K-Waay: Fast and Deniable Post-Quantum X3DH without Ring Signatures

Daniel Collins, Loïs Huguenin-Dumittan, Ngoc Khanh Nguyen, Nicolas Rolin, Serge Vaudenay

USENIX Security 2024

EPFL

The Signal Protocol

- **Two–party messaging between Alice and Bob.**
- **De–facto standard: *The Signal Protocol.***
- **Two main components: key exchange and messaging proper.**
- **Key exchange: X3DH (Extended Triple Diffie–Hellman)**
  - Recently: PQXDH (cf. this session!)

Extended Triple Diffie–Hellman (X3DH)

- **X3DH provides secrecy, authentication, deniability, support for offline users…**
- **Due to Shor's algorithm, *post-quantum* cryptography is needed.**
  - X3DH is classically secure [CCDGS20,VGIK20] but not post-quantum.
- **Well-understood how to build post-quantum messaging (Double Ratchet [ACD19]).**
  - Less so for X3DH.

Post-Quantum X3DH-Like Key Exchange

- **Challenge: post-quantum security *and* deniability.**
- **Recently: Signal deployed PQXDH (X3DH + PQ KEM).**
  - Secure against store-now, decrypt-later quantum attacks.
- **Hashimoto et al. [HKKP21, HKKP22] and Brendel et al. [BFGJS22] proposed fully PQ-secure X3DH protocols.**
  - Rely on PQ ring signatures and KEMs.
  - Ring signatures can be a bottleneck.

Split-KEM

- [BFGJS20]: key encapsulation mechanism (KEM) where Encaps/Decaps take as input public *and* secret keys.
- Key generation: $(pk_A, sk_A) \leftarrow \mathsf{KeyGenA}$, $(pk_B, sk_B) \leftarrow \mathsf{KeyGenB}$
- Encapsulation: $(K, ct) \leftarrow \mathsf{Encaps}(pk_B, sk_A)$
- Decapsulation: $K \leftarrow \mathsf{Decaps}(pk_A, sk_B, ct)$
- Morally generalises Diffie–Hellman/NIKE with a ciphertext.
- Was not *formally* shown to imply key exchange by [BFGJS20].

Our Protocol: K-Waay

- *K-Waay*: deniable PQ X3DH based on *split-KEM* and KEMs.

- Uses split-KEM with ephemeral keys, ephemeral KEM, long-term KEM and long-term signatures for prekeys.
  - Security against different combinations of key exposure.

K–Waay

Init:
upload signed prekey bundles

Alice $(\mathsf{lpk_B}, \mathsf{vk_B})$		Bob $(\mathsf{lpk_A}, \mathsf{vk_A})$
$(\mathsf{esk}_\mathsf{A}^\mathsf{skem}, \mathsf{epk}_\mathsf{A}^\mathsf{skem}) \xleftarrow{\$} \mathsf{KeyGenA}_\mathsf{sKEM}$		$(\mathsf{esk}_\mathsf{B}^\mathsf{skem}, \mathsf{epk}_\mathsf{B}^\mathsf{skem}) \xleftarrow{\$} \mathsf{KeyGenB}_\mathsf{sKEM}$
		$(\mathsf{esk}_\mathsf{B}^\mathsf{kem}, \mathsf{epk}_\mathsf{B}^\mathsf{kem}) \xleftarrow{\$} \mathsf{KeyGen}_\mathsf{KEM}$
	$\xleftarrow{\{\mathsf{epk}_\mathsf{B}^\mathsf{skem}, \mathsf{epk}_\mathsf{B}^\mathsf{kem}\}_{\mathsf{sk_B}}}$ $\xrightarrow{\{\mathsf{epk}_\mathsf{A}^\mathsf{skem}\}_{\mathsf{sk_A}}}$	

- - - - - - - - - - - - - - - - - - - -

$K_\ell, \mathsf{ct}_\ell \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{lpk_B})$		
$K_e^\mathsf{skem}, \mathsf{ct}_e^\mathsf{skem} \xleftarrow{\$} \mathsf{Encaps_{sKEM}}(\mathsf{epk}_\mathsf{B}^\mathsf{skem}, \mathsf{esk}_\mathsf{A}^\mathsf{skem})$		
$K_e^\mathsf{kem}, \mathsf{ct}_e^\mathsf{kem} \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{epk}_\mathsf{B}^\mathsf{kem})$		
$K \leftarrow \mathsf{KDF}(K_\ell, K_e^\mathsf{skem}, K_e^\mathsf{kem})$	$\xrightarrow{\mathsf{ct}_\ell, \mathsf{ct}_e^\mathsf{skem}, \mathsf{ct}_e^\mathsf{kem}}$	
		$K'_\ell \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{lsk_B}, \mathsf{ct}_\ell)$
		$K_e'^\mathsf{skem} \leftarrow \mathsf{Decaps_{sKEM}}(\mathsf{epk}_\mathsf{A}^\mathsf{skem}, \mathsf{esk}_\mathsf{B}^\mathsf{skem}, \mathsf{ct}_e^\mathsf{skem})$
		$K_e'^\mathsf{kem} \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{esk}_\mathsf{B}^\mathsf{kem}, \mathsf{ct}_e^\mathsf{kem})$
		$K'_\ell \stackrel{?}{=} \perp \vee K_e'^\mathsf{skem} \stackrel{?}{=} \perp \vee K_e'^\mathsf{kem} \stackrel{?}{=} \perp$
		$K' \leftarrow \mathsf{KDF}(K'_\ell, K_e'^\mathsf{skem}, K_e'^\mathsf{kem})$

