Quantum Secure Signal Protocol

Extended Tripple Diffie Hellman

Emily Ehlert

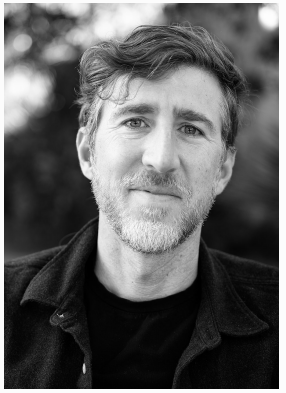
13. Oktober 2023

0. Structure

1. Introduction / History
2. Properties
3. Extended Tripple Diffie-Hellman Handshake (X3DH)
4. Quantum Secure X3DH
    1. Using SIDH
    2. Using KEMs
    3. CRYSTALS-Kyber
    4. Tutanota’s Implementation
    5. Signal’s PQXDH
5. Summary

1. Introduction / History

Perrin



Marlinspike

- End to end encrypted messaging protocol
- Uses the Extended Tripple Diffie-Hellman key exchange and Double-Ratchet algorithm
- Developed in 2013 by Trevor Perrin and Moxie Marlinspike at Open Whispher Systems
- In 2016 name was changed from TextSecure to Signal Protocol
- Used in a variety of apps like Signal, WhatsApp, Google RTS Messaging, Facebook Messenger, ...

Signal

Logo of the Signal App

2. Properties

- **Confidentiality, Integrity and Authentication**
- **Forward secrecy**
  - Compromising all key material does not enable decryption of previously encrypted data.
- **Backward / post-compromise / future secrecy**
  - Compromising all key material does not enable decryption of succeeding encrypted data.
- **Plausible deniability**
  - **Message repudiation**
    - Given a conversation transcript and all cryptographic keys, there is no evidence that a given message was authored by any particular user.
  - **Message unlinkability**
    - If a judge is convinced that a participant authored one message in the conversation, this does not provide evidence that they authored other messages.

Definition taken from Unger et al. 2015, pp. 232-249

3. Function

3.1 Extended Tripple Diffie-Hellman (X3DH)

- Why do we need X3DH and can’t just use Diffie-Hellman (DH) by itself?
- X3DH requires always online server to work
- Assume Alice wants to write Bob
- Protocol consists of three phase
  1) Key Upload
  2) Generating Handshake
  3) Receiving Handshake

3.1 Extended Tripple Diffie-Hellman (X3DH)

1. Key Upload

- Public keys uploaded to the server
- Identity key ${}^{\text{pre}}\text{IK}_\text{X}$
  - Used for authentication, can be checked before or afterwards
- (Signed) Prekey ${}^{\text{pre}}\text{SPK}_\text{X}$
  - Used for forward secrecy
- Prekey signature $\text{Sig}({}^{\text{pre}}\text{IK}_\text{X}, \text{Encode}({}^{\text{pre}}\text{SPK}_\text{X}))$
  - Used for authenticity verification
- Set of one-time prekeys ${}^{\text{pre}}\text{OPK}^{i}_\text{X}$
  - Used for better forward secrecy

3.1 Extended Tripple Diffie-Hellman (X3DH)

2. Sending Message

- Alice downloads Bob $({}^{\text{pre}}\text{IK}_\text{B},\ {}^{\text{pre}}\text{SPK}_\text{B},\ \text{Sig}({}^{\text{pre}}\text{IK}_\text{B}, \text{Encode}({}^{\text{pre}}\text{SPK}_\text{B})))$ and if available ${}^{\text{pre}}\text{OPK}_\text{B}$
- Alice verifies signature of signed key
- Generate ephemeral (temporary) key pair ${}^{\text{pre}}\text{EK}_\text{A}$
- Performes 3 (or 4) DH calculations
  - ${}^{\text{pre}}k_1 = \text{DH}({}^{\text{pre}}\text{IK}_{\text{A;priv}}, {}^{\text{pre}}\,\text{SPK}_{\text{B;pub}})$
  - ${}^{\text{pre}}k_2 = \text{DH}({}^{\text{pre}}\text{EK}_{\text{A;priv}}, {}^{\text{pre}}\,\text{IK}_{\text{B;pub}})$
  - ${}^{\text{pre}}k_3 = \text{DH}({}^{\text{pre}}\text{EK}_{\text{A;priv}}, {}^{\text{pre}}\,\text{SPK}_{\text{B;pub}})$
  - $({}^{\text{pre}}k_4 = \text{DH}({}^{\text{pre}}\text{EK}_{\text{A;priv}}, {}^{\text{pre}}\,\text{OPK}_{\text{B;pub}}))$
- Calculate session key
  - $\text{SK} = \text{KDF}({}^{\text{pre}}\text{k}_1 \,||{}^{\text{pre}}\text{k}_2 \,||\ {}^{\text{pre}}\text{k}_3\ (||\ {}^{\text{pre}}\text{k}_4))$

