C. Alcaraz, J. Lopez, R. Roman, and H-H. Chen, "Selecting key management schemes for WSN applications", *Computers & Security*, vol. 31, pp. 956966, 2012.
http://doi.org/10.1016/j.cose.2012.07.002
NICS Lab. Publications: https://www.nics.uma.es/publications

Selecting Key Management Schemes for WSN Applications

Cristina Alcaraz[1], Javier Lopez[1], Rodrigo Roman[2], Hsiao-Hwa Chen[3]

[1]Computer Science Department, University of Malaga,
Campus de Teatinos s/n, 29071, Malaga, Spain

[2]Institute for Infocomm Research, 1 Fusionopolis Way,
#19-01 Connexis, South Tower, Singapore 138632

[3]Department of Engineering Science, National Cheng Kung University,
No.1 University Road, Tainan City, 70101, Taiwan

[1]{alcaraz,jlm}@lcc.uma.es, [2]rroman@i2r.a-star.edu.sg, [3]hshwchen@ieee.org

October 27, 2015

Abstract

Key management in wireless sensor networks (WSN) is an active research topic. Due to the large number of key management schemes (KMS) proposed in the literature, it is not easy for a sensor network designer to know exactly which KMS best fits in a particular WSN application. In this article, we offer a comprehensive review on how the application requirements and the properties of various key management schemes influence each other. Based on this review, we show that the KMS plays a critical role in determining the security performance of a WSN network with given application requirements. We also develop a method that allows the network designers to select the most suitable KMS for a specific WSN network setting. In addition, the article also addresses the issues on the current state-of-the-art research on the KMS for homogeneous (i.e. non-hierarchical) networks to provide solutions for establishing link-layer keys in various WSN applications and scenarios.

Keywords: Sensor Networks, Key Management, Network Design, Optimization, Security

1 Introduction

Wireless sensor networks (WSN) help to bridge the gap between the real world and computer systems and the Internet. By using small-sized and power-constrained pervasive devices, called sensor nodes, physical information such as temperature, humidity, and so forth, can be accessible easily from any other application. However, there are many challenges in managing these types of distributed and pervasive sensor networks, especially when hundreds or thousands of these wireless-enabled and self-configurable devices, which may not necessarily belong to the same network, are deployed within the same monitoring area. Security is one of the major challenges when implementing WSNs.

Due to many specific characteristics of a WSN, it is difficult to setup a secure link-layer channel (i.e. based on pairwise key) between neighboring nodes. This issue has been extensively investigated by researchers recently, and many key management schemes (KMS) have been proposed in the open literature. Most of them were proposed on the assumption that they could be used in any WSN scenario. However, the internal design of the different schemes varies substantially in most cases, hence providing different features or properties. With a large number of available KMS in the literature, it is not an easy task for a network designer to pinpoint the right scheme or protocol for WSN applications without resorting to an exhaustive search, which could be a very time consuming process.

Motivated by this fact, in this paper we conduct a detailed analysis of the different KMS properties, highlighting the relationship between these properties and the requirements of WSN applications. Using this analysis and a specific method (the SenseKey tool), we will highlight the importance of selecting the most suitable KMS protocols for limited or critical contexts, such as smart grids and energy control substations. This paper expands the concepts presented in a previous version [Roman et al.(2011)Roman, Lopez, Alcaraz, and Chen], providing not only a wider overview of the existing KMS properties and protocols, but also a study that uncovers open issues by examining the suitability and applicability of the actual state of the art. Note that our analysis will focus only on homogeneous sensor networks (i.e. networks where all peers engaged in the communication process have similar computational capabilities), thus we will not consider those KMS protocols that try to take advantage of the existence of a powerful device. Moreover, in this article we will use the abbreviation “KMS” as a generic term and the word “protocol” (or “KMS protocol”) to denote a particular key management scheme for WSN applications.

The rest of the paper is organised as follows. In Section 2 we will categorize the different KMS protocols and highlight the properties that define their overall behaviors. The issues on how those properties can be mapped to the requirements of WSN applications are discussed in Section 3. In Section 4 we introduce and develop a method that assists the network designer in the selection of the most appropriate protocol for different WSN application scenarios. We also provide in this section an example of how we can use this method to select a certain KMS. In Section 5, we discuss the issues of whether the current state-of-the-art research in KMS can offer a viable solution to the link-layer key management problem, and thus determine where our future research effort should be focused in order to design more suitable KMS protocols for real-world WSN applications. This is followed by the conclusions of this paper in Section 6.

2 Security and Key Management

The relevance of security in WSNs is substantiated by many existing threats that may hinder several major functionalities of the world-wide networks. Due to the inherent openness of the wireless channels and the limited capabilities of the sensor nodes, it can be relatively easy for a knowledgeable adversary to monitor or even take control of the behavior of an unprotected WSN [Zhou et al.(2008)Zhou, Fang, and Zhang]. As these types of intrusions are unavoidable, a sensor network must be prepared to prevent or minimize the effect of such attacks by using various possible mechanisms such as secure communication channels, secure protocols (e.g. routing, aggregation, time synchronization), context awareness and trust measures, secure location mechanisms [Zhu et al.(2011)Zhu, Xiang, Zhou, Deng, and Bao], etc.

The foundation for the creation of secure protocols is the security primitives. These security primitives, such as symmetric key cryptography (SKC) and public key cryptography (PKC), allow the construction of a secure communication channel at the link layer between two or more devices, providing confidentiality, integrity, and authentication. Together with other specific protection mechanisms, such as time-stamping, it is possible to prevent most external attacks such as message injection, eavesdropping, and packet relaying, thus protecting the information flow from any unintended recipients.

However, each device that wants to open a secure channel with its neighbors must share between them some security credentials, i.e. the link-layer (point-to-point) secret keys. KMS aim to solve the problem of creating, distributing, and maintaining those secret keys. The design of a good KMS for sensor networks is not a trivial task, as it is unwise to rely on centralized entities due to the distributive and self-configurable nature of the WSN networks. Also, the existing constraints (e.g. memory, computational capabilities, etc.) of sensor nodes discourage the use of resource-intensive algorithms for most WSN scenarios. In addition, there are many other factors, such as network size, nodes connectivity, energy spent in key setup processes, etc, which also affect the design of a KMS.

