Concrete NTRU Security and Advances in Practical Lattice-Based Electronic Voting

Patrick Hough[1] Caroline Sandsbråten[2] Tjerand Silde[2]

[1] Mathematical Institute
Oxford University

[2] Department of Information Security and Communication Technology
Norwegian University of Science and Technology

Outline

1. Electronic voting, an overview.
2. Why NTRU and how hard is it?
3. A new electronic voting scheme.

Electronic Voting

Electronic Voting

Steady adoption including for national voting in Estonia, Switzerland, France, and Australia, as well as for corporations and organisations.

Electronic Voting

Steady adoption including for national voting in Estonia, Switzerland, France, and Australia, as well as for corporations and organisations.

Benefits

- Efficiency (speed and cost)
- Higher turnout
- Verifiability
- Distribution

Electronic Voting - The Protocol

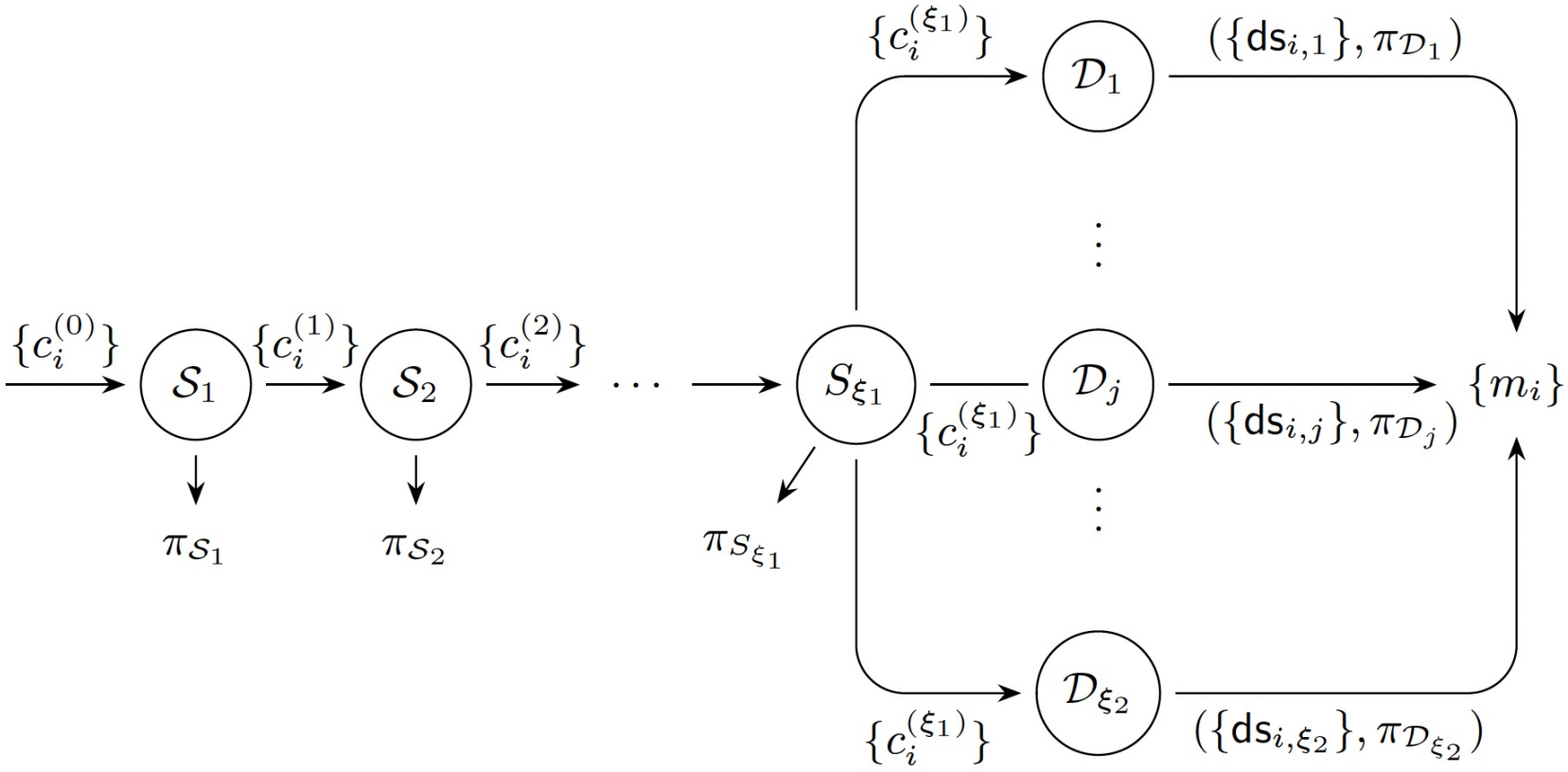

Figure: Voting protocol with sets of shuffle servers $\mathscr{S}_i$ and decryption servers $\mathscr{D}_j$ [ABGS23].

Electronic Voting - Security Assumptions

Deployed schemes based on discreet log-like assumptions.

Electronic Voting - Security Assumptions

Deployed schemes based on discreet log-like assumptions.

State-of-the-art post-quantum work based on RSIS and RLWE.

Verifiable Mix-Nets and Distributed Decryption for Voting from Lattice-Based Assumptions

CCS 2023, *Diego F. Aranha, Carsten Baum, Kristian Gjøsteen, Tjerand Silde*

Notation

Notation

- We denote $R_q = \frac{\mathbb{Z}[X]}{(X^d+1,q)}$ and $R_p = \frac{\mathbb{Z}[X]}{(X^d+1,p)}$ with $p < q$.

Notation

- We denote $R_q = \frac{\mathbb{Z}[X]}{(X^d+1,q)}$ and $R_p = \frac{\mathbb{Z}[X]}{(X^d+1,p)}$ with $p < q$.
