

## WSAN4CIP

### Deliverable 4.1

#### Concepts for providing dependable and secure elementary services in WSANs

Editor:	Alban Hessler, NEC
Deliverable nature:	Report (R)
Dissemination level: (Confidentiality)	Public (PU)
Contractual delivery date:	31 <sup>st</sup> March 2010
Actual delivery date:	14 <sup>th</sup> April 2010
Suggested readers:	Research Community
Version:	1.0
Total number of pages:	51
Keywords:	WSAN, middleware, QoS, energy management, dependability, security

---

#### *Abstract*

Deliverable D4.1 presents the conceptual work achieved within the work package Service Protection and summarizes its first results. We show how various modules can help to increase the dependability of the system at different layers. More specifically, we first show how to build elementary security services based on the uniqueness of the clock skews of nodes, hence providing security services such as fingerprinting of nodes. Then, we investigate distributed data storages based on erasure coding, review the related state of the art and also present a mechanism to detect pollution attacks in such distributed storages without the use of any cryptography. Also, power saving techniques for wireless sensor nodes are studied, in order to build a QoS and energy manager middleware, a cross-layer component that will regulate the QoS state of nodes across the network in an autonomous fashion. Finally, deliverable D4.1 also outlines the design rationales for the interface between WSANs and a CIP application. It provides the basis for specification and implementation of the dependable services and middleware architecture.

---

---

**Disclaimer**

---

This document contains material, which is the copyright of certain WSAN4CIP consortium parties, and may not be reproduced or copied without permission.

All WSAN4CIP consortium parties have agreed to full publication of this document.

The commercial use of any information contained in this document may require a license from the proprietor of that information.

Neither the WSAN4CIP consortium as a whole, nor a certain party of the WSAN4CIP consortium warrant that the information contained in this document is capable of use, or that use of the information is free from risk, and accept no liability for loss or damage suffered by any person using this information.

**Impressum**

[Full project title] Wireless Sensor Networks for the Protection of Critical Infrastructures

[Short project title] WSAN4CIP

[Number and title of work-package] WP4 Service Protection

[Document title] Concepts for providing dependable and secure elementary services in WSANs

[Editor: Name, company] Alban Hessler, NEC Europe Ltd.

[Work-package leader: Name, company] Alban Hessler, NEC Europe Ltd.

[Estimation of PM spent on the Deliverable] 20 PM

**Copyright notice**

© 2010 Participants in project WSAN4CIP

## **Executive summary**

Deliverable D4.1 presents the conceptual work achieved within the work package Service Protection and summarizes its first results. We show how various modules can help to increase the dependability of the system at different layers. More specifically, we first show how to build elementary security services based on the uniqueness of the clock skews of nodes, hence providing security services such as fingerprinting of nodes. Then, we investigate distributed data storages based on erasure coding, review the related state of the art and also present a mechanism to detect pollution attacks in such distributed storages without the use of any cryptography. Also, power saving techniques for wireless sensor nodes are studied, in order to build a QoS and energy manager middleware, a cross-layer component that will regulate the QoS state of nodes across the network in an autonomous fashion.

Finally, deliverable D4.1 also outlines the design rationales for the interface between WSAWs and a CIP application. It provides the basis for specification and implementation of the dependable services and middleware architecture.

## List of authors

Company	Author
BUTE	Czap László
IHP	Krzysztof Piotrowski
INOV	José Gonçalves, Antonio Grilo, Augusto Casaca
INRIA	Claude Castelluccia
NEC	Alban Hessler, Dirk Westhoff
TEC	José Luis Serrano Martín
UMA	Daniel Garrido, Manuel Diaz

## Table of Contents

Executive summary .....	3
List of authors.....	4
Table of Contents .....	5
List of Figures.....	6
Abbreviations .....	7
1 Introduction.....	8
2 Dependable and Secure WSAN Services for CIP.....	9
2.1 WSAN CIP Service Architecture.....	9
2.2 Service Protection: A Trade-Off.....	11
3 Service Protection .....	14
3.1 Elementary services .....	14
3.1.1 Toward Clock Skew Based Services .....	14
3.1.2 Defending against Wormhole attacks by Securing Neighbour Discovery in WSNs.....	16
3.2 Distributed Data Storage (DDS) .....	17
3.2.1 Related work on data persistency in coding based DDS .....	18
3.2.2 Related work on data management in DDS.....	24
3.2.3 Related work on support tools for DDS.....	24
3.2.4 Related work on data integrity for coding based DDS .....	25
3.2.5 Our Contribution in coding based DDS.....	26
3.2.6 A non-cryptographic Approach to Providing Integrity in Coding based DDS.....	26
3.3 QoS and Energy middleware manager.....	29
3.3.1 Power saving mechanisms .....	30
3.4 Services applied to CIP .....	31
4 WSAN Service Integration to CI .....	35
4.1 SCADA System Components and Architecture .....	35
4.2 SCADA Communications Protocols Overview.....	36
4.2.1 DNP3 (Distributed Network Protocol) .....	36
4.2.2 ICCP (Inter-Control Center Communications Protocol) .....	37
4.2.3 MODBUS .....	38
4.2.4 OPC (OLE for Process Control) .....	39
4.3 Challenges and Issues Concerning SCADA-WSAN Systems.....	40
4.4 Middleware Issues Concerning SCADA-WSAN Systems .....	42
4.5 SCADA-WSAN Connection Approaches.....	43
4.5.1 Gateway solution .....	43
4.5.2 IP solution.....	44
4.5.3 Comparison Between the Two Proposals .....	45
5 Conclusion.....	46
References .....	47

## List of Figures

Figure 1: CI management system .....	9
Figure 2: Generic node service architecture.....	1
Figure 3 Coarse Granular Trade-Off for a WSN service.....	1
Figure 4: Skew difference between motes. Nodes 8 and 9 are TelosB. Others are MICAz.....	15
Figure 5 : Single-hop WSN Setup with 7 MICAz motes. Node 1 is the sink node connected with computer through MIB510 serial interface. Nodes 2-7 are acting as end nodes and sending packets to sink over the wireless channel. ....	15
Figure 6: First approach: encoding at the source node and distributed storage.....	1
Figure 7: Second approach: Distributed coding and storage system model.....	20
Figure 8: Illustration of the s.l.e. representing the storage system .....	1
Figure 9: Effects of the pollution attack .....	1
Figure 10: Classification of power conservation mechanisms .....	30
Figure 11: Distributed data storage to pursue service despite jamming.....	33
Figure 12 DNP3 Protocol Stack .....	37
Figure 13: ICCP server protocol stack .....	38
Figure 14: MODBUS layers.....	39
Figure 15: Common OPC client-server architecture .....	39
Figure 16: Gateway solution .....	1
Figure 17: IP Solution .....	1

## Abbreviations

API	Application Programming Interface
ATIM	Announcement Traffic Indication Message
CIP	Critical Infrastructure Protection
CPU	Central Processing Unit
DDS	Distributed Data Storage
DPM	Dynamic Power Management
DVS	Dynamic Voltage Scaling
GW	Gateway
ICCP	Inter-Control Center Communications Protocol
IP	Internet Protocol
LEMMA	Latency-Energy Minimization Medium Access
LT	Luby Transform
MAC	Medium Access Control
NAV	Network Allocation Vector
OLE	Object Linking and Embedding
OPC	OLE for Process Control
PLC	Programmable Logic Controller
QoS	Quality of Service
RS	Reed-Salomon
SCADA	Supervisory Control And Data Acquisition
SLE	System of Linear Equations
WSAN	Wireless Sensor and Actuator Network

# 1 Introduction

Deployed WSANs face a fluctuation of unfortunate factors, such as hardware failures or poor wireless medium, which directly impact their capability to deliver the required service. The challenge at hand is to build applications that despite those failures provide *some* service rather than halting completely. This is the property of *failing gracefully*. Ideally, the performance of the service should decline in relation to the severity of the failure, and restore to full performance as soon as the disruption ends. How can we literally build a service that bends but not breaks? The objective of the Service Protection work package is to provide this enhanced dependability property to the WSAN services that are going to be deployed.

The deliverable D4.1 describes the first results of conceptual work of Tasks 4.1, 4.2, 4.3, and 4.4. It serves as the basis for the implementation and the design of and analysis of the services for the application scenario. With this deliverable, also the design rationales for the interface between WSANs and a CIP application are outlined. It provides the basis for specification and implementation of the dependable services and middleware architecture, namely elementary security services, fault tolerant middleware and intelligent energy and QoS management.

Chapter 2 introduces briefly WSAN services for CI, showing its architecture, and how we can assess modules developed in the project.

Chapter 3 then explores the different components that will be deployed on the nodes. For each of them, a state of the art is presented, as well as a sketch of the project contribution. The descriptions of the final solutions will have their place in future deliverables.

Chapter 4 is dealing with the WSAN service integration to CI. After a brief introduction this chapter is dealing with SCADA system components and architecture and SCADA communication protocols. For the latter protocols DNP3, IEC 60870-5-101, MODBUS and OPC are described. Chapter 4 furthermore provides insights into challenges and issues concerning SCADA and WSAN systems, middleware issues and connection of the WSAN to the SCADA systems.

## 2 Dependable and Secure WSAN Services for CIP

With the recent development on embedded systems, batteries and radio technologies, there is now the opportunity to apply wireless sensor and actuator technologies to a myriad of applications. However, despite a clear benefit of cost-efficiency, particularly emphasized by a low cost of installation, WSANs face higher challenges regarding reliability with limited battery autonomy and noisy wireless issues, as well with security issues, as the wireless channel is accessible to an attacker at virtually no cost. In order to facilitate the market penetration of WSANs, WSAN4CIP develops middleware components, which will ease the development of dependable services for CIP. Furthermore, a code generation tool that takes into account the CIP application requirement is also developed, in order to support the developer in its task of writing dependable software for sensor nodes.

To deal with the unreliable nature of the wireless channel, sensor networks can form an ad-hoc network, where routes can be re-configured by the WSAN itself, creating a highly dynamical network structure. Oppositely, wired sensors generally do not have any redundancy route (except eventually for the backbone link). Consequently, the intrinsic properties of a mesh network such a WSAN is unknown to the way SCADA systems used to represent the current status of the network: usually, sensors were reporting via an established route at setup, this for the lifetime of the sensor. We address this issue by integrating WSANs into existing SCADA systems, extending the interfaces to support the management of WSANs, by for example letting the SCADA operator query the current network topology, or the battery status of a node. Furthermore, the operator can influence the network by sending commands to the network.

### 2.1 WSAN CIP Service Architecture

We introduce first the architecture of WSAN4CIP, and show how services play a vital role to provide a dependable framework for WSAN applications.

Figure 1 represents the CI management system, with the building blocks that compose it. It fairly represents the conceptual work that we address: integration tot the CI management with a SCADA interface, development of distributed data storage, development of a QoS and energy manager, that allows the network to adapt parameters dynamically, and finally elementary security services, in order to have solid foundations for other service that rely on location or time.

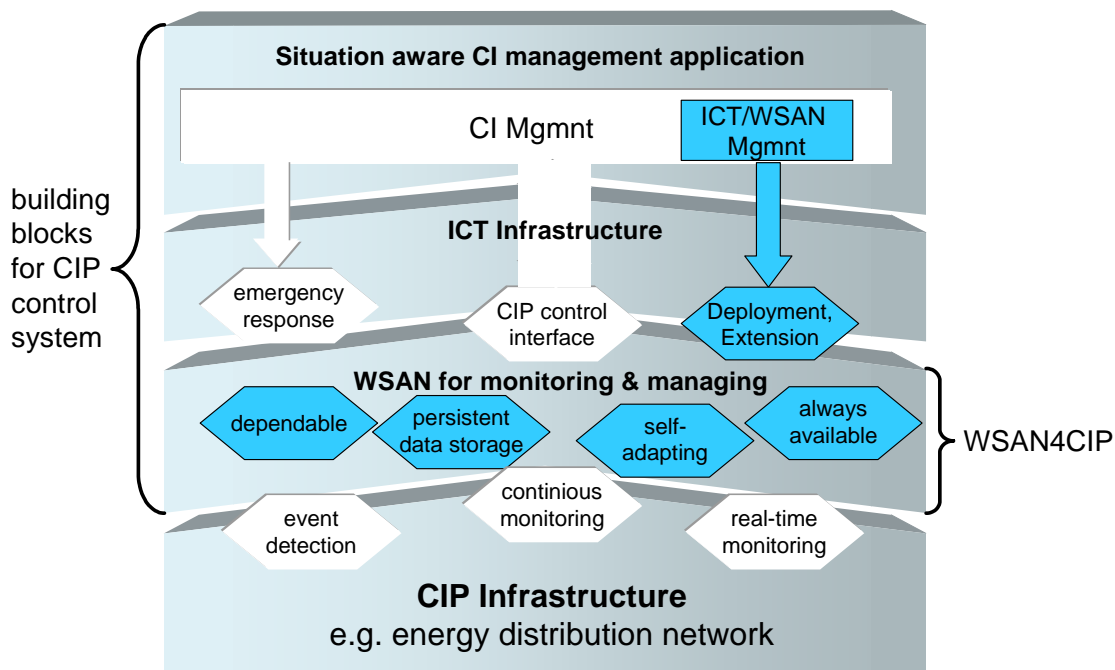
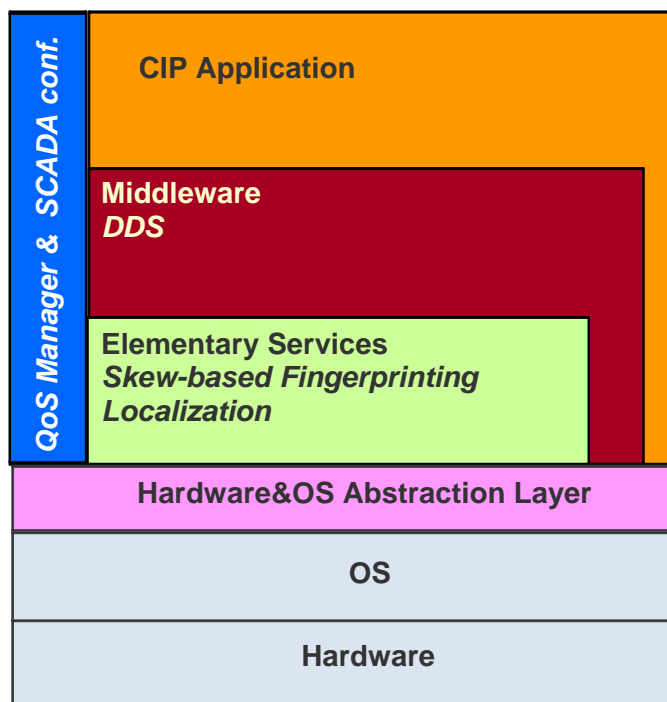


Figure 1: CI management system

From Figure 1, we can also draw the dependencies that act on the WSAN services, namely the sensing and actuating applications such as real-time monitoring or event detection, but also the integration to the CI management, which expects the services to be available all the time.

In this Deliverable, we are principally developing different types of middleware abstraction. First, we build robust elementary services that provide the necessary building blocks for most of applications, such as secure localization or unique fingerprinting of nodes.

On top of it, we develop more complex middleware, e.g. a dependant distributed storage. They are in nature more complex, and provide applications with rich features. Finally, we have a cross-layer middleware that serves for the management of the WSAN. The QoS and energy manager that self-regulates the sensor network to perform quasi optimally, whatever the conditions of the network are, e.g. attacks, poor weather conditions affection the radio reception, etc. There is also a middleware to provide a secure interaction with the SCADA supervisory system. The overall structure is depicted in Figure 2.

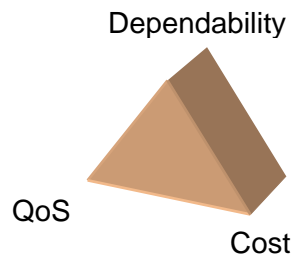


**Figure 2: Generic node service architecture.**

Instantiations of components developed in WP4 are outlined in *italic* in Figure 2. We can observe the clear separations that are both expressed in the architecture, and the tasks of the work package.

## 2.2 Service Protection: A Trade-Off

We outline in this section the important factors that affect a WSAN service in a CIP environment. We categorize three general properties that can be traded off in a WSAN application as shown in Figure 3:



**Figure 3 Coarse Granular Trade-Off for a WSAN service.**

- *Quality of service*: This is a broad term to designate what the service can provide in various performance criteria, such as network latency and throughput, or the precision of the service.
- *Dependability*: This reflects the ability of the service to sustain a range of faults and attacks.
- *Cost*: It covers the costs that are generated by a configuration of the service. The cost covers many aspects such as number of nodes required, hardware requirements, or network lifetime. This is an important factor, as generally decreased cost is the main motivation for an utility to use wireless sensors instead of wired one. Therefore, developed solutions should maintain the cost of the system low.