It should be noted that, for certain network configurations and applications, the KMS must be able to support the negotiation of other types of entity credentials. For example, whenever a limited set of nodes (e.g. a cluster of nodes in a hierarchical network) need to communicate with one another in a secure manner, a mechanism to create and maintain group keys is necessary. Besides, certain applications require the use of a secret key for opening a secure end-to-end channel between two nodes (e.g. a distant node and the base station). Many existing KMS protocols were proposed to provide solutions to these specific situations. Nevertheless, due to the fact that communication security using link-layer keys is one of the foundations for security assurance of sensor networks, the remainder of this paper will focus on the schemes that establish the link-layer keys.

2.1 KMS Frameworks

Due to its importance, the design of KMS for a WSN has received a lot of attention in the literature, and many different types of protocols have been proposed [Camtepe and Yener(2008)]. In fact, since one of the most important sensor network link-layer standards, IEEE 802.15.4, does not specify how secret keys should be exchanged, it is essential to make use of a KMS protocol. In general, for homogeneous sensor networks, these protocols can be classified into four major frameworks: key pool framework, mathematical framework, negotiation framework, and public key framework. Although the core function of all the frameworks is similar (i.e. to bootstrap secret keys needed by the link layer), their underlying mechanisms and design goals are different. In what follows we will introduce all frameworks together with their underlying design choices.

The key pool paradigm plays a pivotal role in the *key pool framework*, one of the most important KMS frameworks that has ever been proposed so far. The basic scheme behind this framework is quite simple [Eschenauer and Gligor(2002)]. First, the network designer creates a key pool, i.e. a large set of precalculated secret keys. Second, before network deployment each node is assigned a unique key chain, i.e. a small subset of the keys from the key pool (key pre-distribution phase). Third, after the network deployment, the nodes exchange their identification numbers of the keys from their key chains, trying to find a common shared secret key (shared-key discovery phase). Finally, in the case that two nodes do not share the same key, they try to find a key path between them in order to negotiate a pairwise key (path-key establishment phase). The principal design goal of the protocols that belong to this framework is to ensure a limited secure connectivity between the nodes, regardless of the size of the network.

Some KMS protocols have been proposed using mathematical algorithms (i.e. linear algebra, combinatorics, and algebraic geometry, etc.) for calculating the pairwise keys of the nodes. Those protocols belong to the *mathematical framework*. Among the KMS protocols based on linear algebra, the most important scheme is the Blom scheme
[Du et al.(2005)Du, Deng, Han, Varshney, Katz, and Khalili]. In this scheme, each node i can calculate the pairwise key it shares with another node j by solving $A(i) \cdot G(j)$, where G is a public Vandermonde matrix and A is calculated using a symmetric random secret matrix D. In the KMS protocols based on combinatorics, generalized quadrangle and symmetric design models [Camtepe and Yener(2007)] are the most widely used. Using generalized quadrangles $GQ(s,t)$ or finite projective planes $FPP(q)$ (incidence structures, whose elements are points and lines, that satisfy various axioms related to the number of points (keys) that can be found in every line (key chain), a network designer can construct a key chain of size $s+1$ or $q+1$, respectively. Finally, for the KMS protocols developed based on algebraic geometry, the most well known scheme is based on the bivariate polynomials [Liu and Ning(2003)]. In this scheme, by using a bivariate polynomial f, every node A in a network is able to obtain a pairwise key with another node y by solving $f(A, y)$. All of the aforementioned protocols allow the creation of pairwise keys between the nodes without an ex-

cessive communication overhead. Unfortunately, these designs are often difficult to apply, and it is not easy to make them scalable either.

It is also possible to let nodes negotiate their keys with their close neighbors right after the deployment of a network. Typically, this requires a few steps in the pre-distribution phase. All of the protocols that generate their keys through mutual agreement belong to the *negotiation framework*. Many of these protocols work under the assumption that there is little or no threat against the integrity of a WSN network in the very early stage of its lifetime [Anderson et al.(2004)Anderson, Haowen, and Perrig]. Nevertheless, it is possible to use other mechanisms and protocols (such as the Guy Fawkes protocol [Seshadri et al.(2008)Seshadri, Luk, and Perrig], etc.) in order to ensure the authenticity of the peers in any stage of the network deployment. The other protocols that are included in this framework are those that can organize a network into dynamic or static clusters [Panja et al.(2006)Panja, Madria, and Bhargava]. Finally, symmetric key agreement schemes of existing WSN standards such as ZigBee PRO [ZigBee Alliance(2008)], WirelessHART [HART Communication Foundation(2009)] and ISA100.11a [ISA100(2011)] also fall within this category.

All of the frameworks mentioned above rely solely 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. For those protocols that belong to the *public key framework*, two nodes need to exchange their public keys and some other information to effectively create their pairwise secret keys. Although resource-constrained sensor nodes are able to use PKC through Elliptic curve cryptography (ECC [Liu and Ning(2008)]), the amount of memory required to implement the algorithm and the time/energy needed to complete the negotiation is, in most cases, substantially higher than other KMS frameworks.

Recently, some improvements over the original KMS protocols have been reported in the literature. These new protocols either provide better properties or are more suitable for certain application scenarios. For example, it is possible in all aforementioned frameworks to take advantage of the *deployment knowledge*, that is the knowledge of the final node locations in the WSN field. Using this knowledge, the building elements of each framework (e.g. keys inside the key pool, polynomials stored inside a node, etc.) can be optimized [Lee and Stinson(2008)]. The *communication overhead* is another factor that is critical in the development of any sensor network protocol. As a result, the optimizations are focused on reducing the number and/or size of the messages employed in the process of discovering or negotiating shared secret keys among the peers [H. Chan(2003)].

Another optimization issue is to achieve a balance between the advantages and disadvantages of different schemes. For example, the underlying structure of mathematical framework protocols can be modified either to enhance certain properties such as *network resilience* [Zhang et al.(2007)Zhang, Tran, Zhu, and Cao] or to provide some properties such as extensibility

[Wen et al.(2009)Wen, Zheng, Ye, Chen, and Qiu]. Also, the building blocks of the existing frameworks can be combined in order to create new protocols that would benefit from most of their advantages while mitigating the problems caused by their disadvantages. For example, the inherent drawbacks of certain mathematical framework protocols can be solved partially by using the key pool paradigm [Liu and Ning(2003)]. Also, it is attractive to use public key cryptography together with the other frameworks, where the PKC primitives could be used only for rekeying and for the cases where two nodes do not have a common secret key.

3 Mapping Application Requirements to KMS Properties

While there are many solutions available that can be used to bootstrap the link-layer keys of a certain WSN deployment, it is always a difficult task to know which solution fits best with a particular WSN. Most of existing KMS protocols could indeed work to distribute secret keys in a wireless sensor network. However, due to their design differences, they may comply with different sets of security and operational requirements, such as their ability to support large networks (i.e. scalability), a secure shared-key discovery phase (i.e. confidentiality), and so on.

