Project no: 269317

nSHIELD

new embedded Systems arcHItecturE for multi-Layer Dependable solutions

Instrument type: Collaborative Project, JTI-CP-ARTEMIS

Priority name: Embedded Systems

D3.1: SPD node technologies assessment

Due date of deliverable: M4 –2011.12.30

Actual submission date: M8 – 2012.04.31

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

Organisation name of lead contractor for this deliverable:

IPS Sistemi Programmabili, Eurotech Group, ETH

Revision [Final v1.0]

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**		
Paolo Azzoni	ETH		
Stefano Gosetti	ETH		
George Dramitinos	ISD		
Carlo Pompili	TELC		
Bharath Siva Kumar	TELC		
Christian Gehrmann	SICS		
Oliver Schwarz	SICS		
Mudassar Aslam	SICS		
Hans Thorsen	T2D		
Paolo Gastaldo	UNIGE		
Alessio Leoncini	UNIGE		
Chiara Peretti	UNIGE		
Daniele Caviglia	UNIGE		
Daniele Grosso	UNIGE		
Luca Noli	UNIGE		
Iñaki Eguia	TECNALIA		
Eider Iturbe	TECNALIA		
Harry Manifavas	TUC		
Alexandros Papanikolaou	TUC		
Konstantinos Fysarakis	TUC		
Georgios Hatzivasilis	TUC		
Dimitrios Geneiatakis	TUC		
Konstantinos Rantos	TUC		
Lorena de Celis	AT		
David Abia	AT		
Antonio Abramo	UNIUD		
Mirko Loghi	UNIUD		
Antonio di Marzo	SESM		
Antonio Bruscino	SESM		
Kyriakos Stefanidis	ATHENA RC		
Spase Drakul	THYIA		
Gordana Mijić	THYIA		
Ljiljana Mijić	THYIA		
Nastja Kuzmin	THYIA		
Balazs Berkes	S-LAB		
Francesco Cennamo	SG		
Luigi Trono	SG		

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	01/03/2012	First ToC
Draft B	20/03/2012	First partners contribution
Draft B v.0.2	22/03/2012	Partners contribution
Draft B v.0.3	14/04/2012	Further contribution and sections update
Prefinal v.0.4	16/04/2012	New partner contributions and integration
Final v0.5	24/04/2012	Sections update
Final v0.6	27/04/2012	Update sections: 2, 3.4.1, 3.4.2. 5
Final v0.7	28/04/2012	Update sections: 2, 3, 6
Final v0.8	29/04/2012	Update sections: 1, 3
Final v1.0	31/04/2012	Final review

Contents

Figures

Tables

Glossary

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

This Page is intentionally left blank

1 Introduction

nSHIELD project proposes a layered architecture to provide intrinsic SPD features and functionalities to embedded systems. In this layered architecture workpackage 3 is responsible for the node layer that represents the lower level of the architecture, a basement constituted of real embedded devices on which the entire project will grow. Some SPD technologies that will be introduced with this workpackage are the extension and evolution of the results obtained with pSHIELD project while, in most cases, start to be investigated in nSHIELD itself.

As already outlined in the TA, workpackage 3 aims to create an Intelligent ES HW/SW Platform that consists of three different kinds of Intelligent ES Nodes: nano node, micro/personal node and power node. These three categories of embedded systems will represent the basic components of the lower part of an SPD Pervasive System that will cover the possible requirements of several market areas: from field data acquisition, to transportation, to personal space, to home environment, to public infrastructures, etc.

This deliverable is focused on the assessment of the SPD technologies that will be studied and developed in work package 3 and that are functional to the nSHIELD SPD applications (Workpackage 7). These technologies will be implemented in the prototypes that will be developed in workpackages 6 and 7 and that will be part of the nSHIELD scenarios demonstrators.

1.1 The technology assessment

The objective of the SPD node (SPDN) technology assessment (TA) is to verify if the SPDN technologies addressed in the nSHIELD project and constrained by the requirements and specifications in D2.2 are fulfilling the needs for the selected application scenarios, described in details in WP7, that drive the identification of the intelligent Embedded System (ES) nodes, which will be developed for the demonstrator. This technology assessment for SPDNs will provide guidelines for the SPDN design as input for other WP3 deliverables as well as for D2.6 that will consider additional requirements proposed in D3.1.

The SPDN technology assessment will validate the nSHIELD node technologies by considering

- Compliance with the WP2 requirements and specifications documents
- Maturity and availability of the SPDN technologies used for platform integration, validation and demonstration and later on for the application demonstrators.

This document will report the current available SPDN technologies, HW and SW module able to implement the SPD functionalities required for each approved technology. Advancements beyond the current State of Art (SoA) will be addressed by identifying the technology gaps that required new solutions for achieving the nSHIELD objectives. These technology gaps will help to identify the risks and their impacts on the design phase.

Conclusions are confirming that the accessed SPDN technologies satisfy the nSHIELD requirements and specifications and that no constraints are imposed on the design of SPDNs.

1.2 Nodes definitions

nSHIELD SPD architecture follows very closely the architecture introduced in pSHIELD project and is based on four layers:

- Node Layer,
- Network Layer,
- Middleware Layer,
- Overlay Layer.

The node layer represents the basement of the nSHIELD SPD Pervasive System and provides the basic components in terms of ES. The node layer consists of three different categories of Embedded Nodes which can be considered three node levels of increasing complexity, features availability and computational power:

- nano Node,
- micro/Personal Node,
- power Node.

The definition of these three categories follows the one introduced in pSHIELD.

Nano Node level typically consists of small and resource limited devices, both in terms of hardware and software (i.e. wireless sensors). Because of their massive distribution in the environment, nano nodes could become an interesting target for attacks and hacking.

Micro/Personal Node level consists of devices richer than the Nano Nodes, in terms of hardware and software resources, network access capabilities, mobility, interfaces, sensing capabilities etc. The specific functions of a Micro/Personal Node are generally referred to:

- secure network access capabilities,
- monitoring and sensing,
- interfacing.

Power Node level represents the first level of massive data elaboration of the SPD pervasive system, with the peculiarity that the computing power is provided directly on the field.

The classification of the nodes in three different classes is provided with more details in the following table. The table specifies the node hardware, software, networking capability, mobility, interfaces and sensing capabilities that are used as criteria for the classification of the nodes.

Table 1 – Characteristics of pSHIELD nodes

	nano	**micro**	**personal**	**power**
hardware	restricted		extendable, personal/mobile board,	typical data center, server board
software	not changeable (sense and send)	process (collect and store)	decision making, (java)	ontologies, reasoning
network access	through gateway		direct wireless/mobile	fixed (backend)
mobility	-		yes	no
interfaces	fixed or wireless		network and usb-like	fixed
sensing	one or more sensors, e.g. light, temperature, position		inbuilt sensors	no

Some examples of nodes that can be classified in the identified categories are:

- Power node: datacenter for ontologies and reasoning;
- Personal node: mobile phone, embedded Linux system;
- Micro-node: SunSpot;
- Nano: GPS unit, wireless sensor mote.

1.3 Document contents

As already mentioned, this deliverable describes the SPD technologies required by the application scenarios that will be developed in WP7. This document focuses on the technologies located at node level, providing both “vertical” technologies that are applicable to a specific class of nodes and “horizontal” technologies that can adopted for all the categories of nodes considered in nSHIELD. The document is structured in the following sections:

1. Introduction: a brief introduction related to the SPD node technology assessment.
2. SDR/Cognitive Enabled node: assessment of SDR/Cognitive Enabled Node (CEN) technologies for generic application scenarios, providing the definition of the cognitive features of a node and introducing a platform for the development of these functionalities.
3. Micro/Personal node: this section introduces the technologies required by scenarios 2 (Voice/Facial Recognition) and 4 (Social Mobility) at node level. It focuses on four main technological areas: intrinsically trusted embedded systems, smartcards for security, SPD and power consumption and biometric algorithms for SPD.
4. Power node: this section describes the technologies that will be adopted in the areas of surveillance, system of embedded systems and SPD for avionics. These technologies will be adopted in scenarios 1 (Railways security), 3 (Dependable Avionic Systems) and 4 (Social Mobility).
5. Dependable self-x Technologies: this section introduces horizontal SPD technologies that will be adopted in task 3.1-3.2-3.3 at different levels, depending on the complexity of the node and considering its HW/SW capabilities, its requirements and its usage. The technologies are focused on the following areas: automatic access control, denial-of-services, self-configuration, self-recovery and quality of service.
6. Cryptographic technologies: this section provides the assessment of horizontal SPD technologies focused specifically on hardware and software cryptography, on the use of crypto technologies to implement SPD embedded devices and prevent physical attacks at this level using defense crypto-based solutions.
7. SPD Node independent technologies: the final section describes a set of SPD technologies that are node independent and is focused on authorization frameworks for SPD and on secure execution environments/trusted virtual domains.

2 SDR/Cognitive Enabled node

The methodology that will be adopted for technology assessment, as described in section 1.1, is used for the current assessment of the micro node level and takes into account two main field of analysis:

1. Functional capabilities of the examined technologies as **SDR/Cognitive Enabled Node (CEN) system model evidence**: the functionalities defined for the SPDNs that are driven by the nSHIELD application scenarios require a restricted range of all requirements for CENs.
2. **Fulfilment of the CEN design**: while the requirements are the evidence that the chosen CEN technologies are suitable for it system design, they do not provide any proof regarding their readiness for the final design. This fulfilment of the CEN design model must be part of the assessment methodology as a step further in order to validate the readiness of the HW and SW modules necessary for the CEN design.

Thus, the CEN system design model as input uses preliminary requirements and specification that are developed in Task 2.1, and as output it provides evidence of technology compliance to the WP2 requirements. Fulfilment of the CEN system model use as inputs HW and SW module analysis to fulfil the CEN design, and as output it provide important issues to be considered, risks, recommendation and new requirements.

Here we are describing in details SDR/**Cognitive Enabled Node** (CEN) technology assessment with emphasis on the new SPD functionalities tailored for a generic application scenario. The primary goal is to define 1) **cognitive features** of such node as an open HW/SW CEN platform and 2) to use this platform for **development of SPD functionalities** for threats (faults, errors, and failures), attributes (confidentiality, integrity, authenticity, availability, reliability, etc.) and means (fault tolerance, fault prevention, fault removal, etc.). Selection of a set of threats attributes and means represent a key design target goals for the selected SPD functionalities for CENs. The secondary goal of this section is to provide some **additional set of requirements and specifications that are specific for CENs**. This means on the end of each subsection (if any) should be clearly captured the new requirements that will be used for the prototype developments in WP3, WP4, WP5, WP6 and WP7.

2.1 SDR/cognitive technology foundation

The research and development (R&D) work related to **Cognitive Enabled Nodes** (CENs) will be based on the SDR/Cognitive radio concepts, requirements and specifications (R&S) provided by Task 2.1 (T2.1) and WP4 that focus on the smart transmission layer which rely on waveform-agile implementation of SDR platform. However, the research results obtained in Task 3.1 (T3.1) are not limited only to CENs, and from a technological point of view will be adopted when needed also for the design and development of **Micro/Personal Nodes** (MPNs). With respect to the current and future capabilities of Micro and Nano-Technologies (MNTs) that represent a foundation for sensing, communication and processing capabilities of CENs & MPNs (or CMPNs) it is clear that border line between CENs and MPNs will change dynamically in the direction from MPNs toward CENs in the years ahead. This means if the current MNT limitations of the CENs are not allowing the implementation of desired SPD functionalities for MPNs or Power Nodes (PNs), or more general **nSHIELD SPD Nodes** (SPDNs) it doesn't means that tomorrow with advancement in MNTs it will not be possible to implement all SPD functionalities for SPDNs. Therefore, the aim of R&D work in T3.1 is twofold:

1. To provide an assessment of the future CEN technology advancements for achieving SPD functionalities of MPNs or even SPD Nodes.

2. To develop a CEN system model for the nSHIELD system with the current capabilities of MNTs.

The complexity and performance increased from CENs (low) towards MPNs (medium) and PNs (high). First, we will focus our assessment work to some essential technologies for SDR/Cognitive Nodes to accomplish the first aim above and to provide a solid foundation for the future developments. Second, starting from this assessment we will analyse fulfilment of the available technologies for CEN design.

2.2 Micronode and nSHIELD node definition

An nSHIELD SPD Node is an Embedded System Device (ESD). When a Legacy ESD equipped with several legacy node capabilities will be used in the nSHIELD network it requires an **nSHIELD Node Adapter** (nSNA). The nSHIELD node is deployed as a hardware/software platform, encompassing intrinsic, innovative SPD functionalities, providing proper services to the other nSHIELD networks and adapters to enable the nSHIELD composability and consequently the desired system SPD. There are three kinds of nSHIELD node deploying each different configuration of Node Layer SPD functionalities of the nSHIELD framework, and comprising different types of complexity:

1. SPD/Cognitive Enabled Nodes (CENs),
2. Micro/Personal Nodes (MPNs) and
3. Power Nodes (PNs).

All of them have a common name SPD Nodes (SPDNs) that are used in the nSHIELD SPD Network.

The technological advancements in computing hardware and software enables a new generation of small ESDs to perform complex computing tasks. Extremely small sensor devices provide advanced sensing and networking capabilities. In parallel, many operating systems targeting these types of devices have been developed to increase their performance. The method for designing nSHIELD SPDNs is twofold:

1. To design completely **new SPD nodes** that are **complaint with the nSHIELD system** design.
2. To keep legacy node technologies as they are compliant with their standards, developed for many applications including those that are targeted in nSHIELD, which means to assume a heterogeneous infrastructure of networked ESDs like IEEE 802.15.4, IEEE 802.11, etc. An ordinary sensor technology (not all, since we need those that are designed for ES) permits to consider an augmentation of SPD functionalities at different levels of the hardware and firmware modules. This means an enhanced **legacy SPD node (LSPDN)** with physical layer and protocol stack composed of existing and new SPD technologies added by nSNA. As result of this integration a new types of networked SPD ESDs will be created. nSHIELD and new SPD ESDs will compose a heterogeneous SPD network infrastructure too.

Developing a SPDN as integrated node of a LSPDN and nSNA we obtain a composable nSHIELD Node. It means that it has all of the desired SPD functionalities and services for the nSHIELD application scenario selected. Additionally to that, the nSHIELD Node keeps some of the desired functionalities of a standardised sensor technology with additional SPD features that make it composable into the nSHIELD framework. The architectural design of the nSHIELD Nodes will relay on the ISO/IEC 9126 standard that has 6 top level characteristics: functionality, reliability, usability, efficiency, maintainability and portability.

2.3 SPD Wireless Sensor Networks

The nSHIELD network architecture is a **homogenous network** (as in Figure 2.2 of the Technical Annex) for the selected application scenarios with a concept of four functional layers with SPD functionalities and SPD core services. By introducing more implicational scenarios and Legacy ES nodes and Legacy ES Networks, the final architecture becomes a **hybrid heterogeneous network** (HHN). It is heterogeneous in the sense of coexistence different technologies (IEEE 802.15.4, IEEE 802.11, UMTS, etc., multi-frequency, multi- technology, multi-layer, multi-architecture) that are connected with unified control and optimisation, and it is hybrid in the sense of a network that is between a centralised and pure decentralised architecture.

Figure 1 illustrates a WSN composed of CENs (nano nodes), Micro/Personal and Power Node which can be used also as a Gateway.

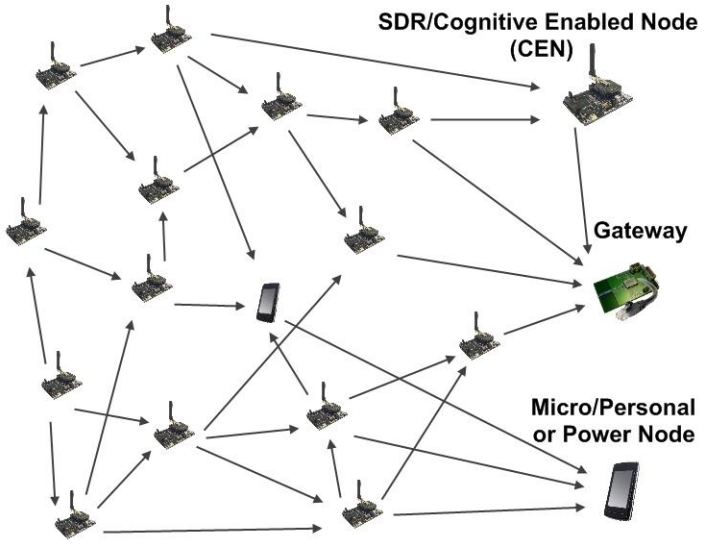

Figure 1 - WSN composed of CENs MPNs and PNs.

We can see that a homogenous or heterogeneous SPD-WSN is a subset of a homogenous or heterogeneous nSHIELD SPD network. Therefore, the smallest and simplest (in complexity) nSHIELD SPD networks is a SPD-WSN composed of wireless sensor CENs (SCENs).

2.4 The CEN system description

2.4.1 Pervasive Systems

Pervasive Computing (PC) also called Ubiquitous Computing (UC) or together Ubiquitous and Pervasive Computing (UPC) is maturing from its origins as an academic research area to a commercial reality. In ubiquitous or pervasive ambient environment, simple and complex services are provided to users, according to their contexts, at anytime, anywhere, and using any available device. Dynamic composition of services for such environment plays an important role, because it composition aims to provide a variety of high level services[1]. Variety of PC nodes and concepts are proposed to accomplish with the UPC requirements.

A key aspect of pervasive computing involves embedding sensing, networking and computation (SNC) into everyday objects and everyday life processes. UPC is the trend towards increasingly ubiquitous connected Embedded Devices (EDs) in the environment. It is a trend about a convergence of advanced electronic, wireless technologies and the Internet. UPC devices are not PCs but very tiny and invisible EDs, either mobile or embedded in almost any type of object imaginable, including cars, tools, appliances, clothing and various consumer goods that are communicating through increasingly interconnected networks.

Among the emerging technologies expected to prevail in the UPC environment of the future are wearable computers, smart homes and smart buildings. The tools expected to support these are: application-specific integrated circuitry (ASIC), speech and gesture recognition, perceptive interfaces; smart matter, field programmable gate area (FPGA), system on a chip (SoC), and micro electromechanical systems (MEMS).

UPC requires a middleware to interface between the networking kernel and the end-user applications running on UPC devices. This UPC middleware will mediate interactions with the networking kernel on the user's behalf and will keep users immersed in the pervasive computing space. The middleware will

[1] K. Tari et al. ," Context-aware Dynamic Service Composition in Ubiquitous Environment," IEEE ICC 2010 proceedings.

consist mostly of firmware and software bundles executing in either client-server or (peer-to-peer) P2P mode. User interfaces are another aspect of middleware.

The nSHIELD system architecture based on the four functional layers is conceptually designed for the development of software components that are reusable across the pervasive computing applications. To achieve this is important to consider the variations and properties like mobility, adaptability, composability, and context awareness that may be required for different nSHIELD applications. However, that various requirements and variations may not always be known a priori and hence developing all the multiple variants may not always be possible or feasible. The term “composability” is widely used in nSHIELD, but for UPC is a property of a software component meaning that it may easily and systematically be combined with other components. Composability of software components in UPC is an important issue and has been given little attention.

2.4.2 SDR/Cognitive functionalities for CEN systems

Understanding the fundamental functionalities of SDR and Cognitive Systems is essential for the SCENs that have SNC capabilities desired. A SCEN has sensing capabilities (it contain at least a sensor, networking capabilities, it represent a node in a network and computational or processing capabilities, it can performs some SDR or cognitive features additionally to some SPD functionalities required for this type of node. Therefore, the first step toward a description of SCENs is to define some key properties of SDR/Cognitive node as a networked node in nSHIELD SPD network that is tailored for a specific application scenario.

2.4.2.1 SDR definitions

First of all, it is useful to review the design of a conventional SDR. Figure 2 shows a block diagram of a generic digital radio, which consists of five sections:

- The **antenna** section, which receives (or transmits) information encoded in radio waves.
- The **RF front-end** section, which is responsible for transmitting/receiving radio frequency signals from the antenna and converting them to an intermediate frequency (IF).
- The **ADC/DAC** section, which performs analog-to-digital/digital-to-analog conversion.
- The **digital up-conversion** (DUC) and **digital down-conversion** (DDC) blocks, which essentially perform modulations of the signal on the transmitting path and demodulation of the signal on the receiving path.
- The **baseband** section, which performs operations such as connection setup, equalization, frequency hopping, coding/decoding, and correlation, while also implementing the link layer protocol.

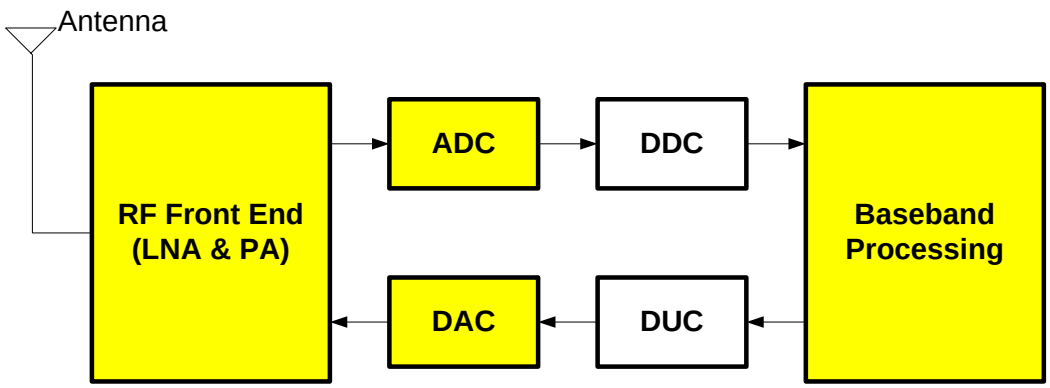

Figure 2 - Schematic block diagram of a digital radio.

Embedded SDR system solution

Waveform processing can be performed on four different types of hardware platforms and configurations:

- General Purpose Processor (GPP)
- General Purpose Processor (GPP) + Digital Signal Processor (DSP)
- Field Programmable Gate Array (FPGA)
- Application Specific Integrated Circuit (ASIC)

While a large number of SDR products has been developed for running on a GPP (for example, in a desktop computer), the constraints of running on a EDs and the interest in using SDR on such devices have presented new challenges for SDRs. The user requirements include small size and limited weight, and long battery life. The challenge is to create SDR systems capable of meeting these constraints when running on the EDs.

2.4.2.2 Cognitive radio

According to Mitola's early vision, a CR would be realized through the integration of model-based reasoning with software radio and would be trainable in a broad sense, instead of just programmable. The radio can reconfigure itself through an ongoing process of awareness (both of itself and the outside world), perception, reasoning, and decision making. The concept of CR emphasizes enhanced quality of information and experience for the user, with cognition and reconfiguration capabilities as a means to this end. Today, however, CR has become an all-encompassing term for a wide variety of technologies that enable radios to achieve various levels of self-configuration, and with an emphasis on different functionalities, ranging from ubiquitous wireless access, to automated radio resource optimization, to dynamic spectrum access for a future device-centric interference management, to the vision of an ideal CR. Haykin, for example, defines CR as a radio capable of being aware of its surroundings, learning, and adaptively changing its operating parameters in real time with the objective of providing reliable anytime, anywhere, and spectrally efficient communication. The U.S. Federal Communications Commission (FCC) uses a narrower definition for this concept: "A Cognitive Radio (CR) is a radio that can change its transmitter parameters based on interaction with the environment in which it operates. The majority of cognitive radios will probably be SDR (Software Defined Radio) but neither having software nor being field programmable is requirements of a cognitive radio." Despite these differences in both the scope and the application focus of the CR concept, two main characteristics appear to be in common in most definitions. They are reconfigurability and intelligent adaptive behavior. Here by intelligent adaptive behavior we mean the ability to adapt without being a priori programmed to do this; that is, via some form of learning. For example, a handset that learns a radio frequency map in its surrounding could create a location-indexed RSSI vector (latitude, longitude, time, RF, RSSI) and uses a machine-learning algorithm to switch its frequency band as the user moves.

From this it follows that cognitive radio functionality requires at least the following capabilities:

- **Flexibility and agility**: the ability to change the waveform and other radio operational parameters on the fly. In contrast, there is a very limited extent that the current multi-channel multi-radio (MCMR) can do this. Full flexibility becomes possible when CRs are built on top of SDRs. Another important requirement to achieve flexibility, which is less discussed, is reconfigurable or wideband antenna technology.
- **Sensing:** the ability to observe and measure the state of the environment, including spectral occupancy. Sensing is necessary if the device is to change its operation based on its current knowledge of RF environment.
- **Learning and adaptability:** the ability to analyze sensory input, to recognize patterns, and modify internal operational behavior based on the analysis of a new situation, not only based on precoded algorithms but also as a result of a learning mechanism. In contrast, the IEEE 802.11 MAC layer allows a device to adapt its transmission activity to channel availability that it senses. But this is achieved by using a predefined listen-before-talk and exponential back off algorithm instead of a cognitive cycle.

Different interpretation of SDR

Table 2 shows a comparison of different interpretations of CR. The most common aspects of all these interpretations are radio spectrum, as well as spectrum efficiency and primary users.

Table 2 - Comparison of different interpretations of CR.

Aspects	Mitola	Haykin	SDR Forum	FCC	Inf. Theory
User's needs	x				
Context	x				
Intellig. & contr.	x	x	x		
Radio/spectr.	x	x	x	x	x
Spectr. effic.		x	x	x	x
Primary users		x	x	x	x
SDR	x	x			
Cooperation				x	
Reliability		x			

We also need to emphasize that there is yet another ambiguity in the definition of CN, since we cannot equate CN and cognitive radio network (CRN). For example, CN is defined as a network constructed of primary and secondary users, where secondary users are considered the cognitive ones. These users simply obtain the additional information on the activity of the primary users to employ better transmission parameters, in this context limited only to coding. Cognitive networks are wireless networks that consist of two types of users:

- **PRIMARY USERS:** These wireless devices are the primary license holders of the spectrum band of interest. In general, they have priority access to the spectrum and are subject to certain quality-of-service (QoS) constraints that must be guaranteed.
- **SECONDARY USERS**: These users may access the spectrum, which is licensed to the primary users. They are thus secondary users of the wireless spectrum and are often envisioned to be cognitive radios. For the rest of this chapter, we assume the secondary users are cognitive radios (and the primary users are not) and use the terms interchangeably. These cognitive users employ their "cognitive" abilities to communicate while ensuring the communication of primary users is kept at an acceptable level.

Types of adaptable radios

Table 2 summarizes some types of adaptable radio devices.

HARDWARE RADIO: The capability of CR devices changing their radio characteristics is implemented completely in hardware. Thus, once in the field the devices will not be able to change their characteristics other than what is already built in. For example, the range of frequency programmed into the hardware always remains the same, even though the user knows that there is an opportunity to work in a different range. Therefore, the scope is limited in this case.

SOFTWARE RADIO: The capability of CR devices changing their radio characteristics also is implemented in software. Thus, the devices are able to change their characteristics from other than what is already built in. For example, contrasting with the preceding, the range of frequency programmed into the hardware may be changed by uploading a new software patch (say, a simple configuration file).

ADAPTIVE RADIO: This is the capability of CR devices where its radio characteristics are changed by mechanisms such as closed-loop or open-loop controllers. Basically, the devices adapt to the surroundings by sensing and using the preprogrammed logic and control techniques.

RECONFIGURABLE RADIO: The radios in CR devices of which the functionalities can be changed manually. A hardware radio and a software radio both are reconfigurable, though in different ways and to different degrees.

POLICY-BASED RADIO: The changes to the radio functionalities of CR devices are governed by the policies. The policy set usually is available as a data set (or database). For example, the frequencies used by military equipment are not allowed to be used by others under all circumstances. Basically the policy set governs the operational characteristics of the CR devices quite immaterial of whether they are capable.

COGNITIVE RADIO: It has been already defined. This includes databases, policies, learning techniques, and so forth.

INTELLIGENT RADIO: This includes cognitive radios, which are also able to learn as well as predict the situations and adapt themselves. In a general and crude sense, it is a software radio. However, with respect to the previous explanation of the software radio, it just specifies the capability to work with a software control, thus an intelligent radio is much more than a simple software radio.

Networking capabilities of CENs

The primary goal for CENs in nSHIELD is to address SPD functionalities, composability, and other application specific requirements specification for a selected scenario. However, this is only possible of we have an open CEN platform that enable us to develop new SPD features targeted in nSHIELD. Our target for an experimental research on SPD-WSN as an nSHIELD SPD network of CENs is hindered by the lack of open, affordable cognitive radio platform and associated software that are capable of operating with the full network protocol stack. To build SPD functionalities on a CEN it requires that it has a minimal set of networking requirements as they are required for a standard WSN composed of wireless sensor nodes (for example nodes complaint with IEEE 802.15.4 standard)!

In this document, we describe our vision of the building blocks needed to create an open CEN platform for cognitive network experimentation and prototyping in nSHIELD. These include mechanisms for spectrum sensing, and/or MAC protocol tailored to dynamic spectrum access, and interfaces for CNs.

A SPD-WSN composed of CENs

A targeted SPD-WSN composed of CENs may have the following target goals with respect to the application scenario selected.

1. **Case1**: Networked CENs that form a SPD-WSN has only a small set of SPD functionalities to demonstrate only some SPD features (like SPD metrics).

2. **Case 2**: Networked CENs that form a SPD-WSN has a complete set (but still limited comparing to MPN or PN) of SPD functionalities to demonstrate a set of SPD features and CN capabilities against selected types of threats typical for such CNs. An example of such scenario will be SMN scenario.

Case 2 as the most complex scenario in nSHIELD that contain also Case1 will be considered as a typical SPD-WSN model composed of CENs.

2.4.3 SPD considerations for CENs

The new features offered by CR introduced new security, privacy, trust and dependability challenges. The objective in this section is to analyse the SPD issues that impact the CEN generic architectures. Therefore, presenting vulnerabilities inherent to CENs, identifying novel types of abuse, classifying attacks, and analysing their impact on the operation of cognitive radio-based systems is the first task in the design process of a CEN prototype.

2.5 Intrinsically secure ES firmware

Modular systems where each component in the system has unique functionality give raise to security dependencies between the different components. In particular, the security of the entire system is dependent upon the components located at the lowest level. Low level components such as firmware controls how software is loaded and plays a crucial role in software platform security architecture.

Micro nodes must address requirements such as power-consumption and size restrictions. Power nodes may include multiple processors, shared secondary cache memory etc. From a security perspective the both node categories should be able to verify the integrity of loaded software before transfer control to it.

The code that is responsible of transfer images from media to internal memory must be trusted and this involves several security design issues. This code can either be hard coded (ROM) or also subject to software upgrades. In particular the latter case is challenging from a hardware/software embedded system security design point of view. Both Global platform and Trust Zone address this kind of problems, and should be analysed and implemented if appropriate to fulfil the nSHIELD requirements and architecture design.

Resilience against tampering in hostile environments must be balanced against requirements of firmware upgrades.

For Power nodes the Firmware could be very complex since it involves many other tasks that are not security related. One approach is to divide the monolithic firmware into two modules, one is resident in read-only-memory and therefore trusted, the other are checked prior to execution and also subject for upgrade. Since modern processors offers many execution states some peripherals must be initialized both by firmware and the operating system, therefore some functionality should in theory be migrated away from firmware to operating system. Suspend and resume of a laptop are one example of such event.

Micro nodes are expected to be less complex to initialize and therefore the entire firmware might be stored in read-only-memory.

Resilience against power failures, are very hard to implement in firmware. The node must keep a copy of the old active firmware when the downloading a new version. In addition, there must be some atomic mechanism that selects which version to run. Fixed firmware eliminates this kind of problems, but also the possibility to upgrades. Another approach to address this problem is to switch to fixed firmware when the products have been shipped in volumes and found stable.

Self-recovery for a system may involve a watchdog and an alternative media with alternative software. From a firmware perspective this involves more hardware to setup and operate.

Flexibility of choosing any file system and media type will directly map to the size and complexity of the firmware. Restrictions and limitations in the firmware could be addressed by partition the media with different file systems for the boot related software and the payload software. If the same file system is used on both primary and secondary media complexity can be reduced.

The firmware could be implemented as a suite of generic components that could be used for anything in the range of a micro to a power node. The components should also be possible to integrate into existing firmware as a security enhancement.

The interface between firmware and the operating system should also include a hyper-visor. If the hyper-visor is configured it should be invoked together with a guest operating system that enables management of additional guest operating system. The firmware should be designed to eliminate threats from any guests targeted the underlying hardware.

2.6 Power supply protection

2.6.1 State of the art

Sustainable operation of battery powered wireless embedded systems, such as SDR/cognitive enabled node or a micro node is a key challenge for every scenario defined in the scope of the project. Over time, Embedded Systems (ES) have evolved and are becoming more and more sophisticated and complex. For this reason, these systems need a better power supply design.

Current devices operate at lower voltages and higher currents than first models. Consequently, power supply requirements may be more demanding, requiring special attention to features deemed less important in past generations.

One of the basic requirements of a power supply for ES is to generate the necessary supply voltages in the best possible quality and a favourable electrical current which lets them make full use of their capabilities.

[1] presents a study of the power consumption of the different types of node, cognitive enabled nodes and micro nodes. After this analysis the protection of the different systems against external attacks is focused on three key points:

- Study how to provide a continuous power supply source, without any cut in time or, at least, how to keep the system running during a period of time long enough to solve the problem with the main source or to send a warning to alert the person in charge. In the previous phase of the project also it was reported [1] different power supply sources. Three main groups were showed:
    - the energy storage systems: batteries, fuel cells, ultra-capacitors, micro-heat engines, nuclear micro batteries
    - the power harvesting methods like solar power, thermal energy, wind power, pressure variations energy and vibrations
    - the power distribution methods: it is possible to distribute power directly to the nodes or even, perform a wirelessly recharging (electromagnetic radio frequency distribution, elastic or acoustic waves and laser beam)
- Design the appropriate protections to avoid system damages, including different operation modes to plug or unplug critical and non - critical sections of the nodes.
- Monitor the power consumption.

2.6.2 Relationship with pSHIELD

During the previous phase of the project two different protection boards were designed, manufactured and tested.

One of them was for a wireless platform which could have up to five different sub- systems connected. It will include not only the necessary protections to avoid damages into the circuit but also the hardware necessary to let the microprocessor controls the power supply of different sub - systems. To monitor power consumption, a current sense amplifier was included in the design. The second protection board contained only the protections needed to avoid damages into the circuit.

Both protection boards have been tested in order to verify that system is protected against over voltages, overloads, short circuit or over temperatures. Both designs have fulfilled the defined requirements since nodes have integrate d not only the necessary protections but also a mechanism to plug - unplug different sub - systems and a sensor to monitor power consumption.

Both designs can be considered as a starting point in the design of a secure power supply nSHIELD.

2.6.3 References

[1] PSHIELD D3.2 SPD nano micro or personal node technologies prototype report http://www.pshield.eu/index.php?option=com_docman&Itemid=37

2.7 Dependable and Secure Firmware

A secure node requires a dependable firmware which is the foundation block for a trusted node. Several mechanisms have been proposed for safeguarding integrity of a system's software, such as applying a sequence of integrity checks, namely a chain of trust [2], during the firmware execution process where the integrity of every piece of code is checked prior to being executed. Yin et.al. proposed in [1] a multi-backup architecture where multiple copies in the form of backups of the firmware in question can be used to safeguard against modified or corrupted firmware due to malicious attacks or system failures.

The integrity of the firmware shall also be protected during upgrades or revisions typically performed for correcting flaws or for adding functionality. The process might be exploited by adversaries to poison the system with their own malicious code. A method for auditing firmware integrity is proposed in [3]. A static kernel is used for recording an "unbroken sequence of application firmware revisions installed on the system", whereas a signed version of this audit log can be provided for attestation purposes.

Advanced security features have also been proposed to protect against firmware unauthorized access and modifications such as the host-based intrusion detection mechanism proposed in [4] in order to protect against malicious code such as rootkits [5]. A key measure in this deployment of secure mechanisms is the protection of the mechanism itself from being attacked and obsolete.

Contemporary web connected embedded systems may even become subject to advanced web-based attacks, like the cross-site scripting attack which can be used against embedded systems to poison the firmware [6]. A malicious script can be injected in the pages stored on an embedded device and, when executed, can become the vehicle to install a modified firmware on the device.

2.7.1 References

[1] H. Yin, H. Dai, and Z. Jia, Verification-based Multi-backup Firmware Architecture, an Assurance of TrustedBoot Process for the Embedded Systems, Proceedings of 2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11.
[2] H. Lohr, A.-R. Sadeghi, and M. Winandy, "Patterns for secure boot and secure storage in computer systems," International Conference on Availability, Reliability and Security, pp. 569–573, 2010.
[3] M. Lemay, CA. Gunter. "Cumulative attestation kernels for embedded systems". 14th European Symposium on Research in Computer Security, ESORICS 2009. Lecture Notes in Computer Science 2009, LNCS (5789), pp.655-70.
[4] A. Cui, SJ. Stolfo. "Defending embedded systems with software symbiotes". 14th International Symposium on Recent Advances in Intrusion Detection Systems. Lecture Notes in Computer Science 2011. LNCS(6961) pp.358-77.
[5] Vasisht, V.R., Lee, H.-H.S.: "Shark: Architectural support for autonomic protection against stealth by rootkit exploits". In: MICRO, pp. 106–116. IEEE Computer Society, Los Alamitos (2008)
[6] B. Bencsáth, L. Buttyán, T. Paulik. XCS based hidden firmware modification on embedded devices. 2011 International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2011, pp.327.

3 Micro/Personal Node

3.1 Micro Node SPDs from Related EU Projects

3.1.1 Trusted Platform Module (TPM)

A significant area of WSN-related security research aims at utilizing TPM hardware and adapting it to the specific needs of resource constrained applications. Such a TPM-related subject is that of implementing the Direct Anonymous Attestation (DAA) scheme specified by the Trusted Computing Group (TCG). [1] provides a detailed report on the implementation of the aforementioned functionality is provided, as well as suggestions for improvements. Moreover, in [2] an anonymous authentication scheme based on an optimized version of DAA is presented, aimed at resource-constrained mobile devices. Functionality includes secure devices authentication, credential revocation as well as anonymity and non-traceability of said devices against service providers. On the subject of TPMs, research has also focused on the security extensions of mobile platforms for hosting Mobile Trusted Module (MTM) functionality. Both software-based and hardware-based MTMs are examined in [3], and respective techniques for dynamic loading of TPM commands are proposed, aiming to alleviate the performance issues arising from the security facilities of mobile platforms. In [5] the server side of Trusted Computing functionality is examined, aiming to provide anonymous and trustworthy service for users, even counteracting certain insider attacks, which the proposed scheme is able to detect.

3.1.2 Complex Programmable Logic Devices (CPLDs)

Another approach to WSN node security is based on the use of low cost, low energy consumption Complex Programmable Logic Devices (CPLDs). A platform which embeds a CPLD in a standard WSN node is presented in [6], resulting in increased performance of a sensor node by a significant factor as well as a considerable reduction in power consumption. This concept is further expanded in [7] and [8] where various networking and security protocols are implemented on the aforementioned platform and real-world performance compared to existing schemes.

3.1.3 Virtualization

Virtualization is a feature that research has shown it adds to the overall security of the system, in various ways. Firstly, it seems to be a remedy for facing the severe security challenges that mobile devices have, given that they are usually targeting a completely open setup [8]. In addition, efficient virtual machines have successfully been implemented in microkernel based systems, thus enabling the reuse of arbitrary operating systems [9]. The overhead imposed on the kernel growth was rather marginal and the overall performance was found to be similar to other virtual machine implementations. An analysis on how and to which degree recent x86 virtualization extensions can influence the response times of a real-time operating system that hosts virtual machines was performed in [10]. In [11] it was shown that a thin and rather simple virtualization layer can add to the overall system's security, as it provides fewer options for attack to a potential adversary. What is more, this approach was found to exhibit significantly better performance, compared to contemporary full virtualization environments. Finally, regarding the way virtual machines should be implemented, it is claimed in [12] that their construction should follow the principle of incremental complexity growth. Namely, additional functionality should not be included in the trusted computing base of a component if the benefits it offers are less than the drawbacks due to larger risk for introduced bugs and errors. Such an approach can be efficiently implemented and it was able to achieve high throughput and good real-time performance.

3.1.4 Dependability

Embedded systems exhibit a significant number of soft errors, the correction of which imposes equally significant hardware and real-time overhead. For improving embedded systems' dependability, the authors of [13] proposed an approach for classifying errors according to their relevance and the impact of their correction to the system that exploited application knowledge. As a result, the imposed correction overhead was reduced.

3.1.5 Cryptography

An overview of the time and energy overhead that popular cryptographic primitives impose on various popular types of wireless sensor nodes is presented in [1]. Whenever strong encryption is required or rather resource-constrained devices, elliptic-curve cryptography (ECC) is always a strong candidate. In [14] the finite fields F_p, F_{2^d} and F_{p^d} are being investigated for suitability for performing ECC on the ATmega128 microcontroller and it turns out that binary fields are most preferable when efficient implementations are required.

An interesting security scheme for WSN that provides transparent security is proposed in [15]. This scheme is effectively a lightweight CBC-X mode cipher that is able to provide encryption/decryption and authentication all in one. Consequently, it exhibits significant energy gains of about 50-60%, compared to TinySec.

3.1.6 References

[1] P. Trakadas, Th. Zahariadis, H.C. Leligou, S. Voliotis, “Analyzing Energy and Time Overhead of Security Mechanisms in Wireless Sensor Networks,” 15th International Conference on Systems, Signals and Image Processing (IWSSIP 2008), Bratislava, Slovak Republic, June 25-28, 2008.

[2] K. Dietrich, J. Winter, G. Luzhnica and S. Podesser – “Implementation Aspects of Anonymous Credential Systems for Mobile Trusted Platforms” - CMS 2011,Oct 19-21, 2011 Ghent, Belgium.

[3] L. Chen, K. Dietrich, H. Löhr, A-R. Sadeghi, . Wachsmann, J. Winter - "Lightweight Anonymous Authentication with TLS and DAA for Embedded Mobile Devices" - Accepted for the 13th Communications Security Conference (ISC 2010), October 25-28, 2010, Boca Raton, Florida, USA.

[4] K. Dietrich, J. Winter - "Towards Customizable, Application Specific Mobile Trusted Modules" - Accepted for the fifth Annual Workshop on Scalable Trusted Computing (ACM CCS STC), Oct. 4-8, 2010, Chicago, USA.

[5] A. Böttcher, B. Kauer and H. Härtig, “Trusted Computing Serving an Anonymity Service”, Trust '08 Proceedings of the 1st international conference on Trusted Computing and Trust in Information Technologies: Trusted Computing - Challenges and Applications, March 2008.

[6] P. Christou, K. Kyriakoulakos, E. Sotiriadis, K. Papadopoulos, G-G. Mplemenos and I. Papaefstathiou, "Low-Power Security Modules optimized for WSNs", 16th International Workshop on Systems, Signals and Image Processing (IWSSIP), Chalkida Greece, June 2009.

[7] G-G. Mplemenos, P. Christou and I. Papaefstathiou, “Using Reconfigurable Hardware Devices in WSNs for Accelerating and Reducing the Power Consumption of Header Processing Tasks” IEEE Advanced Network and Telecommunication Systems (ANTS), India, 14-16 December 2009.

[8] G-G. Mplemenos, K. Papadopoulos, A. Brokalakis, G. Chrysos, E. Sotiriades, I. Papaefstathiou, “RESENSE: Reconfigurable WSN Nodes”, The Second Wireless Sensing Showcase (WiSiG Showcase 09), National Physical Laboratory, July 2009, London, UK.

[9] Jörg Brakensiek, Axel Dröge, Martin Botteck, Hermann Härtig, Adam Lackorzynski, “Virtualization as an enabler for Security in Mobile Devices”, IIES '08 Proceedings of the 1st workshop on Isolation and integration in embedded systems, April 2008.

[10] Michael Peter, Henning Schild, Adam Lackorzynski, Alexander Warg, “Virtual Machines Jailed – Virtualization in Systems with Small Trusted Computing Bases”, Proceedings of the 1st EuroSys Workshop on Virtualization Technology for Dependable Systems, March 2009.

[11] H. Schild, A. Lackorzynski, and A. Warg, “Faithful Virtualization on a Real-Time Operating System”, Proceedings of the Eleventh Real-Time Linux Workshop, pages 237–243, Dresden, Germany, 2009.

[12] Udo Steinberg and Bernhard Kauer, “NOVA: A Microhypervisor-Based Secure Virtualization Architecture”, Proceedings of the 5th European conference on Computer systems (EuroSys '10), 2010.

[13] S. Liebergeld, M. Peter, and A. Lackorzynski, “Towards Modular Security-Conscious Virtual Machines”, Proceedings of Twelfth Real-Time Linux Workshop, Nairobi, Kenya, October 2010.

[14] Andreas Heinig, Michael Engel, Florian Schmoll, and Peter Marwedel, “Using application knowledge to improve embedded systems dependability”, In Proceedings of the Workshop on Hot Topics in System Dependability (HotDep 2010), Vancouver, Canada, October 2010. USENIX Association.

[15] Anton Kargl and Stefan Pyka and Hermann Seuschek, “Fast Arithmetic on ATmega128 for Elliptic Curve Cryptography”, IACR Cryptology ePrint Archive, 2008.

[16] Shiqun Li, Tieyan Li, Xinkai Wang, Jianying Zhou and Kefei Chen, "Efficient Link Layer Security Scheme for Wireless Sensor Networks", Journal of Information And Computational Science, Vol.4, No.2,pp. 553-567, June 2007.

3.2 Smartcards for security services: Authentication Example in the context of nSHIELD

3.2.1 Overview

A smartcard is a tamperproof secure device resilient to physical attacks used to perform secure transactions. Smartcards are used in a plethora of applications require security such as payment applications, healthcare, physical access control to mention a few. Smartcards can provide multiple security levels for sensitive data stored in them. For instance, a security key can be marked as read-only, while the read operation is accomplished only inside the smartcard. Even more the security key can be protected by a PIN to add one more security level. One of the main advantages of smart card solution is that all the sensitive operations are accomplished in the smart card rather than the terminal or application, which in many cases is not considered trustworthy. Smartcards among to others provide the following security services:

1. Message Authentication code
2. Encryption
3. Identity validity
4. Digital signatures
5. Hash functions
6. Secure key management

3.2.2 Communication with smartcards

The smartcards have the structure depicted in Figure 3

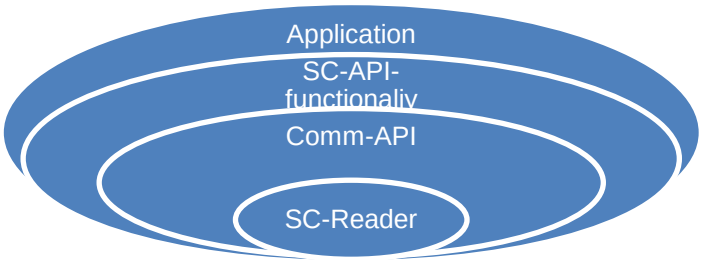

Figure 3 - SmartCard communication structure

It should be noted that even in cases that smartcards do not provide a specific API for communication between the application and the smart card the communication with the can be accomplished by issuing direct command to the smartcard since the smartcards follows the ISO standards [1]. The general structure of a command in smartcards is illustrated in the Table 3

Table 3 - Smartcard request command format

Header					Data
CLA	**INS**	**P1**	**P2**	**Length**	
Class where the command lies	The command itself	Command first parameter	Command second parameter	Data length	Additional data

The command can be issued towards to the smartcard using the underlying communication of the terminal and the smartcard terminal (e.g. serial communication).

For every command issued toward to the smartcard there is a response which its format illustrated in Table 4

Table 4 - Smart card response command format

Data	Response Status
The data returned by the smartcard	Show the result of the requested command, whether the command is successful or failed, and the reason of failure

3.2.3 Smart card file system and data "storage"

Smartcards file system structure is similar to those used in operating system. Particularly the ISO-7816 part 4 defines the structure of the file system as illustrated in the following figure. The master files (MF) can be considered as the root directory, while the dedicated and elementary files are the directories and the data file, in UNIX like operating system, correspondingly.

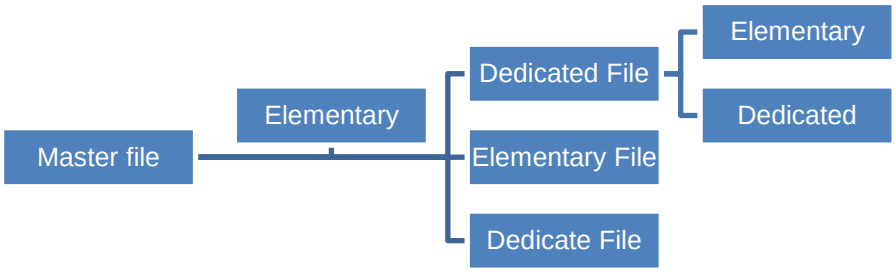

Figure 4 - The logical structure of file system in Smartcards

In smartcards different kind of data can be stored either dynamically or statically, though their capacity is limited. For example, in smartcard can be stored users' data or cryptographic keys for secure transactions. The header in data files defines also the access control rights. Every directory creates a security domain inherit the security policy of its parent. The files in smartcard can be protected with multiple ways:

- Different PIN
- Message authentication code
- Access control restrictions (read, write permissions)
- Digital signatures

This depends on the features incorporated in the smartcard.

3.2.4 Secure services with smart cards

Depending on the type and the manufacturer the smartcards support a number of cryptographic features, including:

- On-card generation of symmetric keys and public key algorithms key pairs
- Digital signatures (based on public key algorithms)
- Symmetric encryption and decryption
- External authentication (host to card)

- Internal authentication (card to host)
- Message authentication code
- Hash functions

Further, smartcards enable protected mode for highly sensitive data, which requires commands to be authenticated and integrity protected either with symmetric or asymmetric keys.

3.2.5 Using smartcards for security services: Authentication Example in the context of nSHIELD

For instance consider the case where an overlay node should authenticate a Micro-Node that incorporates a smartcard module. In that case the overlay node generates a challenge and sends it to the micro node. The Micro-Node passes the challenge to the smartcard and request to create a message authentication code (MAC), assuming that we relying on symmetric key cryptography. The smartcard generate the MAC and send it back to the Micro-Node that forwards the result to the overlay node. The overlay can validate the received MAC either using a TPM or a software based security service. This procedure is illustrated in high level in Figure 5. Note that the symmetric keys requires by the Micro-Node can be either pre-installed in the smartcard or the smartcard itself generate it dynamically.

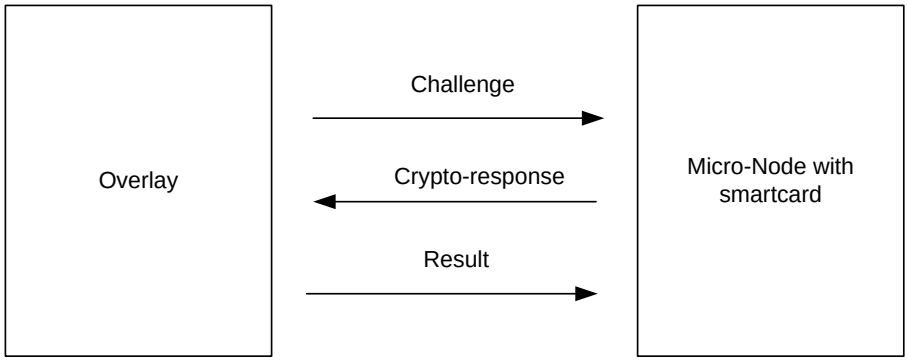

Figure 5 - Example of authentication using smartcards. The overlay authenticates a Micro-Node.

A similar procedure can be followed in the case where Micro-Node needs to authenticate the overlay node. Particularly, the Micro-Node request from smartcard to generate a random number which forward to the overlay node. The overlay node generates the corresponding MAC and send it back to the Micro-Node which request from the smart card to validate the generated MAC. Depending on the result creates either successul or failure response that send to the Micro-Node. Note that the smart card may not be able to validate itself the MAC. In that case, smartcard generate the MAC using the same challenge and the final validation is accomplished by the Micro-Node by comparing the MACs received by the overlay and the smartcard. This procedure is depicted in Figure 6.

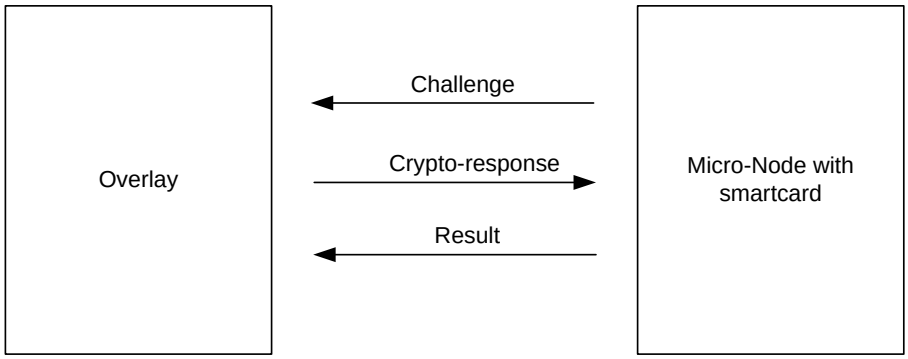

Figure 6 - Example of authentication using smartcards. The Micro-Node authenticates the overlay node

3.2.6 References

[1] ISO/IEC 7816 part 1-15, Available on-line:
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=29257

3.3 SPD and node power consumption

In micro nodes, power consumption must be strongly controlled in order to meet SPD requirements. The issue is even more stringent when portable nodes are considered, where severe size/weight constraints are imposed and the power source has a limited storage as a battery.

However, a careful power management is also required by nodes that are cable-powered, since the dependability can be affected by power failures due to faults or to malicious attacks.

A node, thus, must be aware of its energy budget as well as of any changes on the behaviour of the power supply.

When the energy source is expiring, in fact, a dependable node must report the whole systems (or, at least, its neighbours) about the change in the computational capabilities, and must also shut down in a graceful way, that is avoiding information leaks.

In this regard, focusing at the node level, several strategies of power control/optimization will be enforced. The energy consumption will be regulated by acting on the assessed power knobs, as voltage supply and clock frequency, as well as exploiting low power states where the node does not perform computation. The power regulation strategies will be based on several factors, as the dynamic computational load (measured and estimated), the state of the overall system (in order to predict the required level of service to offer), and the conditions of the environment (in particular for sensors and for power-autonomous systems).

The computational load can be performed both offline and at runtime. The offline profiling can be performed by evaluating the most common execution patterns of the applications that run on the node, in order to collect energy statistics to correlate the execution phases and the power consumption. Such information can be provided to the operating system besides the application executable.

The system, however, will also perform a dynamic evaluation of the computational load, thus adapting its estimations to the actual operating conditions.

Both, the computational load dynamic estimation and the power management, will be implemented as a component of the operating system. Such a software level, in fact, can leverage a nearly complete knowledge of the state of the system (to track the computational power required) and also have access to the low level mechanism to implement the power management.

