Project no: 269317

nSHIELD

new embedded Systems arcHItecturE for multi-Layer Dependable solutions

Instrument type: Collaborative Project, JTI-CP-ARTEMIS

Priority name: Embedded Systems

D2.1: Preliminary System Requirements

Due date of deliverable: M3 –2011.11.30

Actual submission date: M6 – 2012.02.29

Start date of project: 01/09/2011 Duration: 36 months

Organisation name of lead contractor for this deliverable:

THYIA Tehnologije, THYIA

Revision [Draft C]

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 Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	X

Document Authors and Approvals

Authors		Date	Signature
Name	**Company**		
Gordana Mijic	THYIA		
Spase Drakul	THYIA		
Nastja Kuzmin	THYIA		
Ljiljana Mijic	THYIA		
Harry Manifavas	TUC		
Konstantinos Fysarakis	TUC		
Dimitris Geneiatakis	TUC		
Alexandros Papanikolaou	TUC		
Konstantinos Rantos	TUC		
Balázs Berkes	S-LAB		
Alfio Pappalardo	ASTS		
Luigi Trono	SG		
Francesco Cennamo	SG		
Arash Vahidi	SICS		
Christian Gehrmann	SICS		
Hans Thorsen	T2Data		
Paolo Azzoni	ETH		
Stefano Gosetti	ETH		
Antonio Abramo	UNIUD		
Mirko Loghi	UNIUD		
Andrea Morgagni	SE		
Reviewed by			
Name	**Company**		
Approved by			
Name	**Company**		

Applicable Documents		
ID	**Document**	**Description**
[01]	TA	nSHIELD Technical Annex

Modification History		
Issue	**Date**	**Description**
Draft A	29.11.2011	First ToC
Draft B	30.12.2011	Partners contribution
Draft C	29.2.2012	Partners contribution

Contents

Figures

Glossary

Please refer to the Glossary document, which is common for all the deliverables in nSHIELD.

This Page is intentionally left blank

1 Executive Summary

The scope of the present document is to provide top-level requirements extracted from the nSHIELD scenarios defined in the internal deliverables and from the Technical Annex approved by the Artemis JU[1]. The System Requirements Specification (SRS) document covers all requirements on the overall nSHIELD system.

The methodology for specifying the requirements is based on a common agreement described in deliverables ID2.1 titled “Requirements and Specifications Definitions and Rules – Quality Manual for WP2”.

Since the SPD (Security, Privacy, and Dependability) requirements are based on the selected scenarios, i.e., Railroad Security (RS), Voice/Facial Recognition (VFR), Dependable Avionic Systems (DAS) and Social Mobility and Networking (SMN) it is important to take these scenarios in consideration in the early development phase of the high level requirements. Therefore, in chapter 7 a set of high level requirements (HLR) for the selected scenario are defined. Chapter 8 is providing SPD HLR for the nSHIELD system. These means, the system requirements can be considered as requirements that define the system before it will be specified in the deliverable D2.3, where the whole nSHIELD system architecture will be defined. This document covers all requirements on the nSHIELD system functionalities. Special focus is given on SPD and composability. Therefore, for each SPD technology and for each functional layer, a formal set of high level requirements for the functional architecture are defined. Chapter 9, 10, 11 and 12 covers requirements and specifications for the node, network, middleware and overlay functional layers. In Chapter 13 the conclusions are summarized.

In this document the concepts of Security, Privacy and Dependability are formalized via a reference taxonomy that allows describing them with different attributes. The formal definition of each attribute is a fundamental step towards the correct finalization of requirements and specification. This will be the input for the other WPs where the nSHIELD system will be designed with additional details.

The systems definition of the scenarios, the overall system and its four fundamental functional layers definition (i.e., node, network, middleware and overlay) requires also a careful consideration of the system elements that are targeted for nSHIELD.

The deliverable D2.1 is a Preliminary System Requirements document and is a starting point for requirements for the demonstrator. For this purpose one new document will be written. The following deliverable will be D2.2 named Preliminary System Requirements and Specification. The final deliverable D2.6 will be Final System Requirements and Specifications which will consider feedbacks from other WPs. The document D2.6 will be refined on the basis of the results of the validation phase and on the detailed description of the application scenarios from WP7.

[1] Technical Annex of nSHIELD project.

2 Introduction

This document will define the requirements and specifications of the overall nSHIELD system. For each Security, Privacy and Dependability (SPD) technology, for each layer: node, network, middleware and overlay a formal set of high level, architectural, interface and performance requirements will be provided. This task will be influenced by the WP7 application scenarios. These scenarios will be taken as a reference for defining the SPD requirements of each architectural layer (even though the conceived architecture will be able to support any ES scenario).

This deliverable is the main output of Task 2.1 from WP2. The role of the task is to identify the requirements and provides inputs for the specifications of the overall nSHIELD system that will be performed by task 2.3. This task will be strongly influenced by the selected application scenarios Railroad Security (RS), Voice/Facial Recognition (VFR), Dependable Avionic Systems (DAS) and Social Mobility and Networking (SMN).

Requirements and specifications have been be also influenced by the liaisons activated in other WPs, like WP3, WP4, WP5, WP6 and WP7. .

For each layer a formal set of high level, architectural, interface requirements will be identified. The selected scenarios will be taken as a reference for defining the SPD requirements of each architectural layer; however the conceived architecture will be able to support a generic Embedded System (ES) scenario, as well as of the overall system with reduced and clearly identifiable tailoring effort.

An iterative approach will be adopted. A preliminary set of HLR specification will be provided in this phase of the project. The preliminary outcome of this task will be used by WP3, WP4, WP5 and WP6 to develop potential prototypes and to validate them. Vice versa, the application specifications in WP7 will drive development of all requirements in this document. Thus, the requirements and specifications will be refined on the basis of the results of the validation phase and on the detailed description of the application scenarios in WP7.

The application scenarios of reference for the task 2.1 of nSHIELD project are the Railroad Security, Voice/Facial Recognition, Dependable Avionic Systems and Social Mobility and Networking SMN**.** This Deliverable D2.1.1 aims at giving precise definitions characterizing the various concepts that come into play when addressing the security, dependability and privacy of complex systems resulting from the composition of elementary Embedded Systems. For that reason most of the results obtained in this phase will be natively tailored on the selected scenarios, but this doesn't mean that the potentiality of the framework are limited to them. The reusability of the nSHIELD solutions with a minimum tailoring effort is indeed one of its main features in this project.

This will be reflected also in the formalization of System Requirements that will be divided in two sets.

- The first set is about System Requirements specific for the scenarios (i.e. requirements that are valid only in the scope of the selected applications);
- The second set is about general System Requirements that characterize the nSHIELD framework independently from the applications.

3 Terms and Definitions

This section should be coherent with deliverables, D2.5 and D2.3 and provides a set of terms and definition that will be used in the rest of the document as well as in the rest of the project. Since some definitions could vary a bit according to the application scenarios, they will be listed in a separate paragraph in order to facilitate further adaptation of the document. The final adjustment of the terms and definitions that are common will be adopted at the end of the project.

3.1 The nSHIELD System: General Definitions

[nSHIELD System] - The nSHIELD system (a whole composed by several parts) is a set of interacting and/or interdependent system components forming an integrated and more complex system.

The main characteristics of the nSHIELD system are:

1. The *architecture* defined by components and the results of their composition,
2. The *behaviour*, that involves collecting inputs, processing them and producing outputs,
3. The *relations* that the various parts of the system have between each other, both functional and structural
4. The *functionalities* or group of functionalities that the system offers and/or realises.

The nSHIELD system aims to guarantee the following main **taxonomy concepts**:

1. Security,
2. Privacy, and
3. Dependability

for itself and for the application scenario on which it is applied. These attributes are indeed the main goals to be addressed for the new generation Embedded Systems.

The **[nSHIELD functional system]** is organized according to the following layering functional layers:

I. **Node Layer** includes the hardware components that constitute the physical part of the system.

II. **Network Layer** includes the communication technologies (specific for the selected scenarios) that allow the data exchange among nSHIELD components, as well as the external world. These communication technologies, as well as the networks to which nSHIELD is interconnected can be (and usually are) heterogeneous.

III. **Middleware Layer** includes the software functionalities that enable the discovery, composition and execution of the basic services necessary to guarantee SPD as well as to perform the tasks assigned to the system (for example, in the railway scenario, the monitoring functionality)

IV. **Overlay Layer** includes the “embedded intelligence” that drives the composition of the nSHIELD components in order to meet the SPD level desired.

[Component/Sub-system] - A component or sub-system is a smaller, self-contained part of a system. In particular for the nSHIELD system the “interacting components” are Embedded Systems.

[Embedded System/Device] - The Embedded System (or Device) is an electronic system (or device) dedicated to a specific and reduced set of functionalities. It could be an integrated circuit that has input, output and processing capabilities or more commonly it is a small programmable chip. The embedded systems are controlled by one or more main processing cores that are typically either microcontrollers or digital signal processors (DSP).

[Asset categories] - An asset can be grouped in two categories: logical and physical assets. Information, services and software are logical assets, whilst human beings, hardware or particular physical objects are physical assets.

[User] – User is any entity internal or external, human or IT that interacts with the nSHIELD system.

3.2 The nSHIELD System: Application-Oriented Definitions

[Information] - Information is measured data, real-time streams from audio/video-surveillance devices, smart-sensors, alarms, etc.

[nSHIELD Asset] - The nSHIELD assets are information and services.

[SPD Audit] - SPD auditing involves recognizing, recording, storing, and analyzing information related to SPD relevant activities.

[Non-repudiation] - Assuring the identity of a party participating in a data exchange.

[Access control] - Is the process of mediating every request of access to nSHIELD assets determining whether the request should be granted or denied according to the security established policies.

[Identification] - Determining the identity of users.

[Authentication] - Verifying the identity of users and determining their authority to interact with the system.

[Trusted channel] - A communication channel that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.

[Software failures] – Software failures include crashes, incompatibilities, computation errors, etc.

[Hardware failures] – Hardware Failures include transient faults due to radiation, overheating or power spikes.

[Transmission failures] – Transmission Failures include:

- Repetition (a message is received more than once)
- Deletion (a message is removed from a message stream)
- Insertion (a new message is implanted in the message stream)
- Re-sequencing (messages are received in an unexpected sequence)
- Corruption (the information contained in a message is changed, casually or not)
- Delay (messages are received at a time later than intended)
- Masquerade (a non-authentic message is designed thus to appear to be authentic)

[Failure Mitigation Mechanisms] – Failure Mitigation Mechanisms includes:

- hardware redundancy and diversity
- firewall and intrusion detection systems
- self-checking and diagnostics routines
- message sequence numbers
- data checksums
- shared or public key cryptography
- vitality checks through watchdog timers
- software rejuvenation

[Reasoning] – Reasoning is related to finding one or more solutions (HW/SW configuration) that satisfy the desired SPD level.

[Composition] – Composition is related to verifying the possibility of composing the individual elements and composing them logically

[Configuration] – Configuration is the translation of logical configuration into a physical configuration.

4 The Methodology for nSHIELD System Requirements Specification

The nSHIELD requirements elicitation, as well as the overall work carried out in Task 2.1, is structured according to a number of steps: each step is hereinafter presented following a question/answer approach.

Step 1 – What is the nSHIELD System?

The nSHIELD System is a set of interacting and interconnected Embedded Systems with specific composability and SPD Functionalities.

Reference in this document: Chapter 3 – Terms and Definitions and Chapters 7-12 – System Requirements

Step 2 – What is the role of the nSHIELD System?

nSHIELD aims at assuring SPD for a certain asset or goods in a specific scenario. In the following, for convenience, we will replace the expression “assuring SPD” with the generic expression “protecting”, even if its real meaning is different. In Figure 1 this concept is represented: the green boxes represent the interconnected ESs and the gray box is the addressed asset/goods (the SPD functionalities are still missing from this graphical representation because they are identified in the following steps).

Moreover, since the terms SPD seem too general, a set of specific attributes will be introduced to better specify the meaning of “Assuring SPD” (for example assuring integrity, reliability, confidentiality, and so on). These attributes are specified in the SPD Taxonomy.

Reference in this document: Section 5 – Reference SPD taxonomy

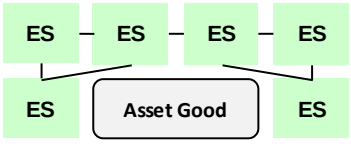


Figure 1: nSHIELD System applied to a specific asset/good.

Step 3 – What are the asset/goods and the scenario addressed by the System?

The selected scenario is railways transportation and the asset/goods is the secure and dependable monitoring of freight trains transporting hazardous materials. Moreover, since nSHIELD should protect itself, it is an asset/good as well.

Reference in this document: Chapter 7 – High level requirements for Scenario

Step 4 – What are the possible menace and/or attack that could affect the protected asset/goods

Once the assets and goods, as well as the application scenarios are clearly identified (from Technical Annex for example), it is easy to enumerate all the possible menaces and attacks that could affect the level of Security, Privacy and Dependability of the system. The output of this activity is a fundamental input for next step. The logical step is represented in Figure 2.

Reference in this document: Chapter 7 – High level requirements for Scenario

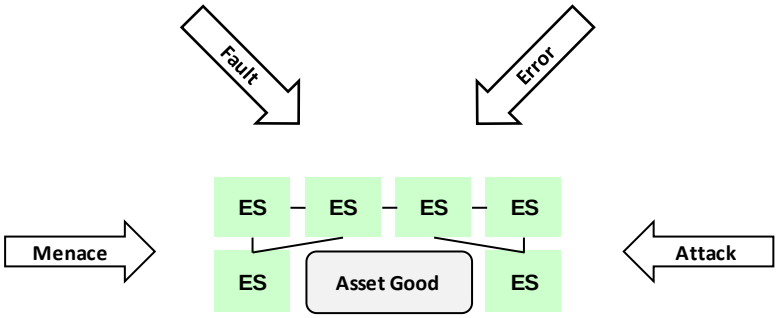


Figure 2: nSHIELD attacks and menaces.

Step 5 – What are the SPD functionalities that can prevent or minimize the effect of the previously identified menace and/or attack?

Starting from the identified menaces and attacks, a set of SPD Functionalities is identified that are able to prevent or mitigate them (see Figure 3). The functionalities are translated in a set of Functional Requirements that are at the basis of the nSHIELD framework. Since these requirements are strictly related to the scenario, they will be listed as Scenario's Requirements.

Reference in this document: Chapter 7 – High level requirements for Scenario

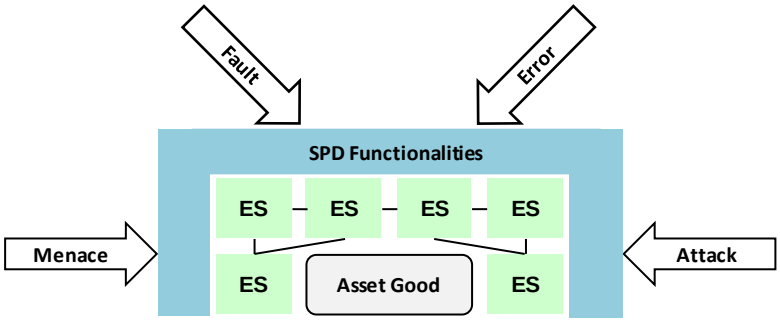


Figure 3: nSHIELD system with SPD functionalities enabled.

Step 6 – What are the features of the system that allow to realize the SPD functionalities?

Once the required SPD Functionalities are captured (in Step 5), in Step 6 a set of System Requirements can be identified allowing to realize these SPD functionalities. These requirements provide the guidelines for the design and development of the four nSHIELD layers: so, these requirements are divided into four categories corresponding to the four different layers.

Reference in this document: Chapter 7-12 – System Requirements

The result of these six steps is the definition, formalization and translation into requirements of the nSHIELD system.

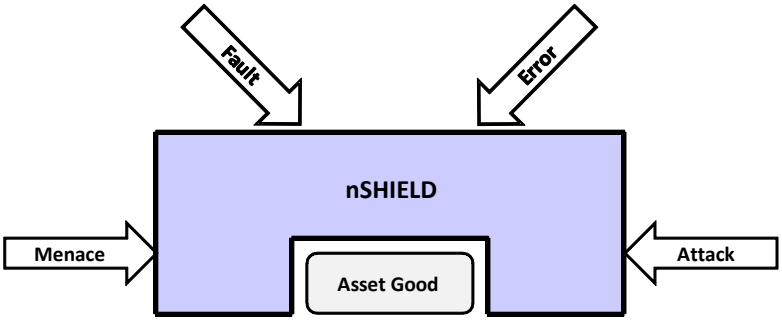


Figure 4: ESs + SPD Functionalities = nSHIELD system.

5 SPD Taxonomy

5.1 Concept Taxonomy

With rapidly developed network technologies and computing technologies, network-centric computing has become the core information platform in our private, social, and professional lives. This information platform is dependent on a computing and network infrastructure, which is increasingly homogeneous and open. The backbone of this infrastructure is the Internet, which is inherently insecure and unreliable. With an ever-accelerating trend of integrating mobile and wireless network infrastructure, things become worse. This is because wireless radio links tend to have much higher bit error rates, and mobility also increases the difficulty of service quality management and security control. The increased complexity of the platform and its easy access has made it more vulnerable to failures and attacks, which in turn has become a major concern for society.

Dependability was first introduced as a general concept covering the attributes of reliability, availability, safety, integrity, maintainability, and so on. With ever increasing malicious catastrophic Internet attacks, Internet providers have realized a need to incorporate security issues. Effort has been made to provide basic concepts and taxonomy to reflect this convergence.

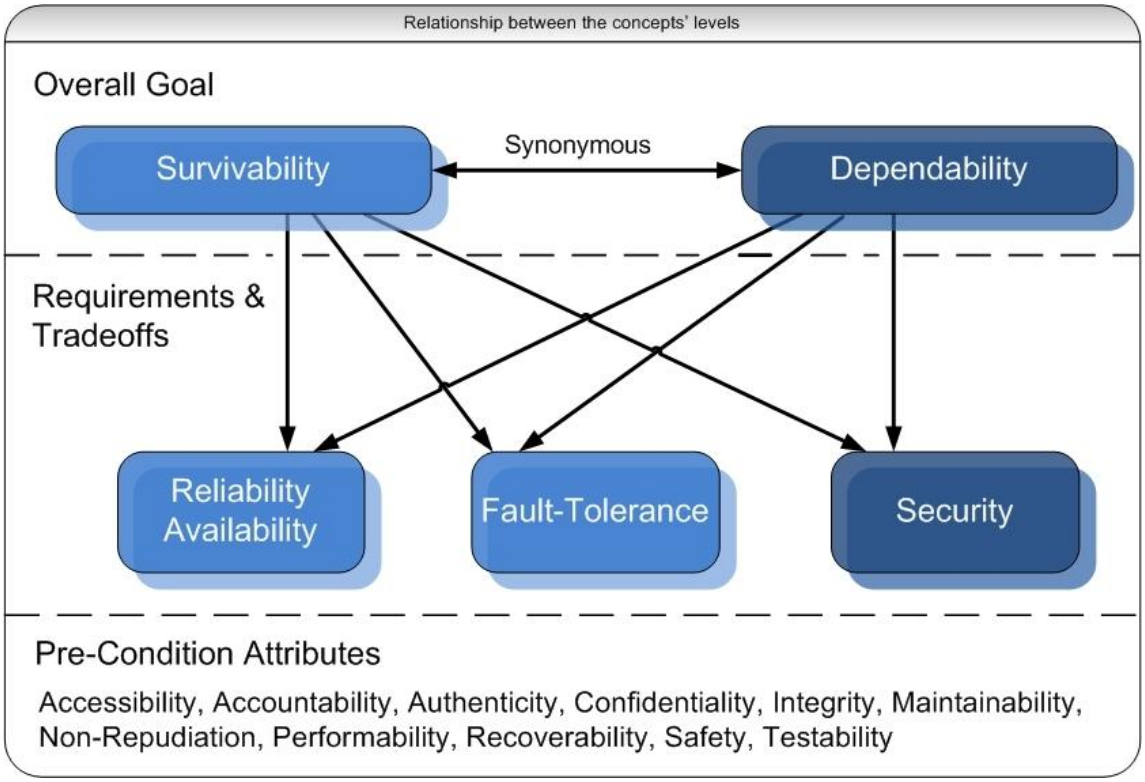


Figure 5: Relationship between the concepts' levels

Although there is some parallelism among the concepts, there is also some degree of hierarchy among them, as shown in Figure 5. In particular, the all-encompassing dependability and survivability concepts can be placed at the top level and all other concepts and their corresponding qualitative and quantitative attributes can be considered as other design requirements that assist in the overall design. [9]

Traditionally there are two different communities separately working on the issues of dependability and security. One is the community of dependability that is more concerned with non malicious faults, to name one of just a few. The other is the security community that is more concerned with malicious attacks or faults.

Figure 6: Reliability vs. Security Perspective of Failure [9]

The analysis of a design concerning dependability usually focuses on *the accidental or random faults*. Modeling these attributes can be performed easily; hence, the ability of optimizing the design can be achieved. There is another failure cause that should be addressed, namely, *the malicious or intentional threats*. These threats are mainly associated with the security concerns. There is no accurate statistical modeling technique available for these types of threats due to the fact that malicious attacks do not usually follow predictable patterns. Because the root causes of system failure in reliability or generally in dependability context (*e.g.*, random accidental failures) are fundamentally different from the root causes of security violations (*e.g.*, intentional attacks), then it is difficult to accurately represent security events using classical stochastic models. Figure 6 shows the pathology of these two different perspectives.

Privacy will be considered as a part of security and will be described in the next version of the document.

5.1.1 Definitions

First of all the terms are defined. [9]

- Availability: Readiness for correct service. The correct service is defined as what is delivered when the service implements a system function.
- Accessibility*:* Ability to limit, control, and determine the level of access that entities have to a system and how much information they can receive.
- Accountability: The ability to track or audit what an individual or entity is doing on a system.
- Authenticity*:* The property of being able to verify the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system.
- Confidentiality*:* Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.
- Dependability: The ability to deliver required services during its life cycle that can justifiably be trusted.
- Fault Avoidance (Prevention): A technique used in an attempt to prevent the occurrence of faults.
- Fault Containment (Isolation): The process of isolating a fault and preventing its effect from propagating.
- Fault Detection: The process of recognizing that a fault has occurred.
- Fault Forecasting (Prediction): The means used to estimate the present number, the future incidence, and the likely consequence of faults.
- Fault Location: The process of determining where a fault has occurred so a recovery can be used.

