Mathematics Interdisciplinary Research **3** (2018) 81 − 87

An Improved Hash Function Based on the Tillich-Zémor Hash Function

Ahmad Gaeini, Mohammad Hossein Ghaffari and Zohreh Mostaghim⋆

Abstract

Using the idea behind the Tillich-Zémor hash function, we propose a new hash function. Our hash function is parallelizable and its collision resistance is implied by the hardness assumption on a mathematical problem. Also, it is secure against the known attacks. It is the most secure variant of the Tillich-Zémor hash function until now.

Keywords: The Tillich-Zémor hash function, Cayley hash function, special linear group.

2010 Mathematics Subject Classification: Primary: 94A60, Secondary: 05C25, 20G40.

How to cite this article
A. Gaeini, M. H. Ghaffari and Z. Mostaghim, An improved hash function based on the Tillich-Zémor hash function, *Math. Interdisc. Res.* **3** (2018) 81-87.

1. Introduction

A hash function $H : \{0,1\}^* \longrightarrow \{0,1\}^n$ maps an input message m of arbitrary length to a fixed-length hash value $h = H(m)$ of size n. If such a function satisfies additional requirements it can be used for cryptographic applications, for example in digital signatures, ID based cryptography, and randomization of plaintexts in probabilistic cryptosystems. The primary properties that a hash function H should possess are as follows:

- Computation of $H(x)$ should be fast and easy, roughly linear time.

⋆Corresponding author (E-mail: mostaghim@iust.ac.ir)
Academic Editor: Mohammad Reza Darafsheh
Received 10 September 2017, Accepted 04 June 2018
DOI: 10.22052/mir.2018.97876.1078

- Preimage resistance: for a given hash value h, it should be computationally infeasible to find any message m, which results in the given hash value $H(m) = h$.
- Second preimage resistance: for a given message m, it should be computationally infeasible to find a second message m' with $m \neq m'$, which results in the same hash value $H(m) = H(m')$.
- Collision resistance: it should be computationally infeasible to find two messages m and m' with $m \neq m'$, which results in the same hash value $H(m) = H(m')$.

Recent developments in the cryptanalysis of hash functions have clearly shown that heuristic security arguments are not good enough. Something provable is needed. Hashing is a vital concept in cryptography and we need hash functions which are efficient and whose security can be trusted at the same time.

In this paper we propose a new hash function which is a modification of a variant of the Tillich-Zémor hash function, a provably secure hash function (in the sense that its security relates to the hardness of mathematical problems). In Theorem 3.3 we show that our hash function is at least as secure as ZesT hash function, a safer modification of the Tillich-Zémor hash function. For more information of security and other aspects of ZesT, we refer the reader to Chapter 9 of [3].

2. Preliminaries

Let n be a positive integer and let $P_n(X)$ be an irreducible polynomial of degree n over the field $\mathbb{F}_2$. Thus $F = \mathbb{F}_2[X]/(P_n(X)) \cong \mathbb{F}_{2^n}$. In this paper, we choose $P_n(X)$ from the following set:

$$\Big\{X^{127} + X + 1,\ X^{251} + X^7 + X^4 + X^2 + 1,\ X^{509} + X^8 + X^7 + X^3 + 1\Big\}.$$

They are irreducible polynomials that allow cheap modular reductions in a computer implementation (Section 4 of [5]).

Denote by $\mathrm{SL}_2(F)$ the group of 2×2 matrices of unitary determinant in the field F. It is known that $\{A_0, A_1\}$ is a generating set for $\mathrm{SL}_2(F)$ (Theorem 3.1 of [6]), where

$$A_0 = \begin{pmatrix} X & 1 \\ 1 & 0 \end{pmatrix}, \qquad A_1 = \begin{pmatrix} X & X+1 \\ 1 & 1 \end{pmatrix}.$$

The Tillich-Zémor hash function $H_{ZT} : \{0,1\}^* \longrightarrow \mathrm{SL}_2(F)$ defined by

$$H_{ZT}(m) = A_{m_1} A_{m_2} \dots A_{m_l},$$

where $m = m_1 m_2 \dots m_l \in \{0,1\}^*$.

