International Journal of Computer Science and Applications

Vol. 14, No. 2, pp. 164 - 180, 2017

Keyed- CAHASH: a New Fast Keyed Hash Function based on Cellular Automata for Authentication

Bouchra Echandouri

Department of Computer Science,Laboratory of computer science research (LRI)
Mohammed V University in Rabat, Faculty of sciences
BP1014 RP, Rabat, Morocco
bouchra.echandouri@gmail.com

Charifa Hanin

Department of Computer Science, Laboratory of computer science research (LRI)
Laboratory of Mathematics, Computing and Applications (LabMiA)
Mohammed V University in Rabat, Faculty of sciences
BP1014 RP, Rabat, Morocco
charifa.hanin@gmail.com

Fouzia Omary

Department of Computer Science, Laboratory of computer science research (LRI)
Mohammed V University in Rabat, Faculty of sciences
BP1014 RP, Rabat, Morocco
omary@fsr.ac.ma

Souad Elbernoussi

Department of Mathematics, Laboratory of Mathematics, Computing and Applications (LabMiA)
Mohammed V University in Rabat, Faculty of sciences
BP1014 RP, Rabat, Morocco
s.elbernoussi@fsr.ac.ma

Private data transmission over arbitrary canal is considered very risky. Hence, the need for designing robust cryptographic solutions ensuring both data integrity and authentication is very attempting. Therefore, Message Authentication Code (MAC) is one of the most provably secure algorithms that ensure integrity and authentication. Using a shared secret key, the receiver checks if an alteration has happened during transmission. This MAC may be derived from hash functions using a secret key. Since it is also easily implemented, this paper concerns the design of a new keyed hash function based on cellular automata. Our suggested MAC process provides a high efficiency and robustness against MAC forgery and key recovery attacks comparing to well-known HMACs standards. Further, the obtained results show that it satisfies data integrity and authentication requirement.

Keywords: privacy; cryptography; information security; data integrity; authentication; hash function; message authentication code; HMAC; keyed hash function; MAC forgery attack; key recovery attack; cellular automata.

1. Introduction

The communication of sensitive data within an arbitrary canal is very risky, since any unauthorized entity can intercept the transmission. Hence, the mechanism that helps secure data transmission is cryptography that is defined as the science that concerns coding and decoding messages. It proposes robust tools ensuring data privacy. Further, data integrity and authentication are essential for secure networks communication as they are the basis of security in many distributed systems.
Wherefore, Message authentication code (MAC) has been described as one of the most robust cryptographic solutions ensuring both data integrity and authentication. The generation MAC algorithm uses the secret key and the message to produce the checksum (called also Tag). It allows the receiver of the Tag to verify its integrity and authentication by verifying if an alteration has happened during transmission. When verifying, the verification MAC algorithm uses the same key and the message to reconstruct the valid Tag if no alteration has affected the message, otherwise it is invalid and has been altered [Smart (2016)].
Particularly, to conceive a MAC there are many ways, one using a symmetric encryption algorithm [Smart (2016)], especially block cipher-based message authentication code (CMAC) [Dworkin (2005)]and cipher block chaining message authentication code (CBC-MAC) [Frankel et al. (2003)]. The CBC-MAC is a mechanism that helps constructing a MAC using a block cipher such as each block depends on the encryption of its previous block. Therefore, the change of one bit in the plaintext will cause a change in the final encrypted block and be unpredictable without knowing a priori the key [Smart (2016)]. These MACs resistance is proved using a strongly secure symmetric encryption algorithm [NIST (2010)]. Additionally, the second using on a keyed hash function [Krawczyk et al. (1997)]. This HMAC is widely used in numerous network protocols such as SSH, IPSec, TLS etc. [Dang (2008)]. Further, the authenticity provided by HMAC offers a high security level and prevents against any malicious entities to forge new modified tag that is valid. Any new maliciously constructed tag is immediately rejected after the verification phase, so the acceptance of the tag become only possible for the rightly constructed tag using the valid key and the valid message [Boyd et al. (2013)].
In order to authenticate data over an arbitrary insecure canal, the main intention of this work is to present a new keyed hash function based on cellular automata, combined with the use of the secret key generation mechanism 'PSOCA' [Hanin et al. (2016)], that is ascertained fast and robust comparing to well-known hash based message authentication .
Cellular automata, firstly introduced by Von Neumann, is defined as a dynamical deterministic system composed of a set of cells, where each cell changes its own state, following a determined set of rules. Particularly, it was used to design efficient number of cryptographic systems due to its properties as homogeneity, locality, parallelism, simplicity and unpredictability [Wolfram (2002)]. Furthermore, it provides a hardware implementation facility, a high randomness quality, and a fast compu-

tation [Shin et al. (2012)]. Moreover, to provide more randomness and robustness against key recovery attack to our key generation approach, 'PSOCA' is used since it is declared secure in [Hanin et al. (2016)]. Accordingly, Hanin et al. conceived this latter [Hanin et al. (2016)], a new pseudo random number generator, based on non-uniform cellular automata combined with a modified binary particle swarm optimization algorithm (MBPSO) to obtain an optimal set of rules. In addition, our suggested MAC provides a high efficiency and robustness against MAC forgery attack. Further, the experimental analysis and the obtained results witness that our proposed MAC meets the data integrity and authentication requirement.
The rest of this paper is organized as follows: In section II the cellular automata and MAC concepts are introduced. Afterwards, a highlight of some related work in the literature is briefly described. Then, in section III, our proposed Message authentication code system overview is given, followed by its Experimental results and Security analysis in the Section VI. Finally, the paper is concluded providing some future work, in the last section.

