DE GRUYTER OPEN

Journal of ELECTRICAL ENGINEERING, VOL. 66, NO. 1, 2015, 47–52

AN ENERGY EFFICIENT PROTOCOL FOR THE INTERNET OF THINGS

Algimantas Venčkauskas — Nerijus Jusas
Egidijus Kazanavičius — Vytautas Štuikys *

The Internet of Things (IoT) is a technological revolution that represents the future of computing and communications. One of the most important challenges of IoT is security: protection of data and privacy. The SSL protocol is the de-facto standard for secure Internet communications. The extra energy cost of encrypting and authenticating of the application data with SSL is around 15 %. For IoT devices, where energy resources are limited, the increase in the cost of energy is a very significant factor. In this paper we present the energy efficient SSL protocol which ensures the maximum bandwidth and the required level of security with minimum energy consumption. The proper selection of the security level and CPU multiplier, can save up to 85 % of the energy required for data encryption.

K e y w o r d s: information security, computer networks, cryptography, energy consumption

1 INTRODUCTION

The Internet of Things (IoT) is a technological revolution representing the future of computing and communications, and its development depends on a dynamic technical innovation in a number of important fields, from mobile devices to wireless sensors and nanotechnology. The IoT describes a vision where objects (things of Internet) become part of the Internet: where every object is uniquely identified and accessible to the network, its position and the status are known, where services and intelligence are added to this expanded Internet by fusing the digital and physical world, ultimately making an impact on our professional, personal and social environments [1, 2]. IoT is very rapidly expanding; in the Internet-connected devices it will grow from 10 billion in 2010 to 50 billion in 2020 [3]. One of the most important challenges of IoT is security, *ie* protection of data and privacy [4]. The solution of these problems can be made by using a variety of methods: VPN, firewall, secure communication protocols, *etc* [5]. The Secure Sockets Layer (SSL) protocol [6] is the de-facto standard for secure Internet communications at the applications level and is supported by virtually all modern Web services and clients. More and more IoT devices (sensors, mobile devices, PDAs, cell phones) provide wireless Web access. There is an increasing demand for establishing secure SSL connections on IoT devices that are relatively constrained in terms of computational and energy resources. The extra energy cost of encrypting and authenticating application data with SSL is around 15 % [7]. For IoT devices, where energy resources are limited, it is a very significant increase in the cost of energy.

The main problem with the SSL protocol is the choice of methods for efficient cryptographic algorithms and encryption keys. This is not described in the SSL standard.

In this paper we present the energy efficient SSL protocol which ensures the maximum bandwidth and the required level of security with minimum energy consumption. We investigated energy consumption dependence on the level of security and the processor performance (the CPU multiplier).

2 RELATED WORK

Reduction of energy consumption of IoT devices in secure communications can be achieved in different ways: by establishing specific protocols, by adding security elements to the communication protocols, by moving the solution of the security problem to the application level and by creating the dedicated hardware for cryptographic operations.

Kamel *et al* [8] propose to build security protocols for IoT environments based on the insecure version of these protocols from the software security components by offering each one a security property. By externalizing the security functionalities and adapting the security level to the user's needs, they provide the security management architecture adapted to IoT environments.

Prasithsangaree *et al* [9] proposed the concept of providing just enough security in which the security level is determined and the security services are provided with the minimum use of energy resources. They proposed the energy-efficient security framework which consists of three techniques for the energy-efficient security protocol design. The first technique suggests replacement of the security algorithm by the one that consumes less energy,

* Faculty of Informatics, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania, algimantas.venckauskas@ktu.lt

DOI: 10.1515/jee-2015-0007, Print ISSN 1335-3632, On-line ISSN 1339-309X © 2015 FEI STU

