Preparing for post-quantum and hybrid cryptography on the Internet

Douglas Stebila McMaster University

Concordia • March 15, 2017

Acknowledgements

Collaborators

- Nina Bindel
- Joppe Bos
- Craig Costello and Michael Naehrig
- Léo Ducas
- Udyani Herath and Matthew McKague
- Ilya Mironov and Ananth Raghunathan
- Michele Mosca and John Schanck
- Valeria Nikolaenko

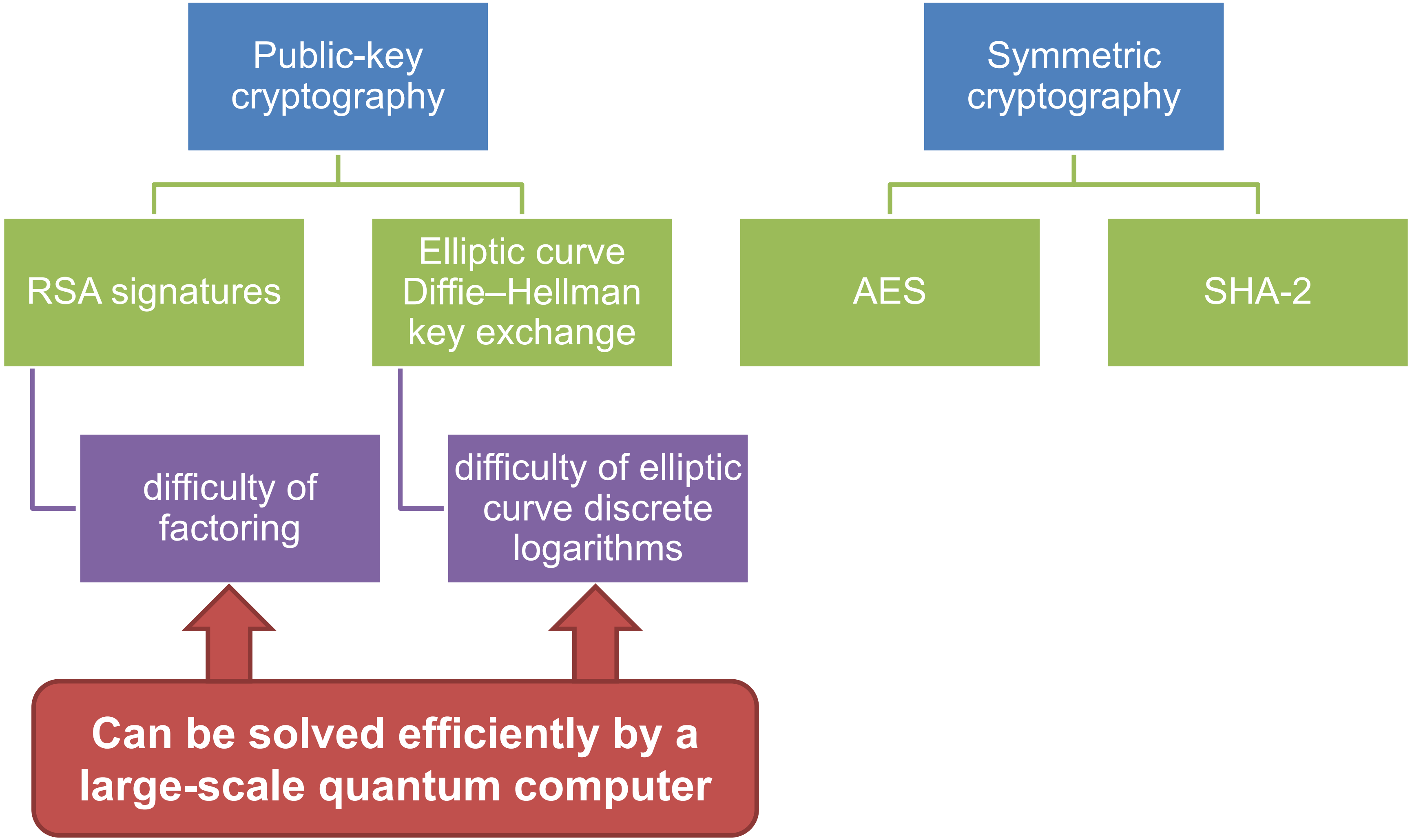
Support

- Australian Research Council (ARC)
- Natural Sciences and Engineering Research Council of Canada (NSERC)
- Queensland University of Technology
- Tutte Institute for Mathematics and Computing

Motivation

Contemporary cryptography

TLS-ECDHE-RSA-AES128-GCM-SHA256



When will a large-scale quantum computer be built?

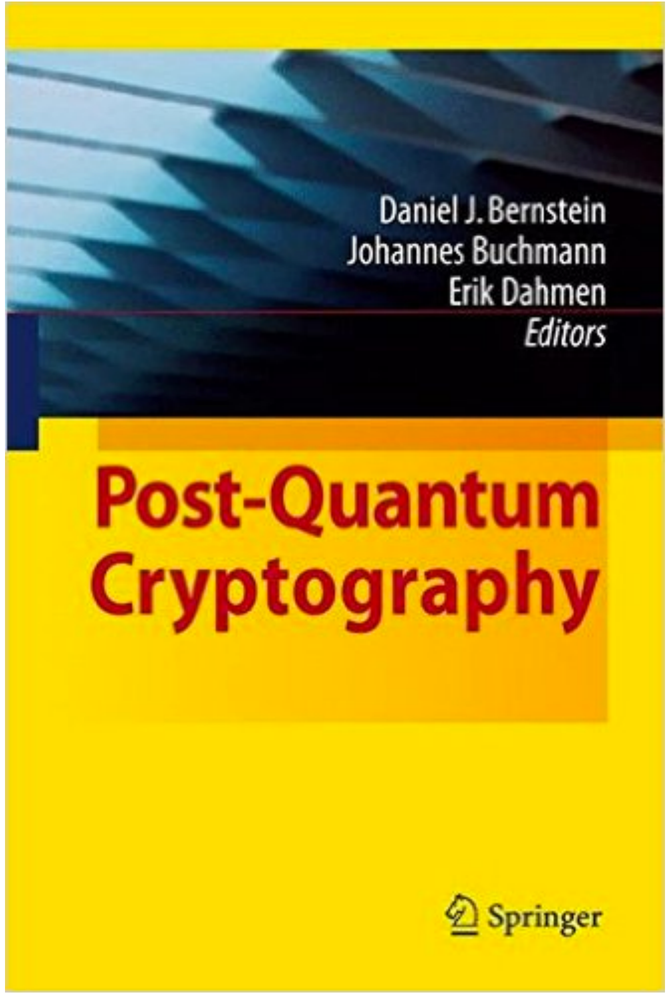
"I estimate a 1/7 chance of breaking RSA-2048 by 2026 and a 1/2 chance by 2031."

— Michele Mosca, November 2015
https://eprint.iacr.org/2015/1075

Post-quantum cryptography in academia

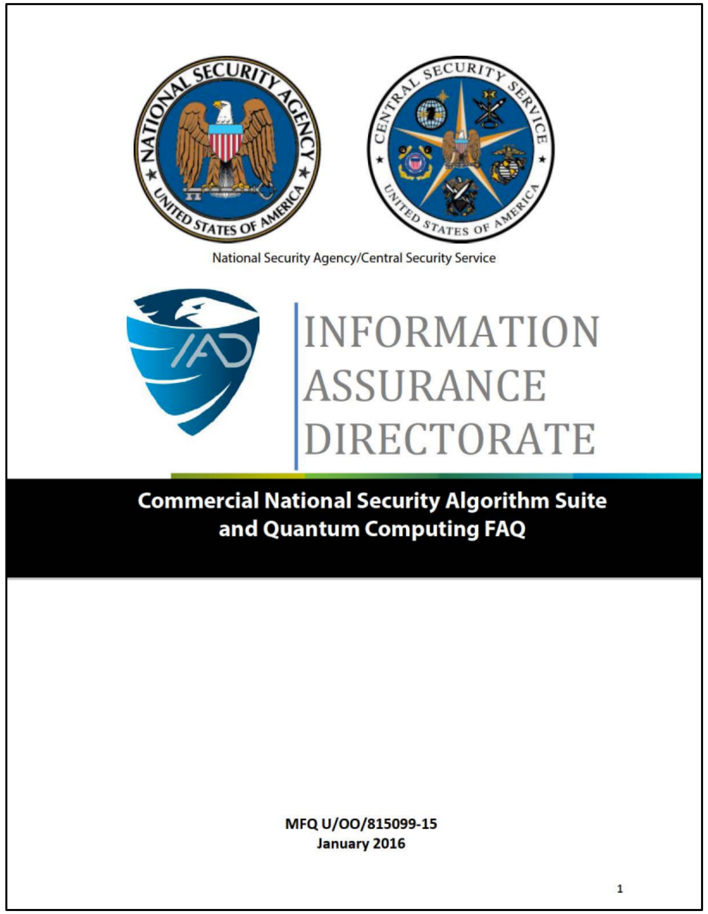
<u>Conference series</u>

- PQCrypto 2006
- PQCrypto 2008
- PQCrypto 2010
- PQCrypto 2011
- PQCrypto 2013
- PQCrypto 2014
- PQCrypto 2016



