PUC

ISSN 0103-9741

Monografias em Ciência da Computação

n° 13/17

A Hybrid Block and Stream Cipher Encryption Algorithm Based on Colision Resistant Hash Functions

Marcio Ricardo Rosemberg

Daniel Schwabe

Marcus Poggi

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 13/17 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena September, 2017

A Hybrid Block and Stream Cipher Encryption Scheme Based on Collision Resistant Hash Functions

Marcio Ricardo Rosemberg, Daniel Schwabe, Marcus Poggi

{mrosemberg, dschwabe, poggi}@inf.puc-rio.br

Abstract. Background: The Vernam Cipher, or one time pad is considered the only unbreakable encryption scheme. It consists of a XOR operation between the plain text and a random key with the same length of the plain text. Once encrypted, the ciphered message gives nothing the cryptanalyst can use to decipher the encrypted message. The major drawback of the Vernam cipher is that it is not feasible to generate and share a key that is as long as the plain text message. **Aim**: In this work, We propose a hybrid encryption algorithm, which combines block an stream cipher characteristics, using hashes, salts and block counters as block key generators to encrypt a variable length message. The strength of the cryptosystem depends on the strength of the hash function. **Method**: We belive that Collision resistant hash functions (CRHF) may be a very effecitve way to generate peseudorandom block keys which can be used in a XOR operation with the plain text. **Results**: In our experiments, we generated one billion keys for each hash function we tested. None of them repeated itself. **Conclusions**: Our experiments show the key generation algorithm mimics the behavior of the Vernam Cipher and that the block cipher key will not be repeated for a long time. The encryption and the decryption algorithms are very simple and easy to implement, both in software and in hardware.

Keywords: Symmetric Key, Vernam Cipher, Encryption, Security, Hash

Resumo. **Introdução Teórica**: a cifra de Vernam é considerada o único esquema de criptografia totalmente segura. Consiste na operação XOR entre o texto simples e uma chave aleatória com o mesmo comprimento da mensagem não cifrada. Uma vez criptografada, a mensagem cifrada não fornece qualquer informação que o criptoanalista possa usar para decifrar a mensagem criptografada. A principal desvantagem da cifra Vernam é que não é viável gerar e compartilhar uma chave que seja tão longa quanto a mensagem de texto simples. **Objetivo**: neste trabalho, propomos um algoritmo de criptografia híbrido, que combina as características de bloco e de fluxo de criptografia, usando hashes, salts e contadores de blocos como geradores de chave de bloco para criptografar uma mensagem de comprimento variável. A força do criptosistema depende da força da função hash. **Método**: acreditamos que as funções de hash resistentes à colisão (CRHF) podem ser uma maneira muito eficaz de gerar chaves peseudoaleatórias que podem ser usadas em uma operação XOR com texto simples. **Resultados**: em nossos experimentos, geramos um bilhão de chaves para cada função de hash testada. Nenhuma delas se repetiu. **Conclusões**: Nossos experimentos mostram que o algoritmo de geração de chaves imita o comportamento da Cifra Vernam e que a chave de bloco não será repetida por muito tempo. Os algoritmos de cifragem e decifragem são muito simples e fáceis de implementar, tanto em software como em hardware.

Palavras-chave: Chave Simétrica, Cifra Vernam, Criptografia, Segurança, Hash

This work has been sponsored by the Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil (CNPQ).

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Table of Contents

1 Introduction

In this section, we briefly explain our proposal and our motivation.

Symmetric key encryption is essentially divided in two categories: Block Ciphers and Stream Ciphers (Paar & Pelzl, 2010).

Block Ciphers consists of two algorithms: one for encryption and another for decryption (Cusick & Stanica, 2009). With block ciphers, the message is broken into blocks of fixed sizes and each block is encrypted with the encryption key.

With Stream Ciphers, the plain text is combined with a pseudorandom cipher bit stream and each bit of the plain text is combined with the cipher bit stream in a XOR operation, implementing in practice, the Vernam Cipher (CryptoMuseum, 2012).

We belive that Collision Resistant Hash Functions (CRHF) may be a very effecitve way to generate peseudorandom keys which can be used in a XOR operation with the plain text.

