UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingeniería

CI: A New Encryption Mechanism for Instant Messaging in Mobile Devices

Iván Mauricio Del Pozo Falconí

Ingeniería en Sistemas

Trabajo de titulación presentado como requisito
para la obtención del título de
Ingeniero en Sistemas

Quito, 10 de julio de 2015

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

COLEGIO DE CIENCIAS E INGENIERÍA

HOJA DE CALIFICACIÓN
DE TRABAJO DE TITULACIÓN

CI: A New Encryption Mechanism for Instant Messaging in Mobile Devices

Iván Mauricio Del Pozo Falconí

Calificación:

Nombre del profesor, Título académico Mauricio Iturralde , Ph,D.

Firma del profesor ________________________________

Quito, 10 de julio de 2015

Derechos de Autor

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Firma del estudiante: ____

Nombres y apellidos: Iván Mauricio Del Pozo Falconí

Código: 00103773

Cédula de Identidad: 1719810622

Lugar y fecha: Quito, julio de 2015

Agradecimientos

Agradezco a mi familia por siempre estar pendiente de mí y por el apoyo incondicional que me brindaron a lo largo de toda mi formación universitaria y principalmente a mi padre y mi madre, motores responsables de muchas de mis metas alcanzadas, ya que soy quien soy gracias a ellos.

A Mauricio Iturralde, mi director de tesis, por fomentar en mi la investigación y además enseñarme lo fascinante que puede llegar a ser nuestra carrera.

A los profesores que son parte del Departamento de Ingeniería en Sistemas y a todos los que alguna vez tuve la dicha de ser su alumno dentro de la Universidad, los cuales aportaron no solo en mi formación académica sino también personal.

A mis amigos cercanos y compañeros universitarios quienes compartieron conmigo días y noches difíciles de estudio y sacrificio, quienes también me alentaron a nunca desertar en ningún momento.

RESUMEN

Gracias al Internet hoy en día nos podemos comunicar fácilmente con cualquier persona de cualquier parte del mundo, mediante algún tipo de mensajería instantánea. La mensajería instantánea en dispositivos móviles puede ser considerada uno de los servicios más utilizados en las comunicaciones móviles. Muchas de estas comunicaciones manejan información y conversaciones delicadas. Por ello surge la necesidad de cifrar y ocultar cierta información, para maximizar la confidencialidad y seguridad de los usuarios. Actualmente existen algoritmos los cuales algunos han sido vulnerados, otros son robustos pero no son rápidos y otros que no son robustos, son rápidos pero los hace vulnerables para que terceros puedan saber su contenido. A continuación se presenta un algoritmo de cifrado simétrico, orientado para el uso en una aplicación de mensajería de texto instantánea, el cual envía toda la información mediante la sucesión de números primos formados a partir de una matriz bi-dimensional y una clave secreta. La solución propuesta se ha comparado con otros algoritmos conocidos como DES, RSA, MD5, AES. Los resultados demuestran que el mecanismo es adecuado para el cifrado de los dispositivos móviles de mensajería instantánea de texto debido a su baja complejidad, rendimiento y robustez.

Palabras clave: algoritmo, cifrado, descifrado, confidencialidad, información, mensajería.

TABLE OF CONTENTS

1._ Introduction **7**

2._ Literature Review **8**

3._ Proposed Encryption Mechanism **8**

3.1._ Proposed Cryptographic Algorithm **8**

3.1.1._ Keyword analysis **8**

3.1.2._ Analysis of the list of prime numbers and the list of characters belonging to the dictionary. **11**

3.1.3._ Analysis of the message to encrypt **11**

3.2 Complexity Analysis **12**

3.3 Encryption Protocol **12**

4._ Numerical Results **13**

5._ Conclusions and future Work **14**

References **14**

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 00 (2014) 000–000

Procedia
Computer Science

www.elsevier.com/locate/procedia

International Workshop on Mobile Computing Security, (MCS 2015)

CI: A New Encryption Mechanism for Instant Messaging in Mobile Devices

Ivan Del Pozo and Mauricio Iturralde

Universidad San Francisco de Quito, USFQ

Abstract

Instant Messaging in mobile devices can be considered one of the most used services in mobile communications. Security properties such as integrity and confidentiality must be maximized. This paper proposes a new symmetric encryption mechanism for instant text messaging in mobile devices. Our mechanism uses a sequence of prime numbers obtained from a bi-dimensional matrix and a secret key for the encryption process. The proposed solution has been compared with other well-known algorithms such as DES, RSA, MD5, AES. Results show that our mechanism is suitable for the encryption of instant text messaging in mobile devices due to its low complexity, performance, and robustness