2009

Post-quantum cryptography in government


National Security Agency/Central Security Service

INFORMATION ASSURANCE DIRECTORATE

Commercial National Security Algorithm Suite and Quantum Computing FAQ

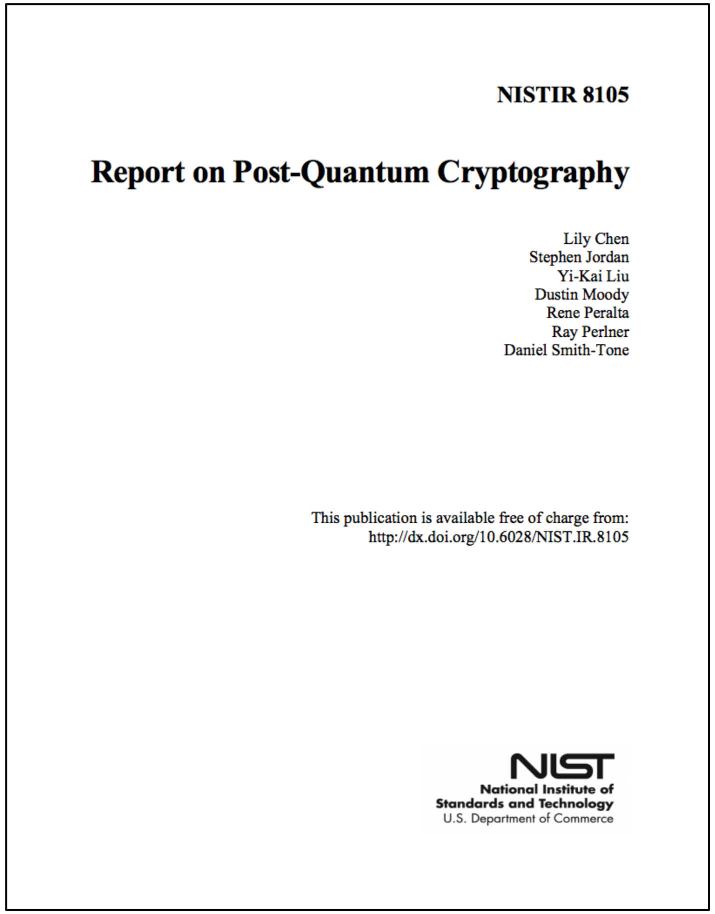
MFQ U/OO/815099-15
January 2016

1

Aug. 2015 (Jan. 2016)

"IAD will initiate a transition to quantum resistant algorithms in the not too distant future."

– NSA Information Assurance Directorate, Aug. 2015


NISTIR 8105

Report on Post-Quantum Cryptography

Lily Chen
Stephen Jordan
Yi-Kai Liu
Dustin Moody
Rene Peralta
Ray Perlner
Daniel Smith-Tone

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.IR.8105

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Apr. 2016

NIST Post-quantum Crypto Project timeline

September, 2016	Feedback on call for proposals
Fall 2016	Formal call for proposals
November 2017	**Deadline for submissions**
Early 2018	Workshop – submitters' presentations
3-5 years	Analysis phase
2 years later	Draft standards ready

http://www.nist.gov/pqcrypto

Post-quantum / quantum-safe crypto

No known exponential quantum speedup

Hash-based

- Merkle signatures
- Sphincs

Code-based

- McEliece

Multivariate

- multivariate quadratic

Lattice-based

- NTRU
- learning with errors
- ring-LWE

Isogenies

- supersingular elliptic curve isogenies

Lots of questions

- Design better post-quantum key exchange and signature schemes
- Improve classical and quantum attacks
- Pick parameter sizes
- Develop fast, secure implementations
- Integrate them into the existing infrastructure

This talk

- Frodo
  - Key exchange protocol from the learning with errors problem
- Open Quantum Safe project
  - A library for comparing post-quantum primitives
  - Framework for easing integration into applications like OpenSSL
- Hybrid key exchange and digital signatures
  - In TLS
  - In X.509v3, S/MIME

Learning with errors problems

Solving systems of linear equations

$\mathbb{Z}_{13}^{7\times 4}$ × **secret** $\mathbb{Z}_{13}^{4\times 1}$ = $\mathbb{Z}_{13}^{7\times 1}$

$$\begin{pmatrix} 4 & 1 & 11 & 10 \\ 5 & 5 & 9 & 5 \\ 3 & 9 & 0 & 10 \\ 1 & 3 & 3 & 2 \\ 12 & 7 & 3 & 4 \\ 6 & 5 & 11 & 4 \\ 3 & 3 & 5 & 0 \end{pmatrix} \times \begin{pmatrix} \blacksquare \\ \blacksquare \\ \blacksquare \\ \blacksquare \end{pmatrix} = \begin{pmatrix} 4 \\ 8 \\ 1 \\ 10 \\ 4 \\ 12 \\ 9 \end{pmatrix}$$

Linear system problem: given **blue**, find **red**

Solving systems of linear equations

$$\underbrace{\begin{pmatrix} 4 & 1 & 11 & 10 \\ 5 & 5 & 9 & 5 \\ 3 & 9 & 0 & 10 \\ 1 & 3 & 3 & 2 \\ 12 & 7 & 3 & 4 \\ 6 & 5 & 11 & 4 \\ 3 & 3 & 5 & 0 \end{pmatrix}}_{\mathbb{Z}_{13}^{7\times 4}} \times \underbrace{\begin{pmatrix} 6 \\ 9 \\ 11 \\ 11 \end{pmatrix}}_{\text{secret } \mathbb{Z}_{13}^{4\times 1}} = \underbrace{\begin{pmatrix} 4 \\ 8 \\ 1 \\ 10 \\ 4 \\ 12 \\ 9 \end{pmatrix}}_{\mathbb{Z}_{13}^{7\times 1}}$$

Easily solved using Gaussian elimination (Linear Algebra 101)

Linear system problem: given blue, find red

Learning with errors problem

random $\mathbb{Z}_{13}^{7\times 4}$ **secret** $\mathbb{Z}_{13}^{4\times 1}$ **small noise** $\mathbb{Z}_{13}^{7\times 1}$ $\mathbb{Z}_{13}^{7\times 1}$

$$\begin{pmatrix} 4 & 1 & 11 & 10 \\ 5 & 5 & 9 & 5 \\ 3 & 9 & 0 & 10 \\ 1 & 3 & 3 & 2 \\ 12 & 7 & 3 & 4 \\ 6 & 5 & 11 & 4 \\ 3 & 3 & 5 & 0 \end{pmatrix} \times \begin{pmatrix} 6 \\ 9 \\ 11 \\ 11 \end{pmatrix} + \begin{pmatrix} 0 \\ -1 \\ 1 \\ 1 \\ 1 \\ 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 4 \\ 7 \\ 2 \\ 11 \\ 5 \\ 12 \\ 8 \end{pmatrix}$$

Learning with errors problem

random $\mathbb{Z}_{13}^{7\times 4}$		secret $\mathbb{Z}_{13}^{4\times 1}$		small noise $\mathbb{Z}_{13}^{7\times 1}$		$\mathbb{Z}_{13}^{7\times 1}$

$$
\begin{pmatrix}
4 & 1 & 11 & 10 \\
5 & 5 & 9 & 5 \\
3 & 9 & 0 & 10 \\
1 & 3 & 3 & 2 \\
12 & 7 & 3 & 4 \\
6 & 5 & 11 & 4 \\
3 & 3 & 5 & 0
\end{pmatrix}
\times
\begin{pmatrix} \\ \\ \\ \end{pmatrix}
+
\begin{pmatrix} \\ \\ \\ \\ \\ \\ \end{pmatrix}
=
\begin{pmatrix} 4 \\ 7 \\ 2 \\ 11 \\ 5 \\ 12 \\ 8 \end{pmatrix}
$$

Computational LWE problem: given **blue**, find **red**

Decision learning with errors problem

random $\mathbb{Z}_{13}^{7\times 4}$ × **secret** $\mathbb{Z}_{13}^{4\times 1}$ + **small noise** $\mathbb{Z}_{13}^{7\times 1}$ = **looks random** $\mathbb{Z}_{13}^{7\times 1}$

