The migration to post-quantum cryptography

Peter Schwabe

Max Planck Institute for Security and Privacy

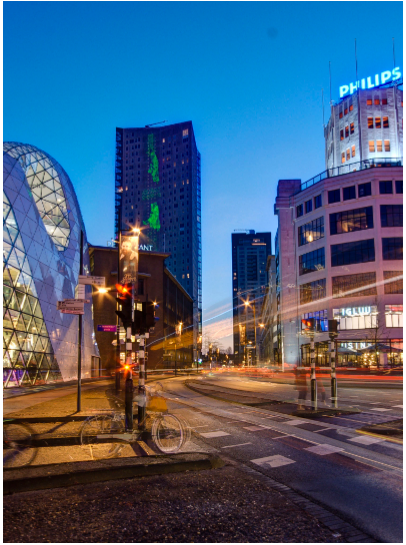
May 9, 2025

About me

- **2001–2007: Aachen**
  Studied Computer Science (Diplom)

About me

- **2001–2007: Aachen**
  Studied Computer Science (Diplom)
- **2008–2011: Eindhoven**
  Ph.D. in Department of Mathematics



About me

- **2001–2007: Aachen**
  Studied Computer Science (Diplom)
- **2008–2011: Eindhoven**
  Ph.D. in Department of Mathematics
- **2011–2012: Taipei**
  Postdoc at Academia Sinica and NTU

About me

- **2001–2007: Aachen**
  Studied Computer Science (Diplom)
- **2008–2011: Eindhoven**
  Ph.D. in Department of Mathematics
- **2011–2012: Taipei**
  Postdoc at Academia Sinica and NTU
- **Since 2013: Nijmegen**
  From Assistant to Full Professor

Since Sep. 2020: MPI-SP

- Located in **Bochum**
- Founded in 2019
- Currently 12 PIs
- Aim to have
  - 6 Departments
  - 12 Research Groups
  - Around 250 people total
- Currently on RUB campus

[A small demo]

ECDH and X25519

Let G be a finite cyclic group with generator g.

Alice		Bob
$A \leftarrow g^a$		$B \leftarrow g^b$
	A $\longrightarrow$	
	$\longleftarrow$ B	
$K \leftarrow B^a = (g^b)^a = g^{ab}$		$K \leftarrow A^b = (g^a)^b = g^{ab}$

ECDH and X25519

- Diffie, Hellman, 1976: Use $G = GF(q)^*$
- Miller, Koblitz (independently), 1985/86: Use group of points on an elliptic curve
- Bernstein, 2006: Use specific elliptic curve over $GF(2^{255} - 19)$

(EC)DH is everywhere

Nitrokey
secure your digital life

WireGuard
FAST, MODERN, SECURE VPN TUNNEL

Tor

zoom

The Discrete Logarithm Problem

Definition

Given $P, Q \in G$ such that $Q \in \langle P \rangle$, find an integer k such that $P^k = Q$.

The Discrete Logarithm Problem

Definition

Given $P, Q \in G$ such that $Q \in \langle P \rangle$, find an integer k such that $kP = Q$.

The Discrete Logarithm Problem

Definition

Given $P, Q \in G$ such that $Q \in \langle P \rangle$, find an integer k such that $kP = Q$.

