On the Security of NTS-KEM in the QROM

Varun Maram

9th May 2020

Applied Cryptography Group
Department of Computer Science, ETH Zurich

Introduction

NTS-KEM

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

Introduction

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

Introduction

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

- (Tightly) IND-CCA secure in the ROM
- IND-CCA security in the QROM?

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

Introduction

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

- (Tightly) IND-CCA secure in the ROM **(bug!)**
- IND-CCA security in the QROM?

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

Introduction

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

- (Tightly) IND-CCA secure in the ROM
- IND-CCA secure in the QROM (quadratic loss)

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

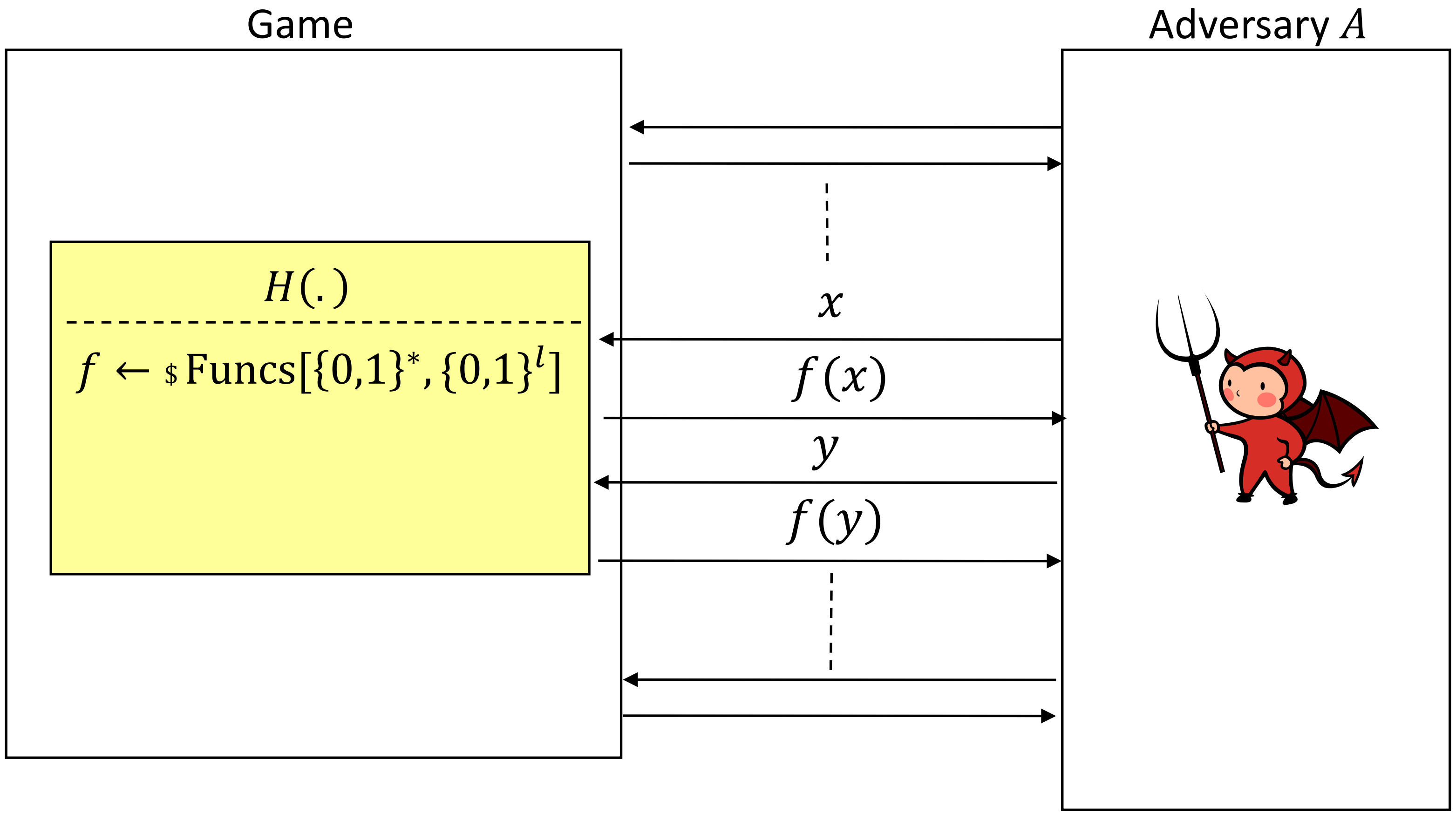
KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

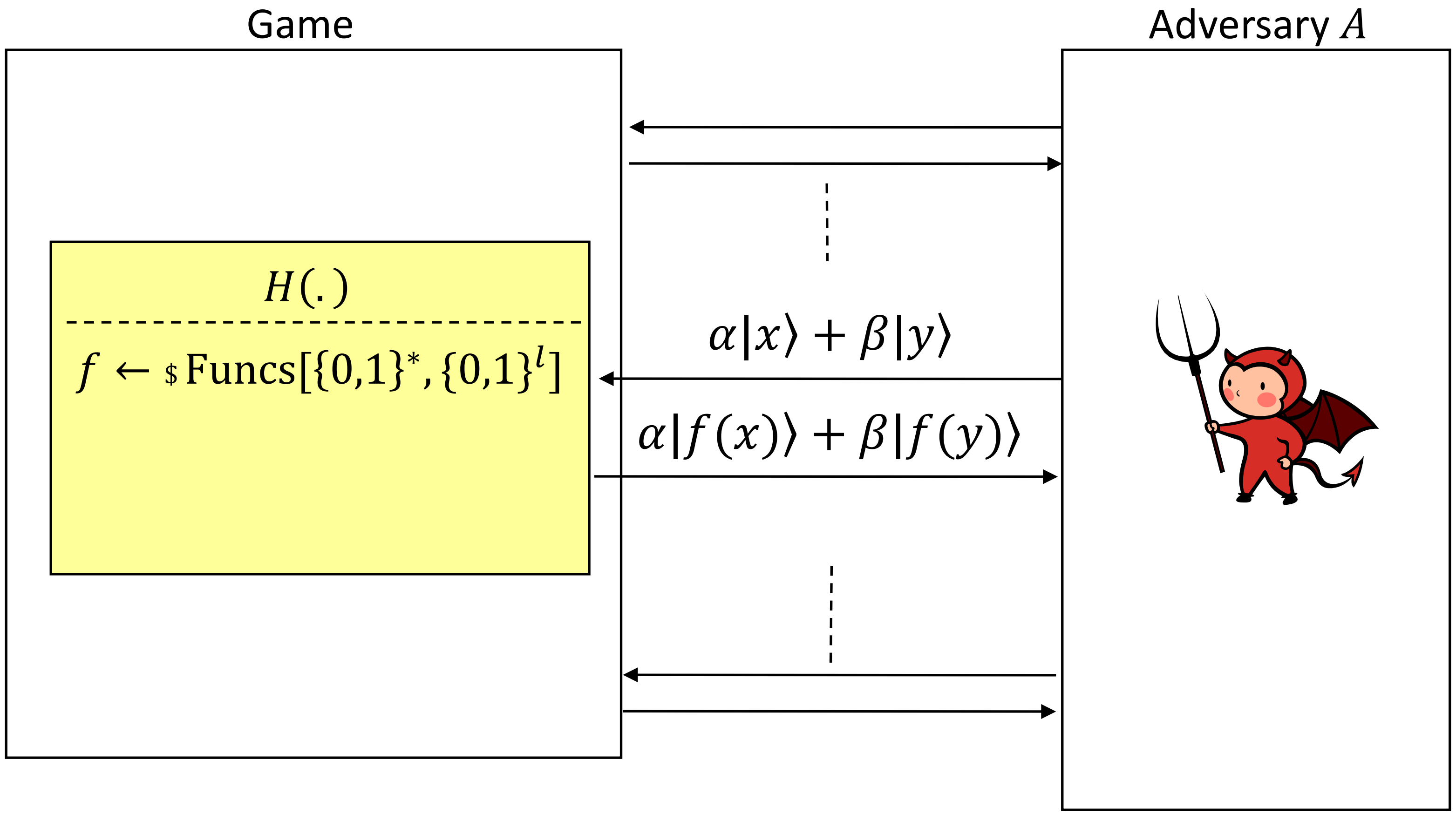
Quantum Random Oracle Model

- **(Classical) ROM:** Modelling a hash function $H(.) : \{0,1\}^* \rightarrow \{0,1\}^l$ as a random oracle:



Quantum Random Oracle Model

- **QROM:** Modelling a hash function $H(.): \{0,1\}^* \rightarrow \{0,1\}^l$ as a *quantum* random oracle:



IND-CCA-secure KEMs in the QROM

OW-secure PKE

IND-CCA-secure KEMs in the QROM

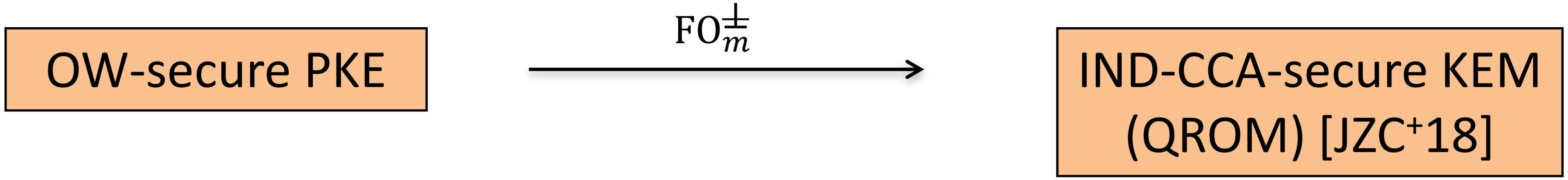
OW-secure PKE $\xrightarrow{\mathsf{FO}_m^{\perp}}$ IND-CCA-secure KEM (ROM) [HHK17]

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_\$ \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_\$ \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM $= \mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

[HHK17]: Hofheinz, D., Hövelmanns, K., and Kiltz, E., *A Modular Analysis of the Fujisaki-Okamoto Transformation*, TCC 2017.

IND-CCA-secure KEMs in the QROM

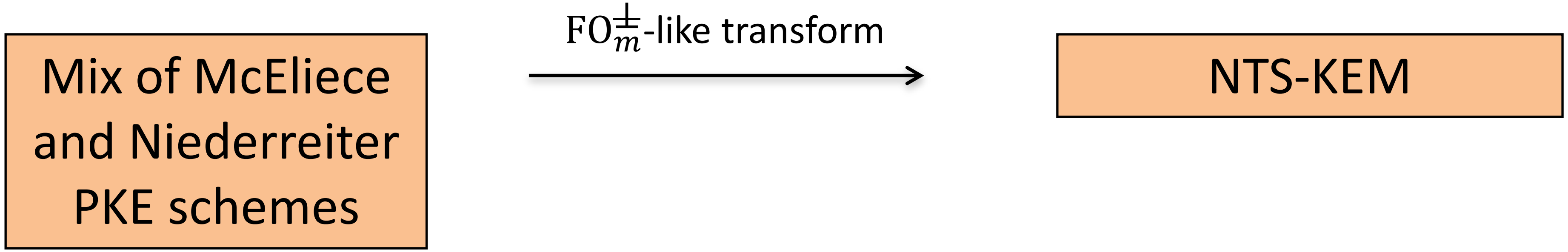


$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM $= \mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

[JZC+18]: Jiang, H., et. al., *IND-CCA-Secure KEM in the Quantum Random Oracle Model, Revisited*, CRYPTO 2018.

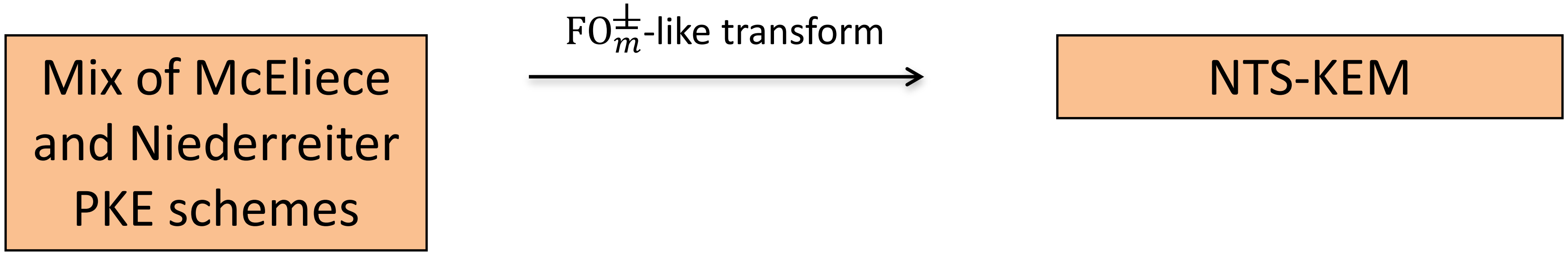
NTS-KEM Specification



$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

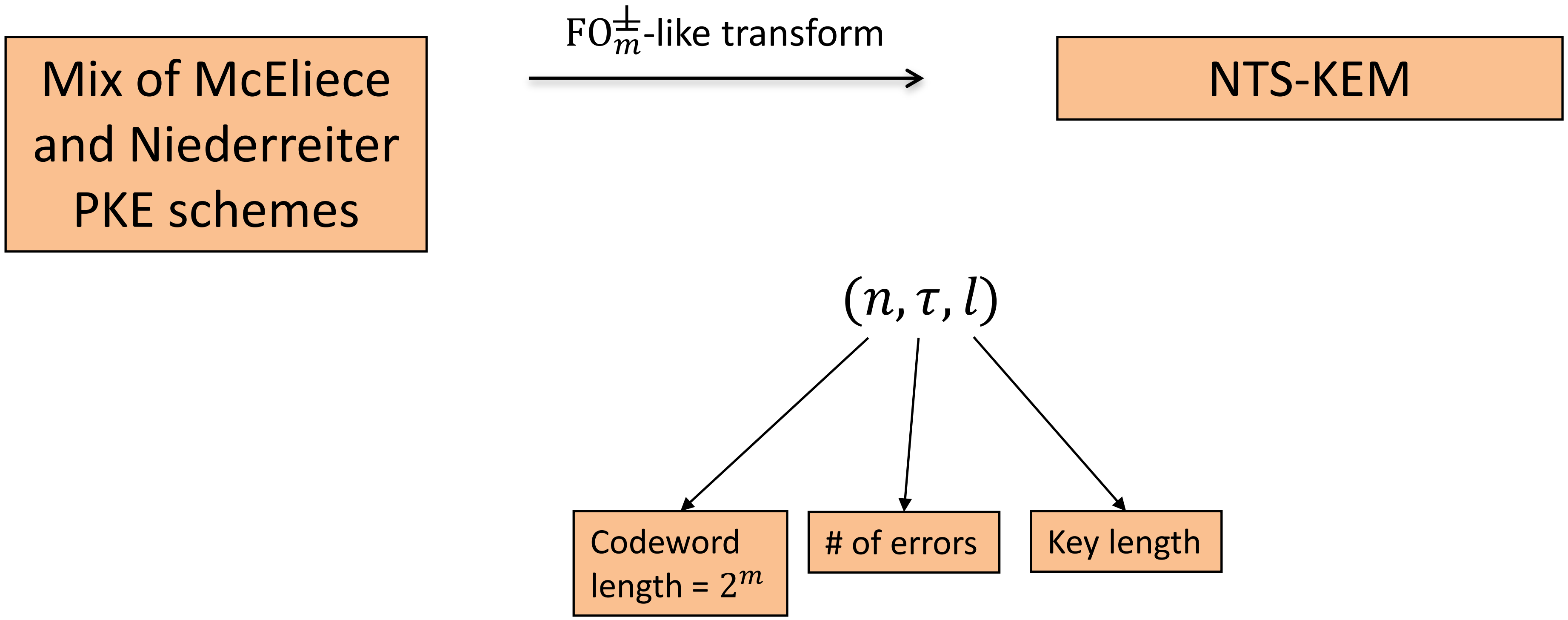
Fig. IND-CCA secure KEM = $\mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

NTS-KEM Specification

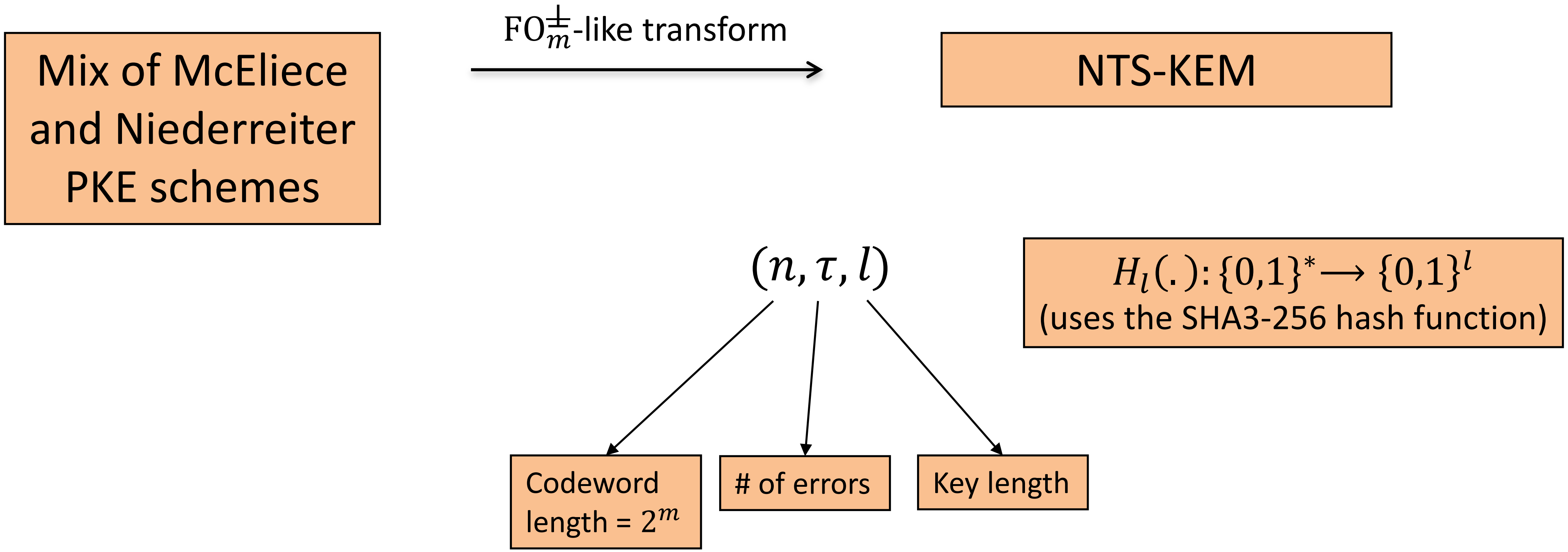


(n, τ, l)