4	1	11	10
5	5	9	5
3	9	0	10
1	3	3	2
12	7	3	4
6	5	11	4
3	3	5	0

4
7
2
11
5
12
8

Decision LWE problem: given blue, distinguish green from random

Toy example versus real-world example

$\mathbb{Z}_{13}^{7\times 4}$

4	1	11	10
5	5	9	5
3	9	0	10
1	3	3	2
12	7	3	4
6	5	11	4
3	3	5	0

$\mathbb{Z}_{4093}^{640\times 256}$

256 (columns) × 640 (rows)

2738	3842	3345	2979	…
2896	595	3607		
377	1575			
2760				
…				

640 × 256 × 12 bits = **245 KiB**

Ring learning with errors problem

random

$\mathbb{Z}_{13}^{7\times 4}$

4	1	11	10
10	4	1	11
11	10	4	1
1	11	10	4
4	1	11	10
10	4	1	11
11	10	4	1

Each row is the cyclic shift of the row above

Ring learning with errors problem

random

$\mathbb{Z}_{13}^{7\times 4}$

4	**1**	**11**	**10**
3	**4**	**1**	**11**
2	**3**	**4**	**1**
12	**2**	**3**	**4**
9	**12**	**2**	**3**
10	**9**	**12**	**2**
11	**10**	**9**	**12**

Each row is the cyclic shift of the row above

…

with a special wrapping rule: *x* wraps to –*x* mod 13.

Ring learning with errors problem

random

$\mathbb{Z}_{13}^{7\times 4}$

4	**1**	**11**	**10**

Each row is the cyclic shift of the row above

…

with a special wrapping rule:

x wraps to –*x* mod 13.

So I only need to tell you the first row.

Ring learning with errors problem

$\mathbb{Z}_{13}[x]/\langle x^4 + 1 \rangle$

	$4 + 1x + 11x^2 + 10x^3$	**random**
×	$6 + 9x + 11x^2 + 11x^3$	**secret**
+	$0 - 1x + 1x^2 + 1x^3$	**small noise**
=	$10 + 5x + 10x^2 + 7x^3$	

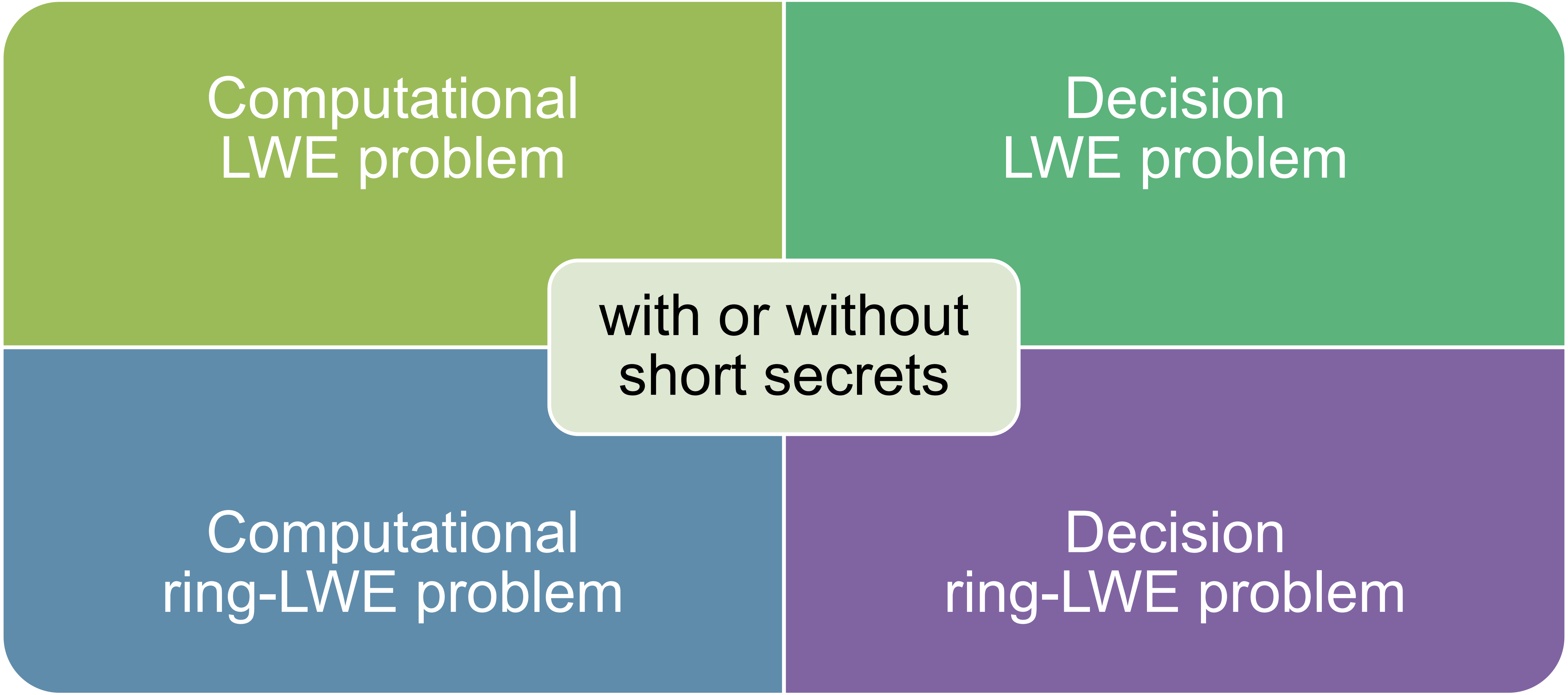
Ring learning with errors problem

$\mathbb{Z}_{13}[x]/\langle x^4 + 1 \rangle$

	$4 + 1x + 11x^2 + 10x^3$	random
×		secret
+		small noise
=	$10 + 5x + 10x^2 + 7x^3$	

Computational ring-LWE problem: given **blue**, find **red**

Problems


Computational
LWE problem
Decision
LWE problem
with or without
short secrets
Computational
ring-LWE problem
Decision
ring-LWE problem

Key agreement from LWE

Bos, Costello, Ducas, Mironov, Naehrig, Nikolaenko, Raghunathan, Stebila.
Frodo: Take off the ring! Practical, quantum-safe key exchange from LWE.
ACM Conference on Computer and Communications Security (CCS) 2016.

https://eprint.iacr.org/2016/659

LWE and ring-LWE public key encryption and key exchange

Regev
STOC 2005
- Public key encryption from LWE

Lyubashevsky, Peikert, Regev
Eurocrypt 2010
- Public key encryption from ring-LWE

Lindner, Peikert
ePrint 2010, CT-RSA 2011
- Public key encryption from LWE and ring-LWE
- Approximate key exchange from LWE

Ding, Xie, Lin
ePrint 2012
- Key exchange from LWE and ring-LWE with single-bit reconciliation

Peikert
PQCrypto 2014
- Key encapsulation mechanism based on ring-LWE and variant single-bit reconciliation

Bos, Costello, Naehrig, Stebila
IEEE S&P 2015
- Implementation of Peikert's ring-LWE key exchange, testing in TLS 1.2

"NewHope"

Alkim, Ducas, Pöppelman, Schwabe.
USENIX Security 2016

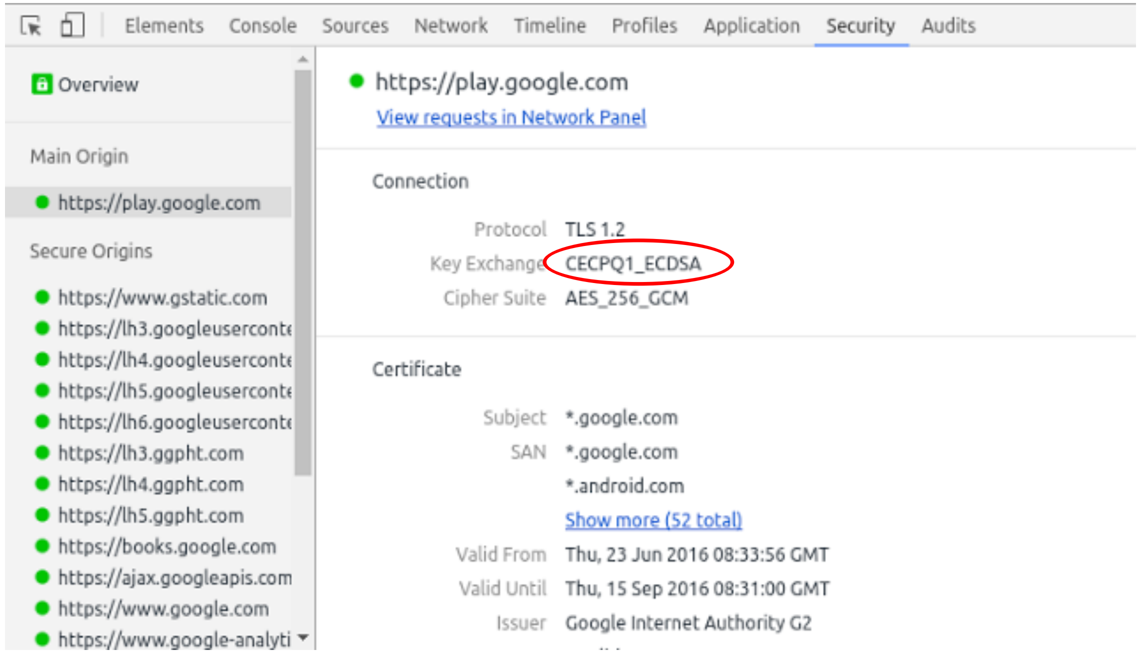
- New parameters
- Different error distribution
- Improved performance
- Pseudorandomly generated parameters

