Practical, Quantum-Secure Key Exchange from LWE

Douglas Stebila
McMaster University

4th ETSI/IQC Workshop on Quantum-Safe Cryptography • September 21, 2016

Acknowledgements

Collaborators

- Joppe Bos
- Craig Costello and Michael Naehrig
- Léo Ducas
- Ilya Mironov and Ananth Raghunathan
- Michele Mosca
- 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

LWE-Frodo

- Key exchange protocol from the learning with errors problem
- Experimental results in TLS

Open Quantum Safe

- A library for comparing post-quantum primitives
  - Starting with key exchange
- Framework for easing integration into applications like OpenSSL

Why key exchange?

Premise: large-scale quantum computers don't exist right now, but we want to protect today's communications against tomorrow's adversary.

- Signatures still done with traditional primitives (RSA/ECDSA)
  - we only need authentication to be secure *now*
  - benefit: use existing RSA-based PKI
- Key agreement done with ring-LWE, LWE, …
  - Also consider "hybrid" ciphersuites that use post-quantum and traditional elliptic curve

Learning with errors problems

Solving systems of linear equations

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

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

$$
\underset{\mathbb{Z}_{13}^{7\times 4}}{\textbf{random}}\;
\begin{bmatrix}
4 & 1 & 11 & 10 \\
5 & 5 & 9 & 5 \\
3 & 9 & 0 & 10 \\
1 & 3 & 3 & 2 \\
12 & 7 & 3 & 4 \\
6 & 5 & 11 & 4 \\
3 & 3 & 5 & 0
\end{bmatrix}
\times
\underset{\mathbb{Z}_{13}^{4\times 1}}{\textbf{secret}}\;
\begin{bmatrix} 6 \\ 9 \\ 11 \\ 11 \end{bmatrix}
+
\underset{\mathbb{Z}_{13}^{7\times 1}}{\textbf{small noise}}\;
\begin{bmatrix} 0 \\ -1 \\ 1 \\ 1 \\ 1 \\ 0 \\ -1 \end{bmatrix}
=
\underset{\mathbb{Z}_{13}^{7\times 1}}{}\;
\begin{bmatrix} 4 \\ 7 \\ 2 \\ 11 \\ 5 \\ 12 \\ 8 \end{bmatrix}
$$

Learning with errors problem

random $\mathbb{Z}_{13}^{7\times 4}$					secret $\mathbb{Z}_{13}^{4\times 1}$		small noise $\mathbb{Z}_{13}^{7\times 1}$		$\mathbb{Z}_{13}^{7\times 1}$
4	1	11	10						4
5	5	9	5						7
3	9	0	10	×		+		=	2
1	3	3	2						11
12	7	3	4						5
6	5	11	4						12
3	3	5	0						8

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

Decision learning with errors problem

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

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

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.

$\Rightarrow$ Save communication, more efficient computation

Problems

Computational
LWE problem

Decision
LWE problem

with or without
short secrets

Computational
ring-LWE problem

Decision
ring-LWE problem

Key agreement from ring-LWE

Ding, Xie, Lin

ePrint 2012

- Key exchange from LWE and ring-LWE

Peikert

PQCrypto 2014

- Key encapsulation mechanism based on ring-LWE

BCNS15

Bos, Costello, Naehrig, Stebila. *IEEE Security & Privacy 2015*

- Selected parameters for the 80-bit quantum security level
- Integrated into TLS
- Communication size: 8 KiB roundtrip
- Standalone runtime: 1.4–2.1ms / party
- TLS performance impact: 1.08–1.27x slower

“NewHope”