2. Preliminaries

2.1. *Hash based Message Authentication Code*

A Message Authentication Code (MAC) is a secret key algorithm that ensures data integrity and authentication. That involves the use of a secret key k and two algorithms(eq. 1, 2), namely a MAC generation algorithm $MACG_k$ and a MAC verification algorithm $MACV_k$. The $MACG_k$ algorithm takes an arbitrary message M and generates a unique fixed length termed Tag or checksum, the generated tag is joined to the message M to construct an authenticated message (see figure 1).

$$Tag = MACG_k(M) \tag{1}$$

Therefore, the $MACV_k$ algorithm uses the same key k and the message M to reconstruct the valid Tag (eq. 2) if no alteration has affected the message, otherwise it is invalid and has been altered [Smart (2016)] (see figure 1).

$$MACV_k(M, tag) = \{valid, invalid\} \tag{2}$$

Generally, there are two major methods to construct MAC algorithm, the first is by using block ciphers [Smart (2016)], such as block cipher-based message authentication code (CMAC) [Dworkin (2005)] and cipher block chaining message authentication code (CBC-MAC) [Frankel et al. (2003)].
Moreover, the second is based on hash function, called also keyed hash function [Krawczyk et al. (1997)]. This algorithm involves a key k from a set of secret keys K that are combined with hash functions H to generate the checksum. Thus, they are families of hash functions providing authentication by the use of a key [Hans et al. (2015)], expressed as follows:

$$H_{k\in K} : \{0,1\}^* \rightarrow: \{0,1\}^n \tag{3}$$

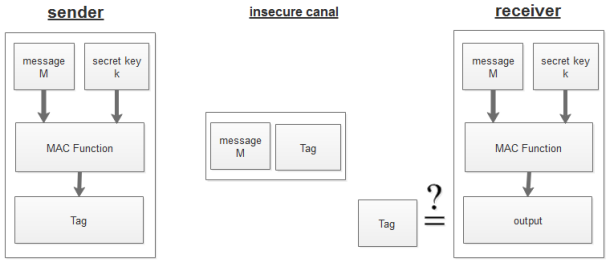

Fig. 1. MAC protocol.

These functions (eq. 3) map an arbitrary length size input message M using the key into a fixed length size output.
A Secure MAC based hash function, termed MAC(), that maps a key k and a message m is a secure keyed hash function if it fulfill these five required properties[Boyd et al. (2013)].

- Given k and m, it is easy to generate MAC(k, m);
- Given k and MAC(k,m), it is computationally infeasible to find m;
- Given k it is computationally infeasible to find two different values m and m' such that MAC(k,m) =MAC(k,m');
- Given (possibly many) pairs of m and MAC(k, m), it is computationally infeasible to compute k;
- Without a prior knowledge of k, it is computationally infeasible to compute MAC(k, m) for any m.

MAC based hash function robustness against attacks resides in the robust cryptographic properties of the hash function H.
A proven secure hash function with inputs m, m' and outputs h, h', should satisfy a three required properties, i.e. the pre-image resistant, 2nd-preimage resistant and collision resistant [Smart (2016)]:

- Pre-image resistance: it is easy to generate h value given a message m, but impossible to find a message m given a hash value h;
- 2nd-preimage resistance: it is computationally infeasible to find another input message m' with the same output hash value h=h' ;
- Collision resistance: it is computationally infeasible to find any pair of messages m and m' that hash the same output h=h'.

2.2. *Cellular Automata*

Von Neumann and Ulam pioneered the concept of cellular automata(CA) in 1940 to study biological manners such as self-replicating [Neumann et al. (1966)]. Afterwards, in 1980, CA was developed by Stephen Wolfram [Wolfram (2002)], in the aim

	7	6	5	4	3	2	1	0
N_i^t	111	110	101	100	011	010	001	000
$f(N_i^t)$	0	0	0	1	0	1	1	1

Table 1. Representation of rule 23 with 3-neighborhood (r = 1)

to generate pseudo random number [Wolfram (1986)]. A CA is a dynamical system that can be viewed as an array of cells, where each cell's state interact with a set of cells' state called neighbors, according to a transition function. It has several characteristics such as homogeneity, locality, parallelism, simplicity and unpredictability [Wolfram (2002)], that help facilitate the hardware and software implementations [J. N. Rao et al. (2012)]. These properties allows to CA to become an efficient tool to develop cryptographic systems, especially to design a robust keyed hash function. Mathematically, a CA is defined as a quadruple A= {S,D,N,f} where:

- S is the sate set determining the set of possible cells' states.
- D is the dimension or the size of CA
- N is the neighboring states following the existing neighborhood structure. It is worth noting that the classical neighbors' type are 3-neighbrhood, Von Neumann and Moore Neighborhood [J. N. Rao et al. (2012)]
- f is the transition rule(eq. 4). It is defined as a Boolean function $f : S^n \rightarrow S$ that maps one state to its next state. Particularly, the state transition at a time (t+1) , denoted s_i^{t+1} ,is expressed by the following equation:

$$s_i^{t+1} = f(s_{i-r}^t, s_{i-r+1}^t, ..., s_i^t, ..., s_{i+r-1}^t, s_{i+1}^t) = f(N_i^t). \tag{4}$$

Where r is the neighborhood radius and N_i^t are the neighbors of the i^{th} cell at a time t.
It is worth noting that this transition rule depends only on the possible state values and the neighborhood radius r that defines the number of neighbors.
Also, this function can be represented as a truth table or as a decimal number termed rule number, showed in (table 1).

