International Journal of Computer Applications (0975 - 8887)
Volume 176 - No.16, April 2020

Building Provably Secure Block Ciphers from Cryptographic Hash Functions

Charles F. de Barros
Department of Computer Science
Federal University of São João del Rei
Minas Gerais, Brazil

ABSTRACT

This paper presents a proposal for the construction of provably secure block ciphers based on cryptographic hash functions. The core idea consists of using a hash function to generate pseudorandom strings to be combined with the message blocks. Each one of these strings depend on the previous ciphertext block (or the initialization vector, in the case of the first message block), the secret key k and a *block key* derived from k. One of the main features of the proposed construction is that it allows keys of arbitrary length, since the key itself is never directly combined with the message. Furthermore, even if an adversary manages to guess all of the block keys, he can't efficiently retrieve the master secret key or the message, provided that the underlying hash function is cryptographically secure. Finally, the proposal also embeds an authentication tag in the initialization vector. Hence, instead of being randomly chosen, the IV is always dependent on the key and the message, which is crucial to generate confusion, diffusion and avalanche effect, since any minor change in the key or in the message will cause the IV to be drastically different, due to the properties of the HMAC, and because of the chained nature of the construction, this change will propagate to all ciphertext blocks.

General Terms

Block Ciphers, Cryptographic Hash Functions, Symmetric-Key Criptography

Keywords

Block Ciphers, Cryptographic Hash Functions, Symmetric-Key Criptography

1. INTRODUCTION

Block ciphers and hash functions are fundamental primitives of any cryptographic protocol. Although each serves a different purpose, they are usually combined in order to achieve different goals simultaneously, namely confidentiality, integrity and authentication. Authenticated encryption is a typical example in which both primitives are combined, the block cipher being used to provide confidentiality, and the hash function being used to provide integrity and authentication, by means of an HMAC.

Since they are meant to achieve different cryptographic goals, block ciphers and hash functions have different design principles. The former must satisfy certain criteria, such as confusion and diffusion, while the latter should have properties like collision resistance. It can be said that these primitives are inherently different, mainly because block ciphers are reversible (it is always possible to decrypt an encrypted message, given the proper key), while hash functions are, by design, irreversible.

Nevertheless, there is a certain similarity in the way block ciphers and hash functions are built. In fact, both are based on iterations: block ciphers consist of iterations of a round function, while hash algorithms are built upon iterations of a compression function. As a matter of fact, it is possible to build hash functions from block ciphers, due to the fact that block ciphers are natural compression functions, which makes them suitable to be used at the core of the so-called Merkle-Damgård [1] construction for cryptographic hash functions. This is indeed the underlying design of well known cryptographic hash functions, such as MD5, SHA-1 and SHA-2 [2]. Recently, NIST released SHA-3 [3], a new member of the Secure Hash Algorithm family of standards. This new cryptographic hash function is actually based on a primitive family called Keccak [4], which is built upon a new design principle known as sponge construction.

On the other hand, the construction of block ciphers from hash functions has already been proven possible. There are concrete examples in the literature, such as SHACAL [5] and its successor SHACAL-2, based on the compression functions of SHA-1 and SHA-2, respectively. Nevertheless, the design principle of ciphers like SHACAL is different from the one presented in this paper. While SHACAL is based on turning the compression function of the underlying hash algorithm into a block cipher, the construction presented in this paper uses the entire hash function to generate pseudorandom strings to be combined with the message blocks.

There are also constructions of block ciphers based on a result due to [6], which shows how to obtain a secure block cipher from three pseudorandom functions, combined in a three round Feistel network. These constructions typically use both hash functions and stream ciphers as sources of pseudorandomness. BEAR and LION [7] are examples of provably secure block ciphers based on this result. They are built upon a Feistel structure, in a DES-like manner, with BEAR using two hash function calls and one stream cipher call, while LION uses the hash function once and calls the stream cipher twice.

