



Post-quantum cryptography

Peter Schwabe

peter@cryptojedi.org

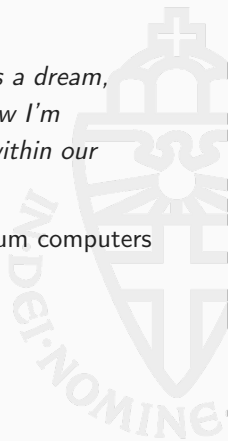
<https://cryptojedi.org>

March 22, 2017



"In the past, people have said, maybe it's 50 years away, it's a dream, maybe it'll happen sometime. I used to think it was 50. Now I'm thinking like it's 15 or a little more. It's within reach. It's within our lifetime. It's going to happen."

—Mark Ketchen (IBM), Feb. 2012, about quantum computers



Shor's algorithm (1994)

- Factor integers in polynomial time
- Compute discrete logarithms in polynomial time



Shor's algorithm (1994)

- Factor integers in polynomial time
- Compute discrete logarithms in polynomial time

Today's asymmetric crypto

- Based on factoring: RSA encryption and signatures, or
- Based on discrete logs: DH, ElGamal, DSA, ECDH, ECDSA



Digital Signatures

- Alice generates **key pair** (sk_A, pk_A) , publishes pk_A
- Alice takes document m , combines with sk_A to obtain **digital signature** σ , publishes (m, σ)
- Everybody can use pk_A to verify that
 - m was signed by Alice (by sk_A)
 - m has not been modified, since it was signed

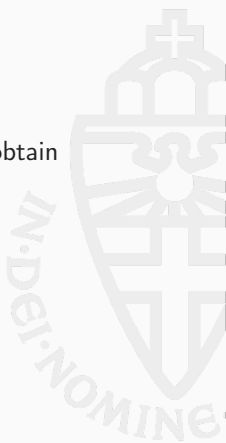
Used in TLS to verify authenticity of web servers



Key encapsulation

- Alice generates **key pair** (sk_A, pk_A) , publishes pk_A
- Bob generates random value r , combines with pk_A to obtain
 - ciphertext c , and
 - shared key k
- Alice receives c , combines with sk_A to obtain k

Used in TLS to agree on (symmetric) session keys



Digital Signatures: EdDSA

- Public key size: 32 bytes
- Signature size: 64 bytes
- Speed (ballpark): 100,000 cycles for each operation

Key encapsulation: ECDH

- Public key size: 32 bytes
- Ciphertext size: 32 bytes
- Speed (ballpark): 100,000 cycles for each operation



Hash-based signatures

- Very strong security arguments
- Small public keys (e.g., XMSS-T: 64 bytes)
- Signatures of some KB
- Signing speed: several million cycles



Hash-based signatures

- Very strong security arguments
- Small public keys (e.g., XMSS-T: 64 bytes)
- Signatures of some KB
- Signing speed: several million cycles
- **Stateful**



Hash-based signatures

- Very strong security arguments
- Small public keys (e.g., XMSS-T: 64 bytes)
- Signatures of some KB
- Signing speed: several million cycles
- **Stateful**
- SPHINCS: stateless hash-based signatures
 - 41 KB signatures
 - Signing speed: ≈ 50 Mio cycles



Hash-based signatures

- Very strong security arguments
- Small public keys (e.g., XMSS-T: 64 bytes)
- Signatures of some KB
- Signing speed: several million cycles
- **Stateful**
- SPHINCS: stateless hash-based signatures
 - 41 KB signatures
 - Signing speed: ≈ 50 Mio cycles

Alternatives

- Multivariate signatures
- Lattice-based signatures



Code-based KEMs

- Traditional McEliece/Niederreiter: good security record
- Fast for encapsulation/decapsulation
- Large public keys (> 500 KB)

SIDH

- Relatively young, needs more analysis
- Small public keys and ciphertexts (< 1 KB)
- Slow (≈ 50 Mio cycles)

Lattice-based KEMs

- Need more analysis to understand parameter choices
- Fast, reasonably small public keys and ciphertexts
- Currently very active research area



Ring-Learning-with-errors (RLWE)

- Let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$
- Let χ be an *error distribution* on \mathcal{R}_q
- Let $\mathbf{s} \in \mathcal{R}_q$ be secret
- Attacker is given pairs $(\mathbf{a}, \mathbf{as} + \mathbf{e})$ with
 - \mathbf{a} uniformly random from \mathcal{R}_q
 - \mathbf{e} sampled from χ
- Task for the attacker: find \mathbf{s}



Ring-Learning-with-errors (RLWE)

