Anonymous, Robust Post-Quantum Public Key Encryption

Varun Maram
Applied Cryptography Group
ETH Zurich

APPLIED CRYPTO GROUP

Joint work with Paul Grubbs and Kenneth G. Paterson

[Full version of paper: https://eprint.iacr.org/2021/708.pdf]

NIST PQC Round-3 KEMs

PQC Standardization Process: Third Round Candidate Announcement

NIST is announcing the third round finalists of the NIST Post-Quantum Cryptography Standardization Process. More details are included in NISTIR 8309.

July 22, 2020

It has been almost a year and a half since the second round of the NIST PQC Standardization Process began. After careful consideration, NIST would like to announce the candidates that will be moving on to the third round.

Third Round Finalists	Alternate Candidates
Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

ORGANIZATIONS

Information Technology Laboratory

Computer Security Division

Cryptographic Technology Group

NIST PQC Round-3 KEMs

PQC Standardization Process: Third Round Candidate Announcement

NIST is announcing the third round finalists of the NIST Post-Quantum Cryptography Standardization Process. More details are included in NISTIR 8309.

July 22, 2020

It has been almost a year and a half since the second round of the NIST PQC Standardization Process began. After careful consideration, NIST would like to announce the candidates that will be moving on to the third round.

Third Round Finalists	Alternate Candidates
Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

ORGANIZATIONS

Information Technology Laboratory

Computer Security Division

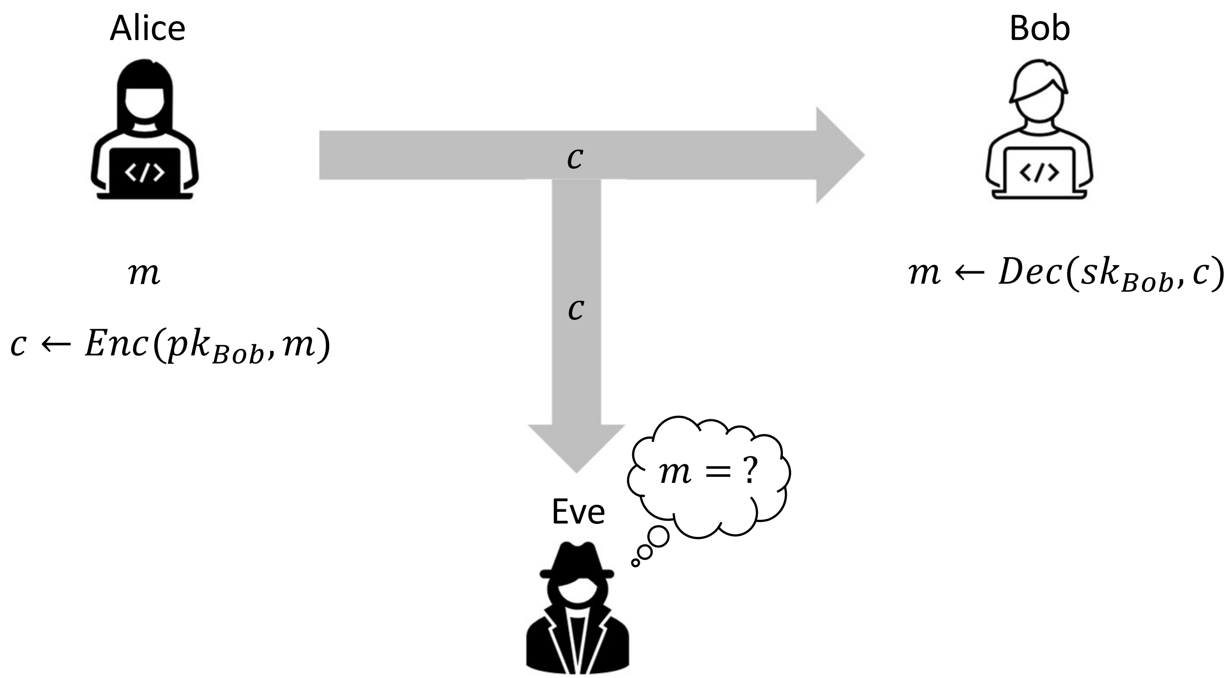
Cryptographic Technology Group

4.A.2 Security Definition for Encryption/Key-Establishment

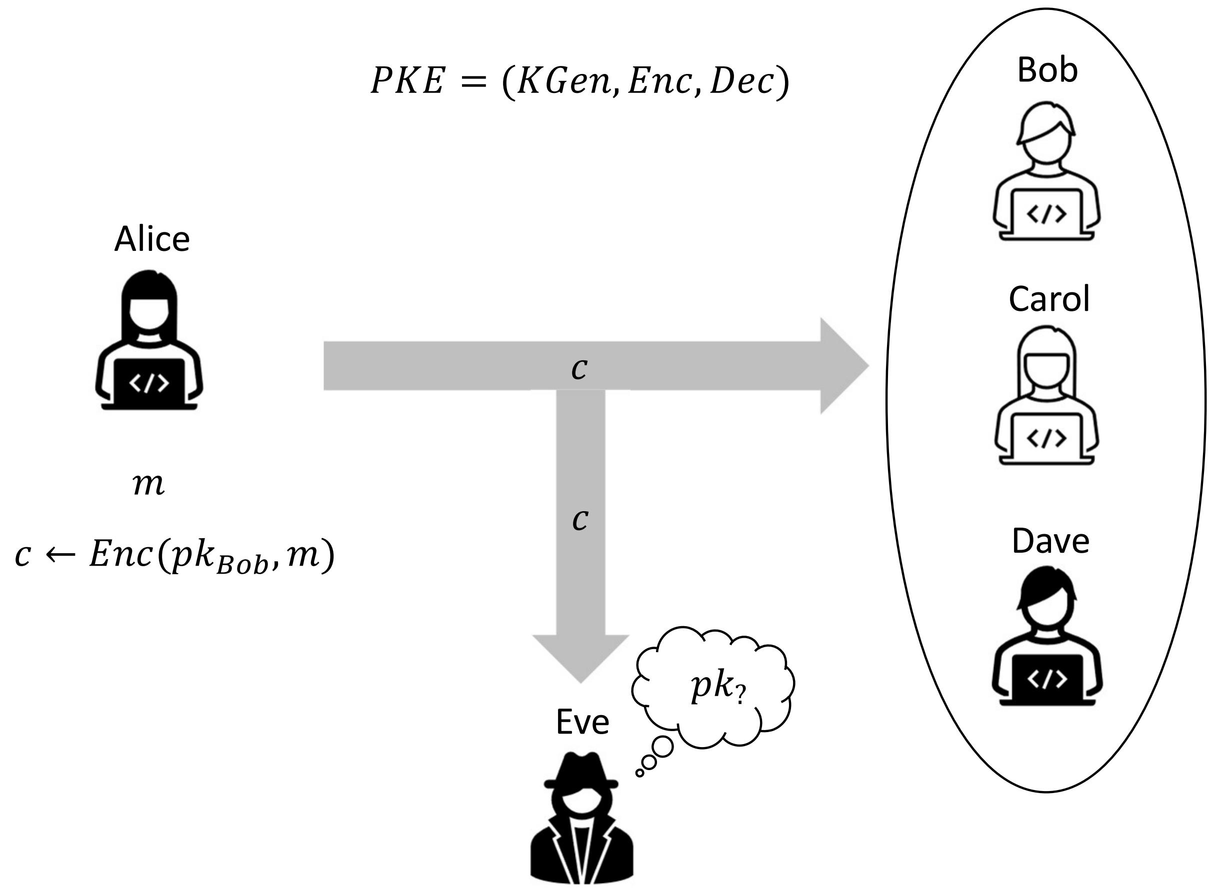
NIST intends to standardize one or more schemes that enable "semantically secure" encryption or key encapsulation with respect to adaptive chosen ciphertext attack, for general use. This property is generally denoted *IND-CCA2 security* in academic literature.

IND-CCA Security

$$PKE = (KGen, Enc, Dec)$$



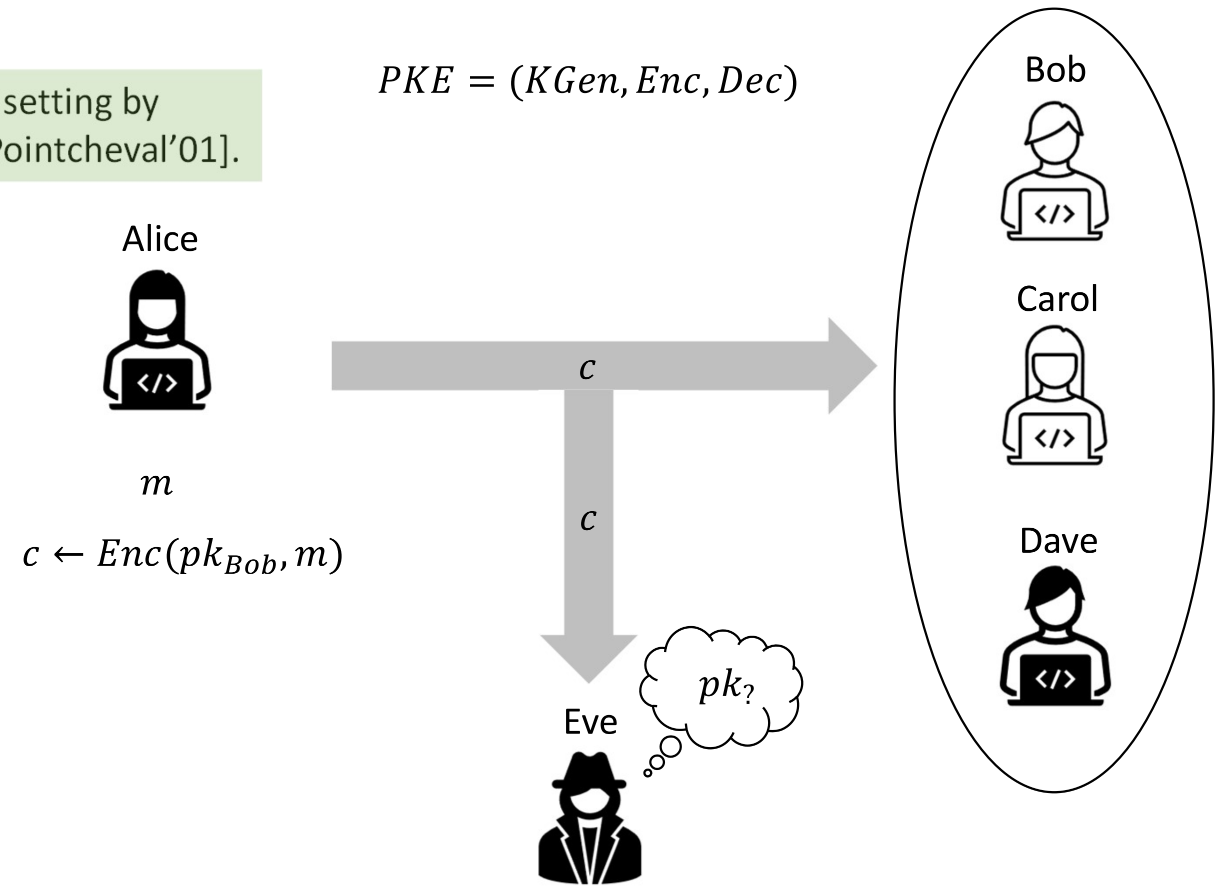
Anonymity (ANO-CCA security)



Anonymity (ANO-CCA security)

Formalized in a public-key setting by [Bellare-Boldyreva-Desai-Pointcheval'01].

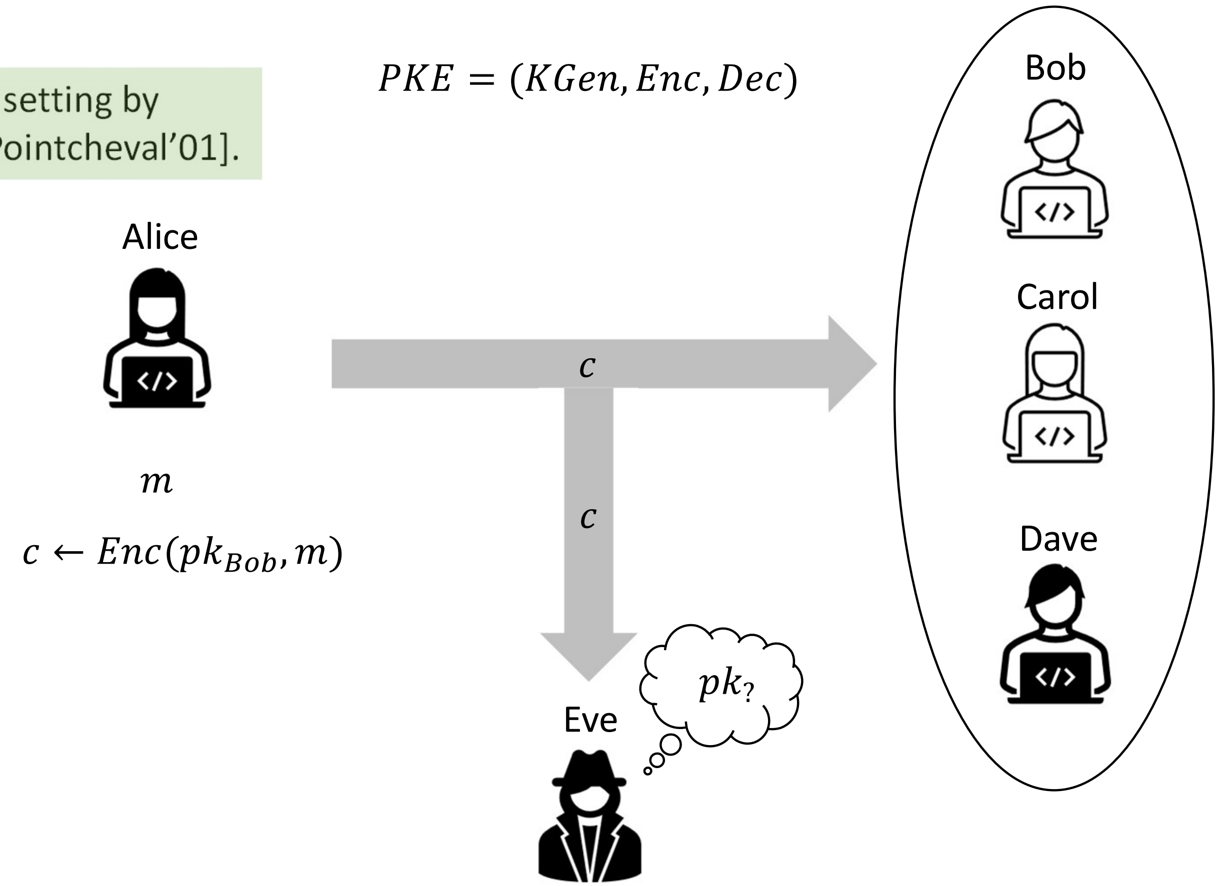
$PKE = (KGen, Enc, Dec)$



Anonymity (ANO-CCA security)

Formalized in a public-key setting by [Bellare-Boldyreva-Desai-Pointcheval'01].

$PKE = (KGen, Enc, Dec)$



Anonymity (ANO-CCA security)

Formalized in a public-key setting by [Bellare-Boldyreva-Desai-Pointcheval’01].