K–Waay

Send:
- Encaps with two KEMs and Split-KEM

- KDF call

Alice $(\mathsf{lpk_B}, \mathsf{vk_B})$		Bob $(\mathsf{lpk_A}, \mathsf{vk_A})$
$(\mathsf{esk}_A^{\mathsf{skem}}, \mathsf{epk}_A^{\mathsf{skem}}) \xleftarrow{\$} \mathsf{KeyGenA_{sKEM}}$		$(\mathsf{esk}_B^{\mathsf{skem}}, \mathsf{epk}_B^{\mathsf{skem}}) \xleftarrow{\$} \mathsf{KeyGenB_{sKEM}}$
		$(\mathsf{esk}_B^{\mathsf{kem}}, \mathsf{epk}_B^{\mathsf{kem}}) \xleftarrow{\$} \mathsf{KeyGen_{KEM}}$
	$\xleftarrow{\{\mathsf{epk}_B^{\mathsf{skem}}, \mathsf{epk}_B^{\mathsf{kem}}\}_{\mathsf{sk_B}}}$ $\xrightarrow{\{\mathsf{epk}_A^{\mathsf{skem}}\}_{\mathsf{sk_A}}}$	
$K_\ell, \mathsf{ct}_\ell \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{lpk_B})$		
$K_e^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{skem}} \xleftarrow{\$} \mathsf{Encaps_{sKEM}}(\mathsf{epk}_B^{\mathsf{skem}}, \mathsf{esk}_A^{\mathsf{skem}})$		
$K_e^{\mathsf{kem}}, \mathsf{ct}_e^{\mathsf{kem}} \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{epk}_B^{\mathsf{kem}})$		
$K \leftarrow \mathsf{KDF}(K_\ell, K_e^{\mathsf{skem}}, K_e^{\mathsf{kem}})$	$\xrightarrow{\mathsf{ct}_\ell, \mathsf{ct}_e^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{kem}}}$	
		$K'_\ell \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{lsk_B}, \mathsf{ct}_\ell)$
		$K_e'^{\,\mathsf{skem}} \leftarrow \mathsf{Decaps_{sKEM}}(\mathsf{epk}_A^{\mathsf{skem}}, \mathsf{esk}_B^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{skem}})$
		$K_e'^{\,\mathsf{kem}} \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{esk}_B^{\mathsf{kem}}, \mathsf{ct}_e^{\mathsf{kem}})$
		$K'_\ell \stackrel{?}{=} \perp \vee K_e'^{\,\mathsf{skem}} \stackrel{?}{=} \perp \vee K_e'^{\,\mathsf{kem}} \stackrel{?}{=} \perp$
		$K' \leftarrow \mathsf{KDF}(K'_\ell, K_e'^{\,\mathsf{skem}}, K_e'^{\,\mathsf{kem}})$

K–Waay

Send:
- Encaps with two KEMs and Split-KEM

- KDF call

Alice $(\mathsf{lpk_B}, \mathsf{vk_B})$		Bob $(\mathsf{lpk_A}, \mathsf{vk_A})$
$(\mathsf{esk}^{\mathsf{skem}}_{\mathsf{A}}, \mathsf{epk}^{\mathsf{skem}}_{\mathsf{A}}) \xleftarrow{\$} \mathsf{KeyGenA_{sKEM}}$		$(\mathsf{esk}^{\mathsf{skem}}_{\mathsf{B}}, \mathsf{epk}^{\mathsf{skem}}_{\mathsf{B}}) \xleftarrow{\$} \mathsf{KeyGenB_{sKEM}}$
		$(\mathsf{esk}^{\mathsf{kem}}_{\mathsf{B}}, \mathsf{epk}^{\mathsf{kem}}_{\mathsf{B}}) \xleftarrow{\$} \mathsf{KeyGen_{KEM}}$
	$\xleftarrow{\{\mathsf{epk}^{\mathsf{skem}}_{\mathsf{B}}, \mathsf{epk}^{\mathsf{kem}}_{\mathsf{B}}\}_{\mathsf{sk_B}}}$ $\xrightarrow{\{\mathsf{epk}^{\mathsf{skem}}_{\mathsf{A}}\}_{\mathsf{sk_A}}}$	
$K_\ell, \mathsf{ct}_\ell \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{lpk_B})$		
$K^{\mathsf{skem}}_e, \mathsf{ct}^{\mathsf{skem}}_e \xleftarrow{\$} \mathsf{Encaps_{sKEM}}(\mathsf{epk}^{\mathsf{skem}}_{\mathsf{B}}, \mathsf{esk}^{\mathsf{skem}}_{\mathsf{A}})$		
$K^{\mathsf{kem}}_e, \mathsf{ct}^{\mathsf{kem}}_e \xleftarrow{\$} \mathsf{Encaps_{KEM}}(\mathsf{epk}^{\mathsf{kem}}_{\mathsf{B}})$		
$K \leftarrow \mathsf{KDF}(K_\ell, K^{\mathsf{skem}}_e, K^{\mathsf{kem}}_e)$	$\xrightarrow{\mathsf{ct}_\ell, \mathsf{ct}^{\mathsf{skem}}_e, \mathsf{ct}^{\mathsf{kem}}_e}$	
		$K'_\ell \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{lsk_B}, \mathsf{ct}_\ell)$
		$K'^{\mathsf{skem}}_e \leftarrow \mathsf{Decaps_{sKEM}}(\mathsf{epk}^{\mathsf{skem}}_{\mathsf{A}}, \mathsf{esk}^{\mathsf{skem}}_{\mathsf{B}}, \mathsf{ct}^{\mathsf{skem}}_e)$
		$K'^{\mathsf{kem}}_e \leftarrow \mathsf{Decaps_{KEM}}(\mathsf{esk}^{\mathsf{kem}}_{\mathsf{B}}, \mathsf{ct}^{\mathsf{kem}}_e)$
		$K'_\ell \stackrel{?}{=} \perp \vee K'^{\mathsf{skem}}_e \stackrel{?}{=} \perp \vee K'^{\mathsf{kem}}_e \stackrel{?}{=} \perp$
		$K' \leftarrow \mathsf{KDF}(K'_\ell, K'^{\mathsf{skem}}_e, K'^{\mathsf{kem}}_e)$

K–Waay

$$\text{Alice } (\mathsf{lpk_B}, \mathsf{vk_B}) \qquad\qquad \text{Bob } (\mathsf{lpk_A}, \mathsf{vk_A})$$

$$(\mathsf{esk}_\mathsf{A}^\mathsf{skem}, \mathsf{epk}_\mathsf{A}^\mathsf{skem}) \xleftarrow{\$} \mathsf{KeyGenA}_\mathsf{sKEM} \qquad\qquad (\mathsf{esk}_\mathsf{B}^\mathsf{skem}, \mathsf{epk}_\mathsf{B}^\mathsf{skem}) \xleftarrow{\$} \mathsf{KeyGenB}_\mathsf{sKEM}$$

$$(\mathsf{esk}_\mathsf{B}^\mathsf{kem}, \mathsf{epk}_\mathsf{B}^\mathsf{kem}) \xleftarrow{\$} \mathsf{KeyGen}_\mathsf{KEM}$$

$$\xleftarrow{\{\mathsf{epk}_\mathsf{B}^\mathsf{skem}, \mathsf{epk}_\mathsf{B}^\mathsf{kem}\}_{\mathsf{sk_B}}} \qquad \xrightarrow{\{\mathsf{epk}_\mathsf{A}^\mathsf{skem}\}_{\mathsf{sk_A}}}$$

$$K_\ell, \mathsf{ct}_\ell \xleftarrow{\$} \mathsf{Encaps}_\mathsf{KEM}(\mathsf{lpk_B})$$

