

Cryptographic hash functions from expander graphs

Denis X. Charles
Microsoft Research
Redmond, WA 98052
USA

Eyal Z. Goren
McGill University
Montréal
Canada H3A 2K6

Kristin E. Lauter
Microsoft Research
Redmond, WA 98052
USA

Abstract

We propose constructing provable collision resistant hash functions from expander graphs in which finding cycles is hard. As examples, we investigate two specific families of optimal expander graphs for provable collision resistant hash function constructions: the families of Ramanujan graphs constructed by Lubotzky-Phillips-Sarnak and Pizer respectively. When the hash function is constructed from one of Pizer’s Ramanujan graphs, (the set of supersingular elliptic curves over \mathbb{F}_{p^2} with ℓ -isogenies, ℓ a prime different from p), then collision resistance follows from hardness of computing isogenies between supersingular elliptic curves. For the LPS graphs, the underlying hard problem is a representation problem in group theory. Constructing our hash functions from optimal expander graphs implies that the outputs closely approximate the uniform distribution. This property is useful for arguing that the output is indistinguishable from random sequences of bits. We estimate the cost per bit to compute these hash functions, and we implement our hash function for several members of the Pizer and LPS graph families and give actual timings.

1 Introduction

With the untimely demise of SHA-1, NIST is soliciting proposals for new cryptographic hash functions to standardize. The goal is to construct an efficiently computable hash function which is collision resistant. A hash function is called a *provable collision resistant hash function* if to compute a collision is to solve some hard mathematical problem such as factoring or discrete log, for example as in the scheme proposed in [8]. We propose constructing provable collision resistant hash functions from expander graphs. Expander graphs are graphs in which the neighbor set of any “not too large” subset of vertices contains many new vertices. This property of expander graphs leads to other interesting properties, such as the fact that random walks quickly approximate the uniform distribution. In our construction the input to the hash function is used as directions for walking around a graph, and the ending vertex is the output of the hash function. Our construction can be applied to any expander graph, and can be thought of as a proposal to construct collision-resistant hash functions from any expander graph in which finding cycles is a hard problem. We give here two families of optimal expander graphs, and investigate the efficiency and collision resistance properties of these two families. The two families are the Ramanujan graphs constructed by Pizer and Lubotzky-Phillips-Sarnak (LPS) respectively. Ramanujan graphs are optimal expander graphs, in a technical sense (see section 2), and thus have excellent mixing properties. The rapid mixing property implies in particular that the output of our hash functions closely approximates the uniform distribution, and so to that extent is indistinguishable from random sequences of bits. For these two families of expander graphs, the collision resistance of the hash functions follows from arithmetic properties of the graphs’ constructions.

When constructing a hash function from the Ramanujan graph of supersingular elliptic curves over \mathbb{F}_{p^2} with ℓ -isogenies, ℓ a prime different from p , as in Pizer ([18]), finding collisions is at least as hard as computing isogenies between supersingular elliptic curves. This is believed to be a hard problem (see Section 5 below), and the best algorithm currently known solves the problem in $O(\sqrt{p} \log^2 p)$ time. Thus we propose to set p to be a 256-bit prime, to get 128 bits of security from the resulting hash function. To compute the hash function from Pizer’s graph when $\ell = 2$ requires roughly $2 \log_2(p)$ field multiplications

per bit of input to the hash function. This is roughly the same efficiency as a provable hash based on the ECDLP, and relatively inefficient compared to the provable hash function VSH [8]. Our construction has the advantage that the output of our hash function is $\lceil \log_2(p) \rceil$ bits, and efficiency may be improved with optimizations. We note that the improvements to VSH-DL given in [15] also achieve a compressed output by sacrificing some performance.

Hash functions from LPS graphs are more efficient to compute than those from Pizer’s graphs. Applying our construction gives a hash function which is an improvement on those proposed by Zémor and Tillich [25], [27], and subsequent extensions. Finding collisions reduces to a group theoretic problem which is also believed to be difficult (see Section 6). To compute the hash function requires only 7 field multiplications per bit of input, but the field size may need to be bigger (1024 bit prime p instead of 256 bits, for example), and the output is $3 \log_2(p)$ bits. We have implemented this hash function for primes of varying size and we give unoptimized timings in Section 6.

These hash functions may be too inefficient to be applied in all situations, but would be appropriate for some protocols where a secure hash function is needed and other operations are on the same order of magnitude. This is the case, for example, for public key cryptographic protocols such as DSA or ECDSA or for key derivation functions (KDFs) used for authenticated key exchange as in NIST’s SP 800-56A. Our proposed hash functions are appropriate for use in any protocol requiring collision resistance, pre-image resistance, and indistinguishability from a random string (assuming the input is at least $\log_2(p)$ bits). An important property of our hash functions is that the hard mathematical problem underlying the collision resistance appears to be independent from other known hard problems such as factoring and ECDLP (elliptic curve discrete logarithm problem). For the Pizer graph, the hard mathematical problem is finding an isogeny between two given supersingular elliptic curves, and we explain in Section 5 how this problem is related to the problem of finding lattice vectors of a given norm. For the LPS graphs, the underlying hard problem is a representation problem in group theory.

2 Background and Definitions

2.1 Hash functions

A hash function maps bit strings of some finite length to bit strings of some fixed finite length, and must be easy to compute. We are concerned in this paper with unkeyed hash functions which are collision resistant. Unkeyed hash functions do not require a secret key to compute the output. We will define families of hash functions, and use an index to specify a member of the family.

2.2 Elliptic curves

Let p be a prime greater than 3 and q a power of p . An *elliptic curve* E over the field \mathbb{F}_q of q elements can be given by a *Weierstrass equation*

$$E: y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}_q,$$

where the polynomial $x^3 + ax + b$ has no repeated roots. One adds a “point at infinity” 0_E , which, when the curve is given in projective space as $y^2z = x^3 + axz^2 + bz^3$, is the point $(0: 1: 0)$. There is a group structure on an elliptic curve, given by rational functions, such that for every finite extension \mathbb{F}_{q^r} the \mathbb{F}_{q^r} -rational points of E , $E(\mathbb{F}_{q^r}) := \{(x, y) : y^2 = x^3 + ax + b, x, y \in \mathbb{F}_{q^r}\} \cup \{0_E\}$, form an abelian group for which 0_E is the identity. Given two elliptic curves E_1, E_2 over \mathbb{F}_q , a *homomorphism* $f: E_1 \rightarrow E_2$ is a morphism of algebraic curves that sends 0_{E_1} to 0_{E_2} . A non-zero homomorphism is called an *isogeny*. An isogeny is automatically surjective on points over the algebraic closure and has a finite kernel. For separable isogenies, the cardinality of this kernel is called the *degree of the isogeny*. For example, for any positive integer n , the multiplication-by- n map $[n]: E \rightarrow E$ is an isogeny of degree n^2 . If p does not divide n , then $\ker[n] \cong (\mathbb{Z}/n\mathbb{Z}) \times (\mathbb{Z}/n\mathbb{Z})$. In particular, if $\ell \neq p$ is a prime, there are precisely $\ell + 1$ subgroups of $E[\ell]$ of order ℓ . Another example of an isogeny is the Frobenius morphism. Let E/\mathbb{F}_q be given by the equation $y^2 = x^3 + ax + b$. Denote by $E^{(p)}$ the elliptic curve given by the equation

$y^2 = x^3 + a^p x + b^p$. There is a canonical isogeny $Frob : E \rightarrow E^{(p)}$ given by $(x, y) \mapsto (x^p, y^p)$. This isogeny is called the Frobenius morphism and it has degree p .

The j -invariant of E is the quantity $1728 \frac{4a^3}{4a^3 + 27b^2}$. Two elliptic curves over \mathbb{F}_q are isomorphic over a finite extension \mathbb{F}_{q^r} if and only if they have the same j -invariant. Given an element $j \in \mathbb{F}_q$, there is an elliptic curve E over \mathbb{F}_q with $j(E) = j$. For example, for $j \neq 0, 1728$ one may take $E: y^2 = x^3 + \frac{3j}{1728-j}x + \frac{2j}{1728-j}$ ($y^2 = x^3 + x$ if $j = 1728$ and $y^2 = x^3 + 1$ if $j = 0$).

An elliptic curve E over \mathbb{F}_q is called *supersingular* if for every finite extension \mathbb{F}_{q^r} there are no points in $E(\mathbb{F}_{q^r})$ of order p . The j -invariants of supersingular elliptic curves are called *supersingular j -invariants*. They all lie in \mathbb{F}_{p^2} , in particular there are finitely many such j -invariants. Elliptic curves which are not supersingular are called *ordinary*. The property of being supersingular can be recognized from the Weierstrass equation of the curve [22, Chapter 5, Thm 4.1] or from its zeta function. For more background on elliptic curves over finite fields, isogenies, and supersingular elliptic curves, see [22, Ch. 3,5].