- Further performance improvements by others [GS16,LN16,…]

Google Security Blog

Experimenting with Post-Quantum Cryptography

July 7, 2016



https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html

Ring-LWE

$$\mathbb{Z}_{13}^{7\times 4}$$

4	1	11	10

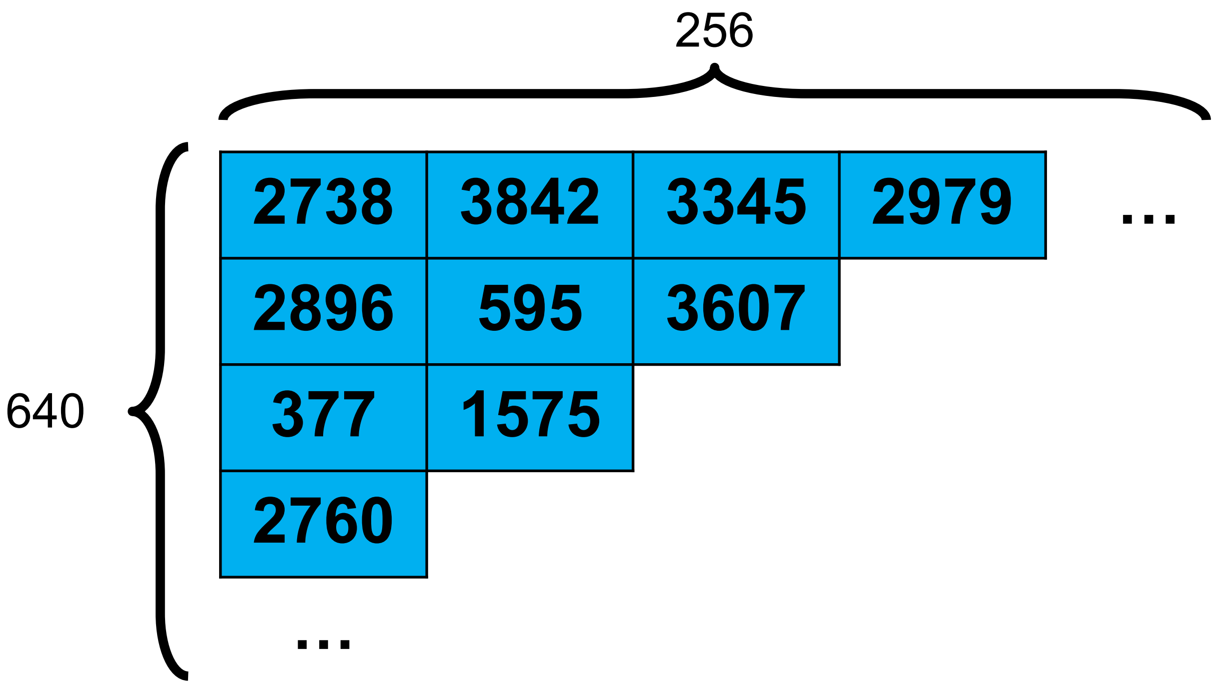
Cyclic structure

⇒ Save communication,
more efficient computation

4 KiB representation

LWE

$$\mathbb{Z}_{4093}^{640\times 256}$$



640 × 256 × 12 bits = **245 KiB**

Ring-LWE

$\mathbb{Z}_{13}^{7\times 4}$

4	1	11	10

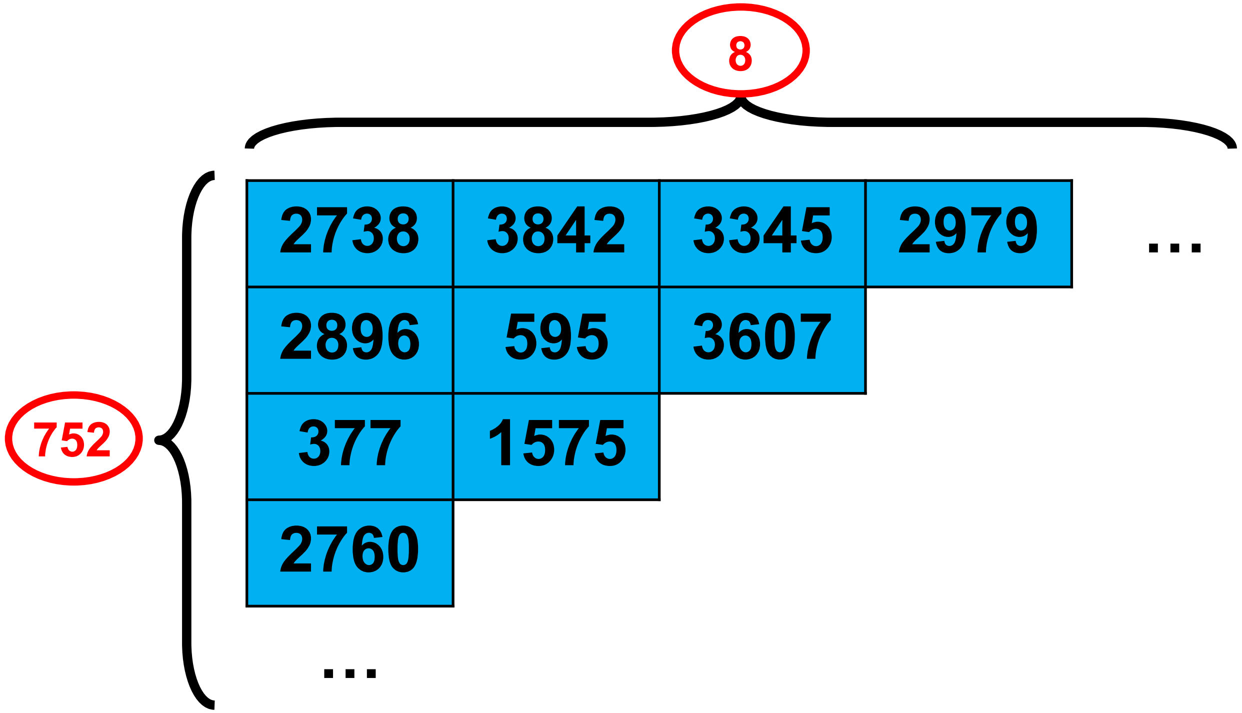
Cyclic structure

⇒ Save communication,
more efficient computation

4 KiB representation

LWE

$\mathbb{Z}_{2^{15}}^{752\times 8}$



752 × 8 × 15 bits = **11 KiB**

Why consider (slower, bigger) LWE?

Generic vs. ideal lattices

- Ring-LWE matrices have additional structure
  - Relies on hardness of a problem in **ideal** lattices
- LWE matrices have no additional structure
  - Relies on hardness of a problem in **generic** lattices
- NTRU also relies on a problem in a type of ideal lattices

- Currently, best algorithms for ideal lattice problems are essentially the same as for generic lattices
  - Small constant factor improvement in some cases
  - Very recent quantum polynomial time algorithm for Ideal-SVP (http://eprint.iacr.org/2016/885) but not immediately applicable to ring-LWE

If we want to eliminate this additional structure, can we still get an efficient protocol?

Decision learning with errors problem with short secrets

Definition. Let $n, q \in \mathbb{N}$. Let χ be a distribution over $\mathbb{Z}$.

