Design Technique for Secure Embedded Devices: Application for Creation of Integrated Cyber-Physical Security System

Vasily Desnitsky, Dmitry Levshun, Andrey Chechulin, and Igor Kotenko*
Laboratory of Computer Security Problems
St. Petersburg Institute for Informatics and Automation (SPIIRAS)
39, 14 Liniya, St. Petersburg, Russia
{desnitsky, levshun, chechulin, kotenko}@comsec.spb.ru

Abstract

As elements of complex information systems, embedded devices define informational and physical connections between the level of software control of the system on the one hand, and its technical environment and users on the other. Operating in a potentially volatile and untrusted cyber-physical environment, using insufficiently secure communication channels and sensors as well as various external influences cause such devices are subject to specific attacking actions. As a result the design of such systems is a challenging task often requiring expert based solutions. The main contribution of the paper is a design technique for secure embedded devices on the basis of combinations of security components, optimization approach and developed software tools for decision making support. The correctness of the technique is confirmed by its use in the development of the integrated cyber-physical security system.

Keywords: design of secure cyber-physical systems, embedded security, cyber-physical security, security components

1 Introduction

Currently there is a trend of rapid development of information systems with a variety of embedded devices and their integration into existing engineering infrastructures of manufacturing, electric power industry, medicine, transport management, objects of cyber-physical security i.e. rooms and so on. Besides its complicated informational component the business logic of such systems depends increasingly on elements of the physical layer such as sensors, actuators, single-board microcontrollers of a low energy consumption. The critical nature of such systems as well as high degree of interaction between embedded devices, other elements of hardware/software environment and users of the system lead to importance of developing mechanisms to protect such devices from information security threats.

We comprehend an embedded device as an electronic device with a limited set of highly specialized functions in the system, working as a rule at the junction of software and hardware levels of the system. Such devices are designed for, first, wireless and wired information exchange with both the user interface and objects of the technical environment (such as RFID scanners, dashboards, voice analyzers, etc.) and, second, information preprocessing of data from sensors, databases, application servers, clustered systems, etc.

The key features of embedded devices, having significant effect on the processes of the choice of individual components and their integration into the system, are the increased requirements to energy consumption of the devices, possibilities of their autonomous operation, increased requirements to productivity, physical characteristics and mobility of devices, to supported API requirements to drivers, compatibility and cost.

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 7:2 (June 2016), pp. 60-80
*Corresponding author: Tel: +7-812-328-7181

The presence of both pure software and hardware/software modules interacting with each other in an embedded device as well as possible variability of cyber-physical environment of embedded devices determine the susceptibility of such devices to specific sets of attacks on assets of the device.

The above mentioned features of embedded devices lead to the need to develop specialized approaches to the development of such devices, enabling improvement of the protection of final products and services [1].

The paper lies at the cross of computer security and embedded device engineering fields. It is focused on solving issues of secure system engineering in combining individual components of embedded devices into a unified consistently working software/hardware complex. This combination is carried out on the basis of established security requirements, limitations of the devices and connections between the components.

This paper is a logical continuation of the works previously published on the design and verification of systems with embedded devices. In [2] and [3] the general guidelines were proposed for combining security components for systems with embedded devices, taking into account metrics of their resource consumption, optimization approach for combinatorial enumeration of available alternatives of security components and the heuristic based approach to determine the importance of various resource consumption metrics.

This paper is the extended version of one published in proceedings of the 24th Euromicro International Conference on Parallel, Distributed and Network-based Processing [4]. The contribution of this paper is the developed technique describing actions a system designer needs to carry out to construct a device of effective implementation of the protection embedded. Additionally the paper provides a proof of concept for the technique by its practical application to the design of an integrated system of cyber-physical security, using programmable microcontrollers and sensors. Practically the technique allows evaluation of available alternatives of security components to select the required ones to increase efficiency metrics such as the energy consumption and cost of the combined protection.

The paper consists of the following sections. Section 2 contains a review of the related papers in the field. Section 3 reveals the essence of the proposed technique. Sections 4-6 encompass a proof of concept of the proposed technique: in section 4 we formulate the requirements to the integrated cyber-physical security system; section 5 contains a description of the process of selecting software/hardware means to implement in the system; section 6 provides a detailed description of the developed system. Section 7 presents an analysis of the obtained results. Conclusion finalizes the paper, highlighting the main outcomes.

2 Analysis of the relevant literature

An embedded device is a special purpose electronic computing device linking information system it operates in with elements of the physical environment of the system [5], [6] and [7]. Such devices are aimed at gathering and subsequent processing input data from various sensors, namely measuring modules, hardware interfaces, user interface and other embedded devices. Typically an embedded device involves also some responses to both informational events in the system and processes in the physical environment, involving a variety of output devices of textual and multimedia information, actuators, heating elements, etc. Particularities of an embedded device functioning in an interactive mode enabling interaction between hardware/software modules of the device, its technical environment, its users, local and remote application servers and databases significantly affects the processes of embedded device development.

Currently in its many aspects the development of embedded devices is based on the reuse of already tested hardware / software components, security algorithms, cryptographic primitives and protocols [8],

that is supported both at the level of standard electronic modules/hardware interfaces and by software components produced within specific platforms, such as Raspberry Pi, Arduino, Beagle board, Android, Java Embedded, etc. In fact an embedded device is organized as a set of interrelating pure software and software/hardware components, including ones providing some security requirements. We will collectively refer to embedded device components, not only pure security components (e.g. encryption module), but also the ones supporting the feasibility of the security requirements (e.g. Wi-Fi module) as security components.

When designing embedded devices the choice of specific security components is often carried out by experts due to their predetermined subjective preferences and the already known solutions. Usually in the process of combining these components into a single unit an insufficient focus is put on, first, individual characteristics of the security components and their implementations, second, possible side effects and hidden conflicts between the components due to a lack of their a priori coherence between themselves and the limits of the software / hardware platform.

In [9] for embedded devices operating at the cross of informational and physical levels it is explained the importance of trade-offs between security and non-functional characteristics of embedded devices, applying optimization approaches to combine security components.

In [10] Knezevic et al. explain the importance of research in design of secure embedded devices on the basis of the use of protection means characterized by a high provided security level and acceptable energy and computational costs. As for the required hardware and energy resources, attacks that could be classified to DoS and aimed at energy exhaustion of the device are especially interesting [11] and [12]. Such attacks are not detectable by common antiviruses and other conventional solutions, but cause uncontrolled resource waste by the most energy consuming hardware modules, such as Wi-Fi interface, GPS, Bluetooth and display. Thereby fulfillment of such attacks hinders further operation of the business functions of the device. Therefore, the embedded device should contain software and hardware modules against relevant security threats, considering the device limitations and side effects.

As a way of reaching a trade-off between the protection of a device and its energy consumption in [13] reconfigurable security primitives based on dynamic adaptation of the architecture of the device, depending on the device state and parameters of its environment are proposed. The proposed adaptation uses the ability to dynamically switching between multiple protection modules as well as the possibility of their updates.

In the existing literature there are many examples of building separate system of physical/cyber security through the use of embedded devices merged in a single network to enable their centralized management. For example [14] comprises an analysis and test results of a distributed access control system. In [15] information on possible consumers and the distinctive features of distributed systems for perimeter control is provided.

In these works, the architecture of the developed solutions is based on an expert based approach. In addition combination of physical protection and cyber one is not occurred practically in existing security systems. Therefore as a rule these subsystems operate independently from each other. According to this tendency complex attacks getting widespread recently will not be detected, because to protect against such attacks it is required to take into account information coming from multiple heterogeneous sources to both physical and cyber levels simultaneously. Therefore, to meet the ongoing challenges it is needed to implement a complex security approach. The essence of the approach lies in, first, unification of information from multiple heterogeneous sources of both physical and cyber levels and, second, consideration of a necessity to protect such system against attacks already at the system development stage becomes rather important.