$PKE = (KGen, Enc, Dec)$

Alice

m

$c \leftarrow Enc(pk_{Bob}, m)$

c

c

Eve

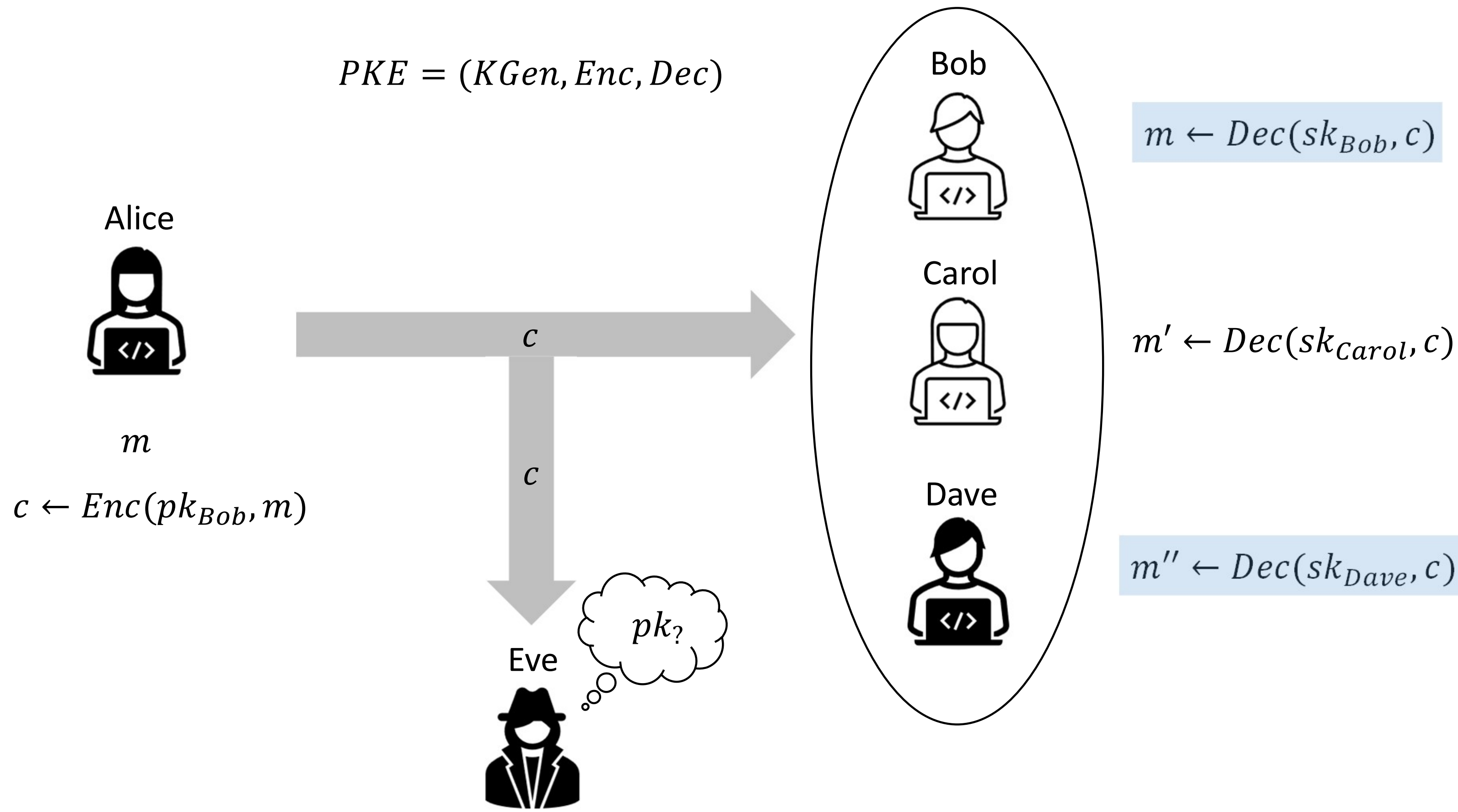
$pk_?$

Bob

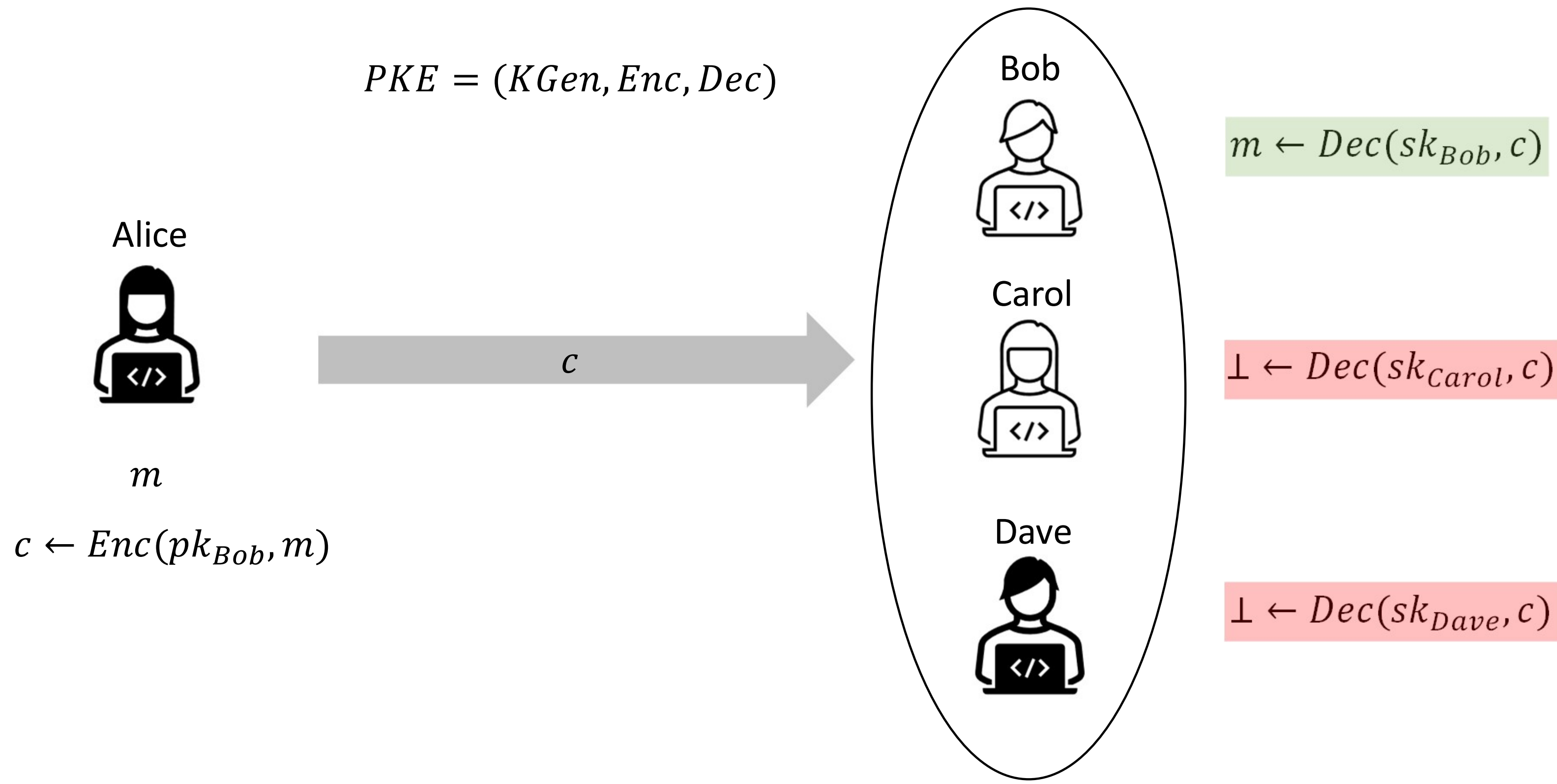
Carol

Dave

Anonymity (ANO-CCA security)



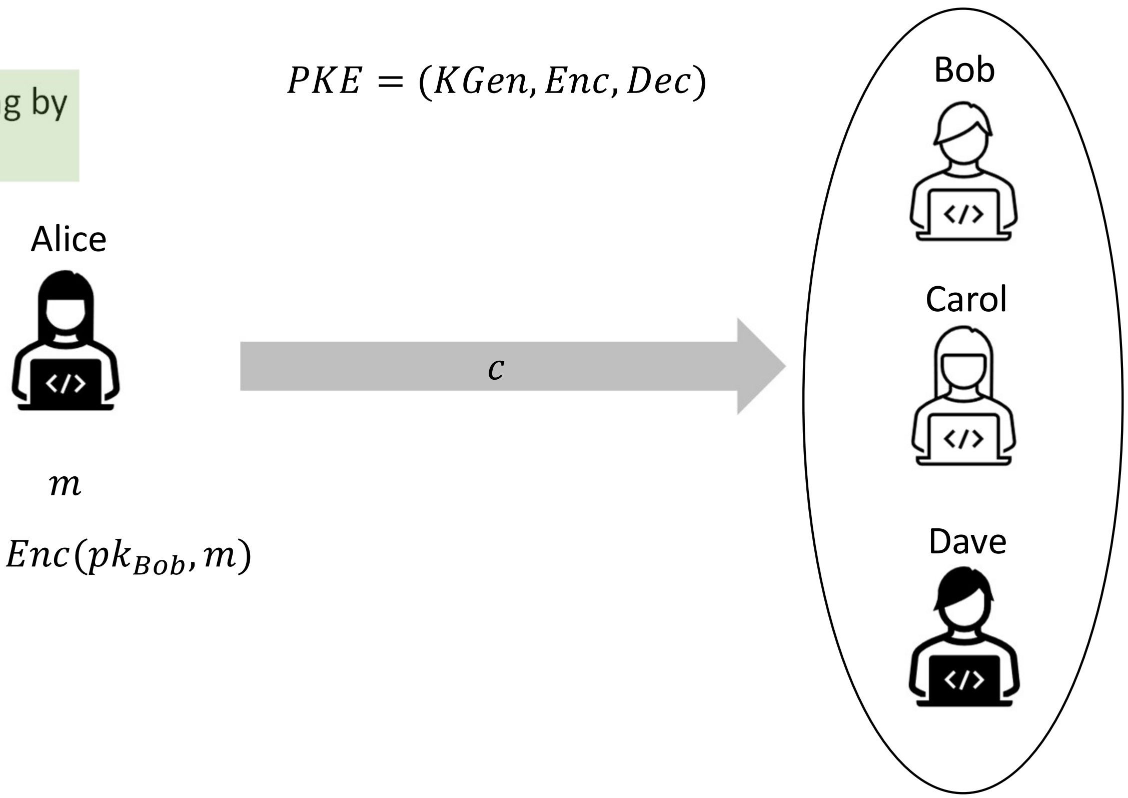
Robustness (SROB-CCA security)



Robustness (SROB-CCA security)

Formalized in a public-key setting by [Abdalla-Bellare-Neven'10].

$PKE = (KGen, Enc, Dec)$



m

$c \leftarrow Enc(pk_{Bob}, m)$

$m \leftarrow Dec(sk_{Bob}, c)$

$\perp \leftarrow Dec(sk_{Carol}, c)$

$\perp \leftarrow Dec(sk_{Dave}, c)$

KEM-DEM Paradigm

Public-Key Encryption/KEMs

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

Public-Key Encryption/KEMs

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

KEM-DEM Paradigm

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

$$PKE = (KGen, Enc, Dec)$$

IND-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

$KEM = (KGen, Encap, Decap)$

KEM

IND-CCA secure

Public-Key Encryption/KEMs

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

$PKE = (KGen, Enc, Dec)$

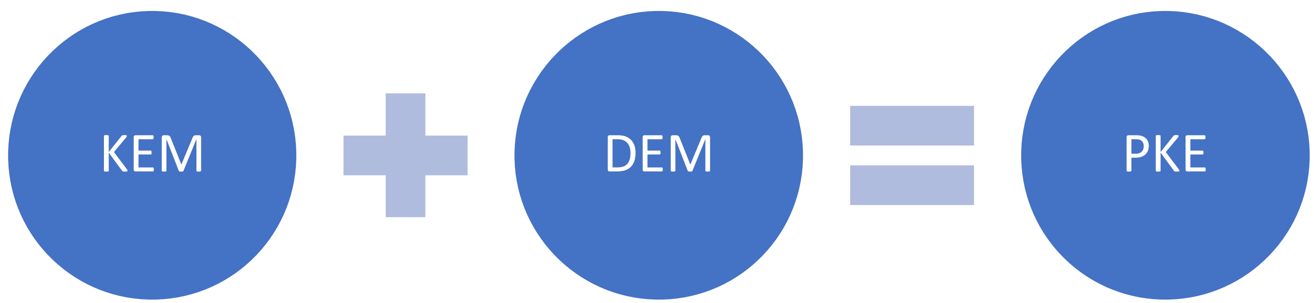
PKE

IND-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



IND-CCA secure

(one-time) authenticated encryption

IND-CCA secure ✓

KEM-DEM Paradigm

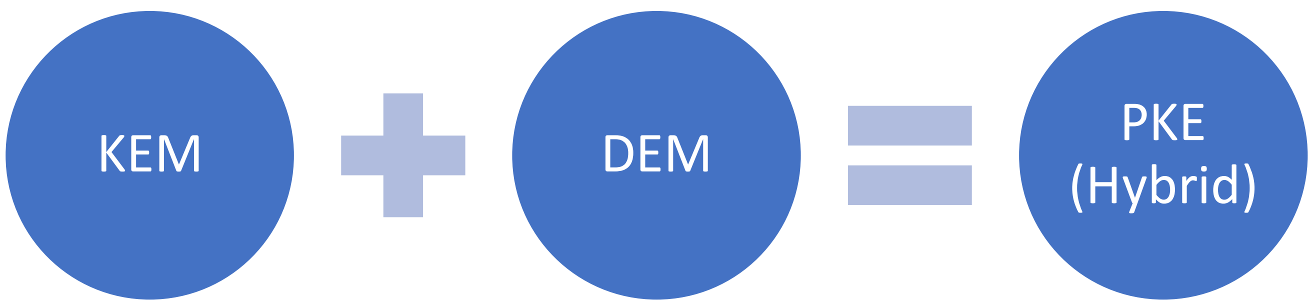
Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA secure

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

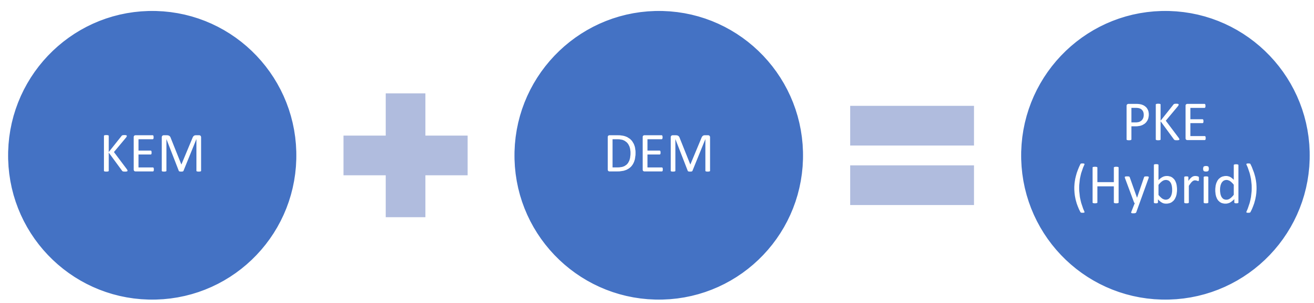
$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



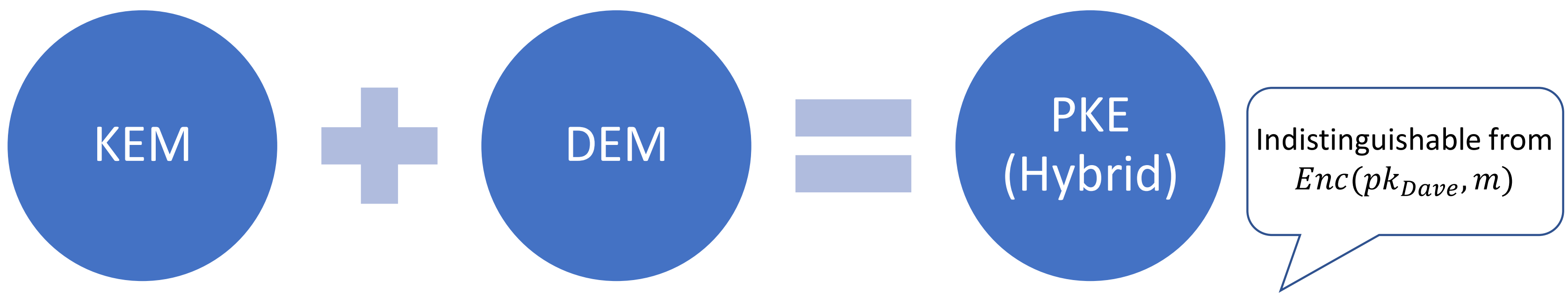
$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure +
ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure +
ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

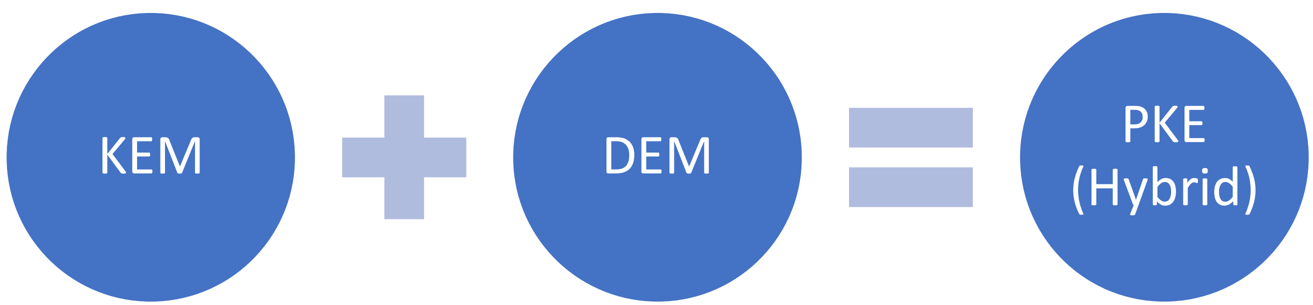
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

KEM + DEM = PKE (Hybrid)

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

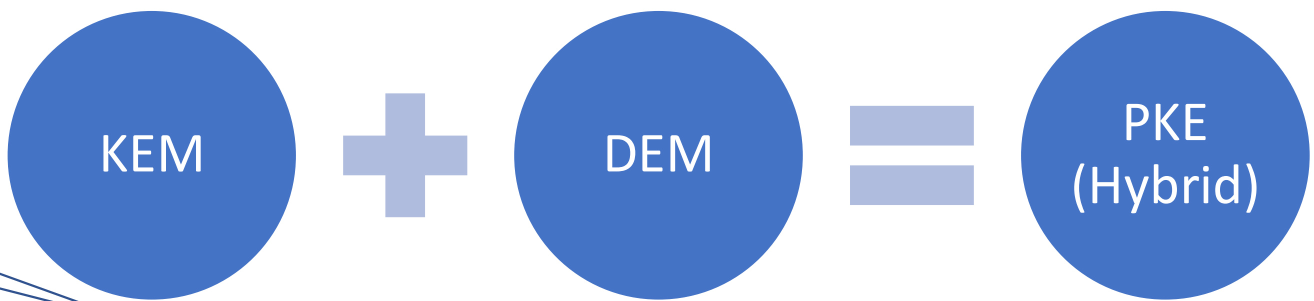
KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