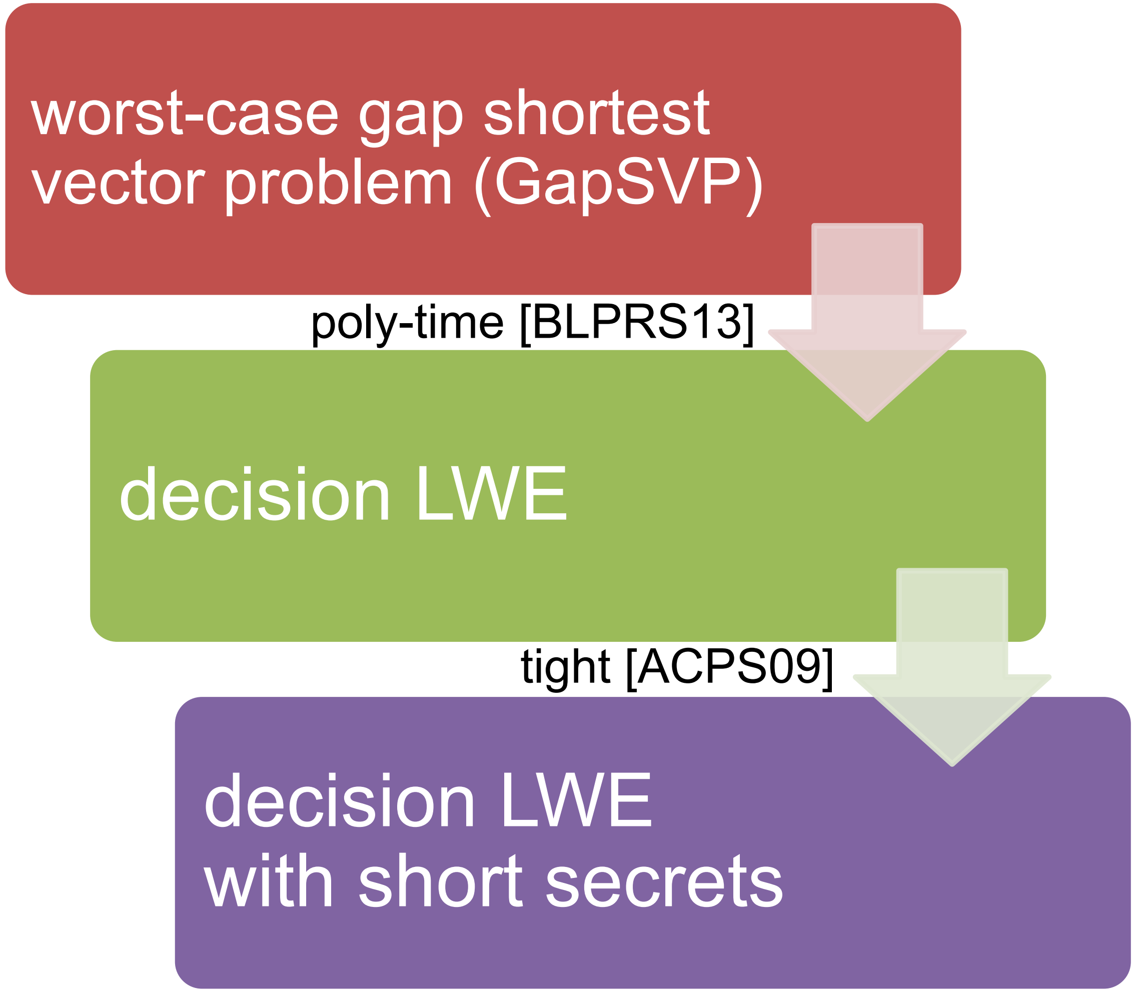
Let $\mathbf{s} \stackrel{\$}{\leftarrow} \chi^n$.

Define:

- $O_{\chi,\mathbf{s}}$: Sample $\mathbf{a} \stackrel{\$}{\leftarrow} \mathcal{U}(\mathbb{Z}_q^n)$, $e \stackrel{\$}{\leftarrow} \chi$; return $(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + e)$.
- U: Sample $(\mathbf{a}, b') \stackrel{\$}{\leftarrow} \mathcal{U}(\mathbb{Z}_q^n \times \mathbb{Z}_q)$; return $(\mathbf{a}, b')$.

The *decision LWE problem with short secrets* for n, q, χ is to distinguish $O_{\chi,\mathbf{s}}$ from U.

Hardness of decision LWE



Practice:

- Assume the best way to solve DLWE is to solve LWE.
- Assume solving LWE involves a lattice reduction problem.
- Estimate parameters based on runtime of lattice reduction algorithms.
- (Ignore non-tightness.)

Basic LWE-DH key agreement (unauthenticated)

Based on Lindner–Peikert LWE public key encryption scheme

public: “big” A in $\mathbf{Z}_q^{n \times m}$

Alice

secret:
random “small” s, e in $\mathbf{Z}_q^m$

Bob

secret:
random “small” s', e' in $\mathbf{Z}_q^n$

Alice → Bob: $b = As + e$

Bob → Alice: $b' = s'A + e'$

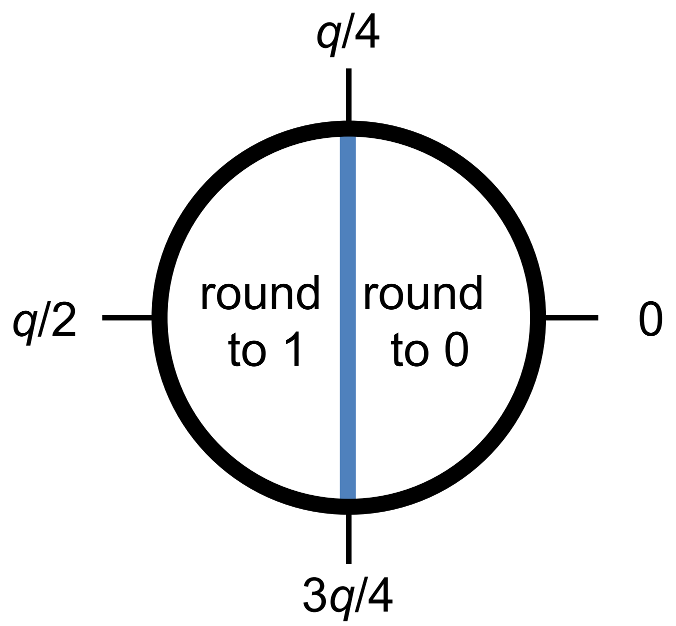
Alice — shared secret:
$b's = s'As + e's \approx s'As$

Bob — shared secret:
$s'b \approx s'As$

These are only approximately equal ⇒ need rounding

Basic rounding

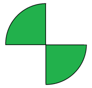
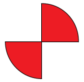
- Each entry of the matrix is an integer modulo *q*
- Round to either 0 or *q*/2
- Treat *q*/2 as 1

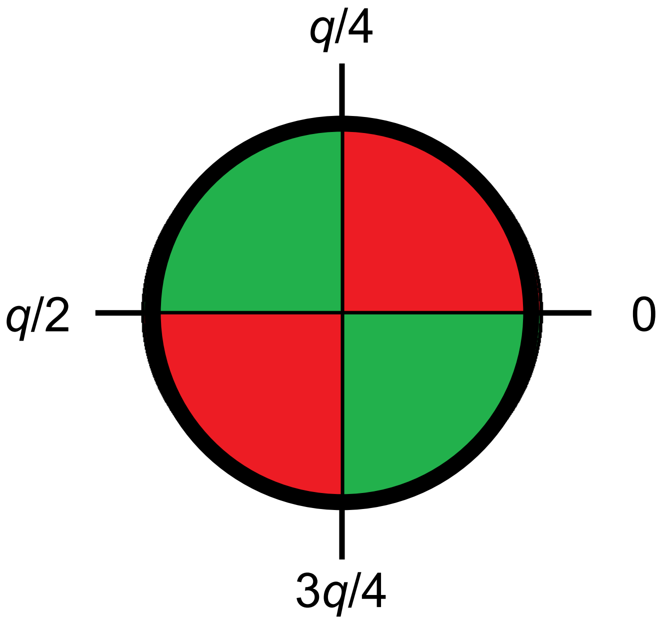


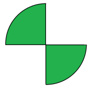
This works most of the time: prob. failure 2^{-10}.

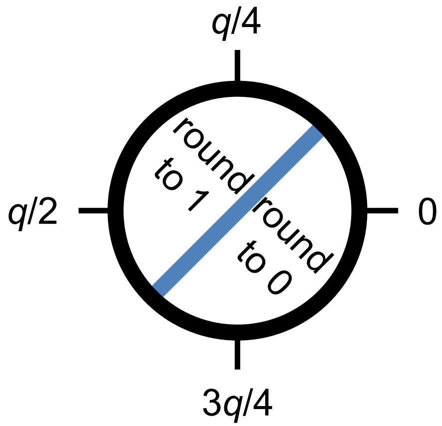
Not good enough: we need exact key agreement.