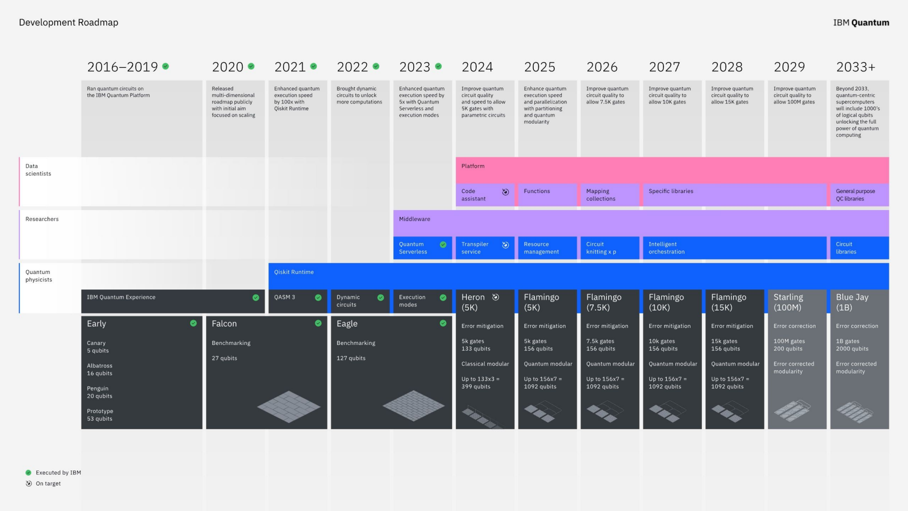
- DH needs group where DLP is hard
- (EC)DLP-based crypto also for signatures (DSA, ECDSA, EdDSA...)
- Prominent alternative: RSA (based on factoring)

Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.



See https://www.ibm.com/quantum/blog/ibm-quantum-roadmap-2025

[Back to our demo]

POST-QUANTUM KEY EXCHANGE

A NEW HOPE

ERDEM ALKIM
LÉO DUCAS
THOMAS PÖPPELMANN
PETER SCHWABE

Key Encapsulation Mechanisms (KEMs)

Initiator		Responder
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KEM.Gen}$		
	pk →	
		$(\mathsf{ct}, K) \leftarrow \mathsf{KEM.Enc}(\mathsf{pk})$
	← ct	
$K \leftarrow \mathsf{KEM.Dec}(\mathsf{ct}, \mathsf{sk})$		

Learning with errors (LWE)

- Given uniform $\mathbf{A} \in \mathbb{Z}_q^{k \times \ell}$
- Given “noise distribution” χ
- Given samples $\mathbf{A}\mathbf{s} + \mathbf{e}$, with $\mathbf{e} \leftarrow \chi$

Learning with errors (LWE)

- Given uniform $\mathbf{A} \in \mathbb{Z}_q^{k \times \ell}$
- Given “noise distribution” χ
- Given samples $\mathbf{A}\mathbf{s} + \mathbf{e}$, with $\mathbf{e} \leftarrow \chi$
- Search version: find $\mathbf{s}$
- Decision version: distinguish from uniform random

Ring Learning with errors (RLWE)

- Given uniform $\mathbf{a} \in \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$
- Given “noise distribution” χ
- Given samples $\mathbf{as} + \mathbf{e}$, with $\mathbf{e} \leftarrow \chi$

Ring Learning with errors (RLWE)

- Given uniform $\mathbf{a} \in \mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$
- Given "noise distribution" χ
- Given samples $\mathbf{as} + \mathbf{e}$, with $\mathbf{e} \leftarrow \chi$
- Search version: find $\mathbf{s}$
- Decision version: distinguish from uniform random

Alice (server)		Bob (client)
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{a}\mathbf{s} + \mathbf{e}$	$\xrightarrow{\mathbf{b}}$	$\mathbf{u} \leftarrow \mathbf{a}\mathbf{s}' + \mathbf{e}'$
	$\xleftarrow{\mathbf{u}}$	

Alice has $\mathbf{v} = \mathbf{u}\mathbf{s} = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}'\mathbf{s}$

Bob has $\mathbf{v}' = \mathbf{b}\mathbf{s}' = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}\mathbf{s}'$

- Secret and noise polynomials $\mathbf{s}, \mathbf{s}', \mathbf{e}, \mathbf{e}'$ are small
- $\mathbf{v}$ and $\mathbf{v}'$ are *approximately* the same

How to build a KEM, part 2

Alice		Bob
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{a}\mathbf{s} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}\quad)}$	
		$\mathbf{u} \leftarrow \mathbf{a}\mathbf{s}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{b}\mathbf{s}'$
$\mathbf{v}' \leftarrow \mathbf{u}\mathbf{s}$	$\xleftarrow{(\mathbf{u}\quad)}$	