Indistinguishable from $Encap(pk_{Dave})$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA + ANO-CCA secure

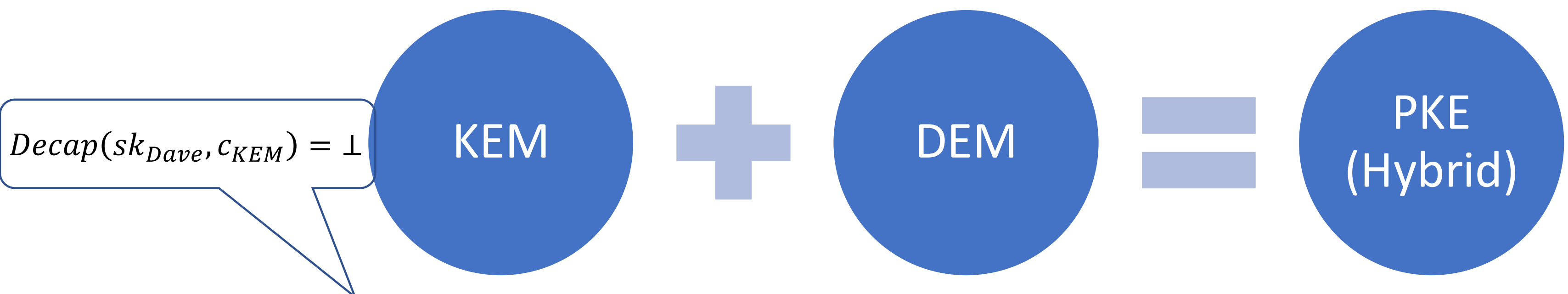
IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson’22]; generalization of [Mohassel’10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

KEM + DEM = PKE (Hybrid)

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure ✓

KEM-DEM Paradigm

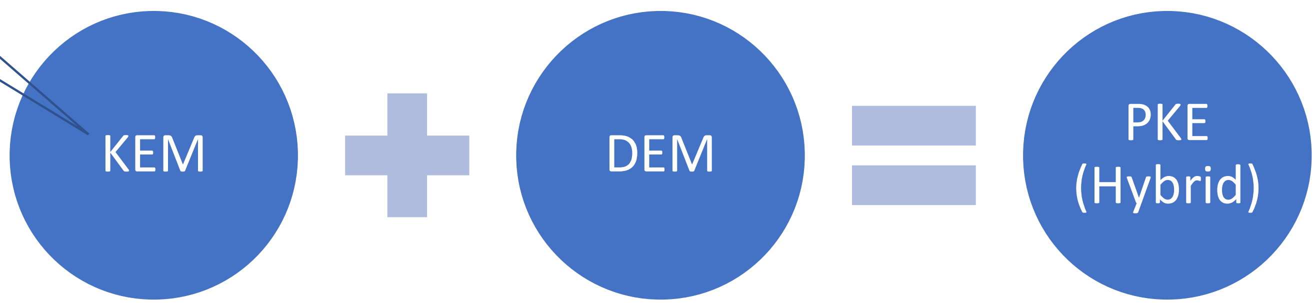
Public-Key Encryption/KEMs
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs
- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

[Mohassel'10] only considered KEMs constructed <u>directly</u> from PKE schemes.

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

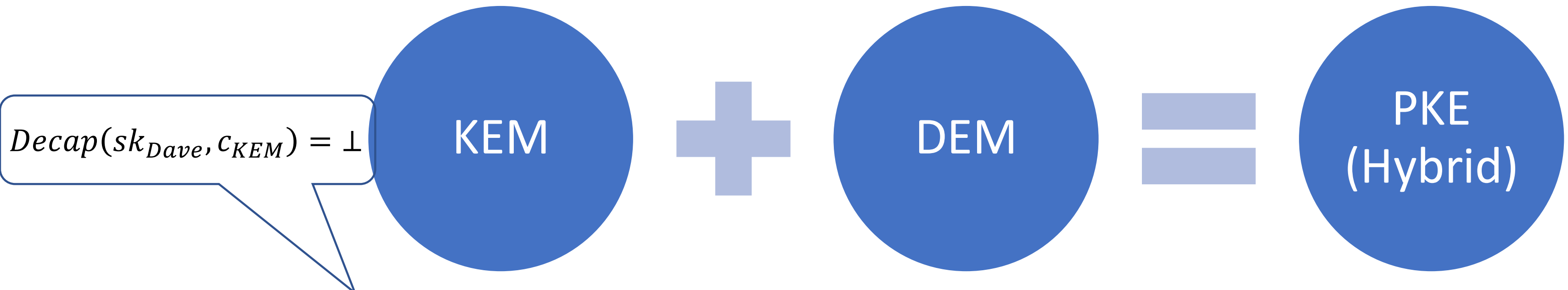
KEM-DEM Paradigm

Public-Key Encryption/KEMs
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs
- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

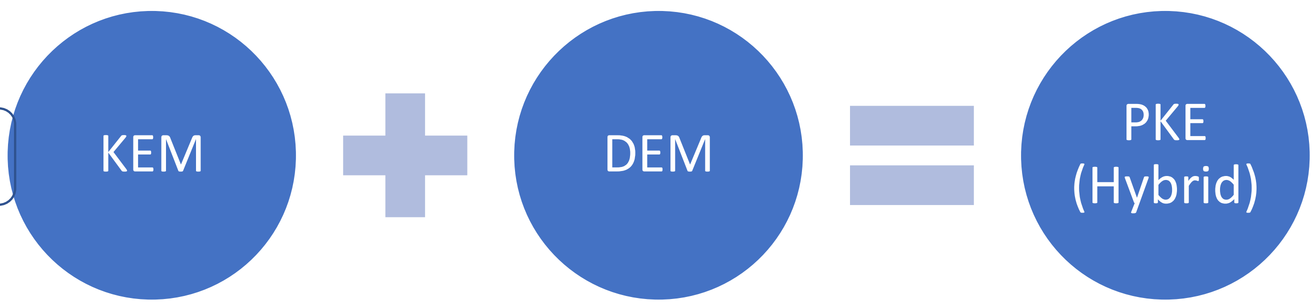
KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$Decap(sk_{Dave}, c_{KEM}) = \bot$



... is also necessary.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure
+ weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

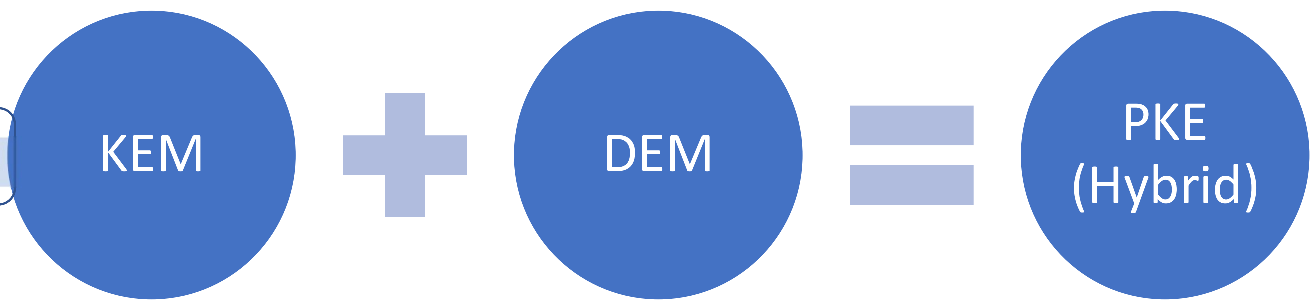
KEM-DEM Paradigm

Public-Key Encryption/KEMs
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs
- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$Decap(sk_{Dave}, c_{KEM}) = \perp$

... is also necessary.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

"Implicit-rejection" KEMs!

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

KEM + DEM = PKE (Hybrid)

$Decap(sk_{Dave}, c_{KEM}) = \perp$

... is also necessary.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

“Implicit-rejection” KEMs!

Cannot be even weakly robust.

Public-Key Encryption/KEMs
- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson’22]; generalization of [Mohassel’10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$Decap(sk_{Dave}, c_{KEM}) = \perp$

KEM + DEM = PKE (Hybrid)

... is also necessary.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

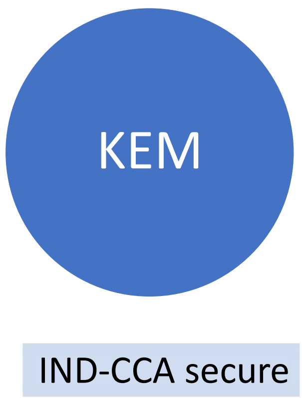
$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