- We denote by $\mathscr{D}_\sigma$ the discrete gaussian distribution with standard deviation σ and by $\mathscr{U}(R_q)$ the uniform distribution over R_q.

Notation

- We denote $R_q = \frac{\mathbb{Z}[X]}{(X^d+1,q)}$ and $R_p = \frac{\mathbb{Z}[X]}{(X^d+1,p)}$ with $p < q$.
- We denote by $\mathscr{D}_\sigma$ the discrete gaussian distribution with standard deviation σ and by $\mathscr{U}(R_q)$ the uniform distribution over R_q.
- $\mathscr{S}_B := \{a \in R\,;\, \|a\|_\infty \leq B\}$.

The NTRU Problem

The NTRU Problem

(Search) $\mathsf{NTRU}_{q,d,\sigma}$

Sample $f, g \leftarrow \mathscr{D}_\sigma^2$ with rejection if f not invertible in R_q and set $h = g/f \mod q$. The search $\mathsf{NTRU}_{q,d,\sigma}$ problem is, given h, to recover any rotation $(X^i f, X^i g)$ of the pair (f, g).

The NTRU Problem

(Search) $\mathsf{NTRU}_{q,d,\sigma}$

Sample $f, g \leftarrow \mathscr{D}_\sigma^2$ with rejection if f not invertible in R_q and set $h = g/f \mod q$. The search $\mathsf{NTRU}_{q,d,\sigma}$ problem is, given h, to recover any rotation $(X^i f, X^i g)$ of the pair (f, g).

Denote by H, F, and G the circulant matrices corresponding to h, f, and g.

The NTRU Problem

(Search) $\mathrm{NTRU}_{q,d,\sigma}$

Sample $f, g \leftarrow \mathscr{D}_\sigma^2$ with rejection if f not invertible in R_q and set $h = g/f \mod q$. The search $\mathrm{NTRU}_{q,d,\sigma}$ problem is, given h, to recover any rotation $(X^i f, X^i g)$ of the pair (f, g).

Denote by H, F, and G the circulant matrices corresponding to h, f, and g.

NTRU Lattice

$$\mathscr{L}^{H,q} := \begin{pmatrix} qI_d & H \\ 0 & I_d \end{pmatrix} \cdot \mathbb{Z}^{2d}.$$

Dense Sublattice

$$\mathscr{L}^{GF} := \begin{pmatrix} G \\ F \end{pmatrix} \cdot \mathbb{Z}^d \subset \mathscr{L}^{H,q}.$$

Why NTRU?

Why NTRU?

- More compact/faster primitives utilizing ZKPs

Why NTRU?