This paper presents a new construction as an improvement of the aforementioned examples in terms of efficiency. In fact, the proposed block cipher requires only two calls to a hash function per

message block. The first call takes as input the previous ciphertext block, combined with the master secret key and a block key, and outputs a string which enciphers the current message block. The second call generates the block key from the master secret key to be combined with the current ciphertext block, in order to produce the string that will encipher the next message block.
The proposed construction presents some interesting features, such as the possibility to use arbitrary length keys, since the message blocks are never combined directly with the key, but with the result of a hash function. Furthermore, the construction is based on the generation of block keys $k_0, k_1, \cdots, k_\ell$ from the master secret key k. Even if an adversary manages to guess all of these block keys, he is unable to efficiently retrieve the message or the master secret key, provided that the hash function is cryptographically secure.

1.1 Roadmap

This paper is organized as follows: section 2 presents a brief review on block ciphers and cryptographic hash functions. In section 3, the construction of a block cipher based on hash functions is presented in details. Section 4 discusses the security of the proposed construction. In section 5, experimental results that corroborate the usability and security of the proposed block cipher are presented. Finally, section 6 presents conclusions and final remarks.

2. THEORETICAL REVIEW

This section presents a brief theoretical review on block ciphers and cryptographic hash functions, discussing the major design principles of these cryptographic primitives, such as confusion and diffusion for block ciphers, and collision-resistance for hash functions.
As usual, the set of binary strings of length n will be denoted by $\{0,1\}^n$, and the set of binary strings of arbitrary length (including 0) shall be represented by $\{0,1\}^*$. Concatenation of strings will be denoted by $||$, and the XOR operation by $\oplus$. The Hamming distance between two binary strings of the same length x and y corresponds to the number of bits in which x and y differ.

2.1 Block Ciphers

Block ciphers, as the name itself suggests, process messages in blocks of fixed length, generating fixed-size ciphertext blocks. The canonical construction of block ciphers consists of iterated round functions. At each round, the result of the previous round is combined with a round key k_i, obtained from the master key through a process called key schedule. In the first round, the message block is combined with the first round key. After a certain fixed number of rounds, the result of the round function is returned as a ciphertext block, as shown in Figure 1
Claude Shannon defined the concepts of confusion and diffusion [8] as crucial properties for any symmetric cryptographic system. Confusion refers to destroying any statistical relationship between the key and the ciphertext, in such a way that any ciphertext bit should depend on several parts of the key. Conversely, if a single bit of the key is changed, we expect several bits of the ciphertext to be affected. Diffusion is meant to hide the relationship between the message and the ciphertext. Changing a single bit of the plaintext should affect many bits of the ciphertext and vice-versa.
These two properties are closely related to the Strict Avalanche Criterion (SAC), introduced by [9], which is one of the main design principles for both block ciphers and cryptographic hash functions. Roughly speaking, an algorithm is said to satisfy the SAC if slight changes in the input cause significant changes in the output. Put into more formal terms, if a single input bit is flipped, we expect every output bit to be flipped with probability $1/2$. In other words, every time an input bit is changed, around 50% of the output bits will be changed.
Note that a block cipher specifies how a single message block is encrypted and decrypted. In order to determine how the entire message (which can be larger than the block size) is processed, a mode of operation must be specified. For instance, in the Electronic CodeBook (ECB) mode, each message block is independently encrypted. Consequently, decryption is also performed in an independent way. This mode of operation is not considered secure, as similar message blocks will always be encrypted as similar ciphertext blocks, which reveals the relationship between the plaintext and the ciphertext.
In the Cipher Block Chaining (CBC) mode of operation, each message block is XORed with the previous ciphertext block prior to being encrypted. The first message block is XORed with a randomly generated Initialization Vector (IV). Decryption is performed in a similar way, XORing the result of each decrypted block with the previous ciphertext block. Other well known modes of operation, such as Cipher FeedBack (CFB), Output FeedBack (OFB) and Counter (CTR), were designed to turn a block cipher into a stream cipher, as the encryption algorithm is used to generate pseudorandom strings to be XORed with the message blocks.