2.2.1. *Boundary Conditions*

At a certain moment especially in the extremities of the cells' space, cells need to interact with neighbors cells. Hence, it is necessary to define the appropriate boundary conditions of a CA. Especially; there exist different types of boundary conditions, namely null boundary conditions, periodic boundary conditions and others[Wolfram (2002)]. Particularly, a CA with a periodic boundary has adjacent extreme cells one to each other and a CA with null boundary has extreme cells connected to logic 0.

2.2.2. *Definitions*

- Uniform CA : If all the cells obey the same rule. Otherwise, it is non-uniform.
- Linear CA: if the rule of a CA cell involves only Xor logic, then it is called a linear rule. A CA with all the cells having linear rules is called a linear CA.
- Nonlinear CA : if the rule of a CA cell involves only AND-OR logic, then it is called a nonlinear rule. A CA with all the cells having nonlinear rules is called a linear CA.
- Group and non-group CA: If a state transition of a CA contains only cyclic states, then the CA is called a group CA; otherwise it is a non-group CA. The rule applied on a uniform group CA is called a group rule; otherwise it is a non-group rule.

Accordingly, cellular automata's logical operation, high speed and low hardware complexity inspired us to propose the design of a new keyed hash function.
In what follows some interesting works on hash based message authentication code that have been performed are highlighted.

2.3. *Related Works*

Hash based MACs are one of the most paramount studied primitives in modern cryptography. Particularly, in practice numerous constructions have been developed.
In [Teng (2016)],Teng and Gong proposed a new message authentication code that focused on the constructions based on universal hash functions such as almost perfect nonlinear(APN) functions. Their algorithm has provable security against substitution, cycling and linear forgery attacks. Furthermore, another MAC scheme using a fast and simple construction of a PRF(Pseudo Random Function) from an Merkle-Damgrd hash function, called AMAC, was proven more efficient than HMAC especially for short messages [Bellare (2016)]. In [Krawczyk et al. (1997)] Krawczyk et al. published the standard HMAC that is derived from a hash function using Merkle-Damgard's method in combination with a shared secret key. This proposed standard gets almost all its security from the elementary hash function used as a black box.
In the work [Bertoni (2013)], the authors introduced Keccak, the SHA-3 standard winner, that helps provide a Message Authentication Codes. The Tag is produced using the keccak hashing mechanism of the concatenation of the key and the input message [Taha (2013)].Also, Auth256 [Bernstein (2014)] is derived from the hash function Hash256 using a secret key that is obtained using counter-mode AES.
Moreover, in [Shin et al. (2012)] Yoo et al. introduced a lightweight authentication scheme based on cellular automata for multi-user in the cloud context that is computed using a one-time password.

Furthermore, in 1995, XOR-MACs have been proposed by Bellare et al. to obtain randomized MACs. Their described construction uses a PRF by applying XOR operation on outputs blocks together [Mihir (1995)].
Thus, the attacks on these aforesaid keyed hash functions have inspired us to propose the design of a new keyed hash function that is based on Cellular Automata and helps meet data integrity and authentication requirements.

3. Our Proposed Keyed Hash Scheme

Our purpose is to design a secure message authentication function performed by a new keyed hash function using on the new mechanism of secret key generation 'PSOCA' proposed in [Hanin et al. (2016)] that are both based on cellular automata. Our suggested message authentication code MAC = { Subkey Generation, Generate-Keyed-CAHASH, Verify-Keyed-CAHASH } is a triple of algorithms with associated key space K, message space M, and Tag space T .

3.1. *Subkeys Generation*

Firstly, a set of used secret keys $\{sk_1, sk_2, sk_{index}, sk_p\}$is generated (see Algorithm 3.1), namely sk_1 and sk_2 are generated using the pseudo random number generator 'PSOCA' [Hanin et al. (2016)].
Pseudo random number generators (PRNGs) are widely used in several fields of cryptography especially for key generation. Hence, different ways are proposed to conceive PRNGs like cellular automata. By exploiting their properties namely homogeneity, locality, parallelism, simplicity and unpredictability, CA are good candidate to generate a high quality of PRNGs. In order to select a new optimal rules combination for CA, Hanin et al. [Hanin et al. (2016)] proposed a new PRNG called 'PSOCA' based on non-uniform CA combined with a modified binary particle swarm optimization (MBPSO). 'PSOCA' algorithm takes a binary number as input (i.e. the seed) to produce a sequence of keys from this seed.
In order to obtain subkeys $Sk_s = \{sk, sk_1, sk_2\}$, 'PSOCA' takes two parameters, a seed (s) and key size (k_z) to provide the first $512-bit$ subkey length sk_1. Thereafter, 'PSOCA' is evolved to generate a $512 - bit$ length sk_2 from sk_1. Further, sk_p is a secret chosen prime number with a specified number of bits (i.e. 13 bits in our case) that purpose to compress the message without loss of information and $sk_{index}$is a secret chosen index from $[0, n]$. (See Algorithm 3.1)