Note that an element of F (respectively, $\mathrm{SL}_2(F)$) can be represented by a bit-string with length n ($4n$). The following proposition is a consequence of Theorem 3.4 of [6].

Proposition 2.1. *For any distinct messages m and m', if $H_{ZT}(m) = H_{ZT}(m')$, then the length of at least one of the messages is at least n.*

Denote by H_{ZT}^{vec} the first row of the Tillich-Zémor, i.e. $H_{ZT}^{vec}(m) = (a, b)$, if $H_{ZT}(m) = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$. Replacing matrix-by-matrix multiplication by a vector-by-matrix multiplication, H_{ZT}^{vec} can be computed about twice faster than H_{ZT}. Now define ZesT-n hash function $H_{ZesT} : \{0,1\}^* \longrightarrow F^2$ as follows

$$H_{ZesT}(m) = H_{ZT}^{vec}\big(m \,||\, (H_{ZT}^{vec}(m) \oplus c_{rnd})\big),$$

where c_{rnd} is a constant, whose bits "look like random"; for example, the binary representation of number π. Note that

$$H_{ZesT}(m) = H_{ZT}^{vec}(m) H_{ZT}(H_{ZT}^{vec}(m) \oplus c_{rnd}).$$

H_{ZesT} is introduced in [5] and Chapter 9 of [3]. It is a safer and faster version of H_{ZT}. Our new hash function is a modification of H_{ZesT}. Today's attacks on H_{ZesT} are based on the factorization algorithms in $\mathrm{SL}_2(F)$. The main idea behind our new hash function is to restrict the type of factorizations that can be used in an attack.

3. New Hash Function

Let t be an integer number greater than 1 and $C \in \mathrm{SL}_2(F) - \{I, A_0, A_1\}$; where I is the identity element of $\mathrm{SL}_2(F)$. Define $H_{ZgT} : \{0,1\}^* \longrightarrow \mathrm{SL}_2(F)$ by

$$H_{ZgT}(m) = \prod_{i=1}^{l} D_i,$$

where,

$$D_i = \begin{cases} A_{m_i} & \text{if} \quad t \nmid i \\ A_{m_i} C & \text{if} \quad t \mid i, \end{cases}$$

and $m = m_1 m_2 \dots m_l \in \{0,1\}^*$.

We define new hash function $H_{ZgesT} : \{0,1\}^* \longrightarrow F^2$ as

$$H_{ZgesT}(m) = H_{ZgT}^{vec}(m) H_{ZgT}\big(H_{ZgT}^{vec}(m) \oplus c_{rnd}\big),$$

where H_{ZgT}^{vec} is the first row of H_{ZgT}.

The parameter C can be defined as $H_{ZT}(c)$ for a long message c. In this case, we are easily able to use Proposition 3.2. The effect of parameter t on the run

time of H_{ZgesT} is analyzed in Section 3.2. At the end of this section we express why we should choose $t < n$.

In our C implementation of this function we used bitwise operations for implementing operations over field $\mathbb{F}_2$. More techniques for reaching better performance are described in the sections 3 and 4 of [5].

Example 3.1. Let m be the string "abc...yz"×5 of length 130. The hexadecimal representation of the $H_{ZesT}(m)$ and $H_{ZgesT}(m)$ for $n = 127$ and $t = 120$, respectively, are:

$$9cfd8651019330895f1696f55f306b088ab5b0a65177a7ec53c4c6b53b09e240,$$

$$1d704b1ee521710e4167f17fe2714b2c1babf2e97c66985d1a10cd499f0fee42.$$

3.1 Security Aspects

In this section, at first, we prove that H_{ZgesT} is at least as secure as H_{ZesT}. Then, we show that H_{ZgesT} is resistant against all the known attacks that work on H_{ZesT}.

