



Pilot SHIELD

pilot embedded Systems
archItecturE for multi-Layer Dependable solutions



Project no: 100204

p-SHIELD

pilot embedded Systems architecture for multi-Layer Dependable solutions

Instrument type: Capability Project

Priority name: Embedded Systems (including RAILWAYS)

D6.2.1: Platform validation and verification

Due date of deliverable: M18 (30th November 2011)

Actual submission date: M17 (11th January 2011)

Start date of project: 1st June 2010

Duration: 19 months

Organisation name of lead contractor for this deliverable: pSHIELD Consortium

Revision [Issue 1]

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2012)		
Dissemination Level		
PU	Public	
PP	Restricted to other programme participants (including the Commission	
RE	Restricted to a group specified by the consortium (including the Commission	X
CO	Confidential, only for members of the consortium (including the Commission Services)	



Pilot SHIELD

pilot embedded Systems
architecturE for multi-Layer Dependable solutions



SEVEN FRAMEWORK
PROGRAMME

Document Authors and Approvals

Authors		Date	Signature
Name	Company		
Andrea Morgagni	Selex Elsag		
Renato Baldelli	Selex Elsag		
Andrea Fiaschetti	Università degli studi di Roma "La Sapienza"		
Vincenzo Suraci	Università degli studi di Roma "La Sapienza"		
Sergio Wanderlingh	Tecnologie nelle Reti e nei Sistemi		
Pedro Miguel Rendeiro de Oliveira	Critical Software		
Mariana Esposito	Ansaldo STS		
Reviewed by			
Name	Company		
Approved by			
Name	Company		

Modification History

Issue	Date (DD/MM/YY)	Description
Draft A	26/10/2011	First issue for comments
Draft B	25/11/ 2011	Incorporate comments from Draft A review
Draft C	23/12/ 2011	Incorporate comments from Draft B review
Issue 1	11/01/ 2012	Incorporate comments from Draft C review



Contents

1	Executive summary.....	8
2	Introduction	9
3	Terms and Definitions	10
4	pSHIELD Scenario Platform.....	12
4.1	Introduction	12
4.2	Platform definition	12
4.3	SPD functionalities definition	16
4.3.1	Class AU – SPD Audit	16
4.3.2	CS – Cryptographic Support	17
4.3.3	Class IA – Identification and authentication.....	18
4.3.4	Class PT – Protection of the SPD functionalities.....	19
4.3.5	Class SM – SPD functions Management.....	19
5	Platform validation process.....	21
5.1	pSHIELD Ontology.....	21
5.1.1	Validation against model representation.....	21
5.1.2	Validation against SPD Composition.....	22
5.2	Middleware and Overlay for discovery and composition	24
5.2.1	Validation against discovery.....	24
5.2.2	Validation against composition	26
5.2.3	Validation against orchestration	26
5.2.4	Validation against the SPD level change	27
5.3	Heterogeneous Wireless Sensors Networks and secure communication	29
5.3.1	General Functionality	29
5.3.2	WSN Cryptography levels description	29
5.3.3	WSN key exchange.....	29
5.3.4	Communication mechanism.....	31
6	Platform verification process	36
6.1	Tests identification and description	36
6.1.1	Test identification.....	36



Pilot SHIELD

pilot embedded Systems
architecturE for multi-Layer Dependable solutions



SEVEN FRAMEWORK
PROGRAMME

6.1.2	Testbed definition	37
6.1.3	Testbed devices initialization.....	39
6.2	Tests.....	41
6.2.1	Audit tests.....	41
6.2.2	CS – Cryptographic Support.....	47
6.2.3	Identification and authentication tests.....	58
6.2.4	Protection of the SPD functionalities tests	61
6.2.5	SPD Functions Management Tests.....	63
6.3	Completeness of tests.....	66
7	Conclusions	67



Pilot SHIELD

pilot embedded Systems
archItecturE for multi-Layer Dependable solutions



SEVEN FRAMEWORK
PROGRAMME

Figures

Figure 4 - 1: Platform structure	13
Figure 4 - 2: Central Unit representation	13
Figure 4 - 3: Real time system clock synchronization process	14
Figure 4 - 4: Crossbow TelosB node.....	15
Figure 5 - 1: pSHIELD Ontology	21
Figure 5 - 2: pSHIELD scenario platform modelling	22
Figure 5 - 3: pSHIELD scenario platform system tree representation	23
Figure 5 - 4: Details of the Discovery core SPD service	25
Figure 5 - 5: Closed-loop SPD level control.....	27
Figure 5 - 6: semantic model of the SPD functionalities	28
Figure 5 - 7: validation results of against the SPD level change	28
Figure 5 - 8 : Exchange public keys protocol between Power and Sensor Node	30
Figure 5 - 9: Example of messages exchanged between Power and Sensor nodes.....	31
Figure 5 - 10: Power node behaviour (UART -> Wireless).....	32
Figure 5 - 11: Power node behaviour (Wireless-> UART)	33
Figure 5 - 12: Sensor node behaviour represented as a states machine	34
Figure 5 - 13: Sensor node behaviour (From Power Node requests).....	35
Figure 6 - 1: Testbed representation.....	38



Pilot SHIELD

pilot embedded Systems
architecturE for multi-Layer Dependable solutions



SEVEN FRAMEWORK
PROGRAMME

Tables

Table 5 - 1 Computable configurations generated by the reasoner	24
Table 6 - 1 Cross-reference verification	66

Glossary

AES	Advanced Encryption Standard
ECDH	Elliptic curve Diffie–Hellman
ESs	Embedded Systems
FFD	Full-Function Device
KF	Knopflerfish
PAN	Personal Area Network
SPD	Security Privacy Dependability
WSN	Wireless Sensor Network



Pilot SHIELD

pilot embedded Systems
archItecturE for multi-Layer Dependable solutions



SEVEN FRAMEWORK
PROGRAMME

This Page is intentionally left blank

1 Executive summary

The purpose of this task is to validate and verify the SPD features and concept integrated on an identified platform for pSHIELD scenario.

The introduction of SPD driven Semantics (developed with the target to address the interoperability among different SPD technologies and defined in task 5.1) has solved the challenging problem of the integrated prototypes features (considering the heterogeneous environment in which the different modules/components have been developed).

The validation and verification methodology has been defined on the basis of evaluation process defined in task 2.2 (multi-technology SPD metrics) which through the Common Criteria approach builds the basis to provide metrics for the integration and test of the specific components/modules which are implemented for demonstration purposes (target of task 6.3).

The structure and content of the document are the following:

- Chapter 1 – purpose of the document and its structure
- Chapter 2 – brief introduction
- Chapter 3 – taxonomy
- Chapter 4 – definition and description of platform for pSHIELD scenario
- Chapter 5 – platform for pSHIELD scenario validation process
- Chapter 6 – platform for pSHIELD scenario verification process
- Chapter 7 – validation and verification processes conclusions

2 Introduction

The pSHIELD activities have led to the development of a set of technological improvements in the different fields: node, network and middleware. These improvements, in a reduced but significant subset, have been implemented into the following prototypes:

- *pSHIELD Ontology*
- *Middleware and Overlay for discovery and composition*
- *Cognitive Radio Node Prototype*
- *FPGA Power Node Prototype*
- *Innovative key Exchange protocol*
- *heterogeneous Wireless Sensors Networks and secure communication*

These prototypes constitute the pSHIELD Technology Demonstrators described in deliverable D6.3.

Subsequently, some of these elements have been further integrated into a common platform specifically tailored on the railway scenario, that constitutes the pSHIELD Integrated Demonstrator.

This integrated platform will be validated and verified according to specific methodologies. In particular:

- The objective of the platform validation activity is to check the consistency of the platform components in terms of functionalities, semantic models (e.g. metrics and descriptions) and interfaces necessary to perform the SPD composability.
- The objective of the platform verification activity is to verify the platform behavior with respect to the selected scenario by means of focused functional tests.

3 Terms and Definitions

Audit	Involves recognizing, recording, storing, and analyzing information related to SPD relevant activities. The resulting audit records can be examined to determine which SPD relevant activities took place.
Authorised User	A user who possesses the rights and/or privileges necessary to perform an operation
Class and Family	The CC has organised the components into hierarchical structures: Classes consisting of Families consisting of components. This organisation into a hierarchy of class - family - component - element is provided to assist consumers, developers and evaluators in locating specific components
Common Criteria	The Common Criteria for Information Technology Security Evaluation (abbreviated as Common Criteria or CC) is an international standard (ISO/IEC 15408) for computer security certification. It is currently in version 3.1. Common Criteria is a framework in which computer system users can specify their security functional and assurance requirements, vendors can then implement and/or make claims about the security attributes of their products, and testing laboratories can evaluate the products to determine if they actually meet the claims. In other words, Common Criteria provides assurance that the process of specification, implementation and evaluation of a computer security product has been conducted in a rigorous and standard manner.
Composability	Is the possibility to compose different (possibly heterogeneous) SPD functionalities (also referred to as SPD components) aiming at achieving in the considered system of Embedded System Devices a target SPD level which satisfies the requirements of the considered scenario.
Cryptographic Algorithms	Algorithms to hiding the information, to provide security and information protection against different forms of attacks
Discovery	Provide to the pSHIELD Middleware Adapter the information, raw data, description of available hardware resources and services in order to allow the system composability
Life-Cycle support elements	It is the set of elements that support the aspect of establishing discipline and control in the system refinement processes during its development and maintenance. In the system life-cycle it is distinguished whether it is under the responsibility of the developer or

RE

	the user rather than whether it is located in the development or user environment. The point of transition is the moment where the system is handed over to the user.
Overlay Layer	The “embedded intelligence” that drives the composition of the pSHIELD components in order to meet the desired level of SPD. This is a software layer as well.
Personal Area Network	It is a computer network used for communication among computer devices, including telephones and personal digital assistants, in proximity to an individual's body.
PAN Coordinator	The ZigBee device which is responsible for starting the formation of a ZigBee network. The ZigBee PAN coordinator chooses the PAN ID. There is only one ZigBee PAN Coordinator in any ZigBee network; it's ZigBee address is always 0.
TinyOS	This operating system (OS) is a free and open source operating system and platform that is designed for WSNs.
User session	A period of interaction between users and SPD functional components.
Wireless Sensor Network	It consists of spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants and to cooperatively pass their data through the network to a main location.

RE

4 pSHIELD Scenario Platform

4.1 Introduction

pSHIELD project aim to validate and verify the SHIELD integrated system by means of an application scenario.

In particular the chosen application scenario is the “monitoring of freight trains transporting hazardous material”.

In the previously mentioned meaningful scenario the developed SPD functionalities will be integrated in a complete platform which will be validated and verified.

In this paragraph the platform and the integrated SPD functionalities are defined.

4.2 Platform definition

The hypothesized platform is composed by a central unit connected by means of a ciphered wireless network to remote sensors. These components are supplied with the related configuration manuals.

In this platform the assets to protect are data sent by remote sensors to central unit, where data are recorded inside the central unit itself.

Platform logical structure can be represented graphically as depicted in the following figure:

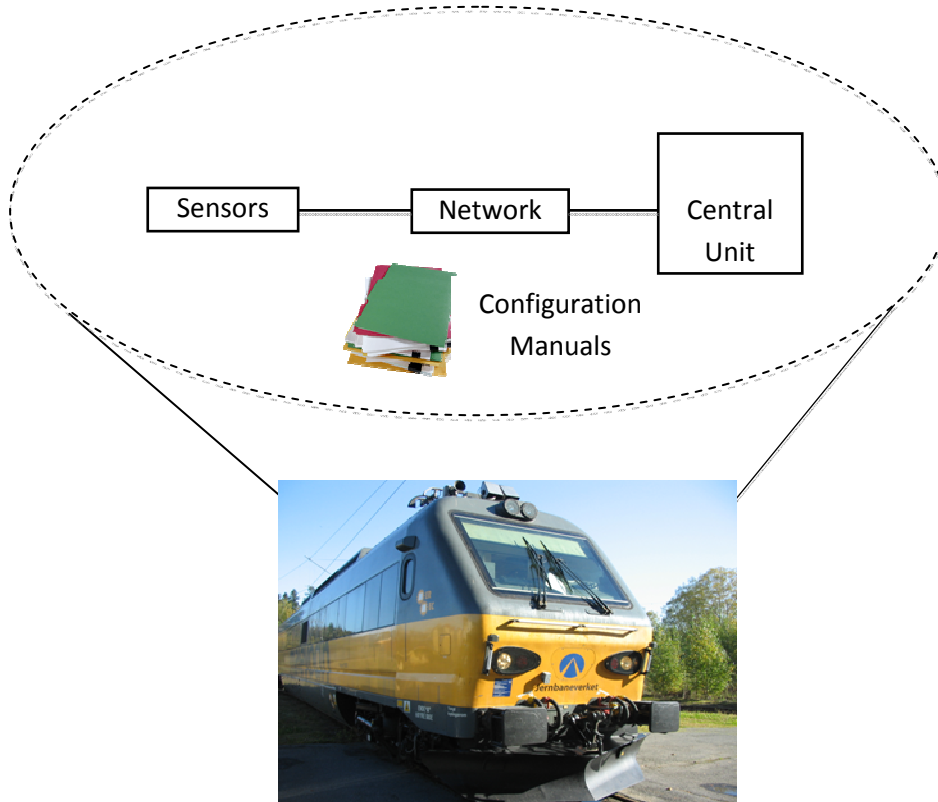


Figure 4 - 1: Platform structure

The detailed composition of the platform components is the following:

1. Central unit:

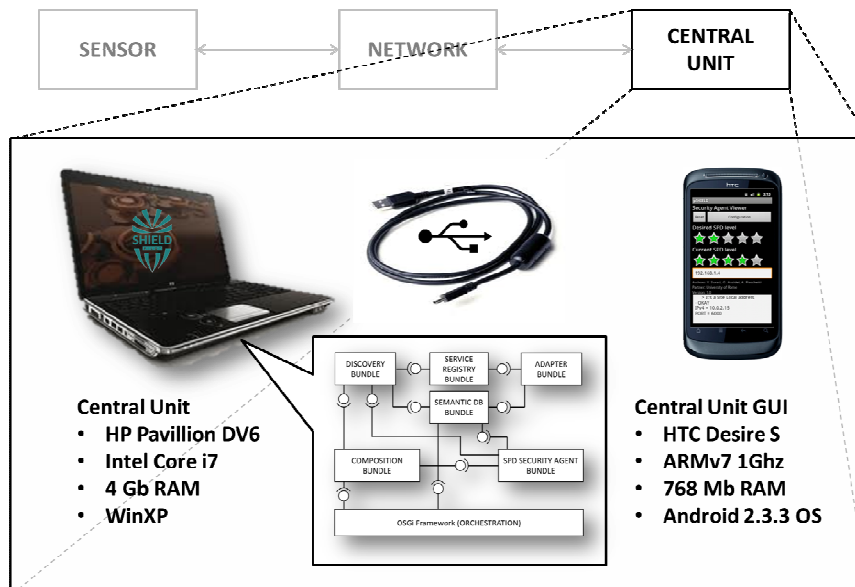


Figure 4 - 2: Central Unit representation

The above figure represents the prototype reference schema, where the Central Unit box has been expanded to show the hardware components used in the testbed. It is composed by a Central Unit machine and an USB connected mobile terminal.

