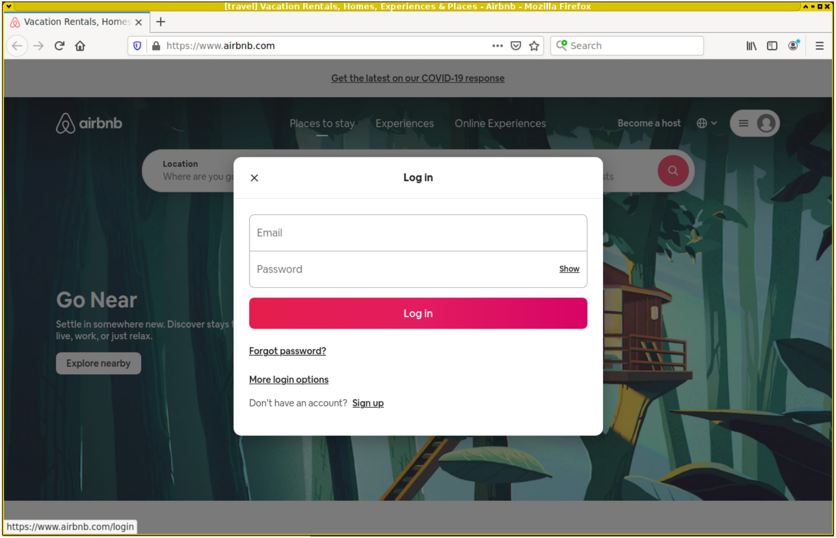
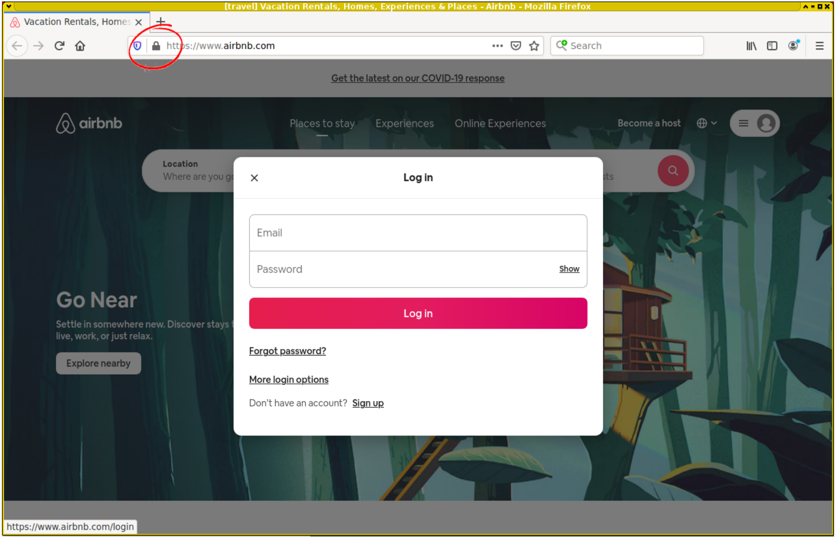
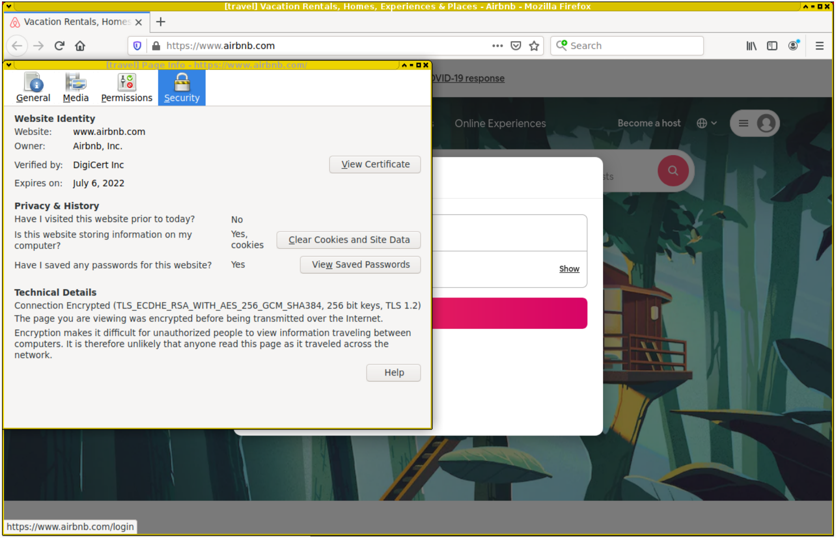
MAX PLANCK INSTITUTE
FOR SECURITY AND PRIVACY