Algorithm 3.1. *Subkeys Generation(s, k_z,n)*
Input: seed s, key size k_z,number of blocks n
Output: sk_1, sk_2, sk_p , sk_{index}
$sk_1 \leftarrow$ PSOCA(s,k_z) [Hanin et al. (2016)]
$sk_2 \leftarrow$ PSOCA(sk_1, k_z) [Hanin et al. (2016)]
$sk_p \leftarrow$ choose a prime number
$sk_{index} \leftarrow$ choose an index $\in [0, 1]$

For each session, Once the subkeys $sk = \{sk_1, sk_2, sk_{index}, sk_p\}$, are generated and shared secretly (i.e. over a secure medium) between the legitimate senders and receivers.

3.2. *MAC Generation Algorithm (Generate-Keyed-CAHASH)*

The sender may generate a Tag using our proposed keyed hash algorithm(see figure 2). This probabilistic authentication algorithm takes as input a message m∈M, a secret key k∈K and outputs an authentication Tag ∈ T.(see Algorithm 3.2)

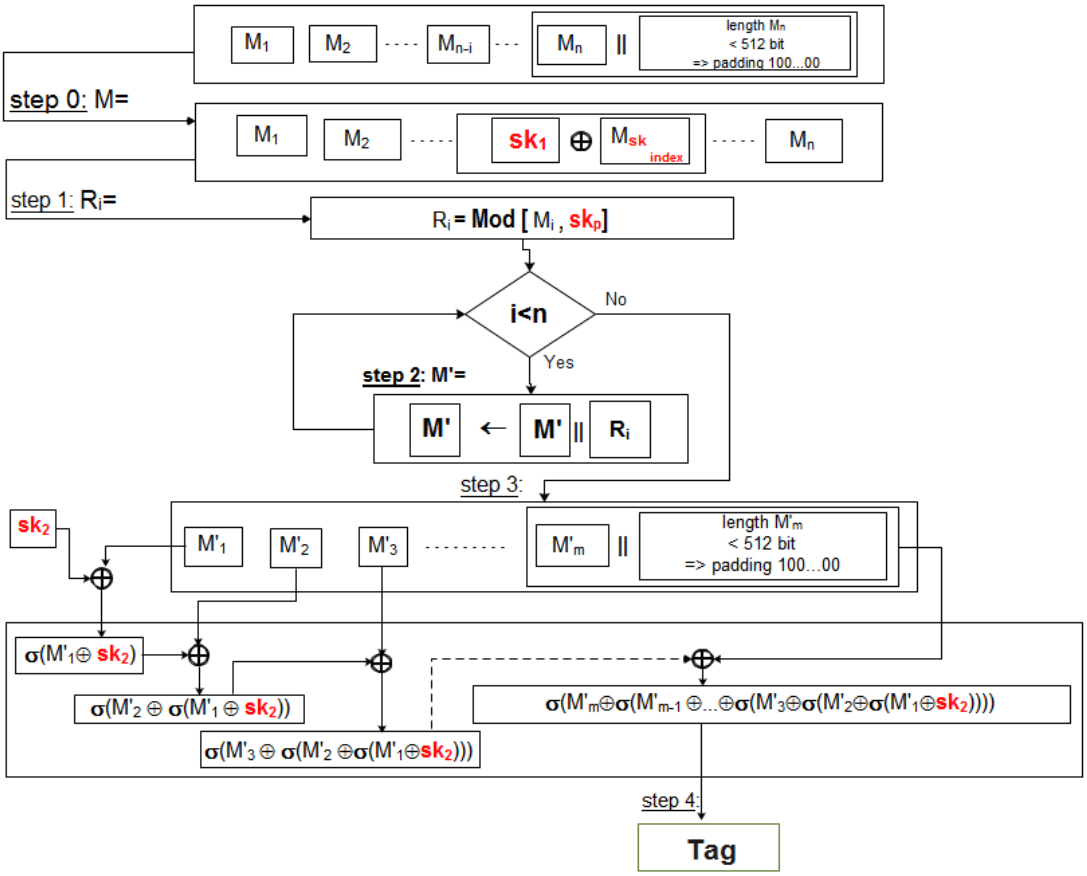

Fig. 2. Our proposed keyed hash generation algorithm (Keyed-CAHASH).