NTS-KEM Specification

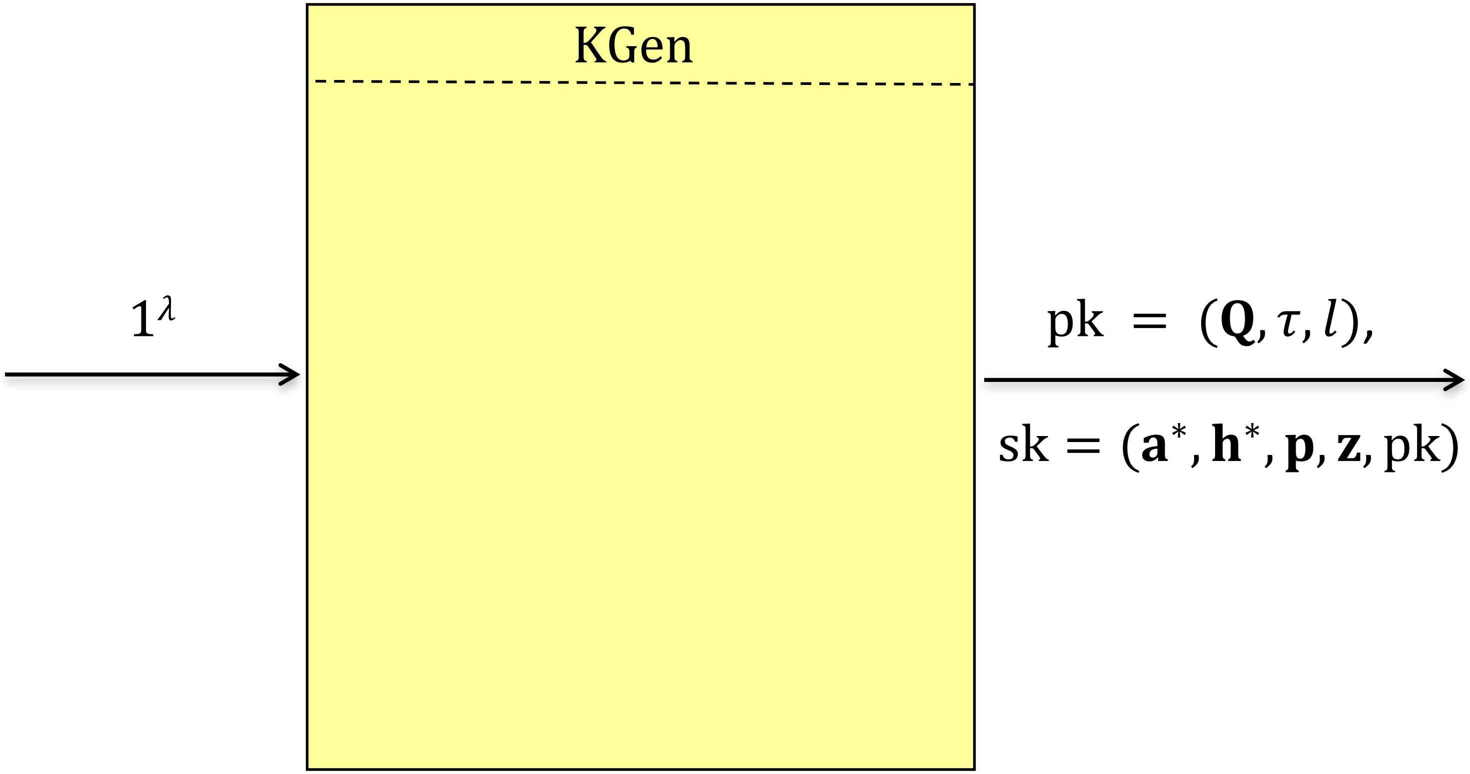


NTS-KEM Specification



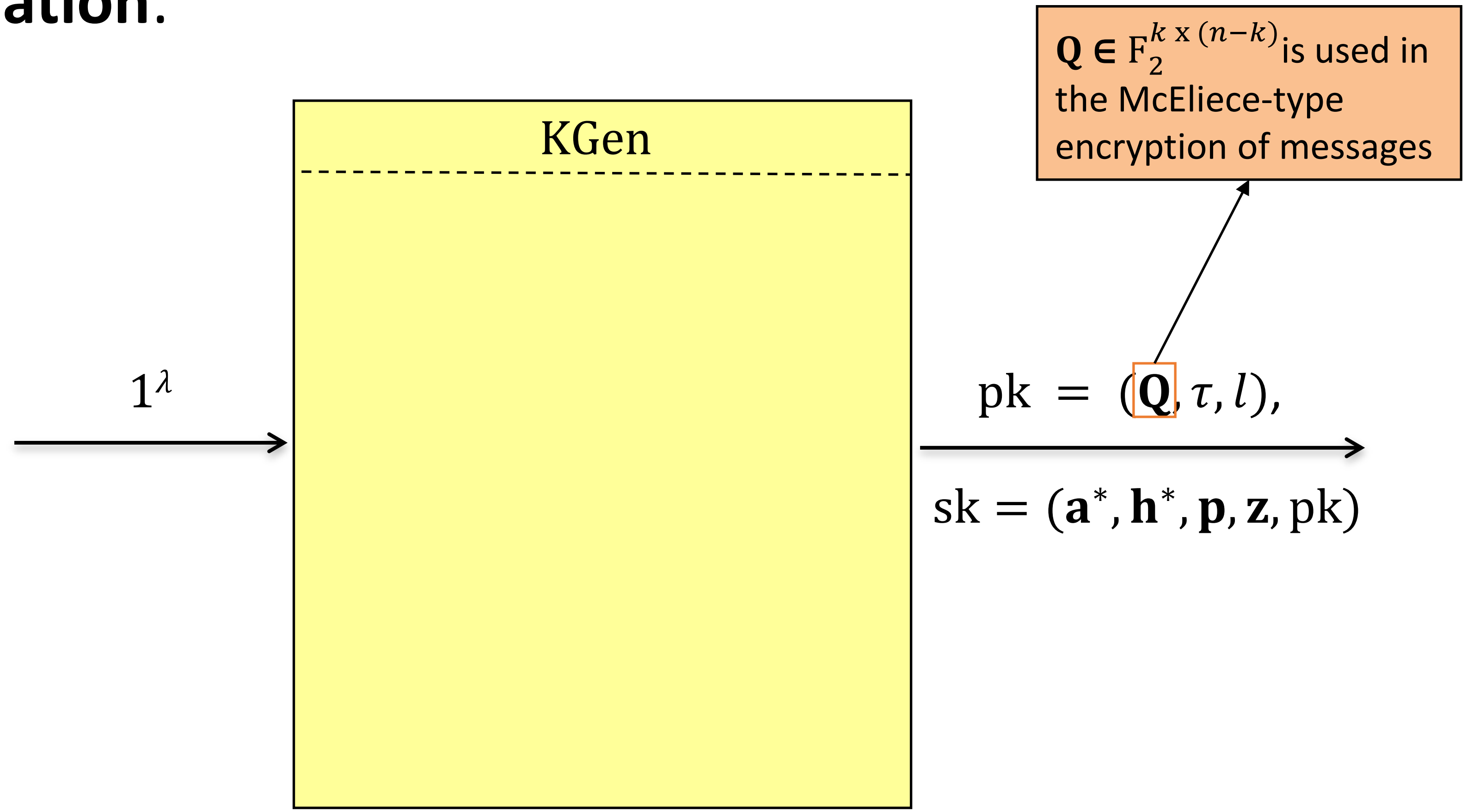
NTS-KEM Specification

- **Key generation**:



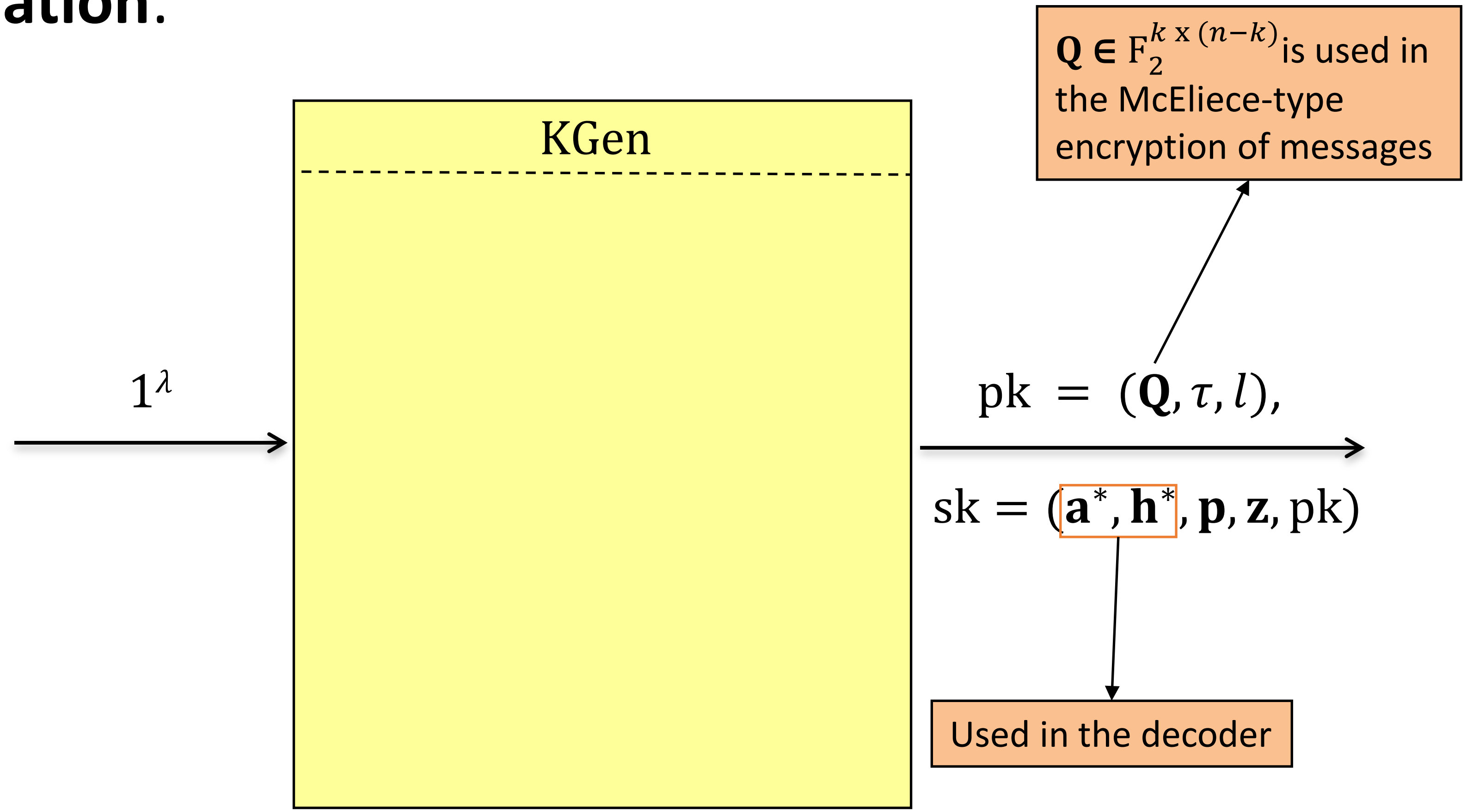
NTS-KEM Specification

- **Key generation**:



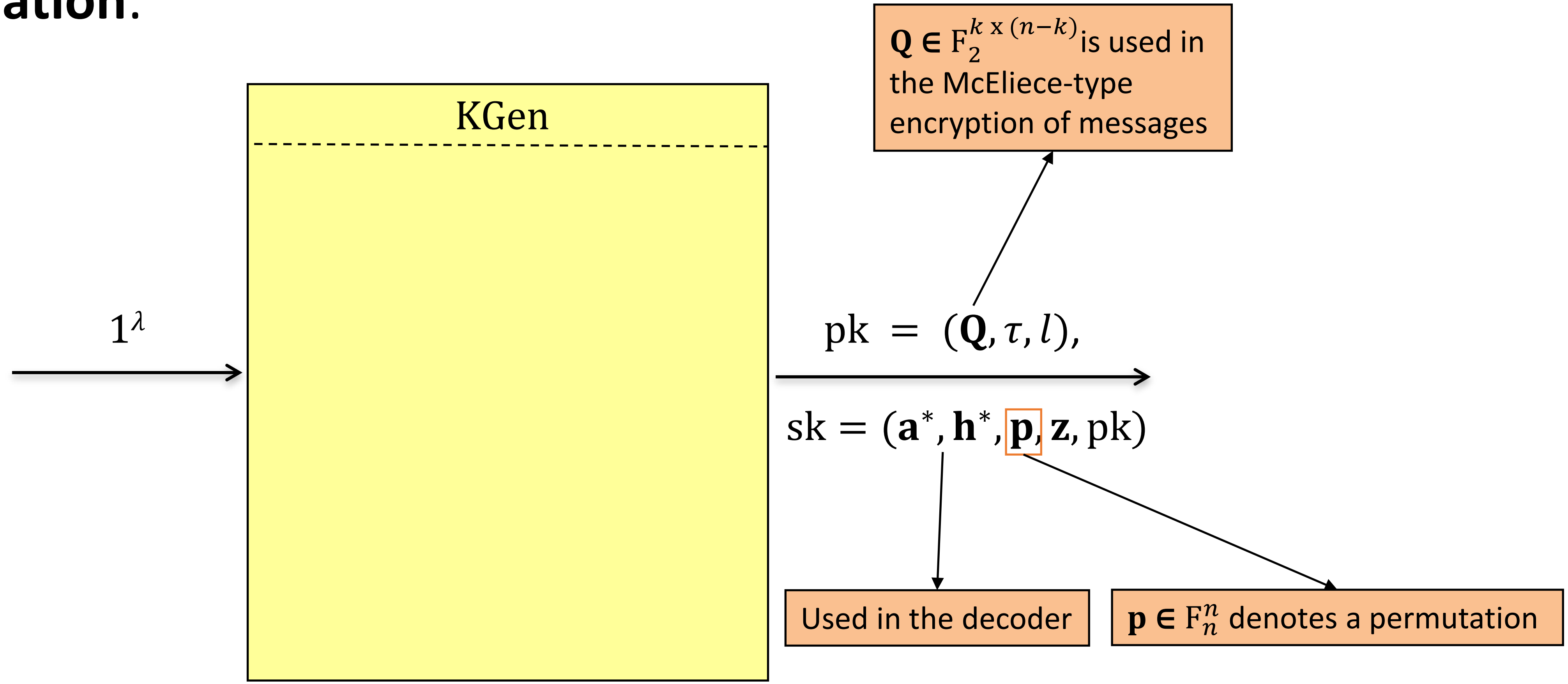
NTS-KEM Specification

- **Key generation**:



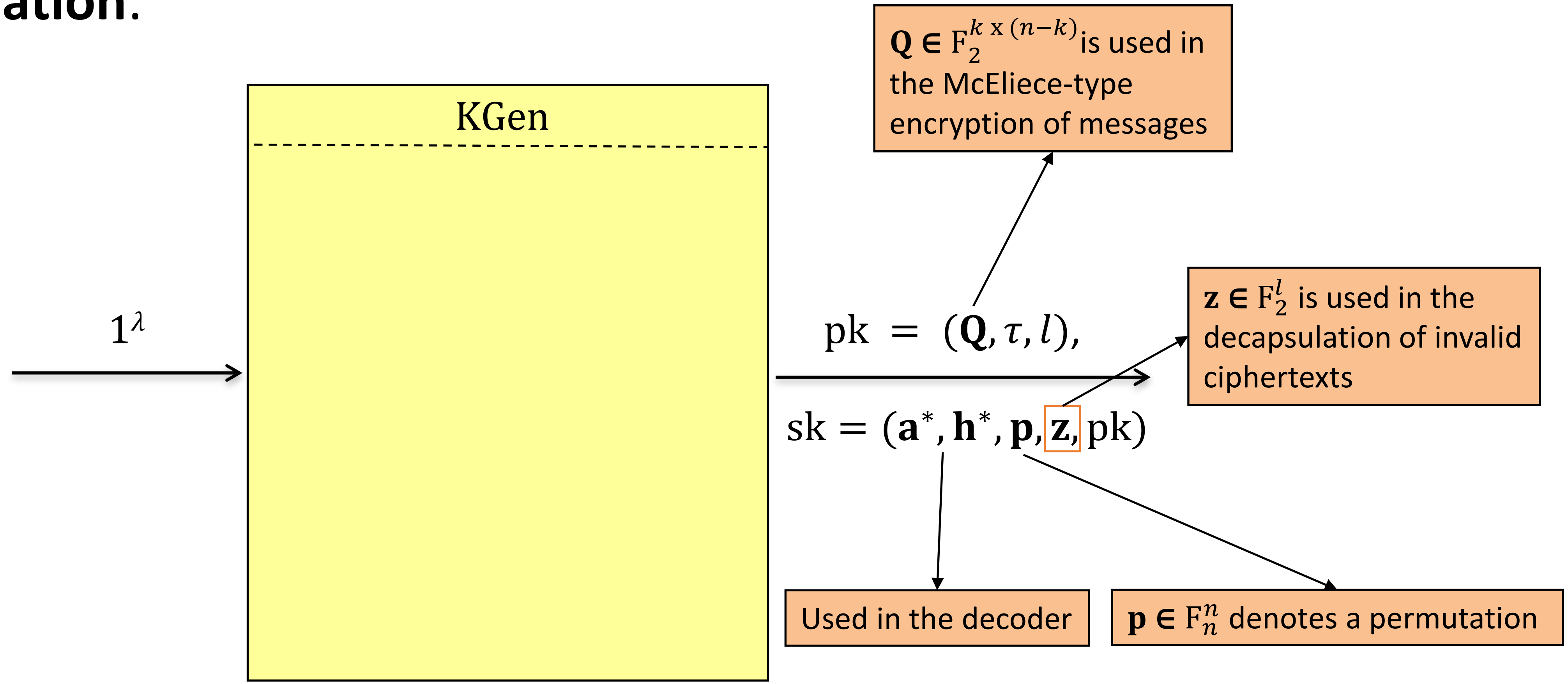
NTS-KEM Specification

- **Key generation**:



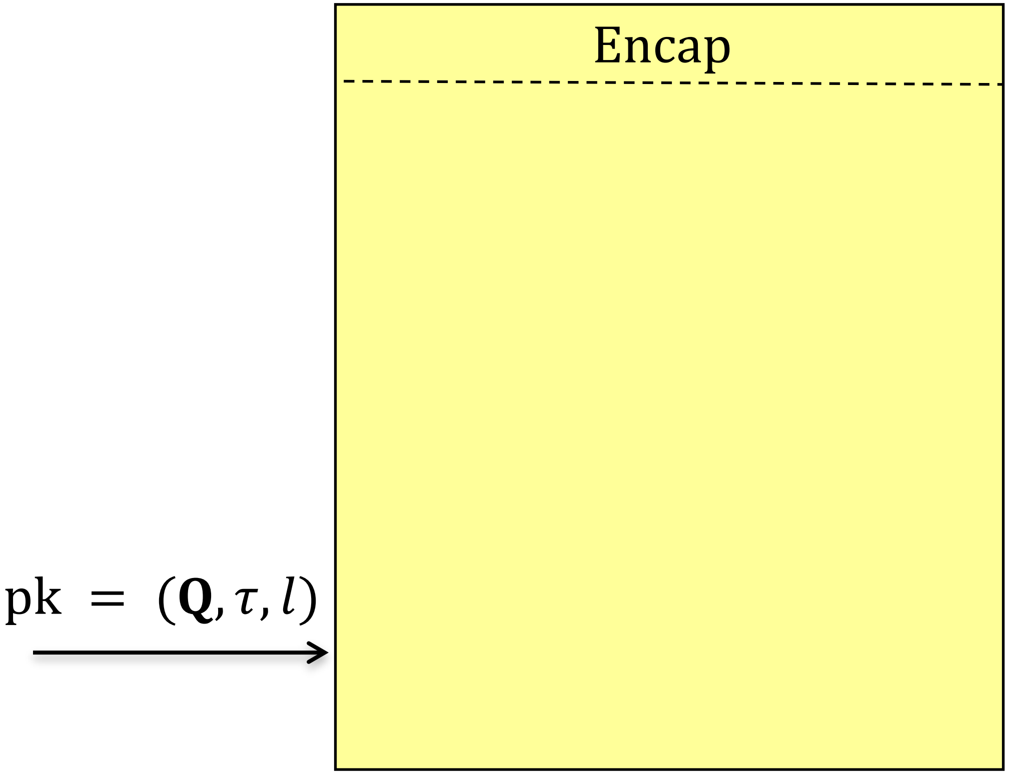
NTS-KEM Specification

- **Key generation**:



NTS-KEM Specification

- **Encapsulation**:



NTS-KEM Specification

- **Encapsulation**:

Encap

Sample error at random:

- $\mathbf{e} \in \mathrm{F}_2^n$ (with $\mathrm{hw(e)} = \tau$)

$\mathrm{pk} = (\mathbf{Q}, \tau, l)$

NTS-KEM Specification

- **Encapsulation**:

Encap

Sample error at random:

- $\mathbf{e} \in \mathbb{F}_2^n$ (with $\mathrm{hw}(\mathrm{e}) = \tau$)

Compute $\mathbf{k_e} = H_l(\mathbf{e}) \in \mathbb{F}_2^l$

$\mathrm{pk} = (\mathbf{Q}, \tau, l)$ →

NTS-KEM Specification

- **Encapsulation**:

Encap

Sample error at random:

- $\mathbf{e} \in \mathrm{F}_2^n$ (with $\mathrm{hw(e)} = \tau$)