In this work, We propose a hybrid encrytion algorithm which combines block an stream cipher caracteristics, using hashes, salts and block counters as block key generators to encrypt a variable length message. The strengh of the cryptosystem depends on the strengh of the hash function.

1.1 Motivation

Recently, several symmetric key encryption algorithms have been broken. RC4 was broken in less than one minute in 2007 (Tews, et al., 2007). In 2016, the security of Triple DES was lowered from 2^{168} to 2^{113} and Blowfish have been proven to be unsecure. Both Blowfish and Triple DES are vulnerable to SWEET32 attacks (Bhargavan & Leurent, 2016). As a result, new encryption algorithms are always in demand.

The AES (Advanced Encryption Standard) is the most popular and one of the most secure, block cipher, encryption algorithm in use. Because it has several modes of operation, AES is not trivial to be used by the developer. One of the most secure modes of operation for AES is the CBC (Cipher Block Chaining). The first ciphered block is used to encrypt the second and so on. For the first block, CBC requires an IV (Initialization Vector), which consists of a block composed of random bits. The IV is transmitted with the encrypted message to allow decryption by the receiver. A nonrandom IV weakens the encryption and allows dictionary and differential attacks to have a better chance to decipher the encrypted message (Preneel, 2007). Also, AES requires a key with a specific length. AES-128 requires a 128 bit long key, while AES-192 and AES-256 require 192 and 256 bit long keys, respectively. If one chooses a weak key or uses a shorter key padded with spaces or nulls, one wakens the encryption algorithm.

We also encountered a problem in the development of mobile applications in the AES algorithm of the Android Operating System. Its Java Cipher Class does not allow all types of padding schemes. Padding is required because AES uses fixed 128 bits length blocks. When a block does not have enough bits, the block must be padded to with some padding scheme. The Cipher class of the Android SDK currently supports PKCS5, ISO10126 or No Padding. However, Swift for IOS accepts PKCS7, ISO97971, AnsiX923, ISO10126, ZeroPadding and NoPadding. Finally, we found Windows libraries accepting PKCS7, AnsiX923, ZeroPadding and Padding with spaces (20 hex). Even though this problem is more related to Software Engineering or Human–computer interaction, it gave us the initial motivation to propose an encryption scheme without the need for padding.

Summarizing the motivation, our proposed encryption scheme focus on the following characteristics:

- Requires only the plain text, the encryption key and the Hash Algorithm to be used in the encryption process.
- Requires only the ciphered message, the encryption key and the Hash Algorithm used in the decryption process.
- Keys can be of any length
- No padding scheme is needed
- No real time synchronization required between the sender and the receiver
- Easy to implement on almost any platform and programming language
- Flexibility and Modularity

By Flexibility and Modularity, we mean that the user or software developer can choose a CRHF from a set, according to his/hers security requirements. If the CRHF becomes unsecure, all he/she has to do is to adopt another CRHF. Neither the encryption or the decryption algorithm need changes in their coding.

2 Background

In this section, we provide the necessary background for the understanding of our proposal.

2.1 The Vernam Cipher

The Vernam Cipher (Vernam, 1926) (Kahn, 1967) or one time pad is considered the only unbreakable encryption scheme. It consists of a XOR operation between the plain text and a random key of the same length of the plain text. Once encrypted, the ciphered message gives nothing the cryptanalyst can use to decipher the encrypted message. On the other hand, the Vernam Cipher has two drawbacks:

1) It is not feasible to negotiate a key between two parties, in which the key size is as big as the message size.
2) If two parties are capable of negotiating a key with the same length of the message, they can also exchange messages without the need for encryption.

2.2 Collision Resistant Hash Functions

Collision Resistant Hash Functions (CRHF) are used for various applications such as message authentication, digital signatures, pseudorandom bit generation, integritty assurance and so on (Akhimullah & Hirose, 2016). Hashes are mathematical functions that compress an input of arbitrary length to a result with a fixed length. Hash functions are also used to allocate as uniformly as possible storage for the records of a file. Yet, CRHF are hashes which collisions are hard to find.

The formal definition of a Collision resistant hash function appears to be credited to Damgård (Damgard, 1988).

