Self-adaptability in Secure Embedded Systems: an Energy-Performance Trade-off

N. Botezatu, V. Manta, and A. Stan

***Abstract*—Securing embedded systems is a challenging and important research topic due to limited computational and memory resources. Moreover, battery powered embedded systems introduce power constraints that make the problem of deploying security more difficult. This problem may be addressed by improving the trade-off between minimizing energy consumption and maintaining a proper security level.**

This paper proposes an energy-aware method to determine the security resources consistent with the requirements of the system. The proposed solution is based on a multi-criteria decision mechanism, the Weighted Product Model (WPM), used to evaluate the relations between different security solutions and to select the appropriate one based on variable runtime requirements.

***Index Terms*—encryption algorithms, power-aware system, self-adaptable security, weighted product model**

I. Introduction

ADVANCES in embedded systems design have led to the implementation of self-adaptable mechanisms, with the purpose of increasing flexibility of operation, without or with minimum user intervention. The ability to adapt to variable internal and external conditions, allows embedded systems to be used with increased performance in various environments.

The security of embedded systems poses some specific problems due to the limited computational resources available and the overhead introduced by security. Moreover, mobile or battery powered systems create constraints on power consumption which translate to stricter limitations on computational performance.

This problem may be addressed by minimizing power consumption while maintaining a proper security-performance ratio. Previous research [1] shows that the solution is not always straightforward. Subsequently, the aim of this paper is to approach the problem of securing resource-constrained embedded systems from a runtime self-adaptable perspective.

To be more precise, we propose a method for choosing security resources that are consistent with the applications running on the system, while trying to optimize power consumption and performance. The problems raised by this approach consist in: the need to determine the relevance between different security resources, based on several parameters which do not always have a straightforward relation; the existence of several applications running at the same time, with different security requirements, which can bring processing overheads and increased complexity in the process of adapting security.

The rest of the paper is organized as follows: Section II briefly outlines the related work; Section III presents the proposed method, explains its components and system interactions. Section IV presents a case study in order to show how this approach works and Section V highlights the contributions of this work and proposes some future research perspectives.

Manuscript received March 4, 2011; revised April 1, 2011.

N. Botezatu is a PhD student at "Gh. Asachi" Technical University of Iasi, Faculty of Automatic Control and Computer Engineering (e-mail: nbotezatu@cs.tuiasi.ro)

V. Manta is with the "Gh. Asachi" Technical University of Iasi, Faculty of Automatic Control and Computer Engineering (e-mail: vmanta@cs.tuiasi.ro).

A. Stan is with the "Gh. Asachi" Technical University of Iasi, Faculty of Automatic Control and Computer Engineering (e-mail: andreis@cs.tuiasi.ro)

II. Related work

We focus on giving an overview on papers that cover the aspects of adaptable security with respect to energy considerations.

This paper is base on our previous work [2], where we covered the aspects of self-adaptable security at a theoretical level. It proposes a security architecture for embedded systems and discusses the guidelines used for constructing an adaptable mechanism.

In [3] the self-adaptable security is addressed at system level, centered on the concept of security policy. The authors propose a domain-independent methodology for system security adaptability.

In [4] and [5], the authors propose the use of AHP to define priorities among different system requirements and to compare different security solutions. The approaches rely on the solution of an optimization problem which is addressed by a Linear Programming formulation.

AHP was also used in [6], where a context aware and adaptive security manager (CASM) is presented. The CASM is used to secure 802.11 wireless networks.

In [7] a battery power optimization technique for embedded systems is presented. The authors performed real experiments to model the relationship between power consumption and security. Also, the evaluation of security was bound to a vulnerability metric.

A conceptual model for designing adaptable security services is presented in [8]. The proposed model is used to describe and compare existing security services, by describing their requirements.