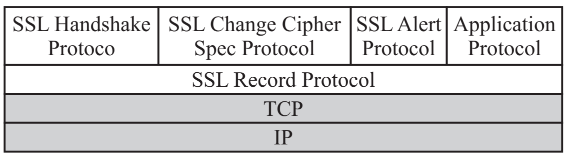

Fig. 1. The SSL protocol stack

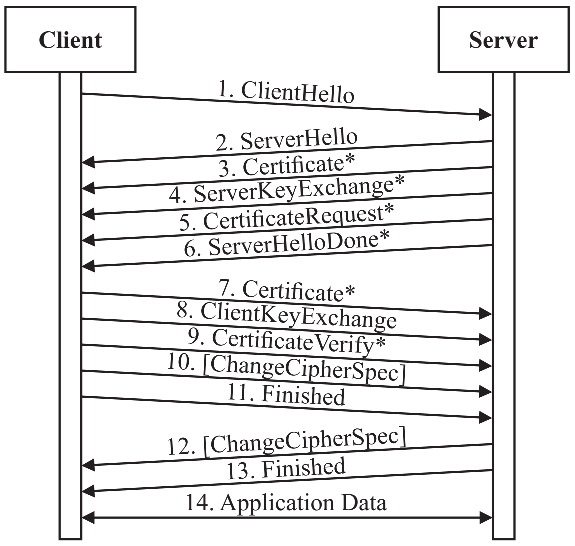

Fig. 2. SSL Handshake between Client and Server, * Indicates optional or situation-dependent messages that are not always sent

the second suggests the modification of the standard protocols in order to achieve energy efficiency, and the third technique suggests the Greenfield approach, where a new system is designed to provide the energy- efficient security protocol.

Cano and Domenech-Asensi [10] present a secure energy-efficient solution for IoT devices. Security is provided at the application layer, thus, making it possible to apply different security policies depending on the data to be sent with the corresponding resource saving. Their proposed solution does not require an Internet connection, the packets will only be sent when necessary, thus, saving the bandwidth and the battery.

In order to improve the efficiency of secure communication processing on the embedded devices, Potlapally *et al* [11] propose to use the embedded processors. They examine the impact of customizing an embedded processor by synergistically 1) configuring the architectural parameters, such as instruction and data cache sizes, processor-memory interface width, write buffers, *etc*, and 2) extending the base instruction set of the processor by using the custom instructions for both cryptographic and protocol processing.

Cryptographic keys are generated by using the software generators, utilizing device characteristics [12, 13], *etc*.

However, such solutions are not flexible, not suitable to the existing web applications causing the need to reprogram not only the client part of the system, but also a server part or the need for additional hardware.

In this paper we propose to modify the SSL protocol on the client side only. By establishing the connection to the server, the client selects the cryptographic algorithm and the length of the encryption key taking into account the energy requirements and the level of security, as defined in the security policy.

3 BASICS OF THE SSL PROTOCOL

The SSL protocol is one of the most widely used security protocols on the Internet. It provides the basic security services of encryption, the server and client authentication, and the integrity protection for the data exchanged over the potentially insecure networks [6]. The SSL protocol operates typically between the Transmission Control Protocol (TCP) layer and the Application layer, as shown in Fig. 1.

The SSL protocol consists of two main layers, the SSL record protocol provides the basic services to the higher-layer protocols: SSL handshake, SSL cipher change, SSL alert, and Application protocol. Now we describe the basic operating principles of the SSL protocol (Fig. 2).

The cryptographic parameters of the session state are produced by the SSL handshake protocol (steps 1–13). When the SSL client and the server first start communicating, they agree on the protocol version, select the

Table 1. Data security policy matrix

Levels/ Objectives	U	SU, R	C	S	TS
C	NUL	DES RC4	AES128 3DES	AES192	AES256 Blowfish
I	NUL	MD5	SHA1	SHA256	SHA512

Table 2. The matrix of energy needs and the speed of cryptographic algorithms

Algorithm/ CPU clock	NUL	MD5	$\cdots$	AES256
c_1	$0,0$	e_{12}, s_{12}	$\ldots$	e_{1n}, s_{1n}
$\ldots$	$\ldots$	$\ldots$	$\ldots$	
c_m	$0,0$	e_{m2}, s_{m2}	$\ldots$	e_{mn}, s_{mn}

cryptographic algorithms, optionally authenticate each other and exchange the session encryption key. The CipherSuite list, passed from the client to the server in the client hello message, contains combinations of the cryptographic algorithms supported by the client in the order of the client's preference (first choice first). Each CipherSuite defines the key exchange algorithm, the cipher algorithm, the secure hash functions and the compression algorithm. The server will select the CipherSuite.