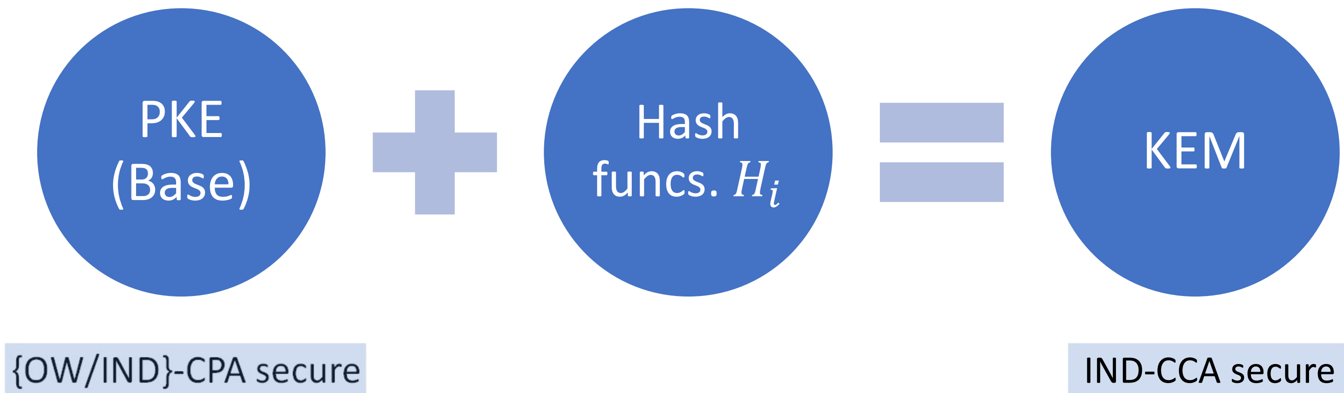
IND-CCA secure + ANO-CCA secure

Fujisaki-Okamoto Transformation

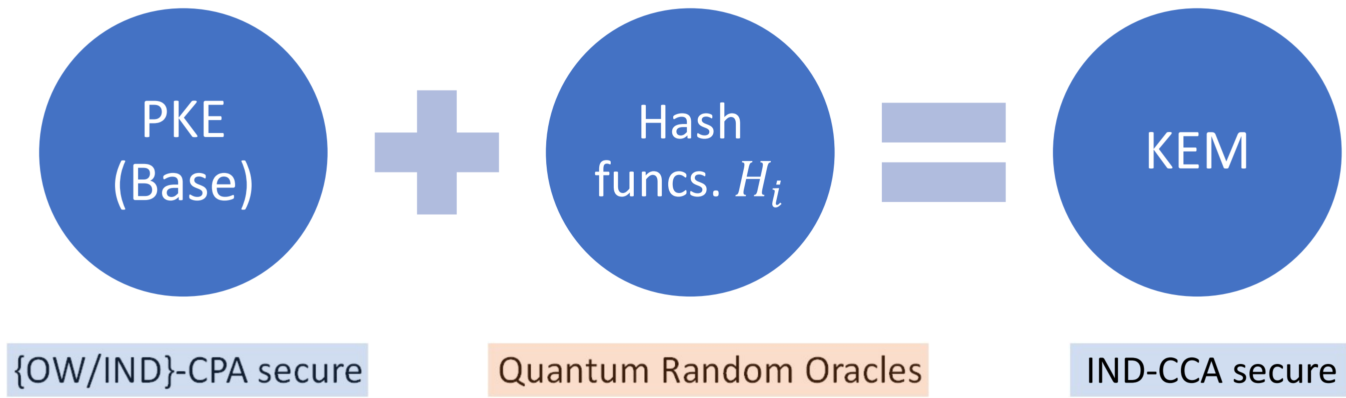


Fujisaki-Okamoto Transformation

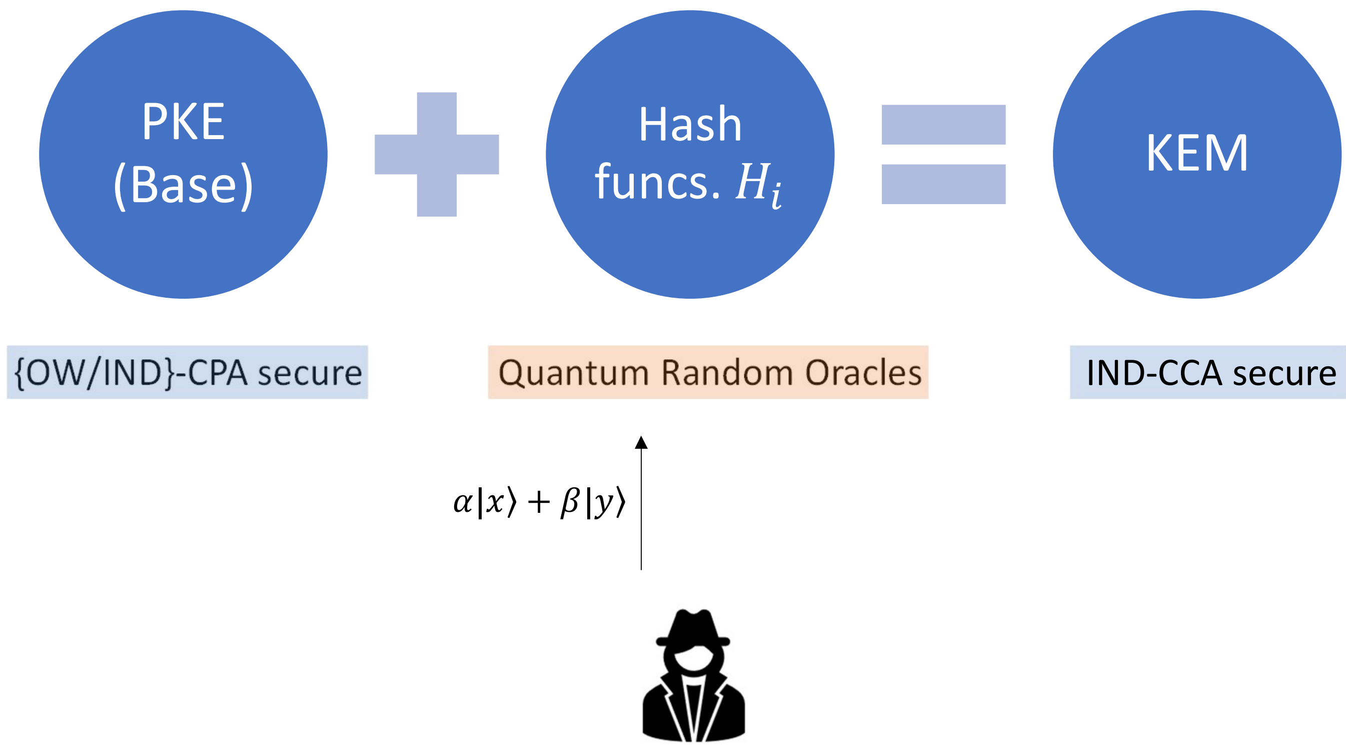
Fujisaki-Okamoto Transformation



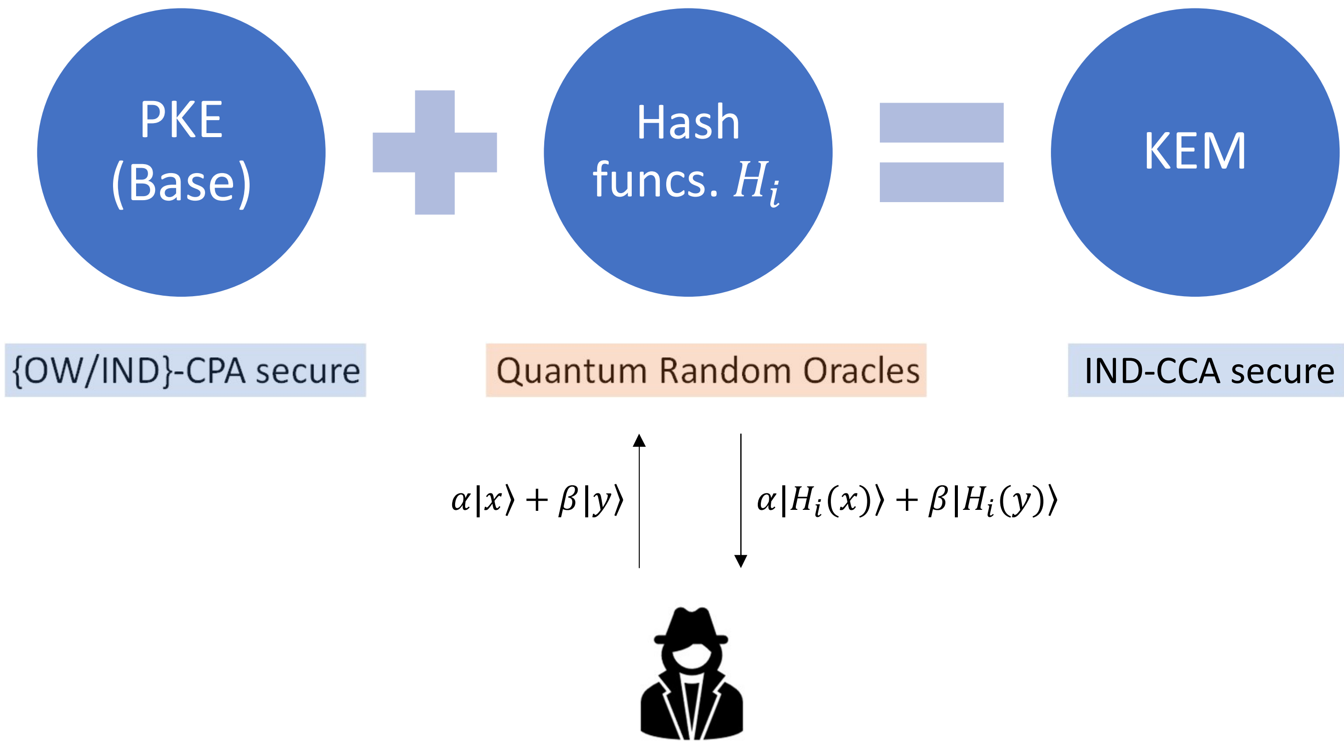
Fujisaki-Okamoto Transformation



Fujisaki-Okamoto Transformation



Fujisaki-Okamoto Transformation



Fujisaki-Okamoto Transformation

Classic McEliece

CRYSTALS-KYBER

SABER

NTRU

Fujisaki-Okamoto Transformation

- Classic McEliece
- CRYSTALS-KYBER
- SABER
- NTRU

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathsf{FO}^{\not\perp}$

Fujisaki-Okamoto Transformation

Classic McEliece

CRYSTALS-KYBER

SABER

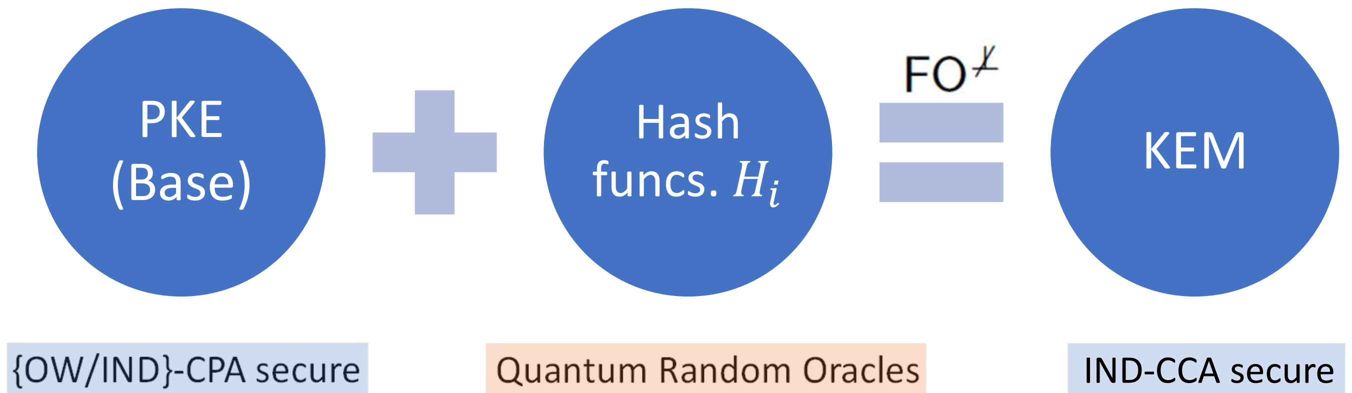
NTRU

FrodoKEM

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

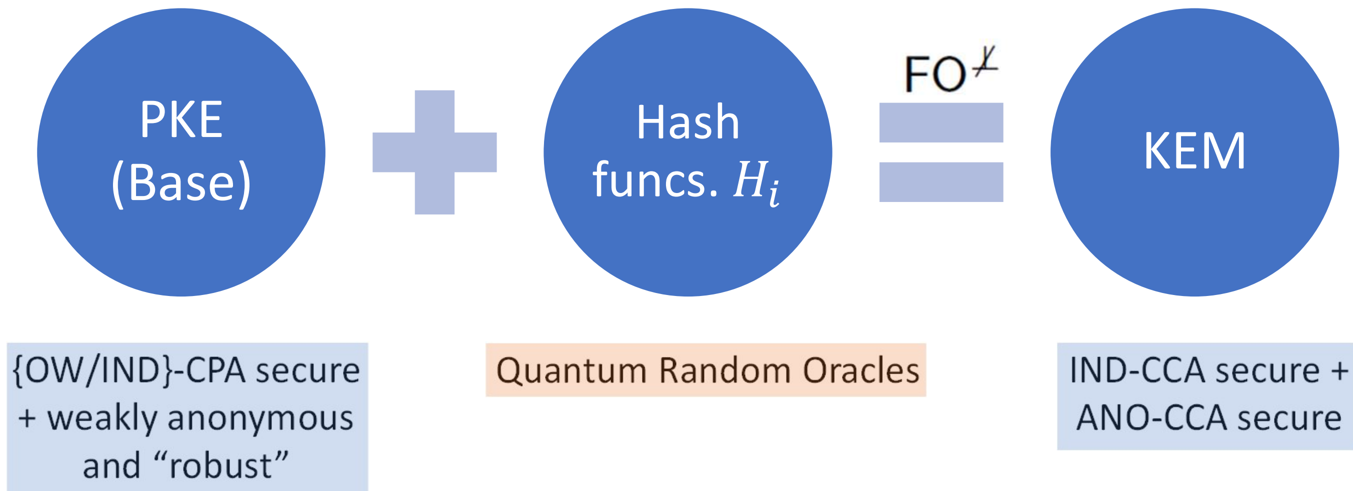
$\mathsf{FO}^{\not\perp}$

Anonymity from FO transforms



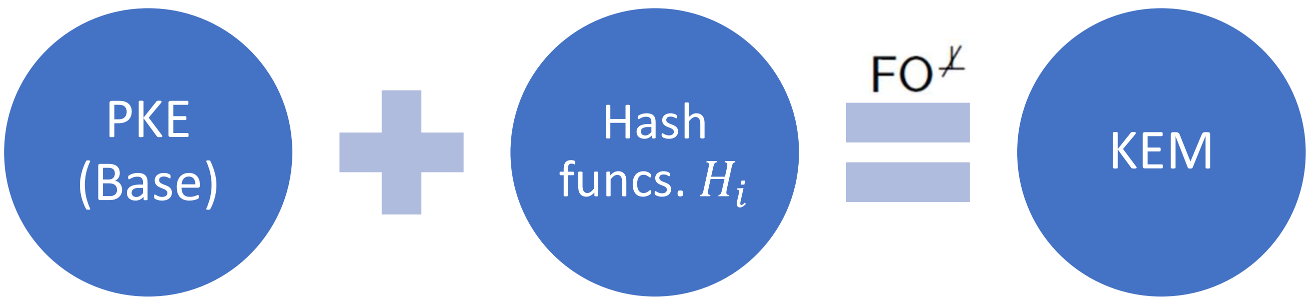
Shown in [Jiang-Zhang-Chen-Wang-Ma’18]

Anonymity from FO transforms



Shown in [Grubbs-Maram-Paterson'22]

Anonymity from FO transforms



{OW/IND}-CPA secure + weakly anonymous and "robust"

Quantum Random Oracles

IND-CCA secure + ANO-CCA secure

Shown in [Grubbs-Maram-Paterson'22]

Extended [Jiang et. al.'18]'s proof techniques from a *single-key* setting (IND-CCA) to a *two-key* setting (ANO-CCA).

KEM-DEM Paradigm

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

"Implicit-rejection" KEMs!

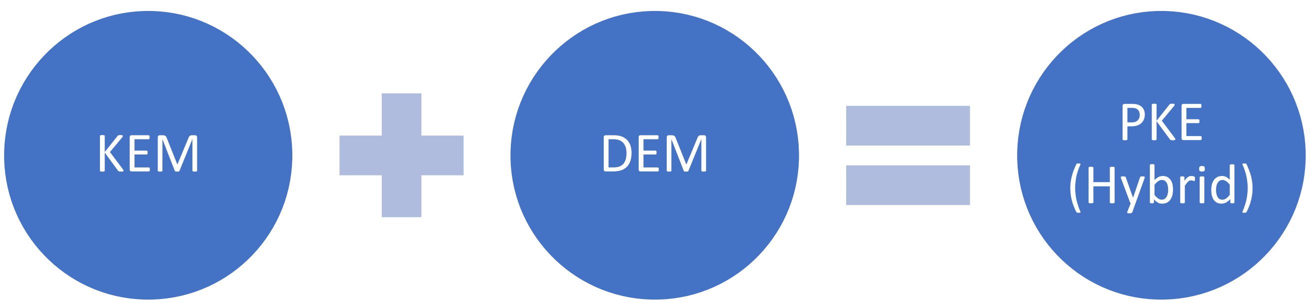
Cannot be even weakly robust.

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + weakly robust

... is also necessary.

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

"Implicit-rejection" KEMs!

Cannot be even weakly robust.

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$FO^{\pm}$-based KEM + DEM = PKE (Hybrid)

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

KEM-DEM Paradigm

Public-Key Encryption/KEMs

Classic McEliece
CRYSTALS-KYBER
NTRU
SABER

"Implicit-rejection" KEMs!

Cannot be even weakly robust.

Public-Key Encryption/KEMs

BIKE
FrodoKEM
HQC
NTRU Prime
SIKE

Shown in [Grubbs-Maram-Paterson'22]; generalization of [Mohassel'10].

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$c \leftarrow Enc^{base}(pk_{Bob}, m)$ should have large enough entropy.

$FO^{\pm}$-based KEM + DEM = PKE (Hybrid)

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

Classic McEliece (CM)

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$FO^{\pm}$-based KEM + DEM = PKE (Hybrid)

$c \leftarrow Enc^{base}(pk_{Bob}, m)$ should have large enough entropy.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

Classic McEliece (CM)

<u>Public-Key Encryption/KEMs</u>

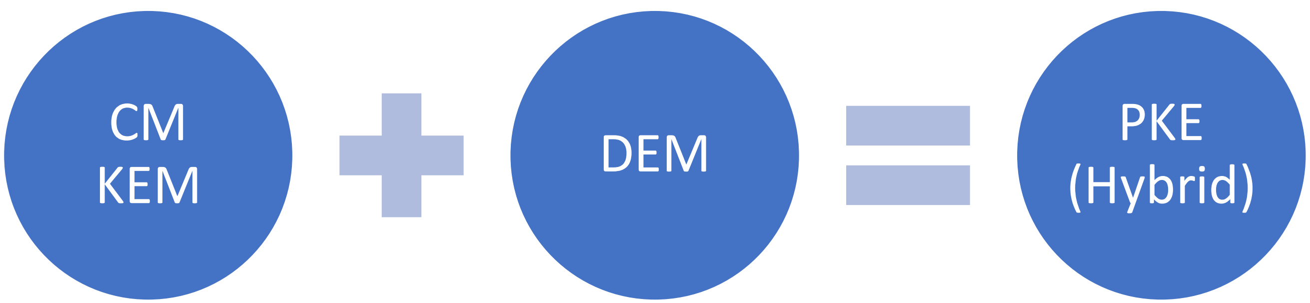
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

<u>Public-Key Encryption/KEMs</u>

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

CM uses a *deterministic* base PKE scheme.



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

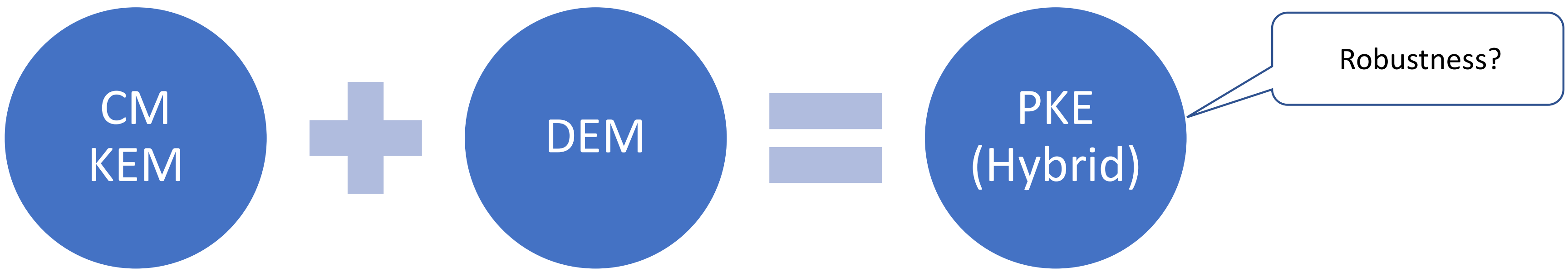
(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

Classic McEliece (CM)

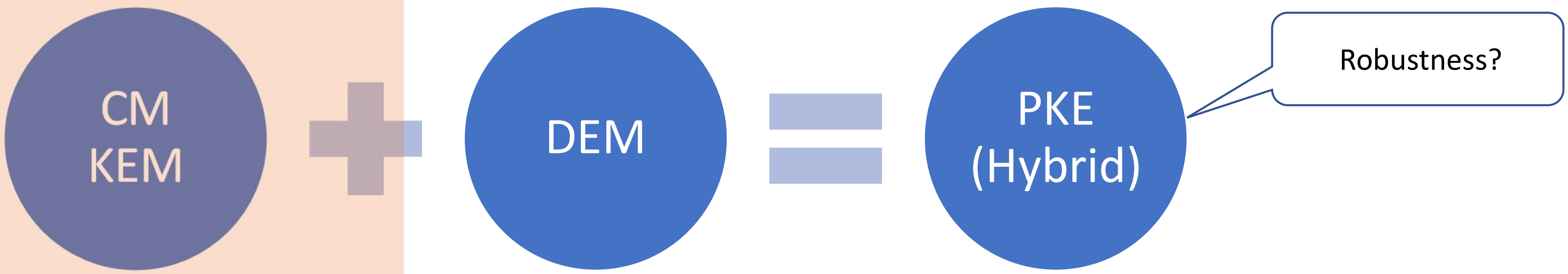
$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

Classic McEliece (CM)

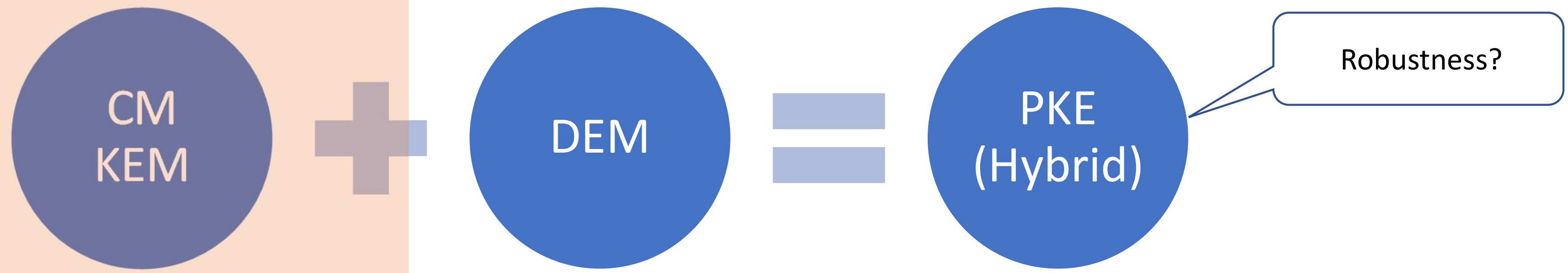
$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

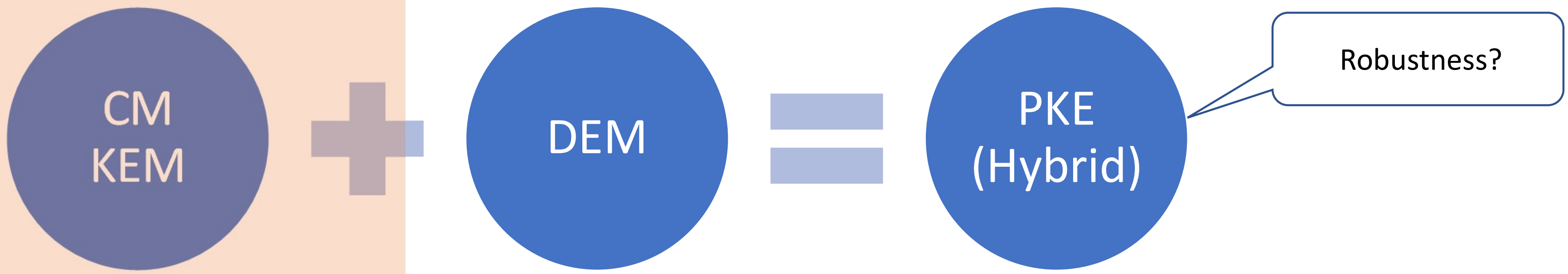
For *any* message m,

we can construct a ciphertext

$c \leftarrow (c_{KEM}, c_{DEM})$ such that,

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

For *any* message m,

we can construct a ciphertext

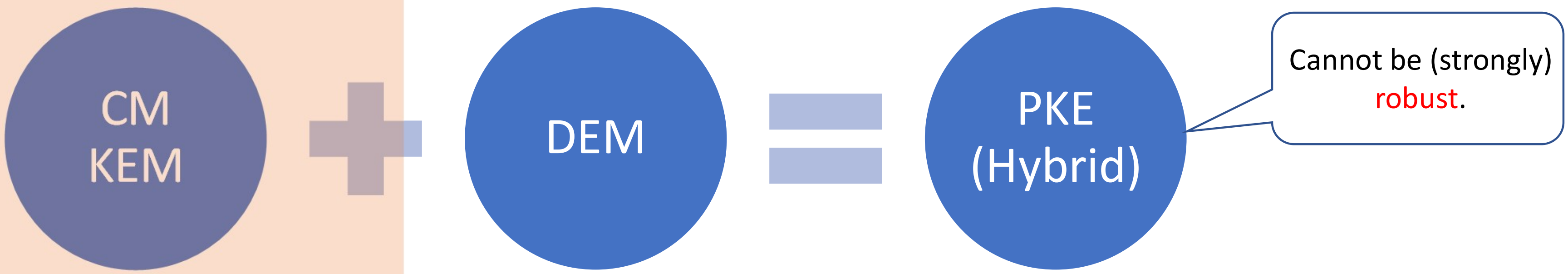
$c \leftarrow (c_{KEM}, c_{DEM})$ such that,

for *any* CM private key sk_* ,

$Dec(sk_*, c) = m\ (\neq \bot)$!