2.2 Cryptographic Hash Functions

A hash function takes as input a message of arbitrary size and returns a digest of fixed size. In other words, it is a function $\mathrm{H} : \{0,1\}^* \to \{0,1\}^n$ for some fixed value of n. A crucial feature of cryptographic hash functions is *one-wayness*. Roughly speaking, a one-way function is easy to compute, but difficult to invert. In order to be cryptographically secure, a hash function H must satisfy the following conditions:

(1) Preimage resistance: given $y \in \{0,1\}^n$, it is computationally infeasible to find $x \in \{0,1\}^*$ such that $\mathrm{H}(x) = y$;
(2) Second preimage resistance: given $x \in \{0,1\}^*$, it is computationally infeasible to find $x' \in \{0,1\}^*$ such that $\mathrm{H}(x) = \mathrm{H}(x')$;
(3) Collision resistance: it is computationally infeasible to find a pair x, x', with $x \neq x'$, such that $\mathrm{H}(x) = \mathrm{H}(x')$.

Hash functions must also satisfy criteria related to diffusion and avalanche effect. Hence, a cryptographically secure hash function is supposed to be highly sensitive to small changes in its input.
Until SHA-2, hash functions were built using the Merkle-Damgård construction, which is based on the iteration of one-way compression functions, as shown in Figure 2. A compression function takes two values x, y as input and outputs a single value z. Block ciphers are commonly used in such constructions, as they are natural compression functions. In fact, a block cipher takes a key and a message as inputs and returns a ciphertext. Different ways of using block ciphers as compression functions have been proposed in the literature, such as the Davies-Meyer [10], the Matyas-Meyer-Oseas [11] and the Miyaguchi-Preneel [12] constructions.
In 2015, NIST released a new member of the Secure Hash Algorithm family, called SHA-3. It is a hash algorithm based upon the Keccak family of functions, which relies on the sponge construction, a generalization of hash functions with arbitrary-length output. A sponge construction consists of two phases: absorbing and squeezing. There is a state of $b = c + r$ bits,

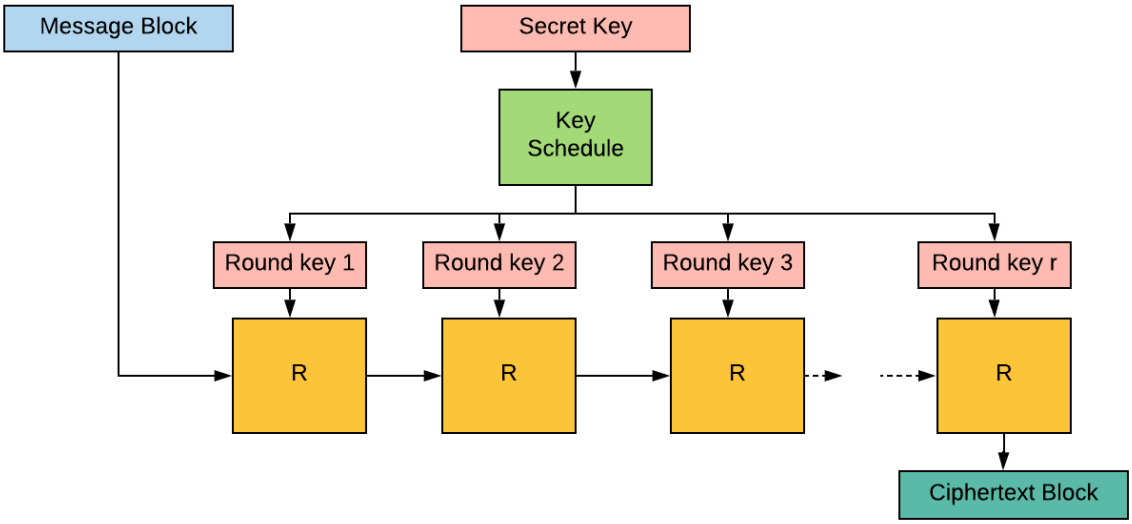

Fig. 1. Internal structure of a block cipher, based on the iteration of a round function R.