- Fault Masking: The process of preventing faults in a system from introducing errors into the informational structure of that system.
- Fault Removal: The means used to reduce the number and severity of faults.
- Fault Restoration (Recovery): The process of remaining operation or gaining operational status via reconfiguration event in the presence of faults.
- Fault Tolerance: Ability to continue the performance of its tasks in the presence of faults.
- Graceful Degradation: The ability of a system to automatically decrease its level of performance to compensate for hardware or software faults.
- Integrity: Guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity.
- Maintainability: The ease with which a system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment.
- Non-Repudiation (Non-Repudiability): Assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the information.
- Performability: The degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage. It is also defined as a measure of the likelihood that some subset of the functions is performed correctly.
- Reliability: A conditional probability that a system performs its intended tasks correctly throughout a complete interval of time.
- Safety: The property that a system does not fail in a manner that causes catastrophic damage during a specified period of time.
- Security: Ability to guard and protect from unwanted happenings or actions and preserve confidentiality, integrity, and availability
- Survivability: Ability to fulfil its mission in a timely manner in the presence of attacks, failures, or accidents.
- Testability: The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met.
- Traceability: The ability to verify the history, location, or application of an item by means of documented recorded identification

In the following two sections are introduced two approaches which are mentioned in pSHIELD [8], [11].

In the next version of the document the two approaches will be integrated.

5.2 Concept Taxonomy Approaches

5.2.1 Approach 1

5.2.1.1 Dependability Concept Taxonomy

There are a lot of definitions for dependability. Here are some of them:

- Dependability is the ability of a system to deliver the required specific services that can "justifiably be trusted" [1].
- Dependability is the ability of a system to avoid failures that are more frequent or more severe than is acceptable to the users [12].
- Dependability is a system property that prevents a system from failing in an unexpected or catastrophic way. [13]

Dependability measures the degree to which a system is operable at any random time during a specific mission profile, given that its services are available at the start of the mission.

Figure 7 shows a generalized view of dependability attributes along with its threats and the means to achieve dependability.

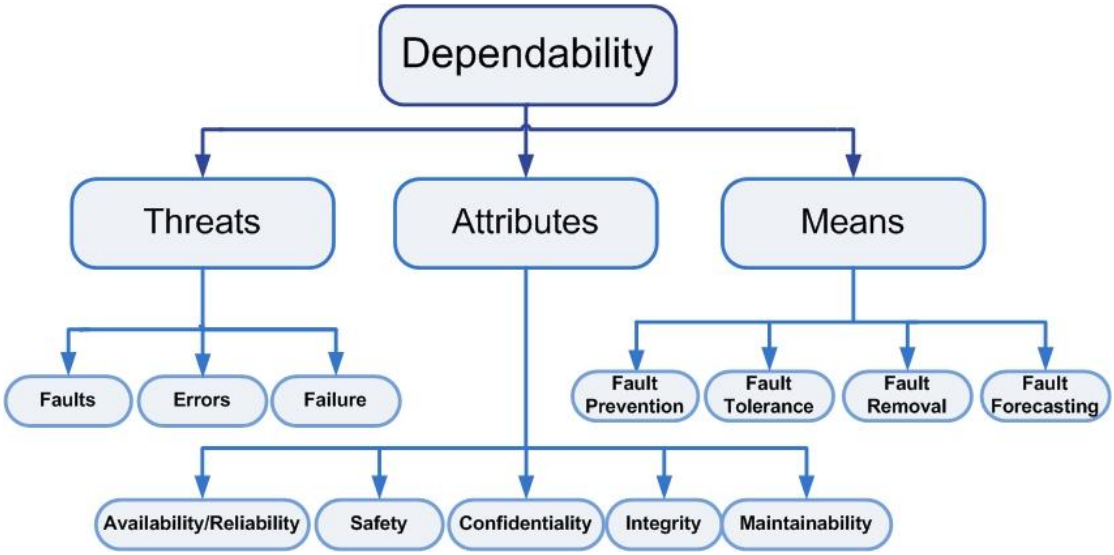


Figure 7: Dependability Concept Taxonomy.

The notions introduced here can be grouped into three classes:

- **Threats** to dependability: faults, errors, failures; they are undesired N but not in principle unexpected N circumstances causing or resulting from un-dependability;
- **Means** for dependability: fault prevention, fault tolerance, fault removal, fault forecasting; these are the methods and techniques enabling one
  a) to provide the ability to deliver a service on which reliance can be placed, and
  b) to reach confidence in this ability;
- **Attributes** of dependability: availability, reliability, safety, confidentiality, integrity, maintainability;
  a) enable the properties which are expected from the system to be expressed, and
  b) allow the system quality resulting from the impairments and the means opposing to them to be assessed.

Attributes

Depending on the application(s) intended for the system, different emphasis may be put on different facets of dependability, i.e., dependability may be viewed according to different, but complementary, *properties,* which enable the *attributes* of dependability to be defined:

- the *readiness for usage* leads to **availability**,
- the *continuity of service* leads to **reliability**,
- the *non-occurrence of catastrophic consequences on the environment* leads to **safety**,
- the *non-occurrence of unauthorised disclosure of information* leads to **confidentiality**,
- the *non-occurrence of improper alterations of the system* leads to **integrity**,
- the *ability to undergo repairs and evolution* leads to **maintainability**.

Dependability is a collection of related measures including some attributes such as reliability, availability, and safety [9] and is not a single property measure. Different authors describe dependability of a system as a set of properties or attributes. For instance, dependability concept includes some attributes such as reliability, maintainability, safety, availability, confidentiality, and integrity where the last three are shared

with the security concept. Some of these attributes are quantitative (e.g., reliability and availability) and some of them are qualitative (e.g., safety).

Associating integrity and availability with respect to authorised actions, together with confidentiality, leads to **security**.

Threats

The importance of assets SPD attributes is usually expressed in terms of the consequential damage resulting from the manifestation of threats.

Threats are expressed in the following taxonomy:

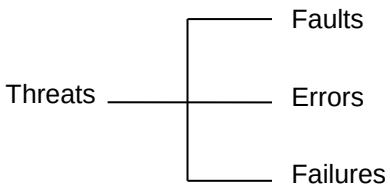


Figure 8: Threats taxonomy

where a fault is defined as a cause of an error and a failure is linked to the error that is outside of the error tolerance boundary and is caused by a fault.

A system **failure** occurs when the delivered service deviates from fulfilling the system **function**, the latter being what the system *is aimed at*. An **error** is that part of the system state which is *liable to lead to subsequent failure*: an error affecting the service is an indication that a failure occurs or has occurred. The *adjudged or hypothesised cause* of an error is a **fault**.

The creation and manifestation mechanisms of faults, errors, and failures may be summarised as follows:

- A **fault** is **active** when it produces an error. An active fault is either
  a) an internal fault which was previously **dormant** and which has been activated by the computation process, or
  b) an external fault. Most internal faults cycle between their dormant and active states. Physical faults can directly affect the hardware components only, whereas human-made faults may affect any component.
- An **error** may be latent or detected. An error is **latent** when it has not been recognised as such; an error is **detected** by a detection algorithm or mechanism. An error may disappear before being detected. An error may, and in general does, propagate; by propagating, an error creates other Ñ new Ñ error(s). During operation, the presence of active faults is determined only by the detection of errors.
- A **failure** occurs when an error "passes through" the system-user interface and affects the service delivered by the system. A component failure results in a fault:
  - for the system which contains the component, and
  - as viewed by the other component(s) with which it interacts; the failure modes of the failed component then become fault types for the components interacting with it.

These mechanisms enable the "fundamental chain" to be completed:

... -> failure -> fault -> error -> failure -> fault -> ...

Fault classification

We have provided a set of elementary fault classes with minimum overlapping. An intuitive choice is to start two classes namely, human-made faults (HMF) and nonhuman-made faults (NHMF).

Human-Made Faults (HMF)

Human-made faults result from human actions. They include absence of actions when actions should be performed (i.e., omission faults). Performing wrong actions leads to commission faults.

HMFs are categorized into two basic classes: faults with unauthorized access (FUA), and other faults (NFUA).

Faults with Unauthorized Access (FUA)

The class of Faults with unauthorized access (FUA) attempts to cover traditional security issues caused by malicious attempt faults. We have investigated FUA from the perspective of availability, reliability, integrity, confidentiality and safety.

Malicious attempt fault has the objective of damaging a system. A fault is produced when this attempt is combined with other system faults.

FUA and confidentiality. Confidentiality refers to the property that information or data are not available to unauthorized persons or processes, or that unauthorized access to a system's output will be blocked by the system's filter. Confidentiality faults are mainly caused by access control problems originating in cryptographic faults, security policy faults, hardware faults, and software faults. Cryptographic faults can originate from encryption algorithm faults, decryption algorithm faults, and key distribution methods. Security policy faults are normally management problems and can appear in different forms (e.g., as contradicting security policy statements).

FUA and integrity. Integrity refers to the absence of improper alteration of information. An integrity problem can arise if, for instance, internal data are tampered with and the produced output relies on the correctness of the data.

FUA and availability & reliability. In general Availability refers to a system's readiness to provide correct service and reliability refers to continuity of correct service, but according to interpretation proposed before these attributes have been considered as one because both guarantee the correct service with an error $e(t) < \varepsilon$. A typical cause of such faults is some sort of denial of service (DoS) attack that can, for example, use some type of flooding (SYN, ICMP, UDP) to prevent a system from producing correct output. The perpetrator in this case has gained access to a system, albeit a very limited one, and this access is sufficient to introduce a fault.

FUA and safety. Safety refers to absence of catastrophic consequence on System users end environment. A safety problem can arise if, for instance, an unauthorized system access can cause the possibility of human lives being endangered.

Not Faults with Unauthorized Access (NFUA)

There are human-made faults that do not belong to FUA. Most of such faults are introduced by error, such as configuration problems, incompetence issues, accidents, and so on.

Nonhuman-Made Faults (NHMF)

NHMF refers to faults caused by natural phenomena without human participation. These are physical faults caused by a system's internal natural processes (e.g., physical deterioration of cables or circuitry), or by external natural processes. They can also be caused by natural phenomena. For example, in

communication systems, a radio transmission message can be destroyed by an outer space radiation burst, which results in system faults, but has nothing to do with system hardware or software faults.

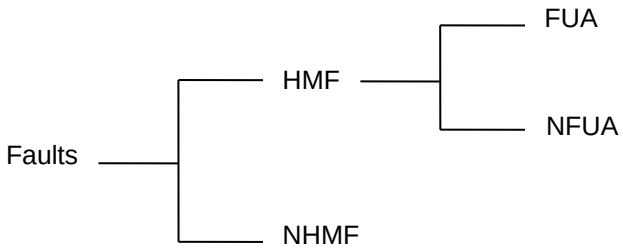


Figure 9: Faults taxonomy

Tree representation of faults

From above discussions, we propose the following elementary fault classes.

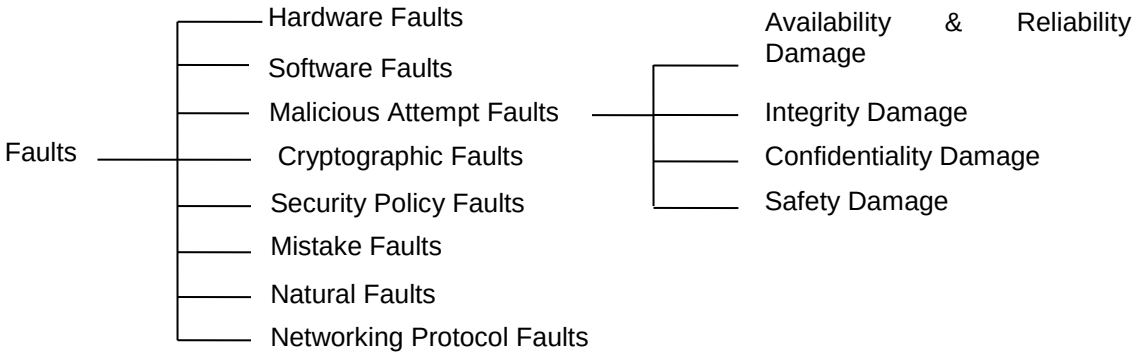


Figure 10: Fault classes

Errors

An error is the part of a system's total state that may lead to a failure—a failure occurs when the error causes the delivered service to deviate from correct service. The cause of the error has been called a fault.

An error is detected if its presence is indicated by an error message or error signal. Errors that are present but not detected are latent errors.

Since a system consists of a set of interacting components, the total state is the set of its component states. The definition implies that a fault originally causes an error within the state of one (or more) components, but service failure will not occur as long as the external state of that component is not part of the external state of the system.

Whenever the error becomes a part of the external state of the component, a service failure of that component occurs, but the error remains internal to the entire system.

Whether or not an error will actually lead to a service failure depends on two factors:

1. The structure of the system, and especially the nature of any redundancy that exists in it:
   - protective redundancy, introduced to provide fault tolerance, that is explicitly intended to prevent an error from leading to service failure.

- unintentional redundancy (it is in practice difficult if not impossible to build a system without any form of redundancy) that may have the same—presumably unexpected—result as intentional redundancy.

2. The behavior of the system: the part of the state that contains an error may never be needed for service, or an error may be eliminated (e.g., when overwritten) before it leads to a failure.

Failures

Failure occurrence has been defined with respect to the function of a system, not with respect to its specification. Indeed, if an unacceptable behaviour is generally identified as a failure due to a deviation from the compliance with the specification, it may happen that such a behaviour complies with the specification, and be however unacceptable for the system user(s), thus uncovering a specification fault. In the latter, recognising that the event is undesired (and is in fact a failure) can only be performed after its occurrence, for instance via its consequences.

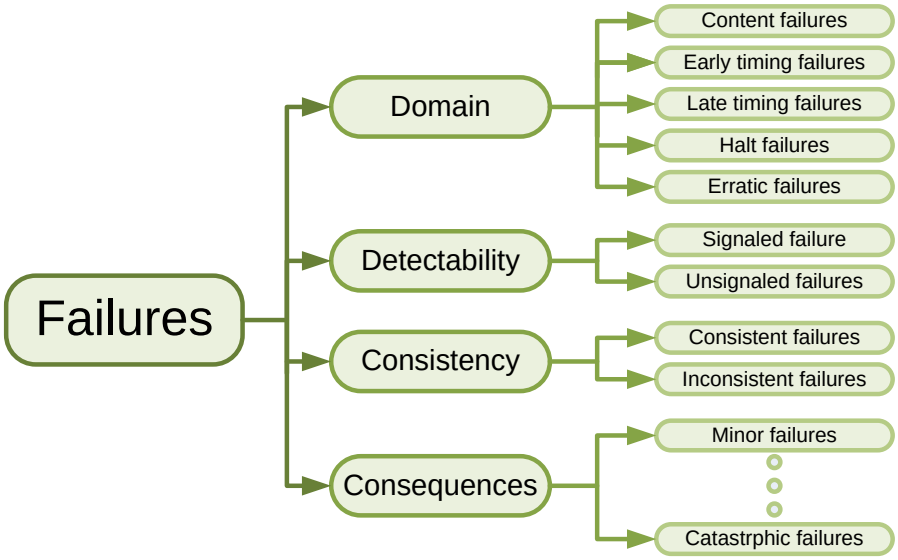


Figure 11: Failures

Means

Dependability of a system can be achieved by the joint and balanced deployment and operation of a set of four techniques:

- **Fault prevention**: how to prevent fault occurrence or introduction,
- **Fault tolerance**: how to ensure a service up to fulfilling the system's function in the presence of faults,
- **Fault removal**: how to reduce the presence (number, seriousness) of faults,
- **Fault forecasting (or prediction)**: how to estimate the present number, the future incidence, and the consequences of faults.

Fault Prevention

Prevention techniques aim at eliminating or reducing the likelihood of faults to arise.

Fault prevention is part of general engineering and there are facets of fault prevention that are of direct interest regarding dependability and security.

Prevention of development faults is an obvious aim for development methodologies, both for software (e.g., information hiding, modularization, use of strongly-typed programming languages) and hardware (e.g., design rules). Improvement of development processes in order to reduce the number of faults introduced in the produced systems is a step further in that it is based on the recording of faults in the products, and the elimination of the causes of the faults via process modifications.

Fault Tolerance

Fault tolerance, which is aimed at failure avoidance, is carried out via error detection and system recovery. Figure 12 gives the techniques involved in fault tolerance.

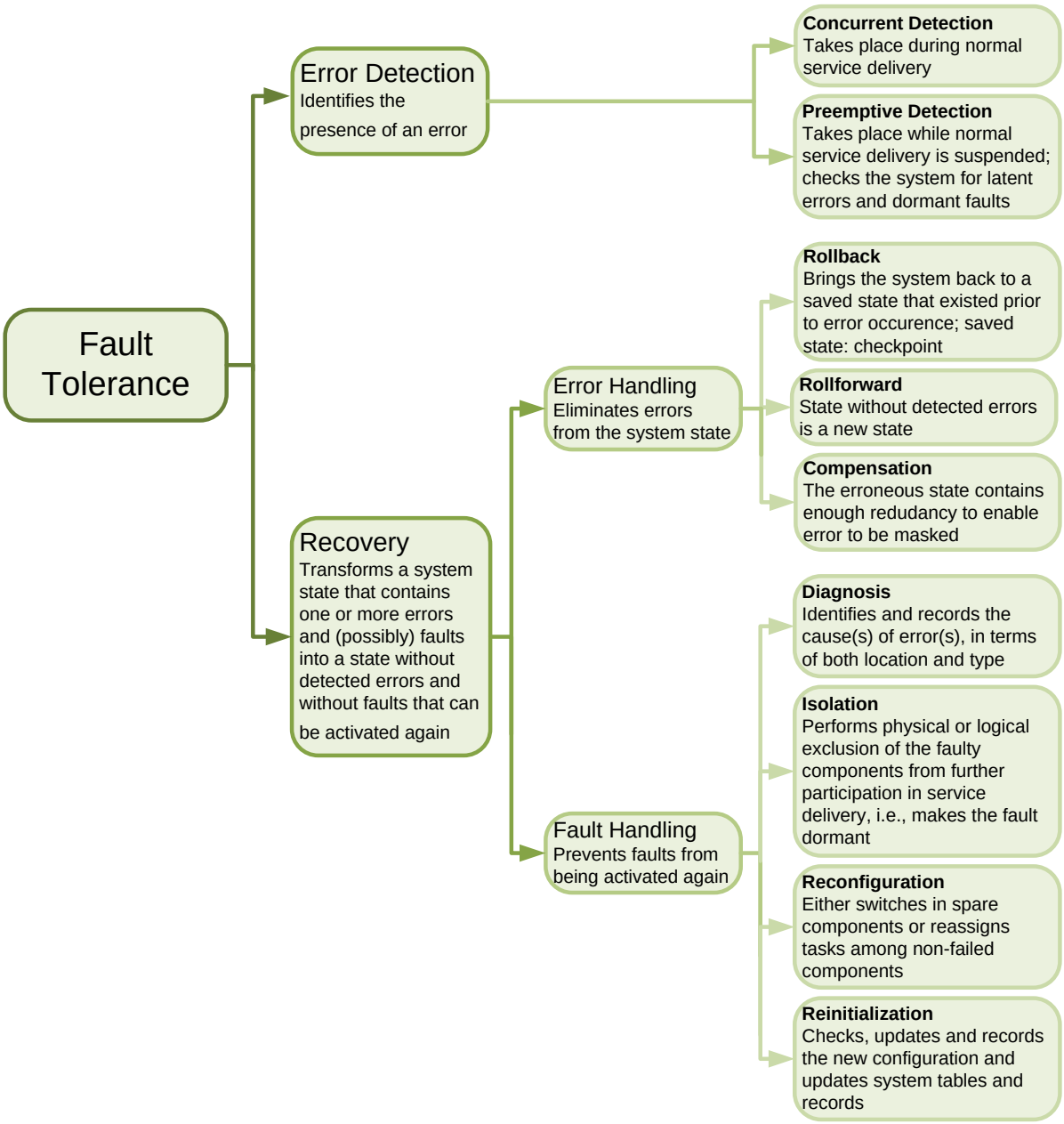


Figure 12: Fault tolerance techniques

Fault Removal

Next we consider fault removal during system development, and during system use.

Fault Removal during Development

Fault removal during the development phase of a system lifecycle consists of three steps: verification, diagnosis, and correction. We focus in what follows on verification, that is the process of checking whether the system adheres to given properties, termed the verification conditions; if it does not, the other two steps have to be undertaken: diagnosing the fault(s) that prevented the verification conditions from being fulfilled, and then performing the necessary corrections. After correction, the verification process should be repeated in order to check that fault removal had no undesired consequences; the verification performed at this stage is usually termed nonregression verification.

Fault Removal during Use

Fault removal during the use of a system is corrective or preventive maintenance. Corrective maintenance aims to remove faults that have produced one or more errors and have been reported, while preventive maintenance is aimed at uncovering and removing faults before they might cause errors during normal operation. The latter faults include 1) physical faults that have occurred since the last preventive maintenance actions, and 2) development faults that have led to errors in other similar systems. Corrective maintenance for development faults is usually performed in stages: The fault may be first isolated (e.g., by a workaround or a patch) before the actual removal is completed. These forms of maintenance apply to nonfault-tolerant systems as well as to fault-tolerant systems, that can be maintainable online (without interrupting service delivery) or offline (during service outage).

Fault Forecasting

Fault forecasting is conducted by performing an evaluation of the system behavior with respect to fault occurrence or activation. Evaluation has two aspects:

- qualitative, or ordinal, evaluation, that aims to identify, classify, and rank the failure modes, or the event combinations (component failures or environmental conditions) that would lead to system failures;
- quantitative, or probabilistic, evaluation, that aims to evaluate in terms of probabilities the extent to which some of the attributes are satisfied; those attributes are then viewed as measures.