$$K_e^\mathsf{skem}, \mathsf{ct}_e^\mathsf{skem} \xleftarrow{\$} \mathsf{Encaps}_\mathsf{sKEM}(\mathsf{epk}_\mathsf{B}^\mathsf{skem}, \mathsf{esk}_\mathsf{A}^\mathsf{skem})$$

$$K_e^\mathsf{kem}, \mathsf{ct}_e^\mathsf{kem} \xleftarrow{\$} \mathsf{Encaps}_\mathsf{KEM}(\mathsf{epk}_\mathsf{B}^\mathsf{kem})$$

$$K \leftarrow \mathsf{KDF}(K_\ell, K_e^\mathsf{skem}, K_e^\mathsf{kem}) \qquad \xrightarrow{\mathsf{ct}_\ell, \mathsf{ct}_e^\mathsf{skem}, \mathsf{ct}_e^\mathsf{kem}}$$

$$K'_\ell \leftarrow \mathsf{Decaps}_\mathsf{KEM}(\mathsf{lsk_B}, \mathsf{ct}_\ell)$$

$$K'^\mathsf{skem}_e \leftarrow \mathsf{Decaps}_\mathsf{sKEM}(\mathsf{epk}_\mathsf{A}^\mathsf{skem}, \mathsf{esk}_\mathsf{B}^\mathsf{skem}, \mathsf{ct}_e^\mathsf{skem})$$

$$K'^\mathsf{kem}_e \leftarrow \mathsf{Decaps}_\mathsf{KEM}(\mathsf{esk}_\mathsf{B}^\mathsf{kem}, \mathsf{ct}_e^\mathsf{kem})$$

$$K'_\ell \stackrel{?}{=} \bot \vee K'^\mathsf{skem}_e \stackrel{?}{=} \bot \vee K'^\mathsf{kem}_e \stackrel{?}{=} \bot$$

$$K' \leftarrow \mathsf{KDF}(K'_\ell, K'^\mathsf{skem}_e, K'^\mathsf{kem}_e)$$

K–Waay

Receive: corresponding Decaps and KDF calls

Alice $(\mathsf{lpk_B}, \mathsf{vk_B})$		Bob $(\mathsf{lpk_A}, \mathsf{vk_A})$
$(\mathsf{esk}_\mathsf{A}^{\mathsf{skem}}, \mathsf{epk}_\mathsf{A}^{\mathsf{skem}}) \xleftarrow{\$} \mathsf{KeyGenA}_{\mathsf{sKEM}}$		$(\mathsf{esk}_\mathsf{B}^{\mathsf{skem}}, \mathsf{epk}_\mathsf{B}^{\mathsf{skem}}) \xleftarrow{\$} \mathsf{KeyGenB}_{\mathsf{sKEM}}$
		$(\mathsf{esk}_\mathsf{B}^{\mathsf{kem}}, \mathsf{epk}_\mathsf{B}^{\mathsf{kem}}) \xleftarrow{\$} \mathsf{KeyGen}_{\mathsf{KEM}}$
	$\xleftarrow{\{\mathsf{epk}_\mathsf{B}^{\mathsf{skem}}, \mathsf{epk}_\mathsf{B}^{\mathsf{kem}}\}_{\mathsf{sk_B}}}$ $\xrightarrow{\{\mathsf{epk}_\mathsf{A}^{\mathsf{skem}}\}_{\mathsf{sk_A}}}$	
$K_\ell, \mathsf{ct}_\ell \xleftarrow{\$} \mathsf{Encaps}_{\mathsf{KEM}}(\mathsf{lpk_B})$		
$K_e^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{skem}} \xleftarrow{\$} \mathsf{Encaps}_{\mathsf{sKEM}}(\mathsf{epk}_\mathsf{B}^{\mathsf{skem}}, \mathsf{esk}_\mathsf{A}^{\mathsf{skem}})$		
$K_e^{\mathsf{kem}}, \mathsf{ct}_e^{\mathsf{kem}} \xleftarrow{\$} \mathsf{Encaps}_{\mathsf{KEM}}(\mathsf{epk}_\mathsf{B}^{\mathsf{kem}})$		
$K \leftarrow \mathsf{KDF}(K_\ell, K_e^{\mathsf{skem}}, K_e^{\mathsf{kem}})$	$\xrightarrow{\mathsf{ct}_\ell, \mathsf{ct}_e^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{kem}}}$	
		$K'_\ell \leftarrow \mathsf{Decaps}_{\mathsf{KEM}}(\mathsf{lsk_B}, \mathsf{ct}_\ell)$
		$K_e'^{\mathsf{skem}} \leftarrow \mathsf{Decaps}_{\mathsf{sKEM}}(\mathsf{epk}_\mathsf{A}^{\mathsf{skem}}, \mathsf{esk}_\mathsf{B}^{\mathsf{skem}}, \mathsf{ct}_e^{\mathsf{skem}})$
		$K_e'^{\mathsf{kem}} \leftarrow \mathsf{Decaps}_{\mathsf{KEM}}(\mathsf{esk}_\mathsf{B}^{\mathsf{kem}}, \mathsf{ct}_e^{\mathsf{kem}})$
		$K'_\ell \stackrel{?}{=} \perp \vee K_e'^{\mathsf{skem}} \stackrel{?}{=} \perp \vee K_e'^{\mathsf{kem}} \stackrel{?}{=} \perp$
		$K' \leftarrow \mathsf{KDF}(K'_\ell, K_e'^{\mathsf{skem}}, K_e'^{\mathsf{kem}})$

Split–KEM (2)

- **We revisit split–KEM: its original security notions are *insufficient* for X3DH–like key exchange.**
  - Authentication guarantees.
  - Semi–honest deniability, like [BFGJS22].
  - Support ephemeral key–reuse *without* full CCA security!
- **We propose a secure instantiation secure under plain learning–with–errors (LWE).**
  - Based on Frodo key exchange [BCD+16].

Frodo-based Split-KEM (pre-RO transform)

KeyGen: extra noise term, otherwise Frodo.

