ARES CONSOLIDER project (CSD2007-00004) and the CRISIS project (TIN2006-09242). The third author was funded by the Ministry of Education and Science of Spain under the "Programa Nacional de Formacion de Profesorado Universitario".

References

1. S. Petersen, and S. Carlsen. *Wireless Sensor Networks: Introduction to Installation and Integration on an Offshore Oil & Gas Platform.* Proceedings of the 19th Australian Conference on Software Engineering (ASWEC 2008), Perth (Australia), Marh 2008.
2. European Organization for Nuclear Research (CERN). *LHC - The Large Hadron Collider.* `http://lhc.web.cern.ch`, Retrieved on June 2009.

3. W. Xu, W. Trappe, Y. Zhang, and T. Wood. *The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks*. Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc 2005), pp. 46-57, Urbana-Champaign (USA), May 2005.
4. D.R. Raymond, R.C. Marchany, M.I. Brownfield, and S.F. Midkiff. *Effects of Denial-of-Sleep Attacks on Wireless Sensor Network MAC Protocols*. IEEE Transactions on Vehicular Technology, vol. 58, no. 1, pp. 367-380, 2009.
5. Y.W. Law, M. Palaniswami, L. Van Hoesel, J. Doumen, P. Hartel, and P. Havinga. *Energy-Efficient Link-Layer Jamming Attacks against Wireless Sensor Network MAC Protocols*. ACM Transactions on Sensor Networks, vol. 5, no. 1, pp. 6:1-6:38, 2009.
6. D.R. Raymond, and S.F. Midkiff. *Denial-of-Service in Wireless Sensor Networks: Attacks and Defenses*. IEEE Pervasive Computing, vol. 7, no. 1, pp. 74-81, 2008.
7. K. Pongaliur, Z. Abraham, A.X. Liu, L. Xiao, and L. Kempel. *Securing Sensor Nodes Against Side Channel Attacks*. Proceedings of the 11th IEEE High Assurance Systems Engineering Symposium (HASE 2008), pp. 353-361, Nanjing (China), December 2008.
8. A. Becher, Z. Benenson, and M. Dornseif. *Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks*. Proceedings of the 3rd International Conference on Security in Pervasive Computing (SPC'06), pp. 104-118, York (United Kingdom), April 2006.
9. T. Goodspeed. *Wireless Sensor Networks as an Asset and a Liability*. Proceedings of the SOURCE Conference, Boston (USA), March 2009.
10. A. Francillon, and C. Castelluccia. *Code Injection Attacks on Harvard-Architecture Devices*. Proceedings of the 15th ACM conference on Computer and communications security (CCS 2008), pp. 15-26, Alexandria (USA), October 2008.
11. J. Newsome, E. Shi, D. Song, and A. Perrig. *The Sybil Attack in Sensor Networks: Analysis & Defenses*. Proceedings of the IEEE 3nd International Workshop on Information Processing in Sensor Networks (IPSN'04), pp. 259-268, Berkeley (USA), April 2004.
12. C. Karlof, and D. Wagner. *Secure Routing in Wireless Sensor Networks: Attacks and Countermeasure*. Ad-Hoc Networks, vol. 1, no. 2-3, pp. 293-315, September 2003.
13. M. Manzo, T. Roosta, and S. Sastry. *Time Synchronization Attacks in Sensor Networks*. Proceedings of the 3th ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2005), pp. 107-116, Alexandria, (USA), November 2005.
14. V. Shnayder, M. Hempstead, B. Chen, G.W. Allen, and M. Welsh. *Simulating the Power Consumption of Large-Scale Sensor Network Applications*. Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys 2004), pp. 188-200, Baltimore (USA), August 2004.
15. E. Sabbah, A. Majeed, K.-D. Kang, K. Liu, and N. Abu-Ghazaleh. *An Application-driven Perspective on Wireless Sensor Network Security*. Proceedings of the 2nd ACM International Workshop on Quality of Service & Security for Wireless and Mobile Networks, pp. 1-8, Torremolinos (Spain), 2006.
16. S. Ransom, D. Pfisterer, and S. Fischer. *Comprehensible Security Synthesis for Wireless Sensor Networks*. Proceedings of the 3rd international Workshop on Middleware for Sensor Networks, pp. 19-24, Leuven (Belgium), 2008.
17. R. Roman. *Application-driven Security in Wireless Sensor Networks*. Ph.D. Thesis, University of Malaga, June 2008.
18. 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.