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



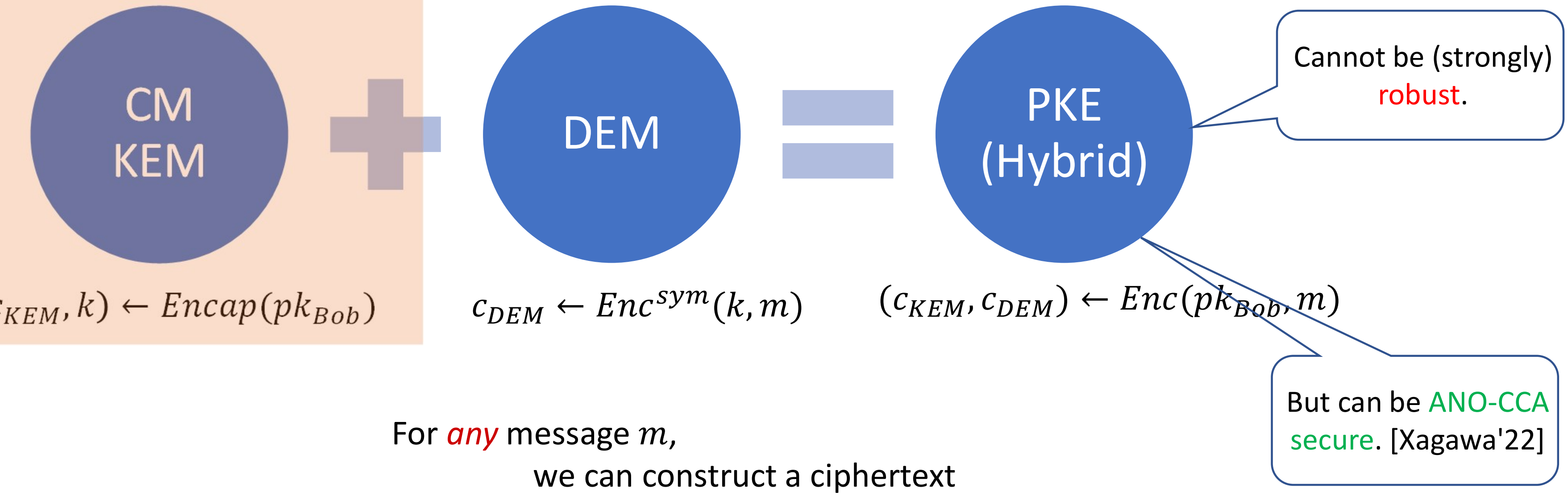
$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

For *any* message m,
we can construct a ciphertext
$c \leftarrow (c_{KEM}, c_{DEM})$ such that,

for *any* CM private key sk_* ,
$Dec(sk_*, c) = m \ (\neq \perp)!$

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $\quad DEM = (Enc^{sym}, Dec^{sym})$ $\quad PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $\quad c_{DEM} \leftarrow Enc^{sym}(k, m)$ $\quad (c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

For *any* message m,

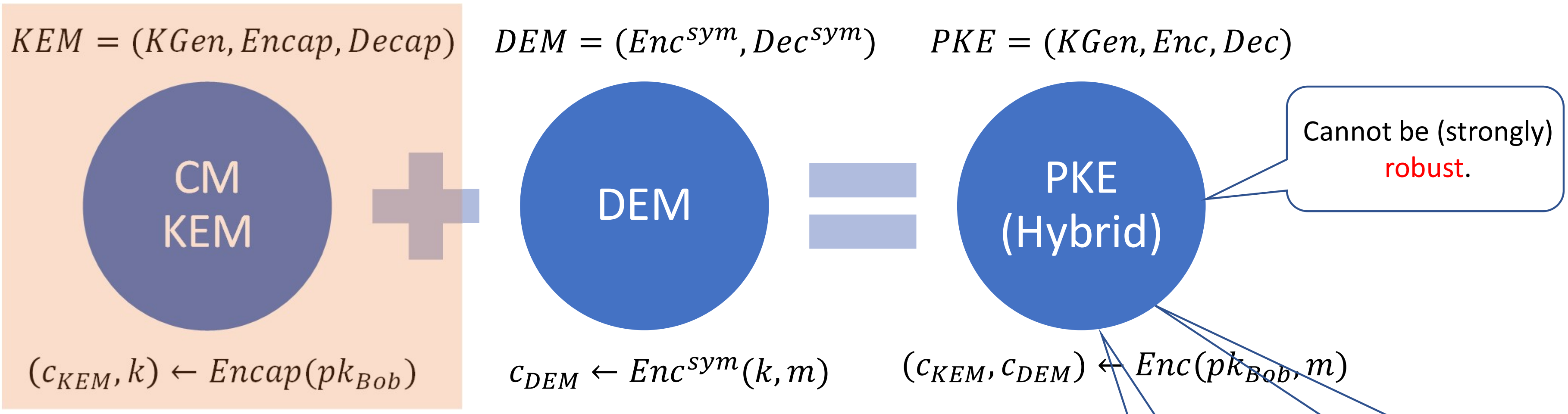
we can construct a ciphertext

$c \leftarrow (c_{KEM}, c_{DEM})$ such that,

for *any* CM private key sk_* ,

$Dec(sk_*, c) = m \; (\neq \bot)!$

Classic McEliece (CM)



For *any* message m,
we can construct a ciphertext
$c \leftarrow (c_{KEM}, c_{DEM})$ such that,

for *any* CM private key sk_* ,
$Dec(sk_*, c) = m\ (\neq \perp)$!

CRYSTALS-KYBER and SABER

Public-Key Encryption/KEMs

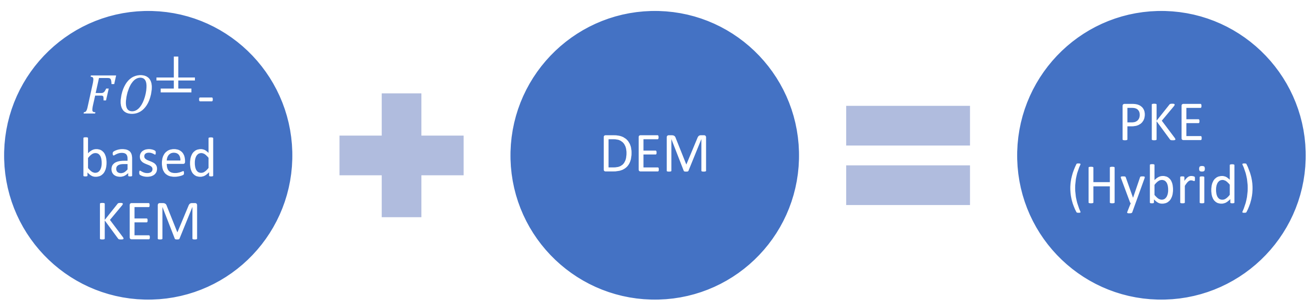
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$c \leftarrow Enc^{base}(pk_{Bob}, m)$ should have large enough entropy.



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

Public-Key Encryption/KEMs

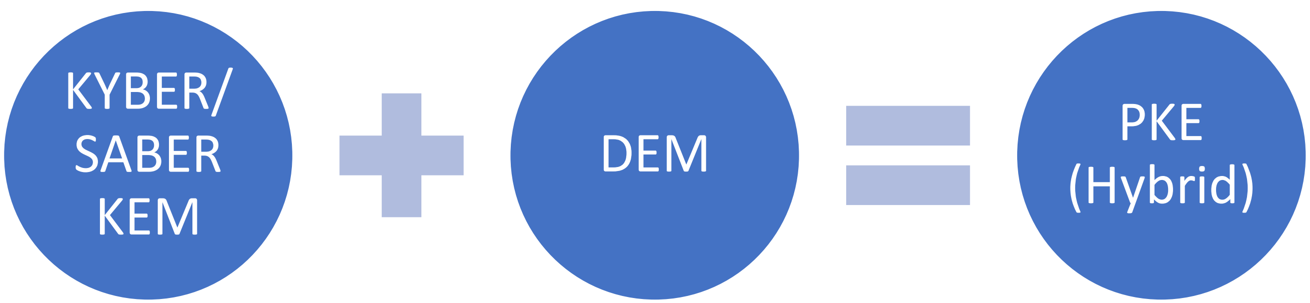
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

$c \leftarrow Enc^{base}(pk_{Bob}, m)$ should have large enough entropy.



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

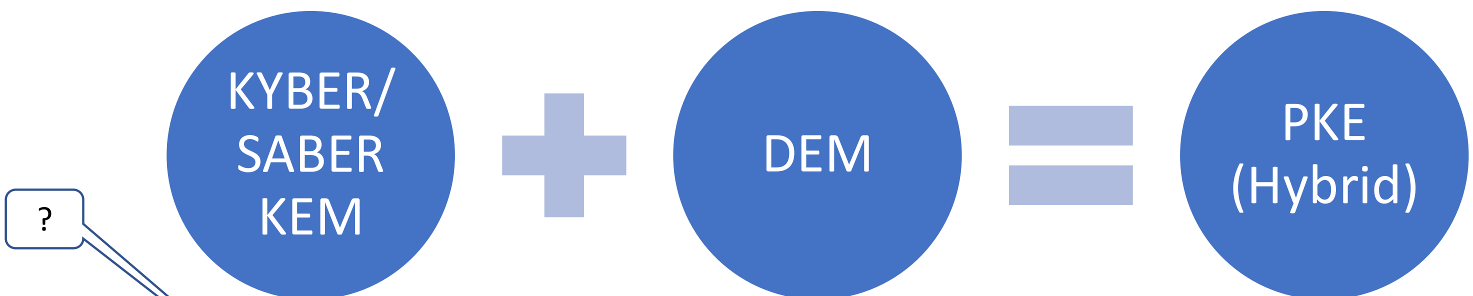
<u>Public-Key Encryption/KEMs</u>

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

<u>Public-Key Encryption/KEMs</u>

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $\quad DEM = (Enc^{sym}, Dec^{sym})$ $\quad PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$r \leftarrow G(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$r' \leftarrow G(m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathsf{FO}^{\not\perp}$

CRYSTALS-KYBER and SABER

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$r \leftarrow G(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$r' \leftarrow G(m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathsf{FO}^{\not\perp}$

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$
$s \leftarrow_{\$} \mathcal{M}$	$m \leftarrow F(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' \leftarrow (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$	$(\hat{k}, r) \leftarrow G(F(pk), m)$	$(\hat{k}', r') \leftarrow G(F(\mathsf{pk}), m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	$k \leftarrow \mathsf{KDF}(\hat{k}, F(c))$	**if** $c' = c$ **then**
	return (c, k)	**return** $\mathsf{KDF}(\hat{k}', F(c))$
		else return $\mathsf{KDF}(s, F(c))$

CRYSTALS-KYBER, Saber

CRYSTALS-KYBER and SABER

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

“$k \leftarrow H(m, c)$”

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$r \leftarrow G(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$r' \leftarrow G(m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathsf{FO}^{\not\perp}$

“$k \leftarrow H(G(m), F(c))$”

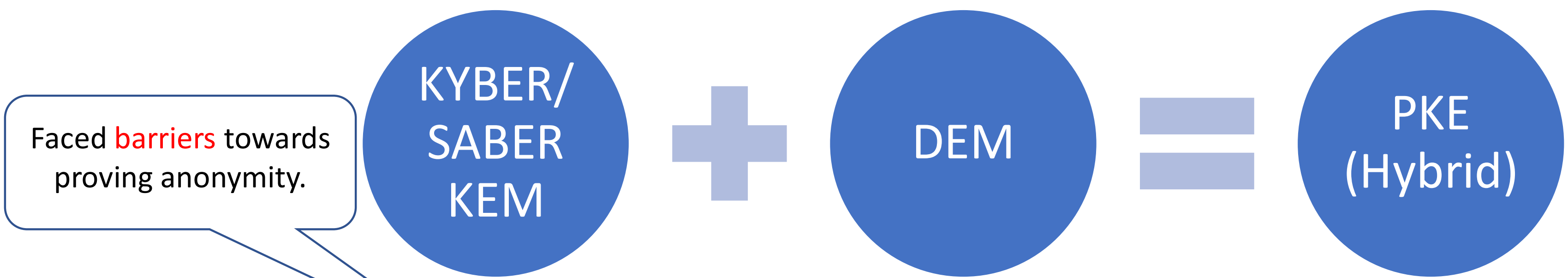
KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$
$s \leftarrow_{\$} \mathcal{M}$	$m \leftarrow F(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' \leftarrow (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$	$(\hat{k}, r) \leftarrow G(F(pk), m)$	$(\hat{k}', r') \leftarrow G(F(\mathsf{pk}), m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	$k \leftarrow \mathsf{KDF}(\hat{k}, F(c))$	**if** $c' = c$ **then**
	return (c, k)	**return** $\mathsf{KDF}(\hat{k}', F(c))$
		else return $\mathsf{KDF}(s, F(c))$

CRYSTALS-KYBER, Saber

CRYSTALS-KYBER and SABER

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

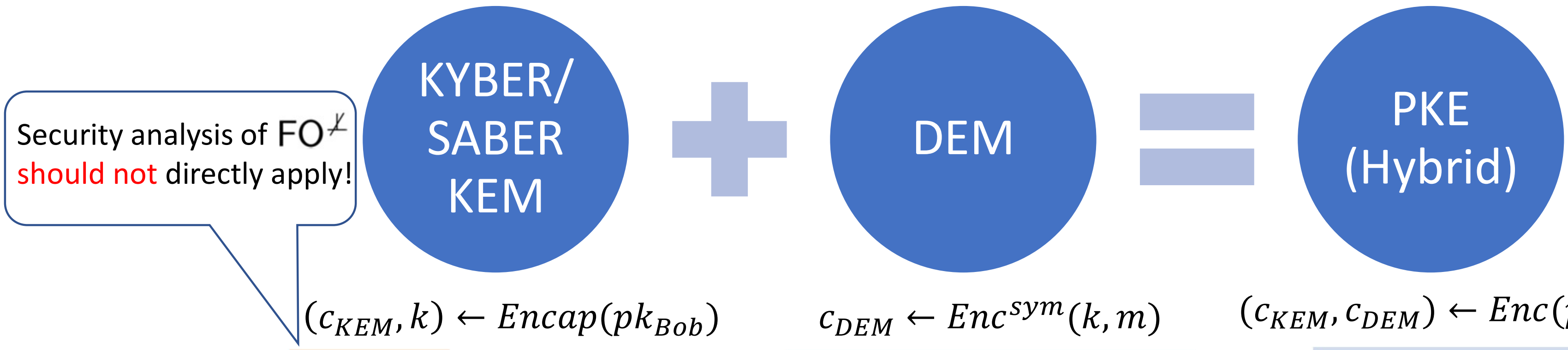
Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

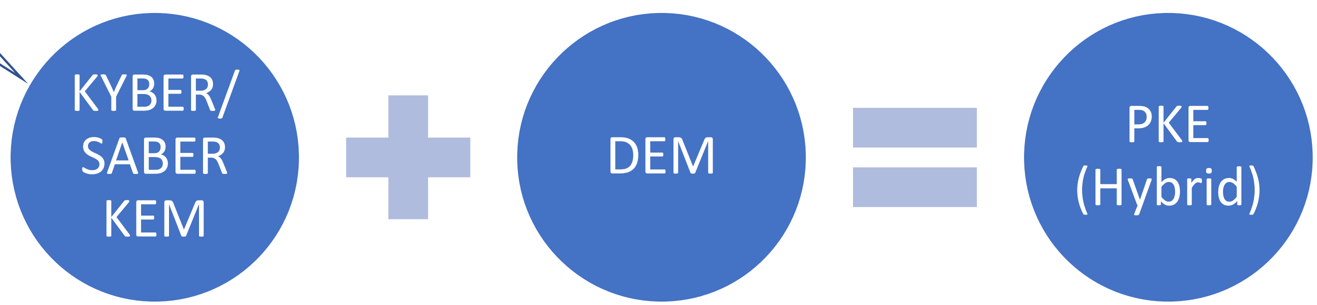
CRYSTALS-KYBER and SABER

<u>Public-Key Encryption/KEMs</u>
- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