While some protocols may seem to be better than the others, it is not clear exactly which one performs best, as most protocols do not specify in which applications they could be most useful. In fact, additional problems can arise when an inappropriate KMS is chosen for a particular application. For example, a basic implementation of the symmetric design approach [Camtepe and Yener(2007)] allows the creation of a pairwise key between any pair of nodes, which is essential for applications with mobile entities. However, from the point of view of security, the resilience of that design is limited: the communication channel will be completely vulnerable against any attacks once an adversary captures a very small set of nodes. Therefore, in order to avoid such situations, time-consuming trial-and-error experiments and exhaustive protocol analyses may seem to be the only options available in the selection process.

However, it might be possible to find a good solution to this problem by making the following assumption: the needs of a particular application will determine the choice of a particular KMS protocol. In fact, this assumption is also applicable to other WSN protocols such as routing, as they must comply with some application-specific requirements. For example, if the knowledge of the physical locations of the nodes is critical, the routing protocols may need to use geographical information (e.g. GPS information) to locate the nodes and forward packets.

Based on the constraints of a particular WSN scenario, it is possible to select a KMS protocol the properties of which can meet the requirements of the application to be deployed. For instance, a long lifetime small sensor network

Table 1: Relevant KMS properties

Property name		**Abbr.**	**Description**
Memory footprint		*Mm*	ROM and RAM used for the protocol
Communication overhead		*Cm*	Number of messages exchanged between peers
Processing speed		*Sp*	Computational cost of the protocol
Network bootstrapping		*Sec*	Confidentiality of the bootstrap process
Network resilience		*Rs*	Resistance against stolen credentials
Connectivity	Global conn.	*GC*	Existence of a key path between any node
	Local conn.	*LC*	Existence of a shared secret between neighbour nodes
	Node conn.	*NC*	Existence of a shared secret between any nodes
Scalability		*Sc*	Support for big networks
Extensibility		*Ex*	Capability of adding new nodes
Energy		*En*	Optimization of the energy usage

needs to pay sufficient attention to the energy overhead that a KMS consumes in the key distribution process, but it may neglect the other aspects such as the scalability of the WSN the KMS can offer. In order to perform this selection process, it is imperative to identify what properties could characterize the behaviors of the KMS protocols. These properties, which are shown in Table 1, were obtained by analyzing both the state of the art and existing surveys. All properties are defined based on the security and operational requirements of sensor networks relevant to the key management. Both these properties and their link to the application requirements are described below.

Memory Footprint (Mm)

A sensor node is usually constrained in terms of memory (e.g. < 4 kB of data memory and < 48 kB of instruction memory). Therefore, it is important to know what is the amount of data memory that a KMS requires for storing the security credentials used for bootstrapping the entire infrastructure. The amount of instruction memory that a KMS needs to implement the whole protocol is also important.

From a system development point of view, it is essential to have enough free memory for implementing different WSN applications and for storing temporary data. As a result, the importance of this property will be associated closely with the complexity of a particular application. In general, the simpler the application is, the more space should be made available for storing security credentials.

Communication Overhead (Cm)

In most KMS protocols, the nodes must exchange information with their peers through communication channels in order to establish their pairwise keys. While some protocols may require the exchange of a small amount of information, the other protocols may need to undergo complex negotiation processes among peers.

There are many WSN scenarios where the communication overhead should be reduced to a minimal level. For instance, when an application needs to establish the links to provide its services within a very short period of time.

Processing Speed (Sp)

Sensor nodes are usually severely constrained in terms of their computing power. Fortunately there are many KMS protocols that are not very computationally intensive. As for the communication overhead property, the time consumed in negotiating the pairwise keys is mainly spent with the duration in sending and receiving messages through the wireless channels. Nevertheless, some protocols may indeed require some computational-intensive tasks.

The processing times for different key management schemes are associated closely with the communication overhead required by the KMS. In particular, some WSN applications may require setting up a secure channel between two previously unknown nodes as fast as possible. A typical example is a WSN where mobile nodes are in motion and they only get close enough within each other's radio range briefly. Thus, a fast establishment of the communication link is critical, and a reduction in the overhead can help to shorten the link establishment time.

Network Bootstrapping (Sec)

It should be noted that the whole process of the keys distribution must be secure by itself. For example, the confidentiality of the key distribution process taking place in the very early stage of the life-time should be assured. Actually, the data exchange in most protocols can also provide information about the secret key (e.g. the index of the keys from a key chain), and adversaries may derive the secret key itself from that information. To mitigate this problem, some protocols do not need to exchange sensitive information (e.g. IDs of the nodes, etc.), and thus they will not reveal any information about the secret key. However, some protocols do assume that the network is less likely to be in danger right after its deployment, and it may tend to exchange some secret information without any protection whatsoever.

For the applications where the deployment environment is secure enough, confidentiality is not an issue since there are no attackers in the area that may steal the security credentials. However, the problems may arise whenever the deployment area is open to public or when the information managed by the sensor nodes is important.

Network Resilience (Rs)

Network resilience indicates the ability to cope with stolen credentials and rogue nodes. The higher the network resilience is, the lower the chance is for a malicious attacker to control a significant part of the network using the stolen credentials. The best resilience is offered by the KMS protocols where nodes share pairwise keys only with their direct neighborhood.

If the environment where a sensor network is deployed is heavily protected or the probability of capturing a set of nodes is extremely low, then the network resilience is not a critical factor. Consequently, the need for network resilience increases with the chance of a node being subverted by an adversary.

Connectivity

Connectivity is a property which is related to the capability for two sensor nodes to share the same security credentials. There are three main connectivity properties, as listed below.

- Global connectivity (GC) is the ratio between the largest size of isolated components in the network and the size of the whole network. If the GC is 100%, it means that there is always a key path, i.e. a secure routing path, between any two nodes in the network.
- Local connectivity (LC) can be defined as the probability that two neighboring nodes share the same secret key right after the network starts to operate. If LC is 100%, then any node can securely communicate with any of its neighbors without negotiation.
- Node connectivity (NC) is the probability of any two nodes in the network sharing the same secret key, regardless of their location in the network. If NC is 100%, then any node in the network can open a pairwise secure channel with any other node.