5.2.1.2 Security Concept Taxonomy

The security concept structure is shown in Figure 13 considering the context of security assurance and it merges all various attributes in one comparable framework.

The concept of security is closely related to the confidentiality, integrity, and availability of assets. According to Neumann [14] “security must encompass dependable protection against all relevant concerns, including confidentiality, integrity, and availability despite attempted compromises, preventing denials of service, preventing and detecting misuse, providing timely responses to threats, and reducing the consequences of unforeseen threats”.

The security concept encompasses protection of systems, networks, and their components from different misappropriate actions as well as protection of information, *e.g.*, protection of data and programs from inappropriate actions.

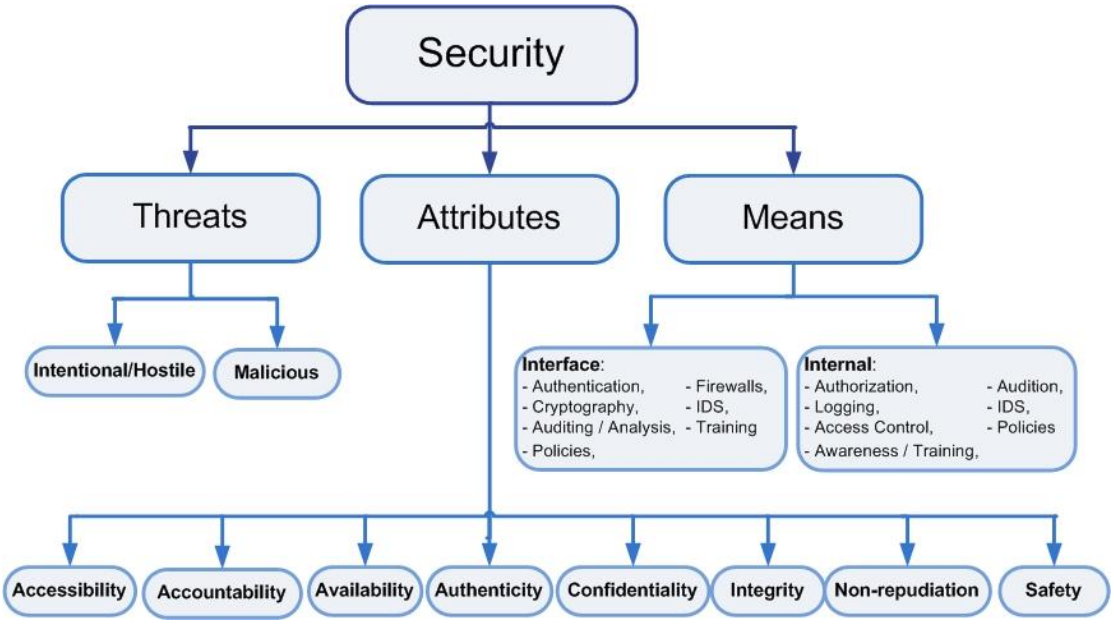


Figure 13: Security Concept Taxonomy.

Attributes

Different researchers assign different attributes to security. There is no universal agreement about the expressions used in the security literature in describing its attributes. For our context we generalize a security concept structure as shown in Figure 13 which brings together all various attributes in one comparable framework. When compared to other concepts, security does share some attribute with other concepts and at the same time it exclusively encloses other attributes.

Threats

Security entails prediction of possible threats, including insider abuses or misuses of the system, as well as outsider invasions or breaches. This concept can be expressed also as resilience of a system to any type of malicious attacks.

Means

The required goal of security is to introduce measures and procedures that preserve confidentiality, integrity, availability, and other attributes such as authenticity and non-repudiation. Control mechanisms implement functions that help harden the system in order to prevent, detect, tolerate, and respond to security attacks. This is done using both theoretical and practical approaches such as cryptography, access controls, authentication, firewalls, risk assessments, policies, auditing, and intrusion detection and prevention systems as well as raising the human awareness and training.

5.2.2 Approach 2

5.2.2.1 Dependability and Security attributes definition

To address Security, Privacy and Dependability in the context of Embedded Systems (ESs) it is essential to define the assets that these kinds of systems aim to protect.

The nSHIELD project assets can be categorised in two principal groups, logical and physical asset. Inside these categories it is possible to define information, services and software as logical assets and human beings, hardware or particular physical objects as physical assets.

These assets are characterized by their Dependability and Security attributes.

The original definition of dependability refers to the ability to deliver a service that can be justifiably trusted. The alternative definition is the ability to avoid service failures that are more frequent and severe than is acceptable. The concept of trust can be defined as accepted dependence, and dependability encompasses the following attributes:

- Availability: Readiness for correct service. The correct service is defined as what is delivered when the service implements a system function.
- Reliability: Continuity of correct service.
- Safety: Absence of catastrophic consequences on the users and environment.
- Integrity: Absence of improper system alterations.
- Maintainability: Ability to undergo modifications and repairs.

Security has not been introduced as a single attribute of dependability. This is in agreement with the usual definitions of security, which is viewed as a composite notion of the three following attributes:

- Confidentiality: the prevention of the unauthorized disclosure of information;
- Integrity: the prevention of the unauthorized amendment or deletion of information;
- Availability: the prevention of the unauthorized withholding of information.

Avizienis et al. [2] merged the attributes of dependability and security together, as shown in Figure 14. Similarly, the above attributes can be reframed as follows:

- Availability: Readiness for correct service. The correct service is defined as delivered system behavior that is within the error tolerance boundary.
- Reliability: Continuity of correct service. This is the same as the conventional definition.
- Safety: Absence of catastrophic consequences on the users and the environment. This is the same as the conventional definition.
- Integrity: Absence of malicious external disturbance that makes a system output off its desired service.
- Maintainability: Ability to undergo modifications and repairs. This is the same as the conventional definition.
- Confidentiality: Property that data or information are not made available to unauthorized persons or processes. In the proposed framework, it refers to the property that unauthorized persons or processes will not get system output or be blocked by the filter.

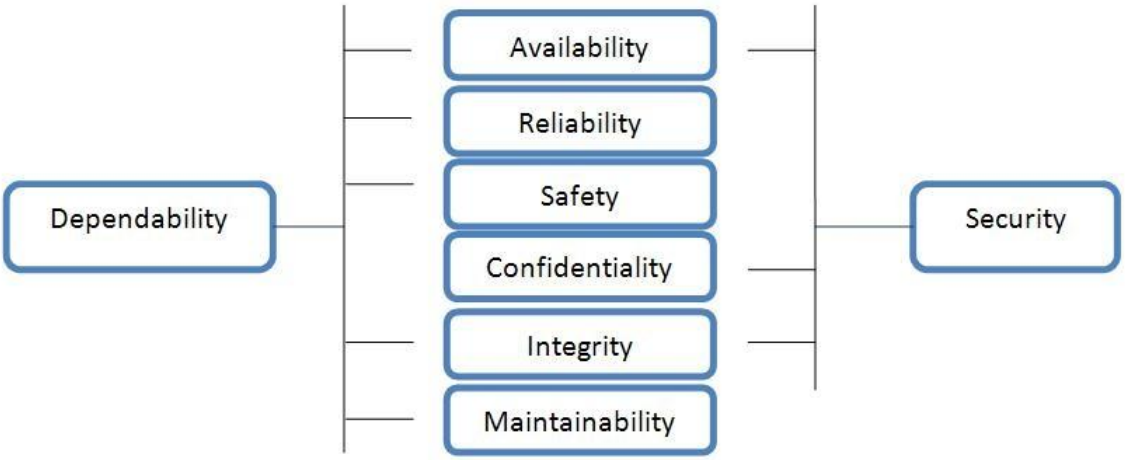


Figure 14: Dependability and Security attributes

6 nSHIELD Scenarios

6.1 Railroad Security Scenario

Rail-based mass transit systems are vulnerable to many criminal acts, ranging from vandalism to terrorism. Therefore, physical security systems for infrastructure protection comprises all railway assets as for tunnel, train on board, platform and public areas, external Areas, technical control room, depots, electrical substations and etc…

The objectives are to forecast critical threats as: aggressions and abnormal behaviours, sabotage and terrorism, vandalism and Graffitism, thefts and pickpocketing.

A modern smart-surveillance system suitable for the protection of urban or regional railways is made up by the following subsystems:

1. Intrusion detection and access control:
    - volumetric sensors for motion detection;
    - magnetic contacts to detect illicit doors opening;
    - glass break detectors;
    - microphonic cables for fence/grill vibration detection;
    - active infrared barriers for detecting intrusions inside the tunnels;

2. Intelligent video-surveillance and Intelligent sound detection:
    - advanced cameras with special features;
    - digital video processing and recording, using efficient data compression protocols;
    - video-analytics of the scenes, using computer vision algorithms;
    - Microphones

3. Dedicated communication network

4. Integrated management system

Distributed smart-sensors are installed along the railway line both in fixed (e.g. bridges, tunnels, stations, etc.) and mobile (passenger trains, freight cars, etc.) locations (Figure 15).

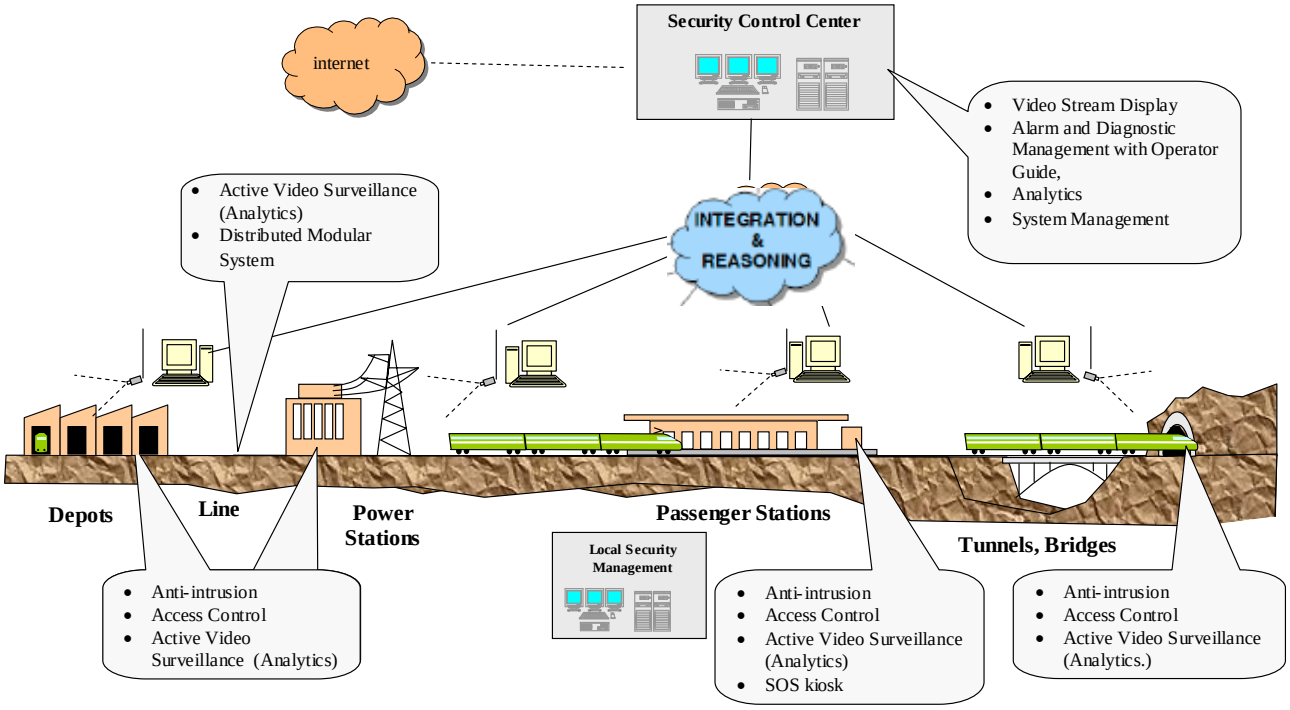


Figure 15: Architecture

They are integrated locally using local wireless infrastructures (e.g. Wi-Fi, ZigBee, etc.) and then data is collected by WSN gateway nodes and transmitted remotely by means of WAN (Wide Area Network). Low/average bandwidth networks are strictly required to transmit alarms to the control centre, which are often already available (like GSM-R for railways) or easy to deploy (like satellite) and provide an extensive coverage of the infrastructure. However, if high-quality video streams from cameras need to be shown to the operators in order to verify the alarm and/or supervise the situation; higher bandwidth is required which can be possible achieved by multiple low bandwidth connections.

This system are already been designed by Ansaldo STS for metro railways, where heterogeneous intrusion detection, access control, intelligent video-surveillance and abnormal sound detection devices are integrated in a cohesive Security Management System (SMS), Figure 16.

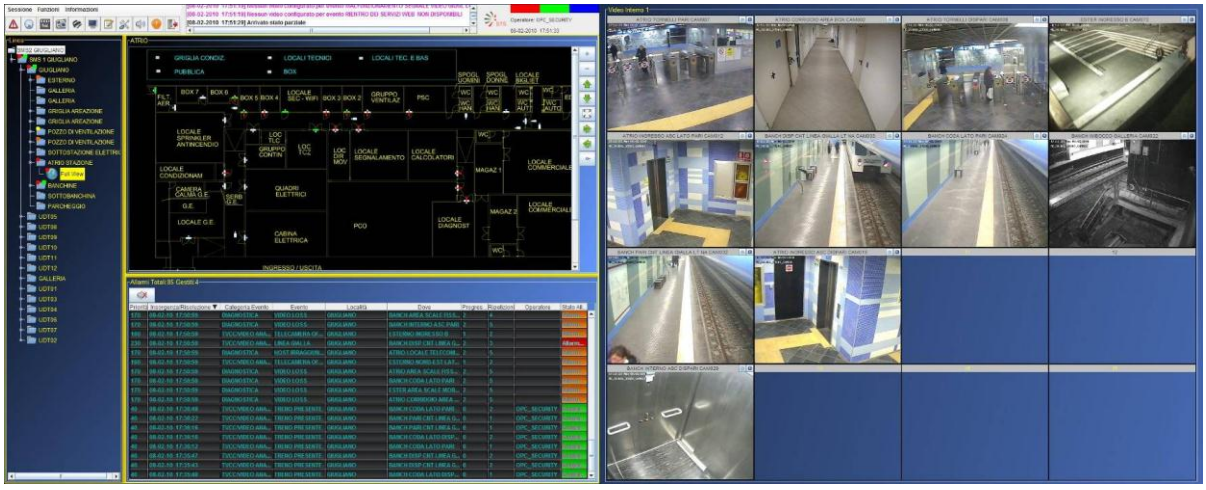


Figure 16: SMS-Security Management System

The core of the SMS consists of a web-based software application featuring a graphical user interface. System architecture is distributed and hierarchical, with both local and central control rooms collecting alarms according to different scopes and responsibilities. In case of emergencies, the procedural actions required to the operators involved are orchestrated by the SMS. Redundancy both in sensor dislocation and hardware apparels (e.g. by local or geographical clustering) improve detection reliability, through alarm correlation, and overall system resiliency against both random and malicious threats. Video-analytics is essential, since a small number of operators would be unable to visually control the large number of cameras which are needed to extensively cover all the areas needing to be protected. Therefore, the visualization of video streams is activated automatically when an alarm is generated by smart-cameras or other sensors, following an event-driven approach. Very high resolution cameras installed close to the turnstiles are used to automatically detect and store the faces of passengers, whose database can be accessed for post-event investigations. Real-time communication between the on-board and the ground is allowed by a wide-band wireless network.

Currently, the security system described above is highly heterogeneous in terms not only of detection technologies (which will remain such) but also of embedded computing power and communication facilities. In other words, sensors differ in their inner hardware-software architecture and thus in the capacity of providing information security and dependability. This causes several problems:

- Information security must be provided according to different mechanisms and on some links - which are not “open” but still vulnerable to attacks - information is not protected by cryptographic nor vitality-checking protocols;
- Whenever any new sensor needs to be integrated into the system, a new protocol and/or driver must be developed and there is no possibility of directly evaluating the impact of such integration on the overall system dependability;
- New dedicated and completely segregated network links often need to be employed in order not to make the sensor network exposed to information related threats;
- The holistic assurance and evaluation of dependability parameters (e.g. for assessment/certification purposes) would be a very difficult task.

In particular both natural and malicious faults can impact on system availability and indirectly on safety, since the SMS is adopted in critical infrastructure surveillance applications.

The problems mentioned above can be solved by adopting the nSHIELD architecture. Cohesion will be assured by wrapping sensors of any nature with homogeneous embedded hardware and software providing information security, by e.g.:

- Cryptographic protocols
- Vitality checking (heartbeat/watchdog timers based on sequence numbers and time-stamping)

The mechanisms provided by nSHIELD would mitigate the effects on the system of the following logical threats:

- Repetition (a message is received more than once)
- Deletion (a message is removed from a message stream)
- Insertion (a new message is implanted in the message stream)
- Re-sequencing (messages are received in an unexpected sequence)
- Corruption (the information contained in a message is changed, casually or not)
- Delay (messages are received at a time later than intended)
- Masquerade (a non-authentic message is designed thus to appear to be authentic)

Some sensing devices will be converted into smart-sensors by integrating the sensor unit with the nSHIELD processing units (both hardware and software) at the node level. The sensor networks will be integrated by the nSHIELD middleware before data is collected by the SMS and used at the presentation level (integration and reasoning).

Description of Attributes, Threats, and Means for Railroad Security Scenario	
Attributes	
Threats	
Mean	

6.2 Voice/FacialVerification Scenario

In the last ten years SPD application scenarios are increasingly introducing the detection and tracking of devices, cars, goods, cars, etc. One of the most important objective of this trend is to increase the intrinsic security, privacy and dependability of the scenario and have more and more services to improve our life (automatic tolling payment, navigation, traceability, logistics…e.g.).Very frequently, these services and functionalities are based on the identification of a device while we are using it, and today there are many solutions to connect and exchange data from machine to machine (RFID, Wi-Fi, 3G connection and others wired and wireless connections through authentication by code or digital signature…).

In the next future, a similar application scenario is trying to introduce similar services performing the recognition, monitoring and traceability of people.

This scenario is oriented to develop new techniques to analyse physical quantities such as the face image and the voice sound that will be used as a “real-time” person profile that, compared with the one stored in an archive, allows the recognition, monitoring and tracking of that person. From a technical point of view, the requirements of this application scenario introduce new challenges derived from the use of embedded systems to provide recognition, monitoring and tracking services. nSHIELD project, with its SPD hardware infrastructure and software layers, represents the correct answer to these important challenges.

Face Recognition

Face images, which are commonly known as displayed portraits, have been used for many decades for manually verifying the identity of individuals. The recognition procedure required a human operator, being completely manual or semi-automatic, and was based on standard computing systems. More recently, digital face images have been used in a variety of applications ranging from analysis of human actions to computer-assisted face recognition and, the design of new algorithms in conjunction with the evolution of hardware capabilities, allowed the possibility to adopt embedded systems to provide the recognition service. Although photographic formats have been roughly standardized,(consider i.ie. application context like the issuing of passports and driver licenses) there is an ever-increasing need of defining a standard data format for digital face images that would allow interoperability among vendors, service providers and device manufacturers.

The International Standard ISO/IEC 19794-5 (ICAO) is aimed at providing a face image format for recognition applications, which enables the exchange of face image data. Typical applications are:

- human examination of facial images with sufficient resolution to allow a human examiner to ascertain small features such as moles and scars that might be used for verifying the identity;
- human verification of identity through a comparison of facial images;
- computer-automated face identification (one-to-many search);
- computer-automated face verification (one-to-one match).

In order to improve face recognition accuracy, the standard ISO/IEC 19794-5 does not specify only a data format, but it also provides:

- scene constraints (pose, expression etc.);
- photographic properties (lighting, positioning, camera focus etc.);
- as well as digital image attributes (image resolution, image size etc.).

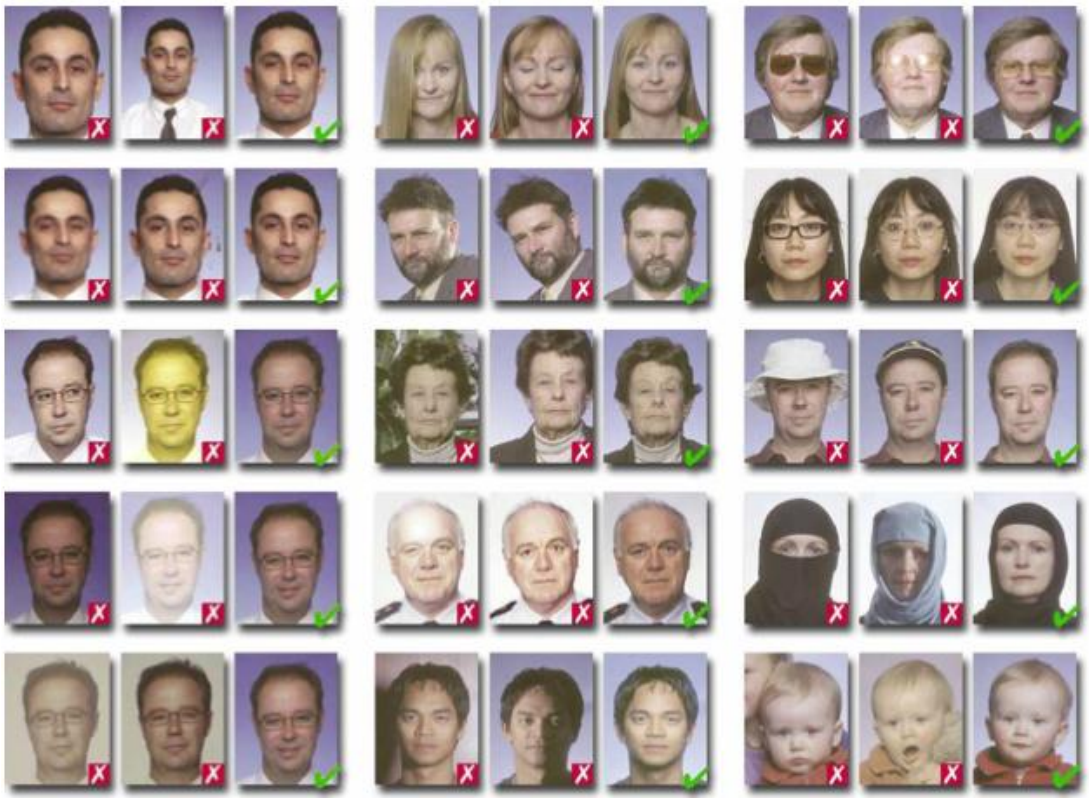


Figure 17: Examples of face recognition results.