Due to the need of having a high performance system, the Central Unit machine is a HP Pavillion DV6 series, equipped with a top gamma Intel Core I7 quad processor with 4 Gbyte DDR3 RAM and the Operating System is Microsoft Windows XP. The OSGi environment runs over the JVM (version 1.6) that has been installed in the Central Unit. The Knopflerfish Java-based implementation of the OSGi has been used. The whole pShield middleware has been made to run into the same OSGi container.

The Central Unit real time system clock has been synchronized via the USB connection with the external GSM device (Fig 4-3).

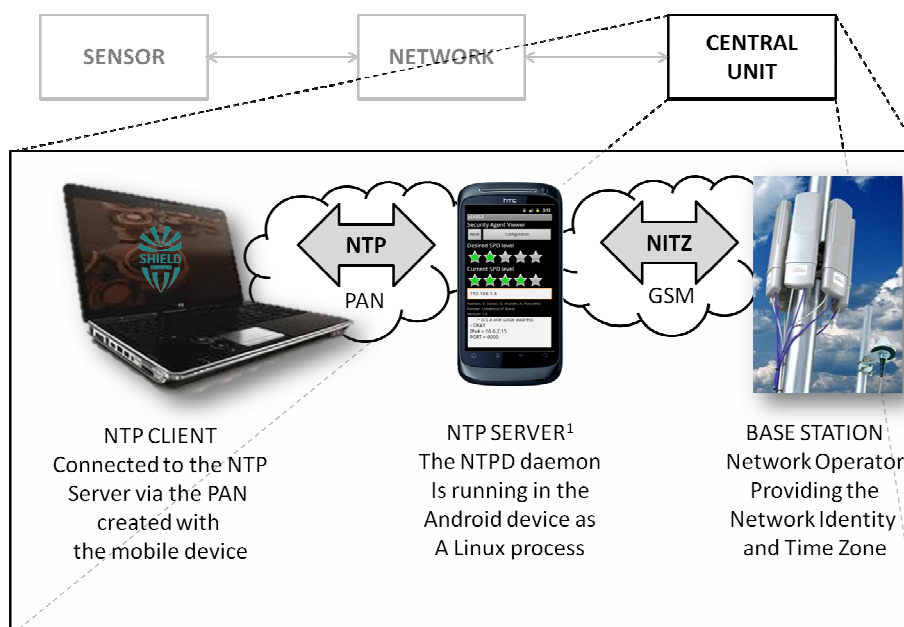


Figure 4 - 3: Real time system clock synchronization process¹

As shown in the above figure, once the mobile device is connected via the USB cable, the Windows XP Android driver establishes a virtual Personal Area Network (PAN) and an internal IP addressing scheme is associated to the mobile device and to the Central Unit. It allows the mobile device to act as a NTP server over the IP-based PAN. The Central Unit runs an NTP client to synch its clock with the GSM network. Indeed the NTPD (NTP server daemon) running in the mobile device is synched with the Network Operator base station clock through the NITZ (Network Identity and Time Zone) protocol.

¹ Source = NTP Server installation guide for linux based systems ([link](#))

The mobile terminal is a HTC Desire S, equipped with a ARMv7 1Ghz processor and 768 Mbyte RAM. It has been used to be at the same time the clock synch as well as the Central Unit Graphic User Interface (GUI).

2. Wireless Sensor Network (WSN): it is composed by Crossbow TelosB devices. These devices are sensor nodes and the PAN coordinator (it is the bridge between the wireless sensor network and the framework in the central unit, it is called Power Node too).

Topology Network, that is implemented by Crossbow TelosB devices can be only Star topology. Each communication will be directly from the PAN coordinator node to the sensor node and vice-versa.

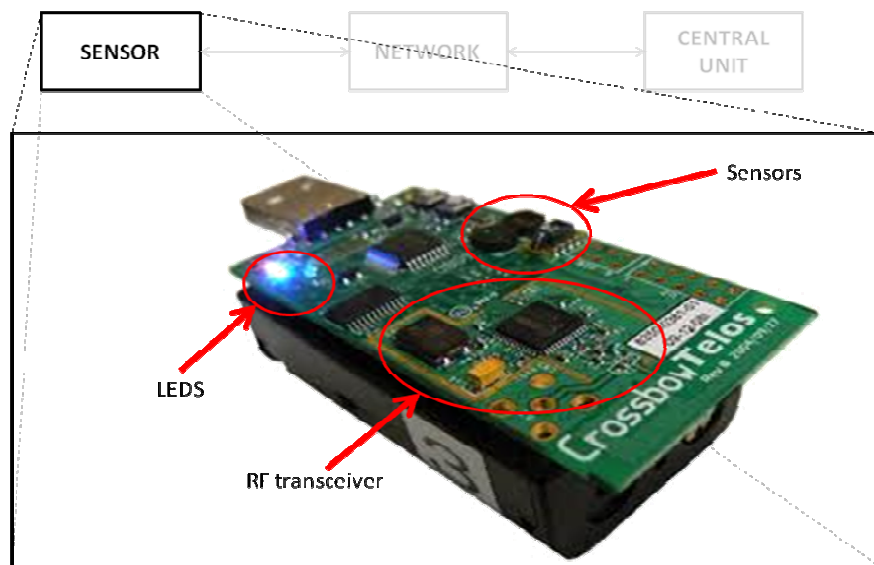


Figure 4 - 4: Crossbow TelosB node

In detail Crossbow TelosB node is an open source platform with the USB programming capability, an IEEE 802.15.4 radio with integrated antenna, a low-power MCU with extended memory, and an sensor suite. Crossbow's TelosB hardware platform features, includes:

- IEEE 802.15.4/ZigBee compliant RF transceiver
- 2.4 to 2.4835 GHz, a globally compatible ISM band
- 250 kbps data rate
- Integrated on-board antenna
- 8MHz TI MSP430 microcontroller with 10kB RAM
- Low current consumption
- 1MB external flash for data logging
- Programming and data collection via USB
- Sensor suite including integrated light, temperature and humidity sensor

The Crossbow TelosB can be powered by two AA batteries. If is plugged into the USB port for programming or communication, power is provided from the host computer and If is always attached to the USB port no battery pack is needed.

3. Configuration Manuals: these are the manuals provided by HW and SW producers; in particular to the aim of verification, the following manuals were considered:
 - Windows XP Professional with SP2 Evaluated Configuration User's Guide – version 3.0 – July 11, 2007
 - MOTE IV - Telos revB (Low Power Wireless Sensor Module) - Manual.
 - CROSSBOW - TPR2400/2420 Quick Start Guide - Rev. A, May 2005 – Doc. 7430-0380-01

4.3 SPD functionalities definition

SPD functionalities used on hypothesized pSHIELD scenario platform are based on D2.2.1 concepts.

The platform performs SPD functions belonging to the following classes:

- 1.AU - SPD Audit
- 2.CS – Cryptographic Support
- 3.IA – Identification and Authentication
- 4.PT – Protection of the SPD functionalities
- 5.SM – Security Management

4.3.1 Class AU – SPD Audit

SPD audit involves recognizing, recording, storing and analyzing information related to SPD relevant activities (i.e. activities controlled by pSHIELD). The resulting audit records can be examined to determine which relevant activities took place and which user is responsible for them.

The identified SPD functionalities of these families are implemented in the Central Unit by Operating System (Microsoft Windows XP, Professional SP 2, CC EAL 4 augmented with ALC_FLR.3 certificated). In particular to implement Audit on pSHIELD the following families and components has been identified:

1)AU_GEN – Audit Data generation with the following components:

AU_GEN.1 - Audit Data generation

- AU_GEN.1.1 - The SPD functionalities shall be able to generate an audit record of the following auditable events:
 - a) Start-up and shutdown of the audit functions;

RE

- b) All auditable events for the basic level of audit.
- AU_GEN.1.2 - The SPD functionalities shall record within each audit record at least the following information:
 - a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and
 - b) For each audit event type, based on the auditable event definitions of the functional components:
 - start-up and Shutdown of the audit functions;
 - reading of information from the audit records.

AU_GEN.2 – User identity association

- AU_GEN.2.1 - For audit events resulting from actions of identified users, the SPD functionalities shall be able to associate each auditable event with the identity of the user that caused the event.

2)AU_SAR – Security audit review with the following components:

AU_SAR.1 - Audit review

- AU_SAR.1.1 The SPD functionalities shall provide authorised users with the capability to read a list of audit information from the audit records.
- AU_SAR.1.2 The SPD functionalities shall provide the audit records in a manner suitable for the user to interpret the information.

AU_SAR.3 – Selectable audit review

- AU_SAR.3.1 The SPD functionalities shall provide the ability to apply a method of selection and/or ordering of audit data based on criteria with logical relations.

4.3.2 CS – Cryptographic Support

This class is used to implement cryptographic functions, the implementation of which could be in hardware, firmware and/or software.

The identified SPD functionalities of these families are implemented in the Crossbow TelosB devices. In particular, to implement cryptographic support on pSHIELD the following families and components has been identified:

1)CS_CKM - Cryptographic key management with the following components:

CS_CKM.1 – Cryptographic key generation

- CS_CKM.1.1 - The Power Node and each single Sensor Node generates cryptographic keys for different levels of encryption. A random key generation with key size 128/256 (when AES is used). A public/private key generation using an algorithms that create a unique key from a random vector implemented by TinyECC library (when ECC is used).

RE

CS_CKM.2 – Cryptographic key distribution

- CS_CKM.2.1 – The WSN sensors shall distribute cryptographic keys in accordance with a WSN broadcast key distribution method (when used AES) and pre-distribution key distribution method (when used ECC); both meet no specific standard.

2)CS_COP – Cryptographic operation with the following component:**CS_COP.1 – Cryptographic operation**

- CS_COP.1.1 - The pSHIELD platform Wireless Sensor Network encrypts and decrypts every message exchanged on network, except the sync messages. The implemented cryptographic algorithms are AES (Advanced Encryption Standard) and ECC (Elliptic curve cryptography). Algorithms are specified by cryptographic key sizes: 128 and 256 bits for AES and 160 bits for ECC. The use of each algorithm/key size is defined according to the specified security level, as described in section 5.3.2.

4.3.3 Class IA – Identification and authentication

The families in this class deal with determining and verifying the identity of users, determining their authority to interact with the pSHIELD, and with the correct association of security attributes for each authorized user and in addition with Wireless sensor network packets traffic authentication. Other requirements classes (e.g. User Data Protection, Security Audit) are dependent upon correct identification and authentication of users in order to be effective.

The identified SPD functionalities of these families are implemented in the Central Unit by Operating System (Microsoft Windows XP, Professional SP 2, CC EAL 4 augmented with ALC_FLR.3 certificated). In particular to implement identification and authentication on pSHIELD the following families and components has been identified:

1)IA_UID - User Identification with the following components:**IA_UID.1 – Timing of identification**

- IA_UID.1.1 - The SPD functionalities shall allow no SPD functionalities mediated actions on behalf of the user to be performed before the user is identified.
- IA_UID.1.2 - The SPD functionalities shall require each user to be successfully identified before allowing any other SPD functionalities mediated actions on behalf of that user.

2)IA_UAU - User Authentication with the following components:**IA_UAU.1 – Timing of authentication**

- IA_UID.1.1 - The SPD functionalities shall allow no SPD functionalities mediated actions on behalf of the user to be performed before the user is authenticated.

- IA_UID.1.2 - The SPD functionalities shall require each user to be successfully authenticated before allowing any other SPD functionalities mediated actions on behalf of that user.

4.3.4 Class PT – Protection of the SPD functionalities

This class contains families of functional requirements that relate to the integrity and management of the mechanisms that constitute the SPD functionalities and to the integrity of SPD functionalities data.

The identified SPD functionalities of these families are implemented in the Central Unit by Operating System (Microsoft Windows XP, Professional SP 2, CC EAL 4 augmented with ALC_FLR.3 certificated). In particular to implement protection of the pSHIELD SPD functionalities the following families and components has been identified:

1)PT_STM – Time stamps with the following components:

PT_STM.1 - Reliable time stamps

- PT_STM.1.1 - The SPD functionalities shall be able to provide reliable time stamps.

4.3.5 Class SM – SPD functions Management

This class is intended to specify the management of several aspects of the SPD functionalities: SPD attributes, data and functions. The different management roles and their interaction, such as separation of capability, can be specified.

This class has several objectives:

- a) management of data, which include, for example, banners;
- b) management of SPD attributes, which include, for example, the Access Control Lists, and Capability Lists;
- c) management of SPD functions, which includes, for example, the selection of functions, and rules or conditions influencing the behaviour of the SPD functionalities;
- d) definition of security roles.

The identified SPD functionalities of these families are implemented in the Central Unit by Operating System (Microsoft Windows XP, Professional SP 2, CC EAL 4 augmented with ALC_FLR.3 certificated). In particular to implement “Security Management” on pSHIELD the following families and components has been identified:

1)SM_MTD – Management of SPD functionalities data with the following components:

SM_MTD.1 - Management of SPD functionalities data

- SM_MTD.1.1 - The SPD functionalities shall restrict the ability to modify the minimum password length, password complexity policy to pSHIELD platform administrator.

2)SM_SMR – Security Roles with the following components:

SM_SMR.1 – Security management roles

- SM_SMR.1.1 - The SPD functionalities shall maintain the roles: pSHIELD platform administrator and operator.
- SM_SMR.1.2 - The SPD functionalities shall be able to associate users with roles.

3)SM_SMF – Specification of Management Functions with the following components:

SM_SMF.1 – Specification of Management Functions

- SM_SMF.1.1 - The SPD functionalities shall be capable of performing the following security management functions:
 - a) modify access control attributes associated with an object
 - b) enable, disable, modify the behaviour of the audit function
 - c) clear the audit trail
 - d) modify the set of events to be audited
 - e) read the audited events
 - f) initialize and modify user security attributes
 - g) modify the minimum allowable password length
 - h) modify the password complexity restriction
 - i) modify the unsuccessful authentication attempts threshold

5 Platform validation process

As introduced in chapter 2, the objective of the platform validation activity is to check the consistency of the platform components in terms of functionalities, semantic models (e.g. metrics and descriptions) and interfaces necessary to perform the SPD composability. So these elements are analyzed in this chapter for each component.

5.1 pSHIELD Ontology

The complete semantic model of the pSHIELD ontology, derived from Task 5.1, is depicted in the following figure.