The proposed technique forms a solution for the design of such a system without obligatory involvement of an expert in the information security field through the use of the developed software decision making tools. At that the tools are based on pieces of expert knowledge in analytical procession of

security requirements to the device and subsequent combination of security components.

3 Technique for design of secure embedded devices

In this section we reveal the most important stages of the developed design technique for secure embedded devices. The distinctive feature of the technique is consideration of functional and non-functional characteristics of security components and device limitations in the use of an optimization approach.

By the security configuration we mean a set of security components with defined functional and non-functional characteristics. The goal of the technique is to determine the most efficient set of security components (optimal configuration) on the basis of the device specification and possible security components. The efficiency here is regarded in a context of non-functional metrics, i.e. optimal energy consumption, costs, weight, etc.

The proposed design technique for secure embedded devices has 7 stages: (1) revelation of functional security requirements; (2) identifying the set of security component alternatives; (3) determination of essential non-functional constraints of the embedded device; (4) calculating values of non-functional metrics for the components; (5) determination of a selection criterion for the security components; (6) calculation of total values of non-functional security configurations; (7) combinatorial enumeration and comparison of security configurations.

Security components are software and hardware modules integrated in the embedded device and implementing some security requirements. A software module for symmetric encryption, such as AES/256, is an example of a software security component. An example of a security component is also a WPA supported Wi-Fi module aimed at providing a secure wireless communication channel and user authentication.

The technique proposed in this paper is aimed at automating the process of selection of efficient security configurations, using intruder models and evaluation of non-functional properties of security components. We assume checking the due security level and security component resistance is carried out within the preliminary stage of the technique and is beyond the scope of this paper. We assume the compatibility of the security components, first, at the level of the operating system/runtime environment and, second, in terms of hardware compatibility of specific hardware interfaces and electrical signal parameters are resolved within the pre-filtration of security components that is also not considered in the paper.

Within the scope of the technique we solve a problem of discrete optimization on a set of configurations with a target function expressed using non-functional parameters and constraints on both functional and non-functional parameters [3]. The target function of the optimization problem represents several non-functional metrics either minimized or maximized, depending on the specific semantics of non-functional metric (p1 $\rightarrow$ min/max, p2 $\rightarrow$ min/max, p3 $\rightarrow$ min/max,...). The choice of the most preferable metrics can be done by means of a heuristic based approach.

Stage 1 of the technique includes the revelation of functional security requirements that should be implemented in the process of development of the combined security mechanism. These requirements are derived based on the analysis of the target device specifications, using an approach to analytical modeling [16] of the embedded device intruders with the use of known intruder models [17] and [18]. As an example, we can specify the following functional security requirement – secrecy of data of the device must be carried out using symmetric encryption with at least 128 bits key length.

Briefly consider the approach. By applying two known intruder classifications proposed by Rae'2003 and Abraham'1991 respectively there are five possible types of intruder access to the embedded device (ED) as well as three possible levels of intruder's capabilities. These two classifications are combined to introduce a matrix of intruder categories, 21 categories in all. In essence each its cell contains a

pair, namely particular intruder type and its level. After a list of functional device features is extracted from the system specifications. These ones can be any business features like a presence of Wi-Fi or any specific sensor, presence of traffic encryption or some specific business procedure and so on.

Then the set of all intruder categories is restricted by selecting only those categories that correspond to the obtained list of the functional device features. For example, if a device has no Internet connection, this means there won't be Type1 intruders, since according to Rae classification Type1 intruder exploits TCP/IP based attacks only [16]. The corresponding formal reasoning rule used within the approach is as follows:

if not(has_Internet_connection(device)) $\rightarrow$ *no type1 intruder.*

So this means, according to conditions of a concrete ED, the approach enables significant restriction of the set of possible intruder categories to consider in the device design process. Basically rules underling the approach have the following form:

(functional_feature, intruder_category) $\rightarrow$ *specific_threats.*

Each rule consists of two parts. The left part contains a particular functional device feature and a few relevant intruder categories, whereas the right part represents one or a number of security threats this functional feature is subject to by these intruders. Another example of rules to deduce relevant threats and subsequent functional security requirements are presented below in a more fully manner:

(functional_feature = "traffic encryption"), ((type1,level3) or (type2,level3) or (type3,level3) or (type4,level3)) $\rightarrow$ *threat = "cryptographic analysis of encrypted messages"*

if low financial value of device business data $\rightarrow$ *no level3 intruders (no reason to protect against level3 intruders)*

if (communication_interface="Wi-Fi", intruder of level1/level2 only)$\rightarrow$ *"crypto analysis attack" is practically impossible.*

So, when considering the Traffic encryption feature, due to the anticipated low cost of data, the protection from the intruder of level 3 (professional team of offenders with access to laboratory equipment) would be useless, whereas level 1 and level 2 intruders are not able to perform the cryptanalysis based attack.

As a part of the underling expert knowledge rules, the most typical ED functional features have been already specified and realized within our implemented Static Testing Tool automating the procedure of deducing needed functional security requirements for embedded devices (Figure 1).

By means of this tool the developer can set the intruder categories due ED functional features, and run the process of deduction of security threats the ED is subject to.

At stage 2 of the technique for each functional security requirements the identification of a number of alternative security components that implement it is carried out. For example for a requirement of business data integrity a set of cryptographic hash functions with collision resistance not less than a specific one, such as SHA-512, MD5, MD6, Whirlpool, etc. are defined.

Stage 3 includes steps for determining non-functional constraints that are essential to the designed device. Source of knowledge for determining non-functional constraints is MARTE methodology (Modeling and Analysis of Real-Time Embedded Systems) [8], where non-functional features of embedded devices are specified by UML. In particular, the methodology uses non-functional constraints based on the following knowledge domain classes: HW_Physical, HW_PowerSupply, HW_StorageManager, HW_Computing and HW_Communication [8].

Generally such constraint is linear and has the following form $p \leq A$, where p specifies some non-functional metric, A is the maximum value of the specified hardware resource to be consumed by security components . For example, $p_{ec} \leq 1200$ sets a limit to 1200 mAh on electricity consumption p_{ec}, allocated to the functioning of the device security components.

At stage 4 we determine values of non-functional constraints of security components, which are determined in the following two ways: analytically – by collecting data from manufacturers of the used

