Hashing with SL_2

Jean-Pierre Tillich and Gilles Zémor

Ecole Nationale Supérieure des Télécommunications
Network Department
46 rue Barrault, 75634 Paris Cedex 13, France
tillich@inf.enst.fr, zemor@res.enst.fr

Abstract. We propose a new family of hash functions based on computations over a finite field of characteristic 2. These functions can be computed quickly, detect small modifications of the input text, and their security is equivalent to a precise mathematical problem. They rely on the arithmetic of the group of matrices SL_2, and improve upon previous functions based on the same strategy.

1 Introduction

We focus on the problem of designing easily computable cryptographic hash functions. Such a function H should map the set of variable length texts over an alphabet $\mathcal{A}$, to a set of (short) fixed length texts that are named *hashcodes.*

$$H : \mathcal{A}^* \longrightarrow \mathcal{A}^n$$

A hash function should have the following properties :

- It should be easily (i.e. quickly) computable.
- It should be computationally difficult to find "collisions", i.e. two texts having the same hashcode. (This is sometimes known as the strong collision criterion).

Hash functions are widely used in numerous cryptographic protocols, and a lot of work has already been put into devising adequate hashing schemes. Despite that, numerous propositions have been shown to be insecure, and the security of those which remain unbroken remains formally unproved.

Following ideas introduced in [Zém] (see also [Zém91]), we elaborate here on a design principle which enables one to obtain the following unconditional security property. *Small modifications of the input text are always detected.* More precisely we will present here a new hash algorithm which meets this property, and which displays several other attractive features. In particular, it improves upon previous proposals of this kind [Zém] [TZ94].

- The algorithm can be easily implemented in software by using only basic operations, namely addition in a finite field of characteristic 2, with 2^n elements $\mathbf{F}_{2^n}$ (with say n in the range 130-170), which allows fast computations.
- It is particularly well suited to parallelisation, and to precomputations.
- The security of the scheme we propose is equivalent to a precise mathematical problem, for which there exist several results in favor of its difficulty.

The hash algorithm we propose can be described as follows.

Defining parameter. An irreducible polynomial $P_n(X)$ of degree n in the aforementioned range.
Algorithm. Let A and B be the following matrices.

$$A = \begin{pmatrix} X & 1 \\ 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} X & X+1 \\ 1 & 1 \end{pmatrix}$$

Define the mapping

$$\begin{aligned} \pi : \{0,1\} &\to \{A, B\} \\ 0 &\mapsto A \\ 1 &\mapsto B \end{aligned}$$

The hashcode of binary message $x_1 x_2 \dots x_k$ is just the matrix product

$$\pi(x_1)\pi(x_2)\dots\pi(x_k)$$

where computations are made in the quotient field $\mathbf{F}_{2^n} = \mathbf{F}_2[X]/P_n(X)$ of 2^n elements. The hashcode is thus some element of the group $SL_2(\mathbf{F}_{2^n})$ of 2×2 matrices with determinant 1 over $\mathbf{F}_{2^n}$. We need $3n+1$ bits to encode the hashed value (that is 390-510 bits).

A design strategy lies behind the construction of this function. It is based on associating to such a function a Cayley graph, and by exploiting the fact that security is related to graphical parameters. We will elaborate on this in the next section. Provable properties of the hash algorithm using this strategy will be given in section 3.

2 A design strategy. Graph-theoretic issues

2.1 The general construction

We have based the design of our hash functions on the following general scheme.

Defining parameter. A finite group G, and a set of generators $\mathcal{S}$ of the same size as the text alphabet $\mathcal{A}$. Choose a function $\pi : \mathcal{A} \to \mathcal{S}$ which defines a one-to-one correspondence between $\mathcal{A}$ and $\mathcal{S}$.
Algorithm.

The hashcode of the text $x_1 x_2 \ldots x_k$ is just the group element

$$\pi(x_1)\pi(x_2)\ldots\pi(x_k)$$

Motivation for this construction is twofold : the hash functions display a concatenation property, and one can associate to such a scheme a Cayley Graph, several parameters of which are relevant to security (namely its girth and expanding properties).

