Cryptographic hash functions from expander graphs

Denis Charles, Microsoft Research
Eyal Goren, McGill University
Kristin Lauter, Microsoft Research

ECC 2006, Fields Institute
September 18, 2006

Background

- Crypto04 Rump session: collisions found in the most commonly used hash functions MD4, MD5, …
- SHA-0, SHA-1 also under attack
- NIST organizes a series of workshops (2005, 2006) and a competition (2007-08) to select new hash functions

Hash functions

- A *hash function* maps bit strings of some finite length to bit strings of some fixed finite length
- easy to compute
- unkeyed (unkeyed hash functions do not require a secret key to compute the output)
- Collision resistant

Collision-resistance

- A hash function *h* is *collision resistant* if it is computationally infeasible to find two distinct inputs, *x, y*, which hash to the same output $h(x) = h(y)$.
- A hash function h is *preimage resistant* if, given any output of h, it is computationally infeasible to find an input, x, which hashes to that output.

Provable hash function

- Goal: to construct efficiently computable collision-resistant hash functions.
- It is a *provable hash function* if to compute a collision is to solve some other well-known hard problem, such as factoring or discrete log.

Related work: (provable hashes)

- VSH [Contini, Lenstra, Steinfeld, 2005]
- ECDLP-based [?]
- Zemor-Tillich `94, Hashing with $SL_2(Z)$
- Joye-Quisquater, `97,
- Quisquater 2004, Liardet 2004
- Goldreich, 2000, One-way functions from LPS graphs

Construction of the hash function:

- k-regular graph G
- Each vertex in the graph has a label

Input: a bit string

- Bit string is divided into blocks
- Each block used to determine which edge to follow for the next step in the graph
- No backtracking allowed!

Output: label of the final vertex of the walk

Simple idea

- Random walks on *expander* graphs are a good source of pseudo-randomness
- Are there graphs such that finding collisions is hard? (i.e. finding distinct paths between vertices is hard)
- Bad idea: hypercube (routing is easy, can be read off from the labels)

What kind of graph to use?

- Random walks on *expander* graphs mix rapidly: log(n) steps to a random vertex
- *Ramanujan* graphs are optimal expanders
- To find a collision: find two distinct walks of the same length which end at same vertex, which you can easily do if you can find cycles

Expander graphs

- G = (V,E) a graph with vertex set V and edge set E.
- A graph is k-regular if each vertex has k edges coming out of it.
- An *expander graph* with N vertices has expansion constant $c > 0$ if for any subset U of V of size

$$|U| \leq N/2,$$

the boundary (neighbors of U not in U)

$$|\Gamma(U)| \geq c|U|.$$

Expansion constant

- The adjacency matrix of an undirected graph is symmetric, and therefore all its eigenvalues are real.
- For a connected k-regular graph, G, the largest eigenvalue is k, and all others are strictly smaller

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

- Then the expansion constant c can be expressed in terms of the eigenvalues as follows:

$$c \geq 2(k - \mu_1)/(3k - 2\mu_1)$$

- Therefore, the smaller the eigenvalue μ_1, the better the expansion constant.

Ramanujan graphs

- Theorem (Alon-Boppana) X_m an infinite family of connected, k-regular graphs, (with the number of vertices in the graphs tending to infinity), that

$$\liminf \mu_1(X_m) \geq 2\sqrt{(k-1)}.$$

- Def. *Ramanujan graph*, a k-regular connected graph satisfying $\mu_1 \leq 2\sqrt{(k-1)}$.

Example: graph of supersingular elliptic curves modulo p (Pizer)

- Vertices: supersingular elliptic curves mod p
- Curves are defined over $GF(p^2)$
- Labeled by j-invariants
- Vertices can also be thought of as maximal orders in a quaternion algebra
- # vertices ~ p/12
- $p \sim 2^{256}$

Pizer graph

- Edges: degree ℓ isogenies between them
- $k = \ell+1$ – regular
- Graph is Ramanujan (Eichler, Shimura)
- Undirected if we assume p == 1 mod 12

Isogenies

- The degree of a separable isogeny is the size of its kernel
- To construct an ℓ -isogeny from an elliptic curve E to another, take a subgroup-scheme C of size ℓ, and take the quotient E/C.
- Formula for the isogeny and equation for E/C were given by Velu.

One step of the walk: (ℓ=2)

- $E_1 : y^2 = x^3 + a_4 x + a_6$
- $j(E_1) = 1728*4a_4^3/(a_4^3+27a_6^2)$
- 2-torsion point $Q = (r, 0)$
- $E_2 = E_1 /Q$ (quotient of groups)
- $E_2 : y^2 = x^3 – (4a_4 + 15r^2)x + (8a_6 – 14r^3)$.
- E_1 à E_2
- (x, y) à $(x + (3r^2 + a_4)/(x-r), y – (3r^2 + a_4)y/(x-r)^2)$

Collision resistance

Finding collisions reduces to finding isogenies between elliptic curves:

- Finding a collisionà finding 2 distinct paths between any 2 vertices (or a cycle)
- Finding a pre-imageà finding any path between 2 *given* vertices
- $O(\sqrt{p})$ birthday attack to find a collision

Hard Problems ?

- **Problem 1**. Produce a pair of supersingular elliptic curves, E_1 and E_2, and two distinct isogenies of degree ℓ^n between them.
- **Problem 2**. Given E, a supersingular elliptic curve, find an endomorphism f : E à E of degree ℓ^{2n} , not the multiplication by ℓ^n map.
- **Problem 3**. Given two supersingular elliptic curves, find an isogeny of degree ℓ^n between them.

Timings

- p **192**-bit prime and $\ell = 2$
- Time per input bit is 3.9 × 10–5 secs.
- Hashing bandwidth: 25.6 Kbps.
- p **256**-bit prime
- Time per input bit is 7.6 × 10–5 secs or
- Hashing bandwidth: 13.1 Kbps.
- 64-bit AMD Opteron 252 2.6Ghz machine.

Other graphs

- Vary the isogeny degree
- Lubotzky-Phillips-Sarnak Cayley graph
  - random walk is efficient to implement
  - Ramanujan graph
  - Different problem for finding collisions