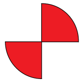
Better rounding

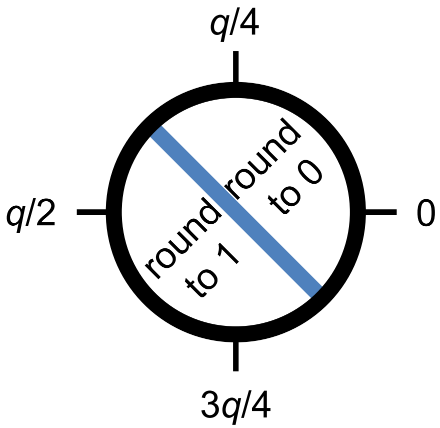
Bob says which of two regions the value is in:  or 



If 

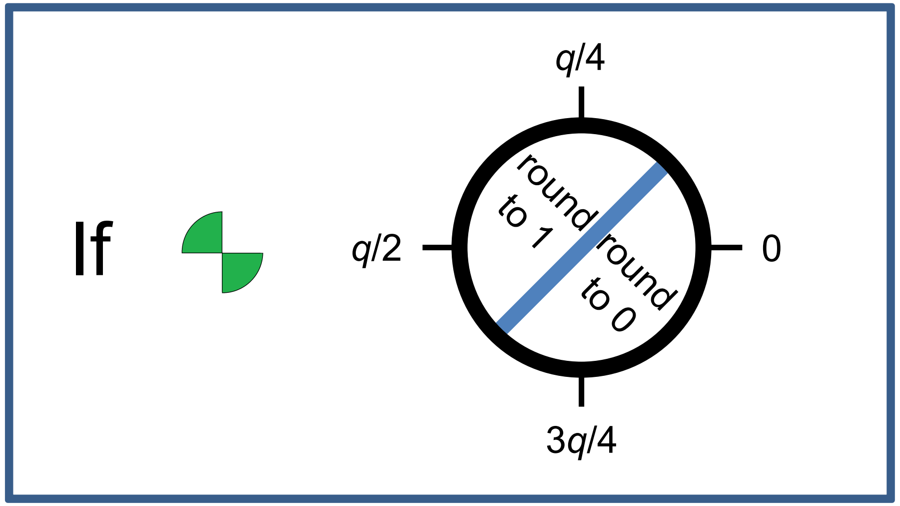


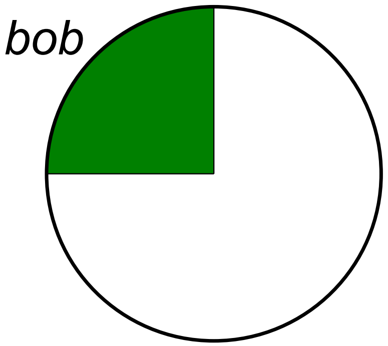
If 

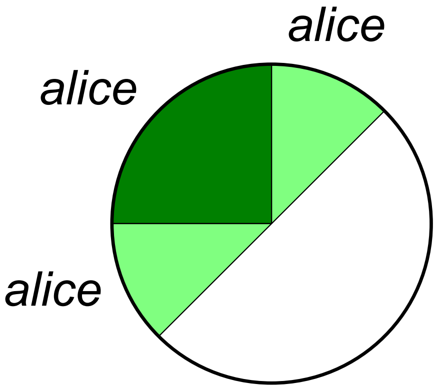


Better rounding

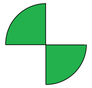
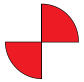
- If | *alice* – *bob* | ≤ *q*/8, then this always works.







- For our parameters, probability | *alice* – *bob* | > *q*/8 is less than $2^{-128000}$.

- Security not affected: revealing  or  leaks no information

Exact LWE-DH key agreement (unauthenticated)

Based on Lindner–Peikert LWE public key encryption scheme

public: “big” A in $\mathbf{Z}_q^{n \times m}$

Alice

secret:
random “small” s, e in $\mathbf{Z}_q^m$

Bob

secret:
random “small” s', e' in $\mathbf{Z}_q^n$

$b = As + e$ →

← $b' = s'A + e'$,  or 

Alice: shared secret:
round($b's$, hint)

Bob: shared secret:
round($s'b$)

"Frodo": LWE-DH key agreement

Based on Lindner–Peikert LWE key agreement scheme

Alice		Bob
$\mathsf{seed}_{\mathbf{A}} \xleftarrow{\$} U(\{0,1\}^s)$		
$\mathbf{A} \leftarrow \mathsf{Gen}(\mathsf{seed}_{\mathbf{A}})$		
$\mathbf{S}, \mathbf{E} \xleftarrow{\$} \chi(\mathbb{Z}_q^{n \times \overline{n}})$		
$\mathbf{B} \leftarrow \mathbf{A}\mathbf{S} + \mathbf{E}$	$\xrightarrow{\mathsf{seed}_{\mathbf{A}}, \mathbf{B}}$ $\in \{0,1\}^s \times \mathbb{Z}_q^{n \times \overline{n}}$	
		$\mathbf{A} \leftarrow \mathsf{Gen}(\mathsf{seed}_{\mathbf{A}})$
		$\mathbf{S}', \mathbf{E}' \xleftarrow{\$} \chi(\mathbb{Z}_q^{\overline{m} \times n})$
		$\mathbf{B}' \leftarrow \mathbf{S}'\mathbf{A} + \mathbf{E}'$
		$\mathbf{E}'' \xleftarrow{\$} \chi(\mathbb{Z}_q^{\overline{m} \times \overline{n}})$
		$\mathbf{V} \leftarrow \mathbf{S}'\mathbf{B} + \mathbf{E}''$
	$\xleftarrow{\mathbf{B}', \mathbf{C}}$ $\in \mathbb{Z}_q^{\overline{m} \times n} \times \mathbb{Z}_2^{\overline{m} \times \overline{n}}$	$\mathbf{C} \leftarrow \langle \mathbf{V} \rangle_{2^B}$
$K \leftarrow \mathsf{rec}(\mathbf{B}'\mathbf{S}, \mathbf{C})$		$K \leftarrow \lfloor \mathbf{V} \rceil_{2^B}$

A generated pseudorandomly

Uses two matrix forms of LWE:
- Public key is n x $\overline{n}$ matrix
- Shared secret is $\overline{m}$ x $\overline{n}$ matrix

Secure if decision learning with errors problem is hard

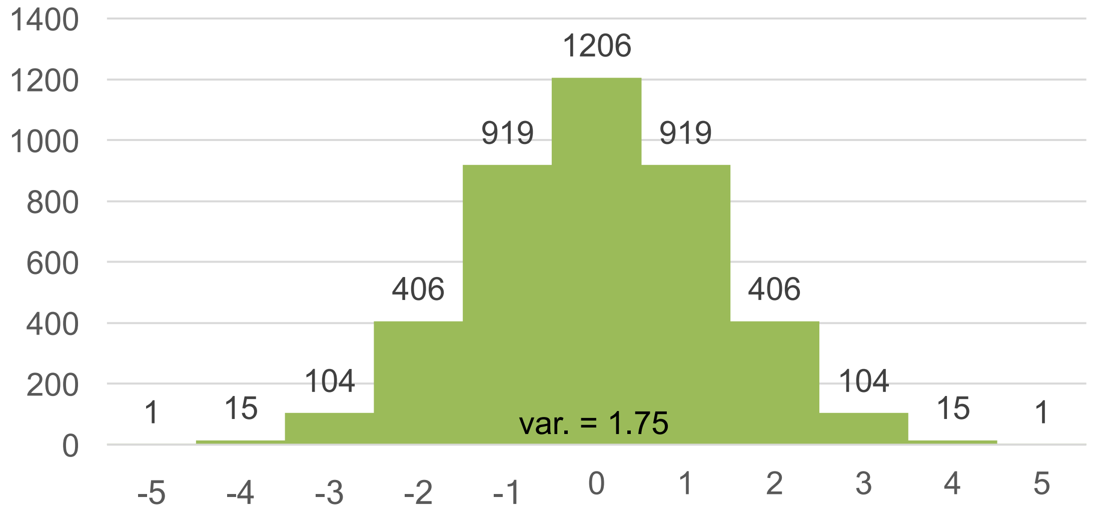
(and Gen is a secure PRF).

Rounding

- We extract 4 bits from each of the 64 matrix entries in the shared secret.
  - More granular form of previous rounding.

Parameter sizes, rounding, and error distribution all found via search scripts.

