POLITECNICO DI MILANO

Dipartimento di
Elettronica, Informazione
e Bioingegneria

Fault attack friendliness of post-quantum cryptosystems

Alessandro Barenghi, Gerardo Pelosi

Fault Diagnosis and Tolerance in Cryptography 2023 - 10th September 2023

Outline

Context

- Post quantum cryptosystems are coming:
  - Draft NIST FIPS 203, 204, 205 (Kyber, Dilithium, SPHINCS$^+$) up for comments
  - Call for Additional Digital Signature Schemes closed last June

Goal of the talk

- An overview of the common traits among post-quantum cryptographic primitives
- Highlight wherever the common traits are fault-fragile

NIST Post Quantum Standardization competition

- First public competition for asymmetric cryptographic primitive design
  - previous ones yielded AES, SHA-3, SHAKE
  - previous asymmetric encryption schemes standardized after being popular
- Began in 2017, now “over but not quite yet”
  - FIPS drafts up for comments until this November

Requirements for the new designs

- NIST requires resistance to “active attackers”
  - For encryption schemes, the attacker has access to a decryption oracle
  - For signature schemes, the attacker has access to a signature oracle
- Side channel attack security explicitly among desirable additional properties

Public Key Encryption schemes

- NIST call asked for two kind of primitives
  - Public Key Encryption (PKEs): encrypt and decrypt a generic message
  - Key Encapsulation Methods (KEMs): encrypt and decrypt a short random key
- KEMs won the “popularity contest”
  - Only one PKE promoted to second round (LEDApkc), merged with a corresponding KEM
  - PKEs are advantageous when small messages are transmitted
- Most KEMs are built... adding components to a PKE!

Underlying hard problems

High level view of hard problems

Given a matrix G and $c = aG + e$, where e is "small", it is hard to find a, e

- message encoded as either a, e or both
- remaining element between a, e, drawn at random
- private key allows to retrieve a, e from c (removing the "error" e from aG)

PQ PKEs may have failures

- Example: if e is too "large", but small enough to be admissible by cipher parameters
- Failures leak information on the private key:
  - Cipher parameters designed so that they occur with negligible probability/never
- In both cases, injecting controlled faults will make failures appear

How to get to active-attacker resistance

Increasing attacker capabilities →

OW-CPA (OW-Passive)

1. Attacker gets the pk
2. Attacker gets a random ciphertext c
3. Attacker wins if it decrypt c

IND-CPA

1. Attacker gets pk and chooses two ptx m_0, m_1
2. Attacker gets either $\mathrm{Enc}(m_0)$ or $\mathrm{Enc}(m_1)$
3. Attacker wins if it guesses which it got

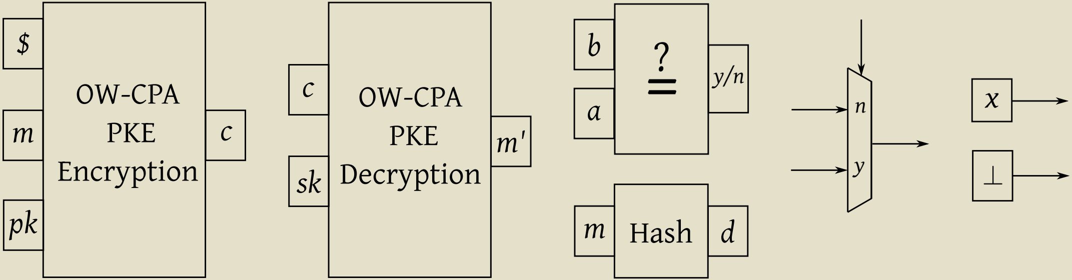
IND-CCA

1. Attacker gets pk and chooses two ptx m_0, m_1
2. As in IND-CPA, but the attacker can also get $\mathrm{Dec}(m_x)$, as long as $x \notin \{0, 1\}$

Separation of concerns approach

Design a PKE, secure under a weak attacker model, “promote it through constructions”.

