On the Benefits of Elliptic Curve Cryptography and Steganography for the Security of Cyber-physical Systems

Laura Vegh*, Liviu Miclea
Faculty of Automation and Computer Science
Technical University of Cluj-Napoca
Romania
*Email: Laura.Vegh [AT] aut.utcluj.ro

Abstract—**Security is an essential aspect in today's society, in a time when we use technology in almost every aspect of our lives. New systems are discovered everyday, with enhanced and complex features. Cyber-physical systems integrate both physical and computational elements and are gaining more and more terrain. The security of such complex systems is a challenge and classical methods such as cryptography and steganography alone might not be enough. The disadvantages of classic public-key cryptography are said to be eliminated through the use of elliptic curve cryptography. In the present paper we propose a security architecture that combines cryptography and steganography. Furthermore, we modify the original encryption algorithm to use elliptic curves and analyze the performance of the two in an attempt to discover the benefits this new type of cryptography, combined with steganography could have on cyber-physical systems.**

Keywords- cryptography; steganography; cyber-physical systems; elliptic curves

I. INTRODUCTION

A. Overview

Nowadays technology plays an important role in our everyday lives. Whether we think about our day-to-day activities, or our jobs, or the way in which we communicate, technology is present in more than one way. Advances are being made each day in both hardware and software. All these aspects bring to our attention the increased need for security. Like the technologies they are designed for, security architectures need to be in constant research so as to adapt to the new discoveries. Solutions are required for complex systems such as cyber-physical systems which integrate physical and cyber elements, supervisory control and data acquisition (SCADA) systems which are used in areas such as water supply and have to operate many times over large geographical areas. The security of such systems is a crucial aspect. The solution should be complex and robust yet simple so as to not overload or slow down the main operations of the system.

Over the years, the area of security has developed a lot, in an attempt to provide solutions for every possible problem. Probably the most known security method today is cryptography. A simple definition presents cryptography as the art of writing in secret code, due to the fact that its main usage is to change the form of a message in a way that it becomes impossible to understand without the correct key. There are several forms of cryptography, each with advantages and disadvantages - the method to use is chosen based on the systems' constraints and requirements and of course, the data to secure. Each cryptographic method has an area for which it is the best choice, for this reason we cannot discuss of 'the best method'.

Symmetric-key cryptography is a method in which only one key is used for both encryption and decryption. The most known algorithm in this category is the Advanced Encryption Standard (AES). Even though symmetric ciphers are susceptible to several attacks such as differential and linear cryptanalysis, or plain-text attacks, they are still used in several applications. The fact that the same key is used for both encryption and decryption makes them suitable for securing stored data and hardware security.

If the goal is to secure communication between two or more parties, the recommended approach will usually be public-key (or asymmetric) cryptography. These methods require the use of a key, named the public key to encrypt data, and another key, named the private key to decrypt the data. The most known public-key algorithms are RSA, Diffie-Hellman, ElGamal and various elliptic curve techniques. Our solution is based on an extension of the ElGamal algorithm, a method based on the division of the private key. We modified the original form of the algorithm so that it uses elliptic curves.

Many encryption algorithms are based on complex mathematical structures that are easy to compute one way, but almost impossible in reverse, such as the discrete logarithm. While this method is considered secure, safe, its efficiency is

sometimes questioned due to the complexity of the computations performed and also due to the usually large keys used. An answer to these problems could be the use of elliptic curve cryptography (in short, ECC), a type of algorithm based on elliptic curves over finite fields. Their main benefit is the reduction of key sizes, thus reducing storage requirements and computational times, while ensuring the same level of security as other consecrated public-key type of algorithms, such as RSA.