19. 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.
20. 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.
21. Winston K. G. Seah, Z. A. Eu and Hwee-Pink Tan. *Wireless Sensor Networks Powered by Ambient Energy Harvesting (WSN-HEAP) - Survey and Challenges*. Proceedings of CTIF Wireless VITAE 2009, Aalborg (Denmark), May 2009.
22. M.T.Penella, and M. Gasulla. *A Review of Commercial Energy Harvesters for Autonomous Sensors*. Proceedings of IEEE Instrumentation and Measurement Technology Conference Proceedings (IMTC 2007), pp. 1-5, Warsaw (Poland), May 2007.
23. Crossbow Technology, Inc. *eKo Pro Precision Agriculture*. `http://www.xbow.com/eko/`, Retrieved on June 2009.
24. D. Galindo, R. Roman, and J. Lopez. *A Killer Application for Pairings: Authenticated Key Establishment in Underwater Wireless Sensor Networks*. Proceedings of the 7th International Conference on Cryptology and Network Security (CANS 2008), pp. 120-132, Hong Kong (PRC), December 2008.
25. S.A. Camtepe, and B. Yener. *Key Management in Wireless Sensor Networks*. On Wireless Sensor Network Security, IOS Press. ISBN: 978-1-58603-813-7, 2008.
26. F. Stajano, D. Cvrcek, and M. Lewis. *Steel, Cast Iron and Concrete: Security Engineering for Real World Wireless Sensor Networks*. Proceedings of the 6th International Conference on Applied Cryptography and Network Security (ACNS 2008), pp. 460-478, New York (USA), June 2008.
27. C. Alcaraz, and R. Roman. *Applying Key Infrastructures for Sensor Networks in CIP/CIIP Scenarios*. 1st International Workshop on Critical Information Infrastructures Security (CRITIS 2006), pp. 166-178, Samos (Greece), August-September 2006.
28. ECRYPT Network of Excellence. *eSTREAM, the ECRYPT Stream Cipher Project*. `http://www.ecrypt.eu.org/stream/`, Retrieved on June 2009.
29. European Organization for Nuclear Research (CERN). *LHC - The Large Hadron Collider*. `http://lhc.web.cern.ch`, Retrieved on June 2009.
30. D. Culler, D. Estrin, and M. Srivastava. *Overview of Sensor Networks*. IEEE Computer, vol. 37, no. 8, pp. 41-49, August 2004.
31. I. Akyildiz, D. Pompili and T. Melodia. *Underwater acoustic sensor networks: Research challenges*. Ad Hoc Networks Jounal (Elsevier), vol. 3, no. 3, pp. 257-279, May 2005.
32. 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.
33. I.F. Akyildiz, and E.P. Stuntebeck. *Wireless Underground Sensor Networks: Research Challenges*. Ad Hoc Networks Journal, vol. 4, no. 6, pp. 669-686, November 2006.
34. M.C. Vuran, and I.F. Akyildiz. *Cross-layer Packet Size Optimization for Wireless Terrestrial, Underwater, and Underground Sensor Networks*. Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM 2008), pp. 226-230, Phoenix (USA), April 2008.

