Constant-time Lattice Reduction for SQIsign

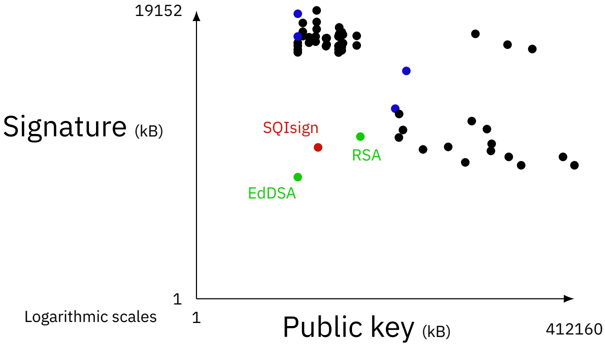
O. Hanyecz, A. Karenin, E. Kirshanova, P. Kutas, **S. Schaeffler**

The Isogeny Club
December 17, 2024

+ Post-quantum signature
→ Submitted to NIST in 2023

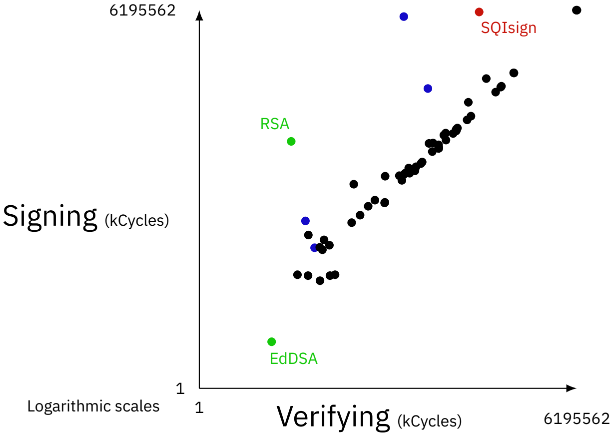
Eprint 2020/1240 and 2022/234
Logo from sqisign.org

Comparison of sizes of round 2 candidates



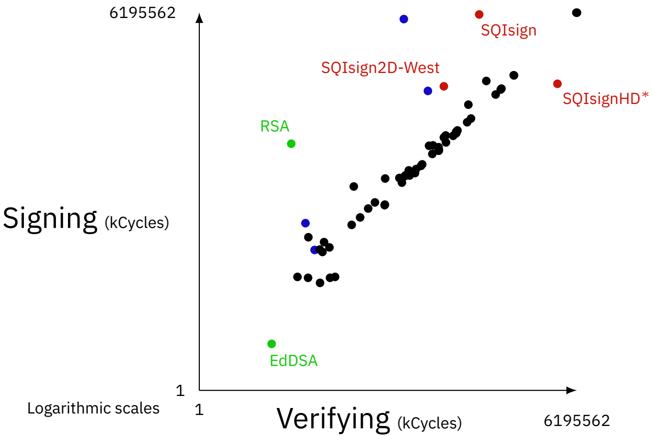
Data from https://pqshield.github.io/nist-sigs-zoo/
(13 December 24)

Comparison by speed



Data from https://pqshield.github.io/nist-sigs-zoo/
(13 December 24)

Comparison by speed



Data from https://pqshield.github.io/nist-sigs-zoo/
Eprint 2023/436 and 2024/760 (13 December 24)

SQIsign current status

+ Small
+ Not that slow any more
+ Submitted for standardization

$\rightarrow$ Scary: Maybe one day it could be used!

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Side channels:

- Runtime
- Power consumption
- Memory accesses
- Faults

Practical security in cryptography

Mathematical security:
Ensure that given only public information, an adversary cannot break the system

Implementation security:
Ensure observing the computation only gives public information to an adversary

Side channels:

- Runtime
- Power consumption
- Memory accesses
- Faults

Outline

1. Practical SQIsign
   - Real-world SQIsign?
   - Lattice reduction in SQIsign
2. Lattice reduction in constant-time
   - Algorithm
   - Parameters and implementation

Short Quaternion and Isogeny Signature

SQIsign signing (all variants)

Core idea: Compute ϕ_{resp} to prove knowledge of $\text{End}(E_1)$ and $\text{End}(E_2)$

$$E_1 \xrightarrow[\phi_{\text{resp}}]{} E_2$$

Infinitely many $\phi_{\text{resp}} : E_1 \to E_2$ exist

Which response?