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

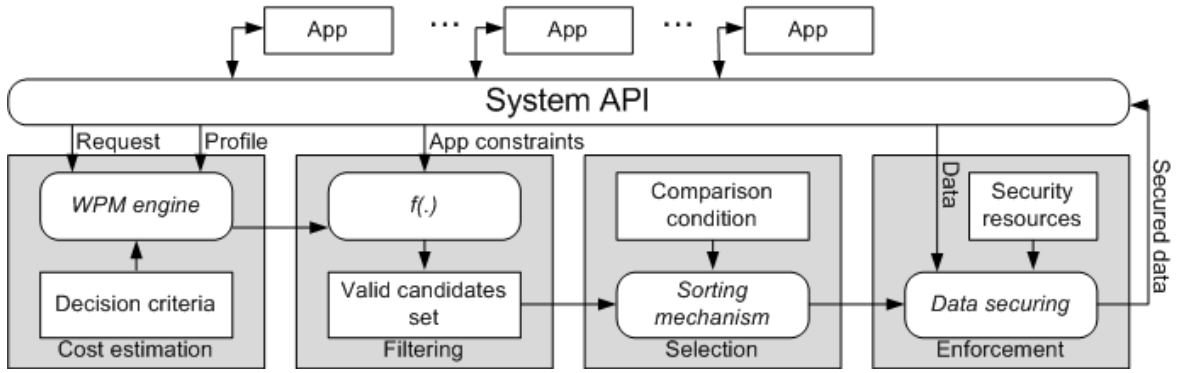


Fig. 1. Self-adaptable architecture

III. Security Adaptation

In this section we present the mechanism used for adapting the security, at runtime. It consists of the following 4 stages (Fig. 1):

- *Cost estimation* – this stage consists in determining a relative cost for all available security resources (SR), based on a multi-criteria decision mechanism, the Weighted Product Model (WPM), which in turn is dependent on the system requirements
- *Filtering stage* – based on the application constraints, regarding which security resource can be used for securing it, only the security resources that match the selection criteria are considered valid candidates for use
- *Selection stage* – the security resources in the valid candidates set are sorted, thus determining the best candidate
- *Enforcement stage* – in this stage the selected security resource is applied for system use

As Fig.1 shows, the system provides an interface for the applications to signal the need for securing specific data. Also, the system provides for the user the possibility to select a usage profile, at runtime. The system input for our selection mechanism is defined as $Sys = \langle req\_data, profile \rangle$, where *req_data* represents the amount of data (e.g. bytes) to be secured when an request is issued by an application and *profile* represents the runtime profile.

Based on this information, when the system is ready to process an application request, the relative cost of every security resource is calculated.

Other inputs to the selection mechanism are represented by the application constraints which are of use in the second stage and the comparison condition which is used to determine the best candidate. More on these inputs will be discussed in subsection B.

A. Energy and performance considerations

Before describing further our method, the objectives of this adaptive model should be considered. Its purpose is to determine the best alternative for securing and application with respect to energy consumption and performance.

The model used for tracking the consumed energy is based on the quantity of data to be secured (e.g. number of bytes to be encrypted) and on the energy used in the setup stage of the security resource (e.g. key generation), as in (1).

$$E = E_{setup} + data \cdot E_{data} \tag{1}$$

The performance of a security resource is expressed as two independent measures: the throughput that can be achieved and a latency expressed as the time elapsed between the selection of a security resource and the start of the securing process. More precisely, it refers to the time needed for the setup phase of a security resource.

For the purpose of characterizing security from the two fore mentioned points of view, a security resource is describes as a tuple $SR = \langle E_{setup}, T_{setup}, E_{data}, T_{data} \rangle$, where E_{setup} represents the energy consumed during the setup process, T_{setup} is the time taken by the setup phase, E_{data} is the energy consumed and T_{data} is the time taken for securing one data unit.

B. Cost calculation

Starting from the premise that this adaptable method is to be used in embedded systems, we selected to investigate the WPM as the mechanism used to estimate the cost of the security resources. The following paragraph presents an overview of the method.

Given a set of m alternatives to be selected, a set of n decision criteria and w_j, $j=1,n$ the relative weight of importance for criterion C_j, the performance value of alternative A_x relative to alternative A_y is calculated as in (2).

$$P(A_x/A_y) = \prod_{j=1}^{n} \left(a_x^j / a_y^j\right)^{w_j}, x = \overline{1,m}, y = \overline{1,m} \tag{2}$$

where a_i^j is the performance value of alternative A_i evaluated in terms of criterion C_j and $\sum w_j = 1$.

If the ratio $P(A_x/A_y) \geq 1$, then the alternative A_x is said to be more desirable than alternative A_y, when the considered decision problem is a maximization one. In the case of a minimization problem, the ratio should be less than one for A_x to be more desirable than A_y.

The WPM has two main advantages when considering our problem: it has a low implementation complexity, expressed as processing overhead, and it is a dimensionless analysis method, meaning it eliminates any units of measure from the performance values of the alternatives.