35. M.A. Hanson, H.C. Powell, A.T. Barth, K. Ringgenberg, B.H. Calhoun, J.H. Aylor, and J. Lach. *Body Area Sensor Networks: Challenges and Opportunities*. IEEE Computer, vol. 42, no. 1, pp. 58-65, January 2009.
36. S. Ullah, H. Higgin, M.A. Siddiqui, and K.S. Kwak. *A Study of Implanted and Wearable Body Sensor Networks*. Proceedings of the 2nd KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications (KES-AMSTA 2008), pp. 464-473, Incheon (Korea), March 2008.
37. I.F. Akyildiz, T. Melodia, and K.R. Chowdhury. *Wireless multimedia sensor networks: A survey*. IEEE Wireless Communications, vol. 14, no. 6, pp. 32-39, December 2007.
38. R. Di Pietro, L.V. Mancini, A. Spognardi, C. Soriente and G. Tsudik. *Catch Me (If You Can): Data Survival in Unattended Sensor Networks*. Proceedings of the 6th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'08), pp. 185-194, Hong Kong (China), March 2008.
39. M. Albano, A. Brogi, R. Popescu, M. Diaz, and J. A. Dianes. *Towards Secure Middleware for Embedded Peer-to-Peer Systems: Objectives & Requirements*. Proceedings of the 2nd Workshop on Requirements and Solutions for Pervasive Software Infrastructures (RSPSI 2007), pp. 1-6, Innsbruck (Austria), September 2007.
40. R. Roman, C. Alcaraz, and N. Sklavos. *On the Hardware Implementation Efficiency of Cryptographic Primitives*. On Wireless Sensor Network Security, IOS Press. ISBN: 978-1-58603-813-7.
41. S. Didla, A. Ault, and S. Bagchi. *Optimizing AES for Embedded Devices and Wireless Sensor Networks*. Proceedings of the 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TRIDENTCOM 2008), Innsbruck (Austria), March 2008.
42. K. Jun Choi, and J.-I. Song. *Investigation of Feasible Cryptographic Algorithms for Wireless Sensor Network*. Proceedings of the 8th International Conference on Advanced Communication Technology (ICACT 2006), Phoenix Park (Korea), February 2006.
43. I. Mantin. *Analysis of the Stream Cipher RC4*. Master's Thesis, Weizmann Institute of Science, 2001.
44. G. Meiser, T. Eisenbarth, K. Lemke-Rust, C. Paar. *Efficient Implementation of eSTREAM Ciphers on 8-bit AVR Microcontrollers*. Proceedings of the International Symposium on Industrial Embedded Systems (SIES 2008), pp. 58-66, Montpellier (France), June 2008.
45. 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.
46. S.C. Seo, D.-G. Han, H.C. Kim, and S. Hong. *TinyECCK: Efficient Elliptic Curve Cryptography Implementation over $GF(2^m)$ on 8-bit MICAz Mote*. IEICE Transactions on Info and Systems, vol. E91-D, no. 5, pp. 1338-1347, 2008.
47. P. Szczechowiak, A. Kargl, M. Scott, and M. Collier. *On the Application of Pairing based Cryptography to Wireless Sensor Networks*. Proceedings of the 2nd ACM conference on Wireless Network Security (WiSec 2009), pp. 1-12, Zurich (Switzerland), March 2009.
48. P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu. *Analyzing and Modeling Encryption Overhead for Sensor Network Nodes*. Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications (WSNA 2003), pp. 151-159, San Diego (USA), September 2003.

49. X. Wang. *Recent Progress on SHA-1.* Rump Session, Crypto 2005.
50. NIST hash function competition. `http://www.nist.gov/hash-competition`. Retrieved on June 2009.
51. L. Eschenauer, and V.D. Gligor. *A Key-management Scheme for Distributed Sensor Networks.* Proceedings of the 9th ACM conference on Computer and communications security (CCS '02), pp. 41-47, Washington DC (USA), November 2002.
52. W. Du, J. Deng, Y. S. Han, P. Varshney, J. Katz, and A. Khalili. *A Pairwise Key Predistribution Scheme for Wireless Sensor Networks.* ACM Transactions on Information and System Security (TISSEC), vol. 8, no. 2, pp 228-258, May 2005.
53. S. A. Camtepe, and B. Yener. *Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks.* IEEE/ACM Transactions on Networking, vol. 15, no. 2, pp. 346-358, April 2007.
54. D. Liu, P. Ning, and R. Li. *Establishing Pairwise Keys in Distributed Sensor Networks.* ACM Transactions on Information and System Security, vol. 8, no. 1, pp. 41-77, February 2005.
55. R.J. Anderson, H. Chan, and A. Perrig. *Key Infection: Smart Trust for Smart Dust.* Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP 2004), pp 206-215, Berlin (Germany), October 2004.
56. A. Seshadri, M. Luk, and A. Perrig. *SAKE: Software Attestation for Key Establishment in Sensor Networks.* Proceedings of the 2008 International Conference on Distributed Computing in Sensor Systems (DCOSS'08), pp. 372-385, Santorini Island (Greece), June 2008.
57. B. Panja, S. Madria, and B. Bhargava. *Energy and Communication Efficient Group Key Management Protocol for Hierarchical Sensor Networks.* Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06), Taichung (Taiwan), June 2006.
58. G. Acs, and L. Buttyan. *Secure Routing in Wireless Sensor Networks.* On Wireless Sensor Network Security, IOS Press. ISBN: 978-1-58603-813-7.
59. Routing Over Low power and Lossy networks (ROLL) Working Group. Internet Engineering Task Force (IETF). `http://www.ietf.org/html.charters/roll-charter.html`. Retrieved on June 2009.
60. H. Alzaid, E. Foo, J.G. Nieto. *Secure Data Aggregation in Wireless Sensor Network: a Survey.* Proceedings of the Australasian Information Security Conference 2008: Conferences in Research and Practice in Information Technology (CRPIT), pp. 93-105, Wollongong, NSW (Australia), March 2008.
61. S. Ozdemir, and Y. Xiao. *Secure data aggregation in wireless sensor networks: A comprehensive overview.* Computer Networks (Elsevier), 2009, Article in press: `http://dx.doi.org/10.1016/j.comnet.2009.02.023`.
62. A. Boukerche, D. Turgut. *Secure time synchronization protocols for wireless sensor networks.* IEEE Wireless Communications, vol. 14, no. 5, pp. 64-69, October 2007.
63. Y. Yang, and Y. Sun. *Securing Time-synchronization Protocols in Sensor Networks: Attack Detection and Self-healing.* Proceedings of the IEEE 2008 Global Telecommunications Conference (GLOBECOM 2008), pp. 1-6, New Orleans (USA), November-December 2008.
64. S. Ganeriwal, C. Popper, S. Capkun, and M.B. Srivastava. *Secure Time Synchronization in Sensor Networks.* ACM Transactions on Information and System Security (TISSEC), vol. 11, no. 4, July 2008.
65. 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.