After the handshake is complete, the client and the server may begin exchanging the application layer data (step 14).

As seen from the above description of the SSL protocol, the client composes a list of potential cryptographic algorithms. After assessing his needs for the level of security and computing the energy resources, the client can choose the best data encryption and hashing algorithms. This is the essence of our proposed method.

4 THE PROPOSED ADAPTIVE SSL PROTOCOL

The core of the proposed adaptive energy consumption SSL protocol is:

1. Data transmission security levels are described in the data protection policy, which provides security objectives, data protection levels, cryptographic algorithms and key lengths that provide an adequate level of security.
2. The processor in the client device can function in a variety of performance modes with different energy consumption level.
3. During the installation of the protocol in the device the database of energy needs for the cryptographic algorithms for different CPU performance and energy demand modes is composed.
4. By initiating the data transfer, the client indicates the appropriate level of security and the volume of data to be transmitted.
5. According to the client's requirements for the level of security and the available energy resources, in the "*ClientHello*" step the optimal in Pareto sense [14] set of the cryptographic algorithms maximizing bandwidth and minimizing energy consumption is chosen.

The SSL protocol implements the following security objectives: confidentiality (C), integrity (I) and availability (A). In addition, we define another characteristic of the protocol — protocol bandwidth (S) which is important for assessing the quality of the protocol. Let us denote the security objective by $o_i \in O = \{C, I, A, S\}$.

For each security objective a finite number of security levels are introduced. The type of data security classification levels will depend on the nature of the organisation, *eg* , in the business sector, levels, such as Public, Sensitive, Private, and Confidential, in the government sector, labels, such as Unclassified (U), Sensitive But Unclassified (SU), Restricted (R), Confidential (C), Secret (S), and Top Secret (TS) are introduced [15]. The suite of security levels can be described as a partially ordered set

$$L = \{U, SU, R, C, S, TS\}\,,\ L_i \in L\,.$$

For different security objectives the security levels can be different.

To achieve the security objectives of the SSL protocol, proper cryptography techniques have to be taken. To ensure confidentiality, SSL uses symmetric encryption algorithms with different key length to encrypt the data. The suite of these algorithms can be described as a partially ordered set.

$$A = \{NUL, DES, RC4, AES128, 3DES,\\ AES192, AES256, \mathit{Blowfish}\}\,,\ a_i \in A\,.$$

To ensure integrity, SSL uses cryptographic hash functions. The suite of hash functions can be described as a partially ordered set

$$H = \{NUL, MD5, SHA1, SHA256, SHA512\},\ h_i \in H\,.$$

In the SSL protocol for IoT devices data availability is ensured indirectly; in the time of session initialization the availability of the required amount of energy should be evaluated. The energy consumed by an IoT device performing a certain cryptographic algorithm, $e_i \in E$ depends on the CPU performance and the CPU multiplier $c_i \in C$ as well.

The data security policy can be described by the matrix $P = O \times L \times A \times H$, which defines the minimum level of cryptographic algorithms required to ensure a certain level of security. The fragment of the matrix is presented in Tab. 1.