- Let $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$
- Let χ be an *error distribution* on \mathcal{R}_q
- Let $\mathbf{s} \in \mathcal{R}_q$ be secret
- Attacker is given pairs $(\mathbf{a}, \mathbf{as} + \mathbf{e})$ with
 - \mathbf{a} uniformly random from \mathcal{R}_q
 - \mathbf{e} sampled from χ
- Task for the attacker: find \mathbf{s}
- Common choice for χ : discrete Gaussian
- Common optimization for protocols: fix \mathbf{a}



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

Alice has $\mathbf{t} = \mathbf{u}\mathbf{s} = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}'\mathbf{s}$

Bob has $\mathbf{t}' = \mathbf{b}\mathbf{s}' = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}\mathbf{s}'$

- Secret and noise polynomials $\mathbf{s}, \mathbf{s}', \mathbf{e}, \mathbf{e}'$ are small
- \mathbf{t} and \mathbf{t}' are *approximately* the same



POST-QUANTUM KEY EXCHANGE



A NEW HOPE

ERDEM ALKIM

LÉO DUCAS

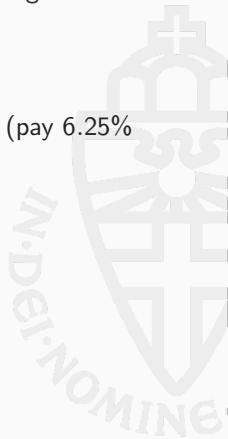
THOMAS PÖPPELMANN

PETER SCHWABE

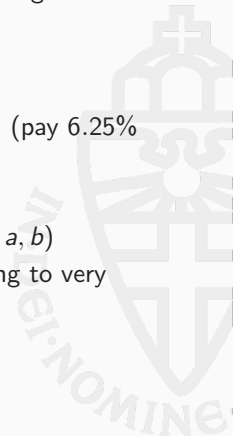
- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconciliation to go from approximate agreement to agreement
 - Originally proposed by Ding (2012)
 - Improvements by Peikert (2014)
 - More improvements in NewHope



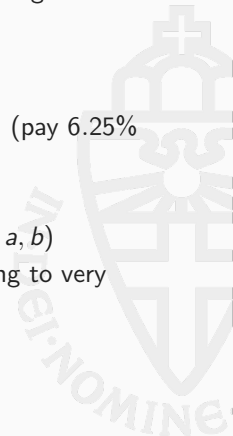
- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconciliation to go from approximate agreement to agreement
 - Originally proposed by Ding (2012)
 - Improvements by Peikert (2014)
 - More improvements in NewHope
- NewHope-Simple (2016): Scrap complex reconciliation (pay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



- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconciliation to go from approximate agreement to agreement
 - Originally proposed by Ding (2012)
 - Improvements by Peikert (2014)
 - More improvements in NewHope
- NewHope-Simple (2016): Scrap complex reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Centered binomial noise ψ_k ($\text{HW}(a) - \text{HW}(b)$ for k -bit a, b)
- Achieve ≈ 256 bits of post-quantum security according to very conservative analysis



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

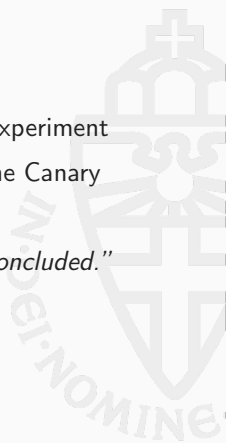


- Improve IEEE S&P 2015 results by Bos, Costello, Naehrig, Stebila (BCNS)
- Use reconciliation to go from approximate agreement to agreement
 - Originally proposed by Ding (2012)
 - Improvements by Peikert (2014)
 - More improvements in NewHope
- NewHope-Simple (2016): Scrap complex reconciliation (pay 6.25% increase in ciphertext size)
- Very conservative parameters ($n = 1024, q = 12289$)
- Centered binomial noise ψ_k ($\text{HW}(a) - \text{HW}(b)$ for k -bit a, b)
- Achieve ≈ 256 bits of post-quantum security according to very conservative analysis
- Choose a fresh parameter \mathbf{a} for every protocol run
- Higher security, shorter keys and ciphertexts, and $> 10\times$ speedup:
 - Key generation: $< 100,000$ cycles
 - Encapsulation: $< 120,000$ cycles
 - Decapsulation: $< 20,000$ cycles

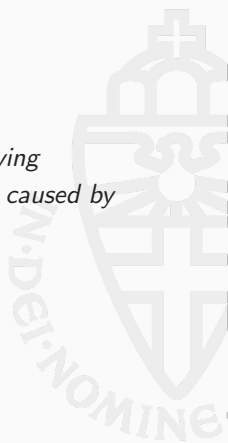
- July 7, 2016, Google announces 2-year post-quantum experiment
- NewHope+X25519 (CECPQ1) in BoringSSL for Chrome Canary
- Used in access to select Google services



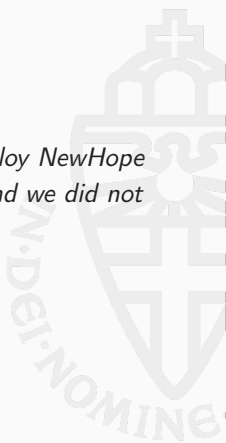
- July 7, 2016, Google announces 2-year post-quantum experiment
- NewHope+X25519 (CECPQ1) in BoringSSL for Chrome Canary
- Used in access to select Google services
- November 28, 2016: *“At this point the experiment is concluded.”*



"[...] we did not find any unexpected impediment to deploying something like NewHope. There were no reported problems caused by enabling it."