Generic example of trade-off:

- Increase the number of nodes: this decreases the cost efficiency as each additional node has a cost; it increases the dependability by improving the redundancy and may increase the quality of service (e.g. richer density of measurements).
- Adding public cryptography hardware support: it increases the dependability by allowing node to use efficiently public cryptography to verify signatures and thus increasing the general security level. But it also decreases the QoS as the security payload may impact the effective throughput of the network. It also obviously increases the cost of the solution.

It is of course not possible to have high quality of service and high dependability without a high cost, it is however possible: 1) to develop protocols and algorithms that provide quasi-optimum dependability and QoS for a fixed cost, and 2) to carefully assess the application requirements in order to provide the WSAN with a service with scaled dependability, QoS and cost.

### *Dependability Assessment*

To perform a dependability assessment, it is first important to understand what adversary we are facing. Over-estimating or under-estimating the threat can lead to poor system design. For CIP environments, we performed a threat analysis in WP1, with different adversaries. Table 1 summarizes the result of this analysis, listing different types of attacks with their associated risk. A risk comprehends the impact on the system and the likelihood of that threat to realize.

**Table 1: Summary of the risk analysis.**

Application type > Adversary model >	Control loop			Surveillance		
	Poor	Clever	Rich	Poor	Clever	Rich
<b>Attack type</b>						
Physical destruction of nodes	M/L	M/L	H/M	M/L	H/M	H
Dismounting and stealing nodes	M/L	H/L	H	L	M/L	H
Dismounting and relocating sensors	L	L	M	M/L	M/L	H
Sensor input manipulation	L	L	M	M	H	H
Jamming	H	H	H	M	H	H
Eavesdropping	M	H	H	L	M	H
Replay of protocol messages	L	H	H	L	H	H
Injection of crafted protocol messages	L	H	H	L	H	H
Corruption of stored data	n/a (*)	n/a (*)	n/a (*)	L	H/L	H
Remote code injection	L	M	M	L	M	M
Installing rogue software on nodes	L	M	H	L	M	H
Deployment of rogue nodes	L	H	H	L	H	H

L=low, M=medium, H=high

(\*) We believe that there could even be data storage in control loop applications. For example, in an on-going jamming attack, data could be stored in the network if it cannot be transmitted to the gateway.

#### QoS Assessment

Finally, metrics need to be defined for assessing the QoS. This is for every function and service different, and thus renders the evaluation of complex services a difficult task. In WP1 of WSAN4CIP, the problem of evaluating the QoS of routing, or of composed layers is a task on its own. There are so many dependencies at a middleware layer that we can only afford to make an informal evaluation of the QoS of the middleware layer. At best, one can evaluate the middleware on very specific conditions, or by simplifying the model.

In WSAN4CIP, we develop services to ensure that those requirements are met. Although, it is important to note that the middleware that we develop do not meet the requirements directly, but helps the application to meet them.

In the elaboration of the different scenarios, a set of requirements were set. As an example, we refer to Table 2. For a specific use case, if all the requirements are met, we can deduce that an excellent QoS is reached, if not, we can consider the QoS as degraded, which is a continuous function until the service is halted. The degradation of QoS has many dimensions, but we will show on a component level of certain QoS properties can degrade gracefully in presence of an adversary or a fault.

**Table 2: Example of requirements from a use case: video surveillance of a substation.**

Class	Parameters	Value
Security	Confidentiality	1
	Integrity	1
	Authentication	1
	Non-repudiation	1
Time	Throughput	768Kbps peak (per camera)
	Jitter	200 ms
	Delay	10 s
	Fairness index	Alarm data is more important
Energy	Energy consumption index	200 mW
	Energy consumption pattern	10 W peak tolerated
Reliability	Packet loss	Data: 90% within 10 seconds Video: 95% within 10 seconds

	Information integrity	Data: 100% Video: 98% per frame
	Degree of autonomy	5 years
	Node density	Power Station area: 5-15 m <sup>2</sup> (1 sensor node is enough) Note: Data transmission shall be done using the network implemented in Scenario 5.

### *Cost Assessment*

The cost assessment of the service is generally derived from the requirements needed for the functioning of the service. That can be computation, memory or energy cost. It can be the reliance on special hardware such as tamper-proof resistant units.

For each service, we might not evaluate each individual cost, but the most prominent ones. Some network costs can be hard to estimate: it comprises a lot of factors, and especially for security services, costs and benefits are hard to evaluate, as they generally depend on a risk analysis.

All in all, our aim is not to rely on expensive hardware and to provide compact and efficient software functions that run on the platform. With this basic design goal, we ensure that the costs of the services we are to deploy will remain low, and therefore beneficial for our end-users, the infrastructure operators.

## 3 Service Protection

In this section, we report on the different work items that WSAN4CIP is addressing to protect CIP services. It relates to the tasks 4.1, 4.2 and 4.3 of the work package on Service Protection, and we show how the output of these tasks can improve the dependability of a CIP application.

We first present elementary services, which provide basic blocks for security. Then we investigate distributed data storage and a QoS manager, which both increase the reliability of the network by increasing data persistency, and service reliability. Finally, we show how those services address the different attacks that a CIP WSN may be subjected to.

### 3.1 Elementary services

One precondition for offering services that are concerned with locality is the ability that the nodes are able to determine each other's locations. This will usually be done in the bootstrapping phase. In this project, we are considering attacks on localization protocols in wireless sensor networks. More specifically,

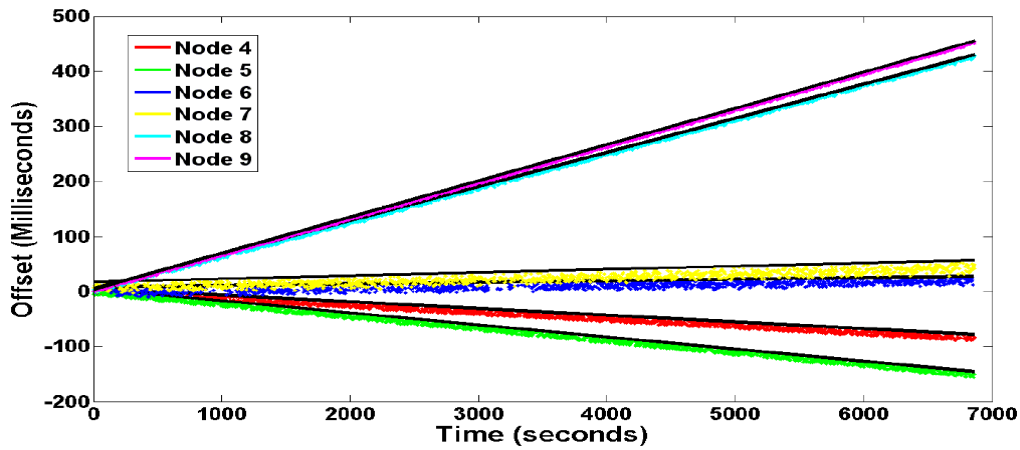
1. We consider the problem of the detection of such attacks. We show that a node clock skew can be used to monitor the localization of a node [84]. Clock drift changes with change of temperature. Clock drift seems to be a function of temperature. Hence, the temperature can be estimated by monitoring the clock drift. This property can be used for estimation of geo-location of sensor nodes. The longitude of a place can be estimated by measuring daily peak temperatures to establish local time. While the latitude can be estimated by measuring change of day lengths over a reasonably long period of time. This process of determining geo-location requires the sink node to be kept in conditioned (fixed) temperature, which seems feasible to achieve for a single sink node in WSN.
2. We study solutions against wormhole attacks and show that most of them are vulnerable and can be defeated. We study the impact of sending false lists of neighbours on the precision of detection of the wormhole attack for three chosen approaches. Attackers send false lists when nodes execute neighbour discovery. We define a new attack, called "parallel wormhole", where wormhole extremities and their neighbours send false lists of neighbours. We evaluated the impact of this introduced attack on the probability of detection of the wormhole attack. Simulation results proved that the three implemented approaches are influenced by the introduction of the new attack. They become unable.

#### 3.1.1 Toward Clock Skew Based Services

Clock skew has been used for remote fingerprinting of hosts on computer networks [85]. Variation of clock skew with respect to temperature has been used for revealing hidden services on Tor network [86]. So far so little has been known about clock skew on wireless sensor nodes and the effect that has temperature on the clock skew of sensor nodes.

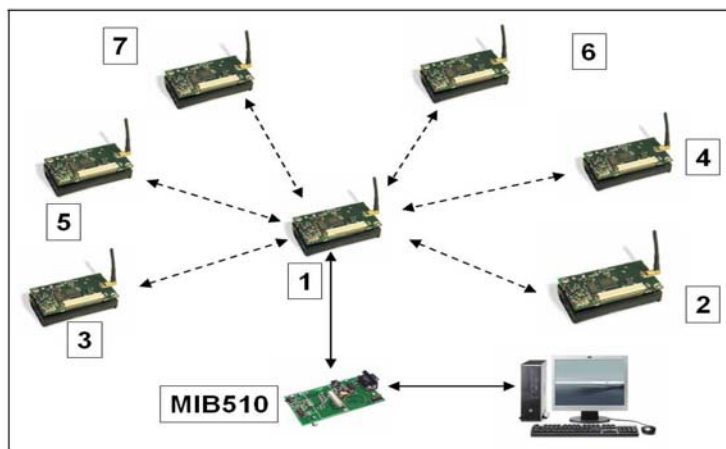
In WSAN4CIP, we did experiments to find out clock skew of different wireless sensor nodes with respect to a sink node and found that different motes have different clock skews. Clock skew varies a little from mote to mote and can be quantified with the unit 'parts per million' (ppm). This identity of motes based on different clock skew can be used for fingerprinting of nodes in WSN. Figure 4 exhibits the skew of different motes. One also observes that two different hardware platforms, telosB and MICAz, can be distinguished thanks to the skew.

Clock skew also varies with the variation of temperature [87]. The variation of clock skew in wireless sensor motes does not cause any problem of clock skew based mote fingerprinting, however. Variation of clock skew (a.k.a. clock drift) increases with the increase of temperature and decreases with the decrease of temperature. So, the clock drift is a function of the temperature. Thus estimating the variation of clock skew, we can estimate the current temperature of the end sensor nodes. Temperature data, sent from sensor motes having on board temperature sensor, can be verified whether those motes are malfunctioning or lying based on the comparison of estimated temperature from variation of clock skew and received temperature data. Estimated temperature data, over the day and for long period, will be useful to estimate the geo-locations of the nodes in WSN.



**Figure 4: Skew difference between motes. Nodes 8 and 9 are TelosB. Others are MICAz.**

The nodes of at the end sensor network will send packets with a timestamp to the sink node. The sink node will forward those packets to the measuring machine. The measuring machine will measure the clock skew of the nodes with respect to the clock of the sink node, which is used as the reference clock. The setup is depicted in . The sink node can also store the packets in its local storage and transmit them later to the measuring machine. Thus, the measuring machine can be offline or thousands of miles away from the sink node. The sink node can also be connected with the measuring machine through a combination of cellular, satellite, Internet and other types of wired and/or wireless networks for monitoring clock skew from an office computer. The measuring machine only needs the packets from the sink node having timestamp of end nodes and sink node.



**Figure 5 : Single-hop WSN Setup with 7 MICAz motes. Node 1 is the sink node connected with computer through MIB510 serial interface. Nodes 2-7 are acting as end nodes and sending packets to sink over the wireless channel.**

**Our Contributions:** We verified that clock skew based fingerprinting of sensor nodes can be successfully done on WSN. Clock skew based mote identification can be utilized for detecting Wormhole and Sybil attacks. Effect of temperature on the clock skew of sensor nodes can also be analyzed, which can be utilized for malicious or malfunctioning mote identification, estimation of temperature and geo-localisation of sensor nodes etc. in WSNs. Using 7 MICAz and 2 TelosB motes, we created single-hop and multiple-hop WSNs. We did experiments for both clock skew based fingerprinting of motes and analyzing effect of temperature on sensor motes by varying the temperature on wireless sensor motes. We implemented the synchronized timestamp sending mechanism as [88] on WSN and implemented the linear programming based skew detection mechanism as [89] using linear time algorithm of [90][91]. We showed that remote sensor mote fingerprinting can be done as remote physical device fingerprinting shown by [85]. We did experiment to verify the variation of clock skew with the variation of temperature in wireless sensor motes as done by [86]

on computer hosts for revealing hidden services in Tor network. We discussed the implication, pros and cons of clock skew based fingerprinting of sensor motes and the effect of temperature on the variation of clock skew of wireless sensor motes.

More information about this work can be found in the paper [84], and more details will be given in Deliverable D4.2.

### 3.1.2 Defending against Wormhole Attacks by Securing Neighbour Discovery in WSNs

In this work, we focus is on a particularly devastating form of attack, called wormhole attack [92]. A wormhole attack is particularly harmful against routing in sensor networks where an attacker overhears packets at one point in the network, tunnels them through the wormhole link to another point in the network. The other wormhole attacker broadcasts the packets among its neighbourhood nodes. A wormhole end can tamper the data, messages, or selectively forward data messages to disrupt the functions of the sensor network. The tunnel creates the illusion that the two end points are very close to each other. This allows an attacker to subvert the correct operation of the routing protocol, by controlling numerous routes in the network. Later, he can use this to perform traffic analysis or selectively drop data traffic.

Many researchers are proposed to detect and defend against wormhole attack. Most solutions used some special hardware such as the directional antenna and the precise synchronized clock to defend the sensor against wormhole attacks during the neighbour discovery process. Other solutions are based on statistical and geometric tests and local neighbourhood information. All tests are applied on lists of neighbours for 1 and 2 hops. These lists are constructed using Neighbour Discovery and the exchange of HELLO messages. For this reason, sending false lists of neighbours may change results of these approaches.

**Related work:** In [93] Hu et al. considered a general mechanism, called packet leashes. They presented two types of packet leashes: geographic leashes and temporal leashes. In geographical leash, each node must know its own location, and all nodes must have loosely synchronized clocks. Node location information is used to bind the distance that a packet can traverse. The detection of a wormhole attack is based on differences in the observed locations and calculated times. In temporal leash, network nodes must have globally synchronized clock which used to bind the propagation time of packets. The disadvantage of using packet leashes is that they require either location information for each node or need tight clock synchronization between the nodes.

Hu and Evans [94] developed a wormhole attack detection solution using special hardware called directional antennas which are not common for WSN. Their assumption is that the neighbour nodes examine the directions of the received signals from each other and a shared witness. Only when the directions of both pairs match, the neighbouring relation is confirmed.

In [95], an authenticated distance bounding technique called MAD is used. This protocol needs a special hardware radio to switch very quickly between send and receive mode. Receiver node has to respond to a series of bit exchanges sent by the sender node who compute the distance between them using timing properties. The scheme proposed in [96] detects wormhole attacks using connectivity information. The detection algorithm looks for forbidden substructures in the connectivity graphs that should not be present in a legal connectivity graph. Indeed, authors use a disk graph model to present the node. The disadvantage of this approach is that the scheme is not investigated in a real deployment. For this reason, authors propose to empirically estimate the forbidden structures in an attacker-free part of the network. However it might be impossible that a part of the network is wormhole-free.

In [97] authors presented a centralized wormhole detection technique which uses inaccurate distance estimations between neighbouring nodes to determine a virtual network layout using multi-dimensional scaling (MDS) technique. Indeed, the algorithm tries to determine a virtual position for every node in such a way that the constraints induced by the connectivity and the distance estimation data are respected. But it is considerably susceptible to distance estimation errors. Without any wormhole, the network layout should be relatively flat but the layout could be warped in presence of wormholes.

To prevent the effect of wormhole attack on the range based localization, authors in [98] presented consistency-based secure localization scheme against wormhole attacks including three phases: wormhole attack detection, valid locators identification and self-localization. This approach requires node classification.

In summary, the solutions developed to detect wormhole attacks often require distance measures, node classification and use sophisticated hardware such as directional antenna, GPS, synchronized clocks.

**Our Contributions:** We will study solutions against wormhole attacks and show that most of them are vulnerable and can be defeated. We will study the impact of sending false lists of neighbours on the precision of detection of the wormhole attack for three chosen approaches. Attackers send false lists when nodes execute neighbour discovery. We define a new attack, called “parallel wormhole”, where wormhole extremities and their neighbours send false lists of neighbours. We evaluated the impact of this introduced attack on the probability of detection of the wormhole attack. Simulation results proved that the three implemented approaches are influenced by the introduction of the new attack. They become unable.