- More compact/faster primitives utilizing ZKPs
- Faster KEMs as compared to RLWE-based schemes
- Very receptive to NTT speedups
- Long standing problem

NTRU Hardness

TABLE NTRU Cryptanalysis

1998	NTRU first appears, Hoffstein, Pipher, Silverman [HPS98]
2016	'Subfield lattice attacks' for overstretched params [ABD16, CJL16]
2017	Asymptotic 'fatigue point' upper bound, Kirchner and Fouque [KF17]
2021	Lower fatigue point and concrete confirmation in ternary case [DvW21]

Overstretched NTRU

Overstretched NTRU

The best known attacks on NTRU are based on the LLL lattice reduction algorithm, which searches for the unusually short vector vector $\begin{pmatrix} g \\ f \end{pmatrix}$ in $\mathscr{L}^{H,q}$. However...

Overstretched NTRU

The best known attacks on NTRU are based on the LLL lattice reduction algorithm, which searches for the unusually short vector vector $\begin{pmatrix} g \\ f \end{pmatrix}$ in $\mathscr{L}^{H,q}$. However...

... when q becomes very large (with respect to d) this problem becomes easy.

Overstretched NTRU

The best known attacks on NTRU are based on the LLL lattice reduction algorithm, which searches for the unusually short vector vector $\begin{pmatrix} g \\ f \end{pmatrix}$ in $\mathscr{L}^{H,q}$. However...

... when q becomes very large (with respect to d) this problem becomes easy.

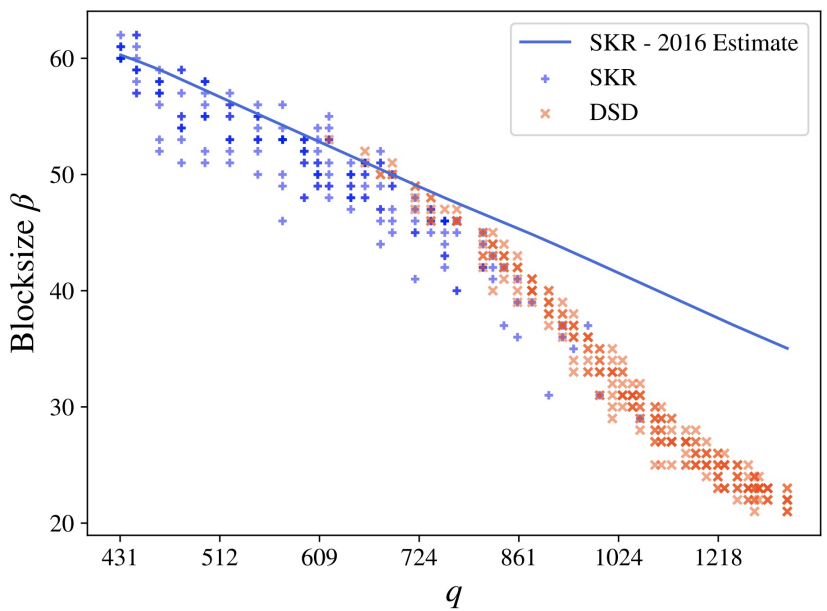

Figure: Progressive BKZ with $d = 127$, $\sigma^2 = 2/3$ [DvW21].

Concrete Fatigue Point

Concrete fatigue point ($\sigma^2 = 2/3$) [DvW21]

Fixing d and σ, the point at which $\Pr(DSD) > \Pr(SKR)$ is determined (experimentally) as $q = 0.004 \cdot d^{2.484}$.

Concrete Fatigue Point

Concrete fatigue point ($\sigma^2 = 2/3$) [DvW21]

Fixing d and σ, the point at which $\Pr(DSD) > \Pr(SKR)$ is determined (experimentally) as $q = 0.004 \cdot d^{2.484}$.

What is the concrete hardness of NTRU for

$$\sqrt{2/3} < \sigma < \sqrt{q}\,?$$

Concrete Fatigue Point

Concrete fatigue point ($\sigma^2 = 2/3$) [DvW21]

Fixing d and σ, the point at which $\Pr(DSD) > \Pr(SKR)$ is determined (experimentally) as $q = 0.004 \cdot d^{2.484}$.