3.1 Extended Tripple Diffie-Hellman (X3DH)

2. Sending Message

- $^{\text{pre}}k_1 = \text{DH}(^{\text{pre}}\text{IK}_{\text{A;priv}}, ^{\text{pre}}\,\text{SPK}_{\text{B;pub}})$

  $^{\text{pre}}k_2 = \text{DH}(^{\text{pre}}\text{EK}_{\text{A;priv}}, ^{\text{pre}}\,\text{IK}_{\text{B;pub}})$
  - Used for bidirectional authentication
  - Two calculations used to provide some forward secrecy
- $^{\text{pre}}k_3 = \text{DH}(^{\text{pre}}\text{EK}_{\text{A;priv}}, ^{\text{pre}}\,\text{SPK}_{\text{B;pub}})$

  $(^{\text{pre}}k_4 = \text{DH}(^{\text{pre}}\text{EK}_{\text{A;priv}}, ^{\text{pre}}\,\text{OPK}_{\text{B;pub}}))$
  - After Handshake complete can not calculate DH_3 or DH_4

3.1 Extended Tripple Diffie-Hellman (X3DH)

2. Sending Message

- Alice calculates Associate Data AD
  - $\mathrm{AD} = \mathrm{Encode}({}^{\mathrm{pre}}\mathrm{IK}_{\mathrm{A;pub}}, {}^{\mathrm{pre}}\mathrm{IK}_{\mathrm{B;pub}})$
- Alice sends first message containing the following
  - $({}^{\mathrm{pre}}\mathrm{IK}_{\mathrm{A;pub}}, {}^{\mathrm{pre}}\mathrm{EK}_{\mathrm{A;pub}})$
  - Identifications of used ${}^{\mathrm{pre}}\mathrm{SPK}_{\mathrm{B}}$ und ${}^{\mathrm{pre}}\mathrm{OPK}_{\mathrm{B}}$
  - Ciphertext encrypted with an AEAD encryption scheme like AES GCM using SK as key (or a derivation of SK) and AD as associated data

3.1 Extended Tripple Diffie-Hellman (X3DH)

3. Receiving Message

- Bob receives message
- Extracts keys and loads own private keys
- Performs DH calculations to receive same SK
- Decrypts ciphertext

4. Quantum Secure X3DH

- Original X3DH uses eliptic curve DH
  - Can be broken effiently with Shor’s quantum algorithm
  - Quantum computers capable enough expected in the 2030s
  - Need to secure communication now
- In 2017 NIST started a Post-Quantum Cryptography Standardization program for KEMs and Signature Schemes
- One very interesting candidate SIKE / SIDH had the possibility of being a great replacement for DH
- Research into making Signal quantum secure was oftentimes built on top of SIDH
- However broken in 2022 on a classical computer
- Now research focusses on using Key Encapsulation Mechanisms (KEMs)

4. Quantum Secure X3DH

1. Using SIDH

- SIDH allows creating a shared secret similar to DH
- Based on Supersingular Isogeny
- Even though SIDH is not secure, looking into possible implementations can give an insight in quantum secure protocol design
- One of the last Signal SIDH publication “Post-Quantum Signal Key Agreement with SIDH” by Samuel Dobson and Steven D. Galbraith was published in March 2022
- Employed a Zero Knowledge Proofs to prevent known adaptive attacks against SIDH

4. Quantum Secure X3DH

1. Using SIDH

- Zero Knowledge Proofs quite complex, thus only used for Identity Keys
- Ephemeral Keys use the Fujisaki–Okamoto transformation to prevent adaptive attacks as well

$$^{\text{post}}k_1 = \text{SIDH}(^{\text{post}}\text{IK}_{\text{A;priv}}, ^{\text{post}}\,\text{IK}_{\text{B;pub}})$$

$$^{\text{post}}k_2 = \text{SIDH}(^{\text{post}}\text{EK}_{\text{A;priv}}, ^{\text{post}}\,\text{IK}_{\text{B;pub}})$$

$$^{\text{post}}k_3 = \text{SIDH}(^{\text{post}}\text{EK}_{\text{A;priv}}, ^{\text{post}}\,\text{SPK}_{\text{B;pub}})$$

$$(^{\text{post}}k_4 = \text{SIDH}(^{\text{post}}\text{EK}_{\text{A;priv}}, ^{\text{post}}\,\text{OPK}_{\text{B;pub}}))$$

$$\text{SK} = \text{KDF}(^{\text{post}}\text{k}_1||^{\text{post}}\text{k}_2||^{\text{post}}\text{k}_3(||^{\text{post}}\text{k}_4))$$

$$\pi = s \oplus H_2(^{\text{post}}\text{k}_1) \oplus H_2(^{\text{post}}\text{k}_2) \oplus H_2(^{\text{post}}\text{k}_3)(\oplus H_2(^{\text{post}}\text{k}_4))$$