2.3 Expander graphs

Let $G = (V, E)$ be a graph with vertex set V and edge set E . We will deal with undirected graphs, and say a graph is k -regular if each vertex has k edges coming out of it. A graph with N vertices is an *expander graph* with *expansion constant* $c > 0$ if, for any subset $U \subset V$ of size $|U| \leq \frac{N}{2}$, the boundary $\Gamma(U)$ of U (which is all neighbors of U minus all elements of U) has size $|\Gamma(U)| \geq c|U|$. It follows from the definition that any expander graph is connected (see [10] for more background on expander graphs).

There is also an algebraic way to define the expansion property of a graph. The adjacency matrix of an undirected graph is symmetric, and therefore all its eigenvalues are real. For a connected graph, G , the largest eigenvalue is k , and all others are strictly smaller ([10, Lecture 9, Fact 5.6, 5.7]). Order the eigenvalues as follows:

$$k > \mu_1 \geq \mu_2 \geq \dots \geq \mu_{N-1}.$$

Then the expansion constant c can be expressed in terms of the eigenvalues as follows: ([2])

$$c \geq \frac{2(k - \mu_1)}{3k - 2\mu_1}.$$

Therefore, the smaller the eigenvalue μ_1 , the better the expansion constant. A theorem of Alon-Boppana says that for an infinite family X_m of connected, k -regular graphs, with the number of vertices in the graphs tending to infinity, that $\liminf \lambda(X_m) \geq 2\sqrt{k-1}$, where $\lambda = \max\{|\mu_1|, |\mu_{N-1}|\}$. This motivates the definition of a *Ramanujan graph*.

Definition 1. A k -regular connected graph is a *Ramanujan graph* if the absolute value of any eigenvalue is either k or not larger than $2\sqrt{k-1}$.

A family of k -regular Ramanujan graphs is optimal with respect to the size of λ .

A random walk on an expander graph mixes very quickly. The endpoint of a random walk approximates the uniform distribution after $O(\log(N))$ steps on an expander graph with N vertices. More quantitatively, define a sequence of random variables X_0, X_1, \dots, X_ℓ , where X_i is the label of the vertex at the i -th step of a random walk on an expander graph on N vertices. Then for every δ there is an $\ell = O(\log(1/\delta))$ such that for every vertex v

$$\left| \Pr[X_\ell = v] - \frac{1}{N} \right| < \delta.$$

The constant implied by the O -notation does not depend on the size of the graph. Thus, the observation made earlier follows, for instance, by setting $\delta = 1/N^2$ (see [10, Lecture 10, Theorem 6]).

3 Construction of a hash function from an expander graph

The use of expander graphs to produce pseudo-random behaviour is well-known to complexity theorists. The idea here is to use expander graphs to produce hash functions which are collision-resistant. We give two examples of such graphs in the following sections.

The input to the hash is used as directions for walking around a graph (*without backtracking*), and the output of the hash function is the ending vertex of the walk. For a fixed hash function, the walk starts at a fixed vertex in the given graph. A family of hash functions can be defined by allowing the starting vertex to vary. We execute a walk on a k -regular expander graph by converting the input to the hash function to a base- $(k - 1)$ number whose digits then dictate which edge to take at each step. Starting at the first vertex, each step of the walk chooses an edge emanating from that vertex to follow to get to the next vertex. At each step in the walk, the choice of the edge to follow is determined by the next digit of the (converted) input. We do not allow backtracking in the walk, so only $k - 1$ choices for the next edge are allowed at each step.

4 Pizer's Ramanujan graphs

We first define the family of graphs ([18]). Let p and ℓ be two distinct prime numbers. Define the graph $G(p, \ell)$ to have vertex set, V , the set of supersingular elliptic curves over the finite field \mathbb{F}_{p^2} . We label vertices with their j -invariants, which can be computed directly from the curve equation and are a priori elements of \mathbb{F}_{p^2} . The number of vertices of $G(p, \ell)$ is $\lfloor \frac{p}{12} \rfloor + \epsilon$, where $\epsilon \in \{0, 1, 2\}$ depending on the congruence class of p modulo 12 ([22, Chapter 5, Thm 4.1]). Later, we will impose $p \equiv 1 \pmod{12}$, in which case $\epsilon = 0$. Since there are roughly $p/12$ distinct j -invariants, we will choose a linear congruential function to map j -invariants from \mathbb{F}_{p^2} injectively into \mathbb{F}_p for the output of the hash function. Thus the output of the hash function will be just $\log_2(p)$ bits. We propose to use a graph of cryptographic size $p \approx 2^{256}$.

The edge set is as follows: given a supersingular j -invariant, j_1 , choose an elliptic curve E_1 with $j(E_1) = j_1$ in the manner described in the next paragraph and a subgroup $H_1 \subseteq E_1$ of order $\ell \neq p$. Connect j_1 to $j_2 := j(E_2)$ where E_2 is the elliptic curve E_1/H_1 . A priori, since there are $\ell + 1$ subgroups of order ℓ this gives a directed $\ell + 1$ -regular graph. However, if we assume that $p \equiv 1 \pmod{12}$, then the graph can be made into an undirected graph as follows: for each subgroup $H_1 \subseteq E_1$ of order ℓ , there is a canonical choice of subgroup $H_2 \subseteq E_2$ (of order ℓ) such that $E_2/H_2 \cong E_1$. Thus, we can identify the edge associated to H_1 with the edge associated to H_2 . The reason for the assumption $p \equiv 1 \pmod{12}$ is that, to say that the subgroup H gives an isogeny $E_i \rightarrow E_j$ is not precise because you need to choose an identification of E_i/H with E_j , and that is not canonical (and the more automorphisms E_j has, the more noncanonical it is). If we assume that $p \equiv 1 \pmod{12}$, then the elliptic curves have no automorphisms other than ± 1 . If the non-canonicity is just up to automorphisms ± 1 , this works because the dual isogeny to $-f$ is minus the dual isogeny of f .

To implement the hash function, the $\ell + 1$ ℓ -torsion subgroups at each node need to be ordered in a consistent way. One method for ordering them is as follows. At a node corresponding to $j = j(E)$, the j -invariant of an elliptic curve E , start with a Weierstrass equation for E in the standard form given in [3, Lemma VIII.3]: $Y^2 = X^3 + 3kX + 2k$, where $k = \frac{j}{j-1728}$ (for $j = 0, 1728$, make some canonical choice). Using this model for the curve, find all the ℓ -torsion of E over some extension field (ℓ should be small for this to be feasible). Next, order the x -coordinates of the points within each $\mathbb{Z}/\ell\mathbb{Z}$ factor using some fixed ordering of the extension field of \mathbb{F}_p . Say P and Q are the smallest points in that ordering in their respective factor. Then order the groups as in [6, Algorithm 1]: $G_1 = \langle Q \rangle$ and $G_{1+i} = \langle P + (i - 1) * Q \rangle$ for $1 \leq i \leq \ell$. When $\ell = 2$, there is a simpler way to order the edges which will be given in Section 4.2 below.

The Ramanujan property of this graph follows from the fact that the adjacency matrix (called the ℓ^{th} Brandt matrix) gives the action of the ℓ^{th} Hecke operator on the space of weight 2 cusp forms of level p (see [12] for a nice exposition of Brandt matrices and [17] for the application to constructing a basis of modular forms). So the bound on the eigenvalues follows from the corresponding result for modular forms (the Ramanujan-Petersson conjecture proven by Eichler and Shimura in this case [20, 9]).

4.1 Walking around the graph

For C a subgroup of the group of points on an elliptic curve E , Vélou in [24] gives explicit formulas for determining the equations for the isogeny $E \rightarrow E/C$ and the Weierstrass equation of the curve E/C . We

give here the formulas when ℓ is an *odd* prime (see Section 4.2 for the formulas when $\ell = 2$). Let E be given by the equation

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

We define the following two functions in $\mathbb{F}_q(E)$. For $Q = (x, y)$ a point on $E - \{0_E\}$, define

$$\begin{aligned} g^x(Q) &= 3x^2 + 2a_2x + a_4 - a_1y \\ g^y(Q) &= -2y - a_1x - a_3, \end{aligned}$$

and set

$$\begin{aligned} t(Q) &= 2g^x(Q) - a_1g^y(Q) \\ u(Q) &= (g^y(Q))^2 \\ t &= \sum_{Q \in (C - \{0_E\})} t(Q) \\ w &= \sum_{Q \in (C - \{0_E\})} (u(Q) + x(Q)t(Q)). \end{aligned}$$

Then the curve E/C is given by the equation

$$Y^2 + A_1XY + A_3Y = X^3 + A_2X^2 + A_4X + A_6$$

where

$$\begin{aligned} A_1 &= a_1, A_2 = a_2, A_3 = a_3, \\ A_4 &= a_4 - 5t, A_6 = a_6 - (a_1^2 + 4a_2)t - 7w. \end{aligned}$$

From the Weierstrass equation of E/C we can easily determine the j -invariant of E/C . We apply Vélu's formulas for subgroups of order ℓ , and it is clear that this procedure can be done using $O(\ell)$ elliptic curve operations for each of the $\ell + 1$ groups of order ℓ .