Keywords: algorithm; encryption; decryption; confidentiality; Instant Messaging;

1. Introduction

Instant Messaging (IM) is one of the most used services in mobile devices[22]. Users tend to transmit all types of information including credit card and bank account numbers by using this service. Attackers consider IM services as a rich source for information for stealing. Sniffing is a common method for catching the instant messaging service that communicates through the network. Therefore, in order to provide to users a secured IM service, properties such as integrity and confidentiality must be maximized[5]. Symmetric and Asymmetric cryptography are mostly used for this task. Although cryptography is a good solution for secure communication in computer science, there are several constraints that need to be taken into account for mobile devices communications. For instance, in order to preserve the battery life of the device, the encryption computational complexity must be minimized. On the other hand, in order to make it harder to break the encryption, the algorithm robustness must be maximized.

A secured text-based communication in mobile devices must focus on maximizing the confidentiality and the data integrity, while, the encryption/decryption computational complexity must be minimized. This paper proposes a new encryption method for instant text messaging application in mobile devices. Our proposed mechanism is based on symmetric encryption and presents a low algorithmic complexity. The following paper is organized as follows:

E-mail address: ivan.delpozo@estud.usfq.edu.ec; miturralde@usfq.edu.ec

1877-0509

In Section 2, a quick review of previous works is presented. In Section 3 we introduce our proposed encryption mechanism which is composed of the ciphering algorithm and the hand-shake protocol. In section 4 we compare our proposed solution against existing algorithms, and discuss the results. Finally, Section 5 concludes this paper.

2. Literature Review

In order to generate a secure data transmission in an IM service, a secure channel using any of the cryptographic techniques must be implemented[7]. The communication between two devices must be understood by the receiver and must be the same as that of the sender. Furthermore, the communication should be encrypted to prevent unauthorized access. The text transmission between two devices must be read and understood only by theinvolved devices. Most of the current algorithms are based on the concepts of confusion and diffusion developed by Claude Shannon on Information Theory in the forties[3]. Several studies have reported different ways of coding, highlights of symmetric and asymmetric cryptography. The most relevant existing cryptography with their features and functionalities, such as Feistel networks[5], DES[5 6 7], DES Multiple[6 7], AES[4 7], Rijndael[4], IDEA[7], RSA[15 19], MD5[12], SHA[13 17].

Nowadays, mobile messaging services is performed by several applications which use different cryptographic solutions such as:

- WhatsApp: This system is called end-to-end, which encrypts the messages when you send it and decrypts, when the recipient receives it. It operates with a simple MD5 hash function of the IMEI number of the mobile turned upside down. An important aspect of WhatsApp HMAC protocol is that it has no sequence number.[21]
- BBM: It uses a point-to-point encryption for both users and content, thus ensuring security. Messages are protected and privacy too. This mechanism uses AES, S/MIME (Secure MIME), or Triple DES as a key encryption algorithm for data encryption and combines all three together, sending the data in the format PKCS No 7. All information is handled through RIM servers. Blackberry holds the best patents in safety.[20]
- Telegram: Telegram implements a proprietary protocol, MTProto, which transmits messages securely between our mobile and server. It allows the creation of a safe chat between two clients, so that, even Telegram servers cannot see what is being sent, they only know that there is traffic. It is encrypted using symmetryc cryptography. All the encryption are based on the DiffieHellman implementation, which is essentially mathematical tricks, with groups of multiplicative integers module p, where $p = \mathbb{N}primes$. They argue saying that it is very easy to operate a with b in order to get c, but from c it is very difficult to know what numbers a and b has generated.[22]

3. Proposed Encryption Mechanism

This mechanism focusses on reaching a trade-off between robustness and low complexity for Instant Text Messages in mobile technology transmission.

3.1. Proposed Cryptographic Algorithm

Consider the dictionary $\mathbb{D}$, which is composed by a list of characters $\mathbb{C}$ and a list of prime numbers $\mathbb{P}$, so that $\mathbb{C} \equiv \mathbb{P}$ where $\mathbb{P} : [\mathbb{N}primes]$ and $\mathbb{C} : [A, B, C...Z] \wedge [a, b, c...z] \wedge [0, 1, 2...9] \wedge$ [, , ., ; , :, −, ?, , , !,′ , ”, @, +, ∗, /, =, |, (,), „ [,],]
The following operations are performed to encrypt the message.

