(Post-quantum) cryptography
Security models, proofs and generic transforms

Andreas Hülsing
Eindhoven University of Technology

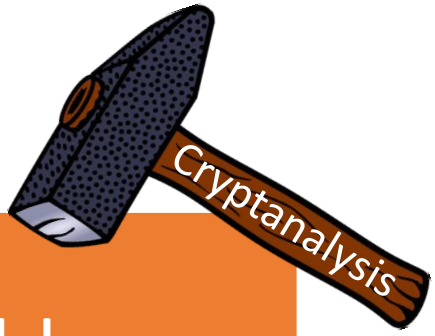
Executive School on Post-Quantum Cryptography
July 2019, TU Eindhoven

Evaluation criteria for crypto

- Performance
  - Speed
  - Size
- Cryptanalysis
  - Maturity, quantity & quality
  - Are the relevant problems analyzed
- Provable security
  - Standard model vs. (Q)ROM
  - Tight vs non-tight reduction
- Implementation security
  - Possibility / availability of protected implementations

Picture: By Rizkyharis - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=52757933

How to build PKC



(Computationally)

hard problem

RSA DL QR DDH

Proof

Problem hard ⇒ Scheme secure

PKC Scheme

RSA-OAEP ECDSA DH-KE

Security proofs

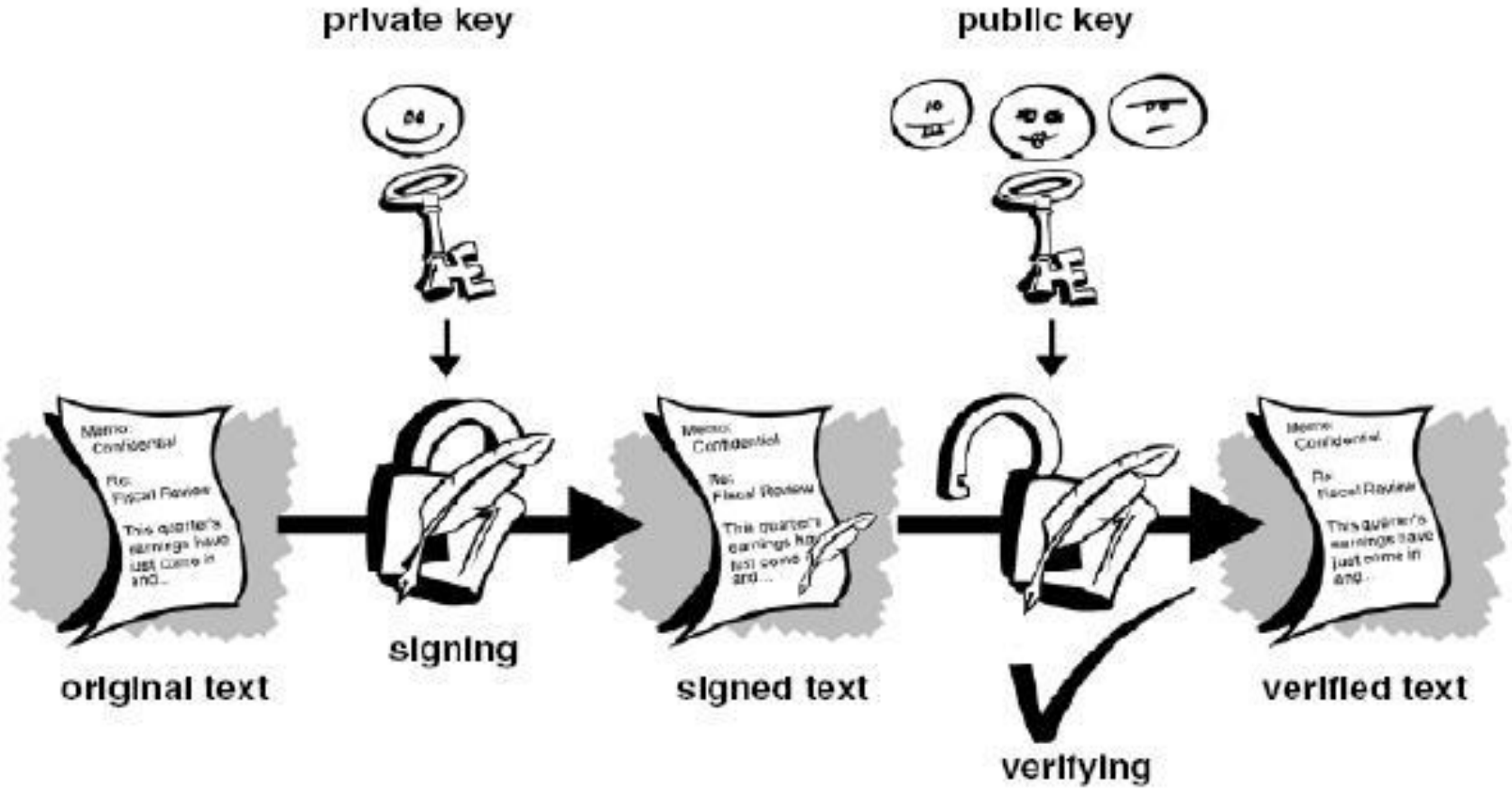
(for the example of digital signatures)

Security proofs

Four dimensions

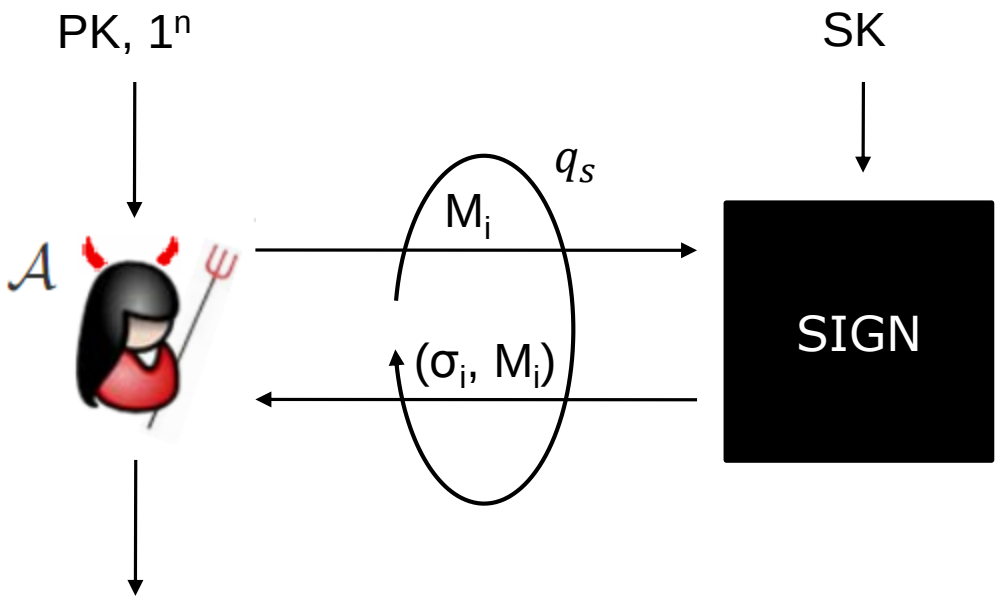
- What are we looking at?
  - Signature scheme? MAC? Key evolving signatures...
  - Determines functionality
- What do we want to prove?
  - EU-CMA, SU-CMA, FS-EU-CMA, EU-RMA...
  - Determines security guarantee
- In which model can we give the proof?
  - Standard model vs ROM vs QROM
  - Heuristic model or not
- Can we get a tight proof?
  - Reduction loss
  - Does proof say something for concrete parameters