<u>Public-Key Encryption/KEMs</u>
- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Is "robust". [Grubbs-Maram-Paterson'22]

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

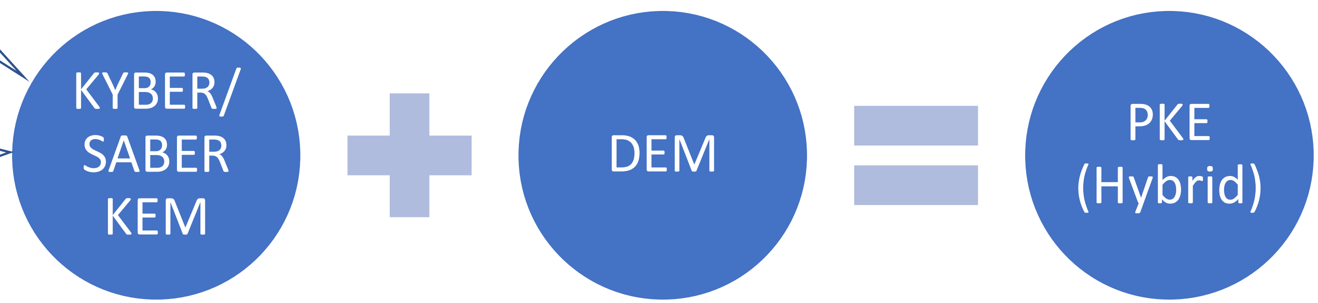
Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

Is "robust". [Grubbs-Maram-Paterson'22]

$Decap(sk_{Bob}, c) \neq Decap(sk_{Dave}, c)$

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

CRYSTALS-KYBER and SABER

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

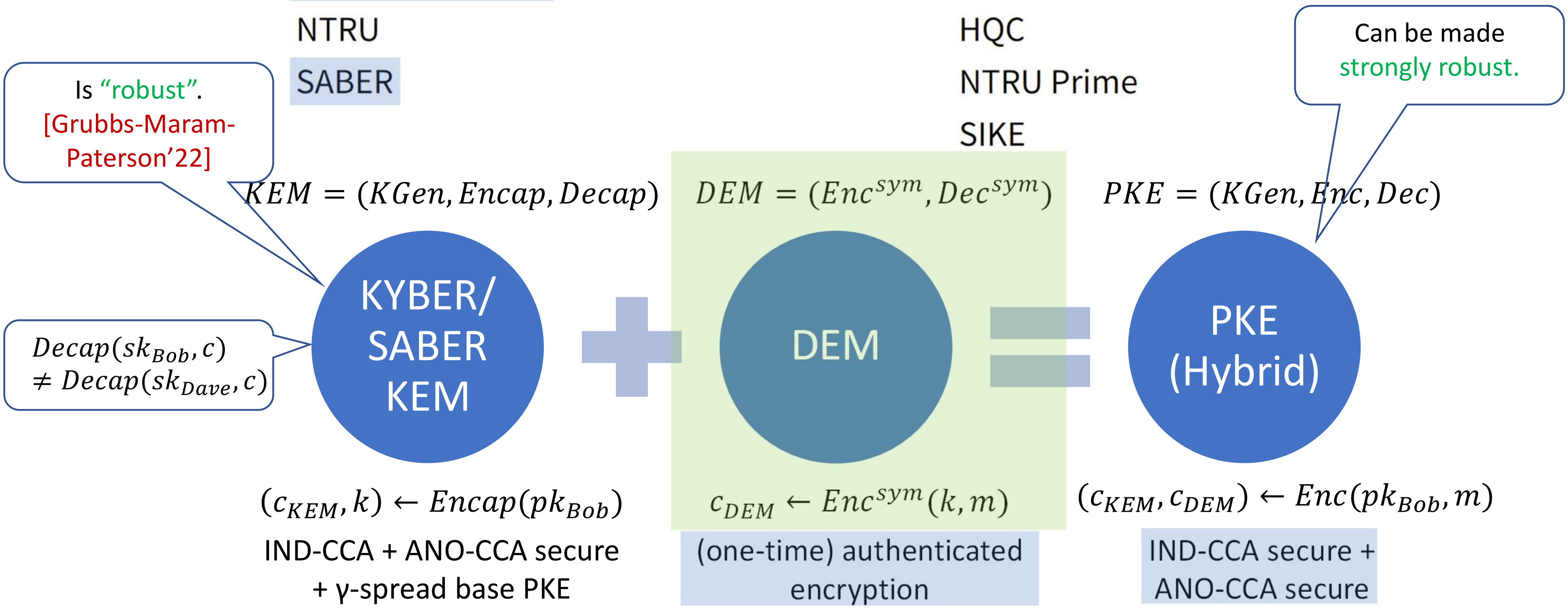
BIKE

FrodoKEM

HQC

NTRU Prime

SIKE



FrodoKEM

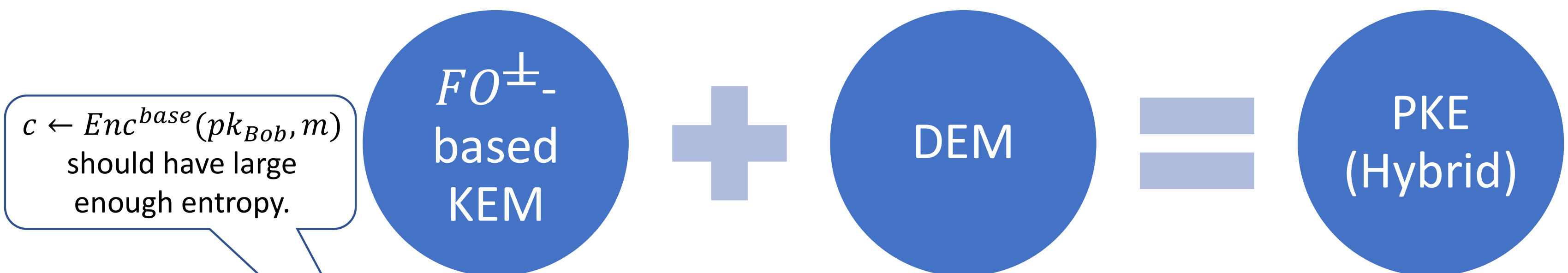
Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$

Frodo KEM + DEM = PKE (Hybrid)

$c \leftarrow Enc^{base}(pk_{Bob}, m)$ should have large enough entropy.

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

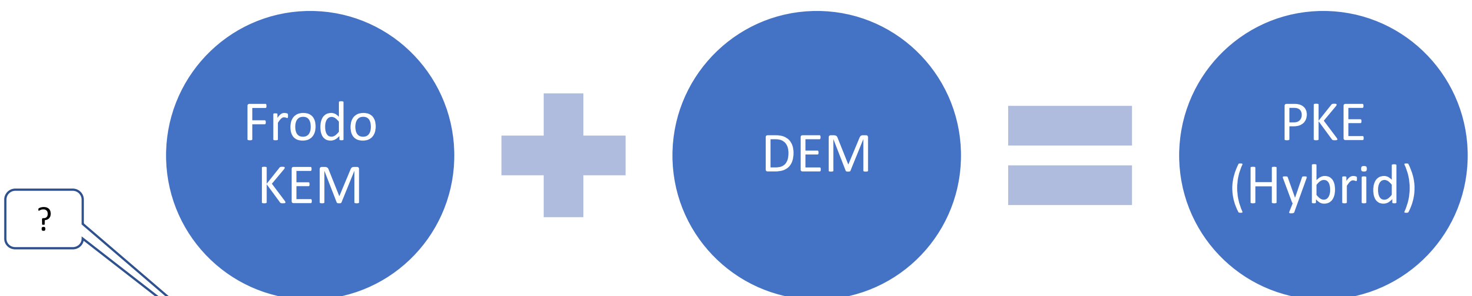
Public-Key Encryption/KEMs

- Classic McEliece
- CRYSTALS-KYBER
- NTRU
- SABER

Public-Key Encryption/KEMs

- BIKE
- FrodoKEM
- HQC
- NTRU Prime
- SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_\$ \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_\$ \mathcal{M}$	$r \leftarrow G(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$r' \leftarrow G(m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathrm{FO}^{\not\perp}$

KGen'	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}', c)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_\$ \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$
$s \leftarrow_\$ \mathcal{M}$	$(\hat{k}, r) \leftarrow G(F(pk), m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' \leftarrow (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$(\hat{k}', r') \leftarrow G(F(\mathsf{pk}), m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(\hat{k}, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(\hat{k}', c)$
		else return $H(s, c)$

FrodoKEM

FrodoKEM

Public-Key Encryption/KEMs

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

Public-Key Encryption/KEMs

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

"$k \leftarrow H(m, c)$"

KGen′	Encap(pk)	Decap(sk′, c)
1 : (pk, sk) ← KGen	1 : $m \leftarrow_\$ \mathcal{M}$	1 : Parse sk′ = (sk, s)
2 : $s \leftarrow_\$ \mathcal{M}$	2 : $r \leftarrow G(m)$	2 : $m' \leftarrow$ Dec(sk, c)
3 : sk′ = (sk, s)	3 : $c \leftarrow$ Enc(pk, m; r)	3 : $r' \leftarrow G(m')$
4 : **return** (pk, sk′)	4 : $k \leftarrow H(m, c)$	4 : $c' \leftarrow$ Enc(pk, m'; r')
	5 : **return** (c, k)	5 : **if** $c' = c$ **then**
		6 : **return** $H(m', c)$
		7 : **else return** $H(s, c)$

$\mathrm{FO}^{\not\perp}$

"$k \leftarrow H(G(m), c)$"

KGen′	Encap(pk)	Decap(sk′, c)
1 : (pk, sk) ← KGen	1 : $m \leftarrow_\$ \mathcal{M}$	1 : Parse sk′ = (sk, pk, F(pk), s)
2 : $s \leftarrow_\$ \mathcal{M}$	2 : $(\hat{k}, r) \leftarrow G(F(pk), m)$	2 : $m' \leftarrow$ Dec(sk, c)
3 : sk′ ← (sk, pk, F(pk), s)	3 : $c \leftarrow$ Enc(pk, m; r)	3 : $(\hat{k}', r') \leftarrow G(F(\mathrm{pk}), m')$
4 : **return** (pk, sk′)	4 : $k \leftarrow H(\hat{k}, c)$	4 : $c' \leftarrow$ Enc(pk, m'; r')
	5 : **return** (c, k)	5 : **if** $c' = c$ **then**
		6 : **return** $H(\hat{k}', c)$
		7 : **else return** $H(s, c)$

FrodoKEM

FrodoKEM

Public-Key Encryption/KEMs

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

Public-Key Encryption/KEMs

BIKE

FrodoKEM

HQC

NTRU Prime

SIKE

"$k \leftarrow H(m, c)$"

KGen′	Encap(pk)	Decap(sk′, c)
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, s)$
$s \leftarrow_{\$} \mathcal{M}$	$r \leftarrow G(m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' = (\mathsf{sk}, s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$r' \leftarrow G(m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(m, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(m', c)$
		else return $H(s, c)$

$\mathsf{FO}^{\not\perp}$

"$k \leftarrow H(G(m), c)$"

Only nested hashing of m and not c.

KGen′	Encap(pk)	Decap(sk′, c)
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$	$m \leftarrow_{\$} \mathcal{M}$	Parse $\mathsf{sk}' = (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$
$s \leftarrow_{\$} \mathcal{M}$	$(\hat{k}, r) \leftarrow G(F(\mathsf{pk}), m)$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c)$
$\mathsf{sk}' \leftarrow (\mathsf{sk}, \mathsf{pk}, F(\mathsf{pk}), s)$	$c \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$	$(\hat{k}', r') \leftarrow G(F(\mathsf{pk}), m')$
return $(\mathsf{pk}, \mathsf{sk}')$	$k \leftarrow H(\hat{k}, c)$	$c' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; r')$
	return (c, k)	**if** $c' = c$ **then**
		return $H(\hat{k}', c)$
		else return $H(s, c)$

FrodoKEM

FrodoKEM

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

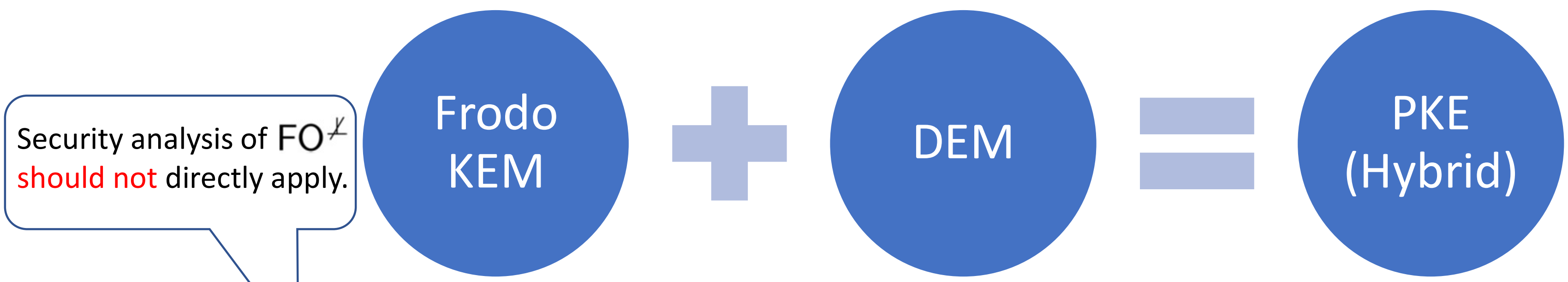
FrodoKEM

HQC

NTRU Prime

SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

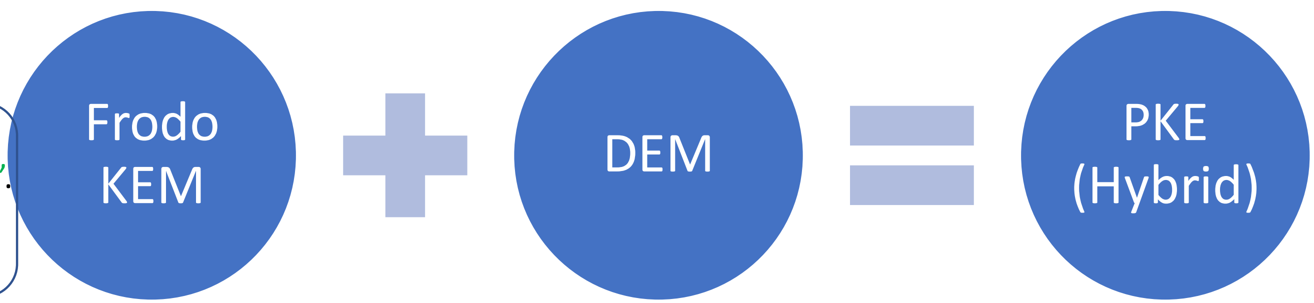
$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



But (provable) IND-CCA security can be "recovered". [Grubbs-Maram-Paterson'21]

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

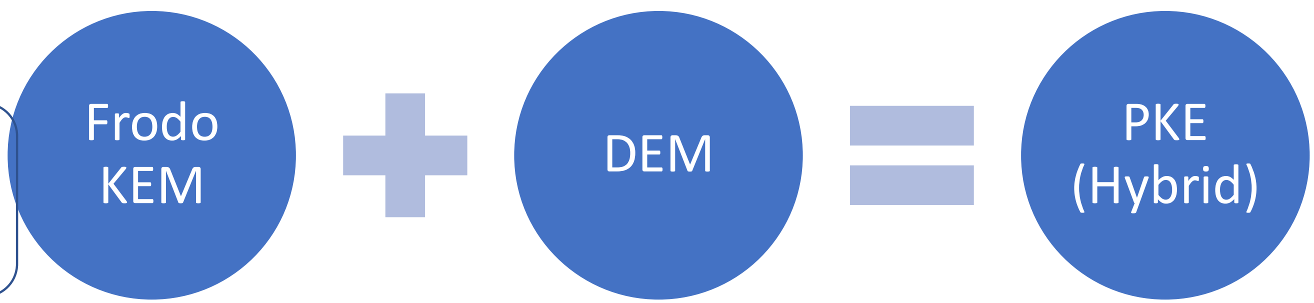
$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



FrodoKEM is ANO-CCA secure in the QROM. [Grubbs-Maram -Paterson'21]

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

FrodoKEM

<u>Public-Key Encryption/KEMs</u>

Classic McEliece

CRYSTALS-KYBER

NTRU

SABER

<u>Public-Key Encryption/KEMs</u>

BIKE

FrodoKEM

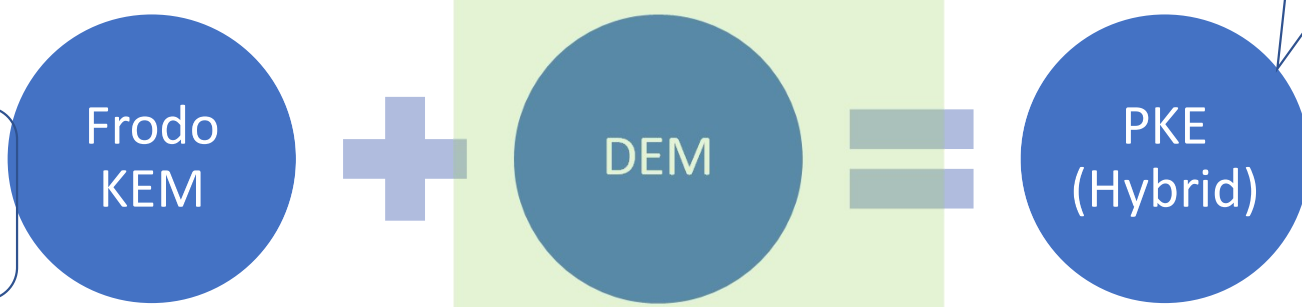
HQC

NTRU Prime

SIKE

FrodoKEM does result in anonymous and robust PKE in a PQ setting.