Error distribution



- Close to discrete Gaussian in terms of Rényi divergence (1.000301)
- Only requires 12 bits of randomness to sample

Parameters

All known variants of the sieving algorithm require a list of vectors to be created of this size

“Recommended”

- 144-bit classical security,
  130-bit quantum security,
  103-bit plausible lower bound

- $n = 752$, $m = 8$, $q = 2^{15}$
- χ = approximation to rounded Gaussian with 11 elements

- Failure: $2^{-38.9}$

- Total communication: 22.6 KiB

“Paranoid”

- 177-bit classical security,
  161-bit quantum security,
  128-bit plausible lower bound

- $n = 864$, $m = 8$, $q = 2^{15}$
- χ = approximation to rounded Gaussian with 13 elements

- Failure: $2^{-33.8}$

- Total communication: 25.9 KiB

Implementations

Our implementations

- Ring-LWE BCNS15
- LWE Frodo

Pure C implementations

Constant time

Compare with others

- RSA 3072-bit (OpenSSL 1.0.1f)
- ECDH `nistp256` (OpenSSL)

Use assembly code

- Ring-LWE NewHope
- NTRU `EES743EP1`
- SIDH (Isogenies) (MSR)

Pure C implementations

Standalone performance

	Speed		Communication		Quantum Security
RSA 3072-bit	Fast	4 ms	Small	0.3 KiB	
ECDH `nistp256`	Very fast	0.7 ms	Very small	0.03 KiB	
Ring-LWE BCNS	Fast	1.5 ms	Medium	4 KiB	80-bit
Ring-LWE NewHope	Very fast	0.2 ms	Medium	2 KiB	206-bit
NTRU `EES743EP1`	Fast	0.3–1.2 ms	Medium	1 KiB	128-bit
SIDH	Very slow	35–400 ms	Small	0.5 KiB	128-bit
LWE Frodo Recom.	Fast	1.4 ms	Large	11 KiB	130-bit
McBits*	Very fast	0.5 ms	Very large	360 KiB	161-bit

First 7 rows: x86_64, 2.6 GHz Intel Xeon E5 (Sandy Bridge) – Google `n1-standard-4`
* McBits results from source paper [BCS13]

Note somewhat incomparable security levels

Open Quantum Safe

https://openquantumsafe.org/

Open Quantum Safe

- MIT-licensed open-source project on Github
  - https://openquantumsafe.org/
  - https://github.com/open-quantum-safe/
- liboqs: C language library, common API

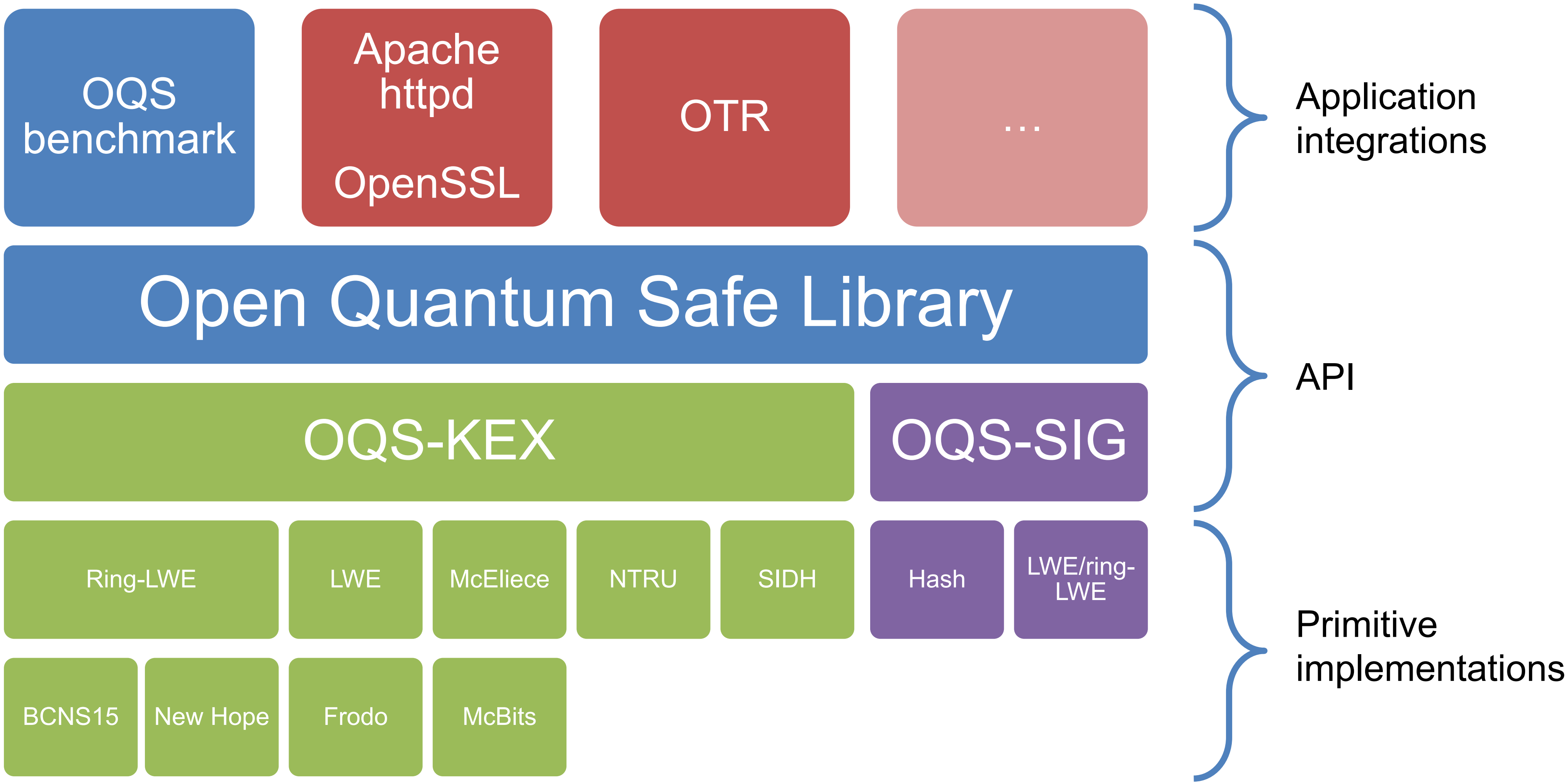
Open Quantum Safe

1. Collect post-quantum implementations together
    - Our own software
    - Thin wrappers around existing open source implementations
    - Contributions from others

2. Enable direct comparison of implementations

3. Support prototype integration into application level protocols
    - Don’t need to re-do integration for each new primitive – how we did Frodo experiments

Open Quantum Safe architecture



liboqs: Current key exchange algorithms

- **Ring-LWE**:
  - BCNS15
  - NewHope
  - MSR NewHope improvements
- **LWE**: Frodo
- **NTRU**
- **SIDH (Supersingular isogeny Diffie–Hellman)**:
  - MSR
  - IQC
- **Code**: McBits

liboqs: Benchmarking

- Built-in key exchange benchmarking suite
  - `./test_kex --bench`
- Gives cycle counts and ms runtimes

liboqs: Application integrations

OpenSSL v1.0.2:

- Ciphersuites using key exchange algorithms from liboqs
- Integrated into `openssl speed` benchmarking command and `s_client` and `s_server` command-line programs

- Track OpenSSL 1.0.2 stable with regular updates
  - https://github.com/open-quantum-safe/openssl

- Successfully used in Apache httpd and OpenVPN (with no modifications!)

OQC contributors and acknowledgements

Project leaders

- Michele Mosca and Douglas Stebila

Planning & discussions

- Scott Vanstone and Sherry Shannon Vanstone (Trustpoint)
- Matthew Campagna (Amazon Web Services)
- Alfred Menezes, Ian Goldberg, and Guang Gong (University of Waterloo)
- William Whyte and Zhenfei Zhang (Security Innovation)
- Jennifer Fernick, David Jao, and John Schanck (University of Waterloo)

Software contributors

- Mike Bender
- Tancrède Lepoint (SRI)
- Shravan Mishra (IQC)
- Christian Paquin (MSR)
- Alex Parent (IQC)
- Douglas Stebila (McMaster)
- Sebastian Verschoor (IQC)

+ Existing open-source code

Getting involved and using OQS

https://openquantumsafe.org/

If you're writing post-quantum implementations:

- We'd love to coordinate on API
- And include your software if you agree

If you want to prototype or evaluate post-quantum algorithms in applications:

- Maybe OQS will be helpful to you

We'd love help with:

- Code review and static analysis
- Signature scheme implementations
- Additional application-level integrations

Hybrid cryptography

Hybrid TLS: joint work with John Schanck

Hybrid signatures: joint work with Nina Bindel, Udyani Herath, Matthew McKague

Hybrid cryptography

- Use of two (or more) algorithms with different security properties

- Example: hybrid key exchange
  - 1 traditional key exchange algorithm (RSA, Diffie–Hellman, elliptic curves)
  - 1 post-quantum key exchange algorithm (LWE, ring-LWE, …)
  - final shared secret = Hash(traditional shared secret, post-quantum shared secret)
  - If **either** key exchange algorithm is secure, the final shared secret is secure.