What is the concrete hardness of NTRU for

$$\sqrt{2/3} < \sigma < \sqrt{q}\,?$$

Knowledge of such behaviour has proven crucial for fine-tuning RSIS and RLWE-based NIST-standardised scheme parameters e.g. Crystals Dilithium, which uses non-ternary secrets.

Concrete Fatigue Point - This Work

Concrete Fatigue Point - This Work

Concrete fatigue point (general σ) - This work

Fixing d and σ, we determine the point at which $\Pr(DSD) > \Pr(SKR)$ is determined (experimentally) as

$$q = 0.0058 \cdot \sigma^2 \cdot d^{2.484}.$$

Crucially, the fatigue point increases *quadratically* in σ.

Concrete Fatigue Point - This Work

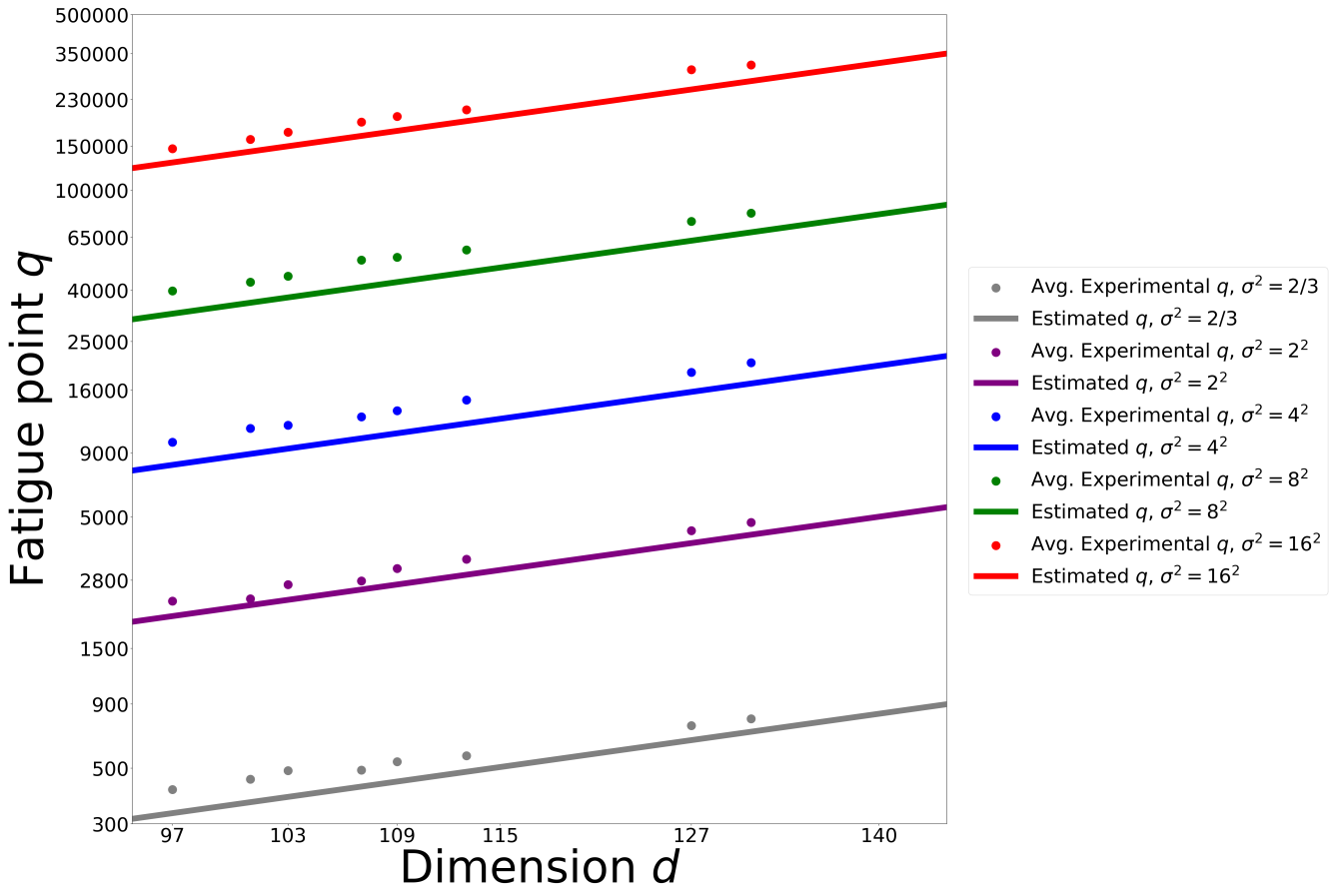

Figure: Experimental fatigue point values for a range of σ, calculated using BKZ with 8 tours on matrix NTRU instances. The straight-colored lines show the estimated values using the (modified) estimator from [DvW21].