4.2 Efficiency of hash functions from Pizer graphs when $\ell = 2$

Here are the steps to compute the output of the hash function when using supersingular elliptic curves and 2-isogenies (i.e. $\ell = 2$). Since there are 3 edges emanating from each vertex, and no backtracking is allowed in a walk, there are two choices of which edge to follow next from each vertex, and this can be determined by 1 bit as follows. Start at a vertex E_1 . Subgroups of E_1 of order 2 are each given by a single two-torsion point on the elliptic curve $E_1 : y^2 = f(x)$. The 3 non-trivial 2-torsion points are $P_i = (x_i, 0)$, where the cubic $f(x)$ factors as

$$(x - x_1)(x - x_2)(x - x_3)$$

over an extension field. As an example, when computing the isogeny ϕ which corresponds to taking the quotient by $\langle P_1 \rangle$, both of the other 2-torsion points are mapped to the same 2-torsion point $\phi(P_2) = \phi(P_3)$ on the isogenous elliptic curve, E_2 . In turn, the isogeny which corresponds to taking the quotient of E_2 by the subgroup generated by $\phi(P_2)$ is the dual isogeny $\hat{\phi}$ and leads back to E_1 . So to choose the next step from E_2 , it suffices to choose between the two other 2-torsion subgroups different from $\langle \phi(P_2) \rangle$. An efficient way to determine the 2 new 2-torsion points on E_2 is to keep \tilde{x}_2 , the x -coordinate of $\phi(P_2)$, and to factor $(x - \tilde{x}_2)$ out of the new cubic $f_2(x)$, leaving a quadratic to be factored. The roots of the quadratic can be ordered according to some convention, and one bit suffices to choose between them for the next step in the walk. So if the input bit length is n , then the hash function takes a walk of length n steps.

Using Vélu's formulas [24] one calculates that if E is given by $y^2 = x^3 + a_4x + a_6$ and the 2-torsion point Q is $(\alpha, 0)$ then the elliptic curve $E/\langle Q \rangle$ can be given by the equation

$$y^2 = x^3 - (4a_4 + 15\alpha^2)x + (8a_6 - 14\alpha^3).$$

Furthermore, the equation for the isogeny is

$$(x, y) \mapsto \left(x + \frac{(3\alpha^2 + a_4)}{x - \alpha}, y - \frac{(3\alpha^2 + a_4)y}{(x - \alpha)^2} \right).$$

Even simpler formulas for 2-isogenies can be found for example in [22, III Example 4.5], but the formula given here shows the dependence on the 2-torsion point $Q = (\alpha, 0)$.

So summarizing, each vertex corresponds to an elliptic curve E_i given by an equation $y^2 = f_i(x)$, where $f_i(x)$ is a cubic. To compute the 2-torsion subgroups at each step, factor the cubic $f_i(x)$. At each step, calculate the 2-torsion by keeping the image of the other 2-torsion point (not used to quotient by), and then factoring the quadratic. After ordering, choose which one to quotient by and apply Vélú's formulas (field operations in \mathbb{F}_p or \mathbb{F}_{p^2}).

Cost per bit of input to the hash function:

1. Find the 2-torsion:
 - a. Apply the isogeny from the previous step to one point: 7 field multiplications.
 - b. Factor out the linear factor from the cubic $f_i(x)$: one field inversion.
 - c. Factor the quadratic by completing the square and taking a square root: roughly $(3/2)\log_2(p)$ field multiplications plus a field inversion if $p \equiv 3 \pmod{4}$. If $p \not\equiv 3 \pmod{4}$, then one can do this with $2\log_2(p)$ multiplications in a residue ring of $\mathbb{F}_p[x]$ (Cippola's method). The construction of the residue ring requires $\log p$ random bits.
2. Order the 2-torsion.
3. Use Vélú to obtain the equation of the next elliptic curve: 9 field multiplications.

In addition, at the first vertex, the cubic defining the curve must be factored, and at the last step, computing the j -invariant requires several field multiplications and 1 field inversion.

An estimate of total cost can be made by estimating a field inversion as 5 field multiplications (and as usual not counting field additions). Here we did not distinguish which field multiplications occur in \mathbb{F}_p and which occur in \mathbb{F}_{p^2} , but that is at most a factor of 3 difference. Also, the above is not optimized, so there may be better ways to do some of the steps. To summarize the efficiency of the hash function under these assumptions: the cost per bit in terms of field multiplications is roughly $2\log_2(p)$.

Timings for the Hash function based on the Pizer graph. We implemented our hash function to find the actual performance of the hash function. Our results are given below. For a prime p of 192-bits and $\ell = 2$, the time per step of the walk (which is also the time per input bit) is 3.9×10^{-5} secs. This translates to a hashing bandwidth of about 25.6 Kbps. For a prime p of 256-bits, the time per input bit is 7.6×10^{-5} secs or 13.1 Kbps. The implementation was done in C, and the computer on which the timings were taken was an 64-bit AMD Opteron 252 2.6Ghz machine.

5 Collision resistance of hash functions from Pizer graphs

5.1 Definitions and hard problems

Definition 2. A hash function h is said to be *collision resistant* if it is computationally infeasible to find two distinct inputs, x, y , which hash to the same output $h(x) = h(y)$. This property is also called *strong collision resistance*. Computationally infeasible can be taken to mean that any probabilistic polynomial time algorithm succeeds with negligible probability.

Definition 3. A hash function h is said to be *preimage resistant* if, given any output of h (for which a corresponding input is not known), it is computationally infeasible to find an input, x , which hashes to that output. A hash function with this property is also called *one way*.

Fix distinct primes p and ℓ and consider the graph $G(p, \ell)$. For reasons discussed above, throughout this section assume that $p \equiv 1 \pmod{12}$, p is a prime of cryptographic size (at least 256-bits), ℓ is a relatively small prime, and let n denote the length of the input to the hash function (for example, $n = 256$). We will relate the collision resistance and preimage resistance properties of the hash function based on this graph to problems involving finding isogenies between elliptic curves, and then argue why these problems are hard.

First we show that distinct paths in the graph $G(p, \ell)$ lead to distinct isogenies.

Proposition 1. *Let $\phi : E \rightarrow E^\dagger$ be an ℓ -power isogeny produced by traversing a path in $G(p, \ell)$ without backtracking. Then ϕ decomposes (or factors) uniquely as a composition of ℓ -isogenies. This means that if we have two factorizations of ϕ as*

$$E = E_0 \xrightarrow{\phi_1} E_1 \xrightarrow{\phi_2} E_2 \xrightarrow{\phi_3} \cdots \xrightarrow{\phi_n} E_n = E^\dagger$$

and

$$E = E_0 \xrightarrow{\phi'_1} E'_1 \xrightarrow{\phi'_2} E'_2 \xrightarrow{\phi'_3} \cdots \xrightarrow{\phi'_n} E_n = E^\dagger$$

then for each i , $\ker \phi'_i = \ker \phi_i$. Consequently, for each i , $E_i \cong E'_i$ and setting $\epsilon_i : E'_i \rightarrow E_i$ to be an isomorphism, we have $\phi_i \circ \epsilon_{i-1}^{-1} = \epsilon_i \circ \phi'_i$.

Proof : First we note that if $\ker \phi$ is cyclic then the factorization of ϕ into degree ℓ isogenies is unique. Indeed, factoring ϕ is equivalent to providing a Jordan-Holder series for its kernel, which is unique if the kernel is a cyclic group.

Now suppose that the kernel of ϕ is not cyclic. We claim that in any factorization of ϕ as above we have “backtracking”, namely, if $\phi = \phi_n \circ \phi_{n-1} \circ \cdots \circ \phi_1$, a composition of isogenies of degree ℓ , there is an index m , $1 < m \leq n$ such that $E_{m-2} \cong E_m$, and, having chosen such an isomorphism, we have $\phi_m = \epsilon_m \hat{\phi}_{m-1}$, for some $\epsilon_m \in \text{Aut}(E_m)$.

Indeed, let m be the minimal integer such that $\phi_m \circ \phi_{m-1} \circ \cdots \circ \phi_1$ is not cyclic and denote this isogeny by Ψ . Note that $m > 1$ and that $\psi := \phi_{m-1} \circ \cdots \circ \phi_1$ is cyclic. The kernel of ψ is thus a cyclic group of E , isomorphic to $\mathbb{Z}/\ell^{m-1}\mathbb{Z}$, while the kernel of Ψ is a subgroup of E of order ℓ^m containing $\ker(\psi)$ and not cyclic. It is therefore isomorphic to $\mathbb{Z}/\ell^{m-1}\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$, where in this isomorphism we can always assume that $\ker(\psi)$ is the first factor $\mathbb{Z}/\ell^{m-1}\mathbb{Z}$. (By the elementary divisors theorem, applied to $\ker(\psi) \subset \ker(\Psi)$.) It then follows that $\ker(\phi_{m-2} \circ \cdots \circ \phi_1)$ is the subgroup $\mathbb{Z}/\ell^{m-2}\mathbb{Z}$ contained in the first factor of $\ker(\Psi)$ and so that $\ker(\phi_m \circ \phi_{m-1}) \cong (\mathbb{Z}/\ell^{m-1}\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}) / (\mathbb{Z}/\ell^{m-2}\mathbb{Z} \times \{0\}) \cong \mathbb{Z}/\ell\mathbb{Z} \times \mathbb{Z}/\ell\mathbb{Z}$.