Digital Signature



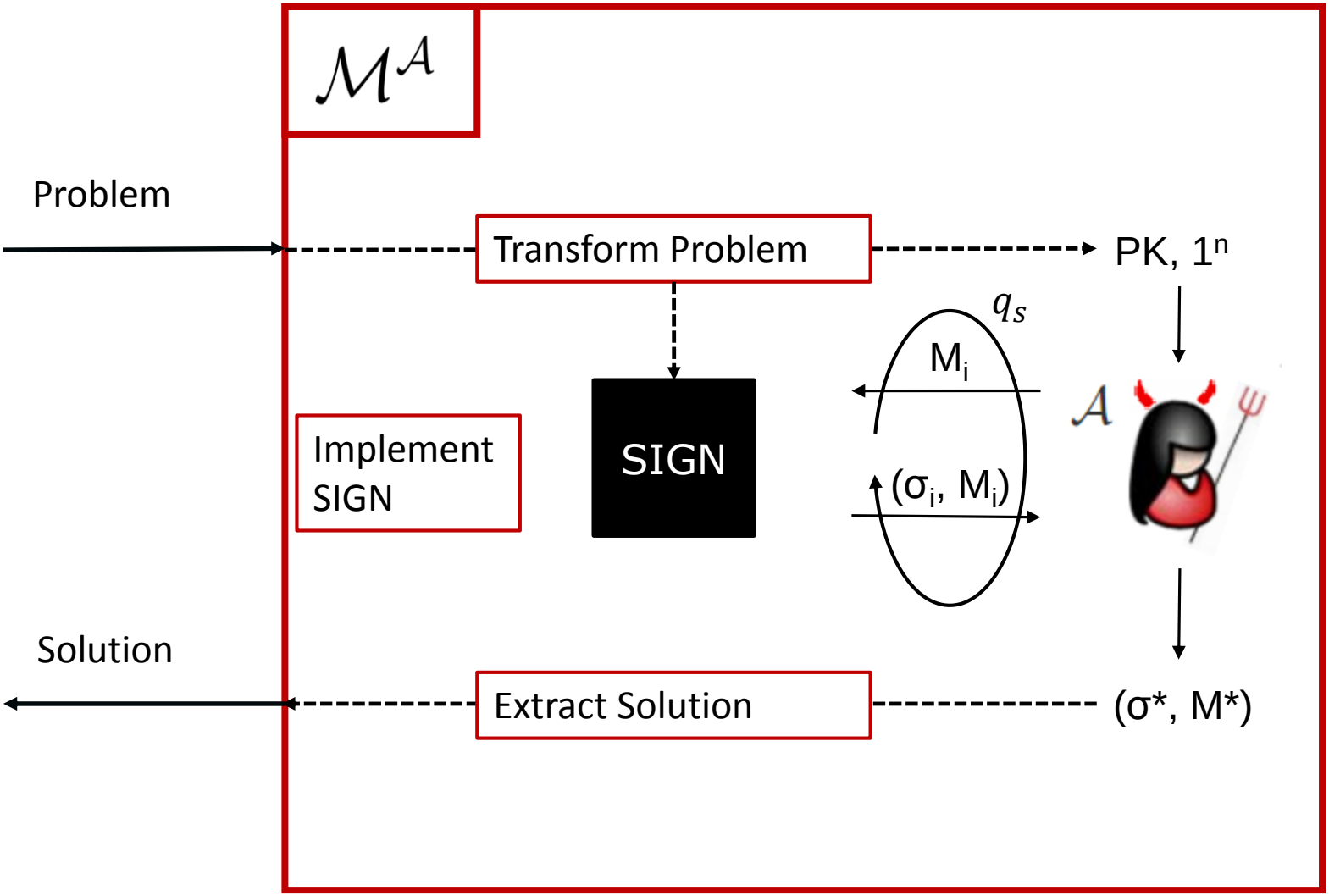
Source: http://hari-cio-8a.blog.ugm.ac.id/files/2013/03/DSA.jpg

Existential unforgeability under adaptive chosen message attacks



Success if $M^* \neq M_i$, $\forall i \in [0, q]$ and Verify(pk,σ*,M*) = Accept

Reduction


$\mathcal{M}^{\mathcal{A}}$
Problem
Transform Problem
PK, 1^n
q_s
M_i
$\mathcal{A}$
Implement SIGN
SIGN
(σ_i, M_i)
Solution
Extract Solution
(σ^*, M^*)

Reduction: Implications

To prove:

„If A can break [security] of [scheme] in time t with probability ε then M^A can solve [problem] in time $f_1(t)$ with probability $f_2(\varepsilon)$.”

Implications:

1. If f_1, f_2 are polynomial functions *(non-tight)*:
   “If [problem] is hard, [scheme] is secure.”
2. If f_1, f_2 are close to identity *(tight)*:
   “The [scheme] is as hard to break as the problem.”

Standard model vs. idealized model

Standard model:

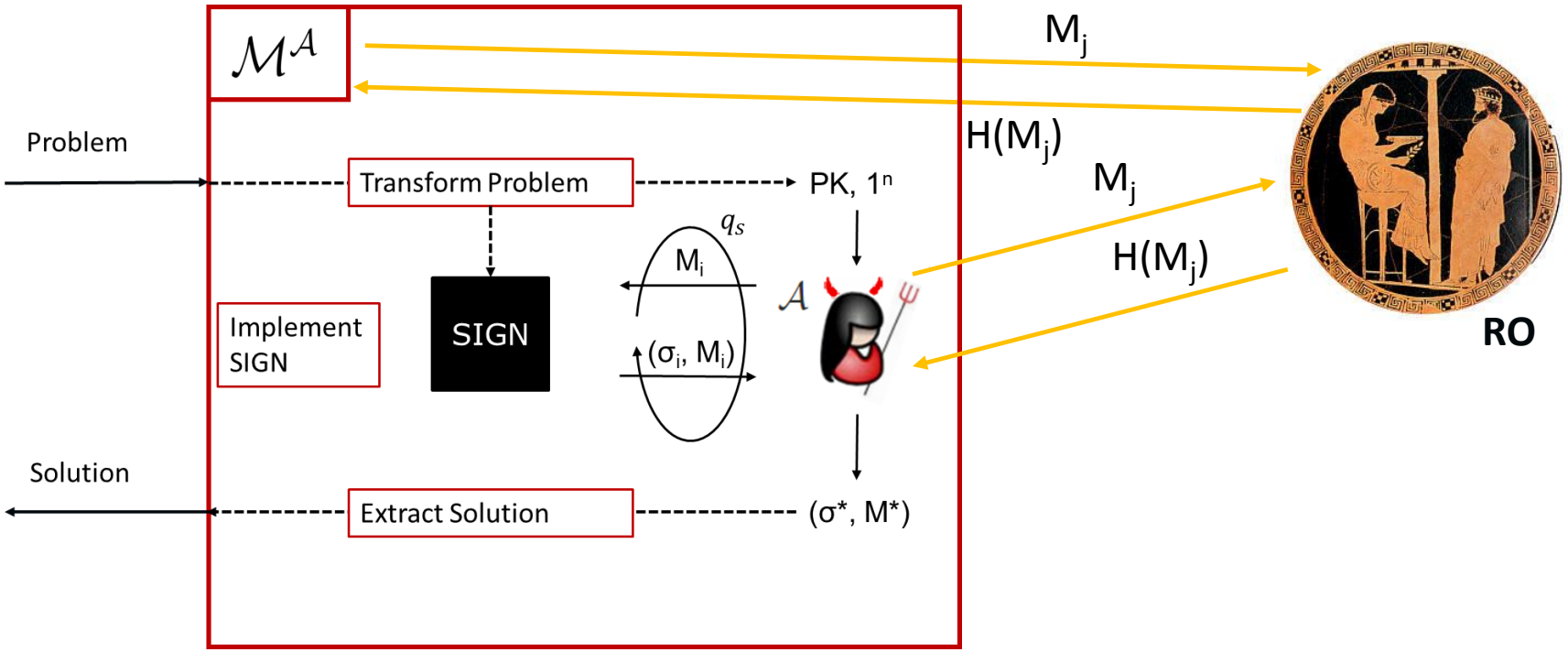
Assume building block has property P (e.g., collision resistance). Use property in reduction.