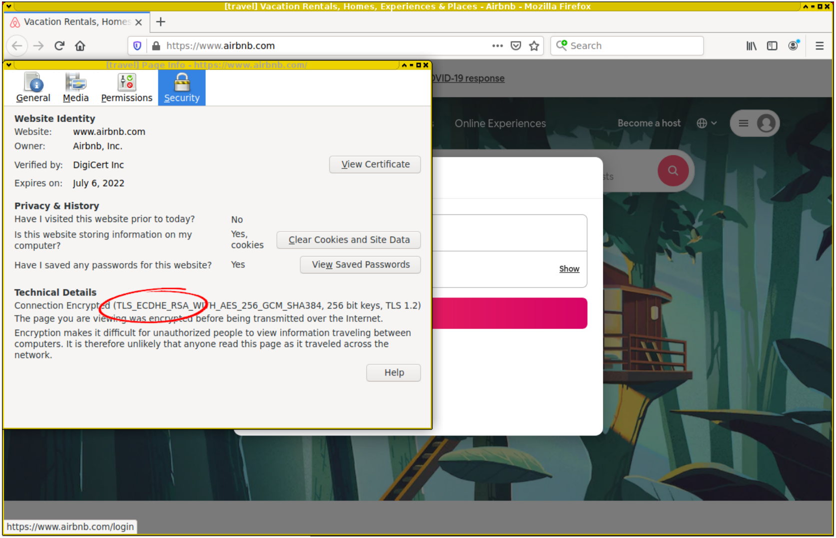
NIST PQC, Kyber, and beyond

August 10, 2022


[travel] Vacation Rentals, Homes, Experiences & Places - Airbnb - Mozilla Firefox
Vacation Rentals, Homes
https://www.airbnb.com
Search
Get the latest on our COVID-19 response
airbnb
Places to stay
Experiences
Online Experiences
Become a host
Location
Where are you g
Log in
Email
Password
Show
Log in
Forgot password?
More login options
Don't have an account? Sign up
Go Near
Settle in somewhere new. Discover stays t
live, work, or just relax.
Explore nearby
https://www.airbnb.com/login


[travel] Vacation Rentals, Homes, Experiences & Places - Airbnb - Mozilla Firefox
Vacation Rentals, Homes
https://www.airbnb.com
Search
Get the latest on our COVID-19 response
airbnb
Places to stay
Experiences
Online Experiences
Become a host
Location
Where are you g
Log in
Email
Password
Show
Log in
Forgot password?
More login options
Don't have an account? Sign up
Go Near
Settle in somewhere new. Discover stays t
live, work, or just relax.
Explore nearby
https://www.airbnb.com/login


[travel] Vacation Rentals, Homes, Experiences & Places - Airbnb - Mozilla Firefox
Vacation Rentals, Homes
https://www.airbnb.com
Search
[travel] Page Info - https://www.airbnb.com/
General
Media
Permissions
Security
Website Identity
Website: www.airbnb.com
Owner: Airbnb, Inc.
Verified by: DigiCert Inc
View Certificate
Expires on: July 6, 2022
Privacy & History
Have I visited this website prior to today? No
Is this website storing information on my computer? Yes, cookies
Clear Cookies and Site Data
Have I saved any passwords for this website? Yes
View Saved Passwords
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
Help
VID-19 response
Online Experiences
Become a host
Show
https://www.airbnb.com/login