Concrete Fatigue Point - This Work

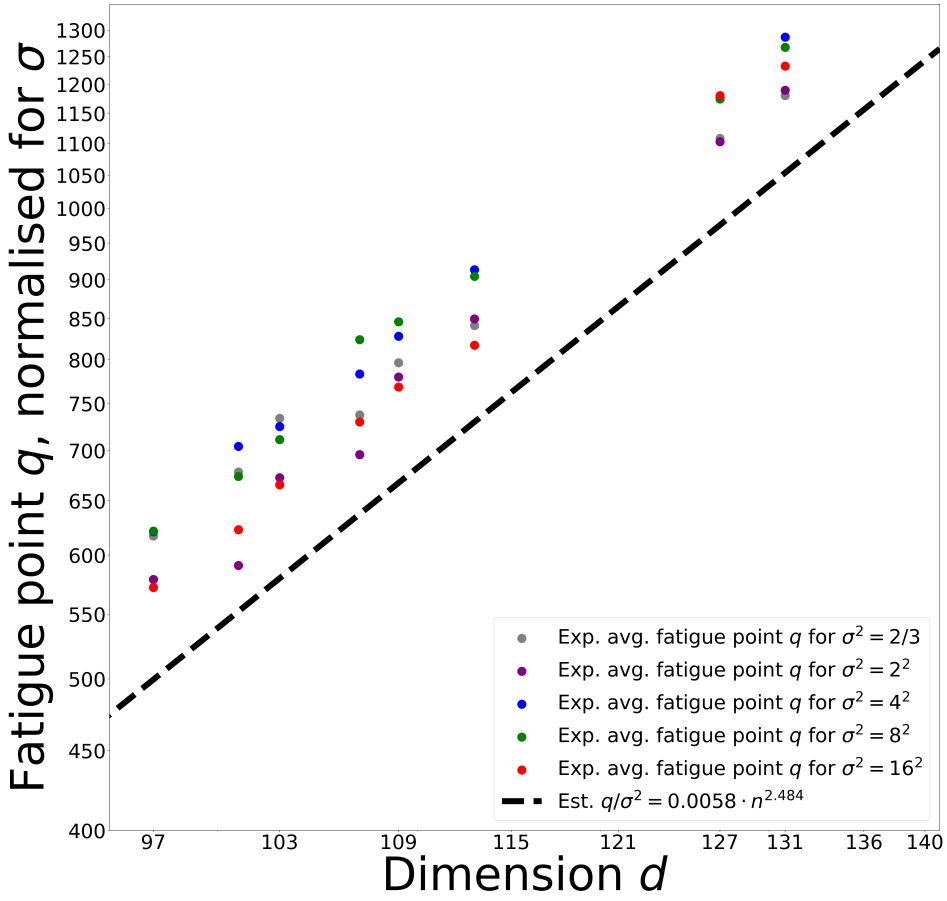

Figure: Experimental values for q/σ^2 illustrate that the fatigue point, when adjusted for σ, is modeled by $q/\sigma^2 = 0.0058 \cdot d^{2.484}$.