A different security method, that has made a comeback during the past years, is steganography. Unlike cryptography, steganography hides the secret data, it does not modify its form at all. There are several ways one can apply steganography, the best one depends on the type of data that needs to be hidden and on the system's specifications. The main differentiation between steganographical methods is brought by the cover file used - image, video or audio files, text files, html and so on. For each type of cover file there are several methods that allow a user to hide the secret message. The main goal of all these methods is to hide the data in such a way that the changes done to the cover file are not visible to the naked eye. For example, for image steganography, the most used method and the one we will use to increase the security of our cyber-physical system is the least significant bit (LSB) method. This method implies, as the name suggests it, modifying the least significant bit of an image by inserting the secret data. The changes done to the original cover file will be minimal. Literature offers us several works that illustrate the use of steganography is various systems and also algorithms meant to improve consecrated methods such as LSB [2][5][6][7].

An approach more and more present in complex information systems is combining steganography with cryptography. There are several ways to combine the two methods. For example, an encrypted message cannot be read without the proper key for decryption. However, it can still be intercepted, as an eavesdropper for example, will easily see that secret code is being transmitted on a certain communication channel. By hiding the encrypted data in a cover file that will not raise any suspicions in the context of the system, the chances of the data being intercepted decrease. And even in the scenario that one would want to analyze that cover file, they would have to perform many more computations to get to the actual data. This is beneficial both because it increases the security of the system and also because it increases the time one has to react in case the secret data has been intercepted. A hacker should first find the secret data, then find a way to extract it. Once the message is extracted, all the attacker has is a cipher that needs to be decrypted before the message is readable.

B. Cyber-Physical Systems

A new paradigm that has gained more and more terrain are cyber-physical systems. Integrating both physical and computational elements, CPS are complex systems, used in a variety of applications. To name a few of the areas of applications, CPS are used in healthcare, power and energy systems, water supply systems, aviation and so on [10]. The critical nature of the applications in which CPS are used increases the need for a robust secure architecture. There are several approaches of CPS security that can be found in literature. Most of them are focused on certain aspects of the system, such as hardware security, or communication security. Other architectures are modeled according to the area of application of the CPS, an approach that always brings forth unique aspects of a system [3]5][12]. There are those who claim that focusing on one or two components at a time is not ideal as it leaves room for mistakes, for unprotected areas that could compromise the system [11]. While this can be true, we should also look at the large number of components CPS usually have and at the fact that sometimes these components are spread over a wide geographical area. These aspects often require different approaches in the area of security. Furthermore, working with each component separately, allows us to see the smaller details which we might overlook if we worked with the entire system at once.

In the present paper we propose a security model for cyber-physical system that includes both cryptography and steganography. We model the CPS as a multi-agent system, a common approach, as agents are autonomous components with decision making capabilities and their systems are decentralized, making them a suitable choice for modeling CPS. In Section 2 we describe the cryptography module - we present first the original encryption algorithm and its implementation and we later describe the changes made in order to use elliptic curve cryptography instead of classic cryptography. In Section 3 we describe the steganography module, its usage and we compare the performances of combining this method with classical cryptography as opposed to ECC. In Section 4 we analyze the results and finally we draw the conclusions in Section 5.

II. CRYPTOGRAPHY MODULE

A. Hierarchical system

Having in mind the complexity of cyber-physical systems, we wanted to design a security architecture that would focus on communication, integrity and access control. The chosen approach has been to create a hierarchical system, in which each user has access only to certain data. These systems are very practical in many applications - for example a medical facility. The approach allows one to create access rights, thus ensuring not only a greater level of security but also confidentiality.

The particularity of the proposed system is that the hierarchy is given by the private key generated by the encryption algorithm. An extension of the ElGamal algorithm, ElGamal with (k+1) degrees of access, has the particularity that the private key is divided in such as way that each user can only access certain messages. From the way the keys are generated, the result is a tree structured hierarchy. The root of the tree has the highest degree of access, while the leaves have the lowest degree of access. A particularity of the system is that encryption is performed by the leaves. In order to decrypt

a message, a user has to be a direct ascendant of the leaf who encrypted the message.