The software component will interact with the synchronization primitives (barriers, semaphores, etc.) and with the input-output requests, in order to detect, for each application, what computational load is required and how much the various tasks are acting coupled. Moreover, the actual implementation of the synchronization primitives will also be investigated to explore possible optimizations.

Busy waits, false cache sharing and other inefficiencies, in fact, can strongly affect the power consumption of these critical components.

After a monitoring infrastructure will be developed, the power management strategies will be inserted as novel scheduling policies that will take into account the energy requirement of the tasks besides their standard priority.

3.4 SPD based on Face and Voice Verification

This section illustrates the technologies that will be studied and developed to provide the SPD features and functionalities to the Face and Voice Verification scenario (WP7). These technologies will be implemented in embedded system prototypes that will be part of the nSHIELD demonstrators.

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. Currently, a similar requirement is emerging in several application contexts, but with people as the main object: similar services are very useful to perform the recognition, monitoring and traceability of people.

The Face and Voice Verification 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.

3.4.1 Biometric Face Recognition

3.4.1.1 Introduction

Several new face recognition techniques have been proposed recently. The new techniques include recognition from three-dimensional (3D) scans, recognition from high resolution still images, recognition from multiple still images, multi-modal face recognition, multi-algorithms and preprocessing algorithms to correct the illumination and pose variations. These techniques represent a potential in order to improve the performance of automatic face recognition.

The goal of the activities performed in Task 3.2 on this topic is to achieve an improvement in terms of performance with the development of algorithms for all of the methods previously listed. The assessment and the evaluation of these techniques require three main elements: sufficient data; a challenging problem that allows the evaluation of the improvement in terms of performance; and the infrastructure that supports an objective comparison among different approaches.

The Embedded Face Recognition System (EFRS) proposed in nSHIELD project addresses all these requirements. The EFRS data corpus must contain at least 50,000 recordings divided into training and validation partitions. The data corpus contains high resolution still images, taken under controlled lighting conditions and with unstructured illumination, 3D scans and contemporaneously collected still images.

The identification of a challenging problem ensures that researchers can work on sufficiently reasonable, complex and large problems and that the results obtained are valuable, in particular when compared between different approaches. The challenging problem identified to evaluate the EFRS consists of six experiments. The experiments measure the performance on still images taken with controlled lighting and background, uncontrolled lighting and back- ground, 3D imagery, multi-still imagery, and between 3D and still images. The infrastructure ensures that results from different algorithms are computed on the same data sets and that performance scores are generated by the same protocol. To measure the improvements introduced by the EFRS, the Face Recognition Vendor Test of 2002 year (FRVT), an independent evaluation on the collected data, will be conducted.

There is an animated debate among researchers in order to understand which face recognition method or technique will have better performance, in particular when the discussion is related to embedded systems. The EFRS should provide answers to some of these questions. Currently the discussion is focused on a key topic: will recognition from 3D imagery be more effective than recognition from high resolution 2D imagery? We are going to state conjectures, and relate them to specific experiments that will allow an

assessment of the conjectures at the conclusion of this project.

3.4.1.2 Design of Data Set and Challenge Problem

The design of the EFRS starts from the performance measured using FRVT, establishes a performance goal that is an order of magnitude greater, and then designs a data corpus and challenge problem that supports will allow to reach EFRS performance goal.

The starting point for measuring the increase in performance is the high computational intensity test (HCInt) of the FRVT. The images in the HCInt corpus are taken indoors under controlled lighting. The performance point selected as the reference is a verification rate of 80% (error rate of 20%) at a false accept rate (FAR) of 0.1%. This is the performance level of the top three FRVT 2002 participants. An order of magnitude improvement in performance that we expect from EFRS requires a verification rate of 98% (2% error rate) at the same fixed FAR of 0.1%.

A challenge to designing the EFRS is collecting sufficient data to measure an error rate of 2%. Verification performance is characterized by two statistics: verification rate and false accept rate. The false accept rate is computed from comparisons between faces of different people. These comparisons are called non-matches. In most experiments, there are sufficient non-match scores because the number of non-match scores is usually quadratic in the size of the data set. The verification rate is computed from comparisons between two facial images of the same person. These comparisons are called match scores. Because the number of match scores is linear in the data set size, generating a sufficient number of matches can be difficult.

For a verification rate of 98%, the expect verification error rate is one in every 50 match scores. To be able to perform advanced statistical analysis, 50,000 match scores are required. From 50,000 match scores, the expected number of verification errors is 1,000 (at the EFRS performance goal).

The challenge is to design a data collection protocol that yields 50,000 match scores. We accomplished this by collecting images for a medium number of people with a medium number of replicates. The proposed EFRS data collection is based on the acquisition of images of 200 subjects once a week for a year, which generates approximately 50,000 match scores.

The design, development, tuning and evaluation of face recognition algorithms require three data partitions: training, validation, and testing. The EFRS challenge problem provides training and validation partitions to developers. A separate testing partition is being collected and sequestered for an independent evaluation.

The representation, feature selection, and classifier training is conducted on the training partition. For example, in PCA-based (Principle Component Analysis) and LDA-based (Linear Discriminant Analysis) face recognition, the subspace representation is learned from the training set. In vector machine (SVM) based face recognition algorithms, the SVM classifier is trained on the data in the training partition.

The challenge problem experiments must be constructed from data in the validation partition. During algorithm development, repeated runs are made on the challenge problems. This allows developers to assess the best approaches and tune their algorithms. Repeated runs produce algorithms that are tuned to the validation partition. An algorithm that is not designed properly will not generalize to another data set.

To obtain an objective measure of performance requires that results are computed on a separate test data set. The test partition measures how well an approach generalizes to another data set. By sequestering the data in test partition, participants cannot tune their algorithm or system to the test data. This allows for an unbiased assessment of algorithm and system performance.

The EFRS experimental protocol is based on the FRVT 2002 testing protocols. For an experiment, the input to an algorithm is two sets of images: target and query sets. Images in the target set represent facial

images known to a system. Images in the query set represent unknown images presented to a system for recognition. The output from an algorithm is a similarity matrix, in which each element is a similarity score that measures the degree of similarity between two facial images. The similarity matrix is comprised of the similarity scores between all pairs of images in the target and query matrices. Verification scores are computed from the similarity matrix.

3.4.1.3 Description of the Data Set

The EFRS data corpus is part of an ongoing multi- modal biometric data collection.

A *subject session* is the set of all images of a person taken each time a person's biometric data is collected. The EFRS data for a subject session consists of four controlled still images, two uncontrolled still images, and one three-dimensional image. Figure 7 shows a set of images for one subject session. The controlled images are taken in a studio setting, are full frontal facial images taken under two lighting conditions (two or three studio lights) and with two facial expressions (smiling and neutral). The uncontrolled images were taken in varying illumination conditions; e.g., hallways, atria, or outdoors. Each set of uncontrolled images contains two expressions, smiling and neutral. The 3D images are taken under controlled illumination conditions appropriate for the sensor (structured light sensor that takes a 640 by 480 range sampling and a registered color image), not the same as the conditions for the controlled still images. In the FRP, 3D images consist of both range and texture channels. The sensor acquires the texture channel just after the acquisition of the shape channel. This can result in subject motion that can cause poor registration between the texture and shape channels. The still images are taken with a 4 Megapixel camera.

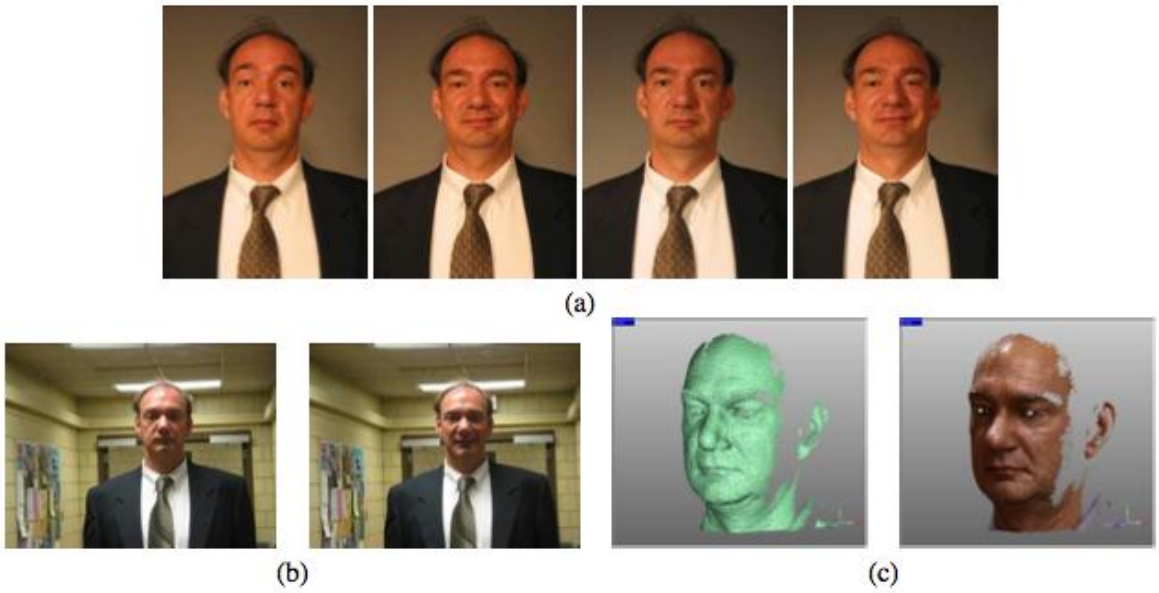

Figure 7 - Images from one subject session. (a) Four controlled stills, (b) two uncontrolled stills, and (c) 3D shape channel and texture channel pasted on 3D shape channel.

Table 5 - Size of faces in the validation set imagery broken out by category.

	Mean	Median	Std. Dev
Controlled	261	260	19
Uncontrolled	144	143	14
3D	160	162	15

Size is measured in pixels between the centers of the eyes. Reported are mean, median, and standard deviation.

Images are either 1704x2272 pixels or 1200x1600 pixels. Images are in JPEG format and storage sizes range from 1.2 Mbytes to 3.1 Mbytes. Subjects are approximately 1.5 meters from the sensor.

Table 5 summarizes the size of the faces for the uncontrolled, controlled, and 3D image categories. For comparison, the average distance between the centers of the eyes in the FRVT 2002 database is 68 pixels with a standard deviation of 8.7 pixels.

The data required for the experiments on the EFRS are divided into training and validation partitions. From the training partition, two training sets are distributed. The first is the *large still training set,* which is designed for training still face recognition algorithms. The large still training set consists of 12,776 images from 222 subjects, with 6,388 controlled still images and 6,388 uncontrolled still images. The large still training set contains from 9 to 16 subject sessions per subject, with the mode being 16. The second training set is the *3D training set* that contains 3D scans, and controlled and uncontrolled still images from 943 subject sessions. The 3D training set is for training 3D and 3D to 2D algorithms. Still face recognition algorithms can be training from the 3D training set when experiments that compare 3D and still algorithms need to control for training.

The validation set contains images from 466 subjects collected in 4,007 subject sessions. The demographics of the validation partition broken out by sex, age, and race are given in Figure 8. The validation partition contains from 1 to 22 subject sessions per subject (see Figure 9).

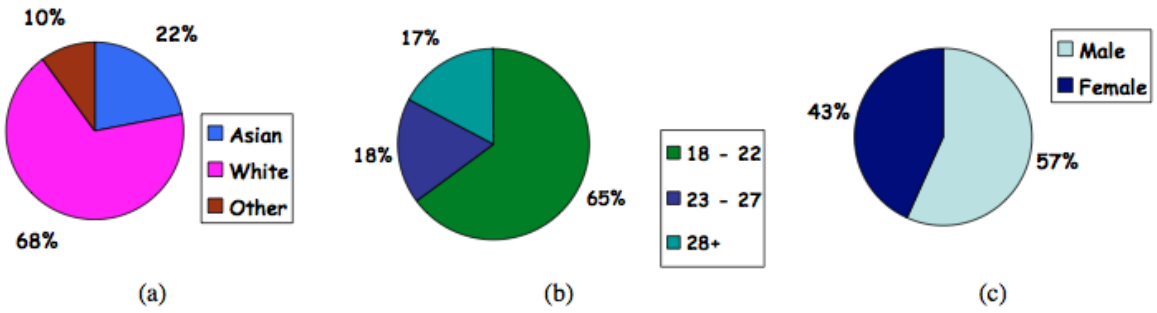

Figure 8 - Demographics of FRP ver2.0 validation partition by (a) race, (b) age, and (c) sex.

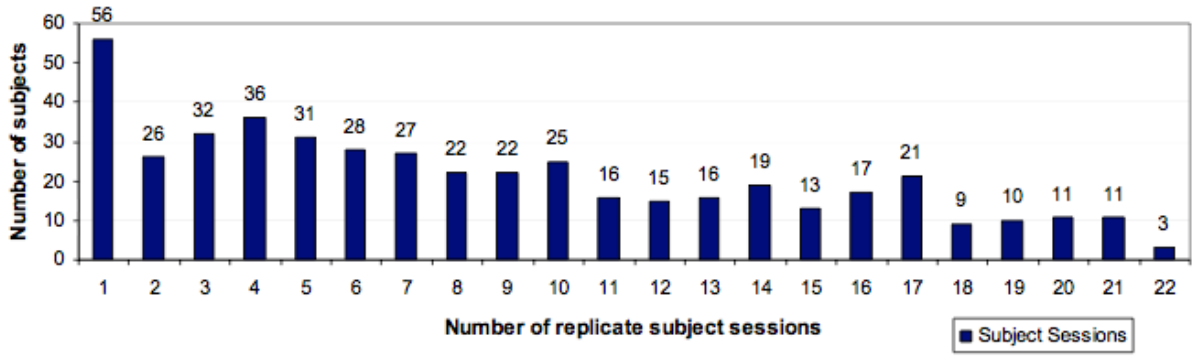

Figure 9 - Histogram of the distribution of subjects for a given number of replicate subject sessions. The histogram is for the ver2.0 validation partition.

3.4.1.4 Description of Experiments

The experiments that will EFRS are designed to improve face recognition in general with emphasis on 3D and high resolution still imagery. EFRS will perform six experiments:

- **Experiment 1** measures performance on the classic face recognition problem: recognition from frontal facial images taken under controlled illumination. To encourage the development of high resolution recognition, all controlled still images are high resolution. In Experiment 1, the biometric samples in the target and query sets consist of a single controlled still image. You can observe that multi-still images of a person can substantially improve performance.
- **Experiment 2** is designed to examine the effect of multiple still images on performance. In this experiment, each biometric sample consists of the four controlled images of a person taken in a subject session. The biometric samples in the target and query sets are composed of the four controlled images of each person from a subject session.
- Recognizing faces under uncontrolled illumination has numerous applications and is one of the most difficult problems in face recognition. **Experiment 4** is designed to measure progress on recognition from uncontrolled frontal still images. In Experiment 4, the target set consists of single controlled still images, and the query set consists of single uncontrolled still images. Proponents of 3D face recognition claim that 3D imagery is capable of achieving an order of magnitude increase in face recognition performance.
- Experiments 3, 5, and 6 examine different potential implementations of 3D face recognition:
    - **Experiment 3** measures performance when both the enrolled and query images are 3D. In Experiment 3, the target and query sets consist of 3D facial images. One potential scenario for 3D face recognition is that the enrolled images are 3D and the target images are still 2D images.
    - **Experiment 5** explores this scenario when the query images are controlled.
    - **Experiment 6** examines the uncontrolled query image scenario. In both experiments, the target set consists of 3D images. In Experiment 5, the query set consists of a single controlled still. In Experiment 6, the query set consists of a single uncontrolled still.

3.4.1.5 Baseline Performance

The baseline performance is introduced to demonstrate that a challenge problem can be executed, can provide a minimum level of performance and a set of controls for detailed studies. A PCA-based face recognition is selected as the baseline algorithm.

The initial set of baseline performance results will be given for Experiments 1, 2, 3, and 4. For Experiments 1, 2, and 4, baseline scores are computed from the same PCA-based implementation. In Experiment 2, a fusion module is added to handle multiple recordings in the biometric samples. The algorithm is trained on a sub- set of 2,048 images from the large training set. The representation consists of the first 1,228 eigenfeatures (60% of the total eigenfeatures). All images were preprocessed by performing geometric normalization, masking, histogram equalization, and rescaling pixels to have mean zero and unit variance. All PCA spaces are whitened. The distance in nearest neighbor classifier is the cosine of the angle between two representations in a PCA-space. In Experiment 2, each biometric sample consists of four still images, and comparing two biometric samples involves two sets of four images. Matching all four images in both sets produces 16 similarity scores. For Experiment 2, the final similarity score between the two biometric samples is the average of the 16 similarity scores between the individual still images.

An example set of baseline performance results is given for Experiment 3 (3D versus 3D face recognition) in the following paragraphs. It has been obtained in previous experiments performed by independent research team and can be considered as a reference point. The baseline algorithm for the 3D scans consists of PCA performed on the shape and texture channels separately and then fused. Performance scores are given for each channel separately and for the shape and texture channels fused. We also fused the 3D shape channel and one of the controlled still images. The controlled still is taken from the same subject session as the 3D scan. Using the controlled still models a situation where superior still camera is incorporated into the 3D sensor. The baseline algorithm for the texture channel is the same as

in Experiment 1.

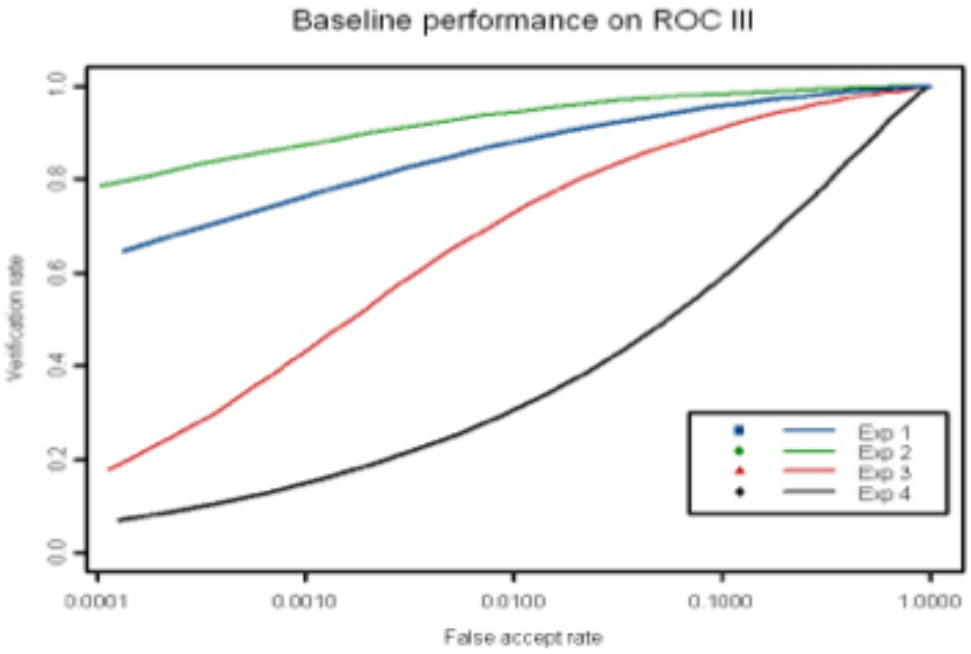

Figure 10 – Example of expected baseline ROC performance for Experiments 1, 2, 3, and 4.

The PCA algorithm adapted for 3D is based on Chang et al[2].

The results obtained in the example of baseline verification performance for Experiments 1, 2, 3, and 4 are shown in Figure 10. Verification performance is computed from target images collected in the fall semester and query images collected in the Spring semester. For these results, the time lapse between images is between two and ten months. Performance is reported on a Receiver Operator Characteristic (ROC) that shows the trade-off between verification and false accept rates. The false accept rate axis is logarithmic. The results for Experiment 3 are based on fused shape and texture channels. The best baseline performance should be achieved by multi-still images, followed by a single controlled still, then 3D scans. The most difficult category should be the uncontrolled stills.

Figure 11 shows another example of baseline performance for five configurations of the 3D baseline algorithms: fusion of 3D shape and one controlled still; controlled still; fusion of 3D shape and 3D texture; 3D shape; and 3D texture. The best result is achieved by fusing the 3D shape channel and one controlled still image. This result suggests that 3D sensors equipped with higher quality still cameras and illumination better optimized to still cameras may improve performance of 3D systems.

[2] Kyong I. Chang, Kevin W. Bowyer, and Patrick J. Flynn, “An evaluation of multi-modal 2d+3d face biometrics”, IEEE Trans. PAMI, vol. 27, no. 4, pp. 619–624, 2005

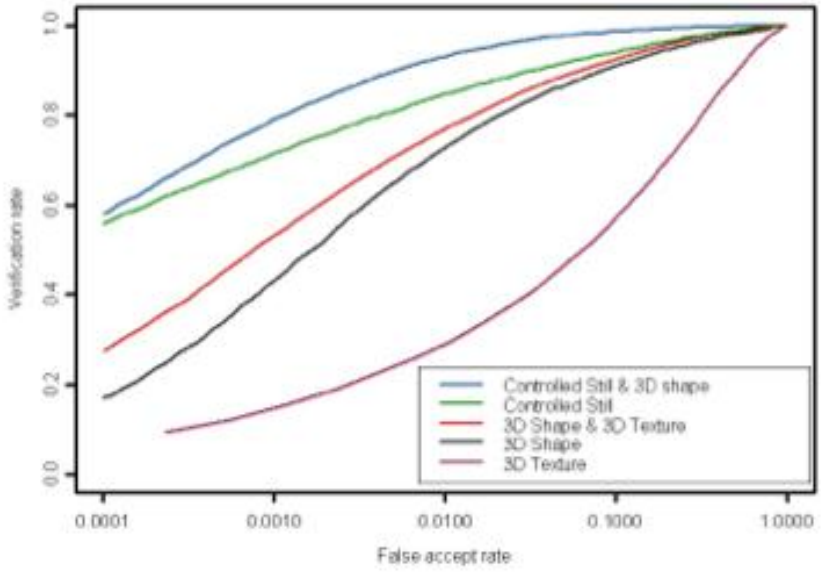

Figure 11 - Example of baseline ROC performance for Experiment 3 component study.

Successful development of pattern recognition algorithms requires that one knows the distributional properties of objects being recognized. A natural starting point is PCA, which assumes the facial distribution has a multi-variate Gaussian distribution in projection space.

In the first facial statistics experiment we examine the effect of the training set size on the eigenspectrum. If the eigenspectrum is stable, then the variance of the facial statistics on the principal components is stable. The eigenspectrum is computed for five training sets of size 512, 1,024, 2,048, 4,096, and 8,192. All the training sets are subsets of the large still training set. The expected eigenspectra should be similar to the ones plotted in Figure 12. The horizontal axis is the index for the eigenvalue on a logarithmic scale and the vertical axis is the eigenvalue on a logarithmic scale. The main part of the spectrum consists of the low to mid order eigenvalues. For all five eigenspectra, the main parts overlap.

The eigenvalues are estimates of the variance of the facespace distribution along the principal axes. Figure 12 shows that the estimates of the variances on the principal components should be stable as the size of training set increases, excluding the tails. The main part of the eigenspectrum is approximately linear, which suggests that to a first order approximation there is a 1/f relationship between eigen-index and the eigenvalues.

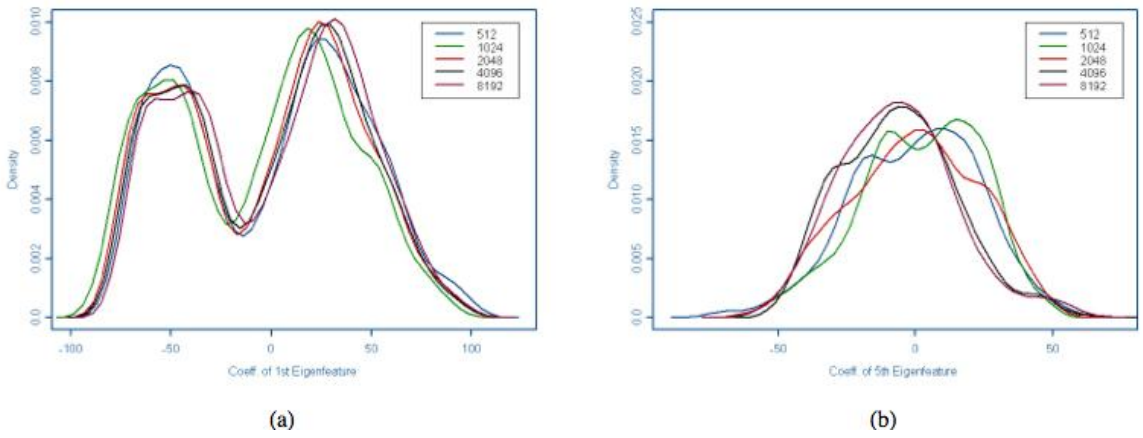

Figure 12 - Estimated densities.

Figure 12 describes an example of performance on Experiment 1 for training sets of size 512, 1,024, 2,048, 4,096, and 8,192. The figure illustrates the estimated densities for the (a) 1st and (b) 5th eigen-coefficients for each training set (the numbers in the legend are the training set size). To generate the curve label 1024 in (a), a set of images are projected on the 1st eigenfeature generated the 1024 training set. The set of images projected onto the eigenfeatures is a subset of 512 images in common to all five training sets. All other curves were generated in a similar manner. Verification performances at a false accept rate of 0.1% is reported (vertical axis). The horizontal axis is the number of eigenfeatures in the representation. The eigenfeatures selected are the first n components. The training set of size 512 approximates the size of the training set in the FERET Sep96 protocol. This curve approximates what was observed by Moon and Phillips[3], where performance increases, peaks, and then decreases slightly. Performance peaks for training sets of size 2,048 and 4,096 and then starts to decrease for the training set of size 8,192. For training sets of size 2,048 and 4,096, there is a large region where performance is stable. The training sets of size 2,048, 4,096, and 8,192 have tails where performance degrades to near zero.

The examples described in this section in order to identify the two most important consequences that we expect from the experiment: first, it is evident that increasing the training set increases also the performance to a point, and second, it is clear that the selection of the cutoff index is not critical.

In the following section we describe the algorithm that will be adopted for face recognition.

3.4.1.6 The Eigenface technique

The Eigenface method starts from the idea to extract the basic faces features: this simplify the problem to a lower dimension. The PCA (Principal Component Analysis) is the selected method (also known in the pattern recognition application as Karhunen-Loève (KL) transform) to extract the principal components of the faces distribution. These eigenvectors are computed from the covariance matrix of the face pictures set (faces to recognize); every single eigenvector represent the feature set of the differences among the face picture set. The graphical representations of the eigenvectors are also similar to faces: for this reason they are called eigenfaces.

The eigenfaces set defines the so called “face space”. In the recognition phase, the unknown face pictures are projected on the face space to compute the distance from the reference faces.

Each unknown face is represented (reducing the dimensionality of the problem) by encoding the differences from a selection of the reference face pictures. The unknown face approximation operation considers only the eigenfaces providing higher eigenvalues (variance index in the face space). In other words in the face recognition the unknown face is projected on the face space to compute a set of weights of differences with the reference eigenvalues. This operation first allows to recognize if the picture is a face (known or not) if its projection is close enough to the reference face space. In this case the face is classified using the computed weights, deciding for a known or unknown face. A recurring unknown face can be added to the reference known face set, recalculating the whole face space. The best matching of the face projection with the faces in the reference faces set allows to identify the individual.

Going in the detail of the “face space” evaluation process requires some introductory considerations. A generic bi-dimensional picture can be gray level converted and eventually adjusted for brightness and contrast. If square shaped (the general case slightly differs) it can be defined by an N x N matrix of pixels, each of them is a point in a N2-dimension space. A set of pictures can hence map to a set of points on this space.

In our case, every picture refers to faces: the representation in the N2 space will not be randomly distributed. Also, the PCA analysis provides the best vectors representing the pictures distribution. It’s possible to gather that these vectors can define a subspace (of the whole space) for generic face pictures (called “face space”). The following figure shows this concept.

[3] H. Moon and P. J. Phillips, “Computational and performance aspects of PCA-based face-recognition algorithms”, Percep- tion, vol. 30, pp. 303–321, 2001

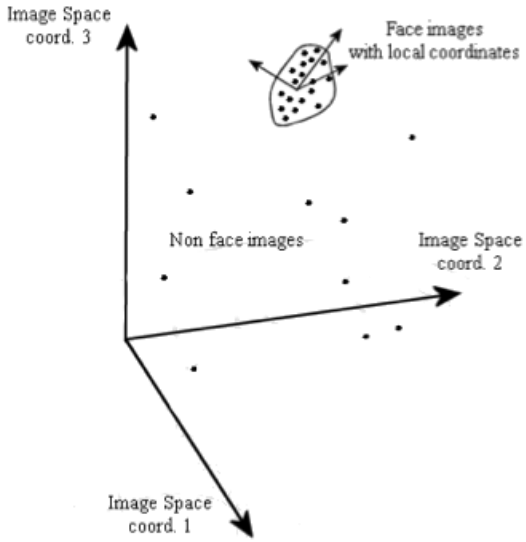

Figure 13 - Space distribution of faces images.

Each vector of the subspace so defined has a dimension N; these vectors are the eigenvector of the covariance matrix corresponding to the original images, and given that shown have the appearance of a face, they are called "eigenface".

More formally, given a training set of images:

$$\Gamma_1, \Gamma_2, \ldots \Gamma_M$$

the average face is computed as:

$$\psi = \frac{1}{M} \sum_{i=1}^{M} \Gamma_i$$

Each face of the training set differs from the average according to the vector:

$$\phi_i = \Gamma_i - \psi$$

This set of vectors of large size is then subjected to analysis of the main components that allows to obtain a set of orthonormal vectors u_i and of scalars λ_i associated with them, that best describes the data distribution. The vectors u_i and the scalars λ_i are respectively the eigenvector and the eigenvalue of the covariance matrix:

$$C = \frac{1}{M} \sum_{i=1}^{M} \phi_i \, \phi_i^T = \frac{1}{M} A \, A^T \quad where\ the\ matrix\ A = [\phi_1 \, \phi_2 \ldots \phi_M]$$

The mechanism that allows to reduce the dimensionality of the problem is based on the identification of the Mi (Mi ≤ M) further eigenvalue of the training set, with which is possible to select the corresponding eigenvector. These form the basis of a new space of representation of data, particularly from the reduced dimensionality. The number of eigenvector considered is chosen heuristically and depends strongly on the distribution of the eigenvalue. To improve the effectiveness of this approximation the background it is normally cut from the images; in this way it makes zero the value of the eigenface outside of the face.

At this point the identification is a simple pattern-recognition process.

Every new image Γ to identify is transformed into the eigenface components through a projection on the "face space" with the simple operation:

$$\omega_k = u^T_K (\Gamma - \psi),$$

with k=1,..., M' (and u^T_K transposed to the base of the transformed space) that consists of multiplications and sums, point to point of the image.

The values thus obtained from a weight vector $\Omega^T = [\omega_1 \omega_2 \ldots \omega_M]$ which expresses the contribution of each eigenface in representing the image data. It is now clear how M' eigenface may constitute a basis set to represent the other images. The vector Ω^T is used to determine, if it exists, which of the predefined classes describes in best way the image (through a nearest-neighbour algorithm type). The simplest way to determine which class best describes the face in question consists in identifying the class k that minimizes the euclidean distance:

$$e_k = \| \Omega - \Omega_k \| = \sqrt{\sum_{k=1}^{M^i} (\Omega - \Omega_k)^2}$$

where Ω_k is the vector that describes the k-th class. A face is classified as belonging to the class k if the minimum distance e_k is below a predetermined threshold value ϑ_e; otherwise the face is classified as unknown. In addition to this and to consider that the image of a generic face should project itself in extreme proximity of the "face space", which in general, as it was built (the faces of the training set), should describe all the images with the appearance of a face. In other words, the distance ε of an image from its projection should be within a certain threshold ϑ_δ.

In general, four possible cases may arise, as shown in Figure 14:

- The carrier is near the "face space" and its projection to a class;
- The carrier is near the "face space", but its projection is not close to any known class;
- The carrier is far from the "face space", but its projection and close to a class known;
- The carrier is far from the "face space" and its projection is not close to any class known.

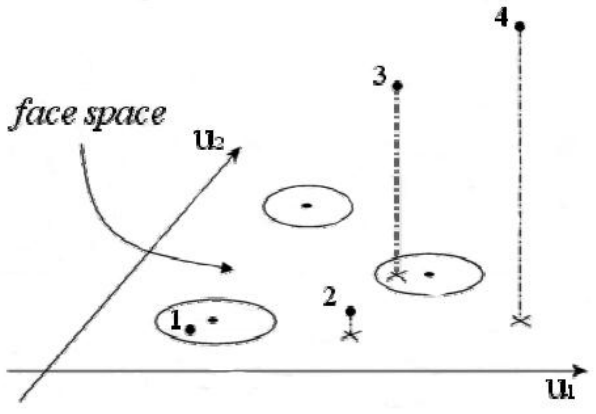

Figure 14 - Example of a simple "face space" consisting of just two eigenface (u_1 ed u_2) and from three individuals known (Ω_1 , Ω_2 e Ω_3).

In the first case, the individual is recognized and identified in the second case it detects only the presence but is not recognized, in the third case could present a typical false-positive, but because of the apparent

distance between the carrier and "face space "and can refuse to recognize, in the fourth case is assumed that it is not even a face, much less known.

Another important peculiarity of this technique consists in being able to use the space formed by the best eigenface to detect faces within an image. The creation of the weight vector is nothing but a projection of the space "facespace" low dimensional (ωk = ukT (Γ − ψ)) so the distance ε between the image and its projection coincides with the distance between the image of the average deducted:

$$\phi = \Gamma - \psi$$

and the projection of the vector of weights in the "face space":

$$\phi_f = \sum_{k=1}^{M^i} \omega_k u_k$$

Note that in this case the appearance of the projected image will not be in general any feature of the face. To detect the presence of a face in the image it is necessary to calculate the distances between different portions of the image and the projection on the face sought. In this way is to generate a map ("facemap") of distances ε(x,y). The only flaw of this approach for the identification of faces is the computational cost, which increases with the granularity with which it analyses the image.

You must then try to extend the eigenface with the aim of making it well suited to managing large databases. We adopt the so-called "modular eigenspace" that, when used in support of traditional Eigenface, can demonstrate an improvement in recognition accuracy. This extension consists in an additional "layer" of key features of the face such as eyes, nose and mouth (Figure 15). In this circumstance one speaks of: eigeneye, eigennose and eigenmouth, or more generally of eigenfeature.

The new representation of the faces can be seen, in a modular fashion, as a description of the entire low-resolution face, combined with a more detailed facial features on the most salient.

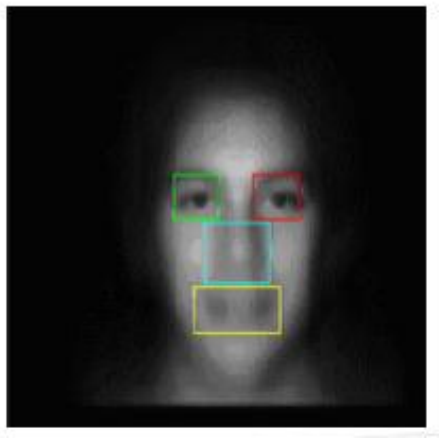

Figure 15 - Eigenface in which domains were identified: eigeneye (left and right), eigennose and eigenmouth.

Of course achieving this technique needs an automated method of detection of the characteristic elements of the image (Figure 16): this can be taken from the mechanism adopted to identify faces offered directly from the Eigenface. Similarly to the distance from the space of faces, in this circumstance is called away from the “feature space”.

Figure 16 - Example of identification of eigenfeature.

This extension is suitable above all to offer a valuable mechanism for modular reconstruction of images, which is advantageous in terms of compression. And thanks to the more details provided by eigenfeature the reconstructed images show a higher quality than the reconstruction from eigenface.

The advantage offered by the eigenfeature is the ability to overcome some weaknesses of the standard eigenface method. In fact, the standard eigenface recognition system can be fooled by gross variations in the input image (hats, beards, etc...).

Finally we will test the use of infrared images with the Eigenface technique. An infrared image (or thermogram) has the characteristic of showing the distribution of heat emitted by an object. While this approach may prove a formidable strength to attack with "mask", and the ability to work with any type of lighting (also absent), may ultimately prove a big problem with people who wear glasses, such as infrared these are very often completely opaque. Considering images of individuals without glasses the benefits are obvious, especially on profile pictures.

3.4.2 Voice Verification

Voice Verification is a technique that allows to verify the identity of a person comparing his voice with a biometric profile. Voice Verification is based on Voice Detection (VD), a technique used to process sound, in which the presence or absence of human speech is automatically detected. The main applications of VD are in speech coding and in speech recognition. It can facilitate the processing of speech, and can also be used to disable some processes during the sections of not-spoken in an audio session: it can avoid the necessity to perform unnecessary encoding or transmission of audio packets of silence in VoIP applications, thus obtaining a gain in computation time and bandwidth in the network. In the case of this project, the VD is mostly oriented to security.

VD is an important technology for developing applications based on speech. Some algorithms have been developed to provide various features and we will consider some of these techniques to optimize tradeoffs between latency, sensitivity, accuracy and computational cost in security applications on embedded systems.

Some VD algorithms also provide further analysis, for example, detection of voiced speech, unvoiced and sustained. VD is usually independent of language. It was initially designed to be used in time assignment speech interpolation systems (TASI).

3.4.2.1 Description of the VD algorithm

The typical VD algorithm adopts in the following approach:

- the first step is a noise reduction, for example by spectral subtraction.

- The second step consists in the extraction of some features or quantities from a section of input signal,
- and finally a classification rule is applied in order to identify the section of speech as speech or not speech. Often this classification is based on one or more threshold values calculated.

The algorithm may provide some further feedback in which the decision of VD is used to improve the estimate of the noise in the phase of noise reduction, or to vary in an adaptive way the threshold.

These feedback operations allow to increase the performance of the VD when you have to do with non-stationary noise (i.e. when it has many variations). Some methods of VD formulate the rule frame by frame using an instantaneous measurement of the distance between speech and noise. These measures include the spectral slope, correlation coefficients, the logarithmic ratio of similarity, and distance measurements cepstral, weighted cepstral and modified. Regardless of the choice of algorithms of VD, we must make a tradeoff between having voice detected as noise and the noise detected as an entry (i.e. between false positive and false negative). An algorithm of VD must be able to detect speech in the presence of a wide variety of acoustic noise in the background. In these conditions of difficult detection is often preferable that the VD perform a fail-safe, i.e. indicating that there is speech when the decision is in doubt, in such a way as to reduce the possibility of losing speech segments. The greatest difficulty in detecting speech in this situation is the signal/noise ratios (SNR) with very low which has to do. It might even be impossible to distinguish between speech and noise using the techniques of simple level detection when some expressions of speech are covered by noise.

3.4.2.2 Evaluation of performance

To evaluate the performance of VD, we compare the output using the test records with those of an ideal VD, created by hand by noting the presence/absence of the voice recordings. The performance of the RV is commonly evaluated using the following parameters:

- FEC (Front End Clipping): cutting introduced in the passage from noise to speech activity;
- MSC (Mid Speech Clipping): cut due to bad speech classification as noise;
- OVER: noise interpreted as speech due to a condition of VD which remains active passing from speech to noise;
- NDS (Noise Detected as Speech): noise interpreted as speech in a period of silence.

Although the method described above provides information on the useful objectives regarding the performance of the VD, it is only an approximate measure of the subjectively effect. For example, the effects of cutting the audio signal may sometimes be hidden by the presence of background noise, depending on the model chosen for the synthesis of the comfort noise, so that some cuts measured with the objective tests are not audible. The type of test requires a number of listeners to judge the recordings containing the results obtained by the algorithm of VD tested. Listeners have to give a score to the following features:

- quality,
- difficulty in understanding,
- audibility of the cut.

These scores, obtained by listening to different sequences of speech, are used to calculate the average results for each feature, thereby obtaining an estimate of the global behavior of the VD. To conclude, where the objective methods are very useful in an early stage to evaluate the quality of the VD, the subjective methods are more significant. Although they are more expensive (because they require the participation of a number of people for a few days), they are generally used when a proposal must be standardized.

3.4.2.3 Algorithm based on Wavelet Packet Transform and Voice Activity Shape

The first VD algorithm implemented is the one proposed by Chiodi and Massicotte[4]. The algorithm is based on the Wavelet Packet Transform (WPT) and can be divided into four phases, as shown in the following figure:

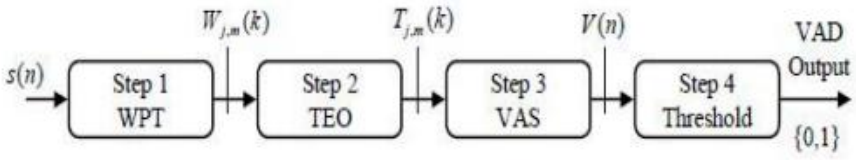

Figure 17 - Scheme of VD algorithm based on WPT

Phase 1: Decomposition by Wavelet

The speech signal s(n) is decomposed into frames of 256 samples each. The choice of size depends on the choice of the frequency range you want to analyze and sample rate. If you want to have more information at low frequencies, the frame must be bigger, but if you want to analyze the signal at high frequencies, the frame must be smaller. For the purposes of this project, 256 is a good value, considering that the sampling frequency is 8 KHz. To decompose the signal, are used filter banks corresponding to this mother wavelet, to obtain the coefficients of approximation and detail data by the following relations:

$$a(k) = \sum_{n=1}^{N} g(n-2k)s(n)$$

$$d(k) = \sum_{n=1}^{N} h(n-2k)s(n)$$

where g(n) and h(n) are to indicate the coefficients of the low-pass filter and the high pass filter, respectively. In this implementation of the algorithm, using the filter coefficients with data from the Daubechies wavelet, since they allow to maintain the selectivity of the frequencies with increasing level of decomposition of the wavelet. DWT is implemented via the cascade of these filters, and you get the tree decomposition corresponding to the Wavelet Packet Transform. This approach makes the DWT filters adaptable to real time applications. Using the WPT, each frame is decomposed into sub band signals S (S = 16). Implementation carried out, using a balance decomposition tree with 4 levels, and the choice of mother wavelet of Daubechies Wavelet fell on to 10 points. Are obtained 16 signals of different size (depending on the level of decomposition), referred to as Wj, m (k), where j is the level of decomposition (in the frequency scale), and j = 1, 2, ..., 2j (j = 8), m is the index of the sub band signal (1 ≤ m ≤ S), and k is the index of the coefficients k = 1, 2, ..., 2j.

The level of decomposition j represents the frequency range of interest to detect speech frames and not spoken.

Phase 2: TEO application

The objective of the operator TEO is to determine the frequency of each sub band signal. It is calculated for each sub band using the equation:

$$Tj_{,m}(k) = \Psi[W_{j,m}(k)]$$

[4] R. Chiodi and D. Massicotte, Voice Activity Detection Based on Wavelet Packet Transform in Communication Nonlinear Channel, 2009 First International Conference on Advances in Satellite and Space Communications.

This operation allows to detect the shape of the frequency and the decay of the moments are not transient and aperiodic signals, and also allows you to suppress the noise.

Phase 3: Extracting Voice Activity Shape

After application of the TEO, the algorithm calculates the variance of each signal TEO, Tj,m(k), and calculates the following summation:

$$V(n) = \sum_{m=1}^{S} var\ (Tj_{,m}(k))\ k = 1,2,\ldots,2^{j} \qquad eq.1$$

where n = 1, 2, ..., N and var(•) is the operator of variance. Each frame is assigned a value V(n). The result that is obtained corresponds to a curve of Voice Activity Shape (VAS) that characterizes the evolution of speech and non-spoken in the observed signal s (n). The value of V (n) is high during periods of voice and low during periods of non-voice.

Phase 4: decision based on thresholding

The algorithm Nails and Massicotte is based on a fixed thresholding on the VAS values, calculated by taking the first 10 frames, measured as noisy. For reasons of efficiency, implementation of the algorithm is not used that fixed thresholding, adaptive thresholding, but a weighted (AWT) in the algorithm described by Chen, Wu, Ruan and Truong. It is calculated by an iterative algorithm, following the steps described below:

1. We put the index k = 1 and we define $V^{(1)}(n) = V(n)$, where V (n) is given by the equation 1

2. $V^{(k+1)}(n)$ is defined by the following equation:

$$V^{(k+1)}(n) = \begin{cases} V^{(k)}(n) & se\ V^{(k)}(n) < E[V^{(k)}(n)] \\ E\left[V^{(k)}(n)\right] & altrimenti \end{cases} \qquad \text{eq. 2}$$

   where $E[V^{(k)}(n)]$ is the average of $V^{(k)}(n)$.

3. You repeat step 2, so as to obtain the measure named *Second Derivative Round Mean* (SDRM), i.e. $E[V^{(2)}(n)]$.

4. You determine the voiced rate in the following speech:

$$p = \frac{Lv}{L} \qquad \text{eq. 3}$$

   where Lv is the length of the regions of $V^{(2)}(n)$ when $V^{(2)}(n) = V^{(1)}(n)$ and L is the length of the input signal.

5. The value of the threshold is included in the range:

$$\left[\frac{E\left[V^{(2)}(n)\right] + E[V^{(3)}(n)]}{2}, p\frac{E\left[V^{(2)}(n)\right] + E[V^{(3)}(n)]}{2}\right]$$

6. Finally, the adaptive threshold value of each frame can be calculated with the aid of the following equation:

$$AWT(i) = \begin{cases} Max\left(Frame(i)\right) + 0{,}1 & se\ Max\left(frame(i)\right) < Noise\_dis(n)] \\ E\left[V^{(2)}(n)\right] + E\left[V^{(3)}(n)\right] & otherwise \end{cases}$$

where AWT(i) is the adaptive threshold value of each frame i, while Frame (i) and Noise_dis (n) are defined by the following relations:

$$\text{Frame(i)} = [V((i-1)*\text{Num}+1), V(i*\text{Num})]$$

$$\text{Noise\_dis(n)} = p\,\frac{E\left[V^{(2)}(n)\right]+E\left[V^{(3)}(n)\right]}{2}$$

where Num = 5 in the implementation used.

3.4.2.4 Algorithm based on Discrete Wavelet Transform and Teager Energy Operator

The second algorithm that we want to implement is the one proposed Wu and Wang[5]. It is based on the use of discrete wavelet transform on the Teager Energy Operator (TEO). The following describes the steps.

Discrete Wavelet Transform

The wavelet transform is based on an analysis of the signal in time and frequency. This analysis adopts a technique of glazing with regions of variable size. It allows the use of long intervals of time where you want to obtain precise information at low frequency, and the use of shorter regions where one wants to have information to high frequency. The speech signals containing many components contain transitional and have the property of non-stationary. When using the properties of multi-resolution analysis of wavelet transform, it is necessary to have a better time resolution at high frequency range to detect transient components that vary rapidly when we require a better frequency resolution in the low frequency to take track in a precise manner of forming that vary slowly over time. Through the multi-resolution analysis, one can obtain a good speech classification in speech, or non-voice components transient. The coefficients of approximation and detail Aj Dj, at level j-th, of the input signal are determined using the quadrature mirror filters (QMF). The sub band signals A and D are the coefficients of approximation and detail and are obtained using the low-pass filters and high-pass, respectively, implemented using the Mother of Daubechies wavelet. In the implementation, we use the Daubechies wavelet of 4 points. Using the discrete wavelet transform, we can divide the speech signal into four not uniform sub-bands. The following figure uses three-level wavelet decomposition. The structure of wavelet decomposition can be used to obtain the periodicity in the most significant sub-bands.

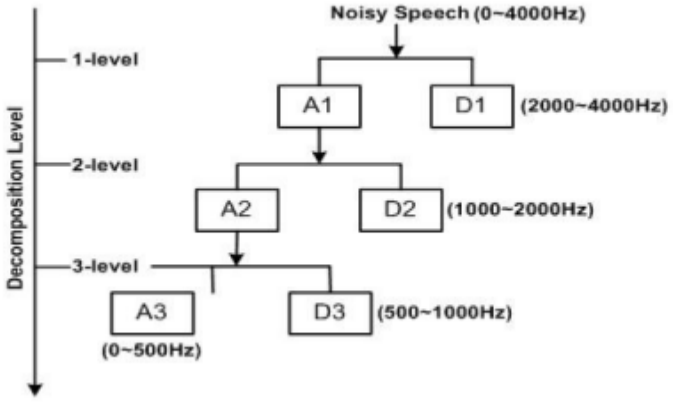

Figure 18 - 3-level wavelet decomposition using

[5] Kun-Ching Wang and Yi-Hsing Tasi, Voice Activity Detection Algo- rithm with Low Signal-to-Noise Ratios Based on Spectrum Entropy, 2008 Second International Symposium on Universal Communication

Teager Energy Operator

As previously mentioned, the operator TEO allows a better discriminability between speech and noise, and further suppresses noise components from the signals spoken noisy.

Moreover, the method of noise suppression based on the TEO can be implemented much more easily in the time domain respect to the traditional approach based on the frequency domain.

Calculation of SSACF

The auto-correlation function (ACF) used to measure the frequency of sequences of the sub-band signals is defined as

$$R(k) = \sum_{n=0}^{p-k} s(n)s(n+k), \quad k = 0, 1, \dots p$$

where p is the length of the ACF and k indicates the number of shift of the samples. This function will be defined here in the domain of the sub-bands and will be called the autocorrelation function for the sub-band signals (SSACF).

It can be deduced from the wavelet coefficients of each sub-band after applying the Teager Energy Operator. You may notice that the SSACF speech voice has more peaks than the spoken voice and not to white noise. In addition, for the spoken voice, the ACF has a frequency higher than the white noise, especially in sub-band A3.

Calculation of DSSACF and MDSSACF

To evaluate the periodicity of the sub-band signals, a method is used for each SSACF Mean-Delta. Initially, it uses a measure similar to the evaluation delta cepstrum to estimate the frequency of the SSACF, i.e. the autocorrelation function Delta for the sub-band signals (DSSACF) calculated as follows:

$$\bar{R}_M = \frac{1}{N_b} \sum_{k=0}^{N_b-1} |\dot{R_M}(k)|$$

where N_b indicates the length of the sub-band signal. The final parameter SAEis obtained by summing the our values of MDSSACF of sub-band signals. In fact, each of them provides information to extract in a precise manner the moments when there is voice activity.

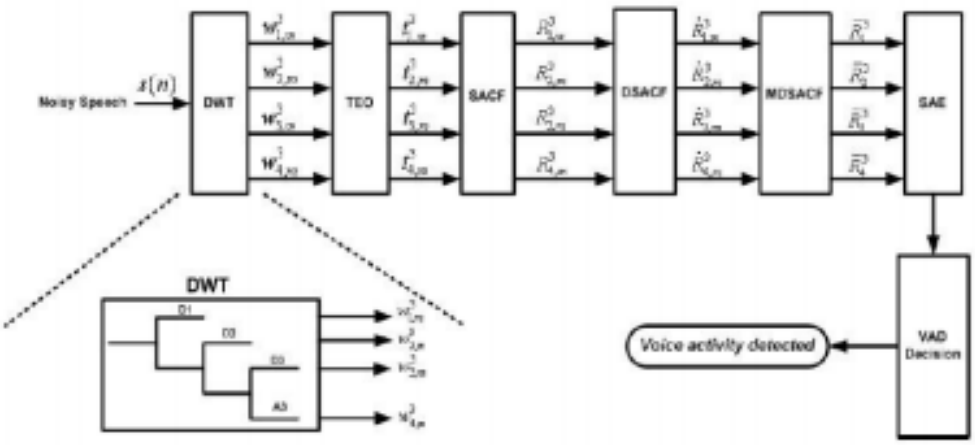

Figure 19 - Block diagram of the Voice verification algorithm

In the above figure you can see the block diagram of the algorithm of voice verification. For a given level of decomposition j, the wavelet transform decomposes the noisy speech signal in the j+1 sub-bands corresponding to the sets of Wavelet coefficients, called $w^{j}_{k,m}$, m. In this case, for the level j = 3:

$$w^{3}_{k,m} = DWT\{s(n), 3\} , n = 1, \ldots , N , k = 1, \ldots , 4$$

where w3k,m indicates the m-th coefficient of the k-th sub-band, while N corresponds to the length of the window. The length of each sub-band is N/2k.For example, if k = 1, w31,m corresponds to the sub-band signal D1.

By applying the operator TEO, you obtain:

$$t^{3}_{k,m} = \Psi d[w^{3}_{k,m}] , k = 1, \ldots , 4$$

The SSACF is obtained by calculating the energy of the signal $t^{3}_{k,m}$ in the following manner:

$$R^{3}_{k,m} = R[t^{3}_{k,m}]$$

where R [·] denotes the operation of auto-correlation. Next, we calculate the DSSACF by the relation:

$$\dot{R}^{3}_{k,m} = \Delta[R^{3}_{k,m}]$$

where $\Delta[\cdot]$ denotes the operation Delta. You obtain the MDSSACF by the relation:

$$\bar{R}^{3}_{k} = E[\dot{R}^{3}_{k,m}]$$

where E[·] denotes the operation of average. At last the SAE parameter is obtained from the relationship:

$$S\,AE = \sum_{k=1}^{4} \overline{R^{3}_{k}}$$

Voice verification decision based on adaptive threshold

To accurately determine the limit for the voice activity, the decision is usually made via threshold. To accurately estimate the noise characteristics that vary over time, we use an adaptive threshold value derived from the statistics of the parameter SAE during noisy frames, and the process of decision recursively updating the threshold using the mean and the variance of the values of SAE. Initially, we calculate the mean and the variance of the initial noise of the first five frames, assuming that these frames contain only noise. Are then calculated thresholds for speech and noise through these relationship:

$$T_s = \mu_n + \alpha_s \cdot \sigma_n$$

$$T_n = \mu_n + \beta_n \cdot \sigma_n$$

where T_s and T_n indicate the threshold of speech and the noise floor, respectively. Similarly, μ_n and σ_n indicate the mean and the variance of the values of the function SAE, respectively. The decision rule is defined as:

if ($S\,AE$(t) > T_s) then $V\,AD$(t) = 1

otherwise if ($S\,AE$(t) < T_n) then $V\,AD$(t) = 0

otherwise $V\,AD$(t) = $V\,AD$(t − 1)

If the detection result is a noisy period, the mean and variance of the values of SAE are thus updated:

$$\mu_n(t) = \gamma \cdot \mu_n(t - 1) + (1 - \gamma) \cdot SAE(t)$$

$$\sigma_n(t) = \sqrt{\left[S\,AE^2_{buffer}\right]mean - \mu_n(t)^2}$$

$$[SAE^2_{buffer}]\text{mean}(t) = \gamma \cdot [SAE^2_{buffer}]\text{mean}(t-1) + (1-\gamma)\cdot SAE(t)^2$$

where [SAE2buffer]mean(t−1) is the average value in the memory of the SAE in a frame containing only noise. The thresholds are then updated using the mean and variance of the current values of the SAE. The two thresholds are updated only during periods of inactivity voice, and not during periods of voice activity.