3.1.1. Keyword analysis

We define our secret password as ***key*** in order to establish a secure communication between users. ***total*** is defined as the number equivalent to the sum of the translation of the key's characters. Then let us consider a ***keyNumber***, the result of ***total*** module cardinality of ***key*** and ***calculation*** as the number obtained as a result of ***total*** divided by ***keyNumber***.

$$\forall\ key\ \in\ \mathbb{P}\ \exists\{total \mid total = \textstyle\sum key\} \wedge$$
$$\exists\{keyNumber \mid keyNumber = total\ mod\ |key|\}\ \wedge\ \exists\{calculation \mid calculation = \frac{total}{keyNumber}\}$$

Depending on the values of ***calculation***, we are going to have two different cases, each one with two immersed sub-cases.

- Case A: If $calculation\ mod\ 10 \geq 3 \Rightarrow calculation\ mod\ 10$; If $calculation\ mod\ 10 < 3 \Rightarrow 5-(calculation\ mod\ 10)$
- Case B: If $\frac{calculation}{10} \geq 3 \Rightarrow \frac{calculation}{10}$; If $\frac{calculation}{10} < 3 \Rightarrow 5 - \left(\frac{calculation}{10}\right)$

These calculations are performed in order to analyse each digit of the numerical value of ***calculation.*** This logical comparison $< 3; \geq 3$ for both module operations and the division, is performed in order to have a 101 character dictionary. The prime numbers that will be generated are between 0 and 523, which means that it will be a 3 digit number. If the dictionary increases and we have a thousand units, then the logical comparison must be $< 4; \geq 4$. Otherwise information would be lost, and it would be impossible to convert all characters to their equivalent in $\mathbb{P}$ Given the previous arguments, it can be concluded that:

$$\forall\ key\ \in\ \mathbb{P}\ \exists\{calculation\}\ \exists\{(Case\ A = calculation\ mod\ 10) \leftrightarrow ((calculation\ mod\ 10) \geq 3)\ \vee (Case\ A = 5 - (calculation\ mod\ 10)) \leftrightarrow ((calculation\ mod\ 10) < 3)\}\ \exists\{(Case\ B = (calculation/10)) \leftrightarrow ((calculation/10) \geq 3)\ \vee (Case\ B = 5 - (calculation/10)) \leftrightarrow ((calculation/10) < 3)$$

Now let us define ***spaceNumbers*** a subset of $\mathbb{N}$, where $spaceNumbers \in [3, 9]$. The cardinality is given by the numerical value of ***keyNumber***. The values that compose it are given by the results of Case A and Case B previously discussed. So we can say that:

$$\forall\ key\ \in\ \mathbb{P}\ \exists\{spaceNumbers\ \epsilon\ \mathbb{N}\ \geq 3 \wedge \leq 9\}\ \exists\{|spaceNumbers| = keyNumber\} \Rightarrow [(spaceNumbers_i = A) \leftrightarrow (|calculation| < 3)]\ \vee\ [(spaceNumbers_i = (Case\ A\ \vee\ Case\ B\ \vee\ (Case\ A \wedge Case\ B))) \leftrightarrow (|calculation| \geq 3)]$$

The average of the number of characters that people tend to select for their passwords is between 8 and 16. They are composed of numbers, letters, and special characters [10]. Based on the average password characters cardinality (we choose 12), this example shows that:

$$If\ k\ \wedge\ n\ \in\ \mathbb{N}\ |\ n = |\mathbb{C}|\ \wedge\ k = |key| \Rightarrow \binom{n}{k} = \binom{101}{12}$$
$$\frac{n!}{k!(n-k)!} = 3,2485E + 287$$

Which represents by binomial coefficient [15] the number of variations that the key can have using just 12 characters from the dictionary $\mathbb{N}$ All the variables will depended directly on ***key.*** Using a larger key will produce a safer message.

3.1.2. Analysis of the list of prime numbers and the list of characters belonging to the dictionary.

Based on the previous analysis and taking into account that our dictionary has 101 characters. The value of $\mathbb{P}$ is equal to the value of $\mathbb{C}$. Therefore, we have a high number of permutations :

$$\forall\ \mathbb{P}\ \wedge\ \mathbb{C}\ \in\ \mathbb{D}\ \exists\{n\ \in\ \mathbb{N}\}\ \exists!\{(\mathbb{P} = n)\ \equiv\ (\mathbb{C} = n)\}$$
$$\Rightarrow 101! = 9,426E + 159$$

This is an interesting fact, because it represents the possible ways that a character can be ordered and assigned to each prime number. If $\mathbb{D}$ tends to be larger, then there will be a greater number of combinations with their equivalents $\mathbb{P}$, which decreases the probability of guessing which character is related to each number.