In terms of implementation, we have used the Java language, with the JADE platform for multi-agent systems. As most encryption algorithms, ElGamal with divided private key has three main phases: key generation, encryption and decryption. The classical algorithm requires the usage of large numbers, of up to 1024 bits. Working with such numbers can prove to be a challenge. The Java language provides a particular type for numbers that reach 1024 bits - the *BigInteger*. This data type proved to be very helpful in the implementation of the encryption algorithm, as it has integrated methods for modulo calculations, a crucial aspect for the ElGamal algorithm as most calculations are performed modulo. Another aspect to note is that we divided the application in two packages: one where the algorithm computations are performed, with a class for key generation, another for encryption and one for decryption; another package where the classes for the CPS can be found. This division makes the application flexible, the algorithm and the system packages can be used separately, a property that proved particularly useful when making changes to the encryption algorithm.

Regarding the system itself, a crucial step towards creating a secure, robust system when using a public-key type of algorithm, is the key generation and distribution. To ensure better confidentiality and access control, generating and distributing the keys is performed by a user outside of the tree structure. In other words, this user - that we will call 'key manager' for further reference - will not be a part of the functional CPS. Furthermore, the key manager is responsible with starting the system - he knows the exact number of users that will be in the system, the number of levels - the value of the variable *k* - and how the system is structured, how many users there are on each level. With this data he proceeds on to computing the keys and distributing them to their rightful owners. The public key is sent only to the leaves, while each user in the system will receive a private key. Since the key distribution step is of utmost importance, we added a steganography module to help protect the keys while they are distributed. We will present this module in detail in Section 3 of the present paper.

B. Elliptic Curve Cryptography

Elliptic curve cryptography aims at eliminating issues that arise in classical cryptography, such as large numbers and slow computation times. An elliptic curve is defined as the set of all points $(x, y) \in (Z_p \times Z_p)$ satisfying the equation:

$$y^2 = x^3 + a \cdot x + b \bmod p \quad (1)$$

Where $a, b \in Z_p$ such that:

$$4a^3 + 27b^2 \neq 0 \bmod p \quad (2)$$

We wanted to keep the hierarchical structure of the system, as a result, the way in which the private keys are computed needed to remain the same. As with the standard algorithm, we are working again with the data type *BigInteger* to represent the private keys. However, this time the keys will be of much smaller size. The public keys are points on the elliptic curve and are therefore represented in Java using the data type *Point*. Regarding the choice for *BigInteger* to represent the private keys, it should be mentioned that elliptic curve cryptography allows for real numbers to be used, it does not require integers. In Java for large numbers the equivalent would be *BigDecimal*. However, we wanted to remain with integers so that we would have the same functions for the modulo operations that we had in the previous implementation. This will allow for a comparison between the two methods focused solely on the key sizes, without risking the results to be altered by the data types used. Keeping the mathematical formulas used to compute the hierarchical private keys was relatively easy since in elliptic curve cryptography the private keys are not points, they are simple numbers. Usual ECC implies that each user generates his private key randomly, an action not allowed in our system, as the keys need to be generated by the same user, the one that gives the access rights.

The first step towards an elliptic curve algorithm implementation is to choose the values of *a* and *b* as presented in equation (1) and (2). Next we choose a point *g*, the generator, on the curve. Finally, we choose a prime number *p*. Computing the private key is done respecting the mathematical formulas requested by the original algorithm [9]. For example, for level 0 the key will be:

$$f^0 : \mathrm{N}^* \times \mathrm{N}^* \times \ldots \times \mathrm{N}^* \to \mathrm{Z}_q^*$$

$$f^0(n_1, n_2, \ldots, n_k) = x_1^{\theta_1(n_1)} x_2^{\theta_2(n_2)} \ldots x_k^{\theta_k(n_k)} \quad (3)$$

For level 1:

$$f_1^1 : \mathrm{N}^* \times \mathrm{N}^* \times \ldots \times \mathrm{N}^* \to \mathrm{Z}_q^*$$

$$f_1^1(n_2, n_3, \ldots, n_k) = x_1^{\theta_1(1)} x_2^{\theta_2(n_2)} \ldots x_k^{\theta_k(n_k)} \quad (4)$$

$$f_2^1 : \mathrm{N}^* \times \mathrm{N}^* \times \ldots \times \mathrm{N}^* \to \mathrm{Z}_q^*$$

$$f_2^1(n_2, n_3, \ldots, n_k) = x_1^{\theta_1(2)} x_2^{\theta_2(n_2)} \ldots x_k^{\theta_k(n_k)} \quad (5)$$

The public key is computed according to the requirements of the elliptic curve cryptography, by multiplying the point G with the user's private key. Since only the leaves can perform encryption, they are the only ones for whom a public key is generated. To note that all computations are performed modulo p.