Ingredients



The Fujisaki-Okamoto (FO) transform

The majority of PQ KEMs are derived from a PKE through the FO transform composing two elements, the T and U transforms [Hofheinz et al., 2017]

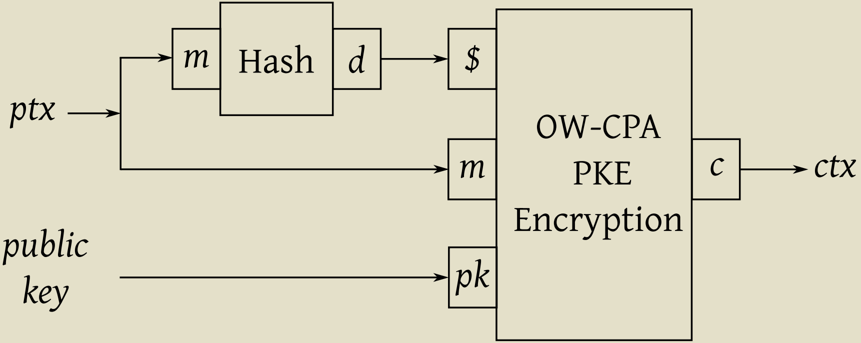
T transform

- T: takes a randomized OW-CPA PKE, “derandomizes” and adds decryption check

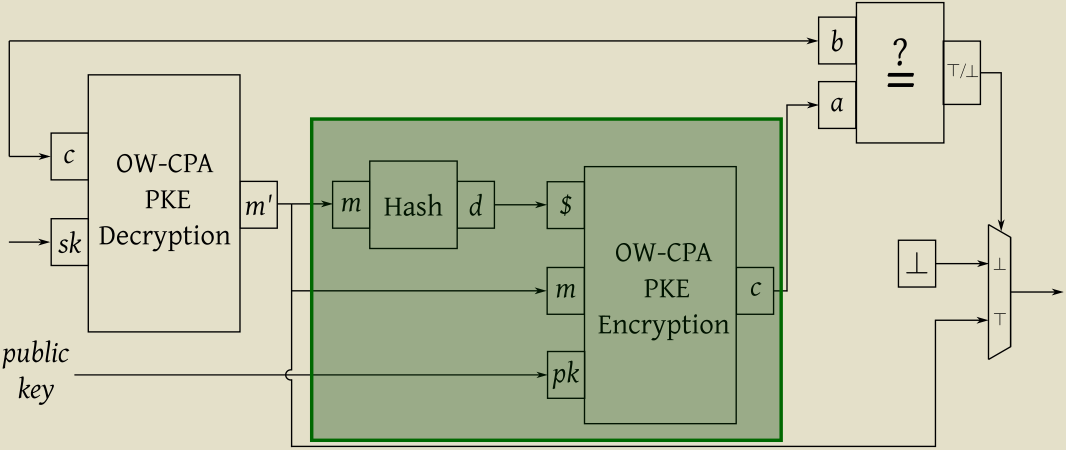
U transform

- U: Takes the output of the T transform, achieves IND-CCA through
  - Feeding the IND-CPA PKE a random message
  - In case of a PKE decryption failure either
    - $U^{\perp}$ fail in decapsulating the key (outputting $\perp$), or
    - $U^{\not\perp}$ output a pseudorandom string depending on a secret and the ciphertext