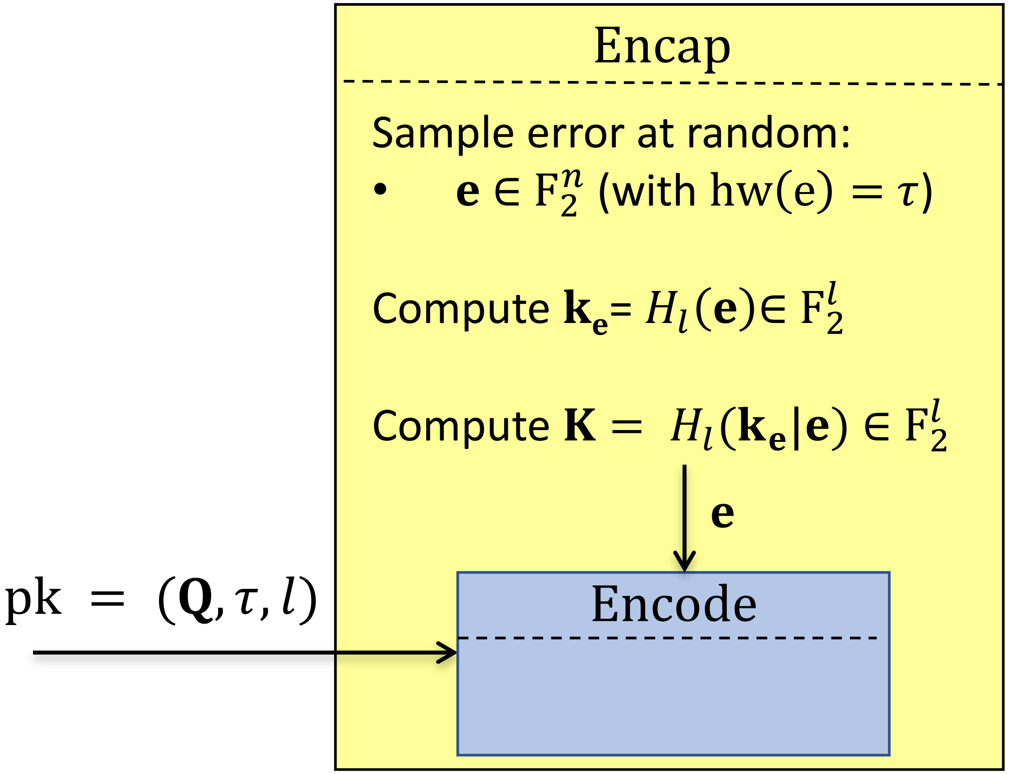
Compute $\mathbf{k_e} = H_l(\mathbf{e}) \in \mathrm{F}_2^l$

Compute $\mathbf{K} = H_l(\mathbf{k_e}|\mathbf{e}) \in \mathrm{F}_2^l$

$\mathrm{pk} = (\mathbf{Q}, \tau, l)$ →

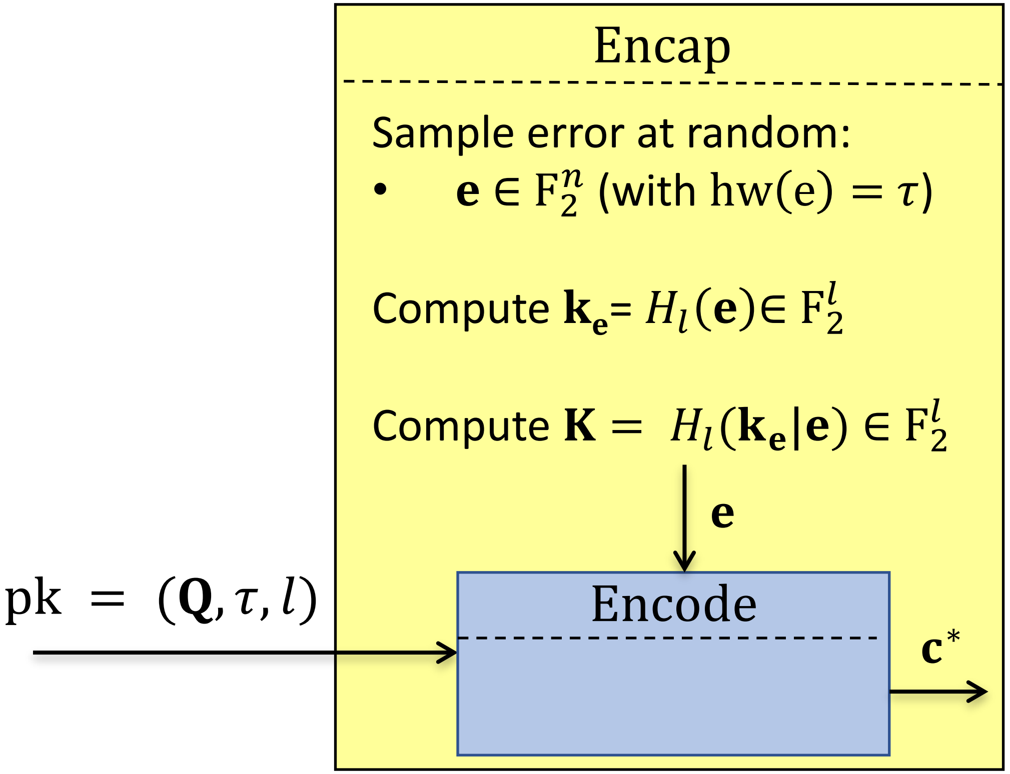
NTS-KEM Specification

- **Encapsulation**:



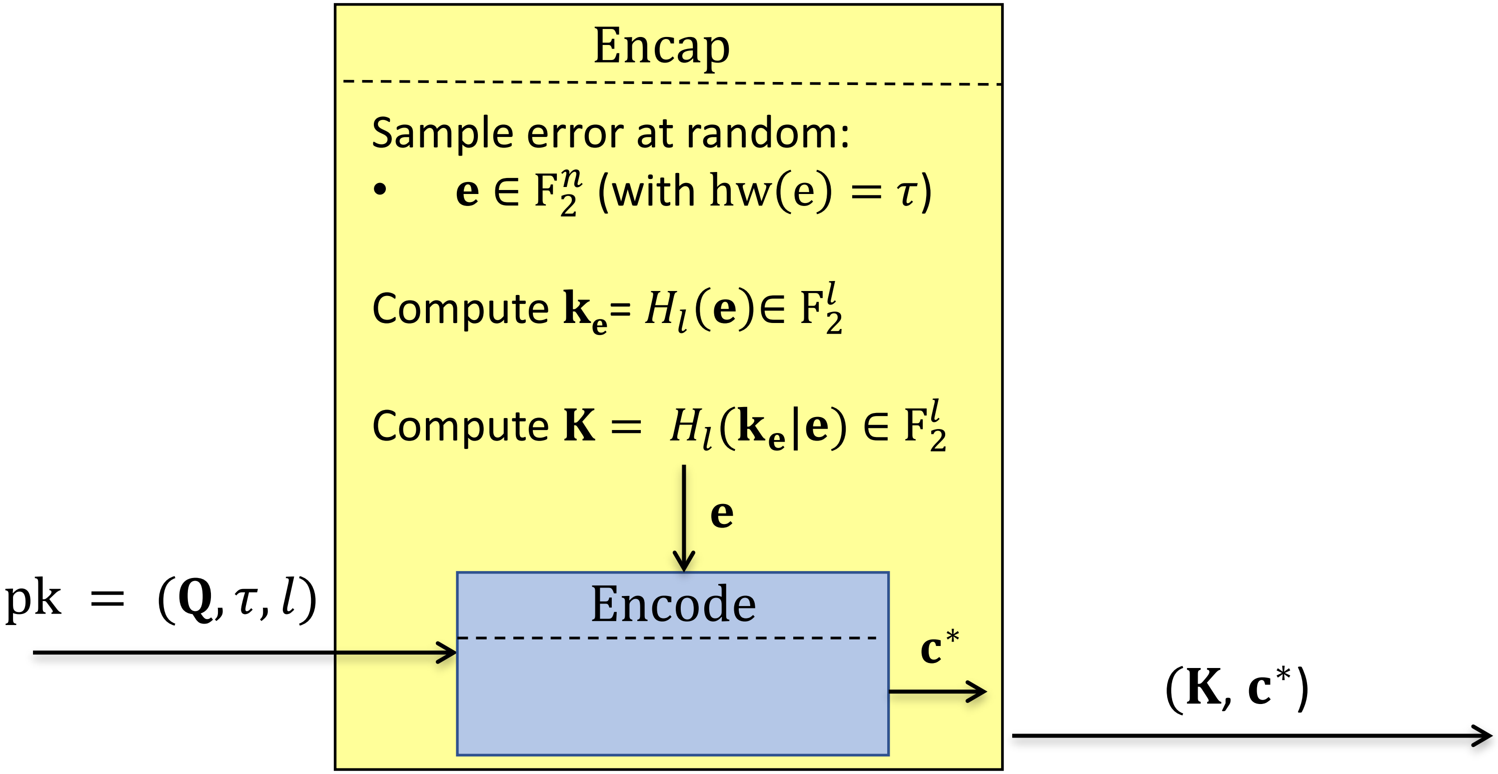
NTS-KEM Specification

- **Encapsulation**:



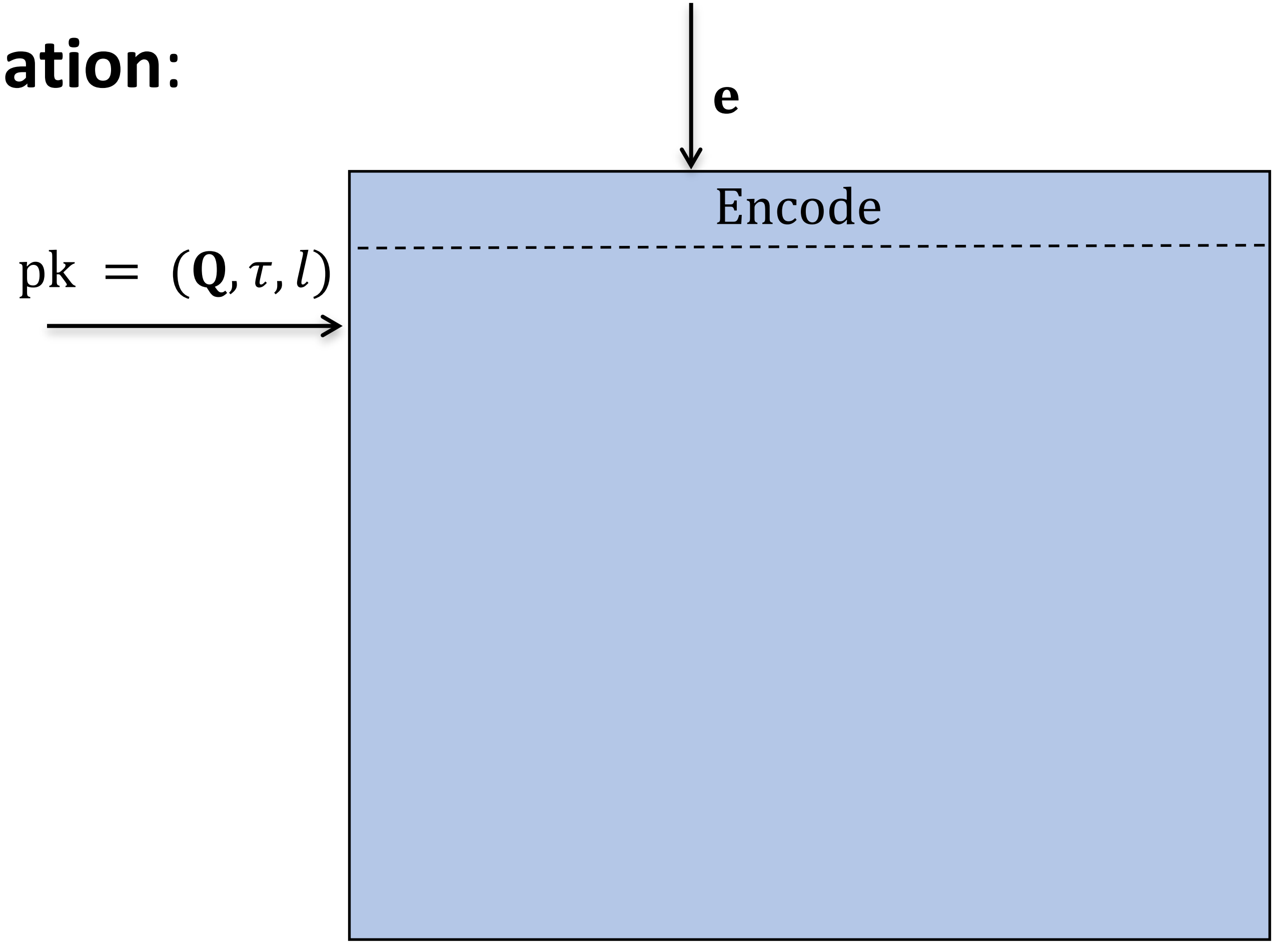
NTS-KEM Specification

- **Encapsulation**:



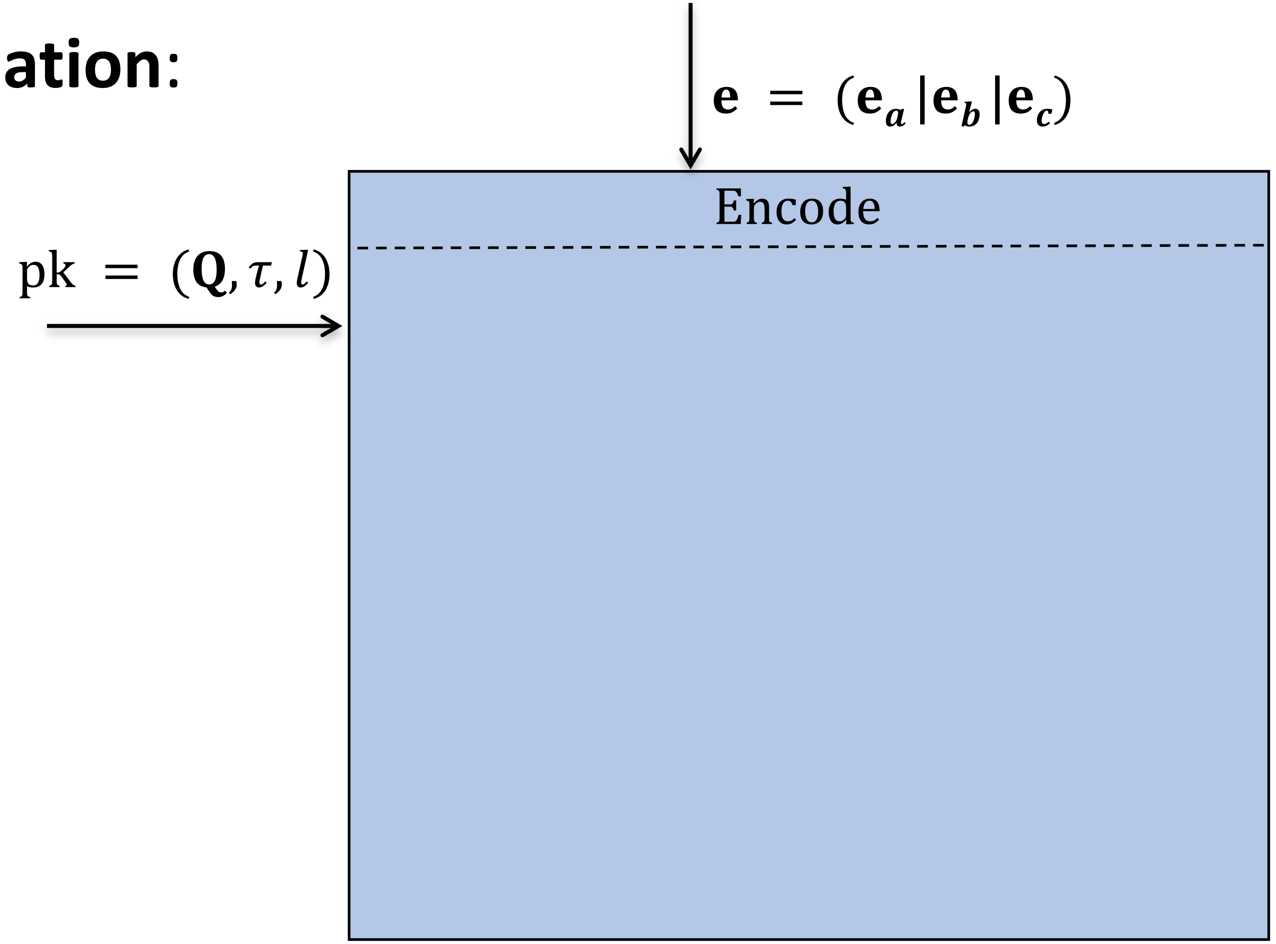
NTS-KEM Specification

- **Encapsulation**:



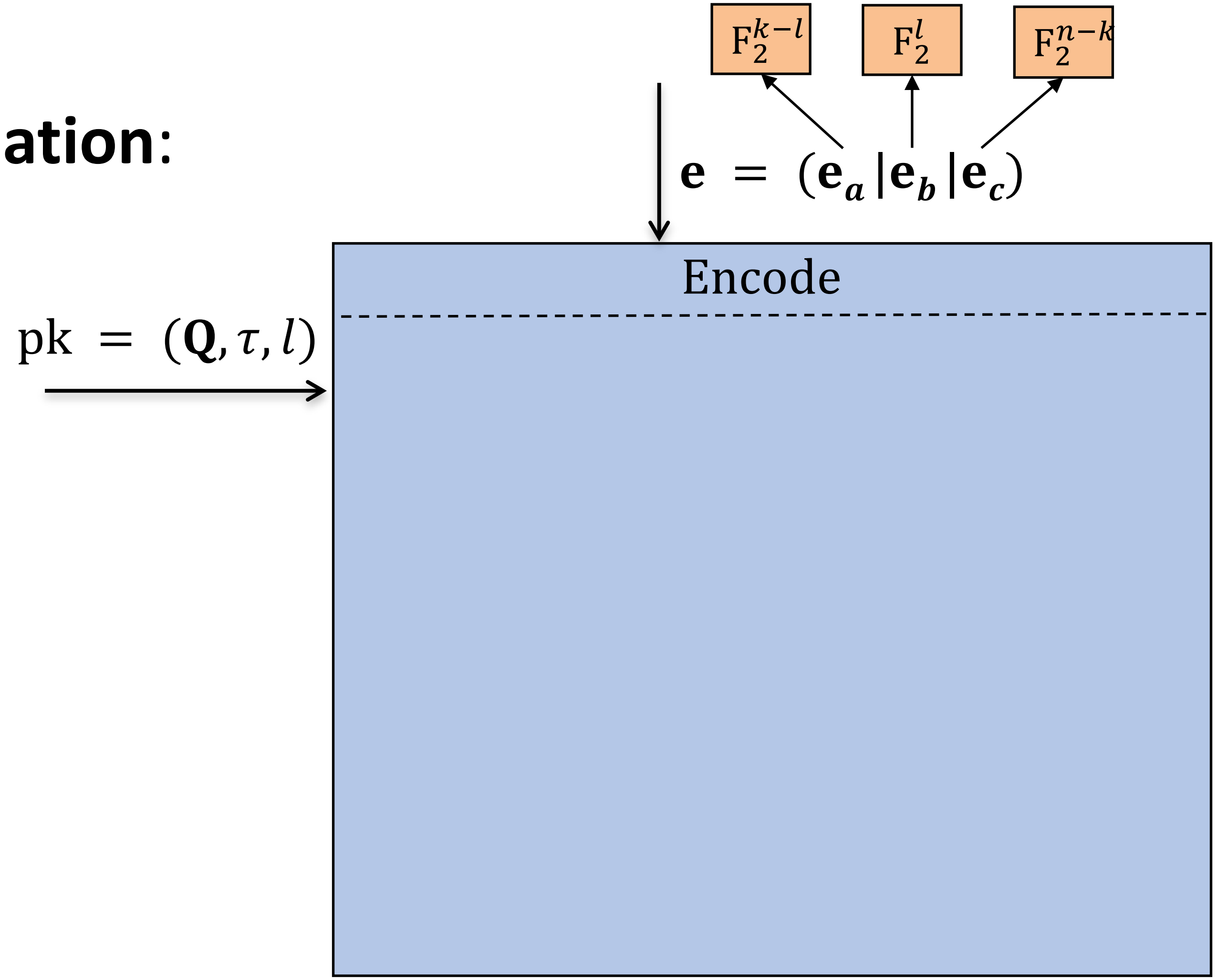
NTS-KEM Specification

- **Encapsulation**:



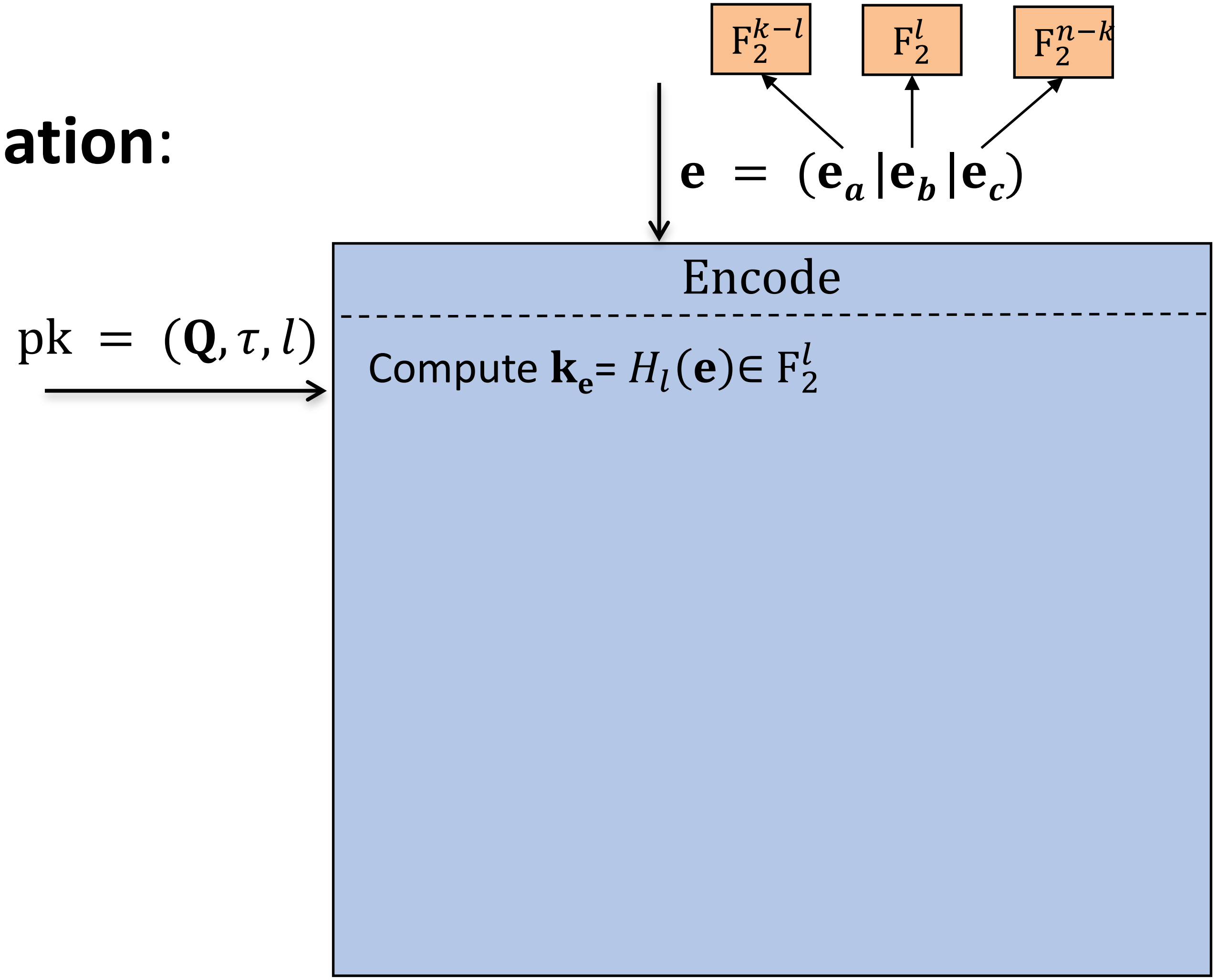
NTS-KEM Specification

- **Encapsulation**:



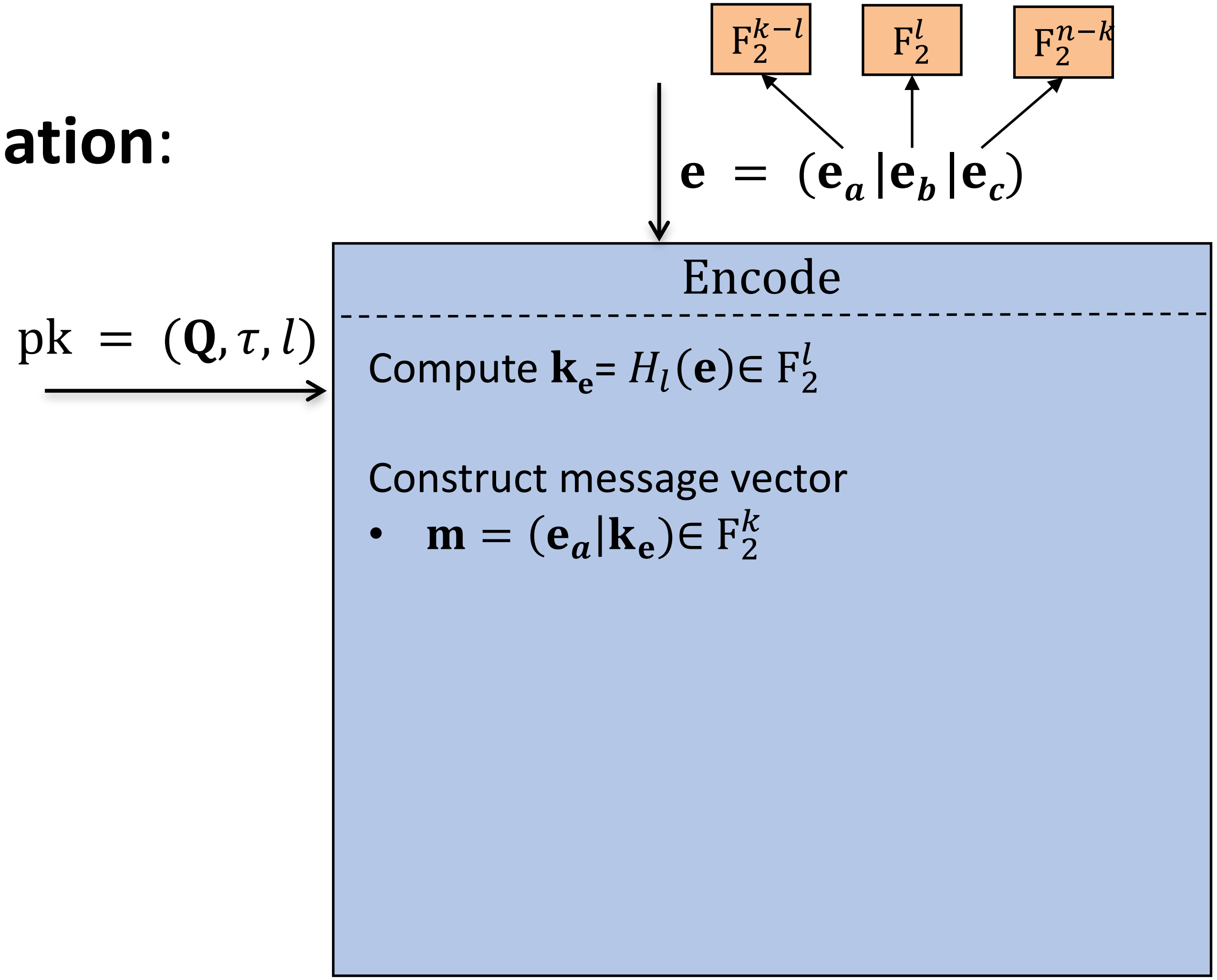
NTS-KEM Specification

- **Encapsulation**:



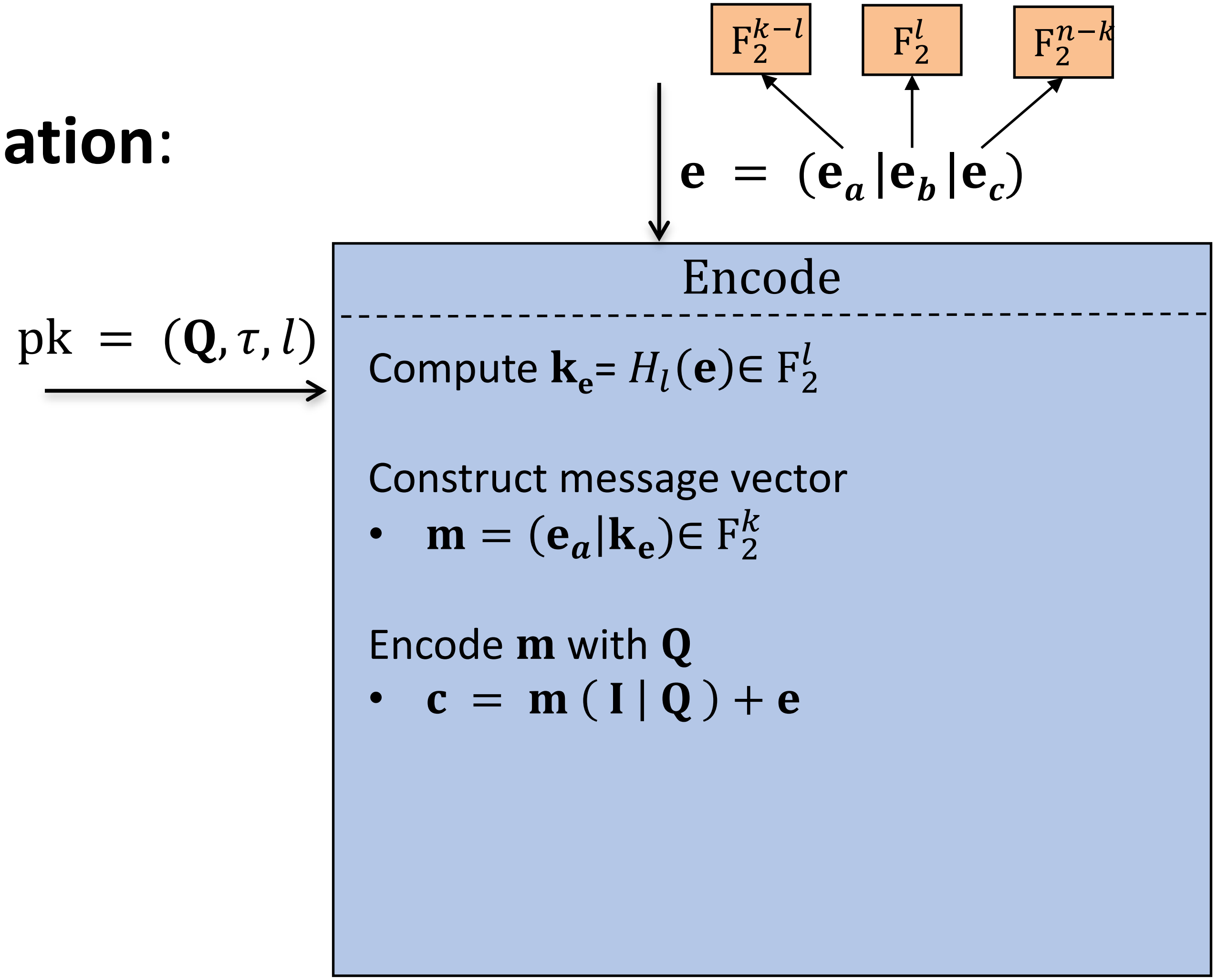
NTS-KEM Specification

- **Encapsulation**:



NTS-KEM Specification

- **Encapsulation**:



NTS-KEM Specification

- **Encapsulation**:

$\mathbb{F}_2^{k-l}$ $\mathbb{F}_2^{l}$ $\mathbb{F}_2^{n-k}$

$\mathbf{e} = (\mathbf{e}_a \,|\, \mathbf{e}_b \,|\, \mathbf{e}_c)$

$\text{pk} = (\mathbf{Q}, \tau, l)$

Encode

Compute $\mathbf{k_e} = H_l(\mathbf{e}) \in \mathbb{F}_2^{l}$

Construct message vector

- $\mathbf{m} = (\mathbf{e}_a | \mathbf{k_e}) \in \mathbb{F}_2^{k}$

Encode $\mathbf{m}$ with $\mathbf{Q}$

- $\mathbf{c} = \mathbf{m}\,(\,\mathbf{I} \,|\, \mathbf{Q}\,) + \mathbf{e}$

$= (\mathbf{e}_a \,|\, \mathbf{k_e} \,|\, (\mathbf{e}_a | \mathbf{k_e})\, \mathbf{Q}) + (\mathbf{e}_a \,|\, \mathbf{e}_b \,|\, \mathbf{e}_c)$

NTS-KEM Specification

- **Encapsulation**:

$\mathbb{F}_2^{k-l}$ $\mathbb{F}_2^{l}$ $\mathbb{F}_2^{n-k}$

$\mathbf{e} = (\mathbf{e}_a \,|\, \mathbf{e}_b \,|\, \mathbf{e}_c)$

$\mathrm{pk} = (\mathbf{Q}, \tau, l)$

Encode

Compute $\mathbf{k_e} = H_l(\mathbf{e}) \in \mathbb{F}_2^{l}$

Construct message vector

- $\mathbf{m} = (\mathbf{e}_a | \mathbf{k_e}) \in \mathbb{F}_2^{k}$

Encode $\mathbf{m}$ with $\mathbf{Q}$

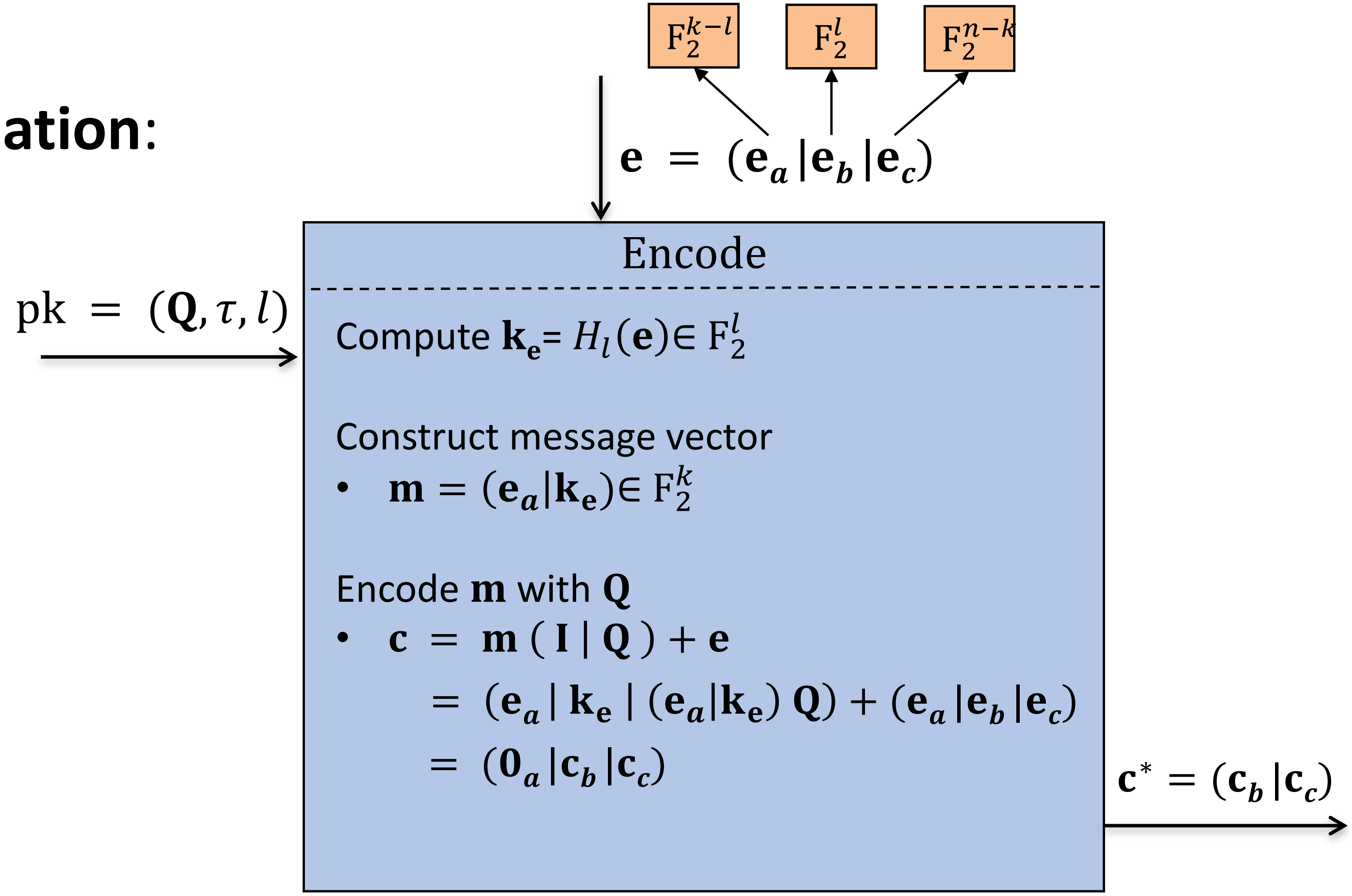
- $\mathbf{c} = \mathbf{m}\,(\,\mathbf{I} \,|\, \mathbf{Q}\,) + \mathbf{e}$

$= (\mathbf{e}_a \,|\, \mathbf{k_e} \,|\, (\mathbf{e}_a | \mathbf{k_e})\,\mathbf{Q}) + (\mathbf{e}_a \,|\, \mathbf{e}_b \,|\, \mathbf{e}_c)$

$= (\mathbf{0}_a \,|\, \mathbf{c}_b \,|\, \mathbf{c}_c)$

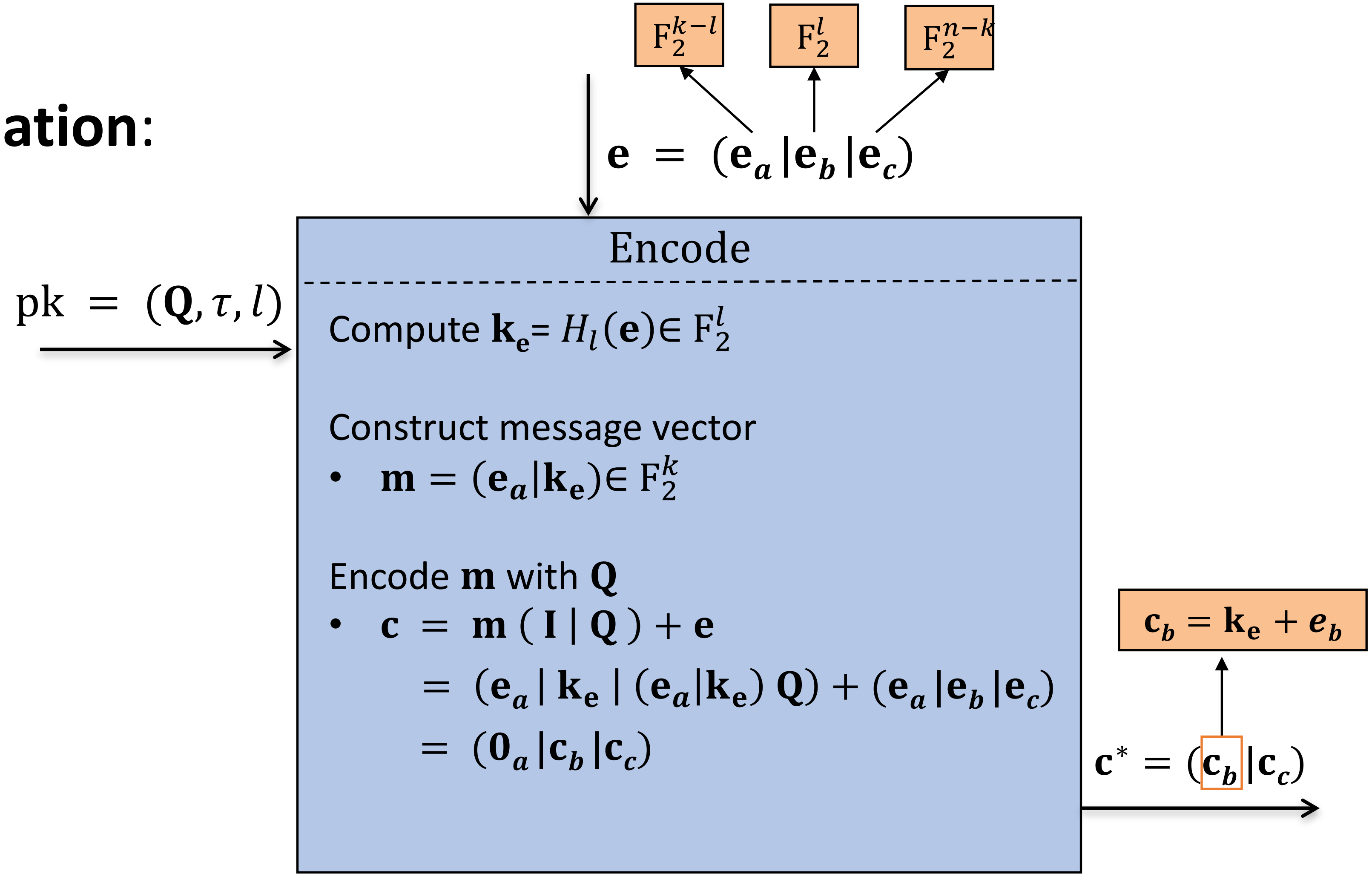
NTS-KEM Specification

- **Encapsulation**:



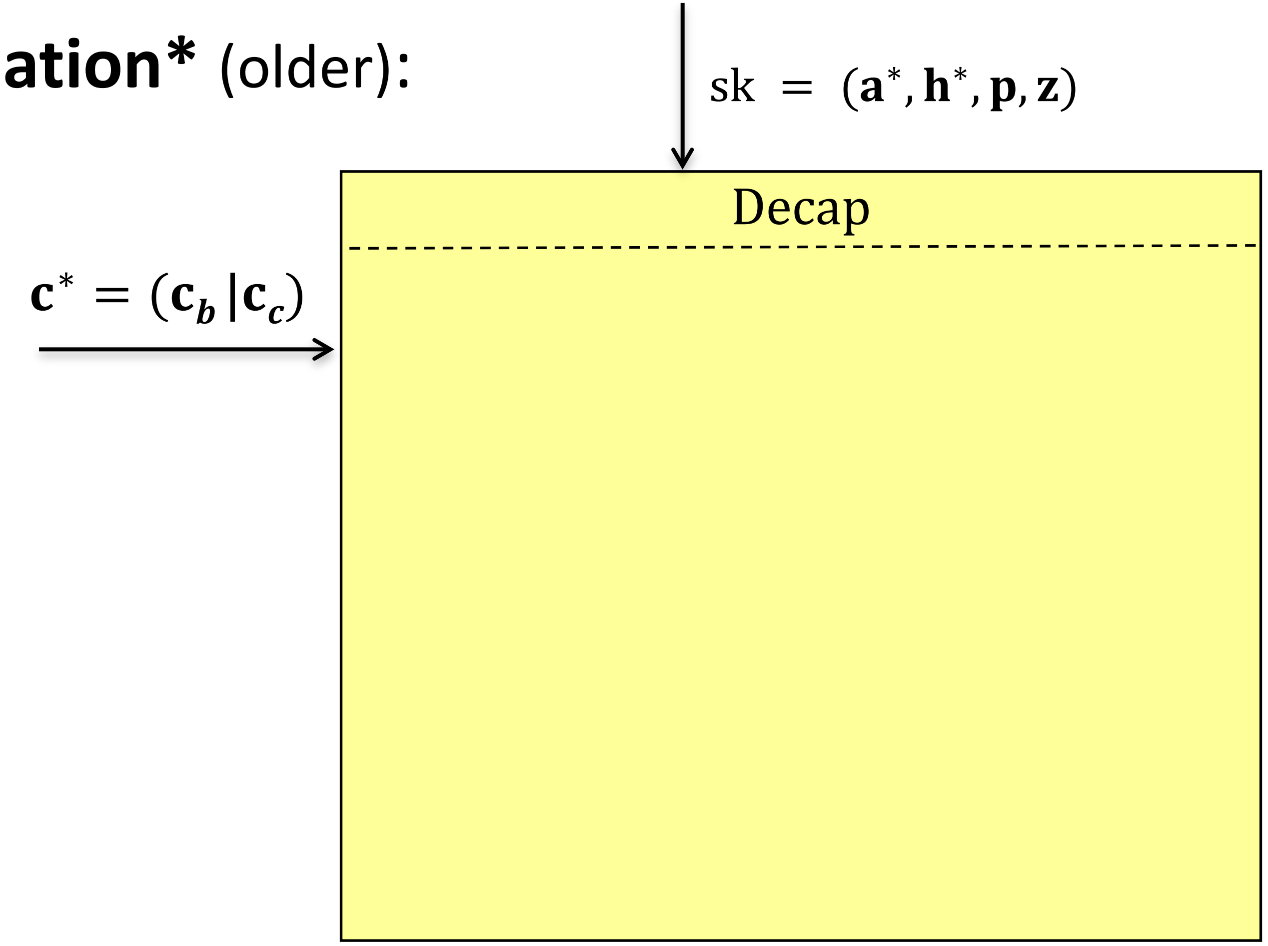
NTS-KEM Specification

- **Encapsulation**:



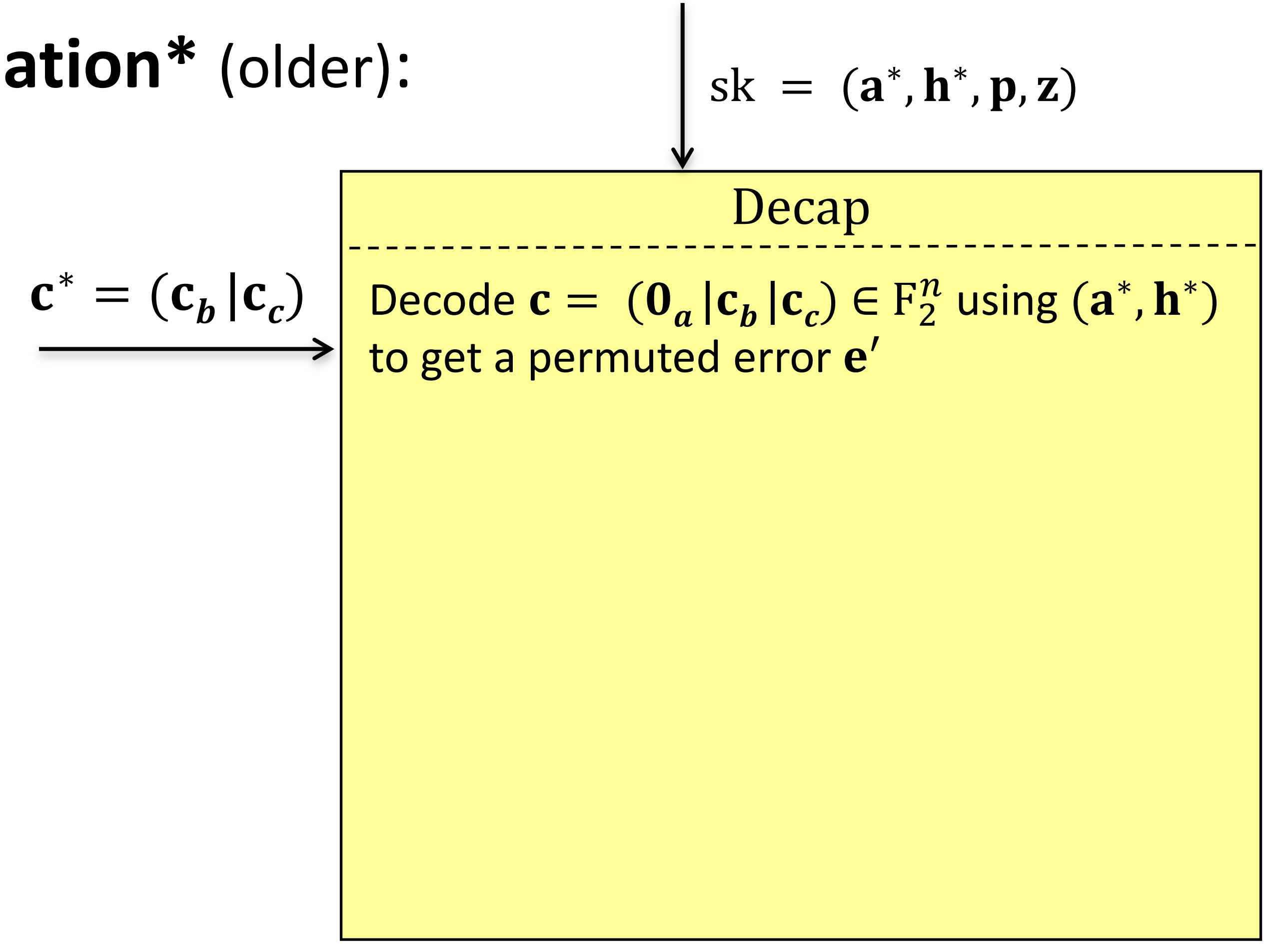
NTS-KEM Specification

- **Decapsulation*** (older):



NTS-KEM Specification

- **Decapsulation*** (older):



NTS-KEM Specification

- **Decapsulation*** (older):

$\mathrm{sk} = (\mathbf{a}^*, \mathbf{h}^*, \mathbf{p}, \mathbf{z})$

$\mathbf{c}^* = (\mathbf{c}_b | \mathbf{c}_c)$

Decap

Decode $\mathbf{c} = (\mathbf{0}_a | \mathbf{c}_b | \mathbf{c}_c) \in \mathrm{F}_2^n$ using $(\mathbf{a}^*, \mathbf{h}^*)$ to get a permuted error $\mathbf{e}'$

Invert permutation using $\mathbf{p} \in \mathrm{F}_n^n$ to obtain error vector $\mathbf{e}$

NTS-KEM Specification

- **Decapsulation*** (older):

$\mathrm{sk} = (\mathbf{a}^*, \mathbf{h}^*, \mathbf{p}, \mathbf{z})$

$\mathbf{c}^* = (\mathbf{c}_b | \mathbf{c}_c)$

Decap

Decode $\mathbf{c} = (\mathbf{0}_a | \mathbf{c}_b | \mathbf{c}_c) \in \mathrm{F}_2^n$ using $(\mathbf{a}^*, \mathbf{h}^*)$ to get a permuted error $\mathbf{e}'$

Invert permutation using $\mathbf{p} \in \mathrm{F}_n^n$ to obtain error vector $\mathbf{e}$

$\mathbf{e} = (\mathbf{e}_a | \mathbf{e}_b | \mathbf{e}_c)$ and compute $\mathbf{k}_\mathbf{e} = \mathbf{c}_b - \boldsymbol{e}_b$

NTS-KEM Specification

- **Decapsulation*** (older):

$\mathrm{sk} = (\mathbf{a}^*, \mathbf{h}^*, \mathbf{p}, \mathbf{z})$

$\mathbf{c}^* = (\mathbf{c}_b | \mathbf{c}_c)$

Decap

Decode $\mathbf{c} = (\mathbf{0}_a | \mathbf{c}_b | \mathbf{c}_c) \in \mathrm{F}_2^n$ using $(\mathbf{a}^*, \mathbf{h}^*)$ to get a permuted error $\mathbf{e}'$

Invert permutation using $\mathbf{p} \in \mathrm{F}_n^n$ to obtain error vector $\mathbf{e}$

$\mathbf{e} = (\mathbf{e}_a | \mathbf{e}_b | \mathbf{e}_c)$ and compute $\mathbf{k_e} = \mathbf{c}_b - \boldsymbol{e}_b$

If $H_l(\mathbf{e}) = \mathbf{k_e}$ and $\mathrm{hw}(\mathbf{e}) = \tau$

$\mathbf{K} = H_l(\mathbf{k_e} | \mathbf{e})$

NTS-KEM Specification

- **Decapsulation*** (older):

$\mathrm{sk} = (\mathbf{a}^*, \mathbf{h}^*, \mathbf{p}, \mathbf{z})$

$\mathbf{c}^* = (\mathbf{c}_b | \mathbf{c}_c)$

Decap

Decode $\mathbf{c} = (\mathbf{0}_a | \mathbf{c}_b | \mathbf{c}_c) \in \mathrm{F}_2^n$ using $(\mathbf{a}^*, \mathbf{h}^*)$ to get a permuted error $\mathbf{e}'$

Invert permutation using $\mathbf{p} \in \mathrm{F}_n^n$ to obtain error vector $\mathbf{e}$

$\mathbf{e} = (\mathbf{e}_a | \mathbf{e}_b | \mathbf{e}_c)$ and compute $\mathbf{k}_\mathbf{e} = \mathbf{c}_b - \boldsymbol{e}_b$

If $H_l(\mathbf{e}) = \mathbf{k}_\mathbf{e}$ and $\mathrm{hw}(\mathbf{e}) = \tau$

$\mathbf{K} = H_l(\mathbf{k}_\mathbf{e} | \mathbf{e})$

$\mathbf{K} = H_l(\mathbf{z} | \mathbf{1}_a | \mathbf{c}_b | \mathbf{c}_c)$

Otherwise

Bug in the initial ROM proof

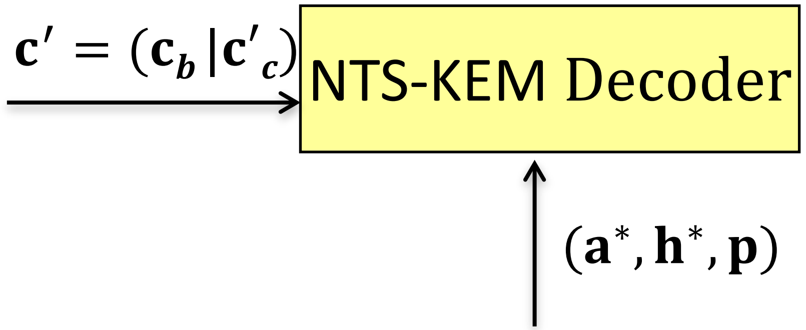
- Failed to account for the following ciphertexts:

$$\mathbf{c}' = (\mathbf{c}_{\boldsymbol{b}} | \mathbf{c}'_{\boldsymbol{c}})$$

$\longrightarrow$

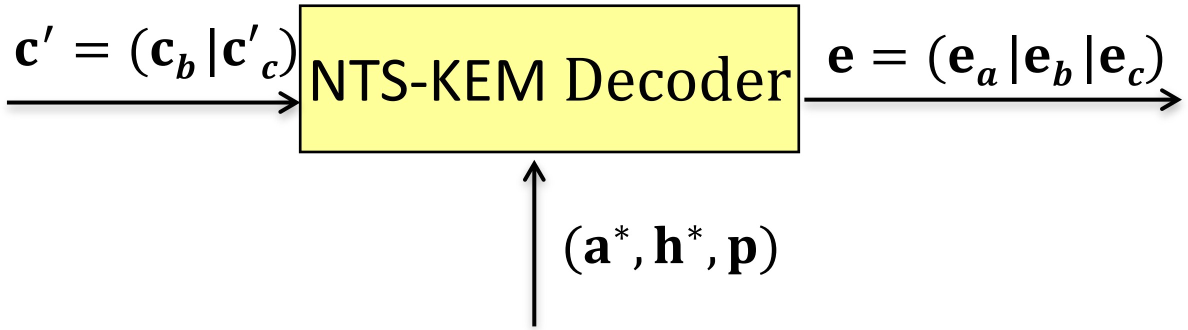
Bug in the initial ROM proof

- Failed to account for the following ciphertexts:



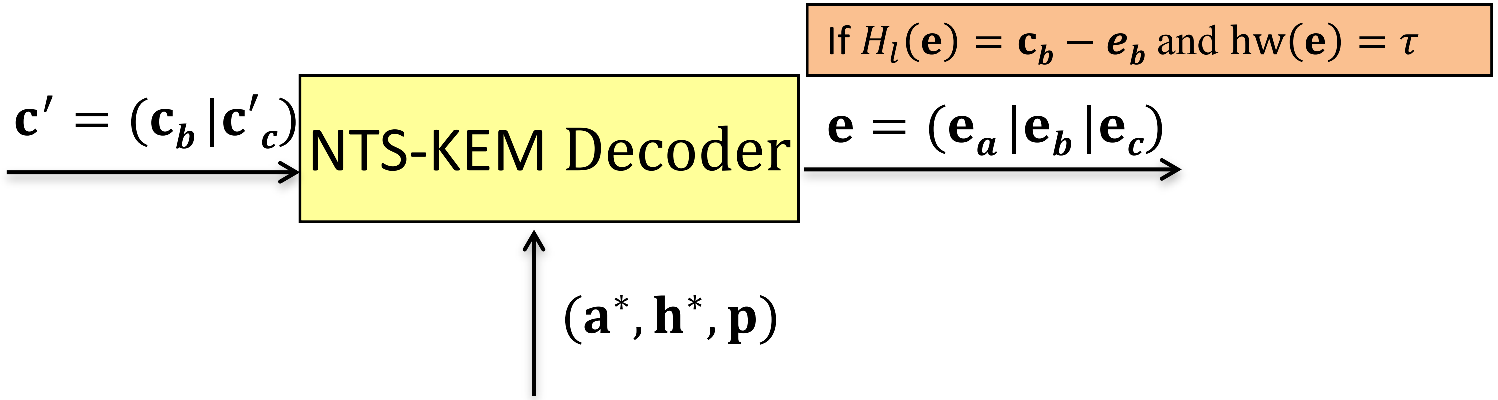
Bug in the initial ROM proof

- Failed to account for the following ciphertexts:



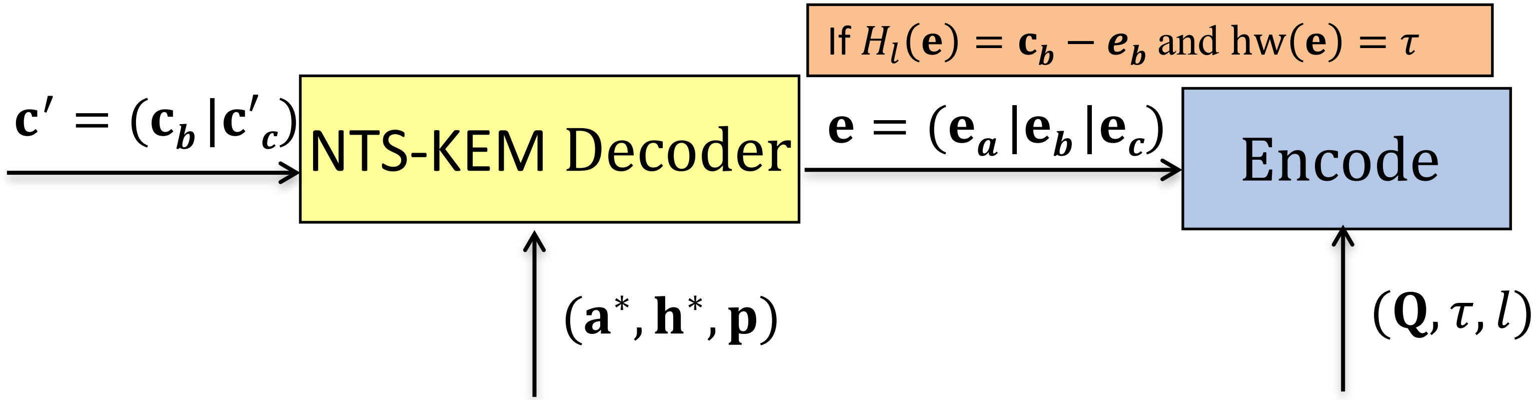
Bug in the initial ROM proof

- Failed to account for the following ciphertexts:



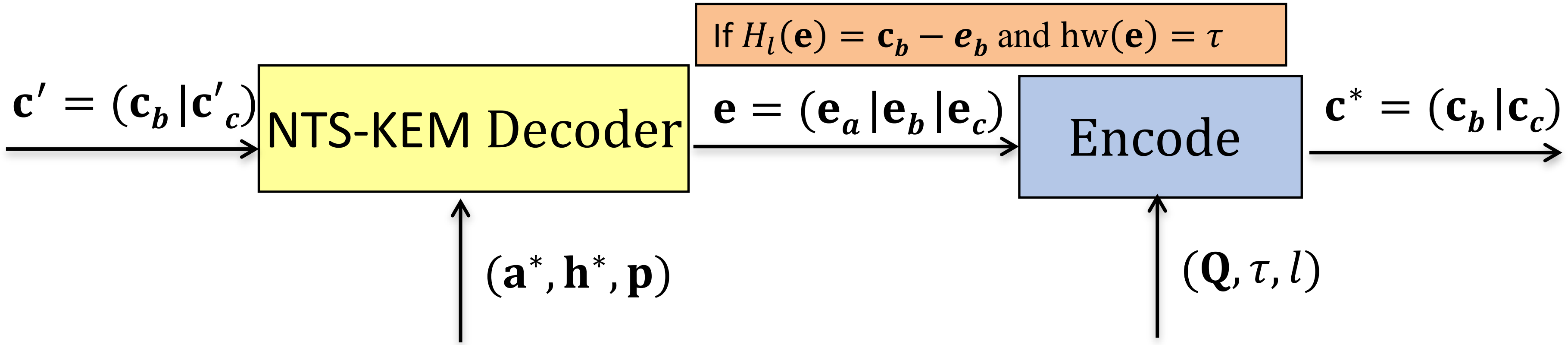
Bug in the initial ROM proof

- Failed to account for the following ciphertexts:



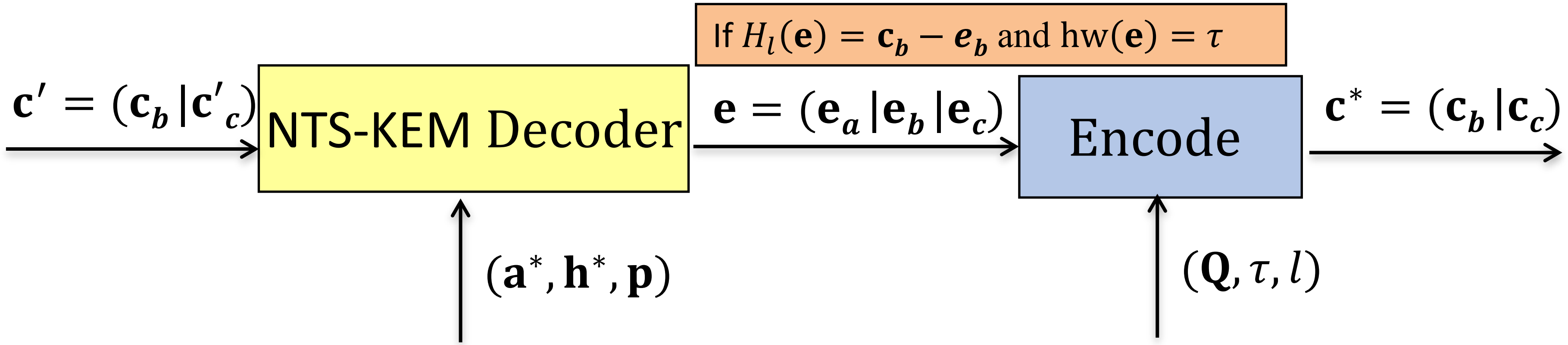
Bug in the initial ROM proof

- Failed to account for the following ciphertexts:

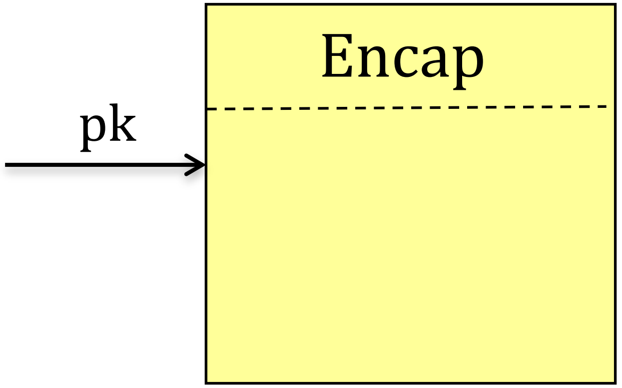


Bug in the initial ROM proof

- Failed to account for the following ciphertexts:

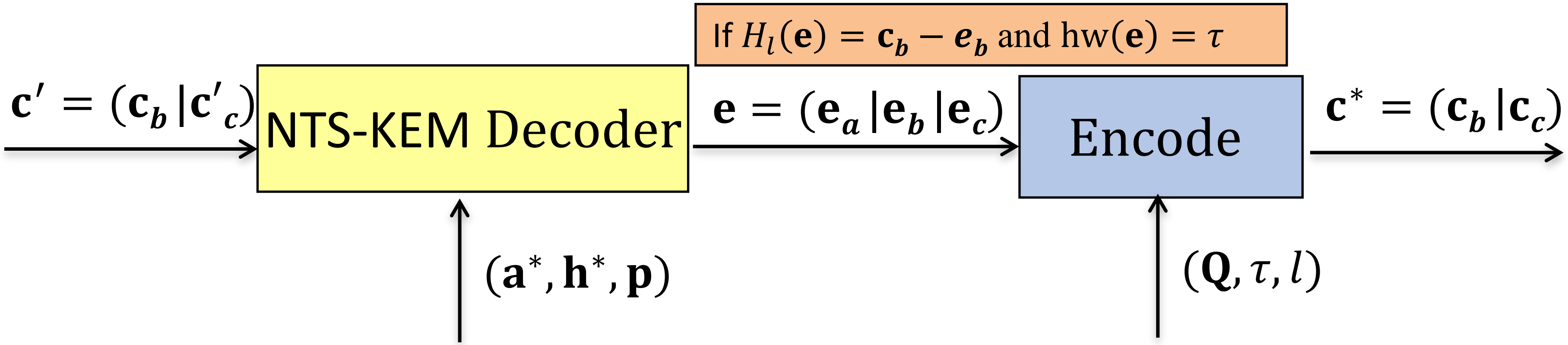


- Because of NTS-KEM correctness:

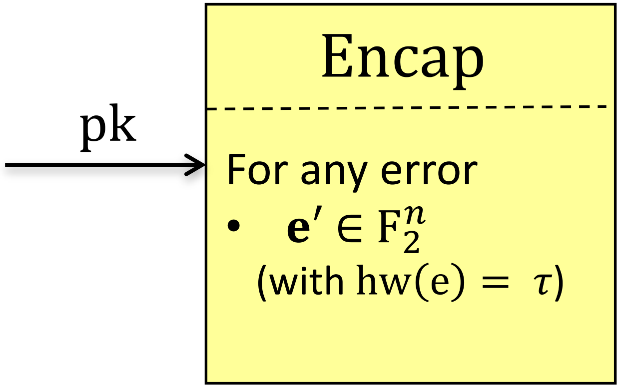


Bug in the initial ROM proof

- Failed to account for the following ciphertexts:

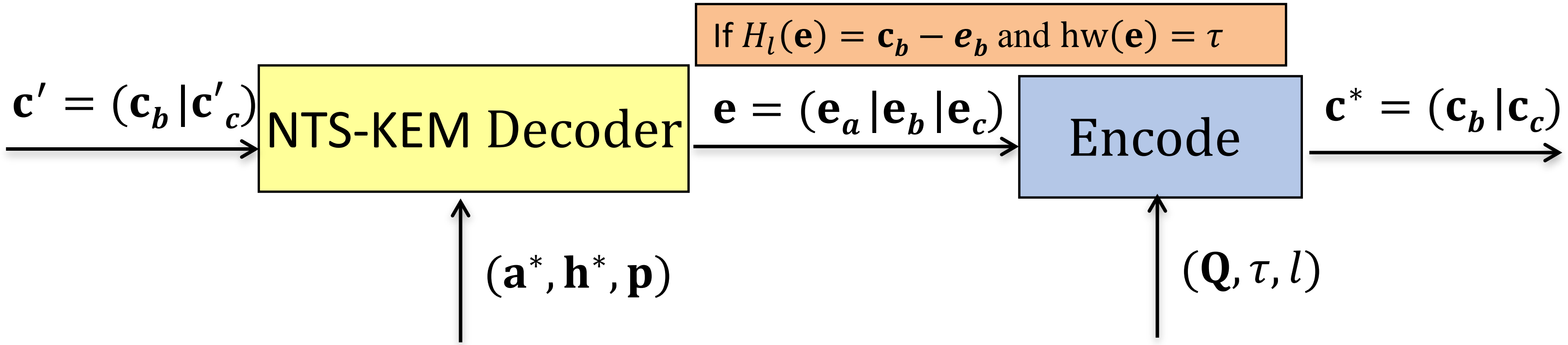


- Because of NTS-KEM correctness:

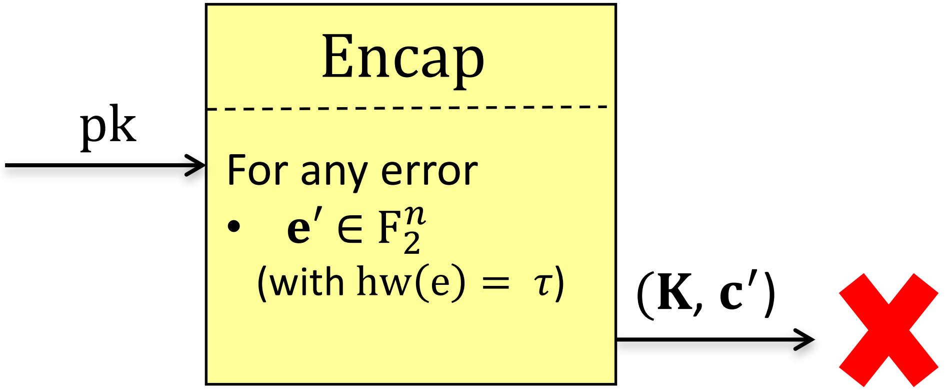


Bug in the initial ROM proof

- Failed to account for the following ciphertexts:

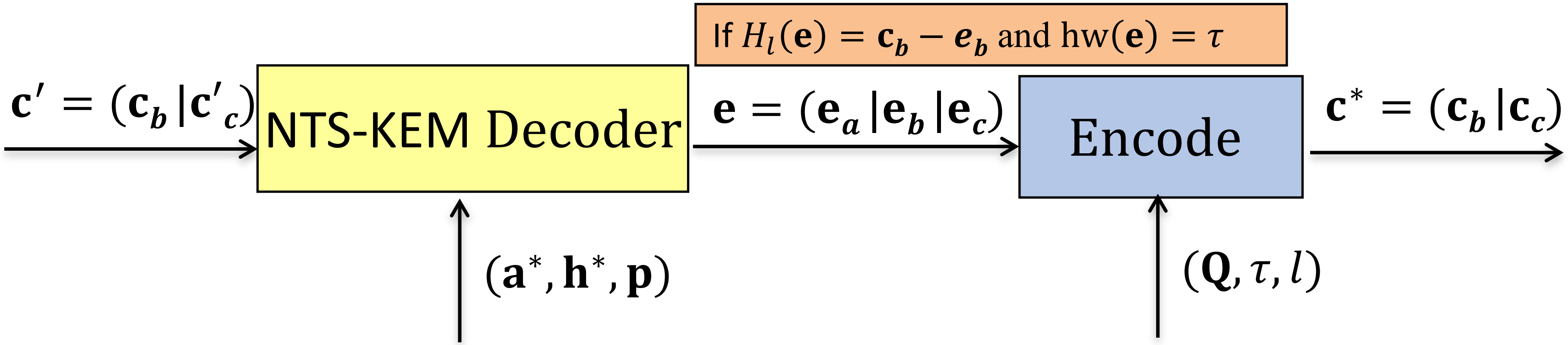


- Because of NTS-KEM correctness:

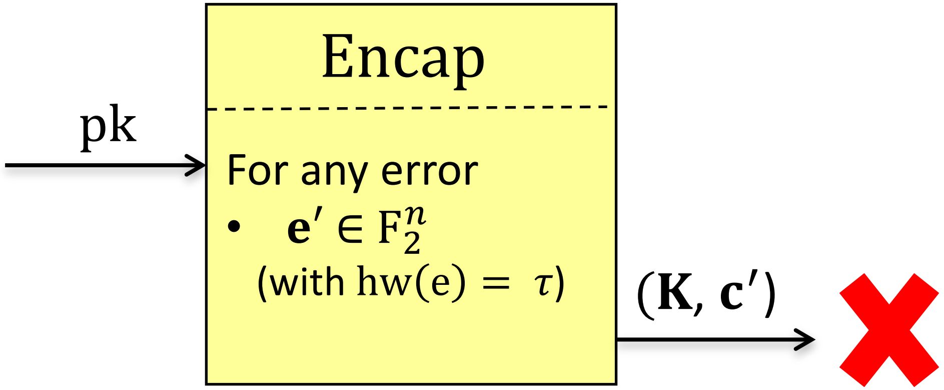


Bug in the initial ROM proof

- Failed to account for the following ciphertexts:



- Because of NTS-KEM correctness:



Invalid $\mathbf{c}'$ "accepted" by older NTS-KEM Decap!

NTS-KEM Specification

- **Decapsulation*** (older):

$\text{sk} = (\mathbf{a}^*, \mathbf{h}^*, \mathbf{p}, \mathbf{z})$

$\mathbf{c}^* = (\mathbf{c}_b | \mathbf{c}_c)$

Decap

Decode $\mathbf{c} = (\mathbf{0}_a | \mathbf{c}_b | \mathbf{c}_c) \in \mathrm{F}_2^n$ using $(\mathbf{a}^*, \mathbf{h}^*)$ to get a permuted error $\mathbf{e}'$

Invert permutation using $\mathbf{p} \in \mathrm{F}_n^n$ to obtain error vector $\mathbf{e}$

$\mathbf{e} = (\mathbf{e}_a | \mathbf{e}_b | \mathbf{e}_c)$ and compute $\mathbf{k}_\mathbf{e} = \mathbf{c}_b - \boldsymbol{e}_b$

If $H_l(\mathbf{e}) = \mathbf{k}_\mathbf{e}$ and $\mathrm{hw}(\mathbf{e}) = \tau$

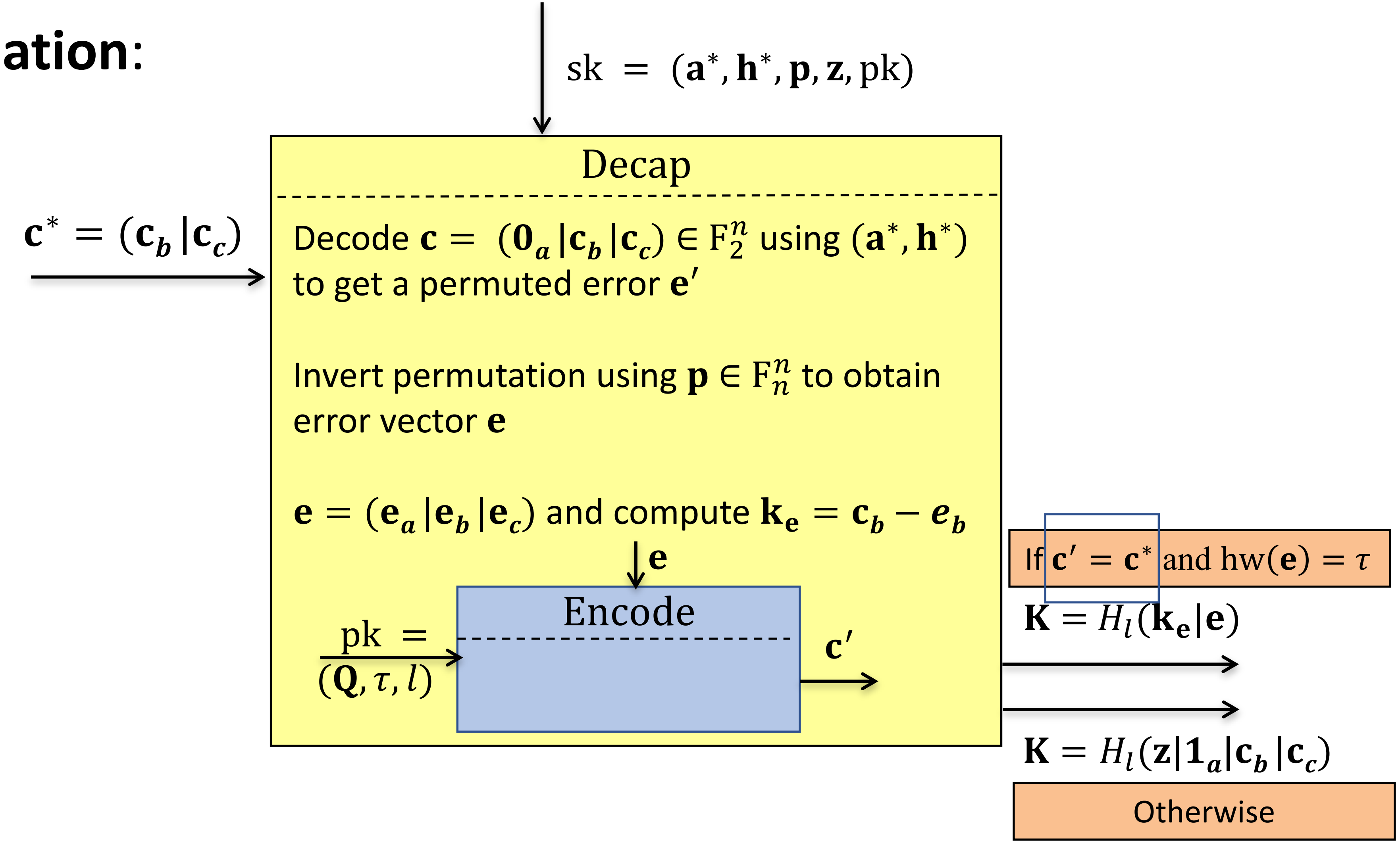
$\mathbf{K} = H_l(\mathbf{k}_\mathbf{e} | \mathbf{e})$

$\mathbf{K} = H_l(\mathbf{z} | \mathbf{1}_a | \mathbf{c}_b | \mathbf{c}_c)$

Otherwise

NTS-KEM Specification

- **Decapsulation:**



Changes to NTS-KEM Decapsulation

If $\text{Encode}(\text{pk}, \mathbf{e}) = \mathbf{c}^*$ and $\text{hw}(\mathbf{e}) = \tau$

Changes to NTS-KEM Decapsulation

If $\text{Encode}(\text{pk}, \mathbf{e}) = \mathbf{c}^*$ and $\text{hw}(\mathbf{e}) = \tau$ $\Rightarrow$ $H_l(\mathbf{e}) = \mathbf{c}_{\boldsymbol{b}} - \boldsymbol{e}_{\boldsymbol{b}}$ and $\text{hw}(\mathbf{e}) = \tau$

Changes to NTS-KEM Decapsulation

If $\text{Encode}(\text{pk}, \mathbf{e}) = \mathbf{c}^*$ and $\text{hw}(\mathbf{e}) = \tau$ $\Rightarrow$ $H_l(\mathbf{e}) = \mathbf{c}_{\boldsymbol{b}} - \boldsymbol{e}_{\boldsymbol{b}}$ and $\text{hw}(\mathbf{e}) = \tau$

Preserves the tightness of intended ROM proof for NTS-KEM.

Changes to NTS-KEM Decapsulation

If $\text{Encode}(\text{pk}, \mathbf{e}) = \mathbf{c}^* \text{ and } \text{hw}(\mathbf{e}) = \tau$ $\Rightarrow$ $H_l(\mathbf{e}) = \mathbf{c}_{\boldsymbol{b}} - \boldsymbol{e}_{\boldsymbol{b}} \text{ and } \text{hw}(\mathbf{e}) = \tau$

Preserves the tightness of intended ROM proof for NTS-KEM.

$\text{FO}_m^{\perp}$-style reencoding check makes NTS-KEM secure in QROM!

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_\$ \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_\$ \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM $= \mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	1 : $\mathbf{m} \leftarrow_{\$} \mathcal{M}$	1 : $\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
2 : $\mathbf{z} \leftarrow_{\$} \mathcal{M}$	2 : $\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	2 : $\textbf{if } \mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
3 : $\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	3 : $\mathbf{K} = H(\mathbf{m})$	3 : $\quad \textbf{return } H(\hat{\mathbf{m}})$
4 : $\textbf{return } (\mathsf{pk}, \mathsf{sk}')$	4 : $\textbf{return } (\mathbf{K}, \mathbf{c})$	4 : $\textbf{else return } H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM = $\mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

If $\mathrm{Encode}(\mathrm{pk}, \mathbf{e}) = \mathbf{c}^*$...

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM = $\mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

$H_l(\mathbf{z}|\mathbf{1}_a|\mathbf{c})$

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM = $\mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

$\mathbf{K} = H_l(\mathbf{k_e} | \mathbf{e})$

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM = $\mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

$\mathbf{m} = (\mathbf{e}_a | \mathbf{k}_\mathrm{e})$
$\mathbf{K} = H_l(\mathbf{k}_\mathrm{e} | \mathbf{e}_a | \mathbf{e}_b | \mathbf{e}_c)$

Comparisons b/w NTS-KEM and $\mathbf{FO}_m^{\perp}$

$\mathsf{KGen}_{\mathrm{KEM}}$	$\mathsf{Encap}(\mathsf{pk})$	$\mathsf{Decap}(\mathbf{c}, \mathsf{sk}')$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathrm{PKE}}$	$\mathbf{m} \leftarrow_{\$} \mathcal{M}$	$\hat{\mathbf{m}} = \mathsf{Dec}(\mathsf{sk}, \mathbf{c})$
$\mathbf{z} \leftarrow_{\$} \mathcal{M}$	$\mathbf{c} = \mathsf{Enc}(\mathsf{pk}, \mathbf{m}; G(\mathbf{m}))$	**if** $\mathsf{Enc}(\mathsf{pk}, \hat{\mathbf{m}}; G(\hat{\mathbf{m}})) = \mathbf{c}$
$\mathsf{sk}' = (\mathsf{sk}, \mathbf{z})$	$\mathbf{K} = H(\mathbf{m})$	**return** $H(\hat{\mathbf{m}})$
return $(\mathsf{pk}, \mathsf{sk}')$	**return** $(\mathbf{K}, \mathbf{c})$	**else return** $H(\mathbf{z} \mid \mathbf{c})$

Fig. IND-CCA secure KEM $= \mathrm{FO}_m^{\not\perp}[\mathrm{PKE}, G, H]$.