When working with elliptic curves, a special attention needs to be given to the way the operations are implemented. More specifically, the point addition and multiplication, are the two operations that are required several times throughout an encryption algorithm using elliptic curves. They are more or less the same with every ECC algorithm, their implementation could aid to the efficiency of the algorithm. Also, on the note

of implementation, generating the elliptic curve and all its points and parameters is a task performed by the key manager.

III. Steganography Module

In a time when hackers get bolder each day, even the safest encryption systems could prove to be at risk. Steganography adds a layer to the security architecture, hiding data under cover files chosen in such a way as not to raise suspicions.

The system described in the previous section is based on the encryption algorithm ElGamal with (k+1) degrees of access. Whether in its classical form, or using elliptic curves, the initial implementation left the door open for unwanted attacks. The most significant vulnerability is found in the key distribution step. Keys are being sent from one agent to another in the form of simple messages. Even though the system is not considered started yet, it is not functional, this vulnerability is still important and should be addressed. The solution was found in the form of steganography. Cover files can have various forms and they should be chosen according to the system's requirements and restrictions. We chose image steganography, more specifically, the least significant bit (LSB) method.

The LSB method is fairly easy to use and it is a common method in image steganography. The fact that it is so well known and commonly used is probably its main disadvantage. However, when choosing a security method one should also consider the context. In our case, we work with a system secured through a robust encryption algorithm. Furthermore, the operation we are trying to secure through steganography takes place before the system is functional, an aspect that reduces the chances of an attack. Also, we need to consider the security architecture is designed for a complex CPS, with many different tasks. For this reason, we should not overload the system with tasks that could be performed in a more efficient manner. All these aspects make the LSB method the perfect choice to hide the secret keys. In an attempt to eliminate any redundant operations, we only hide the private keys, the public keys do not affect the security or the structure of the system if intercepted.

In the classical form of the algorithm, private keys are large numbers of up to 1024 bits, while in the modified form, using elliptic curve cryptography, the private keys are numbers can have a maximum of 521 bits, with a minimum of 192 bits according to NIST standards [8]. For our tests we chose keys of size 256 bits. The embedding and extraction operations, through the LSB method, do not vary with the size of the data. However, in the case of numbers that can reach 1024 bits we concluded after several tests it was necessary to also include in the cover the exact number of bits the key has. Without this operation, in 3 out of 50 tests we reported errors at the extraction step. While a 0.06% error rate could be considered acceptable in some cases, when the functionality of the system relies so heavily on the private keys, no error can be allowed. In the case of elliptic curve cryptography things are easier: there is no need to embed the size of the key, no errors were reported in any of the tests. It can be observed that already elliptic curve cryptography eliminated an operation, thus increasing the efficiency of the key distribution step.

With regards to the cover file, or the 'stego' file as it is called in literature, once the data has been embedded there are no noticeable differences to the naked eye. Figure 1 compares two stego files, originating from the same image, one in which we have a 1024 bits key embedded and the other with a much smaller key. The two images look the same.

Figure 1. Stego images - on the left with 1024 bits embedded and on the right with 256

IV. Results and Analysis

For the tests we have used several systems, some with more users, and some with less. We wanted to compare the efficiency of the classical algorithm ElGamal with (k+1) degrees of access, using keys of 1024 bits to the modified version using elliptic curves and keys of 256 bits. Our interest was focused mainly on the key generation and distribution aspect, but also on encryption and decryption. The steganography module was tested separately. Key generation and distribution is co-dependent with the number of levels and of users in the system. Therefore we performed tests for several structures of the cyber-physical system. Encryption and decryption might depend on the size of the messages, therefore we tested the two algorithms on different text lengths. Finally, we wanted to see how the steganography module might be affected by different sizes of the private keys. This module does not depend on the number of users in the system, as we are only interested in the embedding and extraction operations in one image.