Usually, global connectivity is an essential property because in many WSN scenarios all nodes are equally important for providing the network services. Since in most cases the locations of the nodes inside the network are unknown, it is usually important to have a high local connectivity, in order to ensure that most nodes will be able to set up a pairwise key with their neighborhood, producing the least overhead possible. Finally, node connectivity is indispensable in some WSN application scenarios where nodes are mobile, requiring a secure channel to be opened with any node in their vicinity. Moreover, node connectivity also is of great importance in scenarios where the survivability of the network is at risk due to faulty nodes or broken links: new secure communication links can be created any time between any pair of nodes.

Scalability (Sc) and Extensibility (Ex)

A KMS protocol provides scalability if it is able to support a WSN network with a large number of nodes (at least at an order of thousands). On the other hand, a KMS protocol is extensible if it allows the inclusion of new nodes after its initial deployment. Not all protocols provide scalability and extensibility, due to various reasons (e.g. unoptimized underlying mechanisms, high memory requirements, involvement of the gateway in the negotiation process).

Scalability is not an important issue for the WSN applications that require a small number of sensor nodes (i.e. less than 100). However, as the size of the network increases, the scalability becomes more important: some protocols are not designed to manage a network with a large number of nodes due to memory constraints or other issues. The extensibility property is important for those scenarios where there may be hostile external entities. It is also important for those applications which have to provide a service for a relatively long period of time (where there should be some maintenance policies).

Energy (En)

A sensor node usually relies on batteries for powering itself. Since the establishment of pairwise security credentials between nodes is an energy-consuming task (transmitting the security credentials, sending/receiving data to/from other peers, etc.), it is important to consider how much energy is consumed during the operation of a KMS protocol.

On the other hand, there are also some scenarios where energy saving is not a critical factor. For instance, a WSN with a relatively short lifetime does not need to take into account energy saving as much. Also, some network configurations may make it possible for the nodes to access unlimited energy sources, such as solar energy or even normal power lines. In addition, some WSN applications may need to send security credentials only a few times over the entire lifetime of the network.

4 A Tool for Selecting KMS Protocols

In the previous section, we have discussed that it is of upmost importance to choose a protocol that can best fit a certain WSN application, and that this can be achieved by mapping the requirements of the application to the properties of the KMS protocol to be used. However, this is not an easy task for a network designer because there are too many different types of protocols. A network designer needs some means that enable him/her to select a subset of protocols that are likely to be more suitable for a given application than others.

In fact, there are already some methods developed in the research literature that try to help network designers in this complex task. Some tools (such as [Lee et al.(2007)Lee, Kim, Lee, and Lim]) allow the selection of a KMS protocol according to the threat level of the application context. As there are other factors besides the attacker model, newer tools (such as [Na et al.(2011)Na, Ren, Hori, and Sakurai]) make use of numerical priorities derived from the network designer's input (e.g. in an army context resilience is much more important than scalability). However, these methods, although valid, might not accurately represent the needs of a certain application where certain properties are of equal importance. Besides, sometimes the results are numeric values that do not give explicit feedback on the properties fulfilled by the protocols. Finally, these methods do not consider all properties presented in Section 3. Therefore, in this section we present a method where each property is checked independently from the others. Our method also provides information for network designers about the relevancy of every property, and can be easily automated. Needless to say, our method can be used in conjunction with other methods if deemed necessary by the network designer.

The first step of the method (step 0) consists of constructing the table shown in Table 2, which provides information on how different protocols fit with the properties introduced in Section 3. Actually, the table includes the most relevant KMS protocols mentioned in the literature (in terms of number of citations, novelty, and relevant design). It should be noted that this table can be further extended by adding other new protocols, just by inserting an additional row with the properties of each of the new protocols. Some properties are labeled as design-dependant, since there are protocols that can be tweaked in order to offer better properties. Also, it is indicated whether a certain protocol requires particular deployment knowledge.

Table 2: Major properties for widely used KMS

Protocol	Cm	GC	LC	NC	En	Ex	Mm	Rs	Sc	Sec	Sp
Key Infection [Anderson et al.(2004)Anderson, Haowen, and Perrig]	−	++	++	−−	+	−−	++	++	++	−−	+
SAKE [Seshadri et al.(2008)Seshadri, Luk, and Perrig]		++	++	−−		++		++	++		−−
Generalized Quadrangle [Camtepe and Yener(2007)]	−−	++		−−		−−	++				++
Symmetric Design [Camtepe and Yener(2007)]	++	++	++	++		−−	++	−−			++
Hybrid Designs - Generalized Quadrangle [Camtepe and Yener(2007)]		○	○	○		−	++		○		++
Basic Probabilistic Key Pre-distribution [Eschenauer and Gligor(2002)]	−	−	−	−			○	−	++		
Random Key Pre-distribution [Jaworski et al.(2009)Jaworski, Renand, and Rybarczyk]	+	+	+	−−			○	−	++		
Random Pairwise Key [H. Chan(2003)]		○	○	○		−−	−	−	++	++	
Blom Key Pre-distribution [Du et al.(2005)Du, Deng, Han, Varshney, Katz, and Khalili]	++	++	++	++		−−	○	○	○	+	
Multiple Space Key Pre-distribution [Du et al.(2005)Du, Deng, Han, Varshney, Katz, and Khalili]	−	−	−	−	−−		○	++	++		
Multiple ID-Based one-way Function [Lee and Stinson(2008)]	−−	++	++	−−		−−	++	−−	++		++
Multiple Space Blom (MBS) [Lee and Stinson(2008)]	++					−−	○	○	○		
Deterministic Multiple Space Blom DMBS [Lee and Stinson(2008)]	−	−	−	−			○	++	++		
Robust Continuity Blom [Wen et al.(2009)Wen, Zheng, Ye, Chen, and Qiu]	+	++	++	++		++	○	○	+	+	−
Grid Based Key Pre-distribution [Liu and Ning(2003)]	++	++	++		−−	−−	○	++	○		−−
Polynomial Based Key Pre-distribution [Liu and Ning(2003)]	++	++	++	++	−−	++	○	○	○	++	−−
Random Subset Assignment [Liu and Ning(2003)]	−	−	−	−	−−		○	○	++		−−
PIKE [Chan and Perrig(2005)]	−	−	−	−		−	++		++		
LEAP+ [Zhu et al.(2006)Zhu, Setia, and Jajodia]		++	++	−−		++		○	++		++
Panja [Panja et al.(2006)Panja, Madria, and Bhargava]		++	++	−−		−−	++	−−	++		−−
RPB Scheme [Zhang et al.(2007)Zhang, Tran, Zhu, and Cao]	++	++	++	++		++	−	+	++	++	
CARPY+ [Yu et al.(2010)Yu, Lu, and Kuo]	++	++	++	++		++	−	+	++	++	
OSCAR [Delgado-Mohatar et al.(2011)Delgado-Mohatar, Fuster-Sabater, and Sierra]	+	++	++	++		+	+	++	++	+	
EDDK [Zhang et al.(2011)Zhang, He, and Wei]	+	++	++	+	−−	++	−	++	++	+	−−
Forward Authentication KMS [Huang et al.(2011)Huang, Lia		++	−	−−		++		−−	+	++	
Public Key Cryptography-based KMS [Liu and Ning(2008)]	++	++	++	++	−−	++		++	++	+	−−
ZigBee PRO [ZigBee Alliance(2008)]	−	++	++	++	○	++	++	−	○	+	
ZigBee Smart Energy [Garrison(2011)]	−	++	++	++	−−	++		++	○	++	−−
WirelessHART^{TM} [HART Communication Foundation(2009)]	−	++	++	++	○	++	++	−	○	+	
ISA100.11a Symmetric [ISA100(2011)]	−	++	++	++	○	++	++	−	○	+	
ISA100.11a Asymmetric [ISA100(2011)]	−	++	++	++	−−	++		−	○	++	−−

