v1

Overview of side-channel and fault injection attacks on ML-KEM (CRYSTALS-KYBER) implementations

Simon Pontié simon.pontie@cea.fr

Outline

I. Side Channel Analysis and Fault Injection

II. ML-KEM

III. 3 examples of Fault Injection attack on ML-KEM implementation

IV. 2 examples of Side-Channel Analysis attack on ML-KEM implementation

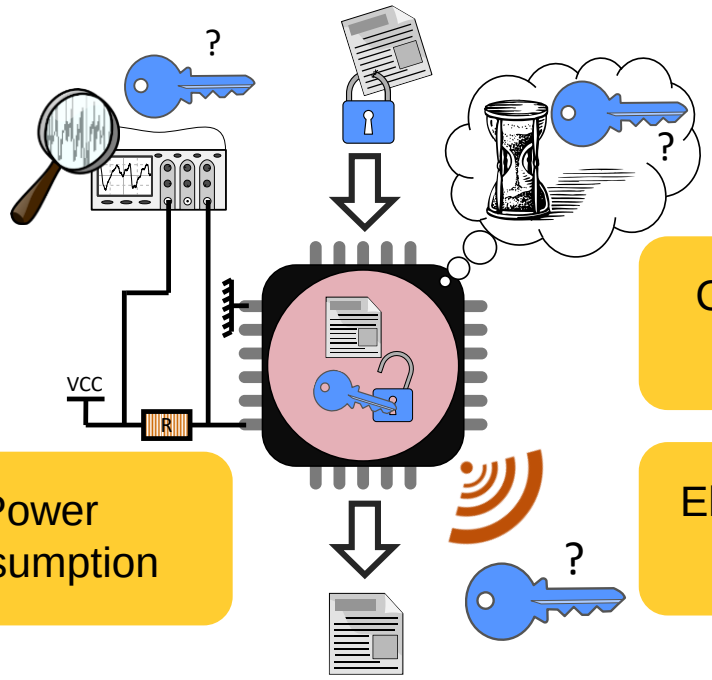
V. Conclusion

Existing survey in the literature: *Ravi, Prasanna, et al. "Side-channel and fault-injection attacks over lattice-based post-quantum schemes (Kyber, Dilithium): Survey and new results." ACM Transactions on Embedded Computing Systems 23.2 (2024): 1-54.*

Side Channel Analysis and Fault Injection

A cryptographic implementation can leak information about a secret even if it is an implementation of a secure cryptographic algorithm.

Side-channel analysis (SCA) attacks

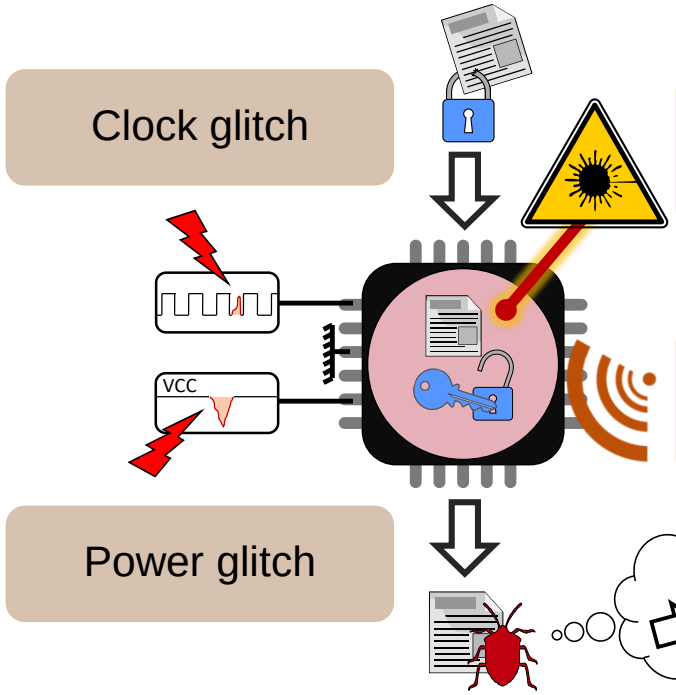


Computational time

Power consumption

Electromagnetic emanations

Fault injection (FI) attacks



Clock glitch

Laser illumination

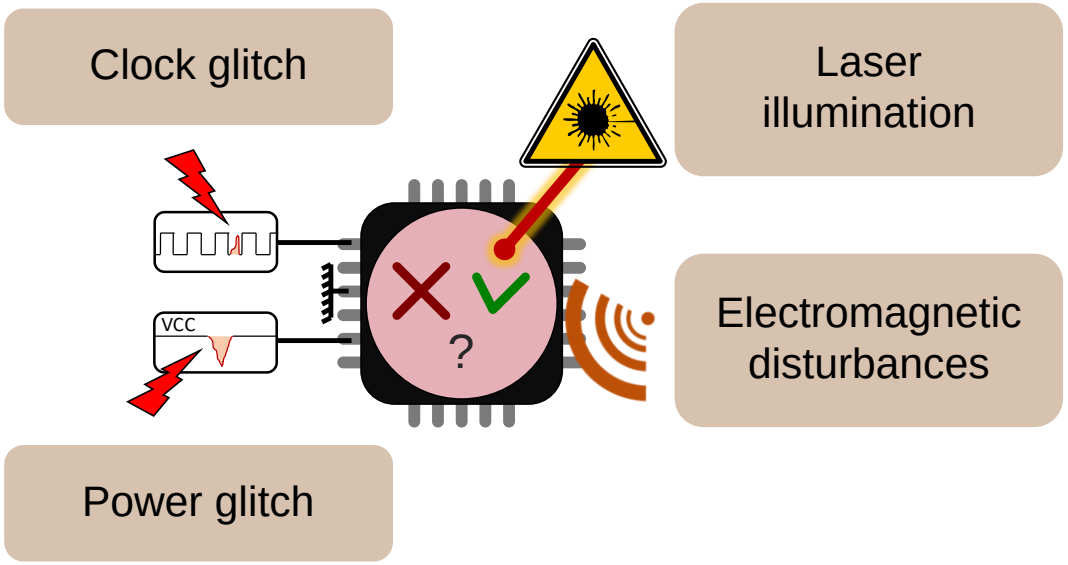
Electromagnetic disturbances

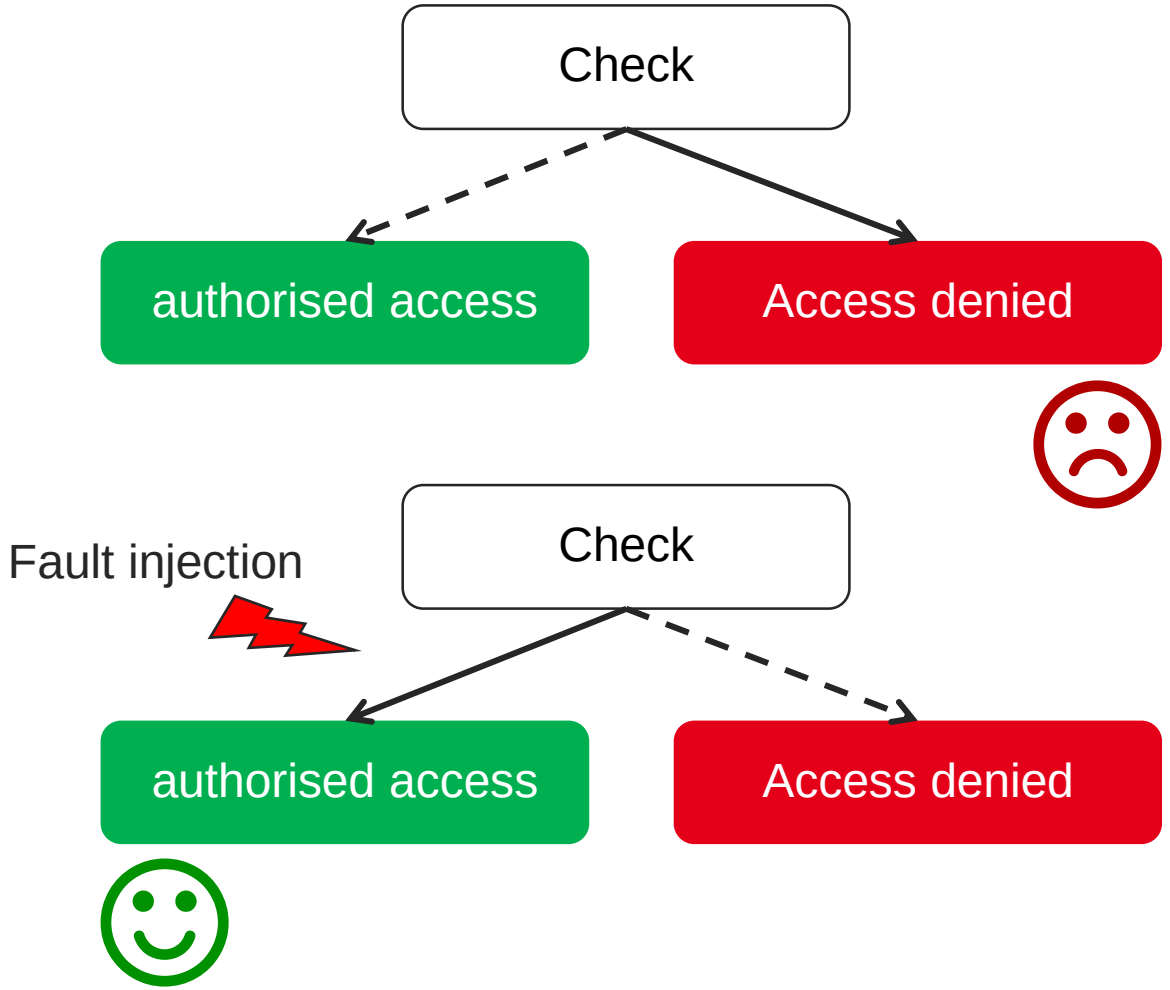
Power glitch

Side Channel Analysis and Fault Injection

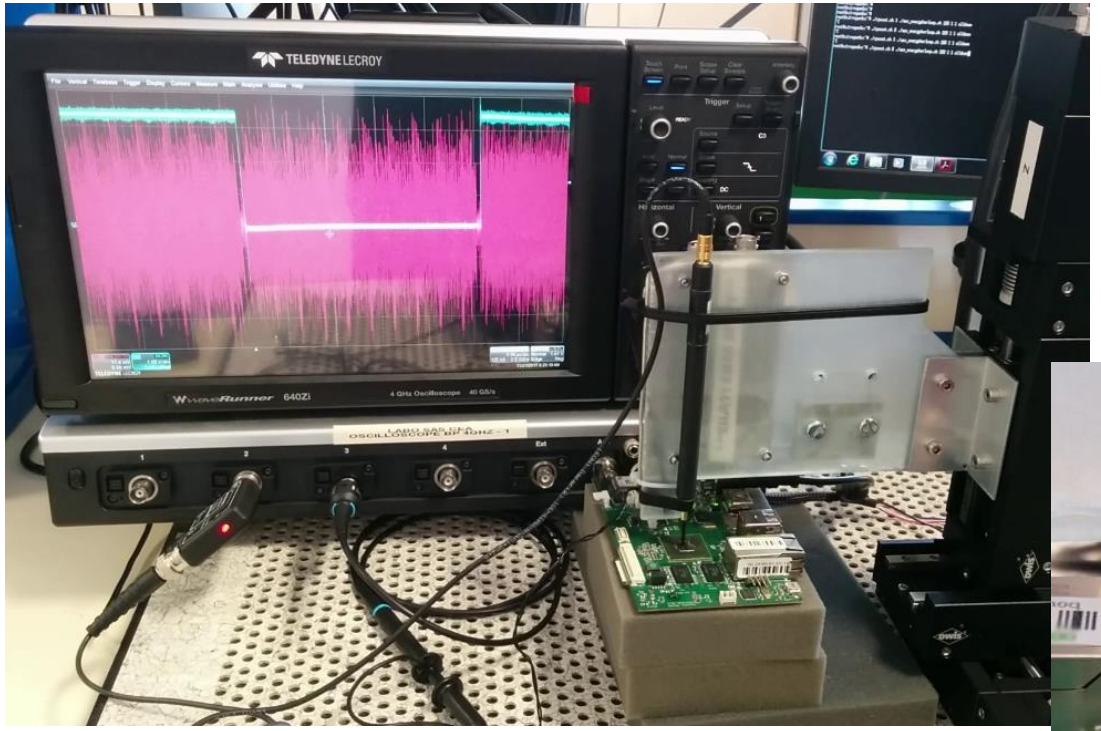
Fault injection attacks can also be used to bypass a security mechanism