T transform encryption



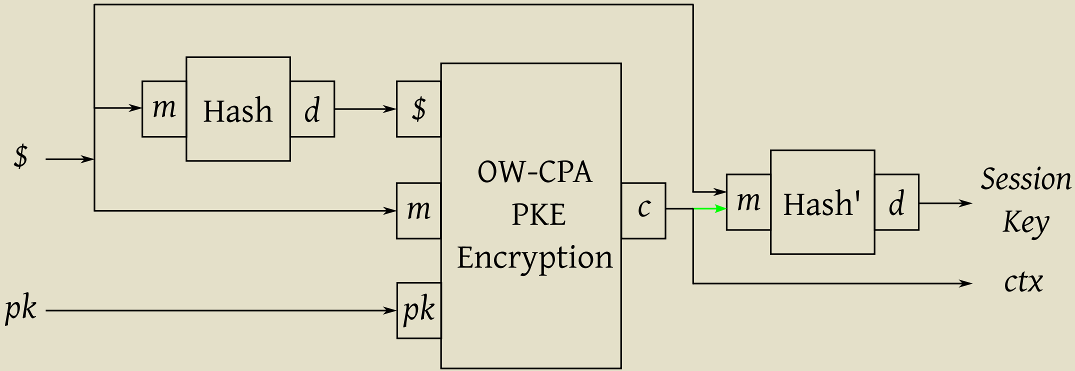
⊤ transform decryption



T transform effects

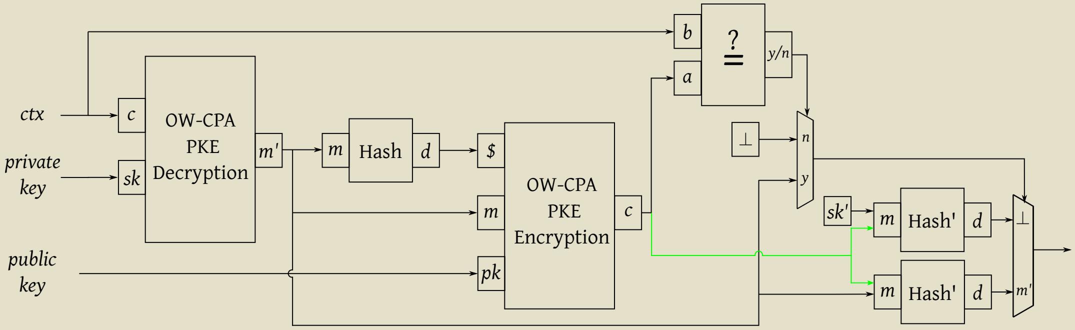
- Obtains a DPKE, allowing reencryption on receiver side
- Achieves rigidity [Bernstein and Persichetti, 2018]:
  - informally: no two distinct ciphertext decrypt to the same plaintext
- Non rigid KEMs allow a CCA attacker to:
  1. Collect a correct m, c pair
  2. Ask the encryption oracle to decrypt $c + \varepsilon$ and see if it yields still m
  3. Employ the information to infer the value of the “small” e
- Fault fragility: non rigidity is restored with a fault:
  - Flipping the result of the comparison
  - Skipping the selection at the end

$U^{\not\perp}$ transform encryption



the green colored arrow is optional

$U^{\not\perp}$ transform decryption

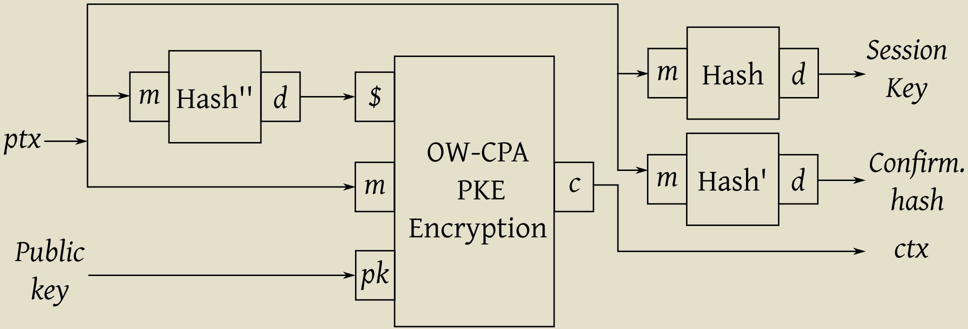


the green colored arrow is optional

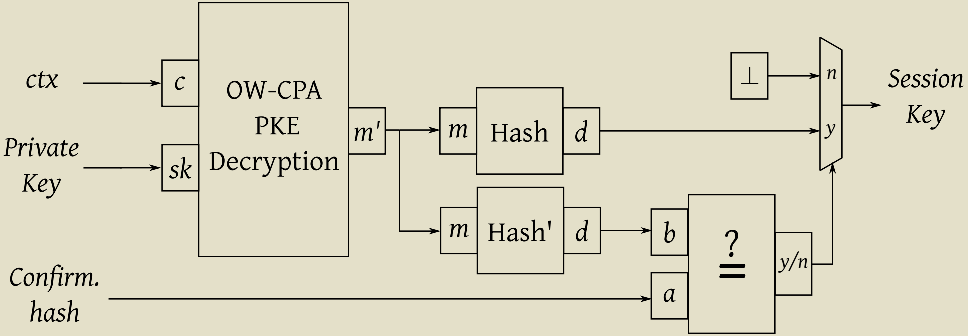
$U^{\not\perp}$ transform effects

- Obtains a KEM employing coins as a message
- Hashes the plaintext (and optionally the ciphertext together)
  - Prevents straightforward differential fault analysis, as a side effect
- Adds (optionally) implicit rejection
  - Implicit rejection effectively hides failures
- 1st Fault fragility of implicit rejection
  - Skipping the final comparison will make them evident again [Oder et al., 2018]
- 2nd Fault fragility of implicit rejection:
  - Run twice the entire decap process, with the message expected to fail
  - One in two cases, inject a fault in the computation of the “garbage answer”
  - Employed in [Bernstein, 2022] to break NTRU, assuming a persistent fault, and an output collection before it takes place