66. R. Roman, J. Lopez, and S. Gritzalis. *Situation Awareness Mechanisms for Wireless Sensor Networks*. IEEE Communications Magazine. vol. 46, no. 4, pp 102-107, 2008.
67. C. Ozturk, Y. Zhang, W. Trappe, and M. Ott. *Source-Location Privacy for Networks of Energy-Constrained Sensors*. Proceedings of the 2004 IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems (WSTFEUS'04), Vienna (Austria), May 2004.
68. E. Aivaloglou, S. Gritzalis, and C. Skianis. *Trust establishment in sensor networks: Behaviour-based, certificate-based and a combinational approach*. International Journal System of Systems Engineering, vol. 1, no. 1/2, pp. 128-148, 2008.
69. R. Roman, M. C. Fernandez-Gago, 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, ISBN: 978-1905886-906, 2009.
70. L. Na, N. Zhang, S.K. Das, B. Thuraisingham. *Privacy Preservation in Wireless Sensor Networks: A State-of-the-art Survey*. Ad Hoc Networks (Elsevier), 2009, Article in press: `http://dx.doi.org/10.1016/j.adhoc.2009.04.009`.
71. A. Yasinsac. *Remote Attestation - Identification*. On Wireless Sensor Network Security, IOS Press. ISBN: 978-1-58603-813-7.
72. L. Sang, A. Arora. *Spatial Signatures for Lightweight Security in Wireless Sensor Networks*. Proceedings of the 27th IEEE Conference on Computer Communications (INFOCOM 2008), pp. 2137-2145, Phoenix (USA), April 2008.
73. B. Danev, and S. Capkun. *Physical-layer Identification of Wireless Sensor Nodes*. Technical Reports 604, ETH Zrich, System Security Group D-INFK, 08 2008.
74. 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.
75. ZigBee Alliance, `http://www.zigbee.org/`, Retrieved on June 2009.
76. HART Communication, `http://www.hartcomm2.org`, Retrieved on June 2009.
77. ISA100.11a, Wireless Systems for Industrial Automation: Process Control and Related Applications, `http://www.isa.org/isa100`, Retrieved on June 2009.
78. IEEE Standard, 802.15.4-2006. *Wireless medium access control and physical layer specifications for low-rate wireless personal area networks*. ISBN 0-7381-4997-7, 2006.
79. 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.
80. Modbus-IDA. The Architecture for Distributed Automation, `http://www.modbus.org`, Retrieved on June 2009.
81. DNP3. DNP Users Group, `http://www.dnp.org`, Retrieved on June 2009.
82. IEC 60870-5-104, International Electrotechnical Commission, `http://www.iec.ch/`, Retrieved on June 2009.
83. PI Profibus - Profinet, `http://www.profibus.com`, Retrieved on June 2009.
84. Fieldbus Foundation, `http://www.fieldbus.org`, Retrieved on June 2009.
85. G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. *RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks*. Request for Comments, September 2007.
86. International Telecommunication Union. *The Internet of Things*. ITU Internet Reports 2005.
87. R. Roman, and J. Lopez. *Integrating Wireless Sensor Networks and the Internet: A Security Analysis*. Internet Research, vol. 19, no. 2, pp 246-259, 2009.