Algorithm 3.2. ***Generate-Keyed-CAHASH*** $(M, sk_1, sk_2, sk_{index}, sk_p)$
Input: Message $M, sk_1, sk_2, sk_{index}, sk_p$
Variable: n, m
Output: Tag
Split M into n blocks of 512 bit
If M≠ multiple of 512 bit Then
Padd M
End if
$M_{sk_{index}} \leftarrow M_{sk_{index}} \oplus sk_1$
For i ←1 to n
$R_i \leftarrow Mod[M_i, sk_p]$
End For
M'← $R_1 \| R_2 \| ... \| R_n$
Split M' into m blocks of 512 bit
If M' ≠ multiple of 512 bit Then
Padd M'
End if
$Tag \leftarrow \sigma(sk_2 \oplus M'_1)$
For i ← 2 *to m*
$Tag \leftarrow \sigma(M'_i \oplus Tag)$;
End for

3.2.1. *Description of function* $\sigma()$

The function $\sigma()$ is the global function, defined as $\sigma : \{0,1\}^* \rightarrow \{0,1\}^*$,that returns all cells' state at the next time step through the evolution of the CA as $x^{t+1} = \sigma(x^t)$, where each cell evolves according to one of rules 23 and 150. These rules are both one-dimensional CA named 'Rule 23' and 'Rule150' respectively (see table 2). Especially, the logic functions of used rules are described as follows: [Wolfram (1986)]
'Rule 150': $s_i^{t+1} = s_{i-1}^t \oplus s_i^t \oplus s_{i+1}^t$
'Rule 23': $s_i^{t+1} = (\bar{s^t}_{i-1} \odot \bar{s^t}_{i+1}) \oplus (\bar{s^t}_i \odot \bar{s^t}_{i+1}) \oplus (\bar{s^t}_{i-1} \odot \bar{s^t}_i)$
(where $\oplus$ and $\odot$ denotes the bitwise XOR, AND operations.)

Rule name	111	110	101	100	011	010	001	000
Rule 23	0	0	0	1	0	1	1	1
Rule 150	1	0	0	1	0	1	1	0

Table 2. Representation of rule 23 and 150 with 3-neighborhood (r = 1)

3.2.2. *Description of our Proposed Function*

Our proposed MAC generation algorithm Generate-Keyed-CAHASH() (see figure 2 and see Algorithm 3.2) begins with splitting the input message M of arbitrary length L, into n fixed-length of 512-bit blocks M_i ,using also a padding operation, when needed. Afterwards, the message becomes $M = M_1, ..., M_n$ with M_i a 512-bit block. Then, "xor" operation is performed to the chosen $M_{sk_{index}}$, from the message M , with the first subkey sk_1. Therefore, our proposed compression function consists on computing each modulo sk_p of M_i, denoted by R_i. Then, M', the concatenation of all residuesR_i , is split into 512-bit blocks M'_i , using also a padding operation, when needed. Thereafter, the result performed by the 'xor' operation on the second subkey sk_2 with the first bloc of M' (i.e. $(M'_1 \oplus sk_2)$) is taken as input parameter for the denoted evolution function $\sigma()$ described above .So on, the overall output is evolved where each previous output is then performed using the "xor" operation with all the previous result, until the last iteration (i.e.($M'_m \oplus ... \oplus \sigma(M'_2 \oplus \sigma(M'_1 \oplus sk_2))$) to generate the authenticated message, denoted Tag (eq. 5). The output of this MAC generation algorithm is a 512-bit length that is used to ascertain the input message.

$$Tag := Generate - keyed - CAHASH(M, sk_1, sk_2, sk_{index}, sk_p) \tag{5}$$

3.3. *MAC Verification Algorithm (Verify-Keyed-CAHASH)*

This deterministic verification algorithm takes as input a message m ∈ M, a secret key k ∈ K and a Tag∈T and outputs an element of the set {valid, invalid}.

Algorithm 3.3. ***Verify-Keyed-CAHASH***
Input: $M, sk_1, sk_2, sk_{index}, sk_p, Tag$
Output: $Valid_{Tag}, Invalid_{Tag}$
Tag' ← $Generate\text{-}Keyed\text{-}CAHASH(M, sk_1, sk_2, sk_{index}, sk_p);$
If Tag' = Tag Then
return $Valid_{Tag};$
else
return $Invalid_{Tag};$
End if

Once the receiver gets the Tag, he recalculates the MAC generation algorithm. The MAC verification algorithm, our proposed Verify-keyed-CAHASH() (see Algorithm 3.3), takes as inputs the message M, the secret keys $\{sk_1, sk_2, sk_{index}, sk_p\}$ and the received Tag (eq.5). The output of the MAC verification algorithm is either INVALID or VALID. If the two Tags match, the sender is authenticated, and the message's integrity is guaranteed.

4. Security Analysis

Generally, the security of a hash based message authentication code leans on the robust cryptographic properties of the hash function. Especially, the collision resis-

tance and compression property [Krawczyk et al. (1997)].
Moreover, a malicious entity, aiming to provide a valid tag value trying to inject a counterfeit message, should have a time complexity that is set to polynomial [Shaoquan (2015)]. Hence, this malicious entity can take one of two attacks, namely key recovery and forgery attack. Especially, the Key recovery attack consists of recovering the secret key from a number of message/Tag pairs, requiring about 2^k operations (k is the secret key bit-length). Such attack is powerful than the forgery one where the malicious entity constructs at least one valid message/Tag pair without knowing the secret key. An existential MAC forgery is said when this attack is possible for a single message. Further, it is said selective MAC forgery if it is possible for a Tag of his choice [Zhu (2015)].
The well-known HMAC suffers from the problem that if a collision in the hash function can be found, its hash based MAC is weak and the generated tag could be forged. Accordingly, In order to design our proposed keyed hash function, a combination of a linear group rule and a nonlinear group rule are used. Particularly, the linear group rule provides the collision resistance and the nonlinear non-group rule provides one-way property [Jeon (2013)]. Especially Rule 150, based on periodic boundary condition, has the highest dependency from neighborhood comparing to almost all linear rules. Also, the rule 23 provides both a high non-linearity and a special transition form [Jeon (2013)]. Combining these two rules (150 and 23) guarantee both, linearity as well as non-linearity, that are the most important properties in cryptography. Hence, low linearity and maximum non-linearity ensure respectively resistance of differential and linear cryptanalysis [[Eli (1991)],[Mitsuru (1994)]].