"[...] if the need arose, it would be practical to quickly deploy NewHope in TLS 1.2. (TLS 1.3 makes things a little more complex and we did not test with CECPQ1 with it.)"



“Although the median connection latency only increased by a millisecond, the latency for the slowest 5% increased by 20ms and, for the slowest 1%, by 150ms. Since NewHope is computationally inexpensive, we’re assuming that this is caused entirely by the increased message sizes. Since connection latencies compound on the web (because subresource discovery is delayed), the data requirement of NewHope is moderately expensive for people on slower connections.”

Are we done? Is the Internet safe again?



Disadvantages of NewHope

- Security analysis assumes that we have an LWE instance
- Structure of **RLWE** is ignored



Disadvantages of NewHope

- Security analysis assumes that we have an LWE instance
- Structure of **RLWE** is ignored
- Somewhat large messages ($\approx 2\text{KB}$ each way)
- Maybe overly conservative security...?



Disadvantages of NewHope

- Security analysis assumes that we have an LWE instance
- Structure of **RLWE** is ignored
- Somewhat large messages ($\approx 2\text{KB}$ each way)
- Maybe overly conservative security...?
- “Only” does ephemeral key exchange
- Must not reuse keys/noise
- No CCA security

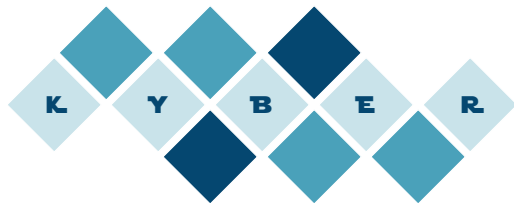


Disadvantages of NewHope

- Security analysis assumes that we have an LWE instance
- Structure of RLWE is ignored
- Somewhat large messages ($\approx 2\text{KB}$ each way)
- Maybe overly conservative security...?
- “Only” does ephemeral key exchange
- Must not reuse keys/noise
- No CCA security

Back to the drawing board!





THE KEM

Shi Bai

Eike Kiltz

John M. Schanck

Joppe Bos

Tancredè Lepoint

Peter Schwabe

Léo Ducas

Vadim Lyubashevsky

Damien Stehlé

The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)



The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)
- Less structured underlying problem: good for security



The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)
- Less structured underlying problem: good for security
- Use Targhi-Unruh CCA transform to build **CCA-secure KEM**
 - Can be used just like NewHope (but can cache keys!)
 - Can also be used for KEM-DEM to encrypt messages
 - Can be used in authenticated key exchange (without signatures)



The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)
- Less structured underlying problem: good for security
- Use Targhi-Unruh CCA transform to build **CCA-secure KEM**
 - Can be used just like NewHope (but can cache keys!)
 - Can also be used for KEM-DEM to encrypt messages
 - Can be used in authenticated key exchange (without signatures)
- Choose $d = 3, n = 256, q = 7681$ for very conservative security



The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)
- Less structured underlying problem: good for security
- Use Targhi-Unruh CCA transform to build **CCA-secure KEM**
 - Can be used just like NewHope (but can cache keys!)
 - Can also be used for KEM-DEM to encrypt messages
 - Can be used in authenticated key exchange (without signatures)
- Choose $d = 3, n = 256, q = 7681$ for very conservative security
- Public key: 1088 bytes
- Ciphertext: 1184 bytes



The design of Kyber (WiP)

- Use **Module-Lattices** and MLWE
 - RLWE: large polynomials (e.g., $n = 1024$)
 - MLWE: matrices of smaller polynomials (e.g., $n = 256$) of small dimension (e.g., $d = 3$)
- Less structured underlying problem: good for security
- Use Targhi-Unruh CCA transform to build **CCA-secure KEM**
 - Can be used just like NewHope (but can cache keys!)
 - Can also be used for KEM-DEM to encrypt messages
 - Can be used in authenticated key exchange (without signatures)
- Choose $d = 3, n = 256, q = 7681$ for very conservative security
- Public key: 1088 bytes
- Ciphertext: 1184 bytes
- Performance similar to NewHope (for sufficiently large values of “similar”)

<http://pq-crystals.org/kyber>