Idealized model:

Assume a building block behaves perfectly (e.g. hash function behaves like truely random function). Replace building block by an oracle in reduction.

Random Oracle Model (ROM)

- Idealized Model
- Perfectly Random Function



How to implement RO? (In theory)

”Lazy Sampling”:

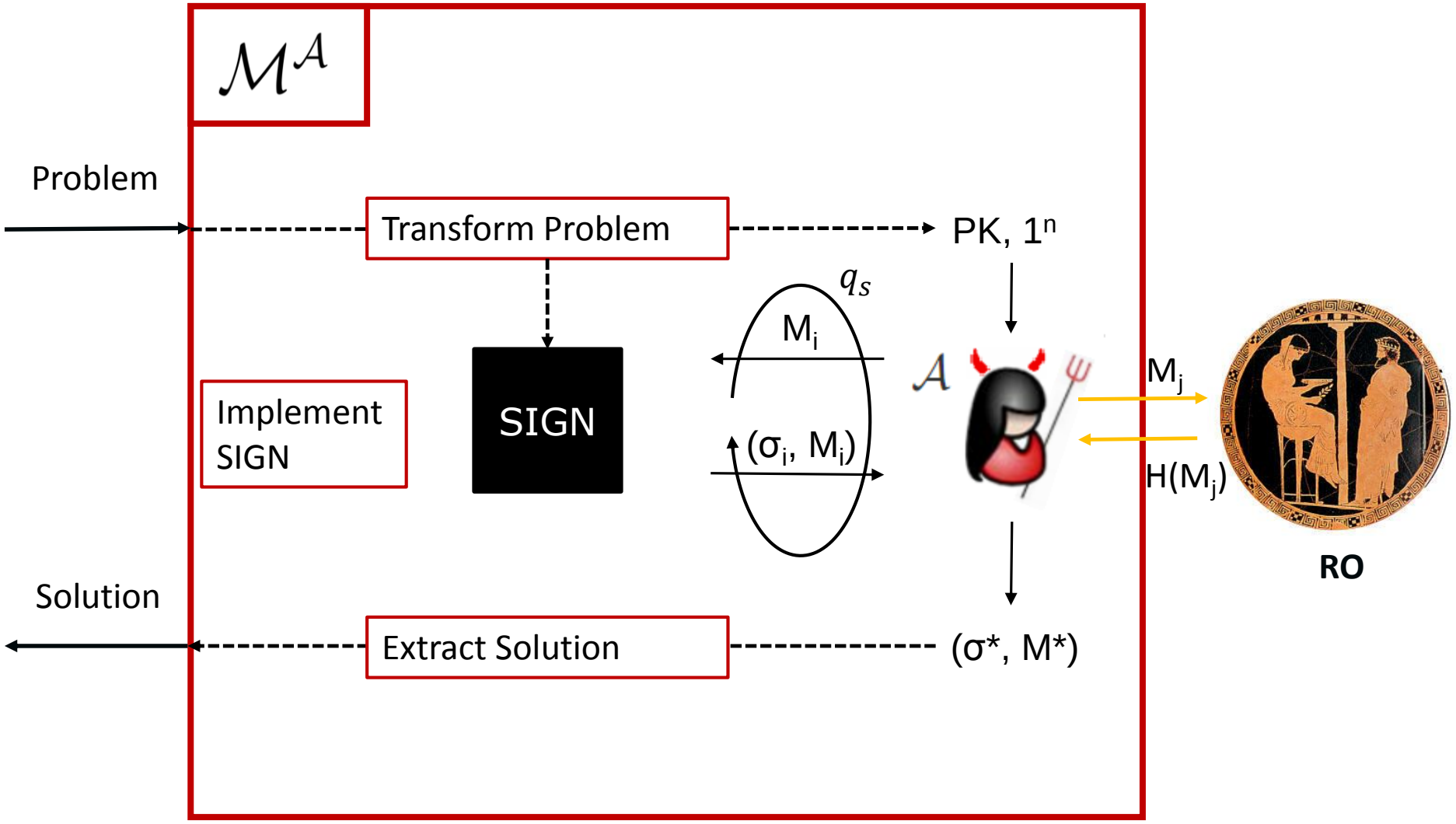
1. Keep list of (x_i, y_i)
2. Given M_j, search for $x_i = M_j$
3. If $x_i = M_j$ exists, return y_i
4. Else sample new y from Range, using uniform distribution
5. Add (M_j, y) to table
6. Return y