Setting of the voice verification parameters

The algorithm parameters to be tested in this project will take approximately the following values:

- dimensions of frame=256 samples per frame
- $M = 8$
- $\alpha_s = 5$
- $\beta_n = -1$
- $\gamma = 0.95$

4 Power Node

4.1 Power Node SPD – Surveillance and anti-tampering

Surveillance and anti-tampering are two key capabilities of many real world systems. Surveillance mechanisms are typically implemented using visual and/or infrared cameras. They are commonly utilized to monitor an asset or area under protection and to detect threats or hazardous situations through either operator visual inspection or image processing algorithms that typically require high performance processing nodes. Anti-tampering mechanisms, on the other hand, aim primarily to protect the embedded system itself by detecting any attempt for compromising system integrity and, sometimes, by applying countermeasures such as the emergency erasure of flash memories containing important data (cryptographic keys for example). Anti-tampering mechanisms are based both on traditional technologies such as secure packaging and use of seals as well as on sensors detecting the unjustified alteration of a periodically measured physical attribute, such as the electrical resistance of a protective enclosure.

In the context of nSHIELD, ISD aims to explore a novel acoustic based technology that can be used for monitoring and protecting both assets as well as the embedded system itself. Acoustic based systems have been in use for decades in underwater environments in the form of passive and active sonars. Recently, a lot of effort has targeted the exploitation of the potential of acoustic based systems as a technology that can complement camera based inspection. The main advantages of acoustic based technology, compared to camera based inspection, are the following:

- Sound processing is computationally much less demanding than image processing.
- There are no “out-of-view” areas due to obstacles or due to overcrowding and the data acquired are not affected by variable illumination conditions.
- The reduction in cost and in processing power enables building systems with a really large number of acoustic sensors.

On the other hand acoustic processing faces many challenges mainly due to the interference of background sounds. That’s why most commercial applications target abnormal events that can be easily extracted from the ambient noise of the environment, such as gunshots [1]. In this type of applications a geographically distributed network of sensors is used to constantly process sound data, perform a detection of a gunshot acoustic signature, and use the known relation is sensors’ placement to compute the rough area in which the event occurred [2].

Most research efforts in the area target the efficient detection and classification of the abnormal events of interest [3, 4]. However not much attention has been placed on key system features that can simplify processing tasks and hopefully boosts the widespread usage of acoustic based detection. Typically, a single sensor is used, or a network of sensors that are geographically distributed and interact in a loosely coupled fashion.

In the context of nSHIELD, ISD will develop a novel audio based surveillance system that aims to overcome the most important limitations of non-military grade systems currently utilized in acoustic based research providing correlated data acquisition from a large number of overlapping sensors.

More specifically, the system will be able to interface hundreds of hardware synchronized microphones and transfer the combined audio stream to memory in real time. It will be the only acoustic based system with sensors hardware synchronized, meaning there is neither time difference nor time drift among samples captured by different sensors. This feature dramatically simplifies data correlation for two reasons. First, by correlating the synchronized samples from multiple sensors placed at known locations and by taking advantage of the system’s inherent redundancy to make sure that sound sources are captured by multiple sensors, applications will be able to simplify ambient noise extraction, to perform detection of irregular acoustic events using simple peak detection and to identify the direction of any threat using triangulation. When sensors are not hardware synchronized, their sampling rates are controlled by individual crystal clocks, and they drift. This makes it difficult to accurately correlate sensor

data. Moreover, at high speed motion applications (railway case) even a few milliseconds of difference in the time domain result in significant difference in the space domain, making difficult to estimate the relative positions of the microphones at their points of capture.

The data acquisition will be performed by an FPGA-based board that will collect the data samples and will deliver them to a processing unit over a standard PCI family bus. Interfacing the final product to the end application will be performed by software running on an embedded PC. For demonstration purposes, in the context of the project, a standard PC may also be used to interface the prototype board.

4.1.1 References

[1] C. Clavel, T. Ehrette, and G. Richard, “Event detection for an audio-based surveillance system,” in IEEE International Conference on Multimedia and Expo, Amsterdam, July 2005.

[2] http://www.shotspotter.com.

[3] S. Ntalampiras, I. Potamitis and N. Fakotakis, "On acoustic surveillance of hazardous situations", IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.

[4] Jean-Luc Rouas, Jérôme Louradour and Sébastien Ambellouis, "Audio Events Detection in Public Transport Vehicle", in "9th International IEEE Conference on Intelligent Transportation Systems (ITSC'2006).

4.2 System of Embedded System - SoES

Nowadays new interesting trends are characterizing the Embedded System field, in particular, they are even more physical and logically interconnected. These trends, of *hyper connectivity*, are driven by multiple needs, first is the gain of amount and the complexity of the services provided by the system, and second the growing interest around the distributed embedded. In such scenario we identify a new system typology commonly named System of Embedded System or simply Large System.

The SoES are usually large and composed by nodes that are heterogenic and independent. Such heterogeneity is mostly related to the nature of service provided by each node. The independency of nodes, instead, is strictly related to their autonomous evolution. Therefore, according to these considerations, the problem arises is about the design and the development of a large-scale system with specific SPD constraint and value. These two activities are extremely complex and the solution is generally very expansive. To simplify the design process, the idea is to use the concept of reusability and composability. We assume that is possible to put together components that are SPD compliant and obtain a System that is SPD compliant de facto. In this context, the main objective is to develop a new subcomponent that can be easily integrate into the every single node of the system without have to restructure architecture and ensuring the overall SPD properties of the SoES. This methodology is strongly related to pSHIELD project, where the composability criteria of the architecture design were defined. Therefore, following the identified mechanisms in pSHIELD, it will be developed a new methodology to be valid during the system design and system re-design.

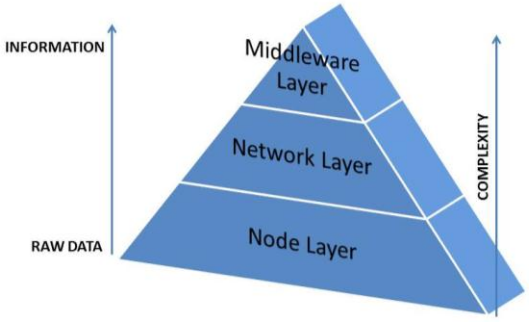

Figure 20 - Architecture Layers

A possible solution to create such gain factor is based on the development of a sub-component able to provide high functionality level of integration and that will be characterized by a set of properties that can confer Security, Privacy and Dependability to the entire architecture in which it will be integrated. This sub-component will be developed as a custom IP to simplify the integration and the adaptation of independent nodes at the lower level of the architecture.

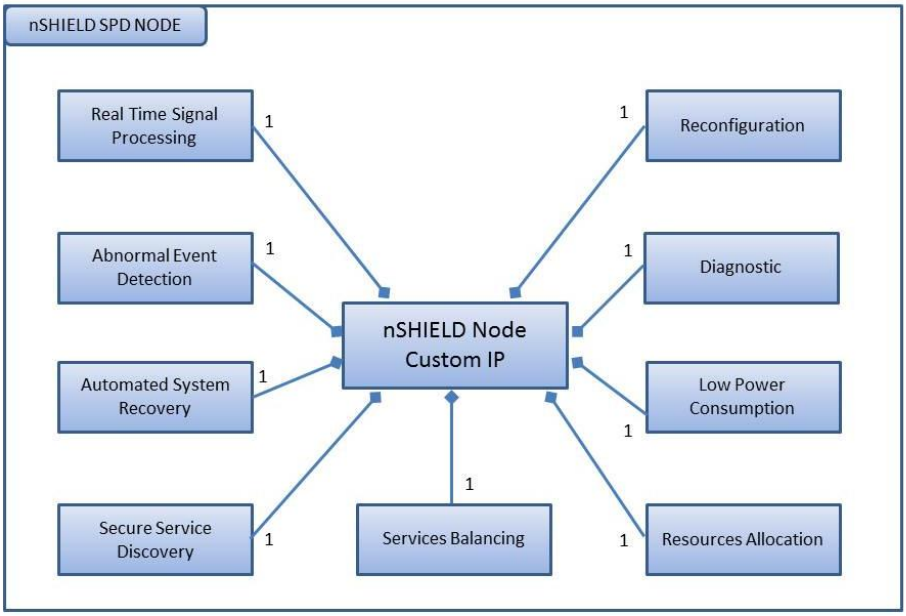

Figure 21 - Custom IP core

The custom IP will be providing functionalities and capabilities as follow:

- Real time signal processing: to meet the stringent time constraint of which critical systems are characterized;
- Abnormal event detection: to identify anomalous operations that can cause failures and/or can make the entire system insecure;
- Automated system recovery: to resume correct system functionalities after a fault;
- Secure service discovery: to provide the secure sub-components integration in the SPD node with a module identification;
- Services balancing: to ensure the system reliability with a dynamic load distribution on more components;
- Resources allocation: giving to the entire system the benefits of using more resources in the critical operations where a more computational power is required;
- Low power consumption: to reduce the contribution of the power consumption;
- Diagnostics: to provide a diagnostic service to the entire system with a status information exchanging;
- Reconfiguration: providing different functionalities through a dynamic reconfiguration of the custom IP to manage different operating modes.

The integration of this custom IP in the pre-existing embedded system will be facilitated providing discovery and composability features. In addition, the Dependability of the network, which the IP is part of, will be improved by detecting abnormal events and recovering the system functionalities by a FPGA reconfiguration; the redundancy of hardware components will also assure the Dependability of the entire node architecture. Regarding the Security, the custom IP will receive encrypted data by others

components and it will be able to decrypt it; another security aspect will be guaranteed by the diagnostic and self-reconfiguration property. The custom IP will be able to manages a dynamic resources allocation by a service balancing to assure a real time signal processing in every working condition, this is one of the most important requirements in some scenarios (i.e. the Avionic System), due to critical time constraints that have to be met.

At this architectural level, the heterogeneous nature of nodes makes the developing of a software support necessary for services exchanging with upper layer components. For this reason, in conjunction with the custom IP core, a software library will be developed to distribute SPD aspects as functionalities to microprocessor layer.

4.3 Power node for Avionics System

This section illustrates the Selex Galileo approaches to define the concept of the “new” Avionics Architecture by defining, developing and validating the “avionics module”. The way of built with this different “avionics node” through the SPD features and functionalities will be part of to the Dependable Avionics scenario (WP7). These technologies will be implemented in embedded system prototypes that will be part of the nSHIELD demonstrators.

The nSHIELD drivers are to provide a scalable solution, to define a minimal set of nodes, to increase the number of supported function and to demonstrate fault tolerance and reconfiguration and so the high dependability of the SG for nSHIELD solution. An “avionics node” can be built with different pieces of the following part:

- HW
- SW

4.3.1 Current System Configuration

The current implementation of a typical avionic system is mainly based on a Federated Architecture where several LRUs with remote processing capability and dedicated I/Os provides the requested functionality.

All the avionics functionalities such as

- Navigation Management (including Guidance and Flight Management)
- Communication (including also identification)
- Plant Management
- Flight Control
- Payload/Sensors Management

are based on an HW and SW Platform which usually differs between them according to the application where are used.

The trend for modern aircrafts is to support aircraft application with an Integrated Modular Avionic (IMA) platform. Platform that have to include always more functionality, starting from the navigation/mission functionality to the flight control function including also communication functionality.

To implement all the functionalities required therefore the IMA platform should be reconfigurable, reconfigurable means that IMA should be able to change the configuration of the avionic platform by moving application

Re-configuration should therefore improve the operational reliability of the aircraft while preserving (or improve) current levels of safety (aircraft systems have to enforce stringent safety requirements that address the effects of failures on the life of passengers). Operational reliability strong addresses the effect of failures on economic aspects of flight operations

Avionics systems rely on computing platforms, and these platforms must be designed to provide the required levels of safety, maintenance, overall flight functions (flight management, mission and navigation). The avionics functions must be sustained appropriately in order to ensure a safe flight, yet the hardware components on aircraft operate in a hazardous environment while running software that itself might contain defects.

Many techniques have evolved for constructing dependable computer platforms for avionics systems. Architectures have been developed that use various forms of redundancy and reconfiguration to allow continued operation when components fail. In addition, in many cases replicated components are separated within an airframe to prevent their simultaneous loss in the event that there is damage to the airframe. Various techniques are employed to aid in the correct construction of software, and software development is required to follow a rigorous process. Finally, various analysis techniques can be used to estimate some of the important probabilities related to dependability of computing platforms.

A dependable avionics system, therefore, is one that can be trusted to support safe aircraft operations.A dependable avionics system needs to include the following attribute: Reliability, Availability, Safety, Confidentiality, Integrity, and Maintainability. This means that the avionics system shall be nSHIELD compliant (SPD features satisfied).

The general dependability requirements for an avionics system/sub-systems, the SPD NODEs, shall include the nSHIELD SPD attributes.

4.3.2 Distributed configuration

With the IMA architecture (or distributed architecture) the avionics computer that implement flight management or flight control functions, needs to take in considerations also data confidentiality and data integrity, that are becoming increasingly important with increasing interconnectedness of dependable systems (i.e. also the communication control function: wide/narrow data link management system)

Either in an IMA or distributed architecture the fundamental components of avionics system architecture are computers (i.e. HW), data busses (i.e. Network) and the application (i.e. either SW services and/or the real application SW). These components may be configured in various computing system architectures, where the purpose of architecture is to meet the functional demands of the computing platform and the dependability requirements.

The key objective in moving towards a distributed Integrated Modular Avionic (IMA) architecture is to realize a new platform Hardware and Software based.

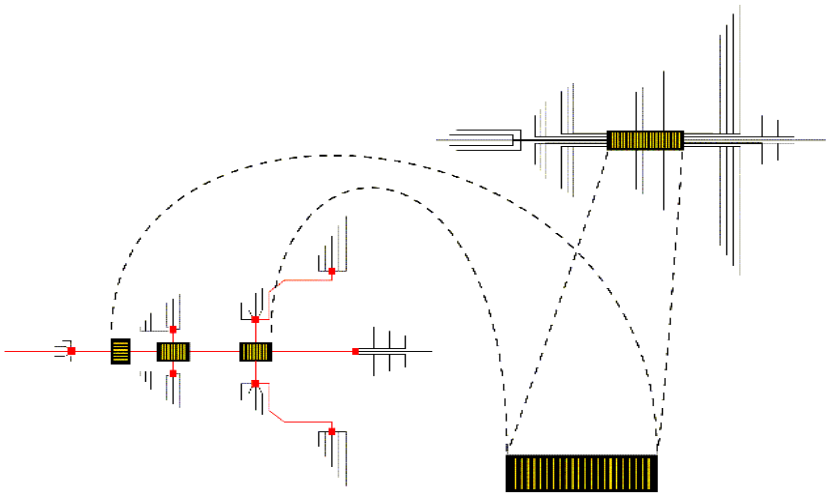

Figure 22 - Modular Avionics Architecture to/from SELEX GALILEO Distributed Modular Avionics Architecture

The following figure shows a possible application/scenario for the Distributed Modular Avionics Architecture. The functionalities are distributed through the “computer #1”, “computer #2” and “computer #3”. The three computer have to be compliant the nSHIELD solution for Dependability, Security and Privacy.

The source/sink box represents the following equipment

- EDU: Electrical Distribution Unit
- WBDL: WideBand Data Link
- SATCOM: Satellite Communications
- FMS: Flight Management System
- A/C : AIRCRAFT
- NSU: Navigation Sensor Unit

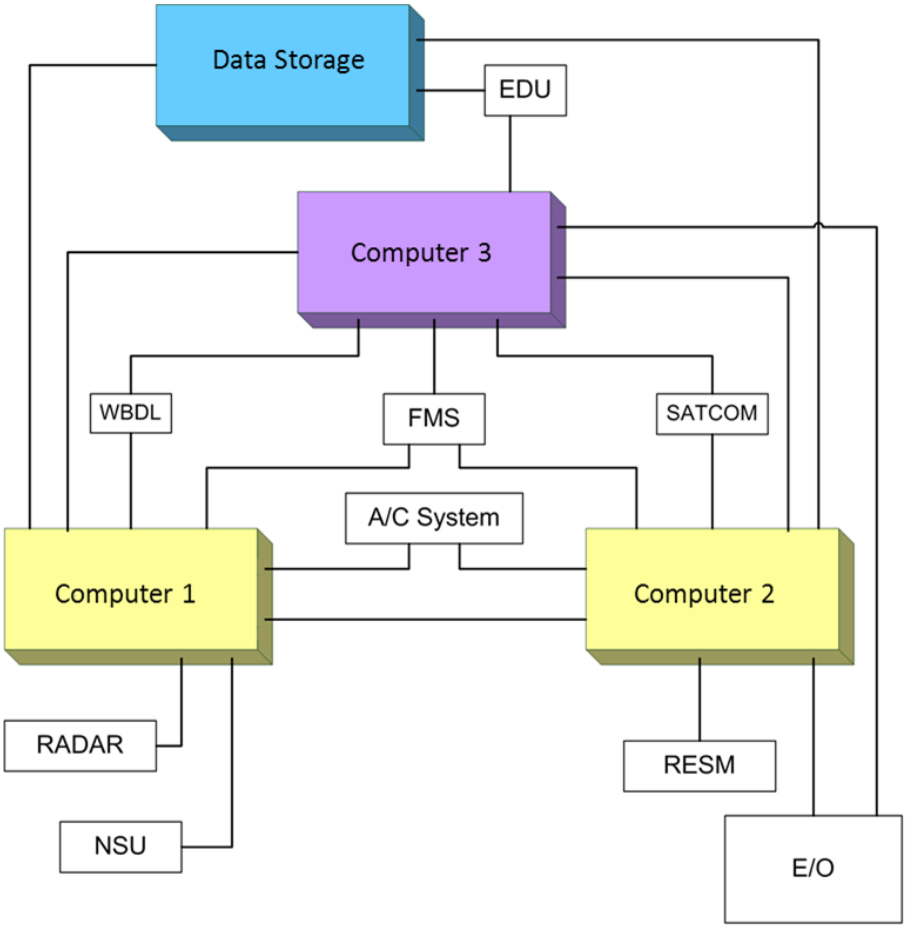

Figure 23 – Selex Galileo Distributed Modular Avionics Architecture for Surveillance System

5 Dependable self-x Technologies

This task will provide horizontal SPD technologies that will be adopted in task 3.1-3.2-3.3 at different levels, depending on the complexity of the node and considering its HW/SW capabilities, its requirements and its usage. The research will rely mainly on the technologies described in the following sections.

5.1 Introduction

In computer networking: "**Resilience"** is the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation." Threats and challenges for services can range from simple misconfiguration over large scale natural disasters to targeted attacks. As such, network resilience touches a very wide range of topics. In order to increase the resilience of a given communication network, the probable challenges and risks have to be identified and appropriate resilience metrics have to be defined for the service to be protected.

These services include:

- supporting distributed processing
- supporting networked storage
- maintaining service of communication services such as
  - video conferencing
  - instant messaging
  - online collaboration
- access to applications and data as needed

5.1.1 Applications

Resilient networks are mainly focused about four application fields:

- dependable surveillance systems for urban railways security,
- dependable system for voice/facial recognition,
- dependable avionic system
- social mobility and networking dependable system

Above mentioned application scenarios correspond to future product and services markets that are expected to exhibit fast growth rates due to socio-economic trends.

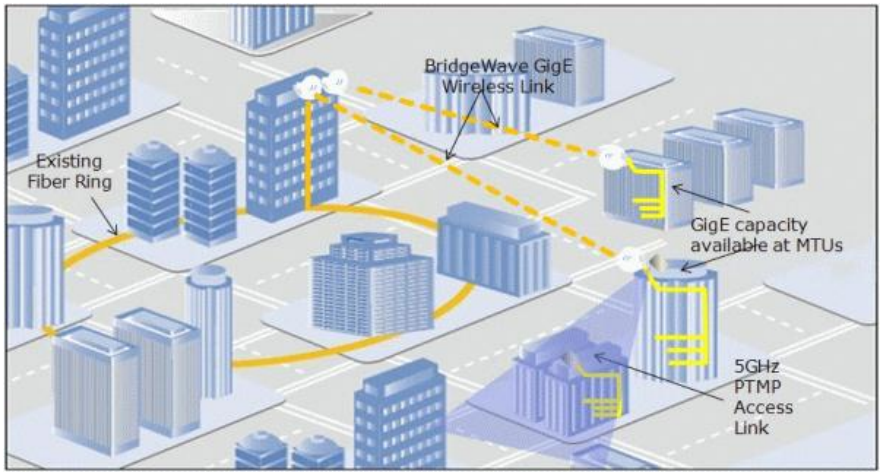

Figure 24 - Resilient network example

5.1.2 Literature

Below are summarized several works done about resilient networks.

Note that one of the most popular attacks to the networks are **DoS** or **DDoS** attacks (see also paragraph 5.2).

A **denial-of-service attack** (**DoS attack**) or **distributed denial-of-service attack** (**DDoS attack**) is defined as an attempt to make a computer or network resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS attack may vary, it generally consists of the concerted efforts of a person, or multiple people to prevent an Internet site or service from functioning efficiently or at all, temporarily or indefinitely.

Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root name servers. The term is generally used relating to computer networks, but is not limited to this field; for example, it is also used in reference to CPU resource management.

One common method of attack involves saturating the target machine with external communications requests, such that it cannot respond to legitimate traffic, or responds so slowly as to be rendered effectively unavailable. Such attacks usually lead to a server overload. In general terms, DoS attacks are implemented by either forcing the targeted computer(s) to reset, or consuming its resources so that it can no longer provide its intended service or obstructing the communication media between the intended users and the victim so that they can no longer communicate adequately.

Denial-of-service attacks are considered violations of the IAB's Internet proper use policy, and also violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations.

When the DoS Attacker sends many packets of information and requests to a single network adapter, each computer in the network would experience effects from the DoS attack.

The work reported in [1] describes very well the weakness of Mobile Ad Hoc Network (MANET), in particular:

1. **Multi-hop communications**: The communication in MANET between any two remote nodes is performed by numerous intermediary nodes whose functions are to relay data-packets from one point to another. Thus, ad hoc network requires the support of multi-hop communications
2. **Constrained Resources**: Generally, most MANET devices are small hand-held devices ranging from personal digital assistants (PDAs) and laptops down to cell phones. These devices indeed have limitations because of their restricted nature; they are often battery-operated, with small processing and storage facilities.
3. **Infrastructure less**: MANETs are formed based on the collaboration between autonomous nodes, peer-to-peer nodes that need to communicate with each other for special purpose, without any pre-planned or base station.
4. **Dynamic Topology**: MANET nodes are free to move, hence the connectivity between nodes in MANET can change with time, because nodes can move arbitrarily; thus the nodes can be dynamically inside and outside the network, constantly changing their links and topology, leading to change in the routing information all the time due to the movement of the nodes.
   Therefore, the communicated links between nodes in MANET can be bi-directional or unidirectional.
5. **Limited Device Security:** MANETs devices are usually small and can be transported from one place to another, then they are not constrained by location. Unfortunately, as a result these devices could be easily lost, stolen or damaged.
6. **Limited Physical Security:** Generally, MANETs are more susceptible to physical layer's attacks than wired network; the possibility of spoofing, eavesdropping, jamming and denial of service (DoS) attacks should be carefully considered. By contrast the decentralized nature of MANET makes them better protected against single failure points.

7. **Short Range Connectivity:** MANETs rely on radio frequency (RF) technology to connect, which is in general considered to be short range communication. For that reason, the nodes that want to communicate directly need to be in the close frequency range of each other. In order to deal with this limitation, multi-hop routing mechanisms have here fore to be used to connect distant nodes through intermediary ones that operate as routers.

The main security requirements of MANETs are also described in [1]:

1. **Authentication**: Authentication is essential to verify the identity of every node in MANET and its eligibility to access the network. This means that, nodes in MANETs are required to verify the identities of the communicated entities in the network, to make sure that these nodes are communicating with the correct entity.
2. **Authorization and Access Control**: Each node in MANET is required to have the access to shared resources, services and personal information on the network. In addition, nodes should be capable of restricting each other from accessing their private information. There are many techniques that can be used for access control such as Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role Based Access Control (RBAC).
3. **Privacy and confidentiality**: Each node has to secure both the information that is exchanged between each other; and secure the location information and the data stored on these nodes. Privacy means preventing the identity and the location of the nodes from being disclosed to any other entities, while confidentiality means keeping the secrecy of the exchanged data from being revealed to those who have not permission to access it.
4. **Availability and survivability**: The network services and applications in MANET should be accessible, when needed, even in the presence of faults or malicious attack such as denial-of service attack (DoS). While survivability means the capability of the network to restore its normal services under such these conditions. These two requirements should be supported in MANET.
5. **Data integrity**: The data transmitted between nodes in MANET should be received to the intended entities without been tampered with or changed by unauthorized modification. This requirement is essential especially in military, banking and aircraft control systems, where data modification would make potential damage.
6. **Non-repudiation**: This ensures that nodes in MANET when sending or receiving data packets should not be able to deny their responsibilities of those actions. This requirement is essential especially when disputes are investigated to determine the misbehaved entity. Therefore digital signature technique is used to achieve this requirement to prove that the message was received from or sent by the alleged node.

One important issue is the definition of the metrics useful to define the networks grade of resilience.

[3] Gives a set of network properties that are broadly classified in six categories, as shown in the below table, in order to define the metrics used to quantify resilience for most network scenarios.

Table 6 - Networks properties

Density	number of nodes, area of spread, distribution pattern, rate of topology change
Mobility	velocity of the node, mobility model, predictability
Channel	capacity distribution, propagation model, bit error rate, error rate model
Node resources	electrical power, computing power, memory, tx/rx power, location awareness
Network traffic	distribution, packet size, source/sink placement, QoS
Derived properties	degree of connectivity, propagation delay, queuing delay, node willingness

5.1.3 State of the art

There are existing approaches in security which have been applied to MANETs are for example using traditional cryptographic solutions based on public key certificates to maintain trust, in which a Trusted Third Party (TTP) or Certificate Authority (CA) certifies the identity associated with a public key of each communicated entities.

Solutions focused on message confidentiality, integrity and non-repudiation, they do not consider however the trust management of the communicated entities, and how these certified entities act is left to the application layer.

There are solution based on behavior detection algorithm combined with threshold cryptography digital certificates to satisfy prevention and detection to securely manage Mobile Ad hoc Network.

Different approach based on protecting the packets sent between nodes by choosing the secure routing path to the destination node based on the redundancies routes between nodes to maintain the availability requirement.

Securing the routing in mobile ad hoc network (MANET) has also been given much attention by the researchers; many approaches, therefore, have been proposed to deal with external attack. Also in this scenario different approach where studied.

There are approaches to protect the packet sent to multi receivers by using keyed one-way hash function supported by windowed sequence number to ensure data integrity.

[1] Proposes an approach based on Discretionary Access Control (DAC) to ensure data confidentiality and privacy of the originator node in MANETs. In this scenario nodes sent with the transmitting packets privacy information used to the receiving node to know if there are and which are nodes in the network allowed to receive the packet too.

[2] Points out the attention about mobile network for disaster recovery like natural disaster like hurricanes or terrorist attach like 9/11 one; but we also can think to the Fukushima atomic site disaster. In such cases the needful are: find and rescue possible people in trouble, identify new incoming risks, keep communications between the people involved in rescue actions also if the communication with the Headquarter is temporary down.

The proposed solution is a distributed and flat architecture with respect to a centralized and hierarchical architecture. Indeed, a number of radio resource and mobility management functionalities, traditionally performed by central controllers, now have to be distributed across the network elements. The network is based on auto-configurable systems with a fully integrated service architecture that can be deployed as a single node solution for local communication or be configured to operate as an ad hoc network of nodes.

Figure 25 shows the network architecture:

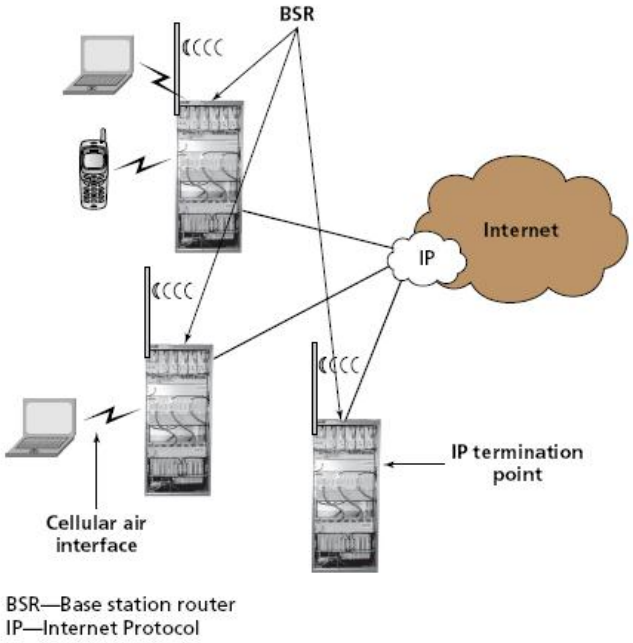

Figure 25 - Distributed and flat architecture

The key features of such a network are:

1. *Simplicity.* Integrating functionalities leads to deployment of fewer network element types, reduced maintenance and troubleshooting, and significant concomitant cost reductions.
2. *Flexibility.* Single network architecture can be employed and managed independently of the air interface technologies being used. Furthermore, the architecture is amenable to different deployment scenarios, including macro cells for wide area coverage, micro cells for hotspot coverage, and pico cells for in-building coverage.
3. Scalability. Because of the absence of central controller elements, the architecture can easily be scaled to the required size. In other words, the deployment of additional access points or base stations does not entail the deployment of additional central controllers and a possible redesign of the RAN.
4. Interoperability. The proposed architecture essentially decouples the evolution of the air interface technology and the network infrastructure. In other words, the evolution of the air interface is not hampered or tied to the network infrastructure, and vice versa. Through the use of standardized IP interfaces, inter-operability between different networks (possibly deployed by different emergency response agencies and first responder units) is achieved.
5. Performance. The integration of different network functionalities leads to the collapse of the protocol stack in a single network element and thereby eliminates transmission delays between network elements and reduces the call setup time and packet fragmentation and aggregation delays. Furthermore, the ability to implement cross-layer optimizations provides additional performance enhancements and resulting capacity gains. Finally, local communication between mobile terminals connected to the same base station is optimized through direct routing at the base station router.

A different problem for Mobile ad Hoc Networks is related to malicious attack from eavesdropping nodes. In this scenario is important that resilient networks maintain an acceptable channel throughput. [4] Shows different mathematical model to characterize Byzantine adversary in different network scenario from encoding transmission point of view. A Byzantine attacker is a malicious adversary hidden in a network, capable of eavesdropping and jamming communications.

5.1.4 The Wireless Sensor Network specific example

A specific set of MANETs: the Wireless sensor networks. Fault detection in this kind of networks is well described in [5].

Wireless sensor networks can be organized in two main scenarios: fully-distributed and hierarchical models.

Distributed model encourages sensor nodes to self-manage themselves: the more decision a node can make affects the less number of communication messages need to be delivered to the base station. In particular, neighbor coordination is a typical example of fault management distribution. Nodes coordinate with their neighbors to detect the suspicious node before consulting with the base station.

In a hierarchical architecture there is the possibility either for the self-fault detection node scenario or for the passive approach.

The passive approach consists in demanding to the Cluster Head or to the Group Coordinator the node fault detection.

For example a possible fault scenario of a node is the battery expiration: in a self-fault detection model the node is responsible for sending to the Base Station (or to the Group Coordinator) information about its low battery level. In a passive model if the Group Coordinator does not receive messages from the node for a defined time interval it can suppose that the node is power off. Of course the choice about the two models can be affected by the acceptable risk level of losing a node.

5.1.5 Market solutions

Allied Telesis: Building Resilient Networks
(http://www.alliedtelesis.com/resources/literature/literature.aspx?id=5)

These kind of products (switch, Router, ect…) implements the "VCStack Solution" (Virtual Chassis Stacking), with high bandwidth and configurable layout, that implements algorithms like the X-ring function in order to build resilient networks (http://www.lantechcom.tw/global/eng/Download/ePaper/X-Ring.pdf).

In the X-Ring topology, every switch should enable X-Ring function and assign two member ports in the ring. Only one switch in the X-Ring group would be set as a backup switch that would be blocked, called backup port, and another port is called working port. Other switches are called working switches and their two member ports are called working ports. When the failure of network connection occurs, the backup port will automatically become a working p y g port to recovery the failure.

The ring master can negotiate and place command to other switches in the X-Ring group. If there are 2 or more switches in master mode, then software will select the switch with lowest MAC address number as the ring master. The X-Ring master ring mode will be enabled by the X-Ring configuration interface. Also, user can identify the switch as the ring master from the R.M. LED panel of the LED panel on the switch.

The system also supports the coupling ring that can connect 2 or more X-Ring group for the redundant backup function and dual homing function that prevent connection lose between X-Ring group and upper level/core switch. Figure 26 shows the X-Ring algorithm.

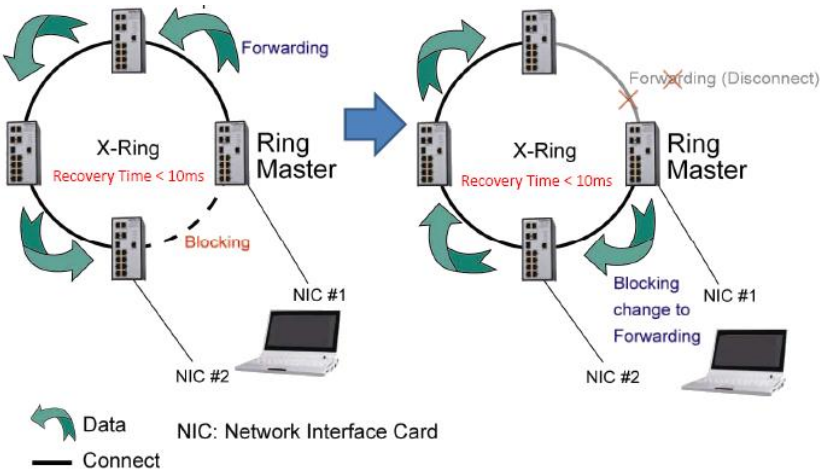

Figure 26 - X-Ring algorithm

Another market example is ExtremeXOS Operating System (www.extremenetworks.com). It is a modular, time hardened, extensible network operating system for robust, high performance networks. ExtremeXOS is built on a high availability architecture with rapid failover features such as Ethernet Automatic Protection Switching (EAPS), which helps reduce network downtime and ensure access to mission-critical applications such as CRM, data warehouses and VoIP for carrier and voice grade networks.

Its main features are:

- Memory protection for processes
- Self-healing process recovery via process restart or hitless failover
- Dynamic loading of new functionality
- Scriptable CLI for automation and event-triggered actions
- XML open APIs for integrating third-party applications
- Dual-stack IPv4 and IPv6 support
- Extensibility
- Integrated Security
- Modular Operating System
    - Preemptive scheduling and memory
    - process monitoring and restart processes that have become unresponsive can be automatically restarted.
    - allows applications, including security stacks such as SSH and SSL, to be upgraded while the switch is running, which reduces downtime due to updates which leads to higher availability
- Capability of preserving the state of resiliency and security protocols such as STP, EAPS and Network Login, thus allowing hitless failover between management modules/redundant masters in case a module or master fails.
- Capability to restart without disrupting traffic forwarding.
- Possibility to update the static routing table after restart incrementally.
- CPU Denial of Service Protection.
- Extensibility
    - Dynamic Module Loading
    - Automating network administration through scripting.
    - XML Application Programming Interfaces
- Ease of Management
    - Link Layer Discovery Protocol

ExtremeXOS also provides a very high level integrated security:

Network Login supports three methods: 802.1x, Web-based and MAC-based. All methods can be enabled individually or together to provide smooth implementation of a secured network.

Dynamic security policies configuration: in order for instance to not allow connections after work time.

MAC Security: allows the lockdown of a port to a given MAC address and limiting the number of MAC addresses on a port.

IP security framework protects the network infrastructure, network services such as DHCP and DNS and even host computers from spoofing and man-in-the-middle attacks.

Identity Manager allows network managers to track users who access their network. User identity is captured based on Network Login authentication, LLDP discovery and Kerberos snooping

Secure management provides authentication and protection against replay attacks, as well as data privacy via encryption.

Resiliency Features: the Virtual Router Redundancy Protocol (VRRP) enables a group of routers to function as a single virtual default gateway.

5.1.6 References

[1] International Journal of Wireless & Mobile Networks (IJWMN) Vol. 4, No. 1, February 2012 DOI: 10.5121/ijwmn.2012.4117 225

Data Confidentiality in Mobile Ad hoc Networks

Hamza Aldabbas, Tariq Alwada'n, Helge Janicke, Ali Al-Bayatti
Software Technology Research Laboratory (STRL), De Montfort University,
Leicester, United Kingdom
{hamza, tariq, heljanic, alihmohd}@dmu.ac.uk

[2] **911-NOW: A Network on Wheels for Emergency Response and Disaster Recovery Operations**

David Abusch-Magder, Peter Bosch, Thierry E. Klein,
Paul A. Polakos, Louis G. Samuel, and Harish Viswanathan

[3] **Poster: Towards Quantifying Metrics for Resilient and Survivable Networks**

Abdul Jabbar Mohammad_‡, David Hutchison†, and James P.G. Sterbenz_†
_Information and Telecommunication Technology Center (‡student)
The University of Kansas, Lawrence, Kansas 66045–7612
Email: {jabbar,jpgs}@ittc.ku.edu
†InfoLab21, Lancaster University, Lancaster, LA1 4WA, UK
Email: {dh,jpgs}@comp.lancs.ac.uk

[4] **Resilient Network Coding in the Presence of Byzantine Adversaries**

S. Jaggi M. Langberg S. Katti T. Ho D. Katabi M. M´edard jaggi@mit.edu mikel@caltech.edu skatti@mit.edu tho@caltech.edu dk@mit.edu medard@mit.edu

[5] **Self-Managed Fault Management in Wireless Sensor Networks**

Mengjie Yu, Hala Mokhtar, Madjid Merabti
School of Computing & Mathematical Science
Liverpool John Moores University, Byrom Street, Liverpool, UK L3 3AF
M.Yu@2001.ljmu.ac.uk,H.M.Mokhtar@ljmu.ac.uk,M.Merabti@ljmu.ac.uk

5.2 Countermeasures against Distributed Denial of Service Attacks

5.2.1 Introduction

It is a fact that Distributed Denial of Service (DDoS) attacks have become one of the most difficult problems in the field of network security. DDoS attacks have the characteristic that they are very easy to be implemented and very difficult to be effectively stopped. Over the years, countless incidents of major DDoS attacks have been reported and they are string to be used as a blackmailing weapon against organizations and corporations that rely on network access and availability. Moreover the tools that are used to implement such attacks tend to be more and more sophisticated and automated, in such a degree that even simple individuals with little knowledge of network programming can use them to launch powerful DDoS attacks against major targets.

The basic idea behind DDoS attacks is to force a large number of individual systems connected to the Internet, to send bulk traffic to the same destination at the same time. The aggregated traffic that those systems produce can easily cripple the available network or system resources of the recipient. Thus the recipient, the victim, of this attack will no longer be able to have reliable network access or serve legitimate clients, if the victim is a network server.

In today's DDoS attacks, a small set of systems that are usually called "agents" control a vast amount of systems that are usually called "daemons" or "zombies". Those "zombie" systems will eventually launch the attack when instructed by the agents. The attacker, in order to be able to launch an effective DDoS attack, needs a large number of compromised systems that will act as "zombies". This large number of systems can be obtained by any hacking procedure. The most popular way though, is the use of Internet worms. Those worms can infect a very large number of systems in a matter of hours, which can be used for the DDoS attack. We should note here that most of the recent worms that have been discovered do not cause actual damage to the infected system but almost all of them install some kind of backdoor to the systems or an actual application that can be used for a DDoS attack.

One unique characteristic of DDoS attacks, which makes them so difficult to defend against, is that during the actual attack there is only "one way" connection with the victim and no confirmation of the reception of the packets or any other form of interaction between the "zombies" and the victim is needed. This, unlike any hacking attempts that need to establish a "two way" connection with the victim, gives DDoS attacks the major advantage of being more or less completely untraceable. Due to the lack of any form of interaction between the "zombies" and the victim, the packets of a DDoS attack, produced by the "zombie" systems, do not contain the true source IP address thus there is no obvious or simple way to know the true sources of the DDoS attack traffic. Moreover there is no simple way to distinguish the attack traffic from the traffic produced by legitimate clients.

The defence against a DDoS attack is a two steps process. At first, the victim has to identify whether it is subject to an on-going DDoS attack or experiences a sudden bandwidth overload due to other reasons. Various methodologies have been proposed that can identify the existence of an on-going DDoS in a network or system. Those methodologies rely on installed IDS systems and use pattern recognition, trained neural networks or other methods to identify the characteristics of DDoS attack traffic. Those systems can alert the victim but cannot take actions to prevent the attack. In the second step, the victim has to take some form of countermeasures in order to stop the on-going attack and prevent if possible further attacks from the same source. There are many proposals on countermeasure systems against DDoS attacks and all of them can to prevent or at least limit the impact of such an attack. All those systems have innate limitations that prohibit them to be identified as complete solutions to the problem. The main approach of countermeasure systems is to try and trace the attack back to its sources. This way enables the victim to the packets originating from those sources and provides its services to the other legitimate users.

In this section we will present some of the major methodologies that have been proposed so far and we will discuss their drawbacks along with some future trends in this field

5.2.2 Traceback

The most common approach in order to effectively defend against a DDoS attack is to try to identify the sources of this attack. This is a very difficult task due the reasons we have mentioned in the introduction but not impossible. The fact that the source IP address is not a reliable source of information, made the researchers to explore different ways to identify the true sources of an incoming attack.

One of the first approaches, tried to eliminate the phenomenon of false a.k.a. “spoofed” source IP addresses. This can be achieved, if all the internet routers employ ingress filtering [8]. In ingress filtering, the router checks if an incoming packet in its ingress interface is valid for that interface. The validity of the packet is decided based on the information that the router has about the possible IP ranges that the incoming packets can have as source IP address. This method cannot be used in transit routers due to the wide range of possible IP addresses for each interface. It also poses a heavy computational burden to the routers due the additional lookups needed for this operation and the effectiveness of such a method relies heavily on the extent of its deployment.

One of the more interesting proposed methods for IP traceback is probabilistic packet marking (PPM) [13]. According to this method, the packets are marked, with low probability, while passing through the internet routers. This marking holds information about this particular part of the complete path of the packet. Using this method, the victim can identify the source of large steams of packets by combining the path information of different packets that belong to the same stream. In PPM the marking is being overwritten if another router along the path decides to mark the same packet. Thus, a large number of packets are needed in order to be able to identify the source. Although packet marking is a very promising approach in traceback, PPM has many limitations such as very large computational complexity during path reconstruction especially on highly distributed DoS attacks. It also suffers from the false marking phenomenon in which a sophisticated attacker can inject specially marked packets into the attack stream forcing the victim to reconstruct false paths. As a last remark, we could say that this marking scheme is not capable to provide real time filtering of the incoming packets because a large number of packets is needed in order to be able to identify the source of those packets.

A solution to the high computational complexity of PPM has been proposed by Song and Perrig in [16]. The proposed advanced marking scheme, using hash values of the edge fragments, achieves better precision i.e. less false positives and lower computation overhead during highly distributed DoS attacks. The drawback of this scheme is that it requires from the victim to have an updated map of all upstream routers.

Another extension of the PPM that significantly reduces the number of packets needed to be able to reconstruct the attack path has been proposed in [10]. In this scheme, additional packets are created during the marking procedure, thus resulting in higher network overhead. On the other hand, in [19] a router maintains a compensation table to record the information of marked packets which are remarked by this router. This results in the reduction of the needed packet for path reconstruction but the increase of the required memory capacity and computation overhead at the routers.

Another proposed solution to the computational overhead problem of PPM is based on an algebraic approach to IP traceback [7]. This approach is based on mathematical techniques used in error correcting codes in order to encode the path in multiple packets. The reconstruction algorithm of this approach is much more efficient ($O(n^{2.5})$) than the one in PPM ($O(n^{8})$). A more simplified algebraic marking scheme [6] combines the use of a map of all upstream routers with the current algebraic marking scheme and achieves not only greatly simplifies the path reconstruction procedure but also minimizes the false positives produced by this procedure. One major disadvantage of both algebraic marking schemes is that there is no authentication of the markings. Thus any compromised router could inject false markings in the stream and produce false results. The only scheme that is robust against false markings from compromised routers is the authenticated marking scheme [Song] that uses message authentication codes (MAC) and time-related chains between routers.

As we can see, a lot of work has been done in the field of IP traceback based on PPM. We can find one last very interesting proposal based on PPM in [1]. It shows a new marking technique which is effective even if the number of bits used for the marking is 1. It also shows that the number of packets needed for

path reconstruction increases exponentially with the path length but decreases doubly exponentially with the number of bits used for the marking.

A different marking scheme proposed in [3], requires all the routers to mark the traversing packets. It is called Deterministic Packet Marking (DPM). In DPM all the edge routers inject (mark) part of their IP address into each traversing packet. With the phrase “edge router” we mean the first routers along the packet’s path. This scheme may increase the computational overhead on the routers but provides a very simple traceback procedure to the victim because there is no need of path reconstruction from the victim. After a very small, compared to PPM, number of incoming packets from the same source, the victim is able to determine the approximate source of those packets with ease.

An enhancement of DPM proposed in [17] further reduces the number of packets needed in order to be able to identify the source to 1 packet. This enables the victim to perform per packet filtering in real time. The drawback of the last marking scheme is that it also requires a map of all the upstream routers and that there is a substantial large fault probability depending on the total number of edge routers.

One last marking scheme that is quite different than the above is based on geographic information rather than the IP address [2]. The scheme called directed geographical traceback (DGT) exploits the fact that the path from one node to the other in the Internet is highly correlated with their geographical locations. In this scheme the routers inject (mark) direction information into the packet that shows the relative geographical position of the next router. This scheme depends on the fact that all the routers will have knowledge of the relative geographical information of their neighbours.

Combining packet marking with agent design in [20], we find another approach that is able to identify the approximate source of the attack with a single packet. This approach involves the use of controller systems inside administrative domains that are involved with the management of the DDoS attack as well as agents that are deployed on all the edge routers of the administrative domains. The approach is similar to the Centertrack [18] approach.

Another combination of packet marking and existing technologies i.e. Pushback [9], can be found in [11]. Pushback is a mechanism that can be implemented in internet routers and uses congestion signatures in order to identify traffic that follows DDoS attack characteristics and proceed to filter the traffic. The proposed methodology does not mark incoming packets based on a fixed probability but start to mark the packets when the pushback mechanism identifies abnormal traffic. This has the advantage that the possible computational overhead on the routers is only employed during an active DDoS attack.

Turning our discussion from packet marking to logging, we can say that packet logging is the most straightforward method to use for traceback reasons. According to this method, the routers keep logs of preferably every packet that traverses through them and those logs can be used to trace a packet back to its source by continuously auditing the router logs. In its generic form, the aforementioned method is not practical because the amount of information needed for such detailed logging is prohibiting. Also, there are regulations that protect personal information of individuals, thus logging the content of the packets is in most countries prohibited. In [15] we find a new approach of IP traceback, named Source Path Isolation Engine (SPIE), based on packet logging which overcomes the aforementioned problems and achieves effective traceback of a single packet delivered by the network in the recent past. According to SPIE, the routers keep in their log files, digests of the packets headers instead of the whole packets. SPIE also uses Bloom filters to minimize the memory requirement up to 0.5% of link bandwidth. However, like most of traceback systems, it can produce false results if there are compromised routers along the attack path. There is also some false positive probability and this scheme cannot be effectively used for per packet filtering.

Lastly, one more effort to IP traceback is made by the definition of the ICMP Traceback message (ITrace) [4] by the IETF. This ITrace message is used to carry information on the routes that a packet has taken. This way it utilizes out of band messaging to achieve packet traceback. The generation of the ITrace message is based on a very low probability (1/20000). The generated message is send either to the destination or the origin of the packet. So in case of DDoS attack, the destination, the victim, system can use this information to traceback the attack path. However this out of band communication increases

network load by 1% approximately. It also cannot provide per packet filtering capabilities to the victim because of the low probability of the generated ITrace message.

One attempt to enhance the existing ITrace scheme is made in [12]. According to that, the ITrace messages can be modified to carry the whole attack path from the origin until the router that produces the message. This way the path reconstruction from the victim can be done very easily by only identifying the attack packets.

5.2.3 Evaluation

As we saw, there is a lot of interest in IP Traceback as a key to the solution of the DDoS attack problem. The main three approaches that have been used are packet marking, packet logging and out of band signalling. All these methods have their disadvantages but all can be used effectively in certain scenarios of DDoS attacks. We also saw that packet marking receives most of the attention by the researchers.

Packet marking is a very appealing solution due to the reason that it overloads the IP header of the packet; it has innate backward compatibility problems. Those compatibility problems can have more serious impact with the wider adoption of the IPv6 protocol. Most, if not all, packet marking schemes are incompatible with the IPv6 protocol and have to be significantly changed in order to be able to work under this protocol. Nevertheless, breaking an underutilized protocol such as IP packet fragmentation in order to provide traceback capabilities to an existing protocol such as the IP, is more like patchwork than a concrete solution. Packet marking can be combined with the IPSec protocol to provide those capabilities in a more elegant way.

On the other hand, the major obstacles in packet logging have been overridden by current research efforts, but still packet logging requires considerable amount of memory and processing power from the routers to be effective. A wide adoption of packet logging methodology for traceback reasons could result in a global system capable of not only tracing DDoS attacks but also protecting networked systems from most of hacking attempts. It could also give a solution to the internet worms, another major problem in network security nowadays.

One of the first coordinated attempts to provide a global and standardized solution to the IP traceback problem and the DDoS attack problem is the ITrace message proposed by IETF. This method combines low computational and network overhead with effective traceback of recent DDoS attacks.

Unfortunately none of the proposed methods give a concrete solution to the DDoS attack problem because none of them (efficiently) enables the victim to filter the incoming packets in real time so that it can protect itself from the impact of an on-going DDoS attack. Some of the proposed solutions promise single packet IP traceback but the real need is the ability of performing this traceback procedure in real time for each and every packet. For this reason, IP traceback methodologies as countermeasures against DDoS attacks have to be combined with existing traffic regulation methodologies in order to give better and faster results against DDoS attacks.

5.2.4 References

[1] M. Adler, “Trade-offs in probabilistic packet marking for IP traceback”, in the Journal of the ACM, Vol. 52, No. 2, pp. 217-244, March 2005

[2] N. Ansari, “Directed geographical traceback”, in Proceedings of the IEEE ITRE, 2005

[3] A. Belenky and N. Ansari, “IP Traceback with deterministic packet marking”, in IEEE Communications Letters, Vol. 7, No. 4, pp. 162-164, April 2003

[4] S. Bellovin et al, “ICMP Traceback messages”, IETF Internet Draft, 2003

[5] H. Burch and B. Cheswick, “Tracing anonymous packets to their approximate source”, in LISA XVI, December 2000

[6] Z. Chen and M. Lee, “A simplified algebraic marking scheme for IP traceback”, 2003

[7] D. Dean, M. Franklin and A. Stubblefield, “An algebraic approach to IP traceback”, in ACM Transactions on Information and System Security, Vol. 5, No. 2, May 2002

[8] P. Ferguson and D. Senie, “Network ingress filtering: Defeating denial-of-service attacks which employ IP source address spoofing”, RFC 2827, 2000

[9] S. Floyd, S. Bellovin, J. Ioannidis, K. Kompella, R. Mahajan, V. Paxson, “Pushback message for controlling aggregates in the network”, Internet Draft, 2001

[10] J. Gomes, F. Jin, H. Choi and H. Choi, “Enchanced probabilistic packet marking for IP traceback”, in Proceedings of the IEEE Workshop on Information Assurance and Security, pp. 30-37, June 2002

[11] H. Lee, “Advanced packet marking mechanism with pushback for IP traceback”, in ACNS ’04, LNCS 3089, pp. 426-438, 2004

[12] H. C. J. Lee, V. L. L. Thing, Y. Xu and M. Ma, “ICMP Traceback with cumulative path, an efficient solution for IP traceback”, in ICICS 2003, LNCS 2836, pp. 124-135, 2003

[13] S. Savage, D. Wetherall, A. Karlin and T. Anderson, “Network support for IP traceback” in IEEE Transactions on Networking, Vol. 9, No. 3, pp. 226-237, June 2001

[14] M. Shung and J. Xu, “IP traceback-based intelligent packet filtering a novel technique for defending against Internet DDoS attacks”, in IEEE Transactions on Parallel and Distributed Systems, Vol. 14, No. 9, pp. 861-872, September 2003

[15] A. C. Snoeren, C. Partridge, . A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent and W. T. Strayer, “Single-packet IP traceback”, in IEEE/ACM Transactions on Networking, Vol. 10, No. 6, pp. 721-734, December 2002

[16] D. X. Song and A. Perrig, “Advanced and authenticated marking schemes for IP traceback”, in Proceedings of the IEEE INFOCOM, 2001

[17] K. Stefanidis and D. N. Serpanos, “Packet-marking scheme for DDoS attack prevention”, in Proceedings of Security and Protection of Information, 2005

[18] R. Stone, “CenterTrack: An IP overlay network for tracking DoS floods”, in proceedings of 9th Usenix Security Symposium, August 2000

[19] Y. K. Tseng and W. S. Hsieh, “CPPM – Compensated probabilistic packet marking for IP trace backing”, IEICE Transactions on Communications, Vol. E87-B, No. 10, pp. 3096-3098, October 2004

[20] U. K. Tupakula and V. Varadharajan, “A practical method to counteract denial of service attacks”, in Proceedings of the ACSC2003, Vol. 16, 2003

5.3 Automatic Access Control

Access controls are mechanisms that are made to protect networks and services from unauthorized access to the target’s resources. Many methods have been proposed to implement access control in a network, depending on the intelligence of the nodes, the memory capabilities and the predefined profiles, and are based on:

- ***Profile authentication:*** If the node has some characteristics, it can join to the network.
- ***Access Code:*** Typical password access, based on memory data, switch configuration, or any other procedure
- ***Predefined topology:*** Only pre-established nodes can join to the network, like MAC filtering in a

WiFi.

The denial of service (DoS) attack is an attempt of a malicious entity to obstruct the availability of a network or a service. It is extremely difficult to counter against such attacks, especially in wireless networks. There are many methods to carry out a DoS attack. In the general case, an attacker sends a large amount of requests to a server. The server is kept occupied as it tries to serves those request, exhausts its resources, like power, processing power, network bandwidth and memory, and becomes unavailable to serve requests derived from legitimate users. The past years revealed the fact that poor design decisions in network protocols and operating systems can become a serious obstacle in DoS and Distributed DoS resilient systems and services. The IP protocol is vulnerable to such attacks and basic software design methodologies don't take into account security requirements that would enable DDoS resilient services.