A collision resistant hash function is a function h satisfying the following conditions (Preneel, 2003):

1. The description of h must be publicly known and should not require any secret information for its operation (*extension of Kerckhoffs's principle*).

2. The argument X can be of arbitrary length and the result h(X) has a fixed length of ***n*** bits.

3. Given h and X, the computation of h(X) must be "easy".

4. The hash function must be one-way in the sense that given a Y in the image of h, it is "hard" to find a message X such that h(X) = Y and given X and h(X) it is "hard" to find a message X' ≠ X such that h(X') = h(X).

5. The hash function must be collision resistant: this means that it is "hard" to find two distinct messages that hash to the same result.

In order to satisfy conditions 4 and 5, a CRHF must be resistant to preimage and second preimage attacks (Rogaway & Shrimpton, 2004). For a preimage attack to be successful, the computational complexity required is, at least, $O(2^n)$, while for a second preimage attack, the computational complexity required is, at least, $O(2^{n/2})$.

2.3 Block Cipher vs Stream Cipher

Block cipher operates on fixed length blocks of bits (Stallings, 2011). If the plain message is shorter than the block size, some sort of padding is required.

Block ciphers use two different algorithms: one for encryption and another for decryption (Bellare & Rogaway, 2005).

The goal of stream ciphers is to mimic the Vernam Cipher by continuously generating and synchronizing new keys between the sender and the receiver (El-Razouk, et al., 2014). They are more difficult to implement than block ciphers, because the keystream cannot repeat itself during the session. There are two types for operation modes: Synchronous Stream Cipher and Self-synchronizing stream ciphers.

In Synchronous Stream Cipher, the keystream is generated independently of the plain text and of the ciphered message. The keystream is commonly produced by a pseudorandom generator, parameterized by the secret key of the whole scheme. In this mode, the sender and receiver must be synchronized for decryption to be successful. One way to achieve synchronization is to send an IV in the open before each encrypted frame (Fontaine, 2011) (Rueppel, 1986).

In a Self-synchronizing, or asynchronous, stream cipher, the keystream depends on the secret key of the scheme and also on a fixed number of ciphered text digits that have already been produced by the sender, or read by the receiver. The idea of self-synchronization was patented in 1946 and has the advantage that the receiver will automatically synchronize with the keystream generator after receiving a certain number of ciphered text digits (Fontaine, 2011) (Daemen & Kitsos, 2008).

2.4 Counter Mode Encryption

In counter mode encryption, block ciphers use sequential numbers which are combined with the encryption key in the encryption algorithm (Schneier, 1996). The result of the operation is the block key. Finally a bitxor operation is performed with the block key and the plaintext block. After each block encryption, the counter increments by some constant or a function. Essentially, counter mode transforms a block cipher scheme into a stream cipher one. The decryption algorithm of the block cipher is not needed. Only the encryption algorithm is used.

Counter mode encryption has the following properties (Fay, 2016) :

- No error propagation: An error in a block B_i has no effect on B_{i+1}
- Synchronization: "Encryption and Decryption work synchronously as long as the counters are in sync". If the sender and receiver loose synchronization, the receivers cannot decrypt the ciphered messages.
- Parallelizability: Fixed length block ciphers using counter mode encryption can be encrypted and decrypted in parallel, because each block is independent from the others.

2.5 Salts and Nonces

Salts and Nonces are random data with different goals. Salts protect passwords against dictionary attacks, when they are used as while nonces are one time random numbers to be used as initialization vectors for encryption algorithms to protect both the encryption key and the ciphered message, ensuring that the same plaintext encrypted with the same key produces a different ciphered message because the nonce will be different. Usually there is only one nonce for the entire encryption process, while salts are abundant and preferably a unique for each password stored in a database.

2.6 The Birthday Attack

The Birthday Attack weakens any hash function by exploiting the mathematics behind the birthday problem in probability theory (Jin, et al., 2017) (McKinney, 1966).