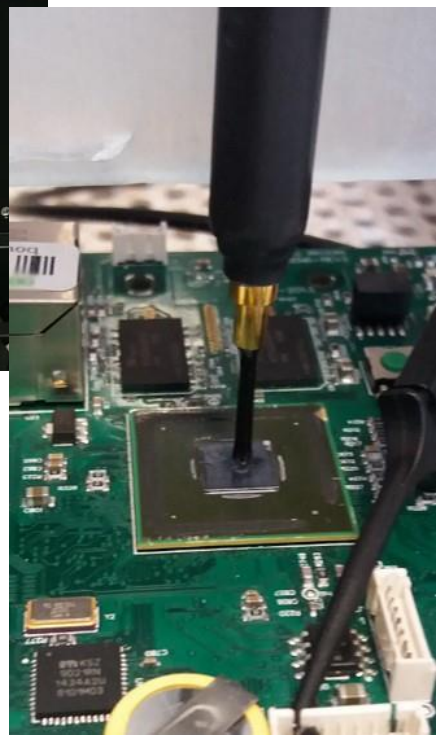
Fault injection (FI) attacks

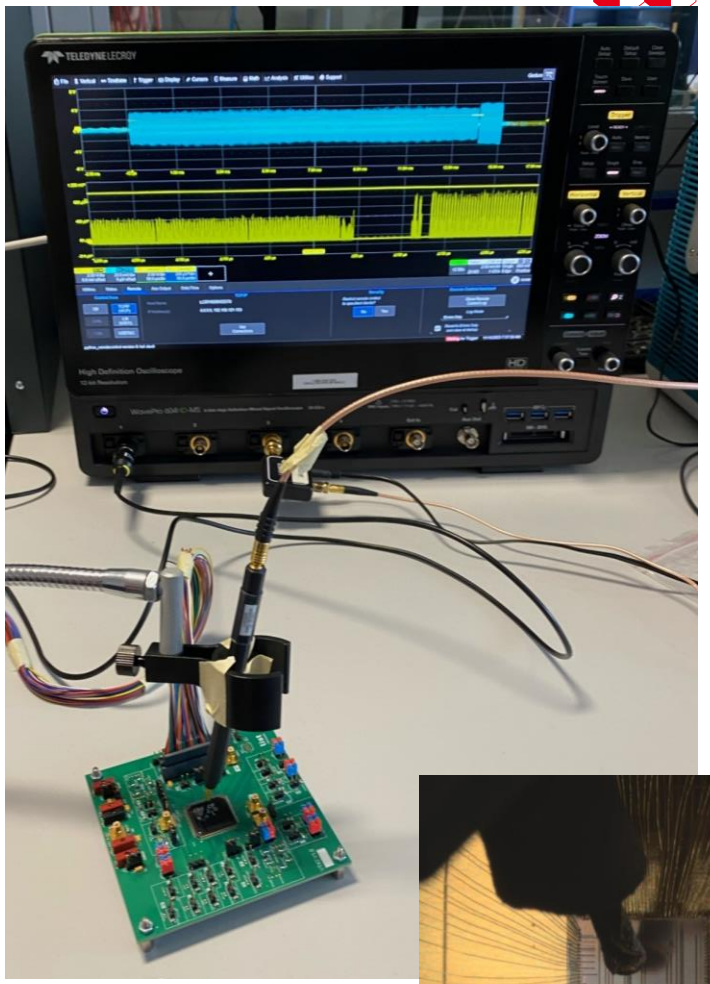


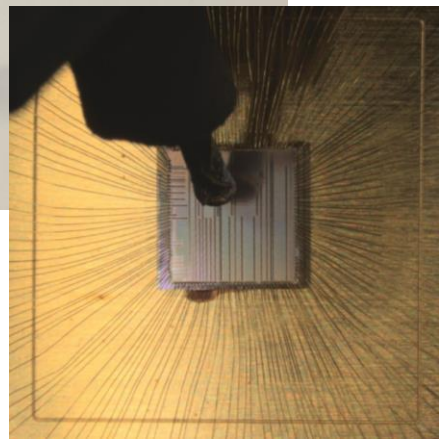


Side Channel Analysis

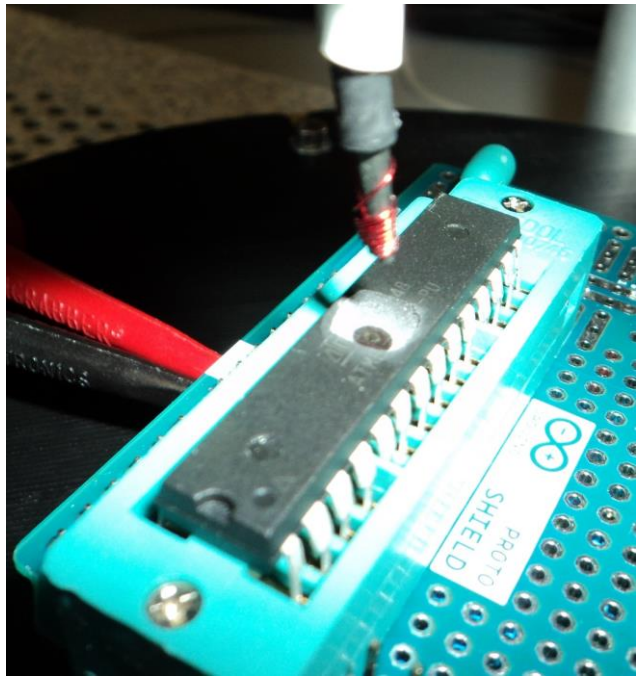


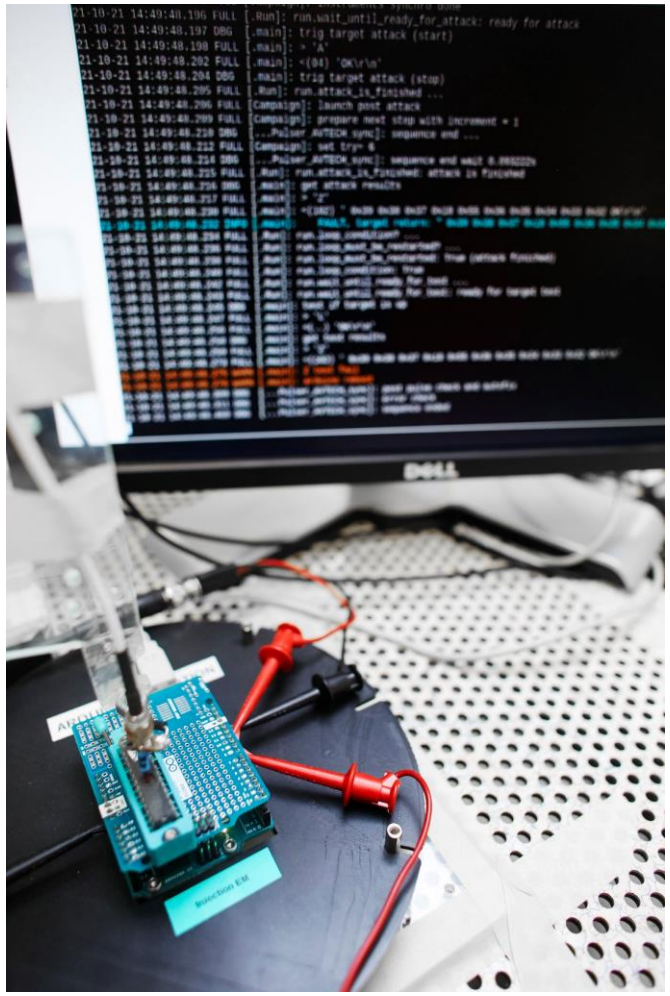


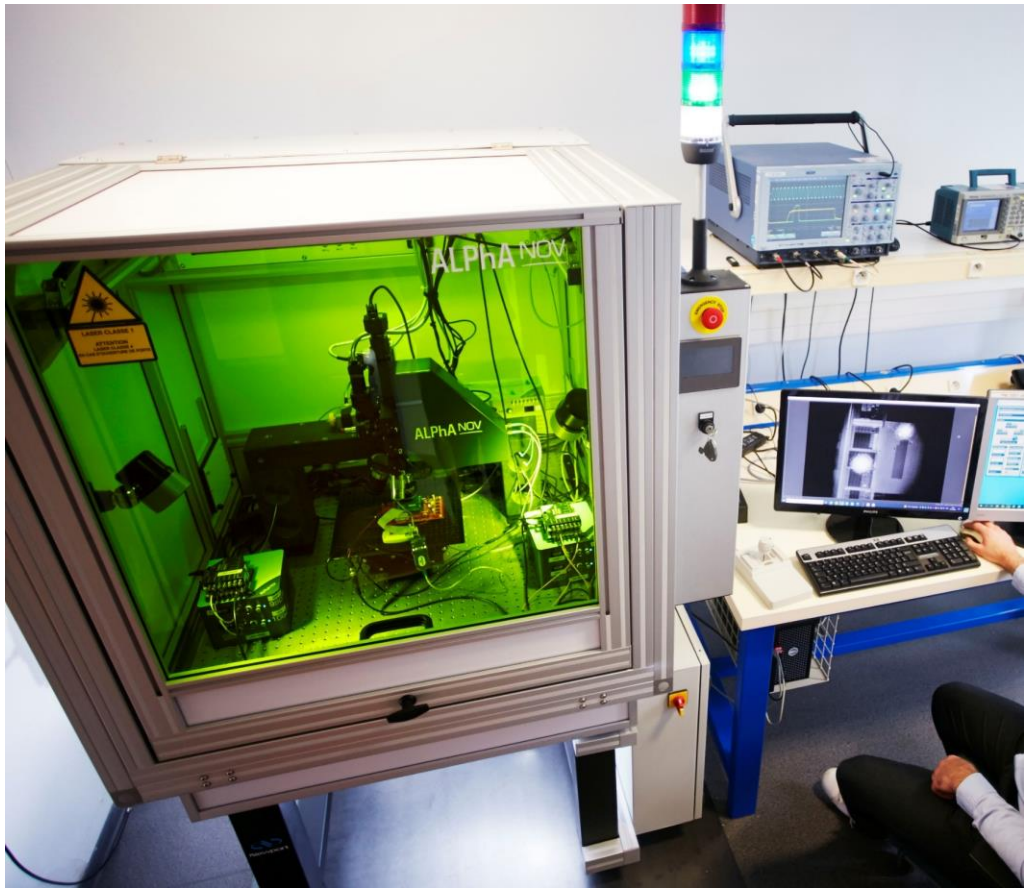




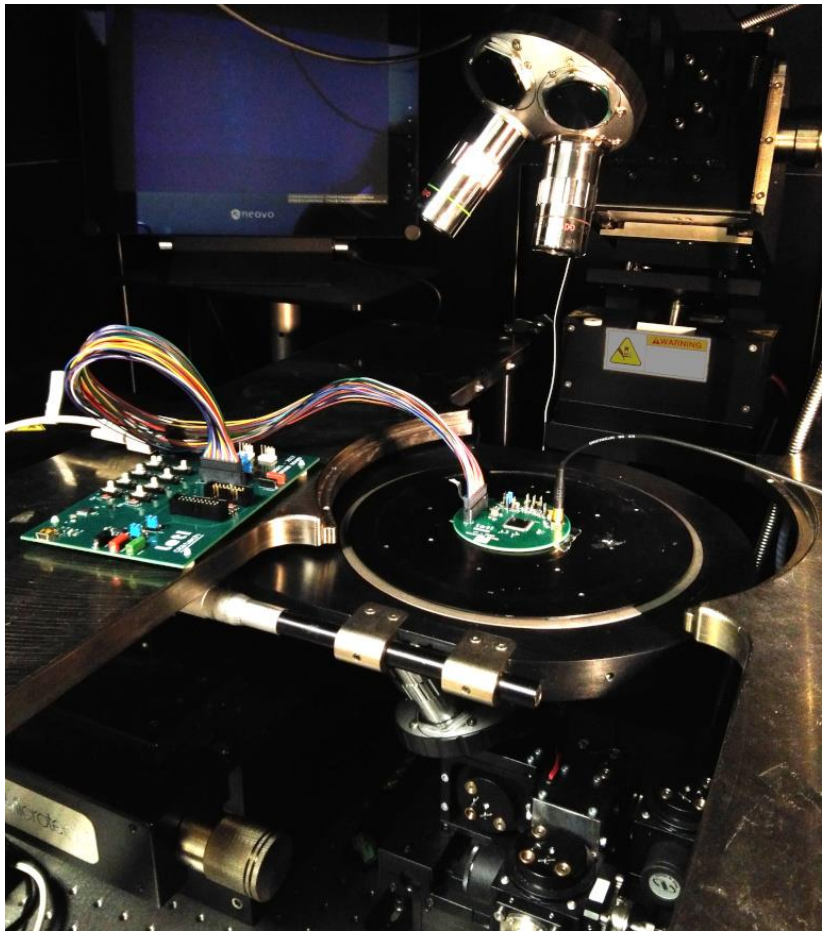
Fault Injection

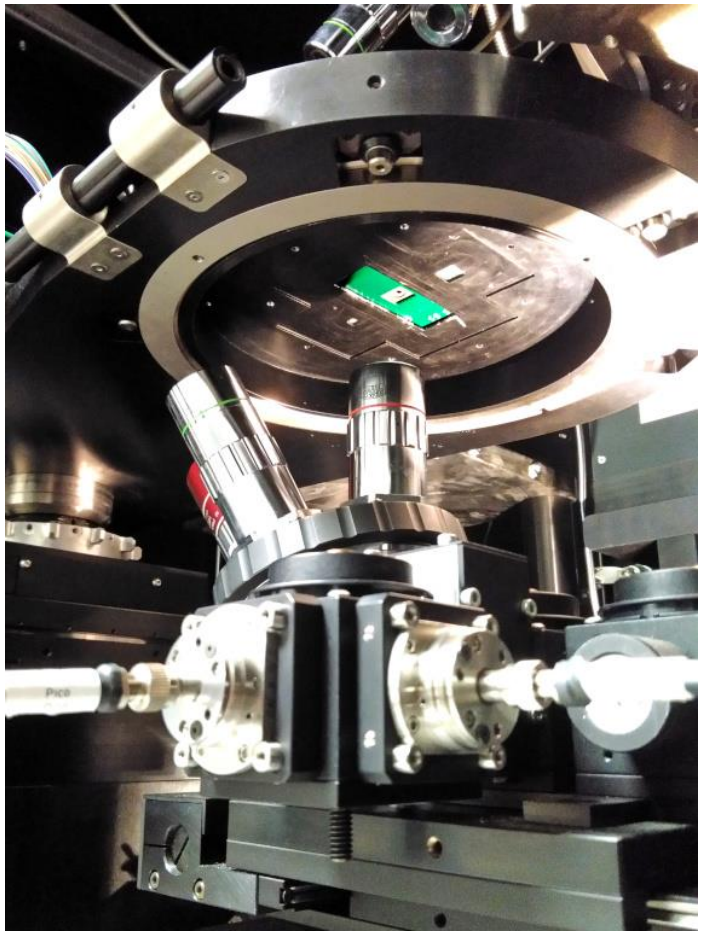






Fault Injection





ML-KEM (close to Crystal-Kyber)