- Bob needs to verify π by recovering s
- Protocol relies solely on SIDH

4. Quantum Secure X3DH

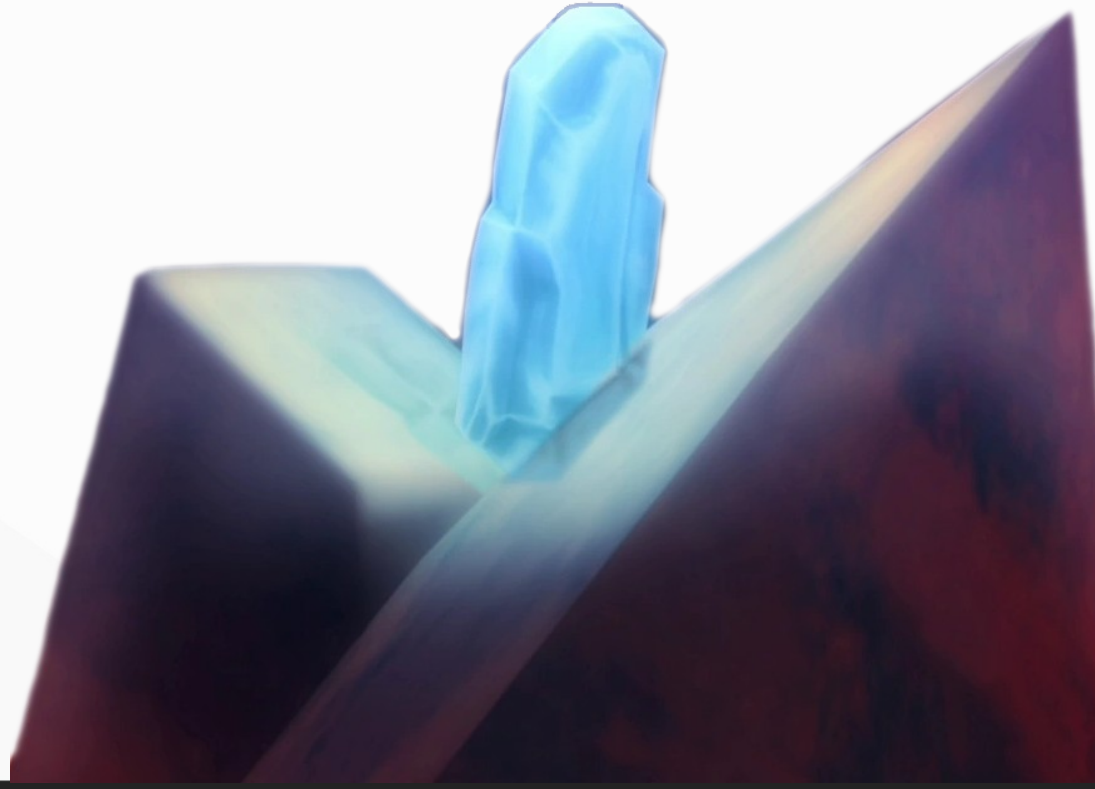
2. Using KEMS

- Key Encapsulation Mechanisms allow a key to be encapsulated using a public key
- The private key can then recover key through decapsulation
- KEMs in the NIST competition do not allow the user to choose the key being encapsulated
- Makes it difficult to utilize it as DH replacement with (semi) static keys
- One one KEM as already announced as winner by NIST: CRYSTAL-Kyber