## 3.2 Distributed Data Storage (DDS)

When categorizing wireless sensor networks (WSNs) with respect to the frequency and real-time responsiveness with which the environmental information is provided to authorised parties, in principle we face two cases: The first type of WSNs, which we term synchronous WSNs are WSNs where the monitored data is fluctuating and the system is most likely to be used for some real-time control monitoring. In contrast, we define an asynchronous WSN as a network that provides information to an authorised reader only seldom. The network continuously monitors and stores environmental data as a function over the time and over the monitored region. Consequently, after a period of monitoring and storing, the WSN contains a fine granular environmental fingerprint. Thus, complementary to the functionalities of a synchronous WSN, an asynchronous WSN also needs to provide components of a distributed database. However, there are several reasons why the implementation of a distributed database for an asynchronous WSN is much more challenging than that of its centralised counterpart located at a powerful server. First and foremost, we cannot assume the devices of a WSN to be tamper-resistant. Consequently, assuming that the WSN is deployed in public, untrusted, or even hostile environments, the entries in the database must be protected and concealed. Secondly, the overall storage space of a WSN is extremely limited and to orders of magnitude more restricted compared to its conventional database counterpart. Even a large scaled WSN will not even compare to the storage capacity of a traditional database. Thirdly, since in WSNs energy consumption is the major metric, database entries and database queries need to be translated in terms of number of hops, transmission distance and CPU cycles to accurately estimate the resulting energy consumption of the database operations. Finally, single nodes, or even whole regions of the WSN, may exhaust earlier than others. Consequently, a strategy for replicating the data storing at different devices is advantageous. However, replicated storage of data at multiple sensor nodes also increases the energy consumption for storage, data queries and data response operations. Also, persistent memory, which is an extremely scarce resource in WSNs, should not be wasted with too much redundant information. In recent advances, coding techniques have been applied to provide persistency at a relatively low memory cost. We discuss the different proposed solution below in the state of the art, and further step beyond it.

In our view, distributed data storage (DDS) middleware has to fulfil the following set of requirements:

1. *Persistency* - the distributed data storage should ensure the persistency of the data despite a reasonable number of dysfunction, failure or destruction of nodes.
2. *Transmission over the wireless* - transmission over the wireless is costly. In particular for a continuous redundant storage of data it may turn out to be extremely energy wasting;
3. *Memory restrictions* - a good balance between the accuracy of the representative data per time slot of the past and the occupied memory of these data is required;
4. *Security* - due to the wireless medium and the potential physical weakness of the device transmitted data are required to be authenticated and concealed;

5. *Disaster radius* - finally, in case the maximum area of destroyed nodes in case of a disaster can be estimated in advance, the disaster radius should be reflected in the storage policy of redundant data within the network;
6. *Query efficiency* - if the system is expected to be queried often, the efficiency of queries must be considered. In practical terms, that means that a query complexity in terms of communication and computation on the sensor side must remain as low as possible.
7. *Network topology* - the network topology itself may be of significant importance. This is in particular true if one considers not purely quadratic or circular shape of the WSN, but also WSN topologies that reflect the shape of the critical infrastructure to be monitored, e.g. railroads, pipelines, roads, etc.

The topic of secure and dependable distributed storage of environmental measurements within the WSN itself has gained momentum during the last years.

Solutions that store monitored data are in particular beneficial for unattended WSNs. In such a setting, the network itself has to store the monitored data for a time duration. Early work of distributed and encrypted data storage in unattended WSNs resulted from the EU project UbiSec&Sens with work on *tinyPEDS* [20], distributed shared memory, respectively the usage of bloom filters [21]. The focus in UbiSec&Sens is on the confidentiality of the data as well as on robustness and reliability in case of exhausting nodes. However, in the meanwhile more elaborated approaches have been proposed proposing to apply erasure codes respectively network coding for the storage of the data.

The main direction of this entire works is that the approach of network coding is valuable in effectively storing and transmitting monitored data to some  $n$ -hop neighbours of the node who monitored an environmental event. However, we believe that current related work does not consider all the listed requirements. In particular, security aspects have often been disregarded. Also, most approaches that we are going to present below are not implemented, as they rely on idealistic networks models.

### 3.2.1 Related Work on Data Persistency in coding based DDS

From the different requirements that must meet a DDS in wireless sensor networks, data persistency despite the failure of a subset of nodes is generally top ranked.

Generally, and if not stated otherwise, the metric to measure persistency is how much source data can be retrieved after that the failure of a certain number of storage nodes. In most papers, the network has two states: In the first state, the collection and distribution state, when data is collected, shared, and stored. The second state happens after an attack/failure phase, where the data is collected from all the surviving nodes, and the sink is trying to recover as much data as possible. Some pieces of data might be lost due to deletion of the source data or the impossibility of decoding retrieved encoded data.

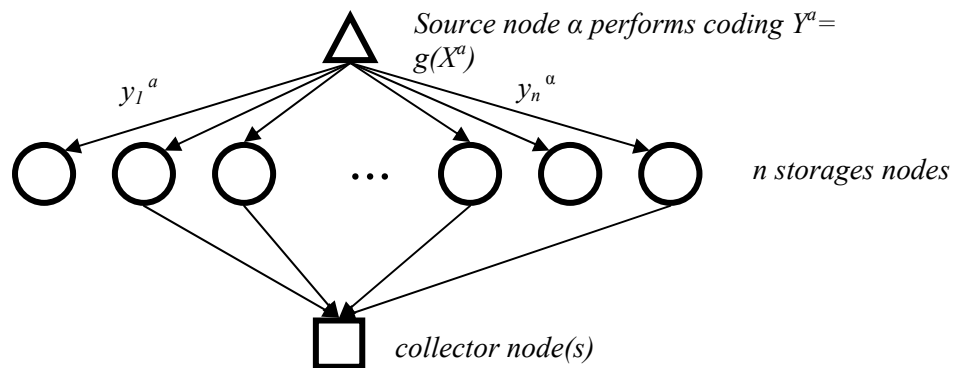
In this state of the art, we principally focus on data storages which are using coding to increase the data persistency. In a matter of fact, without coding, a random distribution of replicates of  $N$  source pieces of data in the network would requires a sink to collect  $O(N \log N)$  symbols to recover the  $N$  original pieces, according to the coupon collection problem. Therefore, erasure coding enhances persistency of the data at a lower communication and memory cost than a simple replication mechanism.

To apply coding, one must first decide what the source symbols are. As sensors collect environmental data principally, there are mainly two directions:

1. A node centric approach that only consider one piece of source data. A node measures a sample (the source data) and applies an erasure code such as Reed-Salomon (RS) on it, and distributes splits of the encoded data to a set of storage nodes. Figure 6 shows this approach.
2. The second approach will code over the samples of several nodes. For a specific time period, each node measures a sample. Nodes share their sample, generally without any coding, to their neighbourhood. Nodes then apply an erasure code on the packets they had received. Generally, a rateless code is applied, because the coding is applied in a distributed manner, and not by the source

itself (with codes like fountain codes, the node does need to have access to the whole source data to successfully encode). Such a distributed coding is represented in Figure 7.

HybridS [47] applies the first approach. Each source node first authenticates the data it measured and encrypts it with a temporary key  $k_r$ . It then encrypts  $k_r$  using a key shared between the node and the sink only, and destroy  $k_r$ . Then it applies  $(m, n)$  Shamir's secret sharing scheme on the encrypted key, in which  $m$  of  $n$  shares are required in order to reconstruct the secret. It then encodes the authenticated and encrypted source data with a  $(m, n)$  erasure code, in this case Reed Salomon. That is the data is split into  $n$  shares of size  $l/m$ , where  $m$  shares are needed to reconstruct the data. It then distributes the output of both coding and secret splitting to a set of randomly selected  $n$  nodes, which will store them. HybridS. Consequently, the sink can reconstruct the data by retrieving at least  $m$  shares from those  $n$  storage nodes. The scheme offers strong dependability properties such as confidentiality, integrity and persistency.



**Figure 6: First approach: encoding at the source node and distributed storage.**

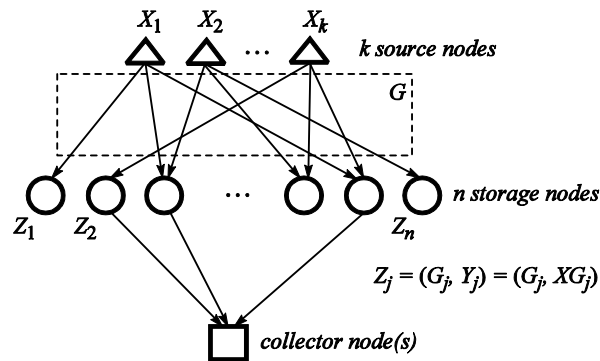
In a recent paper, Wang et al. [48] propose a DDS based on the same building blocks as HybridS. In addition, their scheme offers a dynamic integrity insurance to ensure that distributed data shares are correctly stored over the lifetime. Indeed, an algebraic signature is applied to each stored data, which is the concatenation of a secret share and an encoded RS block. Thus, a node can launch a verification process by issuing a challenge to all the share-holders (In the HybridS context, that are the  $n$  nodes that store encoded data related to the source data.). If the data integrity check is successful, the challenger is ensured that there was no modification of the encoded data at any of the nodes, although a false negative result is possible. From a security point of view, this is the work offering the most security features, such as persistency, integrity, confidentiality and authentication.

Following the second approach to apply erasure codes, numerous papers aim at using fountain codes as a base to provide persistency in the network. The main properties of rateless erasure codes are that they are resilient to erasure (*erasure* property) and can generate a quasi infinite number of codewords (*rateless* property).

Dimakis et al. [49] [50] are the first to propose a decentralised fountain codes scheme for DDS, because of their rateless and low complexity properties. In fountain codes, the pieces of the source data are assumed to be of the same length and of finite number. The encoding operation is simply to the bit-wise operation XOR (exclusive OR) on  $d$  source pieces,  $d$  is thus the *degree* of the encoded packets. The decoding operation consists of solving the linear equation formed by the codewords. The authors lay the ground for a decentralized schema using erasure codes, with proofs that the stored data can be decoded, or that there is no isolated nodes when using random walks. They furthermore study such storages in specific settings, such as a grid topology.

The network model on which their papers are based is common to many papers of the related work. It assumes two categories of nodes,  $k$  collecting nodes which measure at each period of time a sample  $x_i$  and  $n$  storage nodes, which store one codeword per period. The model in question is depicted in Figure 7.

Like many related papers, Dimakis et al. rely on random walks to distribute the source data among the storage nodes. It has the advantage of offering a solid basis for analysis with established properties, such as the expected number of walks needed to disseminate a source packet in a mesh network.



**Figure 7: Second approach: Distributed coding and storage system model.**

Lin et al. [82] tackles the problem of offering a decentralized fountain codes scheme. Like in the previous papers, the authors propose a random walk scheme to disseminate the data. Furthermore, they proposed two decentralized algorithms which aim at approaching the robust soliton degree distribution, with minimal RAM requirements on the storing nodes, their algorithm only requires to keep in memory a buffer for the encoded data and another buffer to treat incoming packets. With their algorithm, they approach persistency performance very close to a centralized solution.

Aly et al. studied in a series of papers ([54][52][53]) the use of different coding schemes in the same network model as represented in Figure 7. They consider Raptor and Fountain codes under specific knowledge assumptions (with or without global information) and study the persistency of the data if a subset of nodes fails.

Kong et al.[55] further develop the concepts of distributed coding and random walks by developing algorithms that are truly distributed and does not require any global information, such as the number of nodes in the network or the number of sensing nodes, thus increasing the self-configurability of the DDS. They study two solutions: one based on LT codes, and another based on Raptor codes.

Kamra et al. work [56] aims at finding the optimal degree distribution for fountain codes when using an iterative decoder such as in LT codes, where the objective is to decode at least  $r$  pieces out of  $k$  source data. Other papers generally focus on finding distribution to recover all the pieces.

All the works mentioned above consider that the different pieces of data to be stored are of equal importance. Lin et al. [57] tackles the issue of persistence priority, using less efficient (in terms of decoding) random linear codes. They twist the encoding matrix, such that using the Gaussian-Jordan elimination algorithm, high priority data is decoded first. Thus, in case of node failures, it is highly likely to recover the important data, while it becomes less likely to recover less important data.

Also non-autonomous networks consider coding based distributed data storage. For example, Kamra et al. devised Growth Codes [51]. The scheme copes with the bottleneck issue of data gathering in sensor networks. The idea is to use rateless erasure codes such as Fountain Codes on the way to the sink. This prevents data loss if nodes fail before the sink could collect everything. But furthermore, it adapts, or "growths", the degree distribution of the code along with the collection of the data to the sink. The idea is that the sink collects low degree codewords first, and then codewords with higher and higher degree. This allows the sink to be able to decode more codewords than using a naïve degree distribution such as the *robust soliton*.

Table 3 gives an overview on coding based distributed data storage. We see that many fountain code based data storage do not consider security. In the literature, the two works are disjoint, as we it was reported in the related work on data integrity previously. Another remark is that coding based DDS requires a lot of messages exchanges. This can also explain that there are actually no known implementations of those

protocols (code running on real mote), as implementing a robust dissemination of the data would not be a trivial task.

**Table 3: Summary of existing coding based DDS.**

<i>Paper</i>	<i>Persistence</i>	<i>Confidentiality</i>	<i>Authentication &amp; Integrity</i>	<i>Coding</i>	<i>Energy efficiency</i>	<i>Dissemination method</i>	<i>Comments</i>
[49] [50]	.	Not considered	Not considered	Fountain code	High comm. OH	Fast random walks	One of the early papers for fountain code based DDS
[54][52] [53]	-	Not considered	Not considered	Fountain code, Raptor codes	High comm. OH, low comp OH	Random walks	This series of papers offer a good analysis of the different fountain code solutions.
[57]	Consider also local disaster, data persists if less than half of the nodes survives	Not considered	Not considered	Fountain code with increasing degree distribution with time	low comm. OH, low comp OH	Cluster-based convergecast towards sink	Different network model, where the DDS is used principally to thwart the bottleneck effect of tree topologies
[57]	Higher priority codewords more likely to be decoded	Not considered	Not considered	Priority coding with Gauss-Jordan elimination decoder	High comm. OH, high comp OH.	Random multicast	Data has different priority levels. Data with higher priority has more chance to survive.
[55]	About $1.15k$ nodes need to be queried	Not considered	Not considered	LT code / Raptor codes	High comm. OH, low comp OH.	Random walks	Relax assumption on global knowledge from nodes $(n,k)$ . No buffer needed.
[56]	Dependant on the objective parameter $r$	Not considered	Not considered	LT codes with distribution dependant on $r$	High comm. OH, low comp OH.	Not specified	Study a specific problem related to fountain code based DDS
[82]	Decoding ratio depends on walks lengths.	Not considered	Not considered	LT codes with degree distribution	High comm. OH, low comp OH.	Random walks with stops using	Only one space buffer needed.

<i>Paper</i>	<i>Persistence</i>	<i>Confidentiality</i>	<i>Authentication &amp; Integrity</i>	<i>Coding</i>	<i>Energy efficiency</i>	<i>Dissemination method</i>	<i>Comments</i>
				closed to the robust soliton		Metropolis algorithm	
[47]	Depends on RS coding rate.	Shamir's secret sharing	Sink can authenticate retrieved data.	Reed-Salomon	High comm. OH	Random multicast 1-N	Node centric distributed data storage.
[48]	Depends on RS coding rate.	Shamir's secret sharing	Nodes can verify data integrity via a challenge.	Reed-Salomon	High comm.. and comp. OH	Random multicast 1-N	Node centric distributed data storage. Very good security properties.

If not specified,  $n$  represents the number of storing nodes and  $k$  the number of source nodes. A node can be both measuring and storing in some network models.

### 3.2.2 Related Work on Data Management in DDS

There is a broad previous art on database and data management for sensor networks. These works focus on providing a useful interface to collect data in a sensor network, and dependability is at most a secondary goal of the design. We report here on general work on DDS, not focusing any more on solutions based on coding. User experience and query processing have here more relevance.

TinyDB[61] was one of the early proposals for distributed data acquisition in sensor networks, and developed a rich and user friendly query language, while optimizing communication processing for the queries, using in-network processing. The database is acquisitional; sensors will collect and report data according to the query. TinyDB only buffers data for a short amount of time, to allow historical queries, but does not aim at providing persistency.

Data Centric Storage [64] explores the idea of linking where in the network the data is stored according to the data values. Consequently, a query knows exactly which nodes to ask to retrieve the query answer. The approach is further improved by DIFS[66] and DIM[68] which respectively improve the scalability of the data and the provide a support for range queries.

There are furthermore two-tier approaches, which propose a network model, composed of RFD and FFD devices. In brief, the RFDs report measurements to the FFDs, which process them and eventually report to the base station. In TSAR [60], the upper tier is responsible for indexing and caching data from the underlying layer, therefore executing queries very efficiently. PRESTO [70] relies on the statistical properties such as the spatial and time correlation of the collected data. The upper layer is responsible to calculate a model, built on previous sensor reading. The lower layer will then only report data if their reading deviates too much from the model.

The same group of authors of TSAR and PRESTO, who also developed Capsule [59], takes advantage of the recent advances in NAND Flash memories to elaborate a more energy efficient database, StonesDB [58], based on the fact that operation on the Flash will become cheaper and cheaper.