Required: $\phi_{\text{resp}} : E_1 \to E_2$

- Representable (HD or smooth)
- Independent of secrets
- Short (small degree)

Set S of isogenies $\phi : E_1 \to E_2$ is isomorphic to an ideal in a quaternion algebra

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
- \+ with quadratic form called norm N

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
+ with quadratic form called norm N

Ideal:

- rank 4 lattice
  - Lattice: set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis

Quaternion ideals

Quaternion algebra:

- 4-dimensional vector space over $\mathbb{Q}$
- with quadratic form called norm N

Ideal:

- rank 4 lattice
  - Lattice: set of $\mathbb{Z}$-linear combinations of a $\mathbb{Q}$-basis
- integer norm: $N(I) = \gcd_{x \in I} N(x)$

Correspondence with sizes

The signer knows an ideal I corresponding to response set S such that:

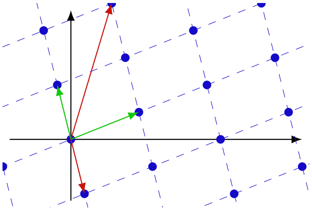
1. To each $x \in I$ corresponds an isogeny ϕ_x
2. With $\deg(\phi_x) = \frac{N(x)}{N(I)}$

$\rightarrow$ Sufficient to find some short element(s) in *lattice* I!

Lattice reduction

Given: A lattice basis B

Find: A *reduced* basis B' of the same lattice as B



Reduced: Containing only vectors which are

- of somewhat small norm
- and somewhat orthogonal

Several definitions exist

Lattice reduction in lattice cryptography?

Lattice-crypto:

- Large dimension over $\mathbb{Q}$
- Often smaller integers

$\rightarrow$ Optimization for high dimension

- Lattice reduction in cryptanalysis

$\rightarrow$ Need fast reduction

SQIsign:

- Dimension 4 over $\mathbb{Q}$
- Large integers

$\rightarrow$ Optimization for large coefficients

- Lattice reduction used constructively

$\rightarrow$ Need secure reduction

Lattice reduction algorithm: LLL

Require: B basis of a lattice L of rank d and $c \in]1/4, 1[$
Ensure: B an c-LLL-reduced basis of L

$B^* :=$ Gram-Schmidt-Orthogonalize(B)
while B is not reduced **do**
    Size-reduce B , update B^*
    for i from 1 to $d-1$ **do**
        if not LLLcondition(c, i, B, B^*) **then**
            Swap b_i, b_{i+1} in B, update B^*, continue
        end if
    end for
end while
return B

From “Factoring polynomials with rational coefficients” by A. Lenstra, H. Lenstra, L. Lovász, 1982
Description adapted from H. Cohen’s “A Course in Computational Algebraic Number Theory”, 1993

Lattice reduction algorithm: Greedy

Require: B basis of a lattice L of rank $d \leq 4$, G its Gram matrix
Ensure: B a Minkowski-reduced basis of L, G its Gram matrix

```
done := False
while not done and d > 1 do
    Sort (b_1, ..., b_d) by norm, adapt B and G;
    b_1, ..., b_{d-1}, G' := Greedy(b_1, ..., b_{d-1}) adapt B and G
    b_d := b_d - c where c is closest to b_d in the lattice of b_1, ..., b_{d-1}, adapt B and G;
    done := (N(b_d) >= N(b_{d-1}))
end while
return B, G
```

From "Low-Dimensional Lattice Basis Reduction Revisited" by P. Q. Nguyen, D. Stehlé, 2004
(DOI 10.1007/978-3-540-2)

Lattice reduction algorithm: BKZ-2

Require: B basis of a lattice L of rank 4, parameter $\delta < 1$
Ensure: B a reduced basis of L

LLL-reduce(B)
while First tour or B has changed in previous tour **do**
    for i from 1 to 3 **do**
        $b :=$SVP(b_i, b_{i+1})
        if δ-condition(b, B) **then**
            Insert b in B
        end if
        LLL-reduce(B)
    end for
end while
return B

From “Lattice basis reduction: Improved practical algorithms and solving subset sum problems.”
by C. P. Schnorr and M. Euchner, 1991 (DOI 10.1007/3-540-54458-5_51); Description from Eprint 2011/198