4.1. *A key Recovery Attack*

One of the main security aspects of message authentication code, especially the keyed hash function, is to be strongly resistant against key recovery attack.

4.1.1. *Exhaustive Research Key*

This attack is one of attacks where a malicious entity tries obtaining all possible valid keys using a predicting tool.
Accordingly, taking into account a high performing machine that needs (10^{-10}) seconds to test the subkeys' validity. Then, given two 512-bit length subkeys, our algorithm has $(2 * 2^{512})$ possible keys, n possible index and 2^{12} possible prime number. Therefore, exhaustive research key attack requires approximately $(10^{-10} * (2 * 2^{512} + n + 2^{12}) = n * 10^{-10} + 2.681562e + 144)$ seconds (i.e more than $(8.497531700777161e + 136)$ years) is required to get the right key. Thus, an exhaustive key search is computationally infeasible [Zhu (2015)].

4.1.2. *Key Sensitivity to One-bit Changes*

In order to estimate the key sensitivity level of our proposed keyed hash. The following experimentation is performed using the avalanche effect notion that defines the capability of a function to spread a little change in the input through the output (i.e. if a one bit changed in the initial message input, it affects the output). Especially, The average of all resulting Hamming distance of all obtained output should be half of the output size (i.e (Hamming$(H(x), H(y)) = n/2$) [J. C. H (2005)]
Therefore, the Tag is performed using keyed-CAHASH. Figure 3 shows the average of hamming distance of output bits changes in the key, given a random one bit difference in the input, using a set of 1024 pairs of keys. It shows that the change in one bit of the input key affects $\simeq 50\%$ change in the output Tag.

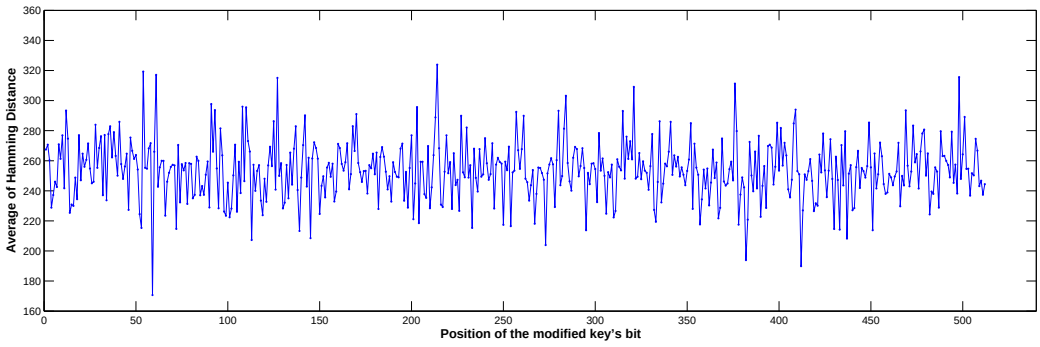

Fig. 3. Original key Sensitivity to one-bit changes.

Furthermore, the presented results in the paper [Hanin et al. (2016)] proved that 'PSOCA' has efficient properties of pseudo random number generator. Thus, the use of 'PSOCA' in our proposed keyed hash function helps generate subkeys that are unpredictable and indistinguishable.

4.2. *MAC Forgery Attack*

Forgery attack is the most known attack on MACs function, where the malicious entity can provide at least a valid message/tag pair without holding the secret key[Zhu (2015)].
Mainly, the strongest known attack on HMAC, birthday attack, is based on the frequency of collisions [Bellare (1996)]. It is a type of forgery attack on MACs that proves the maximum limit on their security, especially to those based on iterative hash functions. This attack requires using about $O(2^{n/2})$message/tag known pairs (i.e. n as the tag length)[Marsaglia (1996)].

	A_{min}	A_{max}	A_{mean}	A(%)
HMAC(Sha2-512)	323.60	350.00	337.74	65.9027
HMAC(MD5)	76.90	91.50	84.92	66.3489
Our proposed-512	215.90	298.50	255.71	49.94

Table 3. Our proposed keyed-CAHACH performance comparing to well-known HMACs

Since our keyed hash function is described as a MAC based on iterated evolutions. Then, to succeed a MAC forgery attack on our proposed approach, a malicious entity needs to know about $O(2^{256})$message/tag [Marsaglia (1996)].

4.2.1. *Original Message Sensitivity to One-bit Changes*

Using the avalanche effect notion [J. C. H (2005)]. A message input M is taken and its Tag is performed using keyed-CAHASH. The experimentations result show the average of hamming distance of output bits changed, given a random one bit difference in the input, using a set of 1024 pairs of messages (see figure 4).
The table 3 presents avalanche effect values ($A_{min}, A_{max}, A_{mean}$,A(%)) of the change in one bit of the input. Nearly, $\simeq$50% in our output tag files are affected by the change comparing to well-known HMACs.