Alkim, Ducas, Pöppelman, Scwabe.
USENIX Security 2016

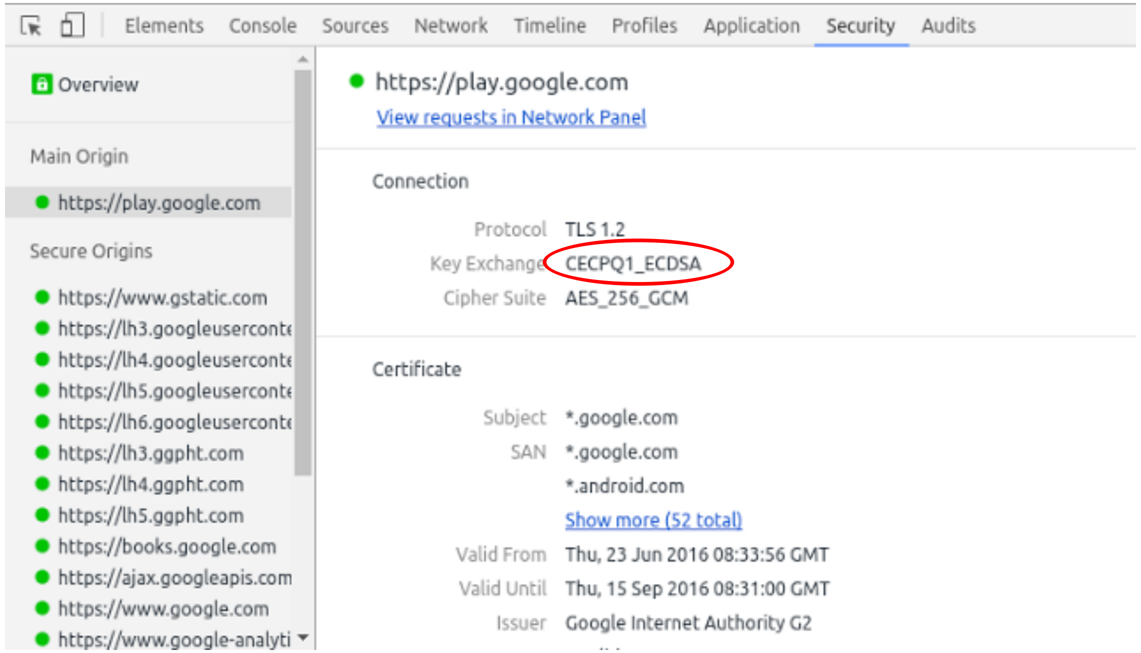
- New parameters
- Different error distribution
- Improved performance
- Pseudorandomly generated parameters

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

Google Security Blog

Experimenting with Post-Quantum Cryptography

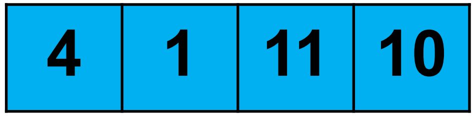
July 7, 2016



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

Ring-LWE

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



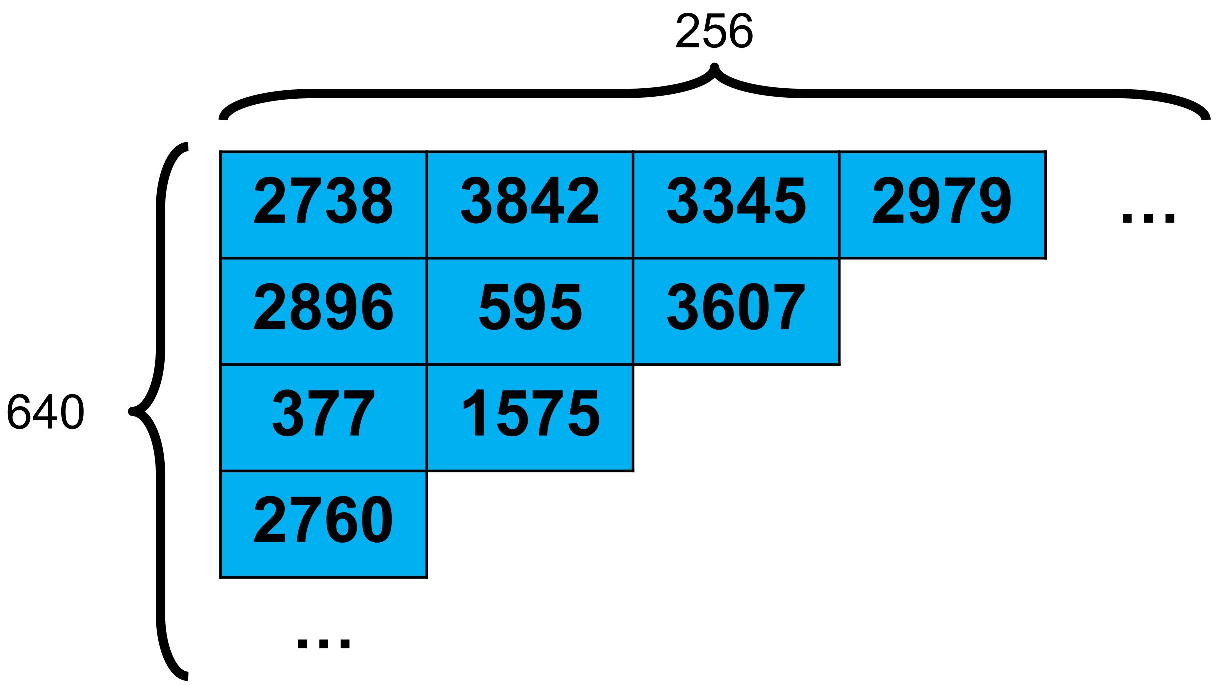
Cyclic structure

$\Rightarrow$ 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

$\Rightarrow$ Save communication,
more efficient computation

4 KiB representation

LWE

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

8

752

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

752 × 28 × 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?