In the FP6 UbiSec&Sens, one partner developed tinyPEDS [20], a DDS that feature security aspects such as confidentiality of the stored data, using privacy homomorphic transformations. TinyPEDS has optional modular security features such as access control, memory integrity or data filtering (to discard faulty measurements). Also, a distributed shared memory tinyDSM [102] was developed, which allows a node to replicate a variable among its neighbourhood. This variable can be read in a distributed manner. Furthermore, the node owning the variable can update the value, which will automatically update the replicates in the network. This offers a powerful middleware to share data for distributed algorithms.

### 3.2.3 Related Work on Support Tools for DDS

An important part of distributed data storage is the file system abstraction used by the sensor to store the data. Its efficiency is important in many aspects: software complexity, energy efficiency, memory efficiency among others. It is not an easy task to manage Flash memory: generally, it can be only written once, and a further write operation requires the erasure of a full memory block. The block size depends on the hardware, but is generally of the size of 512 or 1024 bytes. Therefore, Flash memory cannot be used like a RAM from a developer point of view. TinyOS 2.1 offers in its distribution 3 different API to store and read from the Flash:

1. Configuration: Simulate a RAM API. Used to store small pieces of data that can be rewritten frequently.
2. Logging: API to store series of data pieces. It follows an "append only" policy.
3. Block: API for large pieces of data.

Storing data on a sensor mote require some understanding of the underlying hardware. As the memory resource is scarce, the developer must know what type of API is best suited for its application.

Matchbox [101] is a storage middleware for tinyOS. It provides a file system to the application, a feature that is not supported by tinyOS native API.

More recently, *Capsule* [59], which will be included in the latest release of tinyOS, offers new features and improved performance. It is based on an object abstraction which can meet different application needs: streams (logging), queues, temporary array or file storage. It further allows the establishment of checkpoint that can be used to restore the memory to a known state.

### 3.2.4 Related Work on Data Integrity for coding based DDS

An algorithm to detect errors in communication systems based on network coding principles is presented in [37]. In that algorithm, a hash value is appended to each data packet. It is assumed that the destination node receives at least one or more unmodified packets, and checks the inconsistency of the decoded packets using the appended hash values.

One important result for correcting errors introduced by a Byzantine adversary in network coding based communication systems is presented in [38]. In that paper, the authors introduce an information-theoretically rate optimal code. The packets from the adversarial nodes are intuitively considered as packets coming from a second source, and the packets arriving at the destination are linear combinations of the source's batch of packets and the adversary's batch of packets. Linear independence is assumed within and between these batches. The destination node is assumed to receive all the packets destined to it, and then, it tries to distill out the original data packets from the polluted set of packets. Compared to these works, we cannot assume any encoding of packets at the source nodes, because there is more than one source.

Due to the distributed source classical error detection and codes (such as Reed-Solomon codes) are also not appropriate; furthermore, unlike our solution they require additional redundancy. This holds for the error correction code proposed for network coding [39][40] also. This work describes a Reed-Solomon like code construction that operates with subspaces instead of Galois symbols.

Cryptographic techniques have also been proposed to detect attacks in coding based communication and storage systems. For instance, in practical P2P file sharing systems, data blocks are often hashed and the hash values are made available at a central trusted publisher. By comparing the hash of each downloaded data block to the corresponding hash available at the publisher, a node can verify whether a downloaded block is valid or not.

In order to make this idea work in network coding based P2P file sharing systems, the usage of homomorphic hash functions [41] is proposed in [42][43]. In the proposed scheme, the hash of an encoded packet can be easily derived from the hashes of the blocks contributing to the encoding. It is assumed that the hash value of every block of a given file is obtained by the nodes in a secure way when they first join the system. These hash values are then used to verify the integrity of the encoded packets as they are downloaded. To reduce the computational overhead caused by homomorphic hash functions, the scheme proposed in [43] also requires the nodes to cooperate and alert each other when a maliciously modified block is detected. In this way, a given node does not verify each and every block itself, but it can rely on alerts from other peers.

In any case, every scheme that uses hash functions (be it homomorphic or not) requires the existence of a secure channel between the data sources and the destinations through which the genuine hash values of the original data blocks can be obtained. We do not assume such secure channel in our approach.

Another approach to prevent the pollution attack is to require the source nodes to digitally sign the data blocks before they are injected in the system. However, in order to make this work in systems where intermediate nodes combine data blocks received from different sources; the digital signature scheme must have some homomorphic properties, similar to the case of homomorphic hash functions described above. Recently homomorphic digital signature schemes have been proposed for network coding based content distribution in [43][45][46].

Unlike the approach based on homomorphic hash functions, the approach of using homomorphic digital signatures does not require a pre-existing secure channel between the sources and the destinations. However, it has two other problems: first, homomorphic signature schemes are computationally even more expensive, and second, they need a public key infrastructure (PKI) for the management of the signature verification

keys. These problems hinder their usage in practical applications; in particular, due to the large computational complexity they cannot be used in sensor networks, and due to the PKI requirement, it is unlikely that they will ever be used in large scale P2P content distribution systems.

### 3.2.5 Our Contribution in coding based DDS

It is our intention to build a practical coding based distributed data storage. As we have seen in the related work, there is not known implementation that runs on existing hardware of the protocols we presented. It is our aim to provide a middleware built on efficient and realistic message services. We will also look at the issue of efficient querying, a subject often overlooked in the related work. Another interesting approach would be to look at self-healing data storage, where the memory of failed nodes is “regenerated” by the network, in order to ensure a high level of persistency throughout the lifetime of the network.

Another contribution of WSAN4CIP is a method to detect pollution attacks on coding based DDS without the use of any cryptography. We present it in details in the following section.

### 3.2.6 A non-cryptographic Approach to Providing Integrity in coding based DDS

#### *Introduction*

We consider multiple, distributed sources that generate data that must be stored efficiently in multiple storage nodes, each having constrained communication, computation, and storage capabilities. Storing encoded data, instead of raw data, can help to increase the efficiency of the system. In [33][34], for instance, the following scheme is proposed: There are  $k$  source nodes, each producing a single data packet of interest (per time epoch), and there are  $n$  storage nodes that are used as a distributed memory for the  $k$  data packets. Each storage node can store a single data packet. Instead of storing raw data packets, each storage node stores a linear combination of a subset of them. The random coding techniques (distributed erasure codes, fountain codes) introduced in [34][49] [35] ensure that, for appropriately selected parameters, a collector node can reconstruct all the  $k$  data packets with high probability by downloading the encoded packets from *any*  $k$  storage nodes and solving a system of linear equations (s.l.e). Thus, the collector node can retrieve the interested data from  $k$  nearby nodes, which results in decreased delay in data reconstruction and lower traffic load in the network.

While coding may increase the efficiency of distributed storage systems in a benign environment, it also has a potential problem in hostile environments, where an adversary may attack the storage nodes. In particular, the problem that we consider is the so called *pollution attack* [36], whereby the adversary modifies some of the stored encoded data, which results in erroneous decoding of a large part of the original data upon retrieval. Note that these coding schemes mix (typically, linearly combine) blocks of the original data, therefore, a single corrupted encoded block can affect the decoding of multiple data blocks. This amplification effect of the pollution attack is particularly annoying and undesirable.

Our main contribution is a novel information theoretic approach to counteract pollution attacks in coding based distributed storage systems. Compared to other approaches in the same vein, we do not add redundancy to the data packets, but rather, we take advantage of the inherent redundancy provided by the coding scheme itself. This redundancy comes from the fact that the content of each storage node corresponds to the same data block vector. The price of this property is only a slightly increased communication overhead for the attack detection.

The general model of the distributed storage systems that we consider is taken from [34] and it is illustrated in Figure 7. The system consists of  $k$  source nodes,  $n$  storage nodes, and one or more collector nodes. Note that these are roles, and therefore, the sets of source nodes, storage nodes, and collector nodes may overlap. Only the collector node is assumed to be a powerful computer (base station), while source and storage nodes may be low capacity devices. Each source node  $i$  generates a data block  $X_i$  and transfers it to some randomly selected subset of the storage nodes. Each storage node  $j$  computes a random linear combination of all the data blocks that it receives; the result is a single code block  $Y_j$ . Formally, we can write that  $Y_j = XG_j$ , where  $X = (X_1, X_2, \dots, X_k)$  is the row vector of all the data blocks, and  $G_j = (g_{1j}, g_{2j}, \dots, g_{kj})^T$  is a column vector,

the non-zero elements of which are the random coefficients used in the linear combination. Here,  $g_{ij} \in GF(q)$  for all  $i = 1, 2, \dots, k$  and  $j = 1, 2, \dots, n$  and for some  $q$ . Each storage node  $j$  stores the pair  $Z_j = (G_j, Y_j)$ , which represents the equation  $Y_j = XG_j$ . The entire system is represented by the system of linear equations (s.l.e.)  $Y = XG$ , where  $Y = (Y_1, Y_2, \dots, Y_n)$  is the row vector of all code blocks, and  $G = (G_1, G_2, \dots, G_n)$  is a  $k \times n$  matrix that contains the coefficient vectors in its columns. Matrix  $G$  is also called generator matrix.

For appropriately selected values of  $k$  and  $q$ , any  $k \times k$  sub-matrix of  $G$  is non-singular with high probability. Therefore, the collector node can reconstruct all the data blocks with high probability by downloading the equations from any  $k$  storage nodes and solving the obtained s.l.e. for  $X$ . In the rest of the deliverable, we assume that this property of  $G$  holds.

In fact, each data block  $X_i$  can itself be a column vector of  $m$  symbols  $(x_{m1}, x_{m2}, \dots, x_{mi})^T$  where  $x_{il} \in GF(q)$  for all  $i = 1, 2, \dots, k$  and  $l = 1, 2, \dots, m$ . In that case, each code block  $Y_j$  is also a column vector  $(y_{j1}, y_{j2}, \dots, y_{jm})^T$  of  $m$  symbols in  $GF(q)$ . The linear combination  $Y_j = XG_j$  is computed in symbol-by-symbol manner, meaning that  $y_{lj} = \sum_{i=1}^k x_{il} g_{ij}$  for all  $j = 1, 2, \dots, n$  and  $l = 1, 2, \dots, m$ . Thus, one can think of  $X$  and  $Y$  in the s.l.e.  $Y = XG$  as matrices of size  $m \times k$  and  $m \times n$ , respectively. This view is illustrated in Figure 8.

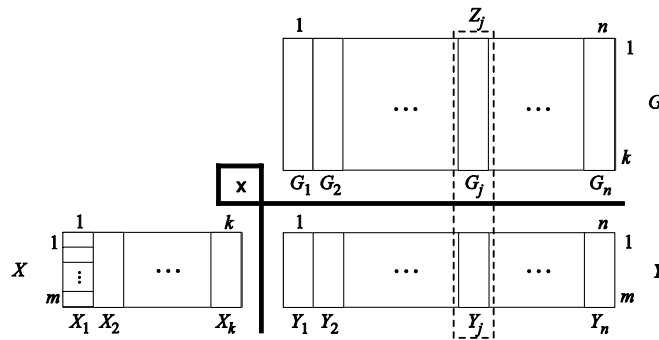


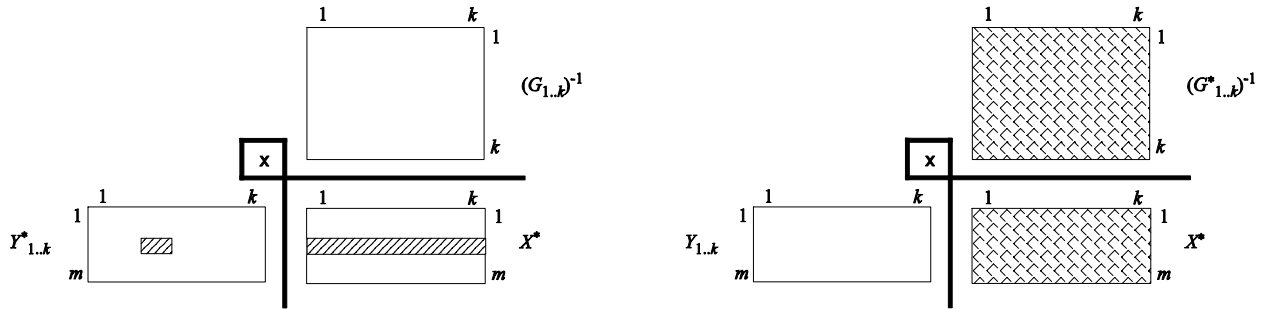
Figure 8: Illustration of the s.l.e. representing the storage system

### Adversary model

We assume that the adversary has access to  $t$  storage nodes, and she can observe and modify the equations stored by them. This means that if the adversary has access to storage node  $j$ , then she can modify both  $G_j$  and  $Y_j$  stored by node  $j$ . Let  $G^* = G + \Delta G$  and  $Y^* = Y + \Delta Y$  be the modified generator matrix and the modified code block vector after an attack, where the modifications made by the adversary are contained in matrix  $\Delta G$  and vector  $\Delta Y$ . We further allow the adversary to compromise the communication links of the  $t$  storage nodes. It gives more possibility to the adversary, but does not extend the possible effect of the attack. For simplicity, we refer to nodes that store modified data as compromised nodes, and not distinguish them upon the way of modification. Note that the adversary has no access to the source nodes, rather she aims at compromising the output of the storage system. The rationale behind this assumption is that storage nodes are exposed to attacks for an extended period of time, whereas the source nodes must be attacked during the limited time period of data generation. Data distribution from the source nodes to the storage nodes typically takes place on a wireless channel, which is exposed to various attacks. Accordingly, our applied model of adversary is realistic in most cases.

Recall that when reconstructing the data blocks, the collector node chooses the  $k$  storage nodes, from which it downloads the  $k$  linear equations, randomly. Therefore, the adversary has no information on which storage nodes will be chosen when she performs the attack. At the same time, the collector node does not know which storage nodes are compromised. In the sequel, we will assume without loss of generality that the adversary randomly chooses the  $t$  storage nodes to be compromised, and the collector node downloads the equations of the first  $k$  storage nodes, where the order of the storage nodes is defined randomly by the collector node. Thus, the set of equations downloaded by the collector node is  $Z_{1:k}^* = (G_{1:k}^*, Y_{1:k}^*)$ , where  $G_{1:k}^* = (G_1^*, G_2^*, \dots, G_k^*)$  and  $Y_{1:k}^* = (Y_1^*, Y_2^*, \dots, Y_k^*)$ .

Let us now investigate the effect of an attack. The collector node solves the s.l.e.  $Y_{1..k}^* = XG_{1..k}^*$  for  $X$ , and obtains the result  $X^* = Y_{1..k}^*(G_{1..k}^*)^{-1}$ . Let us suppose for the moment that the adversary modifies only the code blocks, meaning that  $G^* = G$ . In this case,  $X^* = Y_{1..k}(G_{1..k})^{-1}$ . The modification induced by the attack in the decoded data blocks can be computed as follows:  $\Delta X = \Delta Y(G_{1..k})^{-1}$ . This means that (a) if a given row of  $\Delta Y_{1..k}$  contains only zeros, then the corresponding row of  $\Delta X$  will contain only zeros too, and (b) a non-zero element in a given row of  $\Delta Y_{1..k}$  will affect the entire corresponding row in  $\Delta X$ . Thus, a modification made by the adversary in a given row in any of the first  $k$  code blocks will, in general, affect all decoded data blocks, but the effect will be limited to the corresponding row. This is illustrated on the left hand side of Figure 9.



**Figure 9: Effects of the pollution attack**

Now, let us suppose that the adversary modifies only the coefficient vectors, meaning that  $Y^* = Y$ . In this case,  $X^* = Y_{1..k}(G_{1..k}^*)^{-1}$ . If at least one of the first  $k$  coefficient vectors has been modified by the adversary, then  $G_{1..k}^* \neq G_{1..k}$  and thus  $(G_{1..k}^*)^{-1}$  can be completely different from  $(G_{1..k})^{-1}$ . Therefore, in general, such a modification affects all decoded data blocks in every row. This is illustrated on the right hand side of Figure 9.

If the adversary modifies both the coefficient vectors and the code blocks, then these effects are combined. In the general case, the modification induced by the attack on the decoded data blocks can be derived as follows:  $\Delta X = (\Delta Y_{1..k} - X\Delta G)(G_{1..k}^*)^{-1}$ .

Actually, these observations illustrate the amplification effect of the pollution attack: a small amount of modifications in the stored coded information can result in a large amount of modifications in the decoded data. In the worst case all data blocks are entirely destroyed. This is highly non-desirable, and requires the development of some countermeasures. Below, we address this problem by proposing a mechanism to detect and recover from such attacks.

**Attack detection**

The basic idea of our attack detection mechanism is the following: In most cases, the adversary cannot enforce a particular solution  $X^* = Y_{1..k}(G_{1..k}^*)^{-1}$ , because it is unlikely that she compromises all the first  $k$  equations – the probability of this event is  $\frac{\binom{k}{t}}{\binom{m}{t}} \approx \left(\frac{k}{m}\right)^t$ . In most of the cases,  $X^*$  can be treated as a random vector, except if all the first  $k$  equations are intact, in which case  $X^* = X$  will hold. Of course,  $X^*$  is not really random, but is out of control of both the adversary and the collector, hence from the analysis point of view we can treat it as random.