Notation

Advantage	++
Disadvantage	−−
Advantage, depending on the design of the protocol	+
Disadvantage, depending on the design of the protocol	−
Either advantage or disadvantage, depending on the design of the protocol	○
Node locations are known before deployment	▨

Based on the table generated in step 0, the following series of well-defined steps take place:

1. The network designer identifies the properties of the specific WSN scenario or application by analyzing its requirements, according to the descriptions given in Section 3. Then, he/she decides which of these properties are essential for a KMS in the context of the WSN application (*main properties*), and which are important but not essential (*secondary properties*).

2. The network designer selects from Table 2 all the KMS protocols that fulfill one or more of the *main properties* with the sign "++" or "+". These protocols conform the *KMS candidate set* for the particular WSN application.

3. From the KMS candidate set, the network designer discards all the protocols that have one of its main properties or secondary properties with the sign "$--$" or "$-$". Note that if a property is affected by the variables used in the design of the protocol, it should be ignored. As a result, the KMS candidate set will only contain the protocols that are at least good enough to fit the major properties of the application.

4. From the resultant KMS candidate set, the network designer discards all the protocols that do not have all the main properties marked with the sign "++" or "+". As a result of this step, the KMS candidate set will contain the protocols that provide the essential properties required for the WSN application.

5. Finally, the network designer reviews the protocols still contained in the KMS candidate set and choose the most suitable one for the application. Note that it is possible to order the protocols inside KMS candidate set in terms of their properties. Also, the network designer must pay attention to the design parameters of a protocol if any of the main properties or secondary properties are dependant on them.

In order to facilitate the use of our proposed KMS selection method, we have implemented it as a web tool known as SenseKey[1] tool, which is very simple and easy to use. In this tool, the user has to enter the main properties and secondary properties of the WSN application, together with the rest of the information (e.g. the existence of deployment knowledge, etc.). The interface of the SenseKey tool also offers context-sensitive help for each listed property. Once the properties are selected, the SenseKey tool will output a list of suitable KMS protocols, and they will be presented to the user in order of suitability.

It must be noted that the method may produce an empty KMS candidate set, meaning that there is no protocol that can completely fulfill the application requirements. However, it is important to realize that it is still possible to manually identify the protocols that partially comply with steps 3 and 4 in

[1]`http://www.lcc.uma.es/~roman/KMSCRISIS/`

the method. Therefore, it is possible to apply one of those partially suitable protocols for that scenario.

Use Case: Energy Control Substation of a Smart Grid

In order to understand the functionality of SenseKey for real applications, an example of its usability is presented in this Section. The initial step to use the web solution is precisely to identify the minimum requirements to be met in a certain context. In our case, the application context would be an energy control substation of a Smart Grid. We must extract and assume a set of requirements related to the application so as to identify later on those properties required by SenseKey. Namely:

Extracting application requirements: Our context of application will present the following characteristics.

1. Our application context is 'critical'. This means that any unforeseen change in a part of the system or its sensitive information (e.g. values of readings such as values of voltage v_i, alarms or commands) may cause a (minor or major) effect over the system or systems. This in a Smart Grid system can be seen as the impact over the rest of domains that compound the system itself (such as generation, distribution and transmission systems), with a high probability of provoking a cascading effect over other critical infrastructures [Alcaraz and Lopez(2012)]. Therefore, this application context requires the following property $\Rightarrow$ {Criticality}.
2. Our energy control substation is located in an 'isolated remote point' from the central system, where there is not any kind of human surveillance. This means that our application context requires $\Rightarrow$ {Remote Location, Isolation}.
3. All the devices of the substation are essential, and each device must be able to establish a secure channel with other devices, including handheld interfaces (e.g. smartphones or PDAs). Hence, also required is $\Rightarrow$ {Connectivity}.
4. Our context is 'long-life' and it requires a continued 'maintenance' $\Rightarrow$ {Maintenance}.
5. Despite that substations present small local dimensions whose control can be reduced to a small number of sensor nodes, it is possible to expand the network in the future. This means that the context needs the property $\Rightarrow$ {Growth}.
6. Lastly, our context requires 'performance', 'business continuity' and 'productivity'. Therefore, also required is $\Rightarrow$ {Performance}.

Once we have identified the application requirements, we need to link these requirements with the WSN requirements, so as to discover the main and secondary properties that will be used in SenseKey.

Identifying properties: let SK represent the set of properties of SenseKey {Cm, GC, LC, NC, En, Ex, Mm, Rs, Sc, Sec, Sp} and let SR denote the set of identified requirements {Criticality, Remote Location, Isolation, Maintenance, Growth, Performance}. Then, we must associate $x \rightarrow y$, $\forall$ $x \in SR$ and $\forall$ $y \in SK$. Namely,

x = Criticality $\rightarrow$ y = Sec & Rs
x = Connectivity $\rightarrow$ y = GC & LC & NC
x = Remote Location $\rightarrow$ y = Sec & Rs
x = Isolation $\rightarrow$ y = Sec & Rs
x = Maintenance $\rightarrow$ y = Ext
x = Growth $\rightarrow$ y = Sc
x = Performance $\rightarrow$ y = Cm

As for the importance of these properties, security at bootstrapping is important for this scenario but not essential, as the substation is a protected environment. Communication overhead is also important but not essential. All other properties are essential: all nodes must be able to open a secure channel with each other, the network must be extensible, and the resilience must be very high. Note that memory and energy are not important in this context since industrial nodes have enough memory to execute traditional KMS and store the security credentials used for bootstrapping. Speed is not an issue as most nodes are static, and dynamic nodes are worn by operators who do not move very quickly. In addition, as nodes are generally deployed in static areas, it is possible for them to depend on external power supplies. Precisely, this allows the use of 'deployment knowledge' (i.e. the nodes are configured at known points within network and substation).