KeyGenA(1^λ)

1 : $\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A} \leftarrow\!\!\$\ \chi(\mathbb{Z}_q^{n\times\bar{n}})$

2 : $\mathbf{F}_\mathsf{A} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

3 : $\mathbf{B}_\mathsf{A} \leftarrow \mathbf{A}\mathbf{S}_\mathsf{A} + \mathbf{D}_\mathsf{A}$

4 : $\mathsf{pk}_\mathsf{A} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{A})$

5 : $\mathsf{sk}_\mathsf{A} \leftarrow (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A})$

6 : **return** $(\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{A})$

Encaps($\mathsf{pk}_\mathsf{B} = (\mathbf{A}, \mathbf{B}_\mathsf{B}), \mathsf{sk}_\mathsf{A} = (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A})$)

1 : // We assume A encapsulates

2 : $\mathbf{E}_\mathsf{A} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

3 : $\mathbf{V} \leftarrow \mathbf{S}_\mathsf{A}\mathbf{B}_\mathsf{B} + \mathbf{E}_\mathsf{A}$

4 : $\mathsf{ct} \leftarrow \mathsf{HelpRec}(\mathbf{V})$

5 : $\mathbf{K} \leftarrow \mathsf{Rec}(2\mathbf{V}, \mathsf{ct})$

6 : **return** $(\mathbf{K}, \mathsf{ct})$

KeyGenB(1^λ)

1 : $\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B} \leftarrow\!\!\$\ \chi^{\bar{n}\times n}$

2 : $\mathbf{F}_\mathsf{B} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

3 : $\mathbf{B}_\mathsf{B} \leftarrow \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$

4 : $\mathsf{pk}_\mathsf{B} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{B})$

5 : $\mathsf{sk}_\mathsf{B} \leftarrow (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B})$

6 : **return** $(\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{B})$

Decaps($\mathsf{pk}_\mathsf{A} = (\mathbf{A}, \mathbf{B}_\mathsf{A}), \mathsf{sk}_\mathsf{B} = (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B}), \mathsf{ct}$)

1 : $\mathbf{V}' \leftarrow \mathbf{B}_\mathsf{A}\mathbf{S}_\mathsf{B} + \mathbf{F}_\mathsf{B}$

2 : $\mathbf{K}' \leftarrow \mathsf{Rec}(2\mathbf{V}', \mathsf{ct})$

3 : **return** $\mathbf{K}'$

Frodo–based Split–KEM (pre–RO transform)

KeyGen: extra noise term, <u>otherwise Frodo</u>.

$\mathsf{KeyGenA}(1^\lambda)$

$\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A} \leftarrow\!\!\$\ \chi(\mathbb{Z}_q^{n\times\bar{n}})$

$\mathbf{F}_\mathsf{A} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

$\mathbf{B}_\mathsf{A} \leftarrow \mathbf{A}\mathbf{S}_\mathsf{A} + \mathbf{D}_\mathsf{A}$

$\mathsf{pk}_\mathsf{A} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{A})$

$\mathsf{sk}_\mathsf{A} \leftarrow (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A})$

$\mathbf{return}\ (\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{A})$

$\mathsf{KeyGenB}(1^\lambda)$

$\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B} \leftarrow\!\!\$\ \chi^{\bar{n}\times n}$

$\mathbf{F}_\mathsf{B} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

$\mathbf{B}_\mathsf{B} \leftarrow \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$

$\mathsf{pk}_\mathsf{B} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{B})$

$\mathsf{sk}_\mathsf{B} \leftarrow (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B})$

$\mathbf{return}\ (\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{B})$

$\mathsf{Encaps}(\mathsf{pk}_\mathsf{B} = (\mathbf{A}, \mathbf{B}_\mathsf{B}), \mathsf{sk}_\mathsf{A} = (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A}))$

// We assume A encapsulates

$\mathbf{E}_\mathsf{A} \leftarrow\!\!\$\ \chi^{\bar{n}\times\bar{n}}$

$\mathbf{V} \leftarrow \mathbf{S}_\mathsf{A}\mathbf{B}_\mathsf{B} + \mathbf{E}_\mathsf{A}$

$\mathsf{ct} \leftarrow \mathsf{HelpRec}(\mathbf{V})$

$\mathbf{K} \leftarrow \mathsf{Rec}(2\mathbf{V}, \mathsf{ct})$

$\mathbf{return}\ (\mathbf{K}, \mathsf{ct})$

$\mathsf{Decaps}(\mathsf{pk}_\mathsf{A} = (\mathbf{A}, \mathbf{B}_\mathsf{A}), \mathsf{sk}_\mathsf{B} = (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B}), \mathsf{ct})$

$\mathbf{V}' \leftarrow \mathbf{B}_\mathsf{A}\mathbf{S}_\mathsf{B} + \mathbf{F}_\mathsf{B}$

$\mathbf{K}' \leftarrow \mathsf{Rec}(2\mathbf{V}', \mathsf{ct})$

$\mathbf{return}\ \mathbf{K}'$

Frodo-based Split-KEM (pre-RO transform)

Encaps:
Frodo.

$\mathsf{KeyGenA}(1^\lambda)$

1: $\mathbf{S}_{\mathsf{A}}, \mathbf{D}_{\mathsf{A}} \leftarrow\$ \chi(\mathbb{Z}_q^{n\times\overline{n}})$
2: $\mathbf{F}_{\mathsf{A}} \leftarrow\$ \chi^{\overline{n}\times\overline{n}}$
3: $\mathbf{B}_{\mathsf{A}} \leftarrow \mathbf{A}\mathbf{S}_{\mathsf{A}} + \mathbf{D}_{\mathsf{A}}$
4: $\mathsf{pk}_{\mathsf{A}} \leftarrow (\mathbf{A}, \mathbf{B}_{\mathsf{A}})$
5: $\mathsf{sk}_{\mathsf{A}} \leftarrow (\mathbf{S}_{\mathsf{A}}, \mathbf{D}_{\mathsf{A}}, \mathbf{F}_{\mathsf{A}})$
6: $\mathbf{return}\ (\mathsf{pk}_{\mathsf{A}}, \mathsf{sk}_{\mathsf{A}})$

$\mathsf{Encaps}(\mathsf{pk}_{\mathsf{B}} = (\mathbf{A}, \mathbf{B}_{\mathsf{B}}), \mathsf{sk}_{\mathsf{A}} = (\mathbf{S}_{\mathsf{A}}, \mathbf{D}_{\mathsf{A}}, \mathbf{F}_{\mathsf{A}}))$

1: // We assume A encapsulates
2: $\mathbf{E}_{\mathsf{A}} \leftarrow\$ \chi^{\overline{n}\times\overline{n}}$
3: $\mathbf{V} \leftarrow \mathbf{S}_{\mathsf{A}}\mathbf{B}_{\mathsf{B}} + \mathbf{E}_{\mathsf{A}}$
4: $\mathsf{ct} \leftarrow \mathsf{HelpRec}(\mathbf{V})$
5: $\mathbf{K} \leftarrow \mathsf{Rec}(2\mathbf{V}, \mathsf{ct})$
6: $\mathbf{return}\ (\mathbf{K}, \mathsf{ct})$