Besides FO transform - Plaintext Confirmation - Encap



KEM with Plaintext Confirmation - Decapsulation



Plaintext confirmation

History and effects

- Dent [Dent, 2002] proposed plaintext confirmation as a building block for KEMs
- Dent's idea prevents tampering with the ciphertext, as the attacker is not able to predict the value of the decrypted (modified) plaintext
- A variant introduced in [Baldi et al., 2020] and also used in BIKE allows also to check that the ptx fed to the KEM is obtained via a SHAKE (or another XOF)

- Fault fragility: instruction skipping/comparison altering still works
  - Smaller attack surface w.r.t. implicit rejection, while performing similar task

Signature algorithms

From interactive to non interactive

- A very popular approach to design a signature is:
  1. Design an interactive identification scheme between a prover and a verifier
  2. Remove the interactivity turning it into a signature via [Fiat and Shamir, 1986]
- Dilithium, selected for standardisation, also employs a similar framework

High level view of an ID scheme using a hard problem P

1. Generate a keypair: public key: an instance of P, private key: the solution
2. Prover: build an instance P' related to P, solve it with the knowledge of the private key
3. Prover: convince verifier that you know the private key showing either
   - that you generated P' from P or
   - showing the solution to P' without revealing the private key

The CROSS [Baldi et al., 2023] ID scheme

param: group $G \subset \mathbb{F}_q^n$, Hard to obtain $\mathbf{e}$, given $\mathbf{s}, \mathbf{H}$

sk: restricted vector $\mathbf{e} \in G$

pk: parity-check matrix $\mathbf{H} \in \mathbb{F}_q^{n \times r}$, syndrome $\mathbf{s} = \mathbf{H}\mathbf{e}^\top$

PROVER		VERIFIER
Sample $\texttt{Seed} \xleftarrow{\$} \{0;1\}^\lambda$, $(\mathbf{u}', \mathbf{e}') \xleftarrow{\texttt{Seed}} \mathbb{F}_q^n \times G$		
Compute $\mathbf{d} \in G$ such that $\mathbf{d} \star \mathbf{e}' = \mathbf{e}$		
Set $\mathbf{u} = \mathbf{d} \star \mathbf{u}'$ and $\widetilde{\mathbf{s}} = \mathbf{u}\mathbf{H}^\top$		
Set $c_0 = \mathrm{Hash}(\widetilde{\mathbf{s}}, \mathbf{d})$, $c_1 = \mathrm{Hash}(\mathbf{u}', \mathbf{e}')$	$\xrightarrow{(c_0, c_1)}$	
	$\xleftarrow{\beta}$	Sample $\beta \xleftarrow{\$} \mathbb{F}_q^*$
Compute $\mathbf{y} = \mathbf{u}' + \beta\mathbf{e}'$ \\Uniformly random over $\mathbb{F}_q$		
Set $h = \mathrm{Hash}(\mathbf{y})$ \\First response		
	$\xrightarrow{h}$	
		Sample $b \xleftarrow{\$} \{0, 1\}$
	$\xleftarrow{b}$	
If $b = 0$, set $\texttt{rsp} = (\mathbf{y}, \mathbf{d})$ \\Second response		
If $b = 1$, set $\texttt{rsp} = \texttt{Seed}$ \\Second response		
	$\xrightarrow{\texttt{rsp}}$	
		Verify c_b using $\texttt{rsp}$

The Fiat-Shamir transform for 5-pass schemes

PROVER (sk)		VERIFIER (pk)
Prepare $\mathtt{Com}$	$\xrightarrow{\mathtt{Com}}$	
	$\xleftarrow{\mathtt{Ch}_1}$	Sample $\mathtt{Ch}_1$
Compute $\mathtt{Rsp}_1$	$\xrightarrow{\mathtt{Rsp}_1}$	
	$\xleftarrow{\mathtt{Ch}_2}$	Sample $\mathtt{Ch}_2$
Compute $\mathtt{Rsp}_2$		
	$\xrightarrow{\mathtt{Rsp}_2}$	
		Accept or reject