The energy needs and the speed of cryptographic algorithms, depending on the processor performance modes

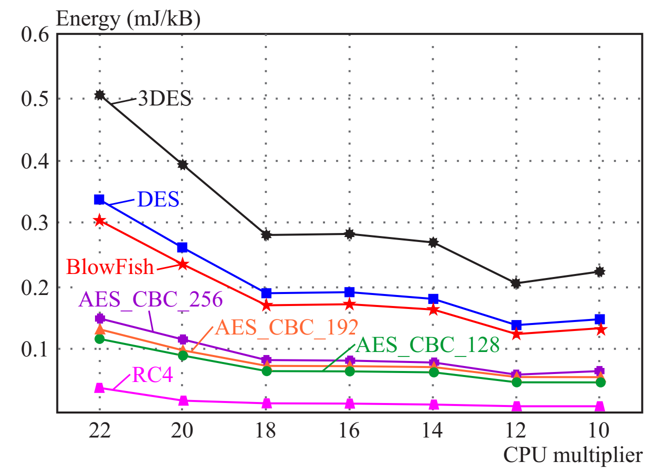

Fig. 3. Energy consumption dependence on different CPU multiplier

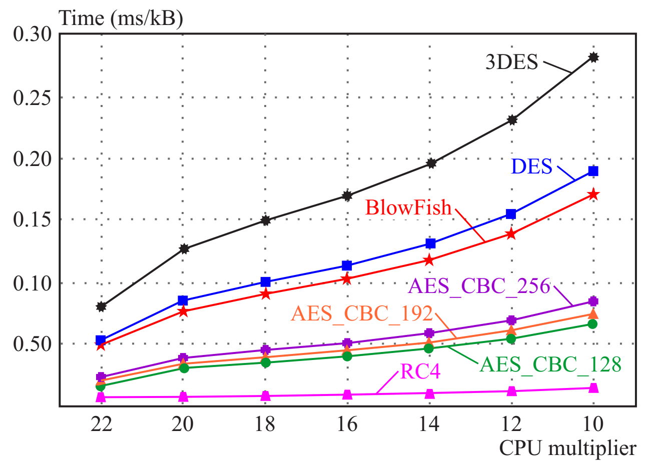

Fig. 4. Time encryption dependence on different CPU multiplier

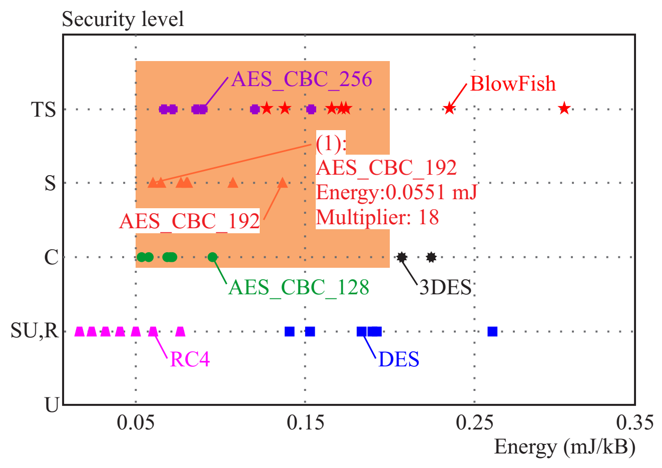

Fig. 5. The relationship between the encryption algorithms realizing a certain level of security and energy costs, depending on the CPU multiplier

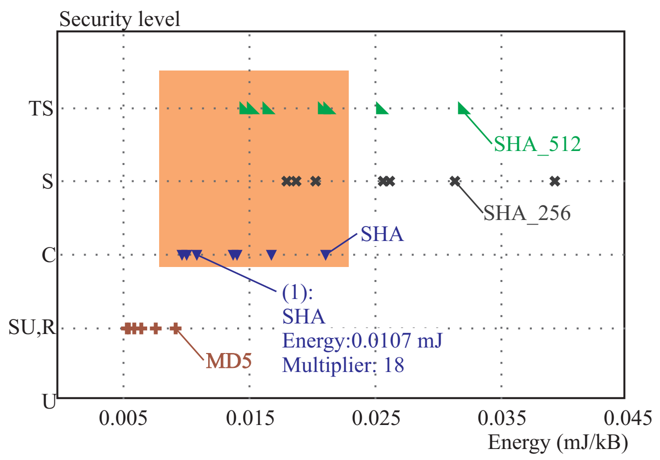

Fig. 6. The relationship between the hash functions realizing a certain level of security and energy costs, depending on the CPU multiplier