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I
PERFORMANCE VALUE EXPRESSIONS

Criterion	Expresion
energy	$([data / th_{setup}] + active\_SR) \cdot E_{setup} + data \cdot E_{setup}$
throughput	T_{data}
latency	$([data / th_{setup}] + active\_SR) \cdot T_{setup}$

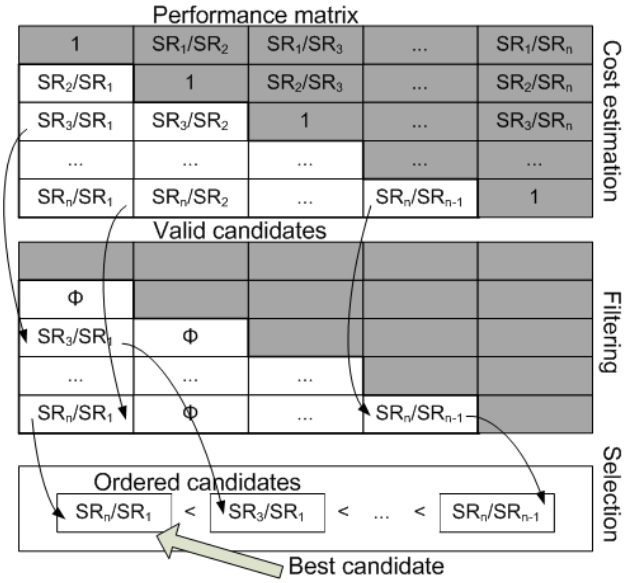


Fig. 2. Sample decision process

C. Evaluation elements

In particular, the elements considered for the WPM are the following: the alternatives are represented by the security resources; the evaluation criteria are represented by the energy, throughput and latency characteristics of the security resources.

The performance values for the energy criterion are based on (1) and may vary with every evaluation request due to changes in data payload. In general, the setup process for a security resource is rerun after an elapsed period of time or after a certain amount of data being secured. We take into account the latter of the two cases, by setting a threshold value, th_{setup}, for the data payload, which retriggers the setup process.

The throughput criterion has a constant value at runtime, based on the characterization values of the security resources, whereas the latency criterion varies with the number of reruns of the security setup process.

One more aspect is taken into account when determining the performance values for the energy and latency criteria: the active security resource. The active security resource may be more favorable from the previous two criterion perspective because is not necessarily to run the setup process, translating in a lower energy consumption and a lower latency. Table I presents the formulas used to determine the performance values. *data* represents the amount of data to be secured, the square brackets denote the integer part of the operand and $active\_SR = \{0, 1\}$ equals 0 when the security resource being evaluated is already in use and 1 when it is not used.

The weights for the importance of the 3 criteria are given by the profile system parameter which is runtime variable and is user selectable.

D. Process description

The first step is to calculate the relative ratios between all security resources, resulting in a cost matrix, as in Fig. 2. Only the lower left portion of the matrix is further considered, as the other half represents the inverse ratio and the diagonal always evaluates to 1.

Then, the valid candidates set is determined, based on the application security constraints. Let $S = \{SR_1, SR_2, ..., SR_n\}$ be the set of all security resources implemented in the system. Every application App_x is characterized by a subset $S_{App_x} \subset S$ which represents the security resources that can be used by it. The selection of the subsets is not in the scope of this paper. Based on this subset, the WPM ratios that contain at least one alternative that is not contained in it, are not included in the valid candidates set.

The last of the selection steps consists in ordering the remaining values in the matrix (i.e. the valid candidates set). We consider one security resource to be more desirable than another, if their relative cost is less than 1. This is the comparison condition, as presented in Fig. 1. Based on this, all relative dependencies between the security resources are converted to an absolute expression, which points out the security resource with the lowest cost. It represents the best candidate, which is used, in the last stage of the method, to secure the application data.

IV. CASE STUDY

This section presents a case study, in order to demonstrate how the adaptation mechanism works. We assume to have a system running three applications and four security resources.

In order to evaluate the performance of the symmetric ciphers (i.e. the security resources), we used an ARM9 platform running an implementation of Crypto++ library, under a Linux 2.x kernel. The power consumption and execution time measurements were recorded using a Tektronix TDS2024 oscilloscope, sampling a resistor. The resistor was series connected with the power connector. The input data was represented by files of 4 kB in size, encrypted in CBC mode.

The evaluation of the adaptation mechanism was performed using a SystemC implementation and simulation of the system in Fig.1.