Concatenation property. If x and y are two texts, then their concatenation xy has hashed value $H(xy) = H(x)H(y)$. This clearly allows an easy parallelisation of the scheme, and precomputations when parts of the message are known in advance.

Parameters of the associated Cayley graph. We can associate to this scheme the Cayley graph $\mathcal{C}(G, \mathcal{S})$: its vertex set is G and there is a directed edge from g_1 to g_2 if and only if $g_1^{-1}g_2 \in \mathcal{S}$. The following parameters are of fundamental importance when studying the security of the hash function.

The girth of the graph. We shall use the following definition of the *directed girth.*

Definition 2.1 — *Call the* directed girth *of a graph $\mathcal{G}$, the largest integer ∂ such that given any two vertices v and w, any pair of distinct directed paths joining v to w will be such that one of those paths has length (i.e. number of edges) ∂ or more.*

It is readily seen that for the associated Cayley graph $\mathcal{C}(G, \mathcal{S})$, this notion is translated into the following property of the hash function.

Proposition 2.2 — *If we replace k consecutive symbols of a text*

$$x = x_1 x_2 \ldots x_i \boxed{x_{i+1} \ldots x_{i+k}} x_{i+k+1} \ldots x_t$$

by a string of h consecutive symbols so that the resulting text

$$x' = x_1 x_2 \ldots x_i \boxed{y_{i+1} \ldots y_{i+h}} x_{i+k+1} \ldots x_t$$

has the same hashed value, then $\sup(k, h) \geq \partial$.

In other words, if we can obtain $\mathcal{C}(G, \mathcal{S})$ with a large ∂, we protect against local modifications of the text.

Expanding properties. A desirable feature of any hash function is the equidistribution of the hashed values. This property can be guaranteed if the associated Cayley graph $\mathcal{C}(G, \mathcal{S})$ satisfies

Proposition 2.3 — *If $\mathcal{C}(G, \mathcal{S})$ is a Cayley graph such that the gcd of its cycle lengths equals 1, then for the corresponding hash function, the distribution of hashed values of texts of length n tends to equidistribution when n tends to infinity.*

This is proved by classical graph-theoretic (or Markov Chain) methods by studying the successive powers A^n of the adjacency matrix of the graph. To prove that this property occurs in practice, we need to evaluate the speed with which equidistribution is achieved. The best results will be achieved if the Cayley graph $\mathcal{C}(G, \mathcal{S})$ sufficiently resembles a random graph. This can be obtained for graphs with a high *magnifying* or *expansion* coefficient (for more details see [Zém]).

2.2 The choice of SL_2.

The groups $SL_2(\mathbf{F}_q)$ of 2×2 matrices of determinant one over a finite field $\mathbf{F}_q$ seem to us to be a promising choice for devising quality hash functions. There are several reasons for this. By choosing simple matrices for generators, one obtains fast hash functions: this is because multiplication by such a matrix, i.e. processing one bit of text, amounts to a few additions in $\mathbf{F}_q$: in this paper, we focus on the case when $q = 2^n$ which provides the fastest computations of all. It is comparatively easy to obtain Cayley graphs over those groups that have large girths, [Mar82]: [Mar88] is a record-breaking construction. Cayley graphs over $SL_2(\mathbf{F}_q)$ tend to display good expanding properties: this is justified both theoretically [Sar90] and experimentally [JR93].