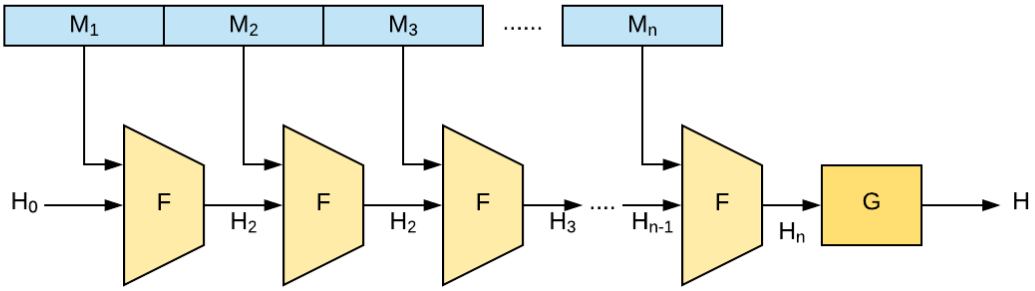

Fig. 2. The Merkle-Damgård construction, based on the iteration of a one-way compression function F. Each hash value H_i (starting from an initial value H_0) is given as input to the compression function, together with a message block in order to generate the next hash value. The last one is passed to a finalization function G, which outputs the final digest of the message.

each one initialized to zero. The first r bits of the state (called the outer state) are XORed with the r-bit message blocks, interleaved with the application of a function f, which is a fixed-length transformation or permutation. The last c bits of the state are also given as input to the function f. After all message blocks are absorbed, the squeezing phase begins. The outer state is returned as output blocks, also interleaved with applications of the function f.

Certain applications require a *keyed hash function*, which is simply an ordinary hash function that receives a secret key as input, together with the message, usually by means of concatenation. In this paper, given a hash function H and a secret key k, the keyed hash function shall be given by

$$\mathrm{H}_k(x) = \mathrm{H}(k||x||k). \tag{1}$$

An important example of keyed hash function is an HMAC (Hash-based Message Authentication Code), which is used to provide integrity and authentication. Given a message m and a secret key k, a canonical HMAC construction is given by

$$\mathrm{HMAC}(k, m) = \mathrm{H}(k'||\mathrm{H}(k''||m)), \tag{2}$$

where k' and k'' are derived from k by XORing k with constant values [13].

3. DESCRIPTION OF THE PROPOSED BLOCK CIPHER

The core idea of the proposed construction is to use a cryptographically secure hash function to generate a pseudorandom string to be combined with each message block. Each string depends on the previous ciphertext block (or the IV in the case of the first block) and a block key, derived from the master secret key. Let H be a hash function which produces N-bit digests, for a given block size N. Let $M = M_0||M_1||\cdots||M_\ell$ be a message split into blocks of N bits, after some suitable padding procedure, and k an arbitrary-length *master secret key*. We define the initialization vector (IV) as the HMAC of M and k, such that

$$\mathrm{IV} = \mathrm{HMAC}(k, M), \tag{3}$$

as defined by (2). At each step, the hash function outputs the digest of the result from the previous step, combined with a *block key* k_i, which is defined as follows:

$$k_i = \begin{cases} \mathrm{H}(k) & \text{if } i = 0 \\ \mathrm{H}_k(k_{i-1}) & \text{otherwise,} \end{cases} \tag{4}$$

where H_k is the keyed hash function defined in (1). The ciphertext consists of N-bit blocks $C_{-1}||C_0||C_1||\cdots||C_\ell$, where each block C_i is given by

$$C_i = \mathrm{H}_k(C_{i-1} \oplus k_i) \oplus M_i, \tag{5}$$

for all $i = 0, 1, \cdots, \ell$, and C_{-1} is the initialization vector IV. Figure 3 shows an overview of the encryption process.
Decryption is performed by retrieving each one of the block keys k_i and computing

$$M_i = C_i \oplus \mathrm{H}_k(C_{i-1} \oplus k_i),$$

for $i = 0, 1, \cdots, \ell$. Figure 4 shows an overview of the decryption procedure.
Note that, due to the construction of the IV as an HMAC of the message, the proposed block cipher naturally offers authenticated encryption. Also note that the proposed construction is not suitable for ECB mode, due to its intrinsic chained nature, where each message block is encrypted using a string which depends on the previous ciphertext block. As a matter of fact, the block cipher presented in this paper already contains a certain *mode of operation* inherent to its structure. This seems to be a unique feature among block ciphers, that typically describe how a single block is processed, making it mandatory to specify a mode of operation in order to encrypt and decrypt messages larger than the block size.