Why use hybrid cryptography?

- "Hedging our bets"
- Don't trust RSA/DH to remain secure
  - => Want something post-quantum
- Not sure which post-quantum algorithm/parameters is really secure
  - => Don't want to rely on a single post-quantum algorithm
- Maybe need to use RSA/DH for compliance reasons

Concerns with hybrid cryptography

- If the individual algorithms are secure, is the combination secure?

- Degraded computational performance
- Increased bandwidth

- Backwards compatibility

Hybrid key exchange in TLS

TLS 1.3

- Client can list all supported key exchange algorithms
- But server can only pick one of these

Possible solutions

- Add hybrid key exchange algorithms to the list:
  - define new codepoints for ECDH nistp256 + NewHope, ECDH nistp256 + Frodo-Recom., ECDH nistp256 + NTRU, ECDH curve25519 + NewHope, …
  - => combinatorial explosion of algorithms
  - Not the elegant way

Hybrid key exchange in TLS

TLS 1.3

- Client can list all supported key exchange algorithms
- But server can only pick one of these

Possible solutions

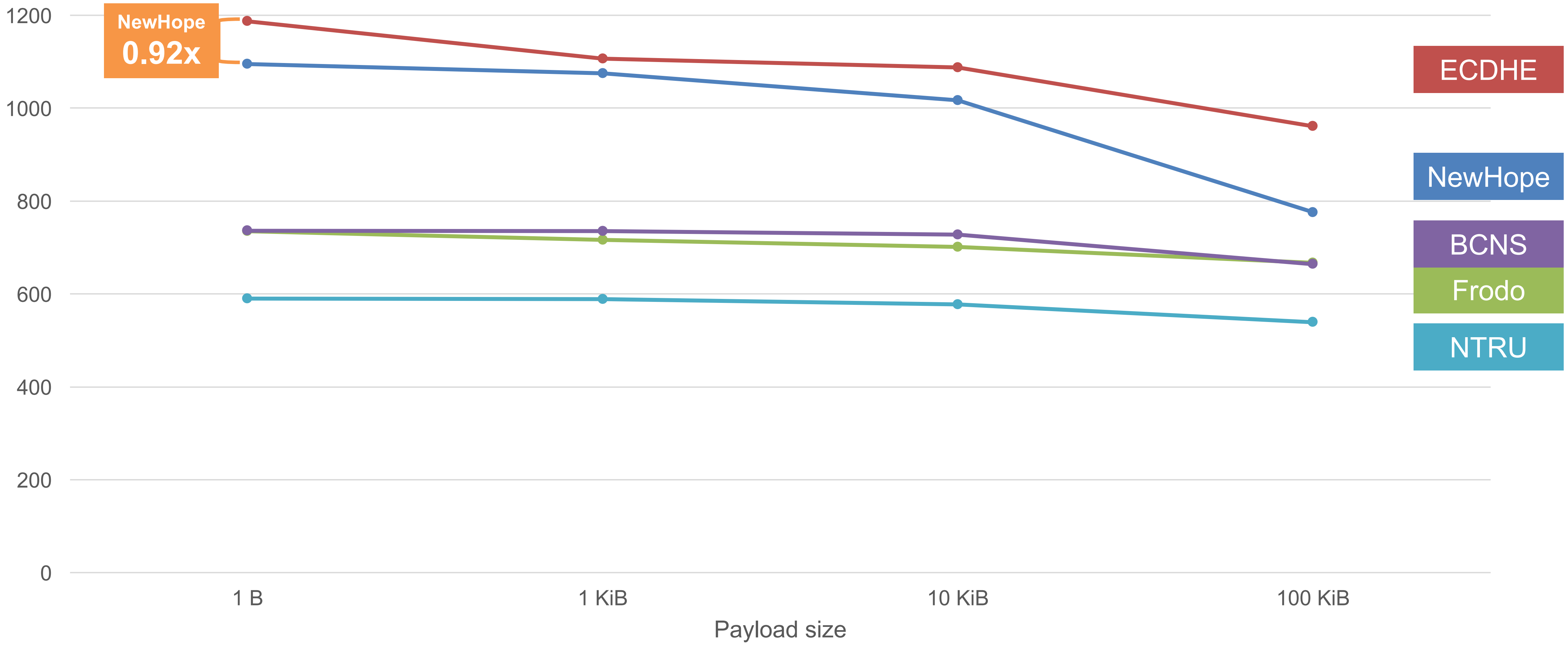
- Use ClientHello extension to request use of a second key exchange algorithm and carry public key
- Use ServerHello extension to carry public key
  - Elegant
  - Backwards compatible with servers that don't understand the extension
  - New Internet-Draft coming from Schanck & Stebila soon
  - Alternative Internet-Draft coming from Whyte et al. as well

Need to update proofs of TLS
Requires stronger security of post-quantum key exchange (IND-CCA KEM)

TLS connection throughput – hybrid w/ECDHE

ECDSA signatures

bigger (top) is better



x86_64, 2.6 GHz Intel Xeon E5 (Sandy Bridge) – server Google `n1-standard-4`, client `-32`

Note somewhat incomparable security levels

Hybrid signatures in X.509 certificates

- How to convey multiple public keys in a single certificate?
- How to sign a single certificate with multiple CA algorithms?

- **X.509 extensions**
  - Can carry arbitrary additional data
  - Put a second "post-quantum" certificate as an extension inside a traditional (RSA/ECDSA) certificate
  - Post-quantum aware software recognizes both and processes both
  - Old software ignores "non-critical" extensions
    - => backwards compatible

Hybrid signatures in X.509 certificates - Compatibility

	Extension size (and corresponding example signature scheme)				
	1.5 kB (RSA)	3.5 kB (GLP [19])	9.0 kB (BLISS [16])	43.0 kB (SPHINCS [6])	1333.0 kB (TESLA-416 [2])
Libraries					
GnuTLS 3.5.8	✓	✓	✓	✓	×
Java SSE 1.8.0	✓	✓	✓	✓	×
mbedTLS 2.3.0	✓	✓	✓	×	×
OpenSSL 1.0.2g	✓	✓	✓	✓	×
Web browsers					
Apple Safari 5.1.7	✓	✓	✓	×	–
Google Chrome 55.0.2883.87	✓	✓	✓	✓	–
Microsoft IE 11.0.38	✓	✓	✓	×	–
Mozilla Firefox 51.0.1	✓	✓	✓	✓	–
Opera 42.0.2393.137	✓	✓	✓	✓	–

Hybrid signatures in S/MIME encrypted email

- How to convey multiple signatures on a single message?

- S/MIME data structures allow multiple parallel signatures
  - But most software tries to validate **all** parallel signatures and rejects if any of them fail
  - => Not backwards compatible

- Various options with extension fields (attributes)

Research in hybrid cryptography

- For each type of primitive (key exchange, public key encryption, digital signatures), what possible ways can we combine algorithms?
  - $s_1 = \text{Sign}_1(sk_1, m)$; $s_2 = \text{Sign}_2(sk_2, m)$; $sig = (s_1, s_2)$
  - $s_1 = \text{Sign}_1(sk_1, m)$; $s_2 = \text{Sign}_2(sk_2, s_2)$; $sig = (s_1, s_2)$
  - $s_1 = \text{Sign}_1(sk_1, m)$; $s_2 = \text{Sign}_2(sk_2, m \,||\, s_1)$; $sig = (s_1, s_2)$

- Are these schemes secure against quantum adversaries?
- How quantum is the adversary?
  - Classical adversary now, quantum later
  - Quantum adversary with **only classical** access to signing/decryption oracles
  - Quantum adversary with quantum access to **random oracle**
  - Quantum adversary with quantum access to **signing/decryption oracles**

Summary

Preparing for post-quantum and hybrid cryptography on the Internet

Douglas Stebila
McMaster University

- **Learning with Errors (LWE)** can achieve reasonable key sizes and runtime with more conservative assumption
- **Open Quantum Safe** project allows for prototyping and comparison on post-quantum algorithms
- **Hybrid cryptography** will probably play a role in the transition

LWE key exchange (Frodo)

- https://eprint.iacr.org/2016/659
- https://github.com/lwe-frodo

Open Quantum Safe

- https://openquantumsafe.org/
- https://eprint.iacr.org/2016/1017