ML-KEM-512	ML-KEM-768	ML-KEM-1024
Security level $= 1$	Security level $= 3$	Security level $= 5$
$q = 3329$	$q = 3329$	$q = 3329$
$n = 256$	$n = 256$	$n = 256$
$k = 2$	$k = 3$	$k = 4$
$\eta_1 = 3$	$\eta_1 = 2$	$\eta_1 = 2$
$\eta_2 = 2$	$\eta_2 = 2$	$\eta_2 = 2$
$d_u = 10$	$d_u = 10$	$d_u = 11$
$d_v = 4$	$d_v = 4$	$d_v = 5$

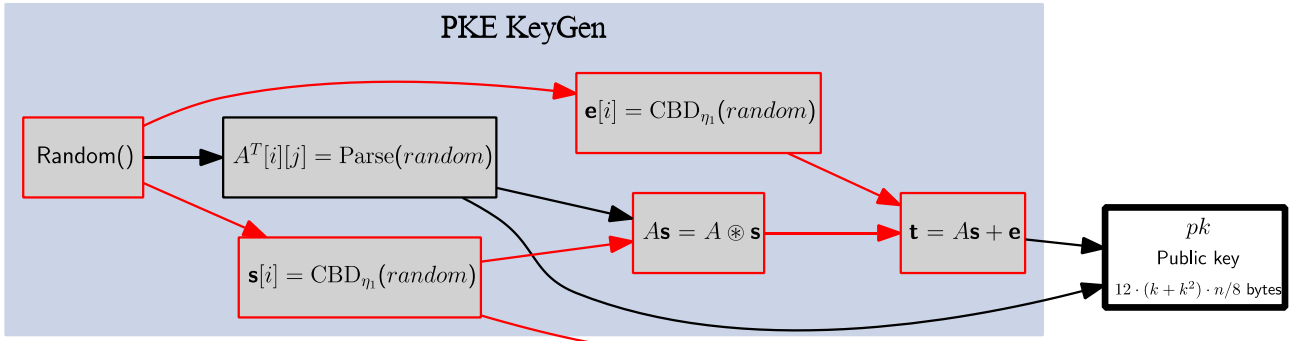
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ (integer ring)
$R_q = \mathbb{Z}_q[X]/(X^n + 1)$ (polynomial ring, n-degree polynomial)
R_q^k (a k-dimensional vector of polynomials with polynomials in R_q)
$R_q^{k \times k}$ (module of dimension $k \times k$, k-by-k matrix of polynomials with polynomials in R_q)
$\zeta = 17$ (primitive n-th root of unity modulo q)

FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard

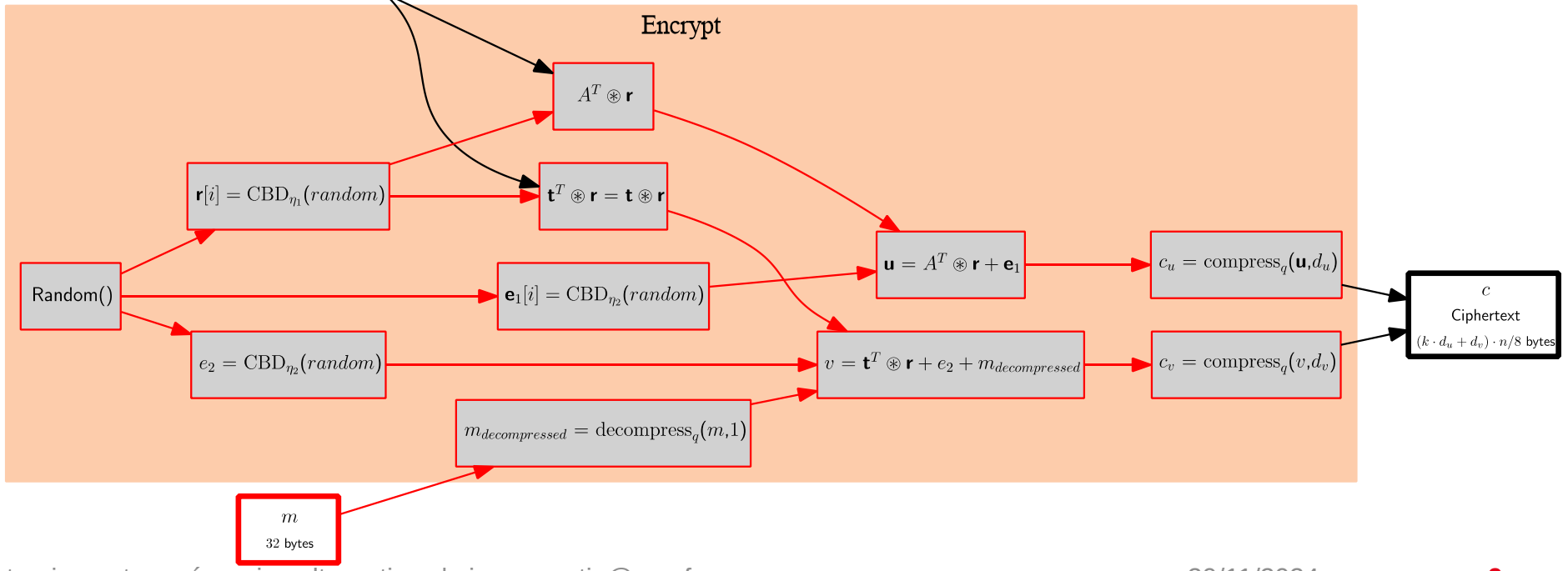
https://doi.org/10.6028/NIST.FIPS.203, Published August 13, 2024

Simplified PKE in ML-KEM

- Key generation

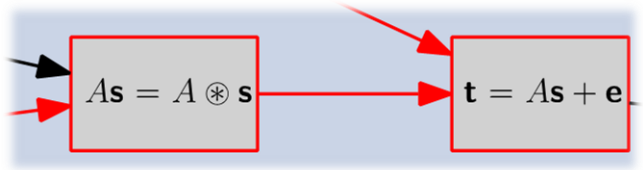


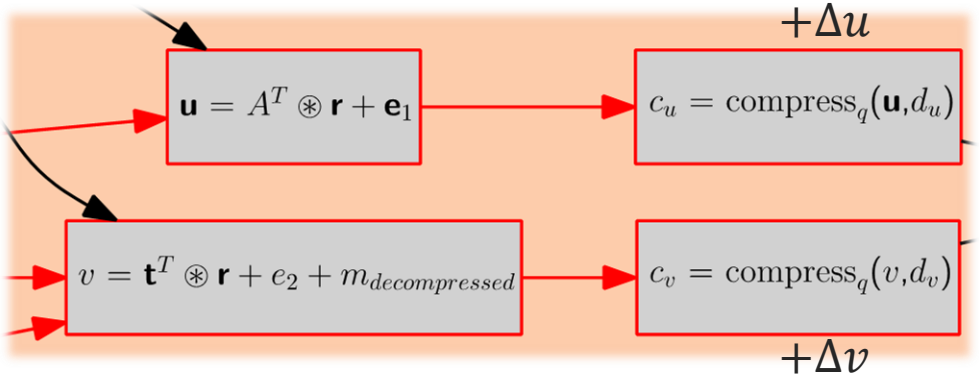
- Message Encryption

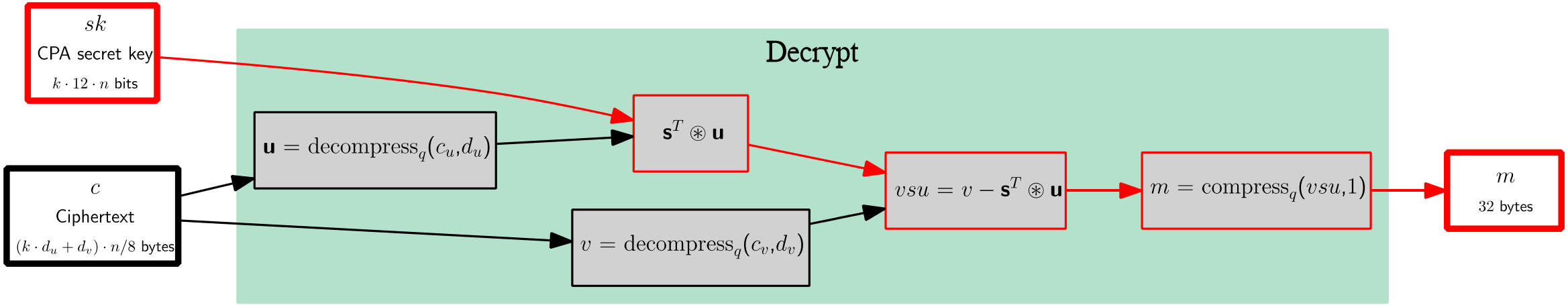


Simplified PKE in ML-KEM

- Message decryption



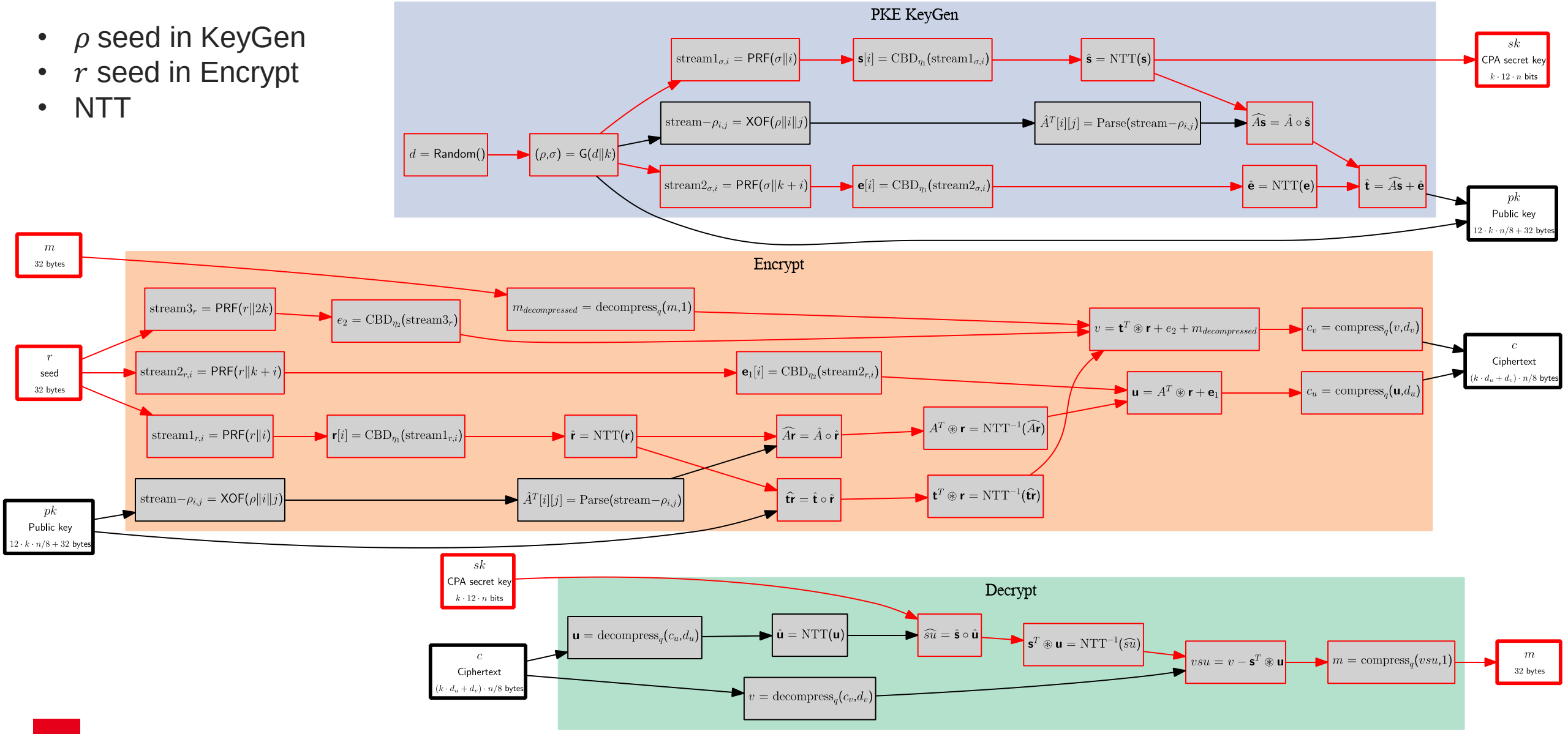




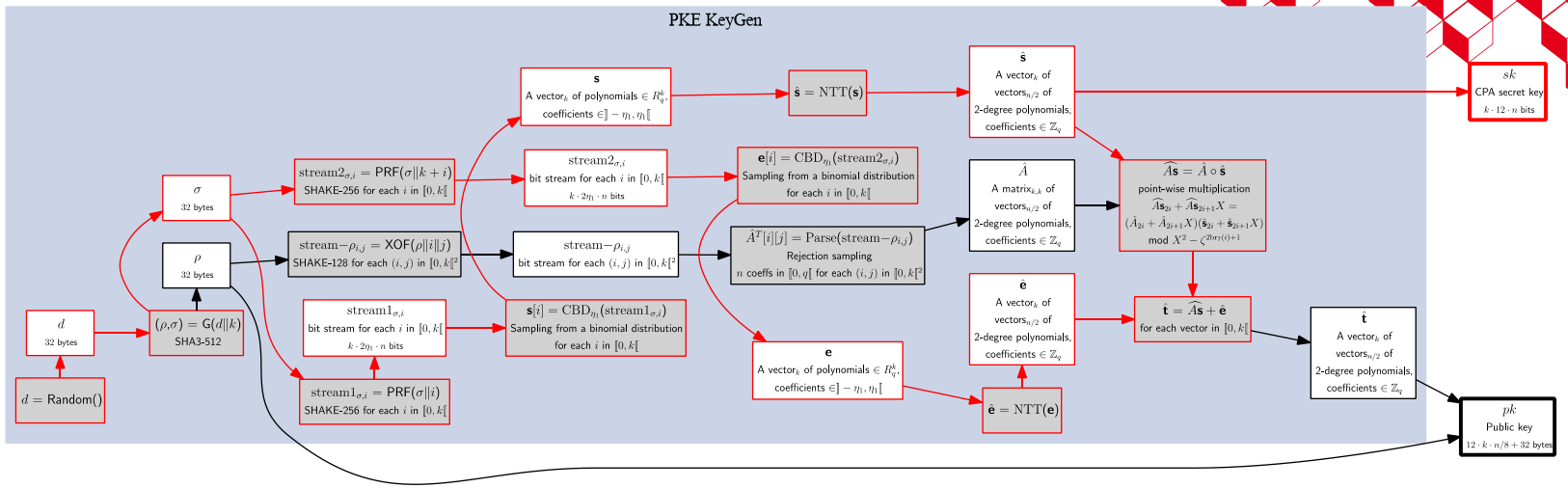
$$
\begin{aligned}
vsu &= v - s^T \circledast u \\
&= (s^T \circledast A^T \circledast r + e^T \circledast r + e_2 + m + \Delta v) - (s^T \circledast A^T \circledast r + s^T \circledast e_1 + s^T \circledast \Delta u) \\
&= m + e^T \circledast r + e_2 + \Delta v - s^T \circledast e_1 - s^T \circledast \Delta u \\
&\simeq m
\end{aligned}
$$