Proposition 3.2. *If we know a message c such that $H_{ZT}(c) = C$, then for every message m we can efficiently find a message $\hat{m}$ such that $H_{ZgT}(m) = H_{ZT}(\hat{m})$ and $H_{ZgesT}(m) = H_{ZesT}(\hat{m})$.*

Proof. Let c be a bitstring such that $H_{ZT}(c) = C$ and $m = m_1 m_2 \dots m_l$. Set $\hat{m} = m_1 \dots m_t\, c\, m_{t+1} \dots m_{2t}\, c\, m_{2t+1} \dots\, m_l$. It is easily seen that $H_{ZgT}(m) = H_{ZT}(\hat{m})$. Thus, $H_{ZgesT}(m) = H_{ZesT}(\hat{m})$. □

Since finding above c is possible by the techniques like in the sections 5 and 6 of [4], the following theorem is an immediate consequence of the above proposition.

Theorem 3.3. *Breaking the preimage and collision resistance of H_{ZgesT} respectively leads to breaking the preimage and collision resistance of H_{ZT} (and H_{ZesT}).*

Proof. We just prove the theorem for H_{ZT}. The similar argument can be applied for H_{ZesT}. For preimage resistance part, let $h \in \mathrm{SL}_2(F)$ with $H_{ZgesT}(w) = h$. By Proposition 3.2, there exists $\hat{w}$ such that $H_{ZT}(\hat{w}) = H_{ZgesT}(w) = h$. For collision resistance part, let $H_{ZgesT}(m) = H_{ZgesT}(m')$ for $m \neq m'$. By Proposition 3.2, there exist $\hat{m}$ and $\hat{m'}$ such that $H_{ZgesT}(m) = H_{ZT}(\hat{m})$ and $H_{ZgesT}(m') = H_{ZT}(\hat{m'})$. So $H_{ZT}(\hat{m}) = H_{ZgesT}(m) = H_{ZgesT}(m') = H_{ZT}(\hat{m'})$. □

Now, we claim that H_{ZgesT} is more secure than H_{ZesT}. The best known attacks on H_{ZT} produce preimages of length $O(n^2)$ in time $O(n^4)$, preimages of length $O(n^3)$ in time $O(n^3)$ (Section 7 of [4]) and collisions of length $O(n)$ in time $O(2^{n/2})$ (Section 3 of [2], see also Section 4 of [1]). In fact, it is trivial that any factorization of a given element of $\mathrm{SL}_2(F)$, leads to a preimage for H_{ZT} (and H_{ZesT}); but looking at the definition of H_{ZgT} we found out that a factorization

with an arbitrary form does not lead to a preimage for H_{ZgT} (and H_{ZgesT}). They should have a special form of factorization:

$$A_{m_1}A_{m_2}\dots A_{m_t}CA_{m_{t+1}}A_{m_{t+2}}\dots A_{m_{2t}}CA_{m_{2t+1}}\dots A_{m_l},$$

but the known factorization algorithms do not have control on the form of the factorization.

Finally, by Proposition 2.1, if we choose $t < n$, then no collision for H_{ZT} can directly used to find a collision for H_{ZgT}. So, no known attack exists that can break H_{ZgesT}.

3.2 Performance

For input bitstring with length l, computing H_{ZgesT} needs $\lfloor l/t \rfloor + \lfloor n/t \rfloor$ matrix multiplications more than H_{ZesT}. Also, in an implementation of H_{ZgesT}, we need a counter to find out when we should multiply by C. The cost of these additional computations is at most 6%, for long inputs, $t = 8\lfloor n/8 \rfloor$ and $B = \begin{pmatrix} x^3+1 & x^2 \\ x & 1 \end{pmatrix}$.