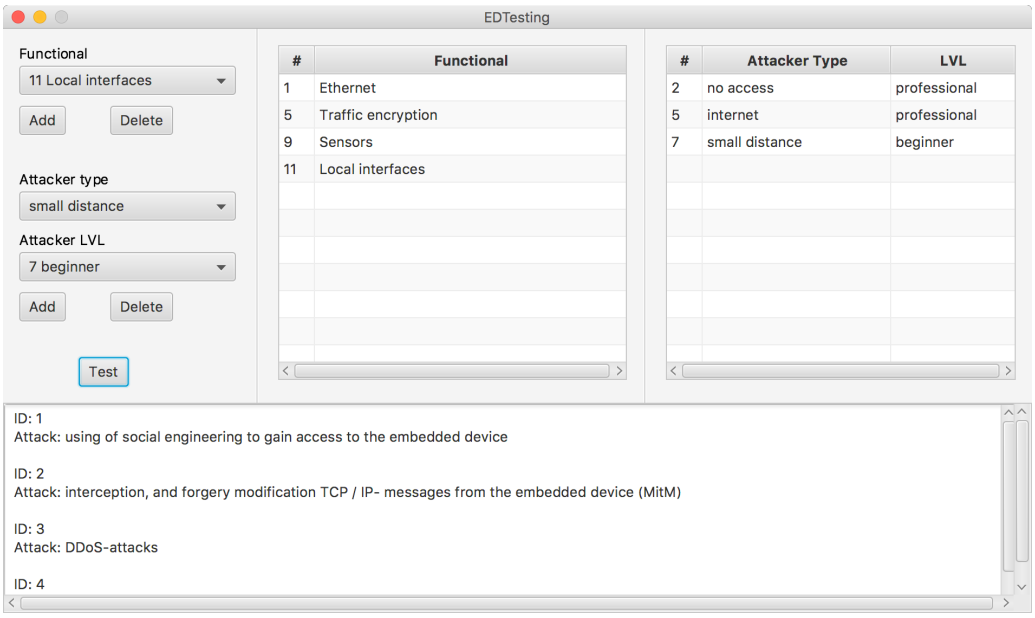

Figure 1: User interface of Static Testing tool

hardware and software modules; empirically – basing on software modeling of security components (when possible). For example, the metric of cost of hardware components, their physical dimensions or the peak energy consumption are determined by the former method. The latter is applicable for example for assessing the amount of RAM required by a component for encryption/decryption of data passed via a secure communication channel or assessing the value of the communication channel bandwidth to enable remote attestation component functioning. If the use of the analytical or empirical approaches for some non-functional constraints is complicated, its numerical value can be established by an expert, basing on past experience in working with the same or similar components.

At stage 5 we set a selection criterion to choose a security component among a set of available ones. In fact, at this stage we divide the most significant and minor limitations, depending on the relative importance of each of them with the heuristic proposed in [3]. The heuristic specifies a general algorithm to prioritize the non-functional constraints of an embedded device. Essentially for each non-functional constraint a set of specific functional and non-functional characteristics of embedded devices is singled out, for instance presence of permanent power source, possibility of replacing the device or battery without harming the services it provides, dependency of achievement between business objectives of the device and its energy, etc. Each characteristic is ranked by three numerical values where 1 means low importance and 3 means high importance. As a result the specification of the target embedded device is analyzed for the presence of the features. For each non-functional constraint the maximum rank value is determined on all features found at the device. After that the non-functional constraints are ordered by decreasing these ranks. If some constraints got the same rank values they are ordered among themselves according to an order predefined by an expert [3] by default.

In case of a high criticality of several non-functional constraints a convex hull on the set of security configurations is constructed. Then the optimal solution is determined as a one lying on the boundary of this convex hull.

At stage 6 we compute the total numerical non-functional values of possible security configurations, and then at **stage 7** the configurations are enumerated and compared to each other according to the established selection criterion, and thus the optimal solution is determined. If a large number of consid-

ered functional security requirements or large amount of available alternatives of security components it is reasonable to apply a developed software tool Configurator [2] allowing automating the process of enumeration and computation.

If a solution of the optimization problem does not exist, recommendations to revise the resource constraints can be offered at stage 7. We should note that in the general case the choice of security components is an iterative process. If any change in the system specifications or its implementation the resulting set of components may change.

4 Requirements for the developed integrated system for cyber-physical security

To demonstrate the efficiency of the proposed technique and the developed software a task of building an integrated system of cyber-physical security was selected. The goal of this system is to combine systems of physical and cyber security to reduce risks associated with complex attacks stretched over time and happening on both the physical and cyber levels. Risk reduction is achieved through the use of correlation of data coming from heterogeneous sources, automatically generating incidents and determining attack scenarios, which can be revealed in contemporary security systems at the investigation stage only.

For example the physical security systems include the following ones – access control system, closed-circuit television system, security alarm system, fire alarm system, automatic fire extinguishing system, warning and evacuation management system, engineering equipment monitoring and control system, light control system, climate control system, automatic telephone system and others. Cyber security systems include the following ones – unauthorized access control system, antivirus, operating system, data leak prevention system, intrusion detection system, intrusion prevention system, network firewall, backup system, email gateway and others.

Connecting to systems of physical and cyber security is accomplished through hardware and software interfaces. The task of hardware and software interfaces is data acquisition from heterogeneous sources and their subsequent transfer to the server of the integrated security system. Hardware interfaces connect to the system of physical security at the level of detectors, alarms, sensors and external electronic components by means of special adapters. Each adapter corresponds to a wired or wireless transmission interface used in the system of physical security. Data collected by the hardware interfaces are transmitted via a secure channel to hubs aimed at reducing the load on the server of integrated cyber-physical security system through preprocessing and compilation of incoming data. Software interfaces are connected to the systems of cyber security as well as to physical security systems at the level of the central server, when it exists, by means of special drivers. The drivers can use both the already existing application programming interfaces and specially designed agents to collect the necessary log entries.

Server of the integrated cyber-physical security system performs the normalization and preprocessing the data collected through hardware and software interfaces prior to the entry into the storage. Then the data are subjected by a process of correlation to detect security incidents, scenarios of attacks and anomalous activity. On the basis of the obtained results the reports are generated, countermeasures are developed and security assessment of the organization is implemented. In addition the results of the work of integrated security system can be used as input data to information and security events management systems. Integration with systems of this type is necessary for more detailed and high-level processing of detected security incidents, scenarios, attacks and anomalous activity.

Thus, to implement the integrated cyber-physical security system it is necessary to implement reliable and secure hubs and hardware interfaces as embedded devices of different functionality. Hardware interfaces are located in the immediate vicinity of the detectors, alarms, sensors and external electronic

components of physical security systems. Hubs are located in close proximity to the central microcontrollers of these systems.

Let us consider the basic functional requirements for the hardware interfaces in more detail.

Hardware interfaces should work in a local network via a wired data transfer channel. Wired solutions provide higher reliability for signal transmission, however, they require significantly more time and money for assembly and installation. Note that for protection against interception, modification and substitution the data transmission between the concentrator and the hardware interfaces should be pre-encrypted.

Additionally the developed hardware interfaces should support emergency operation mode when the data exchange between the concentrator and the hardware interfaces is impossible (due to a lack of connection with the hub, denial of service on the side of the hub, etc.). In the emergency operation mode information about events connected to the work of physical security systems are written to the local log. Thus, after restoring the data exchange between the hubs and the hardware interfaces, no data are lost. We should note that the developed hardware interfaces should have sufficient memory to store and support the local log. Thus, the functional requirements for hardware interfaces can be summarized in an overall view (table 1).

Functional requirements	**N**	**Description**
To hardware	1	Providing of interaction with detectors, alarms and sensors by both wired and wireless connections, as well as external electronic components of physical security systems: readers, output devices for text and audio information, sound and light signals
	2	Providing interaction with external electronic components: output devices of sound and light signals
	3	Providing transfer of data through a structured cabling system
	4	Providing the possibility of functioning in the absence of power supply for a period not exceeding 24 hours
To software	5	Providing secure storage for the local log
	6	Ensurance of transmitted data encryption

Table 1: Functional requirements to hardware interfaces

Let us consider the basic functional requirements to hubs in more detail.

The hubs need to work in a local network over a wireless data transfer channel. Wireless solution will allow protecting the system from physical attacks on the data transmission channel between the hub and the server of the integrated cyber-physical security system (such as a damage of the Ethernet cable), and to substantially reduce the cost of system installation.

Also, hubs must support the interface for interaction with the remote application server for easy integration into the integrated cyber-physical security system. Interaction with the application server may be performed over HTTP, HTTPS and SOAP protocols. Data transmitted via HTTP and SOAP protocols should be pre-encrypted. In addition, to communicate with the server of the integrated security system, hubs must be capable of running applications developed in a high level programming languages.

Similarly to the hardware interfaces, developed hubs should support emergency operation mode, for cases when data exchange between hubs and the server of the integrated cyber-physical security system is impossible (no server connection, denial of service on the server side, etc.). In the emergency operation mode the data received from the hardware interfaces are written to the local log of the hub. Thus restoring the data exchange between hub and server of the integrated security system, no data are lost. The hubs should have sufficient memory to store and support the local log.