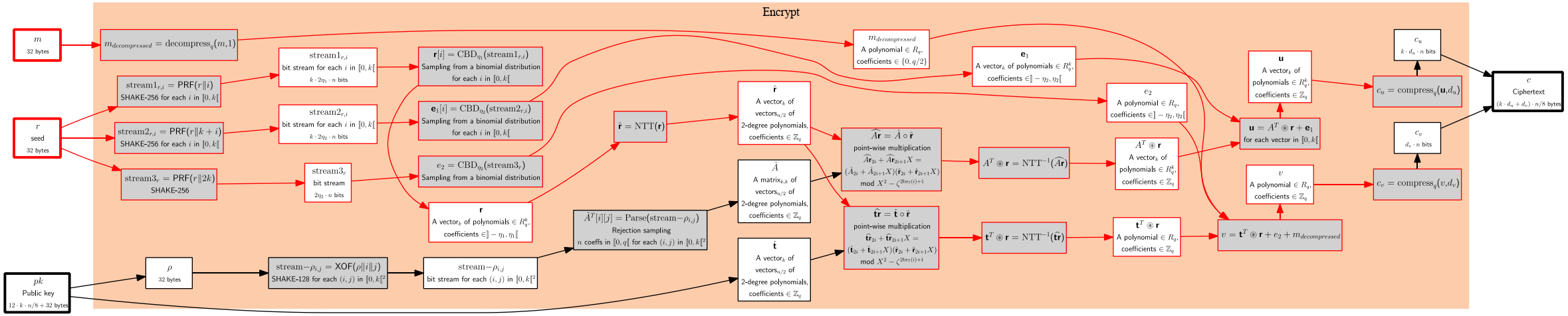
PKE in ML-KEM

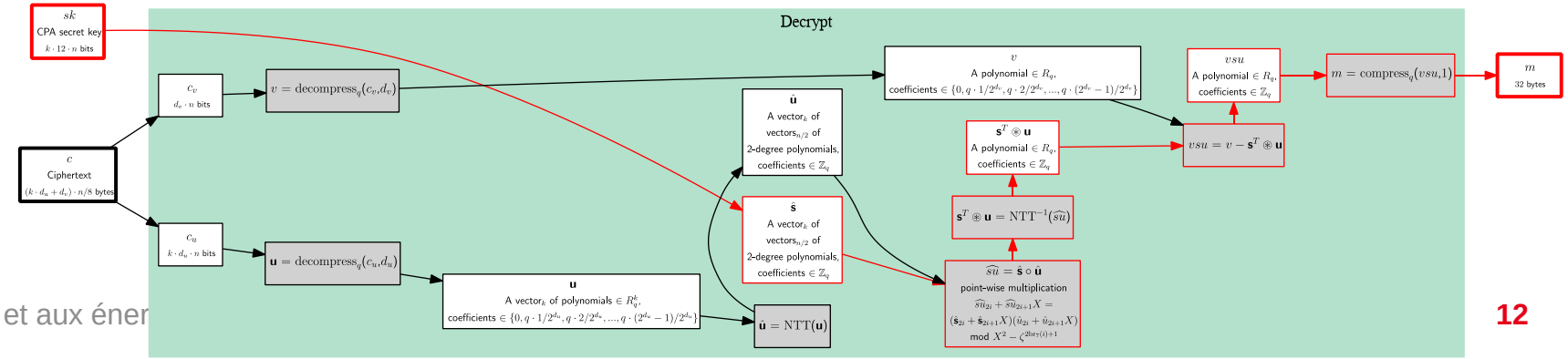
- ρ seed in KeyGen
- r seed in Encrypt
- NTT



PKE in ML-KEM

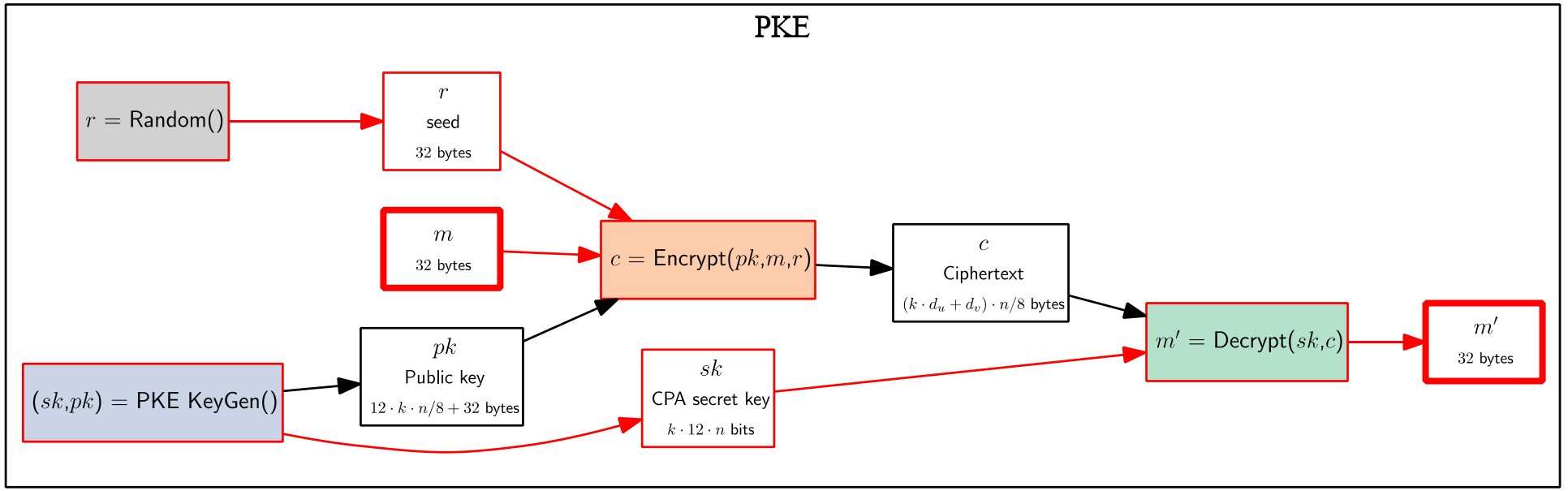

PKE KeyGen
d
32 bytes
d = Random()
(ρ,σ) = G(d||k)
SHA3-512
σ
32 bytes
ρ
32 bytes
stream2σ,i = PRF(σ||k + i)
SHAKE-256 for each i in [0, k[
stream−ρi,j = XOF(ρ||i||j)
SHAKE-128 for each (i, j) in [0, k[²
stream1σ,i
bit stream for each i in [0, k[
k · 2η1 · n bits
stream1σ,i = PRF(σ||i)
SHAKE-256 for each i in [0, k[
s
A vectork of polynomials ∈ Rqk
coefficients ∈] − η1, η1[
s[i] = CBDη1(stream1σ,i)
Sampling from a binomial distribution for each i in [0, k[
ŝ = NTT(s)
stream2σ,i
bit stream for each i in [0, k[
k · 2η1 · n bits
e[i] = CBDη1(stream2σ,i)
Sampling from a binomial distribution for each i in [0, k[
stream−ρi,j
bit stream for each (i, j) in [0, k[²
Â[i][j] = Parse(stream−ρi,j)
Rejection sampling
n coeffs in [0, q[for each (i, j) in [0, k[²
e
A vectork of polynomials ∈ Rqk
coefficients ∈] − η1, η1[
ê = NTT(e)
ŝ
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
Â
A matrixk,k of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
Âs = Â ∘ ŝ
point-wise multiplication
Âs2i + Âs2i+1X = (Â2i + Â2i+1X)(ŝ2i + ŝ2i+1X) mod X² − ζ^(2br7(i)+1)
ê
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
t̂ = Âs + ê
for each vector in [0, k[
t̂
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
sk
CPA secret key
k · 12 · n bits
pk
Public key
12 · k · n/8 + 32 bytes


Encrypt
m
32 bytes
mdecompressed = decompressq(m,1)
mdecompressed
A polynomial ∈ Rq, coefficients ∈ {0, q/2}
r
seed
32 bytes
stream1r,i = PRF(r||i)
SHAKE-256 for each i in [0, k[
stream1r,i
bit stream for each i in [0, k[
k · 2η1 · n bits
r[i] = CBDη1(stream1r,i)
Sampling from a binomial distribution for each i in [0, k[
stream2r,i = PRF(r||k + i)
SHAKE-256 for each i in [0, k[
stream2r,i
bit stream for each i in [0, k[
k · 2η2 · n bits
e1[i] = CBDη2(stream2r,i)
Sampling from a binomial distribution for each i in [0, k[
stream3r = PRF(r||2k)
SHAKE-256
stream3r
bit stream
2η2 · n bits
e2 = CBDη2(stream3r)
Sampling from a binomial distribution
r
A vectork of polynomials ∈ Rqk
coefficients ∈] − η1, η1[
r̂ = NTT(r)
e1
A vectork of polynomials ∈ Rqk
coefficients ∈] − η2, η2[
e2
A polynomial ∈ Rq
coefficients ∈] − η2, η2[
r̂
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
Âr = Â ∘ r̂
point-wise multiplication
Âr2i + Âr2i+1X = (Â2i + Â2i+1X)(r̂2i + r̂2i+1X) mod X² − ζ^(2br7(i)+1)
A^T ⊛ r = NTT^−1(Âr)
A^T ⊛ r
A vectork of polynomials ∈ Rqk
coefficients ∈ Zq
u
A vectork of polynomials ∈ Rqk
coefficients ∈ Zq
u = A^T ⊛ r + e1
for each vector in [0, k[
cu
k · du · n bits
cu = compressq(u,du)
Â
A matrixk,k of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
Â[i][j] = Parse(stream−ρi,j)
Rejection sampling
n coeffs in [0, q[for each (i, j) in [0, k[²
t̂
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
t̂r = t̂ ∘ r̂
point-wise multiplication
t̂r2i + t̂r2i+1X = (t̂2i + t̂2i+1X)(r̂2i + r̂2i+1X) mod X² − ζ^(2br7(i)+1)
t^T ⊛ r = NTT^−1(t̂r)
t^T ⊛ r
A polynomial ∈ Rq, coefficients ∈ Zq
v
A polynomial ∈ Rq, coefficients ∈ Zq
v = t^T ⊛ r + e2 + mdecompressed
cv
dv · n bits
cv = compressq(v,dv)
c
Ciphertext
(k · du + dv) · n/8 bytes
ρ
32 bytes
stream−ρi,j = XOF(ρ||i||j)
SHAKE-128 for each (i, j) in [0, k[²
stream−ρi,j
bit stream for each (i, j) in [0, k[²
pk
Public key
12 · k · n/8 + 32 bytes


Decrypt
sk
CPA secret key
k · 12 · n bits
c
Ciphertext
(k · du + dv) · n/8 bytes
cv
dv · n bits
v = decompressq(cv,dv)
cu
k · du · n bits
u = decompressq(cu,du)
u
A vectork of polynomials ∈ Rqk
coefficients ∈ {0, q · 1/2^du, q · 2/2^du, ..., q · (2^du − 1)/2^du}
v
A polynomial ∈ Rq,
coefficients ∈ {0, q · 1/2^dv, q · 2/2^dv, ..., q · (2^dv − 1)/2^dv}
û
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
ŝ
A vectork of vectorsn/2 of 2-degree polynomials, coefficients ∈ Zq
û = NTT(u)
s^T ⊛ u
A polynomial ∈ Rq, coefficients ∈ Zq
s^T ⊛ u = NTT^−1(ŝû)
ŝû = ŝ ∘ û
point-wise multiplication
ŝû2i + ŝû2i+1X = (ŝ2i + ŝ2i+1X)(û2i + û2i+1X) mod X² − ζ^(2br7(i)+1)
vsu
A polynomial ∈ Rq, coefficients ∈ Zq
vsu = v − s^T ⊛ u
m = compressq(vsu,1)
m
32 bytes

PKE in ML-KEM

ML-KEM-768
Security level $= 3$
$q = 3329$
$n = 256$
$k = 3$
$\eta_1 = 2$
$\eta_2 = 2$
$d_u = 10$
$d_v = 4$
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ (integer ring)
$R_q = \mathbb{Z}_q[X]/(X^n + 1)$ (polynomial ring, n-degree polynomial)
R_q^k (a k-dimensional vector of polynomials with polynomials in R_q)
$R_q^{k \times k}$ (module of dimension $k \times k$, k-by-k matrix of polynomials with polynomials in R_q)
$\zeta = 17$ (primitive n-th root of unity modulo q)



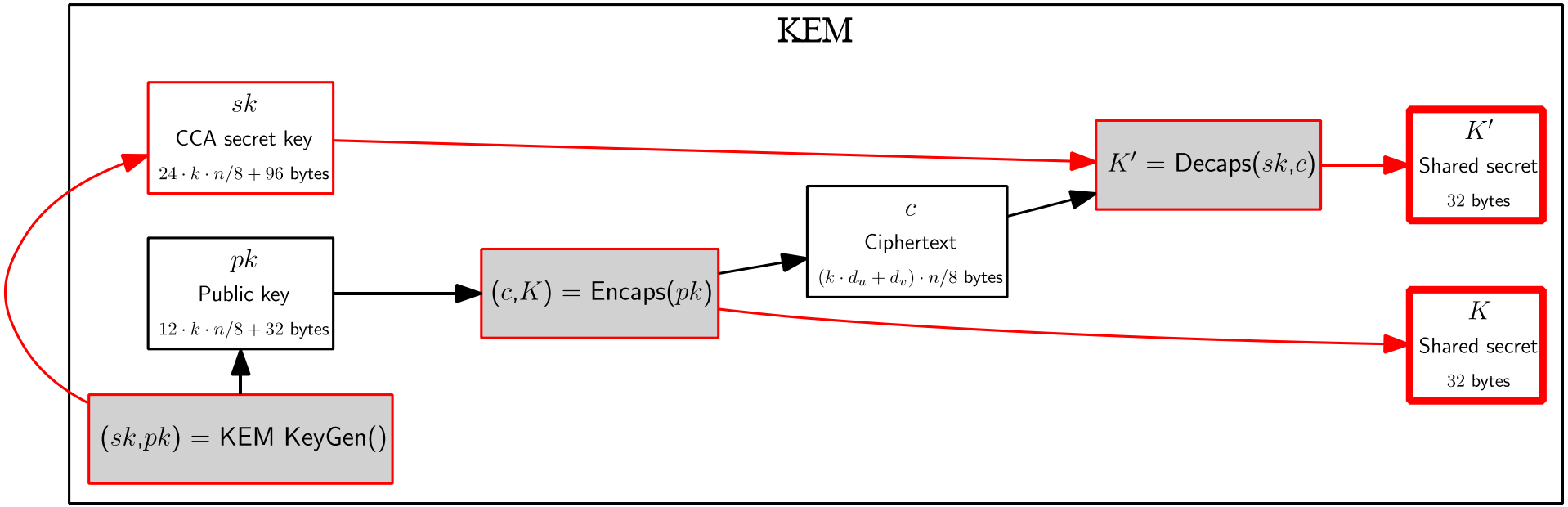
PKE = a Public-Key Encryption scheme from the MLWE problem.