$\mathsf{KeyGenB}(1^\lambda)$

1: $\mathbf{S}_{\mathsf{B}}, \mathbf{D}_{\mathsf{B}} \leftarrow\$ \chi^{\overline{n}\times n}$
2: $\mathbf{F}_{\mathsf{B}} \leftarrow\$ \chi^{\overline{n}\times\overline{n}}$
3: $\mathbf{B}_{\mathsf{B}} \leftarrow \mathbf{S}_{\mathsf{B}}\mathbf{A} + \mathbf{D}_{\mathsf{B}}$
4: $\mathsf{pk}_{\mathsf{B}} \leftarrow (\mathbf{A}, \mathbf{B}_{\mathsf{B}})$
5: $\mathsf{sk}_{\mathsf{B}} \leftarrow (\mathbf{S}_{\mathsf{B}}, \mathbf{D}_{\mathsf{B}}, \mathbf{F}_{\mathsf{B}})$
6: $\mathbf{return}\ (\mathsf{pk}_{\mathsf{B}}, \mathsf{sk}_{\mathsf{B}})$

$\mathsf{Decaps}(\mathsf{pk}_{\mathsf{A}} = (\mathbf{A}, \mathbf{B}_{\mathsf{A}}), \mathsf{sk}_{\mathsf{B}} = (\mathbf{S}_{\mathsf{B}}, \mathbf{D}_{\mathsf{B}}, \mathbf{F}_{\mathsf{B}}), \mathsf{ct})$

1: $\mathbf{V}' \leftarrow \mathbf{B}_{\mathsf{A}}\mathbf{S}_{\mathsf{B}} + \mathbf{F}_{\mathsf{B}}$
2: $\mathbf{K}' \leftarrow \mathsf{Rec}(2\mathbf{V}', \mathsf{ct})$
3: $\mathbf{return}\ \mathbf{K}'$

Frodo-based Split-KEM (pre-RO transform)

Decaps: Noise from KeyGen added to V′

$\mathsf{KeyGenA}(1^\lambda)$

1 : $\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A} \leftarrow\!\!\$\; \chi(\mathbb{Z}_q^{n\times\overline{n}})$

2 : $\mathbf{F}_\mathsf{A} \leftarrow\!\!\$\; \chi^{\overline{n}\times\overline{n}}$

3 : $\mathbf{B}_\mathsf{A} \leftarrow \mathbf{A}\mathbf{S}_\mathsf{A} + \mathbf{D}_\mathsf{A}$

4 : $\mathsf{pk}_\mathsf{A} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{A})$

5 : $\mathsf{sk}_\mathsf{A} \leftarrow (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A})$

6 : **return** $(\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{A})$

$\mathsf{Encaps}(\mathsf{pk}_\mathsf{B} = (\mathbf{A}, \mathbf{B}_\mathsf{B}), \mathsf{sk}_\mathsf{A} = (\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A}, \mathbf{F}_\mathsf{A}))$

1 : // We assume A encapsulates

2 : $\mathbf{E}_\mathsf{A} \leftarrow\!\!\$\; \chi^{\overline{n}\times\overline{n}}$

3 : $\mathbf{V} \leftarrow \mathbf{S}_\mathsf{A}\mathbf{B}_\mathsf{B} + \mathbf{E}_\mathsf{A}$

4 : $\mathsf{ct} \leftarrow \mathsf{HelpRec}(\mathbf{V})$

5 : $\mathbf{K} \leftarrow \mathsf{Rec}(2\mathbf{V}, \mathsf{ct})$

6 : **return** $(\mathbf{K}, \mathsf{ct})$

$\mathsf{KeyGenB}(1^\lambda)$

1 : $\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B} \leftarrow\!\!\$\; \chi^{\overline{n}\times n}$

2 : $\mathbf{F}_\mathsf{B} \leftarrow\!\!\$\; \chi^{\overline{n}\times\overline{n}}$

3 : $\mathbf{B}_\mathsf{B} \leftarrow \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$

4 : $\mathsf{pk}_\mathsf{B} \leftarrow (\mathbf{A}, \mathbf{B}_\mathsf{B})$

5 : $\mathsf{sk}_\mathsf{B} \leftarrow (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B})$

6 : **return** $(\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{B})$

$\mathsf{Decaps}(\mathsf{pk}_\mathsf{A} = (\mathbf{A}, \mathbf{B}_\mathsf{A}), \mathsf{sk}_\mathsf{B} = (\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}, \mathbf{F}_\mathsf{B}), \mathsf{ct})$

1 : $\mathbf{V}' \leftarrow \mathbf{B}_\mathsf{A}\mathbf{S}_\mathsf{B} + \mathbf{F}_\mathsf{B}$

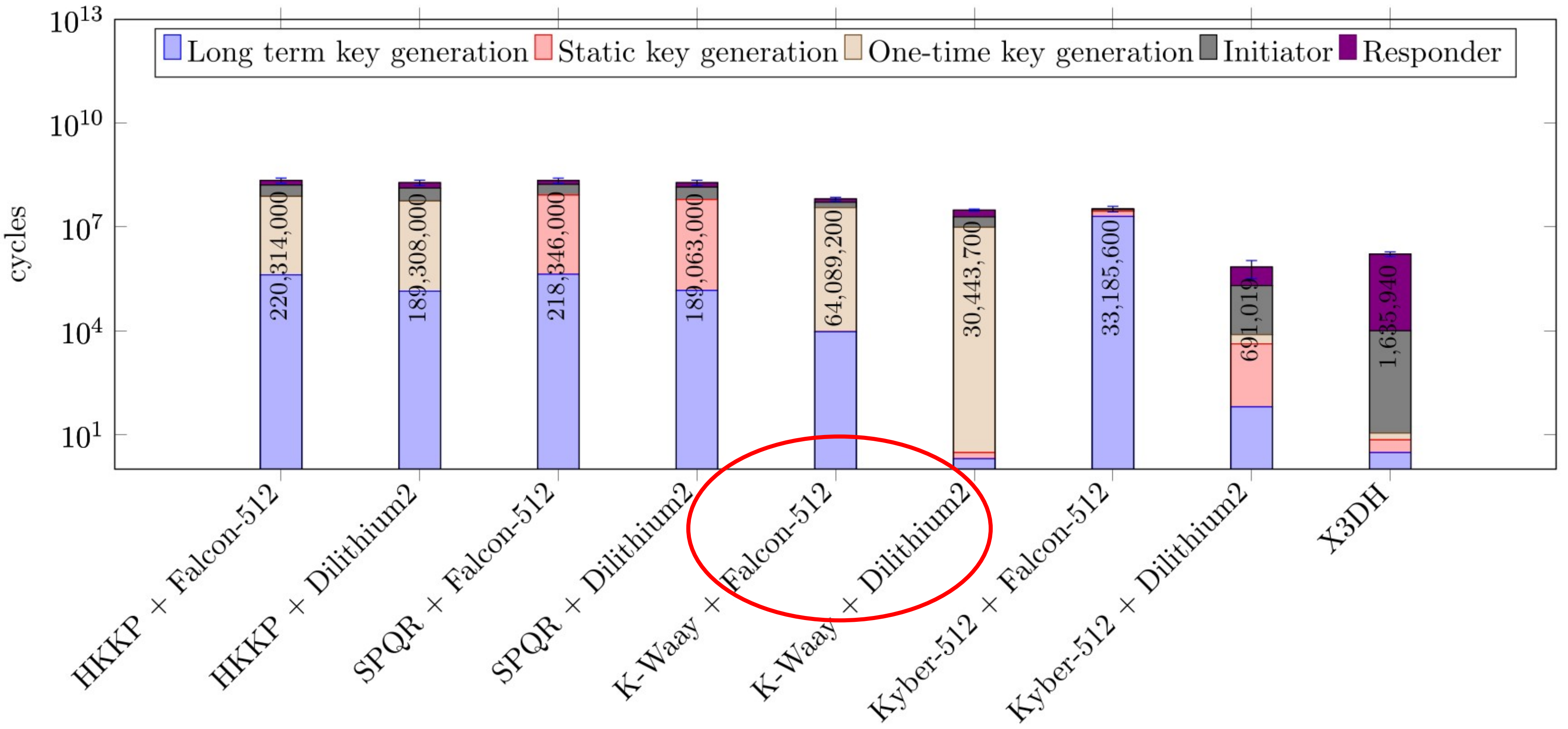
2 : $\mathbf{K}' \leftarrow \mathsf{Rec}(2\mathbf{V}', \mathsf{ct})$