Now, suppose that we have an additional intact equation:  $Y_{k+1} = XG_{k+1}$  (i.e., the collector downloaded  $Z_{k+1} = (G_{k+1}, Y_{k+1})$ ). If  $X^*$  is random, then it will not satisfy the additional intact equation with high probability, while it will satisfy it with probability 1 if  $X^* = X$ . Thus, we can detect if the decoded data block vector  $X^*$  is polluted with the help of an additional intact equation. In the rest of this section, we develop an attack detection algorithm based on the principle described above.

### Algorithm

The proposed attack detection algorithm works in the following way: The collector downloads the first  $k$  equations  $Z_{1..k}^*$  and computes  $X^* = Y_{1..k}^*(G_{1..k}^*)^{-1}$ . Then, the collector downloads the next equation  $Z_k^*$ . If  $Y_{k+1}^* - X^* G_{k+1}^*$ , then no attack is detected (and the collector accepts  $X^*$  as the correct solution). Otherwise, if  $Y_{k+1}^* \neq X^* G_{k+1}^*$ , an attack is signalled.

### Conclusion

We presented a non-cryptographic approach to detect pollution attacks in coding based DDS. It uses natural properties of the system, i.e. redundancy and linear system over-determination, to achieve its goal. We will present in deliverable D4.4 an efficient recovery algorithm, as well as communication and computation cost of the algorithms.

## 3.3 QoS and Energy Middleware Manager

QoS is very important for applications with specified performance requirements. These requirements include reliability [11][12][16], security [13][14][15], real-time [16][17][18][19] and fault-tolerance [9][10]. However, usually in the WSN all these quality parameters are driven by the most important factor – the energy. Actually, the energy is also the connector between different requirements. The connection shall be realised in a way that allows some on-the-fly trade-offs and energy budget shifts between the types of quality to fulfil the common goal.

The realisation of the QoS Manager Middleware can be split into several phases, where the requirements are mapped into the quality parameters. In each phase other limitations play the main role. First, the general application requirements need to be converted into a set of modules needed to be compiled together (for instance using the ConfigKit [99]). At this moment the WSN node hardware and the software are seen as a, more or less, static combination. The required software modules are examined regarding the available memory, the code memory to store the program as well as the RAM to keep the variables, stack etc. The set that fits both, the hardware and the requirements, the best, is chosen and compiled. If all the required functionality cannot be included due to hardware limitations, then the overall requirements need to be reviewed. Next phase is the run-time, but seen only from a single node perspective. At that moment the dynamic interdependencies of the modules and their competition for resources play the main role. If the node is about to send a message, that should be encrypted, but the energy level is too low for the default encryption, then maybe it is better to use a less secure but also less energy consuming encryption scheme for that message or even to skip the encryption in order to allow the message to be sent. The third and last phase is an extension of the second. It includes the same decisions, but in a distributed manner on the nodes in the WSN. Of course all the problems are scaled from the node to the network perspective. So if a node wants to access some information and there are several nodes that can deliver that data, then the source and the route to it may be chosen based on the energy cost basis.

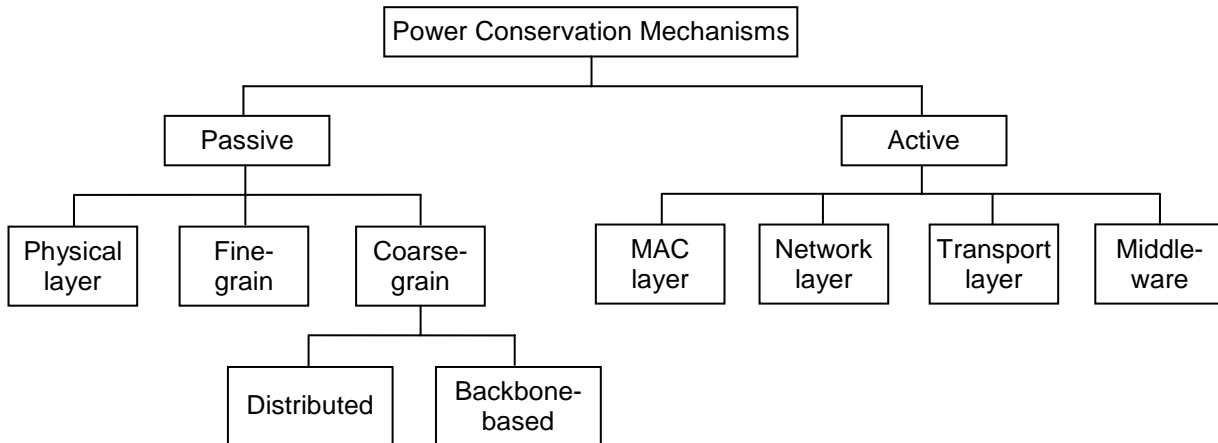
In an unusual case where the energy is not limited, there may be other limitations. An example is the network bandwidth. In applications with high bandwidth demand, possibly combined together with real-time requirements it may happen that some nodes may represent bottlenecks and cause data losses. In order to prevent such a situation and avoid the need to drop packets, some of them may be routed to less loaded nodes or routes.

In both cases, there is a need to provide a layer that provides the neighbouring nodes with information they need to take the decisions. In WSN4CIP, we propose to develop an intelligent QoS middleware that provides an energy-aware cross-layer solution. Knowing the restrictions and the state of the other nodes and the application requirements the network can take the decisions more aware and thus, it can perform better to fulfil the goal. This set of information includes, for instance, the node status (alarm/no alarm/scheduled test), the energy left in the batteries, the packet loss pattern and the packet transmission delay and allows to dynamically reconfigure parameters such as the energy saving mode of the CPU and peripherals, the idle periods, as well as the radio transmission power level. Additionally, the level of data aggregation, encryption

and the sample rate of sensor data can also be controlled. Anyway, such a distributed QoS management implies additional energy and traffic overhead, thus, there is a need to research the possibilities, examine their costs and then, specify their application areas.

### 3.3.1 Power Saving Mechanisms

Power conservation mechanisms can be classified into two main categories [1]: Active and Passive (Figure 10). *Active* refers to mechanisms that achieve energy conservation by smartly utilizing energy-efficient protocols (by not turning off the radio interface), while *Passive* refers to mechanisms that save a node's power by turning off the radio (transceiver) interface module when there is no communication activity.



**Figure 10: Classification of power conservation mechanisms**

Periodic hibernation is a synonym to Passive power conservation that is frequently found in the literature. The idea of turning off the radio transceiver was firstly introduced in IEEE 802.11. According to this standard, a sensor node may switch to sleep mode in accordance with the Network Allocation Vector (NAV). Also, every sensor node in the network must wake up during the Announcement Traffic Indication Message (ATIM) period, during which transmitters inform their destinations not to turn to power save mode. If no notification is received, the sensor node can turn to power save mode and wake up in the next ATIM period.

Physical layer power conservation techniques can produce substantial energy savings by reducing to the minimum the energy of the Central Processing Unit (CPU) in idle system states. Dynamic Voltage Scaling (DVS) and Dynamic Power Management are typical examples of this passive power conservation mechanism.

According to fine-grain power conservation mechanisms, the Medium Access Control (MAC) layer is responsible for deciding if there is a frame transmission that is destined to it, and then turn off the radio for that transmission frame to save power if there is none.

Coarse-grain power conservation mechanisms use higher layer application information to decide when to turn off the radio interface module. There are two implementation approaches: the distributed approach, and the backbone-based approach. In the distributed approach, sensor nodes independently schedule their sleeping intervals based on both internal information and/or neighbour information. The coordination of sleeping schedules is done implicitly through exchanged beacon or hello messages. In the backbone-based approach, a backbone infrastructure is required to be set up and the power conservation application resides on the backbone sensor nodes. The power conservation application can have a better view of their local network environment.

As for the active power conservation mechanisms, at the MAC layer, one approach is to adaptively adjust the transmission power to an appropriate level just enough to reach the next hop. An additional objective is to reduce the collision probability by reducing the radio range.

At the network layer, two basic active power saving schemes are used. Power-aware routing has the objective of finding routes that consume the least possible power. Maximum lifetime routing, in contrast, balances energy dissipation among sensor nodes to prolong the operational lifetime of the network.

At the transport layer, protocols can alter the packet retransmission behaviour by reducing unnecessary retransmissions to a minimum and thus achieving lower power consumption.

At the middleware level, protocols can perform data aggregation [2], reducing the total number of transmitted messages and thus saving energy. Messages can aggregate data from different source nodes or from different samples from the same node.

Naturally, power conservation mechanisms can be combined. For instance, LEMMA (Latency-Energy Minimization Medium Access) [3] is a new MAC protocol that combines active and passive techniques. LEMMA uses carrier sense mechanisms to evaluate the real interference level between nodes and uses this information to schedule active and sleeping slots so as to minimize collisions between the different nodes' transmissions.

Another interesting research topic is combining energy conservation mechanisms with quality of service (QoS) constraints. Authors of [4] use a DVS mechanism to reduce energy consumption while guaranteeing soft real-time constraints in job execution. LEMMA [3] is an example of MAC protocol that minimizes latency and energy. [5] presents a routing protocol that meets QoS objectives of throughput and latency as well as energy conservation to maximize network lifetime. [6] and [7] are other example of routing protocols for wireless sensor networks combining energy and QoS requirements.

Sensor power consumption can vary significantly according to what the sensor is doing. For instance, the Silex SX-560 [8] Embedded Intelligent Programmable 802.11a/b/g Module has a typical consumption of 280mA (0.92W), needing 400mA (1.3W) while transmitting, 200mA (0.66W) while receiving, 53mA (0.175W) when idle and 73mA (0.24W) when in an energy saving mode that just answers beacon polls. Naturally, if the sensor node is permanently active, its response to events can be immediate. On the other hand, power-saving modes add a delay when responding to events, caused by the latency in waking the sensor node and taking some action, but allows saving energy. For the Silex SX-560, this delay is about 1 second, when in energy saving mode.

### 3.4 Services applied to CIP

In this section, we show how the different middleware pieces developed in WSAN4CIP serve the CIP application. To guide this evaluation, we refer to the risk assessment, already reported previously in section 2.2.

**Table 4. Risks covered by the service protection work package<sup>1</sup>.**

Attack type	Task	Involved modules
Physical destruction of nodes	All	Intrinsic property of the middleware: Middleware should be resilient to some node failures. Distributed data storage.
Dismounting and stealing nodes	All	Same as physical destruction.
Dismounting and relocating sensors	T4.1	Secure Localization, Skew-based services
Sensor input manipulation	None	Solutions available in the UbiSec&Sens toolbox [83]
Jamming	T4.3, T4.2	Distributed Data Storage, QoS middleware manager
Eavesdropping	WP3	Traffic analysis prevention.
Replay of protocol messages	WP2, WP3	-
Injection of crafted protocol messages	T4.1	Wormhole attack detection.
Corruption of stored data	T4.2	Distributed Data Storage
Remote code injection	WP2	-
Installing rogue software on nodes	WP2	-
Deployment of rogue nodes	WP2	-

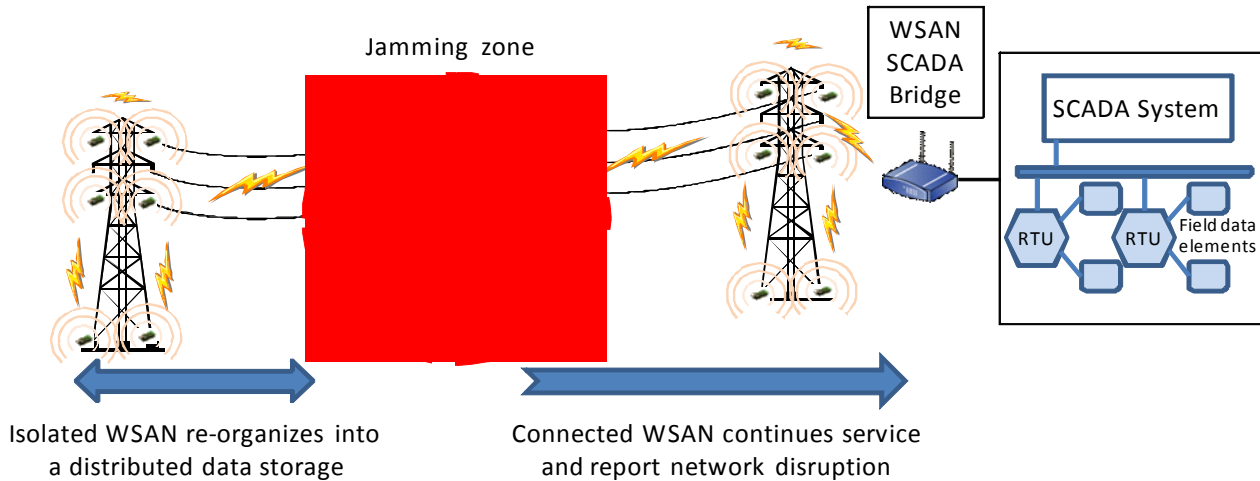
<sup>1</sup> Many risks that are covered in this work package are also covered in other packages. For example, jamming is also covered in work package on Dependable Networking, by applying multipath routing.

As we can see from Table 4, many of the risks identified in our preliminary studies are covered. The table might however minimize the impact of middleware such as the QoS and energy manager middleware. Indeed, this middleware tries to configure the node parameters at any point of time, and thus covers many other “risks” such as network congestion or poor wireless medium conditions. Point by point:

- *Physical destruction of nodes*: The functions and middleware provided in the work package should use one of the greatest features of a WSN: redundancy. All the algorithms that we develop can sustain the loss of nodes (except the sink node, which is assumed to be protected). Furthermore, distributed data storage permits a node to save its data beyond its destruction, by storing its data to its neighbour.
- *Dismounting and stealing nodes*: From this work package point of view, this is similar to the destruction of a node.
- *Dismounting and relocating sensors*: Security elementary services can prevent the relocation of nodes. Skew based services can detect a change in the temperature pattern of a node. Secure localization further prevents the move of nodes.
- *Sensor input manipulation*: Faulty or manipulated of measurements of correlated events can be detected by RANBAR [83], develop in UbiSec&Sens.
- *Jamming*: The DDS and QoS middlewares can mitigate jamming attacks. The QoS manager will prevent that a node spent uselessly its energy in a jammed network, while the DDS might continue collecting data in the disconnected part of the network (as it cannot report to the sink any more). This is discussed in more details below.
- *Eavesdropping*: We do not consider eavesdropping as a major issue in this work package (Service Protection). Although, preventing traffic analysis (developed in WP3) can increase the robustness of a middleware against a smart attacker.
- *Replay of protocol messages*: The specification and implementation of algorithms should be robust against replay of messages. Furthermore, transport protocols (WP3) should prevent the replay of message at their respective layers.
- *Injection of crafted protocol messages*: Solutions are similar as for protocol messages, but this might not be enough. Intrusion Detection Systems developed in WP2 may detect unrequested or anomalous traffic, thus thwarting this kind of attacks.
- *Corruption of stored data*: Section 3.2.6 presented a solution to data corruption.
- *Remote code injection, Installing rogue software on nodes, Deployment of rogue nodes*: Secure code update and software attestation (WP2) prevent such attacks.

**An example of service protection: mitigating jamming**

The distributed data storage might not have an obviously application in the domain of CIP. However, Figure 11 shows how a DDS can pursue the mission of the WSA to collect data, despite the jamming or destruction of a part of the network, which leads to a partition of the WSA. This data can be then collected at a later point by a mobile sink or after re-establishment of the connectivity. The data might be useful for auditing accidents.



**Figure 11: Distributed data storage to pursue service despite jamming.**

Another application for the DDS is to distribute the logging files of an IDS middleware (developed in WP2). If the network is against attack, the logging can be used to audit the attack at a later point of time.

When the weather conditions are atrocious, the wireless channel can become prone to higher packet drops, and thus decreases the overall reliability of the WSA, despite having taking such events into account in the link budget before deployment. In that case, the QoS and energy manager will ensure that the network will regulate itself accordingly.



## 4 WSAN Service Integration to CI

Supervisory Control and Data Acquisition systems have developed from been a standalone and compartmentalized mainframe system to an open-protocol networked architecture that communicate data from several stations across large distances.

In addition, their implementations have migrated from proprietary hardware and software to standard hardware and software platforms, located in a corporate network using the well-known TCP/IP communication protocol, leading to reduce the development, operational and maintenance costs, as well as having the information in real-time, providing an executive management to support planning, supervision and decision making. The disadvantage of using a standard platform is the possibility of having intrusions, through external networks as well as internal personnel, using the well-known vulnerabilities in such standard platforms, as the OS Windows, and PCs used in the SCADA system nowadays.