The Birthday Problem concerns the probability that, given a set of k randomly people, two of them will have the same birthday. With just 30 people, the probability is 70%. Applying the birthday problem to hash functions, the attacker wants to create a second message m′ which produces the same hash output of the original message m, h(m′) = h(m). For a hash function of 2^n possible outputs, where n is the fixed length size in bits of the hash output, the probability of finding a collision is

$$p(n) = \sqrt{\frac{\pi}{2}2^n}$$

As a result, the equation lowers the time complexity for obtaining a collision from ***$O(2^n)$*** to ***$O(2^{n/2})$***

3 Related Work

Bruce Schneier proposed a simple way to use a One-way Hash Functions to encrypt data in a stream cipher algorithm (Schneier, 1996):

$$C_i = M_i \oplus H(K,C_{i-1}) \text{ and } M_i = C_i \oplus H(K,C_{i-1})$$

A hash based stream cipher algorithm was published in 1999 (Peyravian, et al., 1999). In this system, the authors use the sha-1 hash of the encryption key and a sequential counter to generate a keystream. Although efficient, the cryptosystem would produce the same ciphered message given the same plaintext and using the same key or session key.

The encryption/decryption algorithm is described as :

$$C_i = M_i \oplus H(i,K) \text{ and } M_i = C_i \oplus H(i,K).$$

The ciphered text is $C = C_1 \| C_2 \| \dots \| C_n$, where $\|$ means string concatenation.

This hash based stream cipher have been cited 18 times. The articles (Gordon & Loeb, 2002), (Campbell, et al., 2003), (Lee, et al., 2006), (Wang, et al., 2011), (Kumari, et al., 2012), (Gordon & Loeb, 2001), (Kumari & Khan, 2014), (Gordon & Loeb, 2004), (Demirkan & Goul, 2013), (Patrick, 2008), (Tesink, et al., 2005), (Elzouka H. , 2006), (Yeh & Chou, 2001), (Elzouka H. A., 2008), (Chen, et al., 2013), (Li, et al., 2003) and (Cheng, 2005) cite this work as background reference. (Huang, Feng, & Zhang, 2001), in a four page short paper, propose an encryption scheme based on one-way hash and the services of a pseudorandom number generator to enhance the algorithm.

Specifically for images encryption, (Cheddad, et al., 2010) proposed an encryption algorithm that uses hashes and the Fourier Transform.

4 The Proposed Encryption Algorithm

In this section, we state our proposal and its advantages

4.1 Formal Description

The encryption processes consists of the following steps:

- The Sender and Receiver negotiate a CRHF (H)
- Generate and distribute a shared key K between the sender and the receiver. The key can be of any length, but the optimal Key size is half the size of H.
- Divide the message M in n blocks of H size, such that $M = m_1 \| m_2 \| \dots \| m_n$, where $\|$ means string concatenation. For compatibility among operating systems regional code pages, M must be converted to CP-1252 encoding.
- For each block, generate a 32 bit integer pseudorandom number (S_i). S_i must be a DWORD little endian format. The same format is required for the counter i.
- Generate the ciphered block with the following formula

$$c_i = S_i \| m_i \oplus H(K, S_i, i)$$

- Calculate the MAC (Message Authentication Code) = $H(K,M,S_{MAC})$ and prepend it to the ciphered message.

As a result, the ciphered message is $C = S_{MAC} \| MAC \| c_1 \| c_2 \| \ldots \| c_n$

The decryption process is different from the encryption:

- Separate both S_{MAC} and MAC from the ciphered message.
- The ciphered block length is l = size of H + 32 bits long, except for the last ciphered block. The last block length may vary from 40 bits (last salt + one character) to l. For each ciphered block c_i of l bits, take the first 32 bits which corresponds to the block salt (S_i). The remaining l-32 bits is the encrypted message block (Φ_i).
- The original message block is obtained from the formula

$$m_i = \Phi_i \oplus H(K, S_i, i)$$

- Reassemble the message $M = m_1 \| m_2 \| \ldots \| m_n$
- Calculate the message's $MAC = H(K,M,S_{MAC})$
- Check if the received and the calculated MAC match. If they do, the message is both authentic and intact.

4.2 Analysis of the Encryption Scheme

Using the counter and having one Salt for each block guarantee the one time pad of the Vernam Cipher and that the same message, encrypted with the same key twice will not produce the same ciphered message, because Salts would be different for most of the blocks.

A successful random collision attack on a block allows an attacker to decrypt only that specific block.