Lattice reduction algorithm: BKZ-2 for analysis

Require: B basis of a lattice L of rank 4, optional T_m max iteration number
Ensure: B a reduced basis of L if T_m large enough

```
while B has changed in previous tour and T_m not reached do
    for i from 1 to 3 do
        b_1, b_{i+1} := HKZ-reduce(b_i, b_{i+1})
        Size-reduce(B)
    end for
end while
return B
```

From "Terminating BKZ" by G. Hanrot, X. Pujol and D. Stehlé, 2011
(Eprint 2011/198)

Constant-time BKZ-2

Require: B basis of a lattice L of rank 4, iteration counts T_{Lagr}, T_{BKZ}
Ensure: B' a reduced basis of L if T_{Lagr}, T_{BKZ} are large enough

$B', G, B := B$, its Gram matrix, its orthogonalization
for j from 1 to T_{BKZ} **do**
  for i from 1 to 3 **do**
    Constant-time size-reduce b'_i, adapt G and B^*;
    Constant-time Lagrange-reduce $(b'_i, b'_{i+1}, T_{Lagr})$, adapt B', B^*, G
    Constant-time size-reduce b'_i then b'_{i+1} , adapt G and B^*
  end for
end for
return B'

Subroutines

- **Partial size-reduction**
  - Runtime only depends on indices
  - Easily constant-time
- **Lagrange-reduction**
  - Euclid-like algorithm
  - Unclear dependency of runtime on inputs

Lagrange reduction

Require: b_1, b_2 basis of a rank-2 lattice L
Ensure: c, d basis of L with c minimal in L

```
while First round or N(d) < N(c) do
    μ = ⌊(N(c+d) − N(c) − N(d)) / 2N(d)⌉;
    c, d := d, c − μd
end while
return c, d
```

$$\mu = \left\lfloor \frac{N(c+d)-N(c)-N(d)}{2N(d)} \right\rceil$$

Towards a constant-time Lagrange-reduction

- Bound the number of loop iterations
- Ensure “additional” iterations don’t harm output
- Optimize: Minimize operations on basis

Constant-time Lagrange reduction

Require: $G \in \mathbb{Z}^{2\times 2}$ Gram matrix of a basis B, T_{Lagr} number of iterations;
Ensure: $U \in \mathbb{Z}^{2\times 2}$ an unimodular matrix such that BU is Lagrange reduced;

$U :=$ dimension 2 identity matrix
for $c = 1$ to T_{Lagr} **do**
    $\mu := \lfloor G_{1,0}/G_{0,0} \rceil$
    $G_{1,1} = G_{1,1} - (2\mu G_{1,0} - \mu^2 G_{0,0})$
    $G_{1,0} = G_{1,0} - \mu G_{0,0}$
    $U_1 = U_1 - \mu U_0$
    CT-SWAP$(G_{0,0}, G_{1,1})$
    CT-SWAP$(U_{0,0}, U_{1,1})$
end for
CT-CONDITIONAL-SWAP$_{G_{1,1} < G_{0,0}}(U_0, U_1)$
return U

Constant-time BKZ-2

Require: B basis of a lattice L of rank 4, iteration counts T_{Lagr}, T_{BKZ}
Ensure: B' a reduced basis of L if T_{Lagr}, T_{BKZ} are large enough

$B', G, B := B$, its Gram matrix, its orthogonalization
for j from 1 to T_{BKZ} **do**
  for i from 1 to 3 **do**
    Constant-time size-reduce b'_i, adapt G and B^*
    Constant-time Lagrange-reduce $(b'_i, b'_{i+1}, T_{Lagr})$, adapt B', B^*, G
    Constant-time size-reduce b'_i then b'_{i+1}, adapt G and B^*
  end for
end for
return B'

Iteration counts

Ensure:
$$\|b_0\| \leqslant 2\left(\tfrac{4}{3}\right)^{3/2} D^{1/4}$$

Require:
$$T_{\mathsf{BKZ}} \geq \frac{2}{\log_2(8/7)} \log_2\left(\log_2\left(\frac{B^*}{D^{1/4}}\right) + \sqrt{5}(\log(4/3))^{1/2}\right)$$

$$T_{\mathsf{Lagr}} \geq 2 + 2\lceil(\log_{\sqrt{3}} 2)\,(9\log_2 B + 12)\rceil$$