Main and Secondary properties: Taking into account the previous discussion, the properties that must be used in SenseKey are the following:

Main properties = {Rs, GC, LC, NC, Ext, Sc}
Secondary properties = {Sec, Cm} and 'deployment knowledge'

SenseKey and Results: The fourth step is to fill in the form with the main and secondary properties. The results indicate that 7 possible protocols could be suitable for this particular scenario. In fact, one of the protocols is one of the standards developed for these kinds of environments, ZigBee Smart Energy 2.0. Nevertheless, SenseKey even provides other protocols that do not have some of the disadvantages of the standard, such as OSCAR.

Table 3: List of different WSN applications

Scenario Type		**Examples**	**Properties**
One-hop networks		Management of industrial machinery	LC
Simple networks	Simple	Office monitoring	GC, *LC*
	Medium	wine production industry	*Sc*, *GC*, *LC*
	Large	Wildfire detection	Sc, *GC*, *LC*, *Cm*
w. Mobile base station	Small	Vehicle tracking	GC, LC
w. Mobile and static nodes	Small	Monitoring of assisted-living residents	NC, GC, *Cm*, *LC*
	Medium	Hazards in safety-critical structures	NC, *Sc*, *GC*, *Cm*, *LC*
Short-lifetime networks	Small	Measuring noise pollution	LC, *Cm*, ~~En~~, ~~Ex~~

5 OBSERVATIONS FROM EXISTING KMS PROTOCOLS

After defining the method used to choose a particular KMS protocol, we focus our attention on carrying out an analysis the purpose of which is twofold. First, we check whether the current key management state-of-the-art research really covers all the range of possible applications. If that is not the case, secondly we have to know where the future research in the area should focus, with the aim of designing new KMS protocols that are suitable for advanced real-world applications.

Some of the applications are summarized in Table 3. In order to facilitate the assessment process, they are grouped according to the mobility of the nodes and the network size. For the applications, the last column of the table lists the main properties (roman type), secondary properties (*italic* type), or properties that should be ignored (in ~~strikethrough~~ text). It should be noted, and even stressed, that the relevance of other different properties depends on a particular application and its working scenario. This is the case of the security and network resilience properties. Their importance within a context is determined by the open-to-public nature of the environment and its associated risks. Equally, extensibility must also be taken into account if a network has a long lifetime or requires constant maintenance. In fact, there are currently applications that may require these three properties, such as the Supervisory Control and Data Acquisition (SCADA) Systems. Alcaraz et al. in [Alcaraz and Lopez(2012)] identified some security and operational requirements which correspond with the three properties mentioned; security, resilience and extensibility. Therefore it is highly recommendable to advance within this research area to design suitable secure protocols that can be matched to the restrictions of the context.

The analysis consisted of two phases. In the first phase, we analyzed the suitability of the protocols by considering only those properties indicated in Table 3. This phase of the analysis reveals that it is possible to find some link-layer secret keys for almost any type of static sensor network if only network size and node mobility are considered. For one-hop networks, simple-small networks, and short-lifetime networks, there is no need to use complex protocols that need key pools or complex negotiations; some simple schemes such as Blom Key pre-distribution and polynomial key pre-distribution scheme will suffice. For the case of simple-medium/large networks, and in order to take scalability into consideration, it is better to use, for instance, cluster-based protocols such as Panja or key pool protocols. It is important to point out that other simple negotiation schemes, such as key infection scheme, may also work well in a static scenario where security on initial deployment is not an issue.

The scenarios with mobile base stations do not pose any problem since a node may share a pre-installed pairwise key with those base stations. As a result, it is possible to use almost any suitable protocol for the static networks. On the other hand, in a network with mobile nodes, the number of protocols that can completely fulfil the requirements is limited. Nevertheless, the Blom and polynomial key pre-distribution schemes may work well for small networks, while other complex protocols (such as CARPY+ and OSCAR) and PKC-based protocols may work better for bigger networks. Due to the low speed of PKC-based protocols, they cannot be used in those scenarios where two mobile nodes are within their radio ranges for only a very short period of time.

In the second phase of the analysis we have taken into account the properties indicated in the table as well as all the other properties, including energy, extensibility, memory, resilience, security, and speed. The results are not so optimistic, as we will explain presently. One of the properties that needs careful consideration is extensibility. Almost all existing KMS protocols can work with only a few nodes after the initial deployment. However, some protocols may only work well with a limited number of nodes that can be added after the initial deployment. Therefore, the network designer may face difficulties in finding a suitable protocol if the WSN application needs to change the number of nodes or add new nodes from time to time.

A similar problem arises with network resilience. Many protocols are partially resilient. In fact, it is always possible to choose a KMS protocol that provides partial resiliency and partial extensibility. However, this means that, in the case they are subverted, the nodes working with these protocols will provide some information about the keys contained in other nodes. The reason is simple: most protocols have been developed based on the assumption that they can include more keys inside the node or embed certain mathematical information (e.g. a column/row of a matrix) in it to share with the other nodes, so that the information can be used to calculate a shared key. Note that a few protocols provide perfect resilience, but they have other disadvantages as well (e.g. high communication and speed overhead while adding a new node in the OSCAR protocol).

Speed is quite an interesting parameter. This property is essential in the networks where mobile nodes must establish a secure channel within a very short period of time (almost immediately) with the other peers. However, there are very few protocols that can ensure *speed* and *node connectivity* properties at the same time. Moreover, if extensibility is required, then only two useful protocols exist. The performance of the first one, which was proposed based on generalized quadrangles, depends highly on the knowledge of the maximum number of sensor nodes in the network. As for the second one, OSCAR, we have already mentioned that this protocol has a high overhead when adding a new node, thus it is only useful for this scenario in the case all nodes belong to the initial deployment. Note that standards like ZigBee can fulfill these needs, but their overall resilience is not very good [Alcaraz and Lopez(2010)].

With regard to the *security* property in the network bootstrapping, if the security of a network during its initial deployment is not important, it can possibly use some negotiation-based protocols such as the Key Infection scheme. However, if the extensibility or even the resilience (in some cases) of the network is important (e.g. in a SCADA system), then the choice can be different. As for the *energy* property, only a few protocols spend a lot of energy negotiating the pairwise keys (e.g. the Key Infection). Finally, most of the existing KMS protocols were proposed and optimized to consume the least amount of memory as possible, and thus memory is not a big issue.