4. SECURITY ANALYSIS

In this section, the security of the proposed construction is discussed from a theoretical point of view. Firstly, it is shown that the proposed block cipher achieves both confusion and diffusion, due to the properties of the underlying hash function. It is also demonstrated that the proposed block cipher is provably secure, under the assumption that the underlying hash function is cryptographically secure. Finally, possible scenarios of partial information recovery attacks are discussed, showing that none of them leads to an efficient break of the cipher.
The proposed construction inherits the properties of confusion and diffusion from the underlying cryptographic hash function. As previously mentioned, confusion means that each bit of the ciphertext should depend on several parts of the key. Cryptographically secure hash functions have the property that

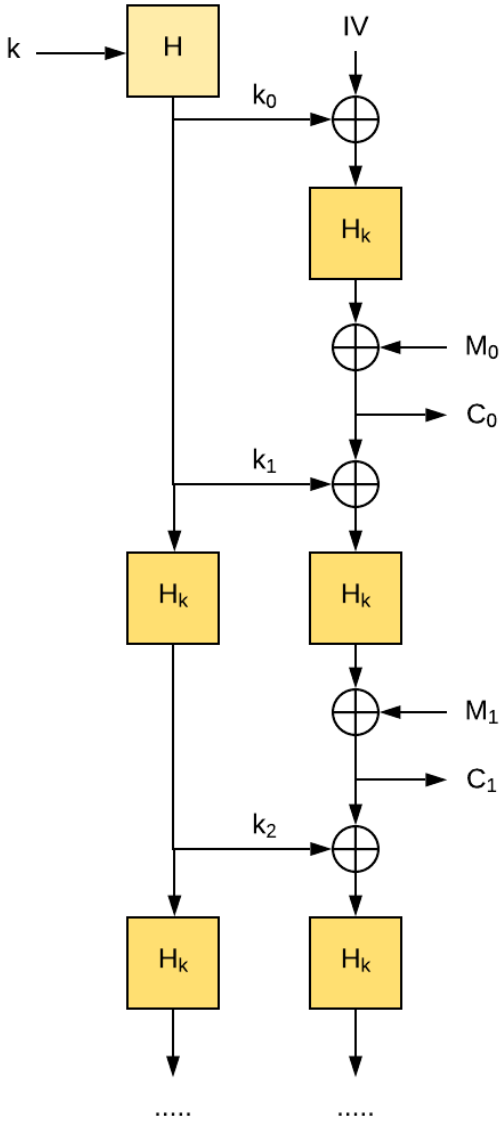

Fig. 3. Overview of the encryption process.

each output bit depends on several input bits. The proposed block cipher achieves confusion by two means: firstly by building each block key k_i from the keyed hash of the previous block key k_{i-1}. Hence, every bit of k_i depends on several parts of k; secondly, by combining each message block with the keyed hash of a value that depends on k_i to obtain the ciphertext block. Therefore, each ciphertext bit depends on several bits of k. Even if a single bit of the master secret key is flipped, it will drastically affect all of the block keys, which consequently will produce an avalanche effect over each ciphertext block.
On the other hand, diffusion means that flipping one single bit of the message should cause several ciphertext bits to flip