At the local Ethernet connection the hubs should provide web interface for configuration and management, which contain data of the internal log of the hub. Thus, the functional requirements to developed hubs can be summarized in an overall view (table 2).

Functional requirements	**N**	**Description**
To hardware	1	Providing interaction with external electronic components: output device of a text, audio information, sound and light signals
	2	Providing wireless data transmission channel
	3	Providing data transmission via Ethernet
	4	Providing a possibility of functioning in the absence of power supply for a period not exceeding 24 hours
To software	5	Providing data exchange via HTTP, HTTPS, SOAP
	6	Providing launching of applications written in Java, Python and C++
	7	Providing secure storage for the local log
	8	Providing encryption of data transmitted over the data channels
	9	Providing configuration and management of hubs through the web interface at a local Ethernet connection

Table 2: Functional requirements for developed hubs

In case of a failure of an electric circuit the hub or hardware interfaces are connected to the energy supply provided by the reserve power source. In such situation the functioning of the devices is not broken, but the maximum time of its functioning depends on the capacitance of the source and quantity of energy they consume. The cost of hubs and hardware interfaces is calculated in a complex way: it takes into account the microcontroller with the entire set of components. Thus, the preference should be given to the alternative that satisfies all functional requirements and at the same time has the optimal ratio final cost / energy efficiency.

Taking into account possible location of the devices, hubs and hardware interfaces should have sizes, allowing placing them in a small space. Thus, the hardware interfaces should be no thicker than 3 cm, for their placement in the close proximity of detectors, alarms, sensors and external electronic components of physical security systems. A hub should be no thicker than 5 cm, for its placement in close proximity to the central microcontroller of these systems, respectively. Note that most of popular microcontrollers have thickness of less than 3 cm. Thus, non-functional requirements can be summarized in the following table (table 3).

Non-functional requirements	**Description**
To power efficiency	Providing functioning of hubs and hardware interfaces in conditions of failure of the electrical circuits which they are connected to due to energy of the reserve power supply
To the cost	Minimizing the cost of the microcontroller, the microcontroller extensions, adapters and peripherals needed to meet functional requirements
To space occupied	Minimization of the size of the microcontroller, the microcontroller extensions, adapters and peripherals, so that the thickness of the hardware interfaces does not exceed 3 cm, and the thickness of the hub does not exceed 5 cm

Table 3: Non-functional requirements for developing hubs and hardware interfaces

External electronic component	Selected physical device	Consumption (milliampere-hour, mAh)	Cost (euro)
The output device for audio signals	DC 12mA 5V 12mm Piezo Alarm Buzzer	12 [20] (at the moment of signal sending)	3
The output device for light signals	RGB Light-emitting Diode	20 [21]	0.1
Total		20 – 32 (average – 22)	3.1

Table 4: The set of external electronic components of hardware interfaces

5 Application of the technique

It should be noticed that the costs of microcontrollers and their additional elements are formed on the basis of official proposals of developers and their distributors. Cheaper alternatives (replicas) are not considered, since it is impossible to guarantee their compatibility with adapters and the common set of external components. The cost of non-specific components is formed basing on proposals of Amazon e-store [19].

5.1 Application of the technique to select components of hardware interfaces

In accordance with the functional requirements to hardware interfaces (table 1) each of the considered alternatives should support interaction with external electronic components, namely output devices for sound and light signals. The set of external electronic components of hardware interfaces with their prices and energy consumption metrics are presented in an unified table (table 4).

Also, in accordance with the functional requirements to hardware interfaces (table 1) each of the considered alternatives should support detectors, alarms and sensors in both wired and wireless versions as well as external electronic components of physical security systems, namely readers, the output devices for text and audio information, as well as for sound and light signals. In compliance with section 4 hardware interfaces are connected to the physical security systems through special adapters, each corresponding to a wired or wireless data interface.

In existing systems of physical security at the level of the adapter connection, mainly the following data interfaces are used – RS-232, RS-485, InfraRed, BlueTooth, NFC, RFID, ZigBee. Depending on the selected microcontroller the support for the most data interfaces for data transfer may be implemented either by default or require acquisition of additional components. Adapters of hardware interfaces to support the most data interfaces of physical security systems, their prices, energy consumption and platform of the microcontrollers are presented in (see table 5).

We assume that the set of external electronic components of hardware interfaces will be entire, so in the design technique (see section 2) there will be no alternatives for each element in the set of external electronic components. The effect of the set (see table 4) of the external electronic components on non-functional requirements (see table 3) will be taken into account when making optimal solution. Also, we assume that the impact of the adapter energy consumption and cost on the non-functional requirements to hardware interfaces will be taken into account when making optimal decisions in the form of averaged value. Given the functional requirements for the hardware interfaces (see table 1) the alternatives presented in table 6 have been formed as the technique output.

In the process of the design technique implementation the following devices have not been added to the output set of alternatives. First, microcontroller Arduino Uno due to its flash memory size of 32 KB, which does not allow adequately supporting local logs in the emergency mode of the device

Data interface	Platform	Selected physical device	Consumption (milliampere-hour,mAh)	Cost (euro)
RS-232	Arduino	RS-232 Shield	70 [22]	18.11
	Raspberry Pi	Serial Pi Plus	120 [23]	13.49
RS-485	Arduino	RS-485 Shield	100 [22]	19.06
	Raspberry Pi	RS-485 Shield V3	150 [23]	24.56
InfraRed	Arduino	Infrared Shield	250 [22]	11.24
	Raspberry Pi	Infrared Shield	260 [23]	11.49
BlueTooth	Arduino	Bluetooth Shield or by default	200 [22]	16.25
	Raspberry Pi	BT-Micro3H2X or by default	150 [23]	30.00
NFC, RFID	Arduino	NFC RFID Shield	120 [22]	23.74
	Raspberry Pi			
ZigBee	Arduino	XBee Shield	200 [22]	17.83
	Raspberry Pi			
The average result for the platform Arduino			157	17.81
The average result for the platform Raspberry Pi			167	20.35

Table 5: Adapters of hardware interfaces

N	Components	Description
1	Arduino Zero	Microcontroller ATSAMD21G18, 32-Bit ARM Cortex M0+, Operating Voltage 3.3V, Digital Pins 20, UART2 (Native and Programming), Analog Input Pins 7, DC Current per I/O Pin 7 mA, Flash Memory 256 KB, SRAM 32 KB
2	Arduino Mega	Microcontroller ATmega2560, Operating Voltage 5V, Digital Pins 54, Analog Pins 16, DC Current per Pin 20 mA, Flash Memory 256 KB, SRAM 8 KB
3	Raspberry Pi A+, microSD 512 MB	Processor Chipset Broadcom BCM2835 32Bit, GPU Videocore IV, Processor Speed Single Core @700 MHz, 256 MB SDRAM @ 400 MHz, MicroSD, 1x USB Ports, Max Power Draw/voltage 1.8A @ 5V, GPIO 40 pin
4	Raspberry Pi B+, microSD 512 MB	Processor Chipset Broadcom BCM2835 32Bit, GPU Videocore IV, Processor Speed Single Core @700 MHz, 512 MB SDRAM @ 400 MHz, MicroSD, 4x USB Ports, Max Power Draw/voltage 1.8A @ 5V, GPIO 40 pin

Table 6: Alternatives of hardware interfaces

operation. Second, microcontrollers Arduino Nano, Arduino Micro and Arduino Mini because of the lack of adapters support (see table 5). In addition, Raspberry Pi 2 and 3 as well as Arduino Yun were not included in the list of microcontrollers since the Raspberry Pi 2 and 3 are more expensive functional analogs of Raspberry Pi A+/B+ (similarly to Arduino Yun and Arduino Zero/Mega), and therefore they are worse because of non-functional requirements.

When analyzing the energy efficiency of individual components and devices both sources from the Internet were used and the experiments with real equipment were carried out.