This PKE is indistinguishability under chosen plaintext attack (IND-CPA)
(under the MLWE hardness assumption)

ML-KEM (Crystal-Kyber)

ML-KEM-768
Security level $= 3$
$q = 3329$
$n = 256$
$k = 3$
$\eta_1 = 2$
$\eta_2 = 2$
$d_u = 10$
$d_v = 4$
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ (integer ring)
$R_q = \mathbb{Z}_q[X]/(X^n + 1)$ (polynomial ring, n-degree polynomial)
R_q^k (a k-dimensional vector of polynomials with polynomials in R_q)
$R_q^{k \times k}$ (module of dimension $k \times k$, k-by-k matrix of polynomials with polynomials in R_q)
$\zeta = 17$ (primitive n-th root of unity modulo q)

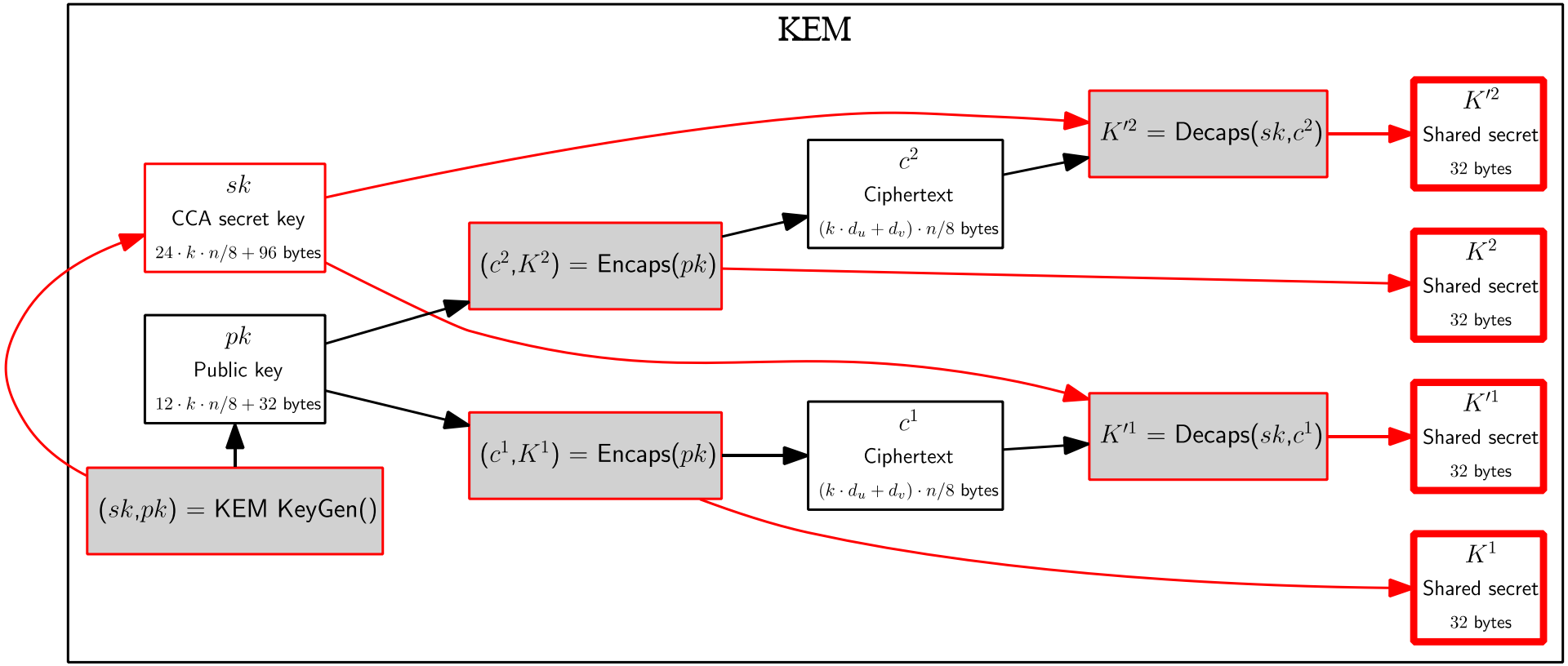
Parameter set	Decapsulation failure rate
ML-KEM-512	$2^{-138.8}$
ML-KEM-768	$2^{-164.8}$
ML-KEM-1024	$2^{-174.8}$



KEM = Key Encapsulation Mechanism

ML-KEM (Crystal-Kyber)

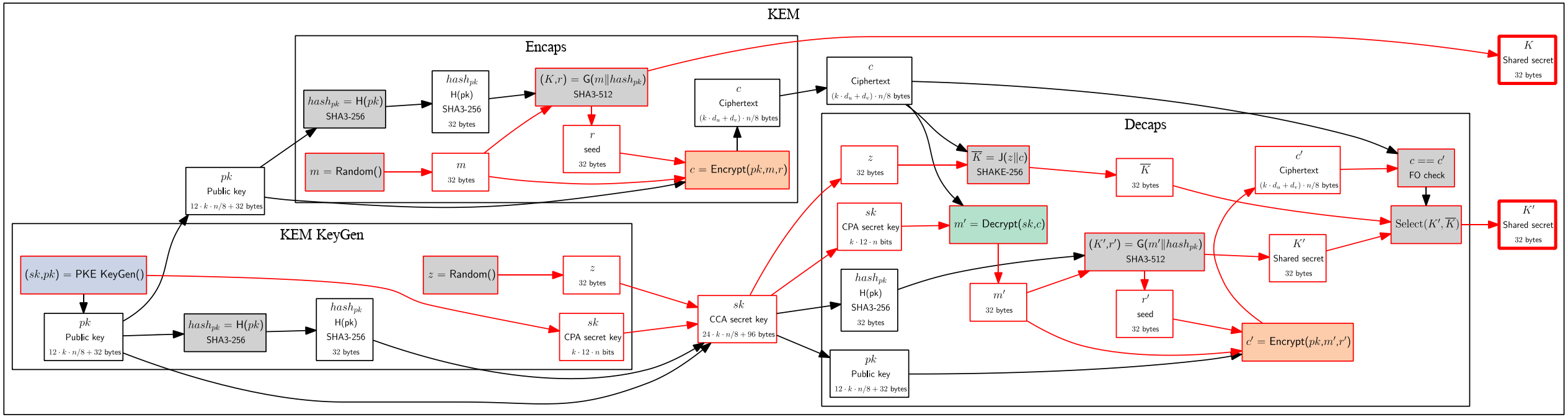
ML-KEM-768
Security level $= 3$
$q = 3329$
$n = 256$
$k = 3$
$\eta_1 = 2$
$\eta_2 = 2$
$d_u = 10$
$d_v = 4$
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ (integer ring)
$R_q = \mathbb{Z}_q[X]/(X^n + 1)$ (polynomial ring, n-degree polynomial)
R_q^k (a k-dimensional vector of polynomials with polynomials in R_q)
$R_q^{k \times k}$ (module of dimension $k \times k$, k-by-k matrix of polynomials with polynomials in R_q)
$\zeta = 17$ (primitive n-th root of unity modulo q)



ML-KEM is indistinguishability under non-adaptative chosen ciphertext attacks (IND-CCA1).
ML-KEM is indistinguishability under adaptative chosen ciphertext attacks (IND-CCA2).

Almeida, José Bacelar, et al. "Formally Verifying Kyber: Episode V: Machine-Checked IND-CCA Security and Correctness of ML-KEM in EasyCrypt." Annual International Cryptology Conference. Cham: Springer Nature Switzerland, 2024.

ML-KEM (Crystal-Kyber)



Fujisaki-Okamoto (FO) transform with implicit rejection

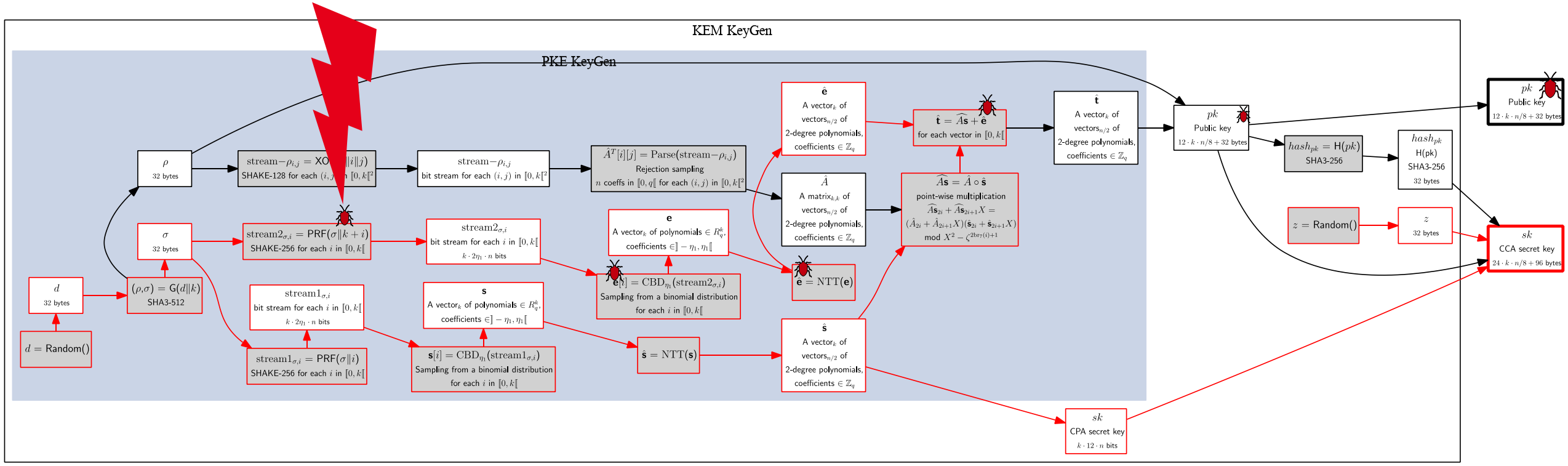
except that the hash of the public key is fed as an additional argument into G

Fujisaki, Eiichiro, and Tatsuaki Okamoto. "Secure integration of asymmetric and symmetric encryption schemes." Journal of cryptology 26 (2013): 80-101.

https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/C0D3W1KoINY/m/99kIvydoAwAJ

PKE KeyGen, Fault injection attack

Encrypt can also be attacked

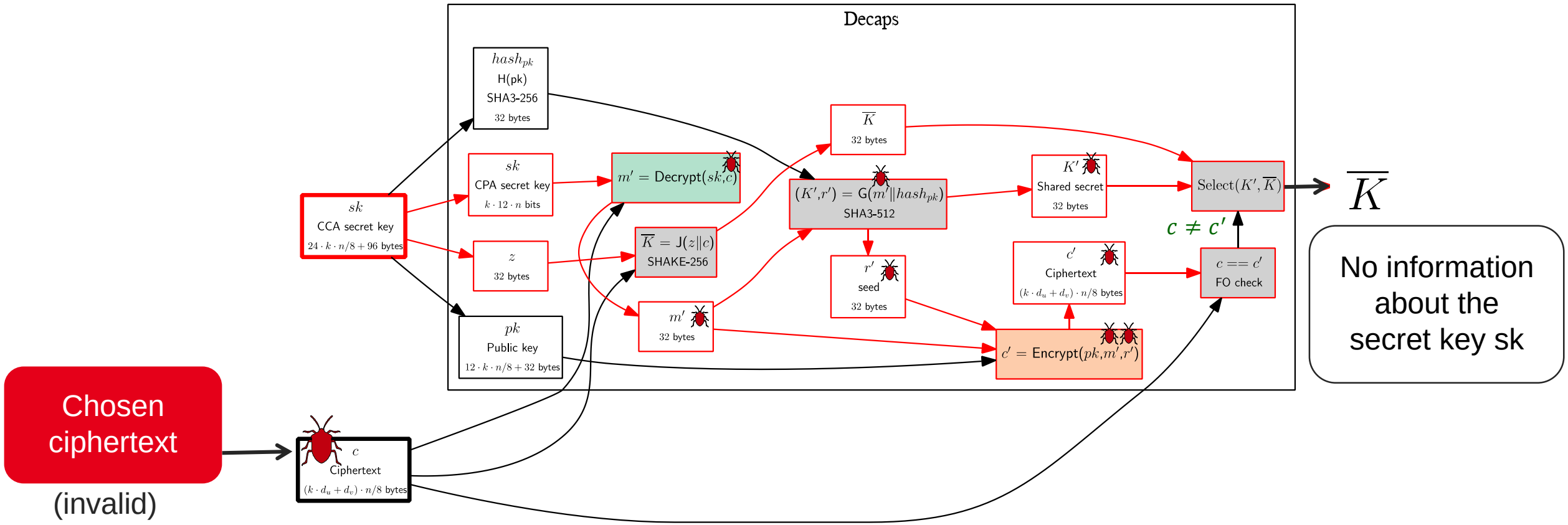


$$t = A \circledast s + e \Rightarrow t = A \circledast s + s = (A + id) \circledast s \Rightarrow s$$

Check counter coin final value after all executions of PRF functions

Ravi, Prasanna, et al. "Number "not used" once-practical fault attack on pqm4 implementations of NIST candidates." Constructive Side-Channel Analysis and Secure Design: 10th International Workshop, COSADE 2019, Darmstadt, Germany, April 3–5, 2019, Proceedings 10. Springer International Publishing, 2019.
Target = STM32F407, FI = EMFI.