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

"Frodo": LWE-DH key agreement

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

A generated pseudorandomly

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

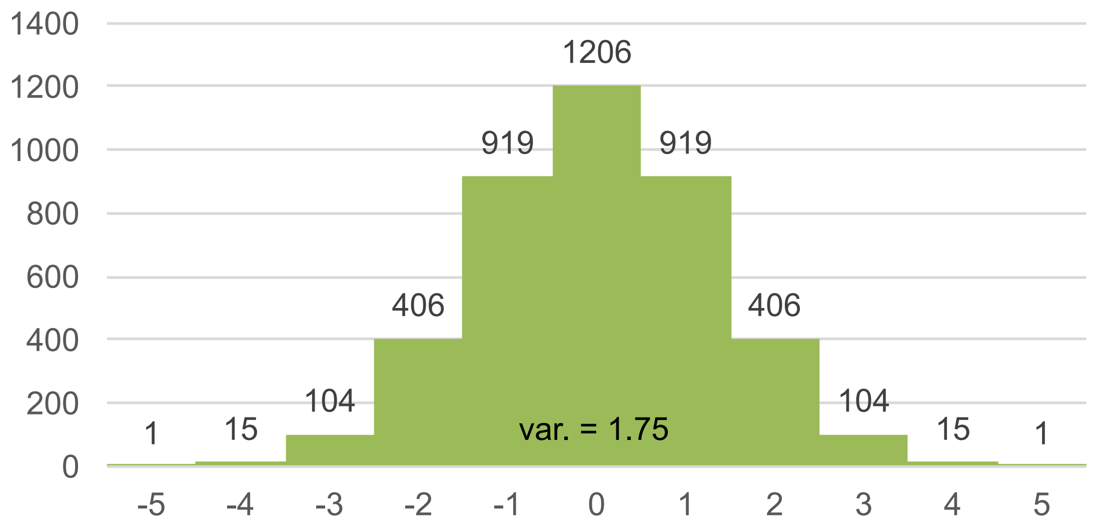
Secure if decision learning with errors problem is hard
(and Gen is a secure PRF)

Rounding

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

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

Standalone performance

Implementations

Our implementations

- BCNS15
- Frodo

Pure C implementations

Constant time

Compare with others

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

Use assembly code

- 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	
BCNS	Fast	1.5 ms	Medium	4 KiB	80-bit
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
Frodo Recommended	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