3 : **return** $\mathbf{K}'$

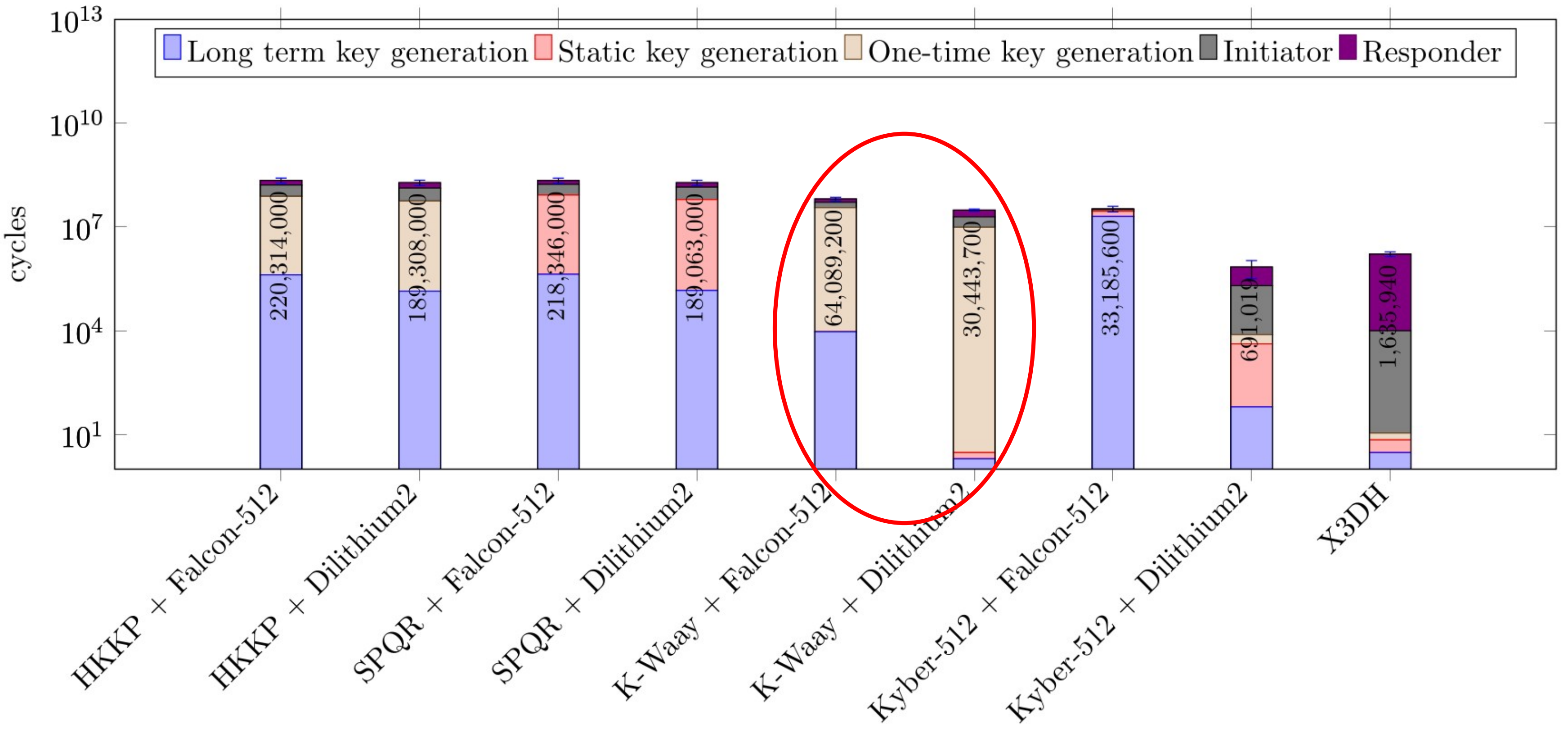
Parameters and Benchmarking

- **We propose parameters for K–Waay.**
- ***Before* a random oracle (RO) transform (extra hash in ciphertext):**
  - >192/128 bits of classical/quantum security (OW–CPA, decaps–OW–CPA, deniability).
- ***After*:**
  - 128–bit security, 64 bits in the QROM (assuming 2^{64} RO queries).
- **Previous PQ X3DH benchmarks: worse concrete security!**
- **We benchmark with Kyber–512 [BDK+18] and Raptor ring signatures [LAZ19] for related work.**