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}'$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}\quad)}$	

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}'$
		$k \xleftarrow{\$} \{0,1\}^n$
		$\mathbf{k} \leftarrow \text{Encode}(k)$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}, \mathbf{c})}$	$\mathbf{c} \leftarrow \mathbf{v} + \mathbf{k}$

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}' + \mathbf{e}''$
		$k \xleftarrow{\$} \{0,1\}^n$
		$\mathbf{k} \leftarrow \text{Encode}(k)$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}, \mathbf{c})}$	$\mathbf{c} \leftarrow \mathbf{v} + \mathbf{k}$

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}' + \mathbf{e}''$
		$k \xleftarrow{\$} \{0,1\}^n$
		$\mathbf{k} \leftarrow \text{Encode}(k)$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}, \mathbf{c})}$	$\mathbf{c} \leftarrow \mathbf{v} + \mathbf{k}$
$\mathbf{k}' \leftarrow \mathbf{c} - \mathbf{v}'$		

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}' + \mathbf{e}''$
		$k \xleftarrow{\$} \{0,1\}^n$
		$\mathbf{k} \leftarrow \text{Encode}(k)$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}, \mathbf{c})}$	$\mathbf{c} \leftarrow \mathbf{v} + \mathbf{k}$
$\mathbf{k}' \leftarrow \mathbf{c} - \mathbf{v}'$		$\mu \leftarrow \text{Extract}(\mathbf{k})$
$\mu \leftarrow \text{Extract}(\mathbf{k}')$		

How to build a KEM, part 2

Alice		Bob
$seed \xleftarrow{\$} \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$		
$\mathbf{s}, \mathbf{e} \xleftarrow{\$} \chi$		$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \xleftarrow{\$} \chi$
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{(\mathbf{b}, seed)}$	$\mathbf{a} \leftarrow \text{Parse}(\text{XOF}(seed))$
		$\mathbf{u} \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}' + \mathbf{e}''$
		$k \xleftarrow{\$} \{0,1\}^n$
		$\mathbf{k} \leftarrow \text{Encode}(k)$
$\mathbf{v}' \leftarrow \mathbf{us}$	$\xleftarrow{(\mathbf{u}, \mathbf{c})}$	$\mathbf{c} \leftarrow \mathbf{v} + \mathbf{k}$
$\mathbf{k}' \leftarrow \mathbf{c} - \mathbf{v}'$		$\mu \leftarrow \text{Extract}(\mathbf{k})$
$\mu \leftarrow \text{Extract}(\mathbf{k}')$		

Encryption scheme by Lyubashevsky, Peikert, Regev. Eurocrypt 2010.

Encode and Extract

- Encoding in LPR encryption: map n bits to n coefficients:
  - A zero bit maps to 0
  - A one bit maps to $q/2$
- Idea: Noise affects low bits of coefficients, put data into high bits

Encode and Extract

- Encoding in LPR encryption: map n bits to n coefficients:
  - A zero bit maps to 0
  - A one bit maps to $q/2$
- Idea: Noise affects low bits of coefficients, put data into high bits
- Decode: map coefficient into $[-q/2, q/2]$
  - Closer to 0 (i.e., in $[-q/4, q/4]$): set bit to zero
  - Closer to $\pm q/2$: set bit to one

NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope

NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope
- NewHope-Simple (2016): Simpler reconciliation (pay 6.25% increase in ciphertext size)

NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope
- NewHope-Simple (2016): Simpler reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Parameters chosen to enable fast implementations (NTT)

NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope
- NewHope-Simple (2016): Simpler reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Parameters chosen to enable fast implementations (NTT)
- Centered binomial noise ψ_k (HW(a)−HW(b) for k-bit a, b)
- Achieve ≈ 256 bits of post-quantum security according to very conservative analysis
- Higher security, shorter messages, and $> 10\times$ speedup