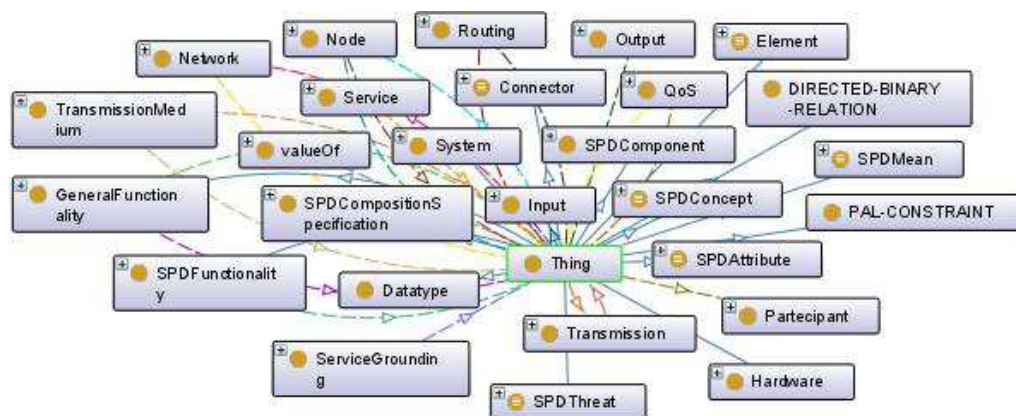


Figure 5 - 1: pSHIELD Ontology

The objective of the demonstrator-tailored Ontology will be to represent all (and only) the information necessary to perform SPD composition of the demonstrator components.

Since a reduced and simple scenario is considered, this model appear to be even too much expressive, and only a subset of the foreseen classes and attributes will be instantiated, on the basis of the architecture of the demonstration platform. The classes and attributes that will not be used, will not be instantiated as well, without affecting the consistency of the model.

5.1.1 Validation against model representation

Since the pSHIELD overall semantic model is consistent and has been designed to describe all the potential elements, configurations and attributes, all its reduced instantiations are validated as well.

1. Is the pSHIELD Ontology able to describe the system's components?
2. Is the pSHIELD Ontology able to describe the SPD Metrics?
3. Is the pSHIELD Ontology able to support SPD Composition (according to "medieval castle" method)?

The expressiveness of the model has been validated at designed level, since on a top-down approach all the information relevant for the pSHIELD purposes have been included in the

semantic model. Then, the consistency of the models has been automatically validated by means of Pellet Reasoner².

The SPD metrics and the composition method have been included in the model as well, by inserting the attribute “SPD_value” and the entity “connector” that support the composition algebra defined in WP2.

On the basis of these argumentations, the pSHIELD Ontology is validated against model representation.

5.1.2 Validation against SPD Composition

The Semantic Engine associated to the selected Ontology will take in input the Scenario parameters and is supposed to generate the correct configuration, on the basis of the Common Criteria approach and of the desired SPD level.

The validation against SPD composition should face the following issues:

1. Is the SPD reasoner able to compute a solution?
2. Is the SPD reasoner able to find always a solution?

Starting from the pSHIELD scenario platform identified in par. 4.2 and SPD functionalities identified in par. 4.3 and associated with it, the resulting scheme for the medieval castle introduced in D2.2.1 (chapter 8) is the following:

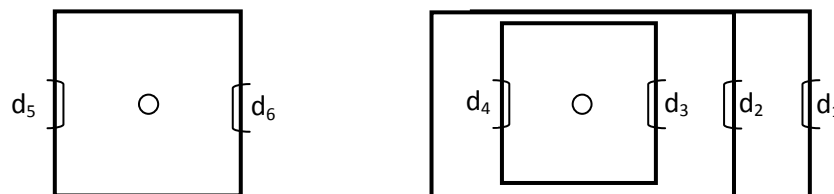


Figure 5 - 2: pSHIELD scenario platform modelling

where:

d_1 = SPD identification and authentication measure (Central Unit)

d_2 = SPD functions management measure (Central Unit)

d_3 = SPD audit measure (Central Unit)

d_4 = SPD functionalities protection measure (Central Unit)

d_5 = SPD identification and authentication measure (WSN)

² <http://www.mindswap.org/2003/pellet/>

d_6 = SPD cryptographic support measure (WSN)

The correspondent system tree representation of the application scenario is:

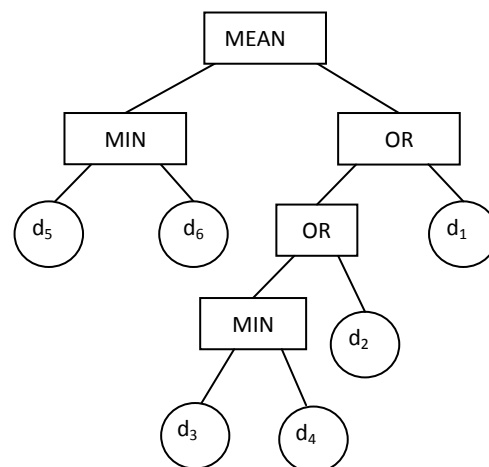


Figure 5 - 3: pSHIELD scenario platform system tree representation

The mathematical expression for the SPD measure of this application scenario system can be defined as follows:

$$d_{\text{TOT}} = \text{MEAN}(\text{MIN}(d_5, d_6), \text{OR}(\text{OR}(d_2, \text{MIN}(d_3, d_4)), d_1)) * d_{\text{LC}}$$

where d_{LC} = SPD measures of life-cycle documentation

In particular, considering the pSHIELD scenario platform WSN, that is the lone part of the system containing selectable SPD functionalities, the following functionalities have been considered:

- d_5 : SPD measures of identification and authentication (WSN)
- d_6 : SPD measures of cryptographic support (WSN)

The WSN SPD level, computed according to the defined algebra, is:

$$\text{MIN}(d_5, d_6)$$

The reasoner is able to compute configurations: thanks to the relations defined in the semantic model, he can recognize elements and attributes (for example it understands that μ TESLA and ECDSA have a different computing contribution to SPD level, or that at least one instance for each entities should be present). Then, through simple algebra rules, it can associate to the configuration the corresponding SPD level.

For validation purposes, in the following table all the computable configurations generated by the reasoner are listed, with the corresponding SPD level (values defined in par. 5.2.4 are considered in the reasoned for computing).

OVERALL SPD	Identification & Authentication	Cryptography
1	μTESLA	AES (128)
1	μTESLA	AES (256)
1	μTESLA	ECC/ECIES
1	ECDSA	AES (128)
3	ECDSA	AES (256)
6	ECDSA	ECC/ECIES

Table 5 - 1 Computable configurations generated by the reasoner

Given this list, we can confirm by inspection that the reasoner is able to find at least one valid configuration (in case it exists) or to inform the user about the absence of solution. Since the algebra is linear and simple, the reasoner can perform even a brute-force search, or an optimized one, to search for the corresponding solution and consequently provide an answer in a finite time.

If the reasoner works for this set of values, then it automatically works for each set of values.

5.2 Middleware and Overlay for discovery and composition

The objective of the Middleware and Overlay platform validation is to focus on the validation of the Core SPD services implemented in the platform. The activity is to check the consistency of the platform components in terms of functionalities necessary to perform the SPD composability. The needed functionalities are: discovery, composition and orchestration. An additional criteria for the middleware and overlay validation is to validate the middleware architecture as a whole against a SPD level change.

5.2.1 Validation against discovery

In order to validate the discovery the following criteria have been taken into account:

1. Is the discovery mechanism technology independent?
2. Is the discovery mechanism using standard protocols?
3. Is the discovery mechanism robust to failures?

The platform discovery functionality is addressed by a discovery framework described in D5.2. For the sake of simplicity the discovery framework is depicted in the above figure:

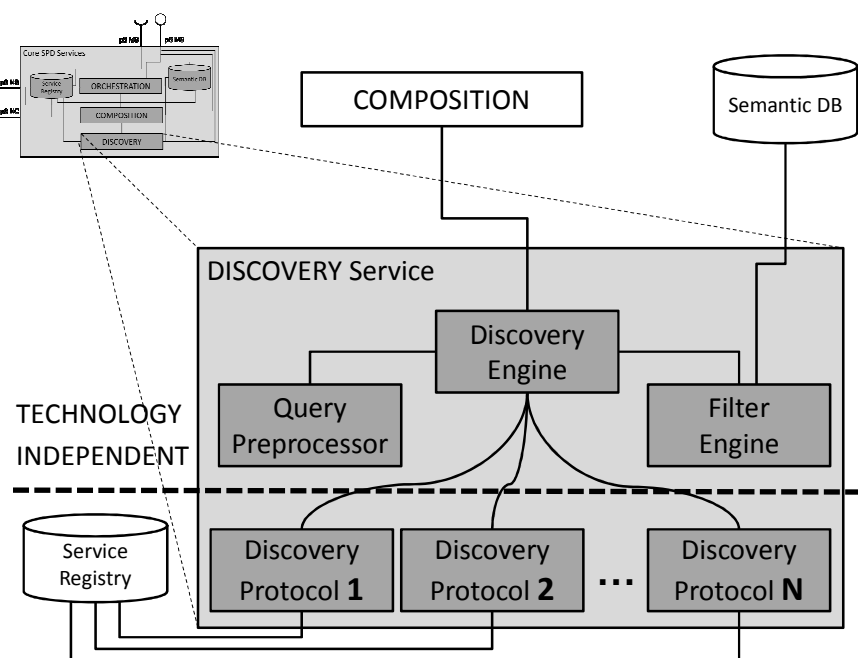


Figure 5 - 4: Details of the Discovery core SPD service

Where the following functional elements can be identified:

- **Discovery Engine:** it is a technology independent functionality in charge to handle the queries to search for available pSHIELD components sent by the Composition service;
- **Query Pre-processor:** it is a technology independent functionality in charge to enrich the query sent by the Composition service with semantic information related to the peculiar context;
- **Filter Engine:** it is a technology independent functionality in charge to semantically match the query with the descriptions of the discovered SPD components;
- **Discovery Protocol:** it is a technology dependent component in charge to discover all the available SPD components description stored in the Service Registry, using a specific protocol (e.g. Service Location Protocol – SLP or Universal Plug and Play Simple Service Discovery Protocol – UPnP SSDP, etc.).

The discovery functionality is thus composed by a technology independent core to manage the discovery of available SPD functionalities independently from the specific discovery protocol and technology and some adapters, that can be dynamically added or deleted to extend the scope of the discovery. These adapter are based on specific, technology dependent protocols to apply for the real search of available SPD functionalities both locally on single the embedded system, as well as remotely, over heterogeneous networks, in a distributed system of embedded devices.

The implemented adapters use the Service Location Protocol (SLP) version 2. It has been defined in RFC 2608 and extended in RFC 3224. SLPv2 allows embedded systems to find services in a local area network without prior configuration. SLP has been designed to scale from small, unmanaged networks to large enterprise networks.

SLP specifies security mechanisms for authentication and authorization mechanisms and extensive assessment on its reliability can be found on literature³.

On the light of the above considerations, it is possible to assert that the platform is validated against the discovery functionality.

5.2.2 Validation against composition

The composition mechanism relies on the semantic composition, thus it has been validated in section 5.1.2.

5.2.3 Validation against orchestration

In order to validate the orchestration the following criteria have been taken into account:

1. Is the orchestration mechanism compliant with standards?
2. Is the orchestration mechanism feasible in limited resources devices?
3. Is the orchestration mechanism technology independent?
4. Is the orchestration mechanism robust to failures?

The pShield orchestration mechanisms has been implemented using OSGi as the reference service platform. OSGi is characterized by the following advantages:

- OSGi is an open standard;
- OSGi has a number of open source implementation (Equinox, Oscar, Knopflerfish);
- OSGi can be executed even over lightweight nodes (Embedded Systems Devices);
- OSGi has been implemented using different programming languages (e.g. Java, C, C#);
- The Java implementations of OSGi is fast to deploy and it is much easier to learn, facilitating even an active and collaborative prototype deployment among partners;
- OSGi plugins are available for a number of IDE tools (i.e. Eclipse, Visual Studio, etc.);
- OSGi can be easily deployed in Windows (XP, 7, Mobile), Linux, MAC and Google (Android) OSes.

More in particular the orchestration platform is based on an open source Knopflerfish OSGi service platform. Knopflerfish (hereafter referred as to KF) is a component-based framework for Java in which units of resources called bundles can be installed. Bundles can export services or run processes, and have their dependencies managed, such that a bundle can be expected to have its requirements managed by the container. Each bundle can also have its own internal classpath, so that it can serve as an independent unit, should that be desirable.

The OSGi environment is robust and secure as deeply analyzed in literature⁴. Thus all the three validation criteria are satisfactory matched.

³ http://www.openslp.org/doc/security/threat_analysis_min_security.html and <http://www.bettstetter.com/publications/vettorello-2001-wadhoc-slp.pdf>

⁴ <http://www.ifi.uzh.ch/pax/uploads/pdf/publication/1099/Bachelorarbeit.pdf>

The objective of the Middleware and Overlay platform verification activity is to check the platform behavior with respect to the selected scenario by means of focused functional tests.

5.2.4 Validation against the SPD level change

Two different type of approach have been taken to validate the Middleware and Overlay platform against the SPD level change. The first approach is on a theoretical basis, the second is bases on concrete field tests.

From a theoretical point of view, the joint operation of the pShield overlay, the pShield Core SPD services (discovery, composition and orchestration) and the pShield middleware layer (that interact with the network and the node layer too) apply as a closed loop system, where the Current SPD level measured by the pShield middleware is continuously compared with the Desired SPD Level by the Overlay. The Overlay applies for configuration rule to react against any potential SPD gap between the desired level and the current level. The configuration rules are then enforced into the system of embedded systems by the Core SPD services and applied concretely by the pShield adapters at middleware, network and node layer.

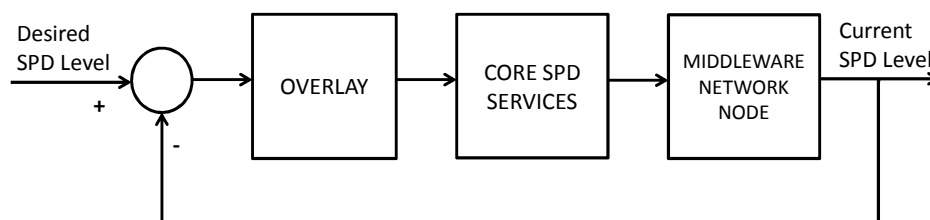


Figure 5 - 5: Closed-loop SPD level control

From a pragmatic point of view, to verify that the Core SPD services are able to maintain a desired SPD level some field tests have been performed. Five different types of SPD functionalities have been defined:

- **Audit** - auditing of user given its account (AU)
- **Accounting** – accounting personnel once identified (AU)
- **Identification** – identification of user, identification of sensor (IA)
- **Cryptographic support** – cryptographic algorithms based on a key Exchange (CS)
- **Key Management** – management of keys (CS)

SPD functions Management (SM) and Protection of SPD functionalities (PT) have been considered as support functionalities and so have not been included in this paragraph.

Auditing, Accounting have been assumed Central Unit SPD functionalities, Identification and Authentication has been considered to be Central Unit and a WSN functionality, while Crypto and Key Management have been assumed to be WSN functionalities.