In wireless networks, DoS attacks that can be performed in the communication link between legitimate tags and readers. These DoS attacks can be classified in three main categories based on the factors that cause them:

- **Kill command:** when a tag is manufactured, it is given a secret password. The password can be easily cracked by an attacker, due to the limited memory of the tag. Then he executes a kill command along with the password to disable the tag permanently.
- **Jamming:** the attacker performs electromagnetic jamming in order to prevent tags from communicating with the readers.
- **De-synchronization:** the attacker uses jamming techniques to de-synchronize the tag and the reader in order to permanently disable the authentication capability of the tag.
- **Tag Data Modification:** the attacker modifies the EPC data of the tag, so it is not recognized anymore by the reader.

5.3.1 Proposed approaches

There is a special type of DoS attack that can exploit vulnerable access control mechanisms. In the basic scenario of an access protocol, clients exchange encrypted messages with an authentication server in order to gain access to the service. The server has to perform an expensive public key process, like a decryption operation, in order to be able to elaborate the message's content. Then an attacker can perform a DoS attack by sending a large number of access request messages to the server.

In order to protect networks from this kind of DoS attack lightweight access control mechanisms have been implemented. The automatic access control stands for a technique where the entities that participate in a network can use a lightweight feature to authenticate each other. Thus, the server won't use an expensive encryption operation that can lead to the DoS attack. The most common technique is to use hash functions. The client sends a hashed secret that is known to the server. Ordinary, the server keeps a map of all these hashed secrets for all of its clients. Then, server only needs to perform a cheap look up function to discover if the user is legitimate. Thereby, the attack can be efficiently encountered. However, several issues can arise from the adaptation of an automatic control mechanism that include mutual authentication, de-synchronization of client and server, DoS attacks in other steps of the propose protocol replay attacks and link-ability of different communications of the same user.

Except from hash functions, other methods that encounter this problem include: authentication using matrix multiplication, pseudorandom number generators and cyclic redundancy check.

Several protocols have been proposed in the literature and associated attacks are considered. Such attacks can involve relevant issues of mutual authentication, de-synchronization of client-server, DoS attacks in other steps of the proposed protocols, replay attacks and link-ability of different communications of the same user. O-FRAP$^+$ [1] and Gossamer [2] are two indicative ultra-lightweight protocols that address this issue. O-FRAP$^+$ uses two keys for every tag, the one from the previous session and the new one for the current session, and the secret key updating procedure is performed in a chain fashion. It provides mutual authentication, privacy-preserving authentication, forward security and resistant against DoS attack. Gossamer makes use of pseudorandom numbers and can effective to provide data confidentiality, tag anonymity, mutual authentication, data integrity, forward security, robustness against replay attacks and DoS attack prevention.

Moreover, even when we use automatic or simple access control the failure of recognizing a legitimate user, leads as to the conclusion that the system can be under attack. Thereby, other mechanisms [3] like anomaly detection, intrusion detection (IDS), intrusion prevention (IPS), intrusion tolerance and mitigation, intrusion response mechanisms and firewalls can take on.

5.3.2 Important Attributes

Six important security aspects should be taken into account when we deal with a proposed automatic access control mechanism:

- ***ID Anonymity:*** The ID of user sent on the communication channel can be used by an attacker to impersonate as the legitimate user.
- ***Forward Secrecy:*** An attacker could use the previously captured messages to find a pattern or information about the next transaction.
- ***Replay Attack:*** An attacker can eavesdrop the messages that are transmitted between the client and the server. Then, he retransmits the same message later to obtain a response.
- ***DoS attack resistance:*** Even in presence of a DoS attack, the proposed protocol should overcome the attack and return to the normal state.
- ***Mutual Authentication:*** Both participants should be able to authenticate each other to avoid an attacker from acting as rogue server or rogue client.
- ***Non-linkability:*** Both insiders and outsiders couldn't be able to ascribe any session to a particular user.

5.3.3 References

[1] Dang Nguyen Duc, Kwangjo Kim: Defending RFID authentication protocols against DoS attacks. In Computer Communications, volume 34, pages 384-390. (2011)
[2] Deepak Tagra, Musfiq Rahman, Srinivas Sampalli: Technique for Preventing DoS Attacks on RFID Systems. In 2010 International Conference on Software, Telecommunitaions and Computer Networks (SoftCOM), pages 6-10. (2010)
[3] Christos Douligeris, Aikaterini Mitrokotsa: DDos attacks and defense mechanisms: classification and state-of-the-art. In Computer Networks, volume 44, pages 643-666, 2004. (2004)

5.4 Quality of service in Embedded Systems

Much work has been done and has been subject of extensive research on the area of QoS[6]. This research deals more the network level of nSHIELD in order to increase reconfiguration and function density of a processing mode, to make a node more secure against side-channel attack and to implement self-healings properties. In next sections we will discuss about QoS, main frameworks and solutions developed in this area and the related work done in pSHIELD regarding reconfiguration.

There are two main efforts (IETF standards) already implemented in the area of QoS for the extension of Internet resources management protocols and models: DiffServ[7] (Differentiated Services) and intServ[8] (Integrated Services). Both of them support real time processing and integrate RVSP[9] (Resource ReSerVation Protocol). These paradigms are network oriented and define different parameters (i.e IntServ

[6] The DIN group "Software and Systems Engineering" for QoS standards.

[7] http://datatracker.ietf.org/wg/diffserv/charter/

[8] http://datatracker.ietf.org/wg/intserv/charter/

[9] http://tools.ietf.org/html/rfc2205

is envisioned to manage real time elements such as remote video, multimedia and virtual reality and DiffServ deals more with the creation of different SLAs to reduce the complexity of IntServ). There are other industrial approaches such as Weapons System Open Architecture (WSOA) developed by Boeing that is based in CORBA middleware.

5.4.1 QoS Adaptation: first approach for Self-X technologies

Embedded systems have the following functionalities that have to be taken into account in order to reach their objective:

- Adaptive real time
- Varying resource needs
- Availability
- Security and safety

It has no sense to analyse self-reconfiguration techniques at node level (as this level is the simplest one.) However, it is very important to analyse the interfaces and connectivity among all nodes in order to increase the composition of trust and define critical areas that might incur into vulnerabilities. QoS adaptation is usually performed by systems software. So that, we have to define where it should be stored (physically and logically) this module (often called resource kernel). For self-adaptation, control systems need to express the value of different modes of operation and configuration to facilitate automatic selection. This information could be made available in a self-descriptive way to analyse it at runtime in an embedded system. The adaptation specification includes the application logic for switching between modes and it must express the requirements on the mode switching process at runtime (e.g. maximum switching delays, consistency checks, fallback into old configuration, etc.).

5.4.2 Research projects for reconfiguration and self x technologies

The on-going ITEA2 project SYLEX is currently developing a framework that will enable optimization of the design and execution of self-x and adaptive real-time embedded systems. But the focus of this project is mainly the integration of techniques that cross multiple-levels of abstraction and addresses multiple non-functional system constraints.

The FP7 project ACTORS (Adaptivity and Control of Resources in Embedded Systems) addresses also the design of complex embedded systems and aims to provide adaptive resource management during runtime, based on feedback control. The focus of this work is to distribute the available system resources to the running tasks dynamically, but also to guarantee resource requirements. The self-x and adaptivity of the entire system is considered in nSHIELD project.

The FP7 project CHAT (Control of Heterogeneous Automation Systems) aims at developing the next-generation of distributed control systems, able to effectively tackle the supervision and control of larger and more complex automated industrial plants, while drastically reducing their infrastructure, maintenance and reconfiguration costs. One of the objectives is to develop middleware architecture for automation (control) components, providing composability, seamless connectivity and dynamical reconfigurability which ensure safety and security requirements.

Indeed, pSHIELD project is a reference of this adaptation and reconfiguration techniques. Adaptation was realised based on the compositions of metrics on top of the medieval castle algorithm approach. The value of each of the nodes was reached by applying Common Criteria vulnerability analysis and this value was launched to control system in order to start with the composition. All interactions of the systems were modelled according to the medieval castle approach, and as a result a final composed value was achieved representing the SPD functionality value of one subsystem.

5.4.3 Self-x technologies analysis nSHIELD layers

The challenge of complexity management of networked embedded systems might require performing an efficient management of different elements that coexist in a particular subsystem which might be

governed by unpredictable behaviours. Self-technologies are based in autonomic computing and self-adapting systems. New systems designs comprise heterogeneous, tightly, and loosely coupled components. Managing this new paradigm where multiple devices take part is a hard task and requires a great effort (cost). Salehie and Tahvildari[10] blame it on the heterogeneity dynamism, and interconnectivity in software applications, services and networks.

The autonomous behaviour is a concept that deals with reconfiguration and adaptation and is goal dependent. Knowledge is the basis of the paradigm and the more the knowledge is structured and managed the more possibilities will be performed for adaptation and self x technologies. Hence, autonomous behaviour is about acquiring knowledge of internal state and changing it depends on the external context.

Self x technology concept is a derivation of autonomous behaviour. It encompasses concepts such as resiliency and fault-tolerance. Self x technologies are built for trying to reduce cost and losses in network management (this later is increasing exponentially due to the complexity and heterogeneity of systems of systems.) Self-x technologies and properties can be enumerated in the following way:

- self-configuring: The ability to readjust itself “on-the fly”
- self-healing: Discover, diagnose, and react to disruptions
- self-optimization: Maximize resource utilization to meet end-user needs
- self-protection: Anticipate, detect, identify, and protect itself from attacks.

Self-* technologies, in summary, comprises several concepts such as, stabilisation, resiliency, tolerance to faults and survivable systems and should be *human supported*. The reason for increasing a system with self-x technologies is the continuous availability and thus the business continuity: main driving force for developing self x based reconfiguration and adaptation mechanisms. Enduring continuity includes resilience against intended, necessary adaptations and unintentional, arbitrary behaviours. nSHIELD objective is to preserve SPD functionalities in different layers, so that self-x technologies are of paramount in order to make systems robust. However it is important to understand different risks that are associated for each layer and to structure these risks and linked safeguards. The ontology for structuring the internal and external knowledge for self-healing nSHIELD environment will be based on metrics which are used to express and limit the actuation for different devices within a subsystem.

Many different processes have been described in the literature in order to perform self x technologies: MAPE-K[11] loop contributes to it by defining the following steps:

- Monitor: The monitor gathers status information from the system through sensors and pre-processes it for the analyse task.
- Analyse: This entity determines whether the received monitored information must follow a designated action. This is generally done by comparing status information to system specific thresholds. These thresholds in nSHIELD are brought to the system by the implementation and deployment of the metrics.
- Plan: A running system often is full of situation specific dynamics. Therefore, an accurate, sound, and planed deployment of the actions demanded by analyse is required. This plan is should be structured in nSHIELD taking into account the policy plan for metrics gathering and deployment.
- Execute: Presents the entity that executes the parts of previously conceived plans on the managed element. The execution should be performed under the metrics components criteria.

[10] Salehie M, Tahvildari L (2005) Autonomic computing: emerging trends and open problems. SIGSOFT Softw Eng Notes 30(4):1–7

[11] http://www.ibm.com/developerworks/library/ac-itito/index.html

- Knowledge: This represents the knowledge base consumed and produced by all four previously mentioned tasks. This knowledge is represented formally by ontology in nSHIELD. This module is fundamental for maintaining SPD functionalities in nSHIELD: utilizing the ontology and metric values, nSHIELD will be able to reconfigure, adapt and self-reconfigure in multiple layers and compose these metrics according to the policy (in this case converging with the nSHIELD QoS or SLA)

These five steps are usually reduced to three main activities: detection, diagnosis and repair:

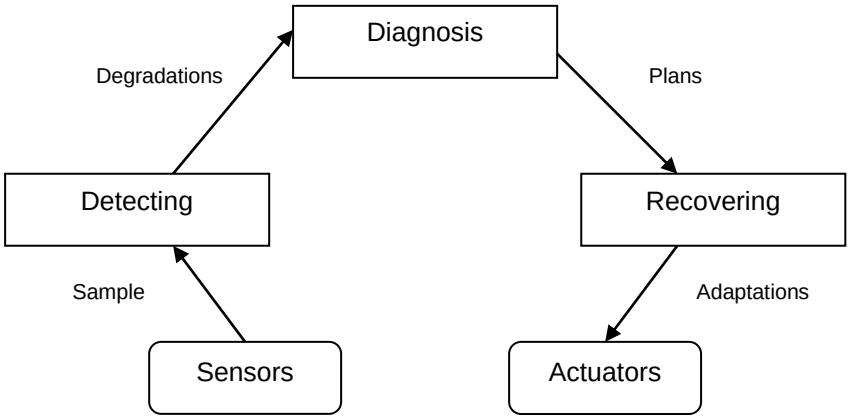

Figure 27 - Steps for self-technology: Healing

In nSHIELD **embedded systems** operate in special constrained environments (node level). Real time environments and critical systems are often being implemented by embedded systems. Embedded systems guarantee somehow the needed reliability by systems of systems. But to provide an appropriate amount of reliability, a certain overhead of extra components is required: Self x technologies can be applied by individual embedded systems (i.e. FPGA that has an important reconfiguration and adaptation component) however, Self x technologies poses the most impact when there are many elements within a subsystem; furthermore, when there is an increasing knowledge management about the complementarity of these elements within a network or moreover, when these elements have different levels of abstraction and are located in different layer definitions.

ES have their own force drivers in order to be governed, **Operating Systems**, however, consume more computational operations and govern more elements. Not all elements within a networked environment are controlled by an operating system. Self-x technologies in OS are easier to develop and deploy (comparing to ES): the main objective of self-healing in operating systems is, aside from failure resilience, to avoid faults requiring a system restart. These include mainly two possible methods: (i) release the supervising kernel from dependencies and (ii) free allocated resources and rerun the failing applications.

The **architecture based** view depicts the real environment in a model of resources and dependencies. This is a difficult task due to the complexity of the real environment; however the abstractions techniques make this model be more understandable for systems developers. For nSHIELD, this view should be an easy activity because is an architectural model based platform. So that, using ADL (Architectural Description Language) and implementing the correct metrics it should be easy to establish a self-reconfiguration built in architecture. A particular instance of this view is the cross/multi-layer based approach. This approach enables reconfiguration and adaptation within resources that are situated in different dimensions or layers. It happens that in one subsystem, the management work should be taken into account for most of the 7 layers of ISO/OSI-7 where all layers have their own type of resources. This **layer approach** could be the solution that nSHIELD might require for its objective; however the best solution should be focus on a hybrid one.

Embedded systems are usually performing their task within a close environment and do not react very well to unexpected external happenings. One solution to face this problem is the **multi agent-based**

paradigm. Agents, by design, can handle unexpected situations in environments with unpredictable behaviour. There is a huge literature behind multi agents-based model. Its capacity for redundancy and hence avoiding DoS is the most powerful advantage. Agents enable flexibility, interoperability and scalability: therefore, nSHIELD will have into account this paradigm in order to include it as a communication framework of one part of the whole nSHIELD platform.

Self x technology is the main factor of **reflective middleware** (It enables analysis by queries on structure and on system states, and adaptation by reconfiguration actions.) For example, OpenORB provides structural reflection, including interface meta-models for external representation and architectural models for internal representation of a component. Furthermore, there are other types of middleware that define their own self-* techniques: such middleware types are Sensorpedia (Web 2.0 based), TinyDB (Database oriented), Mate (Virtual Machine based), Agilla (Mobile Agent), TinyLime (tuple space) and TinyCubus (cross-layered).

Finally, at message and XML level, **web services** can act as a new paradigm for establishing reconfiguration and adaptation for one particular distributed system. Web services enables attestation and thus can operate with some functionalities at node level (see ws-attestation for TPMs) and moreover can orchestrate or even more establish choreography (through BPEL) techniques for self-x technologies. This is an important fact for nSHIELD. However, this has one main disadvantage: it is not for real time and critical systems. A combination of reactive and proactive self-healing is described in the work of Halima et all.[12] The work presents a self-healing middleware called QoS-Oriented Self-Healing (QOSH) that enhances SOAP messages with QoS metadata to monitor QoS degradations.

5.4.4 SLAs contributing to Self-technologies

Although current SLA languages allow expressing Quality of Service (QoS) constraints with different success, the absence of security and dependability aspects in SLAs makes it difficult to create agile QoS for Embedded systems.

A Service Level Agreement (SLA) is a common way to specify the conditions under which a service is to be delivered, but is usually limited to availability guarantees.

The most well-known machine-readable SLA models are the Open Grid Forum's Web Services Agreement (WS-Agreement)[13] and IBM's Web Service Level Agreement (WSLA)[14]. The WS-Agreement specification proposes a domain-independent and standard way to create SLAs and has been up taken in several projects, while its predecessor WSLA seems to be deprecated.

It is difficult to express that a SLA can be derived to establish a kind of level agreement among different embedded systems (at node layer). There is always has to be a control systems responsible for managing the interactions between nodes and this means software processing. However, nSHIELD could analyse the process of including agile Level Agreements in networks of embedded systems. This is one main issue that will be analysed within the project jointly with the analysis of metrics in WP2 as parameters for measuring and composing techniques for increasing QoS.

[12] Halima RB, Drira K, JmaielM (2008) A QoS-oriented reconfigurable middleware for self-healing web services. In: ICWS '08: Proceedings of the 2008 IEEE international conference on web services. IEEE Computer Society, Washington, pp 104–111

[13] A. Andrieux, K. Czajkowski, A. Dan, et al, Web Services Agreement Specification (WS-Agreement), March 14 2007, available at: http://www.ogf.org/documents/GFD.107.pdf

[14] H. Ludwig, A. Keller, A. Dan, et al, Web Service Level Agreement (WSLA) Language Specification, January 28 2003, available at: http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf

The evolution of the system is somehow one important issue to take into account. A threat today might not be a threat in the future. Therefore we need to have at least a classification of different faults and threats that might occur in our system. The main goal is to make the system be resilient, trustworthy and reliable. Self x technologies will enable this at local and global focus.

6 Cryptographic technologies

6.1 Cryptographic Functionalities for SPD Node

The modern day embedded systems (ES) employ increasingly sophisticated communication technologies: low-end systems, such as wireless head-sets use standardised communication protocols to transmit data, remotely-controlled thermostats adjust room temperatures on user request sent from a mobile phone or from the Internet, while smart energy meters automatically communicate with utility providers. Furthermore, wireless sensor networks (WSN), or the recently emerging cyber-physical systems (CPS) are proposed to autonomously monitor and control safety critical infrastructure such as, for example, a nation-wide power grid. The increased complexity of these systems and their exposure to a wide range of potential attacks involving their communication interfaces makes security an extremely important and, at the same time, challenging problem. nSHIELD project recognizes the fact that security, privacy and dependability (SPD) are core characteristics of any modern ES and it proposes to address them as a "built-in" technology rather than as "add-ons". In fact, due to the complexity of networked embedded systems, as well as because of the potentially high cost of failures, SPD must become an integral part of ES design and development.

Hardware (HW) and Software (SW) crypto technologies are fundamental for achieving security of the SPD networks composed of SPD nodes. One of targeted research topics is a study and design of embedded operating systems and firmware for energy-constrained SPD nodes. Choosing the right cryptographic technology for different ES Nodes is one of the most important research efforts dedicated in the design of SPD nodes and SPD network architecture.

6.1.1 Symmetric and asymmetric cryptography

Cryptography is seen as the basis for the provision of different systems security, fundamentally by seeking to achieve a number of goals, that are; confidentiality, authenticity, data integrity and non-repudiation. Typically, security provided through cryptographic means comprises mathematical cyphering algorithms and key management techniques. Common cyphering algorithms are divided into two types; asymmetric and symmetric. Key management is influenced by different factors such as system's architecture and class of devices.

A large number of symmetric ciphers have been designed to date and they vary in their security and performance characteristics. The security of a symmetric cipher cannot be easily established at design time and usually many years of exposure to public scrutiny are required in order to consider a cipher secure. On the other hand, performance characteristics can be measured and the best performing cipher can be objectively selected.

Symmetric key based authentication (i.e. the claimer and the verifier share a key) is vulnerable to the compromise of either party in the authentication. In contrast, there is no secret key shared between the claimer and the verifier when using digital signatures. In public key cryptography (also called asymmetric key cryptography), a pair of keys including public key which is publicly available and private key which is kept as secret, are assigned to each entity. To authenticate to the verifier, the claimer signs a challenge message from verifier using its private key, and appends a digital certificate that confirms the link between the claimer and its public key. The verifier uses the certificate to verify the validity of the signer's public key and validates the integrity and authenticity of the message using the signer's public key. If an entity is no longer trustworthy, its certificate is revoked and the revocation is announced publicly by the certificate authority (CA).

Many implementations use symmetric cryptography, for example keyed hash functions or AES implementations, to meet the constraints of low-power consumption, limited chip area, and restricted computation time in order to produce low-cost devices. But in many application scenarios it is indispensable to obtain the high security level provided by an asymmetric approach. The use of asymmetric instead of symmetric solutions for different devices can radically reduce costs. Public key

approaches are more reasonable in open-loop applications, since no secret keys must be handled by the device. But the integration of public-key cryptography into low-cost devices is technological challenge. Public key cryptography systems are usually based on the assumption that a particular mathematical operation is easy to do, but difficult to undo unless you know some particular secret. This particular secret serves as the secret key. A recent development in this field is the elliptic curve cryptography (ECC).

Protocol level is application specific and includes the design of protocols to be performed on EDs. The PKC (public key cryptosystems) are based on RSA or DSA. ECC (Elliptic Curve Cryptography) and Hyper-ECC (HECC) are based on different algebraic structure and offer equivalent security as RSA, but for much smaller key size. This result in smaller HW and lower power consumption that is extremely important for CMPNs. Modular multiplication forms the basis of modular exponentiation which is the core operation for RSA cryptosystems. Similarly, it is also important for ECCs especially if one use projective coordinates. Montgomery's methods is the most popular for modular multiplication since it avoid time consuming trial division that is common bottleneck of other algorithms. However, it is not enough to have strong cryptographic algorithms. It is also important that their implementation that must be secured. The attack techniques are related to the PHY implementation. For example, the attack can be active or passive. Active attack is performed in such way to alter HW or SW by changing the operating conditions (power supply, temperature, etc.) Passive attack is based on monitoring side-channel information (power supply, EM radiation).

6.1.2 Elliptic Curve Cryptography for CMPNs

Elliptic curve cryptography (ECC) is becoming a powerful cryptographic scheme. Because of its efficiency and security is a good alternative to cryptosystems, like RSA and DSA, not just in constrained devices, but also on powerful computers. ECC is very important in the field of low-resource devices such as smart cards and Radio Frequency Identification (RFID) devices because of the significant improvements in terms of speed and memory compared to traditional cryptographic primitives (e.g. RSA). Memory is one of the most expensive resources in the design of embedded systems which encourages the use of ECC on such platforms. Security, implementation and performance of ECC applications on various mobile devices have been examined and it can be concluded that ECC is the most suitable PKC (Public Key cryptography) scheme for use in a constrained environment.

More and more electronic transactions for mobile devices are implemented on Internet or wireless networks. In electronic transactions, remote client authentication in insecure channel is an important issue. For example, when one client wants to login a remote server and access its services, such as online shopping and pay-TV, both the client and the server must authenticate the identity with each other for the fair transaction.

The remote client authentication can be implemented by the traditional public-key cryptography. The computation ability and battery capacity of mobile devices are limited, so traditional PKC, in which the computation of modular exponentiation is needed, cannot be used in mobile devices. Elliptic curve cryptosystem (ECC), compared with other public-key cryptography, has significant advantages like smaller key sizes, faster computations. Thus, ECC-based authentication protocols are more suitable for mobile devices than other cryptosystem. However, like other public-key cryptography, ECC also needs a public key infrastructure (PKI) to maintain the certificates for users' public keys. When the number of users is increased, PKI needs a large storage space to store users' public keys and certificates. In addition, users need additional computations to verify the other's certificate in these protocols.

6.1.3 Cryptographic Technologies

At node level low-energy low-processing devices are expected to perform cryptographic operations. A TPM is an example of a component providing HW/SW cryptographic technologies. The SW embedded on such a cryptographic component has a direct impact on its:

- size (through its code size and memory footprint: memory elements are taking an important part of the component surface),
- costs (directly linked to the surface of the component),
- speed (optimized code provide its computation results more quickly), and

- power consumption (the quicker you can execute a set of instruction, the quicker you can put the component back in sleep mode where power consumption is reduced).

Algorithmic designs and implementations best suited to constrained devices (e.g., RFIDs, contactless smart cards, sensor nodes, mobile devices) are part of lightweight crypto. Here we are interested in symmetric ciphers, stream ciphers and hash functions. These primitives could be used in a standalone fashion or as building blocks of lightweight crypto/security protocols (e.g., for authentication).

For minimal hardware requirements were developed two symmetric ciphers, i.e. DESL and Present. A lot of effort was put into porting more established algorithms, like AES, IDEA, TEA and the older DES, into low cost implementations. Several mature block ciphers are available and their security (strength against a number of attacks) is well understood. On the other hand, stream cipher designs are still at the edge. A number of efficient hardware designs were and the security they provide on a constrained device is still quite risky. Hash functions designs too, are not lightweight so far. The SHA-3 competition has improved our understanding substantially but still hashes based on block ciphers may have an advantage.

The code optimization (time and memory) and fine-tuning will be done in nSHIELD to improve the characteristics of the component while maintaining the high level of security of the component and reducing the requests of the SW on the HW resources.

Relating to lightweight crypto target platform's special properties will be taken into account when choosing cryptographic algorithms and also a number of tradeoffs, for constrained devices. Protocols for constraint systems will be revisit, which are used in theory but they cannot be used in practice because the primitives they are based on (e.g. hashes) cannot yet be efficiently implemented.

An embedded cryptographic library has to be implemented, which will provide a set of optimized cryptographic algorithms for embedded devices and a standardized approach in SPD node software cryptographic operations. Incorporating Side-Channel Attack (SCA) countermeasures within the optimized implementations should be made carefully such that the optimized implementations do not introduce new leakage channels.

A possible approach is the design and implementation of an embedded operating system with lower resources requirements (e.g. by using a memory management better adapted to the security/integrity requirements that could be put on certain memory location without generating a too important overhead).

6.1.3.1 Asymmetric cryptography for low cost nodes

Asymmetric cryptography algorithms and protocols used with powerful hardware must be adapted to limited devices, both in terms of computing capability and energy constraints. Symmetric ciphers serve mainly for message integrity checks, entity authentication, and encryption, whereas asymmetric ciphers additionally provide key management facilities and non-repudiation. Asymmetric ciphers are computationally far more demanding, in both hardware and software.

All implementations relying on symmetric crypto primitives operate as master-key systems, sharing a master secret over all nodes enabled for verification of the authenticity of other nodes in the system. If one component (e.g. a stolen reader) gets compromised and the master key revealed, the whole system is broken. In asymmetric cryptography, the background system or the reader device may verify the authenticity of the node without knowledge of the node's secret. Compromising such a reader does not do any harm to the overall embedded system and revealing one key does not immediately compromise the whole system, but only the very one entity, since every low cost node would have its own secret.

There are three established families among public-key algorithms of practical relevance: ECC, RSA, and discrete logarithms. ECC and recently Hyper Elliptic Curve Cryptography (HECC) are considered the most attractive for embedded environments because of its smaller operand lengths and relatively lower computational requirements. TinyECC, a software package providing ECC-based operations is intended for sensor platforms running TinyOS. ECC and HECC offer equivalent security as RSA for much smaller

parameter sizes, which is the main benefit. The advantages result in smaller data-paths, less memory and lower power consumption.

The nSHIELD project shall provide an optimized hardware implementation for an ECC or HECC public-key algorithm. The key size will affect the cost as well since it maps the need for short, medium or long term security. Although using a hardware-software code sign can substantially increase public-key performance with minimal area, in some situations public-key cryptography must be implemented purely in software because changes to the hardware aren't possible. For many pervasive computing applications hardware-software code sign produce the best trade-off between size and speed.

Strong asymmetric cryptography shall find its way to low cost nodes in embedded systems, which has for a long time been doubted to be feasible at all. The nSHIELD project should implement a secure authentication protocol based on ECC in a low cost hardware-node as well as in an ES software solution and integrate to prototypes in various scenarios. The implementation has to be parameterized against side-channel attacks (SCA) – which might have a major impact on implementation cost - such as simple power analysis (SPA), differential power analysis (DPA), as well as their electro-magnetic counterparts SEMA and DEMA and fault attacks (DFA).

6.1.4 Main Topics to be covered by Task 3.5

Task 3.5 covers cryptographic technologies providing horizontal SPD technologies that will be adopted at different level depending on the complexity of the node and considering its HW/SW capabilities, its requirements and its use. The research will rely mainly on the hardware and software crypto technologies.

At node level low-energy low-processing devices will perform cryptographic technologies. A node must be trusted through the secure generation of cryptographic keys and limitation of their use, in addition to a hardware pseudo-random number generator and capabilities such as remote attestation and sealed storage. A Trusted Platform Module (TPM) may be used to authenticate hardware devices: each TPM chip is capable of performing platform authentication, since has a unique and secret RSA key burned in as it is produced. Future evolutions of cryptographic/hash functionalities, alternative communication interfaces better adapted to ES and additional cryptographic protocols (e.g. elliptic curves) will be supported. The adoption of the TPM will require the design and implementation of an embedded operating system with lower resources requirements, in order to be suited to the HW features of ES. Algorithms and protocols for asymmetric cryptography, usually used with powerful hardware, could be considered for low cost nodes, from cost point of view. They must be adapted to limited resources devices, both in terms of computing capability and energy constraints. The solution could be an optimized hardware implementation for an elliptic curve cryptography based public-key authentication algorithm. With the use of asymmetric cryptography, the background system or the reader device may verify the authenticity of the node without knowledge of the node's secret – thus compromising such a reader does not do any harm to the overall embedded system. Revealing one key does not immediately compromise the whole system, but only the very one entity, since every low cost node has its own secret. In addition to asymmetric technology, a secure authentication protocol based on ECC shall be implemented in a low cost hardware node as well as in an ES software solution and integrated to prototypes in various scenarios. Thus strong asymmetric cryptography shall find its way to low cost nodes in embedded systems. The implementation shall also be secured against side-channel attacks, such as simple power analysis and differential power analysis, as well as their electro-magnetic counterparts SEMA, DEMA and fault attacks.

In order to face the large amount of data generated by nodes (sensors) data compression techniques are required. These data have either to be processed locally and/or sent to other nodes for further processing. As these nodes often do not have the resources (computational and power) or complete enough information about the extended environment to make proper processing, the latter case (involving data transmission) is very common. An approach utilizing reconfigurable hardware, which accelerates compression algorithms while consuming less power, will be researched, aiming at improving the SPD features of the system and at enabling more reliable and secure transmissions and communications at network level. This approach enables also combining compression with self-re-configurability and self-recovery properties, as this type of hardware can be partially reconfigured, while less energy can be consumed in situations where compression is not needed or can be degraded without altering the node.

6.2 Hardware and Software Crypto Technologies in Relevant EU Projects

The security and constraints stemming from the limited resources of sensor nodes have been investigated in EU projects extensively. [1] constitutes one such attempt at trying to tackle these issues with [2] providing a more detailed look into the smart-home applications.

In terms of network technologies, the utilization of Trusted Platform Modules and Virtualization techniques is an emerging pattern in relevant EU projects. [3] examines the aforementioned topics in combination, aiming to provide a reference design of a Trusted Computing, light-weight virtualization framework specifically aimed at cloud applications.

Anonymous Authentication and Anonymity schemes in general are another key area of current research, since privacy is essential in many applications (e.g. social, medical etc.). An analysis of how Trusted Computing technologies can be used for anonymous authentication and how they can be integrated into common security frameworks (e.g. Java Crypto Architecture) can be found in [4]**.** A Direct Anonymous Attestation protocol utilizing NFC-equipped mobile devices and RFIDs is presented in [5] while [6] proposes an anonymization scheme for trusted third parties which overcome the need for a trusted third party while relying on the TPM's DAA functionality.

Secure routing protocols is a critical research area of networking technologies. [7] provides an overview of security issues and current trends in trusted routing for ad-hoc networks, judging the applicability in WSNs. A secure routing protocol is proposed in [8] where the geographical location of nodes along with other parameters (e.g. their remaining energy for better load balancing and lifetime extension) are taken into account. The interactions between secure routing protocols and the Service Discovery functionality are investigated in [9], which concludes that in some situations there is an efficiency gain if routing protocols allow the higher layers to override the routing decisions.

Intrusion Detections Systems (IDS) are a key tool in safeguarding distributed ES networks. In [10] and [11] dynamic and distributed IDS schemes are proposed, which utilizes agents as local monitors for their neighbours. Defensive techniques for sensor networks based on the nodes' locations are presented in [12]**,** analysing concepts of robust statistics to localize a node in the presence of malicious beacons.

The aspect of reconfigurability and its repercussions on security are considered in [13] and a security architecture is proposed which, based on a middleware layer, offers secure reconfiguration and communication, authenticated downloading from a remote source as well as a rekeying service for key distribution and revocation.

Trusted Software is another important area of middleware layer research and [14] proposes a Trusted Software Stack (TSS) to be integrated into existing security framework (facilitating the adaptation to Trusted Computing technology), including a prototype developed in the .NET programming environment.

A software architecture featuring enforceable security policies along with virtualization provisions is presented in [15]**.**

In [16] a middleware called MWSAN is proposed, that provides high-level services for sensor and actor networks. It follows the component-oriented paradigm and it leaves it up to the developers to configure it according to the actor and sensor resources, by taking into consideration issues such as the network configuration, the quality of service and coordination among actors.

The main features of a secure middleware for embedded peer-to-peer systems, in order to face the various security challenges of the Internet of Things (IoT) are presented in [17]. The presented service model and component-based middleware satisfies necessary principles such as security, heterogeneity, interoperability, scalability and so on.

An extensive overview of a particular category of middleware, the context-aware middleware, is presented in [18], whereas [19] covers service composition mechanisms in ubiquitous computing.

An ontology-based approach has been followed using the Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL) in order to develop monitoring and diagnosis rules [20]. In this way, any malfunctions can be detected and self-healing procedures can be invoked, in an effective, extensible and scalable way. A similar ontology-based approach was also presented in [21].

Finally, middleware can also be used in Kahn Process Networks (KPN) implemented over a Network on Chip (NoC). In [1], a methodology for identifying requirements and implementing fault tolerance and adaptivity is presented.

6.2.1 References

[1] M. Adler, “Trade-offs in probabilistic packet marking for IP traceback”, in the Journal of the ACM, Vol. 52, No. 2, pp. 217-244, March 2005
[2] N. Ansari, “Directed geographical traceback”, in Proceedings of the IEEE ITRE, 2005
[3] A. Belenky and N. Ansari, “IP Traceback with deterministic packet marking”, in IEEE Communications Letters, Vol. 7, No. 4, pp. 162-164, April 2003
[4] S. Bellovin et al, “ICMP Traceback messages”, IETF Internet Draft, 2003
[5] H. Burch and B. Cheswick, “Tracing anonymous packets to their approximate source”, in LISA XVI, December 2000
[6] Z. Chen and M. Lee, “A simplified algebraic marking scheme for IP traceback”, 2003
[7] D. Dean, M. Franklin and A. Stubblefield, “An algebraic approach to IP traceback”, in ACM Transactions on Information and System Security, Vol. 5, No. 2, May 2002
[8] P. Ferguson and D. Senie, “Network ingress filtering: Defeating denial-of-service attacks which employ IP source address spoofing”, RFC 2827, 2000
[9] S. Floyd, S. Bellovin, J. Ioannidis, K. Kompella, R. Mahajan, V. Paxson, “Pushback message for controlling aggregates in the network”, Internet Draft, 2001
[10] J. Gomes, F. Jin, H. Choi and H. Choi, “Enchanced probabilistic packet marking for IP traceback”, in Proceedings of the IEEE Workshop on Information Assurance and Security, pp. 30-37, June 2002
[11] H. Lee, “Advanced packet marking mechanism with pushback for IP traceback”, in ACNS ’04, LNCS 3089, pp. 426-438, 2004
[12] H. C. J. Lee, V. L. L. Thing, Y. Xu and M. Ma, “ICMP Traceback with cumulative path, an efficient solution for IP traceback”, in ICICS 2003, LNCS 2836, pp. 124-135, 2003
[13] S. Savage, D. Wetherall, A. Karlin and T. Anderson, “Network support for IP traceback” in IEEE Transactions on Networking, Vol. 9, No. 3, pp. 226-237, June 2001
[14] M. Shung and J. Xu, “IP traceback-based intelligent packet filtering a novel technique for defending against Internet DDoS attacks”, in IEEE Transactions on Parallel and Distributed Systems, Vol. 14, No. 9, pp. 861-872, September 2003
[15] A. C. Snoeren, C. Partridge, . A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent and W. T. Strayer, “Single-packet IP traceback”, in IEEE/ACM Transactions on Networking, Vol. 10, No. 6, pp. 721-734, December 2002
[16] D. X. Song and A. Perrig, “Advanced and authenticated marking schemes for IP traceback”, in Proceedings of the IEEE INFOCOM, 2001
[17] K. Stefanidis and D. N. Serpanos, “Packet-marking scheme for DDoS attack prevention”, in Proceedings of Security and Protection of Information, 2005
[18] R. Stone, “CenterTrack: An IP overlay network for tracking DoS floods”, in proceedings of 9th Usenix Security Symposium, August 2000
[19] Y. K. Tseng and W. S. Hsieh, “CPPM – Compensated probabilistic packet marking for IP trace backing”, IEICE Transactions on Communications, Vol. E87-B, No. 10, pp. 3096-3098, October 2004
[20] U. K. Tupakula and V. Varadharajan, “A practical method to counteract denial of service attacks”, in Proceedings of the ACSC2003, Vol. 16, 2003

6.3 Cryptography functionalities: An Overview

6.3.1 Lightweight Cryptography (State of the Art)

In symmetric or private key cryptography, a single key is used to perform both encryption and decryption. Symmetric cryptography is widely used due to its performance against asymmetric cryptography. Key distribution and establishment is its main drawback. In order to communicate securely, two entities have to establish a common secret key. If the key is pre-established, then if someone possesses an entity, he can obtain the secret key. For this reason the asymmetric cryptography was introduced. Today, it is used in collaboration with asymmetric cryptography. Symmetric cryptography is used to encrypt/decrypt the main body of data, while asymmetric cryptography is used for symmetric key distribution and digital signatures.

There are two types of symmetric ciphers: the block ciphers and the stream ciphers. Block ciphers use fixed sized blocks of plaintext, while stream ciphers operate on individual bits of plaintext combined with a pseudo-random bit sequence.

Stream ciphers are typically faster and simpler to implement and are better suited for encryption of transmissions of streams of large amount of data. In stream ciphers, keys should never be reused and thus they are considered vulnerable when they are not used carefully.

A block cipher transforms the blocks in a sequence of operations, called rounds. When messages longer than the block size must be encrypted, the original message is partitioned in a sequence of blocks and the encryption of each block depends on the cipher's mode of operation. When messages smaller than the block size must be encrypted, a padding scheme is used to fulfil the missing bits. Some block ciphers can act as stream ciphers when specific modes of operation are used.

AES is the most well-known and studied cipher, and it is used as a comparison unit for the different symmetric cipher proposals. AES is a block cipher that uses 128 and 256 bits keys and acts as stream cipher in CBC mode.

Traditional cryptography implementations focus in providing high levels of security, ignoring the requirements of constrained devices. Lightweight cryptography is a research field that has been developed in recent years due to the widespread use of such systems. LWC is concentrating in implementing cryptographic functionality for devices with constrained capabilities in power supply, connectivity, hardware and software. Hardware implementations are considered more suitable for embedded systems, whereas software and hardware-software implementations are also studied. Hardware implementations try to reduce the number of logic gates that are required to materialize the cipher. This metric is called Gate Equivalent (GE). A small GE predisposes that the circuit will execute its functionality quick and the device will be put in sleep mode, consuming less power. For low cost devices an implementation up to 3000 GE can be acceptable while for even smaller devices, like 4-bit microcontrollers, implementations of 1000 GE are studied. On the contrary, software implementations try to reduce the memory and CPU needs of the cipher.

Lightweight and ultra-lightweight ciphers usually offer 80 to 128 bit security. 80 bit security is considered adequate for constrained devices like the 4-bit microcontrollers. While a security level of 128 bits is typical for mainstream applications, 80 bit security is often a reasonable target for RFID tag based applications. For one way authentication, 64 or 80 bit security could be enough. Traditional cryptographic ciphers, like AES, are still in the foreground as they can provide much larger security levels.

Two main approaches are followed in order to implement lightweight ciphers. In the first case, researchers are trying to improve the performance of well-known and well-studied ciphers such as AES and DES. The state of the art AES [7] hardware implementation uses 2.400 GE. Another approach is to implement new ciphers from the scratch, specific for this domain. PRESENT [8] is such a cipher, which was implemented for lightweight and ultra-lightweight cryptography and one of the first ciphers that offer a 1.000 GE implementation for ultra-constrained devices. Another artefact that is used in order to reduce the

requirements of such ciphers, especially for ultra-lightweight cryptography, is the absence of decryption. This approach is suitable for devices that need only one way authentication. Furthermore, some ciphers propose that the key should be hard-wired on the device to even reduce the GE due to the absence of key generation operations.

6.3.1.1 Block Ciphers

PRESENT is a milestone in the evolution of lightweight block ciphers and is used as the comparison unit for the new lightweight ciphers. It is now under standardization within the upcoming ISO 29192 Standard on Lightweight Cryptography. It uses 64 bits block size and an 80 bit key. The main feature of PRESENT is the replacement of the eight S-boxes with a carefully selected single one. This technique resulted in significant reduction of GE and was later adopted by the posterior ciphers. The design of PRESENT is extremely hardware efficient, since it uses a fully wired diffusion layer without any algebraic unit. Also PRESENT can offer only encryption functionality. In this way it can be used within challenge-response authentication protocols and it could be used for both encryption and decryption of communications to and from the device by using the counter mode.

The KATAN and KTANTA cipher family embraces the hard-wired approach. They are the most hardware efficient block ciphers which require less than 1.000 GE. They provide 80 bit key size and security level and can be scaled down to 462 GE with hardwired key and block size of 32 bits.

Hummingbird is another promising ultra-lightweight cipher, which introduces a hybrid structure of block cipher and stream cipher, with 256 bit key length and 16 bit block size. It has better performance than PRESENT on 4-bit microcontrollers. The next version, Hummingbird-2, optionally produces a message authentication code (MAC) for each message processed and was developed with both lightweight software and lightweight hardware implementations for constrained devices in mind. Hummingbird-2 has 128 bit secret key and 64 bit IV and as its predecessor has been targeted for low end microcontrollers and for hardware implementation in lightweight devices such as RFID tags and wireless sensors. It is believed to be resistant to all standard attacks to block and stream ciphers and is also resistant to chosen-IV attacks. The implementation requires little more than 2.000 GE. Its main drawback is the lengthy initialization process due to its stream cipher feature.

Newer block ciphers include the PRINTcipher, EPCBC, Klein, LED and Piccolo. All these new ciphers combine several techniques that are proposed by the former ciphers and try to increase the resistance in several attacks that were confirmed in AES, PRESENT and other ciphers. The ciphers should be extensively tested for security vulnerabilities before being widely used.

6.3.1.2 Stream Ciphers

Despite the evolution effort in the field of lightweight stream ciphers, they still remain inferior to lightweight block ciphers. The major drawback of stream ciphers is the lengthy initialization phase prior to first usage. The most remarkable of them are the two finalists of the eSTREAM project the Grain [9] and TRIVIUM [10]. The key size of Grain is 80 bits and the IV 64 bits and it requires about 1.300 GE to implement. TRIVIUM comes up with an 80 bit secret key, an 80 bit IV and about 2600 GE to implement and it was designed as an exercise in exploring how far a stream cipher can be simplified without sacrificing its security, speed or flexibility. The two ciphers are the more accepted ones and their resistance on a series of known stream ciphers' vulnerabilities is investigated. PRESENT was also a candidate for eStream hardware implementation so it could be taken into account as an alternative. Also Salsa20/12 is an eStream finalist for software implementation. It uses 256-bit keys and 128-bit initialization vectors.

6.3.1.3 Hash Functions

The hash functions are another research field of LWC. The standardized SHA-1 and SHA-2 are too large to fit in hardware constrained devices. The NIST's SHA-3 competition is going to define a new function to replace the older SHA-1 and SHA-2 in 2012. The finalists in SHA-3 are BLAKE, Grostl, JH, Keccak and Skein. Unfortunately, the SHA-3 finalists aren't much more compact than their antecessors. At the time, all SHA-3 finalists require more than 12.000GE for 128 bit security. Other state of the art lightweight hash functions include Quark, Spongent, Vitamin-H and PHOTON.

PHOTON is hardware oriented and uses the sponge functions framework to keep the internal memory size as low as possible. It requires about 1120 GE for 64 bit collision resistance security.

Spongent processes hash sizes of 88, 128, 160, 224 and 256 bit based on a sponge construction instantiated with a PRESENT-type permutation, following the hermetic sponge strategy. Its smallest implementations require 738, 1060, 1329, 1728 and 1950 GE, respectively. It has considerably smaller footprint than SHA-2, SHA-3 finalists, PRESENT in hashing modes and QUARK.

6.3.1.4 Selection Criteria

The criteria that are used to classify the different implementations of lightweight ciphers are:

- ***Quantitative***
  - The ***footprint / surface area***: the size of the hardware implementation (like GE or FPGA slices)
  - The ***code size:*** the code size for software implementations
  - The ***memory elements:*** the memory that is required for software implementations
  - The ***key size:*** the different key sizes in bits that are supported.
  - The ***level of security***: the bits of security that are offered for each key size
  - The ***performance***: how fast the cryptosystem can operate its functionality
  - The ***power and energy consumption:*** this factor can be of great importance for low cost devices with constrained power and energy resources
  - The ***size of the output***: the transmission time and the number of messages that must be transmitted in order to send a cipher text to another entity depends on that size.
  - The ***implementation cost:*** inexpensive implementations should be proposed to allow the widespread use.
- ***Qualitative***
  - ***Analysis:*** the proposed implementations should be well examined against known types of vulnerabilities and analysis should be made – cryptanalysis, simple power analysis (SPA), differential power analysis (DPA).
  - ***Acceptance:*** the proposed implementations should be compliant with global export control regulations in order not to restrict international trade.
  - ***Key Parameterization:*** key parameterization options should be supported in order to map the requirements of specific application-scenario.
  - ***Cryptographic improvement:*** there should be an improved global architecture of the embedded SW to support future evolution of cryptographic functionalities.

6.3.1.5 nSHIELD: proposed solutions

We propose the investigation of

- AES & PRESENT, as a symmetric block cipher,
- the Grain, as a symmetric stream cipher and
- ECC & NTRU for public key cryptography and signatures

PRESENT may also be a candidate for eStream hardware implementation. Regarding hash functions, nSHIELD could investigate the SHA-3 finalists.

6.3.2 Asymmetric Cryptography (State of the Art)

Asymmetric or public key cryptography makes use of a pair of keys. One key is kept secret and is called private key, while the other is published and is called public key. When data are encrypted by one key, they can only be decrypted by the corresponding second key of the pair. When an entity encrypts data with its private key, they can be decrypted only by the relevant public key, providing authentication of the source. When an entity encrypts data with the public key of a second entity, the data can only be decrypted by the second's entity private key, ensuring that the data can be accessed only by this entity,

and hence establishes confidentiality. The robustness of the asymmetric cryptography depends on the ability of the attackers to correlate the public with the relevant private key of the pair. Although asymmetric cryptography is slower and demands more resources than symmetric cryptography, the asymmetric cryptography is mainly used for secret/session key distribution and electronic signatures, due to the main drawback of symmetric cryptography to provide dependable key distribution mechanisms. The wide use of embedded systems emerges the development of asymmetric cryptology for low cost nodes.

Traditional public key cryptography is based on one-way trapdoor functions. The functions are based on a set of hard mathematical problems. There are three well established cryptosystems:

- RSA – which is based on the Integer Factorization Problem (FP)
- ElGamal – which is based on the Discrete Logarithm Problem In Finite Fields (DLP)
- ECC/HECC – which are based on Elliptic Curve Discrete Logarithm Problem (ECDLP)

Implementations in software, hardware and co-design of both have been invoked for different applications.

6.3.2.1 Traditional Public Key Cryptosystems Comparison

RSA is the most known and widespread algorithm for asymmetric cryptography and supports key sizes from 1024 to 4096 bits. It is used as a comparison unit for the different proposed public key cryptosystems. However, its large hardware footprint and its resource demanding implementations, led researchers to seek for other algorithms for applications in constrained devices.

ECC [1] and HECC [2] are considered the most attractive cryptosystem for embedded systems. They present smaller operand lengths and relatively lower computational requirements. Their main advantage is that they offer shorter keys for the same level of security than RSA. As the level of security goes high, RSA has key sizes growing much faster than ECC. ECC also produces lightweight software implementations due to its memory and energy savings. The most known software implementations [3] are the TinyECC and the WMECC.

A hyper elliptic curve of genus 1 is an elliptic curve. HECC is a generalization of elliptic curves. As the genus goes higher, the arithmetic of encryption gets complicated, but it needs less bits for the same level of security. HECC's operand size is at least a factor of two smaller than the one of ECC. The curves of genus 2 are of great interest for the research community as higher genus curves suffer from security attacks. HECC has better performance than ECC and is more attractive in resource constrained devices.

ElGamal produces no interest for resource constrained platforms. The computation is more intensive than RSA and encryption produces a 2:1 expansion in size from plaintext to ciphertext. It is also considered vulnerable to some types of attacks, like chosen ciphertext attacks.

6.3.2.2 Alternative Public Key Cryptography (APKC)

Alternative public key cryptosystems [4] that follow a different approach become popular due to their performance and their resistance against quantum computing. These alternative cryptosystems are based on:

- Hash-Based Cryptography – general hash functions are used as a base operation for generating digital signatures.
- Code-Based Cryptography – McElice is the most popular scheme which is based on error-correcting codes.
- Multivariate-Quadratic Cryptography – is based on the problem of solving multivariable quadratic equations over finite fields.
- Lattice-Based Cryptography – NTRU is the most popular scheme which is based on the Shortest Vector Problem.

NTRU [5], [6] is the most promising cryptosystem of all APKCs. NTRU is specified by three integer parameters (N, p, q):

- N-1 – the maximal degree for all polynomials in the truncated ring R

- p – a small modulus
- q – a large modulus

where it is assumed that N is prime, q>p and p,q are coprime. Encryption and decryption use only simple polynomial multiplication, which makes them very fast compared to traditional cryptosystems. NTRU is considered to be highly efficient and suitable for embedded systems, while providing a comparable level of security to that of RSA and ECC. In comparison to ECC, NTRU is 1,5 times faster and has only 1/7 memory footprint of ECC at the same level of security in hardware. NTRU's software implementation, in comparison to RSA, is 200 times faster in key generation, the encryption is almost 3 times faster and the decryption is about 30 times faster. On the other hand, NTRU produces larger output, which may impact the performance of the cryptosystem if the number of transmitted messages is crucial. It is considered safe when recommended parameters are used. NTRU can be efficiently used in embedded systems because of the easy key generation process the high speed and the low memory usage. Now the system is fully accepted to IEEE P1363 standards under the specifications for lattice-based public-key cryptography IEEE P1363.1 and to X9.98 Standard for use in the financial services industry.

The main drawback of McEliece and MQ cryptosystems is the large key sizes. In comparison to RSA with 1924 bit key, MQ requires 9690 bytes for the public key and 879 bytes for the private key. The key sizes impact the computations that are performed, the speed, the key storage and the output's size. The advantage of these systems is the fast encryption and decryption process that make them suitable for high performance applications where messages must be assigned in real time.

6.3.3 Dependable Authentic Key Distribution (State of the Art)

Secret key establishment is one of the most fundamental operations for all kind of applications where security is concerned. As it has been described above, the conventional key management techniques utilize the public key cryptography for this purpose. Unfortunately, the limited resources of many embedded devices restrict the use of these key management techniques since their implementations in these systems are slow. Thus, researchers have proposed lightweight key management solutions that are based on symmetric cryptography. There are many proposed schemes [11], [12] for lightweight key distribution and each one tries to address several problems that can occur. The research in the field results that the pre-distribution strategy usually involves less resources and the distributed homogeneous model is more general. Also the suitability of a scheme depends on the application's needs in terms of security, performance and flexibility. In the sections below, there are presented the basic ideas and some of the newest schemes.

6.3.3.1 The Basic Scheme

Eschenauer and Gligor's method [13] is considered as the 'Basic Scheme'. In key pre-distribution phase, a large key pool is initialized with random keys and their respective identifiers. For each node, k keys are drawn at random from the pool. These keys are loaded into the node's memory forming its key chain. The method makes use of the random graphs theory, and selects the size of the pool and k in such a way that each pair of nodes shares at least one key with an arbitrary probability.

In share-key discovery phase, each node broadcast the identifiers of its key chain, allowing neighbouring nodes to identify the common keys. The common keys can be later used to secure a communication between these nodes. Variants of this approach propose a challenge-response operation to improve security. The disadvantage of this strategy is the greater communication and processing overhead.

In path-key establishment phase, a pair of nodes having no common keys must find an intermediary node. Any node, whose key chain contains keys that are present in both nodes' chain, is a suitable candidate. Upon request, the intermediary can choose unassigned keys from its key chain in order to create an indirect link between the pair of nodes.

The Basic Scheme is simple as it is interesting at providing a connected network with reduced amount of memory for storing keys. However, it has some disadvantages like lack of node to node authentication.

The importance of the scheme is based on the fact that ideas that were introduced by this method are utilized by the later proposed methods which aim to overcome its limitations.

6.3.3.2 Location-independent key distribution schemes

Well known location-independent key distribution schemes includes the Blom's method, the method by Blundo et al, the Basic Scheme, the q-composite key pre-distribution scheme, the multipath key reinforcement, the multiple space key pre-distribution scheme, the polynomial pool-based key pre-distribution scheme, the combinatorial design-based key pre-distribution scheme, the expander graph-based key pre-distribution scheme, the PIKE, the random assignment set selection key pre-distribution scheme, the pseudo-random function-based key pre-distribution scheme, the method by Tsai et al, the BABEL key pre-distribution scheme, the method by Law et al, the random perturbation-based key establishment scheme and the non-interactive key establishment scheme.

Table 7 - Classification criteria for authentic key distribution

Efficiency metrics	***Memory:*** the amount of memory that is needed for storing data
	Processing: the number of processor cycles needed to establish keys
	Bandwidth: the amount of data exchanged between nodes during the key generation process
	Energy: the energy consumption involved in the key agreement process
	Key Connectivity: the probability that nodes are able to established shared keys. When only the connectivity between a pair of neighbour nodes is considered, the metric is called local connectivity. When the connectivity of the whole network is considered, the metric is called global connectivity
Security metrics	***Node authentication:*** the key management technique should guarantee mutual identity authentication for the communicating nodes in a secure way
	Resilience: refers to the resistance of the scheme against node capture. The scheme's resilience is given by the fraction of the network communications that are exposed to an adversary, excluding the communications in which the compromised node is directly involved
	Node revocation: upon the discovery of compromised nodes, the scheme should provide efficient ways to dynamically revoke them from the network
Flexibility metrics	***Lack of prior deployment knowledge:*** nodes are deployed dynamically and at random. More flexible techniques do not depend on the nodes positioning for initializing the network keys
	Scalability: during the network lifetime, its size may vary dynamically. The scheme must support large networks and allow the introduction of new nodes without loss of security

6.3.3.3 Location-dependent key distribution schemes

On the contrary, well known location-dependent key distribution schemes include the LEAP key distribution scheme, the group-based key distribution scheme, the attack probability-based distribution scheme, the location-aware key establishment (LKE) key distribution scheme, the traversal design-based key distribution scheme and the secure walking Global Positioning System (GPS) key distribution scheme.