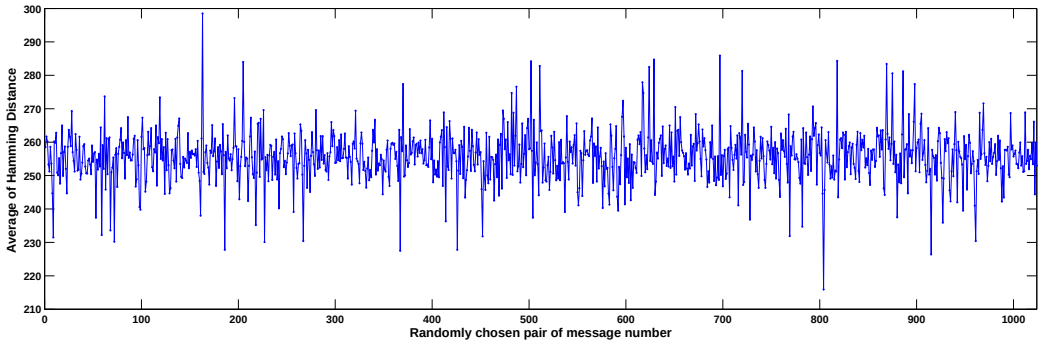

Fig. 4. Original message Sensitivity to one-bit changes.

4.2.2. *Statistical Tests*

Fundamentally, the transferred tag should be indistinguishable from any different tag. In what follows, our proposed keyed hash function's randomness quality is studied by checking if it bypasses standard statistical attacks. Generally, a secure hash function's output should be statistically indistinguishable from a random function's

Test name	Number of P-value	Interpretation
diehard birthdays	0.90315496	PASSED
diehard operm5	0.87063499	PASSED
diehard rank 32x32	0.98995661	PASSED
diehard rank 6x8	0.29111246	PASSED
diehard bitstream	0.91261990	PASSED
diehard opso	0.43442292	PASSED
diehard oqso	0.53436364	PASSED
diehard dna	0.50201120	PASSED
diehard count 1s str	0.67388288	PASSED
diehard count 1s byt	0.37199848	PASSED
diehard parking lot	0.91507220	PASSED
diehard 2dsphere	0.86872396	PASSED
diehard squeeze	0.37402560	PASSED
diehard sums	0.03667907	PASSED
diehard runs	0.93794343	PASSED
diehard craps	0.38196019	PASSED
sts runs	0.56014598	PASSED

Table 4. Diehard Tests' result on our approach

output. For this manner, there is a number of tests batteries aiming to evaluate the randomness of such algorithm as DIEHARD test [Marsaglia (1997)].
The DIEHARD test contains a package of various statistical tests, where each test has its own evaluating tools. These tests are considered strong at verifying the randomness up to an extreme level. To achieve good results it verifies the random numbers' p-value that is in the range $0.025 < p < 0.975$. Hence, for our approach experiments, our keyed hash function has been generating many tags in a way to obtain a 10Mb data stream. This stream is created using a counter mode scheme employing our proposed MAC generation algorithm to compute successive output values that are concatenated to create the data stream. Thereafter, this produced stream is statistically analyzed using Diehard [Marsaglia (1997)]. Then, these final results are averaged and then reported in the table 4.
It is worth noting that the binary stream obtained using our proposed design has successfully passed almost all essential DIEHARD tests. Furthermore, our proposed keyed hash function ascertains a good randomness behavior and could be statistically indistinguishable from random sequences.

5. Performances Analysis and Comparison

The main purpose of our work is to propose an efficient new keyed hash function with high computation speed that ensures data integrity and authentication. Our new keyed hash function based on cellular automata (keyed-CAHASH) was turned

Message size (Kilo byte)	HMAC(Sha2-512) (s)	HMAC(MD5) (s)	KMAC(KeccakMAC-512) (s)	keyed-CAHASH (s)
1	0.0136	0.0132	0.01257	0.0110
10	0.1981	0.1905	0.03318	0.02262
20	0.3850	0.2812	0.04041	0.04871

Table 5. Speediness Comparison between our keyed-CAHASH MAC function and others HMACs

on an Intel Core i5-34227, OS 64-bit, 1.8Ghz processor with 4 GB RAM. The table 5 shows that our proposed algorithm is fast comparing to other well-known hash based message authentication code while achieving a satisfying performances using simple software implementations.

6. Conclusion

HMAC is widely used in numerous network protocols such as SSH, IPSec, TLS etc. [Dang (2008)]. Further, the authenticity provided by HMAC offers a high security level against forgery attacks [Boyd et al. (2013)]. In order to authenticate data over an arbitrary insecure canal, the main intention of this work is to present a new keyed hash function based on cellular automata.
As ascertained above, the main advantage of our proposed approach is that it is fast and robust against MAC forgery attacks comparing to well-known hash based message authentication codes. Besides ensuring authentication and data integrity, in terms of future work, an extension of this suggested scheme is intended by providing the non-repudiation property through conceiving a digital signature [SMART (2016)].