Three Cryptographic algorithms have been considered:

- **AES 128** – Advanced Encryption Standard (128 key size) (SPD metric = 1);
- **AES 256** – Advanced Encryption Standard (256 key size) (SPD metric = 3);
- **ECC/ECIES** (Elliptic Curve Integrated Encryption Scheme) (SPD metric = 5);

These functionalities have been modeled semantically and their mutual relationships are shown in the following figure:

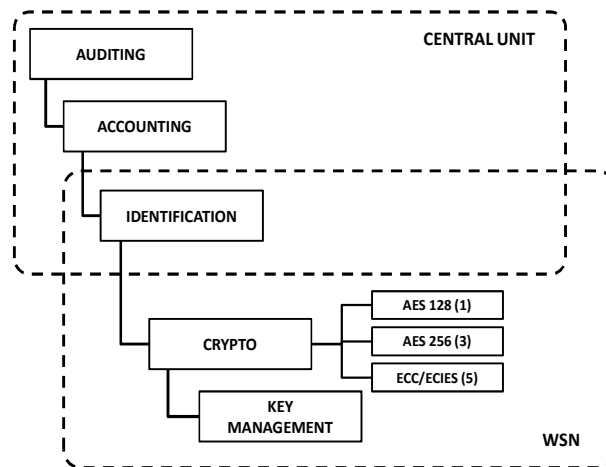


Figure 5 - 6: semantic model of the SPD functionalities

The following tests have been done:

1. Switch the SPD level to 1;
2. Switch the SPD level to 3;
3. Switch the SPD level to 5;

The following results have been obtained:

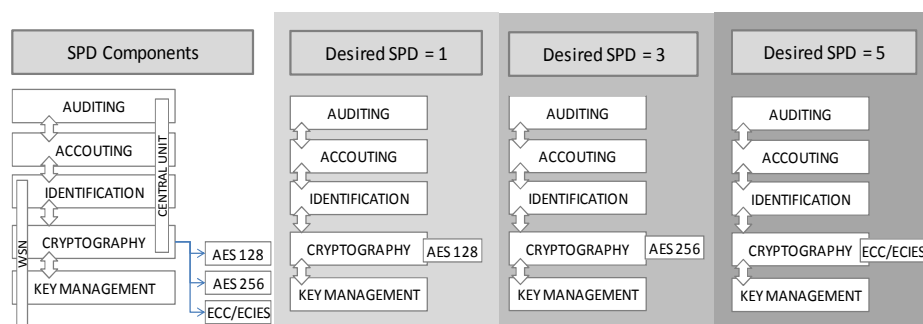


Figure 5 - 7: validation results of against the SPD level change

Thus an empirical validation of the platform has been done resulting in a successful composition of SPD functionalities. While the static process is convincing, the main concern is about the time required by the system to enforce the required SPD level. After a number of empiric tests, we have gathered some statistics showing that the convergence time for the railway scenario ranges from 15 to 60 seconds.

During the transient state, the system is always coherent, meaning that all the adapters work properly and the overall SPD level is always beyond the starting level.

On the light of the above consideration we can assert that the platform successfully matches with the validation criteria.

5.3 Heterogeneous Wireless Sensors Networks and secure communication

5.3.1 General Functionality

From the WSN point of view, the CryptographicHashing has three levels of cryptography:

- AES (128) – the first level of encryption
- AES (256) – the second level of encryption
- ECC/ECIES (Elliptic Curve Integrated Encryption System) (160) – as the third and the highest level of encryption for WSN

5.3.2 WSN Cryptography levels description

In order to define the different cryptography levels in the WSN, a predefined security levels was set in the nodes.

All nodes starts without encryption and only after the framework sets an encryption level, the commands for changing encryption are sent via radio and the communication starts to be encrypted. This is the approach presented for the demonstrator to easily show the nodes communication without encryption.

For any different approach or configuration of the start-up encryption level it demands minor changes in the nodes firmware.

5.3.2.1 Level 1 and 2 – AES

The Level 1 and 2 uses AES (Advanced Encryption Standard) a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data. As said before on subsection **Errore. L'origine riferimento non è stata trovata.**, it is used two different sizes of cryptographic keys: for level one is used 128 bits and for level two is used 256 bits.

5.3.2.2 Level 3 – ECC – ECIES

The Level 3 uses ECC with the ECIES scheme from TinyECC library. TinyECC is a software package providing ECC-based PKC operations that can be flexibly configured and integrated into sensor network applications. It is used to provide a public key encryption scheme (ECIES) to the demonstrator.

5.3.3 WSN key exchange

From the WSN point of view, there is one level of key exchange. Since the main objective of this prototype was to demonstrate SPD composability mechanism, there were not implemented specific key management mechanisms that guarantee the nodes authenticity. This task can be accomplished by the implementation of key pre-distribution mechanisms, as described in D4.2 "SPD Network Technologies Prototype Report. There are a maximum number of Sensor Nodes that can be connected in WSN. For demonstration purpose this number is four (4).

The scheme implemented in this demonstrator is depicted in Figure 5-8. The Power Node initializes the protocol by generating the private and the public key. After generation is concluded it sends the public key through radio. If a sensor node is connected it will receive the public key and stores in the memory.

The sensor node will also perform the same procedure. When it boots it will create the private key and then send the public key to the network.

The Power Node will also receive the public keys from sensor nodes and store them in the memory. The Power node and all the nodes will use the generated private and public keys to secure the communication. After exchange of the public keys by radio, the nodes will send and receive encrypted messages. The ECIES scheme grants the confidentiality (due to encryption) and non-repudiation.

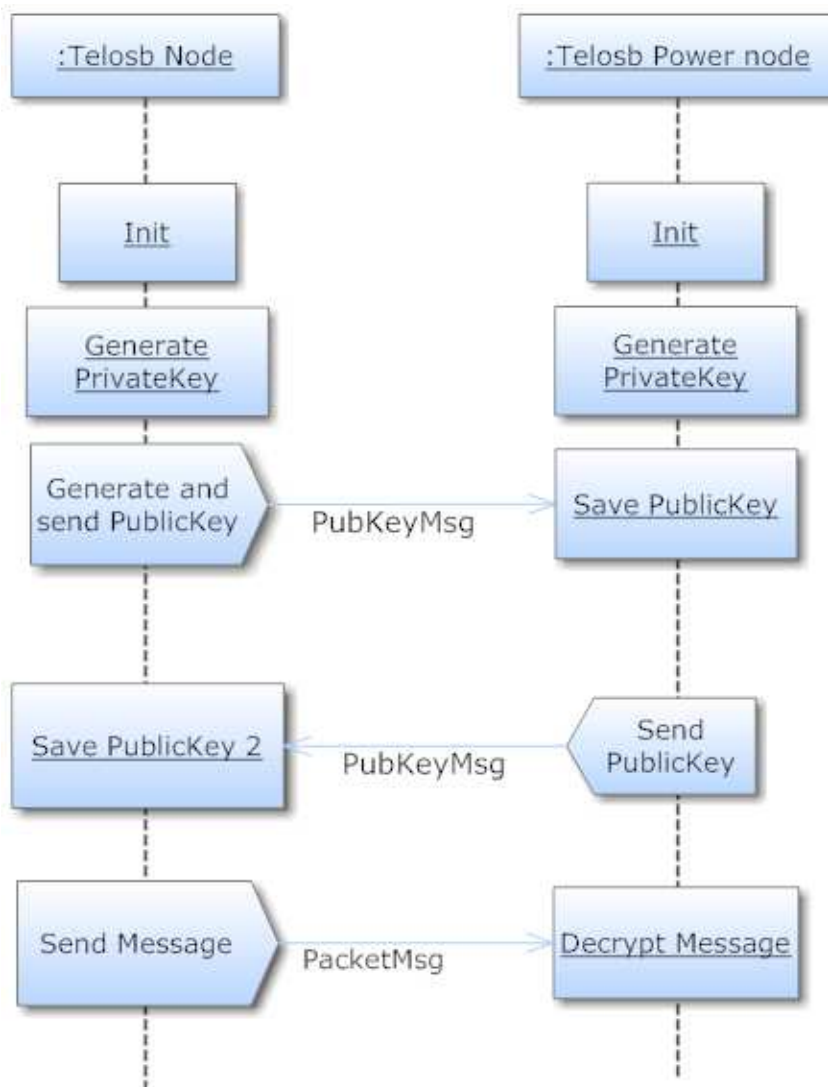


Figure 5 - 8 : Exchange public keys protocol between Power and Sensor Node

5.3.4 Communication mechanism

In the wireless sensor network the communication mechanism has three main types of entities that communicate with each other: OSGI, Power Node and Sensor Nodes. Through the diagram in Figure 5 - 9 it is possible to see an example of messages exchanged between them.

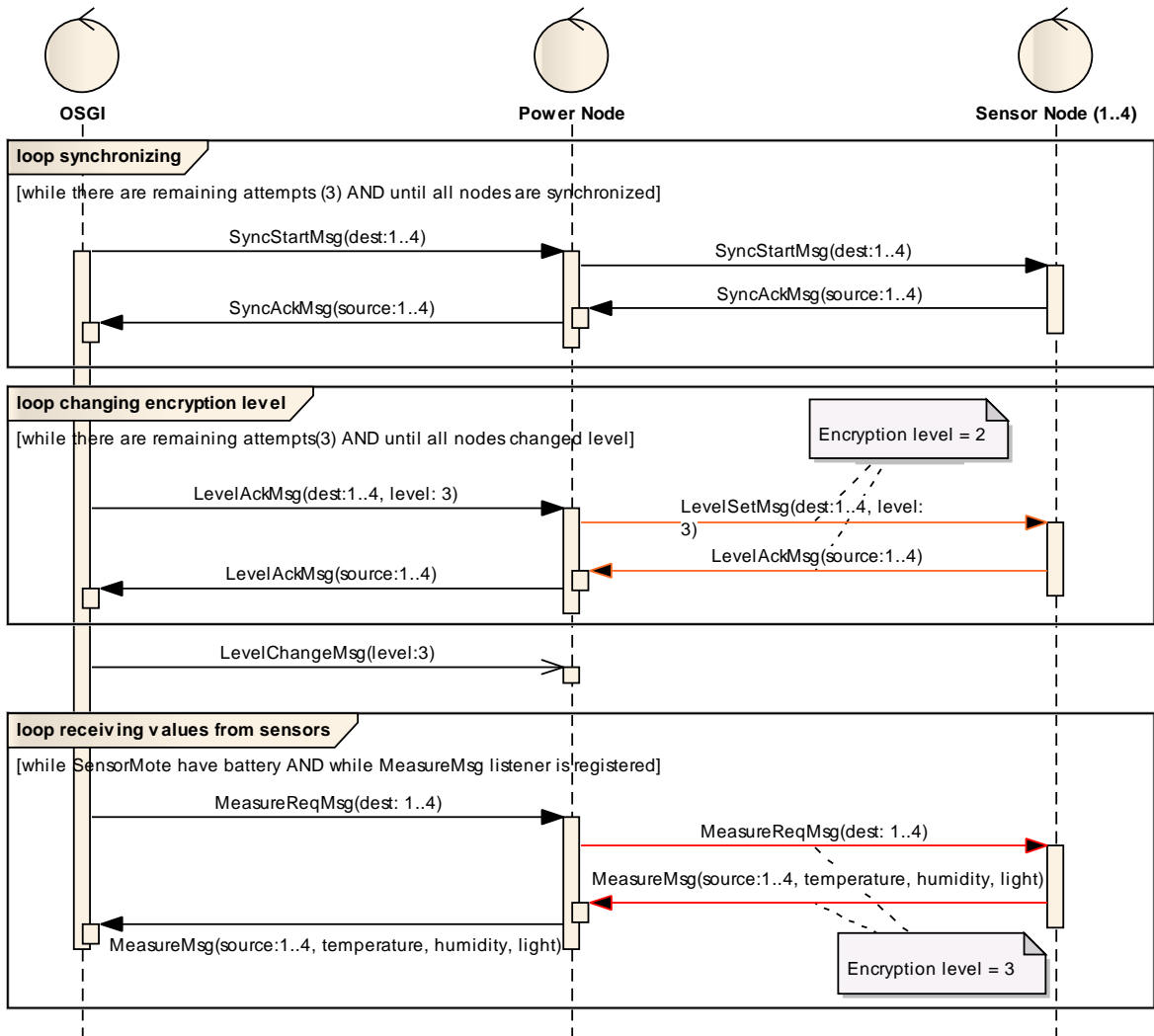


Figure 5 - 9: Example of messages exchanged between Power and Sensor nodes.

There are also three types of communications scopes:

- Synchronizing – When pSHIELD system starts, the synchronism between Power Node and the various Sensor Nodes is performed. OSGI sends one start message, for each existent sensor node, to Power Node which will forward the same message for its destination. After each Sensor Node receiving this start message, they will send back an acknowledge message to Power Node and this one will forward the message to OSGI. From now on there are synchronism between the nodes and can be exchanged others types of message.
- Changing encryption level – If there are synchronism between the nodes, every time it is requested a change of encryption level by OSGI's system, it will send a message, for each registered sensor node (synchronized with power node), with the requested level, to Power Node. Power Node will forward the message, encrypting it before sending. Sensor nodes will send back an acknowledge message, encrypted on the current level,

to Power Node that will decrypt it and will send back to OSGI's system. After OSGI's system receives the acknowledge message from all nodes, or a timeout occurs, it will send other message from OSGI to Power Node to start encrypting messages in the new level. In diagram example the current level is 2 and the requested level is 3.

- Receiving values from sensors – OSGI platform sends, each 120 seconds, a measures request to each Sensors Node using Power node as forwarder, that encrypts the message. When each Sensor Node receives this message, reads its sensor data, and sends them encrypted to the Power Node, which decrypts them and sends back to the OSGI. In the example diagram, the measure messages were encrypted in level 3.

5.3.4.1 Power Node behaviour

Inside the Power Node, predefined actions occur when it receives some message from UART's connection with PC that is executing OSGI's system. In the activities diagram in Figure 5 - 10 it's possible to observe that each time that UART sends a message, Power Node will do different actions according to the type of message:

- LEVEL_CHANGE_MSG – it changes level variable to level got from message and it doesn't do nothing more.
- SYNC_START_MSG – it sets level variable to level 0, adds its public key (Power Node) to the message and then it acts like other messages.
- Others messages (like LEVEL_SET_MSG) – they compute CRC of messages, encrypt them on the current level and send them to Sensor Nodes by wireless.

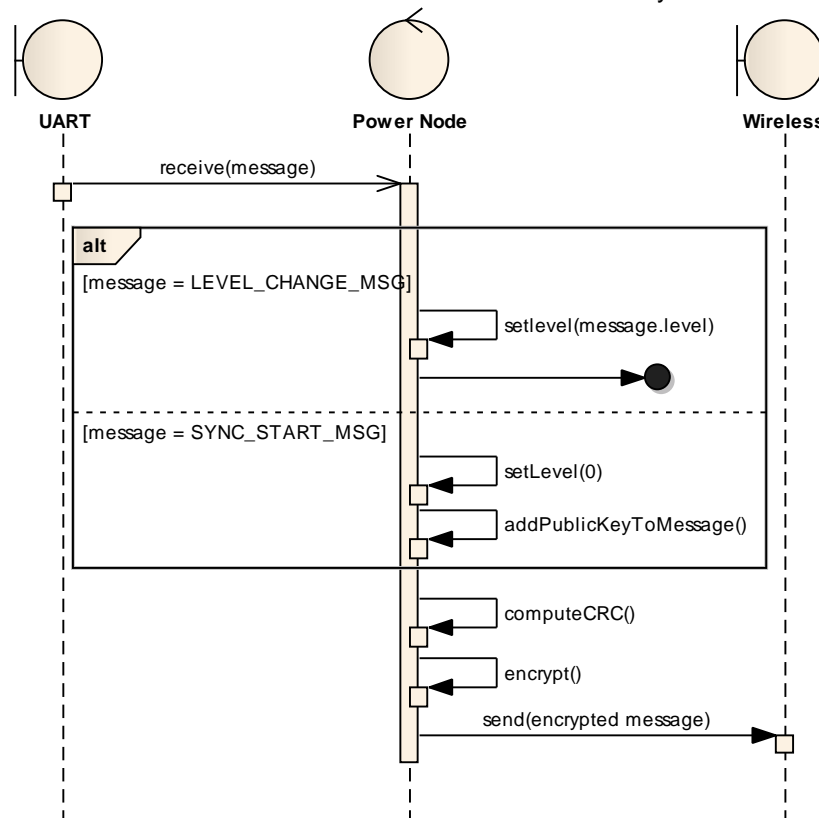


Figure 5 - 10: Power node behaviour (UART -> Wireless)

When Power Node receives messages from Sensor Nodes by wireless, its behaviour can be seen in Figure 5 - 11.