6.3.3.4 nSHIELD: proposed solutions

The suitability of a scheme for key distribution depends on the application's needs in terms of security, performance and flexibility. In nSHIELD we will investigate the use of several schemes for the four scenarios, taking into account the special characteristics of every one of them.

Key distribution mechanisms that make use of asymmetric cryptography can be used by power nodes, as these mechanisms demand more resources but they are more robust. Lightweight key distribution mechanisms that make use of symmetric cryptography can be used by low power nodes.

6.4 SPDs (from pSHIELD to nSHIELD)

Table 8 - SPDs

SPD	pSHIELD	nSHIELD
3. Automatic Access Control and Denial-of-Service	Mentioned in proposal and briefly investigated in D3.4. Access control (IEEE 802.15.4, Wireless Medium Access Control) Denial of Service (physical damage, jamming of communication lines, system overloading, attacks on the system's power lines, battery depletion attacks)	WP3, T3.4 [The state-of-the-art can be found in ***Section 0***] "As part of the activities in SHIELD, it is planned to address the critical design steps that will enable node firmware/software as well as network protocols in an SPD node environment which are resilient to DDoS attacks in conjunction with the implementation of basic access control mechanisms that a node should provide to the applications. Another step is to realize and handle DDoS vulnerabilities in a shared node environment where the possible attacker is an insider who already has the necessary credentials and wants to degrade service availability of part of the node network for his own purposes (per example shared face recognition devices installed on airport gates)." [From ***TA_nSHIELD – 2.2.2 – Progress in specific SPD technologies as expected output of the project***]

4. Lightweight Hardware and Software crypto technologies	In ***D3***, asymmetric cryptography implementations of ECC and RSA were investigated. Asymmetric cryptography was used in order to exchange symmetric keys. Furthermore, the use of SHA-1, AES in CBC mode and a random number generator (RNG) was proposed. Alternative algorithms were considered, some offering better performance, like NTRU for asymmetric as well as PRESENT and Hummingbird for symmetric cryptography. These weren't investigated further. In addition the use of a TPM to store keys and improve performance and security was proposed. In ***D6.1***, a prototype was implemented using the Blowfish cipher. Finally in ***D6.2*** the following were used: AES with key sizes 128bits and 256 bits ECC/ECIES (TinyECC S/W) with key size 160 bits	WP3, T3.5 We propose the investigation of AES & PRESENT, as a symmetric block cipher, the Grain, as a symmetric stream cipher and ECC & NTRU for public key cryptography and signatures PRESENT was also a candidate for eStream hardware implementation, so we could also take it into account as an alternative. Regarding hash functions, we could investigate the SHA-3 finalists. [The state-of-the-art can be found in ***Section 6.3.1 and 6.3.2***]
12. Asymmetric Cryptography for low cost nodes	See SPD 04 above	WP3, T3.5 See SPD 04 above
13. Reputation based schemes for secure routing and intrusion detection system	***D4.2*** describes the proposed IDS which features a distributed architecture and is implemented through a hybrid anomaly detection system. In this system every node runs a detection system, which is in charge of identifying nearby suspicious nodes. These suspicious nodes are temporarily blacklisted and an alarm is sent to the central agent. The central node gathers information from the rest of the nodes and in the case of a false alarm sends a message of false positive to the first node to erase the node from blacklist. If it is a true alarm, the central node will report it to the rest of nodes, in order to have them blacklist the suspicious node. This solution combines misuse and anomaly based techniques in a distributed hierarchy for improving resilience and performance.	WP4, T4.3 "SHIELD will go beyond the state-of-the-art in this technology by adapting it to a mobile ad-hoc environment. In such a network, it may be difficult for the reputation upgrading process to cope up with the node mobility and it might not be appropriate to depend solely upon personal observation. Using second hand information can significantly accelerate the detection and subsequent isolation of malicious nodes in MANETS". [From ***TA_nSHIELD – 2.2.2 – Progress in specific SPD technologies as expected output of the project***]
14. Anonymity and Location-privacy techniques	Mentioned in proposal but no significant research appears on deliverables.	WP4, T4.4 This SPD will be mainly investigated in the nSHIELD scenario of social mobility. We can propose relevant state of the art techniques,

		considering the updated case studies.
15. Reputation based security resource Management Procedures	Mentioned in Proposal as part of Task 4.3 and D3.2 touches the topic for NMP nodes but no visible research @ WP4's deliverables.	WP4, T4.3 "In order to improve this technology, SHIELD project will design an abstract layer that will consider device's security as a service, so that SHIELD project could control the security of one resource and transactions among resources. This will control also traceability and dependence among resources. This remote control of TPM – reputation based- can identify malicious use, corruption and perform a secure flow control of the job." [From ***TA_nSHIELD – 2.2.2 – Progress in specific SPD technologies as expected output of the project***]
18. Dependable authentic key distribution mechanism	In ***D3***, public key cryptography is proposed in order to exchange symmetric keys. ***D4*** examines key distribution mechanisms that rely solely on symmetric cryptography. In ***D6.1***, a key exchange protocol, namely 'Control Randomness Protocol', is implemented. On the first phase, a public key cryptography scheme is used in order to exchange the bundle of symmetric keys that will be used on the second phase. During the second phase, those keys are being used as input for a symmetric key cryptography scheme that handles the actual data exchange. In ***D6.2***, the WSN sensors distribute cryptographic keys in accordance with a WSN broadcast key distribution method (when used AES) and pre-distribution key distribution method (when used ECC).	WP4, T4.4 The suitability of a scheme for key distribution depends on the application's needs in terms of security, performance and flexibility. In nSHIELD we will investigate the use of several schemes for the four scenarios, taking into account the special characteristics of every one of them. Key distribution mechanisms that make use of asymmetric cryptography can be used by power nodes, as these mechanisms demand more resources but they are more robust. Lightweight key distribution mechanisms that make use of symmetric cryptography can be used by low power nodes. [The state-of-the-art can be found in ***Section 6.3.3***]
19. Secure service discovery, composition and delivery	In ***D5.2*** the secure service management is described. In the proposed prototype the OSGI framework is used. In ***D5.4*** several prototypes were	WP5, T5.2

protocols	implemented that use: ***OSGI framework*** to perform Middleware Core Services for discovery and composition of pSHIELD components ***OWL file*** representing the pSHIELD ontology that, together with the pSHIELD middleware, makes the composition possible. In particular this prototype includes the reasoning for Common Criteria compliant composition of SPD metrics. ***Architectural design*** and performance analysis of a Policy Based approach by which the middleware composition could be driven Matlab ***simulation*** and theoretical ***formalization*** of a Hybrid Automata approach to drive the SPD composition in a context-aware way More technologies for service management are presented in ***D5.3*** and ***D5.4.*** In ***D6.1*** the proposed implementations are revisited.	"SHIELD will implement (and if possible) refine these specification to release a very first implementation of some of these mechanism; among them, the most interesting issues is the definition of WS-Security Policy. WS-Security Policy is a standard that regulate a security assertion model, a security binding abstraction and policy considerations." [From ***TA_nSHIELD – 2.2.2 – Progress in specific SPD technologies as expected output of the project***]
21. Policy-based SPD management	***D5.2*** and ***D5.4*** provides an overview of the state-of-the-art for policy-based management (PBM). XACML is recommended for policy specification. In ***D6.1*** and ***D6.3*** a prototype is implemented using XACML and Hybrid-Automata model.	WP5, T5.3 "In this aspect, SHIELD will implement the technologies to provide the ES networks with the ability to adapt the policies at runtime to changes in the environment to react to ongoing attacks. These technologies will be developed starting from the concepts and technologies developed in the SERENITY and MASTER projects (both dealing with dynamic policy management), and adapting them to the particularities of Embedded systems. The objective is that the provided policy management framework is not simply an adaptation of an existing one, but, on the contrary, designed for the particularities of the ESs." [From ***TA_nSHIELD – 2.2.2 – Progress in specific SPD technologies as expected output of the project***]

6.5 Elliptic Curve Cryptography

Elliptic Curves combine different and very diverse areas of mathematics: algebraic number theory, algebraic geometry, complex analysis, representation theory. These have many applications. Those results based on number theory, have very useful applications to cryptography, especially, public key cryptography. Elliptic curves are a fairly diverse, long established branch of mathematics.

Elliptic Curve Cryptography (ECC) is emerging as an attractive public-key cryptosystem for mobile/wireless environments. Compared to traditional cryptosystems like RSA, ECC offers equivalent security with smaller key sizes, which results in faster computations; lower power consumption, as well as memory and bandwidth savings. This is especially useful for mobile devices which are typically limited in terms of their CPU, power and network connectivity. nSHIELD will exploit ECC for the implementation of public-key cryptographic solutions.

6.5.1 Theoretical Foundations

An elliptic curve is a mathematic space suitable for solving a range of problems. By definition [tate], an elliptic curve is an algebraic curve of genus one, also called abelian variety, since it defines an algebraic multiplication.

A possible application of elliptic curves is the demonstration of Fermat's Last Theorem, which states that no positive integers *a*, *b* and *c* can satisfy the equation $a^n + b^n = c^n$ for any integer exponent n > 2. The Frey–Hellegouarch elliptic curve $y^2 = x(x - a^n)(x + b^n)$ was proposed to solve the theorem. A second example of application is the congruent number problem solution. Another elliptic curve application relies on congruent number problem. A congruent number is a positive rational number equivalent to the area of a right triangle with three rational number sides [koblitz1993]. In [tunnell] and [koblitz1993] is shown a link between the question of whether a given number is congruent and the condition that an elliptic curve has positive rank.

An integer factorization algorithm based on elliptic curve properties was proposed by Lenstra [lenstra1987], and this research posed first ideas for the application of elliptic curves to the public key cryptography.

The application of elliptic curves to public key cryptography was proposed by Koblitz [koblitz1987] and Miller [miller] in 1985. In the following years a large number of publications in literature focused on implementing and enforcing security and efficiency of elliptic curve cryptography (ECC). Public key cryptography algorithm was first proposed by Diffie and Hellman [diffie] in 1976, and the well-known first implementation named RSA was performed by Rivest, Shamir and Adleman [rivest] in 1977.

The cryptographic security of RSA implementation is based on the hardness of the integer factorization problem. Analogously, ECC is characterized by the hardness of the elliptic curve discrete logarithm problem, whose time of evaluation represents the strength of the encryption infrastructure, and that overcomes the complexity of the integer factorization problem.

The most difficult integers to factor in practice, using existing algorithms, are those obtained by the product of two large primes of similar size. All cryptographic application exploits only this type of integers. No known algorithms can solve the integer factorization problem in a polynomial time. The work from Montgomery [montgomery1994] presents a number of integer factorization algorithms divided in two categories: algorithms that find small prime factors quickly, and algorithms with complexity depending of the number to be factorized. The following algorithms belong to the first category, and are practically inapplicable to true cryptology problems.

Trial division is an algorithm that presents a complexity of $O(p/\log p)$, where *p* is the second largest prime factor of factorized integer. Pollard's rho algorithm [pollard1978] presents a complexity of $O(p^{1/2})$ to find a factor *p*. Another algorithm invented by Pollard, called "P - 1 method" [pollard1974], can find small prime factors with complexity $O(p)$ in the worst case, but some conditions can decrease dramatically the

discovery time. These conditions includes that *p* is small and *p*-1 is a smooth number (integer which factor completely into small primes [hellman]). The so called Step 2 [brent] method is a variation of the P - 1 method with a bit relaxed constraints on number *p*-1, and complexity almost the same of previous algorithm. Finally, the variant of "P – 1" algorithm called "P + 1" [williams] reduces the failures given by (*p*-1) numbers with very large factors. This algorithm requires that the p + 1 number is a smooth number. But for certain values of p such that $p^2 - 4$ is a quadratic residue, this last method becomes a more expensive variant of the "P – 1" method. The latest integer factorization algorithm analyzed in this work is the elliptic curve method. Recall that "P – 1" and "P + 1" methods require that *p* - 1 and *p* + 1 numbers do not have large prime factors. As stated in [montgomery1994], this condition appears frequently, preventing the latter methods from working. The elliptic curve method already named [lenstra1987] is able to work when "P – 1" and "P + 1" are not applicable.

Despite of last six methods, the following algorithms work for an arbitrary odd integer *N* in order to find two numbers *X* and *Y* such that $X^2 \equiv Y^2 \ (mod\ N)$.

Finding squares through products is described in [coppersmith1993, coppersmith1994] and with other variants; it finds *X* and Y numbers with complexity of about $O(n^3)$. The continued fraction method (CFRAC) is fast but does not find all possible factors, while it looks for congruencies $X^2 \equiv r \ (mod\ N)$ with small *r*. Quadratic sieve [pomerance] and Montgomery's Multiple Polynomial Quadratic Sieve [silverman] are also fast methods but cannot return all possible factors of N. The best known algorithm for integer factorization, in terms of complexity, is the General Number Field Sieve (GNFS) [lenstra1993], that presents a sub-exponential complexity. For a given number n, the computational complexity of GNFS is $O(e^{O((\log n)^{1/3}(\log\log n)^{2/3})})$.

All the algorithms for integer factorization do not seem to be applicable to find the solution of the analog problem over elliptic curves [miller][koblitz1987].

Suitable algorithms for solving the elliptic curve discrete logarithm problem (ECDLP) are the Pohlig-Hellman attack [pohlig], the Pollard's rho attack, the baby-step giant-step attack, the index-calculus attack, the isomorphism attacks [hankerson]. With an optimal selection of the cryptosystem parameters, for example the order of the chosen curve field is divisible by a large prime; the best general-purpose attack known on the ECDLP is the combination of the Pohlig-Hellman algorithm and Pollard's rho algorithm. This technique presents a fully-exponential complexity $O(n^{1/2})$ [hankerson].

Three well known types of public key cryptosystems are compared in Table 9. As shown in the last column, RSA, Diffie-Hellman algorithm and Digital Signature Algorithm (DSA) can be attacked with algorithms with sub-exponential running time. The best known attack on elliptic curve cryptography systems requires exponential time instead. For this reason, elliptic curve cryptography offers the same security level of a given public key cryptographic system with substantially smaller key sizes.

Some comparison of the key length necessary to achieve some comparable security level, in the case of symmetric key systems, finite field systems, integer factorization systems and ECC system are presented in [fips800] and are reported in Table 10.

Table 11 [gupta] presents the estimated workload necessary to evaluate the attacking algorithms for presented cryptosystems. This workload expressed in MIPS year (millions of instructions per second per year) is compared with the minimum acceptable security level. From [certicom-A], one can see that in July of year 2000 the value of 10^{12} MIPS year was considered as the acceptable security level. Today, from [gupta] one can note that 10^{12} MIPS year was acceptable until year 2010. This gap is due to the increasing of computational power following Moore's Law. In this case is very clear the problem of balancing cryptographic system efficiency and overall security performances.

Table 9 - Public key cryptosystems comparison

Public key system	*Algorithm examples*	*Mathematical problem*	*Best solving algorithm and complexity*
Integer factorization	RSA, Rabin-Williams	Given number *n*, find prime factors	General number field sieve, sub-exponential complexity: $\exp [O((\log n)^{1/3} (\log \log n)^{2/3})]$
Discrete logarithm	DH, DSA, ElGamal	Given numbers *g, h* and prime *n*, find *x* such that $h = g^x \bmod n$	General number field sieve, sub-exponential complexity: $\exp [O((\log n)^{1/3} (\log \log n)^{2/3})]$
Elliptic curve discrete logarithm	ECDH, ECDSA	Given an elliptic curve E and points P,Q over E, find k such that $kP = Q$	Pohlig-Hellman and Pollard's rho, exponential complexity: $n^{1/2}$

a L is the public key size, N is the private key size.

Table 10 - Equivalent key bit lengths in terms of security level for different cryptographic schemes

Symmetric key cryptography (e.g. SKIPJACK, TDES, AES)	*Finite field cryptography[a] (e.g. DH, DSA)*	*Integer factorization cryptography (e.g. RSA)*	*Elliptic curve cryptography (e.g. ECDSA)*
80	L = 1024, N = 160	1024	160
112	L = 2048, N = 224	2048	224
128	L = 3072, N = 256	3072	256
192	L = 7680, N = 384	7680	384
256	L = 15360, N = 512	15360	512

aL is the public key size, N is the private key size.

A different complexity of the algorithms that solve ECDLP, with respect to the ones devoted to solve integer factorization, entails that with elliptic curve cryptosystems the desired security level can be attained with significantly smaller keys than is possible with their RSA counterparts. Smaller key sizes involve more speed, and more efficiency in bandwidth and power consumption.

Table 11 - Protection lifetime considerations among different key sizes

Integer factorization cryptography (e.g. RSA)	*Elliptic curve cryptography (e.g. ECDSA)*	*MIPS Years necessary to attack*	*Protection lifetime*
1024	160	10^{12}	Until 2010
2048	224	10^{24}	Until 2030
3072	256	10^{28}	Beyond 2031
7680	384	10^{47}	
15360	512	10^{66}	

aL is the public key size, N is the private key size.

6.5.1.1 Finite Fields

The efficient implementation of finite field arithmetic is an important prerequisite in elliptic curve systems because curve operations are performed using arithmetic operations. Three kinds of fields that are especially useful for an efficient implementation of elliptic curve systems are prime fields, binary fields, and optimal extension fields. In this report will be discussed prime fields and binary fields because they are helpful for the section 3.

Fields are abstractions of familiar number systems (such as the rational numbers Q, the real numbers R, and the complex numbers C) and their essential properties. They consist of a set F together with two operations, addition (denoted by +) and multiplication (denoted by ·), that satisfy the usual arithmetic properties:

- *(*F,+*)* is an abelian group with (additive) identity denoted by 0.
- *(*F\{0}*, ·)* is an abelian group with (multiplicative) identity denoted by 1.
- The distributive law holds: *(a + b) · c = a · c + b · c* for all *a, b, c* ∈ F.

If the set F is finite, then the field is said to be *finite*.

A field F is equipped with two operations, *addition* and *multiplication*. *Subtraction* of field elements is defined in terms of addition: for $a, b \in F, a - b = a + (-b)$ where *−b* is the unique element in F such that *b+(−b)* = 0 (*−b* is called the *negative* of *b*). Similarly, *division* of field elements is defined in terms of multiplication: for $a, b \in F$ with $b \neq 0$, $a/b = a \cdot b^{-1}$ where b^{-1} is the unique element in F such that $b \cdot b^{-1} = 1$ (b^{-1} is called the *inverse* of *b*).

The *order* of a finite field is the number of elements in the field. There exists a finite field F of order *q* if and only if *q* is a prime power, i.e., $q = p^m$ where *p* is a prime number called the *characteristic* of F, and *m* is a positive integer. If *m* = 1, then F is called a *prime field*. If *m* ≥ 2, then F is called an *extension field*. For any prime power *q*, there is essentially only one finite field of order *q*; informally, this means that any two finite fields of order *q* are structurally the same except that the labeling used to represent the field elements may be different. We say that any two finite fields of order *q* are *isomorphic* and denote such a field by F_q.

A subset *k* of a field *K* is a *subfield* of *K* if *k* is itself a field with respect to the operations of *K*; in this instance, *K* is said to be an *extension field* of *k*. The subfields of a finite field can be easily characterized: a finite field F_{p^m} has precisely one subfield of order p^l for each positive divisor *l* of *m*; the elements of this subfield are the elements $a \in F_{p^m}$ satisfying $a \cdot p^l = a$. Conversely, every subfield of F_{p^m} has order p^l for some positive divisor *l* of *m*.

The finite field F_{q^n} can be viewed as a vector space over its subfield F_q . Here, vectors are elements of F_{q^n} , scalars are elements of F_q, vector addition is the addition operation in F_{q^n} , and scalar multiplication is the multiplication in F_{q^n} of F_q- elements with F_{q^n} - elements. The vector space has dimension *n* and has many bases. If *B* = $\{b_1, b_2, \ldots, b_n\}$ is a basis, then $a \in F_{q^m}$ can be uniquely represented by an *n - tuple* $(a_1, a_2 \ldots, a_n)$ of F_q - elements where *a* = $a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$. For example, in the polynomial basis representation of the field F_{p^m} described above, F_{p^m} is an *m* dimensional vector space over F_p and $\{z^{m-1}, z^{m-2}, \ldots, z^2, z, 1\}$ is a basis for F_{p^m} over F_p.

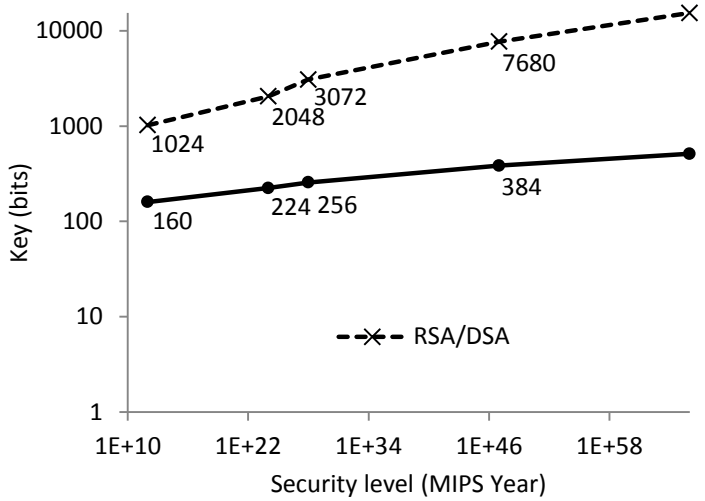

Figure 28 - Security level of integer factorization cryptography systems and elliptic curve cryptographic systems.

The nonzero elements of a finite field F_q , denoted F_q^*, form a cyclic group under multiplication. Hence there exist elements $b \in F_q^*$ called *generators* such that:

$$F_q^* = \{b_i : 0 \leq i \leq q-2\}.$$

The *order* of $a \in F_q^*$ is the smallest positive integer *t* such that $a^t = 1$. Since F_q^* is a cyclic group, it follows that *t* is a divisor of *q* −1.

6.5.1.2 Prime Fields

Let *p* be a prime number. The integers modulo *p*, consisting of the integers {0,1,2, . . . , *p* −1} with addition and multiplication performed modulo *p*, is a finite field of order *p*. We shall denote this field by F_p and call *p* the *modulus* of F_p. For any integer *a*, *a* mod *p* shall denote the unique integer remainder *r*, 0 ≤*r* ≤ *p*−1, obtained upon dividing *a* by *p*; this operation is called *reduction modulo p*.

6.5.1.3 Binary Fields

Finite fields of order 2^m are called *binary fields* or *characteristic-two finite fields*. One way to construct F_{2^m} is to use a *polynomial basis representation*. Here, the elements of F_{2^m} are the binary polynomials (polynomials whose coefficients are in the field $F_2 = \{0,1\}$) of degree at most *m* −1:

$$F_{2^m} = \{a_{m-1}z^{m-1} + a_{m-2}z^{m-2} + \cdots + a_2z^2 + a_1z + a_0 : a_i \in \{0,1\}\}$$

An irreducible binary polynomial *f (z)* of degree *m* is chosen (such a polynomial exists for any *m* and can be efficiently found). Irreducibility of *f (z)* means that *f (z)* cannot be factored as a product of binary polynomials each of degree less than *m*. Addition of field elements is the usual addition of polynomials, with coefficient arithmetic performed modulo 2. Multiplication of field elements is performed modulo the *reduction polynomial f(z)*. For any binary polynomial *a(z)*, *a(z)* mod *f(z)* shall denote the unique remainder polynomial *r(z)* of degree less than *m* obtained upon long division of *a(z)* by *f(z)*; this operation is called *reduction modulo f(z)*.

Assuming we have a multiplicative cyclic group G of order n with generator *g*. The discrete logarithm problem is to find *x* given $y = g^x$ e *g*.

In order for the discrete logarithm is difficult to solve we have to do a proper choice of the group, that is operate in large groups where the discrete logarithm operation becomes intractable.

Here it is used a multiplicative language; in additive language, the problem becomes finding *x* given *y = xg* and *g*. In the second context we can find elliptic curves: the elements of the group are points and the discrete logarithm on elliptic curve, for the same number of bits of the order of the group, is much more difficult to solve compared to conventional discrete logarithm, and therefore inherently more secure.

6.5.2 Elliptic Curves

Given a prime number p different from 2 and 3, and F_p the respective field: an elliptic curve *E* on F_p is denoted with *E*(F_p) and it is defined by the equation:

$$y^2 = x^3 + ax + b$$

where $a, b \in F_p$ satisfy $4a^3 + 27b^2 \neq 0 \bmod p$ (like asking that the polynomial in x don't have multiple roots). A couple (*x*, *y*) where $x, y \in F_p$ is defined a point of the curve if satisfy the previous equation on F_p. It is postulated the *point at infinity*, and it is denoted by ∞. Other three equations are valid for fields different from 2 and 3: this is due to the fact that the equation presented here is a simplified form of the general equation valid for any field, the Weierstrass equation.

Following there is an example of an elliptic curve:

Let $y^2 = x^3 + 2x + 4$ an elliptic curve on F_7, and its point are:

$$E(F_7) = \{\infty, (0,2), (0,5), (1,0), (2,3), (2,4), (3,3), (3,4), (6,1), (6,6)\}$$

There is a method for adding two points on an elliptic curve to produce a third: the addition rule requires some arithmetic operations of addition with the coordinates of points to be added. With this addition rule, the set of points of an elliptic curve make a group with the point at infinity which acts as identity: for this reason $Q = dP$ indicates the sum operation repeated d times of the point *P* to generate the point *Q*.

Let *E* be an elliptic curve defined over the field *K*. There is a *chord-and-tangent rule* for adding two points in *E(F_p)* to give a third point in *E(F_p)*; this rule is right only for real numbers. Together with this addition operation, the set of points *E(F_p)* forms an abelian group with ∞ serving as its identity. It is this group that is used in the construction of elliptic curve cryptographic systems.

The addition rule is best explained geometrically. Let P = (x1 ,y1) and Q = (x2, y2) be two distinct points on an elliptic curve E. Then the sum R, of P and Q, is defined as follows. First draw a line through P and Q; this line intersects the elliptic curve at a third point. Then R is the reflection of this point about the x-axis. This is represented in

Figure 29.

The *double R*, of *P*, is defined as follows. First draw the tangent line to the elliptic curve at *P*. This line intersects the elliptic curve at a second point. Then *R* is the reflection of this point about the *x*-axis. Algebraic formulas for the group law can be derived from the geometric description. These formulas are presented next for elliptic curves *E* of the simplified Weierstrass form in affine coordinates when the characteristic of the underlying field F_p is not 2 or 3.

The group law for an elliptic curve with p ≠2,3:

- *Identity*. *P* + ∞ = ∞ + *P* = *P* for all *P* ∈ *E(F_p)*.
- *Negatives*. If *P = (x, y)* ∈ *E(F_p)*, then *(x, y)* + *(x,−y)* = ∞. The point *(x,−y)* is denoted by −*P* and is called the *negative* of *P*; note that −*P* is indeed a point in *E(F_p)*. Also, − ∞ = ∞.
- *Point addition*. Let *P = (x1, y1)* ∈ *E(F_p)* and *Q = (x2, y2)* ∈ *E*(F_p), where *P* = ±*Q*. Then *P* + *Q* = *(x3, y3)*, where:

$$x_3 = (y_2 - y_1/x_2 - x_1)^2 - x_1 - x_2 \text{ and } y_3 = (y_2 - y_1/x_2 - x_1)(x_1 - x_3) - y_1$$

- *Point doubling*. Let $P = (x_1, y_1) \in E(F_p)$, where $P = -P$. Then $2P = (x_3, y_3)$, where:

$$x_3 = (3x^2_1 + a / 2y_1)^2 - 2x_1 \text{ and } y_3 = (3x^2_1 + a / 2y_1)^2(x_1 - x_3) - y_1$$

An important theorem about the number of points of an elliptic curve is the Hasse's theorem: let *E* be an elliptic curve defined over F_q . The number of points in $E(F_q)$, denoted $\#E(F_q)$, is called the *order* of *E* over F_q. Since the Weierstrass equation has at most two solutions for each $x \in F_q$, we know that $\#E(F_q) \in [1, 2q+1]$. This theorem provides tighter bounds for $\#E(F_q)$ that is:

$$q + 1 - 2\sqrt{q} \leq \#E(F_q) \leq q + 1 + 2\sqrt{q}$$

The interval $[q + 1 - 2\sqrt{q},\ q + 1 + 2\sqrt{q}]$ is called the *Hasse interval.* Since $2\sqrt{q}$ is small relative to *q*, we have $\#E(Fq) \approx q$.

Like any other type of curve also elliptic curves can make profitable use of changes of coordinates. Previously rules of addition and doubling in natural coordinates of elliptic curves were presented: these are called *affine coordinates*.

Let K be any field extended, binary or prime; it is possible define an equivalence relationship between non-zero triple on K, that is passes by the couple(x, y) to the triple (X, Y, Z). In particular, the map is the following:

$$x = X / Z^c \ , \ y = Y / Z^d \quad \text{for some values of } c \text{ and } d$$

In this new reference can be shown that the point at infinity becomes the point for which Z = 0 is: to be precise it is not a point but a line in the new coordinates so generally it is called *line at infinity*. This new type of coordinates is called *projective coordinates*. It is evident that according to the choice of parameters can have different coordinate systems. A particularly good choice is c = 2 and d = 3, in which case we speak of *Jacobian coordinates*. The usual characteristic curve defined for fields with> 3 becomes:

$$Y^2 = X^3 + aXZ^4 + bZ^6$$

in this system the *point to infinity* becomes (1 : 1 : 0) and the negative of (X : Y : Z) is (X : -Y : Z). In this new reference the couple (x, y), matches more than one of a point (X : Y : Z) because there is an added degree of freedom Z. For this reason the writing (X : Y : Z) indicates the equivalence class that corresponds to (x, y) and not the point (X : Y : Z) representative of that class. Following it is showed the new mode in which adding and doubling points:

If $P = (X, Y, Z) \in E(F_p)$ so $(X, Y, Z) + (X, -Y, Z) = \infty$. The point $(X, -Y, Z)$ is indicated by $-P = (X, -Y, Z)$ and is called the *negative of P*.

- *Point addition.* Let $P = (X_1, Y_1, Z_1)$ and $Q = (X_2, Y_2, Z_2) \in E(F_p)$, and let $P \neq \pm Q$. So $P + Q = (X_3, Y_3, Z_3)$, where:

$$\begin{cases} X_3 = (Y_2 Z_1^3 - Y_1)^2 - (X_2 Z_1^2 - X_1)^2 (X_1 + X_2 Z_1^2) \\ Y_3 = (Y_2 Z_1^3 - Y_1)(X_1 (X_2 Z_1^2 - X_1)^2 - X_3) - Y_1 (X_2 Z_1^2 - X_1)^3 \\ Z_3 = (X_2 Z_1^2 - X_1) Z_1 \end{cases}$$

- *Point doubling.* Let $P = (X_1, Y_1, Z_1) \in E(F_p)$ *and let* $P \neq -P$. So $2P = (X_3, Y_3, Z_3)$, where:

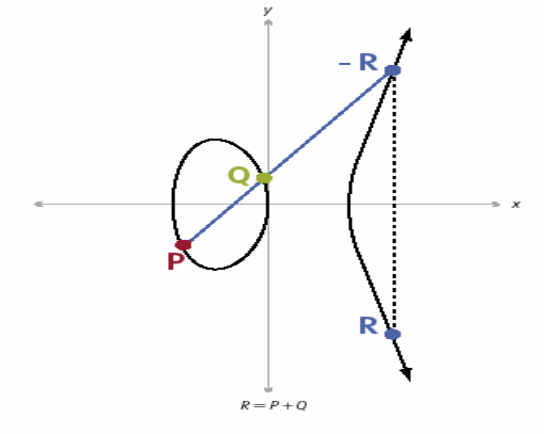

Figure 29 - Example of an elliptic curve, scheme of point sum and point inversion.

$$\begin{cases} X_3 = (3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2 \\ Y_3 = (3X_1^2 + aZ_1^4)(4X_1Y_1^2 - X_3) - 8Y_1^4 \\ Z_3 = 2Y_1Z_1 \end{cases}$$

It is important to note that when the final result is found it must return to affine coordinates because in Jacobian coordinates the result will be ambiguous.

From the computational point of view a superficial analysis would lead one to conclude that the Jacobian coordinates are inappropriate because the number of involved operations is higher. However in the Jacobian coordinates is not performed the operation of "division" that in a finite field is the operation of inversion that is known to be particularly expensive: It is made by the extended Euclidean algorithm, and if possible this type of operation should be avoided. Paying some extra multiplication (fast operation) avoids costly inversions: in the literature assumes that an inversion costs 80 multiplications. There are algorithms particularly efficient where it is known the structure of the elliptic curve and Jacobian coordinates are used: in general more are the parameters of the system that we fix and more we are able to build efficient algorithms for the price of a lower flexibility. There are other types of coordinates and the search is open on this front to get better and better representations that enable performance gains.

Table 12 - Operations needed to perform point addiction and point doubling on an elliptic curve with equation $y^2 = x^3 - 3x + b$, over different representations

Point doubling		***Point multiplication***		***Mixed coordinates***	
2A → A	1I, 2M, 2S	A + A → A	1I, 2M, 1S	J + A → J	8M, 3S
2P → P	7M, 3S	P + P → P	12M, 2S	J + C → J	11M, 3S
2J → J	4M, 4S	J + J → J	12M, 4S	C + A → C	8M, 3S
2C → C	5M, 4S	C + C → C	11M, 3S		

Point representation: A = affine, P = standard projective, J = Jacobian, C = Chudnovsky.

Operation over finite field: I = inversion, M = multiplication, S =squaring.

For example, there are special numeric representations to further improve performance (NAF Not Adjacent Form), special techniques of sliding window, the well-known product of Montgomery, interleaving, the use of the Frobenius map, the decomposition of the operands in multiplication, the

halving and others. As a synthesis of these considerations, Table 12 [hankerson] summarizes the computational costs of the most used coordinate systems for example for the case a = -3 and field prime.

To place elliptic curves in an application context should understand that there are *domain parameters* of a cryptosystem and how to represent a message through a point on the curve.

The basic operation to perform cryptography is the scalar multiplication of a point that is the operation of adding a point k times with itself.

A classical and efficient algorithm that is used is the algorithm of multiplication that a bit at a time of k, determines the operation to do. The algorithm consists in the following steps [hankerson].

INPUT: k = $(k_{t-1}, \ldots, k_1, k_0)$, $P \in E(F_q)$.

OUTPUT: *kP*.

1. $Q \leftarrow \infty$
2. For *i* from *0* to *t – 1* do
   2.1 If $k_1 = 1$ then $Q \leftarrow Q + P$
   2.2 $P \leftarrow 2P$
3. Return(*Q*)

To define a cryptosystem based on elliptic curves is appropriate to make a variety of parameters suitable to the definition of the domain in which we operate. The domain parameters are the following n-tuple of elements:

$$T = (q, fr, a, b, P, n, h)$$

- *q* represents the power of the prime number that describes the field in which it operates, which means that we are operating in F_q
- *fr* indicates the way in which we are representing the elements of F_q. In the case the field is not an extension, or is simply prime, the representation coincides with the elements of the field: in case of extended fields must have recourse to a polynomial representation.
- *a, b* are the coefficient of the curve
- *P* is the base point that is the point from which we start to perform the operation *kP*
- *n* indicates the order of *P.* Using an additive language, order means the value of *n* such that ∞ = *nP*
- *h* is called *cofactor* and is defined $\#E(F_q)/n$

A very simple way to represent a message was created by Koblitz, one of the two fathers of the elliptic curve cryptography this method is called *encapsulation* or *embedding*.

Suppose to operate in F_p and that $p \equiv 3 \bmod 4$ and $0 \le m < p/100 - 1$. Defining $x_i = 100m + i$ with $i \in \{0,1,..99\}$ it should be noted that $\forall x_i \in F_p$.

Defining $s_i = x_i^3 + ax_i + b$ with *a, b* coefficient of the curve: it is calculated the Legendre symbol $\left(\frac{s_i}{p}\right) = s_i^{(p-1)/2} \bmod p$ and it is verified that it is a quadratic residue (that is 1): in this way we can see if the root s_i could be extracted. If the answer is affirmative (ie is 1) for x_i we have found an univocal correspondence with *m*.

One can better understand the hypothesis $p \equiv 3 \bmod 4$: in fact is known that if that congruence is valid the root can be found easily by calculating $y_i \equiv s_i^{(p+1)/4} \bmod p$. If is not valid $p \equiv 3 \bmod 4$ the root can be calculated, however, it must use a more sophisticated called Tonelli's algorithm. In case of contrary response we will try for another x_i so long as at least one allows the solution of the equation. It is known from considerations of number theory that the fields are rich in quadratic residues, and should not be a preoccupation the idea of finding a residue because you can always easily find it.

To have any guarantee of success that we have imposed $0 \le m < p/100 - 1$ that is we have the opportunity to make 100 attempts. It is known that the probability of finding a quadratic residue is 0.5, for this reason the probability of not finding 1 in 100 attempts is $\left(\frac{1}{2}\right)^{100}$ that is 0.

A message encoded in a point will also need to go back, the decoding operation is trivial and gives $m = \lfloor x_i / 100 \rfloor$.

6.5.3 Protocols

A number of protocols can be defined over elliptic curves. This document will refer to the most widespread: Elliptic Curve Digital Signature Algorithm (ECDSA) [ansi962][fips186][ieee][rfc], Elliptic Curve Diffie-Hellman (ECDH) [ansi963][ieee][rfc], Elliptic Curve Menesez-Qu-Vanstone (ECMQV) [ansi963][ieee], Elliptic Curve Integrated Encryption Standard (ECIES).

ECDH is an analogue of the common Diffie-Hellman key exchange, with the same purpose. The pseudocode that describes ECDH algorithm is the following:

1. Alice and Bob share the domain parameters T=(q, a, b, P, n, h)
2. Alice generates $x \in [1, n-1]$ randomly and calculates $X = xP$
3. Bob generates $y \in [1, n-1]$ randomly and calculates $Y = yP$
4. Alice e Bob exchange Y, X (they are called ephemeral keys)
5. Alice calculates $k_a = xY = xyP$
6. Bob calculates $k_b = yX = xyP$
7. Alice e Bob share $k_b = k_a$: in particular the shared secret is the x – coordinate of the point just obtained.

The ECDSA algorithm is a digital sign algorithm analogue to the common DSA algorithm: this signature scheme is a standard that is recognized from NIST and IEEE. The pseudocode that describes ECDSA algorithm is the following:

INPUT: domain parameters, message to sign *m*

OUTPUT: (r, s)

SIGN PROCEDURE:

- $k \in [1, n-1]$ randomly
- $kP = (x_1, y_1)$ $r = x_1 \bmod n$
- If $r = 0$ back to the beginning
- $e = H(m)$ has been used an hash function
- $s = k^{-1}(e + dr) \bmod n$; if $s = 0$ back to the beginning
- Return *(s,r)*

VERIFICATION PROCEDURE:

- $r, s \in [1, n-1]$ otherwise reject
- $e = H(m)$
- $w = s^{-1} \bmod n$
- $u_1 = ew \bmod n$ and $u_2 = rw \bmod n$
- $X = (x_1, y_1) = u_1 P + u_2 Q$; If $X = 0$ reject
- $v = x_1 \bmod n$
- If $v = r$ accept, otherwise reject.

DEMONSTRATION:

- $s = k^{-1}(e + dr) \bmod n$
- $k = s^{-1}(e + dr) \bmod n$
- $k = s^{-1}e + s^{-1}dr \bmod n$
- $k = we + wdr \bmod n$
- $k = u_1 + u_2 d \bmod n$
- $X = (x_1, y_1) = u_1 P + u_2 Q$
- $u_1 P + u_2 Q = (u_1 + u_2 d)P = kP \Rightarrow v = r$

This protocol is secure if ECDLP is untreatable and the hash function is strong; the parameter *k* is called *per message secret* and must be generated and destroyed immediately after: if k was known it would be easy to derive the private key.

6.5.3.1 ElGamal - Message Encryption

ElGamal is an encryption protocol based on DLP. Thus ECDH problem is equivalent to solving the problem of decrypting simple encryption scheme ciphertexts. Since ECDH is equivalent to EC El Gamal, we immediately can infer the equivalence of the problem of decrypting EC EL Gamal ciphertexts and decrypting simple encryption scheme ciphertexts. If Alice wants to send Bob a message M encrypted with ElGamal, Bob has to set up a public key as follows. He picks an elliptic curve E defined over a finite field $\mathbb{F}_q$ such that the DLP is hard for the group E($\mathbb{F}_q$). He chooses a point P on E such that the order of P is (divisible by) a large prime. He picks a random integer s and computes $B = sP$. Bob's public key then is $(\mathbb{F}_q, E, P, B)$, his private key is the integer s.

Table 13 - The ElGamal public key cryptosystem.

Alice	Eve	Bob
Alice picks a random integer $\boldsymbol{k}$ and computes $\boldsymbol{M_1 = kP}$ as well as $\boldsymbol{M_2 = M + kB}$, and sends $\boldsymbol{M_1, M_2}$ to Bob	$\xrightarrow{M_1, M_2}$	Bob decrypts the message by computing $M = \boldsymbol{M_2} - s\boldsymbol{M_1}$

This works because $M_2 - sM_1 = (M + kB) - s(kP) = M + k(sP) - skP = M$. If Alice uses the same "random" integer k for two messages M and M', and if Eve knows the plaintext M, then she can compute M'.

6.5.3.2 Massey-Omura (Shamir's no-key protocol) - Message Encryption

The following cryptosystem was described in an unpublished mansucript of Shamir; it is also called Massey-Omura, and was first discovered by M. Williamson but not published since he was working at the GCHQ (a British Intelligence service) at the time. Table 14 explains the protocol, which is clearly based on the difficulty of the discrete log problem. The advantage of Shamir's no-key protocol is the fact that no keys are involved; the main disadvantage is that for sending one message, Alice has to send two ciphertexts, and wait for Bob's ciphertext to arrive.

Here Alice and Bob agree on an elliptic curve E defined over a finite field $\mathbb{F}_q$, where E is chosen in such a way that the DLP in $E(\mathbb{F}_q)$ is hard. They compute the group order $N = \#E(\mathbb{F}_q)$ and use a method for embedding messages as points M on the elliptic curve E. Then they basically follow the classical protocol of Shamir:

Table 14 - Massey-Omura

Alice	Eve	Bob
Alice and Bob agree upon (E, q)		
Alice picks $\boldsymbol{m_A, m_A^{-1}}$ with $\boldsymbol{m_A m_A^{-1} \equiv 1 \; mod \; N}$		Bob picks a pair m_B, m_B^{-1} with $m_B m_B^{-1} \equiv \boldsymbol{1 \; mod \; N}$
Alice computes $\boldsymbol{M_1 = m_A M}$ and sends $\boldsymbol{m_1}$ to Bob	$\boldsymbol{M_1} \rightarrow$	
	$\boldsymbol{M_2} \leftarrow$	Bob computes $M_2 = m_B M$ and sends M_2 to Alice
Alice computes $\boldsymbol{M_3 = m_A^{-1} M_2}$ and sends $\boldsymbol{M_3}$ to Bob	$\boldsymbol{M_3} \rightarrow$	
		Bob decrypts the message by computing $M = m_B^{-1} \boldsymbol{M_3}$

In fact, $m_B^{-1} M_3 = m_B^{-1} m_A^{-1} m_B m_A M = M$ since $m_A m_A^{-1} M = (kN + 1)M = M$ etc.

6.5.3.3 KMOV - Message Encryption

This protocol, which was suggested by Koyama, Maurer, Okamoto & Vanstone, works as follows. Alice picks primes $p, q \equiv 2 \; mod \; 3$ and forms the RSA-key $n = pq$. It is easy to see that $\#E(\mathbb{F}_p) = p + 1$ for all primes $p \equiv 2 \; mod \; 3$ and elliptic curves $E: Y^2 = X^3 + b$ over $\mathbb{F}_p$. Her public encryption key is a random integer e chosen coprime to $N = lcm(p + 1, q + 1)$, and her private decryption key d is computed via $de \equiv 1 \; mod \; N$. Although $E(Z/nZ)$ is not a group, it is still true that $NP = O$ for (almost) all points on $E(Z/nZ)$.

For sending a message $M = (m_1, m_2) \in E(Z/nZ)$ to Alice, Bob computes $b \equiv {m_2}^2 - {m_1}^3 \; mod \; n$ and computes $C = eM$ on the elliptic curve $: y^2 = x^3 + b$; then he sends the pair (b, C) to Alice, who decrypts it by computing $M = dC = deM$ on $E: y^2 = x^3 + b$ and recovers the original message by dropping the last 10 bits.

6.5.3.4 Demytko - Message Encryption

Let $E: y^2 = x^3 + ax + b$ be an elliptic curve over $\mathbb{F}_p$, and let d denote a quadratic nonresidue $mod \; p$. Then the cubic $E_d: dy^2 = x^3 + ax + b$ can be brought into Weierstrass form by multiplying through with d^3 ands setting $d^2 y = Y$, $dx = X$: this gives $Y^2 = X^3 + d^2 aX + d^3 b$. The elliptic curve E_d is called a quadratic twist of E.

It is an easy exercise to show that if $\#E(\mathbb{F}_p) = p + 1 - a_p$, then $\#E_d(\mathbb{F}_p) = p + 1 + a_p$. Moreover, if x is not the x-coordinate of a point on E, then it is the x-coordinate of a point on E_d. For a point $P = (x, y)$, let $k * x$ denote the x-coordinate of the point kP.

Now Alice chooses an RSA-modulus $n = pq$ and an elliptic curve $E: y^2 = x^3 + ax + b$ defined over $\mathbb{Z}/n\mathbb{Z}$. She also finds quadratic nonresidues $u \bmod p$ and $v \bmod q$, and defines the quadratic twists $E^{+-} = E_u(\mathbb{F}_p) \oplus E(\mathbb{F}_q), E^{-+} = E(\mathbb{F}_p) \oplus E_v(\mathbb{F}_q)$, and $E^{--} = E_u(\mathbb{F}_p \oplus E_v(\mathbb{F}_q)$. Let e be an integer coprime to the groups orders of these four curves (i.e., coprime to $p + 1 \pm a_p$ and $q + 1 \pm a_q$). In order to encrypt a message m, she first computes $(\frac{f(m)}{p})$ and $(\frac{f(m)}{q})$; if both symbols are positive, then m is the x-coordinate of some point $M \in E(\mathbb{Z}/n\mathbb{Z})$. $If\ (\frac{f(m)}{p}) = +1$ and $(\frac{f(m)}{q}) = -1$, then m is the x-coordinate of some point $M \in E^{+-}$ etc. Bob now computes $C = e * M$ and sends C to Alice. For decrypting the message, Alice sets

$$N_1 = lcm(p + 1 - a_p, p + 1 - a_q) \qquad if\ \left(\frac{w}{p}\right) = +1, \left(\frac{w}{p}\right) = +1,$$

$$N_2 = lcm(p + 1 - a_p, p + 1 + a_q) \qquad if\ \left(\frac{w}{p}\right) = +1, \left(\frac{w}{p}\right) = -1,$$

$$N_3 = lcm(p + 1 + a_p, p + 1 - a_q) \qquad if\ \left(\frac{w}{p}\right) = -1, \left(\frac{w}{p}\right) = +1,$$

$$N_4 = lcm(p + 1 + a_p, p + 1 + a_q) \qquad if\ \left(\frac{w}{p}\right) = -1, \left(\frac{w}{p}\right) = -1,$$

where $w \equiv c^3 + ac + b \bmod n$. Then she computes $d_i \equiv e^{-1} mod\ N_i$ and decrypts the message via $d_i * c = d_i e * m = m$.

6.5.3.5 Proxy blind signature scheme

The **blind signature scheme** is a protocol for obtaining a signature from a signer, but the signer can neither learn the messages nor see the signatures the recipients obtain afterwards. In **proxy signature scheme**, the original signer delegates his signing capacity to a proxy signer who can sign a message submitted on behalf of the original signer. A verifier can validate its correctness and can distinguish between a normal signature and a proxy signature. In **multi-proxy signature scheme**, an original signer is allowed to authorize a group of proxy members to generate the multi signature on behalf of the original signer.

A **proxy blind signature scheme** is a digital signature scheme that ensures the properties of proxy signature and blind signature. In a proxy blind signature, an original signer delegates his signing capacity to proxy signer. A **proxy blind signature scheme** is a special form of blind signature which allows a designated person called proxy signer to sign on behalf of two or more original signers without knowing the content of the message or document. It combines the advantages of proxy signature, blind signature and multi-signature.

A proxy blind signature scheme consists of the following three phases:

- Proxy key generation
- Proxy blind multi-signature scheme
- Signature verification

Most of the proxy blind signature schemes were developed based on the mathematical hard problems integer factorization (IFP) and simple discrete logarithm (DLP) which take sub-exponential time to solve. Alghazzawi et al. [7] describe a simple **proxy blind signature scheme** based on Elliptic Curve Discrete Logarithm Problem (ECDLP), which is solved in fully-exponential time. The algorithms for solving the ECDLP become infeasible much more rapidly as the problem size increases more than those algorithms for the IFP and DLP. Thus, ECC offers security equivalent to RSA and DSA while using far smaller key sizes. The benefits of this higher-strength per-bit include higher speeds, lower power consumption, bandwidth savings, storage efficiencies, and smaller certificates. This can be implemented in low power

and small processor mobile devices such as smart card, PDA etc. which work in low power and small processor.

Proposed Protocol

The protocol involves three entities:

- Original signer S,
- Proxy signer s P and
- Verifier V.

It is described as follows.

Proxy Phase

- **Proxy generation**:
  The original signer *S* selects random integer *k* in the interval $[1, n-1]$. Computes $R = kP = (x_1, y_1)$ and $r = x_1 mod\ n$. Where x_1 is regarded as an integer between 0 and *q*–1, then computes $s = (d + k * r)\ mod\ n$ and computes $Q_p = sP$.

- **Proxy delivery**:
  The original signer *S* sends (s, r) to the proxy signer P_s and make Q_p public.

- **Proxy Verification**:
  After receiving the secret key pairs (s, r), the proxy signer P_s checks the validity of the secret key pairs (s, r) with the following equation.

$$Q_p = sP = Q + rR \quad (1)$$

Signing Phase

- The Proxy signer P_s chooses random integer $t \in [1, n-1]$ and computes $U = tP$ and sends it to the verifier *V.*
- After receiving the verifier chooses randomly $\alpha, \beta \in [1, n-1]$ and computes the following
  $\tilde{R} = U + \alpha.P - \beta.Q_p$ (2)
  $\tilde{e} = H(\tilde{R} \parallel M)$ (3)
  $e = (\tilde{e} + \beta) mod\ n$ (4)
  and verifier *V* sends *e* to the proxy signer P_s.
- After receiving *e*, P_s computes the following
  $\tilde{s} = (t - s.e) mod\ n$ (5)
  and sends it to *V*.
- Now *V* computes $s_p = (\tilde{s} + \alpha) mod\ n$ (6)
  The tuples $(M, s_p, \tilde{e})$ is the proxy blind signature.

Verification Phase

The verifier *V* computes the following equation.

$$\gamma = H((s_p.P + \tilde{e}.Q_p) \parallel M) \quad (7)$$

and verifies the validity of proxy blind signature $(M, s_p, \tilde{e})$ with the equality $\gamma = \tilde{e}$.

Proxy blind multi-signature scheme

Kar [6] describes an efficient **proxy blind multi-signature scheme**. It satisfies the security properties of both proxy and blind signature scheme. In the proposed scheme the security is based on the difficulty of

breaking the one-way hash function and the elliptic curve discrete logarithm problem (ECDLP). Signatures can only be generated during valid delegation period. A trusted third party called certificate authority is utilized to ensure that.

Security properties

The security properties for a secure blind multi-signature scheme are as follows:

- **Distinguishability**: The proxy blind multi-signature must be distinguishable from the ordinary signature.
- **Strong unforgeability**: Only the designated proxy signer can create the proxy blind signature for the original signer.
- **Non-repudiation**: The proxy signer cannot claim that the proxy signer is disputed or illegally signed by the original signer.
- **Verifiability**: The proxy blind multi-signature can be verified by everyone. After verification, the verifier can be convinced of the original signer's agreement on the signed message.
- **Strong undeniably**: Due to fact that the delegation information is signed by the original signer and the proxy signature are generated by the proxy signer's secret key. Both the signer cannot deny their behaviour.
- **Unlinkability:** When the signer is revealed, the proxy signer cannot identify the association between the message and the blind signature he generated.
- **Secret key dependencies**: Proxy key or delegation pair can be computed only by the original signer's secret key.
- **Prevention of misuse**: The proxy signer cannot use the proxy secret key for purposes other than generating valid proxy signatures. In case of misuse, the responsibility of the proxy signer should be determined explicitly.

6.5.4 Implementation of Elliptic Curve Cryptosystems

The setup of an elliptic curve cryptographic device requires some high-correlated steps.

1. A finite field (or Galois field) $\mathbb{F}_q$ containing candidate elements to build the elliptic curve points. At this point one has to choose between prime fields ($q = p$ or $q = p^m$ where p is a big prime natural number) and binary fields ($q = 2^m$). Another critical parameter to choose is the field order q.
2. A representation for the finite field elements, in order to map correctly messages into elliptic curve points. For prime fields $\mathbb{F}_p$ an element is a Montgomery residue [montgomery1985]; for binary fields $\mathbb{F}_{2^m}$ an element is a polynomial with binary coefficients.
3. Algorithms for implementing arithmetic operations over the previously chosen finite field. To make possible the evaluation of elliptic curve point operations, one has to define the modular sum, modular squaring, modular multiplication and modular inversion over the finite field elements. The finite field arithmetic is different from binary fields to prime fields. For example, the modular multiplication of binary field elements is attained by the extended Euclidean algorithm [cormen], and in prime fields is attained by the Montgomery multiplication algorithm [montgomery1985].
4. An elliptic curve over $\mathbb{F}_q$. Not all elliptic curves are appropriate, since in some specific conditions the security of the overall cryptographic system can be substantially reduced. These specific conditions can take place when the chosen curve is defined weak. The NIST specification [nist] and Certicom SECG group specification [secg-B] help the designer to choose a non-weak and implementation-efficient curve but impose constraints on previous three facts: the order of the finite field, the representation of field elements and the finite field arithmetic algorithms.
5. A suitable representation of the elliptic curve. A coordinate change, for example from affine coordinates to projective coordinates, cause modification of the elliptic curve point arithmetic algorithms, but can improve computational efficiency. A specific projective transformation that leads to so called jacobian coordinates, reduces to zero the number of finite field modular

inversions, increasing the number of modular sums and modular multiplications needed in elliptic curve point operations. This fact, for some implementation cases, can improve dramatically the overall evaluation efficiency.