This application scenario aims to evaluate the facial image according to the ICAO ISO/IEC 19794-5, which defines the requirements for image geometry and scenery of facial images, and returns Token Face Images and Full-Frontal Face Images that are compliant with the standard. Real-time feedback capabilities must allow capturing and automated enrolment in a faster and more efficient way.

The system will accurately find face and facial features (eyes, mouth, eyebrows etc.) in images with 8-bit grayscale or 24-bit RGB and automatically will extract the following information:

- number of faces,
- origin at upper left,
- centered image,
- pixel aspect ratio,
- resolution,
- width of head,
- length of head,
- head pose,
- position of eyes.

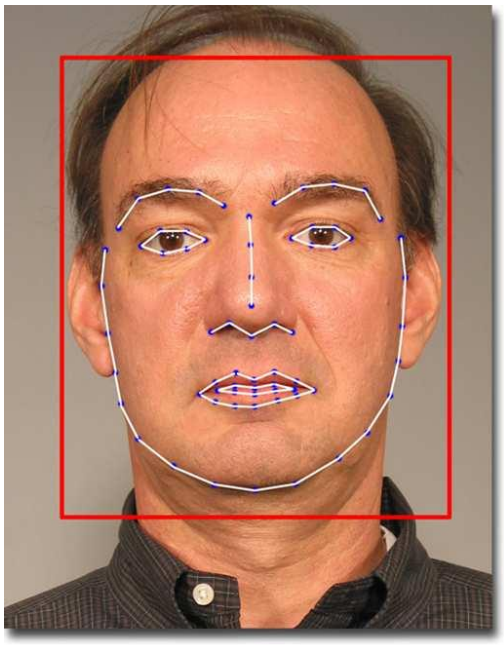


Figure 18: Part of the face recognized.

The system, according to the ICAO ISO/IEC 19794-5 standard, must test the facial area for automatically check the following quality requirements:

- Gray Scale Density,
- Color Saturation,
- Unnatural Color,
- Color Space,
- Video Interlacing,
- Radial Distortion of the Camera Lens,
- Shadows Over the Face,
- No Over or Under Exposure,
- Focus and Depth of Field,
- Eye Glasses,
- Red Eye,
- Eye Patches,
- Shadows in Eye-Sockets,
- Mouth Expression,
- Hot Spots,
- Concealment by hat,
- Background shadows,
- Uniformity of background.

Finally, the system must support (as far as both reading and writing functionalities are concerned) the CBEFF Patron Formats A-C, with ISO/IEC 19785-1 and ANSI INCITS 398-2005 interchange files, and its outputs must be fully compliant with the ISO/IEC 19794-5 standard.

Figure 19: Example of a face recognition software.

Voice Verification

Our voices are unique for each person (including twins), and cannot be exactly replicated. Speech includes two components: a physiological component (the voice tract) and a behavioural component (the accent). It is almost impossible to imitate anyone's voice perfectly. Voice recognition systems can discriminate between two very similar voices, including twins.

The voiceprint generated upon enrolment is characterised by the vocal tract, which is a unique a physiological trait. A cold does not affect the vocal tract, so there will be no adverse effect on accuracy levels. Only extreme vocal conditions such as laryngitis will prevent the user from using the system.

During enrolment, the user is prompted to repeat a short passphrase or a sequence of numbers. Voice recognition project aims to test various audio capture devices (microphones, telephones and PC microphones). The performance of voice recognition systems may vary depending on the quality of the audio signal.

To prevent the risk of unauthorised access via tape recordings, the user must ask to repeat random phrases. This precaution increases the SPD level of the system.

Another goal of the application scenario is to improve the most important weaknesses of voice biometric systems: the high false non-matching rates. Voice verification can be used for government, healthcare, house arrest and probation-related authentication.

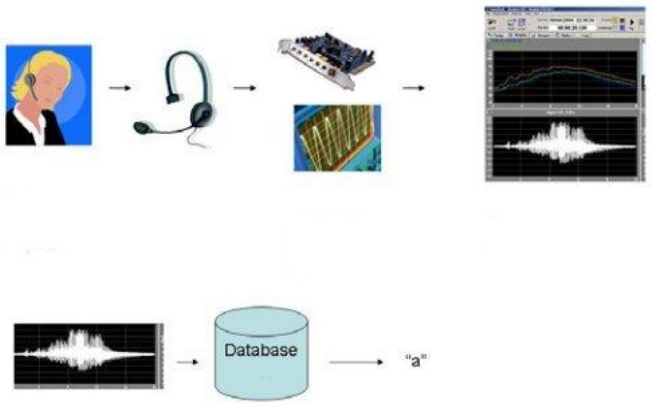


Figure 20: The voice recognition process.

Voice biometrics works by digitizing a profile of a person's speech to produce a stored model voice print, known as template. Biometric technology reduces each spoken word to segments composed of several dominant frequencies called formants. Each segment has several tones that can be captured in a digital format. The tones collectively identify the speaker's unique voice print. Voice prints are stored in databases in a manner similar to the storing of fingerprints or other biometric data.

To ensure a good-quality voice sample, a person must recite some sort of text or pass phrase, which can be either a verbal phrase or a series of numbers. The phrase may be repeated several times before the sample is analysed and accepted as a template in the database. When a person speaks the assigned pass phrase, certain words are extracted and compared with the stored template for that individual. When a user attempts to gain access to the system, his or her pass phrase is compared with the previously stored voice model. Some voice recognition systems do not rely on a fixed set of enrolled pass phrases to verify a person's identity. Instead, these systems are trained to recognize similarities between the voice patterns of individuals when the persons speak unfamiliar phrases and the stored templates.

Voice verification technology uses the different characteristics of a person's voice to discriminate between speakers. These characteristics are based on both physiological and behavioural components. The physical shape of the vocal tract is the primary physiological component. The vocal tract is made up the oral and nasal air passages that work with the movement of the mouth, jaw, tongue, pharynx and larynx to articulate and control speech production. "The physical characteristics of these airways impart measurable acoustic patters on the speech that is produced". The behavioural component is made up of movement, manner, and pronunciation.

The combination of the unique physiology and behavioural aspects of speaking enable verification of the identity of the person who is speaking. This voice verification project will work by converting a spoken phrase from analog to digital format and extracting the distinctive vocal characteristics, such as pitch, cadence, and tone, to establish a speaker model or voiceprint. A template must be generated and stored for future comparisons.

Voice verification systems can be used to verify a person's claimed identity or to identify a particular person, increasing in this way the SPD level of the application context in which they are used. These functionalities are often used where voice is the only available biometric identifier, such as over the telephone. Voice verification systems may require minimal hardware investment, as most personal computers already contain a microphone, and are perfectly suitable for embedded systems. The downside to the technology is that, although advances have been made in recognizing the human voice, ambient temperature, stress, disease, medications, and other physical changes can negatively impact on automated recognition.

Voice verification systems are different from voice recognition systems although the two are often confused. Voice recognition is used to translate the spoken word into a specific response, while voice

verification verifies the vocal characteristics against those associated with the enrolled user. The goal of voice recognition systems is simply to understand the spoken word, not to establish the identity of the speaker. This application scenario, being focused on SPD aspects, considers only the voice verification.

Description of Attributes, Threats, and Means for Voice/Facial Verification Scenario	
Attributes	
Threats	
Means	

6.3 Dependable Avionic System Scenario

The driver of change in avionic system scenarios is the continuing rapid advance of electronics technology, computers, sensors, displays, data-buses etc. There are examples of avionics computers introduced less than a few years ago that are now available under half the size, weight, power consumption, but with considerably enhanced performance and functionality. Avionics components and communication standards are now being replaced increasingly by commercial ones:

- microprocessors, microcontrollers;
- data-buses (e.g. Ethernet, CANbus);
- flat panel AMLCD displays and interfaces (e.g. OpenGL graphics language).

These trends are consolidated in both the civil and military aviation markets. The rapid and continuing advance of electronic technology is constantly driving down the cost of hardware, as the number and cost of the components used falls and much more functionality is achieved with less and less hardware.

Associated with the enhanced capability afforded by the technology, the functionality of avionics systems has continued to rise

- Fly-By-Wire flight controls;
- Flight Management System (FMS),
- Full glass cockpits, large multi-function displays;
- Future Air Navigation System (FANS) capability to operate in the new air traffic management environment;
- Passenger entertainment systems and commercial/business services;
- On-Board central maintenance computers and electronic documentation.

Much effort is spend to ensure that application software can be reused on different hardware in order to avoid the high cost of new software for the application being hosted on different processors, for example, in the event that during the life-cycle of the product the processor becomes obsolete or has insufficient capability to support growth in required functionality.

Hardware independent application software requires embedded software Operating System (OS) or Executive which provides a generic interface to the application code and which translates it for the particular processor and hardware architecture being used.

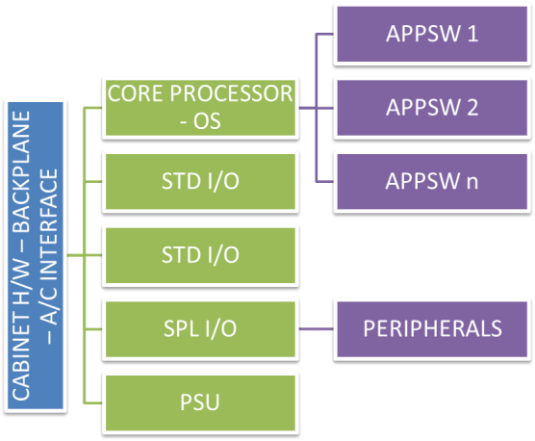
As technology has advanced so there has been a continuing trend of avionics integration.

In the past most integration has been concerned with processes or functions which were already interdependent and by so doing savings could be made in interface hardware, aircraft wiring, power

supplies, etc., with little risk to the certification of the system. This is usually referred to as “vertical” integration. The integration of inertial sensors with computers to produce inertial reference systems (IRS) is an example of vertical integration.

In other cases separate processors have been co-located into a single box because they can share the same I/O hardware on which they both depend, thereby eliminating one set of I/O hardware and simplifying the aircraft wiring.

More latterly there is a trend to go one step further by the integration of largely unrelated avionics functions onto common processors. This may be referred to as “horizontal” integration. There are significantly higher risks because, whilst additional hardware resource may be saved, there are added complexities to provide “equivalent independence” or partitioning within the processing platform. Partitioning is used to ensure that malfunctions within one application (function) cannot affect the others, or that modifications made to one may be certified without the need to revalidate the others. A further complication may be that additional levels of hardware redundancy and associated monitoring and configuration management facilities may be needed to offset the risk of failures within the shared processor causing simultaneous loss of all the otherwise unrelated application functions. The trade between savings of hardware (e.g. processor modules) on the one hand, and the escalation in cost to achieve acceptable levels of segregation and integrity on the other, needs to be carefully weighed. This is a trade that would appear to be more difficult to support as the hardware proportion of overall cost continues to fall with time.



A primary goal of IMA is to establish the application of a “standard” set of hardware modules, directly line-replaceable, encompassing as much of the total avionics suite as possible.

The main drivers for aircraft and systems architecture are:

- reduction in installation complexity – wires, connectors;
- robustness of peripheral interfaces to noise, improved RFI/EMC immunity;
- reduced uncertainties in identifying fault location, i.e. whether fault is in the peripheral, the computer or the wiring;
- simplification in data interfaces; by incorporating special peripheral processing at the “point of action” and reducing communication data flow to higher-order parameters;
- architectural flexibility. The I/O interface is available on aircraft-level data-buses for direct use elsewhere;
- enabling the interface to be independent of the technology of the peripheral;

As electronics are used at the peripherals and serial data-buses provide the interface, the systems become “digital” from end to end.

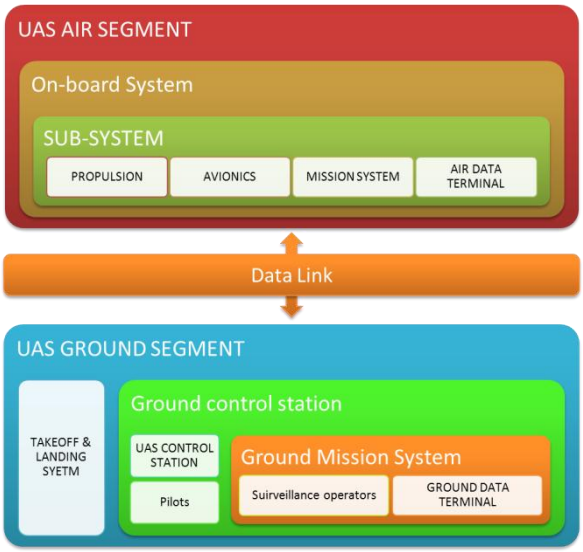
The greater use of smart peripherals and remote data concentrators will also significantly diminish the need for analogue and discrete I/O modules within IMA cabinets and computer LRUs.

The goal of standard, reusable and interchangeable modules is central to the concept of IMA. By rigorous definition and control of each module, both hardware and software, and of its interfaces.

The application scenario for the nSHIELD project is the Avionic System for a Unmanned Aircraft System.

To preserve the main functionalities represents an example application of great interest for the Avionic System. In particular, in this use case, the following requirements have to be fulfilled:

- Secure and dependable handling of on-board computing and sensor processing capabilities
- Secure and dependable Ground Operator sensor control terminals



The nSHIELD philosophy will be applied to preserve the data for the following components

- AIR/GROUND DATA TERMINAL: the nSHIELD solution will guarantee that the data exchanged between the UAV and Ground Station will be preserved by anomalous interface. (Security paradigm oriented)
- Avionics Unit: the nSHIELD solution will guarantee that the data acquired by sensors are protected against the possible corruption. (Dependability paradigm oriented)
- Mission System: the nSHIELD solution will define solution for easy integration of new sensors and/or replaced old version (Dependability paradigm oriented)
- Ground Control System: the nSHIELD solution will be able to manage the different access to the Unmanned System for the Mission Operators and Pilot (Security and Privacy paradigm oriented)

Description of Attributes, Threats, and Means for Dependable Avionic System Scenario	
Attributes	
Threats	
Means	

6.4 Social Mobility and Networking Scenario

The project nSHIELD is considering SMN as a scenario of humans that are moving from one place to other by walking, using public means of transport (train, bus, etc.) or personal one, such a bicycle, car, etc., and they desired to communicate with other persons or things (here we are referring to the future **I**nternet of **T**hings and **H**umans, named here ITH). There will be two different sub-scenarios, indoor and outdoor. Indoor sub-scenario is related to social mobility of people inside the houses, buildings, etc. Outdoor sub-scenarios is related to social mobility of people on the streets in cites/towns, villages, highways, and other roads. In the scope of SMN in the nSHIELD project we are aiming to present this as a common R&D area in which SPD play an extremely important role for a successful implementation of SMN scenario. Figure 21 illustrates how different filed will interact jointly in this scenario. There are four fundamental overlapping areas:

1. New Street and Building Lights (SBLs),
2. Public and Private Transportation Means PPTMs,
3. The future Internet of Things and Humans (ITHs), and
4. Ubiquitous and/ Pervasive Computing (U&PC)

that have a common one, i.e., SPD aspects that are considered in details in this project in which a common nSHIELD system architecture is composed of four layers: node, network, middleware and overlay, that play an important role in the implementation of SMN scenario. In SMN scenarios a focus will be given to intelligent systems and intelligent ICT, since intelligent street lighting is maturing and providing cost-effective approach to manage municipal street lighting. Even though several attempts that have tried to merge the two worlds could not reach the masses, experts expect that future mobile social networking systems possibly even exceed the success of their Internet bound counterparts. We believe that two key features are the user‘s permanent reachability and location awareness, which is called P3 (Peer-to-Peer-to-Place).

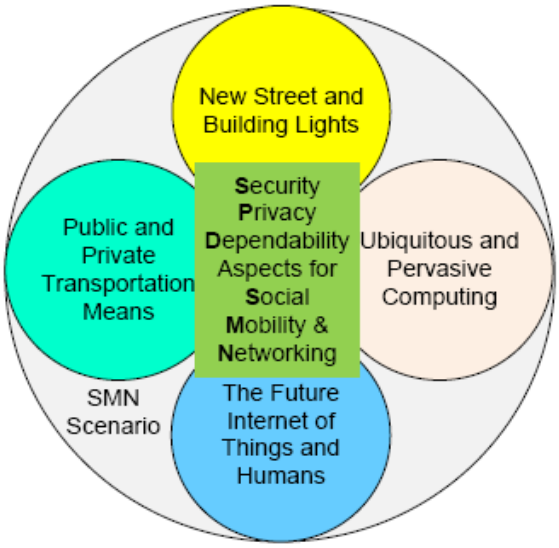


Figure 21: Social Mobility and Networking Scenario.

For that we need to integrate the old and new communication infrastructures. This lead to the creation of large and complex networks called *real-life networks* that include: electrical power grid, World Wide Web, the Internet backbone, collaboration and citation networks, and airline connection networks. Step-by-step, we are moving into a world of ubiquitous and pervasive computing (U&PC), Security, Trust, Privacy and Dependability (STPD) issues will be in focus more than ever before. Sensor networks have been used in numerous applications such as remote sensing, environmental monitoring, habitat, human activity, health monitoring, industrial appliances monitoring, medical applications, space and underwater phenomena monitoring, home and building automation and so on. Capturing sensory data from Body Area Networks (BANs), or Body Sensor Networks (BSNs) and sending it to social networks is challenging task, because it required a number of distributed networks to work together seamlessly. Disseminating the sensory data in real time to one‘s community of interest such as family, friends, family doctors, and emergency services is very crucial and critical. Therefore, SPD aspects are needed in such a complex networks and environments.

First, **PPTMs** play important role in social life of people. For example, bus, tram, trains, ships, cruisers, aircrafts, etc., are place that an individual can interact with other people and surrounding environments. The same when an individual is using his/her bicycle, motorcycle, car, camions, boats/yachts, etc. For example, the railways scenario is easily integrated in SMN scenario. Second, the future **ITH** is one of the most important areas of research in FP7. Additionally, internet based social networking services have experienced an enormous growth over the past years. Simultaneously to the social networking services’ triumphant advance, mobile devices in general and smart phones in particular have rapidly penetrated the consumer market. Recent phones do not only exhibit considerable computing resources but also feature means for Internet access as well as wireless short distance communication such as Bluetooth and Wi-Fi. They seem to be the perfect platform to combine the market potential of traditional social networking services and the success story of mobile devices. Even though several attempts that have tried to merge the two worlds could not reach the masses, experts expect that future mobile social networking systems possibly even exceed the success of their Internet bound counterparts. We believe that two key features

are the user's permanent reachability and location awareness, which is called P3 (Peer-to-Peer-to-Place). Third, **U&PC** devices are very tiny - even invisible - devices, either mobile or embedded in almost any type of object imaginable, including cars, tools, appliances, clothing and various consumer goods - all communicating through increasingly interconnected networks. Fourth, **SBLs** will have impact on streets and building lighting, since it is a European directive acting as new European standard. The state-of-the art of the street lighting and the future expectation in the partner-countries is enormous.