RO

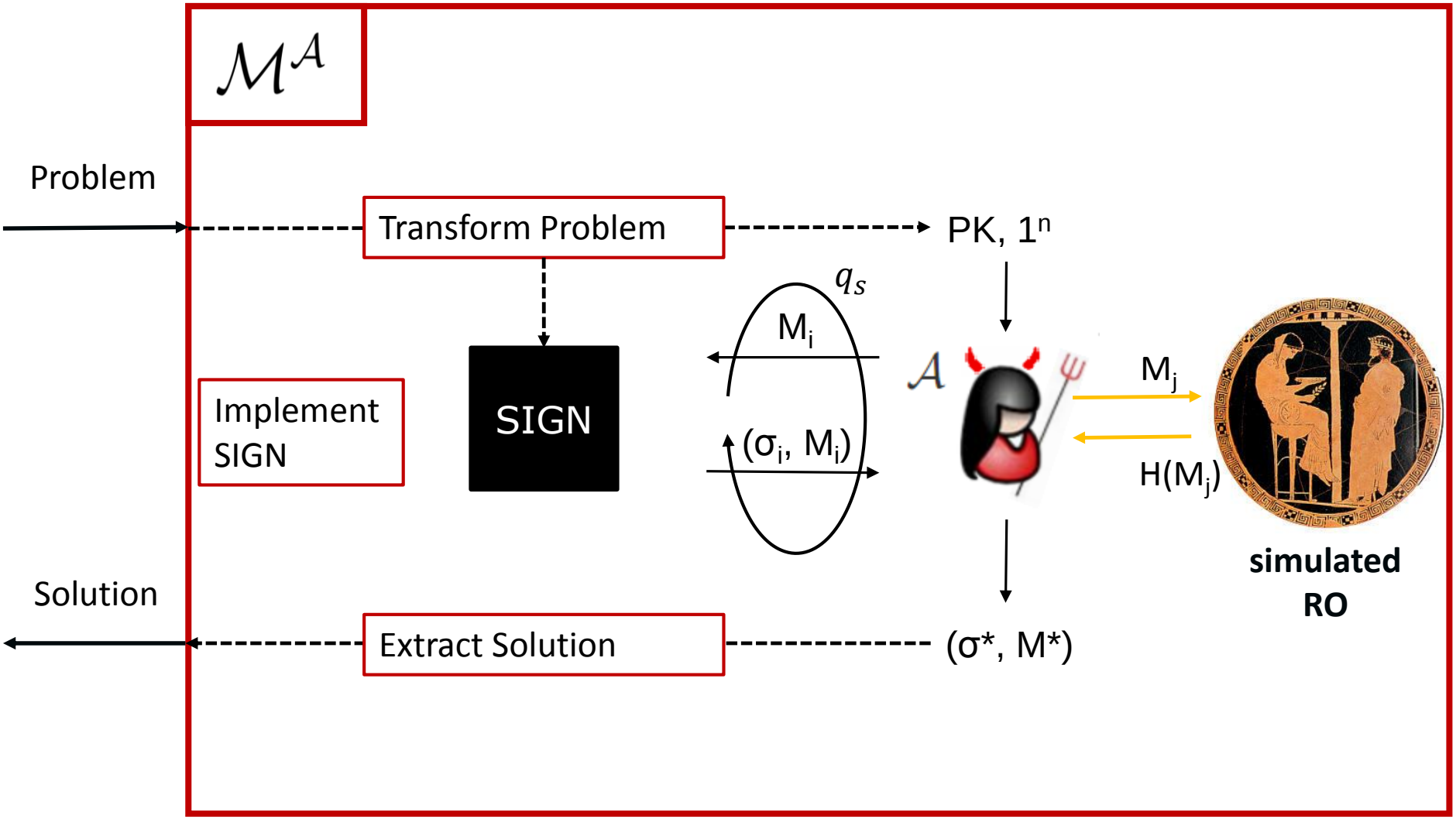
ROM security

- Take scheme that uses cryptographic hash
- For proof, replace hash by RO
  - Different flavors:
    Random function vs. Programmable RO

Programmable RO



Programmable RO



ROM security

- Take scheme that uses cryptographic hash
- For proof, replace hash by RO
  - Different flavors:
    Random function vs. Programmable RO

➢Heuristic security argument

➢Allows to verify construction

➢Worked for ”Natural schemes” so far

➢However: Artificial counter examples exist!

➢Random oracles do not exist!

Quantum security worlds

[Gagliardoni’17]

QS0: Classical security

11010101
01110110

01010101110110

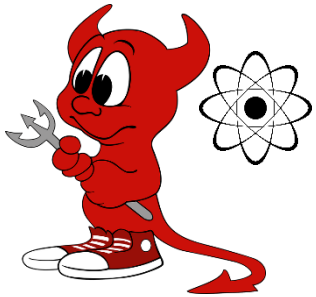
11000101
00010110

11010101
01110110

QS1: Post-quantum security

11010101
01110110

01010101110110



$|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle$
$|1\rangle|1\rangle|1\rangle|0\rangle|1\rangle|1\rangle|0\rangle$

11010101
01110110

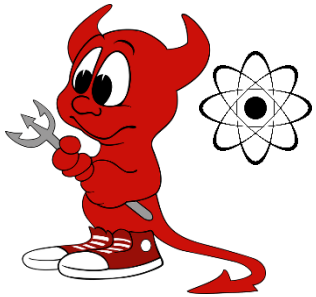
QS1: Post-quantum security

- Adversary can run local quantum computations
- Adversary cannot communicate with honest parties using quantum states!
  - Local interfaces & oracles that do not contain secret information:
    Quantum access
  - Remote interfaces & secretly keyed oracles:
    Classical access

QS2: Quantum security

$|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle$
$|1\rangle|1\rangle|1\rangle|0\rangle|1\rangle|1\rangle|0\rangle$

$|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|1\rangle|1\rangle|0\rangle|1\rangle|1\rangle|0\rangle$



$|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle$
$|1\rangle|1\rangle|1\rangle|0\rangle|1\rangle|1\rangle|0\rangle$

$|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle|1\rangle|0\rangle$
$|1\rangle|1\rangle|1\rangle|0\rangle|1\rangle|1\rangle|0\rangle$

QS2: Quantum security

- Adversary can run local quantum computations
- Adversary can communicate with honest parties using quantum states!
  - Local interfaces & oracles that do not contain secret information:
    Quantum access
  - Remote interfaces & secretly keyed oracles:
    Quantum access
- This assumes a world where users have quantum computers

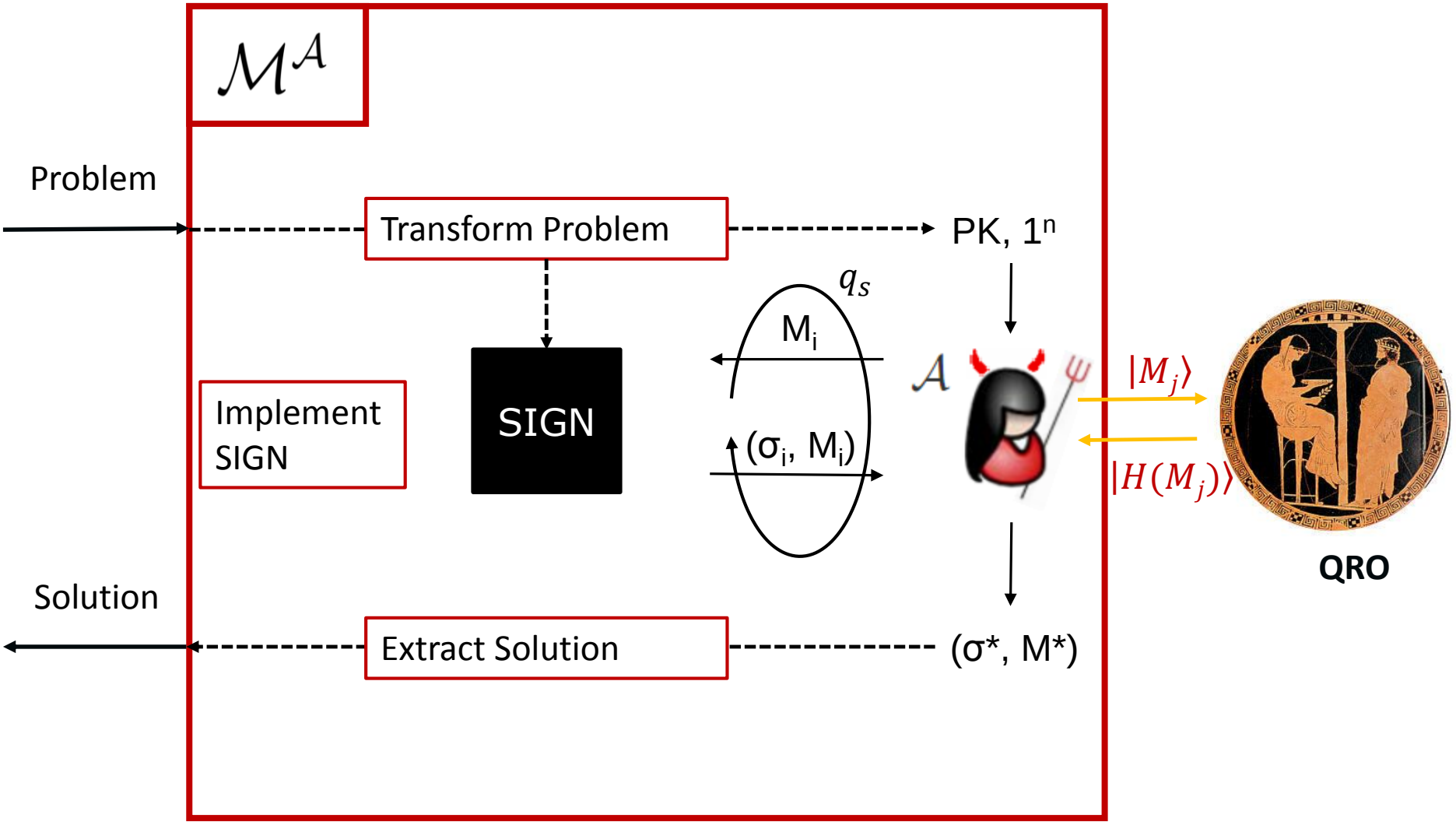
We care about QS1 for practical applications in the foreseeable future!!!