2.3 On the difficulty of finding collisions.

Another attractive feature of the general hashing scheme is that we can express clearly in mathematicals terms the problem of finding collisions. It may readily be checked that the problem reduces to finding two strings of generators (elements of $\mathcal{S}$) such that the corresponding products coincide in G ; i.e. find $s_1, s_2, \ldots s_n$, $\sigma_1, \sigma_2, \ldots \sigma_m \in \mathcal{S}$ such that

$$s_1 s_2 \ldots s_n = \sigma_1 \sigma_2 \ldots \sigma_m$$

equivalently

$$s_1 s_2 \ldots s_n \sigma_m^{-1} \sigma_{m-1}^{-1} \ldots \sigma_1^{-1} = 1. \tag{1}$$

So we see that finding a collision is equivalent to finding factorisations of the form (1). Now it can be argued that there are always trivial factorisations of the form (1) in any finite group (e.g. $s^{|G|} = 1$, for any $s \in \mathcal{S}$). But we must note that only $n \approx \log |G|$ bits are needed to express hashed values, so that we can choose groups of large cardinality (e.g. $|G| = 2^{500}$) for which trivial factorisations involving $N \sim |G|$ elements are useless as actual forgeries (because no text has 2^{500} bits !). This means, broadly speaking, that the strong collision criterion is satisfied whenever it is computationally difficult to find short factorisations of the form (1). This problem is known to be potentially difficult. For instance Jerrum [Jer85] considers the problem of finding the shortest factorisation of an arbitrary element of an arbitrary permutation group over some set of generators, and proves that it is Pspace-complete. This provides additional motivation for pursuing this group-theoretic strategy.

Moreover when we choose $G = SL_2(\mathbf{F}_q)$ as explained above, it seems that the search for short factorisations in G yields challenging and apparently difficult problems, see e.g. [BKL89]. Some work has already been devoted to solve this problem, and the main results are the following probabilistic algorithms.

1. *Subgroup attacks.* A hashing scheme based on matrix computations has been devised in the past [Bos77] with a group of too small a size for which a probabilistic attack was devised in [Cam87]. It is based on the search for texts whose hashcode falls into a subgroup. For our choice of q this algorithm turns out to be inefficient, because all subgroups of $SL_2(\mathbf{F}_q)$ have index larger than $q + 1$ (theorem 3.2).
2. *Density attacks.* A preliminary version of a group theoretic hash function was shown to be insecure in [TZ94]. It was based on the group $SL_2(\mathbf{Z}_p)$ and the two generators $C = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ and $D = \left(\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right)$. The key to the forgery was the following: it is readily seen that a short factorisation over $SL_2(\mathbf{Z}_p)$ of the identity produces collisions (by insertion into any message M). To find such a factorisation, the strategy is to reduce the problem to factorising in an infinite group, in this case $SL_2(\mathbf{Z})$. Look for a matrix U of $SL_2(\mathbf{Z})$ which reduces modulo p to the identity, and can be expressed as a product of C's and D's. In this case this simply means that U should have non-negative coefficients. There is an effective algorithm to obtain the factorisation of U. To have an effective forgery, one must have a way of finding such a matrix U whose factorisation into C's and D's is short. A probabilistic algorithm that does this is given in [TZ94]: it is based on the fact that the set of matrices of $SL_2(\mathbf{Z})$ with non-negative coefficients is a "dense" subset of $SL_2(\mathbf{Z})$. To protect against such attacks, one should choose sets of generators $\mathcal{S}$ that generate sufficiently sparse submonoids of the infinite groups associated to $SL_2(\mathbf{F}_q)$.

3 A specific hash function

In this section we turn to the specific hash function associated to $G = SL_2(\mathbf{F}_q)$ with $q = 2^n$ and with the generators A and B mentioned in the introduction. We address the following topics related to its security, i.e.

- What is the set of hashcodes?
- Why does this hash function detect local modifications of the input.
- Why does this hash function protect against density attacks.