We are therefore left with proving the following assertion: Let $f : E \rightarrow E'$, $g : E' \rightarrow E''$ be isogenies of degree ℓ such that $\ker(g \circ f) \cong (\mathbb{Z}/\ell\mathbb{Z}) \times (\mathbb{Z}/\ell\mathbb{Z})$ then $E \cong E''$ and, having chosen such an isomorphism $\epsilon : E \rightarrow E''$, we have $g = \epsilon \hat{f}$. First, since $\ker(g \circ f) = E[\ell]$, the kernel of an endomorphism (namely $[\ell]$), it follows that $E'' \cong E/E[\ell] \cong E$ and so, again, it is enough to show that $\ker(g) = \ker(\hat{f})$, but clearly $\ker(g) = E[\ell]/\ker(f) = \ker(\hat{f})$. \square

Problem 1. Produce a pair of supersingular elliptic curves over \mathbb{F}_{p^2} , E_1 and E_2 , and two *distinct* isogenies of degree ℓ^n between them, $f_1 : E_1 \rightarrow E_2$, $f_2 : E_1 \rightarrow E_2$. In light of the above proposition, by distinct isogenies we mean isogenies whose kernels have distinct composition series. In particular, composing an isogeny by an automorphism produces an equivalent isogeny.

Problem 2. Given E , a supersingular elliptic curve over \mathbb{F}_{p^2} , find an endomorphism $f : E \rightarrow E$ of degree ℓ^{2n} that is not any automorphism of E composed with the multiplication by ℓ^n map.

Notation: For each vertex in the graph $G(p, \ell)$ represented by the curve E_i (numbered in any order), let h_i denote the hash function defined by letting the vertex corresponding to E_i be the starting vertex for the walk. The collection of h_i form a family of hash functions. Assume the hash functions h_i take input of fixed length n_0 . Let n be the length of the corresponding paths in the graph. Note that if $\ell = 2$, then $n = n_0$.

Theorem 2. *Finding a collision in the hash function h_i implies a solution to Problem 1 with $E_1 = E_i$, and a solution to Problem 2 with $E = E_i$.*

Proof : Finding a collision for a hash function in this family amounts to finding two distinct paths between two vertices. For the hash function h_i , the first vertex is $E_1 = E_i$. Since the hash function takes inputs of fixed length n_0 , the paths must also have the same length, n . By Proposition 1, finding two distinct paths in the graph from the vertex $E_1 = E_i$ to the vertex E_2 allows one to construct two distinct isogenies $\phi_1 : E_1 \rightarrow E_2$ and $\phi_2 : E_1 \rightarrow E_2$, $\phi_1 \neq \phi_2$, via composition of degree ℓ isogenies, where $E_1 = E_i$ and E_2 are supersingular elliptic curves over \mathbb{F}_{p^2} . This shows that finding a collision solves Problem 1. Furthermore, the length constraint on the paths implies that $\deg \phi_1 = \deg \phi_2$, and the fact that the edges of the graph are ℓ -isogenies means that the degree of the two isogenies is ℓ^n . Taking the dual of ϕ_2 , we get an isogeny $\hat{\phi}_2 : E_2 \rightarrow E_1$. Now $\phi_1 \circ \hat{\phi}_2 : E_1 \rightarrow E_1$ is an endomorphism of the elliptic curve E_1 of degree ℓ^{2n} . This endomorphism cannot be the multiplication by ℓ^n map (which also has degree ℓ^{2n}), since $\phi_2 \neq \phi_1$. In other words, a collision also leads to a cycle¹ of even length in the graph. Thus, explicitly finding a collision in this hash function allows one to find two isogenies of the same ℓ -power degree between a pair of supersingular elliptic curves, and to find an ℓ^{2n} -degree endomorphism of a given supersingular elliptic curve $E = E_1 = E_i$ that is not the multiplication by ℓ^n map. \square

It is easy to see from the proof of Theorem 2 that if the attacker is allowed to find collisions between inputs of different lengths, say n and m , then one can solve a variation of Problem 1 where one is asked to find two isogenies $f_1 : E_1 \rightarrow E_2$, and $f_2 : E_1 \rightarrow E_2$ of degrees ℓ^n and ℓ^m .

Problem 3. Given E_1 and E_2 , two supersingular elliptic curves over \mathbb{F}_{p^2} , find an isogeny $f : E_1 \rightarrow E_2$ of degree ℓ^n between them.

Theorem 3. *Finding preimages for the hash function h_i implies a solution to Problem 3 with $E_1 = E_i$.*

Proof : Given an output y to the hash function h_i , let E_2 be a supersingular elliptic curve over \mathbb{F}_{p^2} whose j -invariant corresponds to y . To find an input x , such that $h_i(x) = y$, is to find a path in the graph of ℓ -isogenies from $E_1 = E_i$ to E_2 . \square

Remark 4. Note that a solution to Problem 3 implies a solution to Problem 1. This follows from the fact that a solver for Problem 3 can be used to solve Problem 1 by first taking a random walk on the graph with endpoints E_1 and E_2 , and then asking the Problem 3-solver for another path between them. If the two paths are the same, repeat. Since the graph is an expander there are many distinct paths between any two vertices. The first path was chosen at random, and consequently, the probability that the Problem 3-solver produced the same path is low. Thus with high probability we will get two distinct paths from E_1 to E_2 and hence get a solution for Problem 1. In other words there is a probabilistic polynomial time reduction from Problem 1 to Problem 3.

5.2 On the converses to Theorems 2 and 3.

As observed in Theorem 1, finding a collision implies a solution to Problem 1 and a solution to Problem 2. In the opposite direction, if a solution to Problem 2 is given in “factored” form, then it also implies a solution to Problem 1 and the ability to produce a collision. That is, if a cycle in the graph is found, written in “factored” form as a sequential list of vertices, it can be used to create two distinct paths between two vertices by following the cycle in two different directions until the paths meet. The path can be converted into an isogeny with $O(\ell^{1+\epsilon} \log^2(p))$ amount of work at each step (see [4]). However, if a solution to Problem 1 or 2 is given as an isogeny or an endomorphism, specified either by a recipe for evaluation or by its kernel (the size of the subgroup ℓ^n would presumably be too large to make this practical), then it is not clear how to decompose the isogeny or endomorphism into the path in the graph that would produce a collision. See the paragraph on factoring isogenies below. Note that the same is true for the equivalence of Problem 3 with preimage finding. If a solution to Problem 3 is given in terms of a path in the graph, then it can be used to find preimages.

A note on factoring isogenies: In the last paragraph we encountered the problem of writing an isogeny $f : E_0 \rightarrow E_n$ of degree ℓ^n as a composition of isogenies $\phi_n \circ \phi_{n-1} \circ \dots \circ \phi_1$ where $\deg \phi_i = \ell$.

¹We use the term cycle rather loosely here, as we allow a cycle to intersect itself.

One might be tempted to use Corollary III.4.11 of [22] to solve this problem. The result states that, for non-constant isogenies $\phi : E \rightarrow E_1$ and $f : E \rightarrow E'$ such that ϕ is separable, f factors as

$$\begin{array}{ccc} E & \xrightarrow{\quad f \quad} & E' \\ & \searrow \phi & \nearrow \lambda \\ & & E_1 \end{array}$$

for a unique isogeny λ if and only if $\ker \phi \subseteq \ker f$. For instance, one can use this criterion to find the first step in the “factorization” of the isogeny as follows: given $f : E_0 \rightarrow E_n$, f factors as $f' \circ \phi_1$ iff $\ker \phi_1 \subseteq \ker f$. This can be checked by taking an ℓ -torsion point P that generates a subgroup (a candidate for $\ker \phi_1$) and checking whether $f(P)$ is the identity in E_n . Doing this for each of the $\ell + 1$ possibilities for the subgroup $\ker \phi_1$ we can identify the first step of the factorization. A problem arises with this approach if one carries it to subsequent steps of the factorization. Consider the second step of this process: one needs to check for each possible isogeny $\phi_2 : E_1 \rightarrow E_2$, whether $\ker(\phi_2 \circ \phi_1) \subseteq \ker f$. Since $\deg(\phi_2 \circ \phi_1) = \ell^2$, we know that $\ker(\phi_2 \circ \phi_1) \subseteq E_0[\ell^2]$, the ℓ^2 -torsion points on E_0 . Furthermore, we know that $\ker \phi_1 \subseteq \ker(\phi_2 \circ \phi_1)$. Given that $E_0[\ell^2] \cong \mathbb{Z}/\ell^2\mathbb{Z} \times \mathbb{Z}/\ell^2\mathbb{Z}$, this means we have to find a $P \in E_0[\ell^2]$ of exact order ℓ^2 such that $\phi_1(P)$ lies in $\ker \phi_2$ (again this is tested by checking if $f(P)$ is the identity). Continuing this way, one would need to find points P in $E_0[\ell^k]$ of exact order ℓ^k . The problem is that such points in $E_0[\ell^k]$ are defined over large degree extensions of the field that E_0 is defined over. In general, this degree could be as large as ℓ^k and the finite field would have p^{ℓ^k} elements. Thus, even if f is of degree ℓ^n where n is $O(\log p)$ this approach becomes infeasible.