We begin by testing the execution times of the key generation step, which is dependent upon the cyber-physical system's structure. For further record, all measurements are done in milliseconds. Figure 2 presents the comparative results of our tests using different systems structures - from left to right we have a system of 7, 15, 31, 63, 84 users respectively.

Analyzing the chart, the efficiency of elliptic curve cryptography is easy to observe. Regardless of the number of users in the system, ECC always yields faster results. To note, we perform the same tests with the same number of users but

different system structures, wanting to see if this would affect in any way the execution times. For example, for a system with 31 users, we first tested a structure with 5 levels (k=4) and 16 leaves. A second test was conducted for the same number of users but for a 6 levels (k=5) and 12 leaves. The execution times were practically the same - with classical cryptography there was a difference of 2 milliseconds, which is considered an acceptable error margin, while in the case of ECC there were no differences.

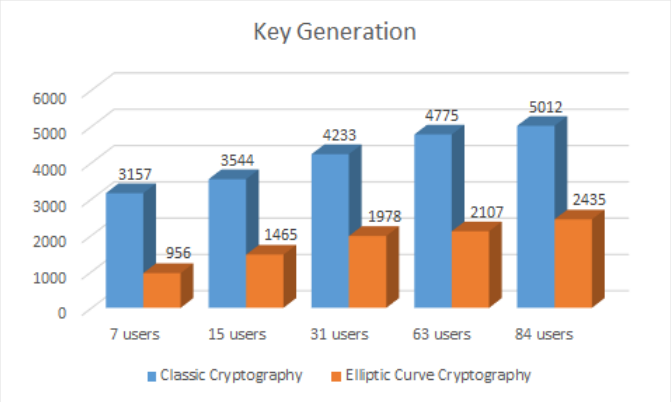

Figure 2.Comparison between key generation execution times

Figure 3 presents the comparison of encryption times of the two types of cryptography, while Figure 4 describes the decryption times. We used several lengths of text. The number of characters shown is that of the original plain text. The tests included in the charts are performed on a system of 6 levels with 63 users of which 32 are leaves. An important aspect of the algorithm used to secure our CPS is that all information is viewed as a set of messages. Before this information is sent to the leaves for encryption, it is first divided in a number of messages equal to the number of leaves. If the character count cannot be divided in equal parts for each leaf, the last leaf will receive all the remaining characters. For example, if we have a string of data of 650 characters, that we want to send to our 32 leaves, we have to divide the data in 32 messages. This will mean 32 messages of 20 characters plus a remainder of 10 characters. These 10 characters are left in the last message, thus the last leaf will receive 30 characters in this case. If this is the case, encryption times for the last leaf will be slightly greater than for the rest of the leaves. In the figures bellow, we averaged the execution times for all the leaves - the differences have always remained within 10 milliseconds and are therefore negligible.