$\mathbf{e} \leftarrow {}_{\$}\mathrm{F}_2^n$ (with $\mathrm{hw}(\mathbf{e}) = \tau$)

$\mathbf{m} = (\mathbf{e}_a | H_l(\mathbf{e}))$

Proving Security of NTS-KEM in QROM

OW-security of PKE $\overset{\text{QROM}}{\Longrightarrow}$ IND-CCA-security of KEM = $\mathrm{FO}_m^{\perp}[\mathrm{PKE}, G, H]$

Proving Security of NTS-KEM in QROM

OW-security of PKE $\overset{\text{QROM}}{\Longrightarrow}$ IND-CCA-security of KEM = $\mathrm{FO}_m^{\perp}[\mathrm{PKE}, G, H]$

errors in NTS-KEM ≈ ***messages*** in FO-transform

Proving Security of NTS-KEM in QROM

OW-security of PKE $\overset{\text{QROM}}{\Longrightarrow}$ IND-CCA-security of KEM = $\mathrm{FO}_m^{\not\perp}[\text{PKE}, G, H]$

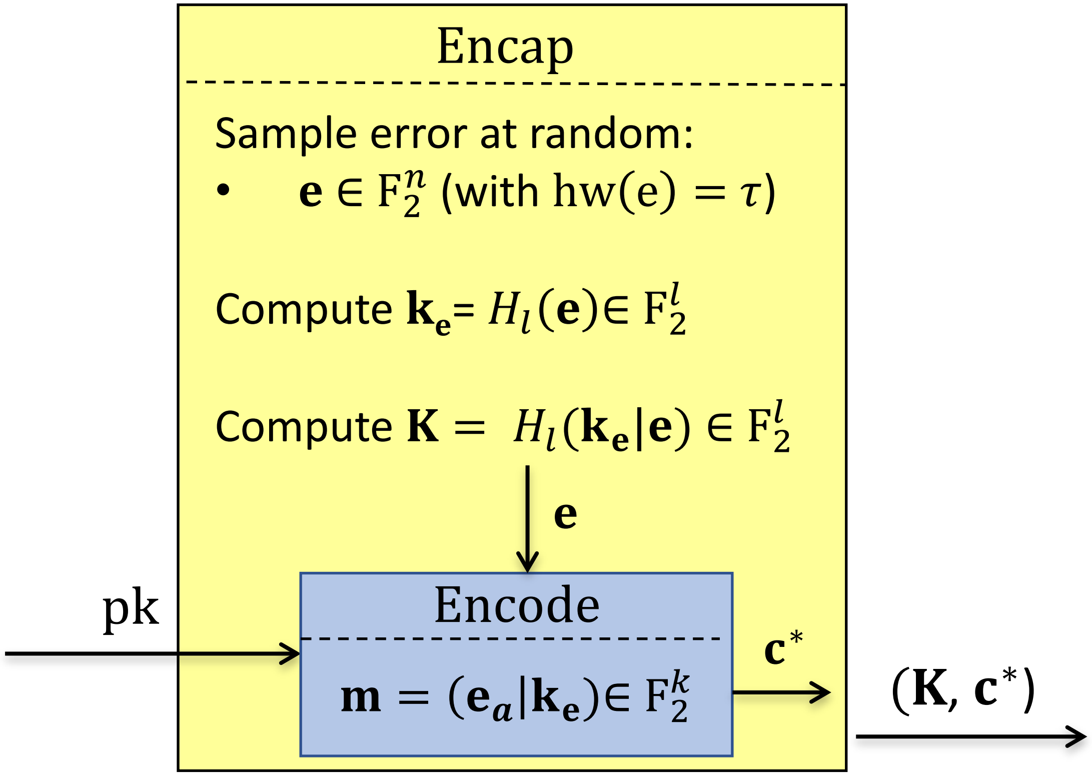
errors in NTS-KEM ≈ ***messages*** in FO-transform

EOW-security [ACP+19] of NTS⁻ $\overset{\text{QROM}}{\Longrightarrow}$ IND-CCA-security of NTS-KEM

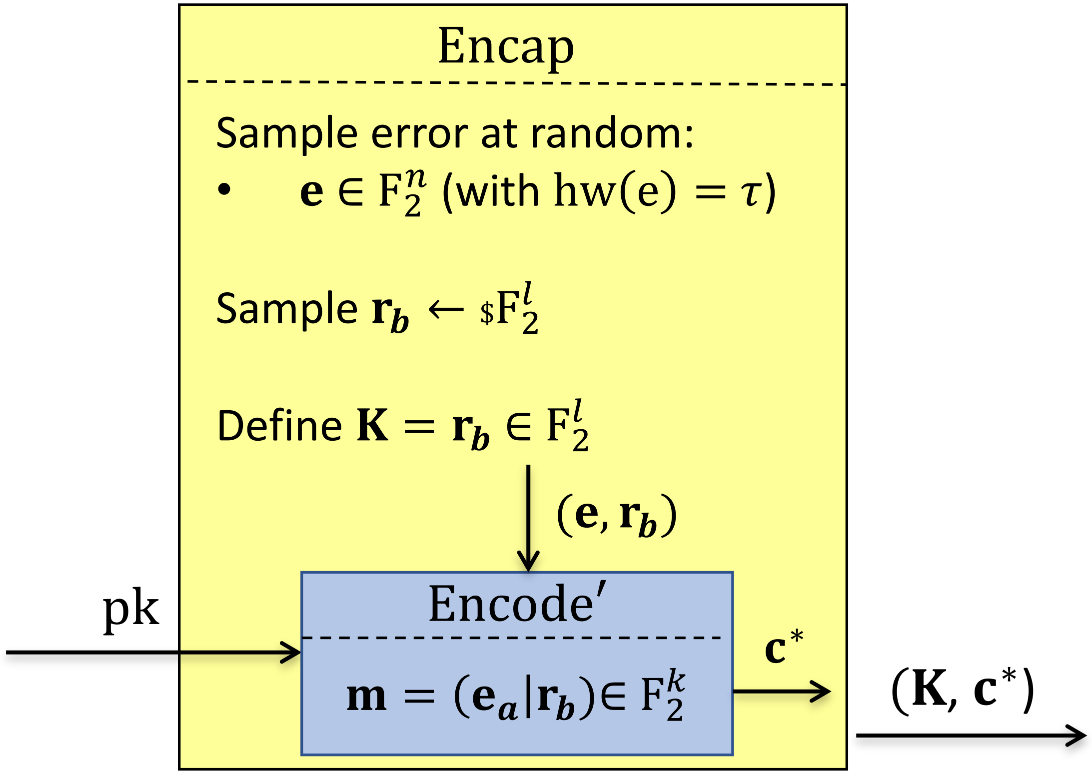
[ACP+19]: Albrecht, M., et. al., *NTS-KEM: NIST PQC Second Round Submission*, 2019. [https://nts-kem.io/]