3.1.3. Analysis of the message to encrypt

Based on the analysis of section 3.1.1,the message to be encrypted ***message***, is a subset of the dictionary $\mathbb{D}$ and the two-dimensional matrix ***A*** of order ***n = 2***, ***m=*** $\frac{|message|}{2}$. Where each character of the message will be stored, so that:

$$\forall\ message\ \in\ \mathbb{D}\ \exists\{|message|\ \wedge\ A_{n*m}\ |n = 2\ \wedge\ m = \frac{|message|}{2}\}$$

Based on this definition, we perform the operation ***module 2***, to the cardinality of ***message***. This way, we determine whether the number of characters entered is an odd or even number. The result of this operation tells us if the last matrix element of ***A*** will store a value or will be *NULL*, so that:

$$\forall\ message\ \in\ \mathbb{D}\ \exists\{A_{n*m}|(a_{2\frac{|message|}{2}} = \mathbb{N}) \leftrightarrow (|message|\ mod\ 2 = 0)\ \vee\ (a_{2\frac{|message|}{2}} = \emptyset) \leftrightarrow (|message|\ mod\ 2 \neq 0)\}$$

Once the size of the matrix ***A*** and the value of the latest matrix element is known, each matrix element is filled from left to right, character by character. Each character is placed in a unique space of the matrix respecting the spaces and punctuations of the ***message***. Once the matrix ***A*** holds the whole message, it translates character by character to its equivalent in the list of prime numbers $\mathbb{P}$. Then ***A*** will be composed of numbers. The ***keyNumber*** value will be added to each value contained by ***A***, which results in:

$$\forall\ message\ \in\ \mathbb{D}\ \exists\{A_{n*m}\} \wedge \exists!\{key\ \wedge\ keyNumber\ |a_{ij} = a_{ij} + keyNumber\}$$

The values to recover from the set ***spaceNumbers***, represent the cardinality of the digits of each matrix element of A, in descending order, like $A_{1,1},\ A_{2,1},\ A_{1,2},\ A_{2,2}...\ A_{n-umpteenth,\ m-umpteenth}...A_{2,\frac{|message|}{2}}$ due to ***spaceNumbers*** cardinality is lower than the amount of elements in the matrix A. When all the numbers are used, ***spaceNumbers*** will use once again (in the same order) the number of the digits of each matrix element of ***A***. Not all the elements of matrix ***A*** have the same number of digits, so:

- Case E: If $|a_{ij}|\ < spaceNumbers_i \Rightarrow$ Zeros will increase to the left until $|a_{ij}| = spaceNumbers_i$
- Case F: If $|a_{ij}|\ = spaceNumbers_i = |a_{ij}|$

So, if an attacker eventually intercepts the message and performs a statistical analysis in order to find any logical pattern, by having different sizes for each element of the matrix, this process can be avoided. Each element of the matrix is differently compounded. In this way, each character is composed by a different number of digits depending on the corresponding sub index number of $spaceNumbers_i$, even if it is the same letter. Thus, the elements of ***spaceNumbers*** define the cardinality of each matrix element. The cardinality of ***spaceNumbers*** defines how often the elements inside are reused, this shows how often the first element of the set is re-iterated. So that:

$$\forall\ message\ \in\ \mathbb{D}\ \exists\{A_{n*m}\} \wedge$$
$$\{\exists!\ spaceNumbers\ |(a_{ij} = Case\ E) \leftrightarrow (|a_{ij}|\ < spaceNumbers_i)\ \vee\ (a_{ij} = Case\ F) \leftrightarrow (|a_{ij}|\ = spaceNumbers_i)\}$$

In the next step, ***A*** is processed in order to have a single flat line, which will be the final encoded message. All elements of ***A*** are stored in the variable ***encodedMessage***. Its order comes to be:
$A_{1,1},\ A_{2,1},\ A_{1,2},\ A_{2,2},\ A_{1,3},\ A_{2,3}...\ A_{n-umpteenth\ m-umpteenth}...A_{2*\frac{|message|}{2}}$.