Lastly, it is worth commenting here that the properties in Table 3 are not necessarily limited to the properties described throughout this paper. The construction of the Table and the SenseKey tool have been designed in such a way that they can accommodate further new properties so as to provide a sustainable selection tool and be open to new possibilities.

5.1 Discussions and Open Issues

As mentioned earlier, while the actual state-of-the-art research in KMS protocols for sensor networks is good enough for its use in certain applications, there are some issues that still remain to be solved. One of the open issues is to develop a KMS protocol with superior extensibility and resilience properties. In fact, many KMS protocols have been designed without paying enough attention to the needs of adding new sensor nodes to the WSN after its deployment. It is important to point out here that standards like ZigBee, WirelessHART and ISA100.11a have fulfilled this particular property, although they have some disadvantages as well. Indeed, though these three standards were intentionally designed for critical contexts (i.e. control and industrial systems) to ensure coexistence, energy, reliability and security, an important number of weaknesses have also been identified and analysed here. Looking at Table 3, it is possible to note that the three standards lack of potential to offer a complete communication and resilience when the pairwise keys are being negotiated. Also, the actual scalability of these protocols is highly dependant on their particular modes of operation (e.g. some ZigBee modes involve the gateway in the key negotiation process).

Continuing with the resilience, most KMS protocols work based on node redundancy (including extra security credentials) in order to allow the nodes to find and share pairwise keys with their neighbourhoods. While there is a need to design new solutions not assuming node redundancy, it is not clear how this can be done. One possible approach is to make use of authenticator tokens like the ones used in OSCAR, which ensure that a node stored certain master keys in the past.

The design of KMS protocols that are particularly suitable for certain scenarios is another open issue. One scenario has been identified in the previous Section: a network with mobile nodes that move at high speed. We can even add an additional scenario: an extensible network with almost no communication overhead, or in other words, the communication overhead is a main property (e.g. underwater sensor networks). Only Public Key Cryptography (e.g. using Identity-Base cryptography [Galindo et al.(2012)Galindo, Roman, and Lopez]) and some novel protocols might cover the needs of this particular scenario.

We can also discuss briefly why some simple protocols, such as polynomial key pre-distribution and Blom key pre-distribution (without key-pool optimizations), can be recommended for most WSN applications. The reason is simple, as most of current WSNs are used in small or medium-size applications, ranging from 50 to 1000 nodes. For this size, the simpler protocols are good enough. Still, the security of these primitives used by the mathematical framework protocols have not been rigorously proven yet. If these primitives were to be found insecure, the number of WSN applications demanding more suitable KMS protocols would increase.

Finally, we would like to address the issues related to the use of public key cryptography. In most WSN applications it is possible to use Elliptic Curve Cryptography for setting up and maintaining the keys for the sensors. However, PKC is still expensive and slow in comparison with other symmetric-based key management protocols, requiring each node to start a negotiation in order to obtain their pairwise keys. As a result, it is clear that for a WSN network with a low complexity other simpler protocols based on symmetric cryptography can be used. Nonetheless, for most networks, PKC can offer the necessary security properties if the overall security requirements of the networks are high and the devices' resources can support it.

6 Conclusions

While key management in WSNs has been exhaustively studied, there are still some open issues that require more research. This paper has shown that the existing protocols are able to satisfy some of the needs of existing sensor network applications. Also, the paper has described the properties characterizing a KMS protocol and we made an effort to map those properties to the requirements of WSN applications. This has led to our proposed KMS selection method, which can be used by network designers to decide which protocols are suitable for protecting their networks. We have developed a simple and easy to use a web interface based on this methodology, known as the SenseKey tool. Note that our method can be used in conjunction with other methods (cf. [Lee et al.(2007)Lee, Kim, Lee, and Lim, Na et al.(2011)Na, Ren, Hori, and Sakurai]) so as to provide network designers with a better overview on the suitability of specific protocols.

Though our work has focused on key management schemes that establish link-layer secret keys among neighbors in sensor networks with homogeneous nodes, it would be very interesting to perform the same analysis with other protocols that can establish other types of keys (e.g. group keys and end-to-end keys) in both homogeneous and heterogeneous (e.g. hierarchical) sensor networks. In fact, these other protocols may have other specific properties, such as self-healing (i.e. recovery of a group key). We are currently working in this direction.

References

[Roman et al.(2011)Roman, Lopez, Alcaraz, and Chen] R. Roman, J. Lopez, C. Alcaraz, H. Chen, SenseKey - Simplifying the Selection of Key Management Schemes for Sensor Networks, in: 5th International Symposium on Security and Multimodality in Pervasive Environments (SMPE 2011), Singapore, 789–794, 2011.

[Zhou et al.(2008)Zhou, Fang, and Zhang] Y. Zhou, Y. Fang, Y. Zhang, Securing Wireless Sensor Networks: a Survey, IEEE Communications Surveys & Tutorials 10 (3) (2008) 6–28.

[Zhu et al.(2011)Zhu, Xiang, Zhou, Deng, and Bao] W. Zhu, Y. Xiang, J. Zhou, R. Deng, F. Bao, Secure Localization with Attack Detection in Wireless Sensor Networks, International Journal of Information Security 10 (2011) 155–171.

[Camtepe and Yener(2008)] S. Camtepe, B. Yener, Key Management in Wireless Sensor Networks, in: J. Lopez, J. Zhou (Eds.), Wireless Sensor Network Security, IOS Press, 2008.

[Eschenauer and Gligor(2002)] L. Eschenauer, V. Gligor, A Key-Management Scheme for Distributed Sensor Networks, in: 9th ACM Conference on Computer and Mommunications Security (CCS), Washington, DC, USA, 41–47, 2002.

[Du et al.(2005)Du, Deng, Han, Varshney, Katz, and Khalili] W. Du, J. Deng, Y. Han, P. Varshney, J. Katz, A. Khalili, A Pairwise Key Predistribution Scheme for Wireless Sensor Networks, ACM Transactions on Information and System Security (TISSEC) 8 (2) (2005) 228–258.

[Camtepe and Yener(2007)] S. Camtepe, B. Yener, Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks, IEEE/ACM Transactions on Networking 15 (2) (2007) 346–358.

[Liu and Ning(2003)] D. Liu, P. Ning, Establishing Pairwise Keys in Distributed Sensor Networks, in: 10th ACM Conference on Computer and Communications Security (CCS), Washington D.C., USA, 52–61, 2003.