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



FrodoKEM is ANO-CCA secure in the QROM. [Grubbs-Maram -Paterson'21]

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA + ANO-CCA secure + γ-spread base PKE

$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure + ANO-CCA secure

Other Contributions

Other Contributions

Encap(pk)	Decap(sk, c)
$m \leftarrow_{\$} \mathcal{M}$	Parse $c = (c_1, c_2)$
$c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c_1)$
$c_2 \leftarrow H'(m)$	$c_1' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
	if $c_1' = c_1 \land H'(m') = c_2$ **then**
$c \leftarrow (c_1, c_2)$	
$k = H(m, c)$	**return** $H(m', c)$
return (c, k)	**else return** $\perp$

$\mathsf{HFO}^{\perp}$

Other Contributions

Encap(pk)	Decap(sk, c)
$m \leftarrow_{\$} \mathcal{M}$	Parse $c = (c_1, c_2)$
$c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c_1)$
$c_2 \leftarrow H'(m)$	$c_1' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
	if $c_1' = c_1 \wedge H'(m') = c_2$ **then**
$c \leftarrow (c_1, c_2)$	
$k = H(m, c)$	**return** $H(m', c)$
return (c, k)	**else return** $\perp$

$\mathsf{HFO}^{\perp}$

Results in IND-CCA secure KEMs in the QROM. [Jiang-Zhang-Ma'19]

Other Contributions

Encap(pk)	Decap(sk, c)
$m \leftarrow_{\$} \mathcal{M}$	Parse $c = (c_1, c_2)$
$c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c_1)$
$c_2 \leftarrow H'(m)$	$c_1' \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
$\boxed{c_2 \leftarrow H'(m, c_1)}$	**if** $c_1' = c_1 \wedge H'(m') = c_2$ **then**
$c \leftarrow (c_1, c_2)$	$\boxed{\textbf{if } c_1' = c_1 \wedge H'(m', c_1) = c_2 \textbf{ then}}$
$k = H(m, c)$	**return** $H(m', c)$
return (c, k)	**else return** $\perp$

$\mathsf{HFO}^{\perp}$ $\boxed{\mathsf{HFO}^{\perp'}}$

Results in IND-CCA secure KEMs in the QROM. [Jiang-Zhang-Ma'19]

Other Contributions

$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathsf{sk}, c)$
$m \leftarrow_{\$} \mathcal{M}$	Parse $c = (c_1, c_2)$
$c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m; G(m))$	$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, c_1)$
$c_2 \leftarrow H'(m)$	$c'_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m'; G(m'))$
$\boxed{c_2 \leftarrow H'(m, c_1)}$	**if** $c'_1 = c_1 \wedge H'(m') = c_2$ **then**
$c \leftarrow (c_1, c_2)$	$\boxed{\textbf{if } c'_1 = c_1 \wedge H'(m', c_1) = c_2 \textbf{ then}}$
$k = H(m, c)$	**return** $H(m', c)$
return (c, k)	**else return** $\perp$

$\mathsf{HFO}^{\perp}$ $\boxed{\mathsf{HFO}^{\perp'}}$

Results in IND-CCA secure KEMs in the QROM. [Jiang-Zhang-Ma'19]

Results in IND-CCA, ANO-CCA and SROB secure KEMs in the QROM. [Grubbs-Maram-Paterson'22]

Conclusions

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).
- Hybrid PKE schemes derived from Classic McEliece cannot be (strongly) robust.

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).
- Hybrid PKE schemes derived from Classic McEliece cannot be (strongly) robust.
  - Though they can be made ANO-CCA secure as shown in [Xagawa'22].

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).
- Hybrid PKE schemes derived from Classic McEliece cannot be (strongly) robust.
  - Though they can be made ANO-CCA secure as shown in [Xagawa'22].
- We identified barriers towards proving IND-CCA and ANO-CCA security of CRYSTALS-KYBER and SABER in the QROM.

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).
- Hybrid PKE schemes derived from Classic McEliece cannot be (strongly) robust.
  - Though they can be made ANO-CCA secure as shown in [Xagawa'22].
- We identified barriers towards proving IND-CCA and ANO-CCA security of CRYSTALS-KYBER and SABER in the QROM.
  - At the same time, we showed they do result in strongly robust hybrid PKE schemes.

Conclusions

- We provide insights into obtaining anonymous and robust hybrid PKE schemes – via the KEM-DEM composition – when the KEM is implicitly rejecting (i.e., non-robust).
- We showed that the $\mathsf{FO}^{\not\perp}$ transform does result in ANO-CCA secure and "robust" KEMs in a post-quantum setting (i.e., the QROM).
- Hybrid PKE schemes derived from Classic McEliece cannot be (strongly) robust.
  - Though they can be made ANO-CCA secure as shown in [Xagawa'22].
- We identified barriers towards proving IND-CCA and ANO-CCA security of CRYSTALS-KYBER and SABER in the QROM.
  - At the same time, we showed they do result in strongly robust hybrid PKE schemes.
- Finally, we showed that FrodoKEM does result in ANO-CCA secure and strongly robust hybrid PKE schemes in the QROM.

Extra Slides

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output $\text{ENCODE}(e, T)$ is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output $\text{ENCODE}(e, T)$ is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output $\textsc{Encode}(e, T)$ is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Fix any "message" $e = \binom{e_{n-k}}{0^k}$:

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Fix any "message" $e = \binom{e_{n-k}}{0^k}$:

- ($n - k \geq t$ in all CM parameters)

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Fix any "message" $e = \binom{e_{n-k}}{0^k}$:

- ($n - k \geq t$ in all CM parameters)
- $C_0 = (I_{n-k}|T)\binom{e_{n-k}}{0^k} = e_{n-k}$ – i.e., independent of public-key T.

Classic McEliece (CM)

2.2.3 Encoding subroutine

The following algorithm ENCODE takes two inputs: a weight-t column vector $e \in \mathbb{F}_2^n$; and a public key T, i.e., an $(n-k) \times k$ matrix over $\mathbb{F}_2$. The algorithm output ENCODE(e, T) is a vector $C_0 \in \mathbb{F}_2^{n-k}$. Here is the algorithm:

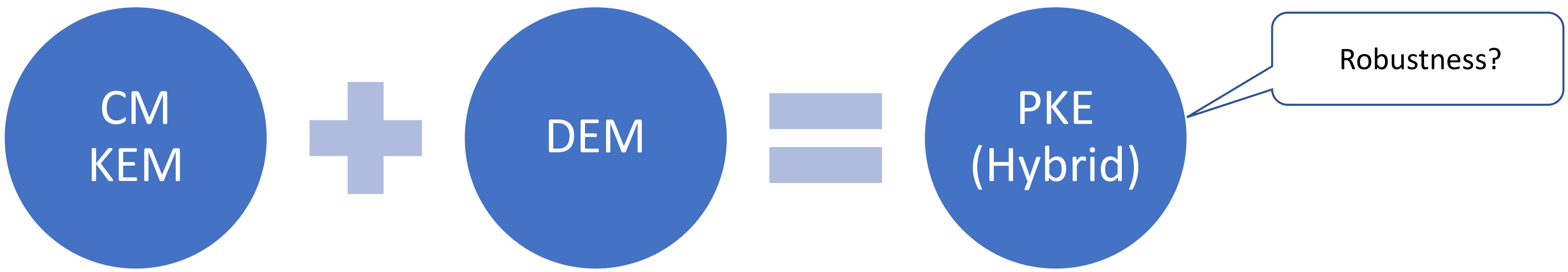
1. Define $H = (I_{n-k} \mid T)$.
2. Compute and return $C_0 = He \in \mathbb{F}_2^{n-k}$.

Fix any "message" $e = \binom{e_{n-k}}{0^k}$:

- ($n - k \geq t$ in all CM parameters)
- $C_0 = (I_{n-k}|T)\binom{e_{n-k}}{0^k} = e_{n-k}$ – i.e., independent of public-key T.
- Because of perfect correctness, C_0 must decrypt to fixed e under *any private key* of CM's base PKE scheme.

Classic McEliece (CM)

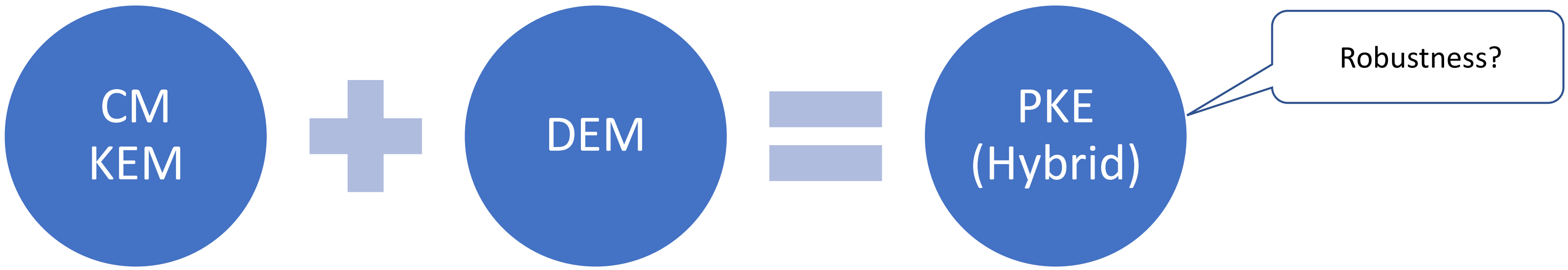
$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

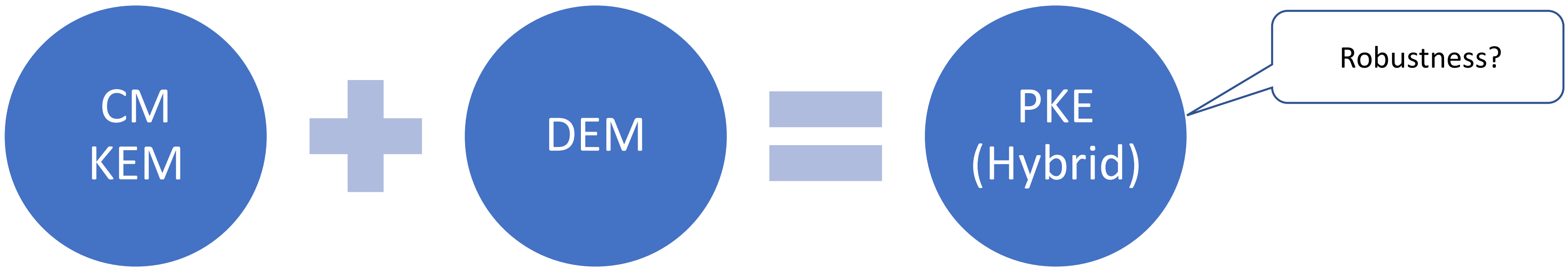
2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

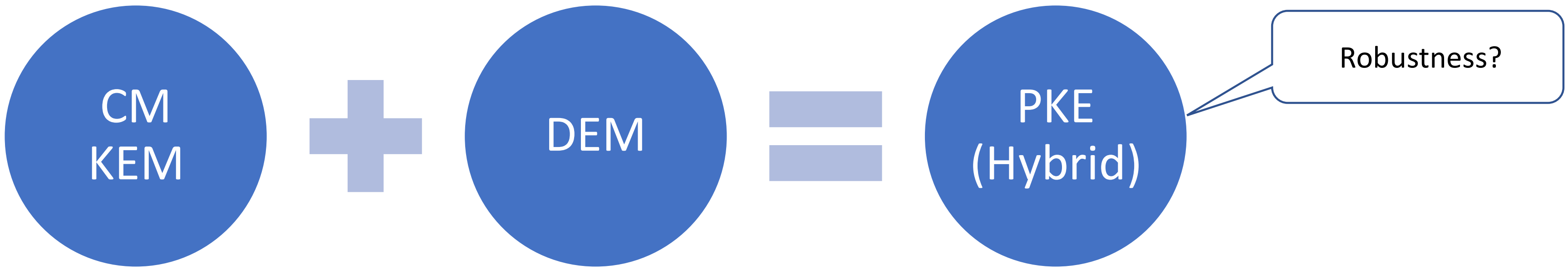
The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

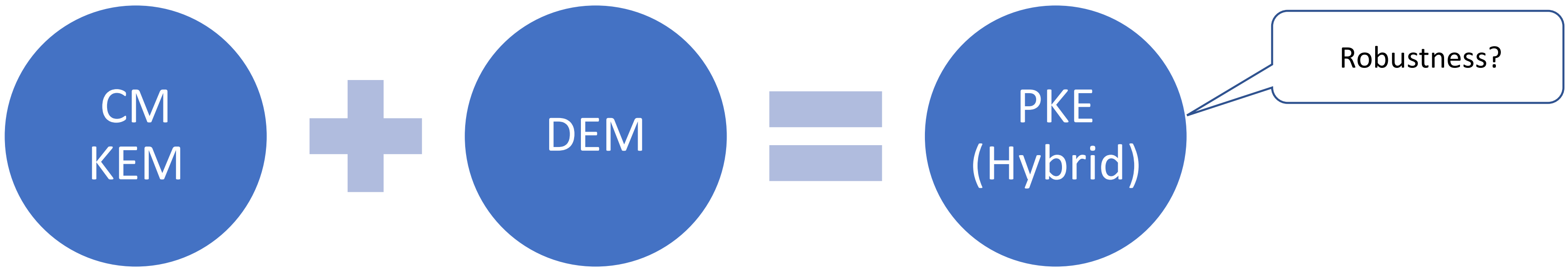
1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

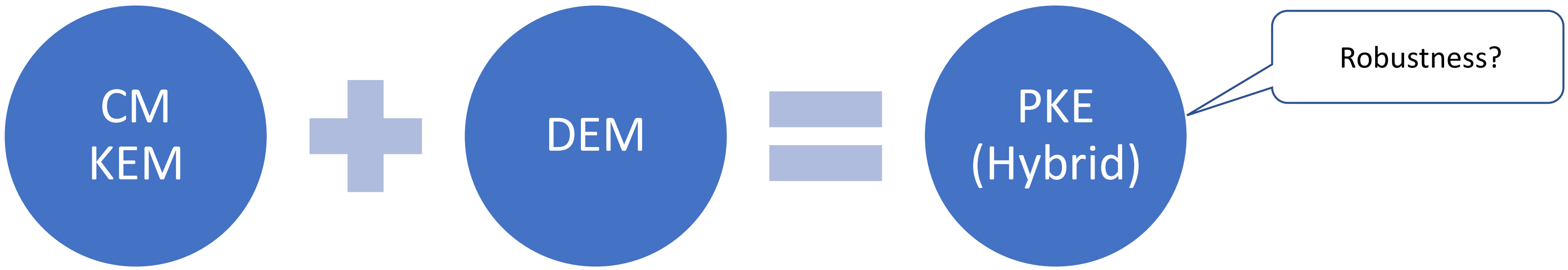
1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

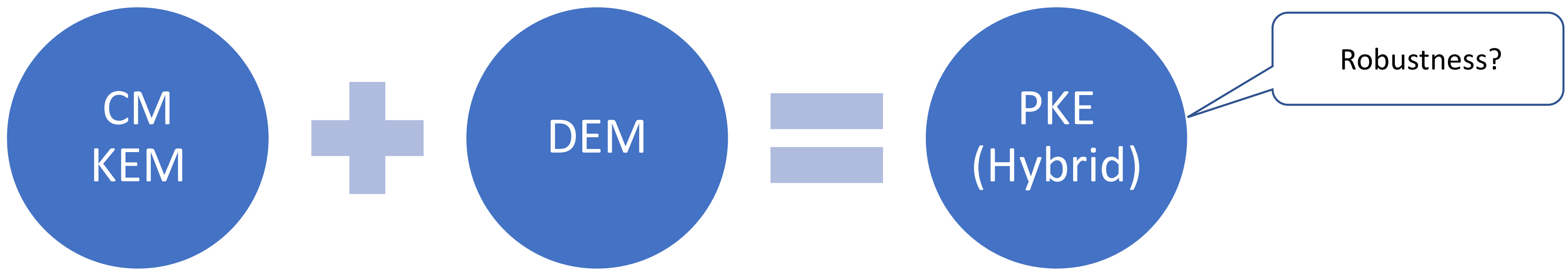
1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DEM} \leftarrow Enc^{sym}(k, m)$.