So that:

$$\forall\ message\ \in\ \mathbb{D}\ \exists!\ \{encodedMessage\ \wedge\ A_{n,*m}|encodedMesagge = A_{1,1},\ A_{2,1},\ A_{1,2},\ A_{2,2}...\ A_{n-umpteenth\ m-umpteenth}$$

Where ***encodedMessage*** is the encrypted message. When not knowing which element of ***A*** corresponds to which character from the message, and if this process encrypts a message of 65,000 characters length, by a permutation without repetition we have that:

$$65,000! \Rightarrow tends\ to\ \infty$$

So we can conclude that with a longer message, the analysis task of the encrypted message becomes harder

3.2. Complexity Analysis

Not surprisingly, the computational complexity of this process results as an order polynomial complexity quadratic $O(n^2)$

3.3. Encryption Protocol

The authentication designed protocol is composed by 3 entities such as user "A", user "B", and a sever "S", as shown in Fig. 1. User ***A*** sends to the server the following request $\{A, N_A, B,_{timestamp}\}\ K_{AS}$ encrypted with a key K_{AS} between ***A*** and ***S***. The server sends to ***B*** the following request $B\{A, N_A, B\}\ K_{BS}$, which represents that ***A*** wants

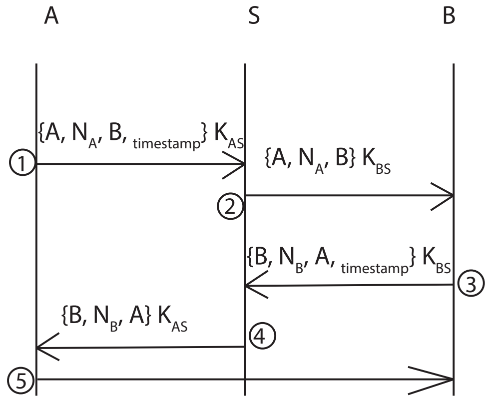

Fig. 1: Hand Shake Protocol

to communicate with ***B*** encrypted with a key K_{BS}. ***B*** responds to the server $\{B, N_B, A,_{timestamp}\}$ K_{BS} which validate who is the receiver. Finally the server refers to $A\{B, N_B, A\}$ K_{AS} which confirms that he received the statement and establishes the secure communication.

In order to avoid catch and replay attacks all the protocol uses a time stamp.

4. Numerical Results

The tests were performed on a equipment with the following characteristics. OS = Windows 7-64 Bits, RAM memory = 8 GB, DISK memory = 320 GB, Processor = Intel I5 3^{th} generation. The performance of our proposed algorithm was tested by loading a text message with a considerable quantity of characters. We started with 100 up to 65.000 characters per message. This message was also encrypted using the well-known algorithms, such as AES, MD2, MD5, SHA-256, SHA-512 and RSA.

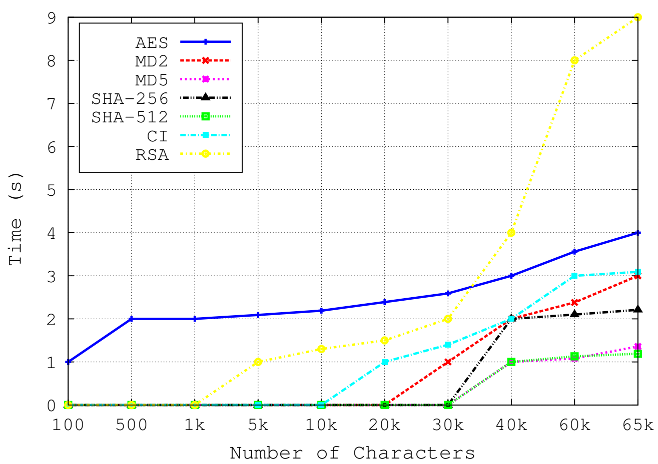

Fig. 2: Algorithm efficiency

As shown in Fig. 2, our proposed algorithm shows a good performance when encrypting up to 10,000 characters. When encrypting a phrase composed by more than 10,000 characters an increase of 11.1% was detected. On the other hand, a significant increase is perceived with 60,000 characters with an increase of 33.3% compared to the beginning. However, since this encryption algorithm was created to be implemented in any instant messaging application; The high time needed for more than 65,000 characters can be rejected. A message of this nature of service does not exceed the amount of 2,500 characters. For this reason it would be an optimal algorithm for accomplishing its purpose. The lowest performance is shown by RSA, with a key size of only 11, reaches a maximum of 100% at the time of conversion to 65,000 characters. This means that, RSA could not be the appropriate algorithm to be implemented for IM services. The fastest algorithm according to the study is the SHA-512; noting just an increase in its conversion time to 65,000 characters of 11,1% . This algorithm is based on hash functions, which from an input, generate a unique alphanumeric output normally of fixed length. It is oriented for database use. Its main disadvantage is to be a one-way function, meaning it can encrypt but not decrypt. It also has weaknesses that have been discovered, such as collisions in the hash space.