The same key may be reused to encrypt other messages, since the sequence of Salts generated for each block will be different.

Because the ciphered block is composed of the salt block and the encrypted block, there is no need for synchronization between the sender and the receiver. However, there is a significant increase in the size of the ciphered message. For a plain message of 2GB, the ciphered message increases, according to the following table:

Hash Length (bits)	Hash Length (Bytes)	Number of Blocks	Total Salt Cost (MB)	Final Size (GB)	Increase
160	20	107.374.183	409,6	2,40	20%
256	32	67.108.865	256	2,25	13%
384	48	44.739.243	170,7	2,17	8%
512	64	33.554.432	128	2,13	6%
1024	128	16.777.216	64	2,06	3%

Table 1 – Salt cost

From

Hash Length (bits)	Hash Length (Bytes)	Number of Blocks	Total Salt Cost (MB)	Final Size (GB)	Increase
160	20	107.374.183	409,6	2,40	20%
256	32	67.108.865	256	2,25	13%
384	48	44.739.243	170,7	2,17	8%
512	64	33.554.432	128	2,13	6%
1024	128	16.777.216	64	2,06	3%

Table 1, we can infer that the larger the size of the CRHF, the stronger the cipher and the lower the salt cost.

The cipher strength depends only on the strength of CRHF to resist to preimage, second preimage or collision attacks. Because the encryption algorithm applies the CRHF to the key, the salt and the counter, the compression function of the hash is either not used or used a few times, strengthening the resistance to second preimage attacks to a complexity nearing ***O(2^n^)*** (Kelsey & Schneier, 2005). Nonetheless, because of the Birthday Attack, the cipher strength is ***$O(2^{n/2})$*** per block, which is why the optimum secret key length is half the size of the CRHF output length. If the key length is lower than the CRHF output length, it is easier for the attacker to perform a brute force attack on the key. Nevertheless, if the key length in greater than the CRHF output length, a collision attack would take less time than a brute force attack on the key.

Assuming the attacker knows the size of the encryption key (not the key value itself) and that a dictionary attack will fail, the decision to attack either the key, by brute force, or the hash, by trying random collisions, will depend on which is smaller: the key size or the CRHF output size.

Recently, there have been significant improvements on Hash functions. In (Su, et al., 2016), the authors claim that a non-iterative hash function for small messages can produce a hash output complexity of ***$O(2^m)$***, where $80 \leq m \leq 232$ and $80 \leq m \leq n \leq 4096$ and n being the size of the message to be hashed. Thus, the algorithm is capable of resisting the Birthday Attack or Meet in the Middle Attacks (Stallings, 2011).

Chaos based one-way hash functions (CBHF) offers as much collision resistance as CRHF, but, because chaotic systems are defined in the real number field (Yang, et al., 2009), the authors claim that the collision attack is more difficult in CBHF than it is in CRHF. CBHF also features better distribution than CRHF (Ahmad, et al., 2017).

Recent studies on Lattice hash functions also claims better distribution and equivalent collision resistance, when compared to CRHF (Wang, et al., 2011).

The following table compares the cipher strength of the proposed algorithm with various CRHF:

CRHF	Block Size (bits)	Output size (bits)	Security bits
RIPEMD-160	512	160	80
SHA-256	512	256	128
SHA-384	1024	384	192
SHA-512	1024	512	256
Whirlpool	512	512	256
Su, et al., 2016	80 - 4096	80 - 232	80 - 232

Table 2 – CRHF comparison

4.3 Stream or Block Cipher

According to Rueppel's definition (Rueppel, 1992): *"Block ciphers operate on data with a fixed transformation on large blocks of plaintext data; stream ciphers operate with a time-varying transformation on individual plaintext digits."* the proposed encryption scheme is a stream cipher. However, if we consider the fact that stream ciphers use the same algorithm for both encryption and decryption, while block ciphers have different algorithms, the proposed encryption scheme is a block cipher, since it adds the block salts into the ciphered blocks in the encryption algorithm and removes them from the ciphered blocks in the decryption algorithm. As a result, the proposed encryption scheme is a hybrid with both stream and block cipher characteristics.