FO transform, Fault injection attack



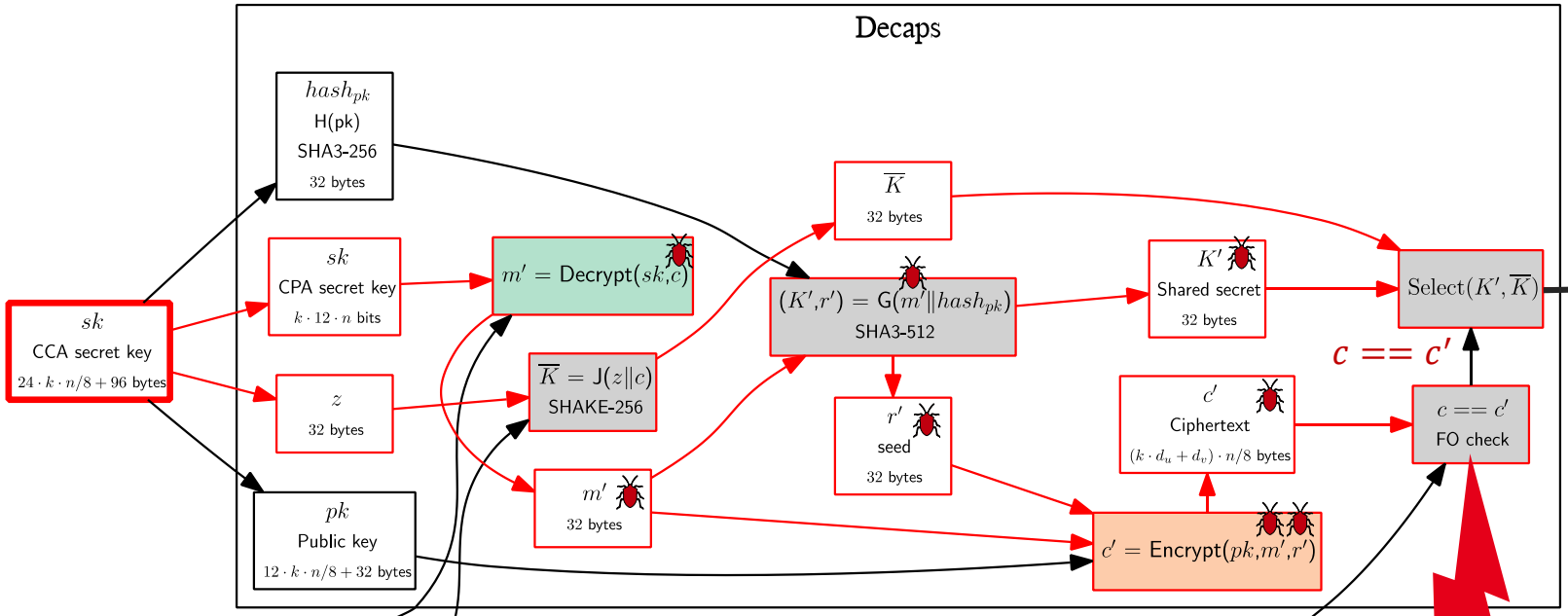
Fujisaki-Okamoto (FO) transform with implicit rejection

except that the hash of the public key is fed as an additional argument into G

Fujisaki, Eiichiro, and Tatsuaki Okamoto. "Secure integration of asymmetric and symmetric encryption schemes." Journal of cryptology 26 (2013): 80-101.

https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/C0D3W1KoINY/m/99kIvydoAwAJ

FO transform, Fault injection attack



Chosen ciphertext

(invalid)

Fault injection attack

Information about the secret key sk

Compare K' to 2 candidates => 1536 queries to recover sk for ML-KEM 512

Compare K' to 2^a candidates => $\frac{1536}{a}$ queries to recover sk for ML-KEM 512

Redundancy
Ciphertext sanity check

Xagawa, Keita, et al. "Fault-injection attacks against NIST's post-quantum cryptography round 3 KEM candidates." Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part II 27. Springer International Publishing, 2021.
Target = STM32F415, FI = CW Clock glitch.

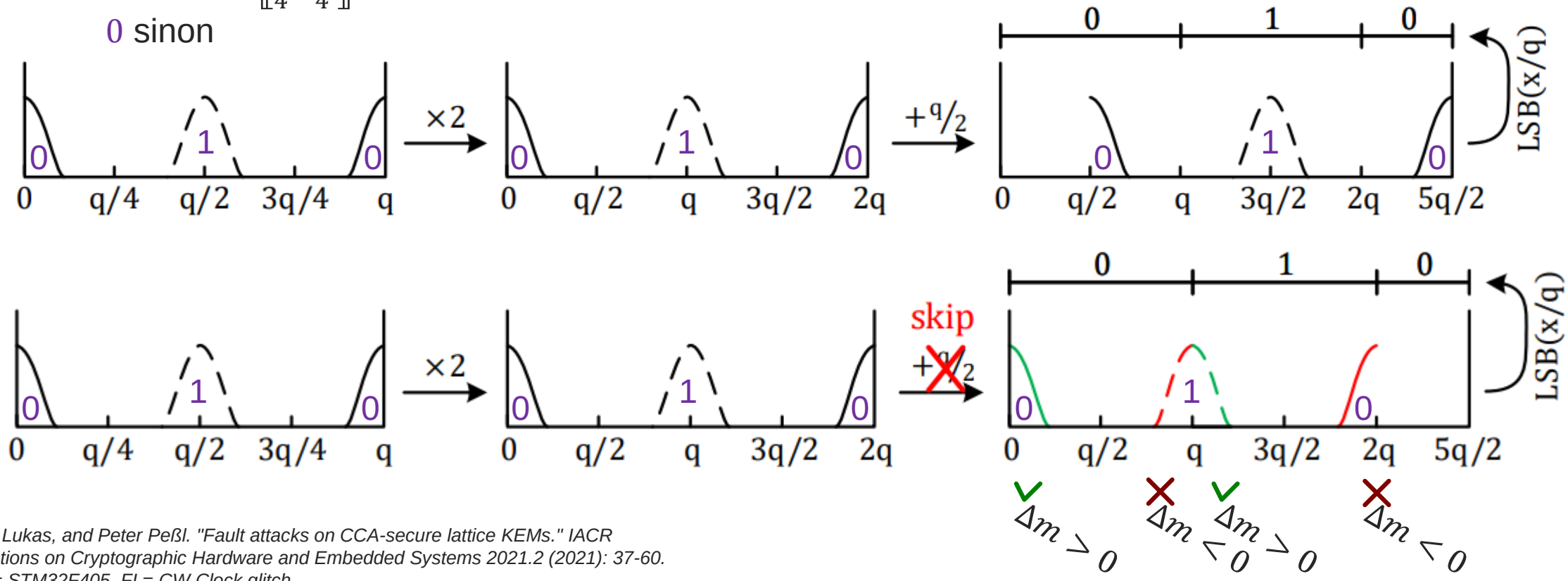
Decrypt, Fault injection attack

$vsu = m + \Delta m \quad \Delta m = e^T \circledast r + e_2 + \Delta v - s^T \circledast e_1 - s^T \circledast \Delta u$

$\Delta m \in \left[\!\left[-\frac{q}{4}, \frac{q}{4}\right]\!\right]$ for valid message decoding.

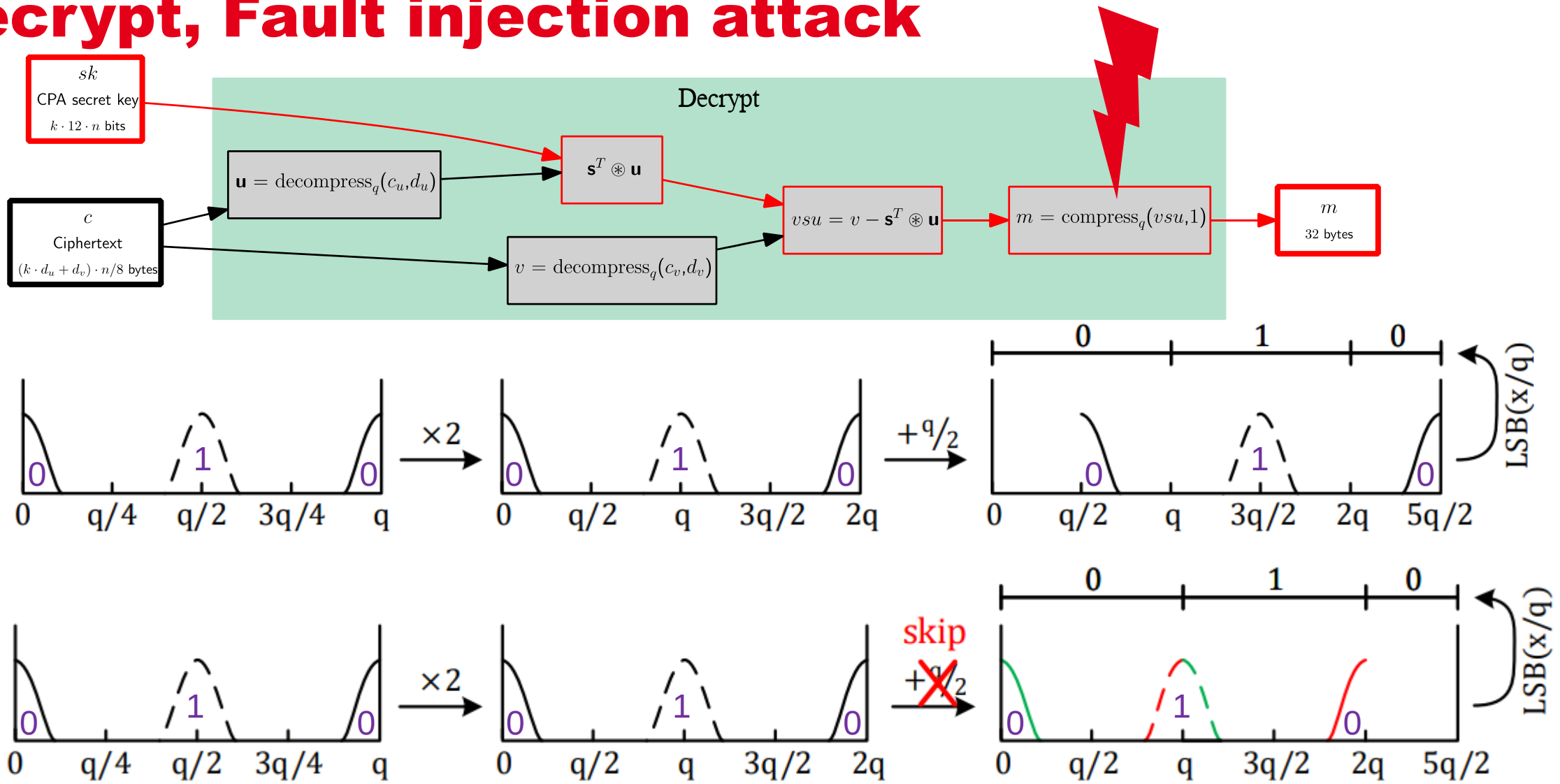
$m[i] = 1$ if $vsu \in \left[\!\left[\frac{q}{4}, \frac{3q}{4}\right]\!\right]$

0 sinon



Prokop, Lukas, and Peter Peßl. "Fault attacks on CCA-secure lattice KEMs." IACR Transactions on Cryptographic Hardware and Embedded Systems 2021.2 (2021): 37-60.
Target = STM32F405, FI = CW Clock glitch.