Description of Attributes, Threats, and Means for Social Mobility and Networking Scenario	
Attributes	
Threats	
Means	

7 High Level Requirements for Scenarios

7.1 Functional Requirements

7.1.1 Railway Scenario

{REQ_D2.1.1_0201.A Application scenario – Information availability
Information shall be provided continuously according to soft or real-time constraints.
}

{REQ_D2.1.1_0202.A Application scenario – Information integrity
Sensed, stored or transmitted data shall not be corrupted accidentally or in a malicious way.
}

{REQ_D2.1.1_0203.A Application scenario – Information deletion
Sensed, stored or transmitted data shall not be lost in a malicious way.
}

{REQ_D2.1.1_0204.A Application scenario – Information masking
Non-authentic data shall not be recognized as authentic.
}

{REQ_D2.1.1_0205.A Application scenario – Information privacy
Information shall be accessed only by authorized users.
}

{REQ_D2.1.1_0206.A Application scenario – ESs integration/expansion
Whenever any new ES needs to be integrated into the system, there should not be the need to develop a specific SPD assurance mechanism.
}

{REQ_D2.1.1_0207.A Application scenario – ESs integration/expansion
Whenever any new ES needs to be integrated into the system, it should be possible to easily evaluate the impact of the modification on the overall system SPD.
}

{REQ_D2.1.1_0208.A Application scenario – SPD parameters assurance/evaluation
The holistic assurance and evaluation of SPD parameters (e.g. for assessment/certification purposes) shall be possible.
}

{REQ_D2.1.1_0209.A Application scenario – Mechanisms for failure mitigation
Mechanisms to mitigate the effects on the system of software, hardware, and transmission failures shall be provided.
}

{REQ_D2.1.1_0210.A Application scenario – ESs vitality checking
Vitality checking mechanisms for the ESs shall be provided, in order to recognize their unavailability.
}

{REQ_D2.1.1_0211.A Application scenario – Dynamic adaptivity to the available resources
The ESs should be able to dynamically adapt the quality of the transmitted data and/or their SPD level according to the available resources and performance constraints (e.g. available bandwidth, CPU load, and memory occupation).
}

{REQ_D2.1.1_0212.A Application scenario – Maintainability
In case of permanent failures, failed components should be replaceable with a limited effort, and with no SPD impact or with predictable SPD impact of the repair intervention.
}

7.1.2 Voice/Facial recognition Scenario

{REQ_D2.1.1_0801.A Voice/Facial Recognition Scenario: Biometric data security
Biometric data should be stored in a way it's not possible to rebuild the analog source (image or voice).
}

{REQ_D2.1.1_0802.A Voice/Facial Recognition Scenario: Biometric data privacy
The original voice or image signal must be deleted after analysis.
}

{REQ_D2.1.1_0803.A Voice/FacialRecognition Scenario: Data storage
Data should be not easy available. The biometric and matching process is executed within the embedded system, which make available only the results of the elaboration.
}

{REQ_D2.1.1_0804.A Voice/FacialRecognition Scenario: Collaborative process
In the case the recognition system is not completely autonomous; the user may collaborate to the recognition result.
}

{REQ_D2.1.1_0805.A Voice/Facial Recognition Scenario: Data encrypting
Each data stored must be encrypted before being sent. I.e., the encryption policy could be based on SSL protocol.
}

{REQ_D2.1.1_0806.A Voice/Facial Recognition Scenario: Heartbeat
The recognition system shall send a periodic keep alive message independent from reads, to inform the supervisor if the system fails.
}

{REQ_D2.1.1_0807.A Voice/Facial Recognition Scenario: thresholds
The recognition system should not be a deterministic system. It is a self-learning system that needs a training procedure. For this reason it doesn't provide a specific result but a matching score, so it's necessary to find the correct thresholds to determinate the results.
}

7.1.3 Avionics Scenario

{REQ_D2.1.1_1201.A Avionics Scenario – Data Availability
Each Data shall be provided continuously according to the own relevant-time constraints.
}

{REQ_D2.1.1_1202.A Avionics Scenario – Data Acquisition Integrity
The integrity of each Acquired Data from Sensors and/or Equipment shall be guaranteed through a defined acquisition check.
}

{REQ_D2.1.1_1203.A Avionics Scenario – Data Storing Integrity
The integrity of each of each stored data into the Database shall be guaranteed through a defined a storing procedure.
}

{REQ_D2.1.1_1204.A Avionics Scenario – Data Transmission Integrity

The integrity of each transmitted data to any equipment shall be guaranteed through a defined transmission check.
}

{REQ_D2.1.1_1205.A Avionics Scenario – Data Preservation
Each Data shall be preserved against any type of loss or accidentally deletion.
}

{REQ_D2.1.1_1206.A Avionics Scenario – Data Masking
Each data shall be masked/unmasked through the dedicated procedure.
}

{REQ_D2.1.1_1207.A Avionics Scenario – Information privacy
Data shall be accessed only by authorized operators.
}

{REQ_D2.1.1_1208.A Avionics Scenario – ESs integration/expansion
Any Avionic ES shall be integrated into the Avionic System without developing a specific SPD assurance mechanism.
}

{REQ_D2.1.1_1209.A Avionics Scenario – ESs integration/expansion
The impact of the modification on the overall system SPD shall be easily evaluated, after the integration/expansion of any Avionic ESs.
}

{REQ_D2.1.1_1210.A Avionics Scenario – nSHIELD compliant
Any Avionic ESs should be easily recognized as nSHIELD compliant.
}

{REQ_D2.1.1_1211.A Avionics Scenario – Procedure against SW failure
Procedure against the ES software failures shall be provided.
}

{REQ_D2.1.1_1212.A Avionics Scenario – Procedure against HW failure
Procedure against the ES hardware failures shall be provided.
}

{REQ_D2.1.1_1213.A Avionics Scenario – Procedure against TX/RX failure
Procedure against the TX/RX failure inside the Avionic System shall be provided.
}

{REQ_D2.1.1_1214.A Avionics Scenario – SPD Static configuration
SPD Static configuration on Avionic Systems shall be provided.
}

{REQ_D2.1.1_1215.A Avionics Scenario – SPD dynamic configuration
SPD dynamic configuration on Avionic Systems shall be provided.
}

{REQ_D2.1.1_1216.A Avionics Scenario – Dynamic adaptivity to the available resources
The ESs should be able to dynamically adapt the quality of the transmitted data and/or their SPD level according to the available resources and performance constraints (e.g. available bandwidth, CPU load, and memory occupation).
}

{REQ_D2.1.1_1217.A Avionics Scenario – Maintainability

In case of permanent failures, failed components should be replaceable with a limited effort, and with no SPD impact or with predictable SPD impact of the repair intervention.
}

7.1.4 Social Mobility and Networking Scenario

{REQ_D2.1.1_2001.A Permanently reachability and location awareness of data
In the nSHIELD data shall be permanently reachable and location aware.

Traceability:

}

{REQ_D2.1.1_2002.A Integration of infrastructures
The nSHIELD system should integrate the old and new communication infrastructures.

Traceability:

}

{REQ_D2.1.1_2003.A Real-life networks
The nSHIELD system should include large and complex networks called *real-life networks* that include: electrical power grid, World Wide Web, the Internet backbone, collaboration networks, and airline connection networks.

Traceability:

}

{REQ_D2.1.1_2004.A Security, Trust, Privacy and Dependability
The nSHIELD system shall provide Security, Trust, Privacy and Dependability (STPD).

Traceability:

}

{REQ_D2.1.1_2005.A Security of data
In the nSHIELD data from embedded systems shall be secure.

Traceability:

}

{REQ_D2.1.1_2006.A Privacy of data
In the nSHIELD users' data shall be shared only with authorized people.

Traceability:

}

{REQ_D2.1.1_2007.A Dependability of data
In the nSHIELD the handling of data from embedded system shall depend on the user preferences, the situation or context and goal.

Traceability:

}

{REQ_D2.1.1_2008.A Body Area Networks (BANs), Body Sensor Networks (BSNs)
The nSHIELD system shall collect sensory data from Body Area Networks (BANs), or Body Sensor Networks (BSNs) and send it to social networks.

Traceability:

}

{REQ_D2.1.1_2009.A Real time dissemination of data
The nSHIELD shall provide dissemination of sensory data in real time.

Traceability:

}

{REQ_D2.1.1_2010.A Communication through interconnected networks
The nSHIELD system should provide communication through interconnected networks.

Traceability:

}

{REQ_D2.1.1_2011.A Testing the effectiveness of SPD functionalities
The nSHIELD project shall test the effectiveness of the proposed SPD functionalities:

- Comparing the required SPD levels with the ones actually achieved by implementing the SPD nSHIELD solutions;
- Integrating them in a complete platform;
- Testing.

Traceability:

}

7.2 Structural Requirements

7.3 nSHIELD Applications

7.4 nSHIELD Services

8 SPD High Level Requirements for nSHIELD System

8.1 nSHIELD System

{REQ_D2.1.1_2101.A Authorisation
Access to nSHIELD system's resources shall be granted only to authorised entities following a successful entity identification and authentication.
Traceability:
}

{REQ_D2.1.1_2102.A Information access
Access to nSHIELD system's data shall be granted on a need to know basis and only after proper authorisation.
Traceability: {REQ_D2.1.1_2101.A
}

{REQ_D2.1.1_2103.A Policy driven SPD management
The nSHIELD system SPD functionalities shall be configurable by authorised entities to satisfy policy requirements.
Traceability:
}

{REQ_D2.1.1_2104.A Security reconfiguration
The nSHIELD system security functionalities shall be reconfigurable by authorised entities to satisfy modifications to policy requirements.
Traceability:
}

{REQ_D2.1.1_2105.A Modular security architecture
The nSHIELD system security architecture should be based on independent security modules to allow dynamic security functions deployment
Traceability: {REQ_D2.1.1_2103.A
}

{REQ_D2.1.1_2106.A Adaptable security
The nSHIELD system overall security should be adaptable to provide enhanced protection against emerging threats.
Traceability: {REQ_D2.1.1_2104.A
}

{REQ_D2.1.1_2107.A System Composability
The nSHIELD system's composition of a valid set of SPD modules shall enhance security and not introduce vulnerabilities to overall system's protection...
Traceability:
}

{REQ_D2.1.1_2108.A Expandable security
The nSHIELD system functionalities shall be expandable to accommodate appropriate measures against emerging threats.
Traceability: {REQ_D2.1.1_2104.A
}

{REQ_D2.1.1_2109.A Compromised system detection
The nSHIELD system shall be able to detect and easily identify unauthorised disclosure, destruction, removal, modification or interruption or use of information and resources occurred either in the supply chain, during deployment or normal operation.
Traceability:

}

{REQ_D2.1.1_2110.A Defence in depth
The nSHIELD system should adopt a layered architecture of security measures to enhance robustness and cope with failures.
Traceability:
}

{REQ_D2.1.1_2111.A Fail-safe measures
The nSHIELD system should incorporate fail-safe measures to avoid jeopardising the overall system's security or dependability in the event of failure.
Traceability:
}

{REQ_D2.1.1_2112.A Security attestation
The nSHIELD system should be evaluated and certified, and its accreditation level should be communicated to concerned parties.
Traceability: {REQ_D2.1.1_2103.A
}

{REQ_D2.1.1_2113.A Auditing
The nSHIELD system shall provide auditing functionalities. The level of auditing should be configurable.
Traceability: {REQ_D2.1.1_2103.A
}

{REQ_D2.1.1_2114.A System monitoring
The nSHIELD system shall provide monitoring functionalities for security functions.
Traceability:
}

{REQ_D2.1.1_2115.A Standards compliance
The nSHIELD system should adhere to applicable international standards.
Traceability:
}

{REQ_D2.1.1_2116.A Malware protection
The shield system shall deploy measures to protect against potentially hazardous (e.g. malicious) code attempting to bypass controls and make unauthorised use of its resources.
Traceability:
}

{REQ_D2.1.1_2117.A Covert channel protection
The nSHIELD system shall take measures to protect against unauthorised leakage of information through covert channels.
Traceability:
}

{REQ_D2.1.1_2118.A Firmware updates
All firmware updates shall be authorized and accomplished in a way that the system's overall security level is not reduced or jeopardised.
Traceability:
}

{REQ_D2.1.1_2119.A Denial of service
The nSHIELD system shall deploy appropriate measures to protect against denial of service attacks and ensure availability.
Traceability:

}

{REQ_D2.1.1_2120.A Security through obscurity
The nSHIELD system security should not be based on obscurity.
Traceability:
}

{REQ_D2.1.1_2121.A System documentation
The nSHIELD system security functionality, parameterization, management and procedures regarding secure deployment, handling, operation and device termination shall be well documented and available to interested parties.
Traceability:
}

{REQ_D2.1.1_2122.A Unambiguous communications
The system shall be able to communicate with other system components and devices through well-defined interfaces and messages that will ensure interoperability and exchange of unambiguous messages.
Traceability:
}

{REQ_D2.1.1_2123.A Secure communications
The system shall support the establishment of secure channels with other devices or components at the network and/or application layer.
Traceability:
}

{REQ_D2.1.1_2012.A nSHIELD System (nSS)
The **nSHIELD System** shall be composed as a heterogeneous system of a group of interacting, interrelated, or interdependent composable embedded devices and sub-systems with SPD functionalities, other sub-systems and elements, e.g., legacy devices (LDs), and external systems, e.g., legacy, public information systems, and other systems forming a complex whole.

Traceability:

}

{REQ_D2.1.1_2013.A nSHIELD System Components
The nSHIELD system shall have the following functional layers: Node, Network, Middleware and Overlay.

Traceability:

}

{REQ_D2.1.1_2014.A Sensors systems
The nSHIELD system shall integrate sensors systems relevant to all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2015.A Actuators
The nSHIELD system shall integrate actuators relevant to all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2016.A Nodes

The nSHIELD system shall integrate nodes as elements of the nSS relevant to all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2017.A Communications of messages, voice and video
The nSHIELD system shall provide communications of messages, voice and video.

Traceability:

}

{REQ_D2.1.1_2018.A System Security, Privacy and Dependability
The nSHIELD system shall be a secure and dependable system and must respect privacy policies.

Traceability:

}

{REQ_D2.1.1_2019.A Security attributes
The nSHIELD system should have security attributes:

- Accessibility
- Accountability
- Availability
- Authenticity
- Confidentiality
- Integrity
- Non-repudiation
- Safety

Traceability: {REQ_D2.1.1_2018.A
}

{REQ_D2.1.1_2020.A Dependability attributes
The nSHIELD system should have dependability attributes:

- Availability
- Reliability
- Safety
- Confidentiality
- Integrity
- Maintainability

Traceability: {REQ_D2.1.1_2018.A
}

{REQ_D2.1.1_2021.A Security, Privacy and Dependability (SPD) functionalities
The nSHIELD system shall implement SPD functionalities for all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2022.A SPD transmission
The nSHIELD system should perform SPD smart driven transmission for all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2023.A Trusted and dependable connectivity
The nSHIELD system shall allow trusted and dependable connectivity for all chosen scenarios.

Traceability: {REQ_D2.1.1_2022.A

}

{REQ_D2.1.1_2024.A SPD core service
The nSHIELD shall allow the SPD core services for all chosen scenarios.

Traceability: {REQ_D2.1.1_2021.A

}

{REQ_D2.1.1_2025.A SPD metrics
The nSHIELD system shall have SPD metrics for all chosen scenarios.

Traceability: {REQ_D2.1.1_2022.A

}

{REQ_D2.1.1_2026.A Semantics and ontology
The nSHIELD system shall support semantics and ontology technologies for all chosen scenarios.

Traceability:

}

{REQ_D2.1.1_2027.A Policy based management
The nSHIELD system should support policy based management.

Traceability:

}

{REQ_D2.1.1_2028.A Compatibility
The nSHIELD system shall be compatible with the supported communication standards.

Traceability

}

{REQ_D2.1.1_2029.A Redundancy
The nSHIELD should support redundancy functions.

Traceability:

}

{REQ_D2.1.1_2030.A Audit Functionalities
The nSHIELD system shall guarantee Audit functionalities

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_2017.A

}

{REQ_D2.1.1_2031.A Cryptographic Support
The nSHIELD system shall guarantee cryptographic support.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_2017.A

}

{REQ_D2.1.1_2032.A Non-repudiation functionalities
The nSHIELD system shall guarantee non-repudiation functionalities

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_2017.A

}

{REQ_D2.1.1_2033.A Access control functionalities
The nSHIELD system shall guarantee access control functionalities on users, assets and operations among them.

Traceability:

}

{REQ_D2.1.1_2034.A Identification and Authentication functionalities
The nSHIELD system shall guarantee users Identification and Authentication functionalities

Traceability:

}

{REQ_D2.1.1_2035.A Management of Security Functionalities
The nSHIELD system shall guarantee the management of Security Functionalities

Traceability:

}

{REQ_D2.1.1_2036.A Composability
The nSHIELD SPD modules should be designed and developed to be seamlessly composable by means of open, dependable interfaces.

Traceability:

}

{REQ_D2.1.1_2037.A Static and dynamic composability
The nSHIELD SPD modules should allow both a static and dynamic composability of SPD functionalities, to guarantee the agreed level of measured SPD metrics of the overall system.

Traceability:

}

{REQ_D2.1.1_2038.A Homogeneous metadata
Heterogeneous measurements and parameters in the nSHIELD system should be converted by the security agents in *homogeneous metadata* by extensively using properly selected semantic technologies.

Traceability:

}

{REQ_D2.1.1_2039.A Replacement of modules
Single nSHIELD SPD modules should allow replacement once the measured SPD metrics do not satisfy the required SPD levels.

Traceability:

}

{REQ_D2.1.1_2040.A Heterogeneity
The nSHIELD system should integrate heterogeneous platforms exploiting their composability capabilities, aims at offering optimized resources management, through the “ad-hoc” formation and collaboration of sub-networks, according to each time needs and availabilities.

Traceability:

}

{REQ_D2.1.1_2041.A Traceability
In the nSHIELD system sensor networks should allow product traceability all along the path between assembly line and customer delivery.

Traceability:

}

8.1.1 nSHIELD Node

8.1.1.1 SPD Node

{REQ_D2.1.1_2042.A Modules of node level
At node level of nSHIELD system there should be two modules: trusted platform module (TPM) and cryptographic techniques (CRYPTO).

Traceability:

}

{REQ_D2.1.1_2301.A Power management
In the nSHIELD system the hardware shall provide frequency dividers and variable supply voltage, in order to meet the most appropriate energy/performance trade-off.

Traceability:

}

{REQ_D2.1.1_2302.A Power management interface
In the nSHIELD system the hardware shall provide an appropriate interface, such as I/O ports or memory mapped registers, in order to allow the software to drive the frequency dividers and the supply multiplexers.

Traceability: {REQ_D2.1.1_2301.A

}

{REQ_D2.1.1_2303.A Workload assessment
In the nSHIELD system the operating system shall provide information on the past and on the current workload, for example the number of processes and the expired deadlines in a given amount of time, to allow the management software to choose the most convenient level of performance.

Traceability: {REQ_D2.1.1_2301.A

}

8.1.1.2 Legacy Node

8.1.2 nSHIELD Network

8.1.2.1 SPD Network

{REQ_D2.1.1_2304.A Remote management
In the nSHIELD system the operating system or the middleware shall implement routines to send requests to neighbours, to force remote nodes to change their working point, in order to react to node failures, or to accept portion of a distributed computation.

Traceability:

}

8.1.2.2 Legacy Network

8.1.3 nSHIELD Middleware

{REQ_D2.1.1_2305.A Status delivering
In the nSHIELD system the operating system or the middleware shall implement routines to send information about the node status to its neighbours (periodically or on demand).

Traceability: {REQ_D2.1.1_2304.A

}

{REQ_D2.1.1_2306.A Global status inferring
In the nSHIELD system the operating system or the middleware shall implement routines that elaborate data received from neighbours and estimate the current status of the system.

Traceability: {REQ_D2.1.1_2304.A

}

8.1.4 nSHIELD Overlay

8.1.5 nSHIELD SPD Metrics

{REQ_D2.1.1_2043.A Security, Privacy and Dependability Metrics
The nSHIELD system should have security, privacy and dependability metrics.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_2044.A Attributes of dependability
The nSHIELD system shall support the attributes: Availability, Reliability, Safety, Integrity, Maintainability, and Confidentiality that make it dependable.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_2045.A Measuring attributes of dependability
The nSHIELD system shall support methods to measure attributes of dependability, i.e., a set of dependability metrics.

Traceability: {REQ_D2.1.1_2043.A

}

{REQ_D2.1.1_2046.A SMART Metrics
Metrics used in the nSHIELD system shall be SMART, i.e. specific, measurable, attainable, repeatable and time-dependent.

Traceability: {REQ_D2.1.1_2045.A

}

{REQ_D2.1.1_2047.A Static and Dynamic Metrics
The nSHIELD system should use static and dynamic metrics.

Traceability: {REQ_D2.1.1_2046.A

}

{REQ_D2.1.1_2048.A SPD level at each layer and for the overall system
The nSHIELD project should identify the embedded system desired SPD level at each layer (Node, Network, Middleware, and Overlay) and for the overall system with respect to SPD metrics.

Traceability: {REQ_D2.1.1_2043.A

}

{REQ_D2.1.1_2049.A Situational-aware and context-aware capabilities
The nSHIELD framework situational-aware and context-aware capabilities should be analyzed with respect to precision, classification and assessment required metrics in order to provide a quantitative and qualitative evaluation of obtained SPD performances.

Traceability: {REQ_D2.1.1_2043.A

}

{REQ_D2.1.1_2050.A Validation of performance of nSHIELD framework
Metrics should validate and define performance of nSHIELD framework in terms of required SPD levels independently from a specific domain.

Traceability: {REQ_D2.1.1_2043.A

}

{REQ_D2.1.1_2051.A Human factors
The nSHIELD system should consider human factors, i.e., knowledge or awareness of individuals or groups.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_2052.A Threats, attributes and means
The nSHIELD system shall have threats, attributes and means.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_2053.A Security assurance (SA) assessment
The nSHIELD system should allow security assurance (SA) assessment.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_2054.A Aggregation of SA values
The nSHIELD system should allow aggregation of SA values representing the SA levels of an entity.

Traceability: {REQ_D2.1.1_2053.A

}

{REQ_D2.1.1_2055.A Emerging attributes
The nSHIELD system should allow emerging attributes during the aggregation.

Traceability: {REQ_D2.1.1_2053.A

}

{REQ_D2.1.1_2056.A Composition and decomposition
The nSHIELD system should allow composition and decomposition.

Traceability: {REQ_D2.1.1_2053.A

}

{REQ_D2.1.1_2057.A Attribute-dependency graph
The nSHIELD system should allow representation as an attribute-dependency graph.

Traceability: {REQ_D2.1.1_2053.A

}

{REQ_D2.1.1_2058.A Aggregation operators
The nSHIELD system should allow aggregation operators: max, min, union, multiply, weighted-sum, average, as well as other operators needed for aggregation of the SPD metrics.

Traceability: {REQ_D2.1.1_2053.A

}

{REQ_D2.1.1_2059.A Attack graphs
The nSHIELD system should use attack graphs for static metrics evaluation.

Traceability: {REQ_D2.1.1_2053.A

}

8.2 nSHIELD Reference System Architecture

{REQ_D2.1.1_2124.A Anonymity on location-based services
The nSHIELD system should provide anonymous location-based services.

Traceability:

}

{REQ_D2.1.1_2125.A Lightweight crypto
The nSHIELD system should provide lightweight crypto functionalities suitable for constrained devices.

Traceability:

}

{REQ_D2.1.1_2126.A Side-channel attacks
The nSHIELD system should employ appropriate measures to protect against side-channel attacks.

Traceability:

}

{REQ_D2.1.1_2127.A Intrusion detection system
The nSHIELD system should employ an intrusion detection system.

Traceability:

}

{REQ_D2.1.1_2128.A Heterogeneous measurements and metadata
The nSHIELD system should be able to process heterogeneous measurements homogenized through the use of semantics.

Traceability:

}

{REQ_D2.1.1_2129.A Dynamic selection of SPD modules
SPD modules should be dynamically selectable through the deployment of appropriate control algorithms.

Traceability:

}

{REQ_D2.1.1_2060.A Web Service
In the nSHIELD system each device or software component should be represented as a web service.
Traceability:
}

{REQ_D2.1.1_2061.A nSHIELD Service Oriented Architecture (SOA)
The nSHIELD system should be a SOA based system.
Traceability:
}

{REQ_D2.1.1_2062.A Semantic model-based architecture
The nSHIELD system should have a generic semantic model-based architecture.

Traceability:

}

{REQ_D2.1.1_2063.A Semantic Model Driven Architecture
The nSHIELD system should define

- Device Ontology
- Security Ontology
- Software Components Ontology

Traceability: {REQ_D2.1.1_2062.A

}

{REQ_D2.1.1_2064.A Conceptual design
The nSHIELD system architecture based on the four functional layers (node, network, middleware and overlay) should be conceptually designed for the development of software components that are reusable across the pervasive computing applications.