A New E-Voting Scheme

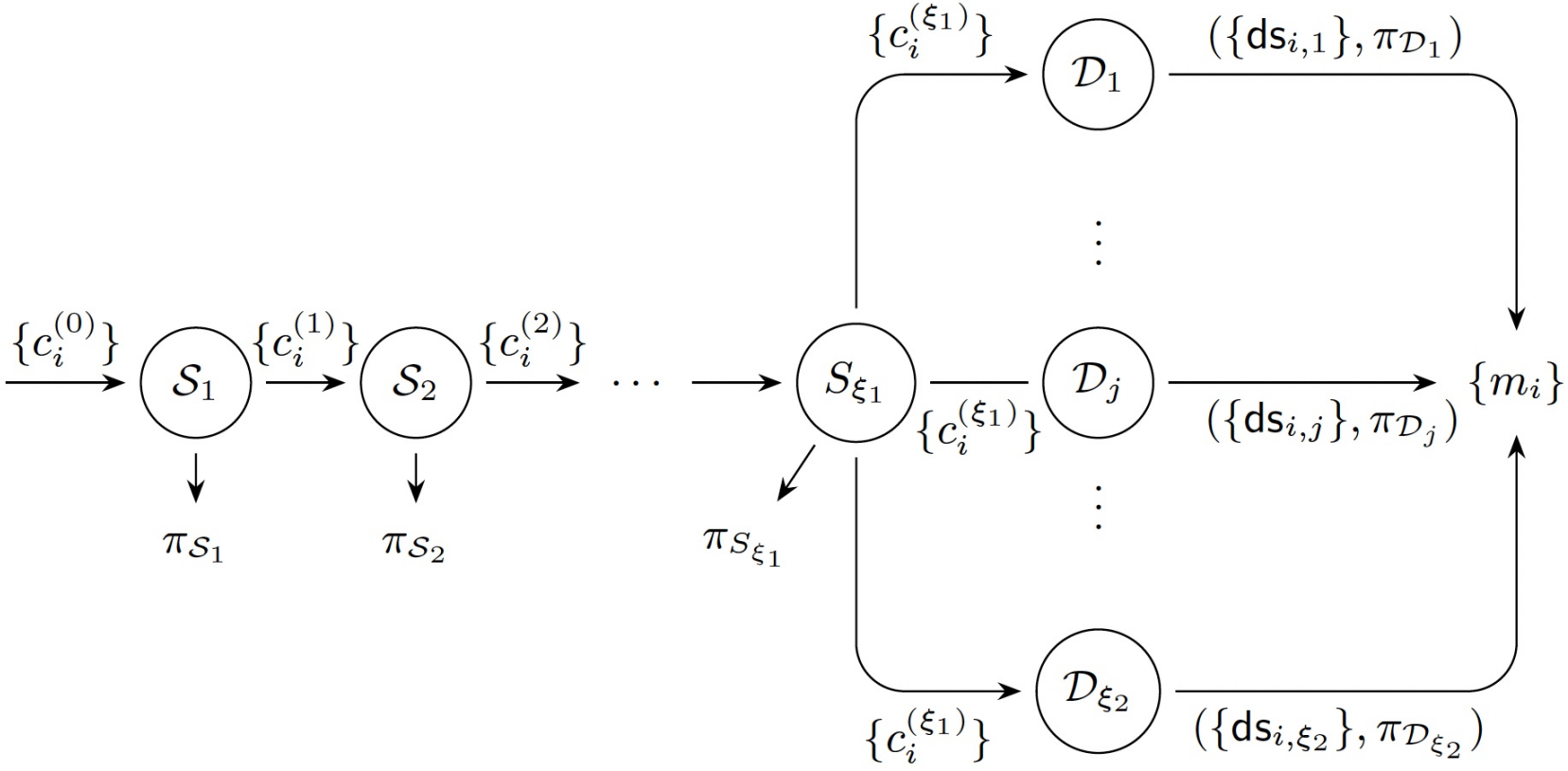

Figure: Voting protocol with sets of shuffle servers $\mathscr{S}_i$ and decryption servers $\mathscr{D}_j$ [ABGS23].

A New E-Voting Scheme - Building Blocks

A New E-Voting Scheme - Building Blocks

Shuffle

- Following framework of [ABGS22], adapted for NTRU-based PKE.

A New E-Voting Scheme - Building Blocks

Shuffle

- Following framework of [ABGS22], adapted for NTRU-based PKE.

Distributed decryption

- Based on NTRUEncrypt - Steinfeld & Stehlé, 2013. Variant using perfect correctness and security relying both on RLWE *and* NTRU.

A New E-Voting Scheme - Building Blocks

Shuffle

- Following framework of [ABGS22], adapted for NTRU-based PKE.

Distributed decryption

- Based on NTRUEncrypt - Steinfeld & Stehlé, 2013. Variant using perfect correctness and security relying both on RLWE *and* NTRU.