In order to have a complete vision of SCADA systems, the following section presents the definition of their main components. Next, the main communication protocols for SCADA systems are outlined. Then, challenges and middleware issues related to SCADA systems interacting with wireless and actuator sensor networks (WSAN) are explained. Finally, different solutions for the integration between SCADA and WSAN are proposed.

### 4.1 SCADA System Components and Architecture

A SCADA system, related to the hardware components, consists of a number of remote terminal units (or RTUs) collecting field data and sending that data back to a master station via a communication system. The master station displays the acquired data and also allows the operator to perform remote control tasks.

These hardware components are included in the following groups:

- **Master Terminal Unit (MTU):** A master terminal unit (MTU) is equivalent to a master unit in master/slave architecture [22]. The MTU gathers data from the distant site, presents it to the operator through the HMI interface, and transmits control signals to the remote site. The transmission rate of data between the MTU and the remote site is relatively low and the control method is usually open loop because of possible time delays or data flow interruptions. This MTU component is substituted in almost every case by some RTUs or PLCs (defined below).
- **Remote Terminal Unit (RTU):** An RTU (Remote Terminal Unit) is a microprocessor controlled electronic device which provides intelligent I/O collection and processing, such as reading inputs from switches, sensors, and transmitters, and then arranging the representative data into a digital format that the supervisory system can understand. The RTU also converts and sends output electrical signals provided by the SCADA system from their digital form into that which can be understood by field-controllable devices such as discrete outputs (for opening or closing a switch or a valve) and analogue outputs (for setting the speed of a pump). The data rate between the RTU and controlled devices is relatively high and the control method is usually closed loop.
- **Programmable Logic Controller (PLC):** A Programmable Logic Controller (PLC) is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines. A PLC can either work with local physically connected inputs and outputs or with remote inputs and outputs provided by RTUs. A typical PLC can provide for two different types of control: discrete and continuous. A Programmable Logic Controller (PLC) differs from an RTU unit in that PLCs are more suitable for local area control (plants, production lines, etc.) where the system utilizes physical media for control while RTUs are more suitable for wide geographical telemetry, often using wireless communications.
- **Human Machine Interface (HMI):** The HMI (Human Machine Interface) is the means by which the user (operator) interacts with the SCADA system, providing a clear and easy-to-understand computer representation of what is, in fact, being controlled or monitored by the SCADA system. Further, it provides for interaction in a variety of formats, including graphics, schematics, windows, pull-down menus, touch-screens, and so on.

- **Hybrid controllers:** Hybrid controllers are specialized devices that provide for capabilities not found in standard discrete and continuous control modules for PLC systems. Capabilities such as adaptive control, artificial intelligence, and fuzzy logic are afforded by typical hybrid controllers.
- **Operator:** The human operator monitors the SCADA system and performs supervisory control functions for the remote plant operations.

The SCADA software can be divided into two types, proprietary or open software:

- On the proprietary software side, companies develop this specific software to communicate with their hardware, selling them as ‘turn key’ solutions. The main problem with these systems is the overwhelming reliance on the supplier of the system.
- On the other side, open software systems have gained popularity because of the interoperability they bring to the system. Interoperability is the ability to mix different manufacturers’ equipment on the same system. GeFanuc iFix [23], Citect [24], and WonderWare [25] are just three of the open software packages available on the market for SCADA systems. Some packages are now including asset management integrated within the SCADA.

## 4.2 SCADA Communications Protocols Overview

As the SCADA industry matured and vendors began to adopt open standards, the total number of SCADA protocols commonly in use was reduced to a small number of popular protocols that were being promoted by industry professional organizations. In the next sections, some of the most used protocols in the SCADA systems nowadays are presented.

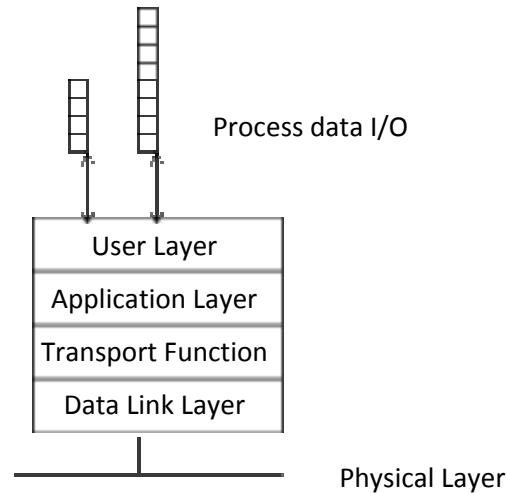
### 4.2.1 DNP3 (Distributed Network Protocol)

DNP3 [26] is a very simple and open client/server SCADA protocol that was originally designed for very low speed serial communication in process control networks, also adapted to Internet technologies by using an extra TCP/IP layer. DNP3 is most commonly used in electric utilities, but there is nothing specific to that industry in the protocol. The client, also referred to as the master, is typically a HMI or control server that issues DNP3 requests to a PLC, RTU, or other field device acting as a DNP3 server, also called the slave. Common request types include read requests, write requests, starting and stopping applications, freezing values to buffers, and a variety of administrative and diagnostic requests. It is widely used by utilities such as water and wastewater companies and electricity suppliers, in the transportation, oil and gas pipeline sectors, for the exchange of data and control instructions between master control stations and remote computers or controllers called outstations, which provides the master control station with information such as pressures, status of a circuit breaker, analogue signals representing such items as temperature or power, and information files.

Within the DNP3 protocol stack are different layers. The most important are the following ones, enumerated according to the path followed from having the process data I/O down to the serial/IP physical communication layer:

- The User Layer is the vendor’s applications code that inputs and outputs data at the master station and the outstation, from the DNP3 application layer.
- The Application Layer is where the monitoring and control functions take place. DNP3 requests and responses are constructed using function codes and objects defined in the DNP3 standard. A DNP3 frame comprises a header and data. The header comprises the source device address, the destination device address, frame size and data link control information. Some function codes in the header include reads, writes, administrative and diagnostic purposes.
- The transport function splits the application message into segments no longer than 250 bytes, and actually can be considered an addition to the Application Layer. On the receiving side, the transport function reassembles the packets into the application layer message.
- The Data Link Layer adds the addressing and error detection and possible correction, prior to sending the frame over the physical layer, with as long as 292 bytes. This error detection and correction are relatively robust compared to other control system protocols. A 16-bit CRC is sent

for the header and every 16 data bytes. The DNP3 Data Link layer also sends acknowledgements for certain data link layer frame types and can detect lost or duplicate frames.



**Figure 12 DNP3 Protocol Stack**

Figure 12 represents the serial communication stack which places the DNP3 Data Link Layer Frame directly on the physical layer. If DNP3 is sent over an IP network, the Data Link Frame is encapsulated in either a TCP or UDP packet. As it was mentioned before, DNP3 is a request/response protocol, but can also support unsolicited response, such as the response sent by an outstation to the master station when an object value exceeds a set threshold, without receiving a request.

In 2007, a specification of Secure DNP3 was released [27], adding user and device authentication as well as data integrity protection to the DNP3 protocol. Secure DNP3 is a bi-directional protocol that provides protection between master stations (HMI, control servers) and outstations (PLCs, RTUs, IEDs).

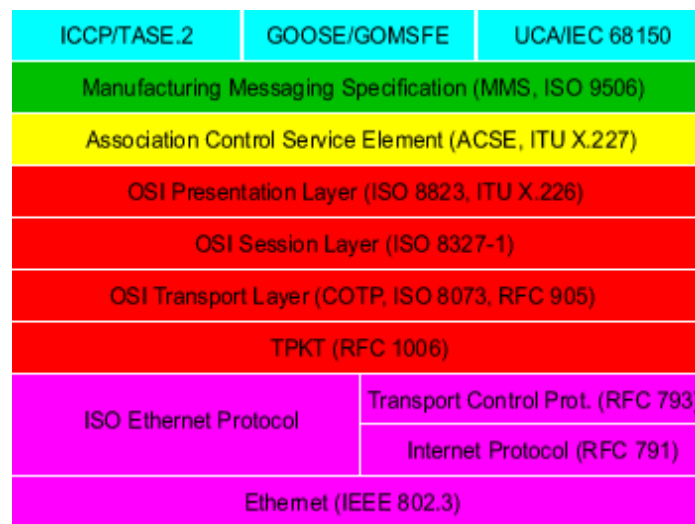
Secure DNP3 only modifies DNP3 protocol at the application layer, working the same whether it is DNP3 over TCP/IP, UDP/IP, or serial and DNP3 can be secured in hybrid networks such as TCP/IP to serial communications through a gateway.

#### 4.2.2 ICCP (Inter-Control Center Communications Protocol)

The Inter Control Center Protocol (ICCP) [28], also known as TASE.2, is an international standard commonly used in the electric industry for communication between energy control centers including communication between different companies. It was developed as an effort to define a more international robust standard than DNP3 to serve the SCADA needs for real-time data. This protocol is also used for communication between generation plants and energy management systems (EMS) within an organization or region, over local and wide area networks (LANs & WANs). Basic ICCP functionality is specified below:

- Transmit real-time data indications from SCADA systems.
- Remotely control other centers' assets and programs, allowing an ICCP client to remote control programs executing on an ICCP server.
- Monitor time-series data, allowing a client to request a report from a server of historical time series data between a start and end date.
- Additional User Objects, to manage scheduling and accounting info, outage and plant information.
- Provide event notification to a client, like error conditions and device state changes at a server.

ICCP protocol is roughly based on OSI protocol stack (see Figure 13), following the client / server principles. ICCP can operate over point-to-point links between control centers, configured as a mesh to provide redundancy, or operate over a typical router setup.



**Figure 13: ICCP server protocol stack**

The logical connection between each client-server is called “association”. There can be multiple associations between a client and more than one server and also a client may establish more than one association with the same server. Multiple associations with the same server can be established at different levels of quality of service so that high priority real-time data is not delayed by lower priority or non real-time data transfers. Associations are expected to remain unless an error occurs in the communication medium or on either end.

Commercial ICCP products are generally available for one of these three configurations:

- As a native protocol embedded in the SCADA host. The ICCP management tools and interfaces are all part of the complete suite of tools for the SCADA. This offers maximum performance, because of direct access to the database of the SCADA host, without requiring any intervening buffering.
- As a networked server, the ICCP is not restricted to the SCADA environment but is open to other systems, such as a separate data historian or other databases. Also the security is easier to manage, while the ICCP server is segregated from the operational real-time systems.
- As a gateway processor, the approach is similar to the networked server, except it is intended for legacy systems with minimal communications networking capability. The ICCP gateway processor can communicate with the SCADA host via a serial port in a similar manner to the SCADA RTUs.

### 4.2.3 MODBUS

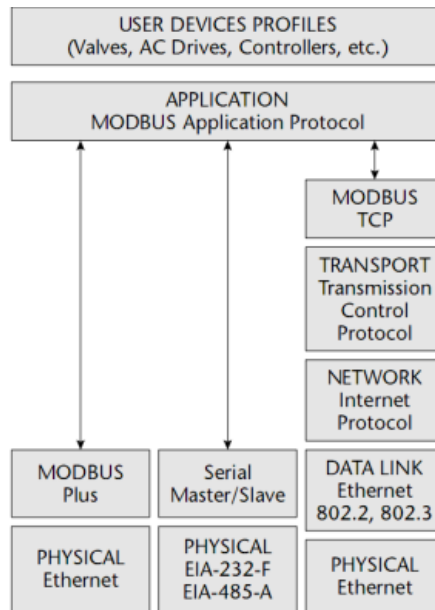
MODBUS [29] is a very simple client server protocol that was originally designed for very low speed serial communication in process control networks. MODBUS is used across many of the critical infrastructure industries including water, oil and gas, and manufacturing. The client is typically a HMI or control server that issues MODBUS requests to a PLC, RTU, or other field device acting as a MODBUS server. Common request types include read requests, write requests, and diagnostics.

Originally, MODBUS was used over serial communication. With the introduction of Ethernet interface cards in PLC's and other controllers, MODBUS was slightly modified to operate over TCP communications, calling this modification MODBUS TCP. Now there are also gateway products that convert MODBUS by serial port to MODBUS TCP and vice versa.

The MODBUS protocol is based on the master/slave principle, communicating one master and up to 247 slaves, with an initialization for transaction only from the master part only. These transactions are either of a query/response type where only a single slave is addressed, or a broadcast/no response type where all slaves are addressed.

The protocol provides frames for the transmission of messages between master and slaves. The information in the message includes the address of the controller to which the request is being directed, the function that the receiver must perform, the data needed to perform the requested function and a means of checking errors. The slave reads the messages, and if there is no error, it performs the task and sends a response back to the

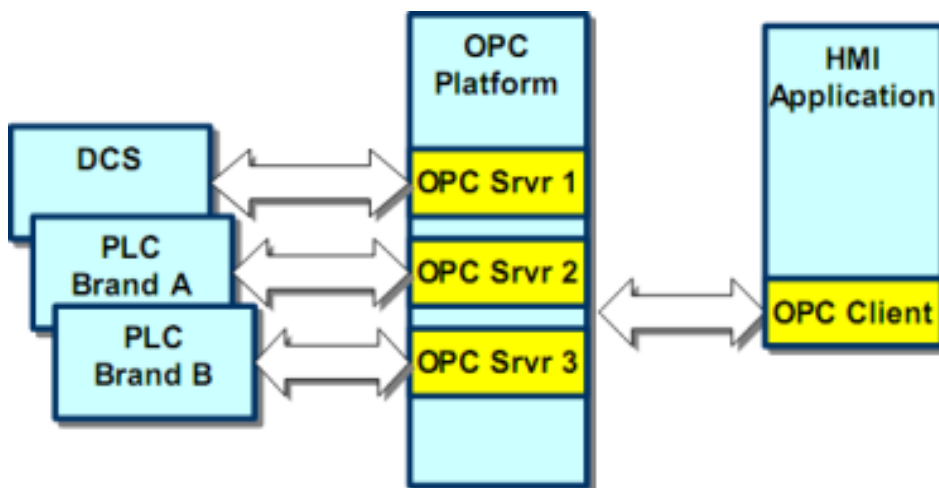
master. The information in the response message includes the responding device address (a field between 1 and 247), an echo with the action performed, the results of this action performed and a means of checking errors. If the initial message is of a broadcast type, there is no response from the slaves. The communications layers of the MODBUS protocol are given in Figure 14.



**Figure 14: MODBUS layers**

**4.2.4 OPC (OLE for Process Control)**

OLE for Process Control (OPC) [30] is a software interface technology used to facilitate the transfer of data between industrial control systems, Human Machine Interfaces (HMI), supervisory systems and enterprise systems such as historical databases. It was developed in response to the need for a standardized method for allowing different control systems to interface with each other, providing a common interface for communicating with diverse industrial control products, regardless of the software or hardware used in the process. OPC clients and servers run on Windows systems and require DCOM [31]. Figure 15 shows a common OPC client-server architecture.



**Figure 15: Common OPC client-server architecture**

An OPC client is an application that accesses data held by OPC servers. For example, an HMI package may contain an OPC client that allows it to access data provided by an OPC server application resident on another machine. The HMI package could also act as an OPC server, allowing other OPC clients to access the data it has aggregated either directly from field controller or from other OPC servers.

OPC is an Application Programming Interface (API), and not a communication protocol. For most developers using the OPC API, the underlying network transport or encoding used by the API to exchange data is irrelevant. The information available from an OPC server is organized into groups of related items for efficiency. These servers can contain multiple groups of items. There are two types of groups: public groups (available for access by any client) and local groups (only accessible by the client that created it).

Some of the most significant OPC Standards & APIs within the OPC specification are the following ones:

- **OPC Data Access (OPC-DA).** OPC-DA is primarily used to provide real-time access to process control and manufacturing data elements where each data element is called a point and has three attributes: value (the actual data being read or written), quality (how trustworthy the data is) and time stamps (time value for all the variables).
- **OPC Alarms & Events (OPC A&E).** Unlike DA, A&E supplies a value only when a specific event occurs. These values include process alarms, operator actions, informational messages, and tracking/auditing messages.
- **OPC Data Exchange (OPC-DX).** It is a set of interfaces that provide interoperable data exchange and server-to-server communications between devices and controllers connected to Ethernet networks using different protocols.
- **OPC Security.** This specification is designed to ensure that OPC Servers implement operating system security APIs in a consistent way. It defines three security levels (disabled, DCOM and OPC). Also, some background definitions of common security concepts are provided, such as authentication or authorization.
- **OPC Unified Architecture (OPC-UA).** OPC-UA integrates the functionality of previous specifications into a single integrated namespace, leaving behind COM/DCOM in favor of other transports mechanisms, SOAP and HTTP(S). This reflects Microsoft's goal of retiring DCOM in favor of .NET and movement towards Service Oriented Architectures (SOA), making much easier the development of OPC clients and servers on non-Microsoft platforms, using Java, Perl, Python, and other languages that support SOAP.