Speed

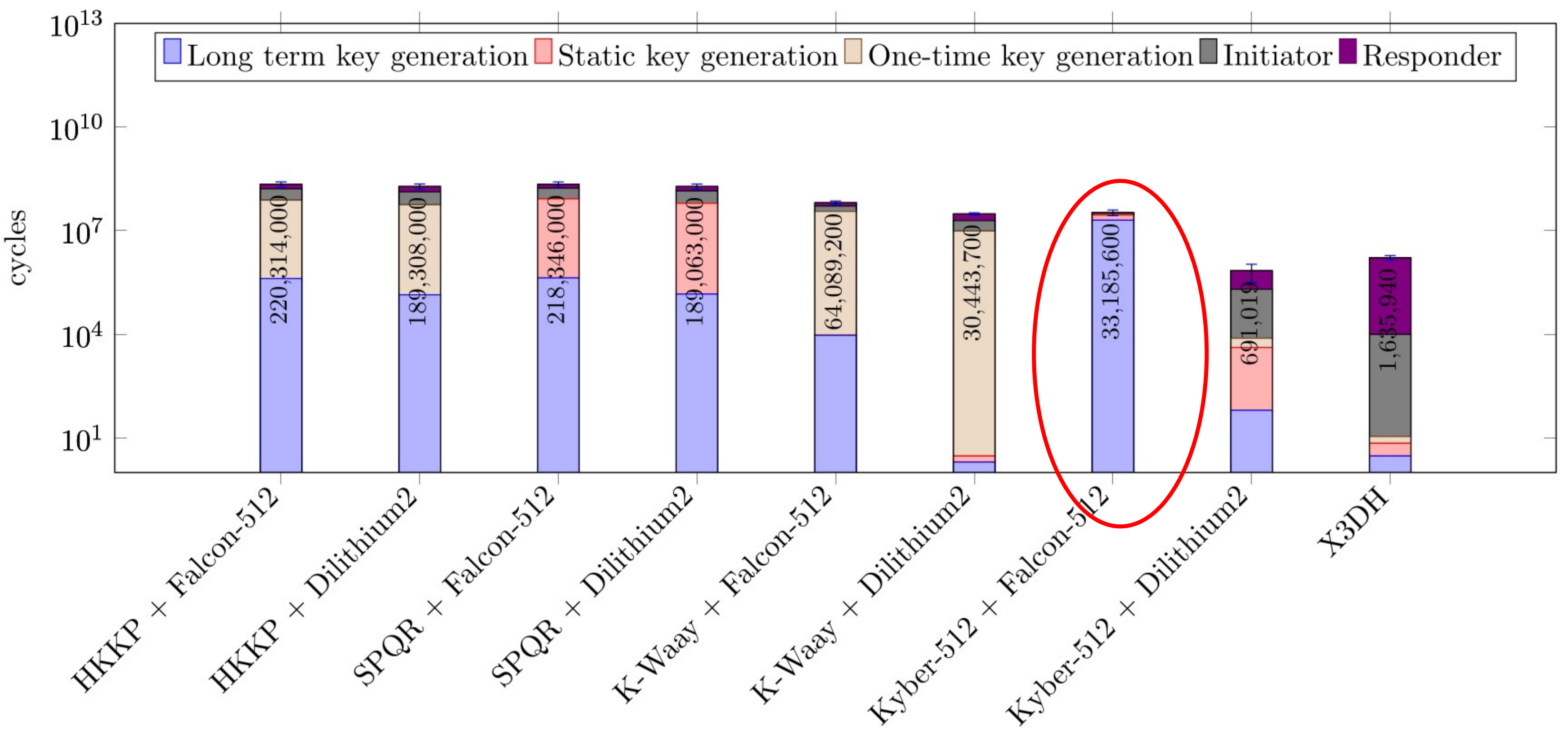

Long term key generation
Static key generation
One-time key generation
Initiator
Responder
cycles
10^13
10^10
10^7
10^4
10^1
220,314,000
189,308,000
218,346,000
189,063,000
64,089,200
30,443,700
33,185,600
691,019
1,635,940
HKKP × Falcon-512
HKKP × Dilithium2
SPQR × Falcon-512
SPQR × Dilithium2
K-Waay × Falcon-512
K-Waay × Dilithium2
Kyber-512 × Falcon-512
Kyber-512 × Dilithium2
X3DH

Speed



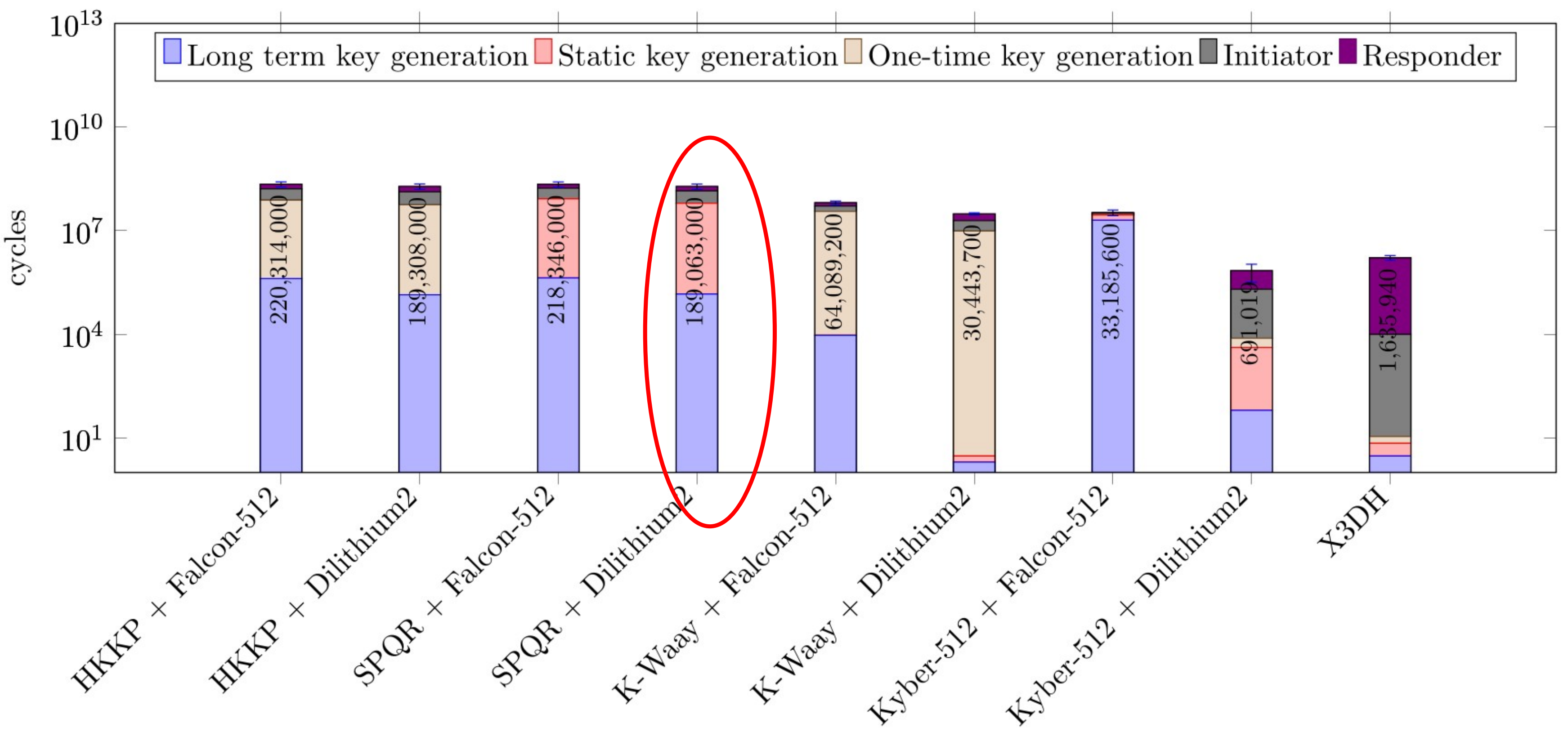
Long term key generation
Static key generation
One-time key generation
Initiator
Responder
cycles
10^13
10^10
10^7
10^4
10^1
220,314,000
189,308,000
218,346,000
189,063,000
64,089,200
30,443,700
33,185,600
691,019
1,635,940
HKKP × Falcon-512
HKKP × Dilithium2
SPQR × Falcon-512
SPQR × Dilithium2
K-Waay × Falcon-512
K-Waay × Dilithium2
Kyber-512 × Falcon-512
Kyber-512 × Dilithium2
X3DH