4. Quantum Secure X3DH

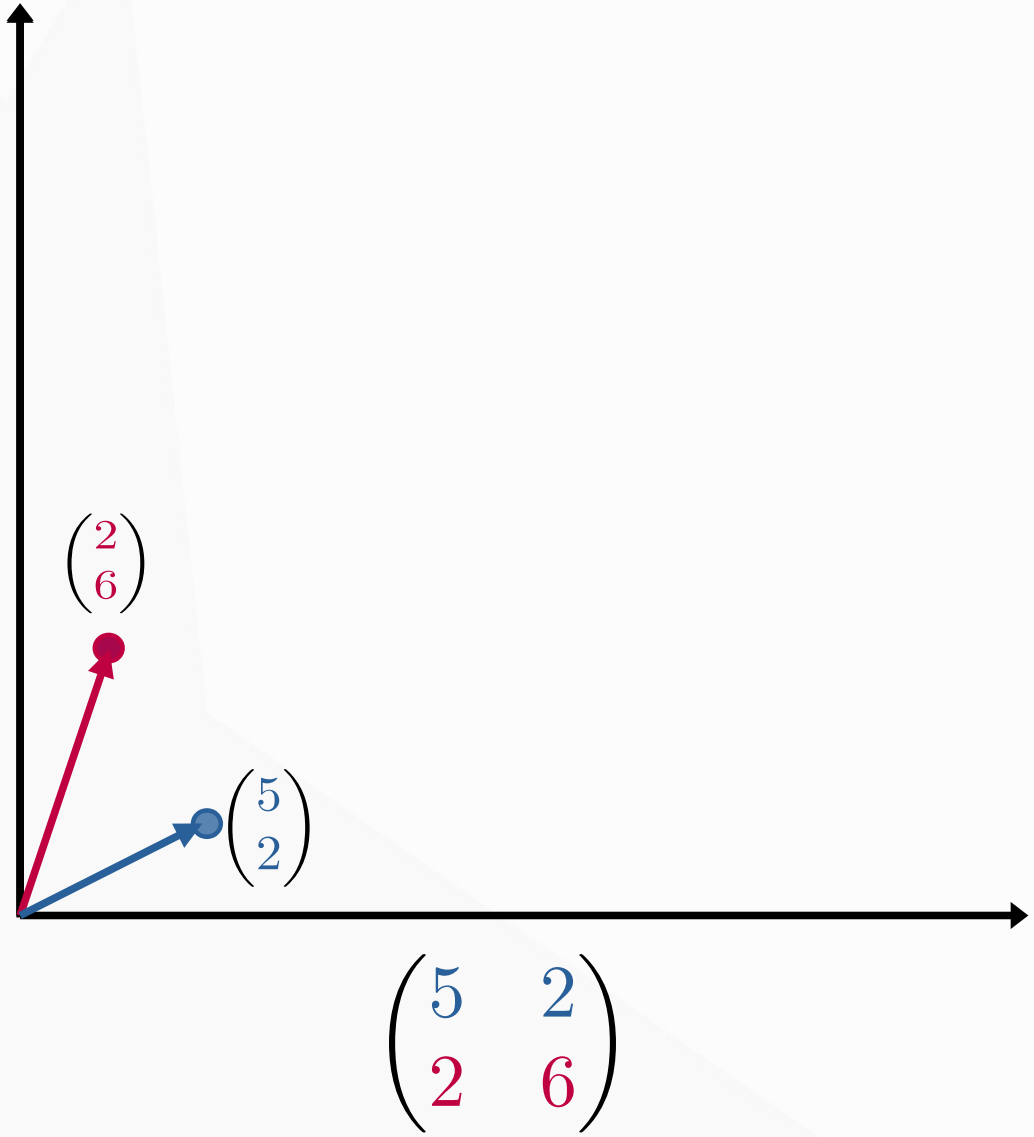
3. CRYSTALS-Kyber

- International research effort based on work published by Odeg Regev
- Kyber belongs to Cryptographic Suite for Algebraic Lattices (CRYSTALS)
- Lattice-based approach
  - Other approach is code-based
- What are Lattices?