QS1: Post-quantum security

- Adversary can run local quantum computations
- Adversary cannot communicate with honest parties using quantum states!
  - **Local interfaces & oracles that do not contain secret information:**
    Quantum access
  - Remote interfaces & secretly keyed oracles:
    Classical access

What happens in ROM?

Quantum-accessible ROM (QROM)



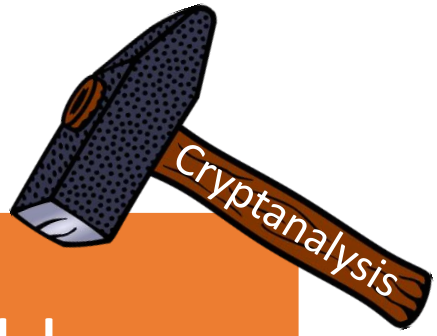
The QROM

- ROM „lives“ in QS0. Wrong model for post-quantum security!
- QROM: Adaption of ROM for QS1 – post-quantum security.
- Proofs in QROM more involved as „looking at queries“ disturbes adversary -> complicates adaptive behavior
- Only weak separation known (using squareroot speed-up of Grover)

Generic transforms

(for the example of KEM construction)

How to build PKC



(Computationally)

hard problem

RSA DL QR DDH

Simple cryptographic scheme

Proof

Problem hard

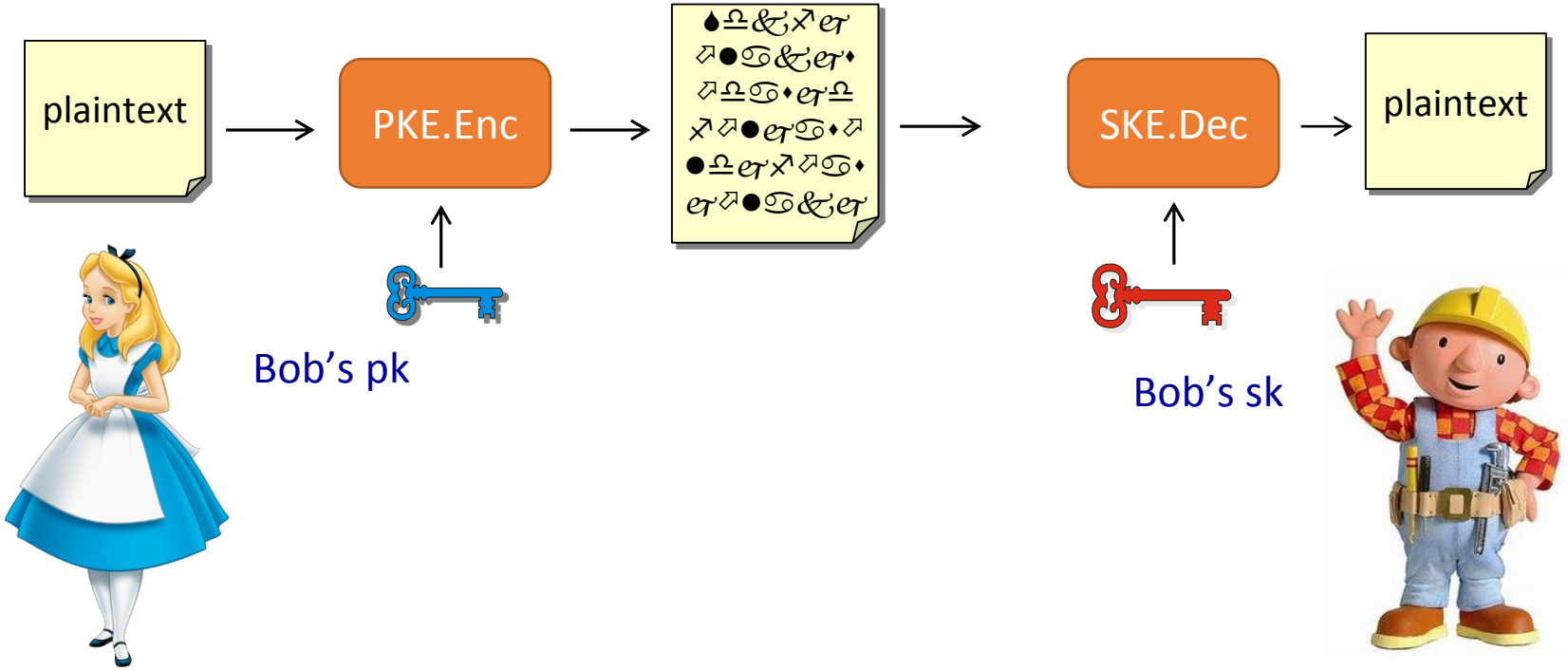
⇒

Scheme secure

PKC Scheme

RSA-OAEP ECDSA DH-KE

Public key encryption (PKE)



Public key encryption (PKE)

$\mathrm{Gen}(1^n)$: is a probabilistic algorithm that outputs a key pair (sk, pk).