5 Our Experiments

In this section, we report how we conducted our experiments

5.1 Research Questions

We already know that the counter produces the avalanche effect on hash functions (Peyravian, et al., 1999) (Sanchez-Arias, et al., 2017). What we do not know is how the salt, combined with the counter, is going to affect the hash output. Hence, our primiry reaserach question is: Does the combination of a secret string, a salt and a counter can generate block keys that will not repeat itself for a long time?

The null hypothesis is: The cipher key repeats itself for a different salt and counter pair.

We can reject the null hypothesis, if we generate a large number of keys and they do not repeat themselves. The experiment must be repeated for each CRHF we intend to use for block keys generation.

5.2 Implementation

We selected four CRHF for testing purposes: RIPEMD160, SHA-256, SHA-384 and SHA-512. For each CRHF, we generated one billion keys, divided in 200 sets of 5 million keys. We used a pseudo-random algorithm to generate each salt ranging from 0 to 2^{31}. The counter started at 0 and was incremented by one for each key generated. The first set had counters ranging from 0 to 4,999,999. The second had counters ranging from 5,000,000 to 9,999,999, and so on. Because of memory constraints, each set was saved in a CSV (comma separated values) file and loaded into memory only when necessary. Each CSV file contains the block cipher key in a hexadecimal format, the salt and the counter, both 32 bit integers converted to strings.

We used for the secret key the string: "`Key_&_TesT-2017`" without the quotes.

After the generation of the sets, each set was loaded into memory and confronted with the others for duplicated keys. As expected, there were no key duplications and the null hypothesis was rejected for all hash algorithms.

5.3 Analyses of the results

The results shows that even with a relatively small salt length (32 bit positive integers), block cipher key collisions do not occur.

With 1 billion different block cipher keys it is possible to transmit the following amounts of data: 20GB with RIPEMD160, 32GB with SHA-256, 48GB with SHA-384 and 64GB with SHA-512.

Our results show that it is even safe to reuse the same secret several times, resetting the counter. The probability of the same salt occur with the same counter is 1 in 64K (birthday paradox on a 32 bit salt). As a result, repeating the same secret key 1024 times, resetting the counter on each cycle, allows us to safely encrypt files or transmit messages up to 20TB with RIPEMD160, 32TB with SHA-256, 48TB with SHA-384 and 64TB with SHA-512.

If the chosen CRHF becomes unsafe, all the developer has to do is to choose another CRHF. The algorithm itself remains unchanged.

6 Conclusions and Future Works

We presented an encryption scheme with block and stream cipher characteristics, based on a Collision Resistant Hash Function. Our experiments show the algorithm mimics the behavior of the Vernam Cipher and that the block cipher key will not be repeated for a long time.

The encryption scheme is very simple and easy to implement, both in software and in hardware.

We intend to continue to test the encryption scheme with other CRHF and to integrate it with an authentication protocol, capable of negotiating a session key, which will then be used with the encryption scheme to guarantee confidentiality.

References

Ahmad, M., Khurana, S., Singh, S., & AlSharari, H. (2017). A Simple Secure Hash Function Scheme Using Multiple Chaotic Maps. In: *3D Research v.8 n.13* (p. 13). Springer.

Akhimullah, A., & Hirose, S. (2016). Lightweight Hashing Using Lesamnta-LW Compression Function Mode and MDP Domain Extension. *Fourth International Symposium on Computing and Networking* (pp. 590-596). Hiroshima: IEEE.

Bellare, M., & Rogaway, P. (2005). Block Ciphers. In: *Introduction to Modern Cryptography* (pp. 39-40). Mihir Bellare and Phillip Rogaway.

Bhargavan, K., & Leurent, G. (2016). On the practical (in-) security of 64-bit block ciphers: Collision attacks on HTTP over TLS and OpenVPN. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 456-467). ACM.

Campbell, K., Gordon, L. A., Loeb, M. P., & Zhou, L. (2003). The economic cost of publicly announced information security breaches: empirical evidence from the stock market. *Journal of Computer Security* (pp. 431-448). IOS Press.

Cheddad, A., Condell, J., Curran, K., & and McKevitt, P. (2010). A hash-based image encryption algorithm. *Optics communications v.283 n. 6* (pp. 879--893). Elsevier.