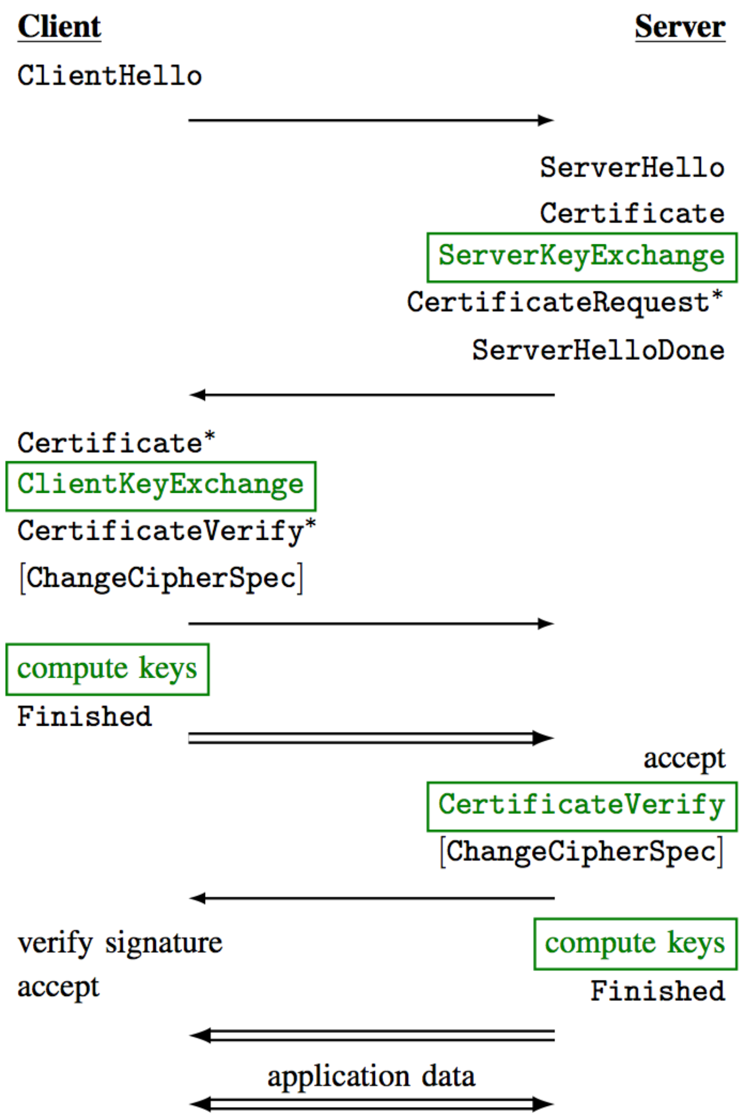
TLS integration and performance

Integration into TLS 1.2

<u>New ciphersuite:</u>

`TLS-KEX-SIG-AES256-GCM-SHA384`

- `SIG` = RSA or ECDSA signatures for authentication
- `KEX` = Post-quantum key exchange
- AES-256 in GCM for authenticated encryption
- SHA-384 for HMAC-KDF



TLS performance

Handshake latency

- Time from when client sends first TCP packet till client receives first application data
- No load on server

