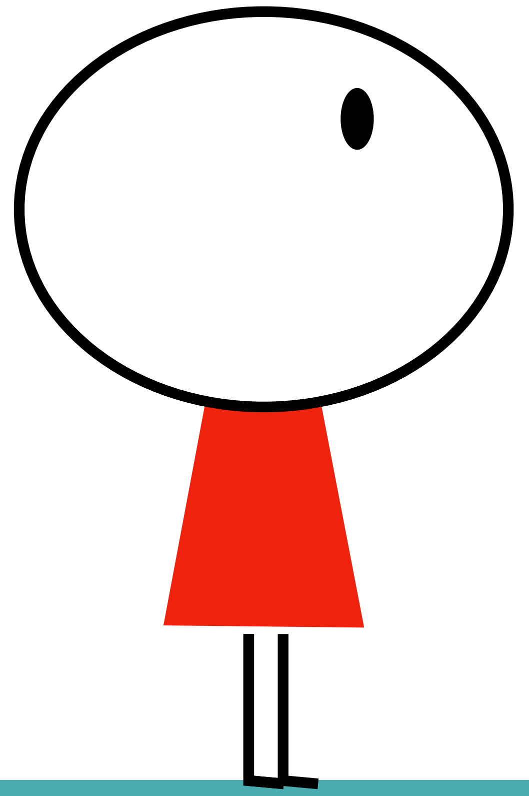
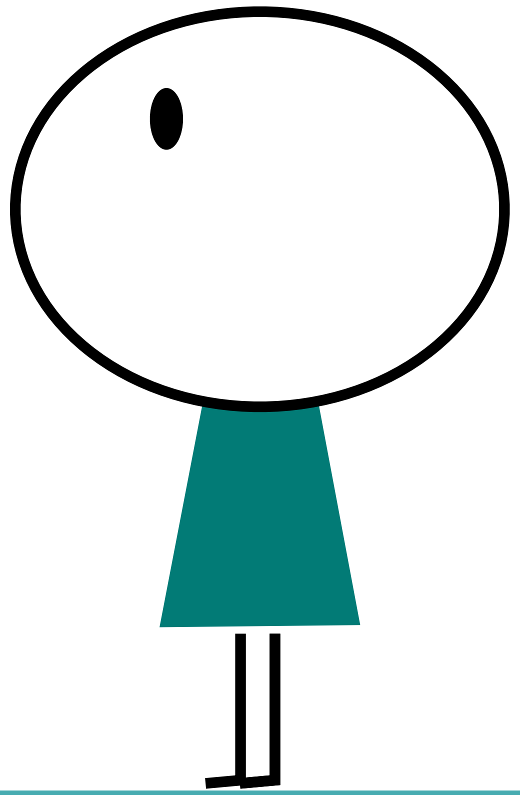
Side-channel countermeasures for lattice-based cryptography

- VeriSiCC Seminar -

Sept 22nd 2022





Mélissa Rossi

Lattices and hard problems

A lattice Λ is an additive subgroup generated by n linearly independent vectors of $\mathbb{R}^n$.

Lattices and hard problems

A lattice Λ is an additive subgroup generated by n linearly independent vectors of $\mathbb{R}^n$.

Lattices and hard problems

A lattice Λ is an additive subgroup generated by n linearly independent vectors of $\mathbb{R}^n$.

Lattices and hard problems

Given a lattice Λ

Find the vector $\mathbf{v}$ that has the smallest nonzero norm

Short Vector Problem (SVP)

Lattices and hard problems

« Linear system solving with noise »

Given the pair $\left(\mathbf{A} \in \mathbb{Z}_q^{m\times n}, \mathbf{b} = \mathbf{A}\mathbf{s}+\mathbf{e} \in \mathbb{Z}_q^m\right)$ where

$\mathbf{A}$ is sampled uniformly at random

$\mathbf{e}$ and $\mathbf{s}$ are sampled following a small distribution χ

Find $\mathbf{s}$

Learning With Errors (LWE)