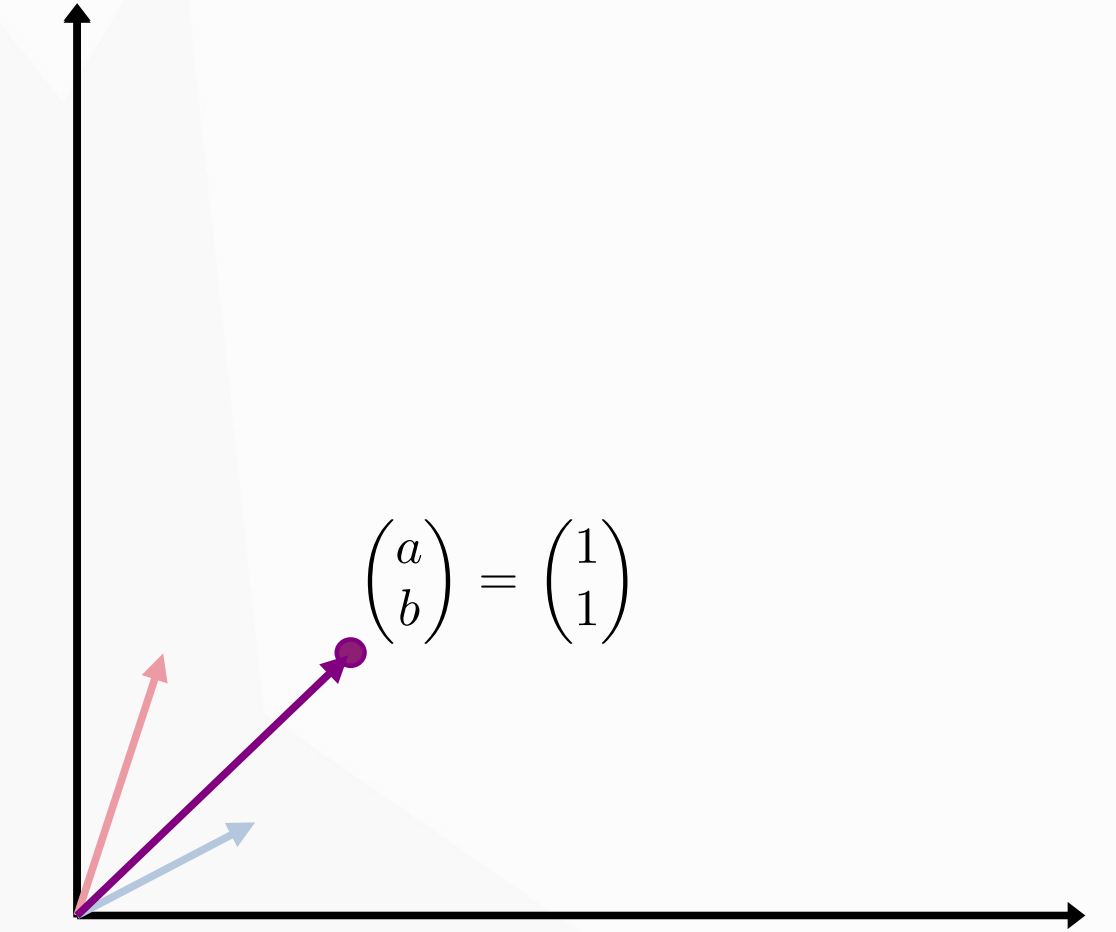
4. Quantum Secure X3DH

3. CRYSTALS-Kyber



4. Quantum Secure X3DH

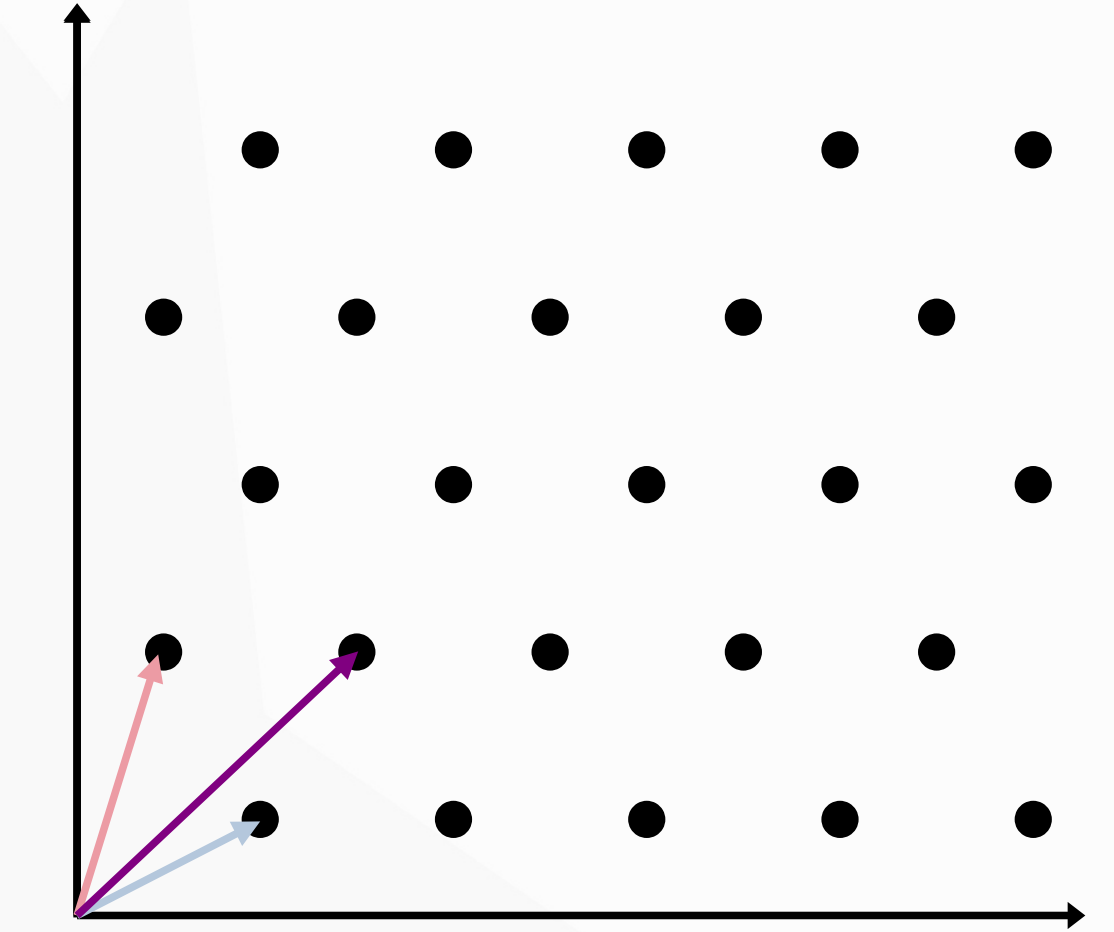
3. CRYSTALS-Kyber



$$\begin{pmatrix} 5 & 2 \\ 2 & 6 \end{pmatrix} \cdot \begin{pmatrix} a \\ b \end{pmatrix} = a \cdot \begin{pmatrix} 5 \\ 2 \end{pmatrix} + b \cdot \begin{pmatrix} 2 \\ 6 \end{pmatrix}$$