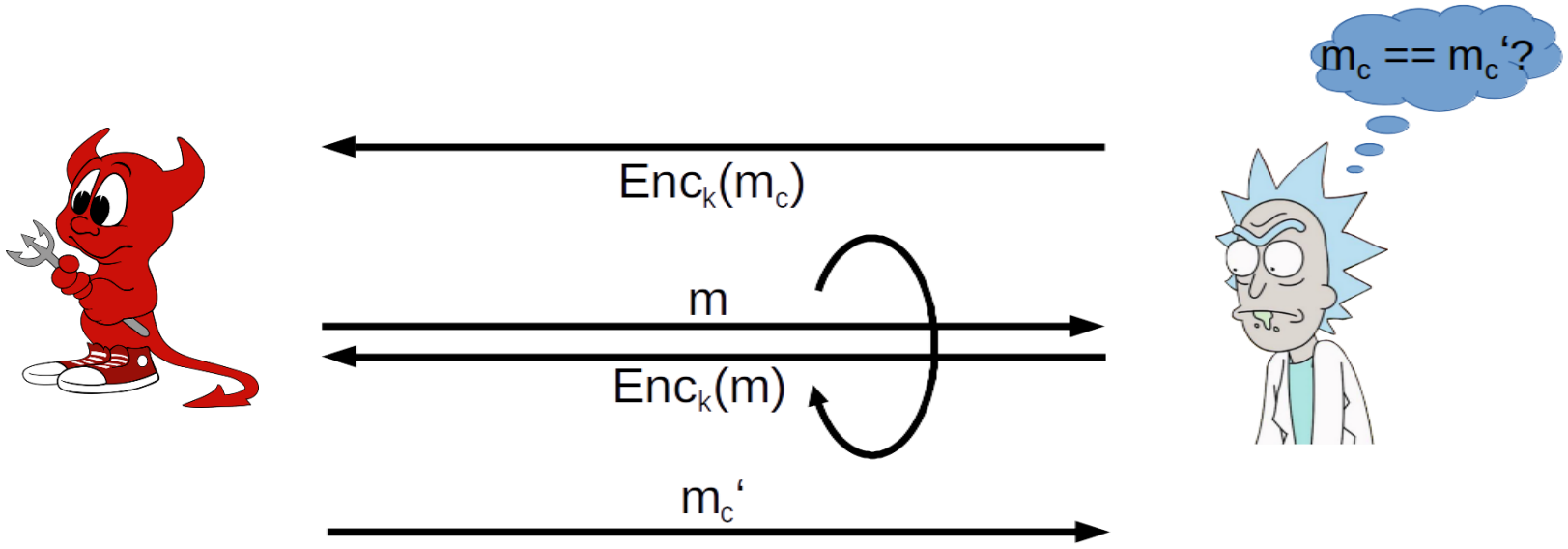
$\mathrm{Enc}(pk, m)$: is a possibly probabilistic algorithm that on input of a public key and a message outputs a ciphertext $c = \mathrm{Enc}_{pk}(m)$.

$\mathrm{Dec}(sk, c)$: is a deterministic algorithm that on input of a secret key and a ciphertext outputs a plaintext $m' = \mathrm{Dec}_{sk}(c)$.

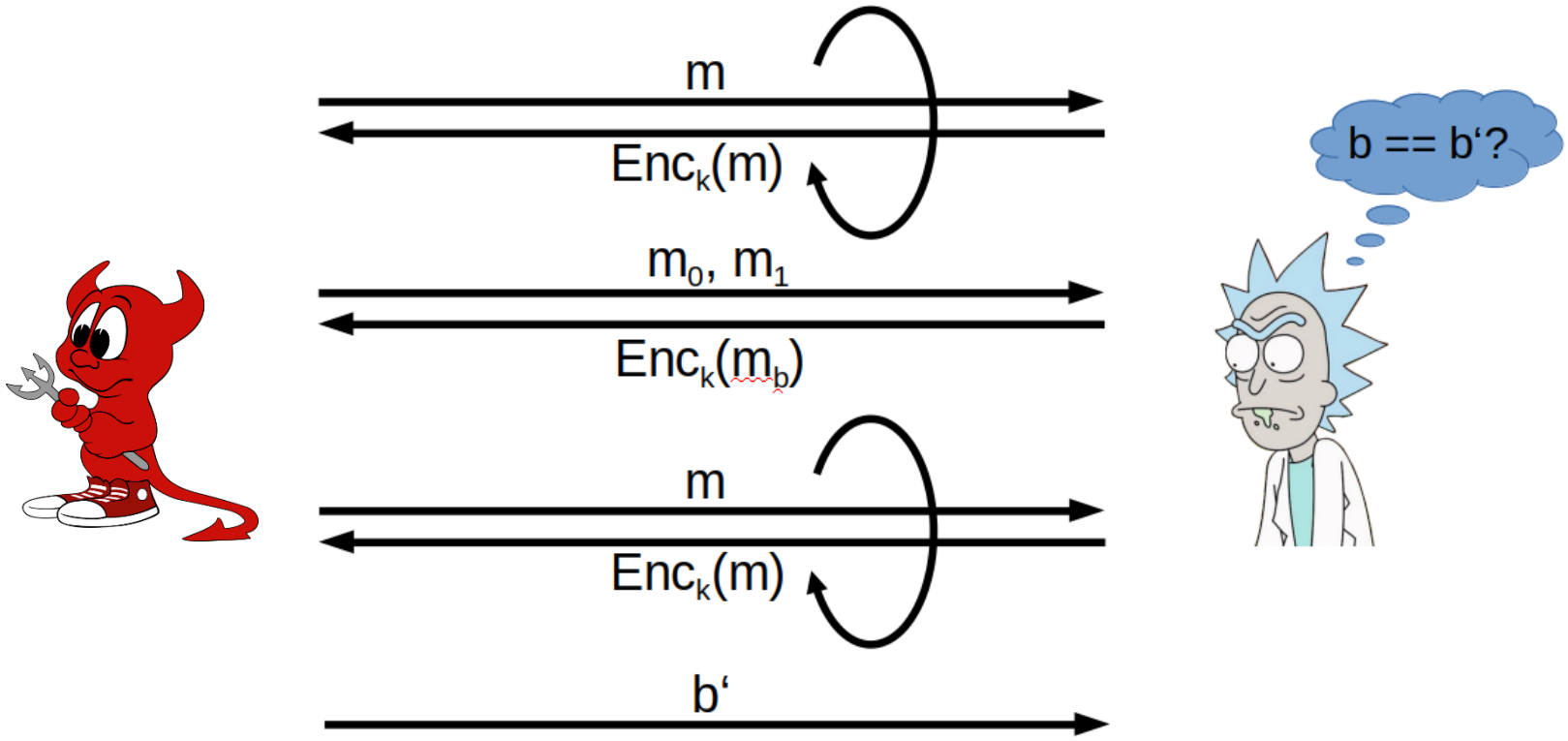
Correctness: $\left(\forall m \in M, (sk, pk) \in Gen(1^n)\right)$:

$$\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m)) = m$$

Weak passiv security (OW-CPA)



Strong passiv security (IND-CPA)