Lattice-based algorithms

Signature schemes

Public key encryption schemes

1 Strong hardness properties

Lattice-based algorithms

Signature schemes

Public key encryption schemes

1 Strong hardness properties

2 Simple designs (but complex analysis)

Lattice-based algorithms

Signature schemes

Public key encryption schemes

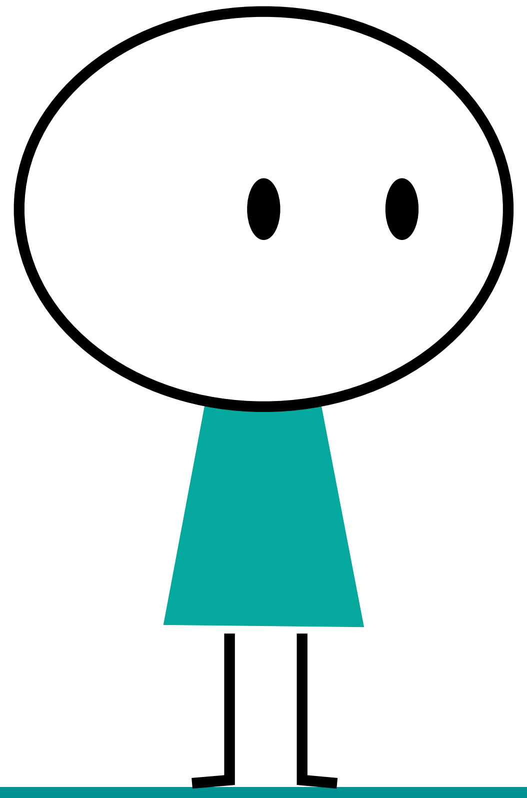
1 Strong hardness properties

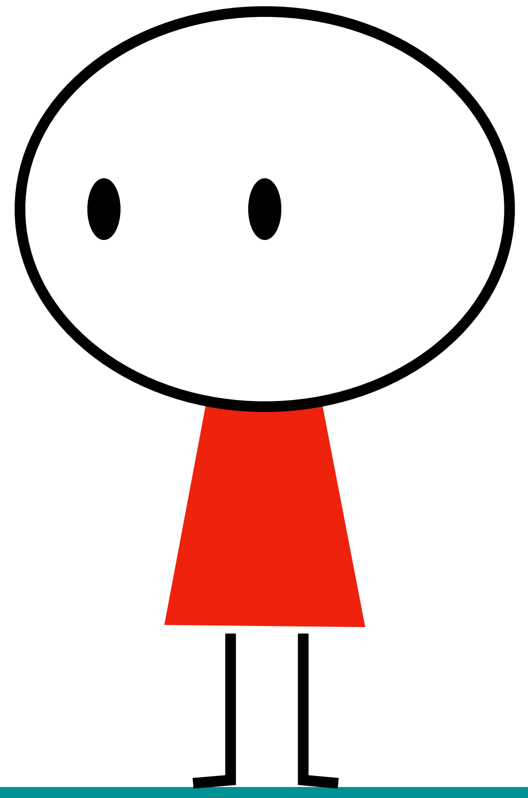
2 Simple designs (but complex analysis)

3 Concrete candidates schemes
NIST round 2: 12 out of 26 candidates
NIST round 3: 5 out of 7 candidates
NIST first standards: 3
NIST round 4: ?