4. Quantum Secure X3DH

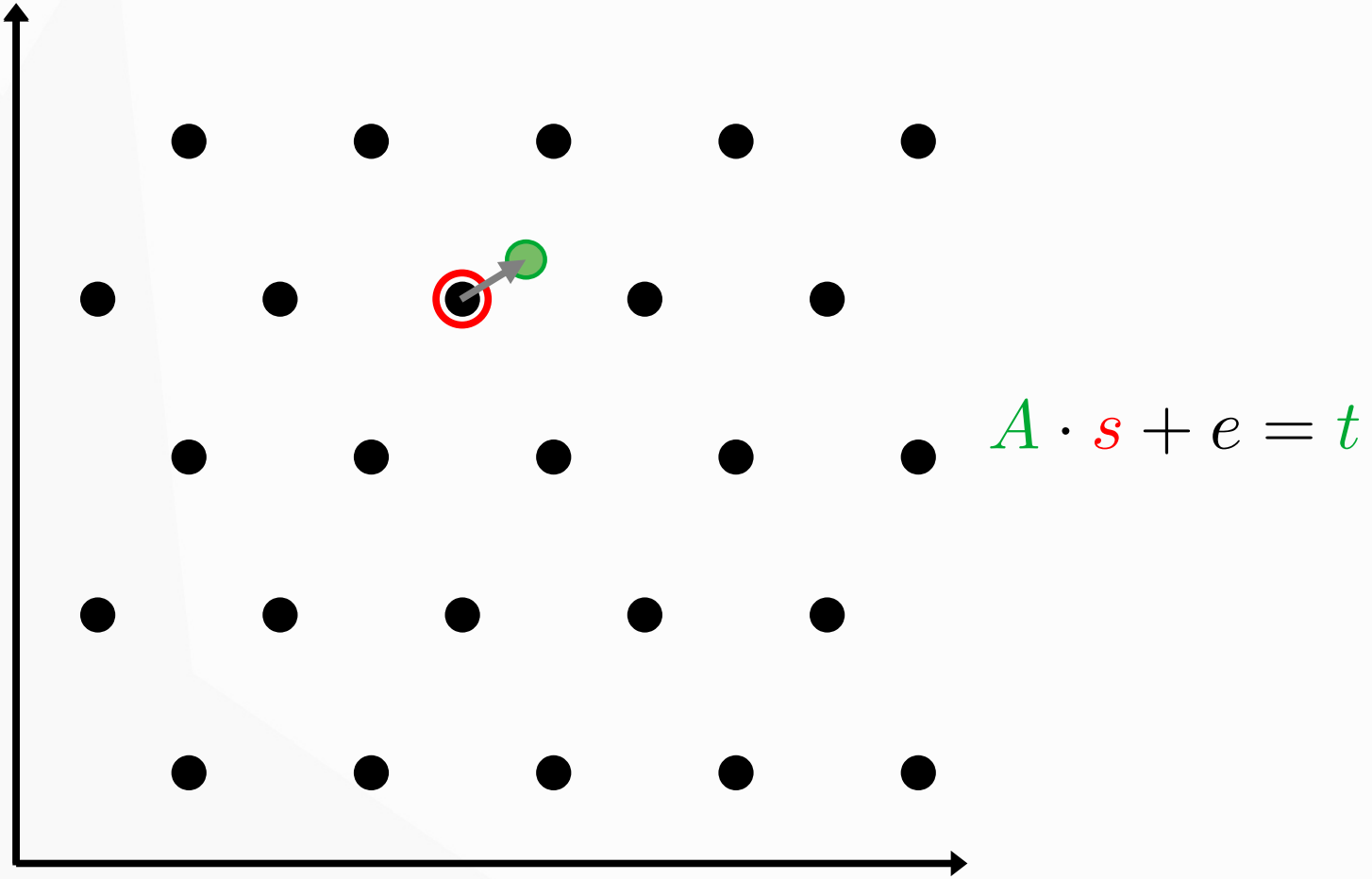
3. CRYSTALS-Kyber



$$\begin{pmatrix} 5 & 2 \\ 2 & 6 \end{pmatrix} \cdot \begin{pmatrix} a \\ b \end{pmatrix} = a \cdot \begin{pmatrix} 5 \\ 2 \end{pmatrix} + b \cdot \begin{pmatrix} 2 \\ 6 \end{pmatrix}$$