A. System description

The system implements four security resources presented in Table II. They were chosen due to their use among

TABLE II
SECURITY RESOURCES CHARACTERISTICS

Security resource	Encryption method	Setup energy - E_{setup} (µJ)	Setup time - T_{setup} (µsec)	Encryption energy - E_{data} (µJ/byte)	Encryption time - T_{data} (µsec/byte)
SR_1	3DES	89	305	6,7	11,7
SR_2	DES	29,6	99,2	2,1	4,1
SR_3	AES	7,7	62,9	1,3	7,2
SR_4	Blow Fish	3057	875	0,8	2

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE III
RUNTIME PROFILE CRITERIA WEIGHTS

Runtime profile	Energy criterion	Throughput criterion	Latency criterion
Low consumption (LC)	0.8	0.1	0.1
High performance (HP)	0.2	0.4	0.4
Balanced (B)	0.5	0.25	0.25

different implementations of TLS/SSL [9], for bulk data encryption.

The amount of data considered for the requests is the only stochastic input of the system and is generated using a Bounded Pareto distribution. The lower limit is set at one unit, the upper one is set at 8k units and the alpha parameter was chosen to be one. These values are consistent with the requests from all three applications.

The three applications send security requests to the system on a time basis and in a successive manner ($App_1 \rightarrow App_2 \rightarrow App_3 \rightarrow App_1 \rightarrow ...$). The elapsed time between two consecutive requests is chosen to be longer than the time taken to secure the largest type of request, issued by any of the three applications, with the lowest performance cipher. The reason is to eliminate any wait times from the performance measurements.

The only difference between the applications is represented by the security constraints. The corresponding sets are: $S_{App_1} = \{SR_2, SR_4\}$, $S_{App_2} = \{SR_1, SR_4\}$, $S_{App_3} = \{SR_1, SR_2, SR_3\}$. The elements for these sets were chosen to illustrate all the types of dependencies that can appear in the adaptation process, as it will be further discussed in the next subsection.

Finally, in order to see how our mechanism varies with the criteria weights, there were considered three runtime profiles (Table III). The Low consumption profile gives greater importance to the security resources that are more energy efficient; the High performance one considers more important the security resources with higher throughput and lower latency, whereas the Balanced profile gives equal importance to both energy and performance aspects. Every system run had a constant profile selection for the entire simulation period.

B. Static analysis

In order to determine how the system adapts, relevance matrixes were calculated for every value in the range of the input distribution, for every runtime profile. The relations between some of the security resources are constant over the entire input set, while for others the relations change based on a threshold value for the input data amount. Table IV shows the relations between the security resources cost, for the LC profile. The upper half displays the relations true when none of the two security resources is already in use. Relations in the lower half assume the security resources on the rows are in use. The << and >> symbols show a constant relation over the input set. For the variable relations, the threshold value is given.

The reason for this is given by the relations between the setup energy and the energy need for securing data, which are not consistent for all the security resources. For example, it is more convenient to secure a large amount of data with SR_4, when the energy consumed during the process is at least equal to the one consumed for the setup. The same explanation is also valid for the latency criterion.

TABLE IV
SECURITY RESOURCES RELATIONS FOR LC RUNTIME PROFILE

	SR_1	SR_2	SR_3	SR_4
SR_2	$SR_2 << SR_1$	–	–	–
SR_3	$SR_3 << SR_1$	$SR_3 << SR_2$	–	–
SR_4	$SR_4 > SR_1$, data ≤ 429 $SR_4 < SR_1$, data > 429	$SR_4 > SR_2$, data ≤ 4185 $SR_4 < SR_2$, data > 4185	$SR_4 >> SR_3$	–
SR_1	–	$SR_1 < SR_2$, data ≤ 27 $SR_1 > SR_2$, data > 27	$SR_1 < SR_3$, data ≤ 5 $SR_1 > SR_3$, data > 5	$SR_1 < SR_4$, data ≤ 2080 $SR_1 > SR_4$, data > 2080
SR_2	$SR_2 << SR_1$	–	$SR_2 << SR_3$	$SR_2 << SR_4$
SR_3	$SR_3 << SR_1$	$SR_3 << SR_2$	–	$SR_3 << SR_4$
SR_4	$SR_4 << SR_1$	$SR_4 << SR_2$	$SR_4 << SR_3$	–

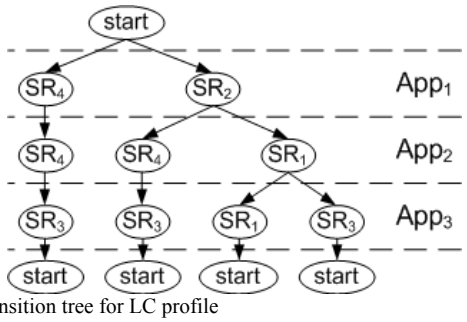


Fig. 3. Transition tree for LC profile

Based on these observations, for every runtime profile, there can be built a state tree, which shows all possible transitions between security resources. Fig. 3 presents such a sample tree for the LC profile. The adaptation sequences are limited by three factors: security constraints of the applications; bounds of the input data and active security resource at the estimation stage.

