University of BRISTOL

28th May 2018

Compact, Scalable, and Efficient Discrete Gaussian Samplers for Lattice-Based Cryptography

James Howe[†], Ayesha Khalid[‡], Ciara Rafferty[‡], Francesco Regazzoni[*], and Maire O'Neill[‡].

[†]University of Bristol, UK; [‡]Queen's University, Belfast; and [*]Università della Svizzera Italiana, Switzerland.

Outline

- Motivation

University of BRISTOL

Outline

- Motivation
- Introduction to post-quantum cryptography

Outline

- Motivation
- Introduction to post-quantum cryptography
- Lattice-based cryptography and the Learning with Errors problem

Outline

- Motivation
- Introduction to post-quantum cryptography
- Lattice-based cryptography and the Learning with Errors problem
- Gaussian samplers

Outline

- Motivation
- Introduction to post-quantum cryptography
- Lattice-based cryptography and the Learning with Errors problem
- Gaussian samplers
- Mathematical optimizations

University of BRISTOL

28th May 2018

Outline

- Motivation
- Introduction to post-quantum cryptography
- Lattice-based cryptography and the Learning with Errors problem
- Gaussian samplers
- Mathematical optimizations
- Hardware design

Outline

- Motivation
- Introduction to post-quantum cryptography
- Lattice-based cryptography and the Learning with Errors problem
- Gaussian samplers
- Mathematical optimizations
- Hardware design
- Results and performance analysis

Motivation

- What happens when quantum computers become a reality 10-15 years from now?
- Commonly used public-key cryptographic algorithms (based on integer factorization and discrete log problem) such as:

RSA, DSA, Diffie-Hellman Key Exchange, ECC, ECDSA

will be vulnerable to Shor's algorithm and will no longer be secure.

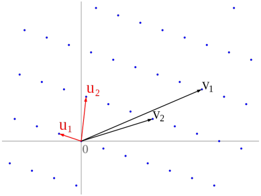
- "Worse than Y2K: quantum computing and the end of privacy" – *Forbes*, 2018.
- "The quantum clock is ticking on encryption - and your data is under threat" – *Wired*, 2016.
- "Unbreakable: The race to protect our secrets from quantum hacks" – *New Scientist*, 2018.

Post-Quantum Cryptography

- NIST have started a post-quantum standardisation “competition”.
  - Similar to previous AES and SHA-3 standardisations.
- ETSI researching industrial requirements for quantum-safe real-world deployments.
- Why focus on lattice-based cryptography?
  - More versatile than code-based, isogeny-based, multivariate-quadratic, and hash-based schemes.
  - Can be used for encryption, signatures, FHE, IBE, ABE etc...
  - Theoretical foundations are well-studied.

Lattice-Based Cryptography

- Lattice-based cryptography is important in its own right.
  - Benefits from simple mathematical operations such as integer multiplication, addition, and modular reduction.
- Lattice-based cryptography is flourishing:
  - 40% lattice-based NIST PQC submissions.
  - NewHope key exchange created.
  - Ring-LWE encryption and BLISS signatures outperform RSA and ECC in s/w and h/w.



- Lattice-based cryptography is already being considered:
  - VPN strongSwan supports post-quantum mode.
  - NewHope awarded Internet Defense Prize Winner 2016.
  - Google experimenting with NewHope key exchange.

The Learning With Errors Problem

- There is a secret vector $\mathbf{s} \leftarrow \mathbb{Z}_q^n$.
- An oracle (who knows $\mathbf{s}$) generates a uniform matrix $\mathbf{A}$ and noise vector $\mathbf{e}$ distributed normally with standard deviation αq.
- The oracle outputs:

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \mod q).$$

- The distribution of $\mathbf{A}$ is uniformly random, $\mathbf{b}$ is pseudo-random.

The Learning With Errors Problem

- There is a secret vector $\mathbf{s} \leftarrow \mathbb{Z}_q^n$.
- An oracle (who knows $\mathbf{s}$) generates a uniform matrix $\mathbf{A}$ and noise vector $\mathbf{e}$ distributed normally with standard deviation αq.
- The oracle outputs:

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \mod q).$$

- The distribution of $\mathbf{A}$ is uniformly random, $\mathbf{b}$ is pseudo-random.
- Can you find $\mathbf{s}$, given access to $(\mathbf{A}, \mathbf{b})$?