An energy consumption of **alternative 1** includes the energy consumption of Arduino Zero, the adapter and the set of external electronic components. An energy consumption of the microcontroller Arduino Zero is 20 mAh [24]. Thus the energy consumption of alternative 1 is 199 mAh. An energy

consumption of **alternative 2** includes the energy consumption of Arduino Mega, the adapter and the set of external electronic components. An energy consumption of a microcontroller Arduino Mega is 200 mAh [22]. Thus the energy consumption of alternative 2 is 379 mAh. An energy consumption of **alternative 3** includes the energy consumption of the Raspberry Pi A+, the adapter and the set of external electronic components. Minimum energy consumption of Raspberry Pi A+ without connected external devices is 500 mAh [22]. Thus the energy consumption of alternative 3 is 689 mAh. An energy consumption of **alternative 4** includes the energy consumption of Raspberry Pi B+, the adapter and the set of external electronic components. The minimal energy consumption of Raspberry Pi B+ without connected external devices is 700 mAh [22]. Thus, the energy consumption of alternative 4 is 889 mAh.

The cost of external components, including output devices for sound and light signals is 3.1 euro (table 4).

Let us consider features of the proposed alternatives affecting the overall price.

The cost of **alternative 1** includes Arduino microcontroller Zero 42.90 euro [25], adapter 17.81 euro (average price, see table 5). The final cost taking into account the external electronic components is 63.81 euro. The cost of **alternative 2** includes Arduino Mega microcontroller 35 euro [25], adapter 17.81 euro (average price, see table 5). The final cost taking into account the external electronic components is 55.91 euro. The cost of **alternative 3** includes microcontroller Raspberry PiA+ 19.38 euro [26], Micro SD card 2 euro [20], adapter 20.35 euro (average price, see table 5). The final cost taking into account the external electronic components 44.83 euro. The cost of **alternative 4** includes microcontroller Raspberry Pi B+ 23.00 euro [26], Micro SD card 2 euro [20], adapter 20.35 euro (average price, see table 5). The final cost taking into account the external electronic components is 48.45 euro.

The sizes of hardware interfaces directly depend on the size of their largest part, namely their microcontrollers. The thickness of all microcontrollers meets a limit of 3 cm (to accommodate the close vicinity of the detectors, alarms, sensors and external electronic components of physical security systems). For non-functional requirements the data obtained is presented in table 5.1.

The set of security components	Energy consumption(mAh)	Cost (euro)	Size (mm)
Arduino Zero	199	63.81	69*53*8
Arduino Mega	379	55.91	102*53*8
Raspberry Pi A+	689	44.83	86*53*7
Raspberry Pi B+	889	48.45	60*36*7

Table 7: Comparison of alternatives of hardware interfaces

Alternative 1 shows the best power efficiency with the greatest cost. Alternative 2 has the power efficiency below the average, despite the fact that its cost is higher than the average values of the given analogs. Alternative 3 has power efficiency below the average and minimal cost. Alternative 4 has the minimal energy efficiency and the cost higher than alternative 3. Thus further consideration of alternative 4 is not reasonable.

The requirement of power efficiency means that hardware interfaces should support operation even in case of a failure in the electricity net they are connected to. This one is achieved by means of a reserve power source. An alternative can work with the help of the reserve power source, depending directly on the capacity of the reserve power source. For 24 hours operation batteries with the following capacities are required, alternative 1 requires 5000 mAh, alternative 2 demands 10000 mAh, alternative 3 requires 17000 mAh. The required capacity are calculated on the base of previously obtained values of average energy consumption.

As a source of the reserve energy Power Bank is considered, as its size meets the non-functional requirements. The cost of the Power Bank depends on the capacity and number of supported recharge

cycles and is equal to: 8, 14, and 31 euro for capacities of 5000, 10000 and 17000 mAh respectively. Thus, the total cost of alternative 1 is 71.81 euro, for alternative 2 it is 69,91 euro and for alternative 3 it is 75.83 euro (including the cost of the reserve power source).

Therefore considering the cost metric the optimal set of components is alternative 2. The final set of components of hardware interfaces of the integrated cyber-physical security system is Arduino Mega, adapter (see tab. 6), 12mA DC 5V 12mm Piezo Alarm Buzzer, RGB Light-emitting Diode, power bank 10000 mAh. The average cost is 69,91 euro and average energy consumption is 379 mAh.

5.2 The application of the technique to select hub components

In accordance with the functional requirements to the hubs (table 2), each of the considered alternatives should support interaction with external electronic components, namely output devices of text and audio information, sound and light signals. The set of external electronic components of the hubs, their prices and energy consumption are presented in table 8.

External electronic component	**Selected physical device**	**Consumption (milliampere-hour, mAh)**	**Cost (euro)**
Device for output of text information	DC 5V Character LCD 16x2	100 [27]	5
Device for output of audio signals	DC 12mA 5V 12mm Piezo Alarm Buzzer	12 [20] (at the time of the alarm)	3
Device for output of light signals	RGB Light-emitting Diode	20 [21]	0.10
Total		120 – 132 (average – 122)	8.10

Table 8: The set of external electronic components of hubs

We assume that the set of external electronic components of hubs will be entire, so the design technique (see section 2) will not include a search for alternatives for each element in the set of external electronic components. The effect of the set (see 8) of external electronic components on non-functional requirements (see table 3) will be taken into account when finding optimal solution. By considering functional requirements alternatives presented in table 9 are formed as the technique output.

In the process of implementation of the design technique the following devices are not added to the output set of alternatives, first, microcontroller Arduino Mega because of the lack of support for running applications written in Java, Python, C++, second, Raspberry Pi A+ because of the lack of support for data transmission via Ethernet. In addition Raspberry Pi 2 and Raspberry Pi B+ are not included in the list of microcontrollers due to the lack of support for the radio channel data transfer in contrast to the Raspberry Pi 3.

When analyzing the power efficiency of individual components and devices both sources from the Internet were used and experiments with real equipment were carried out.

An energy consumption of **alternative 1** includes the energy consumption of the Arduino Yun and the set of external electronic components. An energy consumption of the Arduino Yun microcontroller in its turn depends on CPU load and can vary between 200 and 250 mAh [19]. Thus the energy consumption of alternative 1 is 322-372 mAh. An energy consumption of **alternative 2** includes the energy consumption of Raspberry Pi 3 and the set of external electronic components. Minimum energy consumption Raspberry Pi 3 without connected external devices is 700 mAh [22]. Thus the consumption of alternative 2 is 824 mAh. An energy consumption of **alternative 3** includes the energy consumption of Beaglebone Black, Compact USB Wi-Fi Adapter and the set the external electronic components. An en-

N	Components	Description
1	Arduino Yun, microSD 512 MB	The internal memory of the Arduino Yun is limited to 8MB, which is insufficient to meet the functional requirements of the software. The internal memory of the Arduino Yun can be expanded using a microSD
2	Raspberry Pi 3, microSD 512 MB	Microcontrollers of the Raspberry Pi platform by default use memory cards instead of the internal memory
3	Beaglebone Black, Compact USB Wi-Fi Adapter	Beaglebone Black does not support data transfer over a wireless data channel, and therefore does not meet one of the functional requirements to the hardware. It is used a Compact USB Wi-Fi adapter designed specifically for the Beaglebone Black, to neutralize this disadvantage
4	Intel Galileo Gen 2P Board, Intel Galileo Wi-Fi Kit	Intel Galileo Gen 2P Board does not support data transfer over a wireless data channel, and therefore does not meet one of the functional requirements to the hardware. Intel GalileoWi-Fi Kit is designed specifically for IntelGalileoGen2PBoardto neutralize this disadvantage

Table 9: Alternatives of the hubs selected in the design technique for secure embedded devices

ergy consumption of Beaglebone Black is 210-460 mAh [28]. An energy consumption of Compact USB Wi-Fi Adapter is 120 mAh (the value is obtained experimentally). Thus the energy consumption of alternative 3 is 452-702 mAh. An energy consumption of **alternative 4** includes the energy consumption of Intel Galileo Gen 2P Board, Intel Galileo Wi-Fi Kit and the set of external electronic components. Minimum energy consumption of Intel Galileo Gen 2P Board without connected external devices is 800 mAh [29]. Consumption of Intel Galileo Wi-Fi Kit is about 400 mAh [20]. Thus the energy consumption of alternative 4 is 1322 mAh.