Chen, B., Huang, Y., & Shi, Y. (2013). A Hybrid Mutual Identity Authentication Technology with its Application. *Computer security n.12*, pp. 34-37.

Cheng, Y. (2005). A New Text Digital Watermarking Algorithm. *Science & Technology and Engineering v.5 n.14*, pp. 1006-1008.

CryptoMuseum. (2012). *The Vernam Cipher*. Acesso em 16 de 05 de 2017, disponível em Crypto Museum: http://www.cryptomuseum.com/crypto/vernam.htm

Cusick, T. W., & Stanica, P. (2009). *Cryptographic Boolean functions and applications.* Academic Press. pp. 158–159. ISBN 9780123748904.

Daemen, J., & Kitsos, P. (2008). The self-synchronizing stream cipher moustique, new stream cipher designs. *Lecture notes in computer science, vol 4986* (pp. 210–223). Springer.

Damgard, I. (1988). Collision free hash functions and public key signature schemes. *Advances in Cryptology, Proc. Eurocrypt'87, LNCS 304* (pp. 203–216). D. Chaum and W.L. Price, Eds., Springer-Verlag.

Demirkan, H., & Goul, M. (2013). Taking value-networks to the cloud services: security services, semantics and service level agreements. *Information Systems and e-Business Management v.11 n.1* (pp. 51-91). Springer.

Ellison, C., Hall, C., Milbert, R., & Schneier, B. (2000). Protecting secret keys with personal entropy. *Future Generation Computer Systems v.16 n.4* (pp. 311-318). Elsevier.

El-Razouk, H., Reyhani-Masoleh, A., & Gong, G. (2014). New Implementations of the WG Stream Cipher. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems V.22 N.9*, pp. 1865-1878.

Elzouka, H. (2006). *A New Robust and Secure Steganographic System for Greyscale Images.* Alexandria: Computer Engineering Department. Arab Academy for Science and Technology.

Elzouka, H. A. (2008). FPGA based implementation of robust watermarking system. *Fifth International Conference on Information Technology: New Generations ITNG 2008.* (pp. 1274-1278). IEEE.

Fay, R. (2016). Introducing the counter mode of operation to Compressed Sensing based encryption. *Information Processing Letters v.116 n.4* (pp. 279-283). Elsevier.

Fontaine, C. (2011). Self-Synchronizing Stream Cipher. *Encyclopedia of Cryptography and Security*, 1175-1176.

Fontaine, C. (2011). Synchronous Stream Cipher. *Encyclopedia of Cryptography and Security*, 1274-1275.

Gordon, L. A., & Loeb, M. P. (2001). Using information security as a response to competitor analysis systems. *Communications of the ACM v.44 n.9* (pp. 70--75). ACM.

Gordon, L. A., & Loeb, M. P. (2004). The economics of information security investment. In: *Economics of Information Security* (pp. 105-125). Springer.

Gordon, L., & Loeb, M. (2002). The economics of information security investment. *ACM Transactions on Information and System Security (TISSEC) v.5 n.4* (pp. 438-457). ACM.

Huang, Z., Feng, X., & Zhang, H. (2001). Hash - based Encryption Scheme. *Communications Technology n.7*, pp. 87-89.

Jin, Z., Lai, Y., Hwang, J., Kim, S., & Teoh, A. (2017). A New and Practical Design of Cancellable Biometrics: Index-of-Max Hashing. *arXiv preprint arXiv:1703.05455.*

Kahn, D. (1967). The Codebreakers: The Story of Secret Writing. *New York: Macmillan Publishing Co.*

Kelsey, J., & Schneier, B. (2005). Second preimages on n-bit hash functions for much less than 2n work. In: R. Cramer (Ed.), *EUROCRYPT v. 3494 of LNCS* (pp. 474–490). Springer.

Kumari, S., & Khan, M. (2014). Cryptanalysis and improvement of 'a robust smart-card-based remote user password authentication scheme'. *International Journal of Communication Systems v.27 n.12* (pp. 3939-3955). Wiley Online Library.

Kumari, S., Gupta, M. K., & Kumar, M. (2012). Cryptanalysis and security enhancement of Chen et al.'s remote user authentication scheme using smart card. *Central European Journal of Computer Science v.2 n.1* (pp. 60-75). Springer.