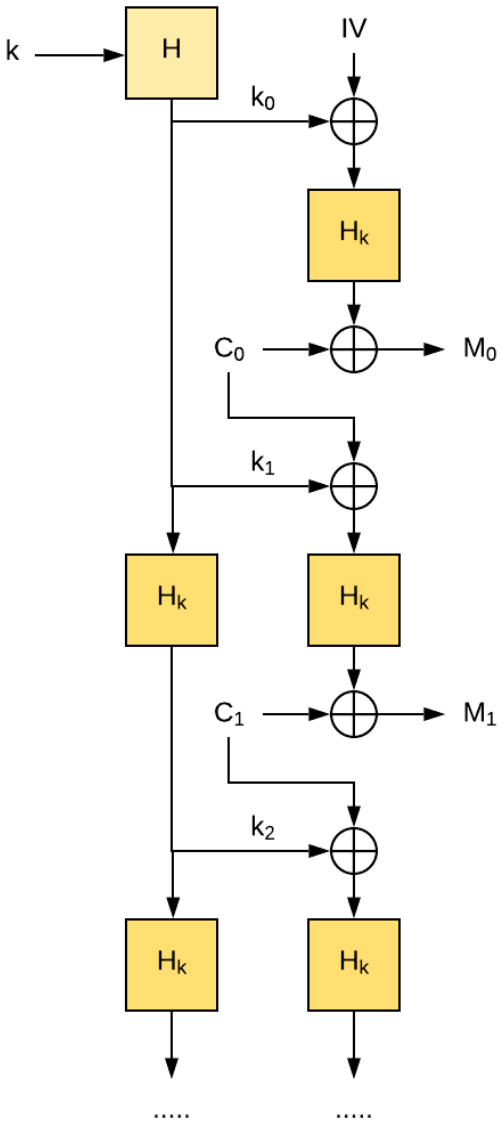

Fig. 4. Overview of the decryption process.

and vice-versa. The proposed construction achieves diffusion by building the IV as an HMAC of the message. If a single bit of the plaintext is changed, it will drastically affect the IV, which will also produce an avalanche effect over each ciphertext block due to the chained nature of the construction.

The provable security of the proposed construction can be demonstrated by showing that any algorithm that efficiently retrieves the master secret key, given access to a pair of a message and the corresponding ciphertext, yields an efficient way to find preimages and collisions for the underlying hash function.

DEFINITION 1. *Given a block size N, a key retrieving oracle is a polynomial-time algorithm $\mathcal{A}$ that takes as input a plaintext*

$$M = M_0||M_1||\cdots||M_\ell$$

and a ciphertext

$$C = C_{-1}||C_0||C_1||\cdots||C_\ell,$$

where each block of both M and C is N bits long, and outputs a master secret key k such that equations (3)*,* (4) *and* (5) *hold for all $i = 0, 1, \cdots, \ell$.*

THEOREM 2. *A key retrieving oracle for the proposed construction yields an efficient way to find preimages for the hash function* H.

Proof: let $\mathcal{A}$ be a key retrieving oracle, and Y a hash value for which a preimage X such that $\mathrm{H}(X) = Y$ is unknown. An adversary may proceed as follows: build a pair (M, C) of a plaintext and a ciphertext, choose a block index i such that $C_i \oplus M_i = Y$ and make M_j, C_j random for all $j \neq i$. Give (M, C) to the oracle, which outputs k such that

$$C_i = \mathrm{H}_k(C_{i-1} \oplus k_i) \oplus M_i.$$

Given the value of k, compute k_i and obtain $X = k||(C_{i-1} \oplus k_i)||k$, which is a preimage of Y.

□

THEOREM 3. *A key retrieving oracle for the proposed construction yields an efficient way to find collisions in the hash function* H.

Proof: let $\mathcal{A}$ be a key retrieving oracle. The goal is to find collisions $X \neq Y$ such that $\mathrm{H}(X) = \mathrm{H}(Y)$. An adversary may proceed as follows: build a pair (M, C) of a plaintext and a ciphertext, choose a block index i such that $C_{i-1} = C_i = C_{i+1}$ and $M_{i+1} = M_i$. Give (M, C) to the oracle, which outputs k such that

$$C_i = \mathrm{H}_k(C_{i-1} \oplus k_i) \oplus M_i$$

and

$$C_{i+1} = \mathrm{H}_k(C_i \oplus k_{i+1}) \oplus M_{i+1}.$$

Since $M_i = M_{i+1}$ and $C_i = C_{i+1}$,

$$\mathrm{H}_k(C_{i-1} \oplus k_i) = \mathrm{H}_k(C_i \oplus k_{i+1}).$$

Provided also that $C_i = C_{i-1}$, there are two possibilities:

(1) $k_i \neq k_{i+1}$: in this case, the desired collision has been found, namely

$$X = k||(C_{i-1} \oplus k_i)||k$$

and

$$Y = k||(C_i \oplus k_{i+1})||k.$$

(2) $k_i = k_{i+1}$: in this case, $\mathrm{H}_k(k_{i-1}) = \mathrm{H}_k(k_i)$. If $k_{i-1} = k_i$, proceed until finding the first index $j > 0$ such that $k_j = k_{j+1}$, but $k_{j-1} \neq k_j$, and the desired collision is found, namely

$$X = k||k_{j-1}||k$$

and

$$Y = k||k_j||k.$$

If there is no such index, then $k_0 = k_1$. But even in this case there is a collision, because

$$k_0 = \mathrm{H}(k) = \mathrm{H}(k||k_0||k) = k_1,$$

where, obviously, $k \neq k||k_0||k$. Hence, the desired collision is

$$X = k$$

and

$$Y = k||\mathrm{H}(k)||k.$$

□

The previous theorems show that the proposed construction is provably secure, provided that the underlying hash function is cryptographically secure. Finally, some possible scenarios of partial information recovery attacks are discussed, where the adversary is supposed to have access to some additional information besides the ciphertext.

(1) Partial recovery of the message: assume that the adversary is able to recover some message block M_i. Then he would be able to compute

$$\mathrm{H}_k(C_{i-1} \oplus k_i) = C_i \oplus M_i.$$

Provided that the hash function is cryptographically secure, the adversary cannot efficiently compute k_i or k, which gives him no efficient way to recover any other message block.

(2) Partial recovery of the key: assume that the adversary is able to recover some block key k_i. Then, he could compute

$$C_{i-1} \oplus k_i,$$

but since he does not know the master secret key k, he cannot compute $\mathrm{H}_k(C_{i-1} \oplus k_i)$ and, consequently, cannot retrieve any information about the message. Retrieving any other block key is also infeasible, because each one of them depends on the master secret key. Retrieving k is also infeasible, provided that H is preimage resistant.

(3) Recovery of all block keys k_i, for $i = 0, 1, 2, \cdots, \ell$: even in this case the adversary is not able to retrieve the master secret key. In fact, given any k_i and $\mathrm{H}_k(k_i)$, it is infeasible to recover the unknown secret key k, as long as the hash function is cryptographically secure.

5. EXPERIMENTAL ANALYSIS

This section provides experimental data that corroborate the usability and security of the proposed block cipher. The experiments were carried out with an implementation in Python 3.6.9 on Linux Mint 19.3, running on Intel Core i3-7020U processor with 4 GB of RAM.

The implementation used the package PyCryptodome. The block size was set to 256 bits, as the chosen hash algorithm was SHA-256. On the other hand, the master key size ranged from 128 to 512 bits. Master secret keys and messages were obtained with the random byte generator available in the sub-package `Crypto.Util`.

The experiments focused on key and message sensitivity, evaluating how small changes in the key or the message affect the ciphertext. The goal was to assess how much confusion and diffusion the proposed block cipher can offer.

In the first experiment, several pairs of messages differing by only one bit were encrypted under the same key, and the average Hamming distance between the resulting ciphertexts was computed. If this value was approximately equal to half the length of the ciphertext, this would be an evidence of diffusion. The experiment was performed according to the following steps:

(1) Fix a test parameter N, which is the message size, in bits;
(2) Generate a random 512-bit key k;
(3) For i in the range from 0 to $N - 1$:
   (a) Generate 100 pairs of N-bit messages, such that messages in each pair differ by exactly one bit in the position i;
   (b) Encrypt the messages in each pair, using the key k, and compute the Hamming distance between the resulting ciphertexts;
   (c) Compute the average distance d_i by adding the measured Hamming distances and dividing the result by 100;
(4) Compute the overall average distance δ by adding the d_i's and dividing the result by N.

Table 1. Results of the message sensitivity test for different values of N.