C. Experimental results

All simulations were run for a total of 2.5 MB of data. Each simulation scenario was executed several times, the results considered in this subsection are their mean values.

First, we evaluated the usage level of the security resources, for all three applications, under all the runtime profiles (Fig. 4). The results show that when using the HP and B profiles, less security adaptations occur: App_1 uses only SR_2 and for App_3, SR_3 is used for less than 1% of the data. Another interesting observation is that SR_2 is never used by App_3, although it is included in its security constraint set. This situation appears because: SR_3 always has a better cost than SR_2; and even though SR_1 has roughly three times the cost of SR_2, it is better suited when it is already in use.

Next, all three metrics considered for the decision criteria were evaluated. When considering the throughput, in all simulation scenarios for the adaptable method, the results were better than in the case of running the system with only one constant security resource (Fig. 5). The same remarks are also valid for the latency (Fig. 6). It can be observed that in the case of App_1 and App_3 the throughput is worse for the HP and B profiles, compared to the LC one. This is

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

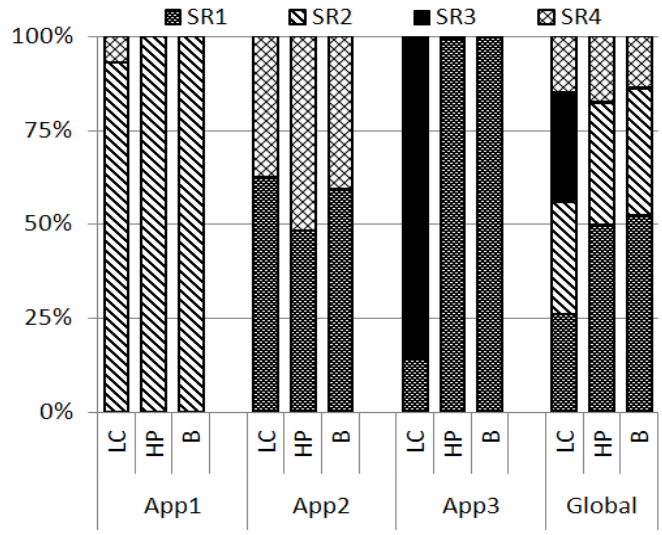


Fig. 4. Security resources usage distribution

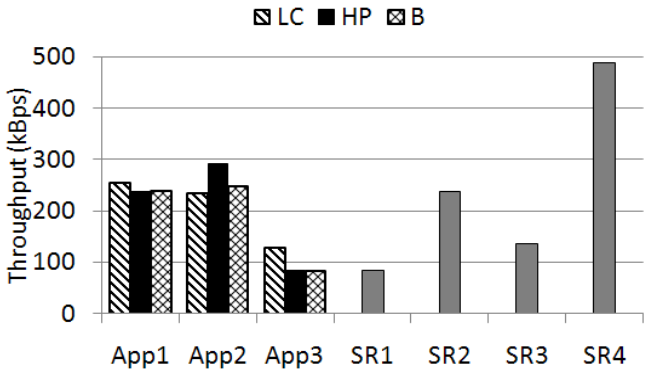
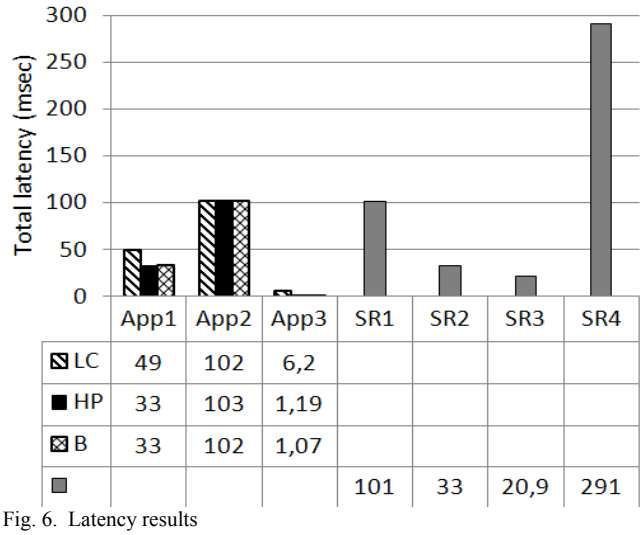