LWE-based public key encryption in a nutshell

- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

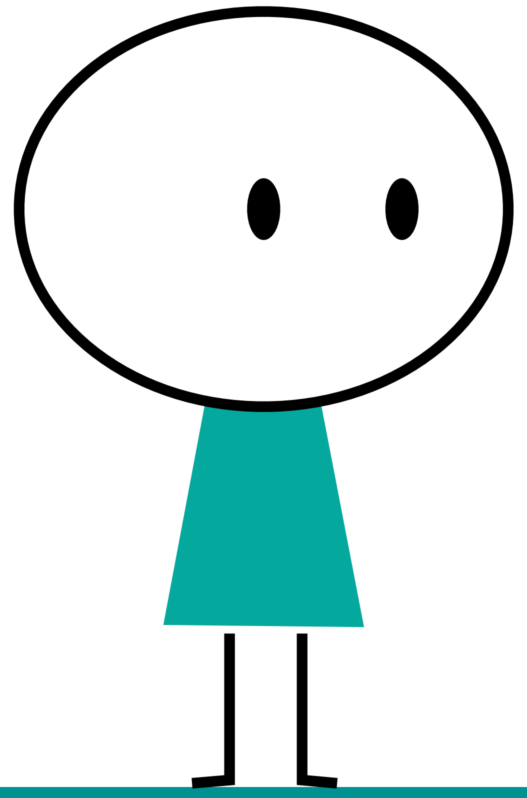


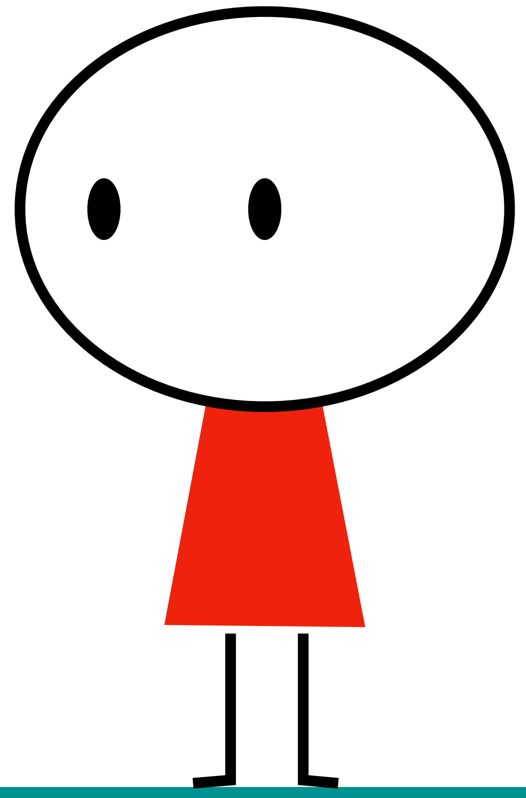


LWE-based public key encryption in a nutshell

- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

$$\left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z} + \mathbf{e}\right)$$

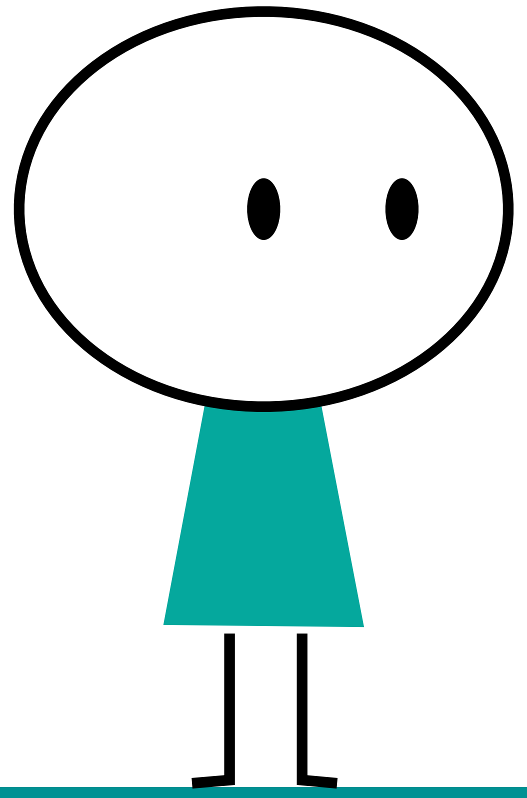


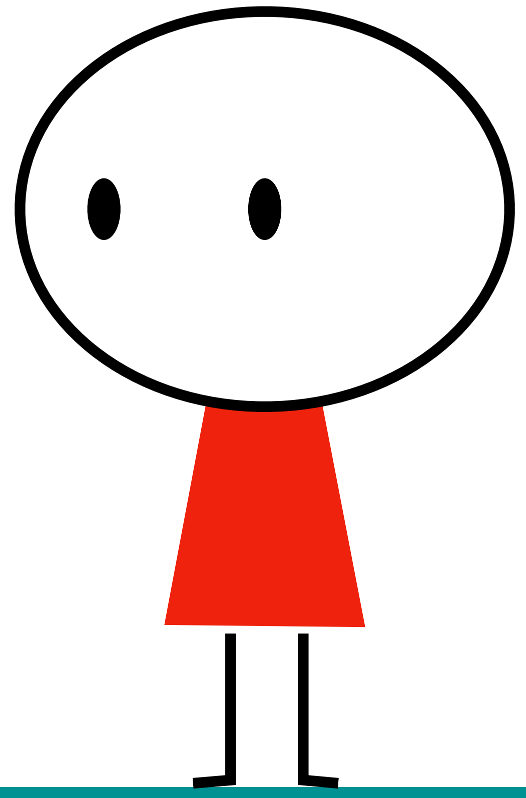


LWE-based public key encryption in a nutshell

- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

$$(\mathbf{A}, \mathbf{b}) \longleftarrow \left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z} + \mathbf{e}\right)$$