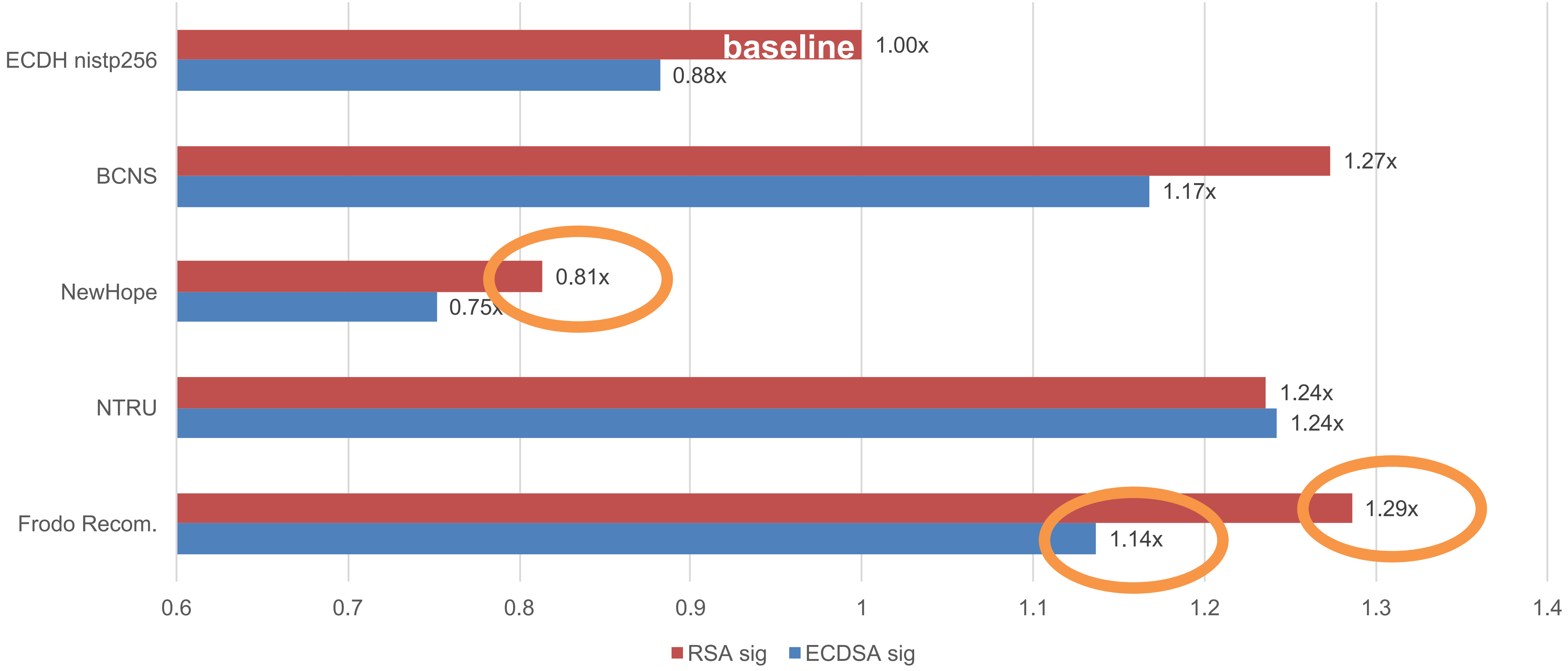
Connection throughput

- Number of connections per second at server before server latency spikes

TLS handshake latency

compared to RSA sig + ECDH nistp256

smaller (left) is better



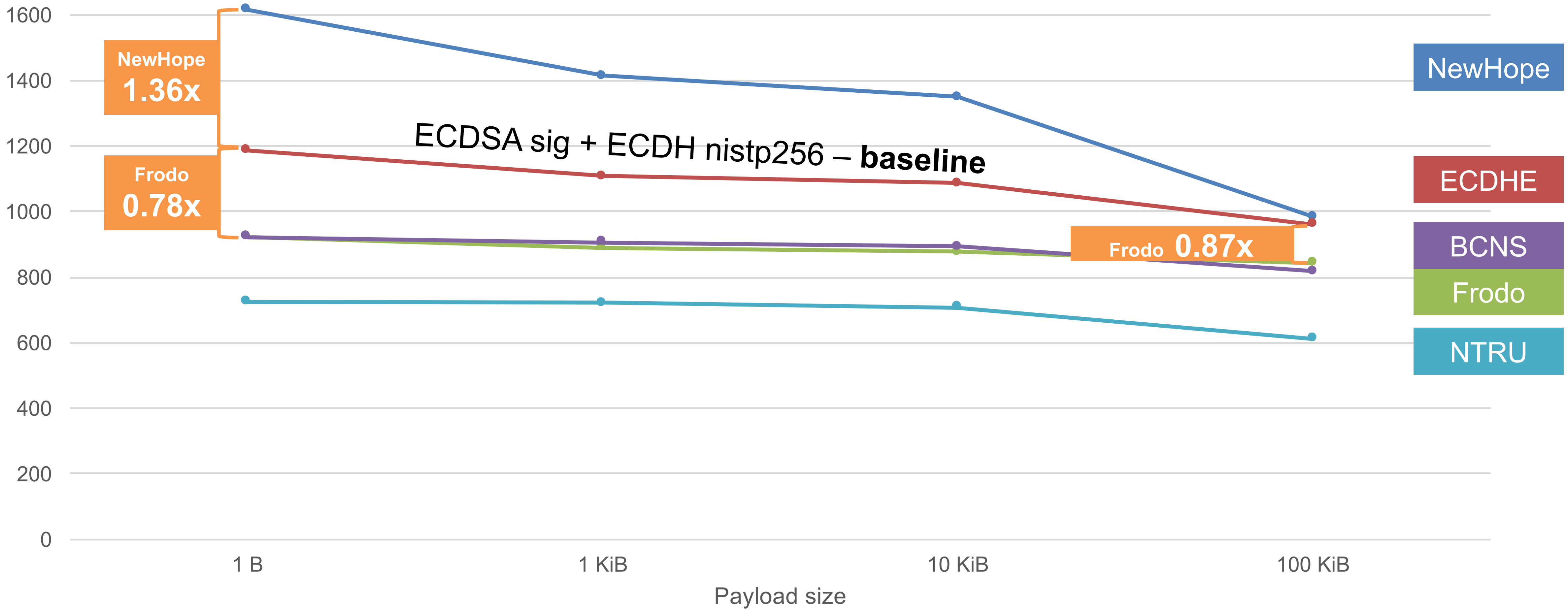
x86_64, 2.6 GHz Intel Xeon E5 (Sandy Bridge) – server Google `n1-standard-4`, client `-32`