B: Square root of largest diagonal coefficient of Gram matrix of the input
B^*: Square root of largest norm of a vector in the orthogonalization of the input
D: Lattice volume

For SQIsign inputs: large ideals in HNF

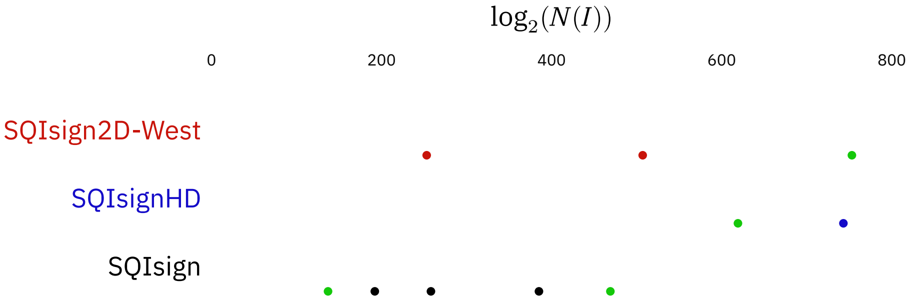
Ensure:
$\deg(\phi_{b_0}) \leqslant \frac{128}{27}\sqrt{p}$

Require:
$$T_{\mathsf{BKZ}} \geq \frac{2}{\log_2(8/7)} \log_2\left(\frac{1}{2}\left(\log_2(N(I)) - \frac{1}{2}\log_2(p/4)\right) + \sqrt{5}(\log_2(4/3))^{1/2}\right)$$

$$T_{\mathsf{Lagr}} \geq 2 + 2\left\lceil (\log_{\sqrt{3}} 2)\left(\frac{9}{2}\log_2(N(I)) + 12\right)\right\rceil$$

$N(I)$ ideal norm
p prime, parameter of the algebra

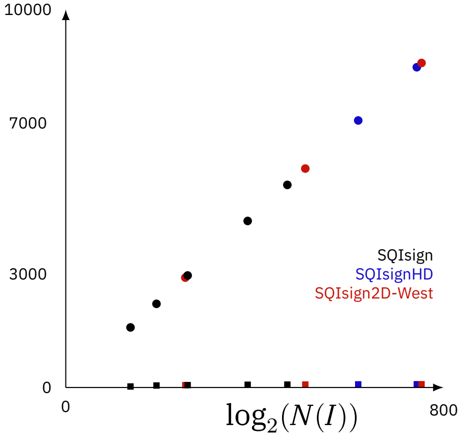
SQIsign LLL calls at level 1



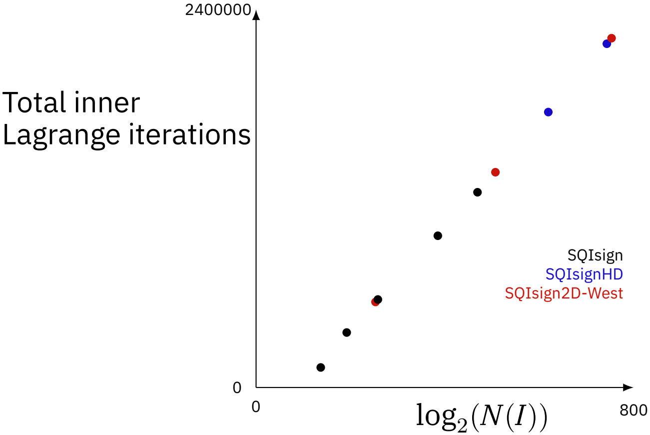
Parameter example with SQIsign LLL calls at LVL1

Iterations:

- T_{Lagr}
- T_{BKZ}



Runtime estimation SQIsign LLL calls (lvl1)



Gap to practice

Example:

bitsize $N(I)$	T_{Lagr} theory	T_{Lagr} no failure observed	T_{BKZ} theory	T_{BKZ} no failure observed
260	2986	9	64	3

Possible reasons:

? LLL analysis not tight

? Worst case never met

And if we did less iterations?