NTS⁻ : Decoupling Random Oracles

- **NTS-KEM Encapsulation:**

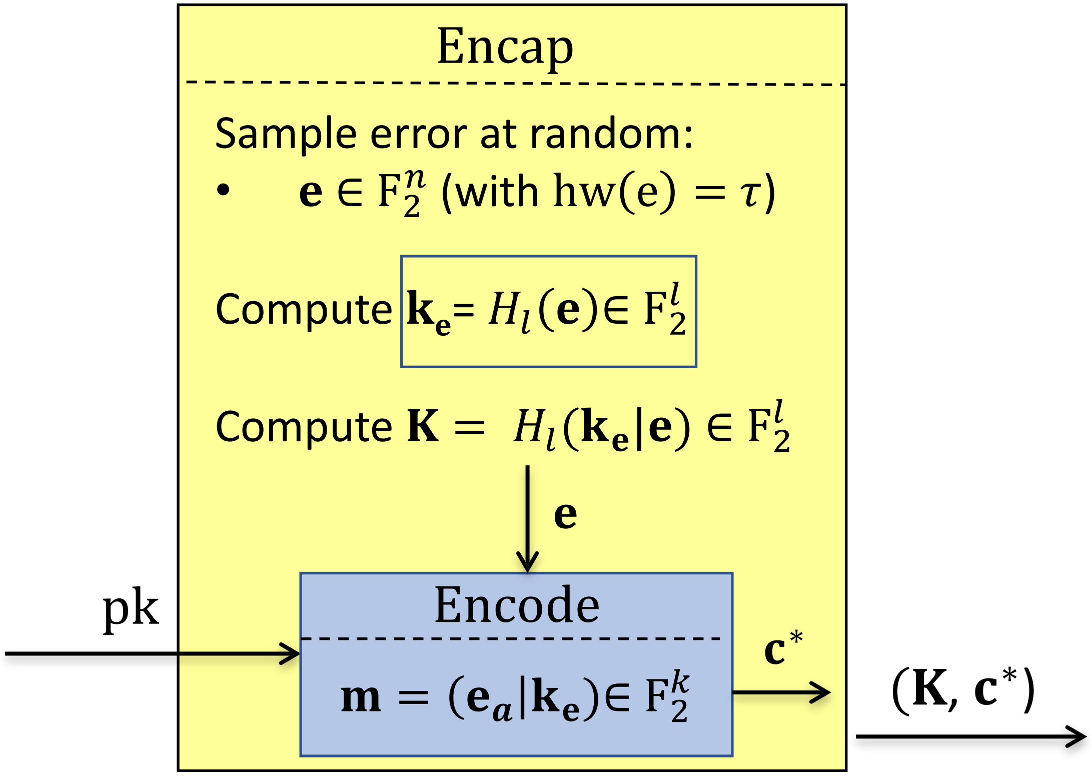


- **NTS⁻ Encapsulation:**

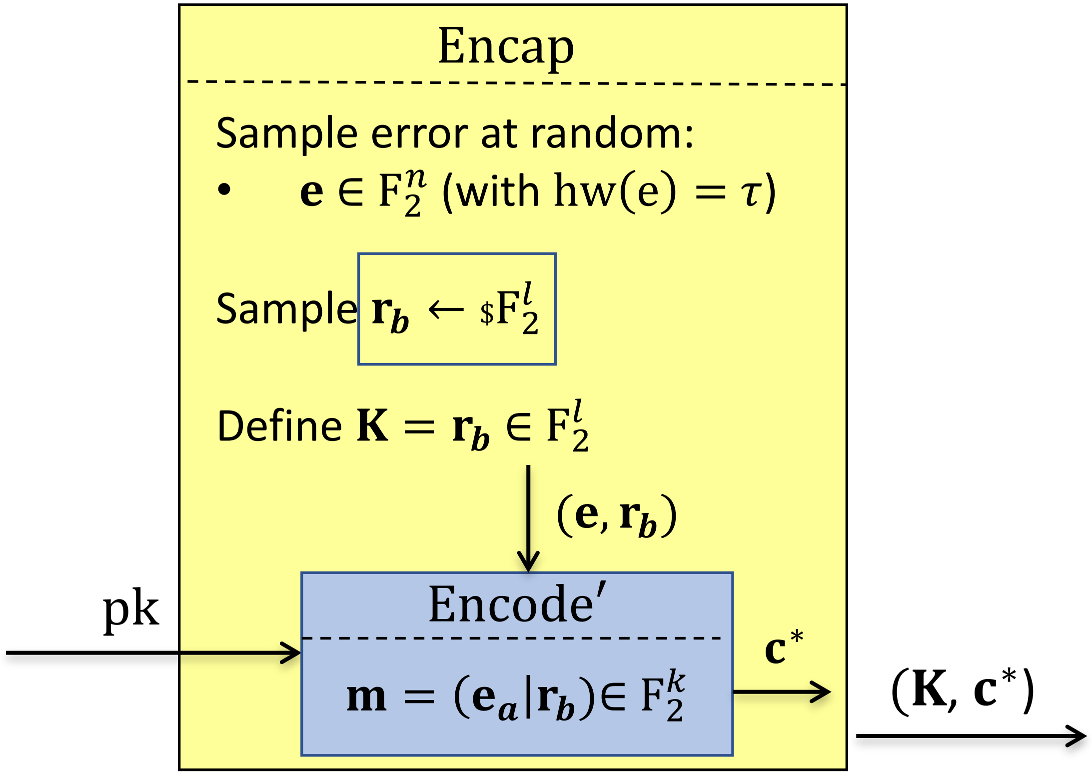


NTS^{-} : Decoupling Random Oracles

- **NTS-KEM Encapsulation:**

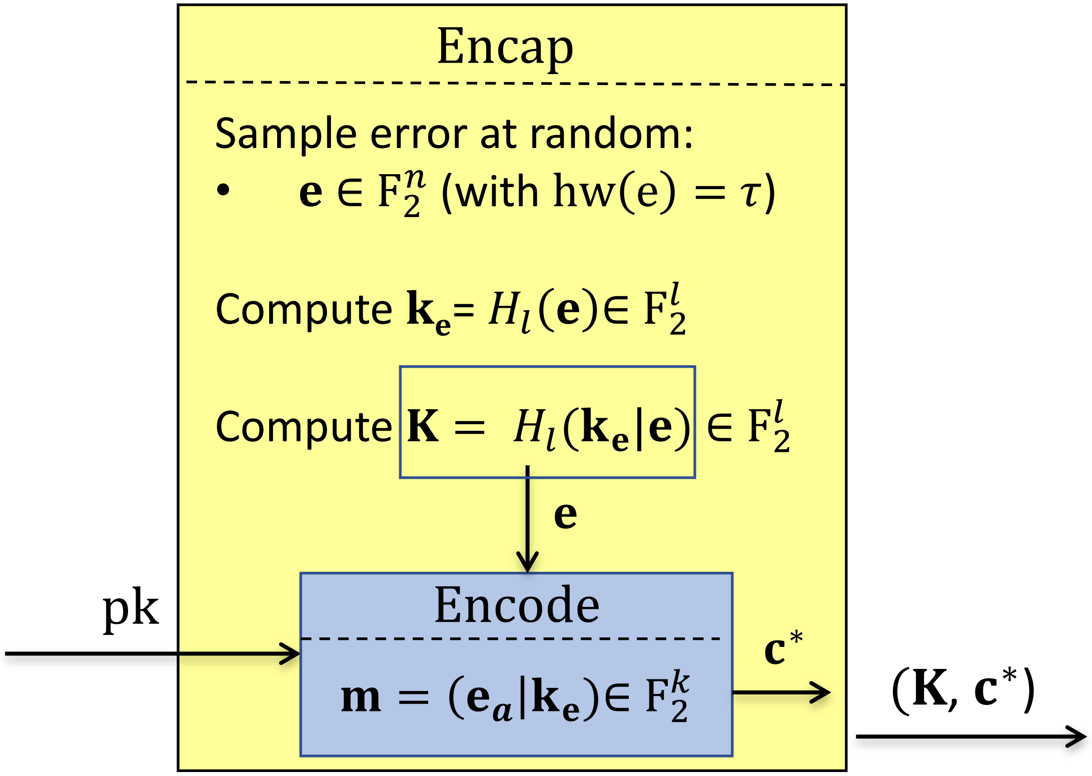


- **NTS^{-} Encapsulation:**

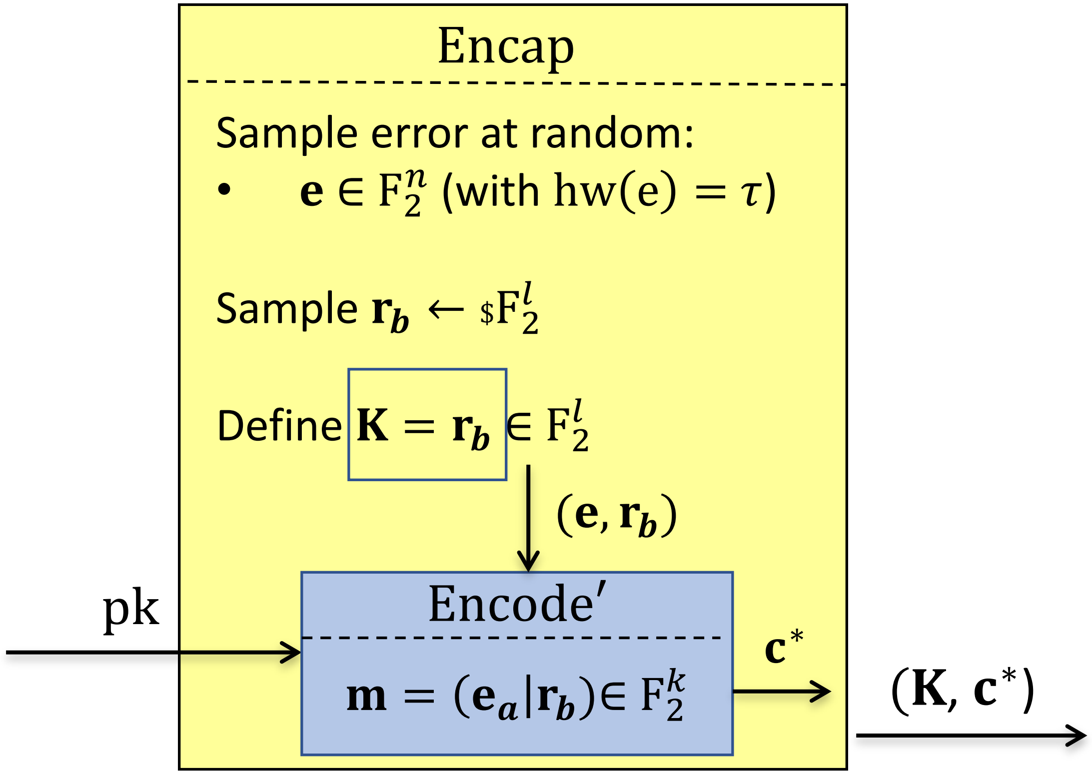


NTS⁻ : Decoupling Random Oracles

- **NTS-KEM Encapsulation:**

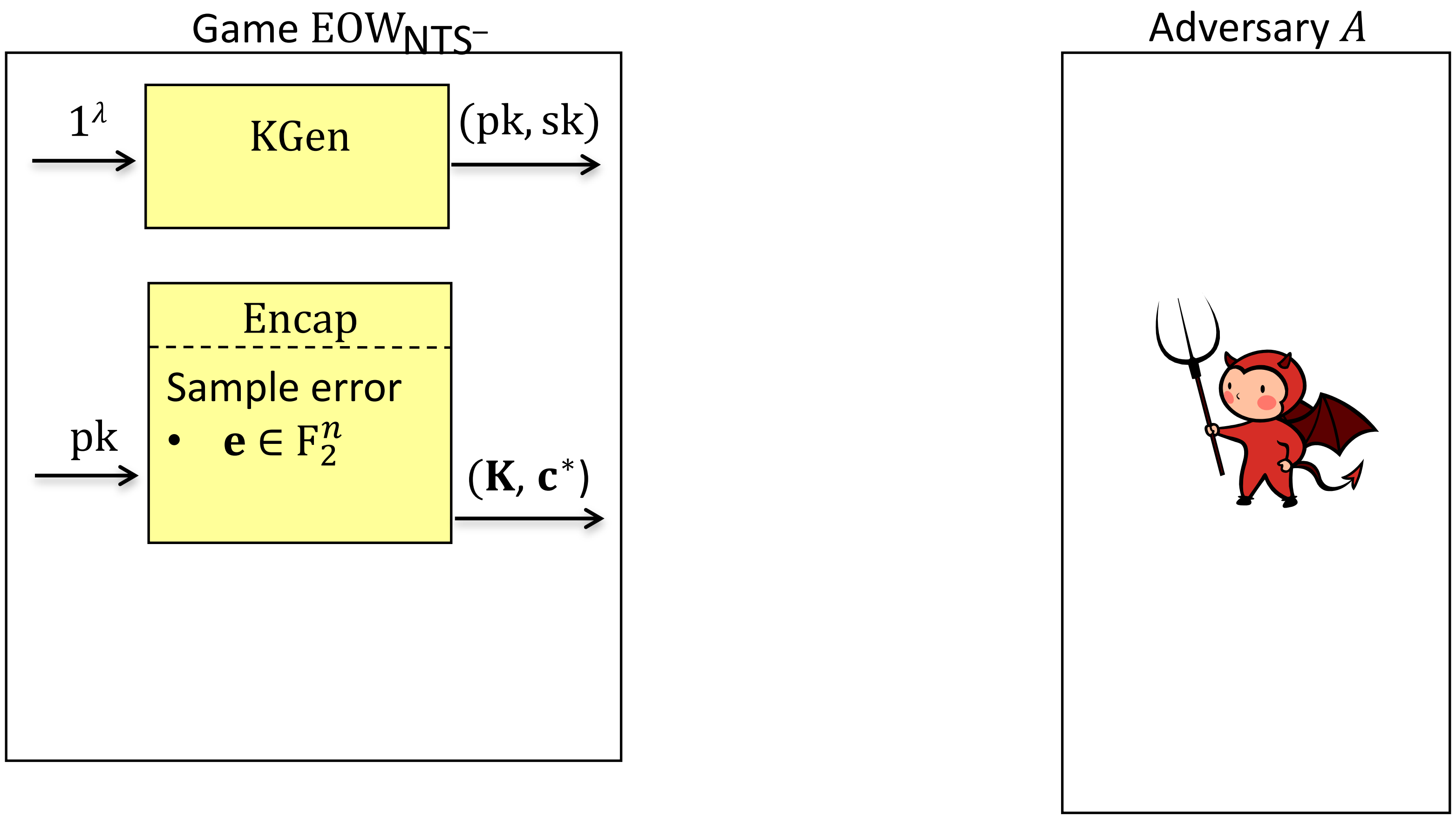


- **NTS⁻ Encapsulation:**



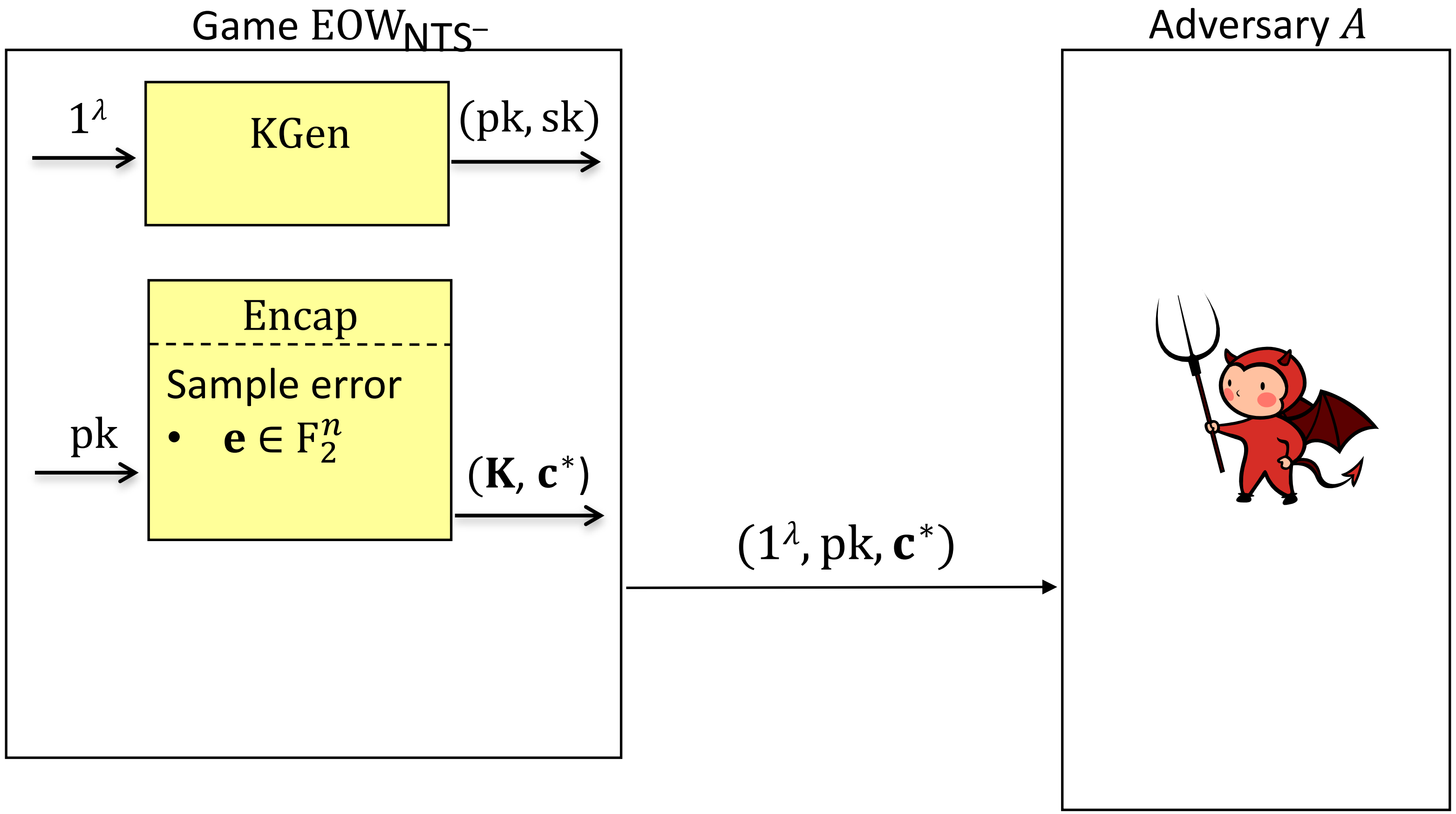
NTS^{-} : Decoupling Random Oracles

- **Error One-wayness:** NTS^{-} is (t, ε)-EOW-secure if there is no adversary that runs in time at most t and wins the below game with probability at least ε.



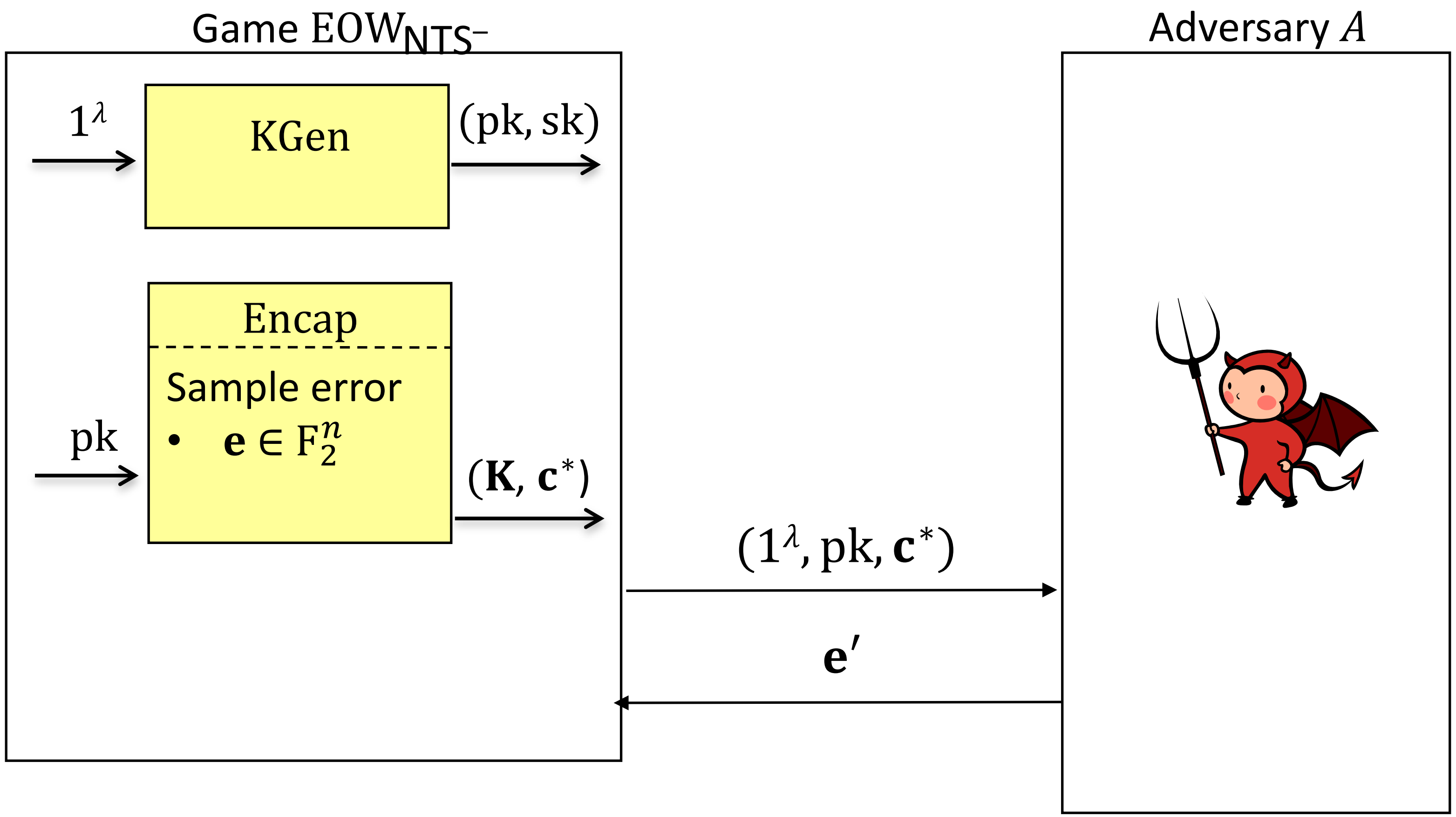
NTS⁻ : Decoupling Random Oracles

- **Error One-wayness:** NTS⁻ is (t, ε)-EOW-secure if there is no adversary that runs in time at most t and wins the below game with probability at least ε.



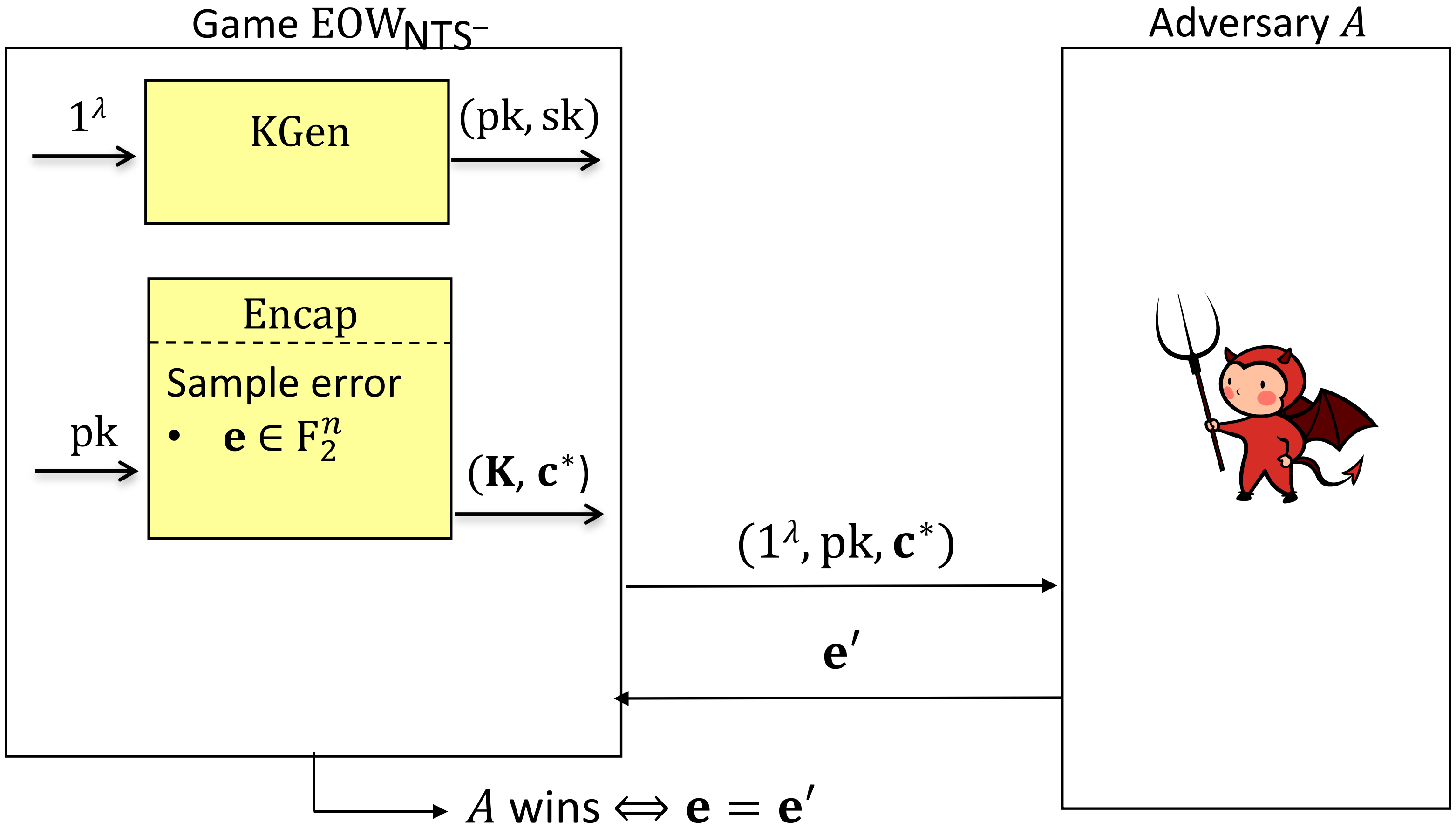
NTS^{-} : Decoupling Random Oracles

- **Error One-wayness:** NTS^{-} is (t, ε)-EOW-secure if there is no adversary that runs in time at most t and wins the below game with probability at least ε.

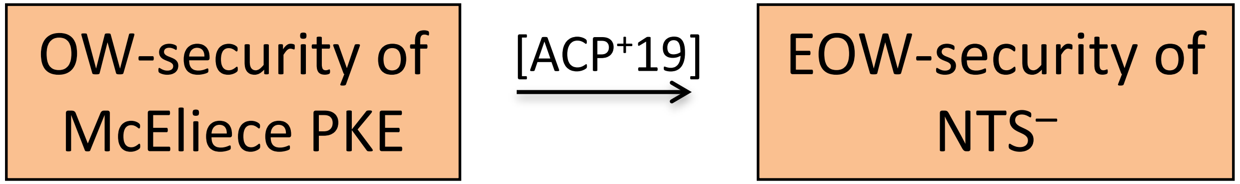


NTS^- : Decoupling Random Oracles

- **Error One-wayness:** NTS^- is (t, ε)-EOW-secure if there is no adversary that runs in time at most t and wins the below game with probability at least ε.

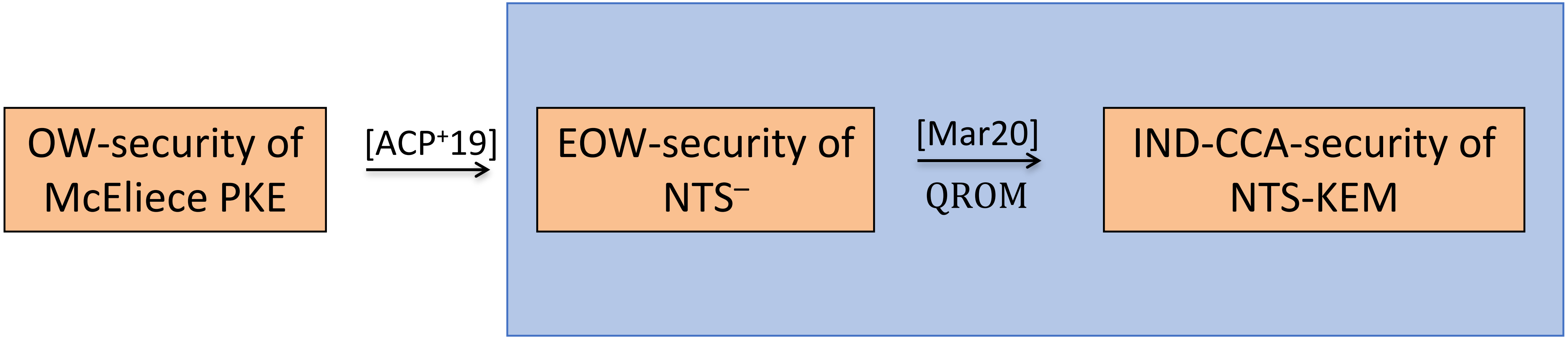


Proving Security of NTS-KEM in QROM



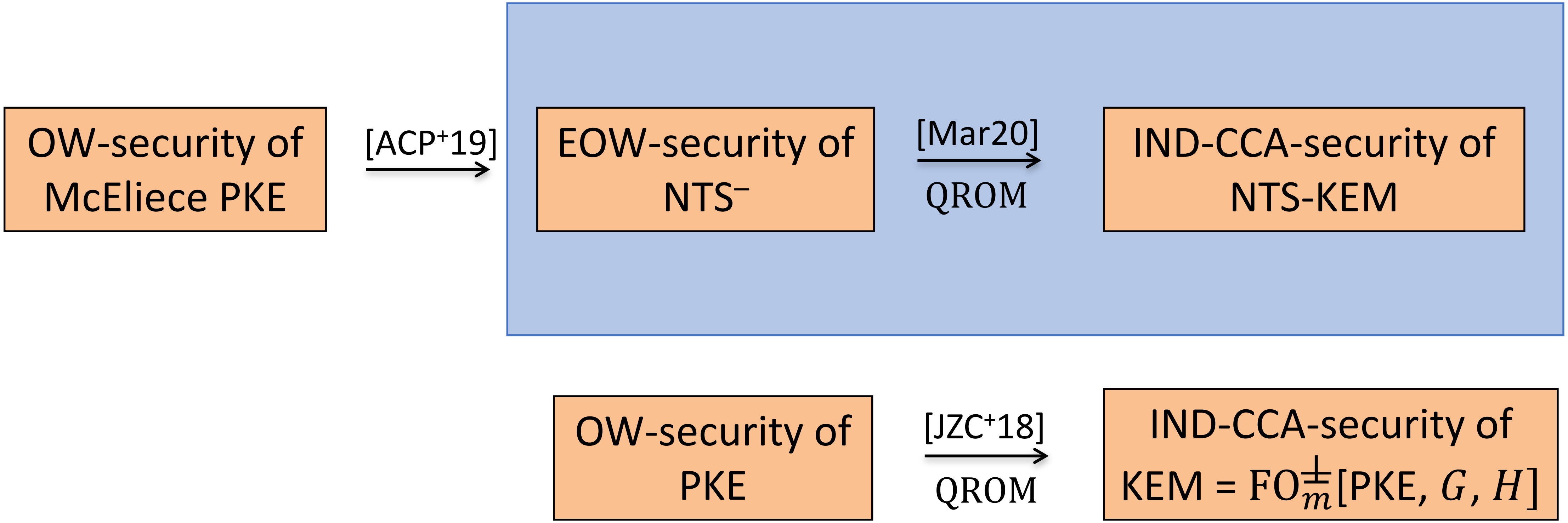
[ACP+19]: Albrecht, M., et. al., *NTS-KEM: NIST PQC Second Round Submission*, 2019. [https://nts-kem.io/]

Proving Security of NTS-KEM in QROM



[ACP+19]: Albrecht, M., et. al., *NTS-KEM: NIST PQC Second Round Submission*, 2019. [https://nts-kem.io/]
[Mar20]: Maram, V., On the Security of NTS-KEM in the QROM, 2020 [https://eprint.iacr.org/2020/150.pdf]

Proving Security of NTS-KEM in QROM



[ACP+19]: Albrecht, M., et. al., *NTS-KEM: NIST PQC Second Round Submission*, 2019. [https://nts-kem.io/]
[Mar20]: Maram, V., On the Security of NTS-KEM in the QROM, 2020 [https://eprint.iacr.org/2020/150.pdf]
[JZC+18]: Jiang, H., et. al., *IND-CCA-Secure KEM in the Quantum Random Oracle Model, Revisited*, CRYPTO 2018.

Conclusion

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission

- (Tightly) IND-CCA secure in the ROM **(bug!)**
- IND-CCA security in the QROM?

Conclusion

NTS-KEM (merged with **Classic McEliece**)

A code-based key-encapsulation mechanism submitted to NIST Post-Quantum Cryptography Standardization Process

Submitters:

Martin Albrecht, Royal Holloway University of London
Carlos Cid, Royal Holloway University of London
Kenneth G. Paterson, Royal Holloway University of London and ETH Zürich
CJ Tjhai, Post-Quantum Ltd
Martin Tomlinson, Post-Quantum Ltd

Email: authors@nts-kem.io

- (Tightly) IND-CCA secure in the ROM
- IND-CCA secure in the QROM (quadratic loss)

Document

The main document submitted to NIST:

Updated second round submission (2019-11-29) (Changes), Second round submission (Changes), First round submission

Source Code

C source code for reference implementation, optimized generic 64-bit, SSE2/SSE4.1 and AVX2 implementations

KATs

Know-Answer-Test vectors and intermediate values:

Updated second round submission (2019-11-29), Second round submission, First round submission