Zero-knowledge proofs

- Required for verifiability of shuffle and distributed decryption.

Distributed Decryption - NTRU

Distributed Decryption - NTRU

KeyGen(λ)

- $h := g/f$, with $f, g \leftarrow \mathscr{D}_\sigma$ and $f \equiv 1 \mod p$.
- For $i \in [N-1]$, $\mathsf{sk}_i \leftarrow \mathscr{U}(R_q)$ and $\mathsf{sk}_N := f - \sum_{i \in [N-1]} \mathsf{sk}_i$.
- $\mathsf{pk} = h$, $\mathsf{sk} = (g, f)$.

Distributed Decryption - NTRU

KeyGen(λ)

- $h := g/f$, with $f, g \leftarrow \mathscr{D}_\sigma$ and $f \equiv 1 \mod p$.
- For $i \in [N-1]$, $\mathsf{sk}_i \leftarrow \mathscr{U}(R_q)$ and $\mathsf{sk}_N := f - \sum_{i \in [N-1]} \mathsf{sk}_i$.
- $\mathsf{pk} = h$, $\mathsf{sk} = (g, f)$.

Enc(m)

- $c = p(hs + e) + m$, where $s, e \leftarrow \mathscr{S}_\nu$ and p is a small prime.

Distributed Decryption - NTRU

KeyGen(λ)

- $h := g/f$, with $f, g \leftarrow \mathscr{D}_\sigma$ and $f \equiv 1 \mod p$.
- For $i \in [N-1]$, $\mathsf{sk}_i \leftarrow \mathscr{U}(R_q)$ and $\mathsf{sk}_N := f - \sum_{i \in [N-1]} \mathsf{sk}_i$.
- $\mathsf{pk} = h$, $\mathsf{sk} = (g, f)$.

Enc(m)

- $c = p(hs + e) + m$, where $s, e \leftarrow \mathscr{S}_\nu$ and p is a small prime.

DistDec(c, $\mathsf{sk_i}$)

- $\mathsf{ds}_i := \mathsf{sk}_i \cdot c + pE_i$, where $E_i \leftarrow \mathscr{S}_{2^{40}}$

Distributed Decryption - NTRU

KeyGen(λ)

- $h := g/f$, with $f, g \leftarrow \mathscr{D}_\sigma$ and $f \equiv 1 \mod p$.
- For $i \in [N-1]$, $\mathsf{sk}_i \leftarrow \mathscr{U}(R_q)$ and $\mathsf{sk}_N := f - \sum_{i \in [N-1]} \mathsf{sk}_i$.
- $\mathsf{pk} = h$, $\mathsf{sk} = (g, f)$.

Enc(m)

- $c = p(hs + e) + m$, where $s, e \leftarrow \mathscr{S}_\nu$ and p is a small prime.

DistDec(c, $\mathsf{sk_i}$)

- $\mathsf{ds}_i := \mathsf{sk}_i \cdot c + pE_i$, where $E_i \leftarrow \mathscr{S}_{2^{40}}$

Comb($\{\mathsf{ds}_i\}$)

- $$v = \left(\sum_{i \in [N]} \mathsf{ds}_i \mod q\right) \mod p$$
  $$= \left(f \cdot c + p \sum_{i \in [N]} E_i \mod q\right) \mod p$$
  $$= \left(p(gs + fe) + fm + p \sum_{i \in [N]} E_i \mod q\right) \mod p$$
  $$= m$$

Distributed Decryption - NTRU

KeyGen(λ)

- $h := g/f$, with $f, g \leftarrow \mathscr{D}_\sigma$ and $f \equiv 1 \mod p$.
- For $i \in [N-1]$, $\mathsf{sk}_i \leftarrow \mathscr{U}(R_q)$ and $\mathsf{sk}_N := f - \sum_{i \in [N-1]} \mathsf{sk}_i$.
- $\mathsf{pk} = h$, $\mathsf{sk} = (g, f)$.

Enc(m)

- $c = p(hs + e) + m$, where $s, e \leftarrow \mathscr{S}_\nu$ and p is a small prime.