4. Quantum Secure X3DH

3. CRYSTALS-Kyber



In a known lattice defined by matrix **A** it is NP-hard given a public point **t** to find the closest vector **s**, the secret, on the lattice. (Closest Vector Problem)

4. Quantum Secure X3DH

3. CRYSTALS-Kyber

- Kyber does not use scalars but polynoms

$$A \cdot s + e = t$$

$$\begin{pmatrix} x+1 & 3x^2-4 \\ x^2-1 & 2x^2-2x \end{pmatrix} \cdot \begin{pmatrix} 3x^2+2x \\ -x-4 \end{pmatrix} + \begin{pmatrix} -6x-10 & 12x^2+3 \end{pmatrix} = \begin{pmatrix} -7x^2+6 & 4x \end{pmatrix}$$

- Kyber uses polynoms with degree 256
  - Allows 256 bit to be encrypted
- Furthermore depending on security level: 2, 3 or 4 dimensional vectors
- Kyber works in a modulo space of size 3329

4. Quantum Secure X3DH

3. CRYSTALS-Kyber

- Given a message m and a public key (A, t)
- Convert m to binary to polynomial (remove zero terms)
- Encrypt a message:
  - Scale m with large factor resulting
  - Calculate ciphertext
    - $v = t \cdot e_1 + e_2 + m$
    - $u = A \cdot e_1 + e_3$
- Decrypt a message with private key **s**
  - $d = v - su = t \cdot e_1 + e_2 + m - s(A \cdot e_1 + e_3)$ with $A \cdot s + e = t$
  - $d = (A \cdot s + e_4) \cdot e_1 + e_2 + m - s(A \cdot e_1 + e_3)$
  - $d = A \cdot s \cdot e_1 + e_4 \cdot e_1 + e_2 + m - s(A \cdot e_1 + e_3)$
  - $d = m + Ase_1 - Ase_1 + e_4e_1 + e_2 - se_3 \approx m$

4. Quantum Secure X3DH

4. Tutanota's Implementation

- Tutanota is an German open-source email service
- Provide E2EE for mails send between customers
- Proposed a modified Signal protocol enabling quantum-secure communication
- Currently only a prototype, does not appear to be actively used
- Uses Kyber-786 as KEM and Dilithium 1280x1024 as quantum-secure signature scheme
- Hybrid scheme, combines conventional and quantum-secure cryptography
- Only consider X3DH replacement

Tutanota®

4. Quantum Secure X3DH
4. Tutanota's Implementation

- Uses all components of traditional X3DH
- Additional each participant has PQ-secure
  - one-time prekeys ${}^{post}\mathrm{OPK}_X^i$
  - signed semi-static prekey ${}^{post}\mathrm{SPK}_X$
  - long-term identity keys ${}^{post}\mathrm{IK}_X$
- When Alice downloads key bundle from server verify signatures
  - $\mathrm{verify}({}^{post}\mathrm{IK}_{B,\mathrm{pub}}, {}^{post}\mathrm{SPK}_{B,\mathrm{pub}}, {}^{post}\mathrm{sig}_B)$
- Perform the following encapsulations

$$(c_2, {}^{\mathrm{post}}\, k_2) = \mathrm{encaps}({}^{\mathrm{post}}\mathrm{IK}_{\mathrm{B;pub}})$$
$$(c_3, {}^{\mathrm{post}}\, k_3) = \mathrm{encaps}({}^{\mathrm{post}}\mathrm{SPK}_{\mathrm{B;pub}})$$
$$((c_4, {}^{\mathrm{post}}\, k_4) = \mathrm{encaps}({}^{\mathrm{post}}\mathrm{OPK}^i_{\mathrm{B;pub}}))$$

4. Quantum Secure X3DH
4. Tutanota's Implementation

- Combine all keys with KDF
  - $\mathrm{SK} = \mathrm{KDF}({}^{pre}k_1 \,||\, {}^{**}k_2 \,||\, {}^{**}k_3 \,(||\, {}^{**}k_4))$
- Since ${}^{post}k_1$ can not exist, must perform authentication of Alice afterwards
  - Sign all data send to Bob for the handshake
  - ${}^{post}\mathrm{sig}_A = \mathrm{sign}({}^{post}\mathrm{IK}_A, \mathrm{data})$
  - Enables mutual authentication, but weakens Deniability
- Bob receives message
  - Verify signature
  - Decapsulate keys
  - Calculate SK

4. Quantum Secure X3DH

4. Tutanota’s Implementation

How to improve Deniability?