Traceability:

}

{REQ_D2.1.1_2065.A Web services
In the nSHIELD the SPD-WSN should allow Web services and access its NMP sensors and actuators through a SPD node that act as Gateway to connect SPD-WSN on IP based network.

Traceability:

}

8.2.1 Assurance process requirements

{REQ_D2.1.1_1501.A Configuration management(ACM)
Units/components developed for nSHIELD project SHALL have version identifiers unique per each release of unit/component. These version identifiers will be used to identify versions and trace version changes.

Traceability: [TA].AT4 – *Reduce the* effort and time required for re-validation and recertification after change

}

{REQ_D2.1.1_1502.A Deployment manual(ADO)
nSHIELD partners developing system components SHALL provide description on how to securely generate and deploy system components, and how to operate them safely

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection, [TA].AT2 – Reduce the costs of development cycles, especially in sectors requiring qualification or certification, [TA].IP3 – Test, validation and verification tools:

}

{REQ_D2.1.1_1503.A Automated testing tools
Assigned nSHIELD project partner(s) SHALL create automated testing tool(s) to generate arbitrary (even potentially faulty or malicious) input to exercise and evaluate input/output and internal interfaces of the critical components. Documentation, especially guidance documents (see {REQ_D2.1.1_1504.A) MAY be used as a basis to generate relevant test excitation.

Traceability: [TA].IP3 – Test, validation and verification tools, {REQ_D2.1.1_1505.A

}

{REQ_D2.1.1_1504.A Guidance documents (AGD)
nSHIELD partners developing system components SHALL provide user documentation on how to use the system securely. The guidance shall include warnings about actions that that can cause errors and lead to faults or failures in the secure environment.

Traceability: [TA]. AT2 – Reduce the costs of development cycles, especially in sectors requiring qualification or certification, [TA].IP3 – Test, validation and verification tools, {REQ_D2.1.1_1503.A

}

{REQ_D2.1.1_1505.A Vulnerability Assessment (AVA)
Using results from test, validation, and verification tools, assigned nSHIELD project partner(s) SHALL carry out security analysis on the system components for all scenarios in order to find potential threats that could leave the nSHIELD system vulnerable.

Traceability: [TA].IP2 – Architectural Dependability, [TA].IP3 – Tst, validation and verification tools:

}

9 Node Requirements and Specifications

NOTE: In the following, the expression "nSHIELD node" is referred to all three node levels of increasing complexity: nano node, micro/personal node and power node.

9.1 High level requirements

Below, we list the identified nSHIELD requirements. The requirements have been derived based on missing needs and under the assumption that some important high level requirements are inherited from the pSHIELD requirements document [8]. The pSHIELD inherited requirements are indicated in the tractability field.

{REQ_D2.1.1_1701.A Node – dependability mechanism
A nSHIELD node should be designed with built-in mechanisms improving system dependability.

Traceability: {REQ_D2.1_20016.A, Technical Annex – Abstract

}

{REQ_D2.1.1_1702.A Node – composability
A nSHIELD node shall support composability; the nodes shall be assembled (physically as well as logically) in various combinations to satisfy specific user requirements.

Traceability: {REQ_D2.1_20016.A, Technical Annex – 1.1.2. IP1 - Composability

}

{REQ_D2.1.1_1703.A Node – built-in security mechanisms
A nSHIELD node should be designed with security built-in mechanisms.

Traceability: {REQ_D2.1_20016.A, Technical Annex - Abstract

}

{REQ_D2.1.1_1704.A Node – integrity protection
A nSHIELD node should provide mechanisms that guarantee data integrity based on hardware "hooks" and secure key installation.

Traceability: {REQ_D2.1_20016.A, Technical Annex – Task 3.1.

}

{REQ_D2.1.1_1705.A Node – Minimised development overhead
Functionalities in an nSHIELD node should not significantly increase time and costs for development of tasks.

Motivation:

- Artemis targets: AT1 reduce the cost of the system design and AT2 Reduce the costs of development cycles, especially in sectors requiring qualification or certification.

}

{REQ_D2.1.1_2066.A Node - Power Node SPD level
In the nSHIELD the highest required SPD level for the Power Node shall be clearly identified.

Traceability:

}

9.2 Mid-level requirements

{REQ_D2.1.1_1706.A Node – Verifiable security
The security of an nSHIELD node shall be verifiable.

Traceability: {REQ_D2.1.1_1701.A, {REQ_D2.1.1_1703.A

Motivation:

- industry priorities: IP2 Architectural Dependability and IP3 Test, validation and verification tools
- Artemis targets: AT4 Reduce the effort and time required for re-validation and recertification after change.

}

{REQ_D2.1.1_1707.A Node – Modular architecture
nSHIELD nodes should have a modular architecture where software and hardware components can be exchanged without any major difficulties.

Traceability: {REQ_D2.1.1_1702.A

Motivation:

- industry priorities: IP1 Composability and IP2 Architectural Dependability.
- Artemis targets: AT4 Reduce the effort and time required for re-validation and recertification after change.

}

{REQ_D2.1.1_1708.A Node – Minimal trusted core
An nSHIELD node shall have a trusted core that contains only the minimum functions needed to function securely.

Traceability: {REQ_D2.1.1_1701.A {REQ_D2.1.1_1704.A {REQ_D2.1.1_1705.A

Motivation:

- industry priorities: IP1 Composability and IP2 Architectural Dependability and IP3 Test, validation and verification tools
- Artemis target AT3: Manage the complexity with effort reduction.

}

{REQ_D2.1.1_1709.A Node – Minimised overhead in the product lifecycle for security enhancements
Development, deployment and operation of nodes should not be increase time and cost due to nSHIELD functionalities

Traceability: {REQ_D2.1.1_1705.A

}

9.2.1 Rationale

	{REQ_D2.1.1_1709.A Node – Minimised overhead in the product lifecycle for security enhancements.	{REQ_D2.1.1_1706.A Node – Verifiable security	{REQ_D2.1.1_1707.A Node - Modular architecture	{REQ_D2.1.1_1708.A Node – Minimal trusted core
{REQ_D2.1.1_1701.A Node – dependability mechanism		X		X
{REQ_D2.1.1_1702.A Node – composability			X	
{REQ_D2.1.1_1703.A Node – built-in security mechanisms		X		
{REQ_D2.1.1_1704.A Node – integrity protection				X
{REQ_D2.1.1_1705.A Node – Minimised development overhead	X		X	X

9.3 Low level requirements

{REQ_D2.1.1_1710.A Node – Support for tasks
An nSHIELD node should allow execution of tasks (e.g. user applications) at different security levels.

Traceability: {REQ_D2.1.1_1707.A

Motivation:

- industry priorities: IP1 Composability

}

{REQ_D2.1.1_1711.A Node – Well-defined interfaces
Components in an nSHIELD node shall have well-defined interfaces between tasks and between a task and the trusted core.

Traceability: {REQ_D2.1.1_1707.A, {REQ_D2.1.1_1706.A

Motivation:

- Artemis target: AT3 Manage the complexity with effort reduction.

}

{REQ_D2.1.1_1712.A Node – Secure isolation
The trusted core in an nSHIELD node should provide secure isolation between tasks.

Traceability: {REQ_D2.1.1_1707.A {REQ_D2.1.1_1706.A, {REQ_D2.1.1_1708.A

Motivation:

- Requirements: {REQ_D2.1.1_1708.A

}

{REQ_D2.1.1_1713.A Node – Support for task development
Implementation of low level functions should be based on high level languages.

Traceability: {REQ_D2.1.1_1709.A

Motivation:

- Cost of security audits
- Portability
- Maintenance

}

{REQ_D2.1.1_1714.A Node – Transparency
The security functions provided by an nSHIELD should be transparent to tasks whenever possible

Traceability: {REQ_D2.1.1_1707.A

Motivation:

- Artemis targets: AT1 reduce the cost of the system design and AT2 Reduce the costs of development cycles, especially in sectors requiring qualification or certification.

}

9.3.1 Rationale

	{REQ_D2.1.1_1713.A Node – Support for task development	**{REQ_D2.1.1_1714.A Node – Transparency**	**{REQ_D2.1.1_1712.A Node - Secure isolation**	**{REQ_D2.1.1_1711.A Node - Well-defined interfaces**	**{REQ_D2.1.1_1710.A Node – Support for tasks**
{REQ_D2.1.1_1707.A Node - Modular architecture		X	X	X	X
{REQ_D2.1.1_1708.A Node – Minimal trusted core			X		
{REQ_D2.1.1_1706.A Node – Verifiable security			X	X	
{REQ_D2.1.1_1709.A Node – Minimised overhead in the product lifecycle for security enhancements.	X				

9.4 Security

{REQ_D2.1.1_2130.A Node – Code execution
An nSHIELD node should verify only authorized code (booting, kernel and application) runs on the system.

Traceability:

}

{REQ_D2.1.1_2131.A Node – Data Freshness
An nSHIELD node should include data freshness checks to avoid replay attacks.

Traceability:

}

{REQ_D2.1.1_2132.A Node – Default Firmware/Software settings
An nSHIELD node should, at its default state, have its firmware and software update features turned OFF.

Traceability:

}

{REQ_D2.1.1_2133.A Node – Firmware/Software binaries
Firmware/software binaries intended for the update of nSHIELD nodes should feature integrity checks and be digitally signed by an entity authorized for the task, the signature being verifiable by the nSHIELD nodes.

Traceability:

}

{REQ_D2.1.1_2134.A Node – Digital Signatures
An nSHIELD node should be able to verify digital signatures even in cases where a trusted third party is not available.

Traceability:

}

{REQ_D2.1.1_2135.A Node – Dynamic security behaviour
An nSHIELD node shall be able to detect and report attacks or changes to the environment, triggering appropriate security policy adjustments.

Traceability:

}

{REQ_D2.1.1_2136.A Node – Policy updates
An nSHIELD node shall not accept security policy updates from unauthorized entities.

Traceability:

}

{REQ_D2.1.1_2137.A Node – TPM Generation and storage of Cryptographic keys
An nSHIELD node shall have an improved TPM architecture supporting lightweight and secure methods to generate and store cryptographic keys.

Traceability:

}

{REQ_D2.1.1_2138.A Node – TPM Lightweight Cryptography
An nSHIELD node shall support extended TPM functionality to include lightweight cryptographic mechanisms, specifically designed for resource-constrained environments.

Traceability:

}

{REQ_D2.1.1_2139.A Node – TPM Low Power mode
An nSHIELD node's TPM shall be able to enter into a low power state without compromising its security.

Traceability:

}

{REQ_D2.1.1_2140.A Node – TPM Context-based encryption keys
An nSHIELD node should extend TPM key generation functionality to include key generators and key parameters that depend on the context available.

Traceability:

}

{REQ_D2.1.1_2141.A Node – Elliptic Curve Cryptography (ECC)
An nSHIELD node shall include an optimized hardware implementation of an ECC or HECC public key algorithm.

Traceability:

}

{REQ_D2.1.1_2142.A Node – Key parameterisation
An nSHIELD node shall offer key size parameterisation options that map the requirements of the specific application/scenario, based on the need for SHORT, MEDIUM or LONG-TERM security.

Traceability:

}

{REQ_D2.1.1_2143.A Node – Secure key distribution mechanism
An nSHIELD node shall support a secure low-cost key distribution mechanism. The mechanism's parameters (e.g. algorithm, key length, usage, entropy etc.) will be defined by the security policy requirements, taking into consideration any restrictions that participating parties impose.

Traceability:

}

{REQ_D2.1.1_2144.A Node – Third-party key management
An nSHIELD node may offer support for third-party key management services to compensate for the shortage of ES computational power in constrained environments.

Traceability:

}

{REQ_D2.1.1_2145.A Node – Situational-aware and context-aware SPD
An nSHIELD node shall be able to provide situational-aware and context-aware SPD services.

Traceability:

}

{REQ_D2.1.1_2146.A Node – Security context establishment
An nSHIELD node should allow security context establishment and sharing, allowing more efficient keys or key material to be exchanged, thereby increasing the overall performance and security of the subsequent communications.

Traceability:

}

{REQ_D2.1.1_2147.A Node – Protection against Side-Channel Attacks (SCA)
An nSHIELD node should incorporate SCA countermeasures to protect itself against such attacks as simple power analysis (SPA), differential power analysis (DPA), as well as their electromagnetic counterparts SEMA and DEMA and fault attacks (DFA).

Traceability:

}

{REQ_D2.1.1_2148.A Node – Physical/tamper resilience
An nSHIELD node shall be designed to not compromise the SPD of the rest of the SHIELD system/deployment in the case of a malicious user gaining physical possession of the device. The device shall be resilient to tampering, micro-probing and reverse-engineering

Traceability:

}

{REQ_D2.1.1_2149.A Node – Dynamic security behaviour
An nSHIELD node shall be able to detect and report attacks or changes to the environment, triggering appropriate security policy adjustments.

Traceability:

}

{REQ_D2.1.1_2150.A Node – TPM additional cryptographic protocols
For a nSHIELD node, the TPM platform should be extended in order to implement additional cryptographic protocols (e.g. elliptic curves, Nth degree Truncated Polynomial Ring Cryptography)

Traceability:

}

{REQ_D2.1.1_2151.A Node – Nth degree Truncated Polynomial Ring Cryptography (NTRU)
An nSHIELD node shall include an optimized hardware implementation of an NTRU public key algorithm. (NTRU is patented by NTRU Cryptosystems Inc.)

Traceability:

}

{REQ_D2.1.1_2067.A Node – integrity to unauthorized accesses
A nSHIELD node should be designed with mechanisms that improve its resilience to unauthorized information alteration (integrity).

Traceability:

}

{REQ_D2.1.1_2068.A Node – availability for authorised users
A nSHIELD node should be designed with mechanisms that improve its availability for authorized users.

Traceability:

}

{REQ_D2.1.1_2069.A Node – secure firmware upgrade
A nSHIELD node should provide a mechanism that allows secure upgrading of the firmware from a remote site as well as local site.

Traceability:

}

{REQ_D2.1.1_2070.A Node – secure boot
A nSHIELD node should provide mechanisms that guarantee a secure boot.

Traceability:

}

{REQ_D2.1.1_2071.A Node – TPM
A nSHIELD node should be TPM compliant.

Traceability:

}

{REQ_D2.1.1_2072.A Node – TPM cryptographic/hash improvement
A nSHIELD node should have an improved global architecture of the embedded SW of the TPM to support future evolution of cryptographic/hash functionalities.

Traceability:

}

{REQ_D2.1.1_2073.A Node – TPM alternative communication interfaces
For a nSHIELD node, the TPM platform should be extended in order to have alternative communication interfaces, better adapted to the embedded applications than the LPC (low pin count) currently supported.

Traceability:

}

{REQ_D2.1.1_2074.A Node – TPM new specialized/dedicated commands
For a nSHIELD node, the TPM platform should be extended in order to add some specialized/dedicated commands (e.g. to further develop on-the-fly encryption).

Traceability:

}

{REQ_D2.1.1_2075.A Node – TPM additional cryptographic protocols
For a nSHIELD node, the TPM platform should be extended in order to implement additional cryptographic protocols (e.g. elliptic curves)

Traceability:

}

{REQ_D2.1.1_2076.A Node – TPM and Smartcard
The nSHIELD Node layer should support TPM and Smartcards.

Traceability:

}

{REQ_D2.1.1_2077.A Node – Lightweight HW and SW crypto technologies
The nSHIELD Node layer should support lightweight HW and SW crypto technologies.

Traceability:

}

{REQ_D2.1.1_2078.A Node – Asymmetric cryptography for low cost nodes
The nSHIELD Node layer should support asymmetric cryptography for low cost nodes.

Traceability:

}

{REQ_D2.1.1_2079.A Node – Intrinsically secure ES firmware
The nSHIELD Node layer should support intrinsically secure ES firmware.

Traceability:

}

{REQ_D2.1.1_2080.A Node - Elliptic Curve Cryptography
In the nSHIELD the Elliptic Curve Cryptography should be implemented on energy constrained NMP-SPD nodes.

Traceability:

}

{REQ_D2.1.1_2081.A Node - TPM or SW-TPM
In the nSHIELD the TPM or SW-TPM should be implemented on NMP-SPD nodes in order to guaranty enhanced security mechanisms in SPD-WSN.

Traceability:

}

{REQ_D2.1.1_2082.A Node - SW-TPM with ECC
In the nSHIELD the SW-TPM with ECC should be implemented on NMP-SPD nodes in order to guaranty enhanced security mechanisms in SPD-WSN.

Traceability:

}

{REQ_D2.1.1_2083.A Node - NMP-SPD node attributes
In the nSHIELD the NMP-SPD nodes networked in a SPD WSN should guaranty at least confidentiality, integrity, authenticity and system integrity.

Traceability:

}

{REQ_D2.1.1_2084.A Node - Lightweight security
In the nSHIELD the NMPS node security design should be lightweight as always in order to fit into the inherent resource-constrained nature of the sensor nodes.

Traceability:

}

9.5 Dependability

{REQ_D2.1.1_2152.A Node – Continuous power supply source
An nSHIELD node's power supply should offer a continuous power, voltage and current to the powered devices.

Traceability:

}

{REQ_D2.1.1_2153.A Node – TPM Remote attestation
An nSHIELD node should utilize the TPM remote attestation functionality to ensure the integrity of the nodes prior to the transmission of sensitive information or resource allocation.

Traceability:

}

{REQ_D2.1.1_2154.A Node – TPM Remote attestation
An nSHIELD node should utilize the TPM remote attestation functionality to ensure the integrity of a node prior to resource allocation.

Traceability:

}

{REQ_D2.1.1_2155.A Power Node –Virtualization
An nSHIELD power node should employ virtualization techniques to allow concurrent virtual nodes to run independently onto the system, thus offering a virtualized hardware redundancy mechanism. This is especially important in applications where dependability is a key parameter (e.g. avionics).

Traceability:

}

{REQ_D2.1.1_2156.A Redundancy
An nSHIELD node may be designed with built-in hardware and firmware redundancy in applications where dependability is an extremely high priority (e.g. avionics).

Traceability:

}

{REQ_D2.1.1_2157.A Node – Runtime Reconfiguration
An nSHIELD node should support the concept of runtime reconfiguration, being able to modify or change the functionality configuration of the device during both normal operation and fault, through either hardware or software changes.

Traceability:

}

{REQ_D2.1.1_2158.A Node – Alternative power supply sources
An nSHIELD node should support alternative power modes, depending on the specific application and environmental conditions (e.g. vibration generator, micro-solar cells).

Traceability:

}

{REQ_D2.1.1_2159.A Node – Resilience to Denial of Service (DoS) Attacks

An nSHIELD node shall incorporate mechanisms that increase its resilience to DoS attacks and ensure its availability to authorized entities.

Traceability:

}

{REQ_D2.1.1_2160.A Node – Dependable authentic key distribution mechanisms
An nSHIELD node shall support secure and dependable low-cost key distribution mechanisms for initialisation or re-keying.

Traceability:

}

{REQ_D2.1.1_2085.A Node – system availability
A nSHIELD node should be designed with mechanisms that improve system availability.

Traceability:

}

{REQ_D2.1.1_2086.A Node – system integrity
A nSHIELD node should be designed with mechanisms that improve system integrity.

Traceability:

}

{REQ_D2.1.1_2087.A Node – safety
A nSHIELD node should be designed with mechanisms that improve safety.

Traceability:

}

{REQ_D2.1.1_2088.A Node – system maintainability
A nSHIELD node should be designed with mechanisms that improve system maintainability, such as self-reconfigurability, self-recovery or firmware upgrade mechanisms.

Traceability:

}

{REQ_D2.1.1_2089.A Node – Self Test
A nSHIELD node should perform a complete self-test of all functions.

Traceability:

}

{REQ_D2.1.1_2090.A Node – uninterruptible Power Supply
For a nSHIELD node power supply should be provided continuously, without any cut in time neither in the power, voltage or current levels, to correctly bias the devices.

Traceability:

}

{REQ_D2.1.1_2091.A Node – Power Supply monitoring

For a nSHIELD, power supply should be monitored continuously in order to prevent any system power risk, which might affect the node.

Traceability:

}

{REQ_D2.1.1_2092.A Node – Power Supply fault tolerance
A nSHIELD should support mechanisms to protect itself from any power supply failure.

Traceability:

}

{REQ_D2.1.1_2093.A Node – remote powering
A nSHIELD should be able to support remote powering, at least to some modules of the device, allowing some functionalities to become operational in case of power failure.

Traceability:

}

{REQ_D2.1.1_2094.A Node – fault-tolerance
A nSHIELD power node should be designed with hardware and firmware redundancy to implement fault-tolerance.

Traceability:

}

{REQ_D2.1.1_2095.A Node – fail-controlled
A nSHIELD power node should fail in a controlled way, meaning that node failures are, to an acceptable extent, halting and signalled.

Traceability:

}

{REQ_D2.1.1_2096.A Node – Power supply protection
The nSHIELD Node layer should have power supply protection.

Traceability:

}

{REQ_D2.1.1_2097.A Node – Self-re-configurability and self-recovery
The nSHIELD Node layer should provide self-re-configurability and self-recovery of sensing and processing tasks.

Traceability:

}

{REQ_D2.1.1_2098.A Node – Easy and dependable interfaces with sensors
The nSHIELD Node layer should have easy and dependable interfaces with sensors.

Traceability:

}

{REQ_D2.1.1_2099.A Node – Embedded camera array

The nSHIELD Node layer should provide embedded camera array auto-calibration and auto configuration.

Traceability:

}

{REQ_D2.1.1_20100.A Node - Node replacement
In the nSHIELD throughout the lifetime of a deployment, nodes may be relocated or replaced due to outages, and discharged batteries.

Traceability:

}

9.6 Privacy

{REQ_D2.1.1_2161.A Node – Physical/tamper resilience
An nSHIELD node shall be designed to not compromise the privacy of the contained information in the case of a malicious user gaining physical possession of the device. The device shall be resilient to tampering, micro-probing and reverse-engineering.

Traceability:

}

{REQ_D2.1.1_2162.A Node – ECC Authentication
An nSHIELD node should include an optimized hardware implementation for an ECC-based public-key authentication algorithm,.