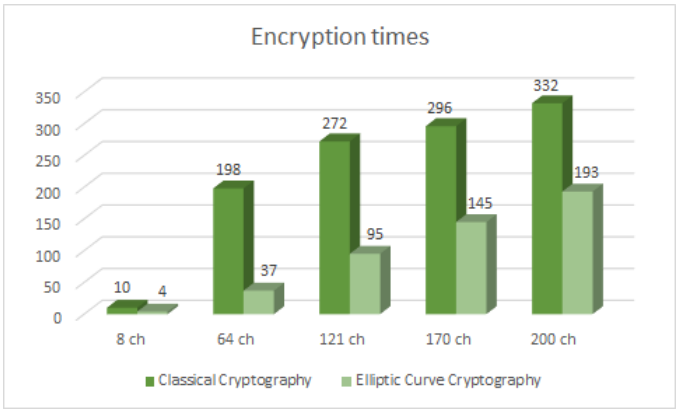

Figure 3.Comparison between encryption execution times

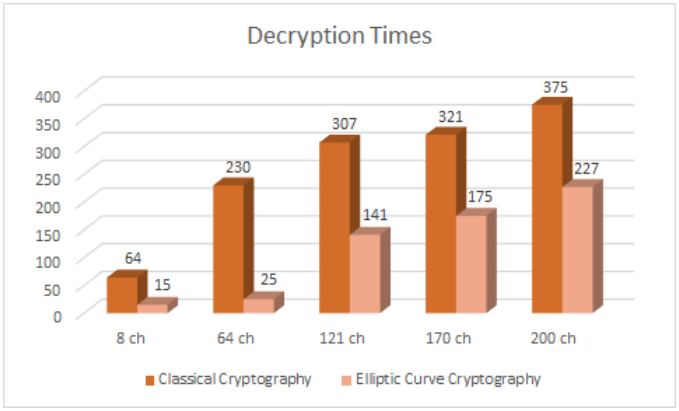

Figure 4.Comparison between decryption execution times

Again, the efficiency of ECC is demonstrated in both cases, regardless of the number of characters to be encrypted and/or decrypted. Decryption execution times are slightly bigger than those of encryption for both types of cryptography. This is common with many algorithms, including for example RSA. This is usually due to the computations required for the two operations, for instance modular exponentiation which will be easy and fast to compute one way but much harder in reverse. We will however not dwell further into these details as they do not consist the main subject of our paper. To note, ElGamal with (k+1) degrees of access can be used for texts larger than 200 characters. We believe however, that the tests performed are sufficient to prove the efficiency of elliptic curve cryptography as opposed to classic cryptography in the context of the hierarchical system.

Our final performance analysis concerns the steganography module. We are interested in seeing how the embedding and extracting phases respectively are affected by the use of classical cryptography versus elliptic curve cryptography. As mentioned in the previous section, we only hide the private keys of the system. This means we are going to analyze how embedding and extracting 1024 bits of data will perform versus the same operations for 256 bits of data. Figure 5 presents the results.

We tested on an image of 1728 X 2592 which will have 4478976 pixels, or for a "truecolor" type of image, having 24 bits per pixels, we have a total of 107495424 bits to hide our data. The chart in figure 5 proves that elliptic curve cryptography improves the efficiency of the steganography module. To note, we use a 3 bit LSB algorithm. Tests [13] have proven the 2 bit LSB is faster than 3 bit LSB. However, the same paper proves data extraction with 3 bit LSB is far more precise than in the case of 2 bit LSB. Even though we are embedding a rather small amount of data as compared to what the image used could hide, we still want to make sure there will be no errors in the extraction phase, as the structure and security of the entire cyber-physical systems depends on it.

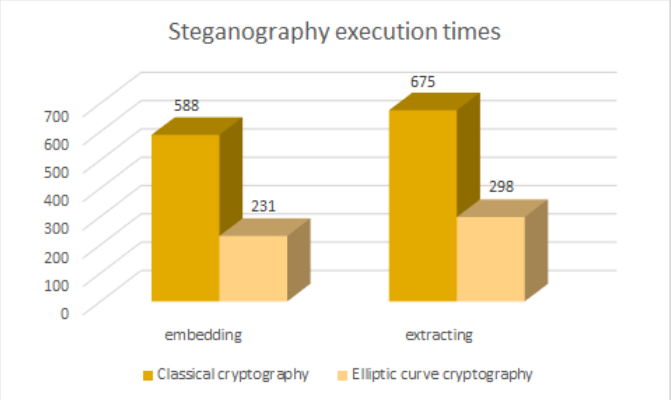

Figure 5. Comparison between the execution times of the embedding and extraction phases of the LSB steganography module when using classical cryptography with 1024 bit keys as opposed to elliptic curve cryptography with 256 bit keys

V. CONCLUSIONS

We described an architecture that used both cryptography and steganography to ensure the security of a cyber-physical system. Furthermore, due to the use of an encryption algorithm with divided private key - ElGamal with (k+1) degrees of access - we ensure not only the security of communications, but also a better control of the access to the system and to data exchanged within the system.