Speed



Long term key generation
Static key generation
One-time key generation
Initiator
Responder
cycles
10^13
10^10
10^7
10^4
10^1
220,314,000
189,308,000
218,346,000
189,063,000
64,089,200
30,443,700
33,185,600
691,019
1,635,940
HKKP × Falcon-512
HKKP × Dilithium2
SPQR × Falcon-512
SPQR × Dilithium2
K-Waay × Falcon-512
K-Waay × Dilithium2
Kyber-512 × Falcon-512
Kyber-512 × Dilithium2
X3DH

Speed



Long term key generation
Static key generation
One-time key generation
Initiator
Responder
cycles
10^{13}
10^{10}
10^{7}
10^{4}
10^{1}
220,314,000
189,308,000
218,346,000
189,063,000
64,089,200
30,443,700
33,185,600
691,019
1,635,940
HKKP × Falcon-512
HKKP × Dilithium2
SPQR × Falcon-512
SPQR × Dilithium2
K-Waay × Falcon-512
K-Waay × Dilithium2
Kyber-512 × Falcon-512
Kyber-512 × Dilithium2
X3DH

Size

Scheme	|lpk|	|prek|	|ct|
K-Waay + Dilithium	2112	24520	1632
K-Waay + Falcon	1697	22790	1632
HKKP [Has+22]	1700	1700	4056
HKKP [Has+22] + Dilithium2	3012	4120	4056
HKKP [Has+22] + Falcon	2597	2390	4056
SPQR [Bre+22]	3400	1632	4824
SPQR [Bre+22] + Dilithium2	4712	4052	4824
SPQR [Bre+22] + Falcon	4297	2322	4824

Wrapping Up

- K–Waay: faster deniable PQ X3DH.

- What about split–KEM from module LWE?
  - No fundamental barrier.
  - Keys should be smaller.

- Thanks! @dcol97 on X

Full version: https://eprint.iacr.org/2024/120

Bibliography

- [BCD+16]: Bos, Costello, Ducas, Mironov, Naehrig, Nikolaenko, Raghunathan, Stebila: Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. CCS 2016
- [CCDGS17/20]: Cohn-Gordon, Cremers, Dowling, Garratt, Stebila: A Formal Security Analysis of the Signal Messaging Protocol. EuroS&P 2017/JoC 2020
- [BDK+18]: Bos, Ducas, Kiltz, Lepoint, Lyubashevsky, Schnack, Schwabe, Seiler, Stehle: CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM. EuroS&P 2018
- [ACD19]: Alwen, Coretti, Dodis: The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol. EUROCRYPT 2019
- [LAZ19]: Lu, Au, Zhang: Raptor: A Practical Lattice-Based (Linkable) Ring Signature. ACNS 2019
- [BFGJS20]: Brendel, Fischlin, Günther, Janson, Stebila: Towards Post-Quantum Security for Signal's X3DH Handshake. SAC 2020
- [VGIK20]: Vatandas, Gennaro, Ithurburn, Krawczyk: On the Cryptographic Deniability of the Signal Protocol. ACNS 2020
- [HKKP21/22]: Hashimoto, Katsumata, Kwiatkowski, Prest: An Efficient and Generic Construction for Signal's Handshake (X3DH): Post-Quantum, State Leakage Secure, and Deniable. PKC 2021/JoC 2022
- [BFGJS22]: Brendel, Fiedler, Günther, Janson, Stebila: Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake. PKC 2022
- [HV22]: Huguenin-Dumittan, Vaudenay: On IND-qCCA Security in the ROM and Its Applications. EUROCRYPT 2022

Backup Slide: Comparison Table

Protocol	PQ Conf	PQ Auth	KCI	FS	SSR	RR	Deniability
X3DH [MP16; Coh+20]	✗	✗	✓	PFS	✓	✓	Malicious
PQXDH [KS23; Bha+23]	✓	✗	✓	PFS	✗	✗	Semi-honest+
KEM+Sigs [Has+22]	✓	✓	✓	PFS	✓	✗	✗
HKKP [Has+22]	✓	✓	✓	WFS	✓	✗	Semi-honest
SPQR [Bre+22]	✓	✓	✓	WFS	✗	✓	Semi-honest
K-Waay (Section 5)	✓	✓	✓	WFS	✓	✗	Semi-honest

Backup Slide: Our DAKE Formalism

- **We propose a new Deniable Authenticated Key Exchange (DAKE) formalism to capture K-Waay.**
- **Init(sk, role) → (st, prek): outputs a temporary state and prekey bundle prek.**
- **Send(sk, pk, st, prek) → (k, m).**
- **BatchReceive(sk, st, $\{pk_j, prek_j, m_j\}_j$) → $\{k_s\}_s$: captures key reuse (concurrency).**
  - Split-KEM doesn't need full IND-CCA security, and is thus very efficient!

Our Split-KEM Security Notions (1)

- **UNF-1KCA: adversary cannot forge a ciphertext when given a ciphertext (from Alice).**
- **Note Signal users upload 100 prekey bundles, but they can run out…**
- **Ind-1BatchCCA: ephemeral key reuse *without* full CCA security!**
- **Q(ROM) transform à la [HV22] to achieve these two:**
  - Adds an extra hash $H(pk_A, pk_B, ct, K)$ to Alice's ciphertext.

Our Split-KEM Security Notions (2)

- **Semi-honest deniability.**
- **Even if the adversary is given one split-KEM secret (say Alice's).**
- **Implies semi-honest AKE deniability (morally stronger than [BFGJS22]).**