Traceability:

}

{REQ_D2.1.1_2163.A Node – Location Privacy
An nSHIELD node shall feature privacy-aware management of location, utilizing secure storage and sanitization of such information prior to transmission.

Traceability:

}

{REQ_D2.1.1_2164.A Node – Storage of private information
An nSHIELD node shall incorporate provisions that ensure the long term storage of private information, not allowing the confidentiality of that information to be compromised even under fault conditions.

Traceability:

}

{REQ_D2.1.1_2165.A µNode / Wearable personal node – Anonymity & Location Privacy
An nSHIELD wearable personal node must feature privacy-aware management of location and other sensitive personal information, utilizing secure storage and sanitization mechanisms to be applied to such information prior to transmission.

Traceability:

}

{REQ_D2.1.1_2166.A µNode / Wearable personal node – Storage of private information
An nSHIELD node must incorporate provisions that ensure the long term storage of personal sensitive information, not allowing the confidentiality of that information to be compromised even under fault conditions.

Traceability:

}

{REQ_D2.1.1_2167.A Node – Privacy in different trust domains
An nSHIELD node shall feature the necessary mechanisms for security token exchange to enable the issuance and dissemination of credentials within different trust domains.

Traceability:

}

{REQ_D2.1.1_2168.A Node – NTRU Authentication
An nSHIELD node should include an optimized hardware implementation for an NTRU-based public-key authentication algorithm.

Traceability:

}

{REQ_D2.1.1_20101.A Node – built-in privacy mechanisms
A nSHIELD node should be designed with built-in mechanisms that provide information privacy.

Traceability: Technical Annex – Abstract

}

{REQ_D2.1.1_20102.A Node – asymmetric cryptography
A nSHIELD node should support a hardware implementation of asymmetric cryptography.

Traceability:

}

{REQ_D2.1.1_20103.A Node – automatic access control
A nSHIELD node should support an automatic Access Control.

Traceability:

}

{REQ_D2.1.1_20104.A Node – secure authentication
A nSHIELD node shall support a secure authentication protocols.

Traceability:

}

{REQ_D2.1.1_20105.A Node - Data compression techniques
The nSHIELD Node layer should support data compression techniques.

Traceability:

}

9.7 Composability

{REQ_D2.1.1_2169.A Node – eNetwork / Hybrid network compatibility
An nSHIELD node should be designed to allow switching between infrastructure-centric and ad-hoc networks on demand, in order to adapt continually to changes in the physical environment.

Traceability:

}

{REQ_D2.1.1_2170.A Node – Flexible key distribution mechanisms
An nSHIELD node should support a flexible and secure low-cost key distribution mechanism for initialisation or re-keying, allowing the distribution of authentic public keys via insecure channels, either according to a pre-defined schedule or ad-hoc, while adhering to policy requirements as well as requirements participating parties impose.

Traceability:

}

{REQ_D2.1.1_2171.A Node – Conflict resolution between policy domains
In case nSHIELD nodes in different policy domains need to communicate, there should be mechanisms in place to facilitate the communication and resolve this conflict with the minimum processing and communication overhead.

Traceability:

}

{REQ_D2.1.1_2172.A Node – Dynamic security behaviour
An nSHIELD node shall be able to change its security behaviour based on the dynamic change of policy requirements without requiring reprogramming or shutting down the node.

Traceability:

}

{REQ_D2.1.1_20106.A Node – commands for composition and configuration
A nSHIELD node should be able to support static and dynamic composability through commands received from the nSHIELD overlay.

Traceability:

}

9.8 Performance/Metrics

{REQ_D2.1.1_2173.A Node – Lightweight embedded operating system
nSHIELD nodes' operating system shall have low resource requirements.

Traceability:

}

{REQ_D2.1.1_2174.A Node – Hardware/Software co-design
An nSHIELD nodes should incorporate hardware-software co-design techniques to substantially increase application (e.g. public-key cryptography) performance with minimal device surface area and cost increase.

Traceability:

}

{REQ_D2.1.1_2175.A Node – ES Certifications
An nSHIELD node shall reach a security level which will make it ready for future ES certifications.

Traceability:

}

{REQ_D2.1.1_2176.A Node – Situational-aware and context-aware SPD
An nSHIELD node should be able to provide situational-aware and context-aware SPD services, thus offering optimization of the available system resources and maximization of performance.

Traceability:

}

{REQ_D2.1.1_20107.A Node – Performance Parameters monitoring
A nSHIELD node should monitor its Performance Parameters and report alert or alarm conditions to the external systems when the defined thresholds for alarm/alert conditions are exceeded.

Traceability:

}

{REQ_D2.1.1_20108.A Node – parameters for SPD metrics
A nSHIELD node should continuously provide parameters that will be monitored by the nSHIELD overlay.

Traceability:

}

{REQ_D2.1.1_20109.A Node – low power
HW and SW implementation of a nSHIELD node should take into account power constraints.

Traceability:

}

{REQ_D2.1.1_20110.A Node – Security
The nSHIELD node shall be secure.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_20111.A Node – Cryptographic security
The nSHIELD node shall overcome cryptographic attacks.

Traceability: {REQ_D2.1.1_20110.A

}

{REQ_D2.1.1_20112.A Node – Attack metrics
The nSHIELD node should use attack metrics for measuring cryptographic insecurity, e.g. number of steps and time required for a successful attack.

Traceability: {REQ_D2.1.1_20111.A

}

{REQ_D2.1.1_20113.A Node – Security level
The nSHIELD node should be specified against which attacks security is desired.

Traceability: {REQ_D2.1.1_20110.A

}

{REQ_D2.1.1_20114.A Node – Security of complex system
The security of complex system shall be estimated from the security results on the whole system.

Traceability: {REQ_D2.1.1_20110.A

}

{REQ_D2.1.1_20115.A Node – Cryptographic key size metric
In the nSHIELD system should be used a metric based on the cryptographic key size: the Key-Size Metric as an Attack Metric.

Traceability: {REQ_D2.1.1_20111.A

}

{REQ_D2.1.1_20116.A Node – Attack Steps Metric
The nSHIELD system should use the Attack Steps Metric, i.e. the number of steps required to perform the best known attack.
Traceability: {REQ_D2.1.1_20112.A

}

{REQ_D2.1.1_20117.A Node – Attack Time Metric
The nSHIELD system should use the Attack Time Metric, i.e. the time required to perform the fastest known attack on a specified processor.
Traceability: {REQ_D2.1.1_20112.A

}

{REQ_D2.1.1_20118.A Node - Multi-dimensional metric (MMS) space
In the nSHIELD the Multi-dimensional metric (MMS) space for NMP-SPD nodes of a SPD-WSN should be composed of SPD (security, privacy, dependability) and basic functions (lifetime, coverage, cost and deployment, response time, temporal accuracy and effective sample rate).

Traceability:

}

{REQ_D2.1.1_20119.A Node - NMP-SPD nodes performance
In the nSHIELD the performance of networked NMP-SPD nodes should be evaluated on a SPD-WSN network by using pre-defined attack models tailored for the application-specific scenario.

Traceability:

}

{REQ_D2.1.1_20120.A Node - Self-powering
The nSHIELD node should be self-powered.

Traceability:

}

9.8.1 Dependability

{REQ_D2.1.1_20121.A Node – Hardware Reliability
The nSHIELD node should have reliable hardware.

Traceability: {REQ_D2.1.1_2018.A, {REQ_D2.1.1_2044.A

}

{REQ_D2.1.1_20122.A Node – Availability
The nSHIELD node shall be available for use.

Traceability: {REQ_D2.1.1_2044.A

}

{REQ_D2.1.1_20123.A Node – Availability & repair
The nSHIELD system should be an available and repairable system.

Traceability: {REQ_D2.1.1_2044.A

}

{REQ_D2.1.1_20124.A Node – Redundancy
The nSHIELD components should support redundancy.

Traceability: {REQ_D2.1.1_20121.A

}

9.9 Interfaces

{REQ_D2.1.1_2177.A Node – SCA protection based on EM emissions
An nSHIELD node should include interfaces to monitor its own EM emissions and modify its functionality accordingly, to protect its assets against SCA.

Traceability:

}

{REQ_D2.1.1_2178.A Node – Location awareness
An nSHIELD node should include the necessary interfaces that will enable location-based functionality.

Traceability:

}

{REQ_D2.1.1_2179.A Node – Situation and context awareness
An nSHIELD node should include the necessary interfaces that will enable the provision of situational-aware and context-aware services and SPD.

Traceability:

}

{REQ_D2.1.1_20125.A Node – Middleware Interface
A nSHIELD node should be capable to exchange data, measures as well as parameters with middleware, through a predefined interface.

Traceability:

}

{REQ_D2.1.1_20126.A Node – Remote Commands
A nSHIELD node should accept configuration commands, data, and equipment status form an external system.

Traceability:

}

{REQ_D2.1.1_20127.A Node – network interface
A nSHIELD node should be able to exchange data, measures as well as parameters with network level, through a predefined interface.

Traceability:

}

{REQ_D2.1.1_20128.A Node – overlay interface
A nSHIELD node should be able to send measurements and parameters, and receive commands from the nSHIELD overlay through a dependable interface.

Traceability:

}

9.10 Miscellaneous

{REQ_D2.1.1_2180.A Node – Accommodations for future energy sources
An nSHIELD node should have provisions for future alternative power sources including super-capacitors and wireless power schemes.

Traceability:

}

{REQ_D2.1.1_20129.A Node – self-reconfigurability
A nSHIELD node should provide a mechanism of self-reconfigurability to increase function density, increase security against side-channel attacks, and increase dependability implementing self-healing properties.

Traceability:

}

{REQ_D2.1.1_20130.A Node – TPM improvement of product endurance and lifespan
For a nSHIELD micro/personal node, the TPM platform should be extended in order to implement additional mechanisms to improve product endurance and increase product lifespan.

Traceability:

}

{REQ_D2.1.1_20131.A Node – TPM inexpensive implementation
For a nSHIELD micro/personal node, the TPM platform should be extended in order to have inexpensive implementation to allow widespread use.

Traceability:

}

{REQ_D2.1.1_20132.A Node – TPM Compliance with global export control
For a nSHIELD micro/personal node, the TPM platform must be extended to be compliant with global export control regulations in order not to restrict international trade with TC platforms (PCs)

Traceability:

}

{REQ_D2.1.1_20133.A Node - sub-SPD-WSN
One standalone SPD-WSN composed of SPD nano, micro/personal nodes and/or legacy nodes that have nSHIELD node adapter with SPD functionalities should represent the smallest possible nSHILED SPD network called sub-SPD-WSN.

Traceability:

}

{REQ_D2.1.1_20134.A Node - Legacy nodes
In the nSHIELD the SPD-WSN may contain one or more legacy nodes that are not using nSHIELD node adapter.

Traceability:

}

{REQ_D2.1.1_20135.A Node - Middleware technology
In the nSHIELD the Hydra middleware may be an excellent candidate middleware technology for SPD-NMP Nodes that compose a SPD-WSN.

Traceability:

}

{REQ_D2.1.1_20136.A SPD features
In the nSHIELD the networked SPD and legacy nodes of a WSN should be designed with SPD features on the circuit level, micro-architecture, and architecture, algorithm and protocol level.

Traceability:

}

{REQ_D2.1.1_20137.A Node - Power-supply circuits
In the nSHIELD the NMP-SPD nodes should have power-supply circuits with security and dependability features.

Traceability:

}

{REQ_D2.1.1_20138.A Node - NMP-SPD nodes
In the nSHIELD one NMP-SPD node should be composed of sensors (analog and digital), camera(s), and a TPM and/or SW-TPM unit.

Traceability:

}

{REQ_D2.1.1_20139.A Node - FPGA Power Node

In the nSHIELD the FPGAs should be used when a small number of SPD Power Node units are required due to its flexibility to incorporate the different components of a SPD Node, and reduced development costs.

Traceability:

}

{REQ_D2.1.1_20140.A Node - SPD Node technologies
In the nSHIELD the development of SPD Nodes with a high volume of produced units should use other technologies, such as ASICs, which reduce cost and increase dependability due to the increase in technology robustness.

Traceability:

}

{REQ_D2.1.1_20141.A Node - Certifiability
In the nSHIELD the third-party components (software and hardware) should be already certified, facilitating the process of certification of the whole SPD Node.

Traceability:

}

{REQ_D2.1.1_20142.A Node - High-capability nodes
In the nSHIELD a generic SPD Power Node with a wide range of capabilities and services should be designed and implemented in order to decrease the development time of the application, reduce costs, and facilitate certifiability.

Traceability:

}

{REQ_D2.1.1_20143.A Node - Integration of non-SPD compliant modules
In the nSHIELD the high-capability nodes should provide easy integration of non-SPD compliant modules.

Traceability:

}

10 Network Requirements and Specifications

10.1 Security

{REQ_D2.1.1_2181.A Confidentiality
The nSHIELD network shall support encryption of the packet data.

Traceability: REQ_D2.1_20045.A

Change record

}

{REQ_D2.1.1_2182.A Integrity
The nSHIELD network shall support authentication of the source of a transmitted packet.

Traceability: REQ_D2.1_20047.A, REQ_D2.1_20053.A

Change record

}

{REQ_D2.1.1_2183.A Availability
The nSHIELD network shall ensure the availability of the service.

Traceability: REQ_D2.1_20044.A, REQ_D2.1_20053.A

Change record

}

{REQ_D2.1.1_2184.A Source Authentication
The nSHIELD network shall support authentication of the source of a transmitted packet.

Traceability: REQ_D2.1_20048.A

Change record

}

{REQ_D2.1.1_2185.A Secure Routing
The nSHIELD network shall support secure routing of transmitted packets.

Traceability

Change record

}

{REQ_D2.1.1_2186.A Network Security Cryptographic Support
The nSHIELD network layer shall support both symmetric and asymmetric cryptography.

Traceability: REQ_D2.1_20049.A, REQ_D2.1_20050.A, REQ_D2.1_20051.A

}

{REQ_D2.1.1_2187.A Traceability
The nSHIELD network should provide traceability information on each transmitted packet.

Traceability

Change record

}

{REQ_D2.1.1_2188.A Audit
The nSHIELD network shall provide the required information to be used for auditing purposes.

Traceability: REQ_D2.1_06013.A, [Traceability]

Change record

}

{REQ_D2.1.1_2189.A Reputation-Based Secure Routing
The nSHIELD network should support reputation-based secure routing.

Traceability: REQ_D2.1_06013.A, [Traceability], [Secure Routing]

Change record

}

{REQ_D2.1.1_2190.A Reputation-Based Intrusion Detection
The nSHIELD network should support reputation-based intrusion detection schemes.

Traceability: REQ_D2.1_06013.A, [Traceability], [Secure Routing]

Change record

}

{REQ_D2.1.1_20144.A Network Security
The nSHIELD network layer shall be secure.

Traceability: {REQ_D2.1.1_2012.A

}

{REQ_D2.1.1_20145.A Network Security Protocols
The nSHIELD network layer should have suitable security protocols.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A
}

{REQ_D2.1.1_20146.A Network Security Attributes
The nSHIELD network layer should provide Availability, Confidentiality and Integrity.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20147.A Denial of Service Attack
The nSHIELD network layer shall support anti-replay protection to prevent against a denial of service attack.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20148.A Confidentiality
The nSHIELD network layer shall support data confidentiality to protect, and to encrypt the entire data.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20149.A Encryption Algorithms
The nSHIELD network layer shall support algorithms for encryption.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20150.A Data Integrity
The nSHIELD network layer shall support data integrity to ensure that the contents of the packet do not change in transit.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20151.A Authentication
The nSHIELD network layer shall support data authentication to verify that the packet received is actually from the claimed sender.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20152.A Efficient Security Algorithms
The nSHIELD network layer should support more efficient algorithms for fast and better speed of execution to satisfy the conditions in the case of limited processing and power capabilities of embedded devices.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20153.A Reputation based Secure Resource Management Procedures
The nSHIELD network layer should have reputation based Secure Resource Management Procedures at transmission level.

Traceability:

}

10.2 Dependability

{REQ_D2.1.1_2191.A Fault Tolerance
The nSHIELD network layer shall tolerate faults.

Traceability: REQ_D2.1_20029.A

}

{REQ_D2.1.1_2192.A Fault Recovery
The nSHIELD network layer shall be able to recover from faults.

Traceability: REQ_D2.1_20029.A

}

{REQ_D2.1.1_2193.A Application-Based Dependable Connectivity
The nSHIELD network layer shall provide dependable connectivity for a given application scenario.

Traceability: REQ_D2.1_20023.A, [Fault Tolerance], [Fault Recovery]

}

{REQ_D2.1.1_2194.A Dependable Authentic Key Distribution Mechanisms
The nSHIELD network layer shall provide dependable authentic key distribution mechanisms for the involved nodes.

Traceability:

}

{REQ_D2.1.1_2195.A Reliable Transmission Methodologies
The nSHIELD network layer shall provide waveform-agile and reliable transmission methodologies.

Traceability:

}

{REQ_D2.1.1_2196.A Self-Management and Self-Coordination
The nSHIELD network layer shall provide distributed self-management and self-coordination schemes for unmanaged and hybrid networks.

Traceability:

}

{REQ_D2.1.1_20154.A Dependability
The nSHIELD network layer should provide dependability mechanism.

Traceability: {REQ_D2.1.1_2012.A

}

{REQ_D2.1.1_20155.A Dependability Attributes
The nSHIELD network layer should provide Reliability, Availability, Safety, Maintainability, and Integrity.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20154.A

}

10.3 Privacy

{REQ_D2.1.1_2197.A Content Privacy
The nSHIELD network layer shall provide privacy on the transmitted information content.

Traceability: REQ_D2.1_20054.A

}

{REQ_D2.1.1_2198.A Anonymity
The nSHIELD network layer shall provide anonymity of the source.

Traceability:

}

{REQ_D2.1.1_2199.A Location Privacy
The nSHIELD network layer shall provide location privacy for the source.

Traceability:

}

10.4 Composability

{REQ_D2.1.1_21100.A Multiple Protocol Support
The nSHIELD network layer should provide support for a variety of protocols used by the heterogeneous system of composable, embedded devices.

Traceability: REQ_D2.1_20016.A, REQ_D2.1_20028.A

}

{REQ_D2.1.1_21101.A Application-Based Configurability
The nSHIELD network layer should/may support different configurations, according to the primary field of application.

Traceability: REQ_D2.1_20022.A, [Low network delay]

}

10.5 Performance/Metrics

{REQ_D2.1.1_21102.A Low Network Delay
The nSHIELD network shall feature low delay.

Traceability

Change record

}

{REQ_D2.1.1_21103.A Information Capacity
The nSHIELD network shall provide sufficient information capacity, for conveying information from/to the nodes and thus ensuring the good operation of the applications that exploit it.

Traceability

Change record

}

{REQ_D2.1.1_21104.A SPD Metrics
Appropriate metrics should be chosen for a given scenario.

Traceability: REQ_D2.1_20025.A

Change record

}

10.5.1 Security

{REQ_D2.1.1_20156.A Network – Measuring malicious network traffic
The nSHIELD system shall measure Malicious Network Traffic.

Traceability: {REQ_D2.1.1_2018.A

}

{REQ_D2.1.1_20157.A Network – Comparison of IDS solution
The nSHIELD system shall use metrics to compare different IDS solution.

Traceability:

}

10.5.2 Dependability

{REQ_D2.1.1_20158.A Network – Dependability Metrics for Network
The nSHIELD system shall use Dependability Metrics for Network.

Traceability: {REQ_D2.1.1_2045.A

}

{REQ_D2.1.1_20159.A Network – Resilience Metrics
The nSHIELD system should use Resilience Metrics that quantify the resilience of service-oriented networks under node and edge failures.

Traceability: {REQ_D2.1.1_20158.A

}

10.6 Interfaces

{REQ_D2.1.1_21105.A Secure Channel Establishment Interfaces
The nSHIELD network shall provide well-defined interfaces for establishing secure channels.

Traceability

Change record

}

{REQ_D2.1.1_21106.A Key Exchange Interfaces
The nSHIELD network shall provide well-defined interfaces for cryptographic key exchange.

Traceability

Change record

}

{REQ_D2.1.1_20160.A Network - Routing protocols
In the nSHIELD the routing protocols shall be resilient against compromised nodes that behave maliciously ensuring that sensed information stays within the sensor network and is accessible only to trusted parties is an essential step toward achieving security.

Traceability:

}

10.7 Miscellaneous

{REQ_D2.1.1_20161.A Network - Hybrid heterogeneous SPD-WSN
In the nSHIELD the SPD-WSN should be designed as a hybrid heterogeneous network if it is composed of NMP-SPD nodes and legacy nodes with or without nSHIELD node adapters with centralised and decentralised homogenous sub-SPD-WSNs or sub-WSNs.

Traceability:

}

{REQ_D2.1.1_20162.A Network - Features of SPD-WSNs
In the nSHIELD the SPD-WSNs should have code mobility, flexibility, node mobility, application knowledge, data fusion and QoS features.

Traceability:

}

11 Middleware Requirements and Specifications

11.1 Security

{REQ_D2.1.1_21107.A Secure Service Discovery
The nSHIELD middleware shall support standards for secure service discovery.

Traceability:

}

{REQ_D2.1.1_21108.A Secure Service Composability
The nSHIELD middleware shall support secure service composability.

Traceability:

}

{REQ_D2.1.1_21109.A Secure end-to-end communication
The nSHIELD middleware communication and data messages shall be secured

Traceability:

}

{REQ_D2.1.1_21110.A Access control
The nSHIELD middleware shall support the appropriate model to ensure identify and authorize the requests. Only authorized entities should have access to *nSHIELD* resources.

Traceability:

}

{REQ_D2.1.1_21111.A Registration
The nSHIELD middleware shall support entities registration before any entity is able to access the underlying infrastructure.

Traceability: {REQ_D2.1.1_21110.A

}

{REQ_D2.1.1_21112.A Secure Semantic Data and Models
The nSHIELD middleware shall ensure the validity of the semantic data used by the middleware.

Traceability**:**

}

{REQ_D2.1.1_21113.A Non Repudiation
The nSHIELD middleware shall provide the appropriate evidence for message origination. The nSHIELD entities should not be able to repudiate an action/message given the middleware's evidence.