### 4.3 Challenges and Issues concerning SCADA-WSAN Systems

Integrating WSAN technology as a part of a SCADA system has been proposed as a way to apply the advantages WSANs offer to achieve higher flexibility and control over the system being monitored. WSANs exhibit some unique characteristics that can be used in the context of SCADA systems and that are described in the following items:

- **Dynamism:** WSAN are not static networks and allow connecting new nodes to the network and disconnecting or deactivating nodes deployed. This feature makes wireless sensor networks a flexible tool to monitor systems as its architecture can be tailored to the system changing needs. Furthermore, the lack of wiring between the devices allows node mobility and eases the task of changing the network topology.
- **Retrofit:** SCADA systems are usually deployed in a big scenario. That means that updating and changing the architecture once it has already been deployed is not simple. Reprogramming techniques can be used but since the location of the devices are static and normally those devices require wiring updates are costly. On the hand, diverse reprogramming techniques have been proposed in the context of WSAN that can be used to update the behaviour of the monitoring application deployed in the WSAN. Also, the dynamism WSANs provide (explained in the item above) makes this technology appropriate to changing applications.
- **Ease of installation:** WSANs can be easily deployed in the scenario since an energy source is not needed to power the nodes. Besides, the wireless capabilities they have also help to reduce the cost of deploying a WSAN.
- **Redundancy:** WSAN can be composed of thousand of nodes which cooperate to monitor a scenario. This high density of devices can be used to provide fault tolerance and QoS by exploiting

node redundancy. Nodes deployed in the same are going to receive the same readings. By cooperating between them they can decide which one is going to provide the readings instead of sending all of them. Additionally, aggregation techniques are suitable in this scenario. All these approaches save energy which is a crucial issue in this kind of technology.

In order to deploy the WSAN and communicating it with the SCADA a number of challenges must be addressed:

- Real-time communications and QoS: SCADA systems are used to monitor and control a wide variety of applications ranging from water treatment plants, oil refineries, electrical power transmission. These applications usually have QoS and real-time requirements that must be met in order to guarantee the correct functioning of the system. QoS and real-time support is unfortunately not directly provided by WSANs. What is more, WSANs communications are error-prone and sensitive to network congestion and packet loss. A middleware layer needs to be used to manage the communications in order to provide QoS and real-time communications in case the operating system does not directly support it. In the end, the goal is to get the information that corresponds to the current state of the system.
- Management support: A SCADA system does not only provide a way to monitor a system but also allows users to control and manage it. This means that both, from SCADA to WSAN and from WSAN to SCADA communication are going to be carried out. Therefore, the solution used to connect both systems must allow information exchange in both directions.
- Fault tolerance: Sensor nodes can run out of battery or fail. The system must detect this situation and the SCADA must show the corresponding information. Redundancy can be used to avoid the network to become useless as a result of a crucial node failure.
- Energy saving: As explained in the first part of the section, this technology offers many advantages in the context of monitoring systems and communicating with the SCADA system. However, a design that takes into account energy consumption as a main issue must be followed to avoid situations where manual replacement of the battery must be made.
- Security: With the transition of SCADA system protocols from proprietary to open standard protocols, those systems become exposed to the same vulnerabilities than conventional computers. Secure protocols need therefore to be developed to protect sensitive information and to guarantee the information confidentiality.

The protection of critical infrastructures is a main factor to be considered. Most of the current installed base of SCADA systems in use today utilize protocols that are either inherently insecure by design or that by themselves are not necessarily insecure but are poorly implemented by the SCADA product vendor, which results in SCADA insecurities.

Some typical high-level weaknesses found in SCADA systems that employ standard hardware and software include:

- SCADA does not require any authentication
- SCADA does not require any authorization
- SCADA does not use encryption
- SCADA does not properly handle errors and exceptions

These high-level weaknesses, listed above, open several potential attack vectors, which are listed here:

- Unauthorized disclosure of critical data
- Unauthorized modification and manipulation of critical data
- DoS or DDoS, (Distributed) Denial of service
- Address spoofing
- Unsolicited responses
- Session hijacking

- Protocol / packet fuzzing
- Unauthorized access to audit logs and modification of audit logs
- Unauthorized control

All of these attack vectors can lead to potentially disastrous results, such as:

- Altering or otherwise affecting the HMI display screen, which may cause the SCADA operator to take incorrect corrective actions.
- Permitting an unauthorized person to assume control of the SCADA system.
- Disrupting the process that is under the control of the SCADA system.

#### 4.4 Middleware Issues concerning SCADA-WSAN Systems

The challenges exposed in the previous section influence the design of the software solution used for the SCADA-WSAN integration. We propose a middleware based solution. A middleware helps to hide the complexity of software development for sensor networks. In this sense, aspects such as node configuration, routing, etc, are transparent for the application developer that simply uses an API to access the information provided by the nodes. In addition, the middleware provides a framework to ease the development of complex monitoring applications.

SCADA systems are usually based on fixed schemas for representing the information gathered from the monitored systems. However, wireless sensor networks can modify the way the information which is obtained from the sensors (activate or deactivate sensors in some nodes) and the way that this information is transmitted and aggregated. This allows the SCADA user to adapt many parameters from the sensing and actuation network depending of the situation of the Critical Infrastructure that is being monitored. The developed middleware should be able to face dynamic situations in the network. For instance, in case of emergency or breakdowns, many aspects of the network can be modified in order to improve the response of the systems. Some examples are the following:

- In an emergency situation, some of the sensor routines of the nodes can be disabled. For instance, humidity may not be a risk factor in the case of a severe breakdown in the communication infrastructure. Therefore, the system stops reporting humidity values, thus increasing the QoS for more important data.
- The way of representing the data in the user interface may depend also on the situation of the plant. In normal operational modes, knowing the medium value of some physical parameters in a concrete section of the plant or Critical Infrastructure can be enough for the control of the normal operation. This may lead to configure the network in order to save energy by deactivating some of the sensors, modifying the sampling frequency, etc. However, in the case of a transitory event or an accident, the user may decide to modify many of the parameters of the sensor network by using the API provided by the middleware:
  - Different strategies for data aggregation. In some cases, even the selection of some nodes may help to localize the problem and we need to know the data provided by some concrete sensors or to define new data aggregation schemes on run-time.
  - Modify the behaviour of specific nodes. Some nodes can be configured to monitor some critical variables, even if the power consumption is increased and the node may be out of battery in a short term.
  - Some idle nodes can be activated or even some new nodes can be deployed in the malfunctioning area. This can modify the network topology, the clustering of the nodes, etc. However, it may be that different protocols and algorithms have to be used to minimize the adaptation of the network to the new configuration. A long delay on self-configuration may lead to useless information in these critical situations.

- Reliability and QoS. The user may want to be sure that the data he is receiving in the SCADA is trustable. In this sense, some reliability and QoS parameters can be modified on the fly to maximize the confidence of the user of the data being received. For instance, the user may decide to increase the redundancy level in the nodes and in the communication network paths. Of course, this may lead the middleware to reconfigure itself to provide the information under the new constraints.

From these scenarios, it is clear that the SCADA system has to have access to “meta-information” about the sensor network. This meta-information will be used by the middleware. On normal functioning, the strategies to provide the information to the system are designed to minimized power consumption and to provided default QoS and reliability parameters. In the case of transient behaviour, the network should adapt itself to the transient scenarios. In addition, these scenarios cannot be foreseen in advance and the (expert) user interaction can provide

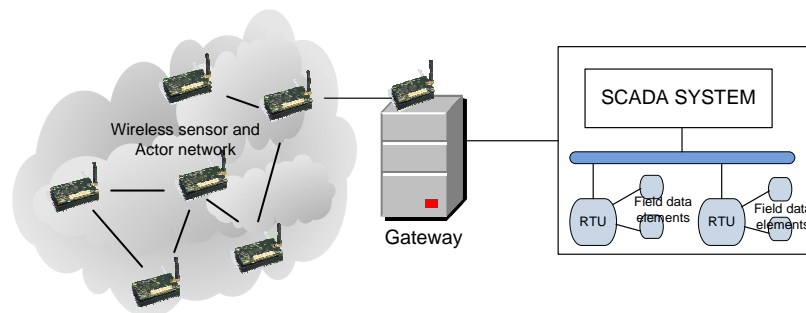
## 4.5 SCADA-WSAN Connection Approaches

In order to get the information from the sensors deployed in the system the SCADA system must be able to query the WSAN. In the same way, management functions need to be carried out to control and modify network parameters. Regardless of whether one solution or another is chosen, information must flow in both directions (from the WSAN to the SCADA and vice versa) and thus the architecture must allow this kind of interaction.

Two different architectures to connect the WSAN with the SCADA are proposed. The first, using an intermediate device in the form of a gateway to connect the SCADA to the WSAN and the second one that allows direct interaction between both sides.

### 4.5.1 Gateway Solution

The first solution considers the WSAN as an independent system. The connection between the wireless sensor network and the SCADA is achieved by using a gateway. The gateway is usually connected to the SCADA and the WSAN, and acts as an information transmitter and packet translator. The whole architecture is depicted in Figure 16.



**Figure 16: Gateway solution**

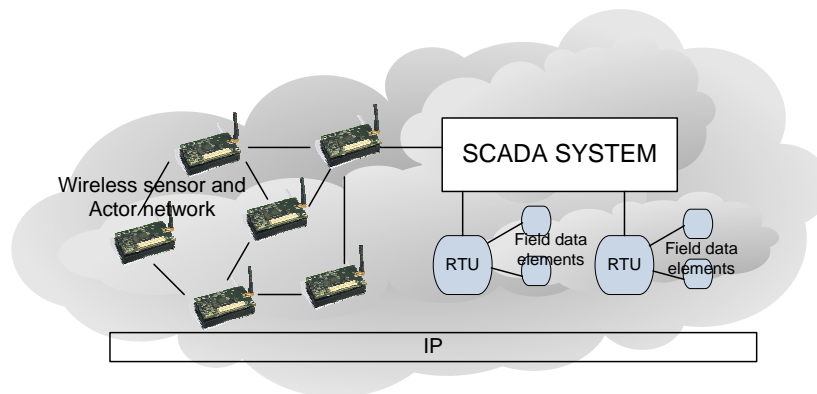
All queries sent to the WSAN by the SCADA are translated into packets that the WSAN can process. In addition, messages need to be translated from the SCADA system format to the WSAN format. This task is carried out by the gateway which needs to implement the logic to map every SCADA message whose destination is the WSAN into a message that the WSAN can understand. Additionally, the gateway need to contain a transceiver device to send information in the same protocol and medium that the WSAN uses and will act as what is known as “base station”. On the other hand, the nodes sending information to the SCADA need to route the messages through the WSAN to the gateway. The gateway will then translate the message and deliver it to the SCADA system.

The use of a gateway allows traffic between the two systems to be isolated from each other. Furthermore, the logic to communicate both systems resides in the gateway rather than in the SCADA system. In case a new WSAN needs to be installed a new gateway needs to be provided since it is where the packet translation takes place. The communication between the gateway device and the SCADA system can be carried out

using some middleware or distributed platform. The use of web services can be a good alternative because they are widely used nowadays and based on standards with several implementations, and finally they are platform independent in terms of operating system or programming language.

#### 4.5.2 IP Solution

The second solution makes use of the 6LowPAN protocol (IPv6 over Low power Wireless Personal Area Networks) [100]. This protocol proposes encapsulation and header compression mechanisms that allow IPv6 packets to be sent to and received from over IEEE 802.15.4 based networks. The architecture of this proposal is shown in Figure 17.



**Figure 17: IP Solution**

The main attractive of this proposal is the use of a standard protocol to communicate the WSAAN with the SCADA system. By using the same protocol both in the SCADA system and the WSAAN the gateway role is not needed since all devices can directly communicate between each other. Only a router with a 6lowpan adaptation layer is needed. There is no need for message mapping on the router. However, there is the need to decide which protocol to use over 6LowPAN at the application layer. In this work we propose the use of Web Services. Web services are defined as a software system designed to support interoperable machine-to-machine interaction over a network. It allows different systems to communicate in a platform independent manner.

However, web services must be handled in an appropriate way by WSAANs as they are composed of resource constrained devices. The use of HTTP needs to be made according to the principles defined in the software architecture technique REST. These key principles ensure that a good use of HTTP is made and so the system can be called “RESTful”. A RESTful web service (also called a RESTful web API) is a simple web service implemented using HTTP and the principles of REST. Such a web service can be thought about as a collection of resources.

The use of RESTful web services is justified as a simple and efficient way to process web services is required for the WSAAN side. In RESTful web services unlike SOAP-based web services, there is no official standard for RESTful web service. This is because REST is an architecture, unlike SOAP, which is a protocol. Even though REST is not a standard, a RESTful implementation such as the Web can use standards like HTTP, URL, XML, GIF, etc. IETF recently created a working group CoRE to integrate future low-power devices into a RESTful architecture.

The use of this architecture makes the SCADA system, which uses IP, able to directly communicate with the sensor nodes of the WSAAN. That eliminates the need of having a gateway device that acts as an intermediary. However, the logic to query and contact the different nodes must reside in the SCADA system application. In case new nodes or a new WSAAN needs to be installed, there is no need to add additional devices and they can be directly used by the SCADA system as all devices communicate using the same protocol. On the one hand the use of standards is also convenient for reasons of efficiency, simplicity, maintenance and economics. The downside of this approach is that the SCADA systems may be exposed to the same vulnerabilities and threats as conventional PCs.

### 4.5.3 Comparison between the two Proposals

Two different proposals have been carried out to connect the WSAN with the SCADA system. The first, using an intermediate device in the form of a gateway to connect the SCADA to the WSAN and the second one that allows directly interaction between both sides. Both approaches have advantages and disadvantages. In this section we justify the election of one of the two approaches for the project.

The implementation of the gateway solution is feasible with the current state of the technology. The gateway device can host a web service or another middleware technology for the connection between the SCADA and the WSAN. This way, the gateway acts as a kind of interface between the SCADA system and the WSAN which is represented by the gateway device. This solution improves the independence between the two parts of the system. If there are changes in the WSAN or updates, the only element which needs to be adapted is the gateway, reducing time and costs. On the other hand, the gateway device is itself a critical element in the architecture because any damage in this element (or attack) can avoid the communication between the SCADA and the WSAN. This can be partially avoided by using several gateway devices if needed. Another disadvantage is the cost of additional devices but it can be argued that the cost of these devices is not very high.

The second solution uses standards protocols for the communication between the SCADA system and the WSAN. These protocols have to be adapted to the special needs of the sensors which have very limited resources in terms of memory and CPU. The main advantage of this solution is the possibility of considering the sensors as “standard” elements which do not require any special protocol or intermediate devices. However, this approach can have some of the vulnerabilities and security problems existing in standard PCs. The final point to be considered for the election of one of the approaches is the current state of the technology. In this case, 6LowPAN or CoRE do not seem to be enough stable to be used in this project, the risk is too high that it may negatively impact the demonstrators that we planned. This reason motivates us to use the gateway solution for WSAN4CIP.

## 5 Conclusion

We presented the conceptual work achieved within WP4 Service Protection. We showed how various modules can help to increase the dependability of the system at different layers. In section 3.4, we exhibited how our work addresses the different tactics of a CI adversary.

More specifically, we showed elementary security services based on the uniqueness of the clock skews of nodes. From this, services such as fingerprinting or node temperature assessment can be derived. Then, we investigated distributed data storages based on coding, reviewed the related state of the art and also presented a mechanism to detect pollutions attacks in such distributed storages without the use of any cryptography. Also, power saving techniques for wireless sensor nodes were studied, in order to build a QoS and energy manager middleware, a cross-layer component that will regulate the QoS state of nodes across the network in an autonomous fashion.

Furthermore, the deliverable D4.1 has discussed the WSAN service integration to the CI. More concretely, SCADA system components and architecture and SCADA communication protocols have been described. For the latter protocols DNP3, ICCP, MODBUS and OPC have been highlighted. These topics have been discussed in Chapter 4. It also offered insights into challenges and issues concerning SCADA and WSAN systems, middleware issues and connection approaches. As an all IP solution is not stable enough and hence too risky for the success of our demonstrators, we opted for a gateway solution to integrate WSANs with the SCADA system.