3.1 The set of hashcodes

In order to show that the set of all hashcodes is not too small we need to clarify what is the subset (actually the subgroup) of $SL_2(\mathbf{F}_{2^n})$ which is generated by A and B. More precisely we show here that this subset is the whole group: this is the following theorem

Theorem 3.1 —
$$< A, B >= SL_2(\mathbf{F}_{2^n})$$
(where $< A, B >$ denotes the group generated by A and B)

From that it follows that the set of hashcodes has $2^n(2^{2n} - 1)$ elements (since $|SL_2(\mathbf{F}_{2^n})| = 2^n(2^{2n} - 1)$), which is sufficiently large to avoid direct probabilistic attacks, which use the birthday paradox for instance. Moreover, it is straightforward to check that we can code efficiently these hashcodes by using just $3n + 1$ bits.

Theorem 3.1 is proven by using some known results about the subgroups of $SL_2(\mathbf{F}_{2^n})$. For our purposes, we use the classification obtained by Dickson (see [Dic58]), and which can also be found in [Suz82] (see theorem 6.25 p.412-413)

Theorem 3.2 — *All possible proper subgroups of $SL_2(\mathbf{F}_{2^n})$ are the following:*

(a) Abelian subgroups.
(b) Dihedral subgroups of order $2d$, where d divides $2^n + 1$ or $2^n - 1$.
(c) The alternating groups A_4, or A_5, or the symmetric group S_4.
(d) The upper triangular subgroup, its subgroups, and their conjugates.
(e) $SL_2(\mathbf{F}_{2^m})$, where m is a divisor of n, (and its conjugates).

It is straightforward to check that none of these cases can occur.
Case (a) is impossible since $A.B \neq B.A$.
Case (b) cannot occur, because neither A, nor B is of order 2.
Case (c) is checked with a little computation: for $n > 2$, A and B generate a subgroup whose order is larger than those of the groups of (c).

Case (d) uses a different approach. Let us note that all the matrices of the upper triangular group (or a conjugate of this group) have a common eigenvector. Since A and $S = A^{-1}B = \left(\begin{smallmatrix} 1 & 1 \\ 0 & 1 \end{smallmatrix}\right)$ have no common eigenvector, A and B can not generate only the upper triangular group (or its conjugates) or its subgroups.
Case (e) is settled by noting that X cannot belong to a subfield $\mathbf{F}_{2^m}$ of $\mathbf{F}_{2^n}$, hence A does not belong to $SL_2(\mathbf{F}_{2^m})$. A and B can not generate a conjugate $U^{-1}SL_2(\mathbf{F}_{2^m})U$ of this group ($U = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in SL_2(\mathbf{F}_{2^n})$) since this would imply

$$U.A^{-1}.B.U^{-1} \in SL_2(\mathbf{F}_{2^m}) \tag{2}$$

$$U.A^{-2}.B.A.U^{-1} \in SL_2(\mathbf{F}_{2^m}) \tag{3}$$

(2) implies that a^2 and d^2 belong to $\mathbf{F}_{2^m}$, hence a and d belong to $\mathbf{F}_{2^m}$. (3) implies that c^2 and b^2 belong to $\mathbf{F}_{2^m}$, hence c and b belong to $\mathbf{F}_{2^m}$. This in turn implies that U belongs to $SL_2(\mathbf{F}_{2^m})$, which would imply that A belongs to this same subgroup, which is impossible.

Moreover, by using the above classification of subgroups and similar arguments, we can show (details will be archived in a technical report) that the Cayley graph corresponding to our hash function satisfies the condition of proposition 2.3, so we obtain an even stronger property on the hashcodes, that is :

Proposition 3.3 — *The distribution of hashcodes of length l tends to equidistribution when l tends to infinity.*

3.2 Protection against local modifications

We have seen in section 2 that if the Cayley graph associated to our scheme has a girth larger than ∂ then any modification of a text obtained by replacing k consecutive bits by h other bits will necessarily change the hashcode if k and h are less than ∂.