6. Algorithms for the elliptic curve points arithmetic, dependent on curve representation. The efficiency can be improved in different ways depending on the overall system implementation and if the desired target platform is hardware or software.

6.5.4.1 Implementation Issues

When one has to implement a particular protocol or functionality concerning elliptic curve cryptography, he has to face with a number or choices and factor that can affect entire system architecture, and consequently the system performances (running time, power consumption, hardware resource needs). Implementation choices regard:

- desired security level of the baseline algorithm (this implies constraints on choosing the appropriate finite field where the curve will lie, and constraints about the specific elliptic curve that implemented protocol will exploit);
- desired security level for the implemented cryptographic protocol (e.g. ECDSA, ECDH);
- methods to maximize the efficiency of finite field arithmetic;
- methods to maximize the efficiency of elliptic curve points operations;
- the application platform (hardware or software);
- constraints on computation resources (processor speed, code size, power consumption);
- constraints on communication resources (bandwidth, response time).

All these choice can affect deeply the application design, and inherently the final device security level. They all are taken together for better results in terms of security and performance.

To ensure avoiding the choose of a weak curve, the National Institute of Standards and Technology [nist] and the Standards to Efficient Cryptography Group [secg-B] published a list of recommended elliptic curve equations. These recommendations are built following studies on specific curves, to ensure usage of non-weak and implementation-efficient equations.

Table 15 - Elliptic curve point representations recommended by NIST for binary fields.

Finite field	*Polynomial basis*	*Normal basis type*
$GF(2^{163})$	$p(t) = t^{163} + t^7 + t^6 + t^3 + 1$	$T = 4$
$GF(2^{233})$	$p(t) = t^{233} + t^{74} + 1$	$T = 2$
$GF(2^{283})$	$p(t) = t^{283} + t^{12} + t^7 + t^5 + 1$	$T = 6$
$GF(2^{409})$	$p(t) = t^{409} + t^{87} + 1$	$T = 4$
$GF(2^{571})$	$p(t) = t^{571} + t^{10} + t^5 + t^2 + 1$	$T = 10$

6.5.4.2 State of the Art

In the late 1990's elliptic curve cryptography systems begin to enter into commercial devices. Some standard organization and private companies, like National Institute of Standards and Technology (NIST) [fips800], Standards for efficient cryptography group [secg-A] and Certicom [certicom-B] started to publish security standards and security protocols based on elliptic curves.

In February of 2005, the National Security Agency of the United States announced a coordinated set of algorithms for U.S. government use called Suite B, including symmetric encryption, key exchange, digital signature and hash functions [nsa]. NSA stated that certified Suite B implementations will be used for the protection of Top Secret information. At the same time, some countries in Europe proposed the same Suite B requirements for information protection. Suite B cryptography recommends use of elliptic curve

Diffie-Hellman (ECDH) in many existing protocols such as the Internet Key Exchange (IKE, mainly used in IPsec [rfc]), transport layer security (TLS [dierks]), and Secure MIME (S/MIME[ramsdell]).

6.5.4.3 Recommended Curves: NIST

Two types of curves are recommended by NIST specification [nist]: the pseudorandom curves and the special curves. Pseudorandom ones are curves with coefficients generated exploiting a seeded cryptographic hash evaluation. Special curves are particular cases of coefficients, curve equation and underlying fields notable for high computational efficiency.

NIST pseudorandom curves over prime fields are called P-192, P-224, P-256, P-384, P-521. All these curves are in the form $E: y^2 = x^3 - 3x + b \bmod p$. The recommendations regards all the curve parameters $T = (p, r, s, c, b, G)$ including the prime modulus p, the order r, the SHA1 seed input s needed to generation of coefficients, the SHA1 output c, the coefficient b satifying $b^2 c = -27 \bmod p$, and the coordinates of the generator point G or (G_x, G_y).

NIST pseudorandom curves over binary fields are called B-163, B-233, B-283, B-409, B-571, all in the form. Every recommended curve is in the form $E: y^2 + xy = x^3 + x^2 + b$.

NIST special curves, also called Koblitz curves, over binary fields are denoted with K-163, K-233, K-283, K-409, K-571, and are in the form $E: y^2 + xy = x^3 + ax^2 + 1$.

All the previous curves over binary fields support two possible representations for point coordinates: a polynomial representation or a normal basis representation. Table 15 shows representation possibilities over recommended binary fields.

6.5.4.4 Recommended Curves: SECG

The Standards for Efficient Cryptography Group (SECG) [secg-B] proposed in year 2000 a set of curves, over prime fields and binary fields.

Over prime fields SECG recommends random curves called secp112r1, secp112r2, secp128r1, secp128r2, secp160r1, secp160r2, secp192r1, secp224r1, secp256r1, secp384r1, secp521r1. Definition of random curves is provided in previous section.

Over binary fields SECG recommends special curves denoted with secp160k1, secp192k1, secp224k1, secp256k1 (Table 16). Definition of random curve is provided in previous section. Note that SECG recommendations include more elliptic curves defined over prime fields than NIST recommendations, and some special curves over prime fields are proposed.

Some cryptographic standards deals with elliptic curves and the SECG curves fit a number of them. In Table 17 one can note some standards like ANSI X9.62 [ansi962], the draft ANSI x9.63 [ansi963], the draft FSML [fsml], IEEE P1363 [ieee], IPSEC [panjwani], the already described NIST [nist], and Wireless Application Forum's WTLS standard (WAP) [wap]. In particular, IPSEC refers to the draft document regarding ECC and submitted to the IPSEC Internet Engineering Task Force working group.

Table 16 - Elliptic curve point representations recommended by SECG for binary fields.

Finite field	*Polynomial basis*
$GF(2^{113})$	$p(t) = t^{113} + t^9 + 1$
$GF(2^{131})$	$p(t) = t^{131} + t^8 + t^3 + t^2 + 1$
$GF(2^{163})$	$p(t) = t^{163} + t^7 + t^6 + t^3 + 1$
$GF(2^{193})$	$p(t) = t^{193} + t^{15} + 1$

$GF(2^{233})$	$p(t) = t^{233} + t^{74} + 1$
$GF(2^{239})$	$p(t) = t^{239} + t^{36} + 1$
$GF(2^{239})$	$p(t) = t^{239} + t^{158} + 1$
$GF(2^{283})$	$p(t) = t^{283} + t^{12} + t^{7} + t^{5} + 1$
$GF(2^{409})$	$p(t) = t^{409} + t^{87} + 1$
$GF(2^{571})$	$p(t) = t^{571} + t^{10} + t^{5} + t^{2} + 1$

Over binary fields SECG recommends random curves called sect113r1, sect113r2, sect131r1, sect131r2, sect163r1, sect163r2, sect193r1, sect193r2, sect233r1, sect283r1, sect409r1, sect571r1. Also special curves over binary fields are mentioned and are called sect163k1, sect233k1, sect239k1, sect283k1, sect409k1, sect571k1. Table 18 shows how these curves are recommended or compliant with several ECC standards mentioned above.

Table 17 - SECG curves over prime fields and compliance with current standards.

Curve	*ANSI X9.62*	*ANSI X9.63*	*FSML*	*IEEE P1363*	*IPSEC*	*NIST*	*WAP*
secp112r1				C	C		R
secp112r2				C	C		C
secp128r1				C	C		C
secp128r2				C	C		C
secp160k1	C	R	C	C	C		C
secp160r1	C	C	C	C	C		R
secp160r2	C	R	C	C	C		C
secp192k1	C	R	C	C	C		C
secp192r1	R	R	C	C	C	R	C
secp224k1	C	R	C	C	C		C
secp224r1	C	R	C	C	C	R	C
secp256k1	C	R	C	C	C		C
secp256r1	R	R	C	C	C	R	C
secp384r1	C	R	C	C	C	R	C
secp521r1	C	R	C	C	C	R	C

C stands for compliant, R for recommended

All SECG curves are provided with coefficients, point generator information, and somewhat all data that NIST presents for an elliptic curve. The curve formulations are explained in detail in SECG publication [secg-A].

For prime fields the curves are in the form $x^2 = y^3 + ax + b$ mod p, including a parameter set $T = (p, a, b, G, n, h)$ where p is defining the finite field GF_p, $a, b \in GF_p$ are the curve coefficients, G is the base point of the curve, the prime n is the order of G, and h is the cofactor of the curve (number of curve points divided by the order of G).

For finite fields the curves are in the form $x^2 + xy = y^3 + ax^2 + b$, with a parameter set $T = (m, f(x), a, b, G, n, h)$ where m is defining the finite field GF_{2^m}, $f(x)$ is an irreducible polynomial of degree m specifying the representation of binary field elements, $a, b \in GF_{2^m}$ are the curve coefficients, G is the base point of the curve, the prime n is the order of G, and h is the cofactor of the curve.

Table 18 - SECG curves over binary fields and compliance with current standards.

Curve	*ANSI X9.62*	*ANSI X9.63*	*FSML*	*IEEE P1363*	*IPSEC*	*NIST*	*WAP*
sect113r1				C	C		R
sect113r2				C	C		C
sect131r1				C	C		C
sect131r1				C	C		C
sect163k1	C	R	R	C	R	R	R
sect163r1	C	C	R	C	R		C
sect163r2	C	R	R	C	C	R	C
sect193r1	C	R	C	C	C		C
sect193r2	C	R	C	C	C		C
sect233k1	C	R	C	C	C	R	C
sect233r1	C	R	C	C	C	R	C
sect239k1	C	C	C	C	C		C
sect283k1	C	R	R	C	R	R	C
sect283r1	C	R	R	C	R	R	C
sect409k1	C	R	C	C	C	R	C
sect409r1	C	R	C	C	C	R	C
sect571k1	C	R	C	C	C	R	C
sect571r1	C	R	C	C	C	R	C

C stands for compliant, R for recommended

6.5.5 Known Attacks against Elliptic Curve Cryptosystems

Here we present some known attacks against elliptic curve cryptosystems. The scope is limited to algorithms solving the elliptic curve discrete logarithm problem (ECDLP), i. e. to determine l given a point P and a point $Q = lP$, and it does not consider attacks against particular elements of digital signature algorithms based on elliptic curves.

1. **Naive Exhaustive Search**: The most simple approach to obtain l is to compute successive multiples of $P: P, 2P, 3P, 4P, \ldots$ until the result is equal to Q. In the worst case, this takes n steps, where n is the order of the point P.

2. **Baby-Step Giant-Step Algorithm**: This algorithm is a time-memory trade-off of the method of exhaustive search. It requires storage for about $\sqrt{n}$ points, and its running time is roughly $\sqrt{n}$ steps in the worst case.

3. **Pollard's Rho Algorithm**: This algorithm is a randomized version of the baby step giant-step algorithm. With some modifications it can be sped up to have an expected running time of $\frac{\sqrt{\pi n}}{2}$ steps and it requires only a negligible amount of storage.

4. **Parallelized Pollard's Rho Algorithm**: The original Pollard's rho algorithm can be parallelized so that when it is run in parallel on r processors, the expected running time is roughly $\frac{\sqrt{\pi n}}{2r}$.

5. **Pollard's Lambda Method**: Pollard also presented a lambda method for computing discrete logarithms which is applicable when l, the logarithm sought, is known to lie in a certain interval. In

particular, when l is known to lie in a subinterval $[0, b]\ of\ [0, n-1]$, where $b < 0{,}39n$, the parallelized version of Pollard's lambda method is faster than the parallelized Pollard's rho algorithm.

6. **Multiple Logarithms**: It turns out that if a single instance of the ECDLP is solved using (parallelized) Pollard's rho method, the following instances (for the same curve E and the same base point P) can be solved faster, since some of the necessary work has already been done in the previous steps. In fact, solving k instances of the ECDLP takes only $\sqrt{k}$ as much work as it does to solve one instance.

Hence, the best known attack against a single instance of the ECDLP is Pollard's rho algorithm and has an expected running time of $\frac{\sqrt{\pi n}}{2} = O(n^{1/2})$, which is fully exponential.

However, there are certain elliptic curves with special vulnerabilities that can be exploited by the following algorithms. These algorithms may have shorter running times as those mentioned before; therefore elliptic curves with these vulnerabilities should be avoided.

1. **Pohlig-Hellman Algorithm**: This algorithm exploits the factorization of n, the order of the point P, and reduces the problem of recovering l to the problem of recovering l modulo each of the prime factors of n. We can then recover l by using the Chinese Remainder Theorem. As a countermeasure, one should select an elliptic curve whose order is a prime or almost a prime (i. e. a large prime times a small integer).

2. **Supersingular Elliptic Curves**: In some cases, the ECDLP in an elliptic curve E defined over a finite field $\mathbb{F}_q$ can be reduced to the ordinary discrete logarithm problem (DLP) in the multiplicative group of some extension field $\mathbb{F}_{q^k}$ for $k \geq 1$. The DLP is the underlying computationally hard mathematical problem for the DSA and one can solve it using the number field sieve algorithm, which has a subexponential running time. To ensure that the reduction algorithm does not apply to a particular curve, one only needs to check that n does not divide $q^k - 1$ for all small k for which the DLP in $\mathbb{F}_{q^k}$ is tractable. In practice, it suffices to check this for $1 \leq k \leq 20$ when $n > 2^{160}$.

3. **Prime-Field Anomalous Curves**: If the number of points on a curve E over $\mathbb{F}_p$ is equal to p, the ECDLP can be solved efficiently. This attack can be avoided by verifying that the number of points on an elliptic curve is not equal to the cardinality of the underlying field.

4. **Curves Defined Over a Small Field**: For elliptic curves E with coefficients in $\mathbb{F}_{2^e}$, Pollard's rho algorithm for computing elliptic curve logarithms in $E(\mathbb{F}_{2^{ed}})$ can be further sped up by a factor of $\sqrt{d}$. For example, if E is a Koblitz curve, then Pollard's rho algorithm for computing elliptic curve logarithms in $E(\mathbb{F}_{2^m})$ can be sped up by a factor of $\sqrt{m}$.

5. **Curves Defined Over $\mathbb{F}_{2^m}$, m Composite**: The Weil descent might be used to solve the ECDLP for elliptic curves defined over $\mathbb{F}_{2^m}$ where m is composite. There exists some evidence that when m has a small divisor l, e.g. $l = 4$, the ECDLP can be solved faster than with Pollard's rho algorithm. Thus, elliptic curves over composite fields should not be used.

Let us take a look how secure elliptic curve cryptography is in practice. Certicom challenge is one source that gives a notion about the security of ECC. The challenge is to compute the ECC private keys from a given list of ECC public keys and associated system parameters. The challenge consists of two levels:

- Level I: 109-bit and 131-bit challenge; considered to be feasible
- Level II: 163-bit, 191-bit, 239-bit and 359-bit challenge; expected to be computationally Infeasible

Concrete recommendations based on the expected cost of the Certicom challenge have been presented. They predict which elliptic curve key sizes can be considered as secure until which year using a model

that incorporates technological and cryptanalytical advances. Table 19 presents their recommendations for future years.

Table 19 - Minimum key size for elliptic curve cryptosystems providing a sufficient level of security

Year	Elliptic Curve Key Size	Infeasible Number of Mips Years	Corresponding Number of Years on 450MHz Pentium II PC
2002	139	$2,06*10^{10}$	$4,59*10^{7}$
2003	140	$3,51*10^{10}$	$7,80*10^{7}$
2004	143	$5,98*10^{10}$	$1,33*10^{8}$
2005	147	$1,02*10^{11}$	$2,26*10^{8}$
2006	148	$1,73*10^{11}$	$3,84*10^{8}$
2007	152	$2,94*10^{11}$	$6,54*10^{8}$
2008	155	$5,01*10^{11}$	$1,11*10^{9}$
2009	157	$8,52*10^{11}$	$1,89*10^{9}$
2010	160	$1,45*10^{12}$	$3,22*10^{9}$
2011	163	$2,47*10^{12}$	$5,48*10^{9}$
2012	165	$4,19*10^{12}$	$9,32*10^{9}$
2013	168	$7,14*10^{12}$	$1,59*10^{10}$
2014	172	$1,21*10^{13}$	$2,70*10^{10}$
2015	173	$2,07*10^{13}$	$4,59*10^{10}$
2016	177	$3,51*10^{13}$	$7,81*10^{10}$
2017	180	$5,98*10^{13}$	$1,33*10^{11}$
2018	181	$1,02*10^{14}$	$2,26*10^{11}$
2019	185	$1,73*10^{14}$	$3,85*10^{11}$
2020	188	$2,94*10^{14}$	$6,54*10^{11}$
2021	190	$5,01*10^{14}$	$1,11*10^{12}$
2022	193	$8,52*10^{14}$	$1,89*10^{12}$

6.5.6 ECC Applications

The number of vendors who have incorporated ECC in their products is rising. An important factor for this emerging trend is the incorporation of ECDSA in several government and major research institution security standards, including IEEE P1363, ANSI X9.62, ISO 11770-3 and ANSI X9.63. ECC is becoming the mainstream cryptographic scheme in all mobile and wireless devices.

ECC applications seen on the market today can be broadly divided into four categories: the Internet, smart cards, PDAs and PCs.

6.5.6.1 Smart cards

The most popular devices for the use of ECC are smart cards. Many manufacturing companies, like Phillips, Fujitsu, MIPS Technologies and DataKey are producing smart cards that use ECDSA algorithms. Smart cards are being used in many situations, like bank (credit/debit) cards, electronic tickets and personal identification (or registration) cards since they are very flexible tools.

6.5.6.2 PDAs

PDAs have more computing power compared to most of the other mobile devices, like cell phones or pagers and because of that are very popular for implementing public key cryptosystems. On the other hand they still suffer from limited bandwidth and this makes them an ideal choice for using ECC.

Security, implementation and performance of ECC applications on various mobile devices have been examined and it can be concluded that ECC is the most suitable PKC scheme for use in a constrained environment.

6.5.7 ECC in Software Trusted Platform Module (TPM)

The TPM provides capabilities for secure storage; secure reporting of platform configuration measurements, and cryptographic key generation. In addition the TPM chip implements tamper-resistance techniques to prevent a wide range of physical and hardware-based attacks.

Trusted computing has applicability to a wide range of embedded systems. Recent efforts to adapt trusted computing standards to resource-constrained environments include the TCG's Mobile Phone Working Group and the Trusted Mobile Platform Alliance. The hardware enhancements, including the addition of the TPM chip, may impose an overhead in the context of cost and size in resource-constrained embedded systems and this is not acceptable. For such systems, one option is to use a software-based TPM (SW-TPM), which implements TPM functions using software that performs in a protected execution domain on the embedded processor itself in order to enable the adoption of trusted computing techniques. It is also important to ensure that the computational and energy requirements for SW-TPMs are acceptable since many embedded systems have limited processing capabilities and are battery-powered.

In terms of protection against physical and hardware attacks SW-TPM is not completely equivalent to a conventional TPM chip. SW-TPM can be executed within protected or isolated execution domains that are provided by embedded CPUs (*e.g.*, ARM TrustZone), and can utilize on-chip storage in order to provide a reasonable degree of tamper-resistance. The question that arises is whether the computational and energy requirements to perform the TPM functions are acceptable.

In the article of Aaraj et al. [1], [2] is performed an evaluation of the energy and execution time overheads for a SW-TPM implementation on a handheld appliance (Sharp Zaurus PDA). The execution time and energy required by each TPM command through actual measurements on the target platform is characterized. It is shown that for most commands, overheads are primarily due to the use of 2,048-bit RSA operations that are performed within the SW-TPM. They replace the RSA algorithm with the Elliptic Curve Cryptography (ECC) specified in the Trusted Computing Group (TCG) standards, in order to alleviate SW-TPM overheads. They also evaluate the overheads of using the SW-TPM in the context of various end applications, including trusted boot of the Linux operating system (OS), a secure VoIP client, and a secure Web browser.

Their experiments indicate that this optimization can significantly reduce SW-TPM overheads. This work demonstrates that ECC-based SW-TPMs are a viable approach to realizing the benefits of trusted computing in resource-constrained embedded systems.

This work contributes in the following way:

- A comprehensive characterization of SW-TPM running on a battery-powered handheld device (Sharp Zaurus PDA) is performed, and the execution time and energy requirements for various TPM commands are measured.
- The overheads imposed by using TPM functions in end applications are evaluated, including trusted boot of the Linux OS, secure file storage utility, secure VoIP client, and secure web browser.
- In order to alleviate the overheads imposed by SW-TPM, it is proposed and evaluated the use of ECC as a replacement for the RSA algorithm specified in the TCG standards. The experiments indicate that results in a substantial reduction in SW-TPM overheads.

This work demonstrates the feasibility of using SW-TPM to realize the benefits of trusted computing in resource-constrained embedded systems.

6.5.7.1 SW-TPM Implementation

The TPM security features are very useful in many embedded systems. Some embedded systems cannot be augmented with a conventional TPM chip because of the area and cost constraints. Here, the feasibility of a SW-TPM is explored, which performs the same functions as a hardware TPM, *i.e.*, supports all the three roots of trust, as well as other cryptographic capabilities. SW-TPM does not provide the same security level as a TPM chip. Executing the SW-TPM in a protected execution domain of the CPU (*e.g.*, ARM Trust-Zone), and using on-chip memory, provides resistance to software attacks, including compromises of the OS, and a limited number of physical attacks.

The implementation of SW-TPM is adapted from the public domain TPM emulator [3], which provides basic TPM functions, such as RSA cryptography and HMAC and SHA-1 hashing functions, and provides several TPM commands.

The emulator has been changed as follows:

- **Random number generation**: A hash-complemented Mersenne Twister (MT) random number generator [4] is used, *i.e.,* we run the output of MT through SHA-1.
- **ECC**: SW-TPM supports ECC in the binary field GF(2^m). ECC on this embedded platform is used because of its small key sizes compared to RSA for offering the same security robustness. Hence, it requires less resource such as processor cycles and energy. ECC-enabled SW-TPM supports key generation and validation, digital signature generation and verification, encryption, and decryption. Supported ECC key sizes are 224 bits (equivalent to 2048-bit RSA keys), 192 bits (not equivalent to RSA key), and 160 bits (equivalent to 1024-bit RSA keys).
- **AES_CBC cryptography**: SW-TPM supports the Advanced Encryption Standard (AES) algorithm, running in Cipher Block Chaining (CBC) mode. This engine is specifically used for ECC encryption and decryption, and for decrypting AIK credentials.

6.5.7.2 Measurement results

The execution time and energy consumed by SW-TPM on the PDA is presented here in order to execute various TPM commands. The presented results are for the original RSA-based SW-TPM and for the proposed ECC-based SW-TPM.

For commands categorized as the storage and key management commands, and TPM_Sign, measurements are performed for different key sizes. For commands that process user data, the data size is varied. The results of these experiments are reported in Table 20. The command executed is presented in Column 1. Columns 2-3 give the key size (K) and data size (D). For commands that do not involve cryptographic operations K (D) is indicated as N/A. Column 4 gives energy measurements in milliJoules (mJ), and column 5 reports the execution times for the TPM commands in milliseconds (msec.). The results indicate that commands involving RSA operations, particularly private key operations, which require manipulation of large numbers, and a resource-consuming modular exponentiation, impose a high execution time overhead. For instance, the TPM_MakeIdentity command, which involves 2048-bit RSA key generation and validation, as well as encryption of the private AIK using the SRK, in addition to other cryptographic functions, takes 29.63 sec. and consumes 70.94 J of energy. Similarly, large execution times and energy consumptions are required for TPM_TakeOwnership, TPM_CreateWrapKey, TPM_Unseal, etc. This overhead is reduced by using ECC: execution time and energy requirements for the TPM_MakeIdentity command are reduced to 2.43 sec. and 5.86 J, respectively. By using ECC, an average reduction of 6.51X and 6.75X can be achieved for execution time and energy, respectively, across all commands.

Table 20 - Energy and execution time for TPM commands

Command	K(bits) ECC/RSA	D(bytes)	PDA measurements	
			Energy (mJ)	Time (msec.)
Authentication commands				
TPM_OIAP	n/a	n/a	0.61	0.21
TPM_OSAP	n/a	n/a	2.38	0.82
Capability commands				
TPM_GetCapability **(Key info.)** **(Manufacturer info.)** **(PCR info.)**	 n/a n/a n/a	 n/a n/a n/a	 0.10 0.10 0.20	 0.04 0.04 0.07
Cryptographic commands				
TPM_GetRandom	n/a	20	0.55	0.19
TPM_Sign	224/2048 224/2048 224/2048 160/1024 160/1024 160/1024 192/512 192/512 192/512	20 50 100 20 50 100 20 50 100	450/2210 492/2221 531/2394 210/806 242/930 319/1006 321/626 350/656 361/760	191/902 204/926 216/960 90/343 114/388 131/409 136/265 148/274 153/305
Identity commands				
TPM_ActivateIdentity	224/2048	n/a	598/12824	348/5239
TPM_Makedentity	224/2048	n/a	5859/70943	2425/29634
Measurements commands				
TPM_PcrRead	n/a	n/a	17.32	6.69
TPM_PcrExtend	n/a	n/a	32.28	12.46
TPM_Quote	224/2048	n/a	762/2475	381/1239
Ownership commands				
TPM_ReadPubek	224/2048	n/a	0.31/3.10	0.12/1.22
TPM_TakeOwnership	224/2048	n/a	5619/66777	2391/28619
Start-up commands				
TPM_init_data	224/2048	n/a	1.71/25.39	0.69/10.52
TPM_Startup	224/2048	n/a	0.48/1.46	0.19/0.58
Storage and key management commands				
TPM_CreateWrapKey	224/2048 160/1024 192/512	n/a n/a n/a	5558/42582 4128/12133 4419/8395	2322/16938 1813/4594 1880/3025
TPM_EvictKey	224/2048 160/1024 192/512	n/a n/a n/a	8.36/37.08 6.70/16.62 6.72/7.73	3.31/14.78 2.71/6.62 2.79/3.10
TPM_GetPubKey	224/2048 160/1024 192/512	n/a n/a n/a	640/10592 471/1504 516/852	229/4388 157/567 172/453
TPM_LoadKey	224/2048 160/1024 192/512	n/a n/a n/a	810/14547 593/4557 737/2092	336/5367 261/1796 301/841
TPM_Seal	224/2048 224/2048 224/2048 160/1024 160/1024 160/1024 192/512 192/512 192/512	20 50 100 20 50 100 20 50 100	1103/3751 1125/3785 1313/4271 769/1898 806/2195 819/2965 1001/1019 1026/1063 1093/1326	463/1476 472/1564 530/1761 322/796 334/967 342/1178 420/427 431/481 444/551
TPM_Unseal	224/2048 160/1024 192/512	256 256 256	1444/14056 952/4679 1279/1778	585/5520 391/1880 525/714

TPM_Unbind	224/2048 160/1024 192/512	256 256 256	1459/10480 974/4269 1284/1616	576/4103 384/1699 524/699

Macromodels that capture the energy for the commands TPM_Sign and TPM_Seal as a function of the key size *K* and data size *D* are presented in Table 20. Values of *K* are up to 2048 (224) bits for RSA (ECC), and *D* assumes values up to 144 Bytes. From the macromodels, and the numbers reported in Table 21 can be concluded that energy and execution time requirements vary more considerably with the key size rather than with the data size (especially with RSA cryptography).

Table 21 - Energy macromodels for the TPM_Sign and TPM_Seal.

Command	Crypto type	Energy model ($C + A*D + B*D^2 + X*K + Y*K^2$ (mJ))
TPM_Sign	ECC	$31.269 + 1.434*D - 0.004*D^2 + 0.642*K + 0.011*K^2$
	RSA	$29.994 + 10.389*D - 0.061*D^2 + 0.349*K + 0.00029*K^2$
TPM_Seal	ECC	$6.415 + 0.067*D - 0.012*D^2 + 3.980*K + 0.005*K^2$
	RSA	$5.766 + 10.033*D - 0.142*D^2 + 2.435*K + 0.00026*K^2$

The presented results are based on the average of several executions (16) of each command, in order to account for uncontrollable variables, such as the randomness of the keys, and to minimize measurement error for commands that require small running times.

It is also important to place the overheads in the context of actual applications not only for evaluating the requirements of SW-TPM in isolation. Trusted extensions for several applications are proposed and the impact of using SW-TPM on their execution time and energy consumption is studied.

6.5.7.3 User applications with SW-TPM

In the paper [1] is presented an experiment where SW-TPM was used in the context of four different applications. The trusted extensions of four applications (Trusted Boot, Secure Storage, Secure Voice over Internet Protocol (VoIP), and Secure Web Browsing) are described and the effect of these extensions in terms of energy and execution time is evaluated. The results of these experiments are presented in Table 22.

In the first column is shown which cryptographic algorithm is used: RSA-enabled SW-TPM or ECC-enabled SW-TPM. Columns 2-3 give energy and execution time for the untrusted application, and the total overhead required by the trusted version of this application, respectively. In the columns 4-5 are given the energy and execution time overheads due to the executed SW-TPM commands. The results indicate that the executed SW-TPM commands, within the applications, require an average of 10.75X less energy and an average of 10.25X less execution time when using ECC, instead of RSA.

Table 22 - Energy and execution time for trusted applications

Cryptographic algorithm	Untrusted application/Overall trust overhead		SW-TPM commands overhead	
	Energy (J)	**Time (sec.)**	**Energy (J)**	**Time (sec.)**
Trusted Boot				
RSA ECC	95.542/198.759 95.542/137.699	48.281/89.495 48.281/63.280	66.806 5.746	28.657 2.442

Secure Storage				
RSA ECC	0.057/75.251 0.057/12.026	0.023/29.732 0.023/4.932	75.059 11.823	29.661 4.859
VoIP: Voice data encrypted				
RSA ECC	159.243/97.353 159.243/9.446	60/40.752 60/4.213	88.778 8.034	37.376 3.579
RSA ECC	318.081/98.228 318.081/10.324	120/41.240 120/4.602	88.778 8.034	37.376 3.579
RSA ECC	804.246/100.867 804.246/12.961	300/42.356 300/6.524	88.778 8.034	37.376 3.579
RSA ECC	1614.909/105.343 1614.909/17.366	600/44.001 600/7.712	88.778 8.034	37.376 3.579
VoIP: Voice data encrypted and hashed				
RSA ECC	159.243/98.893 159.243/9.795	60/40.971 60/4.522	88.778 8.034	37.376 3.579
RSA ECC	318.081/100.121 318.081/11.034	120/41.508 120/4.900	88.778 8.034	37.376 3.579
RSA ECC	804.246/103.612 804.246/14.708	300/43.121 300/7.237	88.778 8.034	37.376 3.579
RSA ECC	1614.909/108.879 1614.909/20.861	600/45.630 600/9.191	88.778 8.034	37.376 3.579
SSL: Server running on PDA				
RSA ECC	0.530/92.455 0.530/7.652	0.213/38.707 0.213/3.387	86.271 7.250	36.124 3.186
SSL: Client running on PDA				
RSA ECC	0.707/2.449 0.707/0.259	0.284/1.175 0.284/0.124	n/a n/a	n/a n/a

6.5.8 Electromagnetic analysis ECC on a PDA

Resistance to attacks on the PDA or cell phone will become a necessity, since a lot of security applications migrate to the wireless device. Such attacks can happen when the device has been stolen or lost and also during everyday use when unintentional electromagnetic (EM) waves radiated from the wireless device during cryptographic computations may reveal confidential data to an attacker. For example an attacker may successfully attack confidential memory in a wireless device obtaining the secret keys stored in there. This attack may be possible through loss or theft of the device. Alternatively the attack can happen also through temporary access to the device by monitoring the EM waves deriving from the device while performing cryptographic computations. In that case the attacker may be able to extract the encryption keys and making future wireless communications insecure. For wireless embedded systems large overheads in energy to achieve resistance to attacks are not practical. Beside smartcard research few researchers have examined secure implementations of cryptographic software (Rijndael [17]) under the threat of EM attacks on 32-bit processors. The cryptographic algorithms which are essential for these applications are typically run by embedded processors in these wireless devices but it is known that they consume a lot of energy. These attack resistant algorithms have been developed for smartcard applications, where energy dissipation is not such important. There is an important need to study EM attacks and energy optimized countermeasures on wireless portable devices, such as PDAs, cell phones, etc.

Thus, although many wireless portable devices offer more resistance to bus probing and power analysis attacks due to their compact size, susceptibility to electromagnetic (EM) attacks must be analysed. Paper from Gebotys et al. [8] presents a real EM-based attack of a PDA running elliptic curve cryptography and a new frequency-based differential EM analysis, which computes the power spectral density and spectrogram. Unlike previous research the new differential analysis does not require perfect alignment of EM traces, thus supporting attacks on real embedded systems.

6.5.8.1 Differential analysis in the frequency domain

Gebotys et al. [8] proposed a performing analysis in the frequency domain, which is an extension of the existing differential side channel attack, where analysis is performed in the time domain. Here are presented two approaches:

- differential frequency analysis (DFA), where the power spectral density (PSD) is used for analysis,
- differential spectrogram analysis (DSA), since a spectrogram is created.

The problem of misalignment (or time-shifts) in traces is solved by analysing signals captured in the frequency domain since fast Fourier transform (FFT) analysis is time-shift invariant.

Both the DFA and DSA are important for attacking real embedded systems where uncorrelated temporal misalignment (or time shifting) of traces is a big concern. Some structures cannot be revealed with time domain analysis. Contrary to that frequency analysis may reveal loops and other repeating structures in an algorithm. There are two problems with using frequency domain signals in differential analysis:

- It reveals no information of when data-dependant operations occur. This timing information is very useful as it helps an adversary focus the signal analysis on these data dependant operations.
- Any peaks in frequency domain due to an event that occurs in a short duration can be visible if the acquisition duration is a lot longer. The solution of these problems is to use spectrogram, which is a time dependant frequency analysis.

The attack of an elliptic curve algorithm presented here uses EM traces. Only when the 1st and 2nd most significant bits of the elliptic point data are used for partitioning, both the DEMA (differential EM analysis) and DSA (differential spectrogram analysis) attacks are successful on the elliptic curve algorithm. This attack works if the MSB's (most significant bit) are correlated with EM activity from the overflow or underflow computations. Using the same algorithm for finite field computations regardless of whether an overflow occurs or not can be a possible countermeasure for the MSB differential analysis attack of ECC.

The analysis techniques proposed here successfully obtain the correct key from elliptic curve cryptography. They are general and applicable to other cryptographic algorithms, power as well as EM, and other embedded systems. A new frequency-based (time-shift invariant) differential analysis is presented using real EM measurements from a PDA device executing Java-based cryptography. Previous differential analysis techniques, which required alignment of traces in the time domain, were not successful in correlating EM signals to bits of the data. This research supports low energy security for embedded systems which will be prevalent in wireless embedded devices of the future.

6.5.9 ECC in wireless sensors

A WSN is a wireless ad-hoc network consisting of resource-constrained sensoring devices (limited energy source, low communication bandwidth, small computational power) and one or more base stations. The base stations are more powerful and collect the data gathered by the sensor nodes so it can be analysed. Routing is accomplished by the nodes themselves as any ad hoc network, through hop-by-hop forwarding of data. Common WSN applications range from battlefield exploration and emergency rescue operations to surveillance and environmental protection.

Security and cryptography on WSNs meet several open problems even though several years of intense research. Given the limited computational power and the resource-constrained nature of the sensoring devices, the deployment of cryptography in sensor networks is a difficult task. Aranha et al. `s paper [12] presents the implementation of elliptic curve cryptography in the MICAz Mote, a sensor platform to develop optimizations specifically:

(i) the cost of memory addressing;
(ii) the cost of memory instructions;
(iii) the limited flexibility of bitwise shift instructions.

This work presents efficient implementations for arithmetic of binary field algorithms such as squaring, multiplication, modular reduction and inversion at two different security levels. These implementations take into account the characteristics of the target platform. The implementation of field multiplication and modular reduction algorithms focuses on the reduction of memory accesses and appears as the fastest result for this platform.

Finite field arithmetic was implemented in C and Assembly and elliptic curve arithmetic was implemented in Koblitz and generic binary curves. Here are obtained the fastest binary field arithmetic implementations in C and Assembly published for the target platform. Significant performance benefits where achieved by the Assembly implementation, resulting from fine-grained resource allocation and instruction selection. The performance of implementations is illustrated with timings for key agreement and digital signature protocols. Results strongly indicate that binary curves are the most efficient alternative for the implementation of elliptic curve cryptography in this platform.

Optimizations produced a point multiplication at the 160-bit security level under 1/3 of a second, an improvement of 72% compared to the best implementation of a Koblitz curve previously published and an improvement of 61% compared to the best implementation of binary curves. When compared to the best implementation of prime curves, is obtained a performance gain of 57%.

6.5.10 Improvements in ECC for resource-constrained devices

Elliptic curve cryptography (ECC) is very important in the field of low-resource devices such as smart cards and Radio Frequency Identification (RFID) devices because of the significant improvements in terms of speed and memory compared to traditional cryptographic primitives (e.g. RSA). Memory is one of the most expensive resources in the design of embedded systems which encourages the use of ECC on such platforms.

Scalar multiplication in ECC implementation is the operation used in many cryptographic primitives for solving the elliptic curve discrete logarithm problem (ECDLP), i.e. finding the discrete logarithm for Q with respect to the elliptic curve point P. It is a process where a secret scalar k is multiplied with a point P on an elliptic curve $E(F_q)$ getting in the point Q and is the most resource-consuming operation.

In embedded systems memory and computational power are scarce resources. In that case, the scalar multiplication can be improved with a method called Montgomery ladder. In this process the y-coordinate of the involved elliptic curve points can be omitted, which lowers the memory requirements for low-resource designs. In addition, it implicitly provides resistance against certain implementation attacks which encourages its use in security-related applications. Meloni [14] proposed another improvement where he showed that points on an elliptic curve can be added quickly when they share a common coordinate, e.g. the projective Z-coordinate. He applied the formula to specific Euclid addition chains to perform a scalar multiplication which improves the speed of ECC implementations and reduces the memory requirements by one coordinate. The proposal of Meloni was extended by Goundar [15] who provided a formula over prime fields that can be applied to classical binary scalar multiplication methods. He introduced a new operation (conjugate co-Z addition) that can be used together with the addition formula of Meloni to perform fast computations with points sharing the same Z-coordinate (co-Z arithmetic).

Performance of scalar multiplication in elliptic curve cryptography implementations can be improved by sharing a common coordinate. Hutter [11] presented a new formula for elliptic curves over prime fields that provide efficient (speed-wise and memory-wise) point addition and doubling using the Montgomery ladder especially applicable to resource-constrained devices. The proposed formula uses out-of-place operations to insure that no additional memory for any implementation of the underlying finite-field operations is required and all computations are performed in a common projective Z-coordinate representation to reduce the memory requirements of low-resource implementations. In terms of memory and speed the results outperform existing solutions and allow a fast and secure implementation suitable for low-resource devices and embedded systems.

The new formula for elliptic curves over finite fields of characteristic $q \neq 2, 3$ apply the co-Z method to the Montgomery ladder scalar multiplication. The given formula perform a differential addition-and-doubling operation of elliptic curve points using x-coordinates only, i.e. two projective X-coordinates of the involved

points and a common Z-coordinate. The formula leads to very efficient scalar multiplications especially suitable to low-resource devices. The practical constraint imposed by the implementations of both the modular multiplication and the modular squaring is considered, which may not support the result to be written in-place, which is overwriting one of the operands. This constraint allows saving memory with many efficient implementations of those. Unfortunately this typically implies the need of more memory than claimed in order to implement formula which have been designed with in-place operations. The formula can be applied on general elliptic curves and allow the integration of conventional countermeasures against implementation attacks. They can be efficiently applied in low-resource implementations of RFIDs, smart cards, and other embedded systems.

6.5.10.1 Key agreement protocol for mobile devices on elliptic curve cryptosystem

A cross-realm client-to-client password-authenticated key agreement (CR-C2C-PAKA) protocol is designed to solve the problem of secure client-to-client communication. It provides a method to achieve authenticated key agreement in a cross-realm setting for clients, who registered in cross realms (servers) with different passwords.

Secure client-to-client communications are required urgently in various areas, such as wireless networks, peer-to-peer networks, client-to-client E-commerce and so on, since the development of the electronic and network technologies is fast. For example, in a wireless network communication environment, a secure peer-to-peer channel, between a client *Alice* registered in one server S_A and another client *Bob* registered in a different server S_B, may be a primary concern over an insecure and public channel. Nowadays, electronic commerce are more popular and convenient, client-to-client businesses are also more and more prevalent day by day and mobile intelligent devices, such as cell phones, PDAs, notebook PCs and so on, are everywhere. Then arise the question, how to design such a Cross-realm client-to-client password-authenticated key agreement (CR-C2C-PAKA) protocol, which can be implemented using mobile intelligent devices.

CR-C2C-PAKA protocol was first proposed by Byun et al. [18] in 2002. A lot of researchers worked on these protocols from then, presented many attacks to show that the previous protocols were not secure and improved protocols to enhance the security. The first provably secure C2CPAKE protocol was introduced by Byun et al. [19] in 2007. Later in 2009 was shown by Feng et al. [20] that the existing protocols designed in secret key setting were not secure against password-compromise impersonation and proposed an improved protocol based on the public key cryptosystem with digital signature system which was proved secure to resist all known attacks, such as off-line password dictionary attack, Denning-Sacco attack, replay attack, one-way man-in-the-middle attack and password-compromise impersonation attack. The secure CR-C2C-PAKA protocol based on smart cards in secret-key setting with modified formal security proof was proposed lately by Jin et al. [21]. The smart cards are high cost and the auxiliary infrastructures and related standards of interfaces are lacking. Because of that such protocols were not widely implemented except in special areas. In 2009, Rhee et al. [22] proposed an improved scheme to enhance the security flaws of Khan-Zhang's scheme [23], which was vulnerable to user impersonation attack without using smart cards. Yang et al. [24] recently introduced elliptic curve cryptosystem into an ID-based remote user mutual authentication with key agreement scheme for mobile devices.

A new improved cross-realm client-to-client password-authenticated key agreement protocol based on elliptic curve cryptosystem for mobile devices is presented in the paper [9]. The proposed protocol is more secure, efficient, convenient, flexible and practical in our daily life. It can be implemented in secret-key (symmetric) setting with the resistance of known attacks including password-compromise impersonation attack. In order to augment the security flaws and increase the efficiency of computation with shorter key size elliptic curve cryptosystem is introduced into this protocol. Compared with the protocols based on smart cards or public key cryptosystems, the new protocol is designed for mobile devices, which are prevalent and convenient than smart cards or public cryptosystems in our daily life. The security of the protocol bases upon elliptic curve discrete logarithm problem. The risky password (verifier) tables or expensive auxiliary equipment are not required in this protocol. Rhee et al.'s remote authentication scheme [25] using elliptic curve cryptosystem is also improved and applied to the presented cross-realm client-to-client password-based authenticated key agreement protocol. Moreover, two additional functions

are provided for users and servers, called secrets update phase and revocation phase for security and flexibility. At last, the security analysis shows that the protocol is secure against known common attacks, including the password compromise impersonation attack in the secret-key setting.

6.5.11 Comparison: ECC vs. Others Alternative Cryptography for Resource-Constrained Devices

Table 23 - ECC vs. Others Alternative Cryptography

Cryptographic algorithm	Properties
ECC	• The smaller ECC keys it turn makes the cryptographic operations that must be performed by the communicating devices to be embedded into considerably smaller hardware, so that software applications may complete cryptographic operations with fewer processor cycles, and operations can be performed much faster, while still retaining equivalent security. This means reduced power consumption, less space consumed on the printed circuit board, and software applications that run more rapidly make lower memory demands. For communication using smaller devices and asymmetric cryptosystem ECC is used. • High security for relatively small key sizes. • Smaller key sizes, faster computations compared with other public-key cryptography • Reduce energy consumption and to prolong life time of sensor nodes • More suitable for mobile devices than other cryptosystem. • To solve the problems, several ID-based authentication protocols on ECC are proposed. • Faster execution timings for the schemes, which is beneficial to systems where real time performance is a critical factor. • In RSA cryptosystem, the security increases sub exponentially whereas in elliptic curve cryptosystem, the security increases directly exponentially. The consequence is smaller key sizes, bandwidth savings, and faster implementations features which are especially attractive for security applications where computational power and integrated circuit space is limited, such as smart cards, PC (personal computer) cards, and wireless devices. • ECC is more appropriate for resource-constrained devices compare to RSA. • Implementation of an ECC cryptographic library exists and also a common hardware architecture for accelerating ECC to be used in open SSL. **Weaknesses:** • ECC needs a key authentication centre (KAC) to maintain the certificates for users' public keys. • When the number of users is increased, KAC needs a large storage space to store users' public keys and certificates. • Users need additional computations to verify the other's certificate in these protocols.
NTRUEncrypt	**Smallest Footprint** • Smallest public key crypto available on market (8 kb) • Ideal for embedded devices where code size is a major limitation ○ Industrial sensors, RFID, medical devices **Highest performing** • Highest performance crypto on the market • 5x to 200x times faster than RSA and ECC

	• Consumes minimal resources including CPU and battery
	  ○ run time memory utilization below 4.5K
	• 60% data throughput improvement (over RSA) when integrated with SSL
	• Significantly reduces server resource utilization for large-scale deployments
	• Ideal for
	  ○ Low power or hard to access environments (battery powered, electric grid, remote sensors)
	  ○ High-volume transaction environments (payment processors, virtualization/cloud computing, etc.)
	Most secure
	
	• Resistant to Quantum Computing attacks
	• The higher level of security, the higher performance gains versus competition
	• Ideal for systems where they can't be updated easily (long-term)
	  ○ Satellites, medical devices, long-term data protection
	Customized for a variety of platforms and implementations
	
	• NTRU in SSL for embedded systems or web application
	• NTRU SDK for C/C++ or Java
	• NTRU has also been flashed onto chips directly, e.g., GPU's as well as integrated circuits, e.g., VHDL
	Sample customers who have deployed NTRU
	
	• **Texas Instruments** embedded NTRU in their OMAP chip, for use in wireless cellular telephony. More than one million OMAP chips that used NTRU as their crypto system have been built and shipped to TI customers
	• **WikID**, an identity management company, uses NTRU in their 2-factor authentication product.
	• **EchoSat**, a provider of payment processing solutions, has incorporated NTRU into its point-of-sale (POS) credit card devices to improve the performance of their payment server. Their server consolidates payments from all their clients (including Citgo gas stations in the US and Canada.) EchoSat also leverages NTRU for its post-quantum cryptography benefits, since their devices need to persist at client sites for years between replacements.

Hummingbird	• Ultra-lightweight cryptographic algorithm • For resource-constrained devices provide the designed security with small block size. • combination of block cipher and stream cipher • hybrid structure • Provide the designed security with small block size which is expected to meet the stringent response time and power consumption requirements in a large variety of embedded applications. • Resistant to the most common attacks to block ciphers and stream ciphers including birthday attacks, differential and linear cryptanalysis, structure attacks, algebraic attacks, cube attacks, etc. • When compared to the ultra-lightweight block cipher PRESENT implemented on similar platforms, experimental results show that after a system initialization procedure Hummingbird can achieve up to 99,2% and 82,4% larger throughput for a size-optimized and a speed-optimized implementations.
PRESENT	• An ultra-lightweight cipher that offers a level of security commensurate with a 64-bit block size and an 80-bit key • The cipher is to be implemented in hardware.

	• Applications will only require moderate security levels. Consequently, 80-bit security will be adequate. Note that this is also the position taken for hardware profile stream ciphers submitted to eSTREAM. • Applications are unlikely to require the encryption of large amounts of data. Implementations might therefore be optimized for performance or for space without too much practical impact. • In some applications it is possible that the key will be fixed at the time of device manufacture. In such cases there would be no need to re-key a device (which would incidentally rule out a range of key manipulation attacks). • After security, the physical space required for an implementation will be the primary consideration. This is closely followed by peak and average power consumption, with the timing requirements being a third important metric. • In applications that demand the most efficient use of space, the block cipher will often only be implemented as encryption-only. In this way it can be used within challenge-response authentication protocols and, with some careful state management, it could be used for both encryption and decryption of communications to and from the device by using the counter mode.

6.5.12 Commercial Products Embedding Elliptic Curve Cryptography

Due to the massive research studies and growing of industrial standards, a number of commercial products including Elliptic Curve Cryptography are available.

The most notable software implementation is the Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL) [miracl]. This is an implementation of methods and functions for cryptography, based on Number Theory and focused on multiple target platform like general purpose processors and embedded systems. MIRACL library functionalities are especially focused on Elliptic Curve Cryptography and Pairing-based Cryptography [boneh]. In order to achieve the maximum portability the MIRACL library provides both C and C++ interface, and the optimization of critical routines resides at assembly level. Developers fine-tuned memory management to best fit platforms with low resources. The MIRACL library implements cryptographic algorithms and arbitrary-precision arithmetic algorithms such as:

- Algorithms for elliptic curve cryptography over both prime fields and binary fields;
- AES encryption;
- RSA cryptography;
- Diffie-Hellman key exchange;
- DSA Digital Signature;
- Industrial standard hash functions like SHA-1 and SHA-2 family;
- Experimental implementations of Pairing-based cryptography.

MIRACL library supports many platforms such as: Intel, Atmel, Sun Oracle, ARM, IBM, Texas Instruments, MIPS Technologies, Analog Devices.

Another notable example of commercial Elliptic Curve Cryptography implementation is the “Cryptographic API: Next Generation” (CNG) module by Microsoft [microsoft]. This is a set of Microsoft Windows API useful to enable cryptography under the Microsoft operating system, developed to be the long-term replacement for the older so-called CryptoAPI. CNG is intended to be used by Windows developers to produce cryptographic applications focused on the secure exchange of documents, and exposes a C/C++ interface. CNG is supported since Windows Server 2008 and Windows Vista. CNG supports the so-called NSA Suite B algorithms [nsa] including all required algorithms: AES (all key sizes), the SHA-2 family (SHA-256, SHA-384 and SHA-512) of hashing algorithms, Elliptic Curve Diffie-Hellman, and Elliptic Curve Digital Signature (ECDSA) over the NIST-standard prime curves P-256, P-384, and P-521 [nist].

Sun Java System Web Server is a hardware Sun product. In addition to the support for RSA keys, Web Server 7.0 introduces support for Elliptic Curve Cryptography (ECC) [sun]. With this product it is possible generating a certificate request or a self-signed certificate using RSA keys or ECC keys. For RSA keys different key sizes can be provided. For ECC keys one should choose a specific curve. A number of curves have been named by various organizations (ANSI X9.62 [ansi962], NIST [nist], SECG [secg-B]). Supported NIST curves over prime fields are P-192, P-224, P-256, P-384, P-521. Supported NIST curves over binary fields are K-163, B-163, K-233, B-233, K-283, B-283, K-409, K-571, B-571. Supported SECG curves over prime fields are secp160k1, secp160r1, secp160r2, secp192k1, secp192r1, secp224k1, secp224r1, secp256r1, secp256k1, secp384r1, secp521r1. Supported SECG curves over binary fields are sect163k1, sect163r1, sect163r2, sect193r1, sect193r2, sect233k1, sect233r1, sect239k1, sect283k1, sect283r1, sect409k1, sect571k1, sect571r1. In particular, curves called secp192r1 and secp256r1 are the curves recommended also in ANSI X9.62 standard.

A widely used product embedding Elliptic Curve Cryptography is Java Standard Edition (Java SE) framework. This tool is a platform for programming in the Java language, to deploy portable applications for general use. Java SE includes a virtual machine, the core engine running Java code, and a set of libraries needed to allow the access of physical platform hardware. Java SE 6 [javase6] and Java SE 7 [javase7] offer implementations of Elliptic Curve Cryptography algorithms, over prime and binary fields.

Java Card [javacard] is another technology with support for Elliptic Curve Cryptography. Java Card is a framework built to allow Java applications to be run securely on smart cards and embedded devices with very low memory and resources. It is used in SIM cards (used in GSM mobile phones) and ATM cards [athena]. Java Card products are based on the Java Card Platform specifications developed by Sun Microsystems. The Java Card supports Elliptic Curve Cryptography algorithms like ECDSA and ECDH, following [ansix962] and [ieee] recommendations.

Certicom Security Builder Crypto [certicom-C] is another cryptographic product with Elliptic Curve Cryptography capabilities. It is a cross-platform cryptographic module including a range of current and legacy algorithms that provide security to constrained environments. It is compliant with NSA Suite B algorithms [nsa]. Security Builder Crypto acts as a software cryptographic provider within the Certicom Security Architecture, which is a modular solution designed to allow developers to quickly and cost-effectively embed security across multiple families and generations of devices.

6.5.13 Hardware implementations of Elliptic Curve Cryptography

A notable hardware implementation relying on Elliptic Curve Cryptography is the IBM Cryptocard [ibm].

An Elliptic Curve Cryptography module implementation in hardware description language is provided by Opencores [ipcores]. This module provides ECDH and ECDSA protocols over binary fields and NIST curves, with a performance of about 5,000 point multiplications per second in the 65 nm ASIC process.

An high number of implementations over low-resource devices can be found in literature [koschuch, kumar2003, kumar2004], and most of them refer to 8-bit CPUs. These works rely on microcontrollers and hardware coprocessors providing cryptographic primitives like elliptic curve point addiction, doubling and multiplication. Complete protocols like ECDH or ECDSA can be implemented more efficiently at higher level.

Another notable hardware implementation is [vanameron], where a Elliptic Curve Cryptography module based on prime field modulo 2^{384} is presented, with a realization on an ARM processor. The result is a little improvement on elaboration time passing from Intel opcode to ARM machine instruction set.

6.5.14 nSHIELD technology challenges

Conventional software-based implementations of ECC are flexible but inefficient, as a general-purpose instruction set architecture (ISA) of the underlying hardware is not optimized for cryptographic computations. In principle, an ISA can be extended to provide partial support for ECC-related arithmetic operations. However, such an approach cannot be applied to the nSHIELD project, which requires one to

address low-resources, low-power devices. Therefore, the technology challenge should be tackled by developing an advanced software layer that can be exploited to implement ECC-related arithmetic into low-resources embedded devices, thus obtaining a dedicated cryptographic co-processor that can support the SPD node.

Such goal can be achieved by taking advantage of the properties of the prime fields, which have been discussed above. In practice, two main aspects motivate this choice:

- prime fields can guarantee a sufficient level of flexibility;
- prime fields best fit implementations of ECC that cannot benefit of the design of a specific, dedicated data path for the underlying hardware layer.

6.6 Cryptographic Key Management and the Controlled Randomness Protocol

6.6.1 Introduction

In real world applications of cryptographic protocols, the key management problem refers to the life cycle management of cryptographic keys. It includes the necessary operations for key generation; distribution; storage; replacement and exchange; usage; and destruction. In order to retain specific security level, keys used in cryptographic algorithms and protocols must be periodically refreshed i.e., new keys are exchanged between communicating parties and old keys are replaced. These precautions ensure that only a specific amount of information is encrypted under the same key and thus, the exposure of information is minimized in case a key is leaked.