The cost of external components, including the output device for the textual information, sound and light signals is 8.10 euro (table 8).

Let us consider proposed alternatives peculiarities affecting the overall price.

The cost of **alternative 1** includes Arduino Yun 52 euro [18] and Micro SD card 2 euro [20]. The final cost taking into account the external electronic components is 62.10 euro. The cost of **alternative 2** includes Raspberry Pi 3 34.32 euro [25] and Micro SD card 2 euro [20]. The final cost taking into account the external electronic components is 44.42 euro. The cost of **alternative 3** includes Beaglebone Black 50 euro [26] and Compact USB Wi-Fi Adapter 11 euro [20]. The final cost taking into account the external electronic components is 69.10 euro. The cost of **alternative 4** includes Intel Galileo Gen Board 2P 62 euro [30] and Intel Galileo Wi-Fi Kit 45 euro [20]. The final cost taking into account the external electronic components is 115.10 euro.

The dimensions of hubs directly depend on the size of their largest part, namely their microcontrollers. The thickness of all of the MCU meets a limit of 5 cm (for placement in close proximity to the central microcontroller of physical security systems). On the basis of non-functional requirements the results presented in table 10 were obtained.

Alternative 1 shows the best power efficiency at a cost below the average. Alternative 2 has a power efficiency below the average, however, it has the cost considerably below than analogs, that gives reasons for considering this option. Alternative 3 has power efficiency and cost being worse than alternative 1. Thus, alternative 3 is not being considered further. Alternative 4 is the most expensive and least energy efficient solution among the rest options, so its further consideration is not reasonable.

The requirement of power efficiency assumes that the hubs should support operation even in case of a failure in the electricity net they are connected to. This one is achieved by means of a reserve power source. For a particular alternative its working time depends on the volume of the reserve power source.

The set of security components	Energy consumption(mAh)	Cost (euro)	Size (mm)
Arduino Yun, microSD 512 MB	350	62.10	73*53*8
Raspberry Pi 3, microSD 512 MB	824	44.42	60*36*7
Beaglebone Black, Compact USB Wi-Fi Adapter	620	69.10	86*53*7
Intel Galileo Gen 2P Board, Intel Galileo Wi-Fi Kit	1322	115.10	123*72*9

Table 10: Comparison of alternatives of hubs

For 24 hours operation batteries with the following capacities are required, alternative 1 requires 9000 mAh, alternative 2 demands 20000 mAh. The required capacity was calculated on the base of previously obtained values of the average energy consumption.

We considered Power Banks as the sources of the reserve power, because their size meets the non-functional requirements. The cost of the Power Banks depends on the capacity and number of supported recharge cycles and is equal to 14 and 30 euro for 10000 mAh and 20000 mAh capacities respectively. Thus, by considering the cost of the reserve power source the total costs of alternative 1 and alternative 2 are 76.10 euro and 74.42 euro respectively.

Therefore alternative 2 represents the optimal set of components. The final set of components for the hub of the integrated cyber-physical security system is as following – Raspberry Pi 3, 512 MB microSD, DC 5V Character LCD 16x2, 12mA DC 5V 12mm Piezo Alarm Buzzer, RGB Light-emitting Diode, Power Bank 20000 mAh, having the cost of 74.42 euro and energy consumption of 824 mAh.

6 Description of the designed integrated cyber-physical security system

The architecture of the designed integrated cyber-physical security system is shown in figure 2:

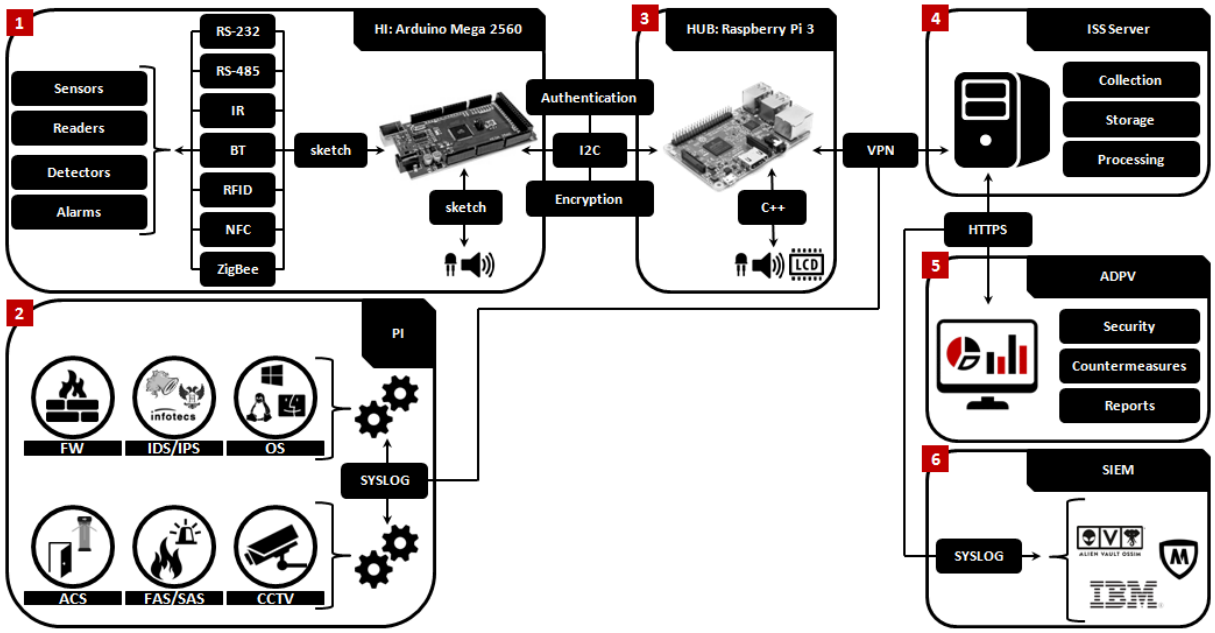

Figure 2: The architecture of the integrated cyber-physical security system

Let us consider the main modules of developed integrated cyber-physical security system, taking into account the requirements to embedded devices (table 1 and 2).

Module 1. Hardware interfaces (HI) – the microcontroller Arduino Mega 2560 (ATmega2560 processor), managing in accordance with the firmware [31] the components of the output light and sound signals (it fulfills requirement 2 of table 1), as well as adapters that implement wired and wireless data interfaces (it fulfills requirement 1 from table 1). This microcontroller provides the real-time events logs of physical security systems (it fulfills requirement 5 of table 1), and transmits information about the occurred events to the hubs (it fulfills requirement 3 of table 1), pre-encrypting the transmitted data and performing mutual authentication procedure (it fulfills the requirement 6 of table 1). This embedded device should be supplemented with an accumulator allowing its operation in case of a short (less than 24 hours) energy absence (fulfillment of the requirement 4 from table 1).

Module 2. Program interfaces (PI) connect to the cyber physical security system through special drivers. The drivers can use both already existing application programming interfaces and specially designed agents to collect the necessary log records.