[travel] Vacation Rentals, Homes, Experiences & Places - Airbnb - Mozilla Firefox
Vacation Rentals, Homes
https://www.airbnb.com
Search
[travel] Page Info - https://www.airbnb.com/
General
Media
Permissions
Security
Website Identity
Website: www.airbnb.com
Owner: Airbnb, Inc.
Verified by: DigiCert Inc
View Certificate
Expires on: July 6, 2022
Privacy & History
Have I visited this website prior to today? No
Is this website storing information on my computer? Yes, cookies
Clear Cookies and Site Data
Have I saved any passwords for this website? Yes
View Saved Passwords
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384, 256 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
Help
OVID-19 response
Online Experiences
Become a host
Show
https://www.airbnb.com/login

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.

"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

Post-quantum crypto

Definition

Post-quantum crypto is (asymmetric) crypto that resists attacks using classical *and quantum* computers.

Post-quantum crypto

Definition

Post-quantum crypto is (asymmetric) crypto that resists attacks using classical *and quantum* computers.

5 main directions

- Lattice-based crypto (PKE and Sigs)
- Code-based crypto (mainly PKE)
- Multivariate-based crypto (mainly Sigs)
- Hash-based signatures (only Sigs)
- Isogeny-based crypto (so far, mainly PKE)

Post-quantum crypto

Definition

Post-quantum crypto is (asymmetric) crypto that resists attacks using classical *and quantum* computers.

5 main directions

- Lattice-based crypto (PKE and Sigs)
- Code-based crypto (mainly PKE)
- Multivariate-based crypto (mainly Sigs)
- Hash-based signatures (only Sigs)
- Isogeny-based crypto (so far, mainly PKE)

The NIST PQC "not-a-competition"

- Inspired by two earlier NIST crypto competitions:
  - AES, running from 1997 to 2000
  - SHA3, running from 2007 to 2012

The NIST PQC “not-a-competition”

- Inspired by two earlier NIST crypto competitions:
  - AES, running from 1997 to 2000
  - SHA3, running from 2007 to 2012
- Approach: NIST specifies criteria, everybody is welcome to submit proposals
- Selection through an open process and multiple rounds
- Actual decisions are being made by NIST

The NIST PQC “not-a-competition”

- Inspired by two earlier NIST crypto competitions:
    - AES, running from 1997 to 2000
    - SHA3, running from 2007 to 2012
- Approach: NIST specifies criteria, everybody is welcome to submit proposals
- Selection through an open process and multiple rounds
- Actual decisions are being made by NIST
- PQC project:
    - Announcement: Feb 2016
    - Call for proposals: Dec 2016 (based on community input)
    - Deadline for submissions: Nov 2017

The NIST competition: initial overview

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.

The NIST competition: Jan 2019

- Announcement planned at Real-World Crypto 2019

The NIST competition: Jan 2019

- Announcement planned at Real-World Crypto 2019
- Due to US government lockdown slightly later

The NIST competition: Jan 2019

- Announcement planned at Real-World Crypto 2019
- Due to US government lockdown slightly later

Encryption / Key agreement

- 9 lattice-based
- 7 code-based
- 1 isogeny-based

The NIST competition: Jan 2019

- Announcement planned at Real-World Crypto 2019
- Due to US government lockdown slightly later

Encryption / Key agreement

- 9 lattice-based
- 7 code-based
- 1 isogeny-based

Signature schemes

- 3 lattice-based
- 2 symmetric-crypto based
- 4 MQ-based

The NIST competition: Jul 2020

- Announcement planned for June 2020

The NIST competition: Jul 2020

- Announcement planned for June 2020
- Due to pandemic (?) slightly later

The NIST competition: Jul 2020

- Announcement planned for June 2020
- Due to pandemic (?) slightly later

Finalists

- 4 key-agreement schemes
  - 3 lattice-based
  - 1 code-based
- 3 signature schemes
  - 2 lattice-based
  - 1 MQ-based

