Efficiency of SIDH/Isogeny Signatures

Robert Gorrie

Department of Computing & Software, McMaster University

November 16th, 2017

Table of Contents

Table of Contents

Current Performance of SIDH

Key Exchange Protocol	Speed	Overhead
Code-based	0.5 ms	360 KiB
NTRU	0.3-1.2 ms	1 KiB
Ring-LWE	0.2-1.5 ms	2-4 KiB
LWE	1.4 ms	v11 KiB
SIDH	15-400 ms	0.5 KiB

https://s3.amazonaws.com/files.douglas.stebila.ca/files/research/presentations/20170918-QCrypt.pdf

Isogeny Based Signatures

- Yoo et. al provide an isogeny based signature scheme built off the Microsoft SIDH 1.0 Library.
- The scheme is constructed using the Zero Knowledge Proof of Identity protocol provided in the original SIDH paper in tandem with Unruh's PQ secure Fiat-Shamir transform.
- The scheme involves performing 248 (seperate) instances of SIDH key exchange with an arbitrary thirdparty
- These instances are parallelizable but overall extremely computationally expensive

Isogeny Signature Parameter Sizes

Scheme	Public-key size	Private-key size	Signature size
Hash-based	1,056	1,088	41,000
Code-based	192,192	1,400,289	370
Lattice-based	7,168	2,048	5,120
Ring-LWE-based	7,168	4,608	3,488
Multivariate-based	99,100	74,000	424
Isogeny-base	768	48	141,312

https://eprint.iacr.org/2017/186.pdf

Table of Contents

Batched Partial Inversions

Inversions are traditionally one of the most expensive operations in modular arithmetic. We implement a procedure that further reduces the inversion count in this signature scheme.

Our procedure utilizes three things:

- The underlying parallel structure of the isogeny signature scheme

Batched Partial Inversions

Inversions are traditionally one of the most expensive operations in modular arithmetic. We implement a procedure that further reduces the inversion count in this signature scheme.

Our procedure utilizes three things:

- The underlying parallel structure of the isogeny signature scheme
- Inversion batching

Batched Partial Inversions

Inversions are traditionally one of the most expensive operations in modular arithmetic. We implement a procedure that further reduces the inversion count in this signature scheme.

Our procedure utilizes three things:

- The underlying parallel structure of the isogeny signature scheme
- Inversion batching
- Partial inversion of $\mathbb{F}_{p^2}$ elements

Batching Inversions

Consider first how we can restructure n $\mathbb{F}_{p^2}$ inversions so that they can be done with 1 $\mathbb{F}_{p^2}$ inversion and roughly $3n$ $\mathbb{F}_{p^2}$ multiplications.

Partial Inversion Procedure

Consider now the following method for reducing one $\mathbb{F}_{p^2}$ inversion to 4 $\mathbb{F}_p$ multiplications and 1 $\mathbb{F}_p$ inversion.

1: $t_0 \leftarrow a_0^2$
2: $t_1 \leftarrow a_1^2$
3: $den \leftarrow t_0 + t_1$
4: $den \leftarrow den^{-1}$
5: $a_0 \leftarrow a_0 * den$
6: $a_1 \leftarrow a_1 * den$

Batched Inversion Procedure

If we combine these two procedures we can reduce n $\mathbb{F}_{p^2}$ inversions to:

- 1 $\mathbb{F}_p$ inversion
- $3(n-1)$ $\mathbb{F}_p$ multiplications
- $2n$ $\mathbb{F}_p$ multiplications
- $2n$ $\mathbb{F}_p$ squarings

Or, roughly 1 $\mathbb{F}_p$ inversion and 7n $\mathbb{F}_p$ multiplications

Performance Increase

The following are measured in billions of clock cycles

Procedure	Without Batching	With Batching
Signature Sign	15.74	15.56
Sign Parallel	10.23	10.13
Signature Verify	11.18	10.8
Verify Parallel	7.27	7.11

- In the serial setting we see a 1.1% and a 3.5% performance increase for Signing and Verifying, respectively.
- Comparatively, in the parallel setting we see a 0.9% and a 2.3% performance increase.

Table of Contents

SIDH Key Compression

A recent paper from Microsoft Research showed that SIDH public keys could be compressed to 330 bytes while retaining 128 bits of security in the quantum setting.

These results, along with other changes, we're implemented in the second installment of the Microsoft SIDH library, which Yoo's signature scheme has yet to be subjected to.

Signature Compression

Because isogeny signatures are composed largely of SIDH public keys, Microsofts recently published techniques can offer further compression of these signatures.

$$\sigma \cong (\mathtt{pk}_i, \mathtt{sk}_i, h_i)$$

$$\text{for } 1 \leq i \leq 248$$

Because public keys are distinctly seperate, we can use the same batched partial inversion procedure to cut down on time spent inverting $\mathbb{F}_{p^2}$ elements in compression and decompression.

Table of Contents

Other Possible Performance Increases

- Bos & Friedberger have shown that using different parameters in the calculation of the underlying modulus can yield more efficient implementations for base field arithmetic.
- Costello & Hisil have recently published algorithms for efficiently computing arbitrary degree isogenies.
- We hope to, through application of Costello and Hisil's work, explore the implications of Bos and Friedbergers findings on the SIDH library.

Preparing SIDH 2.0 for Open Quantum Safe

- Restructure and reorganize codebase with more ephasis on readability.
- Further work to improve the efficiency of signing, verifying, compression, and decompression algorithms
- We hope to work the protocol into an environment where it's practicality can more easily and assuredly be tested

SIDH 2.0 Fork with Signatures:
https://github.com/GorrieXIV/PQCrypto-SIDH-gxivFork