At comparable collision resistances, ZgesT-127, ZgesT-251 and ZgesT-509 are respectively about 12, 8 and 22 times less efficient than SHA-1, SHA-256 and SHA-512. At comparable preimage resistances (which is $2n$ bits for ZgesT), ZgesT-127 and ZgesT-251 are respectively about 6 and 12 times less efficient than SHA-256 and SHA-512.

The run time of some hash functions are presented in Tables 1–3. In Table 1 the parameter t is $8\lfloor n/8 \rfloor$. All tests were performed for a same 500MB random content file on an AMD A6-3500 APU running at 2100 MHz, with 4G RAM.

Table 1: Run time (seconds) of H_{ZgesT} hash functions ($t = 8\lfloor n/8 \rfloor$).

Hash	H_{ZgesT}-127	H_{ZgesT}-251	H_{ZgesT}-509
Time	20.0	29.0	53.9

Table 2: Run time (seconds) of H_{ZesT} hash functions.

Hash	H_{ZesT}-127	H_{ZesT}-251	H_{ZesT}-509
Time	19.3	28.2	50.7

Table 3: Run time (seconds) of SHA hash functions.

Hash	SHA-1	SHA-256	SHA-512
Time	1.6	3.6	2.4

4. Conclusion

In this paper, we proposed a new modified version of the Tillich-Zémor hash function, ZgesT, which is safer, parallelizable and practical. We proved that ZgesT is at least as secure as the Tillich-Zémor and ZesT hash functions. In addition, by

making the background mathematical problem (the factorization problem) harder, it seems today's approaches to attack the Tillich-Zémor based hash functions can not threat ZgesT. As disadvantage, our hash function like other known provable secure hash functions is slower than today used hash functions like SHA hash functions family.

Conflicts of Interest. The authors declare that there is no conflicts of interest regarding the publication of this article.

References

[1] M. Grassl, I. Ilić, S. Magliveras, R. Steinwandt, Cryptanalysis of the Tillich-Zémor hash function, *J. Cryptology* **24**(1) (2011) 148–156.

[2] C. Mullan, B. Tsaban, SL_2 homomorphic hash functions: worst case to average case reduction and short collision search, *Des. Codes Cryptogr.* **81**(1) (2016) 83–107.

[3] C. Petit, *On graph-Based Cryptographic Hash Functions*, PhD thesis, Universit Catholique de Louvain, 2009.

[4] C. Petit, J. -J. Quisquater, Preimages for the Tillich-Zémor Hash Function, In: A. Biryukov, G. Gong, D. R. Stinson (Eds) *Selected Areas in Cryptography*, SAC 2010. Lecture Notes in Computer Science, vol 6544. Springer, Berlin, Heidelberg, 2011.

[5] C. Petit, N. Veyrat-Charvillon, J. -J. Quisquater, Efficiency and pseudorandomness of a variant of Zémor-Tillich hash function, *IEEE International Conference on Electronics, Circuits, and Systems*, 906–909, ICECS 2008, 31 Aug. - 3 Sept. 2008.

[6] J. -P. Tillich, G. Zémor, Hashing with SL_2, In: Y. G. Desmedt (Ed.) *Advances in Cryptology âĂŤ CRYPTO âĂŹ94. CRYPTO 1994*, Lecture Notes in Computer Science, vol 839. Springer, Berlin, Heidelberg, 40–49, 1994.

Ahmad Gaeini
Department of Mathematics, Faculty of Science,
Imam Hossein Comprehensive University,
Tehran, I. R. Iran
E-mail: againi@ihu.ac.ir

Mohammad Hossein Ghaffari
Department of Mathematics, Faculty of Science,
Imam Hossein Comprehensive University,
Tehran, I. R. Iran
E-mail: mhghaffari@alumni.iust.ac.ir

Zohreh Mostaghim
Cryptography and Data Security Laboratory, School of Mathematics,
Iran University of Science and Technology,
Tehran, I. R. Iran
E-mail: mostaghim@iust.ac.ir