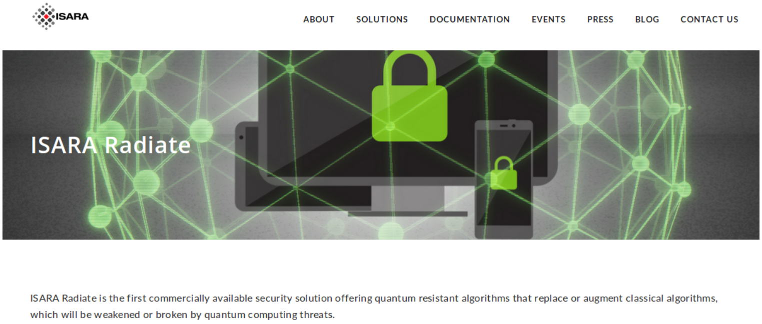
NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope
- NewHope-Simple (2016): Simpler reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Parameters chosen to enable fast implementations (NTT)
- Centered binomial noise ψ_k (HW(a)−HW(b) for k-bit a, b)
- Achieve ≈ 256 bits of post-quantum security according to very conservative analysis
- Higher security, shorter messages, and $> 10\times$ speedup
- Choose a fresh parameter $\mathbf{a}$ for every protocol run

NewHope (USENIX Security 2016)

- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconcilation to go from approximate agreement to agreement
  - Originally proposed by Ding (2012)
  - Improvements by Peikert (2014)
  - More improvements in NewHope
- NewHope-Simple (2016): Simpler reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Parameters chosen to enable fast implementations (NTT)
- Centered binomial noise ψ_k (HW(a)−HW(b) for k-bit a, b)
- Achieve ≈ 256 bits of post-quantum security according to very conservative analysis
- Higher security, shorter messages, and $> 10\times$ speedup
- Choose a fresh parameter $\mathbf{a}$ for every protocol run
- Multiple implementations

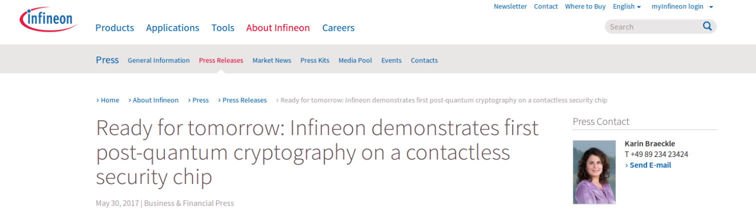
Beyond the paper. . .



“Key Agreement using the ‘NewHope’ lattice-based algorithm detailed in the New Hope paper, and LUKE (Lattice-based Unique Key Exchange), an ISARA speed-optimized version of the NewHope algorithm.”

https://www.isara.com/isara-radiate/

Beyond the paper. . .



Newsletter Contact Where to Buy English myInfineon login

infineon

Products Applications Tools About Infineon Careers

Search

Press General Information Press Releases Market News Press Kits Media Pool Events Contacts

› Home › About Infineon › Press › Press Releases › Ready for tomorrow: Infineon demonstrates first post-quantum cryptography on a contactless security chip

Ready for tomorrow: Infineon demonstrates first post-quantum cryptography on a contactless security chip

May 30, 2017 | Business & Financial Press

Press Contact

Karin Braeckle
T +49 89 234 23424
› Send E-mail

“The deployed algorithm is a variant of “New Hope”, a quantum-resistant cryptosystem”

https://www.infineon.com/cms/en/about-infineon/press/press-releases/2017/INFCCS201705-056.html

Beyond the paper. . .

"We're indebted to Erdem Alkim, Léo Ducas, Thomas Pöppelmann and Peter Schwabe, the researchers who developed "New Hope", the post-quantum algorithm that we selected for this experiment."