## References

- [1] N. Pantazis, D. Vergados. "A Survey on Power Control Issues in Wireless Sensor Networks". *IEEE Communications Surveys & Tutorials*, volume 9, n.4, pp.86-107, 4<sup>th</sup> Quarter 2007.
- [2] Ramesh Rajagopalan, Pramod Varshney. "Data-Aggregation Techniques in Sensor Networks: A Survey". *IEEE Communications Surveys & Tutorials*, volume 8, n.4, pp.48-63, 4<sup>th</sup> Quarter 2006.
- [3] Mário Macedo, António Grilo, Mário Nunes, "Distributed Latency-Energy Minimization and interference avoidance in TDMA Wireless Sensor Networks". *Computer Networks*, volume 53(2009), pp.569-582, April 2009.
- [4] Linwei Niu, Gang Quan. "Energy-Aware Scheduling for Practical Mode Real-Time Systems with QoS Guarantee". *2009 WRI World Congress on Computer Science and Information Engineering*, Volume 3, March 31 2009-April 2 2009, pp.428-432.
- [5] Shanghong Peng, S. Yang, S. Gregori, Fengchun Tian. "An adaptive QoS and energy-aware routing algorithm for wireless sensor networks". *International Conference on Information and Automation, ICIA 2008*, 20-23 June 2008, pp.578-583.
- [6] A. Pourkabirian, A. Haghghat. "Energy-aware, delay-constrained routing in wireless sensor networks through genetic algorithm". *15<sup>th</sup> International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2007*. 27-29 September 2007, pp.1-5.
- [7] Yao Lan, Wen Wenjing, Gao Fuxiang. "A real-time and energy aware QoS routing protocol for Multimedia Wireless Sensor Networks". *7<sup>th</sup> World Congress on Intelligent Control and Automation, WCICA 2008*. 25-27 June 2008, pp.3321-3326.
- [8] "X-560 Embedded Intelligent Module Developer's Reference Guide", Silex Technology Inc, October 2008.
- [9] Ruiz L B, Siqueira I G, Oliverira L B. "Fault management in event-driven wireless sensor networks". In *Proc. the 7<sup>th</sup> ACM/IEEE Int. Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, Italy, 2004.
- [10] Xuanwen Luo, Ming Dong, Yinlun Huang. "On distributed fault-tolerant detection in wireless sensor networks". *IEEE Trans. Computers*, Jan. 2006, 55(1): 58-70.
- [11] Gregory Chockler, Murat Demirbas, Seth Gilbert, Calvin Newport. "A middleware framework for robust applications in wireless ad hoc networks". Technical Report, MIT Computer Science and Artificial Intelligence Laboratory Cambridge, MA 02139, USA. 2006.
- [12] Vana Kalogeraki. "Middleware for reliable real-time sensor data management". *Lecture Notes in Computer Science (LNCS) 4125*, Moro G *et al.* (eds.), Berlin/Heidelberg: Springer, 2007, pp.235-246.
- [13] Perrig A, Szewczyk R, Tygar J, Wen V, Culler D. "SPINS: Security protocols for sensor networks, wireless networks", *ACM Wireless Network*, 2002, 8: 521-534.
- [14] Jeffery Undercoffer *et al.* "On security for sensor networks". CADIP Research Symposium, Technical Presentations, Oct. 25-26, 2002, <http://www.cs.umbc.edu/cadip>.
- [15] C. Karlof, N. Sastry, and D. Wagner. "TinySec: Security for TinyOS". Presentation given at NEST group meeting, Nov. 21, 2002.
- [16] Felemban E *et al.*, "Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks". In *Proc. the IEEE INFOCOM*, Miami, March 13-17, 2005, Vol.4, pp. 2646-2657.
- [17] Stankovic J A *et al.*, "Real-time communication and coordination in embedded sensor networks". In *Proc. IEEE*, 2003, 91(7): 1002-1022.
- [18] Luca Caviglione, Franco Davoli. "Peer-to-peer middleware for bandwidth allocation in sensor networks". *IEEE Communication Letters*, February 2005, 9(3): 285-287.
- [19] Younis M *et al.* "On handling QoS traffic in wireless sensor networks". In *Proc. the 37th Hawaii Int. Conf. System Sciences*, 2004, 40(8): 102-116.

- [20] J. Giroa, D. Westhoff, E. Mykletun, and T. Araki, "TinyPEDS: Tiny Persistent Encrypted Data Storage in Asynchronous Wireless Sensor Networks", *Elsevier Ad Hoc Journal*, Vol. 5, Issue 7, pp. 1073-1089, September 2007.
- [21] C. Jardak, J. Riihijärvi, and P. Mähönen, "Analyzing the Optimal Use of Bloom Filters in Wireless Sensor Networks Storing Replicas", *Proceedings of the IEEE Wireless Communications and Networking Conference - WCNC'09*, Budapest, Hungary, April 2009.
- [22] [http://en.wikipedia.org/wiki/Master-slave\\_\(technology\)](http://en.wikipedia.org/wiki/Master-slave_(technology))
- [23] GeFanuc iFix: <http://www.gefanuc.com/products/3311>
- [24] Citect: <http://www.citect.com/>
- [25] WonderWare: <http://global.wonderware.com/EN/Pages/default.aspx>
- [26] Overview of the DNP3 Protocol: <http://www.dnp.org/About/Default.aspx>
- [27] Secure DNP3 Protocol: [http://www.digitalbond.com/wiki/index.php/Secure\\_DNP3](http://www.digitalbond.com/wiki/index.php/Secure_DNP3)
- [28] Inter Control Center Protocol SISCO Technical Information: <http://www.sisconet.com/techinfo.htm>
- [29] MODBUS Specifications: <http://www.modbus.org/specs.php>
- [30] The OPC Foundation: <http://www.opcfoundation.org/>
- [31] DCOM Specification: [http://msdn.microsoft.com/es-es/library/cc201989\(en-us,PROT.10\).aspx](http://msdn.microsoft.com/es-es/library/cc201989(en-us,PROT.10).aspx)
- [32] Deliverable D1.1, WSAN4CIP
- [33] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Distributed data storage in sensor networks using decentralized erasure codes," in *Proceedings of the Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, November 2004.
- [34] —, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," in *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*. Piscataway, NJ, USA: IEEE Press, 2005, p. 15.
- [35] Dimakis, A. G., Prabhakaran, V., and Ramchandran, K, "Distributed fountain codes for networked storage," in *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, 2006.
- [36] C. Fragouli, J.-Y. L. Boudec, and J. Widmer, "Network coding: an instant primer," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, 2006.
- [37] T. Ho, B. Leong, R. Kötter, M. Medard, M. Effros, and D. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *Proceedings of the 2004 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2004.
- [38] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard, "Resilient network coding in the presence of byzantine adversaries," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Anchorage, Alaska, USA, 2007, pp. 616–624.
- [39] R. Kötter and F. R. Kschischang, "Coding for errors and erasures in random network coding," in *Proceedings of the IEEE Information Theory Symposium (ISIT)*, June 2007.
- [40] Ralf Kötter, Frank R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Transactions on Information Theory*, vol. 54, no. 8, pp. 3579-3591, Aug. 2008.
- [41] M. N. Krohn, M. J. Freedman, and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *Proceedings of 2004 IEEE Symposium on Security and Privacy*, 2004, pp. 226–240.
- [42] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 4, March 2005, pp. 2235–2245.
- [43] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2006.
- [44] K. E. Lauter, D. Charles X, and K. Jain, "Signatures for network coding," in *Proceedings of the 40th Annual Conference on Information Sciences and Systems (CISS '06)*, Mar. 2006.

- [45] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, "An efficient signaturebased scheme for securing network coding against pollution attacks," in *Proceedings of the IEEE INFOCOM '08 Conference*, 2008.
- [46] F. Zhao, T. Kalker, M. Medard, and K. J. Han, "Signatures for content distribution with network coding," in *Proceedings of 2007 IEEE International Symposium on Information Theory (ISIT '07)*, Jun. 2007.
- [47] Ren W., Ren Y., Zhang, H., "HybridS: A Scheme for Secure Distributed Data Storage in WSNs", EUC (2) IEEE Computer Society, 2008, pp. 318-323
- [48] Q. Wang, K. Ren, W.L., Zhang, Y, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance", in *Proceedings INFOCOM 2009. 28th Annual Joint Conference of the IEEE Computer and Communications Societies*. IEEE, 2009
- [49] Dimakis A.G., Prabhakaran V., Ramchandran K., "Decentralized erasure codes for distributed networked storage", *IEEE/ACM Trans. Netw.*, IEEE Press, 2006, Vol. 14(SI), pp. 2809-2816
- [50] Dimakis A.G., Godfrey P.B., Wainwright, M.J., Ramchandran K., "Network Coding for Distributed Storage Systems", in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*. IEEE, 2007, pp. 2000-2008
- [51] Kamra A., Misra V., Feldman J., Rubenstein D., "Growth codes: maximizing sensor network data persistence", *SIGCOMM Comput. Commun. Rev.*, ACM, 2006, Vol. 36(4), pp. 255-266
- [52] S.A. Aly, Z. Kong, E. Soljanin, "Fountain Codes Based Distributed Storage Algorithms for Large-scale Wireless Sensor Networks" in *CoRR*, 2009, Vol. abs/0902.1278
- [53] S.A. Aly, Z. Kong, E. Soljanin, "Raptor Codes Based Distributed Storage Algorithms for Wireless Sensor Networks", in *CoRR*, 2009, Vol. abs/0903.0445
- [54] S.A. Aly, Z. Kong, E. Soljanin, "Fountain Codes Based Distributed Storage Algorithms for Large-Scale Wireless Sensor Networks" , in *IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks*, IEEE Computer Society, 2008, pp. 171-182
- [55] Z. Kong, S.A. Aly, E. Soljanin, E., "Decentralized Coding Algorithms for Distributed Storage in Wireless Sensor Networks", in *CoRR*, 2009, Vol. abs/0904.4057
- [56] Kamra A., Feldman J., Misra V., Rubenstein D., "Data persistence in sensor networks: towards optimal encoding for data recovery in partial network failures", in *SIGMETRICS Perform. Eval. Rev.*, ACM, 2005, Vol. 33(2), pp. 24-26
- [57] Lin Y., Li B., Liang B., "Differentiated Data Persistence with Priority Random Linear Codes", in *ICDCS*, 2007, pp. 47
- [58] Diao Y., Ganesan D., Mathur G., Shenoy P.J., "Rethinking Data Management for Storage-centric Sensor Networks", in *CIDR*, 2007, pp. 22-31
- [59] Mathur G., Desnoyers P., Ganesan D., Shenoy P., "Capsule: an energy-optimized object storage system for memory-constrained sensor devices", In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*, ACM, 2006, pp. 195-208
- [60] P. Desnoyers, D. Ganesan, and P. Shenoy, "TSAR: a two tier sensor storage architecture using interval skip graphs", in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 39-50, ACM Press, 2005.
- [61] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks", *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122-173, 2005.
- [62] J. Hill, P. Levis, S. Madden, A. Woo, J. Polastre, C. Whitehouse, R. Szewczyk, C. Sharp, D. Gay, M. Welsh, D. Culler, and E. Brewer, "TinyOS: <http://www.tinyos.net>", December 2005.
- [63] A. Demers, J. Gehrke, R. Rajaraman, N. Trigoni, and Y. Yao, "The cougar project: a work-in-progress report", *SIGMOD Rec.*, vol. 32, no. 4, pp. 53-59, 2003.
- [64] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with ght, a geographic hash table", *Mobile Networks and Applications*, vol. 8, pp. 427-442, 2003.

- [65] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks", in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 56-67, ACM Press, 2000.
- [66] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy, and S. Shenker, "DIFS: A distributed index for features in sensor networks", *Elsevier Journal of Ad Hoc Networks*, 2003.
- [67] X. Li, Y. J. Kim, R. Govindan, and W. Hong, "Multi-dimensional range queries in sensor networks" in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 63-75, ACM Press, 2003.
- [68] D. Ganesan, D. Estrin, and J. Heidemann, "DIMENSIONS: why do we need a new data handling architecture for sensor networks?", *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 143-148, 2003.
- [69] S. Tilak, N. B. Abu-Ghazaleh, and W. R. Heinzelman, "Collaborative storage management in sensor networks", *CoRR*, vol. cs.NI/0408020, 2004.
- [70] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks.", in *VLDB*, pp. 588-599, 2004.
- [71] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler, "Hood: a neighborhood abstraction for sensor networks", in *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, (New York, NY, USA), pp. 99-110, ACM Press, 2004.
- [72] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks", 2004.
- [73] P. Levis and D. Culler, "Mate: A tiny virtual machine for sensor networks", in *International Conference on Architectural Support for Programming Languages and Operating Systems*, San Jose, CA, USA, Oct. 2002.
- [74] R. G. C. Intanagonwiwat and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", in *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCON)*, 2000.
- [75] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks", in *MobiQuitous*, pp. 109-117, IEEE Computer Society, 2005.
- [76] Crossbow, "Crossbow technology inc. wireless sensor networks"
- [77] L. Buttyan and P. Schaffer, "Panel: Position-based aggregator node election in wireless sensor networks", in *In Proceedings of the 4th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS) (O. 8-11, ed.)*, (Pisa, Italy), IEEE Press, 2007.
- [78] J. Giroa, D. Westhoff, and M. Schneider, "CDA: Concealed data aggregation for reverse multicast traffic in wireless sensor networks", *40th International Conference on Communications*, IEEE ICC 2005, May 2005.
- [79] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale", in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 81-94, ACM Press, 2004.
- [80] O. Ugus, A. Hessler, and D. Westhoff, "Performance of additive homomorphic EC-ElGamal encryption for tinyPEDS", in *6. GI/ITG KuVS Fachgesprch Drahtlose Sensornetze*, 2007.
- [81] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "The TESLA broadcast authentication protocol", *RSA CryptoBytes*, vol. 5, no. Summer, 2002.
- [82] Lin Y., Liang B., Li B., "Data Persistence in Large-Scale Sensor Networks with Decentralized Fountain Codes", in *INFOCOM*, 2007, pp. 1658-1666
- [83] Buttyán, L., Schaffer, P., and Vajda, I.,. "RANBAR: RANSAC-based resilient aggregation in sensor networks". In *Proceedings of the Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks SASN '06*. ACM, New York, 2006

- [84] Uddin B. and Castelluccia, C. “Toward Clock Skew Based Services in Wireless Sensor Networks”, *Int. J. Sensor Networks*, 2010 (to appear).
- [85] Kohno et al., “Remote physical device fingerprinting”, *Proceedings of IEEE Symposium on Security and Privacy*, 211–225, 2005.
- [86] Murdoch, S. J., “Hot or not: Revealing hidden services by their clock skew”, *Proceedings of the 9th ACM Conference on Computer and Communications Security(CCS’06)*, 27–36. October 2006.
- [87] Martinec M., “Temperature dependency of a quartz oscillator”, <http://www.ijs.si/time/#temp-dependency>.
- [88] Zander, S., and Murdoch, S. J., “An Improved Clock-skew Measurement Technique for Revealing Hidden Services”, *17th USENIX Security Symposium*, July 2008.
- [89] Moon, S. B., Skelly, P., and Towsley, D.. “Estimation and removal of clock skew from network delay measurements”, in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Mars 1999.
- [90] Dyer, M. E. “Linear time algorithms for two- and three variable linear programs”, *SIAM Journal on Computing*, 31–45, 1983.
- [91] Megiddo, N. “Linear-time algorithms for linear programming in  $r_3$  and related problems”, *SIAM Journal on Computing*, 759–776. Novembre 1983.
- [92] Cherifa Boucetta, Mohamed Ali Kaafar, “Detecting and Defending Against Wormhole Attacks by Securing Neighbor Discovery in Wireless Sensor Networks”, *In preparation*.
- [93] Y-C. Hu, A. Perrig, and D. B. Johnson, “Packet leashes: A defense against wormhole attacks in wireless networks”, in *INFOCOM*, 2003.
- [94] L. Hu and D. Evans, “Using directional antennas to prevent wormhole attacks”, in *NDSS. The Internet Society*, 2004.
- [95] S. Capkun, L. Buttyan, and J-P. Hubaux, “Sector: secure tracking of node encounters in multi-hop wireless networks”, in *SASN*. S. Setia and V. Swarup, Eds. ACM, 2003, pp. 21-32
- [96] R.Maheshwari, J.Gao, and S.R.Das, “Detecting wormhole attacks in wireless networks using connectivity information”, In *IEEE Conference on Computer Communications INFOCOM*, 2007.
- [97] W. Wang and B. Bhargava, ”Visualization of wormholes in sensor networks”, In *Proceedings of the ACM Workshop on Wireless Security*, 2004
- [98] H. Chen, W. Lou, “Consistency-based Secure Localization Scheme Against Wormhole Attacks”, In *Proceedings of the 4<sup>th</sup> International Conference on Wireless Algorithms, Systems, and Applications*, pp. 368 - 377, 2009.
- [99] P. Langendoerfer, S. Peter, K. Piotrowski, R. Nunes, A. Casaca, “A Middleware Approach to Configure Security in WSN”, *1st ERCIM Workshop on eMobility*, Coimbra, Portugal in conjunction with WWIC 2007, May 2007.
- [100] “IETF 6lowpan working group”, <http://datatracker.ietf.org/wg/6lowpan/>
- [101] Gay, D. The Matchbox File System. Available online: <http://webs.cs.berkeley.edu/tos/tinyos-1.x/doc/matchbox-design.pdf>
- [102] Piotrowski, K., Langendoerfer, P., Peter, S. “tinyDSM: A highly reliable cooperative data storage for Wireless Sensor Networks”, In *Proceedings of the 2009 international Symposium on Collaborative Technologies and Systems* (May 18 - 22, 2009). CTS. IEEE Computer Society, Washington, DC, 225-232. 2009