5. Conclusions and Future Work

In this paper we proposed a new low complexity encryption algorithm for mobile devices communication. Through the study, tests and implementation made, it appears that the encryption algorithm is optimal for the instant messaging service. As seen in the evidence previously discussed, an encryption method for mobile devices must be performed in a low complexity. Our proposed method performs the encryption and decryption task in a short time which optimizes the battery life. The proposed mechanism seems to reach a trade-off between robustness and low complexity. The algorithm runs in a quadratic polynomial complexity. Furthermore, by encrypting the text message to a flat line of numbers, it makes it more difficult to guess the number that the character represents or how many digits each character will have, therefore the overall length of the message. In the case of an attack intercepting the message, it will be difficult for the attack to find an approach to the algorithm, due to all mathematical algorithm implementations and safeties.

This algorithm has been implemented for application layer use and encrypts only text. Further work must focus on image and video encryption mechanisms.

References

1. Peter A. Loscocco et al. 'The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments'. National Security Agency, pp 1-12, 1998.
2. Ray Spencer et al. 'The Flask Security Architecture: System Support for Diverse Security Policies'. NSA, pp 15-17, 1999.
3. Robert J Runser et al. 'Progress toward Quantum Communications Networks: Opportunities and Challenges'. NSA, pp 1-15, 2007.
4. Jamil, T. 'The Rijndael algorithm'. IEEE Potentials. Vol 23(2), pp 36-38, Apr. 2004.
5. Mandeep Singh Narula et al. 'Implementation of Triple Data Encryption Standard using Verilog'. IJARCSSE. Vol 4(1), pp 1-, Jan. 2014.
6. Mandal, A.K. 'Performance Evaluation of Cryptographic algorithms: DES and AES'. IEEE Students' Conf. on, (SCEECS), pp 1-5, Mar. 2012. Bhopal.
7. Sombir Singh et al. 'Enhancing the Security of DES Algorithm Using Transposition Cryptography Techniques' IJARCSSE. Vol 3(6), pp 1-8, Jun. 2013.
8. Frank Greitzer et al. 'Cyber Friendly Fire: Research Challenges for Security Informatics'. IEEE Conf. on (ISI), pp 94-99, Jun. 2013. Seattle.
9. Robert Morris. 'The data encryption standardRetrospective and prospects'. IEEE Comm. Society Magazine, Vol 16(6) , pp 11-14, 1978.
10. Patrick Gage Kelley et al. 'Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms'. IEEE Symposium on (SP), pp 523 - 537, May. 2012. San Francisco, CA.
11. Joseph Bonneau, Sren Preibusch. 'The password thicket: technical and market failures in human authentication on the web'. Computer Laboratory University of Cambridge. 2010, pp 46 - 48.
12. Mary Cindy Ah Kioon et al. 'Security Analysis of MD5 algorithm in Password Storage'. ISCCCA-13. pp 1 -4, Atlantis Press, Paris. 2013.
13. Shay Gueron at el. 'SHA-512/256'. Security Research Lab, Intel Labs, Intel Corporation, USA, pp 1- 6, 2010.
14. Douglas Montgomery, George C.Runger. 'Applied Statistics and Probability for Engineers'. Third edition. 2003. John Wiley & Sons, Inc.
15. Richard Jonhsonbaugh. 'Matemticas Discretas'. Sixth edition. 2005. Pearson Education
16. Bruce Schneier. 'Applied Cryptography: Protocols, Algorithms, and Source Code in C'. Second Edition. 1996. John Wiley & Sons, Inc.
17. National Security Agency (NSA). Recovered on Abril 11, 2015 from < *http : //www.nsa.gov* >
18. Electronic Frontier Foundation (EFF). Recovered on Abril 11, 2015 from < *http : //www.eff.com* >
19. RSA Labs. Recovered on Abril 11, 2015 from < *http : //www.rsa.com.* >
20. BBM. Yours to control. Recovered on Abril 11, 2015 from < *http : //www.bbm.com/bbm/en.html* >
21. WhatsApp. Simple, personal. Real-time messaging. Recovered on Abril 11, 2015 from < *http : //www.whatsapp.com* >
22. Telegram. Taking back our rigth to privacy. Recovered on Abril 11, 2015 from < *https : //telegram.org/* >