The NIST competition: Jul 2020

- Announcement planned for June 2020
- Due to pandemic (?) slightly later

Finalists

- 4 key-agreement schemes
  - 3 lattice-based
  - 1 code-based
- 3 signature schemes
  - 2 lattice-based
  - 1 MQ-based

Alternate schemes

- 5 key-agreement schemes
  - 2 lattice-based
  - 2 code-based
  - 1 isogeny-based
- 3 signature schemes
  - 2 symmetric-crypto based
  - 1 MQ-based

The NIST competition: Aug 2022

- Announcement planned for March 2022

The NIST competition: Aug 2022

- Announcement planned for March 2022
- Due to ??? announcement in July 2022

The NIST competition: Aug 2022

- Announcement planned for March 2022
- Due to ??? announcement in July 2022

4 schemes selected for standardization

- CRYSTALS-Kyber: lattice-based key agreement
- CRYSTALS-Dilithium: lattice-based signature
- Falcon: lattice-based signature
- SPHINCS$^+$: hash-based signature

The NIST competition: Aug 2022

- Announcement planned for March 2022
- Due to ??? announcement in July 2022

4 schemes selected for standardization

- CRYSTALS-Kyber: lattice-based key agreement
- CRYSTALS-Dilithium: lattice-based signature
- Falcon: lattice-based signature
- SPHINCS^{+}: hash-based signature

4 schemes advanced to round 4

- Classic McEliece: code-based key agreement
- BIKE: code-based key agreement
- HQC: code-based key agreement
- SIKE: isogeny-based key agreement

The NIST competition: Aug 2022

- Announcement planned for March 2022
- Due to ??? announcement in July 2022

4 schemes selected for standardization

- CRYSTALS-Kyber: lattice-based key agreement
- CRYSTALS-Dilithium: lattice-based signature
- Falcon: lattice-based signature
- SPHINCS$^+$: hash-based signature

4 schemes advanced to round 4

- Classic McEliece: code-based key agreement
- BIKE: code-based key agreement
- HQC: code-based key agreement
- SIKE: isogeny-based key agreement
- **Additionally:** call for more signature proposals

What now?

- Standards ready “by 2024”
- Time to start upgrading systems!

What now?

- Standards ready "by 2024"
- Time to start upgrading systems!

Store now, decrypt later

- Urgency for key agreement (confidentiality)
- Need PQC **now** for long-term security

What now?

- Standards ready "by 2024"
- Time to start upgrading systems!

Store now, decrypt later

- Urgency for key agreement (confidentiality)
- Need PQC **now** for long-term security

Let's understand Kyber and what it means to use it.

A long time ago (2015) in a galaxy far, far away (Šibenik, Croatia)....

What is a Key Encapsulation Mechanism (KEM)?

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

Ring learning with errors (RLWE)

- Given $\mathbf{a}$, uniformly random
- Given "noise distribution" χ
- Given samples $\mathbf{as} + \mathbf{e}$, with $\mathbf{e} \leftarrow \chi$

Ring learning with errors (RLWE)

- Given $\mathbf{a}$, uniformly random
- 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

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Short answer

In $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned} \mathbf{a} + \mathbf{b} &= 10X^3 + 9X^2 + 2X + 5 \\ &= 3X^3 + 2X^2 + 2X + 5 \end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned} \mathbf{a} - \mathbf{b} &= -2X^3 + X^2 + 2X - 1 \\ &= 5X^3 + X^2 + 2X + 6 \end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\ & 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned}\mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\ & 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \\ = \quad & 24X^6 + 46X^5 + 32X^4 + 32X^3 + 23X^2 + 6\end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned}
\mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\
& 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \\
= \quad & 24X^6 + 46X^5 + 32X^4 + 32X^3 + 23X^2 + 6 \\
= \quad & 3X^6 + 4X^5 + 4X^4 + 4X^3 + 2X^2 + 6
\end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \dots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned}
\mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\
& 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \\
= \quad & 24X^6 + 46X^5 + 32X^4 + 32X^3 + 23X^2 + 6 \\
= \quad & 3X^6 + 4X^5 + 4X^4 + 4X^3 + 2X^2 + 6 \\
= \quad & -3X^2 - 4X - 4 + 4X^3 + 2X^2 + 6
\end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$\begin{aligned}
\mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\
& 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \\
= \quad & 24X^6 + 46X^5 + 32X^4 + 32X^3 + 23X^2 + 6 \\
= \quad & 3X^6 + 4X^5 + 4X^4 + 4X^3 + 2X^2 + 6 \\
= \quad & -3X^2 - 4X - 4 + 4X^3 + 2X^2 + 6 \\
= \quad & -X^2 - 4X + 4X^3 + 2
\end{aligned}$$