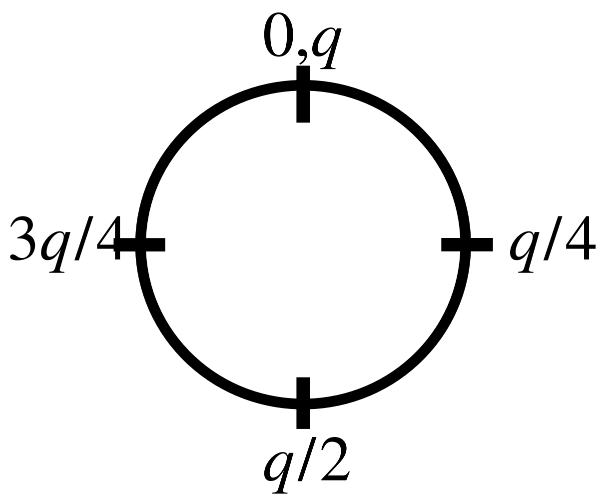
LWE-based public key encryption in a nutshell

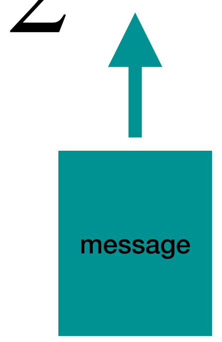
- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

$$(\mathbf{A}, \mathbf{b})$$

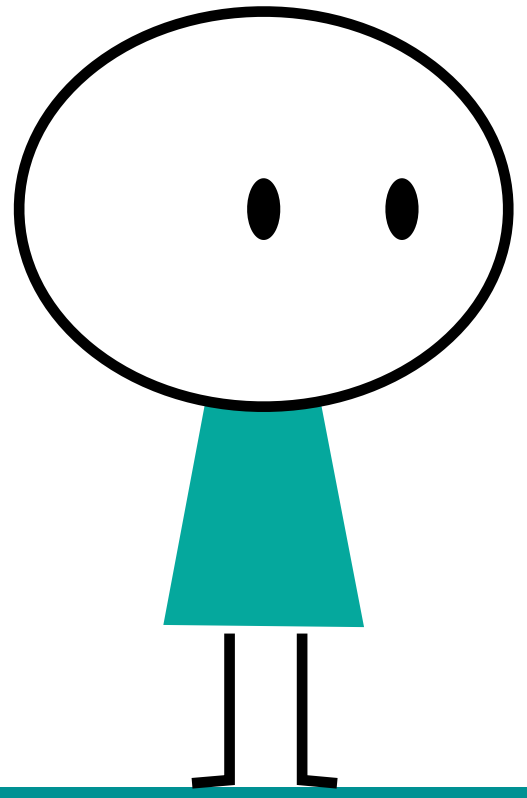
$$\mathbf{b}' = \mathbf{A}^T\mathbf{z}' + \mathbf{e}' \qquad \longleftarrow \qquad \left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z} + \mathbf{e}\right)$$

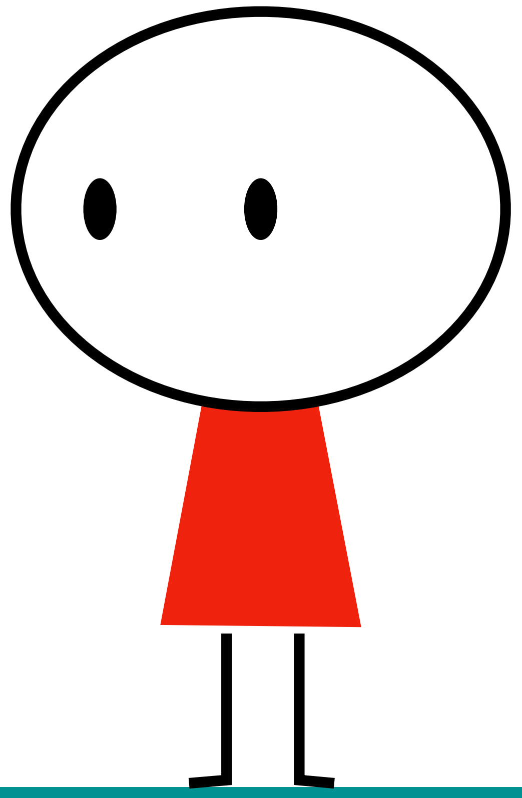
$$v' = \mathbf{b}^T\mathbf{z}' + \mathbf{e}'' + \frac{q}{2}m$$