In the light of recent research, that claims elliptic curve cryptography is the solution to certain disadvantages of classical cryptography such as slow performance due to the large numbers used as keys and the great number of computations required, we wanted to analyze how CPS would benefit from ECC. In order to do this, we first modified the original encryption algorithm, ElGamal with (k+1) degrees of access so that it uses ECC. We implemented and tested the two algorithms using the same computer, thus same processor and same amount of RAM and also the same software and programming language, so that the external environment could not influence the results. The elliptic curve cryptography proved its efficiency in all three stages of the algorithm: key generation, encryption and decryption, by yielding significantly lower execution times. The same can be said for the steganography module, where we wanted to see if the system would be more efficient when embedding 256 bit private keys instead of the 1024 bit keys of classical cryptography. The execution time of both the embedding and the extraction phases decreased when using ECC. Using ECC and steganography to secure cyber-physical systems has proven without a doubt its benefits. CPS are complex systems, used in many critical applications where security is of utmost importance, but so is the overall performance of the system. By using ECC, the performance of the security module is increased, thus leaving space for the different operations the CPS should perform.

ACKNOWLEDGMENT

This paper is supported by the Sectorial Operational Program Human Resources Development (SOP HRD), ID/134378 financed from the European Social Fund and by the Romanian Government.

REFERENCES

[1] Motasem A. Abu-Dawas, Abdulameer K. Hussain, Enhancement of RSA Scheme using Agreement Secure Information for Nearest Parameters, International Journal of Computer and Information Technology, Volume 04-Issue 02, March 2015

[2] Hafid Mammass, Fattehallah Ghadi, Mohamed Elhajji, Secure Watemarking Method with Smart Card, International Journal of Computer and Information Technology, Volume 02-Issue 05, September 2013

[3] K. Usman, S. Aleksandar: Security in cyber-physical energy systems, Workshop on Modelling and Simulation of Cyber-Physical systems (MSCPES), 20-23 May 2013

[4] Riadh W. Y. Habash, Voicu Groza, Kevin Burr: Risk Management for Power Grid Cyber-Physical Security, British Journal of Applied Science and Technology, Volume 3, Issue 4, July 2013

[5] A. H. Ibrahim, W. M. Ibrahim: Text Hidden in Picture using Steganography: Algorithm and Implications for Phase Embedding and Extraction Time, International Journal of Information Technology and Computer Science, Volume 7, No. 3, January/February 2013.

[6] Mamta Juneja, Parvinder Sandhu: An improved LSB based Steganography with Enhanced Security and Embedding/Extraction, 3rd International Conference on Intelligent Computational Systems, Hong Kong, China, January 2013

[7] G. Padmashree, P. S. Venupogapala: Audio Steganography and Cryptography: Using LSB algorithm at 4th and 5th LSB layers, International Journal of Engineering and Innovative Technology, Volume 2, Issue 4, October 2012.

[8] E. Barker, W. Barker, W. Burr, W. Polk, M. Smid: Recommendation for key management - Part 1: general (revision 3), Computer Security, NIST Special Publication 800-57, July 2012

[9] S. Flonta, V. V. Patriciu, L. C. Miclea: Metode criptografice pentru sisteme structurate, U.T. Press, Cluj-Napoca, Romania 2011

[10] E. Wang, Y. Ye, X. Xu, S. Yiu, L. Hui, K. Chow, Security issues and challenges for cyber physical systems, in Green Computing and Communications (GreenCom), 2010 IEEE/ACM Intl Conference on Intl Conference on Cyber, Physical and Social Computing (CPSCom), Dec 2010, pp. 733738.

[11] T. Gamage, B. McMillin, T. Roth: Enforcing information flow security properties in cyber-physical systems: A generalized framework based on compensation, in Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual, July 2010, pp. 158163

[12] Soon M. Chung: Role-Based Access Control for Cyber-Physical Systems Using Shibboleth, Washington University, June 2009

[13] D. Stanescu, V. Stangaciu, I. Ghergulescu, M. Stratulat: Steganography on embedded devices, Applied Computational Intelligence and Informatics, 2009. SACI '09. 5th International Symposium on , vol., no., pp.313,318, 28-29 May 2009