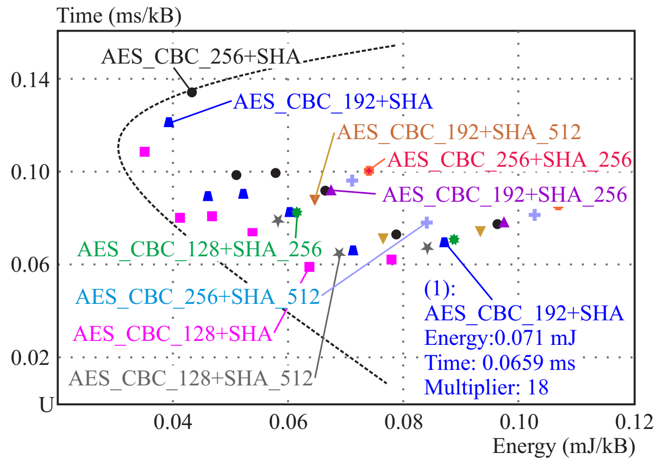

Fig. 7. Set of Pareto-optimal solutions

are described by the matrix $EC = C \times E \times H \times E \times S$. The fragment of the matrix is presented in Tab. 2.

During the initiation of the data transfer session, optimal in the sense of Pareto suite of cryptographic algorithms is chosen. Such algorithm suite provides the necessary level of security, and has the highest bandwidth and requires a minimum of energy. Our multiobjective optimization problem can be presented by the following objective functions

$$\min f_a(a_i, c_i), \text{ where, } a_i \in A, c_i \in C, f_a \text{ — execution time for the algorithm } a_i; \quad (1)$$

$$\min f_h(h_i, c_i), \text{ where, } h_i \in H, c_i \in C, f_n \text{ — execution time for the algorithm } h_i; \quad (2)$$

$$\min f_e(a_i, h_i, c_i), \text{ where, } a_i \in A, h_i \in H, c_i \in C, f_e \text{ — energy consumption during the implementation of algorithms } a_i \text{ and } h_i; \quad (3)$$

with restrictions
$l_i \geq l^*$, $l_i \in L$, l^* — the minimal level of the security required;
$e(c_i) \leq e^*$, $c_i \in C$, e^* — the maximum possible amount of energy.

These criteria are conflicting, because reducing the algorithms' execution time increases the required amount of energy. We will solve the problem by creating a set of Pareto-optimal solutions.

The objective cost function of multiobjective optimization problem can be presented by the following functions

$$\min cost = (w_a f_a + w_h f_h + w_e f_e), \text{ where } w_a, w_h, w_e \text{ — weighting factors of the functions.} \quad (4)$$

The best solution from the set of Pareto-optimal solutions will be found from Equation (4) by selecting the weighting factors of the functions.

5 EXPERIMENTS

For our experiments, we use a PDA with a mobile Intel Core i7-2720QM CPU, frequency of which is 2.2 GHz, the base clock is 100.0 MHz, and the CPU multiplier is 22. We use the cryptographic software library from the OpenSSL [16] to implement the test programs in our experiments, because this library has been widely used in the research community. It has been rigorously reviewed by many cryptographers and programming experts for its correctness and performance. Energy consumption and the speed of cryptographic algorithms can be measured by using various tools and methods [17]. In our experiments we used *Joulemeter* from Microsoft Research [18] as measurements tool.

During the experiment, we measured the energy consumption and execution time encryption algorithms and hash functions used in the SSL protocol by changing the CPU multiplier. The measurement results for the encryption algorithms are presented in Figs. 3 and 4. Increasing the CPU multiplier causes increase of energy consumption required to encrypt the information unit (KB) and decrease of the execution time. Similar results were obtained for hash functions.