Key agreement is the process by which two or more parties agree on a common cryptographic key for a specific timeframe. Key transport is the process by which the agreed key is transferred to the participants. In many scenarios, the two processes occur simultaneously: the participants exchange information by which they both set and exchange the key(s) to be used (or some parts of it). In many scenarios, the key agreement and transport occur as exchange of control messages through a control channel. This channel does not interfere with the data channel in where actual secure data exchange takes place. A public-key cryptosystem (PKC) is commonly used in such setups in order to securely exchange through the control channel the symmetric-key cryptosystem (SKC) encryption/decryption keys used to securely exchange data within the data channel. The latter keys are often called ephemeral or session keys, since their lifetime spans a specific time period i.e., a session and then they are disposed.

In typical resource-limited environment, like the embedded systems in the pSHIELD environment, it is rather costly to implement and use a public-key cryptography (PKC) scheme for secure communication between two entities. When the resource constraints are more severe or the participants are all known beforehand, another option is to replace the “heavy” PKC scheme in the control channel with a lighter SKC scheme. The SKC scheme can use a master key in order to set and transfer the ephemeral keys needed for the data channel. In these cases and for sake of resource economy, the same SKC algorithm can be used in both the “control” and “data” channels albeit with different keys.

An embedded system can incur an interesting trade-off on security level and resource consumption. From a security point of view, the keys must be often refreshed, as explained earlier, in order to maintain the required security level. From a system resource consumption point of view, the keys must be rarely changed, in order to minimize the consumption of precious resources (processor, power and bandwidth). Further, in some usage scenarios, advanced care must be taken in order to ensure that the new keys will be available by the time they must be used, especially when only intermittent connectivity exists.

The “controlled randomness protocol” (CRP) for cryptographic key management is proposed as an improvement for the security level of secure communication protocols. The CRP allows multiple keys to be valid at any given time; it neither alters the total number of keys needed in the underlying cryptographic algorithms, nor the need of a control channel to periodically refresh keys. However, the increased security offered by CRP allows for far less frequent key exchanges.

6.6.2 Protocol Description

Conventional cryptographic schemes operate under the assumption that at most one key is active in any time moment. There is only one exception to this assumption. This is the transition periods when changing a cryptographic key. In these cases, at most two keys can be active in order to cope with delayed messages. We propose a novel approach of having more than one key at any given time moment. The approach is based on the concept of “controlled randomness” i.e., randomly using keys in a controlled environment. The concept of “controlled randomness” can be utilized in any protocol that uses temporal (ephemeral) keys. It increases protocol security with minimal computational overhead.

Assume a time period t = [0; T] composed of time slots t_1, t_2, ..., t_n such as $t = t_1 \cup t_2 \cup ... \cup t_n$. Each time slot t_i represents a session. Within each session one specific, temporal cryptographic key k_i is used in conventional schemes.

The Controlled Randomness Protocol works as follows. Within the time period t every cryptographic key k1, k_2... k_n is valid and can be used. The sender chooses with a uniform distribution a random integer i and encrypts the input data using the key k_i. The receiver has access to a secret mechanism and upon receiving a ciphertext c_i can deduce which of the possible keys was used for the encryption and thus, use the correct one to decrypt the ciphertext. The CRP does not dictate how all these keys are transferred to the receiver. It can be through a control channel using a PKC scheme, or an SKC with master key, or any other method. The CRP dictates how all these keys are used and reused within a time frame composed of many conventional sessions.

Two different methods are proposed for deriving the index, j, of the secret key used for a given ciphertext. The first method is using a synchronized random number generator (RNG) in both the sender and the receiver for the indexes.

The second method involves usage of a Keyed Hash Function (KHF) also known as Message Authentication Code (MAC). The sender and the receiver agree on a set of n encryption keys for a chosen encryption algorithm as usual and additionally on a set of n keys for computing MAC. The sender further uses an RNG. In this cases, the sender works as follows for every plaintext m:

1. Sender chooses a random number j.
2. Sender encrypts m under key k_j to produce the ciphertext $E(m, k_j)$.
3. Sender computes $H(E(m, k_j), h_j)$ i.e., the MAC of the ciphertext using the j-th MAC key.
4. Sender sends $E(m, k_j)||H(E(m, k_j), h_j)$, where || denotes the concatenation operation.

The receiver works as follows to recover m from the quantity $E(m, k_j)||H(E(m, k_j), h_j)$:

1. Receiver computes $H(E(m, k_j), h_j)$ for every possible j = 1, 2, ..., n. This step involves at most n MAC operations. Upon completing all computations, the receiver has derived the secret index j used by the sender.
2. Receiver decrypts $E(m, k_j)$ using the j-th decryption key. This step involves one decryption operation and derives the plaintext m.

6.6.3 Advantages of CRP

The concept of controlled randomness i.e., having multiple active keys at any given time moment, offers superior security characteristics compared to conventional protocols. The system designer can reuse well-known cryptographic blocks in a novel way to achieve increased security with minimal hassle:

- Minimal computational effort can be induced by CRP in the case that both sender and receiver can maintain a synchronized random number generator.

- The synchronization requirement can be relaxed, if the system can sustain some increased computational effort induced by the KHF (MAC) operations.
- In heavily constrained environments, the two above mechanisms can be replaced by sending the random number j with each packet. In this case, some security is indeed sacrificed since an attacker can know which packet is encrypted under what key. Yet, the intermix of keys allows consecutive packets to be encrypted under different keys and thus, protect against some cryptanalysis attacks.

The CRP allows in all above scenarios to extend the lifetime of each key way beyond the time of a conventional session. Further, it allows less frequent exchanges of messages in the control channel (if one is implemented), since less keys are needed to achieve a specific security level for a specific timeframe. An attack on the classical key management protocol with a master key of n bits has complexity $O(2^{2n/3})$; an attack on the RNG for the controlled randomness protocol with l keys has complexity $O(l2^m)$ (usually for m, the period of RNG, it holds m >> n); and an attack on the KHF method has a total complexity of $O(l(2^p + l2^{n/2}))$ where p the size in bits of the MAC keys.

6.7 Electronic Devices for Security Applications

Secure element is a concept that might encompass both hardware and software elements. It is referred usually to a secured storage included in one device or even in embedded devices chips. These devices are lately related to mobility causes or interactions; even though there are different types of secure elements in this section we will deal with secure elements for mobile devices. However there are also different secure elements that might not have to do with mobility objectives (at first sight), such as, Hardware Security Modules (HSMs) or TPMs (Trusted Platform Modules), usually stored in PCs and other stationary devices, servers or workstations.

A secure element is a secure crypto processor that accelerates key access management and enables tamper resistant properties as safeguard. For instance, Smart cards or chip card technology can play the role of secure elements. A Smart card is an embedded system with a simple communication protocol (ISO/IEC 7816-3 T=0/T=1 protocol, single wired protocol ISO/IEC 14443) that can be placed in many formats,

- Contact based cards: The contact cards are the most common type of smart card. This type of smart cards usually fulfills the ISO/IEC 7816 standard. ISO/IEC 7816 is a multi-part international standard broken into fourteen parts. ISO/IEC 7816 Parts 1, 2 and 3 deal only with contact smart cards and define the various aspects of the card and its interfaces, including the card's physical dimensions, the electrical interface and the communications protocols. The following ones are some examples of contact based smart card implementations:
  - Plastic card ISO/IEC 7810 ID1, where ISO/IEC 7810 is an international standard that defines four formats (physical characteristics) for identification cards.
  - Plastic card UICC (Universal Integrated Circuit Card)
  - Usb token
  - Embedded in devices
- Contactless cards: The contactless cards are cards that employ radio frequency (RFID) between the card and the reader. This type of smart cards usually conforms to the ISO/IEC 14443 standard as well as to ISO/IEC 7816 Parts 4, 5, 6, 8, 9, 11, 13 and 15. These last ISO/IEC 7816 Parts are relevant to all types of smart cards (contact as well as contactless).
  - Contactless payment cards (e.g. MasterCard PayPass)

Some of the main applications that can be used by the smart cards are: secure storage of sensible data and information, payment services with the inclusion of NFC technology, authentication services (two factors), legal digital signing, e-pass, e-health and ticketing.

6.7.1 Secure Microcontrollers

When designing secure devices, one approach would be using secure microcontrollers that already provide the basic cryptographic and secure features.

These secure microcontrollers could be organized as follows depending on their application:

- Specially designed for smartcard support:
  - Low computational power (up to 50 MHz).
  - Typically provide an ISO7816 interface (both at electronic and protocol level), although may provide other interfaces like SPI, I2C, UART or USB.
  - Includes security functions.
  - High level of tamper resistance.
- General purpose microcontrollers:
  - Embedded cryptographic functionality.
  - May contain an embedded Trusted Platform Module
  - Support application-level security (e.g. DRM).
- Microcontrollers targeted for use in high-throughput security appliances:
  - High computational power.
  - Examples of use may be VoIP or firewalls.

    Atmel, Infineon and Maxim are examples of manufacturers of these kinds of secure microcontrollers.

6.7.2 External cryptographic modules

A mechanism to provide additional cryptographic and secure functionalities to a general purpose microcontroller is to connect it to an external cryptographic module by means of a standard communication bus.

These external cryptographic modules are typically targeted at these types of applications:

- Payment systems, authentication and digital signatures.
- Secure embedded systems: point-of-sales (POS) terminals, M2M applications, network appliances (firewalls, routers, voice over IP)
- Digital rights management (DRM).

Most of these cryptographic modules are based on two families: smart cards and trusted platform modules (TPM)

- **Smart card**:
  - Oldest specification.
  - Can provide identification, authentication, data storage and application processing
  - Core module available from some manufactures as a chip with other form factor different from a typical smart card.
  - Usually implements ISO 7816 standard that defines:
    - Physical characteristics and electrical interfaces.
    - Transmission protocols.
    - Command and data interchange application protocols.
    - Commands for application management, cryptography, card management and security operations.
  - Availability of low footprint embedded operating systems to run in smart cards:
    - Multos
    - JavaCard
  - Main microcontroller could act as a bridge between an external device (computer) and the smart card core module.

- **Trusted Platform Module:**
  - Specification defined by the Trusted Computing Group (TCG).
  - Provides a trust framework that can be used to support applications such as Digital Rights Management (DRM). Ensures integrity and verification of the application and underlying system.

- Also provides basic cryptographic services facilitating digital signatures, key exchange, etc.
- Each TPM chip has a unique and secret RSA key burned in at manufacturing stage, so it is capable of performing platform authentication. For example, it can be used to verify the access to a host system waiting for the right device.

The main disadvantage of TPM is that it was initially only suited for PCs by using LPC interface but not for other types of microcontrollers, so most of the available chips on the market only provide this kind of interface.

Besides, there are discussions about whether the new trusted platform module is a real competitor of smart cards (which is a portable token than can be utilized across multiple systems) or if both can be complimentary (as TPM was initially thought as a fixed token).

6.7.3 Secure elements in mobility

Mobility is the area where security elements can play a main role. All OS (iOS, Android, BlackBerry, Windows Mobile Phone) have their own access to secure elements. Payments through mobile devices seem to be the one of the first use cases to take seriously the advantages of secure elements (jointly with NFC technologies.)

There are also other uses, such as, legal certificates storage in secure elements for different services; such as: digital signatures from mobile devices, mutual authentication for M2M, VoIP encryption, etc.

Practically there are three types of secure elements implementations:

1. Embedded in mobile phone (iPhones, some Samsun Galaxy series, etc.). There might be some dependencies with handset manufacturers. Secure elements that are embedded in the handset are included by mobile manufacturers and usually are closed to its usage (no possibility of interaction without any kind of partnership with the manufacturer of the handset). Manufacturers usually offer a kind of API for using this storage. The embedded secure elements are generally tamper resistant, and therefore secure by default. They usually pass a Common Criteria [3] security evaluation obtaining as a result the Evaluation Assurance Level (EAL1 through EAL7) certificate.

2. SIM based. SIM (Subscriber Identity Module) can play the role of a secure element. Indeed, SIM can be a tamper resistant secure platform [4]. SIM is an integrated circuit integrated in a SIM card that one particular user might have access to its services by using two passwords: a personal identification number (PIN) and personal unblocking code (PUK). Digital certificates can be stored in SIM. This enables multiple services for authenticating, encrypting, signing and consuming services by utilizing key pars securely stored in SIM.

3. There are also other new usages that can be explored by storing certificates into SIM and including SIM embedded in other devices; such as M2M communications (SML76 family for example [5]). This implies that some devices and applications, such as, smart meters in smart grid, in-car devices, routing and transportation control and building technologies have the skill to authenticate themselves securely without any human interaction. This would also enable a desired traceability if an error occurs during an industrial process for instance. The inclusion of SIM in these processes opens a new era for M2M security view.

4. Removable secure elements (SD Card) A Cryptographic Smart card is an embedded system which can be placed in a mobile phone or in a computer or any other device.

These secure elements have 8-32 bits CPU, a ROM with less than 512 kB, a crypto processor (AES, DES and ECC), a true random generator mechanisms, an EEPROM flash for memory, a communication interface for external data interchanging and memory management unit.

Security is guaranteed in smart cards by the usage of the following components:

- Crypto co-processors: specialised processors that process the cryptographic algorithms (they act as accelerator due to their specialisation)
- TRG (True Random Generator): this component aims at generating random numbers with the objective of composing key pars for the SC
- Memory Management Unit: this unit permits the access to the memory and being set up by the OS (Javacard). Therefore this unit let us secure the access to ROM, RAM and flash component.

Java Card is the usual smart card (development and runtime) environment; Java Card technology is an interoperable platform that enables smart cards and other resource-constrained devices to securely run Java technology-based applications. The interfaces between the javacard and external J2ME application are regulated through JSR177 interface (using APDU protocol).

Next generation of secure elements

Current secure elements are add-ons chips that are embedded in a (mobile) device. The future mobile processors, such as the new generation of ARMs [6] (current main processors for mobile devices), include a TEE (Trusted Execution Environment). This allows the processor as a secure element without any added elements (SDs, SIMs, embedded add-hoc chips). Not only information/data could be stored but also full application could be securely run by using trustlets. This TEE is isolated from the general operating system.

There are already some solutions that can manage the ARMs TTEs. Mobicore [7] from G&D is a new operating system able to manage those sensible application and data that might require an extra security.

The technology above mentioned is one of the examples of next generation for trusted computing technologies (beyond TPM) in the mobile field. Indeed, it provides a trust platform (Trustzone) within the main processor of the mobile. Although it could be a complement of different secure elements, such as, SIM, it provides the capacity to securely run a full application from/through the secure element concept.

nSHIELD should study different perspectives of secure elements and how to implement them. Indeed, there are many stakeholders behind of these technologies and business models might vary depending on the scenario and technology chosen. The emerging payment services might arise disruptive changes of technologies/applications horizon in these terms. nSHIELD should have this into account when assessing the different secure elements as nodes.

6.8 Trusted computing technologies

6.8.1 Background

The increased software and platform complexity in modern IT infrastructures give hard challenges with respect to protecting the computing environments from hostile or weak software. Many sensitive applications rely completely on that it runs on a platform with trusted software and hardware. One fundamental security requirement on all systems to achieve trust in a platform, is a secure boot process, i.e. the platform is only allowed to be booted into a well-known and trusted state. Apart from secure boot, even more challenging, is to ensure that the platform remains in a secure state as long as it is operational. But, the latter can never be achieved without the former, i.e., a secure boot process. A secure boot process is a fundamental prerequisite to be able to fulfil many of the node requirements identified in nSHIELD.

Trusted computing technologies as defined by the Trusted Computing Group (TCG) [1] are slowly starting to become adopted within the IT and telecommunication industry. Trusted computing technologies are built around the usage of a dedicated hardware module, the Trusted Platform Module (TPM) [2], supported by the majority of laptops on the market. It is also starting to be a standard component on almost all x86 platforms. On the embedded side, TPM usage is still limited, but has great potential as an important SPD enabler.

The TPM allows a user to securely create and store secrets, identify itself towards external parties and to report platform configuration status etc. The TCG architecture encompasses several computing platform of different types and the TPM can be used to serve an end-user, IT infrastructure or platform manager security needs. In particular, according to the TCG architecture, the TPM can be used to assist in a so-called authenticated boot. In an authenticated boot process, a platforms state (the sum of its components) is reliably measured and stored, i.e., it is possible at a later point in time (after the boot occurred) to reliable verify the platform software configuration status. This slightly differs from a secure boot process and for managed platforms an authenticated boot is often not enough, but a secure boot process is actually required. Consequently, the TCG mobile phone working group has specified a modified version of the TPM called Mobile Trusted Module (MTM) [3]. Different from the TPM, the MTM and related specifications from the TCG mobile phone working groups defines mechanisms, formats and processes for secure boot. However, the MTM has not to any extent been adopted by the mobile industry and it is not an alternative for most platforms with TPM. An interesting issue then is to study is how well a standard TPM component can be used to implement a secure boot process for a managed nSHEILD platform. We have chosen to look into this issue and will design a secure boot process that only is built upon TPM compliant commands.

Secure boot and trusted boot are not exclusive boot features. The same platform can be configured to support both principles. In nSHIELD we will mainly focus on the secure boot process, but we briefly also discussed how secure and trusted boot can co-exist and be configured on the same platform.

In addition to support trusted/secure boot, the TPM is a hardware unit that supports secure identification, signing for non-repudiation, protection of secrets through the TPM “sealing” functionality and remote attestation. Hence, the TPM is a potential very powerful hardware unit which potentially can provide many of the requested nSHIELD SPD functions.

As part of task T3.5 we will investigate the applicability of TPM both as a cryptographic hardware enabler for secure boot as well as cryptographic module service provider.

6.8.2 Attacks against TPM protected platforms

Considering the security of TPM when attacker has full access to the chip requires careful analysis of the threats which could arise due to the vulnerabilities of the chip design permitting hardware attacks. One such successful attack was recently reported by a security researcher Christopher Tarnovsky at the Black Hat conference. He subverted an Infineon SLE 66 microcontroller—a hardware component that implements the TPM specification. The SLE 66 is designed to protect against EM snooping, various kinds of side channel attacks, and pretty much any other conventional approach that one can think of [4]. The standpoint of the Trusted Computing Group on this has been that —they have never claimed that a physical attack — given enough time, specialized equipment, know-how and money — was impossible [5].

Furthermore, there are other possible attacks which could be performed if the attacker has physical access of the nSHIELD node containing TPM. One such attack is the TPM reset attack [6][7] in which TPM is sent hardware reset by using the ground driven reset line of the Low Pin Count (LPC) bus. Such a reset initializes all Platform Configuration Register (PCR) values to zero and then the attacker can load malicious code before setting the trusted configuration in the PCR registers again. The remote verifier cannot identify the malicious code because it is not reported in the PCR. The author in [6] also point out the theoretical possibility of TPM timing attack.

This clearly implies that for an nSHIELD node with TPM as trust anchor, some considerations must be taken to achieve the desired level of security. For example, the microcontroller chip itself should have defences against physical attacks. Furthermore, other measures can also be taken to build trust in the TPM chip. These may include TPM certification to meet TCG specifications and certified to at least an augmented EAL 4 (Evaluation Assurance Level) against the international Common Criteria certification standards [8].

6.8.3 Scenario for secure boot

In Task 3.1-3.3 we will make a boot design from a life cycle perspective, i.e., we will base our design taking into account the different phases that a typical IT product goes through during the lifetime of the product. When doing the design and analysis we will assume as generic product as possible although we recognize that not all life cycle phases are relevant for all types of products. The product phases we consider are:

- Product hardware manufacture
- Product configuration/customization
- Product first time deployment
- Product operation
- Product software upgrade
- Product recovery at major software failure

Some products also are subject to hardware upgrade. However, we will mainly focus on software upgrade and the connection to a secure boot based on TPM functionality.

The working assumption is that we will base the secure boot design using the TPM. Hence, the boot design will need to be closely aligned with the hardware design and the TPM usage and we expect a close co-operation between T3.5 and the node design task in this regard. In particular, the TPM interface requirements will be considered and also how exactly to use the TPM API to support the secure boot and secure software maintenance on the nodes.

6.8.4 Scenario for TPM as cryptographic module

Current embedded systems typically *do not* have TPM support even if the previous mentioned MTM that was developed by TCG for the mobile market. The reason is that the adoption of the specification in the mobile industry has been very slow. As consequence, TCG recently formed the Trusted Mobility Solutions working group [9]. We share the main scenario with this working group, i.e;

- With the growing usage of networked embedded system there is a need for enhanced protection of these devices as well as controlling their access to networks
- There is a large potential in using TPM/MTM functionality to protect device identity and device integrity
- TPM/MTM can be used to provide secure storage on nSHIELD nodes.
- TPM/MTM can function as an enabler for robust access control and application/data protection mechanisms to enable trusted connections to sensitive nSHIELD application systems

6.8.5 nSHIELD technology challenges

As there is a lack of broad support for TPM in current embedded system, we will work with practical solutions to securely interface TPM modules on major embedded architectures such as ARM based systems. Carful, analysis with respect to the design of hardware interfaces must be done.

In order to provide a secure boot rooted in TPM functionality, we need to make sure that the product always is booted into a secure state with the correct software images. The goal with the design we provide is to give such guarantees while still allowing enough flexibility in terms of software upgrade and recovery. The ultimate goal is to provide a secure boot design that relies on already available standard TPM cryptographic primitives.

The nSHIELD security requirements will be carefully analysed and the feasibility of using TPM functionality to meet the requirements with respect to identification, authentication and protected storage will be evaluated. We will contribute to the nSHIELD SPD architecture framework with respect to the TPM-functionality.

6.9 Anti-tamper Technologies

Secure electronic devices are widely utilized on systems that require functionalities such as user authentication, establishment of trusted communication channels and storing confidential data.

Some basic features that secure electronic devices may offer are:

- Identification.
- Authentication.
- Data encryption.

Anti-tamper mechanisms must be used on these secure devices to prevent access to critical information such as cryptographic keys. As an example, the following requirements are mandatory for FIPS 140-2 level 3 compliance:

- Use of tamper-resistant / tamper-evident coatings or seals.
- Tamper detection and response circuitry that clears keys and sensitive cryptographic material.

Anti-tamper mechanisms are used to prevent any attempt by an attacker to perform an unauthorized physical or electronic action against an electronic device which contains critical information.

It must be taken into account that it is not possible to achieve a 100% level of protection. Usually increasing the complexity of the solution increases the resources required to perform a successful attack as well but also increases the price of the device. Besides, developing a very strong and expensive solution may not compensate the damages caused by a tamper attack. Therefore, the complexity of the anti-tamper solution will vary depending on the desired protection level. Furthermore, in some cases devices could be designed in such a way that they do not require any additional anti-tamper solution for their targeted security level.

Depending on the type of protection provided, anti-tamper mechanisms could be classified into the following categories. Some of these mechanisms are only well suited for a certain range of products whereas they may not be effective on other ones:

- **Tamper Resistance**: this is one of the most basic mechanisms and it is widely used as it is usually quite easy to apply. It consists on using specialized materials to make tampering of a device or module difficult (e.g. using epoxy resin, special enclosures, locks, or security screws). Most times this kind of mechanism provides also tamper evidences as physical changes can be easily detected by a simple visual inspection.
- **Tamper Evidence:** the purpose of this mechanism is to make visible that a tamper attack was made. After a physical attack, evidences of it will remain clearly visible. There are many tamper evident materials and devices available on the market (most common ones are special seals).
- **Tamper Detection**: this is a more advanced mechanism and it is usually presented together with tamper response mechanisms, as it allows the attacked device to be aware of the tamper attempt, which is the first step prior to taking actions against the attack.
- **Tamper Response**: the device will detect the tamper attack and will execute the corresponding countermeasures to make its functionality or critical information not accessible to the attacker. Common actions are disabling the device, erasing private keys or deleting private information. This is the most appropriate anti tamper mechanism when dealing with portable devices that manage confidential information.

In order to qualify the protection provided by secure chips, most of them make reference to FIPS 140-2 standard, which is a U.S. government computer security standard used to accredit cryptographic modules. It defines four level of security:

- Level 1: lowest level of security with no physical requirements.
- Level 2: requires a certain physical protection.
- Level 3: requires countermeasures against tamper attacks (such as clear cryptographic keys).

- Level 4: device thought to work in unprotected environments. It can be quite hard to reach and may be required for military and certain governmental uses.

Based upon previous premises, there are two basic approaches for making electronic devices secure:

- Using single-chip solutions:
  - This is the easiest solution as a large range of this kind of chips is commercially available.
  - All critical data is always kept in a single chip and it is never transferred out to be used by another chip (or it is just transferred under petition of an authenticated user).
  - These secure chips already have some kind of anti-tamper protection.
- Using secure packaging:
  - This solution is commonly used when critical data is transferred among some different chips within the PCB so there is a chance that an external attacker could access de data path.
  - The entire PCB is encapsulated with a tamper mesh connected to a specialised low-power monitoring chip in order to detect any external attack and clear the critical data.
  - Price of the solution may rise as sometimes custom enclosures with the appropriate form factor must be developed.

Besides, as a complementary measure, there are some basic guidelines to offer a basic level of protection when designing PCBs such as:

- Using advanced chip packages such as BGAs instead of others like QF ones.
- Route critical data tracks by intermediate layers.
- Use blind VIAs for interconnection.
- Adoption of measures of this kind could be enough in cases when a non-single chip solution is utilized depending on the targeted security level of the device, while in some other cases they may not be necessary.

6.10 Physical Attacks and Defences

In this section we will discuss malicious attacks that are targeting security chips by measuring or modifying physical parameters. First we will give a classification of these attacks, and then we will discuss each type in detail.

Many different classifications of physical attacks can be found in literature, but they are usually discussed along the following two main aspects:

1. Impact on the normal behavior:

   - **Passive**
     Observing the device's behavior (output, response time, power consumption) without disturbing its operation
   - **Active**
     Tampering with the device's proper functioning (e.g. fault injection, hardware backdoors)
   - **Passive and Active Combined Attacks** (PACA)
     Passive and active techniques applied together

2. Level of physical access to the internals of the chips:

   - **Non-invasive**
     Attacks performed via the original interface. The chip is not modified during the process.
   - **Semi-invasive**
     Requires depackaging, but no electrical contact is made with the chip
   - **Invasive**
     The chip circuitry itself is tapped or modified during the attack

The structure of this section will follow the classification of passive and active attacks and deals with the question of invasiveness within those categories.

6.10.1 Passive Attacks

Passive attacks are analytical attacks aiming to extract information from the chip without modifying its normal operation. Basically we can talk about two types of passive attacks: those that aim to reverse engineer the chip, and Side Channel Analysis attacks.

6.10.1.1 Reverse engineering of circuitry

The aim of a reverse engineering attack is to find out the implementation details of the target chip's functionality. This step also serves as the basis of further attacking techniques e.g. fault injection or side channel attacks.

Decapsulation

Before performing an invasive attack an adversary needs to make samples by extracting the chip's package for further work. There are relatively simple chemical etching processes to depackage a chip, however such an operation is always risky, as the chip's internals may be irreversibly damaged during this process. So the adversaries usually need many samples and many trials to obtain a working result. If the depackaging is not feasible we still may assume that an adversary can get a chip die from the manufacturing and do the bonding by himself. There are ready-made bonding machines on the market for a moderate price. [1] pp.73-79

Deprocessing

Standard CMOS chips have many layers. During fabrication the metal wires are put on the silicon die with a special process. Deprocessing is the opposite of this process: the removing of these layers one-by-one to gain access to deeper layers. Various methods exist to do that:

- *Wet chemical etching*: Layers are removed by different chemicals depending on the top layer.
- *Plasma etching*: Layers are removed by a special gas. This method requires a special chamber.
- *Mechanical polishing*: Layers are polished by a special rough metal. It requires special machines for the fine work.

[1] pp.73-79

Optical Reverse Engineering

During the deprocessing process pictures can be taken of each layer in order to build a simulation of the chip's original operation. This is called optical reverse engineering and is usually done with an electron microscope. It requires high quality lenses and deferent wavelengths depending on the working distance and the required resolution. Other additional features can also help to reach higher resolutions like darkfield illumination, phase contrast, etc. Such equipment is very expensive to buy, however they can be rented on an hourly basis for a reasonable cost that an attacker can afford. [1] pp.79-83

Probe needles on data buses

If the chip works after decapsulation and it is possible to tap the inner buses, an attacker can use needles to connect to the chip's surface and to listen to the data communication. The gathered data can be used to obtain sensitive information, like private keys. This procedure requires a high quality microscope with a long working distance and enough working depth, a device test socket, a stage, and active or passive probes. [1] pp.83-89

6.10.1.2 Side Channel Analysis

Side Channel Analysis attacks aim to extract secret information by measuring physical parameters of the chip. Usually these measurements are done during normal operation without having internal connections to the chip. So these types of attacks can be considered as one of the most powerful passive non-invasive attacks.

Power consumption

Analyzing the power consumption of the chip is a very common side channel attack. Since each microprocessor instruction has a different power consumption profile, measuring the power consumed by the chip during the execution time of a cryptographic algorithm can allow an attacker to deduce what kind of operation the microprocessor is performing and – more importantly – what secrets the processor uses in the actual cryptographic operation.

Power analysis attacks require the attacker to have physical access to the device (but not to its internals). If the attacker is able to provide his own input to the cryptographic algorithm in question, then he can mount a chosen-plaintext attack, or in case he can obtain only the output of the cryptographic operation he can still perform a cipher text-only attack. Power analysis is relatively inexpensive to perform: it does not require specialized equipment, knowledge or resources.

There are three widely used power analysis techniques:

- Simple Power Analysis (SPA)
- Differential Power Analysis (DPA)
- Correlation Power Analysis (CPA)

Simple Power Analysis (SPA)

In a Simple Power Analysis attack, the attacker searches for patterns in power consumption during a security-sensitive operation.

In order to successfully execute such an attack, the attacker needs to know the algorithm (and its exact implementation) used by the target device. On the other hand, SPA only requires a small number of measured power traces to find patterns in the target device's power consumption.

SPA is especially useful for determining the **outcome of a branching instruction**. Since many cryptographic operations (such as the DES key schedule algorithm) use conditional execution that depends on secret data such as the key or sensitive intermediate values, SPA can be used to reveal the secret key used in the algorithm.

The most important countermeasure to prevent simple power analysis attacks is to avoid branching on secret data. However most up-to-date hardware implementation of symmetric cryptographic algorithms has small enough power consumption variation that SPA does not yield secret data.

Differential Power Analysis (DPA)

Differential Power Analysis searches for patterns in power consumption measurements *statistically*: checking the effect of input on power consumption at certain moments. It exploits the fact that **power consumption is different when processing '0' and '1' bit values**.

Unlike SPA, DPA requires a large number of power traces – with a variety of inputs – to find out correlations between the processed data. However, as it does not need detailed knowledge about the

exact cryptographic implementation used by the target[15], this method is non-invasive, and does not depend on knowledge of the plaintext input. So this kind of attack proved to be pretty successful on a large variety of devices.

There are several enhanced variants of DPA. *Automated template DPA*, for example, uses the variance of the power measurements instead of their magnitude hence it requires significantly fewer traces to succeed. *High order DPA functions* combine multiple samples from within a trace. An improved selection function can assign different weights for different traces or divide traces to more than two classes (see next section).

Correlation Power Analysis (CPA)

Correlation Power Analysis is an extension of DPA: instead of trying to divine one bit at a time, the attacker attempts to predict more bits, which usually means in practice the guessing of the Hamming weight of a word.

In CPA, the power usage of the device at a certain time is predicted as a function of certain key bits (depending on the cryptographic algorithm), and stored in a prediction matrix. The measured power values are stored in a consumption vector. The attacker compares the predicted and measured values by using a correlation coefficient; he checks for correlation between the consumption vector and each column of the prediction matrix.

Countermeasures against power analysis attacks

Several known different countermeasures exist against power analysis attacks, but all of them can be categorized into three categories. There are **protocol level** protections that reduce or even completely eliminate the probability of a successful attack through algorithm (re)design. The root of the vulnerability that is exploited by power analysis can be eliminated by decorrelating the observed power consumption profiles and the processed data. This technique is called **hiding**. The third possible solution is called **masking** that refers to the randomization of register values during cryptographic operations with masks.

References: [1] pp.56-59, [2] pp.5-6, [3] pp.18-24, [4], [5], [6], [7], [8], [9], [10], [11], [12].

Electromagnetic Radiation/Photo Emission Analysis

Electromagnetic radiation analysis (EMA) is similar in concept to power consumption analysis: the attacker can measure the strength of the electromagnetic field emanated by the target device while an operation is performed. EMA's main advantage over power analysis is that it usually doesn't require the full depackaging of the chip – the attacker does not need direct physical access to obtain the traces that form the basis of the analysis. Measuring electromagnetic radiation is also inexpensive to perform and does not require special equipment.

Another important advantage of EMA is the possibility to obtain more information than power analysis by positioning the measuring probes (coils) appropriately to focus on the most relevant part of the chip (usually on the cryptographic unit).

The two main types of EMA are very similar to the two main power analysis attack types:

- Simple Electromagnetic Analysis (SEMA) is analogous with Simple Power Analysis
- Differential Electromagnetic Analysis (DEMA) is analogous with Differential Power Analysis

References: [3], [4], [13] pp.56-59.

[15] The attacker still needs to know the used cryptographic algorithm to mount a DPA attack against a device.

Timing Analysis

Measuring the differences of an algorithm's execution time depending on the input parameters is one of the easiest processes that an attacker can carry out. If the execution time depends on secret key bits, then by measuring the decoding of many different messages using the same secret key can reveal the key. If the target is vulnerable to this kind of attack, the secret key can be guessed relatively quickly on a bit by bit basis with a pretty good probability of success.

References: [3] pp. 15-18, [1] pp. 54-55, [15]

6.10.2 Active Attacks

In our terminology an attack is active if it modifies the chip's normal functioning. The modification effect can be permanent, which will have effect on all future computations or temporary, which has only a limited lifetime. Permanent changes usually mean manipulating the circuit layout. The temporary influences are called fault injection attacks.

6.10.2.1 Attacks aiming to modify the circuit layout

Focused Ion Beam (FIB)

The focused ion beam (FIB) technique is frequently used in the semiconductor industry to modify an existing integrated circuit. Gallium ions are accelerated and focused into a beam, which can be as small as 5–10 nm in diameter. While lower beam currents can be used for imaging the integrated circuit (similarly to electron microscopy, but with ions instead of electrons), the higher ion currents can etch or mill the surface. It is also possible to create test points, establish contacts with the interconnection wires, etc. using the ion beam induced deposition.

FIB can be an affordable and particularly effective tool in the hands of an attacker. If a chip can be opened without disabling the normal operation and can be manipulated with a FIB tool, then there is not much left we can do to protect it. So FIB manipulation should be prevented by applying appropriate protective layers and sensors that can detect the breach of these layers.

References: [1] pp. 86-88, [4]

Hardware backdoors

Due to cost-cutting pressures the design and manufacture of the majority of ICs and other components are outsourced to third parties. It is expected by the end of this decade that the majority of ICs will be fabricated in cheap foundries in Far East countries. Without full control over the design and manufacturing process, it is possible for an attacker to modify the planned functionality of the product by inserting back-doors in it. Since the quality process during and after manufacturing are aiming to test the original (planned) functionality of the product – which usually not affected by the backdoors –, it is hard to detect them without targeted tests.

References: [16], [17], [18]

6.10.2.2 Physical Fault Injection Attacks

Fault Injection Attacks are active attacks with transitional effect. Faults are usually induced by influencing the chip's physical environment. They can abuse various known possibilities, e.g.:

- Tapping the wires
- Tampering with the external voltage (power glitches)
- Tampering with the external clock signal
- Inducing radiation (UV light, X-ray or other electromagnetic radiation)

- Tampering with the operating temperature
- Inducing eddy currents

Fault injection attacks can be modelled along different perspectives.

According to the preciosity of the error location:

- Specific location
- Specific region
- Non specific

According to the time of occurrence:

- Random (indeterminate) position
- Within some time interval
- Precisely determined point in time

According to the number of affected bits:

- Single-bit: if it alters exactly one bit
- Multi-bit: e.g., the state of a complete register

According to the effect induced:

- Bit flip: i.e. logic values are inverted
- Fixed state: i.e. logic values are tied to 0 or 1
- Inconsistent behavior (e.g. skipping of instructions on a microcontroller)

There are several practical methods how different types of fault injections can be performed in practice. In the upcoming sections we present several selected methods. What we should learn from the big number of different techniques is that we have to assume that attackers are capable of causing various types of fault injections relatively easily.

References: [3] pp 10-15, [2] pp. 7-10, [4], [19], [20].

Microprobing

Microprobing is an attack performed by connecting probes to the inside wires of the chip. It allows eavesdropping on signals inside a chip or injection of malicious signals and the analysis of reactions. This can be used for extraction of secret keys and memory contents.

The easiest way to read the memory with microprobing is to tap the memory bus. The attacker can use the monotonously increasing program counter to address the memory and observe the read instructions. The only catch left is that the attacker has to prevent the processor from executing jump, call or return instructions. This can be easily achieved with tiny modifications of the instruction decoder or program counter circuit (by cutting the right metal interconnects with a laser).

References: [3] pp 8-10, [4],

Light and X-Ray, Electromagnetic Radiation

Various types of electromagnetic radiations can be used to induce faults in the normal operation of the chip. UV light can be used to disable security fuses in EPROM and microcontrollers (however most modern microcontrollers are less susceptible as they are designed to withstand this). Intense white light is able to induce current and as such faults in the chip. Laser can reproduce a wide variety of faults with an effect similar to white light, but it can more precisely target a small circuit area. X-rays and ion beams can also be used as fault sources; however they are less common in practice. Their main advantage is that the depackaging step can be sometimes skipped.

References: [1] pp. 89-104, [4], [20], [21].

Tampering with the temperature of the chip

Security processors typically store secret keys in Static RAM (SRAM). To ensure security they are usually protected by tamper-sensing enclosures, which on detection of a tampering event powers down the chip. However if the data retention time exceeds the time to open the device and power up the memory, then this kind of protection mechanism can be defeated.

Cooling can increase data retention time in practice up to 10 seconds. Therefore some chips are protected by temperature sensors and zero the memory if the temperature drops down.

The opposite of this attack: localized heating can be used to effect permanent change of a single memory cell.

References: [1] pp. 62-72

Tampering with the external clock frequency

If the attacker temporally changes the external clock frequency, values that take longer to propagate (on the critical path) may not be handled correctly, and that can lead to exploitable flaws. These clock-signal glitch attacks are currently the simplest and most practical attacks to carry out. They are applicable against microcontrollers and some types of smartcards, but less effective against security measures realized by dedicated hardware. Their main use is to skip instructions in one of the following scenarios:

- Skipping conditional jump instructions and test instructions preceding them prevents execution of cryptographic barriers
- Extend the runtime of loops, e.g. in serial port output routines to see more of the memory after the output buffer
- Reduce the number of loops in cryptographic operations to transform the cipher into a weak one

References: [20], [1] pp. 59-61

Power Glitching

Power glitching attacks are based on increasing or dropping the power supply voltage (normally for 1-10 clock cycles) to cause the chip to *misinterpret or skip instructions*. Variations in the supply voltage can shift the threshold level of transistors and cause flip-flops to sample their input at different time or the security fuse to be read incorrectly.

Power glitching attacks are harder to exploit than clock glitches, because they have more parameters to get right: timing, amplitude and rise/fall time.

References: [20], [1] pp. 59-61, [4]

6.10.3 Passive and Active Combined Attacks (PACA)

Applying passive and active techniques at the same time can lead to very powerful attacks. Even if countermeasures exist against both classical kinds of attacks separately, the simple combination of them often not enough to efficiently defend against PACA attacks.

References: [2] pp.20, [22]

6.11 Secure Hardware implementation and testing guidelines

6.11.1 Physical protection of the chip

6.11.1.1 Multi-layering

Using multiple layers in a chip is a good solution for protecting a chip, making smaller chips, hiding data lines and reaching better chip quality. Furthermore multi-layered chips have considerably higher reverse engineering costs, since special and expensive tools and qualified engineers are required to do it.

References: [1] pp. 118-119, [4]

6.11.1.2 Protective Layer

The possibility of measuring the electrical potential on the chip surface is a serious threat that is the basis of many side channel attacks.

A possible defence against it is placing one or more protection layer, on top of important regions (e.g. memory). These protective layers are called shields. If a shield is damaged or detects any signs of an attack, the chip can react to this event. This reaction depends on the chip protection mechanism, but typically means resetting the chip or removing important data.

There are two basic shield types: active shields and passive shields; but there are other protection mechanisms, as well.

Active Shield

Active shields are current-carrying protection layers (typically a net of thin copper wires), whose integrity are continuously monitored by the chip and are able to detect even small changes in the physical environment.

Passive Shield

The passive shield usually is not monitored in every moment. This shield is required for the correct working of the chip. An example is the chip power layer, when lies on the top of the chip. If chemical etching is used for decapsulation, the acid immediately turns the chip wrong, because the power layer is destroyed.

Other protection measures

Applying opaque protective layer that's integrity is continuously monitored by phototransistors is a good hybrid solution.

References: [1] pp. 17-38, [4]

6.11.1.3 Unmarking, remarking and repackaging

Removing the marks from a semiconductor component or placing a custom chip ID inside the chip makes chip identification more expensive. Changing the mark of the real component to a better protected chip's mark can discourage and confuse the attacker.

However these techniques cause problems only to low budget attackers, since e.g. boundary scan on pins or observing signals on the chip interface can give away information on the chip.

References: [1] pp. 116-118

6.11.2 Obfuscating the design

6.11.2.1 Glue logic

Glue logic is obfuscating the transistor placement in the chip. There are automatic tools to solve this problem. Glue logic has a lot of reasons to use, like cloning existing chips without licensing, reaching higher reverse engineering cost, and improving chip performance.

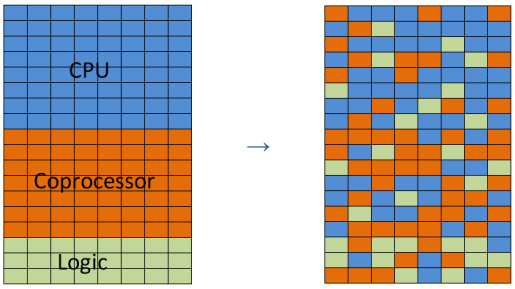

Figure 30 - Appling glue logic

References: [4]

6.11.2.2 Memory Encryption

We have already seen many techniques to read out the content of the memory. A possible countermeasure against this is memory encryption. In this case obtaining the raw memory content is not enough. Some modern chips provide batch- or chip-specific data encryption on memory and/or registers with on the fly encryption and decryption. The key should be located in these secure memory areas.

References: [1] pp. 17-38, [4]

6.11.2.3 Bus scrambling

Bus scrambling is not too difficult to implement by the designer/manufacturer but it makes tapping the bus considerably harder. The scrambling itself can be static, chip-specific or even session-specific.

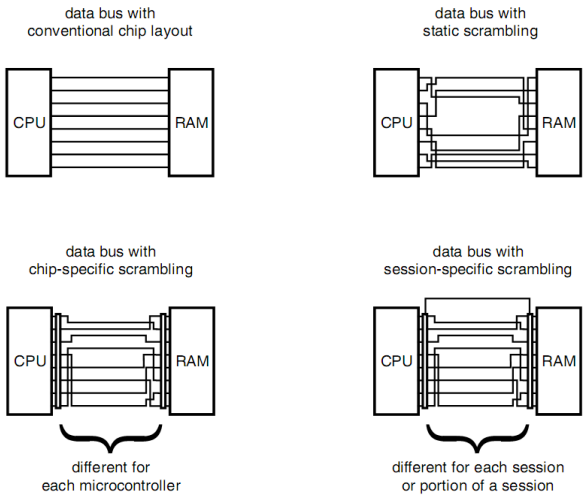

Figure 31 - Bus scrambling methods

References: [23] pp.510-565

6.11.3 Further Protection Measures

6.11.3.1 Physically Unclonable Functions (PUF)

PUFs are basically hardware one-way functions. They are not functions in a mathematical sense (as they can produce different outputs with the same input). Their input is called challenge and the output is called response, the corresponding input and output together called challenge-response pair (CRP).

The typical usage of a PUF has two stages: construction of the CRP database and evaluating the response of the PUF by the corresponding response in the database.

Quite a lot PUF type exists, here are some of them:

- Non-electronic PUFs
  Optical: reflective particle tag
  Paper: random fiber structure (currency notes)
  CD: difference in measured and intended lengths of lands and pits
  RF-DNA: near-field scattering of EM waves by randomly placed copper wires
  Magnetic: inherent uniqueness of magnetic media (credit card fraud prevention)
  Acoustic: characteristic frequency spectrum of an *acoustical delay line*
- Analog Electronic PUFs
  V_T : first IC identification technique (ICID)
  Power Distribution
  Coating
  Changes its CRPs considerably after an invasive attack, thus it can be used as an active shield at the same time)
  LC
- Delay-Based Intrinsic PUFs
  Arbiter
  Ring Oscillator
- Memory-Based Intrinsic PUFs
  SRAM
  Butterfly
  Latch
  Flip-flop

The most important properties of a PUF are these:

- **Evaluateable**: for a given PUF and challenge it is easy to evaluate the response
- **Unique**: contains some information about the identity of the physical entity embedding the PUF
- **Reproducible**: the response for a given challenge is reproducible up to a small error
- **Unclonable**: it is hard to construct a procedure that calculates the responses of a given PUF up to a small error
- **Unpredictable**: given only a set of challenge-response pairs, it is hard to predict the response for a random challenge that is not in the known set of CRPs up to a small error
- **One-way**: given only the response and the PUF, it is hard to find the challenge corresponding to the response
- **Tamper evident**: altering the physical entity embedding the PUF, changes the responses such that with high probability they do not match with the original response, not even up to a small error.

Secret key generation with PUFs is exploiting the intrinsic randomness introduced by the inevitable manufacturing variability. PUF responses are noisy, so an additional intermediate step is required to use them as secret keys. PUFs are highly secure place for a secret key, since it is never stored in non-volatile memory, only in volatile memory for a short time, during operations on it. This provides additional security against probing and other side channel attacks. That means almost tamper-proof key storage.

PUF has other applications, as well, like hardware-entangled cryptography and system identification.

Hardware-entangled cryptography is based on a new class of cryptographic primitives that contain PUFs integrated into them. Their very high security level arises from that they do not use external secret keys. The first result was a PUF-based block cipher.

PUFs' third and most typical usage is system identification. It works similarly to biometric identification: The verifier picks a CRP from the DB and compares the response to the one that was given by the PUF. FAR/FRR values can be fine-tuned with the acceptance threshold level.

References: [24], [25]

6.11.3.2 Unique Chip ID

Every chip with a unique ID can be traced during production, the can use their IDs to generate secret keys and their presence allows the manufacturer to generate blacklists after production.

Chip IDs are usually implemented with PUFs or write once, read multiple (WORM) memory (a.k.a. one-time programmable - OTP - memory).

References: [23] pp.510-565, [4]

6.11.3.3 Sensors

In the previous discussions we have already recommended the use of these types of sensors:

- Photo sensors monitoring the passive shield
- Voltage monitoring (Protection against power glitching)
- External clock frequency monitoring
- Temperature monitoring

However think twice if the planned sensors are all really useful and consider no power or no external clock situations, as well.

References: [23] pp.510-565, [4]

6.11.3.4 Further design guidelines

- Do not use standard cells!
- Use dummy structures to confuse the attacker, nevertheless they require additional room. They can be continuously monitored against tampering, as well.
- Put the memory into the harder-to-reach lower layers!
- Use ion-implanted ROMs!
- Scramble the memory cells inside the memory block!

References: [23] pp.510-565, [4], [26]

6.11.4 Risk analysis

To estimate the security of complex systems, usually risk analysis procedure is carried out. The outcome of this procedure is the list of risks that are consequences of the threats identified, which is used as a basis for mitigation techniques and establishment of the test cases for testing.

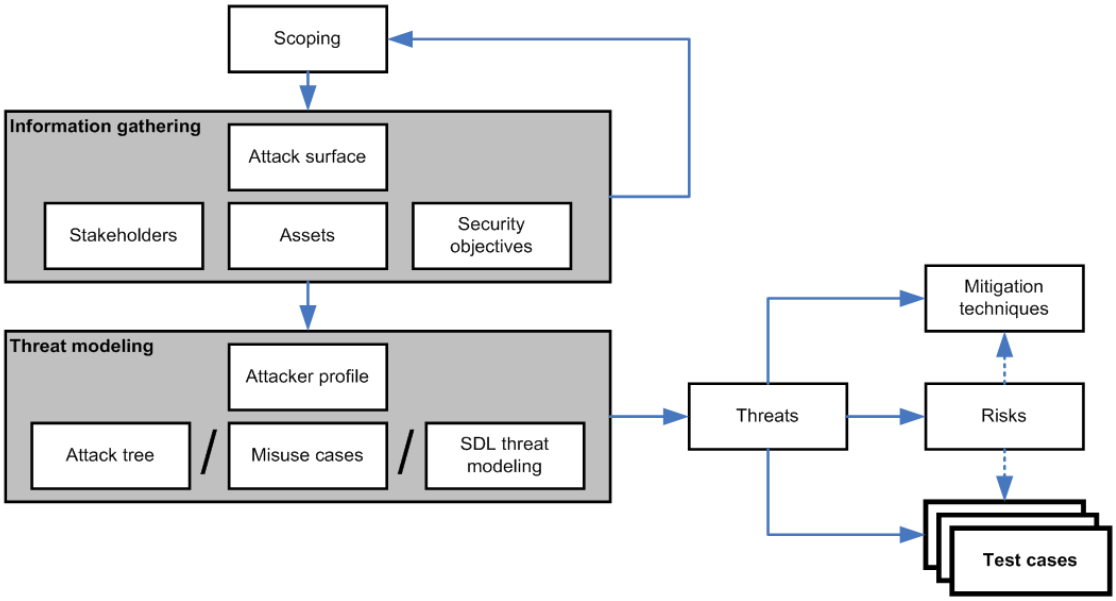

Figure 32 - Steps of test planning

6.11.4.1 Preparation and scoping

The first step in risk analysis is information gathering. All accessible relevant information has to be collected. Studying the architecture, design documents and specifications is required to succeed. Under this learning phase stakeholders, assets and security requirements are identified, the attack surface is defined.

The aim of scoping is to determine what should be checked and what not – it may take several iterations.

6.11.4.2 Threat modelling

Figure 33 shows the classical CIA classification of security properties and their threat counterparts in the more modern STRIDE model.

Threat	Security property
Spoofing	**A**uthentication
Tampering	**I**ntegrity
Repudiation	**N**on-repudiation
Information disclosure	**C**onfidentiality
Denial of service	**A**vailability
Elevation of privilege	**A**uthorization

Figure 33 - The STRIDE model along with the extended CIA model

In the followings some artefacts that are considered useful for testing will be mentioned.

Threat modelling identifies documents and rates threats. It is based on attack trees and misuse cases. The threat model helps to pinpoint exact test cases and sections of code that need close attention.

The attacker profiles describe possible internal and external agents that might want to realize the threats. Internal roles are going to be enlisted. Agents have malicious intent, so understanding their motivations are the most important step.

Attack trees (see Figure 34) are one of the bases that threat modelling relies on. They provide a good overview on security by systematically revealing possible attacks and risks. The starting point is a set of high level attack scenarios, the main goals of the attacker. Then preconditions (sub-goals) are identified step-by-step with AND / OR relationships between conditions, then the preconditions of sub-goals are identified and so forth. Iteration stops when all low level leafs are elementary conditions.

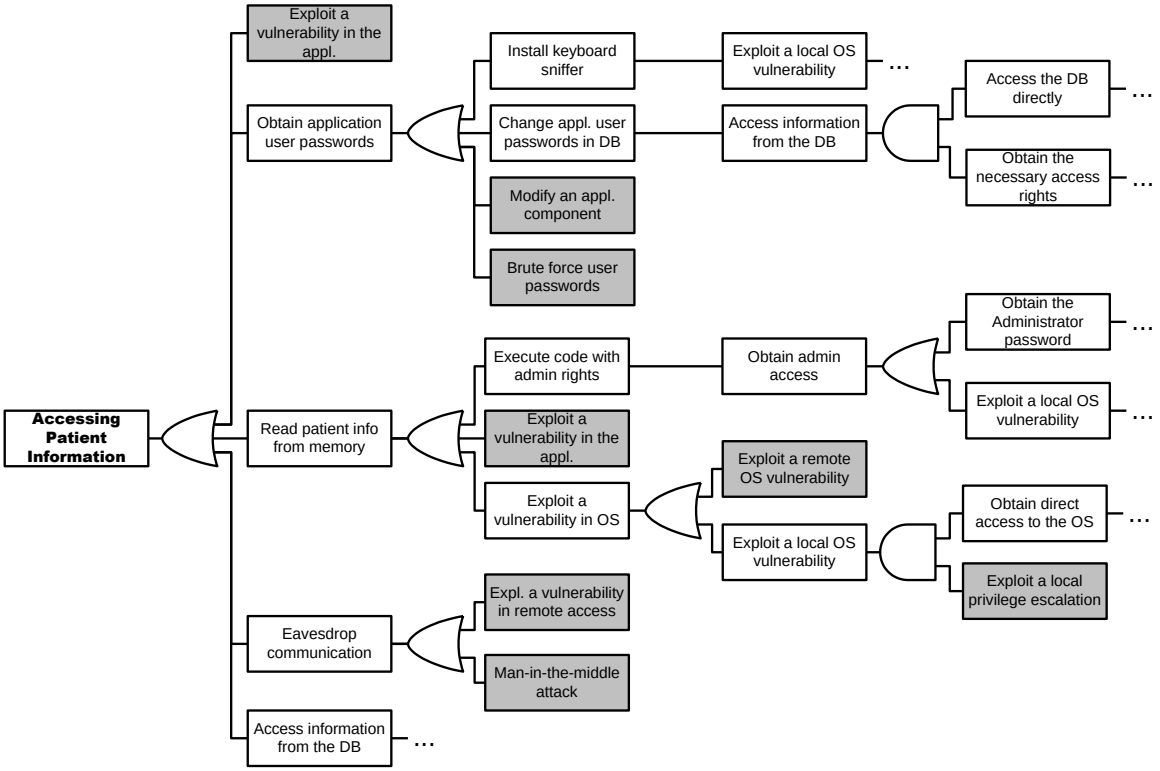

Figure 34 - Attack tree of a hypothetical medical device

With an attack tree in our hands, we can prepare test cases that check if the system is vulnerable to the identified threats.