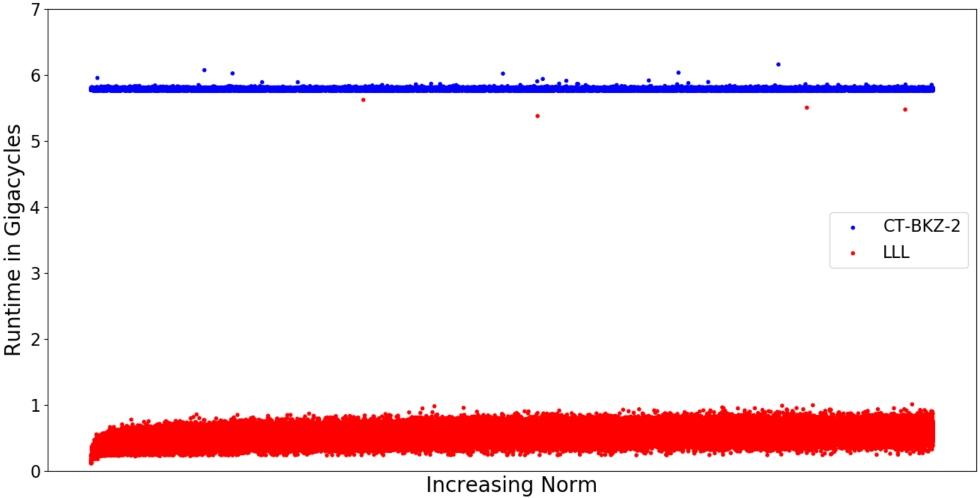
- $+$ Guaranteed constant runtime
- $+$ Output is lattice basis
- $-$ Output might not be sufficiently reduced

$\rightarrow$ Running with reduced tours where risk is acceptable

$\rightarrow$ Reasonable runtime

$\rightarrow$ Experiments possible

CT-BKZ-2 is constant-time



CT-BKZ-2 compared to (unoptimized) LLL

Algorithm Parameters	old LLL	CT-BKZ-2	T_{BKZ}	T_{Lagr}
LVL1	8,72	36,0	5	9
LVL3	17,3	126	6	12
LVL5	26,4	374	10	18

All timings in gigacycles on an Intel i7-11850H processor

For $\log_2(N(I))$ about $260, 385, 508$ respectively
and $\log_2(p)$ equal $254, 378, 502$ respectively

CT-BKZ-2 compared to (unoptimized) LLL

Algorithm	old LLL	CT-BKZ-2
Integers	CT	CT
LVL1	8,72 37 words	36,0 37 words
LVL3	17,3 55 words	126 55 words
LVL5	26,4 72 words	374 72 words

All timings in gigacycles on an Intel i7-11850H processor

For $\log_2(N(I))$ about $260, 385, 508$ respectively
and $\log_2(p)$ equal $254, 378, 502$ respectively

CT-BKZ-2 compared to (unoptimized) LLL

Algorithm	old LLL	CT-BKZ-2	
Integers	CT	CT	short CT
LVL1	8,72 37 words	36,0 37 words	9,78 20 words
LVL3	17,3 55 words	126 55 words	28,8 28 words
LVL5	26,4 72 words	374 72 words	107 37 words

All timings in gigacycles on an Intel i7-11850H processor

For $\log_2(N(I))$ about $260, 385, 508$ respectively
and $\log_2(p)$ equal $254, 378, 502$ respectively

CT-BKZ-2 compared to (unoptimized) LLL

Algorithm	old LLL		CT-BKZ-2	
Integers	CT	non-CT	CT	short CT
LVL1	8,72 37 words	0,0016	36,0 37 words	9,78 20 words
LVL3	17,3 55 words	0,0025	126 55 words	28,8 28 words
LVL5	26,4 72 words	0,0031	374 72 words	107 37 words

All timings in gigacycles on an Intel i7-11850H processor

For $\log_2(N(I))$ about $260, 385, 508$ respectively
and $\log_2(p)$ equal $254, 378, 502$ respectively

Integers and other limitations

- Current bottleneck: constant-time GCD
  - Rationals
  - Constant-time GCD self-implemented
  - Very large numbers
- Other number types?

Future hopes?

$\rightarrow$ Use compiler optimization

$\rightarrow$ Other number types

? Can SQIsign use reduced iterations ?

Future hopes?

→ Use compiler optimization

→ Other number types

? Can SQIsign use reduced iterations ?

Questions?