Where do $\mathbf{a}$, $\mathbf{e}$, and $\mathbf{s}$ live?

Longer answer

Polynomials with n coefficients, each coefficient in $\{0, \ldots, q-1\}$

Arithmetic uses reduction modulo q and modulo $(X^n + 1)$

Example

Let $q = 7$ and $n = 4$.

Let $\mathbf{a} = (4X^3 + 5X^2 + 2X + 2)$ and $\mathbf{b} = (6X^3 + 4X^2 + 3)$

$$
\begin{aligned}
\mathbf{a} \cdot \mathbf{b} = \quad & 24X^6 + 16X^5 + 12X^3 + 30X^5 + 20X^4 + 15X^2 + \\
& 12X^4 + 8X^3 + 6X + 12X^3 + 8X^2 + 6 \\
= \quad & 24X^6 + 46X^5 + 32X^4 + 32X^3 + 23X^2 + 6 \\
= \quad & 3X^6 + 4X^5 + 4X^4 + 4X^3 + 2X^2 + 6 \\
= \quad & -3X^2 - 4X - 4 + 4X^3 + 2X^2 + 6 \\
= \quad & -X^2 - 4X + 4X^3 + 2 \\
= \quad & 4X^3 + 6X^2 + 3X + 2
\end{aligned}
$$

How to build a KEM: the basic idea

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

$$\begin{array}{llll} \text{Alice has} & \mathbf{v} & = \mathbf{u}\mathbf{s} & = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}'\mathbf{s} \\ \text{Bob has} & \mathbf{v}' & = \mathbf{b}\mathbf{s}' & = \mathbf{a}\mathbf{s}\mathbf{s}' + \mathbf{e}\mathbf{s}' \end{array}$$

- 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: the construction

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: the construction

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

How to build a KEM: the construction

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

How to build a KEM: the construction

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

How to build a KEM: the construction

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: the construction

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: the construction

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

This is LPR encryption, written as KEM (except for generation of $\mathbf{a}$)

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

KYBER

THE KEM

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

Two more steps to Kyber

MLWE instead of RLWE

IND-CCA2 Security

Two more steps to Kyber

MLWE instead of RLWE

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

IND-CCA2 Security

Two more steps 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 $\mathrm{HW}(x) - \mathrm{HW}(y)$ 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>
$\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{KeyGen}()$	$\xrightarrow{\mathsf{pk}}$	$x \leftarrow \{0, \ldots, 255\}^{32}$ $k, \mathsf{coins} \leftarrow \mathsf{SHA3\text{-}512}(x)$
	$\xleftarrow{\mathsf{ct}}$	$\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x, \mathsf{coins})$
<u>Decaps((sk, pk), ct):</u> $x' \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$ $k', coins' \leftarrow \mathsf{SHA3\text{-}512}(x')$ $\mathsf{ct}' \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x', \mathsf{coins}')$ **verify if** $\mathsf{ct} = \mathsf{ct}'$		

From passive to CCA security

The Fujisaki-Okamoto Transform

Alice (Server)		Bob (Client)
Gen():		Encaps(pk):
$\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{KeyGen}()$	$\xrightarrow{\mathsf{pk}}$	$x \leftarrow \{0, \ldots, 255\}^{32}$ $k, \mathsf{coins} \leftarrow \mathsf{SHA3\text{-}512}(x)$
	$\xleftarrow{\mathsf{ct}}$	$\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x, \mathsf{coins})$
Decaps((sk, pk), ct): $x' \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$ $k', coins' \leftarrow \mathsf{SHA3\text{-}512}(x')$ $\mathsf{ct}' \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x', \mathsf{coins}')$ **verify if** $\mathsf{ct} = \mathsf{ct}'$		