Another possible approach is to find the first step in the factorization ϕ_1 as we did before, so that $\phi = \phi' \circ \phi_1$, and then consider the isogeny $\phi \circ \hat{\phi}_1$ which we can now evaluate. Unfortunately, this isogeny is $\phi' \circ [\ell]$, where ϕ' is a degree ℓ^{n-1} isogeny. Thus, to find the factorization inductively one would have to evaluate the map $\phi \circ \hat{\phi}_1 \circ [1/\ell]$. Inverting the multiplication by ℓ map on the ℓ -torsion points requires one to find the ℓ^2 -torsion points on E_1 . In subsequent steps one would need to find the inverse of the map $[\ell^k]$ which, again, leads us to the issues raised in the previous paragraph of finding higher ℓ -power torsion. As a consequence, obtaining a converse to Theorem 1 (turning a solution to Problem 1 or 2 into a procedure for finding hash collisions) seems unlikely.

5.3 Hardness of the problems

Hardness of Problem 3 (Preimage resistance)

Since walks on an optimal expander graph quickly approximate the uniform distribution, we can argue heuristically that a Pollard-rho type attack on Problem 3 would succeed in time proportional to the square-root of the graph size, i.e. for the graph $G(p, \ell)$, roughly $\sqrt{\pi p}/24$ times the cost for each step, or $O(\sqrt{p} \log^2 p)$. Such an attack would not always find a path of the correct length, however. This appears to be the best attack known on any of these problems.

Problem 3 was introduced in [11], where it was argued that the problem is hard in both the ordinary and the supersingular cases. In [11], Galbraith gives an algorithm to find an isogeny between two given ordinary, isogenous elliptic curves which runs in time $O(p^{3/2} \log(p))$ assuming the Riemann hypothesis for imaginary quadratic fields. He notes that a similar algorithm to solve the same problem for supersingular elliptic curves runs in time $O(p \log(p))$. The ordinary case can also be described in another language as solving a discrete log problem in orders of class groups of imaginary quadratic number fields, which has been well-studied. Although subexponential index calculus methods apply ([13]), taking quadratic orders with large discriminant makes the problem as hard as factoring integers of that size ([14]). Note the difference between the ECDLP situation and here: problems on supersingular elliptic curves are not necessarily easier than the corresponding problem on ordinary elliptic curves. In fact, for our problem, there is no class group to work in for the supersingular case, and the degree map is a rank 4 quadratic form (the norm form associated to a maximal order in a definite quaternion algebra).

Hardness of Problems 1 and 2 (Collision resistance)

To find a cycle in the graph is to solve Problem 2, so first of all, we will ensure that our graph has no short cycles (i.e. has large girth). We will put restrictions on the congruence class of the prime p to ensure that there are no short cycles in the graph as follows.

Translation into the language of quadratic forms. The problem of finding isogenies can be translated into the language of representation of numbers by quadratic forms. As explained in the proof of Theorem 1, finding two distinct isogenies ϕ_1, ϕ_2 between two elliptic curves E_1 and E_2 of degree ℓ^n leads to an endomorphism of degree ℓ^{2n} of E_1 that is not the multiplication by ℓ^n map. The degree map is a rank 4 positive definite quadratic form, which can also be described as the norm map on a maximal order in a quaternion algebra. The endomorphism ring (over $\overline{\mathbb{F}}_p$) of a supersingular elliptic curve is isomorphic to a maximal order in the quaternion algebra $B = B_{p,\infty}$ over \mathbb{Q} ramified only at p and ∞ ([22, Chapter 5, Theorem 3.1]). The maximal order is a rank 4 \mathbb{Z} -lattice. The existence of an endomorphism of degree ℓ^{2n} implies the existence of a *non-trivial* representation (i.e. not as the norm of ℓ^n) of the number ℓ^{2n} by the quadratic form that is the norm form on the lattice. Note though, that the best known algorithms for determining the endomorphism ring of a supersingular elliptic curve as a maximal order in B are exponential in $\log(p)$ ([5]). Thus the process of translating the problem of finding cycles to the language of quadratic forms seems to be computationally hard in itself.

Ensuring that $G_{p,\ell}$ has no small cycles. We can use the machinery introduced above to efficiently find cases where $G(p,\ell)$ has no small cycles. By choosing p carefully relative to ℓ we can ensure that there are no cycles of length less than some bound. A non-trivial cycle of length $2n$ in the graph of ℓ -isogenies implies that the norm form of some maximal order in B represents ℓ^{2n} in a non-trivial way. If the cycle corresponds to an element x of norm ℓ^{2n} then that implies that the quadratic polynomial $X^2 - \text{Tr}(x)X + \text{Norm}(x)$ is irreducible, and so that p is ramified or inert in the field defined by the polynomial. To illustrate this, take $\ell = 2$ and $n = 1$. Then we consider $X^2 - \text{Tr}(x)X + 4$. Since $b^2 - 4ac < 0$, the trace must satisfy $\text{Tr}(x) \in \{-3, -2, -1, 0, 1, 2, 3\}$, so the field determined by the polynomial is $\mathbb{Q}(\sqrt{-1})$, $\mathbb{Q}(\sqrt{-3})$, $\mathbb{Q}(\sqrt{-7})$, or $\mathbb{Q}(\sqrt{-15})$. One then just needs to make sure p splits in all these fields, which by quadratic reciprocity is a congruence condition. In addition, we require that the graph have no self-loops, which adds the condition that p splits in $\mathbb{Q}(\sqrt{-2})$. So in this example it is enough that $p \equiv 1 \pmod{8}$, $p \equiv 1 \pmod{3}$, $p \equiv 1 \pmod{7}$, and $p \equiv 1 \pmod{5}$, so if p is congruent to 1 modulo $3 \cdot 8 \cdot 5 \cdot 7 = 840$ then there are no cycles of length 2. This idea can be applied in general to make sure there are no short cycles in the graph.

Choosing an appropriate starting vertex. One can apply the idea in the previous paragraph in a different way to exclude short cycles by choosing ℓ, p and the starting vertex carefully. We illustrate this by looking at the case when $p \equiv 1 \pmod{24}$. Here we have one maximal order \mathfrak{m} in $B_{p,\infty}$ whose \mathbb{Z} -basis is given by (see Proposition 5.2 of [17])

$$\frac{1}{2}(1+j), \frac{1}{2}(i+k), \frac{1}{q}(j+ak), k$$

where $i^2 = -1, j^2 = -p, q \equiv 3 \pmod{4}$ is a prime such that $\left(\frac{p}{q}\right) = -1$ and a is an integer such that $q|(a^2p+1)$. Let $q \equiv 3 \pmod{4}$ be a small prime and then pick a prime $p \equiv 1 \pmod{24}$ such that $\left(\frac{p}{q}\right) = -1$. We claim that the graph $G(p,\ell)$ for $\ell \equiv 3 \pmod{4}$ cannot have small cycles starting from any vertex representing a supersingular elliptic curve with endomorphism ring \mathfrak{m} . Indeed, a cycle of length $2t$ gives rise to an endomorphism x of E whose norm is ℓ^{2t} . This means that if $x = \frac{r}{2}(1+j) + \frac{s}{2}(i+k) + \frac{t}{q}(j+ak) + uk$ (where $r, s, t, u \in \mathbb{Z}$), then its quaternionic norm

$$N(x) = \frac{r^2}{4} + \frac{s^2}{4} + p\left(\frac{r}{2} + \frac{t}{q}\right)^2 + p\left(\frac{s}{2} + \frac{ta}{q} + u\right)^2 = \ell^{2t}.$$

Suppose $\left(\frac{r}{2} + \frac{t}{q}\right) = 0$ and $\left(\frac{s}{2} + \frac{ta}{q} + u\right) = 0$. Then $\ell^{2t} = \frac{r^2}{4} + \frac{s^2}{4}$, but since $\ell \equiv 3 \pmod{4}$, all such endomorphisms are the trivial ones coming from multiplication by ℓ^t or $u\ell^t$ (recall that i is an automorphism). Thus, we must have either $\left(\frac{r}{2} + \frac{t}{q}\right) \neq 0$ or $\left(\frac{s}{2} + \frac{ta}{q} + u\right) \neq 0$. In either case, $N(x) \geq \frac{1}{4q^2}p$. Thus $t \gg \log_\ell p$ if q is fixed. This means that there are no non-trivial endomorphisms of degree $< \frac{1}{4q^2}p$.