Module 3. Hubs – by microcontroller Raspberry Pi 3 (BCM2837 processor) that supports code written in C++ (it fulfills requirement 6 of table 2) and controls the output devices of a text and audio information, sound and light signals (it fulfills the requirement 1 from table 2). Event data that are passed by the hardware interfaces are written to the database (fulfillment of requirement 7 of table 2) are subjected to processes of normalization and pre-procession, after that they are forwarded to the server of the integrated cyber-physical security system via VPN or HTTPS protocols (meets requirements 5 and 8 of table 2). This embedded device should be supplemented with a battery allowing its operation in case of short (less than 24 hours) energy absence (fulfillment of requirement 4 from table 2).

Module 4. The server of the integrated cyber-physical security system (ISS Server) comprises collection module (it receives data from the hub), storage (it records information on events of cyber physical security system into the entire database) and processing module (it correlates entry logs from the storage to automatic detection of incidents, attack scenarios and anomalous activity). The results of the processing module are sent to the dashboard via HTTPS protocol. The application server of the integrated cyber-physical security system (4) is represented by the servlets catalog on Tomcat [32] the war application is deployed on. War application is developed within the Spring framework [33], which already has easy to use solutions in the domain of access to a database server (package DAO Support) and implementation of the basic security requirements (package Spring Security). An intermediary between the war application and the database connection pool is C3P0 [34], which is based on JDBC [35]. C3P0 provides great connection flexibility, namely it allows allocating database connections for different users with different rights. C3P0 implements mechanisms to control the load also.

Module 5. The module for analytical data processing and visualization (ADPV) displays detected incidents, scenarios of attacks and anomalous activity. It also produces assessment of security of the organization, develops countermeasures, and generates reports.

Module 6. The information and security events management system (SIEM) – the results of the work of the processing module of the server are also sent in the form of SYSLOG messages via HTTPS protocol to the information and security events management system to, first, provide the high level representation of the detected incidents, attack scenarios and anomalous activity and, second, determine the scale of the attack and possible damage.

Let us consider the used software and communication protocols in more detail.

The data about events that are detected by the physical part of the security system are transferred between hardware interfaces (1) and hubs (3) by I2C protocol. In this connection the Raspberry Pi 3 microcontroller acts as a master microcontroller of the serial data bus, whereas the microcontrollers Arduino Mega 2560 connected to it as slaves. The permissible number of slave microcontrollers is limited to 128.

The interaction between hubs and the server of the integrated cyber-physical security system is done through the virtual private network (VPN). The connection to the local network is performed by wire or wireless link (fulfillment of requirements 2 and 3 from table 2). OpenVPN is used as a VPN client, a freeware implementation of the technology under the GNU GPL.

The data received by the server from the hubs and software interfaces are written into database management system PostgreSQL in SYSLOG format. In addition a special web page is implemented on the hubs and allows us to configure the hub, view the local logs of the device as well as initiate data transfer to the server of the integrated cyber-physical security system. To get an access to the web page you need a direct connection to the microcontroller via an Ethernet cable as well as performing the authentication procedure.

The interaction of the system's components is carried out within the network connection with the use of encrypted messages, because messages are much more reliable and easy to regulate, support asynchronous communication and provide easy integration. Messages are formed in the form of GET requests as this structure is robust and allows easy integration of components.

Thus the embedded devices developed by the proposed technique meet all the functional requirements, balancing the non-functional constraints.

7 Discussion

The design technique for secure embedded devices yields optimal security configuration which meets functional requirements and non-functional ones. The set of alternatives of security components is specified in the special database that also contains information about compatibility (the presence or absence of possible conflicts) between the security components. The optimal solution obtained on the basis of the technique directly depends on the contents of the special database as well as on the specified functional and non-functional requirements. Thus the quality of the set provided by the technique depends on the completeness and relevance of the special database and the correctness of the specified requirements. Therefore in general a case the design technique for secure embedded devices is not a complete replacement for expert decisions.

The efficiency of the proposed technique is confirmed by its practical application to creation of the integrated cyber-physical security system. Figure 3 schematically shows a choice among four alternatives of hardware interfaces (see table 5.1), taking into account non-functional characteristics of the reserve power supplies of these alternatives.

So, the reasonable choice of the required solution, depending on the metrics of energy consumption and cost is carried out by building a convex hull for the set of possible solutions. After that configurations falling on the border (alternatives 1 and 2) are accepted as optimal.

An expert in the field of security components for embedded devices, having knowledge on specialized solutions may select a sufficiently effective set of security components with a high probability. Nevertheless this technique can yield significant benefit when there is a vast number of functional security requirements and considered alternatives of security components. In this case the manual enumerating and calculating of total values of non-functional metrics and subsequent comparisons are fairly difficult, and the benefit of the usage of the developed software tools Static Testing Tool and Configurator is especially visible. These tools allow automating stages 1, 2, 6, 7, which manual execution is difficult.

An advantage of the technique is that it allows replacement of an expert determining selection of the necessary security components. This technique is based on the expert knowledge and allows system developers to focus directly on the development of the business functions, reducing the complexity of the development process.

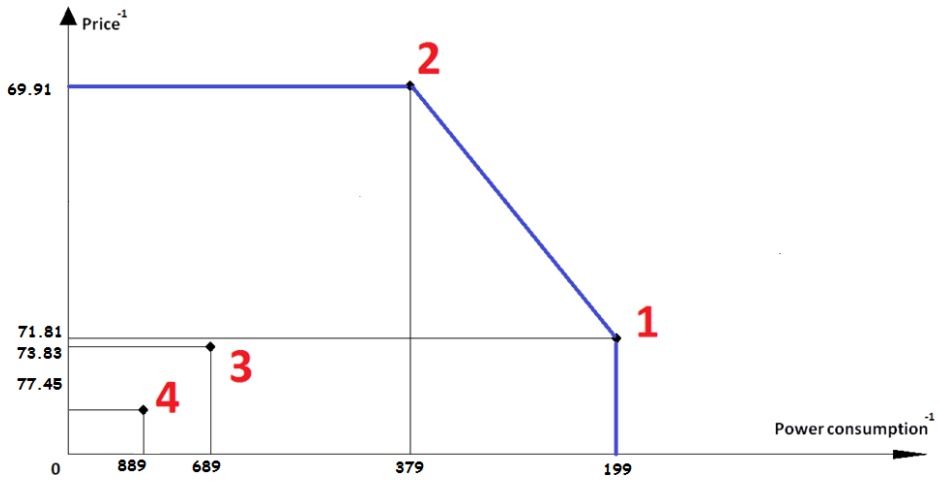

Figure 3: The selection of alternatives of hardware interfaces

8 Conclusion

In this paper we proposed a design technique for secure embedded devices on the basis of combinations of security components. The technique considers functional and non-functional characteristics of security components and device constraints, applying an optimization approach. As a proof of concept the technique applied to construction of an integrated cyber-physical security system. This system represents a set of embedded devices in charge of collection, preprocessing and secure transmission of data and a server responsible for the storage and analysis of data. The most appropriate microcontroller and extra components were selected by the technique.

In further research it is planned to significantly expand the database of embedded device components and their functional and non-functional characteristics. To improve the quality of selected combinations of components we are planning to carry out additional analysis of possible inconsistencies between components as well as to analyze possible attacks due to wrong integration of single components. Also it is planned to involve experts to evaluate the resulting technique and software prototype.

Acknowledgment

The work is performed by the grant of RSF #15-11-30029 in SPIIRAS.

References

[1] C. Gebotys, *Security in embedded devices*. Springer Science+Business Media, LLC 2010, 2010.

[2] V. Desnitsky, I. Kotenko, and A. Chechulin, "Configuration-based approach to embedded device security," in *Proc. of the 6th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security (MMM-ACNS'12), St. Petersburg, Russia*, ser. Lecture Notes in Computer Science, vol. 7531. Springer Berlin Heidelberg, October 2012, pp. 270–285.