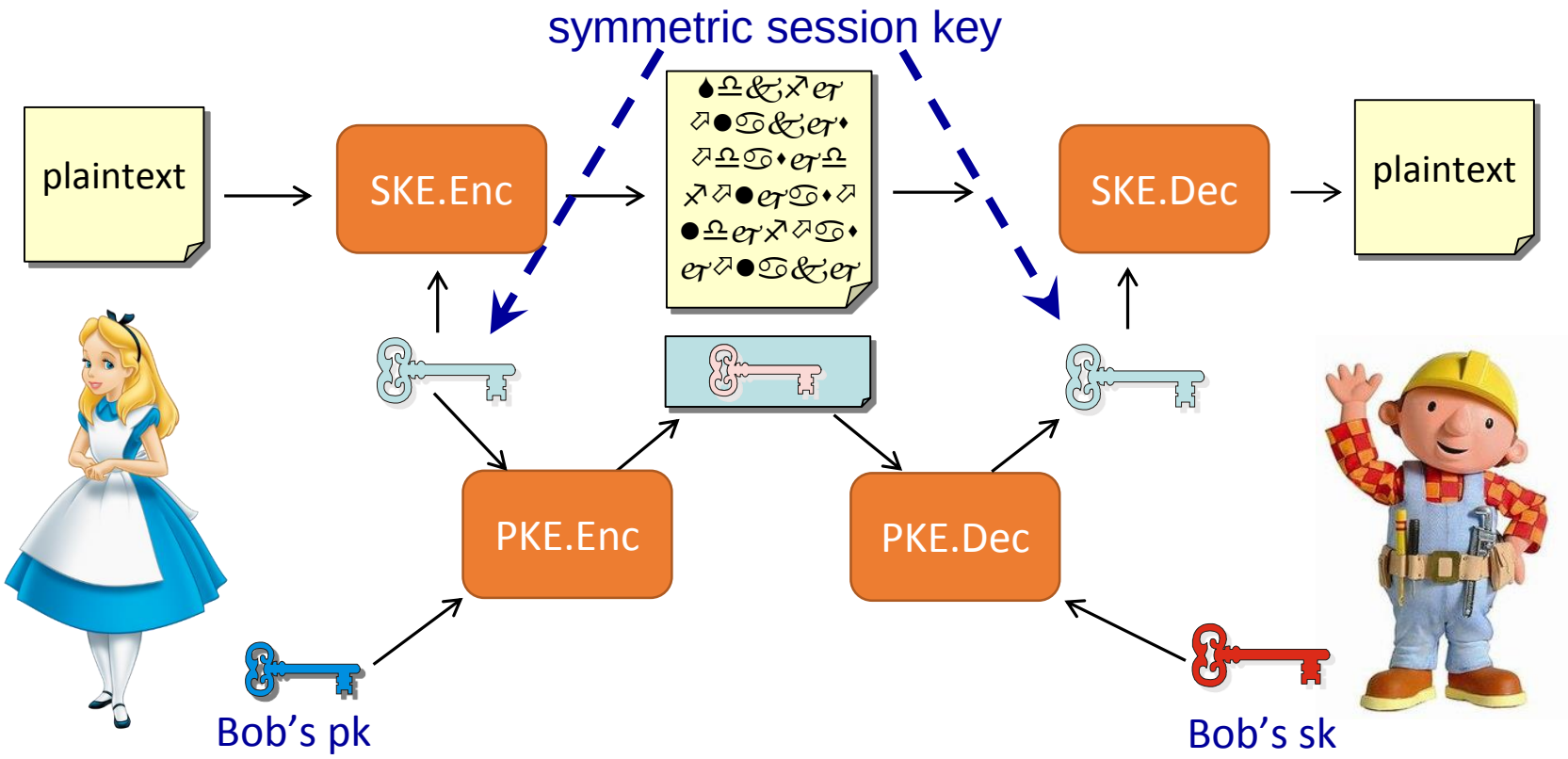


Active security (IND-CCA)

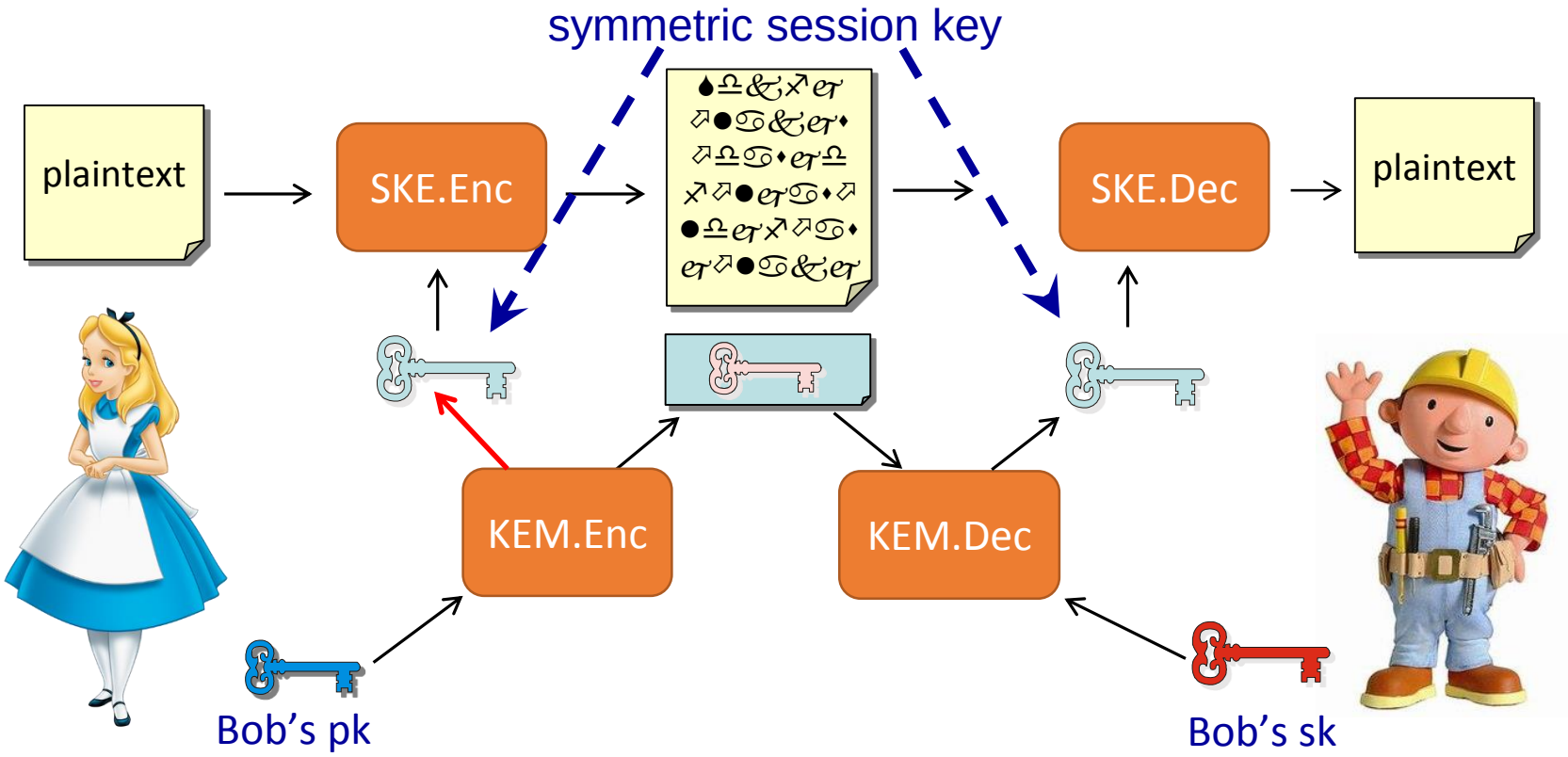
- IND-CPA + access to decryption oracle
- Decryption query for challenge ciphertext forbidden

- Necessary if scheme gets used with non-ephemeral keys!
- In practice, binary response (valid / invalid ciphertext) can be sufficient for attack.

Hybrid encryption



Key encapsulation (KEM)



Key encapsulation (KEM)

$\mathrm{Gen}(1^n)$: is a probabilistic algorithm that outputs a key pair (sk, pk).