If the graph $G(p,\ell)$ does not have small cycles then the best known attack is the Pollard-rho attack which will find a cycle in expected time $O(\sqrt{p} \log^2 p)$. Thus taking $p \approx 2^{256}$ would give roughly 128 bits of security against this attack.

6 LPS Ramanujan graphs

6.1 Definition of the graph $X_{\ell,p}$

An alternative to using the graph $G(p, \ell)$ is to use the Lubotzky-Phillips-Sarnak expander graph ([16]). Let ℓ and p be two distinct primes, with ℓ a small prime and p relatively large. We also assume that p and ℓ are both 1 modulo 4 and ℓ is a quadratic residue (mod p) (this is the case if $\ell^{(p-1)/2} \equiv 1 \pmod{p}$). We denote the LPS graph, with parameters ℓ and p , by $X_{\ell,p}$. We define the vertices and edges that make up the graph $X_{\ell,p}$ next. The vertices of $X_{\ell,p}$ are the matrices in $\text{PSL}(2, \mathbb{F}_p)$, i.e. the invertible 2×2 matrices with entries in \mathbb{F}_p that have determinant 1 together with the equivalence relation $A = -A$ for any matrix A .

We describe the edges that make up the graph next. A matrix A is connected to the matrices $\mathfrak{g}A$ where the \mathfrak{g} 's are the following explicitly defined matrices. Let i be an integer satisfying $i^2 \equiv -1 \pmod{p}$. It follows from the Jacobi formula for the number of representations of an integer as a sum of four squares that there are exactly $8(\ell + 1)$ integer solutions (g_0, g_1, g_2, g_3) to the equation

$$g_0^2 + g_1^2 + g_2^2 + g_3^2 = \ell.$$

Among these there are exactly $\ell + 1$ with $g_0 > 0$ and odd and g_j even for $j = 1, 2, 3$. To each such (g_0, g_1, g_2, g_3) we associate the matrix

$$\mathfrak{g} = \begin{pmatrix} g_0 + ig_1 & g_2 + ig_3 \\ -g_2 + ig_3 & g_0 - ig_1 \end{pmatrix}.$$

This gives us a set S of $\ell + 1$ matrices in $\text{PGL}(2, \mathbb{F}_p)$, but their determinants are squares modulo p (since ℓ is a quadratic residue modulo p) and hence they lie in the index 2 subgroup of $\text{PGL}(2, \mathbb{F}_p)$ namely, $\text{PSL}(2, \mathbb{F}_p)$. If \mathfrak{g} is in S then so is \mathfrak{g}^{-1} , since the inverse (scaled by a square root of ℓ)

$$\mathfrak{g}^{-1} = \begin{pmatrix} g_0 - ig_1 & -g_2 - ig_3 \\ g_2 - ig_3 & g_0 + ig_1 \end{pmatrix}.$$

corresponds to the solution $(g_0, -g_1, -g_2, -g_3)$. Furthermore, since ℓ is small, the set of matrices in S can be found by exhaustive search very quickly. The graph $X_{\ell,p}$ has $p(p^2 - 1)/2$ vertices and is $\ell + 1$ -regular.

The graph $X_{\ell,p}$ is an example of a *Cayley graph*. Given a group G and a subset $S \subseteq G$, the nodes of a Cayley graph are elements of G , and there is an edge between $x, y \in G$ if $x = gy$ or $y = gx$ for some $g \in S$.

Definition of the hash function. Assume that p is of cryptographic size (1024 bits, for example), but that ℓ may be small. The hash function based on the graph $X_{\ell,p}$, denoted $\mathcal{H}_{\ell,p}$, takes an input x and executes a walk without backtracking on the graph $X_{\ell,p}$ and outputs the name of the last vertex on the walk. In more detail, we first order the generators \mathfrak{g} in S in any way. Convert the input of the hash function to a base- ℓ number whose digits then dictate which edge to take at each step. Starting at any vertex, multiply by an element of S determined by the first digit of x to get to the next vertex. Subsequent steps of the walk multiply the previous vertex by an element of $S \setminus \{\mathfrak{g}^{-1}\}$ determined by the next digit of the input, where \mathfrak{g} was the generator used in the previous step. In this way a walk without backtracking is performed on the graph. Once the input is exhausted the last matrix computed is output as the output of the hash function. The vertex corresponding to a 2×2 matrix A can be labelled using the 4-tuple of entries of A or those of $-A$, depending on which is lexicographically smaller in the usual ordering of the set $\{0, \dots, p-1\}^4$. In fact, 3 of the entries suffice to label it, since the determinant is 1.

6.2 Comparison with the Zémor-Tillich hash function

Our construction of a hash function from the graph $X_{\ell,p}$ is related to a previous construction of Zémor and Tillich. In his original paper [25], Zémor proposed using the (directed) Cayley graph of $\text{SL}(2, \mathbb{F}_p)$ with two generators in the set S : $A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ and $B = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. Zémor and Tillich subsequently discovered in [26] that cycles could be found in the graph using a variant of the Euclidean algorithm to decompose elements

of $\text{SL}(2, \mathbb{Z})$ into a product of generators. To avoid these attacks, they proposed in [27] working in the group $\text{SL}(2, \mathbb{F}_{p^n})$, with $p = 2$ and Cayley graph defined by the generators $A = \begin{pmatrix} \alpha & 1 \\ 1 & 0 \end{pmatrix}$, $B = \begin{pmatrix} \alpha & \alpha + 1 \\ 1 & 1 \end{pmatrix}$, where α is a root of the irreducible polynomial $P_n(x)$ defining the extension field \mathbb{F}_{2^n} . Unfortunately, Charney and Pieprzyk [7] found a choice of irreducible polynomial P_n of degree $n = 131$ such that the matrices A and B had very small order, and found a very short relation among A and B yielding a cycle in the graph. Since then, several attempts have been made to fix the Zémor-Tillich hash function ([1, 23]), but none have focused on using the arithmetic properties of an underlying expander graph. In §6.3 below we argue that the hash function based on the graph $X_{\ell,p}$ is not susceptible to these attacks.

6.3 Collision resistance.

In this section we show that finding a collision in the hash function defined by the graphs $X_{p,\ell}$ is equivalent to a certain group theoretic problem and give reasons why we believe this problem to be hard.

Problem 4. Let p and ℓ be two distinct primes congruent to 1 modulo 4 such that ℓ is a quadratic residue modulo p . Let $S = \{\mathfrak{g}_1, \dots, \mathfrak{g}_{\ell+1}\}$ be the generators defined in §6.1 for $\text{PSL}(2, \mathbb{F}_p)$. Find a product (in *reduced form*)

$$\prod_{1 \leq i \leq k} \mathfrak{g}_{\sigma(i)}^{e_i} = 1$$

such that $\sum_i e_i$ is $O(\log p)$ (i.e. small compared to p). By *reduced form* we mean that $\mathfrak{g}_{\sigma(i)} \neq \mathfrak{g}_{\sigma(i+1)}^{-1}$ for each i .

We remark that finding such products with *very large* exponents is not hard. Indeed, it follows from Lagrange's theorem that $\mathfrak{g}^n = 1$ where $n = \#\text{PSL}(2, \mathbb{F}_p)$, but if p is of cryptographic size, such a long cycle is useless in producing colliding inputs.

Proposition 5. *Finding a hash collision in $\mathcal{H}_{\ell,p}$ with inputs of size $O(\log p)$ (i.e. small compared to p) is equivalent to solving problem 4.*

Proof : Suppose $x \neq y$ are two inputs such that $\mathcal{H}_{\ell,p}(x) = \mathcal{H}_{\ell,p}(y)$. This happens if and only if the products of the generators corresponding to the two inputs satisfy:

$$\prod_{1 \leq i \leq r} \mathfrak{g}_{\sigma(i)} = \prod_{1 \leq j \leq s} \mathfrak{g}_{\delta(j)}$$

and this yields

$$\prod_{1 \leq i \leq r} \mathfrak{g}_{\sigma(i)} \left(\prod_{1 \leq j \leq s} \mathfrak{g}_{\delta(j)} \right)^{-1} = 1.$$

Since the set S of generators is closed under inverses this yields a relation between the generators. The reduction of this relation cannot yield the trivial relation since x and y correspond to two distinct paths in the graph. Moreover, this relation is $O(\log p)$ in length owing to the length of the colliding inputs and thus a collision is equivalent to a solution to Problem 4. \square

Large girth. The *girth* of a graph is the length of the smallest cycle in the graph. Proposition 5 says that finding a hash collision is the same as finding a cycle of length $O(\log p)$ in the graph $X_{\ell,p}$. In Sarnak, ([19, §3.4.1]), one finds that the girth, t , is the minimal integer such that ℓ^t is represented over the integers by the quadratic form

$$g_0^2 + 4p^2 g_1^2 + 4p^2 g_2^2 + 4p^2 g_3^2$$

subject to the condition that at least one of g_1, g_2, g_3 is not zero. The argument there shows that $t \geq 2 \log_\ell p$. Thus the girth of the LPS graph is at least $2 \log_\ell p$. In other words, any solution to problem 4 involves products of at least $2 \log_\ell p$ generators. Thus one cannot find short relations of the type which were used to attack the Zémor-Tillich hash functions. We remark that the girth of the LPS graph is

essentially optimal; for example, it is larger than the girth of a random graph, and in ([16] §3.3) is claimed to be the (asymptotically) largest known.