[Anderson et al.(2004)Anderson, Haowen, and Perrig] R. Anderson, C. Haowen, A. Perrig, Key Infection: Smart Trust for Smart Dust, in: 12th IEEE International Conference on Network Protocols (ICNP), Berlin, Germany, 206–215, 2004.

[Seshadri et al.(2008)Seshadri, Luk, and Perrig] A. Seshadri, M. Luk, A. Perrig, SAKE: Software Attestation for Key Establishment in Sensor Networks, in: Distributed Computing in Sensor Systems, vol. 5067 of *LCNS*, Springer Berlin / Heidelberg, 372–385, 2008.

[Panja et al.(2006)Panja, Madria, and Bhargava] B. Panja, S. Madria, B. Bhargava, Energy and Communication Efficient Group Key Management Protocol for Hierarchical Sensor Networks, in: IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, vol. 1, Taichung, Taiwan, 2006.

[ZigBee Alliance(2008)] ZigBee Alliance, ZigBee-08006r03: ZigBee-2007 Layer PICS and Stack Profiles (ZigBee-PRO), Revision 3, `http://www.zigbee.org/`, Retrieved on December 2011, 2008.

[HART Communication Foundation(2009)] HART Communication Foundation, WirelessHART™ Technology, `http://wirelesshart.hartcomm.org/`, Retrieved on December 2011, 1993–2009.

[ISA100(2011)] ISA100, ISA100.11.a-2009: Wireless Systems for Industrial Automation - Process Control and Related Applications, `http://www.isa.org/isa100`, Retrieved on December 2011, 2009–2011.

[Liu and Ning(2008)] A. Liu, P. Ning, TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks, in: International Conference on Information Processing in Sensor Networks (IPSN), St. Louis, USA, 245–256, 2008.

[Lee and Stinson(2008)] J. Lee, D. Stinson, On the Construction of Practical Key Predistribution Schemes for Distributed Sensor Networks Using Combinatorial Designs, ACM Transactions on Information and System Security (TISSEC) 11 (1) (2008) 1–35.

[H. Chan(2003)] D. S. H. Chan, A. Perrig, Random Key Predistribution Schemes for Sensor Networks, in: 2003 IEEE Symposium on Security and Privacy, IEEE, Oakland, USA, 197–213, 2003.

[Zhang et al.(2007)Zhang, Tran, Zhu, and Cao] W. Zhang, M. Tran, S. Zhu, G. Cao, A Random Perturbation-based Scheme for Pairwise Key Establishment in Sensor Networks, in: 8th ACM International Symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc), Montreal, Canada, 90–99, 2007.

[Wen et al.(2009)Wen, Zheng, Ye, Chen, and Qiu] M. Wen, Y. Zheng, W. Ye, K. Chen, W. Qiu, A Key Management Protocol with Robust Continuity for Sensor Networks, Computer Standards and Interfaces 31 (4) (2009) 642–647.

[Lee et al.(2007)Lee, Kim, Lee, and Lim] H. Lee, Y. H. Kim, D. H. Lee, J. Lim, Classification of Key Management Schemes for Wireless Sensor Networks, in: Advances in Web and Network Technologies, and Information Management Workshop: ASWAN, Huang Shan, China, 664–673, 2007.

[Na et al.(2011)Na, Ren, Hori, and Sakurai] R. Na, Y. Ren, Y. Hori, K. Sakurai, A Generic Evaluation Method for Key Management Schemes in Wireless Sensor Network, in: 5th International Conference on Ubiquitous Information Management and Communication (ICUIMC), Seoul, Korea, 2011.

[Jaworski et al.(2009)Jaworski, Renand, and Rybarczyk] J. Jaworski, M. Renand, K. Rybarczyk, Random Key Predistribution for Wireless Sensor Networks using Deployment Knowledge, Computing 85 (1-2) (2009) 57–76.

[Chan and Perrig(2005)] H. Chan, A. Perrig, PIKE: Peer Intermediaries for Key Establishment in Sensor Networks, in: 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), vol. 1, Miami, USA, 524–535, 2005.

[Zhu et al.(2006)Zhu, Setia, and Jajodia] S. Zhu, S. Setia, S. Jajodia, LEAP+: Efficient Security Mechanisms for Large-scale Distributed Sensor Networks, ACM Transactions on Sensor Networks 2 (4) (2006) 500–528.

[Yu et al.(2010)Yu, Lu, and Kuo] C.-M. Yu, C.-S. Lu, S.-Y. Kuo, Noninteractive Pairwise Key Establishment for Sensor Networks, IEEE Transactions on Information Forensics and Security 5 (3) (2010) 556–569.

[Delgado-Mohatar et al.(2011)Delgado-Mohatar, Fuster-Sabater, and Sierra] O. Delgado-Mohatar, A. Fuster-Sabater, J. Sierra, A Light-weight Authentication Scheme for Wireless Sensor Networks, Ad Hoc Networks 9 (5) (2011) 727–735.

[Zhang et al.(2011)Zhang, He, and Wei] X. Zhang, J. He, Q. Wei, EDDK: Energy-Efficient Distributed Deterministic Key Management for Wireless Sensor Networks, EURASIP J. Wireless Comm. and Networking 2011.

[Huang et al.(2011)Huang, Liao, and Tang] J.-Y. Huang, I.-E. Liao, H.-W. Tang, A Forward Authentication Key Management Scheme for Heterogeneous Sensor Networks, EURASIP J. Wireless Comm. and Networking 2011.

[Garrison(2011)] M. Garrison, ZigBee-095449: ZigBee Smart Energy Profile 2.0 Technical Requirements Document, ZigBee Alliance and HomePlug Powerline Alliance liaison, `http://www.zigbee.org/`, Retrieved on December 2011, 2011.

[Alcaraz and Lopez(2012)] C. Alcaraz, J. Lopez, Analysis of Requirements for Critical Control Systems, in: Sixth IFIP WG 11.10 International Conference on Critical Infrastructure Protection, Washington DC, USA, 2012.

[Alcaraz and Lopez(2010)] C. Alcaraz, J. Lopez, A Security Analysis for Wireless Sensor Mesh Networks in Highly Critical Systems, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews 40 (4) (2010) 419–428.

[Galindo et al.(2012)Galindo, Roman, and Lopez] D. Galindo, R. Roman, J. Lopez, On the Energy Cost of Authenticated Key Agreement in Wireless Sensor Networks, Wireless Communications and Mobile Computing 12 (1) (2012) 133–143.