RE

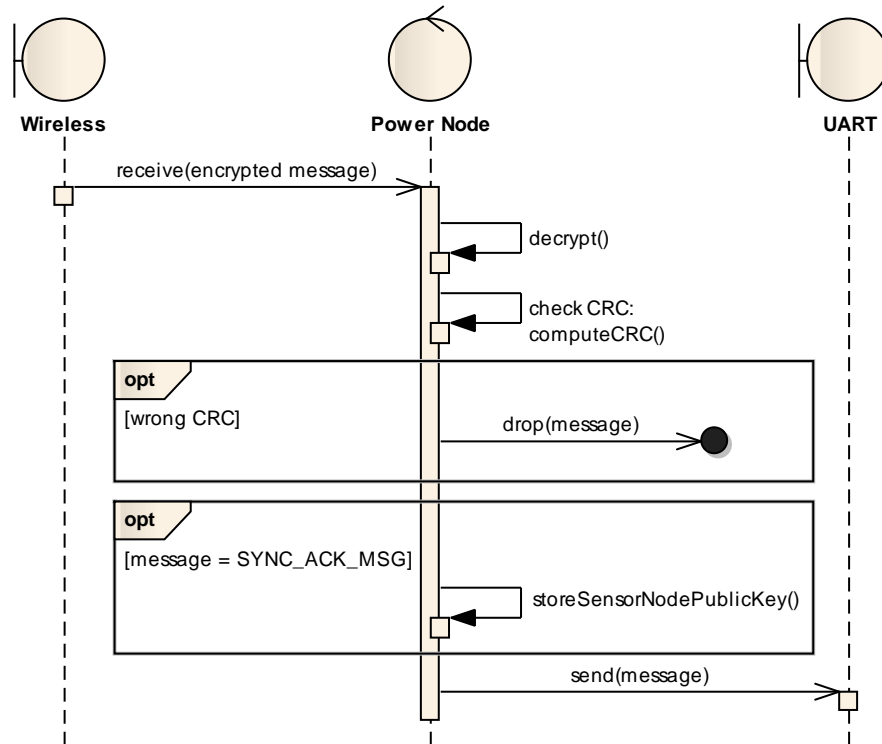


Figure 5 - 11: Power node behaviour (Wireless-> UART)

Firstly it decrypts the message, after that it checks the CRC of decrypted message in order to check the message integrity. If it isn't correct, the message is discarded. Otherwise the decrypted message will be send to UART.

In the case of sync acknowledge messages, they are stored in the Power Node as the Sensor Node's public key.

5.3.4.2 Sensor Node behaviour

Like Power Node, also Sensor Nodes have some important actions happening when they receive Power Node's messages or when sensors readings are obtained. The Sensor Node behaviour can be described as a state machine like it is showed in diagram in Figure 5 - 12. When Sensor Node starts, it waits for synchronism with Power Node. As soon as synchronism is done, the sensor node changes to waiting level state where it waits for a set level's message. When this one is received, the sensor changes to operational mode. In this state three actions can occur:

- Request new sync with the Power Node, passing the sensor node back to the waiting level state.
- Receives a new message to change the level, where some actions happen but the operational state is kept.
- Receives a timer signal (after each 120 seconds) in order to read the sensor data and send it to OSGI's system.

RE

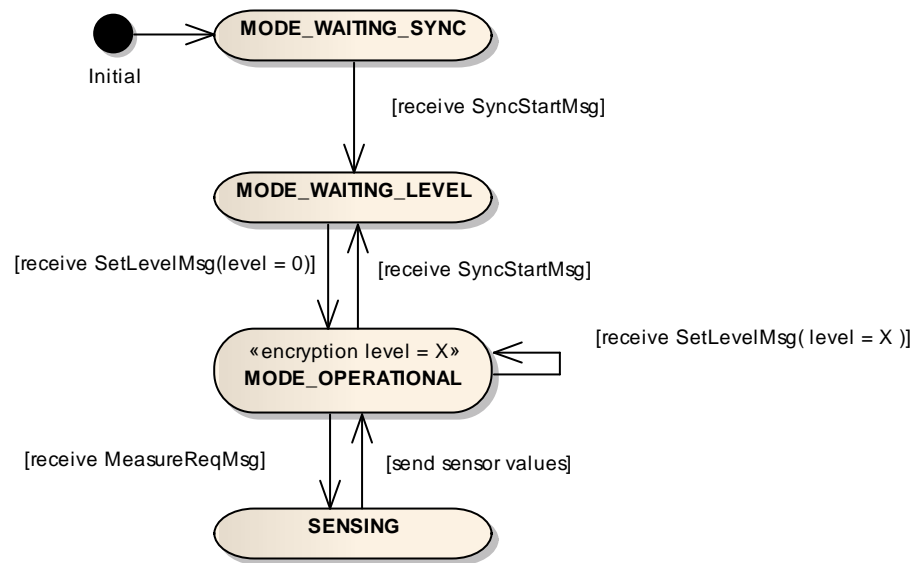


Figure 5 - 12: Sensor node behaviour represented as a states machine

The behaviour of a Sensor Node when receiving a message from wireless network it's depicted in the Figure 5 - 13.

The message received can be unencrypted or encrypted. If it is unencrypted and it is a sync start message, sensor node will store the Power node public's key, set level to 0, change the state to waiting level and send back an acknowledge message with its Public key.

Otherwise, when Sensor Node receives a level set message while being in waiting state, the first step is to decrypt it and after that it checks its CRC. If there is a bad CRC match, probably it's because the level set message it's repeated (Sensor Node already changed its level but it continues receiving messages asking for the level change), so it has to be checked if the message can be decrypted with the last Sensor Node encrypt level. If CRC check fails again, the message is discarded. If CRC check was successful, Sensor Node updates its current level and state and sends back an encrypted acknowledge message.

When Sensor Node is in operational state and receives a measure request message, decrypts it and checks its CRC. If CRC check fails, the message is discarded. If CRC check was successful, Sensor Node reads data from its three different sensors, creates a message with that information, computes CRC message, encrypts it and sends it by wireless to the Power node.

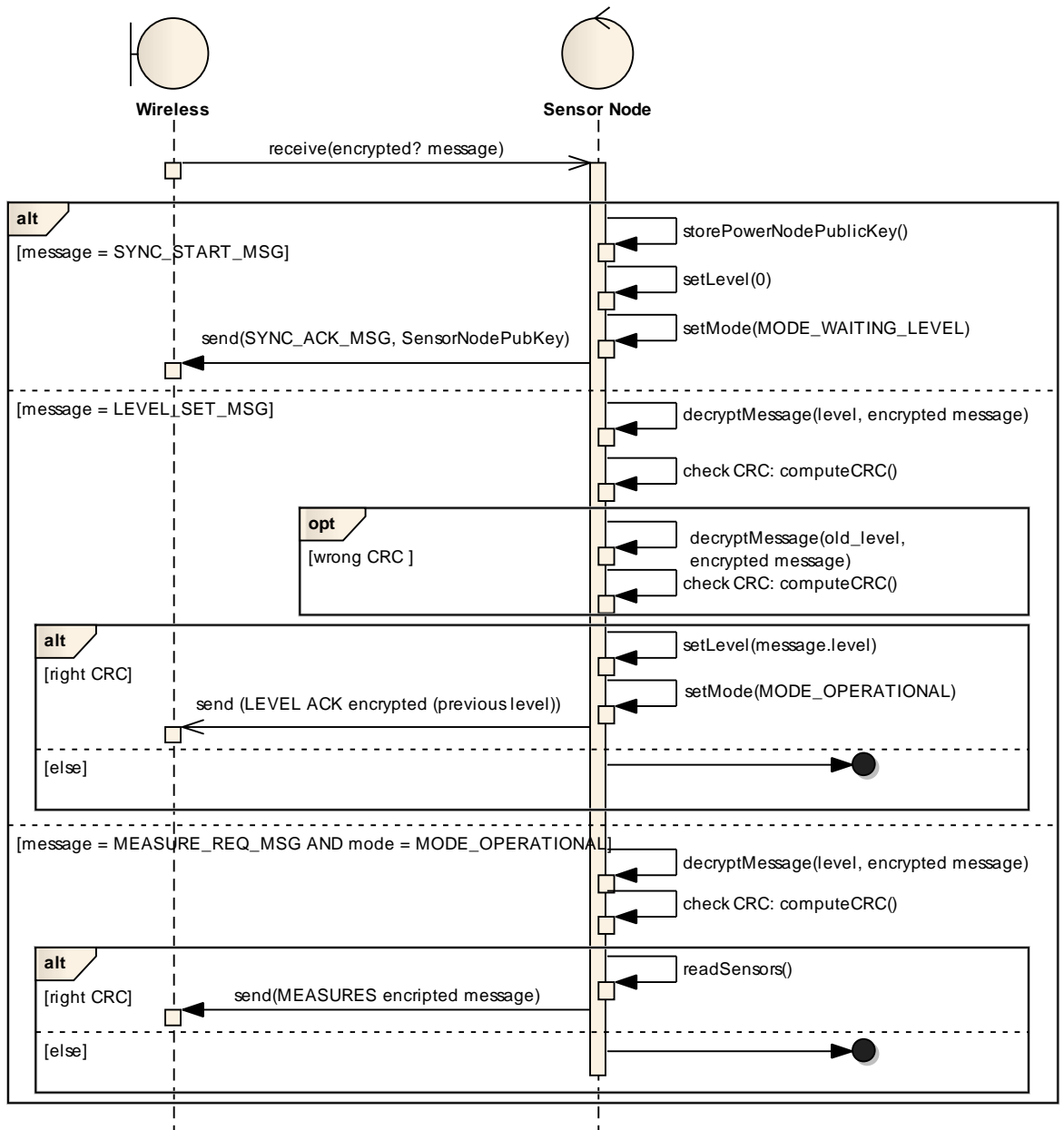


Figure 5 - 13: Sensor node behaviour (From Power Node requests)

6 Platform verification process

Test documentation provides all information useful to repeat tests to ensure that they work as described and that documented results are obtained.

Tests described below clearly identify the purpose of each test so that it's possible to identify the objective. It should also identify the related SPD function for each test (this will assist in meeting the traceability requirements).

6.1 Tests identification and description

The proposed test plan has the purpose to demonstrate that all SPD functions implemented in the pSHIELD for the specific scenario, work properly without errors.

Tests are executed with the aim to demonstrate the correct SPD functions implementation.

Tests can be indicated as positive or negative..

Positive tests have been designed with the purpose to show the correct functioning modifying input parameters coupled with:

- nominal values (with any accepted intermediates value);
- smallest accepted values (the smallest value for ordinal or cardinal number, the smallest string length,...);
- biggest accepted values (the biggest value for ordinal or cardinal number, the biggest string length,...);
- borderline values (value immediately bigger than the smallest accepted value and value immediately smaller than the biggest accepted value);
- values only for mandatory arguments (minimum input);
- values all arguments (maximum input).

Negative tests have been designed in order to verify the correct control over input trying:

- incorrect values;
- values with different type respect to those expected;
- borderline values (value immediately smaller than the smallest accepted value and value immediately bigger than the biggest accepted value).

6.1.1 Test identification

Each test is labeled following this scheme:

[IF]-TX

where [IF] is SPD function class identification, T stands for Test and X is an identification number (e.g. [ID]-T02 indicates test number 2 for identification function); moreover for each test, positive or negative typology is identified.

6.1.2 Testbed definition

In order to achieve tests for pSHIELD platform verification, the following deployment and devices were used.

Wireless Sensor Network (WSN) in the star topology. In the star topology, the communication is established between devices and a single central controller, called the Power node. The Power node is maintained powered while the devices are AA battery powered. After the FFD (full-function device) is activated for the first time, it establishes its own network and becomes the PAN coordinator. The start network chooses a PAN identifier, which is not currently used by any other network within the radio sphere of influence. This ensure us that each star network operates independently.

The WSN is composed by three **Crossbow TelosB** devices powered by two AA batteries and a **TelosB** plugged into USB laptop port as Power node. See 4.2 paragraph for TelosB technical specifications.

The laptop composing the **Central Unit** is a HP Pavillion DV6 series, equipped with a Intel Core I7 quad processor with 4 Gbyte DDR3 RAM and Microsoft Windows XP Operating System in certified configuration which provides a set of security functions verified in the following tests.

The **Central Unit GUI** is provided by a HTC Desire S mobile terminal , equipped with a ARMv7 1Ghz processor and 768 Mbyte RAM. This terminal is connected to the laptop by a usb cable and acts as NTP server too. See 4.2 paragraph for more detail.

The **CC2530DK** is Texas Instrument's second generation ZigBee/IEEE 802.15.4 compliant System-on-Chip with an optimized 8051 MCU core and radio for the 2.4 GHz unlicensed ISM/SRD band. This device enables industrial grade applications by offering state-of-the-art noise immunity, excellent link budget, operation up to 125 degrees and low voltage operation. In addition, the CC2530 provides extensive hardware support for packet handling, data buffering, burst transmissions, data encryption, data authentication, clear channel assessment, link quality indication and packet timing information. The CC2530 Development Kit includes all the necessary hardware to properly evaluate, demonstrate, prototype and develop software targeting for IEEE802.15.4 or ZigBee compliant applications.

The CC2530DK contains the following components

- SmartRF05EB
- CC2530 Evaluation Modules
- Antennas
- CC2531 USB Dongle
- Cables

- Batteries
- Documents

The SmartRF05EB (evaluation board) is the main board in the kit with a wide range of user interfaces: 3x16 character serial LCD, Full speed USB 2.0 interface, UART, LEDs, Serial Flash, Potentiometer, Joystick, Buttons. The EB is the platform for the evaluation modules (EM) and can be connected to the PC via USB to control the EM.

The CC2530EM (evaluation module) contains the RF IC and necessary external components and matching filters for getting the most out of the radio. The module can be plugged into the SmartRF05EB. Use the EM as reference design for RF layout.

The CC2531 USB Dongle is a fully operational USB device that can be plugged into a PC. The dongle has 2 LEDs, two small push-buttons and connector holes that allow connection of external sensors or devices. The dongle also has a connector for programming and debugging of the CC2531 USB controller. The dongle comes preprogrammed with firmware such that it can be used as a packet sniffer device.