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

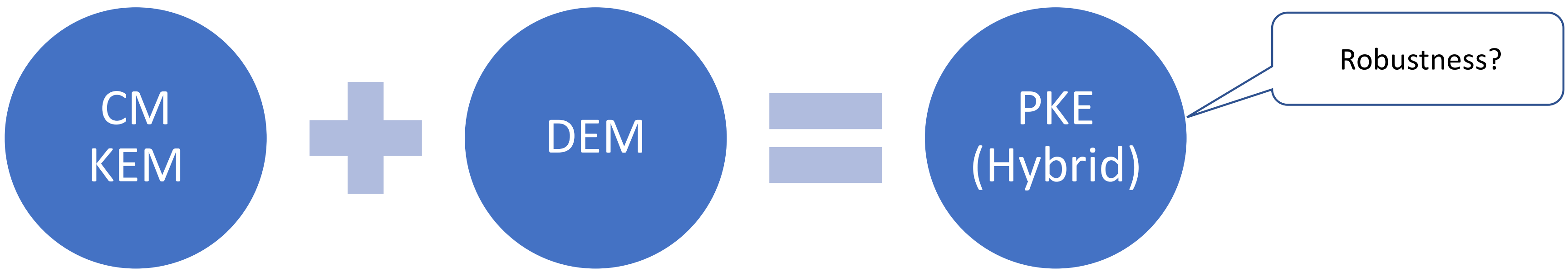
1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DEM} \leftarrow Enc^{sym}(k, m)$.
- Return $c \leftarrow (c_{KEM}, c_{DEM})$.

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

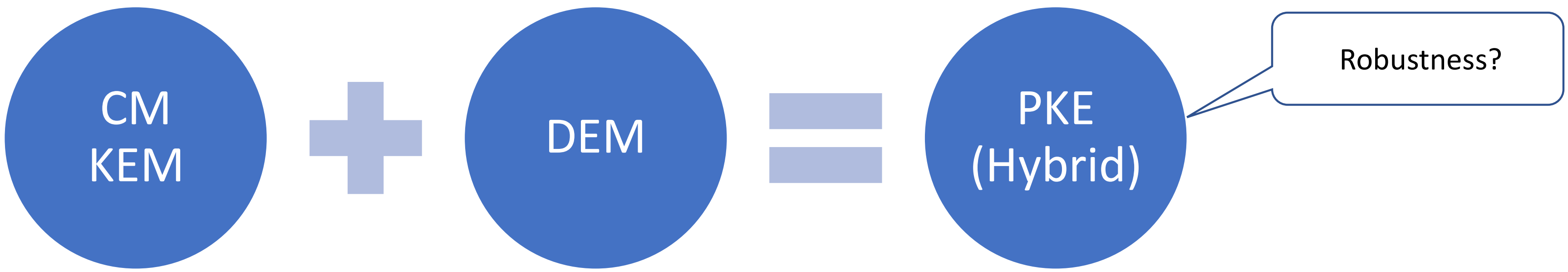
For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DEM} \leftarrow Enc^{sym}(k, m)$.
- Return $c \leftarrow (c_{KEM}, c_{DEM})$.

For *any* CM private key sk_* ,

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

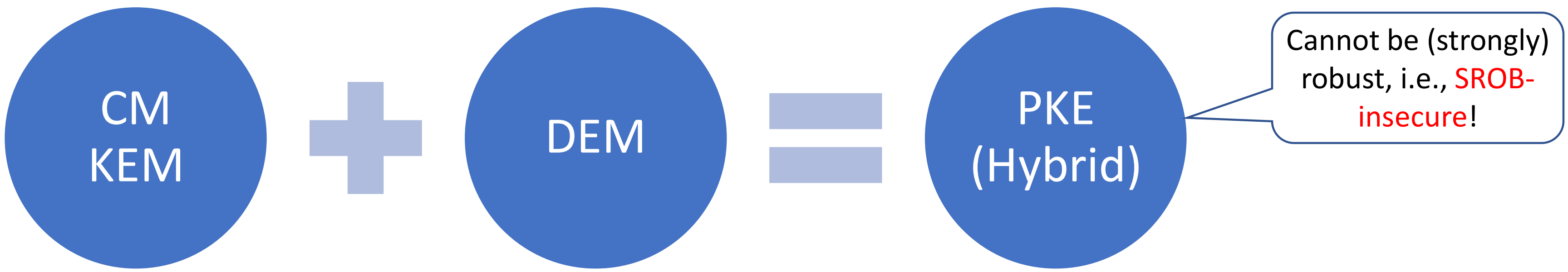
- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DE} \leftarrow Enc^{sym}(k, m)$.
- Return $c \leftarrow (c_{KEM}, c_{DEM})$.

For *any* CM private key sk_*,

$$Dec(sk_*, c) = m \ (\neq \perp).$$

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

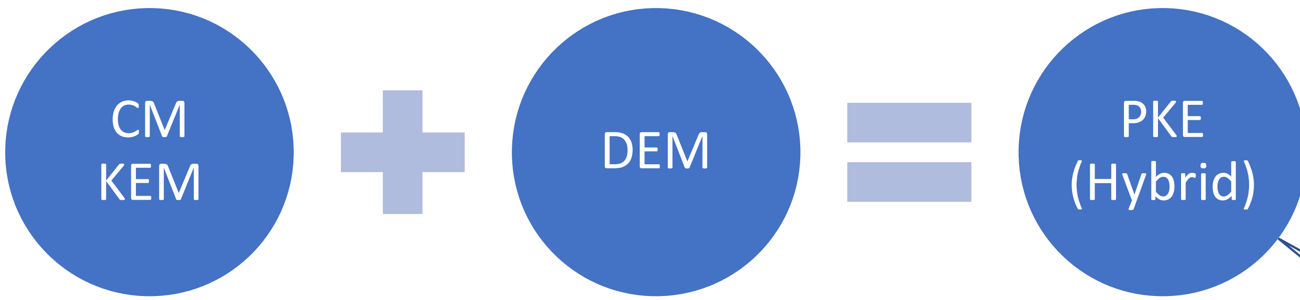
- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DE} \leftarrow Enc^{sym}(k, m)$.
- Return $c \leftarrow (c_{KEM}, c_{DEM})$.

For *any* CM private key sk_* ,

$$Dec(sk_*, c) = m \ (\neq \bot).$$

Classic McEliece (CM)

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



Cannot be (strongly) robust, i.e., SROB-insecure!

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$ $c_{DEM} \leftarrow Enc^{sym}(k, m)$ $(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

But can be ANO-CCA secure. [Xagawa'22]

2.4.5 Encapsulation

The following randomized algorithm ENCAP takes as input a public key T. It outputs a ciphertext C and a session key K. Here is the algorithm:

1. Use FIXEDWEIGHT to generate a vector $e \in \mathbb{F}_2^n$ of weight t.
2. Compute $C_0 = \text{ENCODE}(e, T)$.
3. Compute $C_1 = \mathsf{H}(2, e)$; see Section 2.5.2 for H input encodings. Put $C = (C_0, C_1)$.
4. Compute $K = \mathsf{H}(1, e, C)$; see Section 2.5.2 for H input encodings.
5. Output ciphertext C and session key K.

For *any* message m:

- Fix vector $e = \binom{e_{n-k}}{0^k}$.
- Set $C_0 = e_{n-k}, C_1 = \mathsf{H}(2, e)$ and $c_{KEM} \leftarrow (C_0, C_1)$.
- Compute $k = \mathsf{H}(1, e, c_{KEM})$ and $c_{DEM} \leftarrow Enc^{sym}(k, m)$.
- Return $c \leftarrow (c_{KEM}, c_{DEM})$.

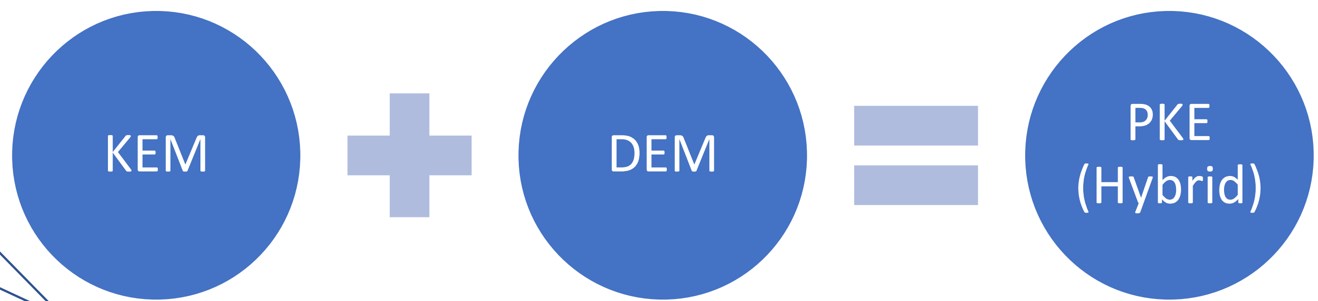
For *any* CM private key sk_*,

$$Dec(sk_*, c) = m \ (\neq \perp).$$

KEM-DEM Paradigm

Public-Key Encryption/KEMs	Public-Key Encryption/KEMs
Classic McEliece	BIKE
CRYSTALS-KYBER	FrodoKEM
NTRU	HQC
SABER	NTRU Prime
	SIKE

$KEM = (KGen, Encap, Decap)$ $DEM = (Enc^{sym}, Dec^{sym})$ $PKE = (KGen, Enc, Dec)$



Indistinguishable from a uniformly random DEM key k'

$(c_{KEM}, k) \leftarrow Encap(pk_{Bob})$

IND-CCA secure

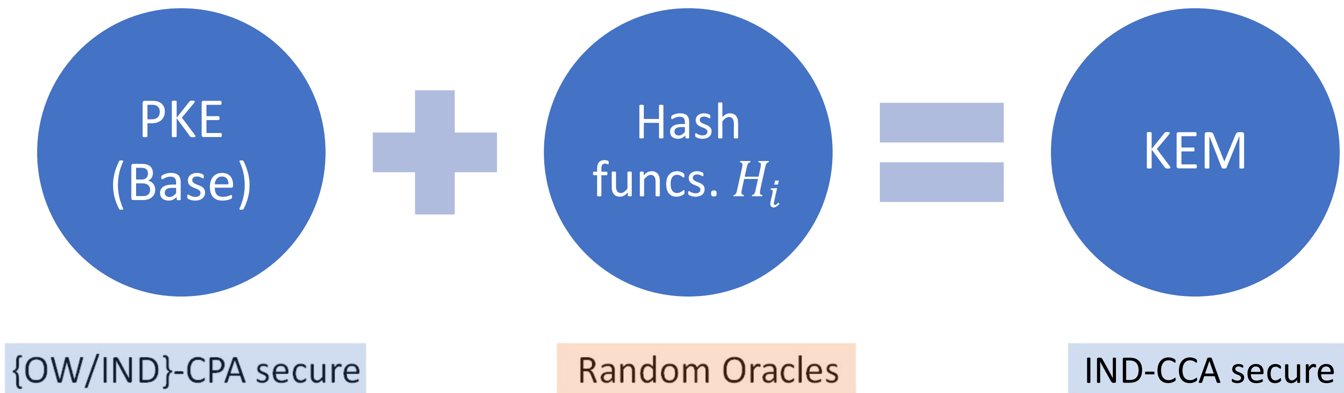
$c_{DEM} \leftarrow Enc^{sym}(k, m)$

(one-time) authenticated encryption

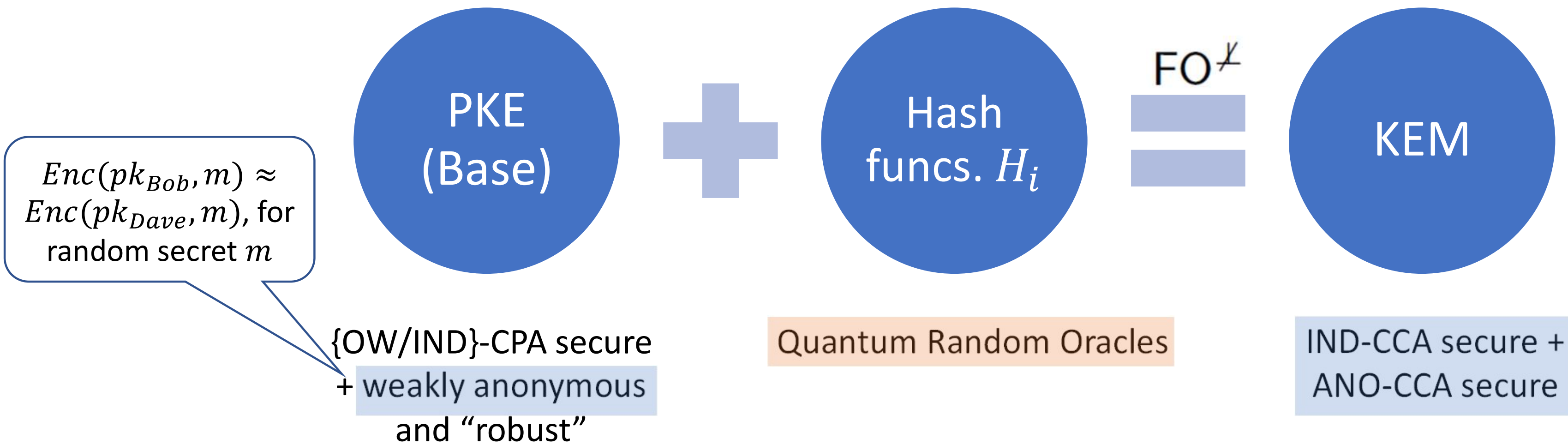
$(c_{KEM}, c_{DEM}) \leftarrow Enc(pk_{Bob}, m)$

IND-CCA secure

Fujisaki-Okamoto Transformation

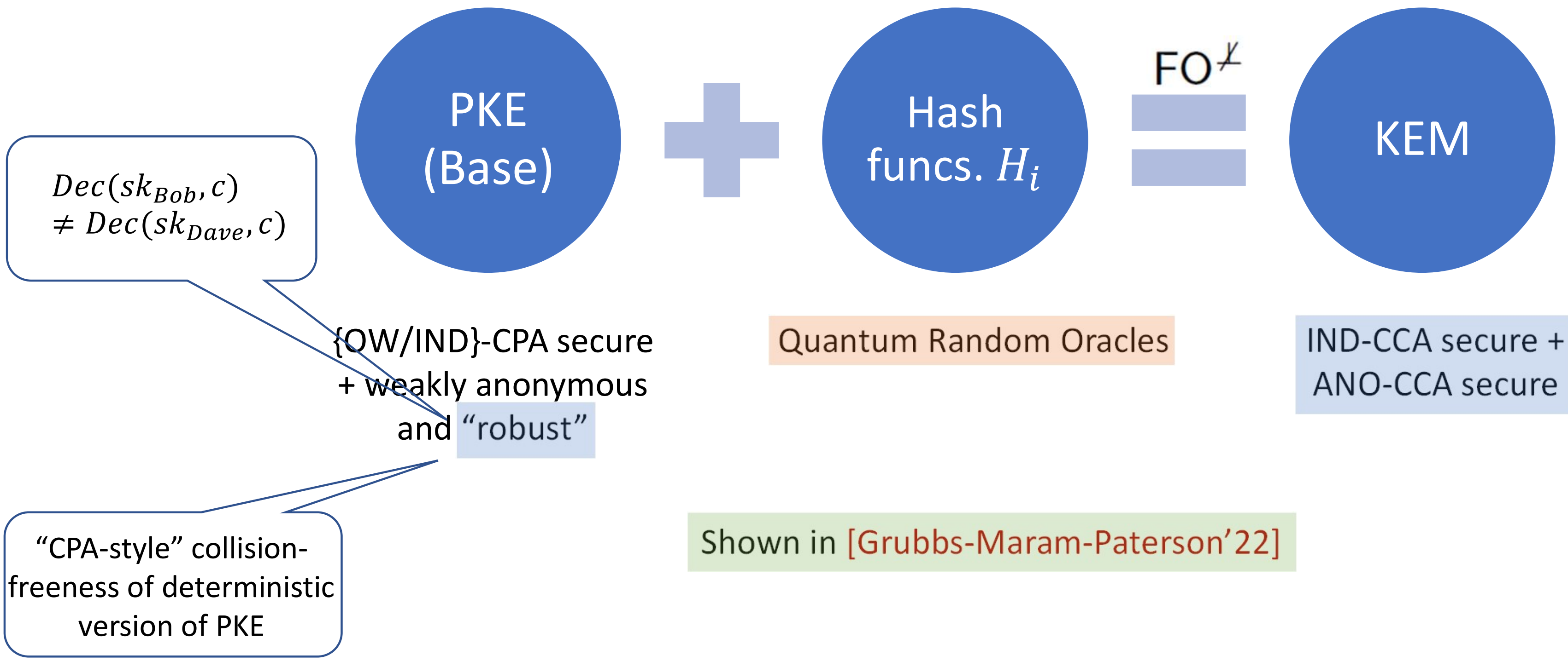


Anonymity from FO transforms



Shown in [Grubbs-Maram-Paterson’22]

Anonymity from FO transforms



Shown in [Grubbs-Maram-Paterson’22]