Beside or instead of attack trees, threat modelling can be based on use-case models also. A classification of traditional use-cases from security perspective:

- **Use cases** – normal behavior assuming correct usage
- **Misuse cases** – unexpected usage, abnormal behavior
- **Abuse cases** – same as misuse cases, but *intentional*

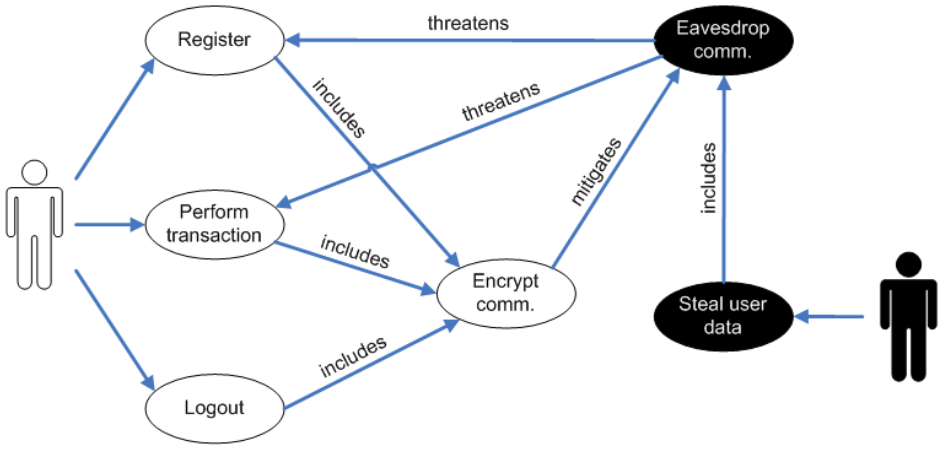

Figure 35 - A misuse/abuse case example

Mitigation objectives are addressing identified threats to an application design. Approaches to threat mitigation are redesigning the application, using standard and unique mitigation techniques and accept risk in accordance with policies.

Validation objectives are helping to ensure that threat models accurately reflect the application design and potential threats. The model, the enumerated threats, mitigations, assumptions or dependencies should be validated.

The threats uncovered by risk analysis are based on threat likelihood, severity and other factors. The goal of risk analysis is to focus the evaluation to the most relevant risks. Categories of risks are design flaws/weaknesses, conceptual error, weak or missing control, implementation bugs and operational vulnerabilities.

6.11.5 Testing guidelines

Security testing verifies if an application cannot in any way be misused by a malicious user. Bugs will sooner or later be triggered by an intelligent adversary (attacker), so we should always seek the answer for the question: is our product do something that it is not supposed to do. The goal of security testing is to get rid of all security-relevant bugs.

Vulnerabilities are side effects or extra functionalities that the attackers can exploit for their own purposes. Security requirements should come from “The system shall” and it must declare “The system shall not” items. This procedure is not just simple verification, because it needs specially trained staff and the security engineer should have the ability to think with the head of an attacker. During this procedure it is always needed prioritization between the possible test cases and the found vulnerabilities.

A possible security vulnerability classification’s elements are bugs and flows.

Bugs are implementation (i.e. coding) level introduction. They give the common security vulnerabilities, but their testing can be automated.

Flaws are design level introduction. They require expertise, are hard to handle or automate. Nevertheless it has become the most prevalent and critical issue.

Attackers nevertheless do not really care about if vulnerability stemmed from a bug or a flaw.

In the following, we also show how security auditing and security testing are different, besides the similarities in their notions.

6.11.5.1 Security audit

Aims of a security audit are:

- Have an independent “second sight” on security features
- Evaluate the overall security level of a solution
- Make sure that security requirements are fulfilled
- Certify compliance with standards or user requirements
- Risk analysis of discovered weaknesses
- In security audit, auditors are checking the whole software development process, they are checking compliance with requirements and they are doing security evaluation of software and hardware and they are evaluating of software development subcontractors through.
- On checking the whole software development process, the following is required: documents studying (e.g. coding guidelines, specifications), doing personal interviews, making quick tests. The developing plans, implementation rules, etc. have to be in accordance with company level rules and guidelines, application specific compliance requirements and international standards.

6.11.5.2 Security testing

Aims of security testing

- Discover implementation bugs
- Check for typical security relevant programming errors
- Similar to functional testing, but need more security knowledge and experience
- On security testing mustn’t have assumptions. Finding assumptions and then making them go away is necessary. There aren’t “can’ts” and “won’ts”. If something is possible, the attacker will find the way to exploit them. The only way to approach far enough tests is taking up security expertise and experience and developing a certain level of paranoia.

6.11.6 Testing techniques

There are lots of approaches of testing:

Figure 36 - Concluding the tests

- Static (on-design or source code) or dynamic(run-time)
- Functional (black-box, specification-based), structural (white-box, implementation-based) or both (Gray-box)
- Manual or automated test execution

Once the test results are available, we can identify the vulnerabilities, reveal missing protections and refine the set of revealed threats. We can make recommendations as feedback to the developers. After framing the recommendations we repeat our risk analysis on identified threats to prioritize the assembled action list.

6.11.6.1 Code review

Code review needs a trained eye for security problems. It should be performed for all high-priority code. This review is very effective, but also very labor-intensive. It can be supported by automated tools.

Possible approaches:

- Identify the variables that an attacker can control
- Follow the data
- Check for patterns of specific problems

6.11.6.2 Penetration testing

Penetration testing is testing an application remotely. The goal is to simulate actual attacks and measure how well the application is able to withstand them. For testing the application as a malicious user, manual techniques are used.

6.11.6.3 Manual run-time verification

During this verification we

- Observe how an application behaves under certain conditions
- Detect certain security issues
- Examine the application's behavior at run-time (black-box)

6.11.6.4 Automated security testing

There are also automated solutions for testing, such as source code/design analysers, binary analysis tools and run-time analysers (e.g. fuzzing tools).

6.12 References

Sec 6.1

[1] P. Trakadas, T. Zahariadis,H.C. Leligou,S. Voliotis, K. Papadopoulos, "AWISSENET: Setting up a Secure Wireless Sensor Network," 50th International Symposium ELMAR-2008, focused on Mobile Multimedia, Zadar, Croatia, 10-13 September 2008, pp. 519-523.

[2] K. Papadopoulos, Th. Zahariadis, N. Leligou, S. Voliotis, "Sensor networks security issues in augmented home environment," 12th IEEE International Symposium on Consumer Electronics (ISCE 2008), Vilamoura, Portugal, ISBN: 978-1-4244-2422-1, April 14-16, 2008.

[3] K. Dietrich, J. Winter – "Towards a Trustworthy, Lightweight Cloud Computing Framework for Embedded Systems" – Accepted for the 4th International Conference on Trust and Trustworthy Computing, 22-24 June 2011, Pittsburgh, PA USA.

[4] K. Dietrich - "Anonymous Client Authentication for Transport Layer Security" - Accepted for the 11th IFIP TC 6/TC 11 International Conference on Communications and Multimedia Security (CMS 2010), May 31 - June 2nd 2010, Linz, Austria.

[5] K. Dietrich - "Anonymous RFID Authentication using Trusted Computing Technologies" - Accepted for the 6th Workshop on RFID Security (RFIDSec 2010), June 7-9, 2010, Istanbul, Turkey.

[6] K. Dietrich, J. Winter - "A Secure and Practical Approach for Providing Anonymity Protection for Trusted Platforms" - Accepted fo the twelfth International Conference on Information and Communications Security (ICICS 2010), December 15-17, 2010, Barcelona, Spain.

[7] Th. Zahariadis, E. Ladis, H.C. Leligou, P. Trakadas, C. Tselikis, S. Voliotis, “Trust Models for Sensor Networks,” 50th International Symposium ELMAR-2008, focused on Mobile Multimedia, Zadar, Croatia, 10-13 September 2008.

[8] T. Zahariadis, H. Leligou, S. Voliotis, S. Maniatis, P. Trakadas, P. Karkazis, “An Energy and Trust-aware Routing Protocol for Large Wireless Sensor Networks”, WSEAS Transactions on Communications, Vol. 8, No. 9, Sept. 2009.

[9] A., J. Kangasharju, “On Interactions between Routing and Service Discovery in Wireless Sensor Networks”, ICOIN 2010, Busan, South-Korea.

[10] A. Fagiolini, F. Babboni, A. Bicchi, “Dynamic Distributed Intrusion Detection for Secure Multi-Robot Systems”, IEEE International Conference on Robotics and Automation, 2009. ICRA '09. pp.2723-2728, 12-17 May 2009.

[11] A. Fagiolini, G. Valenti, L. Pallottino, G. Dini, A. Bicchi, “Local Monitor Implementation for Decentralized Intrusion Detection in Secure Multi–Agent Systems”, IEEE International Conference on Automation Science and Engineering, 2007. CASE 2007, pp.454-459, 22-25 Sept. 2007.

[12] M. García-Otero, F. Álvarez-García, F. J. Casajús-Quirós, "Securing Wireless Sensor Networks by Using Location Information". Procs of the IWSSIP09, the 16th International Workshop on Systems Signals and Image Processing, Chalkida, Greece, Special Session: Security in WSNs, June 18, 2009.

[13] G. Dini, I.M. Savino, “A Security Architecture for Reconfigurable Networked Embedded Systems”, International Journal of Wireless Information Networks, vol.17/1-2, pp 11-25, 2010.

[14] A. Reiter, G. Neubauer, M. Kapfenberger, J. Winter, and K. Dietrich - "Seamless Integration of Trusted Computing into Standard Cryptographic Frameworks" – (awarded best paper at the conference!) Accepted for the 2nd International Conference on Trusted Systems - INTRUST 2010, 13th to 15th December 2010, Bejing, China.

[15] A. Lackorzynski, A. Warg, “VPFS: Taming Subsystems – Capabilities as Universal Resource Access Control in L4”, IIES '09: Proceedings of the Second Workshop on Isolation and Integration in Embedded Systems, March 2009.

[16] J. Barbarán, M. Díaz, I. Esteve, D. Garrido, L. Llopis, B. Rubio , “A Real-Time Component-Oriented Middleware for Wireless Sensor and Actor Networks”, First International Conference on Complex, Intelligent and Software Intensive Systems 2007 (CISIS 2007), pp. 3–10, 2007.

[17] Manuel Díaz, Daniel Garrido, Ana Reyna, “One Step Closer to the Internet of Things: SMEPP”, Future Internet Symposium FIS:2009, Berlin, Germany, 2009.

[18] Kristian Ellebæk Kjær, “A Survey of Context-Aware Middleware”, Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering, Innsbruck, Austria, pp. 148–155, 2007.

[19] Jeppe Brønsted, Klaus Marius Hansen, Mads Ingstrup, “A Survey of Service Composition Mechanisms in Ubiquitous Computing”, In Ubicomp 2007 Workshop Proceedings, pp. 87–92, 2007.

[20] Weishan Zhang and Klaus Marius Hansen, “An OWL/SWRL based Diagnosis Approach in a Pervasive Middleware”, The 20th International Conference on Software Engineering and Knowledge Engineering (SEKE’2008), pp. 893–898, 2008.

[21] Klaus Marius Hansen and Weishan Zhang and Goncalo Soares, “Ontology-Enabled Generation of Embedded Web Services”, The 20th International Conference on Software Engineering and Knowledge Engineering (SEKE’2008), pp. 345–350, 2008.

[22] Onur Derin, Erkan Diken and Leandro Fiorin, “A Middleware Approach to Achieving Fault Tolerance of Kahn Process Networks on Networks on Chips”, International Journal of Reconfigurable Computing, Vol. 2011, Article ID 295385, 2011.

Sec 6.3

[1] Daniel Hein, Johannes Wolkerstorfer, Norbert Felber: *ECC Is Ready for RFID – A Proof in Silicon.* Selected Areas In Cryptography, Lecture Notes in Computer Science, 2009, Volume 5381/2009, pages 401-413. (2009)

[2] Rodrigo Roman, Cristina Alcaraz, Javier Lopez: *A Survey of Cryptographic Primitives and Implementations for Hardware-Constrained Sensor Network Nodes.* Journal of Mobile Networks and Applications, Volume 12, Issue 4, August 2007. (2007)

[3] Nizamuddin, Shehzad Ashraf Ch., Waqas Nasar, Qaisar Javaid: *Efficient Signcryption Schemes based on Hyperlliptic Curve Cryptosystem.* In 7th international Conference on Emerging Technologies (ICET), 2011, pages 1-4. (2011)

[4] Tim Guneysu, Stefan Heyse, Christof Paar: *The Future of High-Speed Cryptography: New Computing Platforms and New Ciphers.* In GLSVLSI '11 Proceedings of the 21st edition of the great lakes symposium on Great lakes symposium on VLSI. (2011)

[5] Xiaoyu Shen, Zhenjun Du, Rong Chen: *Research on NTRU Algorithm for Mobile Java Security.* In International Conference on Scalable Computing and Communications; The Eighth International Conference on Embedded Computing 2009, SCALCOM-EMBEDDEDCOM'09, pages 366-369. (2009)

[6] Abdel Alim Kamal, Amr M. Youssef: *An FPGA Implementation of the NTRUEncrypt Cryptosystem.* In 2009 International Conference on Microelectronics (ICM), pages 209-212. (2009)

[7] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, Huaxiong Wang: *Pushing the Limits: A Very Compact and a Threshold Implementation of AES.* Advances in Cryptology – EUROCRYPT 2011 – 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, volume 6632, page 69. (2011)

[8] Axel York Poschmann: *Lightweight Cryptography – Cryptographic Engineering for a Pervasive World.* PhD Dissertation, Faculty of Electrical Engineering and information Technology Ruhr-University Bochum, Germany. (2009)

[9] Martin Hell, Thomas Johansson, Willi Meier: *Grain – A Stream Cipher for Constrained Environments.* International Journal of Wireless and Mobile Computing, Volume 2, No 1/2007, pages 86-93. (2007)

[10] Christophe De Canniere, Bart Prenel: *Trivium Specifications.* eStream Project http://www.ecrypt.eu.org/stream/triviump3.html (2008)

[11] Chi-Yuan Chen, Han-Chieh Chao: *A survey of key distribution in wireless sensor networks.* Published online in Wiley Online Library, Security and Communication Networks, DOI: 10.1002/sec.354. (2011)

[12] Marcos A. Simplicio Jr, Paulo S. L. M. Barreto, Cintia B. Margi, Tereza C. M. B. Carvalho: *A survey on key management mechanisms for distributed Wireless Sensor Networks.* Computer Networks: The International Journal of Computer and Telecommunications Networking, Volume 54, Issue 15, pages 2591-2612, October 2010. (2010)

[13] L. Eschenauer, V. Gligor: *A key-management scheme for distributed sensor networks.* In Proceedings of the Ninth ACM Conference on Computer and Communications Security (CCS'02), ACM, New York, NY, USA, 2002, pages. 41–47. (2002)

Sec 6.5

[aigner] H. Aigner, H. Bock, M. Hütter, and J. Wolkerstorfer. A low-cost ECC coprocessor for smartcards. In Cryptographic Hardware and Embedded Systems — CHES 2004, LNCS 3156, pp. 107–118, 2004.

[ansi962] ANSI X9.62-1998. "Public Key Cryptography for the Financial Services Industry: the Elliptic Curve Digital Signature Algorithm (ECDSA)". American Bankers Association, 1999.

[ansi963] ANSI X9.63-199x. “Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography”. American Bankers Association, 1999. Working Draft.

[athena] Athena smartcard PayProtect datasheet. http://www.athena scs.com/docs/products-solutions-datasheets/athena-payprotect.pdf.

[beautylabs] Binary finite field library 0.02. http://www.beautylabs.net/software/finitefields.html.

[boneh] D. Boneh, M. Franklin. “Identity-based encryption from the Weil pairing”. SIAM J. Computing, vol. 32, no. 3, pp. 586–615, 2003.

[botan] Botan library. http://botan.randombit.net/.

[brent] R.P. Brent. "Some integer factorization algorithms using elliptic curves". Research Report CMA-R32-45, Centre for Mathematical Analysis, The Australian National University, GPO Box 4, Canberra, ACT 2601, Australia, 1985.

[bsi] Bundesamt für Sicherheit in der Informationstechnik. https://www.bsi.bund.de/DE/Home/home_node.html.

[certicom-A] Certicom. “The Elliptic Curve Cryptosystem”. Whitepaper, 2000. http://www.comms.engg.susx.ac.uk/fft/crypto/EccWhite3.pdf”.

[certicom-B] Certicom. "Certicom corporate overview". http://www.certicom.com/images/pdfs/corp-certicom-121610.pdf.

[certicom-C] Certicom Security Builder Crypto cross-platform cryptographic module datasheet. http://www.certicom.com/images/pdfs/sb/ds-crypto-102210.pdf

[coppersmith1993] D. Coppersmith. "Solving linear equations over GF(2): Block Lanczos algorithm". Linear Algebra and its Applications, vol. 192, pp. 33–60, 1993.

[coppersmith1994] D. Coppersmith. "Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm". Mathematics of Computation, vol. 62, no. 205, pp. 333–350, 1994.

[cormen] T. H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. “Introduction to Algorithms”. MIT Press and McGraw-Hill, 2001.

[crypto] Crypto++ Library. http://www.cryptopp.com/.

[dierks] T. Dierks, C. Allen. “The TLS Protocol - Version 1.0.”, IETF RFC 2246, 1999. http://www.ietf.org/rfc/rfc2246.txt.

[diffie] W. Diffie, M. E. Hellman. "New Directions in Cryptography". IEEE Information Theory, Transactions on, vol. IT-22, no. 6, pp. 644–654, 1976.

[eberle] H. Eberle et al. “Architectural extensions for elliptic curve cryptography over GF(2m) on 8-bit microprocessors. In Proceedings of the 16th IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP 2005), pp. 343–349, 2005.

[fips800] Federal Information Processing Standard 800-57. “Recommendation for Key Management – Part 1: General”. 2007. http://csrc.nist.gov/publications/nistpubs/800-57/ sp800-57-Part1-revised2_Mar08-2007.pdf.

[fips186] Federal Information Processing Standard 186-2. “Digital Signature Standard (DSS)”. 2000. http://www.cs.haifa.ac.il/~orrd/IntroToCrypto/online/fips186-2.pdf.

[fips140] Federal Information Processing Standard 140-2. “Security Requirements for Cryptographic Modules”, 2001.

[fsml] FSML. Financial services markup language . Financial Services Technology Consortium, August,1999. Working Draft.

[gmp] GMP community "The GNU Multiple Precision Arithmetic Library". http: //gmplib.org/.

[guo] X. Guo, P. Schaumont. “Optimized System-on-Chip Integration of a Programmable ECC Coprocessor”. ACM Transactions on Reconfigurable Technology and Systems, Vol. 4, No. 1, 210.

[gupta] V. Gupta, D. Stebila, S. Fung. “Speeding Up Secure Web Transactions Using Elliptic Curve Cryptography”. Proceedings of 11th Network and Systems Security Symposium, pp. 231–239, 2004.

[gura] N. Gura et al. “Comparing elliptic curve cryptography and RSA on 8-bit CPUs”. In Cryptographic Hardware and Embedded Systems CHES 2004, LNCS 3156, pp. 119–132, 2004.

[hankerson] D. Hankerson, A. Menezes, S. Vanstone. "Guide to Elliptic Curve Cryptography", Springer, 2004

[hellman] M.E. Hellman, J.M. Reyneri. "Fast computation of discrete logarithms in GF (q)". Advances in Cryptology: Proceedings of CRYPTO 82, pp. 3–13, 1983.

[ibm] IBM CryptoCards. http://www-03.ibm.com/security/cryptocards/.

[ieee] IEEE P1363. “Standard Specifications for Public-Key Cryptography”. Institute of Electrical and Electronics Engineers, 2000.

[insito] InSiTo library. http://www.flexsecure.eu/insito/index.html.

[ipcores] IPCores. “Elliptic Curve Point Multiply and Verify Core”. http://www.ipcores.com/ elliptic_curve_crypto_ip_core.htm”.

[javacard] Java Card 3.0.1 Platform Specification. http://www.oracle.com/technetwork/ java/javacard/specs-jsp-136430.html”

[javase6] Java Standard Edition 6, API specification. http://docs.oracle.com/javase /6/docs/api/index.html”.

[javase7] Java Standard Edition 7, API specification. http://docs.oracle.com/javase /7/docs/api/index.html”.

[koblitz1987] N. Koblitz. "Elliptic curve cryptosystems". Mathematics of Computation vol. 48, no. 177, pp. 203–209, 1987.

[koblitz1993] N. Koblitz. "Introduction to Elliptic Curves and Modular Forms. New York: Springer-Verlag, 1993.

[koschuch] M. Koschuch, J. Lechner, A. Weitzer, J. Großschädl, A. Szekely, S. Tillich, J. Wolkerstorfer. “Hardware/Software Co-Design of Elliptic Curve Cryptography on an 8051 Microcontroller”, CHES 2006.

[kumar2003] S. S. Kumar et al. “Embedded end-to-end wireless security with ECDH key exchange”. In Proceedings of the 46th IEEE Midwest Symposium on Circuits and Systems (MWSCAS 2003), vol. 2, pp. 786–789, 2003.

[kumar2004] S. S. Kumar and C. Paar. “Reconfigurable instruction set extension for enabling ECC on an 8-bit processor”. In Field Programmable Logic and Application FPL 2004, LNCS 3203, pp. 586–595, 2004.

[lenstra1987] H.W. Lenstra. "Factoring Integers with Elliptic Curves". The Annals of Mathematics, vol. 126, no. 3, pp. 649–673, 1987.

[lenstra1993] A.K. Lenstra and Hendrik. W. Lenstra, Jr., editors. "The development of the number field sieve". Lecture Notes in Mathematics, vol. 1554, Springer–Verlag, 1993.

[microsoft] Microsoft Cryptographic API: Next Generation. http://msdn.microsoft.com/en-us/library/windows/desktop/aa376210%28v=vs.85%29.aspx”.

[miller] V.S. Miller. "Use of elliptic curves in cryptography". Advances in Cryptology: Proceedings of CRYPTO 85, pp. 417–426, 1985.

[maple] Maple help. http://www.maplesoft.com/support/help/AddOns/view.aspx?path=GMP”.

[miracl] MIRACL library, Shamus Software. http://www.shamus.ie/.

[montgomery1985] P. Montgomery. "Modular Multiplication Without Trial Division," Mathematics of Computation, vol. 44, pp. 519–521, 1985.

[montgomery1994] P.L. Montgomery. "A survey of modern integer factorization algorithms". CWI Quarterly, vol. 7, no. 4, pp. 337–366, 1994.

[nist] National Institute of Standards and Technology. “Recommended Elliptic Curves for Government Use”. http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf”.

[nist2007] National Institute of Standards and Technologies. “Crypto++ Library Versions 5.3.0 [32-bit and 64-bit] FIPS 140-2 Level 1 Validation”, 2007. http://csrc.nist.gov/groups/ STM/cmvp/documents/140-1/140sp/140sp819.pdf.

[nsa] NSA Suite B Cryptography. http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml.

[nss] Network Security Services Open Source Crypto Libraries. http://www.mozilla.org/projects/security/pki/nss/overview.html.

[openssh-A] The OpenSSH project. http://www.openssh.org.

[openssh-B] Elliptic Curve Cryptography implementation details in Openssh project. http://openbsd.das.ufsc.br/openssh/txt/release-5.7”.

[openssl] The OpenSSL project. http://www.openssl.org.

[panjwani] P. Panjwani, Y. Poeluev. “Additional ECC groups for IKE”. Internet Engineering Task Force, IPSec working group, 2000. http://www.ietf.org/.

[pohlig] S.C. Pohlig, M. Hellman. "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance". IEEE Information Theory, Transactions on, vol. IT-24, pp. 106–110, 1978.

[pollard1974] J.M. Pollard. "Theorems on factorization and primality testing". Proc. Cambridge Philos. Soc., vol. 76, pp. 521–528, 1974.

[pollard1978] J.M. Pollard. "A Monte Carlo methods for index computation (mod p)". Mathematics Computation, vol. 32, pp. 918–924, 1978.

[pomerance] C. Pomerance. "The quadratic sieve factoring algorithm". Advances in Cryptology: Proceedings of EUROCRYPT 84, vol. 209, 169–182, 1984.

[ramsdell] B. Ramsdell. “S/MIME Version 3 Message Specifica-tion”. RFC 2633, 1999.

[reza] M. Reza, H. Fatemi, I. Jebril, R. Salleh, “An FPGA based co-processor for elliptic curve cryptography”. In Proceedings of the Fifth IASTED International Conference on Communication Systems and Networks (AsiaCSN '08), pp. 73-77, 2008.

[rivest] R.L. Rivest, Adi Shamir, Leonard M. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978.

[rfc] RFC 6071: “IPsec and IKE Document Roadmap”. http://tools.ietf.org/html/rfc6071.

[silverman] R. D. Silverman. "The multiple polynomial quadratic sieve". Mathematics of Computation, vol 48, no. 177, pp. 329–339, 1987.

[secg-A] Standards for Efficient Cryptography Group, "Elliptic Curve Cryptography". http://www.secg.org/download/aid-386/sec1_final.pdf.

[secg-B] Standards for Efficient Cryptography Group, "Recommended Elliptic Curve Domain Parameters". http://www.secg.org/download/aid-386/sec2_final.pdf.

[sun] Sun Java System Web Server 7.0 Update 4 Administrator's Guide. http://docs.oracle.com/cd/E19316-01/820-6600/index.html.

[tate] J.T. Tate. "The Arithmetic of Elliptic Curves". Inventiones mathematicae, vol. 23, pp. 179–206, 1974.

[tunnell] J.B. Tunnell. "A Classical Diophantine Problem and Modular Forms of Weight 3/2". Inventiones mathematicae, vol. 72, pp. 323–334, 1983.

[vanameron] T. Van Ameron and W. Skiba. “Implementing efficient 384-bit NIST Elliptic Curve over prime fields on an ARM946E”. Proceedings of IEEE Military Communications Conference (MILCOM), 2008.

[wap] WAP WTLS. Wireless Application Protocol Wireless Transport Layer Security Specification. Wire-less Application Forum, February, 2000.

[williams] H.C. Williams. “A p + 1 method of factoring”. Mathematics of Computation, vol. 39, no. 159, pp. 225–234, 1982.

[wolfram] Wolfram Mathematica. http://library.wolfram.com/infocenter/Conferences/7518/Macalester_talk.txt.

[ylonen] T. Ylonen, T. Kivinen, M. Saarinen, T. Rinne, S. Lehtinen. “SSH Protocol Architecture”, IETF Internet draft, 2003.

[1] N. Aaraj, A. Raghunathan, S. Ravi, and N. K. Jha: Energy and Execution Time Analysis of a Software-based Trusted Platform Module, Department of Electrical Engineering, Princeton

University, Princeton, NJ 08544 NEC Laboratories America, Princeton, NJ 08540, Texas Instruments R&D Center, Bangalore, India, 2007.

[2] N. Aaraj, A. Raghunathan, S. Ravi, and N. K. Jha: **Analysis and design of a hardware/software trusted platform module for embedded systems,** Journal ACM Transactions on Embedded Computing Systems, Volume 8 Issue 1, December 2008.

[3] M. Strasser, TPM Emulator, [Online]. Available: http://developer.berlios. de/projects/tpm-emulator.

[4] Mersenne Twister Random Numbers Generator. [Online]. Available: http://www.math.sci.hiroshima-u.ac.jp/ m-mat/MT/ewhat-is-mt.html.

[5] A. Weimerskirch, C. Paar, S. Chang Shantz: Elliptic Curve Cryptography on a Palm OS Device, V. Varadharajan and Y. Mu (Eds.): ACISP 2001, LNCS 2119, pp. 502–513, 2001, Springer-Verlag Berlin Heidelberg 2001.

[6] J. Kar, Proxy Blind Multi-signature Scheme using ECC for handheld devices, Department of Information Technology, Al Musanna College of Technology Sultanate of Oman. Available at "International Association for Cryptology Research" http://eprint.iacr.org/2011/043.pdf, 2011.

[7] D. M. Alghazzawi, T. M. Salim and S. H. Hasan, A New Proxy Blind Signature Scheme based on ECDLP, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1, May 2011, ISSN (Online): 1694-0814.

[8] C. Gebotys, S. Ho, A. Ti, EM Analysis of Rijndael and ECC on a PDA, Dept of Electrical and Computer Engineering, University of Waterloo Waterloo, Canada, 2005.

[9] F. Wen, X. Li, S. Cui, Cross-realm Client-to-client Password-based Authenticated Key Agreement Protocol for Mobile Devices on Elliptic Curve Cryptosystem, Journal of Convergence Information Technology, Volume 6, Number 5. May 2011.

[10] W. Chou and Laerence, Elliptic curve cryptography and its applications to mobile device, Project Report, University of Maryland, 2003, http://www.cs.umd.edu/Honors/reports/ECCpaper.pdf

[11] M. Hutter, M. Joye, and Y. Sierra, Memory-Constrained Implementations of Elliptic Curve Cryptography in Co-Z Coordinate Representation, Published in A. Nitaj and D. Pointcheval, Ed., Progress in Cryptology, AFRICACRYPT 2011, vol. 6737 of Lecture Notes in Computer Science, pp. 170-187, Springer, 2011.

[12] D. F. Aranha, R. Dahab, J. Lopez and L. B. Oliveira, Efficient Implementation Of Elliptic Curve Cryptography In Wireless Sensors, Advances in Mathematics of Communications, Volume 4, No. 2, 2010, xxx–xxx.

[13] P. L. Montgomery. Speeding up the Pollard and elliptic curve methods of factorization. Mathematics of Computation, 48(177):243-264, 1987.

[14] N. Meloni. New point addition formul_ for ECC applications. In C. Carlet and B. Sunar, editors, Arithmetic of Finite Fields (WAIFI 2007), volume 4547 of Lecture Notes in Computer Science, pages 189-201. Springer-Verlag, 2007.

[15] R. R. Goundar, M. Joye, and A. Miyaji. Co-Z addition formula and binary ladders. In S. Mangard and F.-X. Standaert, editors, Cryptographic Hardware and Embedded Systems, CHES 2010, volume 2523 of Lecture Notes in Computer Science, pages 65-79. Springer-Verlag, 2010.

[16] D.Chaum, Blind Signature for Untraceable Payments, In Crypto 82, New York, Plenum Press, pp.199-203, 1983.

[17] Dr. B. Gladman, A Specification for Rijndael, the AES Algorithm, at fp.gladman.plus.com/cryptography_technology/ rijndael/aes.spec.311.pdf, 2003.

[18] J.W. Byun, I.R. Jeong, D.H. Lee and C.S. Park, Password-authenticated key exchange between clients with different password, in Proc. ICICS , pp. 134-146, 2002.

[19] J.W. Byun, D.H. Lee and J.I. Lim, EC2C-PAKE:An efficient client-to-client password-authenticated key agreement, Information Science,vol 177,no.19, pp. 3995-4013, 2007.

[20] D.G. Feng and J. Xu, A new client-to-client password-authenticated key agreement protocol, in Proc. IWCC 2009, pp. 63-76，2009.

[21] W. Jin and J. Xu, An efficient and provably secure cross-realm client-to-client password-authenticated key agreement protocol with smart cards, in Proc. CANS 2009, pp. 299-314, 2009.

[22] H.S. Rhee, J.O. Kwon and D.H. Lee, A remote user authentication scheme without using smart cards, Computers Standards & Interfaces ,vol.31,no.1, pp. 6-13,2009.

[23] M.K. Khan and J. Zhang, Improving the security of a flexible biometrics remote user authentication scheme, Computer Standards & Interfaces,vol. 29 ,no.1, pp. 82-85, 2007.

[24] J.H. Yang and C.C. Chang, An ID-based remote mutual authentication with key agreement scheme for mobile devices on elliptic curve cryptosystem, Computers & Security,vol.28 no.3-4, pp. 138-143,2009.
[25] H.S. Rhee, J.O. Kwon and D.H. Lee, A remote user authentication scheme without using smart cards, Computers Standards & Interfaces ,vol.31,no.1, pp. 6-13,2009.
[26] N. Howgrave-Graham, J. H. Silverman, W. Whyte, Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3, NTRU Cryptosystems, 2005.

Sec. 6.7

[1] ISO/IEC 7810: Identification cards – Physical characteristics, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=31432
[2] MasterCard PayPass, http://www.paypass.com/
[3] Common Criteria, http://www.commoncriteriaportal.org/
[4] NXP's Next Generation SWP-SIM Secure Element Bests Conventional SIMs with Increased Security and Performance, NXP, February 2012, http://www.nxp.com/news/press-releases/2012/02/nxp-s-next-generation-swp-sim-secure-element-bests-conventional-sims-with-increased-security-and-performance-.html
[5] Infineon SLM 76 family: M2M platform, http://www.infineon.com/cms/en/product/channel.html?channel=db3a3043156fd5730115f5956f981946

[6] ARM Processors, http://www.arm.com/products/processors/index.php
[7] MobiCore, G&D, http://www.gi-de.com/en/trends_and_insights/mobicore/mobicore_1/mobicore.jsp

Sec. 6.8

[1] Trusted Computing Group, http://www.trustedcomputinggroup.org
[2] TPM Specification, TPM Main Part I-III Design Principles, 2007, http://www.trustedcomputinggroup.org/resources
[3] TCG Mobile Phone Working Group, —TCG Mobile Trusted Module Specification‖, Version 1.0, 2008, http://www.trustedcomputinggroup.org/files/resource_files/87852F33-1D09-3519-AD0C0F141CC6B10D/Revision_6-tcg-mobile-trusted-module-1_0.pdf
[4] Infineon DRM/encryption chip succumbs to physical attack, http://arstechnica.com/security/news/2010/02/infineon-drmencryption-chip-succumbs-to-physical-attack.ars
[5] Black Hat Conference Report About TPMs, http://www.trustedcomputinggroup.org/community/2010/02/black_hat_conference_report_about_tpms
[6] A Security Assessment of Trusted Platform Modules, Computer Science Technical Report, TR2007-597, Evan R. Sparks, Evan.R.Sparks.07@Alum.Dartmouth.ORG, Senior Honors Thesis, http://www.cs.dartmouth.edu/~pkilab/sparks/
[7] Bernhard Kauer. OSLO: Improving the security of Trusted Computing. Technical report, Technische Universitt Dresden, Department of Computer Science, 2007
[8] The Common Criteria Evaluation Scheme, http://www.niap-ccevs.org/cc-scheme/cc_docs/
[9] TCG, Trusted Mobility Solutions, http://www.trustedcomputinggroup.org/developers/trusted_mobility_solutions

Sec. 6.10

[1] Skorobogatov, S.P. "Semi-Invasive Attacks: A New Approach to Hardware Security Analysis." *doctoral dissertation* (Computer Lab., Univ. of Cambridge), 2005. http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-630.pdf
[2] Oswald, David. "Development of an Integrated Environment for Side Channel Analysis and Fault Injection." *master thesis*, 2009. http://www.crypto.rub.de/imperia/md/content/texte/theses/da_oswald.pdf

[3] Koeune, F., and F.-X. Standaert. "A Tutorial on Physical Security and Side-Channel Attacks." *Foundations of Security Analysis and Design III: FOSAD 2004/2005 Tutorial Lectures, Lecture Notes in Computer Science* (Spring) vol 3655 (2005): pp 78-108.

[4] Witteman, Marc. "Advances in Smartcard Security." *Information Security Bulletin* (Riscure) Issue July 2002 (2002). http://www.riscure.com/fileadmin/images/Docs/ISB0707MW.pdf

[5] Kocher, P. "Differential Power Analysis." *Advances in Cryptology – Crypto 99* (Springer LNCS) vol 1666 (1999): pp 388–397. http://www.cryptography.com/public/pdf/DPA.pdf

[6] Popp, Thomas, Stefan Mangard, and Elisabeth Oswald. "Power Analysis Attacks and Countermeasures." *IEEE Design and Test of Computers* vol. 24, no. 6 (2007): pp. 535-543.

[7] Mangard, Stefan, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks: Revealing the Secrets of Smart Cards.* 2007. ISBN: 978-0-387-30857-9 http://www.amazon.com/Power-Analysis-Attacks-Revealing-Information/dp/0387308571

[8] Chari, S., J.R. Rao, and P. Rohatgi. "Template Attacks." *Proc. 4th Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES 02), LNCS 2523* (Springer), 2003: pp. 13-28. http://www.springerlink.com/content/7hr0n9vbc1le5a0u/

[9] Fournier, J.J.A., Simon Moore, Huiyun Li, Robert Mullins, and George Taylor. "Security Evaluation of Asynchronous Circuits." *Proc. 5th Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES 03), LNCS 2779* (Springer), 2003: pp. 137-151. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.58.3060&rep=rep1&type=pdf

[10] Popp, T., and S. Mangard. "Masked Dual-Rail Pre-Charge Logic: DPA-Resistance without Routing Constraints." *Proc. 7th Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES 05), LNCS 3659* (Springer), 2005: pp. 172-186. http://www.iacr.org/archive/ches2005/013.pdf

[11] Mangard, S., T. Popp, and B.M. Gammel. "Side-Channel Leakage of Masked CMOS Gates." *Proc. Topics in Cryptology: Cryptographers' Track at RSA Conf. (CT-RSA 05), LNCS 3376* (Springer), 2005: pp. 351-365. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.7411&rep=rep1&type=pdf

[12] Suzuki, D., and M. Saeki. "Security Evaluation of DPA Countermeasures Using Dual-Rail Pre-charge Logic Style." *," Proc. 8th Int'l Workshop Cryptographic Hardware and Embedded Systems (CHES 06), LNCS 4249* (Springer), 2006: pp. 255-269. http://www.springerlink.com/content/66m3272232465075/

[13] Quisquater, J.J., and D. Samyde. "ElectroMagnetic Analysis (EMA): Measures and Countermeasures for Smart Cards." *International Conference on Research in Smart Cards, E-smart 2001, Cannes, France*, 2001: pp 200–210. http://www.springerlink.com/content/chmydkq8x5tgdrce/

[14] Kocher, Paul C. "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems." *CRYPTO 1996*, 1996: pp 104–113. http://www.cryptography.com/public/pdf/TimingAttacks.pdf

[15] Tehranipoor, M, and F Koushanfar. "A Survey of Hardware Trojan Taxonomy and Detection." *Design & Test of Computers, IEEE*, 2009. http://trust-hub.org/resources/36/download/trojansurvey.pdf

[16] Sanno, Benjamin. "Detecting Hardware Trojans." 2009. http://www.crypto.rub.de/imperia/md/content/seminare/itsss09/benjamin_sanno.semembsec_termpaper_20090723_final.pdf

[17] Chakraborty, R.S., S. Narasimhan, and S. Bhunia. "Hardware Trojan: Threats and emerging solutions." *High Level Design Validation and Test Workshop, 2009. HLDVT 2009. IEEE International*, 2009: pp.166-171. http://trust-hub.org/resources/114/download/PID995123.pdf

[18] Guilley, Sylvain, Laurent Sauvage, Jean-Luc Danger, and Nidhal Selmane. "Fault Injection Resilience." *2010 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2010: pp.51-65. http://hal.archives-ouvertes.fr/docs/00/48/21/94/PDF/fdtc2010.pdf

[19] Hamid, Hagai Bar-El, Hamid Choukri, David Naccache, Michael Tunstall, and Claire Whelan. "The Sorcerer's Apprentice Guide to Fault Attacks." 2004.

http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=B1C01B4E255711EC413BBC2B815EF614?doi=10.1.1.5.6700&rep=rep1&type=pdf

[20] Agoyan, Michel, Jean-Max Dutertre, Amir-Pasha Mirbaha, David Naccache, Anne-Lise Ribotta, and Assia Tria. "Single-Bit DFA Using Multiple-Byte Laser Fault Injection." *IEEE International Conference on Technologies for Homeland Security (HST'2010) Waltham, MA, USA*, 2010.

[21] Giraud, C. "An RSA Implementation Resistant to Fault Attacks and to Simple Power Analysis." *Computers, IEEE Transactions* vol.55, no. 9 (2006): pp.1116-1120.
URL: http://audtm.net/~wcyang/lab/Lab_Paper/SCA/RSA_Impl_SPA.pdf

[22] Amiel, Frederic, Karine Villegas, Benoit Feix, and Louis Marcel. "Passive and Active Combined Attacks: Combining Fault Attacks and Side Channel Analysis." *Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC 2007)*, 2007: pp.92-102.

[23] Rankl, W., and Wolfgang Effing. *Smart Card Handbook (3rd ed.).* New York, NY, USA: John Wiley & Sons, Inc., 2002.
URL: http://www.amazon.com/Smart-Card-Handbook-Wolfgang-Rankl/dp/0470856688

[24] *Towards Hardware-Intrinsic Security: Foundations and Practice.* Springer, 2010.
URL: http://www.springer.com/computer/security+and+cryptology/book/978-3-642-14451-6

[25] Suh, G. E., and S. Devadas. "Physical Unclonable Functions for Device Authentication and Secret Key Generation." *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, 2007: pp. 9-14.

[26] Witteman, M., and M. Oostdijk. "Secure application programming in the presence of side channel attacks." *RSA Conference 2008* (Riscure), 2008.

[27] URL: http://www.riscure.com/fileadmin/images/Docs/Paper_Side_Channel_Patterns.pdf

7 SPD Node independent technologies

7.1 Authorization framework for SPD nodes

A driving force for the emergence of embedded systems with cryptographic capabilities, such as smart card chips and more recently TPMs, is the need to authenticate devices. Less effort has been devoted to the problem of authorization, i.e. determining what privileges a device can assert over a resource (e.g. another embedded system).

For traditional networked computer systems there exist well established authentication and authorization frameworks such as Active Directory, Kerberos and, more recently, XACML. On the web, OAuth is starting to emerge as a de facto standard for handling secure and fine grained API authorization.

For embedded systems there exists no established authentication and authorization framework today. However, as embedded systems are increasingly connected together in larger systems and these systems are dynamic where nodes can be added and removed, this becomes a problem. There has to be a way for nodes to both authenticate other nodes and determine what they are allowed to do.

One attempt using middleware is the EU-funded project SMEPP (Secure Middleware for Embedded Peer-to-Peer Systems) [16]. However, SMEPP assumes network access and many ESs are used in a context with no network access. Consequently, it is desirable to develop an authentication and authorization framework for embedded systems which does not assume network access.

The nSHIELD project will explore approaches for such an authentication and authorization framework for SPD nodes, which can execute on resource constrained ESs and also cope with a scenario where nodes are not continuously connected to each other or to the network.

7.2 Secure execution environment and trusted virtual domains for nano, micro and power nodes

In dynamic systems with frequent updates, it is very hard to provide security guarantees for a particular computing unit, and even harder for whole systems. To handle the associated risks, one need to provide secure execution environments that allow trustworthy, security critical applications to co-exist in the same system with less trustworthy or even insecure and non-security critical applications. This is indeed true for *all three* nSHIELD nodes, i.e. nano, micro and power nodes.

The hardware and software platform security enablers we develop within the project should provide isolation that guarantees secure interaction between software components in networked systems while still allowing components with different levels of trust to co-exist and share system resources. In general, the nSHIELD architectural framework for security allows secure execution and interoperability of services that are executed in a virtualized environment across different computing platforms and organizational and network domains. The project evaluates existing platform security technologies with respect to security and efficiency, and their enhancement towards the applicability within the nSHIELD architecture. We develop enhancements to existing solutions with the goal to improve interoperability and integration into heterogeneous internetworked systems as well as to research novel platform security enabling technologies. Especially for nano and micro nodes, we develop credible security and privacy provisioning for resource constrained devices and environments and devise methods for enhancing end-user visibility and control.

[16] Caro Benito et al, SMEPP- A Secure Middleware for Embedded P2P, ICT mobile summit 2009

7.2.1 Existing technologies

Virtualization allows running several virtual machines (VM or guests) to operate on a single physical device. This is done with the help of an additional software layer, a hypervisor or Virtual Machine Monitor (VMM) (the software managing the virtualization) that runs in the most privileged mode in the system beneath the operating system. Virtualization is often used to operate heterogeneous systems in parallel, to simplify migration or to improve system utilization. In those use cases it can be challenging to ensure the security of the VM. On the other hand, virtualization can be in turn also a mean for providing security, namely:

1. Isolation
2. Monitoring and trust

7.2.1.1 Isolation

Examples for such solutions are UCONKI [1] and SecVisor [2]. The latter ensures that kernel mode pages are not executable in user mode and vice versa. Each attempt to access code of the other mode leads to a trap into the hypervisor, which then can make sure that the operating mode of the processor, is switched accordingly and that only approved entry points to the kernel are used. Hence, for example, a buffer overflow attack in kernel mode will not execute user code and, as executable pages are also marked as read-only, applications cannot modify the kernel. There are also solutions trying mainly to protect the application layer, as for example the approach of Overshadow [3].

One of the main strengths of virtualization in the area of security and trust is the ability to isolate trusted code from non-trusted code [4]. This does not only apply to kernel integrity protection, but also to isolation between two VM or between a VM and a trusted service. Seshadri et al showed in [5] that virtualization along with standard memory protection support can achieve strong isolation. BitVisor [6] furthermore demonstrates how to address input/output security.

To provide an even higher degree of security, efforts were taken to also address the integrity of hypervisors themselves. HyperSafe [7], for example, suggests a way to “lock down” the memory and restricts pointer indexing. Hereby, the control-flow integrity is maintained by comparing referred branch targets with a stored control-flow graph.

Not only hypervisors are used for isolation. Microkernels such as the systems of the L4 family [8] are alternatives. L4 has even been completely formally verified. However, also virtual machine monitors such as the one of the Robin project [9] or the above mentioned SecVisor are upcoming verification targets. This is especially promising as hypervisors have a much thinner code base compared to often complex operating systems.

First efforts have been undertaken to apply virtualization for security on embedded systems as well [10].

The weakness of these solutions is that they only concentrate on single platforms, but do not address distributed systems, especially not heterogeneous ones.

7.2.1.2 Monitoring and Trust

When it comes to monitoring, the use of virtualization has an important advantage: the virtual machine monitor observes the guest from the outside, as it has higher system privileges. It is not part of the monitored system itself, which makes it hard for malware to hide or attack the hypervisor. An example for such a solution is Patagonix [11], which inspects each code before its execution and compares its hash with a prestored value in a database to see if the code is known and trusted. The Livewire [12] approach provides intrusion detection.

Virtualization can be enhanced by trusted computing technology. Yet, there are challenges to overcome as virtual machines usually would not get direct access to the TPM. The Terra hypervisor [13] uses certificates for attestation on various assurance levels. It is capable of isolating virtual machines, so that even the owner of a (physical) machine is prevented from accessing the contents of the virtual machines.

Another approach [14] uses a “Virtual TPM” implemented in software. It allows the virtual machines to communicate directly with a secure software-TPM which itself is linked to a physical TPM. To facilitate trusted computing on embedded systems, Winter [15] suggests the usage of ARM’s virtualization hardware TrustZone [16]. The Trusted Computing Group proposes furthermore a Mobile Trusted Module (MTM) [17]. The field of application of trusted virtualization on embedded systems is wide. Selhorst et al. [18] describe a secure signing environment where process isolation and platform attestation enable the trusted sending of text messages.

One weakness of Trusted Computing is the resources required for operations. Here it is important to compare existing solutions and point out improvements.

A *Trusted Virtual Domain* (TVD) [19][20][21][22] is a coalition of virtual machines that share virtualized resources for I/O and computation. Virtualization of resources as well as machines allows creating arbitrary virtual networks that operate independently from architecture and topology of the underlying hardware platform. In a TVD, interaction between VMs is modeled and regulated through shared TVD resources, for example virtual networks or storage. A TVD establishes a certain level of trust between members of a domain based on an admission policy enforced on these entities upon joining the TVD. The management of the TVD infrastructure is done through a central server (TVD Master) that can be used to define security policies for the TVD, and keep track of the availability and configurations of physical and virtual entities in the TVD. The TVD Master allows definition of the network topology in a way to ensure complete isolation of the TVD-specific data when stored, processed or communicated via network. This means that the physical and virtual entities of a network are connected or not to each other based on the delimited confinement boundaries of the TVD.

In the Trusted Virtual Datacenter described in [23], resource assignment, resource access and communication between virtual machines are controlled by means of a two-sided policy based on the non-hierarchical enforcement model [24]. On the one hand, the policy defines the security context of a virtual machine by “labeling” the set of the resources it can access to. On the other hand, the policy defines some collocation rules, which, for example, enforce restrictions on which virtual machines can run on the same platforms at the same time.

However, this kind of solutions has limitations when considering infrastructures of heterogeneous devices where trust domains are defined, with physical or virtual entities entering and leaving the trust domains. In this case, the challenge, which is not addressed in current solutions, can be summarized in the following points:

- Upon a change in the network topology, e.g., in case a node enters or leaves a trust domain, the component-specific security policies should be automatically adapted and enforced without a change in the high-level security policy. Current fully centralized solutions, do not account for this kind of scenarios.
- The central management service that defines the security policy for the trust domain should be continuously reachable by the physical devices in the domain in order for these to stay synchronized with the security policy updates. The reliance on a single central server for controlling the trust domains represents a potential threat for the maintenance of trust in the domain.
- When a device or component is attacked, or its configuration is (unintentionally) modified in a way to pose a security threat that would undermine the level of trust in the domain, these changes must be reported to the central security management point in order to account for the potential consequences. This requirement cannot be addressed by current solutions; they solely rely on integrity measurements for admission control, but miss to monitor continuously.

7.2.2 The role of secure execution and trusted domains in nSHIELD

The nSHIELD project addresses the following two important SPD technology aspects:

1. A major goal is to provide a secure and dependable architectural framework that allows seamless exploitation of SPD resources in heterogeneous domains.
2. The nSHIELD different nodes contain a number of relatively complex and/or security sensitive software components (especially those that handles authentication, encryption and key exchange). To reduce the software attack threats against these software components they must be isolated from non-trusted software components concurrently running on the system. Furthermore, to prevent root kit attacks and attacks against OS kernels, secure monitoring of and integrity protection of these and other security sensitive software components should be provided.

A major opportunity to address the first aspect is to utilize the trusted virtual domain concept. How to adapt and use that to fulfill the nSHIELD specific requirements will be addressed.

Secure isolation and protection of security sensitive software components is a major an important task in the project. Secure isolation gives in turn secure execution. The project will provide secure isolation and monitoring through own developed virtualization software or what is often referred to as a hypervisor software layer.

7.2.3 References

[1] M. Xu, X. Jiang, R. Sandhu, and X. Zhang. Towards a VMM-based Usage Control Framework for OS Kernel Integrity Protection. In proceedings of the 12th ACM Symposium on Access Control Models and Technologies (SACMAT 2007), June 2007

[2] A. Seshadri, M. Luk, N. Qu, and A. Perrig. SecVisor: A Tiny Hypervisor to Provide Lifetime Kernel Code Integrity for Commodity Oses. In proceedings of the 21st Symposium on Operating System Principles(SOSP 2007), October 2007.

[3] X. Chen, T. Garfinkel, E. C. Lewis, P. Subrahmanyam, C. A. Waldspurger, D. Boneh, J. Dwoskin, and D. Ports. Overshadow: A Virtualization-Based Approach to Retrofitting Protection in Commodity Operating Systems. In proceedings of the 13th Annual International ACM Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), March 2008.

[4] J. Brakensiek, A. Dröge, M. Botteck, H. Härtig, and A. Lackorzynski. Virtualization as an Enabler for Security in Mobile Devices. In First Workshop on Isolation and Integration in Embedded Systems (IIES'08) (Glasgow, UK), April 2008.

[5] J. M. McCune, Y. Li, N. Qu, Z. Zhou, A. Datta, V. Gligor and A. Perrig. TrustVisor: Efficient TCB Reduction and Attestation. Proceedings of IEEE Symposium on Security and Privacy (Oakland 2010), May, 2010.

[6] T. Shinagawa et al., BitVisor: A Thin Hypervisor for Enforcing I/O Device Security. In proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual Execution Environments (VEE '09) (Washington, D.C., USA), March 2009.

[7] Z. Wang and X. Jiang. HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity. In IEEE Symposium on Security and Privacy (SP), 2010.

[8] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. seL4: Formal veryFormal Verification of an OS Kernel, Proceedings of the 22nd ACMSymposium on OS Principles (SOSP '09) (Big Sky, MT, USA), October2009.

[9] H. Tews et al, Nova Micro-Hypervisor Verification Formal, machine-checked verification of one module of the kernel source code (Robin deliverable D.13), 2008, http://www.cs.ru.nl/~tews/Robin/tr.pdf

[10] C. Gehrmann, D., Heradon and K. D. Nilsson, Are there good Reasons for Protecting Mobile Phones with Hypervisors? In: IEEE Consumer Communications and Networking Conference, 9-12 Jan 2011, Las Vegas, Nevada, USA.

[11] L. Litty, H. A. Lagar-Cavilla, and D. Lie. Hypervisor Support for Identifying Covertly Executing Binaries. In proceedings of the 17th USENIX Security Symposium (San Jose, CA, USA), July 28 - August 1, 2008, pp. 243-258.

[12] T. Garfinkel and M. Rosenblum. A Virtual Machine Introspection Based Architecture for Intrusion Detection. In proceedings of the Network and Distributed Systems Security Symposium, February 2003.

[13] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A virtual machine-based platform for trusted computing. In Proceedings of ACM SOSP, 2003.

[14] S. Berger et al, vTPM: Virtualizing the Trusted Platform Module, IBM Research Report, 2006, http://domino.research.ibm.com/library/cyberdig.nsf/papers/A0163FFF5B1A61FE85257178004EEE39/$File/rc23879.pdf
[15] J. Winter, Trusted Computing Building Blocks for Embedded Linux-based ARM TrustZone Platforms, Proceedings of the 3rd {ACM} Workshop on Scalable Trusted Computing, Springer, 2008.
[16] ARM TrustZone, http://www.arm.com/products/processors/technologies/trustzone.php,
[17] Trusted Computing Group (TCG), Mobile Trusted Module (MTM) specification, May2009, http://www.trustedcomputinggroup.org
[18] M. Selhorst et a., Toward a Trusted Mobile Desktop, Trust and Trustworthy Computing: Third International Conference, TRUST 2010, Springer.
[19] S. Cabuk, C. Dalton, H. Ramasamy and M Schunte, Towards automated provisioning of secure virtualized network", Proceedings of the 14th ACM Conference on Computer and Communications Security Alexandria, Virginia, USA,October 28 - 31, 2007, pp. 235-245
[20] S. Berger, S., R. Cáceres D. Pendarakis, R. Sailer, E. Valdez, R. Perez, W. Schildhauer and Srinivasan, Managing security in the trusted virtual datacenter", SIGOPS Oper. Syst. Rev. 42, 1, January 2008, pp. 40-47
[21] [C Serdar, C. Dalton, K. Eriksson, D. Kuhlmann, H. Govind V. Ramasamy, G. Ramunno, A-R. Sadeghi, M. Schunter and C. Stüble, Towards Automated Security Policy Enforcement in Multi-Tenant Virtual Data Centers ", Special Issue of Journal of Computer Science on EU's ICT Security Research, 2009.
[22] H. Löhr, A-R. Sadeghi, C. Vishik, M. Winandy, " Trusted Privacy Domains - Challenges for Trusted Computing in Privacy-Protecting Information Sharing", 5th Information Security Practice and Experience Conference (ISPEC'09), 2009.
[23] S. Berger, et al., "TVDc: Managing Security in the Trusted Virtual Datacenter", Operating Systems Review, 42, 1, 2008, pp. 40-47
[24] W. E. Boebert, R. Y. Kain, "A practical alternative to Hierarchical Integrity Policies", 8th National Computer Security Conference, 1985