The **SmartRF Packet Sniffer** is a PC software application used to display and store RF packets captured with a listening RF HW node. Various RF protocols are supported. The Packet Sniffer filters and decodes packets and displays them in a convenient way, with options for filtering and storage to a binary file format.

The **HP laptop** connected to the CC2530DK board and where SmartRF packet sniffer application is installed is a HP Compaq nc 6120 with a Pentium M 2 GHz processor and 2GB RAM with Microsoft Windows XP Operating System.

In the following figure the described testbed deployment.

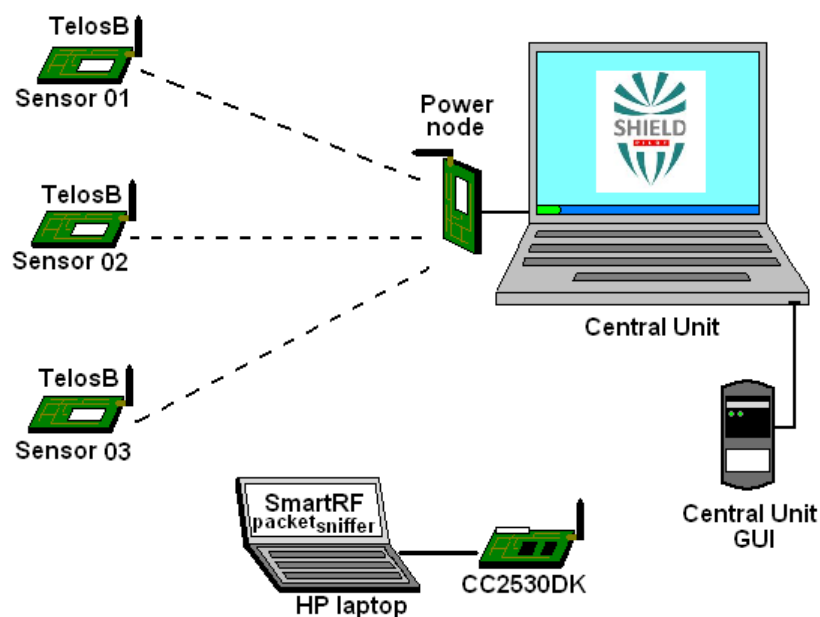


Figure 6 - 1: Testbed representation

6.1.3 Testbed devices initialization

Before starting tests session it is necessary to execute the following steps.

1. HP Pavillion DV6 series:
 - Microsoft Windows XP Operating System certified configuration [9];
2. TelosB sensors:

In order to set the devices operating, after connect them by USB to a computer (Linux Operating system), next commands must be executed:

- See the list of connected notes

```
motelist
```

- Change permissions of each device x connection like:

```
chmod 666 /dev/ttyUSBx
```

- On code directory, install code in mote that will play the role of power node:

```
make -f Makefile.GW telosb install
```

- On code directory, install code in each mote x (existing in motelist) that will play the role of sensor nodes:

```
make -f Makefile.Node telosb install.x bsl,/dev/ttyUSBx
```

Image bellow show a example of last lines of output, in terminal, after the command "make -f Makefile.GW Telosb install"

```
MSP430 Bootstrap Loader Version: 1.39-telos-8
Mass Erase...
Transmit default password ...
Invoking BSL...
Transmit default password ...
Current bootstrap loader version: 1.61 (Device ID: f16c)
Changing baudrate to 38400 ...
Program ...
40374 bytes programmed.
Reset device ...
rm -f build/telosb/main.exe.out build/telosb/main.ihex.out
```

Sensor nodes after being flashed will turn on the blue LED (waiting synchronization with power node).

During this initialization phase some errors can occur:

- In task 2, if the nodes aren't well connected and recognized by computer, it will appear the below message

```
pshield@pshield:~/cvs_pshield/wsn2$ sudo chmod 666 /dev/ttyUSB0
chmod: cannot access '/dev/ttyUSB0': No such file or directory
```

- In task 2, if the nodes aren't well connected and recognized by computer, but the command it's executed without super user permissions, the following output will appear:

```
chmod: changing permissions of `/dev/ttyUSB0': Operation not permitted
pshield@pshield:~/cvs_pshield/wsn2$ chmod 666 /dev/ttyUSB0
```

- In task 3 and 4, if the nodes aren't well connected, recognized by computer and the user haven't the right permissions (see task 2), the last lines of output will be similar to these ones:

```
cp build/telosb/main.ihex build/telosb/main.ihex.out
found no motes (using bsl,auto)
```

3. HP Laptop:

- SmartRF packet sniffer application configuration [10];
- CC2530 DK board configuration [11].

6.2 Tests

6.2.1 Audit tests

[AU]-T01	
Purpose	Reading of information from the audit records
Initialization phase	Windows events and Performance Logs and Alerts are recorded by the Event Log service. The Event Log service starts automatically when Windows XP Professional is started. All users can view the Application and System logs, however, only administrators have access to the Security logs.
Positive test	
Expected result	<p>The Event Viewer Security log displays the following types of events:</p> <p>Success audit. An audited security access attempt that succeeds. For example, the successful attempt by a user to log on the system will be logged as a success audit event.</p> <p>Failure audit. An audited security access attempt that fails. For example, if a user tries to access a network drive and fails, the attempt will be logged as a failure audit event.</p>
Input data	Path to follow
Procedure	<ol style="list-style-type: none"> 1. Log on as an authorized administrator. 2. Click Start, point to All Programs, point to Administrative Tools, and select Computer Management. 3. In the console tree, expand Event Viewer. Select Security and in the details pane, examine the list of audit events. 4. Scroll through the details pane to view the various fields
Output	<p>The event fields described below:</p> <p>The event logs record five types of events:</p> <p>Type:</p> <ul style="list-style-type: none"> • Error - A significant problem, such as loss of data or loss of functionality. For example, if a service fails to load during startup, an error will be logged. • Warning - An event that is not necessarily significant, but may indicate a possible future problem. For example, when disk space is low, a warning will be logged. • Information - An event that describes the successful operation of an application, driver, or service. For example, when a network driver loads successfully, an Information event will be logged

- **Success Audit** - An audited security access attempt that succeeds. For example, a user's successful attempt to log on the system will be logged as a Success Audit event.
- **Failure Audit** - An audited security access attempt that fails. For example, if a user tries to access a network drive and fails, the attempt will be logged as a Failure Audit event.

Date: The date the event took place

Source: The process that raised the event. Time The time the event took place.

Category: The specific class the event is categorized under.

Event: A unique numerical identifier for the event.

User: The user that generated the event.

Computer: The computer on which the event was generated.

Event details provide more information about events than the Events view. This additional information includes the event's source, a description of the event, and details about what is affected by the event. To view additional details for an event, double-click on the event. An event Properties window will appear.

The Description field of the event Properties window provides a longer explanation for the event, including what resources are affected and other technical information.

RE

[AUJ-T02]	
Purpose	Searching for specific events
Initialization phase	No initialization phase
Positive test	
Expected result	The Event Viewer Security let to search for specific events.
Input data	Path to follow and data to fill in the proper fields.
Procedure	<ol style="list-style-type: none"> 1. Log on as an authorized administrator. 2. Click Start, point to All Programs, point to Administrative Tools, and select Computer Management. 3. In the console tree, expand Event Viewer. Right-click Security (or other event log), point to View, and select Find.
Output	<p>The Find in local Security interface will appear. Under Event types, specify the types of events to search for. In Event source, Category, Event ID, User, Computer, or Description, specify additional information, as needed, to further define the search.</p> <p>Click the Find Next button. This will find and highlight the first event matching the search criteria. Clicking the Find Next button again will find the next matching event. This can be done continuously to search through the entire log for each matching event</p>

RE

RE

[AU]-T03	
Purpose	Demonstrate that security relevant event has been recorded.
Initialization phase	Execute test [IA]-T01
Positive test	
Expected result	The security relevant event generated during initialization phase is recorded in the audit file and the Event Viewer Security log displays it.
Input data	Path to follow and data to fill in the proper fields.
Procedure	Follow the procedure of test [AU]-T01 to reach the Event Viewer console and the procedure of test [AU]-T02 to select Logon\Logoff event category.
Output	Successful logon (visible in Properties window), number Event: 528 and Type Date Time Source Category User Computers information field filled with correct data.

RE

RE

[AU]-T04	
Purpose	Demonstrate that security the security relevant event “Logon Failure” has been recorded.
Initialization phase	Execute test [IA]-T01
Positive test	
Expected result	The security relevant event generated during initialization phase is recorded in the audit file and the Event Viewer Security log displays it.
Input data	Path to follow and data to fill in the proper fields.
Procedure	Follow the procedure of test [AU]-T01 to reach the Event Viewer console and the procedure of test [AU]-T02 to select Logon\Logoff event category.
Output	Logon Failure: Unknown user name or bad password (visible in Properties window), number Event: 529 and Type Date Time Source Category User Computers information field filled with correct data.

RE

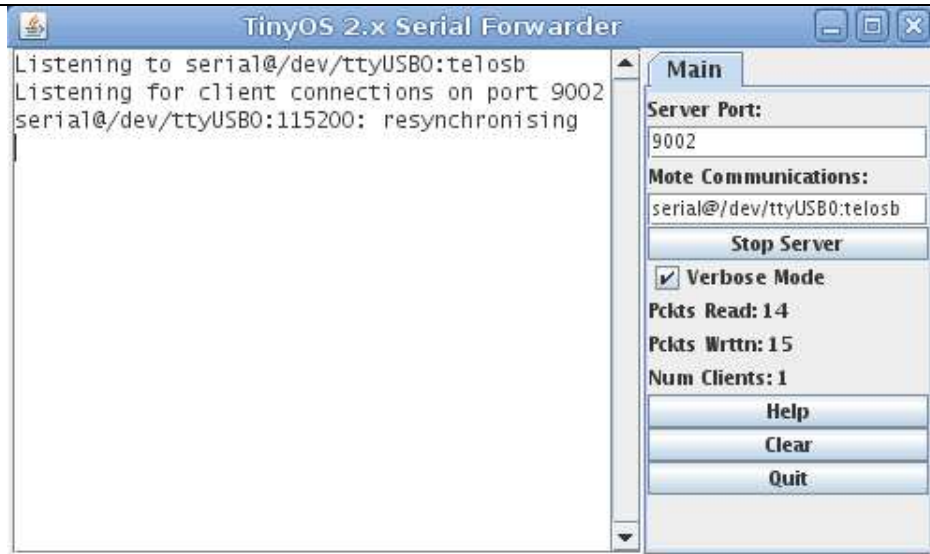
RE

[AU]-T05	
Purpose	Demonstrate that the security relevant event “Logon attempt” has been recorded.
Initialization phase	Execute test [IA]-T01
Positive test	
Expected result	The security relevant event generated during initialization phase is recorded in the audit file and the Event Viewer Security log displays it.
Input data	Path to follow and data to fill in the proper fields.
Procedure	Follow the procedure of test [AU]-T01 to reach the Event Viewer console and the procedure of test [AU]-T02 to select Logon\Logoff event category.
Output	A logon attempt was made by using a disabled account(visible in Properties window), number Event: 531 and Type Date Time Source Category User Computers information field filled with correct data.

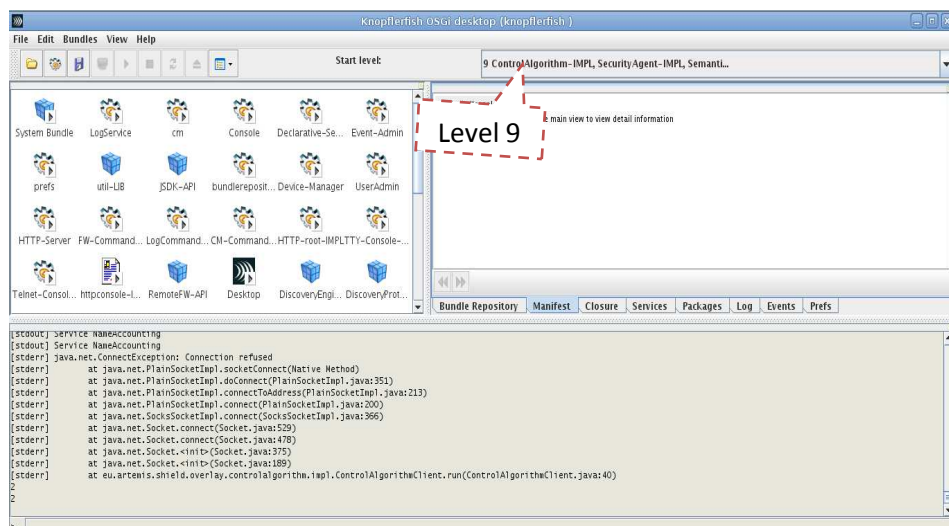
RE

6.2.2 CS – Cryptographic Support

[CS]-T01	
Purpose	Verify the Power Node synchronization with Sensor Node.
Initialization phase	Ensure to have assembled CC2530DK device on SmartRF05 Evaluation Board connected to a computer, and make sure that all system (computer and board) are set to packet sniffer operating mode Errore. L'origine riferimento non è stata trovata..
Positive test	
Expected result	Verify on packet sniffer application interface, the exchange of messages between Power node and Sensor Node that proves the synchronization between these two devices.
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1, 2, 3, 4 or 5 stars
Procedure	<ol style="list-style-type: none"> 1. Start TinyOS 2.x Serial Forwarder <pre>#java net.tinyos.sf.SerialForwarder -comm serial@/dev/ttyUSB0:telosb</pre> 2. Start PShield system (knopflerfish container) (On pShield knopflerfish directory) <pre>#cd osgi</pre> <pre>#sudo java -jar framework.jar -init</pre> 3. Change knopflerfish start level of container to level 9. 4. Start android pSHIELD application (On avd android emulator directory) <pre>sudo ./emulator -avd android &</pre> 5. On android pSHIELD application set the computer running knopflerfish container IP and choice one SPD level, pressing one of the stars (in example, level 1 was chosen).
Output	<ol style="list-style-type: none"> 1. After first procedure the TinyOS 2.x Serial Forwarder GUI opens and make automatically the synchronizing with Power Mote, if it is correctly connected to the machined and it has the correctly permissions:

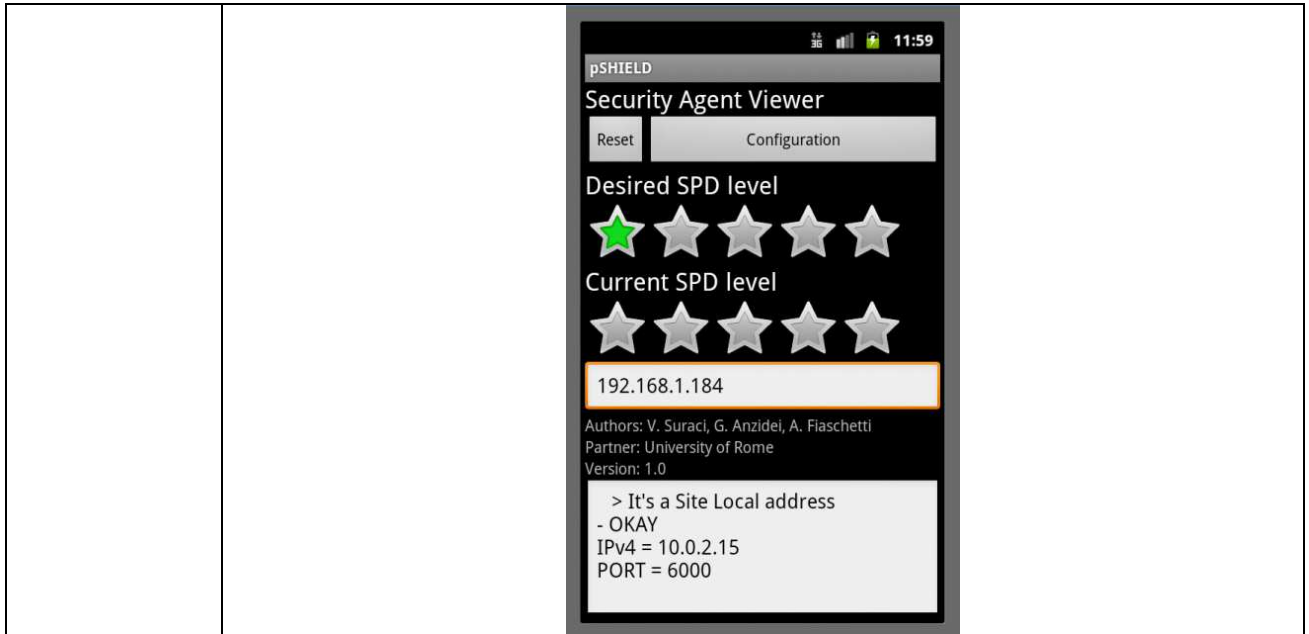


2. After second task, knopflerfish GUI opens like the picture below.



3. After fifth task

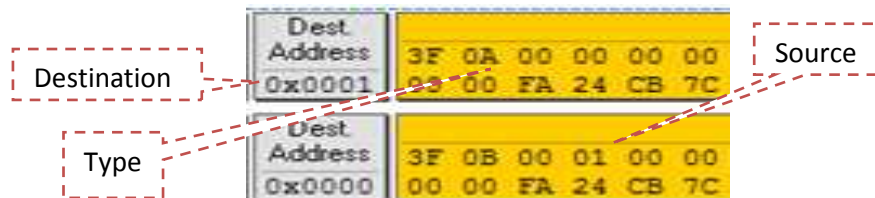
RE



4. Messages caught by packet sniffer regarding to the synchronization protocol

P.nbr.	Time (ms)	Dest. Address	MAC payload	
RX	+0		3F 0A 00 00 00 00 58 75 74 B0 13 E7 04 96 A8 44 E7 C8 AA E8 4A DA 53 0C 8F D3	FCS
1	=0	0x0001	00 00 FA 24 CB 7C 9E 14 6D DA 1B 43 80 A6 95 6E 40 C3 7B D2 0D 47 00 00 0F 7A	OK
P.nbr.	Time (ms)	Dest. Address	MAC payload	
RX	+16		3F 0B 00 01 00 00 58 75 74 B0 13 E7 04 96 A8 44 E7 C8 AA E8 4A DA 53 0C 8F D3	FCS
2	=16	0x0000	00 00 FA 24 CB 7C 9E 14 6D DA 1B 43 80 A6 95 6E 40 C3 7B D2 0D 47 00 00 C2 98	OK

Details of the destination, message type and origin of the messages.



Destinations: 0x0000 (Power Node) and 0x0001 (Sensor Node)

Type: 0A (Synchronization) and 0B (Acknowledge synchronization)

Source: 0000 (Power Node) and 0001 (Sensor node)

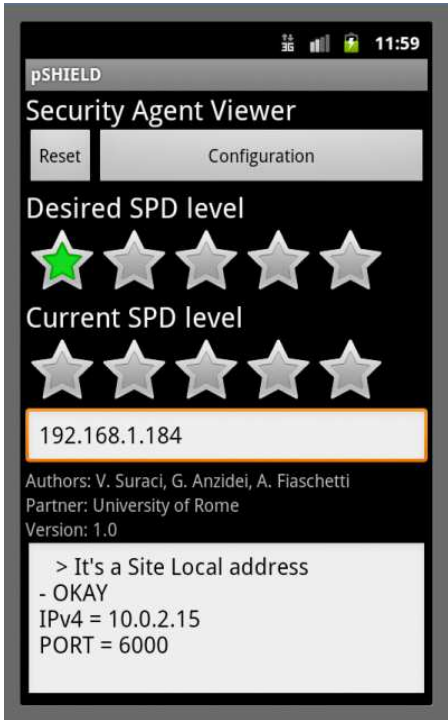
Detail of all message payload and public key exchange (in green).

MAC payload
3F 0A 00 00 00 00 58 75 74 B0 13 E7 04 96 A8 44 E7 C8 AA E8 4A DA 53 0C 8F D3
00 00 FA 24 CB 7C 9E 14 6D DA 1B 43 80 A6 95 6E 40 C3 7B D2 0D 47 00 00 0F 7A
MAC payload
3F 0B 00 01 00 00 58 75 74 B0 13 E7 04 96 A8 44 E7 C8 AA E8 4A DA 53 0C 8F D3
00 00 FA 24 CB 7C 9E 14 6D DA 1B 43 80 A6 95 6E 40 C3 7B D2 0D 47 00 00 C2 98

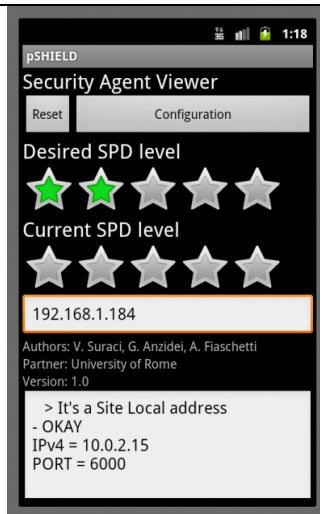
RE

[CS]-T02	
Purpose	Verify the Power Node synchronization with three Sensor Nodes.
Initialization phase	Connect three sensor nodes instead of only one.
Positive test	
Expected result	Verify on packet sniffer application interface, the exchange of messages between Power node and all Sensor Nodes that proves the synchronization between them.
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1, 2, 3, 4 or 5 stars
Procedure	The procedure is the same as the test [CS]-T01.
Output	The output should be similar to test [CS]-T01 output. However, instead of only a pair of messages, for four nodes it gives three pairs.

RE

[CS]-T03	
Purpose	Verify the changing of desired level on Power and a Sensor node
Initialization phase	None
Positive test	
Expected result	Verify on packet sniffer application interface, the exchange of messages between Power node and all Sensor Nodes that proves a changing between current levels to desired level.
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1, 2, 3, 4 or 5 stars
Procedure	<p>With pSHIELD android application, change SPD level from the current level to another one (From the first level for example as the image below).</p> 
Output	After procedure, pSHIELD android application will appear like this:

RE



Messages caught by packet sniffer regarding to the changing level protocol.

Type	P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS	Source
	RX 16	+10117 =31253	0x0001	3F 14 00 00 00 0C 00 01 65 40	OK	
Destination	P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS	Level
	RX 17	+14 =31267	0x0000	3F 15 00 01 00 03 00 01 E3 20	OK	

Destinations: 0x0000 (Power Node) and 0x0001 (Sensor Node)

Type: 14 (Change level) and 15 (Acknowledge change level)

Source: 0000 (Power Node) and 0001 (Sensor node)

Note: Message payload is understandable because it isn't encrypted.

RE

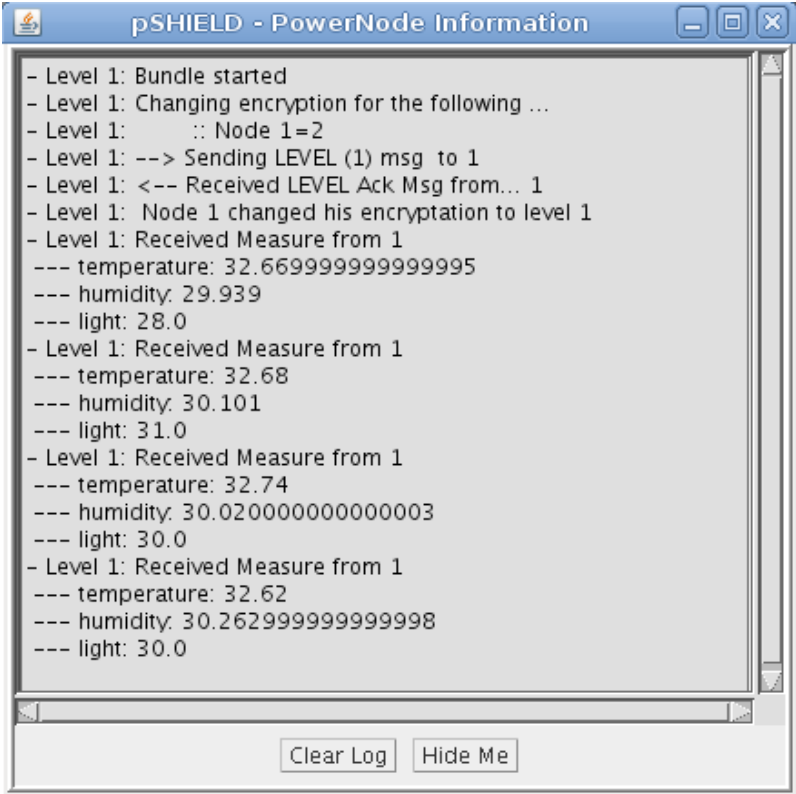
[CS]-T04																										
Purpose	Verify that data transmitted between WSN node is encrypted in level 1 (AES 128).																									
Initialization phase	Execute the test [CS]-T01.																									
Positive test																										
Expected result	Verify on packet sniffer application interface, the exchange of messages between Power node and all Sensor Nodes is incomprehensible without decryption when level of encryption is 1 (AES 128).																									
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1.																									
Procedure	With pSHIELD android application, change SPD level from the current level to level 1.																									
Output	<p>Encrypted messages caught by packet sniffer. Level of encryption = 1 (AES 128).</p> <p style="text-align: center;">Measure message.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>P.nbr.</th> <th>Time (ms)</th> <th>Dest. Address</th> <th>MAC payload</th> <th>FCS</th> </tr> </thead> <tbody> <tr> <td>Type</td> <td>=154904</td> <td>0x0000</td> <td>3F 1E FE 05 F3 2B E3 2E 47 1B 13 5E BD AE A4 31 F0 7D</td> <td>OK</td> </tr> </tbody> </table> <p style="text-align: center;">Change level messages.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>P.nbr.</th> <th>Time (ms)</th> <th>Dest. Address</th> <th>MAC payload</th> <th>FCS</th> </tr> </thead> <tbody> <tr> <td>RX 22</td> <td>+62148 =342045</td> <td>0x0001</td> <td>3F 14 7A 20 B0 00 8F E9 84 D3 BC F4 AF 1E 7D 29 D0 73</td> <td>OK</td> </tr> <tr> <td>RX 23</td> <td>+18 =342064</td> <td>0x0000</td> <td>3F 15 D4 22 E9 46 47 4C 06 87 60 05 83 42 40 99 09 5A</td> <td>OK</td> </tr> </tbody> </table> <p>Type: 1E (Measure), 14 (Change level) and 15 (Acknowledge change level)</p> <p>The messages payload of each message has 18 bytes. The first byte is TinyOS reserved, the second one the message type. The other 16 bytes (128 bits) are the encrypted message.</p>	P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS	Type	=154904	0x0000	3F 1E FE 05 F3 2B E3 2E 47 1B 13 5E BD AE A4 31 F0 7D	OK	P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS	RX 22	+62148 =342045	0x0001	3F 14 7A 20 B0 00 8F E9 84 D3 BC F4 AF 1E 7D 29 D0 73	OK	RX 23	+18 =342064	0x0000	3F 15 D4 22 E9 46 47 4C 06 87 60 05 83 42 40 99 09 5A	OK
P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS																						
Type	=154904	0x0000	3F 1E FE 05 F3 2B E3 2E 47 1B 13 5E BD AE A4 31 F0 7D	OK																						
P.nbr.	Time (ms)	Dest. Address	MAC payload	FCS																						
RX 22	+62148 =342045	0x0001	3F 14 7A 20 B0 00 8F E9 84 D3 BC F4 AF 1E 7D 29 D0 73	OK																						
RX 23	+18 =342064	0x0000	3F 15 D4 22 E9 46 47 4C 06 87 60 05 83 42 40 99 09 5A	OK																						

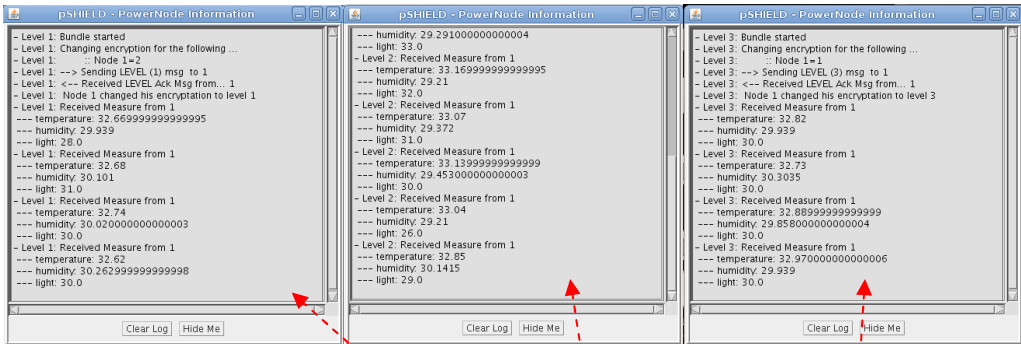
RE

RE

[CS]-T05	
Purpose	Verify that data transmitted between WSN node is encrypted for level 2 (AES 256) and level 3 (ECC / ECIES).
Initialization phase	Execute the test [CS]-T01.
Positive test	
Expected result	Verify on packet sniffer application interface, the exchange of messages between Power node and all Sensor Nodes is incomprehensible without decryption when level of encryption is 2 (AES 256) and 3 (ECC / ECIES).
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 3, 5 stars
Procedure	With pSHIELD android application, change SPD level from the current level to level 2 (3 stars) to test AES 256 encryption. Change SPD level from the current level to level 3 (5 stars) to test ECC / ECIES encryption.
Output	Encrypted messages collected by packet sniffer. For level 2 (AES 256) each message will have 2 + 32 bytes of length. 32 bytes = 256 bits. For level 3 (ECC/ECIES) there aren't a specific message size.