The relevance of our proposition lies in the fact that the girth for the associated graph of the scheme we propose is large :

Theorem 3.4 — *The girth of the Cayley graph $\mathcal{G}(SL_2(\mathbf{F}_{2^n}), A, B)$ is larger than n.*

In other words, for n chosen as explained in the introduction we are *sure* to detect any modification of up to $n = 130 - 170$ consecutive bits of the input text.
Proof.
In the Cayley graph setting, the girth is just the minimum value l for which there exist two different strings of A's and B's $s_1, s_2, \ldots s_l$, and $\sigma_1, \sigma_2, \ldots \sigma_m$ (with $m \leq l$) such that

$$s_1 s_2 \ldots s_l = \sigma_1 \sigma_2 \ldots \sigma_m$$

If we consider now A and B as elements of $SL_2(\mathbf{F}_2[X])$, then the two products $s_1 s_2 \ldots s_l$ and $\sigma_1\sigma_2\ldots\sigma_m$ differ in this infinite group (see lemma 3.5). Since $\mathbf{F}_{2^n}$ is defined as the quotient $\mathbf{F}_2[X]/P_n(X)$, where $P_n(X)$ is some irreducible polynomial over $\mathbf{F}_2$ of degree n, it follows that the only way these products will coincide over $SL_2(\mathbf{F}_{2^n})$, is that one of the matrices $s_1 s_2 \ldots s_l$ or $\sigma_1\sigma_2\ldots\sigma_m$ (computed over $SL_2(\mathbf{F}_2[X])$) has one of its entries equal to a polynomial of degree $\geq n$. It is easy to check that this can happen only if $l \geq n$ or $m \geq n$. □

Lemma 3.5 — *For two different strings of A's and B's , $s_1, s_2, \ldots s_l$, and $\sigma_1, \sigma_2, \ldots \sigma_m$ the products (computed over $SL_2(\mathbf{F}_2[X])$) $s_1 s_2 \ldots s_l$ and $\sigma_1\sigma_2\ldots\sigma_m$ are different.*

Proof.
It is straightforward to show by induction that a product $\sigma_1\sigma_2\ldots\sigma_m$ takes the form $\left(\begin{smallmatrix} P_m(X) & Q_{m-1}(X) \\ P_{m-1}(X) & Q_{m-2}(X)\end{smallmatrix}\right)$ if $\sigma_m = A$ (The P's and Q's denote polynomials whose degree is indicated by the subscript), or $\left(\begin{smallmatrix} P_m(X) & Q_m(X) \\ P_{m-1}(X) & Q_{m-1}(X)\end{smallmatrix}\right)$ if $\sigma_m = B$. Hence two products $s_1 s_2 \ldots s_l$ and $\sigma_1\sigma_2\ldots\sigma_m$ can be equal over $SL_2(\mathbf{F}_2[X])$ only if $m = l$, and $\sigma_m = s_l$. By simplifying on the right by $\sigma_m = s_l$, and by iterating this argument we obtain the lemma. □

3.3 Protection against density attacks

A density attack in the case of this particular hash function takes the following form.

1. find a matrix U of $SL_2(\mathbf{F}_2[X])$ which is equal to the identity modulo $P_n(X)$ (where $P_n(X)$ is the irreducible polynomial of degree n which defines the field $\mathbf{F}_{2^n}$ as $\mathbf{F}_2[X]/P_n(X)$).
2. express this matrix U as a product of A and B in $SL_2(\mathbf{F}_2[X])$ (if possible).
3. this factorisation becomes a factorisation of the identity when computed over $SL_2(\mathbf{F}_{2^n})$.
4. one can deduce from it a message whose hashcode is the identity and which may be inserted in every message M without changing the hashed value of M.