Note somewhat incomparable security levels

TLS connection throughput

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

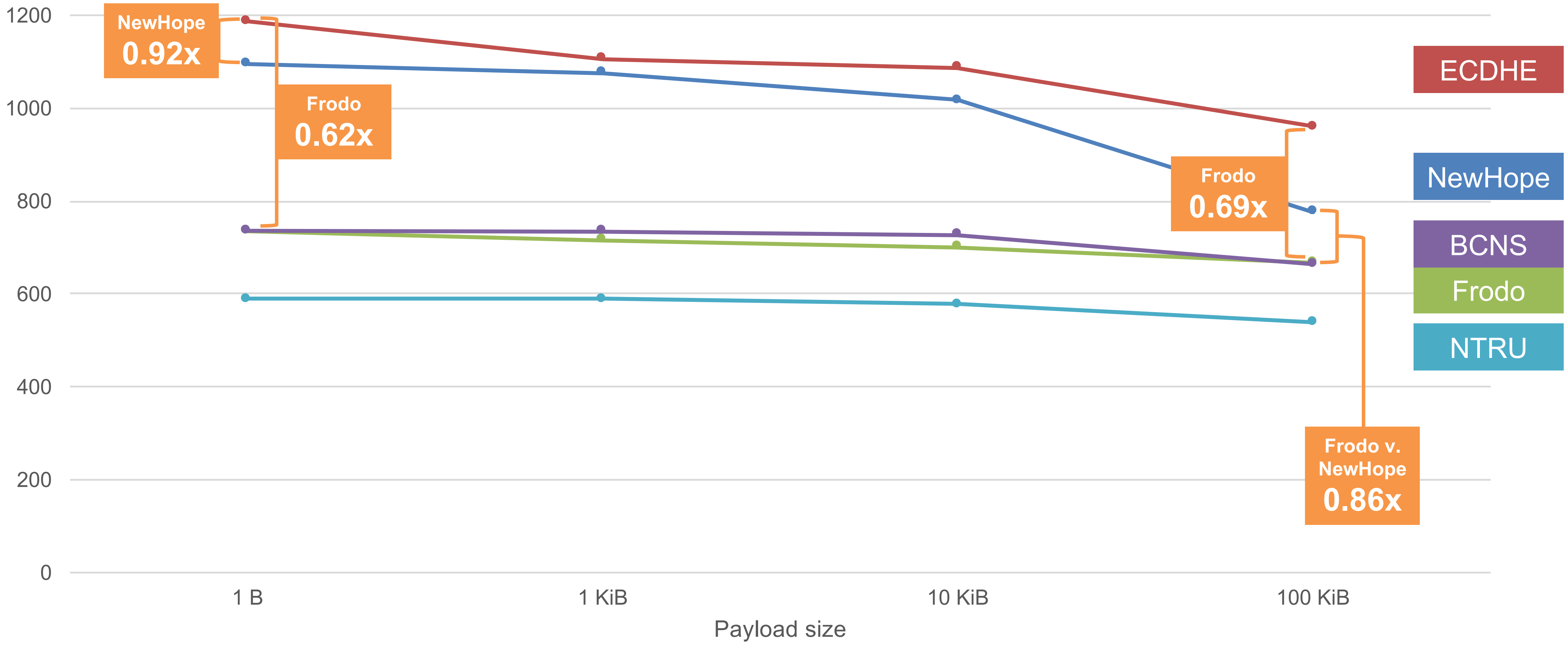
Hybrid ciphersuites

- Use both post-quantum key exchange and traditional key exchange
- Example:
  - ECDHE + NewHope
    - Used in Google Chrome experiment
  - ECDHE + Frodo
- Session key secure if either problem is hard
- Why use post-quantum?
  - (Potential) security against future quantum computer
- Why use ECDHE?
  - Security not lost against existing adversaries if post-quantum cryptanalysis advances