References

Bellare, Mihir, Daniel J. Bernstein, and Stefano Tessaro(2016). Hash-function based PRFs: AMAC and its multi-user security. *Annual International Conference on the Theory and Applications of Cryptographic Techniques,Springer Berlin Heidelberg*, **pp: 566-595**.

Bellare, Mihir, Ran Canetti, and Hugo Krawczyk(1996). Keying hash functions for message authentication. *Annual International Cryptology Conference*, Springer Berlin Heidelberg,**pp. 1-15**.

Bernstein, Daniel J., and Tung Chou (2014). Faster binary-field multiplication and faster binary-field MACs.*International Workshop on Selected Areas in Cryptography. Springer International Publishing.*

Bertoni, G., Daemen, J., Peeters, M., & Van Assche, G. (2013). Keccak. *Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer Berlin Heidelberg,***(pp. 313-314)**

Boyd, Colin, and Anish Mathuria.(2013).Hash Functions, Message Authentication Codes and Key Derivation Functions *Springer Science & Business Media.*

Dang, Quynh. (2008). Recommendation for applications using approved hash algorithms. *US Department of Commerce, National Institute of Standards and Technology.*

Dworkin, Morris(2005).Hash Functions, Message Authentication Codes and Key Derivation Functions *NIST Special Publication.*

Eli Biham and Adi Shamir(1991). Differential cryptanalysis of des-like cryptosystems. *Journal of Cryptology 4(1)*, **pp:3-72**.

Frankel, S. and H. Herbert.(2003).The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec. *No. RFC 3566.*

C. Hanin, F. Omary, S. Elbernoussi and B. Boulahiat.(2016). Design of New Pseudo-Random Number Generator Based on Non-Uniform Cellular Automata. *International Journal of Security and Its Applications.*, **Vol. 10, No. 11**.

Hans Delfs, Helmut Knebl. (2015).Introduction to cryptography, Information Security and Cryptography *Springer-Verlag, Berlin* .

Jeon, Jun-Cheol(2013). Analysis of hash functions and cellular automata based schemes. *International Journal of Security and Its Applications 7, no. 3* **pp: 303-316**.

J. C. H. CASTROA, J.M. SIERRAB, A. SEZNECA, A. IZQUIERDOA, A. RIBAGORDAA(2005). The strict avalanche criterion randomness test.*Mathematics and Computers in Simulation, 68*, **pp: 1-7**

J. N. Rao, and A. C. Singh (2012). A novel encryption system using layered cellular automata, *International Journal of Engineering Research and Applications, vol. 2, no. 6*,**pp: 912-917**.

Krawczyk, Hugo, Ran Canetti, and Mihir Bellare. (1997). HMAC: Keyed-hashing for message authentication. *Cryptography Made Simple. Springer International Publishing*, **https://tools.ietf.org/html/rfc2104 Last visited: april 2017**.

Marsaglia G. Computer code DIEHARD, available at, *http://stat.fsu.edu/pub/diehard/ Last visited: feb 2017*

Menezes, Alfred J., Paul C. Van Oorschot, and Scott A. Vanstone(1996). Handbook of applied cryptography. CRC press, ***pp352****.*

Mihir Bellare, Roch Guerin, and Phillip Rogaway (1995). XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions. In Don Coppersmith, editor, CRYPTO, volume 963 of LNCS, ***pp:15-28****. Springer, Berlin, Heidelberg.*

Mitsuru Matsui(1994). Linear Cryptanalysis Method for DES Cipher. Springer Berlin Heidelberg, ***pp: 386-397****.*

Burks, Arthur W and Von Neumann, John. (1966).The Theory of Self-Reproducing Automata University of Illinois Press.

National Institute of Standards and Technology., Available from: http://www.itu.int/rec/TREC-X.200-199407-I/en derniere Last visited: april 2017.

Shaoquan Jiang(2015). On the Size of Source Space in a Secure MAC. IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 10, NO. 9.

Shin, Sang-Ho, Dong-Hyun Kim, and Kee-Young Yoo. (2012). A lightweight multi-user authentication scheme based on cellular automata in cloud environment.IEEE 1st International Conference on Cloud Networking (CLOUDNET), ***pp****: 176-178.*

SMART, Nigel P (2016). Defining Security. Authentication: Security of Signatures and MACs. Cryptography Made Simple, Springer International Publishing, ***pp. 197-223****.*

Smart, Nigel P. (2016).Hash Functions, Message Authentication Codes and Key Derivation Functions Cryptography Made Simple. Springer International Publishing, ***pp****: 271-294.*

Taha, Mostafa, and Patrick Schaumont(2013). Differential power analysis of MAC-Keccak at any key-length. International Workshop on Security. Springer Berlin Heidelberg, ***68-82****.*

Wu, Teng, and Guang Gong(2016). Two new message authentication codes based on APN

functions and stream ciphers. Security and Communication Networks.

S. Wolfram. (2002).A new kind of science 2nd Edition, John Wiley & Sons, Wolfram Media, ***pp****: 437-440.*

S. Wolfram(1986). Cryptography with cellular automata, The institute for Advanced Study, Princeton NJ08540.

Wolfram, Stephen(1986). Tables of cellular automaton properties. Theory and Applications of Cellular Automata, ***pp.485-557****.*

Zhu, Bo (2015). Analysis and Design of Authentication and Encryption Algorithms for Secure Cloud Systems. ***pp:9-15****.*