The Learning With Errors Problem

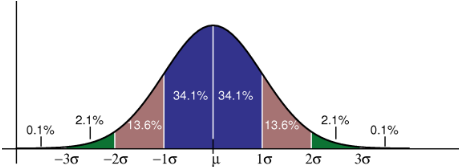
- There is a secret vector $\mathbf{s} \leftarrow \mathbb{Z}_q^n$.
- An oracle (who knows $\mathbf{s}$) generates a uniform matrix $\mathbf{A}$ and noise vector $\mathbf{e}$ distributed normally with standard deviation αq.
- The oracle outputs:

$$(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \mod q).$$

- The distribution of $\mathbf{A}$ is uniformly random, $\mathbf{b}$ is pseudo-random.
- Can you find $\mathbf{s}$, given access to $(\mathbf{A}, \mathbf{b})$?
- Can you distinguish $(\mathbf{A}, \mathbf{b})$ from a uniformly random $(\mathbf{A}, \mathbf{b}')$?

Generating the Error in LWE

- Error adds noise to computations on secret data; computationally hard.
- Look-up table methods: CDT sampler.
- Arithmetic-based methods: discrete Ziggurat sampler.
- Hybrid table / arithmetic methods: Bernoulli and Knuth-Yao samplers.
- Standard deviations depend on cryptographic schemes and parameters:



Generating the Error in LWE

- Error adds noise to computations on secret data; computationally hard.
- CDT sampler previously shown to be the most efficient in hardware.
- Standard deviations depend on cryptographic schemes and parameters:

Scheme	Std. Dev.	Table Size	Clocks
NewHope (KEX)	2.83	3.5 kb	5
LWE (Enc)	3.33	4.0 kb	5
Ring-LWE (Enc)	4.52	5.5 kb	6

Generating the Error in LWE

- Error adds noise to computations on secret data; computationally hard.
- CDT sampler previously shown to be the most efficient in hardware.
- Standard deviations depend on cryptographic schemes and parameters:

Scheme	Std. Dev.	Table Size	Clocks
NewHope (KEX)	2.83	3.5 kb	5
LWE (Enc)	3.33	4.0 kb	5
Ring-LWE (Enc)	4.52	5.5 kb	6
Falcon (Sign)	172	208 kb	11
BLISS (Sign)	215	260 kb	11

Generating the Error in LWE

- Falcon and BLISS samplers require table sizes ∼50x bigger than the smaller encryption schemes.
- Dilithium-G samplers then require ∼100x more than these.
- Table sizes are infeasible, making the sampler's performance inefficient.
- We need optimisation methods to ensure real-world applicability.

Scheme	Std. Dev.	Table Size	Clocks
Falcon (Sign)	172	208 kb	11
BLISS (Sign)	215	260 kb	11
Dilithium-G-I (Sign)	19200	23 Mb	18
Dilithium-G-II (Sign)	17900	22 Mb	18
Dilithium-G-III (Sign)	12400	15 Mb	17

Scalability via Gaussian Convolutions

- We can use convolutions to minimise these large standard deviations.
- Generate samples of smaller standard deviations and to form a sample of a larger target standard deviation as:

$$x := x_1 + kx_2.$$

- We use lemmas to calculate constant(s) and smaller standard deviations.

Scalability via Gaussian Convolutions

- We can use convolutions to minimise these large standard deviations.
- Generate samples of smaller standard deviations and to form a sample of a larger target standard deviation as:

$$x := x_1 + kx_2.$$

- We use lemmas to calculate constant(s) and smaller standard deviations.
- The process in the above equation can be repeated numerous times, further shrinking the standard deviation used in the sampler:

$$x := (x_1 + k'x_2) + k * (x_3 + k'x_4).$$

Scalability via Gaussian Convolutions

- We can use convolutions to minimise these large standard deviations.
- Generate samples of smaller standard deviations and to form a sample of a larger target standard deviation as:

$$x := x_1 + kx_2.$$

- We use lemmas to calculate constant(s) and smaller standard deviations.
- The process in the above equation can be repeated numerous times, further shrinking the standard deviation used in the sampler:

$$x := (x_1 + k'x_2) + k * (x_3 + k'x_4).$$

Scalability via Gaussian Convolutions

- We can use convolutions to minimise these large standard deviations.
- Generate samples of smaller standard deviations and to form a sample of a larger target standard deviation as:

  Level 1: $$x := x_1 + kx_2.$$

- We use lemmas to calculate constant(s) and smaller standard deviations.
- The process in the above equation can be used recursively, further shrinking the standard deviation used in the sampler:

  Level 2: $$x := (x_1 + k'x_2) + k * (x_3 + k'x_4).$$

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

				Convolution Levels		
Type	**Cryptographic Scheme**	**Security (bits)**	**Standard Dev.** (σ)	**Level 1** $(\mathbf{k}, \sigma')$	**Level 2** $(\mathbf{k}', \sigma'')$	**Level 3** $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

Type	Cryptographic Scheme	Security (bits)	Standard Dev. (σ)	Convolution Levels: Level 1 $(\mathbf{k}, \sigma')$	Level 2 $(\mathbf{k}', \sigma'')$	Level 3 $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

Type	Cryptographic Scheme	Security (bits)	Standard Dev. (σ)	Convolution Levels: Level 1 $(\mathbf{k}, \sigma')$	Level 2 $(\mathbf{k}', \sigma'')$	Level 3 $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

Type	Cryptographic Scheme	Security (bits)	Standard Dev. (σ)	Convolution Levels: Level 1 $(\mathbf{k}, \sigma')$	Level 2 $(\mathbf{k}', \sigma'')$	Level 3 $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

Type	Cryptographic Scheme	Security (bits)	Standard Dev. (σ)	Convolution Levels: Level 1 $(\mathbf{k}, \sigma')$	Level 2 $(\mathbf{k}', \sigma'')$	Level 3 $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameters for Gaussian Convolutions

Table: Parameter sets for proposed discrete Gaussian hardware architectures.

Type	Cryptographic Scheme	Security (bits)	Standard Dev. (σ)	Convolution Levels: Level 1 $(\mathbf{k}, \sigma')$	Level 2 $(\mathbf{k}', \sigma'')$	Level 3 $(\mathbf{k}'', \sigma''')$
KEX	New Hope	200	2.83	(-,-)	(-,-)	(-,-)
	BCNS	128	3.19	(1,2.26)	(-,-)	(-,-)
Enc.	Ring-LWE	128	4.52	(1,3.19)	(-,-)	(-,-)
Sign.	BLISS-I	128	215	(11,19.53)	(3, 6.18)	(-,-)
	BLISS-II	128	107	(8, 13.27)	(2, 5.94)	(-,-)
	BLISS-III	160	250	(12,20.76)	(3, 6.57)	(-,-)
	BLISS-IV	192	271	(13,20.78)	(3, 6.57)	(-,-)
	Dilithium-G-I	91	19200	(89,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-II	129	17900	(83,215.69)	(11,19.53)	(3,6.18)
	Dilithium-G-III	158	12400	(57,215.69)	(11,19.53)	(3,6.18)

Parameter Selection for CDT Sampling

- CDT sampling stores CDF values in a lookup and searches the table via binary search to produce one Gaussian sample.
- We can do this in constant-time by fixing the number of table entries.
- What is the maximum standard deviation we can do...

		32 Entry Table		64 Entry Table	
Level	**No. Samples**	$k/k'/k''$	**Max**(σ)	$k/k'/k''$	**Max**(σ)
L0	1	-/-/-	3.39	-/-/-	6.79
L1	2	1/-/-	4.80	3/-/-	21.30
L2	4	1/2/-	10.50	3/13/-	280
L3	8	1/2/5	54.75	3/13/163	45660

Parameter Selection for CDT Sampling

- CDT sampling stores CDF values in a lookup and searches the table via binary search to produce one Gaussian sample.
- We can do this in constant-time by fixing the number of table entries.
- What is the maximum standard deviation we can do...

		5 clock cycles		6 clock cycles	
Level	**No. Samples**	$k/k'/k''$	**Max**(σ)	$k/k'/k''$	**Max**(σ)
L0	1	-/-/-	3.39	-/-/-	6.79
L1	2	1/-/-	4.80	3/-/-	21.30
L2	4	1/2/-	10.50	3/13/-	280
L3	8	1/2/5	54.75	3/13/163	45660

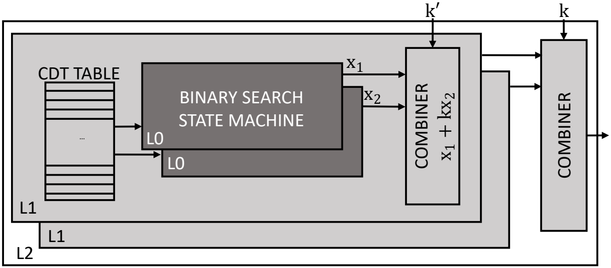
Parameter Selection for CDT Sampling

- CDT sampling stores CDF values in a lookup and searches the table via binary search to produce one Gaussian sample.
- We can do this in constant-time by fixing the number of table entries.
- What is the maximum standard deviation we can do...