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

Open Quantum Safe

Collaboration with Mosca et al., University of Waterloo

https://github.com/open-quantum-safe/

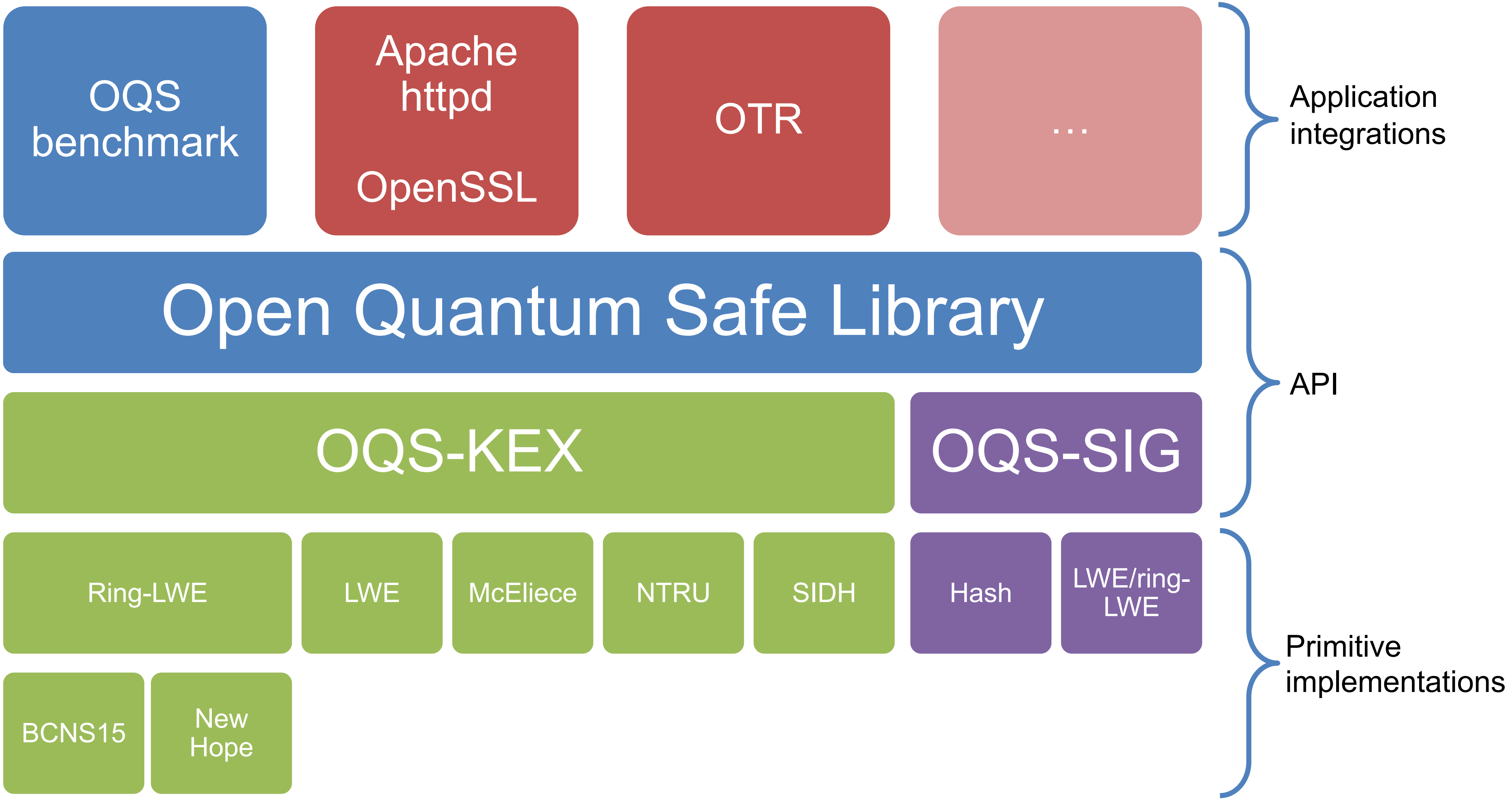
Open Quantum Safe

- Open source C library (MIT License)
- Common interface for key exchange and digital signatures

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


OQS benchmark
Apache httpd
OpenSSL
OTR
...
Application integrations
Open Quantum Safe Library
OQS-KEX
OQS-SIG
API
Ring-LWE
LWE
McEliece
NTRU
SIDH
Hash
LWE/ring-LWE
BCNS15
New Hope
Primitive implementations

Current status

- liboqs
  - ring-LWE key exchange using BCNS15
- OpenSSL
  - integration into OpenSSL 1.0.2 head
  - ring-LWE key exchange as above

Coming soon

- liboqs
  - benchmarking
  - key exchange:
    - LWE-Frodo
    - McEliece, SIDH, NewHope*, NTRU*
      (* via wrappers)
- Integrations into other applications

Getting involved and using OQS

https://github.com/open-quantum-safe/

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:

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

Summary

Practical, quantum-secure key exchange from LWE

Douglas Stebila

McMaster University

- LWE can achieve reasonable key sizes and runtime with more conservative assumption
- Performance differences are muted in application-level protocols

LWE key exchange (Frodo)