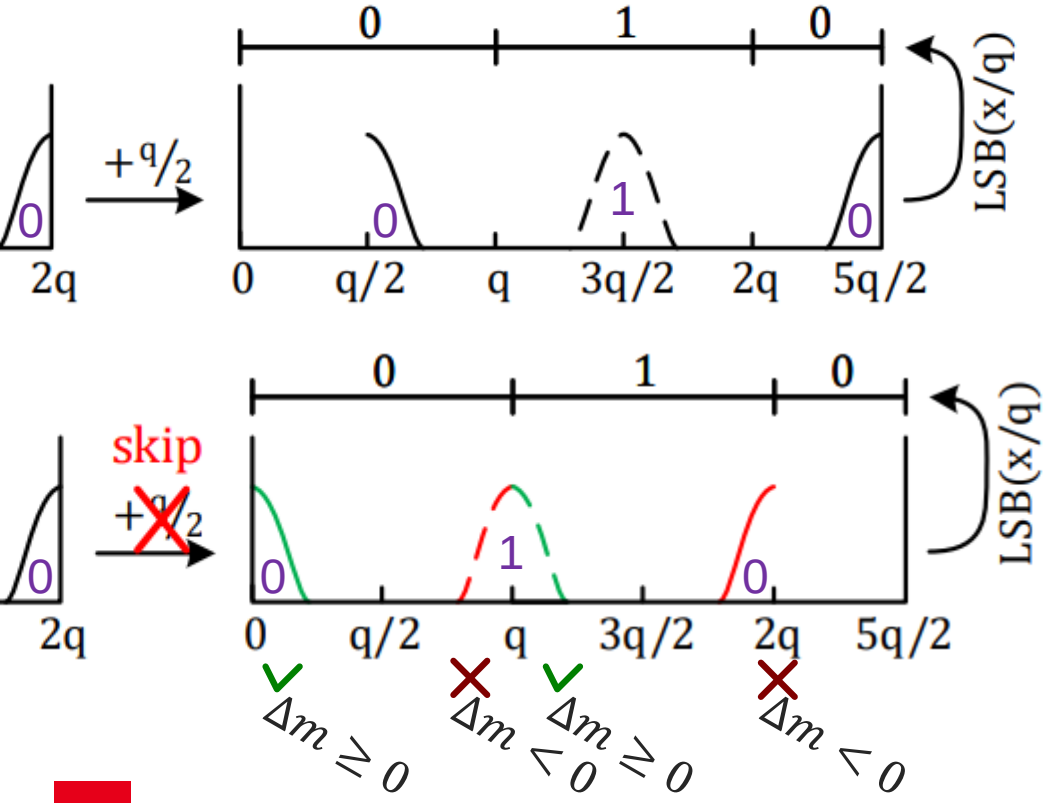
Decrypt, Fault injection attack

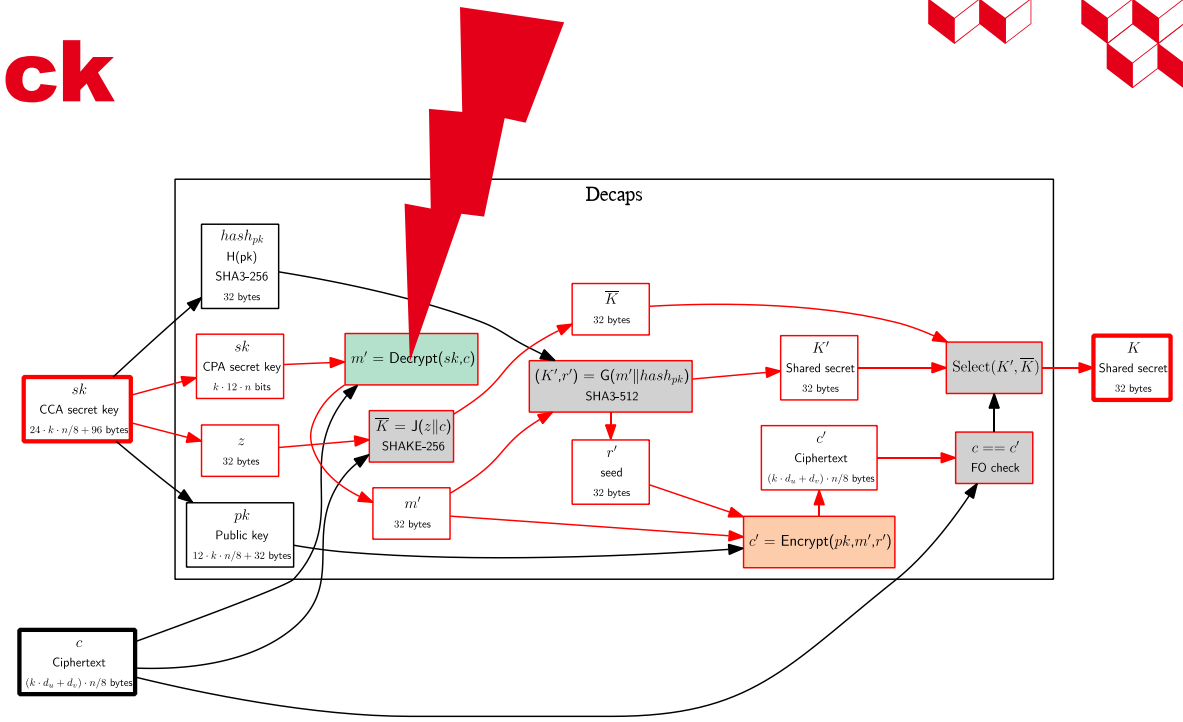


Prokop, Lukas, and Peter Peßl. "Fault attacks on CCA-secure lattice KEMs." IACR Transactions on Cryptographic Hardware and Embedded Systems 2021.2 (2021): 37-60.
Target = STM32F405, FI = CW Clock glitch.

Decrypt, Fault injection attack

$$\Delta m = e^T \circledast r + e_2 + \Delta v - s^T \circledast e_1 - s^T \circledast \Delta u$$





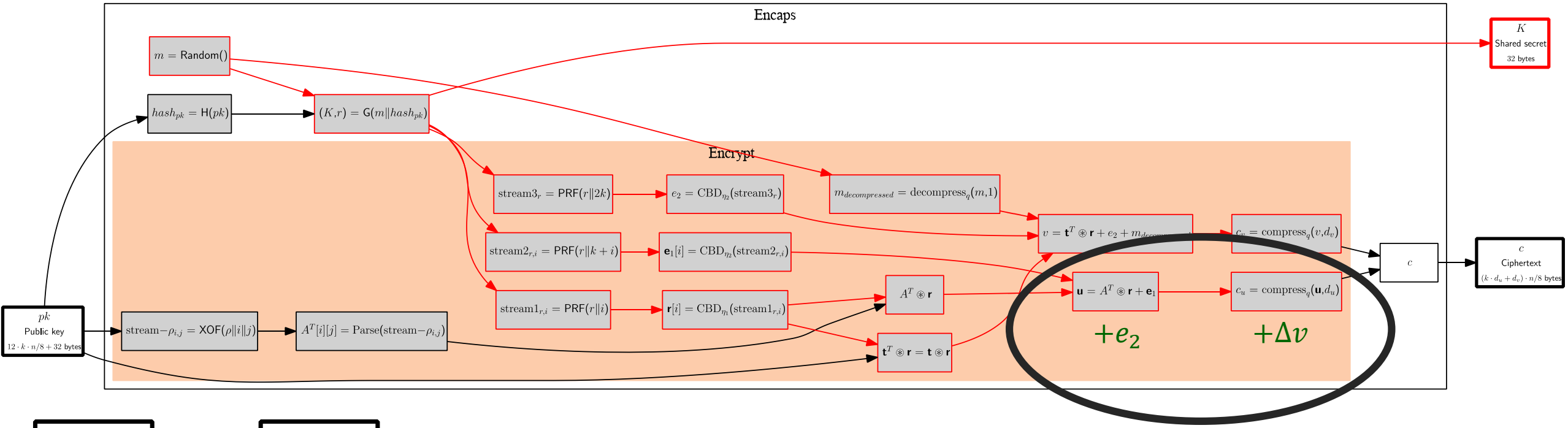
✓ Expected K :
→ Ineffective fault → $\Delta m \geq 0$

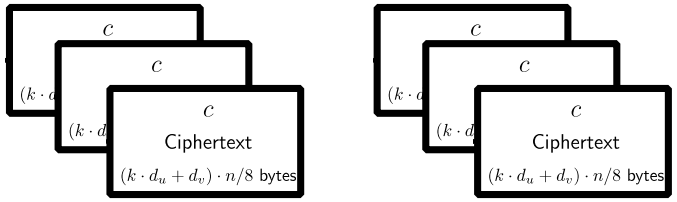
✗ Unexpected K:
→ Effective fault → $\Delta m < 0$

Prokop, Lukas, and Peter Peßl. "Fault attacks on CCA-secure lattice KEMs." IACR Transactions on Cryptographic Hardware and Embedded Systems 2021.2 (2021): 37-60.
Target = STM32F405, FI = CW Clock glitch.

Decrypt, Fault injection attack

Generate valid ciphertexts with $|e_2 - \Delta v| < 10$



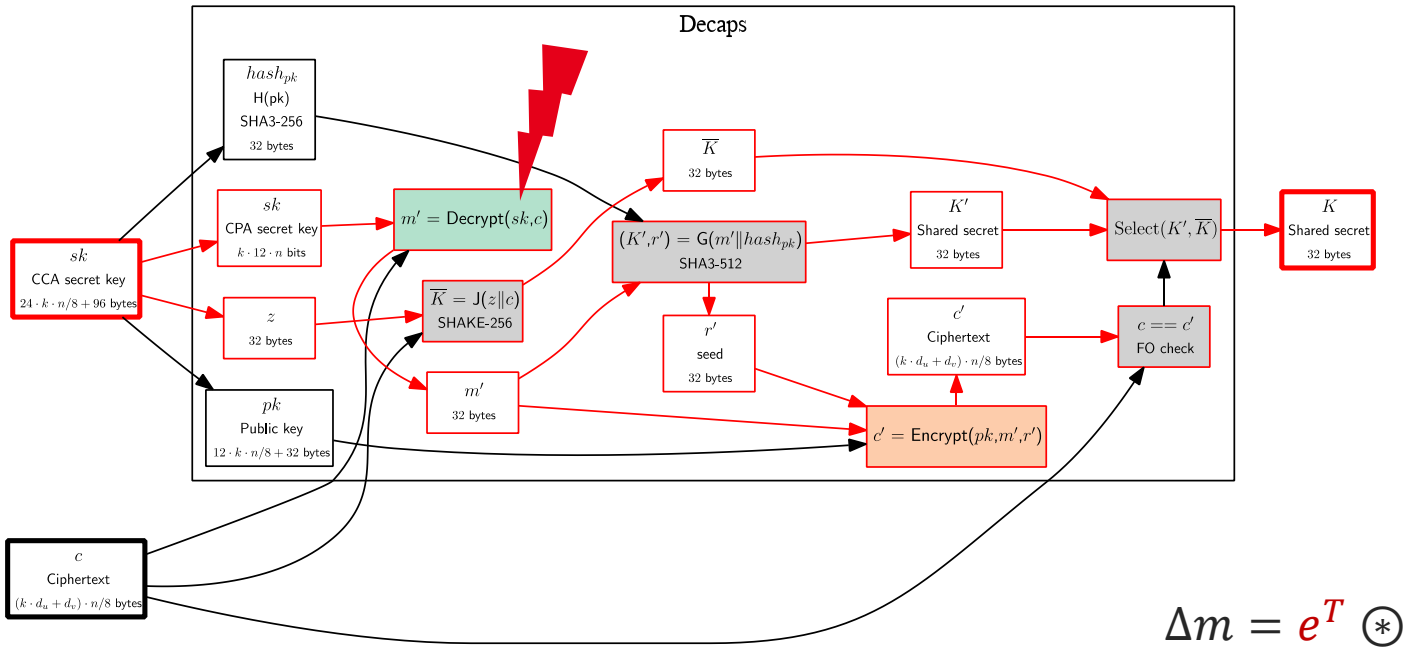


$$\Delta m = e^T \circledast r - s^T \circledast (e_1 - \Delta u) + (e_2 + \Delta v)$$

Small ⇨ More information about s and e in inequalities $\Delta m < 0$

Prokop, Lukas, and Peter Peßl. "Fault attacks on CCA-secure lattice KEMs." IACR Transactions on Cryptographic Hardware and Embedded Systems 2021.2 (2021): 37-60.
Target = STM32F405, FI = CW Clock glitch.

Decrypt, Fault injection attack



Redundancy
Shuffling

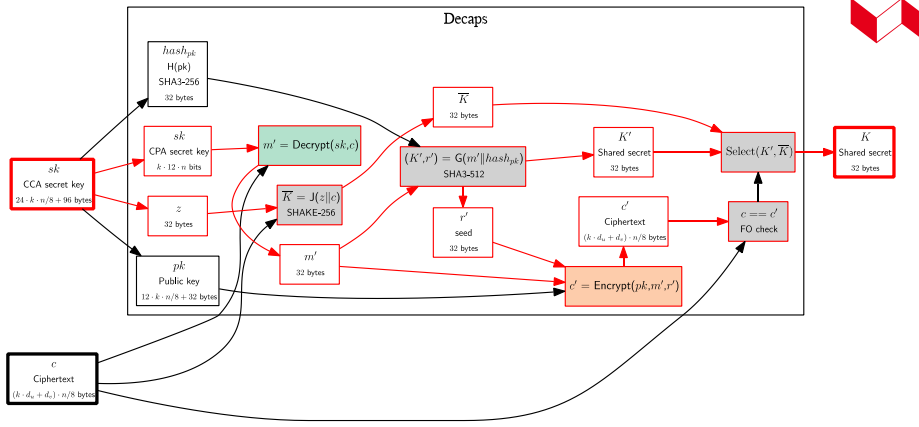
$$\Delta m = e^T \circledast r - s^T \circledast (e_1 - \Delta u) + e_2 + \Delta v$$

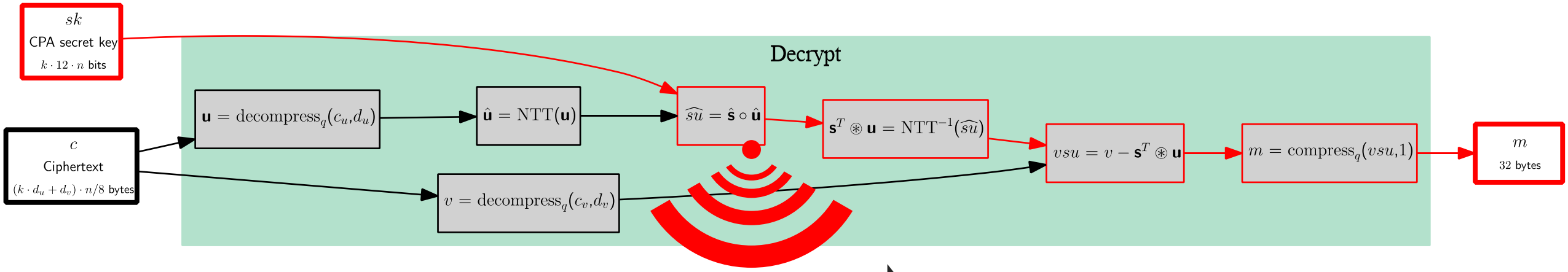
✓ Expected K :
→ Ineffective fault → $\Delta m \geq 0$ ⇨ Filter these inequalities that could be unsuccessful fault injection.