		5 clock cycles		6 clock cycles	
Level	**No. Samples**	$k/k'/k''$	**Max**(σ)	$k/k'/k''$	**Max**(σ)
L0	1	-/-/-	3.39	-/-/-	6.79
L1	2	1/-/-	4.80	3/-/-	21.30
L2	4	1/2/-	10.50	3/13/-	280
L3	8	1/2/5	54.75	3/13/163	45660

Hardware Design for Scalable Gaussian Samplers

- CDT sampling stores CDF values in a lookup and searches the table via binary search to produce one Gaussian sample.
- An L2 sampler, e.g. BLISS $(\sigma = 215)$, comprises of two L1 samplers, each made up of a common CDT sampler and two state machines.



Post-place and Route Results

Table Size	Implementation (Convolution Level)	Precision (λ)	LUT/FF/ Slices	BRAM/ DSP	Clock Cycles	Ops/s ($\times 10^6$)	Ops/s/S ($\times 10^6$/S)
32 Entry Table	BCNS L0	80	115/144/44	0/0	6	16.67	0.38
			51/80/28	1/0	6	16.67	0.60
		128	211/272/84	0/0	6	16.67	0.20
			83/144/55	2/0	6	16.67	0.30
	Ring-LWE L1	80	167/294/63	0/0	6	16.67	0.26
			103/166/53	2/0	6	16.67	0.31
		128	306/550/115	0/0	6	16.67	0.14
			177/294/82	4/0	6	16.67	0.20
64 Entry Table	BLISS-I L2	80	269/347/110	0/0	7	14.29	0.13
			268/347/114	4/0	7	14.29	0.13
		128	390/603/169	0/0	7	14.29	0.08
			399/603/174	8/0	7	14.29	0.08
	Dilithium-G L3	80	1057/1195/332	0/2	8	12.50	0.04
			546/683/222	8/2	8	12.50	0.06
		128	1796/2219/599	0/2	8	12.50	0.02
			777/1195/357	16/2	8	12.50	0.04

Post-place and Route Results

- Thank you for listening. Any questions?

Table Size	Implementation (Convolution Level)	Precision (λ)	LUT/FF/ Slices	BRAM/ DSP	Clock Cycles	Ops/s ($\times 10^6$)	Ops/s/S ($\times 10^6$/S)
32 Entry Table	BCNS L0	80	115/144/44	0/0	6	16.67	0.38
			51/80/28	1/0	6	16.67	0.60
		128	211/272/84	0/0	6	16.67	0.20
			83/144/55	2/0	6	16.67	0.30
	Ring-LWE L1	80	167/294/63	0/0	6	16.67	0.26
			103/166/53	2/0	6	16.67	0.31
		128	306/550/115	0/0	6	16.67	0.14
			177/294/82	4/0	6	16.67	0.20
64 Entry Table	BLISS-I L2	80	269/347/110	0/0	7	14.29	0.13
			268/347/114	4/0	7	14.29	0.13
		128	390/603/169	0/0	7	14.29	0.08
			399/603/174	8/0	7	14.29	0.08
	Dilithium-G L3	80	1057/1195/332	0/2	8	12.50	0.04
			546/683/222	8/2	8	12.50	0.06
		128	1796/2219/599	0/2	8	12.50	0.02
			777/1195/357	16/2	8	12.50	0.04

Conclusions

- This research shows mathematical techniques to make Gaussian samplers practical for large parameters.
- This has been demonstrated for a variety of parameter sizes.
- For performance, the largest parameters have seen a 2.25x improvement in throughput, being reduced from 18 clock cycles, to 8, per variable.
- For area consumption, the largest parameters have seen a 550x improvement, with lookup table sizes reduced from 23 Mb, to 41 kb.
- We would now like to see how these samplers can be used in a cryptographic scheme and whether this technique can be applied elsewhere.

Conclusions

- This research shows mathematical techniques to make Gaussian samplers practical for large parameters.
- This has been demonstrated for a variety of parameter sizes.
- For performance, the largest parameters have seen a 2.25x improvement in throughput, being reduced from 18 clock cycles, to 8, per variable.
- For area consumption, the largest parameters have seen a 550x improvement, with lookup table sizes reduced from 23 Mb, to 41 kb.
- We would now like to see how these samplers can be used in a cryptographic scheme and whether this technique can be applied elsewhere.

Thank you for listening, any questions?