RE

[CS]-T06	
Purpose	Verify that data acquired by one Sensor Node are correctly transmitted to OSGI platform.
Initialization phase	Execute the test [CS]-T01.
Positive test	
Expected result	Receive on OSGI platform GUI windows the sensor measures, with real values, sent over WSN.
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1 stars
Procedure	Wait for data.
Output	<p>GUI window from OSGI pSHIELD platform showing measures given by Sensor nodes through special GUI windows for level 1.</p>  <p>As can be seen, the messages should be presented decrypted.</p>

[CS]-T07	
Purpose	Verify that data acquired by one Sensor Node are correctly transmitted to OSGI platform and it is level independent.
Initialization phase	Execute the test [CS]-T01.
Positive test	
Expected result	Receive on OSGI platform GUI windows sensors measures, real values sent over WSN, showing level independent similar values.
Input data	Start level for Knopflerfish container – 9 Desired SPD level for pSHIELD android application – 1, 3, 5 stars
Procedure	Wait for data.
Output	<p>GUI windows from OSGI pSHIELD platform showing measures given by Sensor nodes for one of each level, through special GUI windows for each of the three levels.</p>  <p>Details from above images:</p> <pre> - Level 1: Received Measure from 1 --- temperature: 32.68 --- humidity: 30.101 --- light: 31.0 - Level 2: Received Measure from 1 --- temperature: 32.85 --- humidity: 30.1415 --- light: 29.0 - Level 3: Received Measure from 1 --- temperature: 32.970000000000006 --- humidity: 29.939 --- light: 30.0 </pre> <p>As it can be seen, for different levels of encryption, besides the measures should be correctly sent and decrypted by Power Node.</p>

RE

[CS]-T08	
Purpose	Reliability of Sensor communication
Initialization phase	Execute the test [CS]-T07.
Positive test	
Expected result	Receive on OSGI platform GUI windows the sensors measures, real values sent over WSN, showing similar values independent of level.
Input data	Only the environment data should change.
Procedure	The system should run for one hour without a break.
Output	The output should be similar from the test [CS]-T07 and the devices should communicate without major interruption for one hour.

RE

6.2.3 Identification and authentication tests

[IA]-T01	
Purpose	Demonstrate that no SPD functionalities mediated action on behalf of a user can be performed before user is identified and authenticated.
Initialization phase	No initialization phase
Positive test	
Expected result	A correct identification and authentication process permit a user to access SPD functionalities mediated action on behalf of the user itself.
Input data	Data to fill in the logon window.
Procedure	<p>Initiate a trusted path for login by pressing CTRL+ALT+DELETE.</p> <p>If the administrator has implemented a log on banner, a message banner will appear on the screen.</p> <p>Read the message and click OK, or hit <Enter> to continue with the logon process.</p> <p>At the Log On to Windows interface, enter a user name and password for an authorised user.</p> <p>Click on the Options >> button. In the Log on to: drop down box select to either log on to the local computer.</p> <p>Click OK.</p> <p>A Windows XP user session start and the user can access SPD functionalities mediated action on behalf of the user itself.</p>
Output	Logon success permit to an authorised user to access SPD functionalities mediated action on behalf of the user itself.

RE

[IA]-T02	
Purpose	Demonstrate that no SPD functionalities mediated action on behalf of a user can be performed before user is identified and authenticated.
Initialization phase	No initialization phase
negative test	
Expected result	A wrong identification and authentication process do not permit a user to access SPD functionalities mediated action on behalf of the user itself.
Input data	Data to fill in the logon window.
Procedure	<p>Initiate a trusted path for login by pressing CTRL+ALT+DELETE.</p> <p>If the administrator has implemented a log on banner, a message banner will appear on the screen.</p> <p>Read the message and click OK, or hit <Enter> to continue with the logon process.</p> <p>At the Log On to Windows interface, enter the user name for an authorised user and a wrong password (e.g.: change the last character of the password).</p> <p>Click on the Options >> button. In the Log on to: drop down box select to either log on to the local computer.</p> <p>Click OK.</p> <p>It appear a logon message with the following contents “The system could not logon you. Make sure your User name and domain are correct, then type your password again. Letter in passwords must be typed using the correct case.</p>
Output	Logon success permit to an authorised user to access SPD functionalities mediated action on behalf of the user itself.

RE

RE

[IA]-T03	
Purpose	Demonstrate that a disabled user cannot access to SPD functionalities mediated action on behalf of the user itself.
Initialization phase	Process of user disabling:
Negative test	
Expected result	An identification and authentication process do not permit a disabled user to access SPD functionalities mediated action on behalf of the user itself.
Input data	Data to fill in the logon window.
Procedure	<p>Initiate a trusted path for login by pressing CTRL+ALT+DELETE.</p> <p>If the administrator has implemented a log on banner, a message banner will appear on the screen.</p> <p>Read the message and click OK, or hit <Enter> to continue with the logon process.</p> <p>At the Log On to Windows interface, enter a user name and password for a disabled user.</p> <p>Click on the Options >> button. In the Log on to: drop down box select to either log on to the local computer.</p> <p>Click OK.</p> <p>It appear a logon message with the following contents "The system could not logon you. Make sure your User name and domain are correct, then type your password again. Letter in passwords must be typed using the correct case.</p>
Output	Logon process do not permit to a disabled user to access SPD functionalities mediated action on behalf of the user itself.

RE

6.2.4 Protection of the SPD functionalities tests

[PT]-T01	
Purpose	Verify that pSHIELD platform receive a reliable timestamp by an authorative external time source.
Initialization phase	No initialization phase
Positive test	
Expected result	pSHIELD Audit Events associated Timestamp is reliable. T.
Input data	No input data.
Procedure	<p>The Windows Time service (W32Time) is designed to maintain date and time synchronization for computers running Windows 2000XP/2003.</p> <p>W32Time is based on the Simple Network Time Protocol (SNTP) as specified in RFC RFC 1769 (now superceded by RFC 2030).</p> <p>As described in par. 4.2 pSHIELD Central Unit have its own clock controlled By synching to a Bluetooth connected external GSM device, that is an authorative source.</p> <p>Test verification is realized checking the configuration indicated in the following steps:On the Windows XP desktop click Start, click Run, type regedit, and then click OK.</p> <p>1 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters\ In the right panel, right-click Type, and then click Modify. In the Edit Value dialog box, under Value data, verify that value is "NTP", and then click OK.</p> <p>2 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Config\ In the right panel, right-click AnnounceFlags, and then click Modify. In the Edit DWORD Value dialog box, under Value data verify that value is "5" and then click OK.</p> <p>3 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient\ </p>

RE

	<p>In the right panel, right-click SpecialPollInterval, and then click Modify.</p> <p>In the Edit DWORD Value dialog box, under Value data verify that value is "TimeInSeconds" and then click OK.</p> <p>4 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpServer\</p> <p>In the right pane, right-click Enabled, and then click Modify.</p> <p>In the Edit DWORD Value dialog box, under Value data verify that data is "1", and then click OK.</p> <p>5 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters</p> <p>In the right panel, right-click NtpServer, and then click Modify.</p> <p>In Edit Value verify that data is "192.168.1.7" (mobile terminal IP address) and then click OK.</p> <p>6 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Config\</p> <p>In the right panel, right-click MaxPosPhaseCorrection, and then click Modify.</p> <p>In the Edit DWORD Value dialog box, click Decimal under Base.</p> <p>In the Edit DWORD Value verify that data is "54000", and then click OK.</p> <p>7 - Locate and then click the following registry subkey: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Config\</p> <p>In the right pane, right-click MaxNegPhaseCorrection, and then click Modify.</p> <p>In the Edit DWORD Value dialog box, click Decimal under Base.</p> <p>In the Edit DWORD Value verify that data is "54000", and then click OK.</p>
Output	The regedit configuration of the Central Unit Operating System is correct.

RE

6.2.5 SPD Functions Management Tests

[SM]-T01	
Purpose	Demonstrate the possibility for the authorized administrators to manage a users' passwords policy which determines settings for password such as enforcement and lifetimes.
Initialization phase	N.A.
Positive test	
Expected result	<p>The security provided by a password system depends on the passwords being kept secret at all times. Thus, a password is vulnerable to compromise whenever it is used, stored, or even known. In a password-based authentication mechanism implemented on a system, passwords are vulnerable to compromise at several essential stages related to password assignment, distributions, management, and use. For this reason it is expected that the system is able to enforce:</p> <ul style="list-style-type: none"> • password history; • maximum password age; • minimum password age; • minimum password length; • password complexity requirements.
Input data	Path to follow and data to fill in the proper fields.
Procedure	<ol style="list-style-type: none"> 1. Log on as an authorized administrator. 2. Click Start, point to Settings and click on then Control Panel 3. Double-click on Administrative Tools, and then on Local Security Policy. 4. In the console tree, expand Account Policy and click on Password Policy.
Output	<p>In the right panel of the console password policy enforcing tools appear described as follows:</p> <ul style="list-style-type: none"> • Enforce Password History: The Enforce password history setting determines how frequently old passwords can be reused. This policy can be used to discourage users from changing back and forth between a set of common passwords. Windows XP Professional can store up to 24 passwords for each user in the password history. By default, this policy is set to zero (0) in Windows XP Professional, which disables the

	<p>password history policy.</p> <ul style="list-style-type: none">• Maximum Password Age: The Maximum password age setting determines how long users can keep a password before they have to change it. The aim is to periodically force users to change their passwords. When this feature is used, set a value that makes sense for the specific network environment it is being applied to. Generally, a shorter period is used when security is very important and a longer period when security is less important. The default expiration date is 42 days; however, it can be set to any value from 0 to 999• Minimum Password Age: The Minimum password age setting determines how long users must keep a password before they can change it. This field can be set to prevent users from cheating the password system by entering a new password and then changing it right back to the old one. By default, Windows XP Professional lets users change their passwords immediately.• Minimum Password Length: The Minimum password length setting establishes the minimum number of characters required for a password. If it hasn't been changed already, the default setting should be changed immediately. The default is to allow empty passwords (passwords with zero characters), which is definitely not a good idea.• Passwords Must Meet Complexity Requirements: Beyond the basic password and account policies, Windows XP Professional includes facilities for creating additional password controls. The functions implemented by enabling the Passwords must meet complexity requirements setting in Password Policy are enforced when a user or administrator attempts to change the password for a user account. For example, in Windows XP Professional, the strong password filter requires that passwords not contain all or part of the user's account name, not be less than six characters in length, and contain characters from at least three (3) of the following four (4) classes:<ol style="list-style-type: none">1. English Upper Case Letters A, B, C, ... Z2. English Lower Case Letters a, b, c, ... z3. Base ten digits 0, 1, 2, ... 94. Non-alphanumeric (—special characters) For example, !, \$, #, %.
--	--

RE

[SM]-T02	
Purpose	Demonstrate the possibility for the authorized administrators to manage local policy which can be used to configure user rights assignment.
Initialization phase	N.A.
Positive test	
Expected result	The system is able to determine which users or groups have logon or task privileges on the computer.
Input data	Path to follow and data to fill in the proper fields.
Procedure	<p>Log on as an authorized administrator.</p> <ol style="list-style-type: none"> 1. Open the Local Security Policy. Click Start, point to Administrative Tools, and then click Local Security Policy. This opens the Local Security Settings console. 2. Expand Security Settings. 3. Within Security Settings, expand Local Policies to reveal the Audit, User Rights Assignment, and Security Options policies. 4. Click the User Rights Assignment object
Output	<p>The details pane will reveal the configurable user rights policy settings as follows described:</p> <ol style="list-style-type: none"> 1. To set a user Logon Right or Privilege, double-click the desired policy in the details pane. This will open the user right Properties dialog window. 2. To remove a Logon Right or Privilege for an account, click the account name to highlight it and click the Remove button. 3. To add a Logon Right or Privilege to an account, click the Add button and browse the appropriate account directory for the desired account.

RE

6.3 Completeness of tests

In the following table it is demonstrated how tests set defined in paragraph 6.2 is complete to verify pSHIELD platform SPD functionalities. Table rows contain SPD functionalities and columns contain tests. SPD functionalities and tests are grouped by classes.

		AU					CS								IA			PT		SM	
		T01	T02	T03	T04	T05	T01	T02	T03	T04	T05	T06	T07	T08	T01	T02	T03	T01	T02	T01	T02
Class	Comp.																				
AU	GEN.1	X	X	X	X	X															
	GEN.2	X	X	X	X	X															
	SAR.1	X	X	X	X	X															
	SAR.3		X	X	X	X															
CS	CKM.1						X	X	X	X	X	X	X								
	CKM.2						X	X	X	X	X	X	X								
	COP.1						X	X	X	X	X	X	X								
IA	UID.1													X	X	X					
	UAU.1													X	X	X					
PT	STM.1																X	X			
SM	MTD.1																			X	
	SMR.1																			X	
	SMF.1																			X	

Table 6 - 1 Cross-reference verification

7 Conclusions

The purpose of the pSHIELD project has been to put the basis of SPD composability in Embedded System domain and to demonstrate it by means of a reduced, but significant scenario. At first, a series of innovative concepts and technologies has been developed in technical WPs (WP3-4-5) and then a subset of these results has been selected for further integration in a common platform that constitute the “pSHIELD Demonstrator”, carefully tailored on railway scenario.

The present document has been focused on: i) the validation and verification of the integrated platform and ii) the validation and verification of its behaviour

- The objective of the platform validation activity has been focused on checking the consistency of the platform components in terms of functionalities, semantic models (e.g. metrics and descriptions) and interfaces necessary to enable and perform the SPD composability.
- The objective of the platform verification activity has been to verify the platform behavior with respect to the selected scenario by means of focused functional tests.

The major platform validation has been based on the presence of the innovative pSHIELD Middleware that is able to dynamically discover and compose the SPD functionalities offered by the system elements. This middleware is the first enabler of the SPD-aware composability.

The second enabler is the semantic model used to describe the pSHIELD components and the SPD functionalities, since it feeds the metrics-driven composition performed at middleware level.

In conclusion the project demonstrator is able to perform SDP-aware composition by using specific Middleware Services and semantic models and this makes it pSHIELD-enabled. However this composition can either be correct or not. The functional tests have verified that this composition is also correct, thus validating also the metrics composition approach.

This platform will be the basis of the enrichment and further development that will be carried out during the prosecution of the research with the nSHIELD project.

References

- [1] Windows XP Professional with SP2 Evaluated Configuration User's Guide – version 3.0 – July 11, 2007
- [2] MOTE IV - Telos revB (Low Power Wireless Sensor Module) - Manual.
- [3] CROSSBOW - TPR2400/2420 Quick Start Guide - Rev. A, May 2005 – Doc. 7430-0380-01
- [4] H.Wang, B. Sheng, C.C. Tan and Qun Li, WM-ECC: an Elliptic Curve Cryptograph Suite on Sensor Motes, Technical report, Oct. 30, 2007
- [5] V. Casola, A. De Benedictis, A. Mazzeo and N. Mazzocca, SeNsIM-SEC: security in heterogeneous sensor networks, May 2011, SARSSI2011
- [6] TinyOs, <http://www.tinyos.net/>
- [7] Cryptography algorithms for TinyOS, <http://tinyos.cvs.sourceforge.net/viewvc/tinyos/tinyos-2.x-contrib/crypto/index.html>
- [8] TinyECC: A Configurable Library for Elliptic Curve Cryptography, in Wireless Sensor Networks, <http://discovery.csc.ncsu.edu/software/TinyECC/>
- [9] Windows XP Professional - Security configuration guide - version 3.0 – July 18, 2007
- [10] SmartRF packet sniffer user manual – SWRU187F – Texas Instruments – revision F – July 07, 2011
- [11] CC2530 Development kit user's guide – SWRU208b – Texas Instruments – revision B – April 23, 2010