Density Attacks: The attacks on the Zémor-Tillich hash function explained in ([26] §3) are referred to as *density attacks*. By Proposition 5, it suffices to find a (short) product $R = \prod_{1 \leq i \leq k} g_{j_i}$, of the generators that yields the identity. As noted in the previous paragraph finding such an R is as hard as finding a cycle in the graph. The density attack attempts to find R by lifting the problem to the integers. More precisely, one constructs lifts $\tilde{g}_i \in \text{PSL}(2, \mathbb{Z})$ for each of the matrices g_i . Then one attempts to find a matrix \tilde{I} that reduces to the identity modulo p and a relation $\tilde{I} = \prod_{1 \leq i \leq k} \tilde{g}_{j_i}$. Reducing this equation modulo p then yields the sought after R . This attack works well for the original scheme proposed by Zémor [25] because the lifts of the generators generate a large submonoid of $\text{SL}(2, \mathbb{Z})$ and it is easy to express lifts of the identity matrix in terms of these lifted generators using a variant of the Euclidean algorithm. Two problems arise in trying this approach for $X_{p,\ell}$: The first is that the lift \tilde{I} need not lie in the group generated by the lifts \tilde{g}_i . For this to happen we need the lifts to generate a sufficiently dense subgroup of $\text{PSL}(2, \mathbb{Z})$ (hence the name “density attack”). Secondly, even if \tilde{I} belongs to the subgroup generated by \tilde{g}_i , finding the product of the generators that yield \tilde{I} is hard, since there cannot be a short expression for \tilde{I} in terms of the generators, due to the girth of the graph.

Timings for the hash function based on the LPS graphs. Our implementation of the hash function based on the LPS graph (with $\ell = 5$) takes 1.6×10^{-5} seconds per step of the walk for a prime p of 1024-bits. At each step of the walk $\log_2 \ell$ bits of the input are consumed and so this translates to a hashing bandwidth of $\frac{\log_2 \ell}{1.6 \times 10^{-5}} \approx 145$ Kbps on an 64-bit AMD Opteron 252 2.6Ghz machine. One disadvantage seems to be that an element of $\text{PSL}(2, \mathbb{F}_p)$ takes 3 elements of \mathbb{F}_p ($3 \log_2(p)$ bits) to represent (using that the determinant is 1), and if $\log_2(p)$ is about 1024, then the output size is too long. For a 192-bit prime p , one step of the walk requires 1.04×10^{-6} seconds. In terms of bandwidth this is about 2.23 Mbps (again with $\ell = 5$). More generally, one step of the walk on this graph costs 8 field multiplications (or 7 if we use Strassen’s method), so estimating the time required to do a field multiplication as α gives a direct estimate of the time required to compute the hash per bit of input as $\frac{8\alpha}{\log_2 \ell}$. One can decrease the computational cost per bit at the expense of storing a larger table (of size $\ell + 1$) of generators for the graph. But, if the table is too large then one will have to account for the memory access cost in the analysis.

7 Generic attacks on expander graph based hash functions

Our purpose in this section is to explain a certain generic method of attack on the collision resistance property of hash functions constructed out of expander graphs in the manner discussed in this paper. Let G be a connected graph and let $w = (v_0; E_1, E_2, \dots, E_n)$ be a walk in G , with initial vertex v_0 and edges E_i . Let v_n denote the vertex where the walk terminates. Let f be an automorphism of the graph G not equal to the identity. We assume that an adversary \mathcal{A} knows f and that the computation of f on any vertex and edge is efficient. Thus, applying f , \mathcal{A} can easily find $f(w) = (f(v_0); f(E_1), \dots, f(E_n))$. If, on the average, the distance in G between v and $f(v)$ is small enough then \mathcal{A} is likely to find a walk $w_{v_0, f(v_0)}$ between v_0 and $f(v_0)$ and a walk $w_{v_n, f(v_n)}$ between v_n and $f(v_n)$ by brute force search. The walks $(w_{v_0, f(v_0)} | f(w))$ and $(w | w_{v_n, f(v_n)})$ are two walks of the same length with the same initial and final vertices. Thus \mathcal{A} can find two different inputs to the hash function hashing to the same value. Alternately, the walk $(w_{v_0, f(v_0)} | f(w) | f(w_{v_n, f(v_n)}))$ represents another input (of different length, usually) hashing to the same value as w . We call such an attack a *generic attack*.

One can easily provide examples of good expanders with an involution f such that the distance between any v and $f(v)$ is one. Indeed, given a good expander graph $H = (V_H, E_H)$ let $G = (V_G, E_G)$ be its extended double cover: if $V_H = \{v_1, \dots, v_n\}$ then $V_G = \{x_1, \dots, x_n, y_1, \dots, y_n\}$ and x_i, y_j are adjacent if $i = j$, or $v_i v_j \in E_H$. This is a connected graph with the involution $f(x_i) = y_i$.

We next discuss our examples of the supersingular graphs and the LPS graphs and explain why the generic attack method fails.

Supersingular graphs. Let $p \neq \ell$ be primes, $p \equiv 1 \pmod{12}$, and let $G = G(p, \ell)$ be the supersingular graph as in section 4. The only obvious automorphism of G we have is the Frobenius automorphism

Fr , sending a supersingular j -invariant j_1 to j_1^p . It also acts on the edges: if H is a subgroup of order ℓ of a supersingular elliptic curve E_1 with $j(E_1) = j_1$ then $Fr(H)$ is a subgroup of order ℓ of $E_1^{(p)}$. The number of fixed points of Fr is the number of supersingular j -invariants defined over \mathbb{F}_p , whose order of magnitude is the class number of $\mathbb{Q}(\sqrt{-p})$, which is asymptotically $O(p^{1/2+o(1)})$ ([21]). More generally, we have the following lemma.

Lemma 6. *Let i be a non-negative integer. The number $\alpha(i)$ of supersingular j -invariants such that $\text{dist}_G(j, j^p) \leq i$ is the number of pairs (E, g) consisting of a supersingular elliptic curve E and an endomorphism g of E of degree $p \cdot \ell^j$, $j \leq i$, up to isomorphism. Assume that $i \leq \log_\ell(p/4)$. Then*

$$\alpha(i) = \ell^{i/2} \tilde{O}(\sqrt{p}).$$

Proof : Given an isogeny $h: E^{(p)} \rightarrow E$ of degree ℓ^j , $j \leq i$, let $g = Fr \circ h$ be the endomorphism of E of degree $p \cdot \text{deg}(g)$. Conversely, an endomorphism g of order $p\ell^j$, $j \leq i$ can be factored uniquely as a composition, up to automorphisms,

$$E \xrightarrow{Fr} E^{(p)} \xrightarrow{h} E,$$

where the order of h is ℓ^j . We note that to give a pair (E, g) is equivalent to giving a supersingular elliptic curve E and an embedding of the ring $\mathcal{O}_{a,j} := \mathbb{Z}[x]/(x^2 + ax + p\ell^j) \hookrightarrow \text{End}(E)$. For such an embedding to exist we must have that p does not split in the quotient field $K_{a,j}$ of $\mathcal{O}_{a,j}$ and that $K_{a,j}$ is a quadratic imaginary field. Since we have $x^2 + ax + p\ell^j = x(x+a) \pmod{p}$, for p not to split we must have $p|a$, while the second condition is simply that $a^2 < 4p\ell^j$. Note that, by assumption, $4\ell^j \leq p$, so this forces a to be zero. Thus, we need to consider pairs consisting a supersingular elliptic curve and an embedding $\mathcal{O}_j := \mathcal{O}_{0,j} = \mathbb{Z}[x]/(x^2 + p\ell^j) \hookrightarrow \text{End}(E)$. Each such embedding extends to an optimal embedding of a unique order of $K_j := \mathbb{Q}(\sqrt{-p\ell^j})$ into $\text{End}(E)$. We have assumed $p \equiv 1 \pmod{12}$, so in particular $p \equiv 1 \pmod{4}$. Then each such order is of the form \mathcal{O}_s with $s \leq j$ and $s \equiv j \pmod{2}$. It is well known that the number of such embeddings is the class number of \mathcal{O}_s and this, in turn, is $\ell^{s/2} \tilde{O}(\sqrt{p})$. Thus, we get the estimate that $\alpha(i)$ is $(\sum_{r=0}^{i/2} \ell^{(i-2r)/2}) \tilde{O}(\sqrt{p}) = \ell^{i/2} \tilde{O}(\sqrt{p})$. \square

The lemma implies that in order to have the distance between two randomly chosen supersingular elliptic curves less than i with probability greater than some constant independent of p and ℓ , one must take i close to the limit posed in the lemma, i.e. $\log_\ell(p/4)$, and this is essentially the diameter of G . This shows that the generic attack using the Frobenius automorphism fails.