$\mathrm{Enc}(pk, m)$: is a possibly probabilistic algorithm that on input of a public key outputs a session key k and an encapsulation $(k, c) = \mathrm{Enc}_{pk}()$.

$\mathrm{Dec}(sk, c)$: is a deterministic algorithm that on input of a secret key and an encapsulation outputs a session key $k' = \mathrm{Dec}_{sk}(c)$.

Correctness: $\left(\forall (sk, pk) \in Gen(1^n)\right)$:

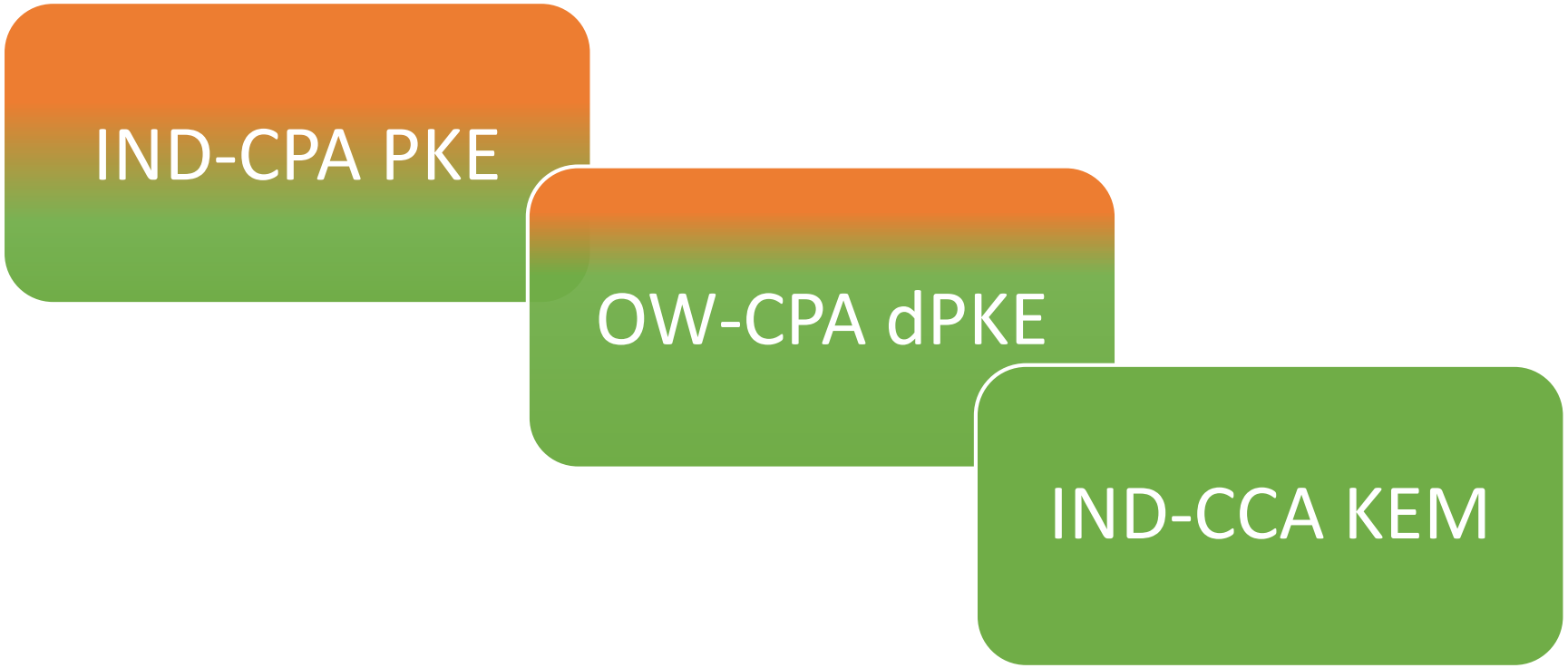
$$(k, c) = \mathrm{Enc}_{pk}(\quad) \Rightarrow k = \mathrm{Dec}_{sk}(c)$$

IND security models for KEM

- IND:
  - Given a pair (c, k) decide if c is encapsulation of k or random
- CPA:
  - Adversary gets pk and can hence generat encapsulations
- CCA:
  - Adversary gets access to decapsulation oracle (which returns $\perp$ for challenge encapsulation)

IND-CPA PKE -> IND-CCA KEM

- Generic transform essentially going back to Fujisaki and Okamoto, CRYPTO‘99



IND-CPA PKE -> OW-CPA dPKE

T-Transform [HHK17]

Start from IND-CPA secure $\mathrm{PKE} = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ with probabilistic Enc and a hash function H. Build OW-CPA secure

$$\mathrm{PKE}' = (\mathrm{Gen}, \mathrm{Enc}', \mathrm{Dec}) = \mathrm{T}(\mathrm{PKE}, \mathrm{H})$$

with deterministic Enc':

$$\mathrm{Enc}'_{pk}(\mathrm{m}) = \mathrm{Enc}_{pk}(m; \mathrm{H}(m))$$

Randomness

- Tight security reduction in ROM
- Non-tight security reduction in QROM

OW-CPA dPKE -> IND-CCA KEM

$U^{\not\perp}$-Transform [HHK17]

Start from OW-CPA secure dPKE $\mathsf{P} = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$, PRF F, and hash function H. Build IND-CCA secure KEM $U^{\not\perp}$(P,F,H):

Keygen():	Encaps(pk):	Decaps(sk, c):
1 $(\mathrm{pk}, \mathrm{sk}_\mathsf{P}) \leftarrow \mathrm{Keygen}_\mathsf{P}()$	1 $m \xleftarrow{\$} \mathcal{M}$	1 parse $\mathrm{sk} = (\mathrm{sk}_\mathsf{P}, \mathrm{prfk})$
2 $\mathrm{prfk} \xleftarrow{\$} \mathcal{K}_\mathsf{F}$	2 $c \leftarrow \mathrm{Encr}(\mathrm{pk}, m)$	2 $m' \leftarrow \mathrm{Decr}(\mathrm{sk}_\mathsf{P}, c)$
3 $\mathrm{sk} \leftarrow (\mathrm{sk}_\mathsf{P}, \mathrm{prfk})$	3 $K \leftarrow H(m, c)$	3 if $m' = \perp$:
4 return $(\mathrm{pk}, \mathrm{sk})$	4 return (K, c)	4 return $\mathsf{F}(\mathrm{prfk}, c)$
		5 else if $\mathrm{Encr}(\mathrm{pk}, m') \neq c$):
		6 return $\mathsf{F}(\mathrm{prfk}, c)$
		7 else: return $H(m', c)$

Fig. 3. Transform $U^{\not\perp}(\mathsf{P}, \mathsf{F}) := (\mathrm{Keygen}, \mathrm{Encaps}, \mathrm{Decaps})$.

Summary

- Security proof ≠ security proof
  - Only tight proofs say something about the security of parameters
  - Only standard model & QROM proofs say something about post-quantum security
- Generic transforms can save you a lot of work
  - Sufficient to construct OW-CPA dPKE for IND-CCA KEM with tight QROM proof
  - Sufficient to construct IND-CPA PKE for IND-CCA KEM with QROM proof

Thank you!
Questions?