According to the security policy matrix P (Tab. 1) and the measurement of the resulting energy consumption, we build matrices, which reflect the relationship between the algorithm, the security level, the CPU multiplier and energy consumption (Figs. 5 and 6). For example, the point (1) corresponds to the following set of parameters: algorithm — $AES192$, level of security — S, CPU multiplier — 18, energy consumption — 0.055 mJ/kB. In the matrix composed we indicate the area that meets the given security requirements and the restrictions of energy consumption. For example, Fig. 5 presents the area in which the level of security is equal to or higher than C and energy consumption is not higher than 0.2 mJ/kB. The measurements show that the energy consumption of hash functions is app. 10 times lower than needed for the encryption algorithms, so the restrictions of energy consumption for hash functions are modified respectively (Fig. 6).

By using the sets of the encryption algorithms and hash functions satisfying the restrictions (Figs. 5, 6) we compose the Pareto-optimal set of solutions taking into account energy consumption and the execution time objective function (Fig. 7).

6 CONCLUSIONS

One of the most important challenges of the Internet of Things is security and limited resources of energy of an IoT device. In this paper we present the adaptive to energy consumption SSL protocol which ensures the maximum bandwidth and the required level of security with minimum energy consumption.

The processor of an IoT device can function in a variety of performance modes with a different energy consumption level. The research has shown that by the management of the CPU operating modes it is possible to optimize energy consumption for the given level of security. Reducing CPU energy consumption (CPU multiplier) with the same energy resources a higher level of security expense of the bandwidth can be achieved. Experiments has shown, that the proper selection of the security level and CPU multiplier, can save up to 85 % of the energy required for data encryption.

The proposed adaptive SSL protocol composes the Pareto-optimal solutions set, which provides the necessary level of security.

Acknowledgements

This research is funded by the European Social Fund under the project "Research and Development of Internet of Things and Services Technologies and Infrastructure for the Smart Environments" (IDAPI, Agreement No VP1-3.1-ŠMM-08-K-01-018).

References

[1] The Internet of Things, International Telecommunication Union, 2005. www.itu.int/osg/spu/publications/internetofthings/.

[2] GAVRAS, A.—KARILA, A.—FDIDA, S.—MAY, M.—POTTS, M. : Future Internet Research and Experimentation, SIGCOMM Computer Communication, rev. **37** No. 3 (July 2007), 89–92.

[3] REED, D. A.—GANNON, D. B.—LARUS, J. R. : Imagining the Future: Thoughts on Computing, Computer **45** No. 1 (Jan 2012), 25–30.

[4] WEBER, R. H. : Internet of Things — New Security and Privacy Challenges, Computer Law & Security Review **26** No. 1 (Jan 2010), 23–30.

[5] KAZANAVICIUS, E.—KAZANAVICIUS, V.—VENCKAUSKAS, A.—PASKEVICIUS, R. : Securing Web Application by Embedded Firewall, Electronics and Electrical Engineering **3**(119) (2012), 65–68.

[6] FREIER, A.—KARLTON, P.—KOCHER, P. : The Secure Sockets Layer (SSL) Protocol Version 3.0, www.rfc-editor.org/rfc/rfc6101.txt..

[7] GUPTA, V.—WURM, M. : The Energy Cost of SSL in Deeply Embedded Systems, Sun Microsystems Inc., Tech. Rep. SMLI TR-2008-173, 2008.

[8] KAMEL, M.—BOUDAOUD, K.—LEQUEUX, S.—RIVEILL, M. : Designing Security Protocols Adapted to the Constraints of Mobile Environments, Embedded and Ubiquitous Computing (EUC), IEEE/IFIP 8th International Conference on, 11-13 Dec 2010, pp. 624–629.