- Split KEMs
  - Theoretical construct introduced by Brendel et al.
  - Enables using (semi-) static keys in the encapsulation process
  - If such a split KEM construction with the same security properties as a regular KEM is possible is not known
- Use Ring Signatures
  - Proposed by Hashimoto et al

4. Quantum Secure X3DH

A) Excursus: Ring Signatures

- Invented by Rivest, Shamir and Kalai in 2001
- Enables semi-anonymous signing of data
- Signing requires supplying a private key and a set of public keys able to verify the signature
- Created with $\mathrm{RS.sign}(\mathrm{sk}_i, \mathrm{data}, (\mathrm{pk}_0, \mathrm{pk}_1, ...))$
- Everyone in the set of public keys could have signed data
- Original paper based on RSA => not PQ-secure
- Some research into PQ-secure ring signatures by Chatterjee et al.
- Very complex construction, requiring multiple advanced cryptographic components

4. Quantum Secure X3DH

5. Signal's PQXDH

- In May 2023 published the first revision of the the PQXDH protocol
- Updated version was incorperated into all Signal clients in September 2023
- Also hybrid protocol
- Uses CRYSTALS-Kyber-1024
- Adds only a single new signed PQ-secure key to calculations
- Let ${}^{post}\mathrm{PQPK}_X$ be either a signed one-time key ${}^{post}\mathrm{PQOPK}_X^k$ or the last resort key ${}^{post}\mathrm{PQSPK}_X$ only used if no one-time key remains
  - Key is only signed with conventional signature scheme, XedDSA

4. Quantum Secure X3DH

5. Signal's PQXDH

- When Alice receives bundle, she verifies both signed keys
- Then calculate regular DH components
- Furthermore perform KEM encapsulation
  - $(c_{\mathrm{pq}}, {}^{\mathrm{post}}\, k_{\mathrm{pq}}) = \mathrm{encaps}({}^{post}\mathrm{PQPK}_{B;pub})$
- Combine key material with KDF
  - $\mathrm{SK} = \mathrm{KDF}({}^{\mathrm{pre}}k1 || {}^{\mathrm{pre}}k2 || {}^{\mathrm{pre}}k3 || {}^{\mathrm{post}}k_{\mathrm{pq}}(|| {}^{\mathrm{pre}}k4))$
- Allows for similar deniability as X3DH
- No PQ-secure authentication
  - However, would require an active quantum-capable attacker
- Protects against passive quantum adversaries

5. Summary

- Signal Protocol is an E2E messaging protocol used in a variety of apps like Signal and WhatsApp
- Handshake Protocol X3DH has remarkable properties like Forward Secrecy and Deniability
- Not PQ-secure since it heavily relies on (EC)DH
- SIDH would have provided a great way of making X3DH quantum secure, while retaining properties => SIDH not secure
- Use of KEMs is difficult since it would require signatures to enable authentication
  - Loss of some deniability
  - Could be improved with Split KEMs or Ring Signatures
- Signal’s new PQXDH does not provide quantum-authentication, however currently not required
  - Enables same deniability as before

Thanks for listening

Photo Credits

- Trevor Perrin – Screenshot from https://www.youtube.com/watch?v=3gipxdJ22iM
- Moxie Marlinspike - By Cmichel67 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=124678255
- X3DH graphics - https://signal.org/docs/specifications/x3dh/
- Tutanota Logo - By tutanota.com, https://commons.wikimedia.org/w/index.php?curid=109546774
- Axolotl images - https://www.ardalpha.de/wissen/natur/tiere/artenschutz/rote-liste/axolotl-schwanzlurch-mexiko-bilder-100.html

References

- M. Marlinspike and T. Perrin, “The X3DH Key Agreement Protocol” 2016. https://signal.org/docs/specifications/x3dh/
- Unger et al. 2015, pp. 232-249 http://ieee-security.org/TC/SP2015/papers-archived/6949a232.pdf
- Wikipedia contributors, “Signal Protocol”, 13. November 2022. https://en.wikipedia.org/w/index.php?title=Signal_Protocol&oldid=1121702733
- PQShield, “Secure Messaging in a Post-Quantum World”, 2022. https://content.pqshield.com/secure-messaging-in-a-post-quantum-world
- I. Duits, “The Post-Quantum Signal Protocol - Secure Chat in a Quantum World” 2019. https://essay.utwente.nl/77239/1/Duits_MA_EEMCS.pdf
- https://tutanota.com/encryption, 10. October 2023
- https://signal.org/blog/pqxdh/, 10. October 2023
- Brendel et al., “Towards Post-Quantum Security for Signal's X3DH Handshake”, 2021, https://doi.org/10.3929/ethz-b-000441452
- Ring signature, https://en.wikipedia.org/w/index.php?title=Ring_signature&oldid=1172131481 (last visited Oct. 11, 2023)
- Chatterjee et al., “A Note on the Post-Quantum Security of (Ring) Signatures” , 2021. https://doi.org/10.48550/arXiv.2112.06078