Message length (N)	Ciphertext length	δ
128	512	255.89
256	768	384.06
384	768	383.97
512	1024	511.97

For the second experiment, several pairs of keys differing by only one bit were used to encrypt the same message, and again the average Hamming distance between the resulting ciphertexts was computed. If this value was approximately equal to half the length of the ciphertext, this would be an evidence of confusion. The experiment was carried out very similarly to the first one, according to the steps below:

(1) Fix a test parameter N, which is the key size, in bits;
(2) Generate a random 512-bit message M;
(3) For i in the range from 0 to $N - 1$:
   (a) Generate 100 pairs of N-bit master keys, such that keys in each pair differ by exactly one bit in the position i;
   (b) Encrypt the message M, using the two keys in each pair, and compute the Hamming distance between the resulting ciphertexts;
   (c) Compute the average distance d_i by adding the measured Hamming distances and dividing the result by 100;
(4) Compute the overall average distance δ by adding the d_i's and dividing the result by N.

Table 2. Results of the key sensitivity test for different values of N. Since all the keys were used to encrypt the same message, the obtained ciphertext length was constant.

Key length (N)	Ciphertext length	δ
128	1024	511.92
256	1024	512.06
384	1024	511.95
512	1024	511.90

Tables 1 and 2 show evidence that the proposed block cipher offers sufficient confusion and diffusion, in the sense that small changes in the message or the key drastically affect the ciphertext.

6. CONCLUSIONS AND FINAL REMARKS

This paper presented a proposal for the construction of a block cipher from cryptographic hash functions. The proposed block cipher inherits the properties of confusion, diffusion and avalanche effect from the underlying hash function and is provably secure, in the sense that any key retrieving algorithm yields an efficient way to find preimages and collisions for the hash function.
It was shown that the proposed construction is based on the generation of block keys from the master secret key in order to encrypt each message block. It was also demonstrated that, even if an adversary manages to guess all of the block keys, he is not able to efficiently retrieve the master secret key or the message, provided that the underlying hash function is cryptographically secure.
Further research on other security aspects of the proposed construction, as well as more practical issues related to its implementation, is highly encouraged.

7. ACKNOWLEDGMENTS

Our thanks to the referees for their useful comments and suggestions that contributed to the improvement of this paper.

8. REFERENCES

[1] Ivan Bjerre Damgård. A design principle for hash functions. In Gilles Brassard, editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 416–427, New York, NY, 1990. Springer New York.

[2] National Institute of Standards and Technology. *FIPS PUB 180-1: Secure Hash Standard*. April 1995. Supersedes FIPS PUB 180 1993 May 11.

[3] National Institute of Standards and Technology. *FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*. August 2015.

[4] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Keccak. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, pages 313–314, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[5] Helena Handschuh, Helena H, and David Naccache. Shacal (- submission to Nessie -), 2000.

[6] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

[7] Ross Anderson and Eli Biham. Two practical and provably secure block ciphers: BEAR and LION. In Dieter Gollmann, editor, *Fast Software Encryption*, pages 113–120, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[8] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal, Vol 28, pp. 656–715*, Oktober 1949.

[9] A. F. Webster and Stafford E. Tavares. On the design of S-boxes. In *Advances in Cryptology*, CRYPTO '85, pages 523–534, Berlin, Heidelberg, 1985. Springer-Verlag.

[10] Bart Preneel. *Davies–Meyer Hash Function*, pages 136–136. Springer US, Boston, MA, 2005.

[11] S.M. Matyas, C.H. Meyer, and J. Oseas. Generating strong one-way functions with cryptographic algorithm. IBM Technical Disclosure Bulletin 27, 5658–5659, 1985.

[12] H. Mirvaziri, K. Jumari, M. Ismail, and M. Z. M. Hanapi. Collision-free hash function based on Miyaguchi-Preneel and enhanced Merkle-Damgård scheme. In *2007 5th Student Conference on Research and Development*, pages 1–6, Dec 2007.

[13] Dr. Hugo Krawczyk, Mihir Bellare, and Ran Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, February 1997.