[9] PRASITHSANGAREE, P.—KRISHNAMURTHY, P. : On a Framework for Energy-Efficient Security Protocols in Wireless Networks, Computer Communications **27** No. 17 (Nov 2004), 1716–1729.

[10] CANO, M. D.—DOMENECH-ASENSI, G. : A Secure Energy-Efficient m-Banking Application for Mobile Devices, Journal of Systems and Software **84** No. 11 (Nov 2011), 1899–1909.

[11] POTLAPALLY, N. R.—RAVI, S.—RAGHUNATHAN, A.—LEE, R. B.—JHA, N. K. : Configuration and Extension of Embedded Processors to Optimize IPSec Protocol Execution, IEEE Transactions on Very Large Scale Integration (VLSI) Systems **15** No. 5 (May 2007), 605–609.

[12] VENČKAUSKAS, A.—JUSAS, N.—KIŽAUSKIENE, L.—KAZANAVIČIUS, E.—KAZANAVIČIUS, V. : Security Method of Embedded Software for Mechatronic Systems, Mechanika **18** No. 2 (2012), 196–202.

[13] VENČKAUSKAS, A.—JUSAS, N.—MIKUCKIENĖ, I.—MACIULEVIČIUS, S. : Generation of the Secret Encryption Key using the Signature of the Embedded System, Information Technology and Control **41** No. 4 (2012), 368–375.

[14] MARLER, R.—ARORA, J. : Survey of Multi-Objective Optimization Methods for Engineering, Structural and Multidisciplinary Optimization **26** No. 6 (2004), 369–395.

[15] PASTORE, M.—EMMETT, D. : COMPTIA SECURITY+ ST. GUIDE DELUXE,, John Wiley & Sons,, 2006, OpenSSL Software Distribution. http://www.openssl.org/.

[16] TOLDINAS, J.—STUIKYS, V.—DAMASEVICIUS, R.—ZIBERKAS, G.—BANIONIS, M. : Energy Efficiency Comparison with Cipher Strength of AES and Rijndael Cryptographic Algorithms in Mobile Devices, Electronics and Electrical Engineering **2**(108) (2011), 11–14.

[17] KANSAL, A.—FENG ZHAO—JIE LIU—KOTHARI, N.—BHATTACHARYA, A. A. : Virtual Machine Power Metering and Provisioning, In Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC '10), ACM, New York, NY, USA, 2010, pp. 39–50.

Received 15 January 2014

Algimantas Venčkauskas (Doc, Ing, PhD) received his Ing degree in 1979 and PhD in 1987 at the Faculty of Informatics, Kaunas University of Technology (KTU), Kaunas, Lithuania. Currently he is professor of Computer Science and head of Department of Computer Science at KTU. He has published more than twenty papers in various International Journals. The most of his research work was orientated on the study of internet technology and information security.

Egidijus Kazanavičius (Prof, Ing, PhD) received his Ing degree in 1979 and PhD in 1988 at the Faculty of Informatics, Kaunas University of Technology, Kaunas, Lithuania. Currently he is professor of Computer Science at Department of Computer Science and director of Centre of Real Time Computer Systems at KTU. He has published more than twenty papers in various International Journals. The most of his research work was orientated on the study of real time computer systems and information security.

Vytautas Štuikys (Prof, Ing, DrHb) received his PhD degree in 1970 and doctor habilitatis in 2002 from Kaunas University of Technology, Kaunas, Lithuania. He is a professor at Software Engineering Department of KTU. His research interests include software and hardware design methodologies, software reuse, component-based programming, metaprogramming and program generation. He has published more than hundred papers in various International Journals. He is an author of several books and a monograph. He is a member of ACM and IEEE.

Nerijus Jusas (MSc) received his MSc degree in 2012 at the Faculty of Informatics, Kaunas University of Technology, Kaunas, Lithuania. Currently he is PhD student and researcher at Department of Computer Science, KTU. He has published some papers in various International Journals. The most of his research work was orientated on the study of internet technology and information security.