LPS graphs. The LPS graphs defined in Section 6 are Cayley graphs. Let $C(G, S)$ be the Cayley graph of a group G relative to a symmetric set of generators S of G , such that $1_G \notin S$. The graph $C(G, S)$ is a simple regular connected graph. The group G acts as automorphisms of $C(G, S)$. Given $x \in G$ we have an automorphism $[x]$ of $C(G, S)$ such that $[x](g) = xg$. Note that if g is connected to gs then $[x](g)$ is connected to $[x](gs)$. The Cayley graph could have other automorphisms. Indeed, any automorphism ϕ of G such that $\phi(S) = S$ induces an automorphism of $C(G, S)$. Those, however, will not be studied here.

Let $x \neq 1_G$. Then $[x]$ has no fixed points. Suppose that for some $g \in G$, $\text{dist}(g, [x]g) = n$, where the distance is the minimal length of a walk in $C(G, S)$ starting at g and ending in xg . Thus, there are elements s_1, s_2, \dots, s_n of S such that $xg = gs_1s_2 \cdots s_n$. Then $x = gs_1s_2 \cdots s_n g^{-1}$. Assume that also $x = hs_1s_2 \cdots s_n h^{-1}$ then $h \in g \text{Cent}_G(s_1 \cdots s_n)$, and vice versa. Note that this condition on h depends only on the product $s_1s_2 \cdots s_n$ and not on the particular choice of elements s_1, s_2, \dots, s_n . We conclude the following:

$$\#\{g \in G : \text{dist}(g, [x]g) \leq n\} = \sum_{\{y \in G : 1 \leq \text{dist}(1_G, y) \leq n, x \sim y\}} \#\text{Cent}(y),$$

where we used the notation $x \sim y$ to indicate that x is conjugate to y . Let x^G denote the conjugacy class of x in G . Since conjugacy is an equivalence relation, we conclude that

$$\#\{g \in G : \text{dist}(g, [x]g) \leq n\} = \sum_{\{y \in x^G : 1 \leq \text{dist}(1_G, y) \leq n\}} \#\text{Cent}(x).$$

Remark that $\#x^G \cdot \#\text{Cent}(x) = \#G$ and so the essential point is how are the lengths of the elements in x^G (relative to the Cayley graph) are distributed. This is an interesting question in general. Here we just note that if G is k regular then there are at most $k \cdot (k-1)^{n-1}$ elements whose distance from 1_G is not

larger than n . In fact, since our interest is in good expanders, we are justified in assuming a worst case scenario.

We now specialize our considerations to the group $\mathrm{PSL}_2(\mathbb{F}_p)$. The centralizer of a non-central element in $\mathrm{SL}_2(\mathbb{F}_p)$ is roughly of size p and is at most of size $p+1$ (that element generates a quadratic algebra in $M_2(\mathbb{F}_p)$ over \mathbb{F}_p isomorphic to $\mathbb{F}_{p^2}, \mathbb{F}_p \oplus \mathbb{F}_p$ or $\mathbb{F}_p[\epsilon]/(\epsilon^2)$). Up to a factor of 2, this is also the size of the centralizer in $\mathrm{PSL}_2(\mathbb{F}_p)$. Thus, for $1 \neq x \in \mathrm{PSL}_2(\mathbb{F}_p)$ the number of vertices g such that the distance in the LPS graph (relative to ℓ and p) between g and $[x]g$ is less than n is at most $(p+1)(\ell+1)\ell^{n-1} \sim p\ell^n$, while the number of vertices is $(p^3-p)/2$. We see that in order to have that the probability of picking an element g such that $\mathrm{dist}(g, [x]g) \leq n$ exceed some constant, we must choose n to be about $2 \log_\ell(p)$, which is essentially the lower bound one has on the girth of the LPS graph. Again, we find that the generic attack method fails.

Acknowledgements The authors thank the anonymous referees for many helpful suggestions to improve the paper.

References

- [1] Abdukhalikov, K. S.; Kim, C.; *On the Security of the Hashing Scheme Based on SL_2* , in Fast Software Encryption 1998, Lecture Notes Comp. Sc. 1372 (1998), 93–102.
- [2] Alon, N.; *Eigenvalues and Expanders*, Combinatorica 6, 83–96, 1986.
- [3] Blake, I.; Seroussi, G.; Smart, N.; *Elliptic Curves in Cryptography*, Lond. Math. Soc., Lecture Note Series, **265**, Cambridge University Press, 1999.
- [4] Bostan, A.; Morain, F.; Salvy, B.; Schost, E.; *Fast algorithms for computing isogenies between elliptic curves*, <http://arxiv.org/abs/cs/0609020>.
- [5] Cerviño, J.M.; *On the correspondence between supersingular elliptic curves and maximal quaternionic orders*, <http://arxiv.org/abs/math/0404538>.
- [6] Charles, D.; Lauter, K.; *Computing Modular Polynomials*, London Math. Soc., Journal of Computational Mathematics, Vol. 8, 195–204 (2005).
- [7] Charnes, C.; Pieprzyk, J.; *Attacking the SL_2 hashing scheme*, in Advances in Cryptology - ASIACRYPT'94, J. Pieprzyk and R. Safavi-Naini, eds., vol. 917 of Lecture Notes in Computer Science, Springer-Verlag, 322–330, 1995.
- [8] Contini, S.; Lenstra, A.K.; Steinfeld, R.; *VSH, an Efficient and Provable Collision Resistant Hash Function*, Eurocrypt 2006, LNCS 4004, 165–182, Springer-Verlag 2006.
- [9] Eichler, M.; *Quaternäre quadratische Formen und die Riemannsche Vermutung für die Kongruenzzetafunktion*. Arch. Math., 5:355–366, 1954.
- [10] Goldreich, O.; *Randomized methods in Computation*, Lecture Notes. <http://www.wisdom.weizmann.ac.il/~oded/rnd-sum.html>
- [11] Galbraith, S.; *Constructing isogenies between elliptic curves over finite fields*, London Math. Soc., Journal of Computational Mathematics, Vol. 2, 118–138 (1999).
- [12] Gross, Benedict H.; *Heights and the special values of L -series*. Number theory (Montreal, Que., 1985), 115–187, CMS Conf. Proc., 7, Amer. Math. Soc., Providence, RI, 1987.
- [13] Hafner, J. L.; McCurley, K. S.; *A rigorous subexponential algorithm for computation of class groups*. Journal of the American Mathematical Society **2** (1989), 837–850.
- [14] Hamdy, S.; Möller, B.; *Security of Cryptosystems Based on Class Groups of Imaginary Quadratic Orders* T. Okamoto (Ed.): Advances in Cryptology ASIACRYPT 2000, LNCS 1976, 234–247, Springer-Verlag 2000.
- [15] Lenstra, A.K.; Stam, M.; Page, D.; *Discrete logarithms variants of VSH*, Proceedings Vietcrypt 2006, LNCS 4341, 229–242, Springer-Verlag 2006.
- [16] Lubotzky, A.; Phillips, R.; Sarnak, P.; *Ramanujan graphs*. Combinatorica 8 (1988), no. 3, 261–277.
- [17] Pizer, A.K.; *An algorithm for computing modular forms on $\Gamma_0(N)$* . J. Algebra 64 (1980), no. 2, 340–390.

- [18] Pizer, A.K.; *Ramanujan Graphs and Hecke Operators*, Bulletin of the AMS, Volume 23, Number 1, July 1990.
- [19] Sarnak, P.; *Some Applications of Modular Forms*, Series: Cambridge Tracts in Mathematics **99**, Cambridge University Press, 1990.
- [20] Shimura, G.; *Correspondances modulaires et les fonctions zeta de courbes algébriques*, J. Math. Soc. Japan 10 (1958) 1–28.
- [21] Siegel, C. L.; *Über die Classenzahl quadratischer Zahlkörper*, Acta Arith. 1 (1935), 83–86.
- [22] Silverman, Joseph, H.; *The Arithmetic of Elliptic Curves*, Graduate Texts in Mathematics, **106**, Springer-Verlag, 1986.
- [23] Steinwandt, R.; Grassl, M.; Geiselmann, W.; Beth, T.; *Weaknesses in the $SL_2(F_{2^n})$ Hashing Scheme*, in CRYPTO 2000, Lecture Notes Comp. Sc. 1880 (2000), 287–299.
- [24] Vélou, Jacques; *Isogénies entre courbes elliptiques*, C. R. Acad. Sc. Paris, **273**, 238–241, 1971.
- [25] Zémor, G.; *Hash functions and Cayley Graphs*, Designs, Codes and Cryptography, 4, 381–394, 1994.
- [26] Zémor, G.; Tillich, J.-P.; *Group theoretic hash functions*, in the First French-Israeli workshop on Algebraic coding, Lecture notes in Computer Science, Vol. 781, Springer-Verlag, 1993.
- [27] Zémor, G.; Tillich, J.-P.; *Hashing with SL_2* , Advances in Cryptology, Crypto'94, Lecture Notes in Computer Science, Vol. 839, 1994.