DistDec($\mathsf{c}, \mathsf{sk_i}$)

- $\mathsf{ds}_i := \mathsf{sk}_i \cdot c + pE_i$, where $E_i \leftarrow \mathscr{S}_{2^{40}}$

Comb($\{\mathsf{ds}_i\}$)

- $$v = \left(\sum_{i \in [N]} \mathsf{ds}_i \mod q\right) \mod p$$
  $$= \left(f \cdot c + p \sum_{i \in [N]} E_i \mod q\right) \mod p$$
  $$= \left(p(gs + fe) + fm + p \sum_{i \in [N]} E_i \mod q\right) \mod p$$
  $$= m \ldots \text{IF} \ldots$$

Distributed Decryption - Correctness Condition

$$\left\| p(gs + fe) + fm + p \sum_{i \in [N]} E_i \right\|_\infty < q/2,$$

Distributed Decryption - Correctness Condition

$$\left\| p(gs + fe) + fm + p \sum_{i \in [N]} E_i \right\|_\infty < q/2,$$

$\Longleftarrow$

$$p \cdot d \cdot \sigma \cdot (2v + 1/2)(1 + 2^{40}) < q/2,$$

Choosing Parameters

Choosing Parameters

Recall distributed decryption requires that

$$p \cdot d \cdot \sigma \cdot (2\nu + 1/2)(1 + 2^{40}) < q/2. \tag{1}$$

Choosing Parameters

Recall distributed decryption requires that

$$p \cdot d \cdot \sigma \cdot (2\nu + 1/2)(1 + 2^{40}) < q/2. \tag{1}$$

Now, we can use the estimator of [DvW21] to select parameters so that

- (1) is satisfied and
- $\mathsf{NTRU}_{q,d,\sigma}$ is hard (128 bits of security).

Voting Scheme Efficiency

Voting Scheme Efficiency

Scheme	Ciphertexts	Shuffle	Dist. Dec.	Total
[5] [KB]	80	370	157	2188
Our [KB]	15	130	85	875
[5] [ms]	0.74	261	138	1182
Our [ms]	0.20	62	328	576

Figure: Per vote comparison to [ABGS23] of ciphertexts, shuffle proofs, decryption proofs, and overall with 4 servers. Shuffles are sequential, while decryption is run in parallel.

Summary Of Results

Summary Of Results

- Experimental generalisation of NTRU concrete fatigue point.
- Recompute secure parameters for selected schemes.
- New voting protocol based on NTRU PKE.
- Compute concrete efficiency of this scheme and provide efficient C-implementation.

Future Directions

Future Directions

- *Return codes*. To extend our scheme and ensure voter verifiability, we need to add return codes to our scheme. This can be done by extending the work of [**?**] from BGV to NTRU. This also includes verifiable encryption.
- A more efficient noise drowning technique for distributed decryption.
- Using more efficient zero-knowledge techniques like SNARKS for better amortization.

Future Directions

Thank you.


Martin R. Albrecht, Shi Bai, and Léo Ducas.
A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes.
In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Heidelberg, August 2016.


Diego F. Aranha, Carsten Baum, Kristian Gjøsteen, and Tjerand Silde.
Verifiable mix-nets and distributed decryption for voting from lattice-based assumptions.
Cryptology ePrint Archive, Report 2022/422, 2022. `https://eprint.iacr.org/2022/422`.


Diego F. Aranha, Carsten Baum, Kristian Gjøsteen, and Tjerand Silde.
Verifiable mix-nets and distributed decryption for voting from lattice-based assumptions.

In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, pages 1467–1481. ACM, 2023.


Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee.
An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low level encoding of zero.
Cryptology ePrint Archive, Paper 2016/139, 2016.
https://eprint.iacr.org/2016/139.


Léo Ducas and Wessel P. J. van Woerden.
NTRU fatigue: How stretched is overstretched?
LNCS, pages 3–32. Springer, Heidelberg, 2021.


Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman.
NTRU: A ring-based public key cryptosystem.
In *Third Algorithmic Number Theory Symposium (ANTS)*, volume 1423 of *LNCS*, pages 267–288. Springer, Heidelberg, June 1998.


Paul Kirchner and Pierre-Alain Fouque.
Revisiting lattice attacks on overstretched NTRU parameters.
In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 3–26. Springer, Heidelberg, April / May 2017.