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

Also back in 2016: NIST PQC

- National Institute of Standards and Technology
- Public call for PQC proposals, aims at finding schemes for standardization
- Similar to earlier AES and SHA-3 efforts
- Process draft online in August 2016, comments by September 2016
- Call for proposals in December 2016, deadline November 2017

Also back in 2016: NIST PQC

- National Institute of Standards and Technology
- Public call for PQC proposals, aims at finding schemes for standardization
- Similar to earlier AES and SHA-3 efforts
- Process draft online in August 2016, comments by September 2016
- Call for proposals in December 2016, deadline November 2017

How it went

Nov. 2017 — 69 proposals —Round 1→ Feb. 2019 — 26 proposals —Round 2→ Jul. 2020 — 7+8 proposals —Round 3→ Jul. 2022 — 4 “winners”

- Second KEM selected for standardization in March 2025

NIST PQC: the beginning

Count of Problem Category	Column Labels		
Row Labels	Key Exchange	Signature	Grand Total
?	1		1
Braids	1	1	2
Chebychev	1		1
Codes	19	5	24
Finite Automata	1	1	2
Hash		4	4
Hypercomplex Numbers	1		1
Isogeny	1		1
Lattice	24	4	28
Mult. Var	6	7	13
Rand. walk	1		1
RSA	1	1	2
Grand Total	**57**	**23**	**80**

Overview tweeted by Jacob Alperin-Sheriff on Dec 4, 2017.

KYBER

Roberto Avanzi
Léo Ducas
Vadim Lyubashevsky
Gregor Seiler
Joppe Bos
Eike Kiltz
John M. Schanck
Damien Stehlé
Jintai Ding
Tancrede Lepoint
Peter Schwabe

From NewHope to Kyber

MLWE instead of RLWE

IND-CCA2 Security

From NewHope to Kyber

MLWE instead of RLWE

- Easily scale security
- Optimized routines the same for all security levels

IND-CCA2 Security

From NewHope to Kyber

MLWE instead of RLWE

- Easily scale security
- Optimized routines the same for all security levels

IND-CCA2 Security

- Support static (or cached) keys
- More robust
- Useful for authenticated key exchange
- Easy to construct PKE

Module Learning with Errors (MLWE)

- RLWE uses arithmetic on large degree polynomials
- For example, NewHope uses $n = 1024$, $q = 12289$

Module Learning with Errors (MLWE)

- RLWE uses arithmetic on large degree polynomials
- For example, NEWHOPE uses $n = 1024$, $q = 12289$
- MLWE uses matrices and vectors of smaller polynomials of small dimension

Module Learning with Errors (MLWE)

- RLWE uses arithmetic on large degree polynomials
- For example, NewHope uses $n = 1024$, $q = 12289$
- MLWE uses matrices and vectors of smaller polynomials of small dimension
- Kyber: $n = 256$, $q = 3329$
  - Security level 1 (AES-128): $d = 2$
  - Security level 3 (AES-192): $d = 3$
  - Security level 5 (AES-256): $d = 4$
- Core arithmetic is in $\mathbb{Z}_{3329}[X]/(X^{256} + 1)$ **for all security levels**

Module Learning with Errors (MLWE)

- RLWE uses arithmetic on large degree polynomials
- For example, NEWHOPE uses $n = 1024$, $q = 12289$
- MLWE uses matrices and vectors of smaller polynomials of small dimension
- Kyber: $n = 256$, $q = 3329$
  - Security level 1 (AES-128): $d = 2$
  - Security level 3 (AES-192): $d = 3$
  - Security level 5 (AES-256): $d = 4$
- Core arithmetic is in $\mathbb{Z}_{3329}[X]/(X^{256} + 1)$ **for all security levels**
- Noise is centered binomial HW(x) − HW(y) for 2-bit x and y

Chosen-ciphertext attacks

- Decryption failures are a function of $\mathbf{s}$, $\mathbf{e}$, $\mathbf{s}'$, $\mathbf{e}'$
- Attacker can choose larger secret/noise $\mathbf{e}'$ and $\mathbf{s}'$
- Observe if decryption fails
- Learn something about $\mathbf{s}$