The Fiat-Shamir transform for 5-pass schemes

PROVER (sk)		VERIFIER (pk)
Prepare $\mathtt{Com}$		
Set $\mathtt{Ch}_1 = \text{Hash}(\mathtt{Msg}, \mathtt{Com})$		~~Sample $\mathtt{Ch}_1$~~
Compute $\mathtt{Rsp}_1$		
Set $\mathtt{Ch}_2 = \text{Hash}(\mathtt{Msg}, \mathtt{Com}, \mathtt{Ch}_1, \mathtt{Rsp}_1)$		~~Sample $\mathtt{Ch}_2$~~
Compute $\mathtt{Rsp}_2$		
	$\xrightarrow{\mathtt{Com}, \mathtt{Rsp}_1, \mathtt{Rsp}_2}$	
		Set $\mathtt{Ch}_1 = \text{Hash}(\mathtt{Msg}, \mathtt{Com})$
		Set $\mathtt{Ch}_2 = \text{Hash}(\mathtt{Msg}, \mathtt{Com}, \mathtt{Ch}_1, \mathtt{Rsp}_1)$
		Accept or reject

From a fault attack perspective

Faults in the control flow

- Signatures obtained from FS-transforming an ID scheme reveal the private key if both commitments are revealed in a single iteration
- Inducing a repetition of a response preparation with a different challenge can be done faulting the protocol repetition counter

Faults in the manipulated data

- Responses depending on private key material may reveal information if properly faulted during preparation (e.g., partial zeroing)
- In signature verification, there are targets beyond the final check (this afternoon’s presentation)

Concluding remarks

- Constructions for IND-CCA KEMs protect against attackers “at the ends”
  - They become a relatively soft target for fault attacks
  - If kept in place by hardening, they help in warding off other fault attacks
  - Silver lining: critical code portions appear few and cheap to harden
- ID scheme + Fiat-Shamir transform based signatures
  - Require care in avoiding control flow altering faults
  - Require care in preparing responses depending on the private key
- Is it possible to design more efficient constructions to ward off fault attacks?

Questions?

Thank you for the attention!

Bibliography I

- Baldi, M., Barenghi, A., Bitzer, S., Karl, P., Manganiello, F., Pavoni, A., Pelosi, G., Santini, P., Schupp, J., Slaughter, F., Wachter-Zeh, A., and Weger, V. (2023).
  CROSS (Codes and Restricted Objects Signature Scheme).
  https://www.cross-crypto.com/cross.html.
- Baldi, M., Barenghi, A., Chiaraluce, F., Pelosi, G., and Santini, P. (2020).
  LEDAcrypt: Low-dEnsity parity-check coDe-bAsed cryptographic systems - specification version 3.0, (April 2020).
  https://www.ledacrypt.org/.
- Bernstein, D. J. (2022).
  A One-Time Single-bit Fault Leaks All Previous NTRU-HRSS Session Keys to a Chosen-Ciphertext Attack.
  In Isobe, T. and Sarkar, S., editors, Progress in Cryptology - INDOCRYPT 2022 - 23rd International Conference on Cryptology in India, Kolkata, India, December 11-14, 2022, Proceedings, volume 13774 of Lecture Notes in Computer Science, pages 617–643. Springer.
- Bernstein, D. J. and Persichetti, E. (2018).
  Towards KEM unification.
  IACR Cryptol. ePrint Arch., page 526.
- Dent, A. W. (2002).
  A Designer's Guide to KEMs.
  Cryptology ePrint Archive, Paper 2002/174.
  https://eprint.iacr.org/2002/174.
- Fiat, A. and Shamir, A. (1986).
  How to prove yourself: Practical solutions to identification and signature problems.
  In Odlyzko, A. M., editor, Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings, volume 263 of Lecture Notes in Computer Science, pages 186–194. Springer.

Bibliography II

- Hofheinz, D., Hövelmanns, K., and Kiltz, E. (2017).
  A modular analysis of the fujisaki-okamoto transformation.
  In Kalai, Y. and Reyzin, L., editors, Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part I, volume 10677 of Lecture Notes in Computer Science, pages 341–371. Springer.
- Oder, T., Schneider, T., Pöppelmann, T., and Güneysu, T. (2018).
  Practical cca2-secure and masked ring-lwe implementation.
  IACR Trans. Cryptogr. Hardw. Embed. Syst., 2018(1):142–174.