Additionally in Kyber:

- Hash the (hash of the) public key into x
  - Multi-target protection (for coins)
  - Turn into contributory KEM
- Hash the (hash of the) ciphertext into the final key

Kyber for Engineers, the baseline

Key exchange today: ECDH

- Key-pair generation $\approx 125,000$ Comet Lake cycles
- Shared-key computation $\approx 125,000$ Comet Lake cycles
- Public keys have 32 bytes

Kyber for Engineers, part I: A KEM is not DH!

Alice		Bob
$A \leftarrow g^a$		$B \leftarrow g^b$
	$\xrightarrow{\quad A \quad}$	
	$\xleftarrow{\quad B \quad}$	
$K \leftarrow B^a = (g^b)^a = g^{ab}$		$K \leftarrow A^b = (g^a)^b = g^{ab}$

Kyber for Engineers, part I: A KEM is not DH!

Alice		Bob
$A \leftarrow g^a$		$B \leftarrow g^b$
	$\xleftarrow{\quad B \quad}$	
	$\xrightarrow{\quad A \quad}$	
$K \leftarrow B^a = (g^b)^a = g^{ab}$		$K \leftarrow A^b = (g^a)^b = g^{ab}$

Kyber for Engineers, part I: A KEM is not DH!

Initiator | **Responder**

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KEM.Gen}$

$\xrightarrow{\hspace{6em}\mathsf{pk}\hspace{6em}}$

$(\mathsf{ct}, K) \leftarrow \mathsf{KEM.Enc}(\mathsf{pk})$

$\xleftarrow{\hspace{6em}\mathsf{ct}\hspace{6em}}$

$K \leftarrow \mathsf{KEM.Dec}(\mathsf{ct}, \mathsf{sk})$

Kyber for Engineers, part II: Performance

Kyber768 (NIST Security level 3)

- Key-pair generation $\approx 40,000$ Comet Lake cycles
- Encapsulation $\approx 55,000$ Comet Lake cycles
- Decapsulation $\approx 45,000$ Comet Lake cycles

Kyber for Engineers, part II: Performance

Kyber768 (NIST Security level 3)

- Key-pair generation $\approx 40,000$ Comet Lake cycles
- Encapsulation $\approx 55,000$ Comet Lake cycles
- Decapsulation $\approx 45,000$ Comet Lake cycles
- Public keys have 1184 bytes
- Ciphertexts have 1088 bytes

Kyber for Engineers, part II: Performance

Kyber768 (NIST Security level 3)

- Key-pair generation $\approx 40,000$ Comet Lake cycles
- Encapsulation $\approx 55,000$ Comet Lake cycles
- Decapsulation $\approx 45,000$ Comet Lake cycles
- Public keys have 1184 bytes
- Ciphertexts have 1088 bytes
- Cycles are dominated by Keccak!

- FO-transform: hide if decryption succeeded
- Use full re-encryption to do this

- FO-transform: hide if decryption succeeded
- Use full re-encryption to do this
- Long computation, one bit of information
- Very hard to protect against SCA/FI

Recommendations

- Start playing with Kyber
- Assume that details may still change

Recommendations

- Start playing with Kyber
- Assume that details may still change
- Always combine with pre-quantum crypto (hybrid KEMs)
- Use Kyber768 (or Kyber1024)

. . . and beyond

- Will need to migrate to PQC in the next 5–10 years
- Use this to migrate to **high-assurance implementations**!
    - Computer-verified correctness
    - Computer-verified security
    - Computer-verified implementation security

https://formosa-crypto.org

Online references

- 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:
  https://pq-crystals.org/kyber
  https://github.com/pq-crystals/kyber