Chosen-ciphertext attacks

- Decryption failures are a function of $\mathbf{s}, \mathbf{e}, \mathbf{s}', \mathbf{e}'$
- Attacker can choose larger secret/noise $\mathbf{e}'$ and $\mathbf{s}'$
- Observe if decryption fails
- Learn something about $\mathbf{s}$
- This is a chosen ciphertext attack (CCA)
- Learn full $\mathbf{s}$ after a few thousand queries

Chosen-ciphertext attacks

- Decryption failures are a function of $\mathbf{s}$, $\mathbf{e}$, $\mathbf{s}'$, $\mathbf{e}'$
- Attacker can choose larger secret/noise $\mathbf{e}'$ and $\mathbf{s}'$
- Observe if decryption fails
- Learn something about $\mathbf{s}$
- This is a chosen ciphertext attack (CCA)
- Learn full $\mathbf{s}$ after a few thousand queries
- NEWHOPE never claimed CCA-security!
- This “attack” is completely expected
- Not a problem for ephemeral $\mathbf{s}$

From passive to CCA security

The Fujisaki-Okamoto Transform (idea)

- Build CCA-secure KEM from passively secure encryption scheme
- Make failure probability negligible for honest $\mathbf{s}'$, $\mathbf{e}'$, $\mathbf{e}''$
- Force encapsulator to generate $\mathbf{s}'$, $\mathbf{e}'$, $\mathbf{e}''$ honestly

From passive to CCA security

The Fujisaki-Okamoto Transform

Alice (Server)		Bob (Client)
<u>Gen():</u>		<u>Encaps(pk):</u>
pk, sk ← KeyGen()	$\overset{\text{pk}}{\rightarrow}$	$x \leftarrow \{0, \ldots, 255\}^{32}$ k, coins ← SHA3-512(x)
	$\overset{\text{ct}}{\leftarrow}$	ct ← Encrypt(pk, x, coins)
<u>Decaps((sk, pk), ct):</u> x' ← Decrypt(sk, ct) $k', coins'$ ← SHA3-512(x') ct′ ← Encrypt(pk, x', coins′) **verify if** ct = ct′		

Kyber today

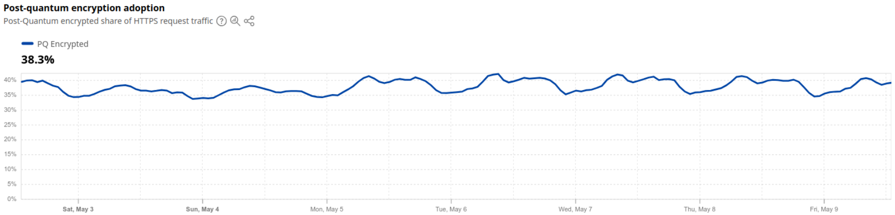
- Various tweaks through NIST PQC rounds
- Standardized in FIPS-203 as “ML-KEM” in 2024

Kyber today

- Various tweaks through NIST PQC rounds
- Standardized in FIPS-203 as “ML-KEM” in 2024
- Used in, e.g., Signal, iMessage, Firefox, Chrome, AWS. . .

Kyber today

- Various tweaks through NIST PQC rounds
- Standardized in FIPS-203 as “ML-KEM” in 2024
- Used in, e.g., Signal, iMessage, Firefox, Chrome, AWS. . .
- Used in TLS 1.3 by e.g. Cloudflare, Google



From `https://radar.cloudflare.com/adoption-and-usage#post-quantum-encryption-adoption`

- Several 100 billion connections per day

Learn more

NIST PQC website:
https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

NIST mailing list:
https://csrc.nist.gov/projects/post-quantum-cryptography/email-list
https://groups.google.com/a/list.nist.gov/g/pqc-forum

Kyber website:
https://pq-crystals.org/kyber

Summerschool on real-world crypto and privacy:
https://summerschool-croatia.cs.ru.nl