✗ Unexpected K:
→ Effective fault → $\Delta m < 0$ ⇨ Solve the system of linear inequalities with belief-propagation techniques (60k or 125k Kyber512 decapsulations to recover s and e).

Prokop, Lukas, and Peter Peßl. "Fault attacks on CCA-secure lattice KEMs." IACR Transactions on Cryptographic Hardware and Embedded Systems 2021.2 (2021): 37-60.
Target = STM32F405, FI = CW Clock glitch.

Decrypt, CPA side-channel attack





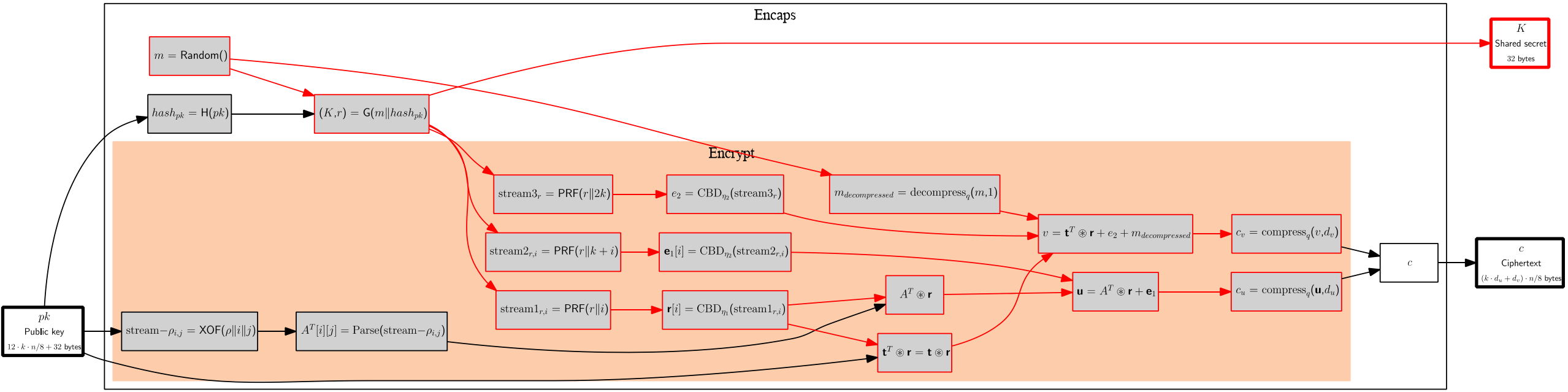
Leakage on $\hat{s}$ ⇨ Correlation Power Analysis (CPA) attack
200 traces to recover $\hat{s}$

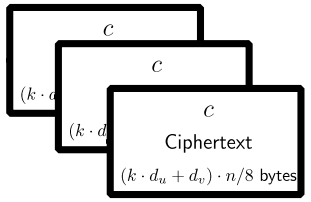
Masking

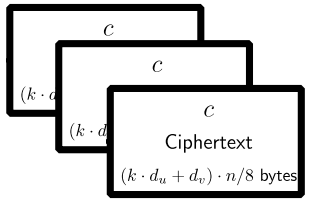
Mujdei, Catinca, et al. "Side-channel analysis of lattice-based post-quantum cryptography: Exploiting polynomial multiplication." ACM Transactions on Embedded Computing Systems 23.2 (2024): 1-23.
Target = STM32F415, SCA = CW Power CPA.

FO-Transform, Template side-channel attack

Build two classes of valid ciphertexts ($m = m_1$, $m = m_2$)

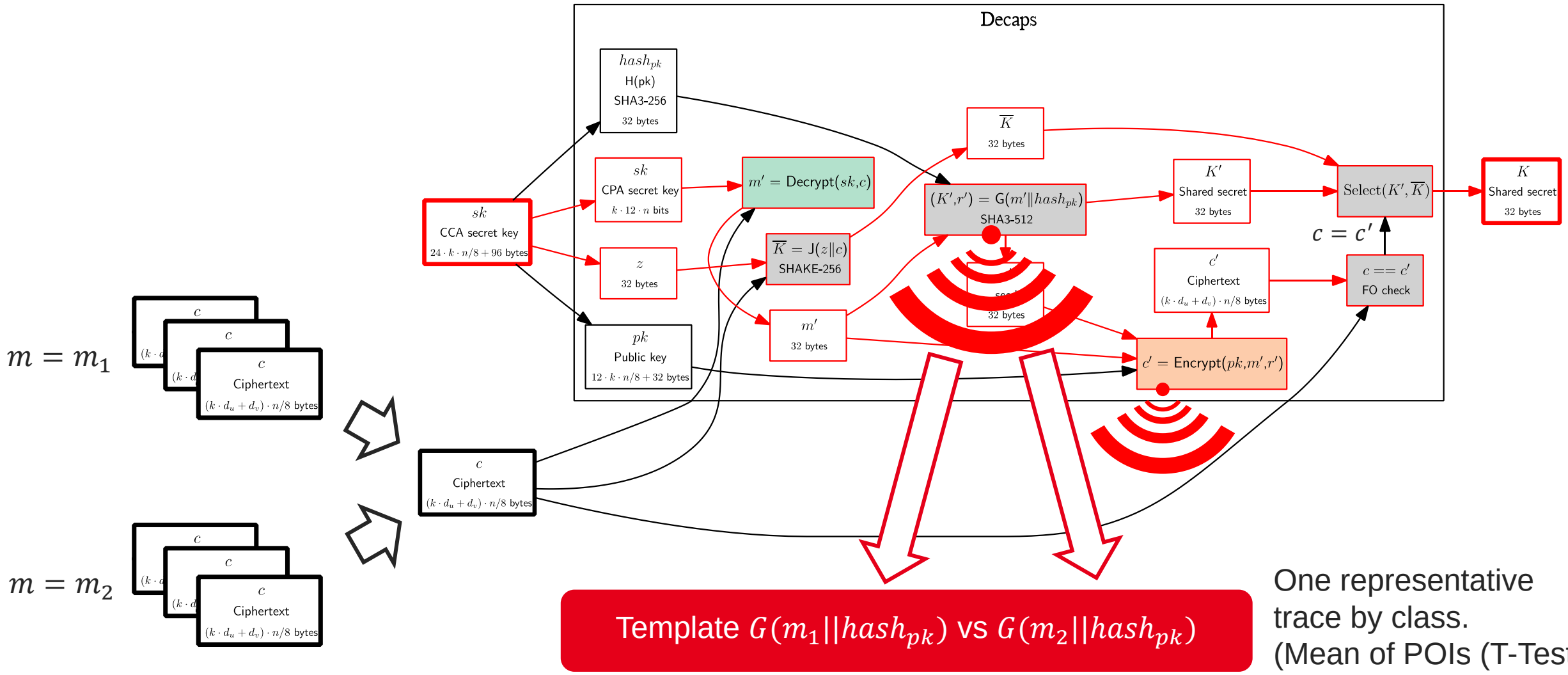


$m = m_1$ 

$m = m_2$ 

Ravi, Prasanna, et al. "Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs." IACR transactions on cryptographic hardware and embedded systems (2020): 307-335.
Target = STM32F407, SCA = EM CPA.

FO-Transform, Template side-channel attack



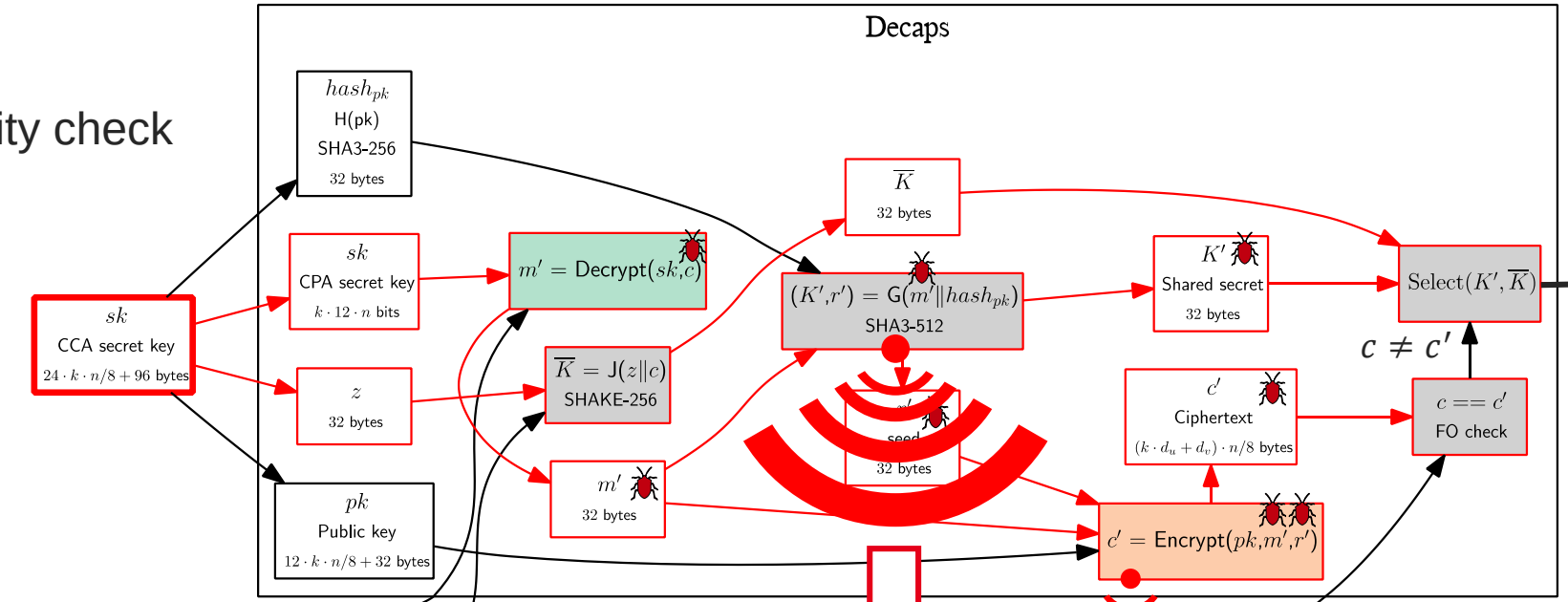
Template $G(m_1||hash_{pk})$ vs $G(m_2||hash_{pk})$

One representative trace by class.
(Mean of POIs (T-Test))

Ravi, Prasanna, et al. "Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs." IACR transactions on cryptographic hardware and embedded systems (2020): 307-335.
Target = STM32F407, SCA = EM CPA.

FO-Transform, Template side-channel attack

Masking
Ciphertext sanity check



No information about the secret key sk in $\overline{K}$

$\overline{K}$

Chosen ciphertext

c
Ciphertext
$(k \cdot d_u + d_v) \cdot n/8$ bytes

(invalid sparse ciphertext)

Plaintext checking oracle

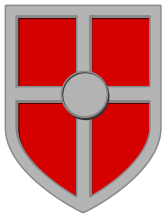
Identify the value of s_i by using several invalid sparse ciphertexts

Guess on s_i → $m' = m_0$ or m_1 → Template $G(m_1||hash_{pk})$ vs $G(m_2||hash_{pk})$ → Valid or invalid guess on s_i

Ravi, Prasanna, et al. "Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs." IACR transactions on cryptographic hardware and embedded systems (2020): 307-335.
Target = STM32F407, SCA = EM CPA.

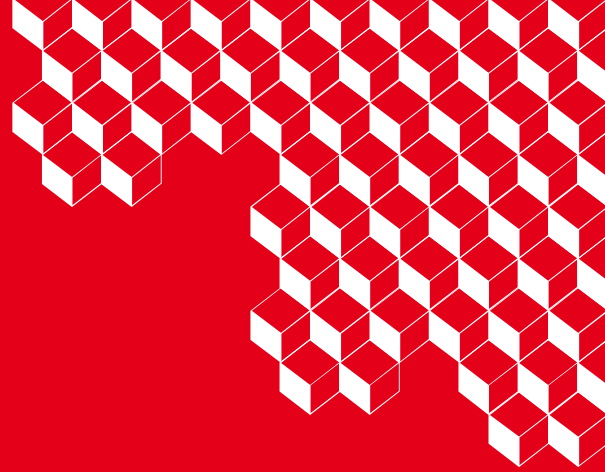
Conclusion

- Countermeasures must be combined to design secure ML-KEM implementation:



  - Masking
  - Shuffling
  - Redundancy
  - Sanity checks

- This overview is an introduction to some existing SCA and FI attacks on ML-KEM implementations:
  - Warning: this overview is not an exhaustive one
    Examples:
    - *Hermelink, Julius, Peter Pessl, and Thomas Pöppelmann. "Fault-enabled chosen-ciphertext attacks on Kyber." Progress in Cryptology–INDOCRYPT 2021: 22nd International Conference on Cryptology in India, Jaipur, India, December 12–15, 2021, Proceedings 22. Springer International Publishing, 2021.*
    - *Ravi, Prasanna, et al. "Fiddling the twiddle constants-fault injection analysis of the number theoretic transform." IACR Transactions on Cryptographic Hardware and Embedded Systems (2023)*
    - *…*
  - Warning: there are attacks against implementation with countermeasures
    Examples:
    - Soft-Analytical Side Channel Attack on NTT to recover secret from masked ML-KEM implementations
      Pessl, Peter, and Robert Primas. "More practical single-trace attacks on the number theoretic transform." Progress in Cryptology–LATINCRYPT 2019: 6th International Conference on Cryptology and Information Security in Latin America, Santiago de Chile, Chile, October 2–4, 2019, Proceedings 6. Springer International Publishing, 2019.
    - *Qiao, Zehua, et al. "When NTT Meets SIS: Efficient Side-channel Attacks on Dilithium and Kyber." Cryptology ePrint Archive (2023).*
    - *…*



Simon Pontié `simon.pontie@cea.fr`