Lee, C., Hwang, M., & Liao, I. (2006). Security enhancement on a new authentication scheme with anonymity for wireless environments. *IEEE Transactions on Industrial Electronics v.53 n.5* (pp. 1683-1687). IEEE.

Li, S., Qin, Z., & Wang, X. (2003). A New Message Digest Codes Generating Algorithm. *Journal of Computer Research and Development v.40 n.3*, pp. 413-416.

McKinney, E. H. (1966). Generalized birthday problem. *The American Mathematical Monthly v. 73 n. 4,*, 385-387.

Paar, & Pelzl. (2010). *Understanding Cryptography.* Berlin: Springer-Verlag p.30.

Patrick, K. N. (2008). *Patente Nº 7,337,319.* USA.

Peyravian, M., Roginsky, A., & Zunic, N. (1999). Hash-based encryption system. *Computers & Security* (pp. 345-350). Elsevier.

Preneel, B. (2003). *Analysis and Design of Cryptographic Hash Functions.* Diss. PhD thesis, Katholieke Universiteit Leuven. pp. 18.

Preneel, B. (2007). *An introduction to modern cryptology.* Dept. Electrical Engineering-ESAT/COSIC.

Rogaway, P., & Shrimpton, T. (2004). Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. *In International Workshop on Fast Software Encryption* (pp. 371-388). Berlin-Heidelberg: Springer.

Rueppel, R. (1992). Stream Ciphers. In: G. Simmons, *Contemporary Cryptology: The Science of Information Integrity* (pp. 65-134). New York: IEEE.

Rueppel, R. A. (1986). *"Stream ciphers." Analysis and Design of Stream Ciphers.* Berlin Heidelberg: Springer pp 5-16.

Sanchez-Arias, G., Garcia, C., & G-Bustelo, B. (2017). Midgar: Study of communications security among Smart Objects using a platform of heterogeneous devices for the Internet of Things. In: *Future Generation Computer Systems v.74* (pp. 444-466). Elsevier.

Schneier, B. (1996). Counter Mode. In: *Applied cryptography: protocols, algorithms, and source code in C (Second Edition)* (pp. 178-179). New York: John Wiley & Sons, Inc.

Schneier, B. (1996). Using one-Way Hash Functions. In: *Applied cryptography: protocols, algorithms, and source code in C (Second Edition)* (p. 296). New York: John Wiley & Sons, Inc.

Stallings, W. (2011). Block Cipher Operation. In: *Cryptography and Network Security Principles and Practice Fifth Edition* (pp. 196-197). Prentice Hall.

Su, S., Xie, T., & Lü, S. (2016). A provably secure non-iterative hash function resisting birthday attack. In: *Theoretical Computer Science v.654* (pp. 128-142). Elsevier.

Tesink, S., MIM, L. R., & Leune, C. (2005). *Improving csirt communication through standardized and secured information exchange.* Tilburg Master Thesis.

Tews, E., Weinmann, R., & Pyshkin, A. (2007). *Breaking 104 Bit WEP in Less Than 60 Seconds.* Cryptology ePrint Archive, Report 2007/120.

Vernam, G. (1926). Cipher printing telegraph system for secret wire and radio telegraph communications. *Journal American Institute of Electrical Engineers Vol. XLV*, 109-115.

Wang, R.-C., Juang, W.-S., & Lei, C.-L. (2011). Robust authentication and key agreement scheme preserving the privacy of secret key. *Computer Communications* (pp. 274-280). Elsevier.

Wang, Y., Wong, K., & Xiao, D. (2011). Parallel hash function construction based on coupled map lattices. In: *Communications in Nonlinear Science and Numerical Simulation v.16 n.7* (pp. 2810-2821). Elsevier.

Yang, H., Wong, K., Liao, X., Wang, Y., & Yang, D. (2009). One-way hash function construction based on chaotic map network. In: *Chaos, Solitons & Fractals v.41 n.5* (pp. 2566-2574). Elsevier.

Yeh, Y.-S., & Chou, J.-S. (2001). RC hash function. *Journal of Information and Optimization Sciences v.22 n.2* (pp. 297-306). Taylor & Francis.