[3] V. Desnitsky and I. Kotenko, "Expert Knowledge based Design and Verification of Secure Systems with Embedded Devices," in *Proc. of the 2014 IFIP WG 8.4, 8.9, TC 5 International Cross-Domain Conference (CD-ARES'14) and the 4th International Workshop on Security and Cognitive Informatics for Homeland Defense (SeCIHD'14), Fribourg, Switzerland*, ser. Lecture Notes in Computer Science, vol. 8708. Springer International Publishing, September 2014, pp. 194–210.

[4] V. Desnitsky, A. Chechulin, I. Kotenko, D. Levshun, and M. Kolomeec, "Application of a Technique for Secure Embedded Device Design Based on Combining Security Components for Creation of a Perimeter Protection System," in *Proc. of the 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP'16), Heraklion, Greece*. IEEE, February 2016, pp. 609–616.

[5] T. Henzinger and J. Sifakis, "The Embedded Systems Design Challenge," vol. 4085, pp. 1–15, August 2006.

[6] A. McEwen and H. Cassimally, *Designing the Internet of Things*. Wiley, 2013.

[7] J. Stankovic, "Research Directions for the Internet of Things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.

[8] O. M. Group, "The UML Profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems," 2011, http://www.omgmarte.org/ [Online; Accessed on June 10, 2016].

[9] D. Hwang, P. Schaumont, K. Tiri, and I. Verbauwhede, "Securing Embedded Systems"," *IEEE Security and Privacy*, vol. 4, no. 2, pp. 40–49, 2006.

[10] M. Knezevic, V. Rozic, and I. Verbauwhede, "Design Methods for Embedded Security"," *Telfor Journal*, vol. 1, no. 2, pp. 69–72, 2009.

[11] B. Moyers, J. Dunning, R. Marchany, and J. Tron, "Effects of Wi-Fi and Bluetooth Battery Exhaustion Attacks on Mobile Devices"," in *Proc. of the 43rd Hawaii International Conference on System Sciences (HICSS'10), Honolulu, HI, USA*. IEEE, January 2010, pp. 1–9.

[12] S. Shin, T. Lee, and H. P. In, ""defending battery exhaustion attacks on mobile systems"," in *Proc. of the 33rd Annual IEEE International Computer Software and Applications Conference (COMPSAC'09), Seattle, WA, USA*, vol. 2. IEEE, July 2009, pp. 347–352.

[13] G. Gogniat, T. Wolf, and W. Burleson, "Reconfigurable Security Primitive for Embedded Systems"," in *Proc. of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06), Honolulu, HI, USA*, vol. 10. IEEE, November 2005.

[14] J. Norman, "Open Source Physical Security," July 2012, http://www.layerone.org/wp-content/uploads/2012/07/LayerOne2012-John_Norman-DIY_Access_Control.pdf [Online; Accessed on June 10, 2016].

[15] I. Nojmol, September 2014, http://www.assaabloy.co.uk/Local/UK/Downloads/ASSA_Smartair_Whitepaper%20V3.pdf [Online; Accessed on June 10, 2016].

[16] J. Ruiz, R. Harjani, A. Mana, V. Desnitsky, I. Kotenko, and A. Chechulin, "A Methodology for the Analysis and Modeling of Security Threats and Attacks for Systems of Embedded Components"," in *Proc. of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Computing (PDP'12), Munich, Germany*. IEEE, February 2012, pp. 261–268.

[17] A. Rae and L. Wildman, "A Taxonomy of Attacks on Secure Devices," *Australian Information Warfare and IT Security*, vol. 1, no. 1, pp. 251–264, 2003.

[18] D. Abraham, G. Dolan, G. Double, and J. Stevens, "Transaction security system," *IBM Systems Journal*, vol. 30, no. 2, pp. 206–228, 1991.

[19] "Amazon," http://www.amazon.com [Online; Accessed on June 10, 2016].

[20] "Intel Galileo distributor shop," http://mouser.com/ [Online; Accessed on June 10, 2016].

[21] "Energy efficiency of DC 5V Character LCD 16x2," http://melt.com/ [Online; Accessed on June 10, 2016].

[22] "Arduino forum," http://forum.arduino.cc/ [Online; Accessed on June 10, 2016].

[23] "Raspberry Pi forum," https://www.raspberrypi.org/forums/ [Online; Accessed on June 10, 2016].

[24] "Arduino tutorials," https://www.arduino.cc/en/Tutorial/ [Online; Accessed on June 10, 2016].

[25] "Arduino shop," http://store.arduino.cc/ [Online; Accessed on June 10, 2016].

[26] "Raspberry-pi distributor shop," http://alliedelec.com/ [Online; Accessed on June 10, 2016].

[27] "Seedstudio (electronic shop)," http://seeedstudio.com/ [Online; Accessed on June 10, 2016].

[28] "Mark VandeWettering's blog," http://brainwagon.org/ [Online; Accessed on June 10, 2016].

[29] "Beaglebone distributor shop," http://digikey.com/ [Online; Accessed on June 10, 2016].

[30] "Beaglebone distributor shop," https://adafruit.com/ [Online; Accessed on June 10, 2016].

[31] "Ebay," http://ebay.com/ [Online; Accessed on June 10, 2016].

[32] "Arduino references," http://arduino.cc/en/Reference/ [Online; Accessed on June 10, 2016].

[33] "Apache Tomcat," http://tomcat.apache.org/ [Online; Accessed on June 10, 2016].
[34] "Spring Framework," http://projects.spring.io/spring-framework/ [Online; Accessed on June 10, 2016].
[35] "C3p0 pooling," http://mchange.com/projects/c3p0/ [Online; Accessed on June 10, 2016].

Author Biography

Vasily Desnitsky got a master degree in at St. Petersburg State University (Russian Federation), Department of Mathematics and Mechanics, Division of Computer Science, Direction of Software Technologies. Vasily obtained PhD degree in computer science on Methods and systems of information protection, information security in SPIIRAS (St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences). At the moment he holds a position of senior researcher at the Laboratory of Computer Security Problems of SPIIRAS. He is the author of more than 50 refereed publications and has a high experience in the research on computer network security and participated as a researcher in computer security R&D projects. His field is embedded security, Internet of Things, software security, computer network security.

Dmitry Levshun is currently pursuing Specialist degree in Information Security at the Saint-Petersburg Electrotechnical University "LETI", Russia. He is working as a developer at the Laboratory of Computer Security Problems of St.Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS). His research interests include embedded systems security and correlation of security events.

Andrey Chechulin received his B.S. and M.S. in Computer science and computer facilities from Saint-Petersburg State Polytechnical University and PhD from St.Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS) in 2013. In 2015 he was awarded the medal of the Russian Academy of Science in area of computer science, computer engineering and automation. At the moment he holds a position of senior researcher at the Laboratory of Computer Security Problems of SPIIRAS. He is the author of more than 60 refereed publications and has a high experience in the research on computer network security and par-ticipated as an investigator in several projects on developing new security technologies. His primary research interests include com-puter network security, intrusion detection, analysis of the network traffic and vulnerabilities.

Igor Kotenko graduated with honors from St.Petersburg Academy of Space Engineering and St. Petersburg Signal Academy. He obtained the Ph.D. degree in 1990 and the National degree of Doctor of Engineering Science in 1999. He is Professor of computer science and Head of the Laboratory of Computer Security Problems of St. Petersburg Institute for Informatics and Automation. He is the author of more than 250 refereed publications, including 12 textbooks and monographs. Igor Kotenko has a high experience in the research on computer network security and participated in several projects on developing new security technologies. For example, he was a project leader in the research projects from the US Air Force research department, via its EOARD (European Office of Aerospace Research and Development) branch, EU FP7 and FP6 Projects, HP, Intel, F-Secure, etc. The research results of Igor Kotenko were tested and implemented in more than fifty Russian research and development projects.