Fig. 5. Throughput results



	App1	App2	App3	SR1	SR2	SR3	SR4
LC	49	102	6,2				
HP	33	103	1,19				
B	33	102	1,07				
				101	33	20,9	291

Fig. 6. Latency results

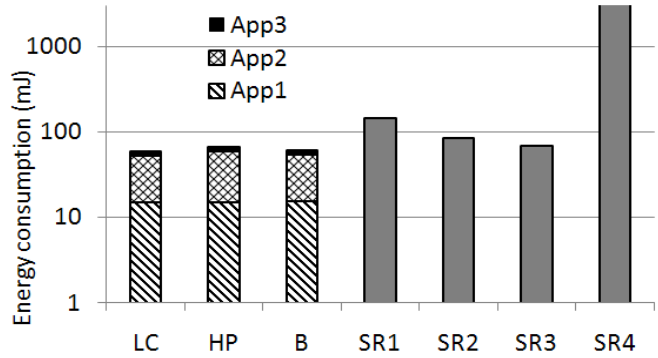


Fig. 7. Energy consumption results

explained by the fact that latency and throughput are of equal importance when estimating the cost – in the fore mentioned cases, the latency values are better for the two profiles. The same situation appears for App_2, this time inversed – better throughput with a slight increase in latency.

By examining the values obtained for the energy consumption, we can conclude that the objective of reduced energy consumption was achieved. The results in Fig. 7 are also presented against the use cases for only one constant security resources. The mean energy saving varies between 12% when compared to the constant use of SR_3 and 97% when compared to the constant use of SR_4.

V. Conclusion and Future Work

In this paper we have proposed a novel self-adaptable security mechanism for embedded systems. The problem of choosing a security resource is formulated as a multi-criteria decision and is addressed by implementation of the Weighted Product Model technique. The advantages of using this solution are twofold: first, because it is a dimensionless analysis model, the decision criteria can be selected based only on its importance and second, different runtime scenarios can be used.

The objective of this method is to reduce the energy consumption with respect to performance metrics. The simulation result obtained for the case study defend our claim, the result being illustrated from both energy and performance perspectives.

Future work includes the evaluation of our mechanism under real life scenarios, through prototype implementations on energy constrained embedded platforms. Also, it is necessary to development a methodology for creating runtime profiles, based on system specific characteristics.

References

[1] S. Ravi, P. Kocher, and S. Hattangady, "Security in Embedded Systems: Design Challenges", in *ACM Transactions on Embedded Computing Systems*, vol. 2, no. 3, Aug. 2004, pp. 461–491.

[2] N. Botezatu, V. Manta, and A. Stan, "Self-adaptable Security Architecture for Power-aware Embedded Systems", in *Proceedings of the 14th International Conference on System Theory and Control*, Sinaia, Romania, 2010, pp. 98–103.

[3] A. Ferrante, et al., "Self-adaptive Security at Application Level: a Proposal", in *Workshop on Reconfigurable Communication-centric SOCs*, Montpellier, France, 2007.

[4] A. Taddeo, P. Marcon, and A. Ferrante, "Negotiation of security services: a multi-criteria decision approach", in *Proceedings of the 4th Workshop on Embedded Systems Security*, Grenoble, France, 2009, Paper 4.

[5] A. Taddeo, and A. Ferrante, "Run-time selection of security algorithms for networked devices" in *Proceedings of the 5th ACM Symposium on QoS and security for wireless and mobile networks*, Tenerife, Spain, 2009, pp. 92–96.

[6] C. Hager, "Context Aware and Adaptive Security for Wireless Networks", Ph.D. dissertation, Dept. Elect. Eng., Virginia Polytechnic Univ., Blacksburg, VA, 2004.

[7] R. Chandramouli, et al., "Battery Power-aware Encryption" in *ACM Transactions on Information and System Security*, vol. 9, no. 2, May 2006, pp. 162–180.

[8] S. Lindskog, Z. Faigl, and A. Brunstrom, "A Conceptual Model for Analysis and Design of Tunable Security Services", in *Journal of Networks*, vol. 3, no. 5, May 2008, pp. 1–12.

[9] N. Potlapally, et al., "Analyzing the energy consumption of security protocols", in *Proceedings of the 2003 International Symposium on Low power electronics and design*, Seoul, Korea, 2003, pp. 30–35.

ISBN: 978-988-18210-6-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)