Point 1 is rather easy to solve (in the same way as over $SL_2(\mathbf{Z})$), nevertheless the key of our proposition is to make point 2 infeasible. What makes this last problem so difficult is that there are very few matrices of $SL_2(\mathbf{F}_2[X])$, which can be expressed as a product of A and B. More precisely

Theorem 3.6 — *Define the set*

$$E_m = \{U = \left(\begin{smallmatrix} P(X) & Q(X) \\ R(X) & S(X)\end{smallmatrix}\right) \in SL_2(\mathbf{F}_2[X])/deg P, Q, R, S \leq m\}.$$

We have $$\frac{|\{U \in E_m \ , \ U \textit{ is a product of } A \textit{ and } B\}|}{|E_m|} = O(\frac{1}{2^m})$$

Sketch of the proof:
This follows from

$$|\{U \in E_m \ , \ U \text{ is a product of } A \text{ and } B\}| = 2^{m+1} - 2$$

(because of the form of products of A and B, see the proof of lemma 3.5) and

$$|E_m| = O(2^{2m})$$

(this comes from a counting argument giving the number of pairs of coprime polynomials of bounded degree). □

The above theorem shows that even if we can find a matrix U satisfying point 1 by random search methods, then since it will have an entry which is a polynomial of degree at least n (see proof of theorem 3.4), U will have a very small probability of being a product of A and B (that is $O(\frac{1}{2^n})$).

4 Concluding Remarks

We have defined a new family of hash functions based on computations in $SL_2(\mathbf{F}_{2^n})$. These functions improve upon previous schemes [Zém], which were defined over the group $SL_2(\mathbf{Z}_p)$. First, computing time is substantially speeded up. It requires at most a few shifts and XOR's of 150-bit quantities per message bit. Because of the concatenation property, precomputations and fast multiplications in fields of characteristic 2 can also be used. Second, we can prove an explicit security property which speaks against the density attacks that have shown the first attempt of [Zém91] to be insecure.

We have chosen the size parameter q so that $\sqrt{q}$ stays too large with respect to computing power. This is to protect against the subgroup approach used in conjunction with birthday attacks.

References

[BKL89] L. Babai, W. M. Kantor, and A. Lubotsky. Small-diameter Cayley graphs for finite simple groups. *Europ. J. of Combinatorics*, 10:507–522, 1989.

[Bos77] J. Bosset. Contre les risques d'altération, un système de certifications des informations. *01 Informatique*, (107), Février 1977.

[BWD93] T. Beth, D. Weber, and F. Damm. Combining data compression with cryptographic hash functions. In *CRYPTO 93*. LNCS Springer-Verlag, 1993.

[Cam87] P. Camion. Can a fast signature scheme without secret key be secure ? In *proc. AAECC*, pages 187–196. Springer-Verlag Lec. N. Comp. Sci. 228, 1987.

[Dic58] L. Dickson. *Linear groups with an exposition of the Galois field theory.* Dover, New York, 1958.

[Jer85] M. R. Jerrum. The complexity of finding minimum length generator sequences. *Theoretical Computer Science*, 36:265–289, April 1985.

[JR93] J.Lafferty and D. Rockmore. Numerical investigation of the spectrum for certain families of cayley graphs. In DIMACS Series in Disc. Math., Theoretic Comput. Sci. in Discrete Mathematics, and Theoretical Computer Science, editors, *1992 DIMACS Workshop on expanders graphs*, pages 63–74, 1993.

[Mar82] G. A. Margulis. Explicit constructions of graphs without short cycles and low density codes. *COMBINATORICA*, 2(1):71–78, 1982.

[Mar88] G. A. Margulis. Explicit group-theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators. *Problemy Peredachi Informatsii*, 24(1):51–60, 1988.

[Sar90] P. Sarnack. *Some applications of modular forms.* Cambridge University Press, 1990.

[Suz82] M. Suzuki. *Group theory, volume I.* Springer-Verlag, 1982.

[TZ94] J-P Tillich and G. Zémor. Group-theoretic hash functions. In *First French-Israeli workshop on algebraic coding.* Springer-Verlag Lec. N. Comp. Sci. 781 p.90-110, to appear in april 1994.

[Zém] G. Zémor. Hash functions and Cayley graphs. To appear in Designs, Codes and Cryptography.

[Zém91] G. Zémor. Hash functions and graphs with large girths. In *EUROCRYPT 91.* LNCS 547 Springer-Verlag, 1991.

This article was processed using the LaTeX macro package with LLNCS style