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

Open Quantum Safe

- https://github.com/open-quantum-safe/

Appendix

Decision learning with errors problem with short secrets

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

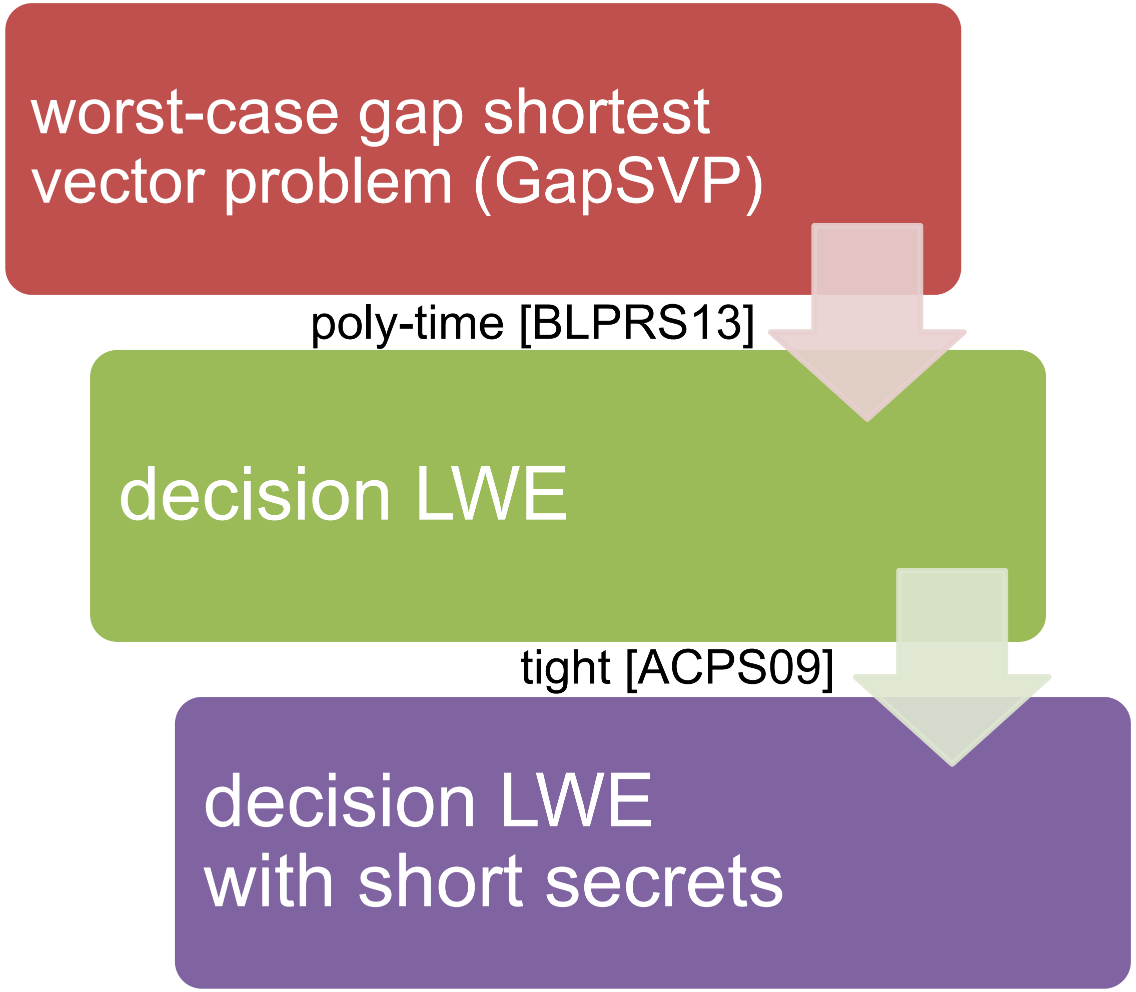
Let $\mathbf{s} \xleftarrow{\$} \chi^n$.

Define:

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



<u>Practice:</u>

- 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.)

Standalone performance

Scheme	**Alice0** (ms)	**Bob** (ms)	**Alice1** (ms)	**Communication** (bytes) **A→B**	**B→A**	**Claimed security** **classical**	**quantum**
RSA 3072-bit	—	0.09	4.49	387 / 0*	384	128	—
ECDH `nistp256`	0.366	0.698	0.331	32	32	128	—
BCNS	1.01	1.59	0.174	4,096	4,224	163	76
NewHope	0.112	0.164	0.034	1,824	2,048	229	206
NTRU `EES743EP1`	2.00	0.281	0.148	1,027	1,022	256	128
SIDH	135	464	301	564	564	192	128
Frodo Recomm.	**1.13**	**1.34**	**0.13**	**11,377**	**11,296**	**144**	**130**
Frodo Paranoid	1.25	1.64	0.15	13,057	12,976	177	161

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

Note somewhat incomparable security levels

Security within TLS 1.2

Model:

- authenticated and confidential channel establishment (ACCE) [JKSS12]

Theorem:

- signed LWE/ring-LWE ciphersuite is ACCE-secure if underlying primitives (signatures, LWE/ring-LWE, authenticated encryption) are secure
  - Interesting technical detail for ACCE provable security people: need to move server's signature to end of TLS handshake because oracle-DH assumptions don't hold for ring-LWE or use an IND-CCA KEM for key exchange via e.g. [FO99]

Open Quantum Safe architecture