Traceability:

}

{REQ_D2.1.1_21114.A Audit
The nSHIELD middleware should support the validity of the information services at a future, using the appropriate evidence.

Traceability:

}

{REQ_D2.1.1_21115.A Integrity
The nSHIELD middleware shall ensure the integrity of any kind of data used by the nSHIELD infrastructure.

Traceability:

}

{REQ_D2.1.1_21116.A Authenticity
The nSHIELD middleware shall ensure the origin of middleware entities

Traceability:

}

{REQ_D2.1.1_21117.A Availability
The nSHIELD middleware shall ensure middleware resources availability

Traceability:

}

{REQ_D2.1.1_21118.A Secure resource management
The nSHIELD middleware shall monitor and assure the availability of the appropriate resources. Only authorized entities shall have access to the nSHIELD resources.

Traceability: {REQ_D2.1.1_21110.A, {REQ_D2.1.1_21113.A

}

{REQ_D2.1.1_21119.A Policy attestation
The nSHIELD middleware should assert the properties of a security policy.

Traceability:

}

{REQ_D2.1.1_21120.A Secure communication with legacy systems
The nSHIELD middleware should secure the communication with legacy systems

Traceability:

}

{REQ_D2.1.1_1506.A Non-repudiation of origin for secure service discovery, composition and delivery protocols
The middleware components for service discovery, composition and delivery protocols SHOULD provide a method to ensure that a subject that receives information during a data exchange is provided with evidence of the origin of the information. This evidence can then be verified by either this subject or other subjects

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1507.A Non-repudiation of receipt for secure service discovery, composition and delivery protocols
The middleware components for service discovery, composition and delivery protocols SHOULD provide a method to ensure that a subject that transmits information during a data exchange is provided with evidence of receipt of the information. This evidence can then be verified by either this subject or other subjects.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_20163.A WS-Security
The nSHIELD middleware should support WS-Security.
Traceability:
}

{REQ_D2.1.1_20164.A Fault recovery
The nSHIELD middleware shall recover from faults.
Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A
}

{REQ_D2.1.1_20165.A Secure Resource Management Procedures
The nSHIELD should support Secure Resource Management Procedures at middleware level.

Traceability:

}

{REQ_D2.1.1_20166.A Secure service discovery, composition and delivery protocols
The nSHIELD middleware layer should support secure service discovery, composition and delivery protocols.

Traceability:

}

11.2 Dependability

{REQ_D2.1.1_21121.A Fault tolerance
The nSHIELD middleware shall tolerate faults.

Traceability:

}

{REQ_D2.1.1_1508.A Access Control Policies for middleware components
Middleware components SHALL have policies that identify sets of the following entities for access control functions: the subjects of access control, the objects of access control and the operations between the object and the subject covered by the policy.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}
{REQ_D2.1.1_1509.A Access Control Functions for middleware components
Middleware component(s) SHALL be implemented to provide the access control based on the Access Control Policies defined.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_20167.A Middleware – Discovery Functionality
The nSHIELD middleware should have discovery functionality in order to catalogue the capability available from nodes and network

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20168.A Middleware - Secure Offline Authentication with mobile devices
The nSHIELD middleware layer should allow secure Offline Authentication with mobile devices.

Traceability:

}

11.3 Privacy

{REQ_D2.1.1_21122.A Node Identity Protection
The nSHIELD middleware shall not reveal the identity of the nodes

Traceability:

}

{REQ_D2.1.1_21123.A Service Identity Protection
The nSHIELD middleware shall not reveal the identity of other services

Traceability:

}

{REQ_D2.1.1_21124.A Location privacy
The nhsield middleware shall protect nodes location

Traceability: {REQ_D2.1.1_21122.A

}

{REQ_D2.1.1_1510.A Anonymity
Middleware component(s) SHOULD have the capability to provide SPD services so that other users or subjects are unable to determine the identity of a user bound to a subject or operation.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1511.A Pseudonimity
Middleware component(s) SHOULD have the capability to provide SPD services so that a user may use a resource or service without disclosing its user identity, but can still be accountable for that use.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1512.A Unlinkability
Middleware component(s) SHOULD have the capability to provide SPD services so that a user may make multiple uses of resources or services without others being able to link these uses together.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}
{REQ_D2.1.1_1513.A Unobservability
Middleware component(s) SHOULD have the capability to provide SPD services so that a user may use a resource or service without others, especially third parties, being able to observe that the resource or service is being used.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1514.A Informed consent
Middleware component(s) SHOULD have the capability to provide SPD services so that users can know the outcomes and effects of all operations requested in advance.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_20169.A Policy-based SPD management
The nSHIELD middleware layer should support Policy-based SPD management.

Traceability:

}

11.4 Composability

{REQ_D2.1.1_1515.A Component authentication
Middleware component(s) SHALL implement and use authentication of other parts of the system (e.g. Nodes)

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection, [TA].IP2 – Architectural Dependability

}

11.5 Performance/Metrics

{REQ_D2.1.1_21125.A Middleware metrics
The nSHIELD middleware shall provide to the agents node's metrics

Traceability:

}

{REQ_D2.1.1_1516.A Middleware components integrity tests
Middleware component(s) SHOULD implement integrity self-test functionality and provide information on their integrity to other parties using the said middleware functionality. These components MAY collect such integrity information from other components (e.g. Nodes) and provide accumulated integrity data for the whole of the SPD services.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection [TA].IP2 – Architectural Dependability

}

{REQ_D2.1.1_20170.A Middleware – Metrics provisioning
The nSHIELD middleware should provide metrics collected from nodes and the network to the overlay.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

11.6 Interfaces

{REQ_D2.1.1_21126.A Communication with agents and nodes
The nSHIELD middleware shall provide the appropriate interfaces to communicate with agents and nodes

Traceability:

}

{REQ_D2.1.1_20171.A Middleware – Interface Capability
The nSHIELD middleware should be capable to interface different kinds of nodes to be specified.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20172.A Middleware – Internal Interfaces
The internal interfaces between the middleware and overlay modules should be defined.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20173.A Middleware – Generic interface
A nSHIELD Middleware should have the generic interfaces to integrate different types of SPD nodes and to retrieve and interpret various types of incoming data.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

11.7 Miscellaneous

{REQ_D2.1.1_20174.A Middleware – Interoperability
A nSHIELD Middleware should address the interoperability between different SPD technologies.

Traceability: {REQ_D2.1.1_2012.A, {REQ_D2.1.1_20144.A

}

{REQ_D2.1.1_20175.A Middleware – Lightweight
The nSHIELD middleware shall be lightweight enough to fit into severely constrained devices.

Traceability:

}

{REQ_D2.1.1_20176.A Middleware - software layer

In the nSHIELD the SPD-WSN should have middleware as software layer that lies between the operating system and the applications on each site of the system.

Traceability:

}

{REQ_D2.1.1_20177.A Middleware - Semantic web services and ontology
In the nSHIELD the SPD-WSN middleware should allow semantic web services and ontology with an overlay SPD Layer that is responsible for cross-layer SPD functionalities and QoS.

Traceability:

}

12 Overlay Requirements and Specifications

12.1 Security

{REQ_D2.1.1_21127.A Secure agent interaction
The federation obtained by the interaction of different nSHIELD security agents shall secure their communication.

Traceability:

}

{REQ_D2.1.1_21128.A Identity Validity
The nSHIELD overlay nodes shall validate all nSHIELD components identities.

Traceability: {REQ_D2.1.1_21127.A

}

{REQ_D2.1.1_21129.A Agent Authenticity
The nSHIELD overlay nodes shall validate the authenticity of other peers.

Traceability: {REQ_D2.1.1_21127.A

}

{REQ_D2.1.1_21130.A Overlay-data confidentiality
The nSHIELD overlay nodes shall protect the confidentiality of the data exchanged/stored by them.

Traceability: {REQ_D2.1.1_21127.A

}

{REQ_D2.1.1_21131.A Overlay data integrity
The nSHIELD overlay nodes shall protect the integrity of data exchanged/stored by them.

Traceability: {REQ_D2.1.1_21127.A, {REQ_D2.1.1_21128.A, {REQ_D2.1.1_21129.A

}

{REQ_D2.1.1_21132.A Overlay data authenticity
The nSHIELD overly nodes shall ensure the origin of the data exchanged among the overlay nodes.

Traceability: {REQ_D2.1.1_21128.A

}

{REQ_D2.1.1_21133.A Overlay – Policy-based security
The overlay nodes shall provide security services based on a pre-defined policy.

Traceability:

}

{REQ_D2.1.1_21134.A Audit Functionalities
The nSHIELD overlay nodes system shall audit the underlying nodes.

Traceability:

}

{REQ_D2.1.1_21135.A Agent Availability
The nSHIELD overlay agents shall ensure their availability

Traceability:

}

{REQ_D2.1.1_21136.A Intrusion Detection and Prevention
The nSHIELD overlay nodes shall detect and prevent denial of service attacks

Traceability: {REQ_D2.1.1_21134.A

}

{REQ_D2.1.1_21137.A Nodes Availability
The nSHIELD overlay nodes shall monitor underlying nodes' availability.

Traceability: {REQ_D2.1.1_21134.A

}

{REQ_D2.1.1_21138.A Monitoring nodes
The nSHIELD overlay nodes shall support continuous monitoring of the parameters that will be provided by the nSHIELD Nodes and Network.

Traceability:

}

{REQ_D2.1.1_21139.A Access control functionalities
The nSHIELD overlay nodes shall control the access to the nSHIELD infrastructure.

Traceability: {REQ_D2.1.1_21135.A

}

{REQ_D2.1.1_21140.A Overlay Nodes – Information/data management
The Information/Data-Management shall store and provide to the adequate module the knowledge collected by the middleware and used by the SPD Security Agents.

Traceability:

}

{REQ_D2.1.1_21141.A Non-repudiation functionalities
The nSHIELD overlay nodes shall provide non-repudiation functionalities.

Traceability:

}

{REQ_D2.1.1_21142.A Anomaly detection mechanism and trust information
If an attacker could compromise an agent or its communication link with the multi-agent system, he would be able to affect the operation of the whole system. Thus, mechanisms for anomaly detection should be considered. Trust information could also be used to continuously rank the agent's behavior.

Traceability: {REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_1517.A Non-repudiation of origin for communication protocols of security agents

The security agents SHOULD provide a method to ensure that a subject that receives information during a data exchange is provided with evidence of the origin of the information. This evidence can then be verified by either this subject or other subjects

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1518.A Non-repudiation of receipt for communication protocols of security agents

The security agents SHOULD provide a method to ensure that a subject that transmits information during a data exchange is provided with evidence of receipt of the information. This evidence can then be verified by either this subject or other subjects.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies, [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_20178.A Overlay – Security Agent

An nSHIELD Security Agent shall realize one or more Overlay functionalities for the sub-system under its responsibility.

Traceability:

}

{REQ_D2.1.1_20179.A Overlay – Security Agents communication

A nSHIELD Security Agent shall communicate with other Security Agents that are responsible of other sub-systems.

Traceability:

}

{REQ_D2.1.1_20180.A Semantic representation of the security knowledge domain

The nSHIELD overlay layer should have Semantic representation of the security knowledge domain.

Traceability:

}

{REQ_D2.1.1_20181.A Key *composability* concept

In the nSHIELD overlay security agents should implement the key *composability* concept.

Traceability:

}

{REQ_D2.1.1_20182.A Monitoring of measurements

In the nSHIELD each security agent should monitor a set of properly selected measurements and parameters taken at any of the three layers: Node, Network, and Middleware layer.

Traceability:

}

{REQ_D2.1.1_20183.A Aggregation of available metadata
In the nSHIELD each security agent should aggregate the available metadata (the ones relevant to monitored measurements and parameters, as well as the ones coming from other security agents), in order to deduce aggregated metadata which form the so-called *dynamic context.*

Traceability:

}

{REQ_D2.1.1_20184.A Continuous monitoring
The SPD metrics in the nSHIELD should be continuously monitored by the security agents.

Traceability:

}

{REQ_D2.1.1_20185.A Static and dynamic composability
In the nSHIELD in the case of failure, the security agent should react discovering, composing and configuring the available SPD modules.

Traceability:

}

12.2 Dependability

{REQ_D2.1.1_21143.A Overlay agents fault tolerance
The nSHIELD overlay agents shall tolerate faults.

Traceability:

}

{REQ_D2.1.1_21144.A Overlay agents Redundancy
The nSHIELD overlay agents shall provide redundant services.

Traceability: {REQ_D2.1.1_21143.A

}

{REQ_D2.1.1_20186.A Semantic representation of the dependability knowledge domain
The nSHIELD overlay layer should have Semantic representation of the dependability knowledge domain.

Traceability:

}

{REQ_D2.1.1_21145.A Conflict resolution mechanism's fault tolerance
The conflict resolution mechanism should be fault tolerant.

Traceability: {REQ_D2.1.1_21154.A}

}

12.3 Privacy

{REQ_D2.1.1_21146.A Overlay agents privacy identity
The nSHIELD overlay agents shall protect the nodes identity

Traceability:

}

{REQ_D2.1.1_21147.A Access Policies for the distributed knowledge
Access policies should be implemented for the distributed knowledge like: everyone, groups of agents or list of agents.

Traceability:{REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_21148.A Locally protected knowledge
Agents should be able to locally protect sensitive data and not distribute them to the other agents. Like specific inner security information and user's personal data and preferences.

Traceability:{REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_20187.A Semantic representation of the privacy knowledge domain
The nSHIELD overlay layer should have Semantic representation of the privacy knowledge domain.

Traceability:

}

{REQ_D2.1.1_20188.A Situational-aware and Context-aware SPD
The nSHIELD overlay layer should have Situational-aware and Context-aware SPD.

Traceability:

}

12.4 Composability

{REQ_D2.1.1_21149.A Overlay agents secure service composability
The nSHIELD overlay agents shall compose secure services

Traceability:

}

{REQ_D2.1.1_21150.A Multi-agent system (MAS)
As agents in nSHIELD must interact with each other, a multi-agent system (MAS) should be implemented.

Traceability:

}

{REQ_D2.1.1_21151.A Full Communication
The proposed MASs should support full communication among all agents.

Traceability:{REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_21152.A Knowledge distribution topologies
Central and distributed topologies should be considered for the knowledge distribution of the multi-agent system, according to the examined scenarios.

Traceability:{REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_21153.A Agents' cooperation
For simplicity, the agents should be honest, perform absolute cooperation with all other agents and no agent should possess secret goals and aspirations. Game theory issues should not be considered in nSHIELD.

Traceability:

}

{REQ_D2.1.1_21154.A Conflict resolution mechanism
As agents publish their local knowledge, conflicts may occur. A conflict resolution mechanism should be implemented which will monitor the distributed knowledge, detect and resolve conflicts, and enforce the solution to the multi-agent system.

Traceability:

}

{REQ_D2.1.1_21155.A Conflict resolution mechanism's implementation
The implementation of the conflict resolution mechanism should be based on the policies mechanisms that already exist in the middleware and overlay, for simplicity.

Traceability: {REQ_D2.1.1_21154.A}

}

{REQ_D2.1.1_21156.A Conflict resolution mechanism and agent's locally protected knowledge
In order to participate in the conflict resolution process, agents should be able to prove their local knowledge based on their inner data. The conflict resolution mechanism should consider the case where one or more agents have made use of locally protected knowledge and can't fully justify their knowledge during this process.

Traceability: {REQ_D2.1.1_21154.A}

}

{REQ_D2.1.1_21157.A Agent Communication Language
The ACL (Agent Communication Language) should be used as a standard language for agent communication.

Traceability:

}

{REQ_D2.1.1_21158.A Agent-oriented tools
For the analysis and design process, agent-oriented tools like AUML (Agent UML), O-MaSE (Organization-based Multiagent System Engineering) and aT3 (agentToll III) should be used.AUML is the most suitable tool for agent development.O-MaSE and At3 can help the developers to analyze, design and implement multi-agent systems.

Traceability: {REQ_D2.1.1_21150.A}

}

{REQ_D2.1.1_20189.A Static and dynamic composability by nSHIELD Overlay

Modules in nSHIELD belonging to different SPD layers (node, network or middleware) should be composed statically or dynamically by the nSHIELD overlay.

Traceability:

}

12.5 Performance/Metrics

{REQ_D2.1.1_1519.A Overlay Integrity tests
Overlay security agents SHOULD implement integrity self-test functionality and provide information on their integrity to middleware functionality interacting with these agents.

Traceability: [TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection, [TA].IP2 – Architectural Dependability, {REQ_D2.1.1_1509.A

}

12.6 Interfaces

{REQ_D2.1.1_21159.A Communication with overlay nodes
The nSHIELD overlay agents shall provide communication interfaces with other overlay nodes

Traceability:

}

{REQ_D2.1.1_1520.A Checking non-repudiation of origin on interfaces of overlay security agents
Security agents SHOULD ensure that information received during a data exchange is provided with evidence of the origin, and SHOULD verify this evidence.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies [TA]. SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_1521.A Non-repudiation of receipt for secure service discovery, composition and delivery protocols
Security agents SHOULD provide a method to ensure that a subject that transmits information during a data exchange is provided with evidence of receipt of the information. This evidence can then be verified by either this subject or other subjects.

Traceability: [TA].2.1.3.9 nSHIELD improved technologies

[TA].SP6 – Inter-networked ES for Security and Critical Infrastructures Protection

}

{REQ_D2.1.1_20190.A Overlay – Overlay interfaces
The nSHIELD Overlay should support two types of interfaces: external interfaces (to communicate with nSHIELD Node and Network Layers) and internal interfaces (to communicate with nSHIELD Middleware).

Traceability:

}

{REQ_D2.1.1_20191.A Overlay – Internal interfaces
The nSHIELD Overlay shall communicate and share information with the middleware via internal interfaces. These interfaces support also the information exchange between Security Agents.

Traceability:

}

{REQ_D2.1.1_20192.A Overlay – Overlay/Node external interfaces
The nSHIELD Overlay shall be able to receive measurements and parameters, and send commands to the nSHIELD Nodes through external interfaces. This interface may be the nSHIELD Middleware itself.

Traceability:

}

{REQ_D2.1.1_20193.A Overlay – Overlay/Network external interfaces
The nSHIELD Overlay shall be able to receive measurements and parameters, and send commands to the nSHIELD Network through external interface. This interface may be the nSHIELD Middleware itself.

Traceability:

}

12.7 Miscellaneous

{REQ_D2.1.1_20194.A Overlay – Input: desired SPD and policies
The nSHIELD Overlay shall be able to receive as input, from the application layer, or the end-user, the desired level of SPD and/or the SPD Policies.

Traceability:

}

{REQ_D2.1.1_20195.A Overlay – Main Functionalities
The nSHIELD Overlay shall realize four main functionalities: discovery, control, composition and configuration. These functionalities could be either strictly separated or partially merged.

Traceability:

}

{REQ_D2.1.1_20196.A Overlay – Functionality: measurement and quantification
The nSHIELD Overlay shall be able to measure and quantify the SPD level of each nSHIELD component as well as the SPD level of the overall system.

Traceability:

}

{REQ_D2.1.1_20197.A Overlay – Functionality: comparison
The nSHIELD Overlay shall be able to compare the desired SPD level with the effective SPD level.

Traceability:

}

13 References

[1] A. Avizienis, J.-C. Laprie, B. Randell, C. Landwehr, Basic concepts and taxonomy of dependable and secure computing, Transactions on Dependable and Secure Computing 1 (1) (2004) 11–33. January–March, ISSN 1545-5971.

[2] C.E. Landwher, A.R. Bull, J.P. McDermott, and W.S. Choi, "A Taxonomy of Computer Program Security Flaws," ACM Computing Survey, vol. 26, no. 3, pp. 211-254, 1994.

[3] www.w3.org/XML/

[4] www.w3.org/TR/soap/

[5] www.oasis-open.org/

[6] Common Criteria for Information Technology Security Evaluation - Part 2: Security functional components.

[7] David A. Wheeler, Secure Programming for Linux and Unix HOWTO, v3.010, 3 March 2003 - http://tldp.org/HOWTO/Secure-Programs-HOWTO/x641.html

[8] V. Drakulovski et al "System Requirements and Specification for the pSHIELD-project", http://www.pshield.eu/index.php?option=com_docman&task=doc_download&gid=238&Itemid=37

[9] M. Al-Kuwaiti, N. Kyriakopoulos, S. Hussein, A Comparative Analysis of Network Dependability, Fault-tolerance, Reliability, Security, and Survivability, IEEE Communications Surveys & Tutorials, Vol. 11, No. 2, Second Quarter 2009.

[10] James McCabe, Practical Computer Network Analysis and Design, Morgan Kaufmann Publishers, Inc., CA, 1998, pp. 1-9.

[11] pSHIELD project, Deliverables D2.1.1, "Preliminary SPD Metrics Specification for the pSHIELD project," September 2011.

[12] A. Avizienis, J.-C. Laprie, and B. Randell, "Fundamental concepts of dependability", Research Report No. 1145, LAAS-CNRS, Apr. 2001.

[13] B. Melhart, and S. White, "Issues in defining, analyzing, refining, and specifying system dependability requirements", *Proc. 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2000)*, Edinburgh, Scotland, UK., Apr. 3-7, 2000, pp. 334-311.

[14] P. Neumann, *Practical architectures for survivable systems and networks*, Technical report, Final Report, Phase Two, Project 1688, SRI International, Menlo Park, California, Jun. 2000.

[15] J. Park, and P. Chandramohan, "Static vs. dynamic recovery models for survivable distributed systems", *Proc. 37th Annual Hawaii International Conference on System Science*, Maui, HI, 5-8 Jan. 2004, IEEE Comp. Soc. Press, 2004, pp. 55-63.

[16] R. Ellison, D. Fischer, R. Linger, H. Lipson, T. Longstaff, and N. Mead, *Survivable network systems: an emerging discipline*, (Report CMU/SEI-2001-TN-001), Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, Mar. 2001.

[17] Hu J, et al. seamless integration of dependability and security concepts in SOA: A feedback control system based framework and taxonomy. J Network Comput Appl (2011), doi:10.1016/j.jnca.2010.11.013

[18] Jonsson E. An integrated framework for security and dependability. Proceedings of the 1998 workshop on new security paradigms. ACM Press; 1998