1-bit encryption





LWE-based public key encryption in a nutshell

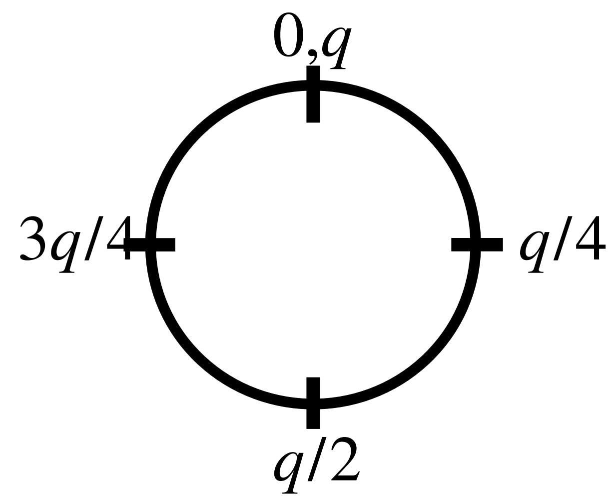
- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

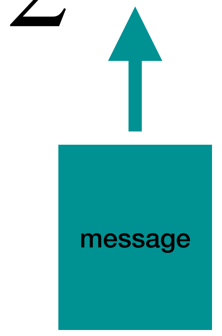
$(\mathbf{A}, \mathbf{b})$

$$\mathbf{b}' = \mathbf{A}^T\mathbf{z}' + \mathbf{e}'$$

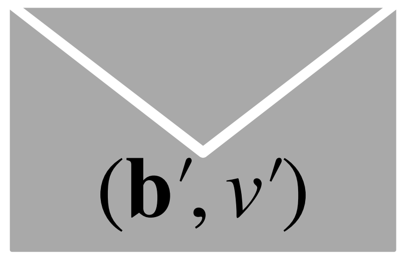
$$\left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z} + \mathbf{e}\right)$$

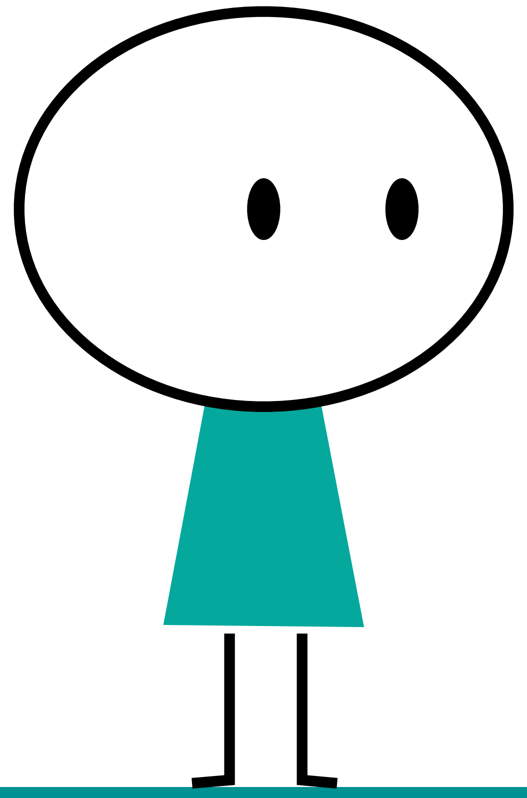
$$v' = \mathbf{b}^T\mathbf{z}' + \mathbf{e}'' + \frac{q}{2}m$$

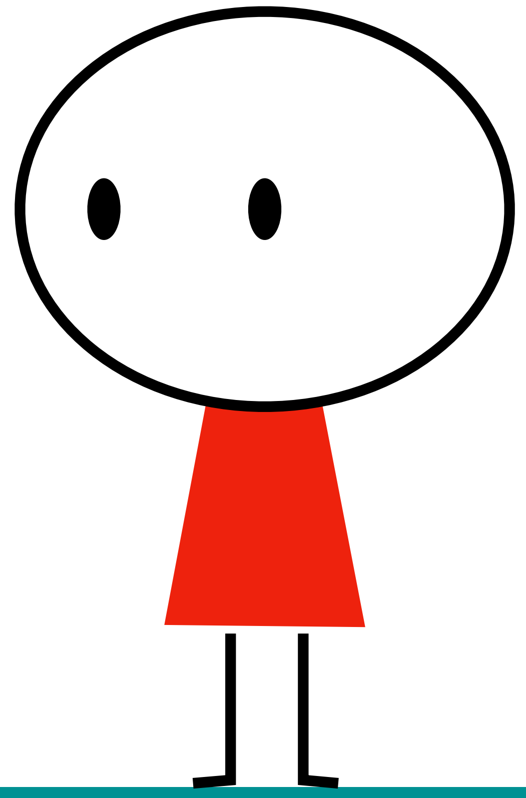




1-bit encryption







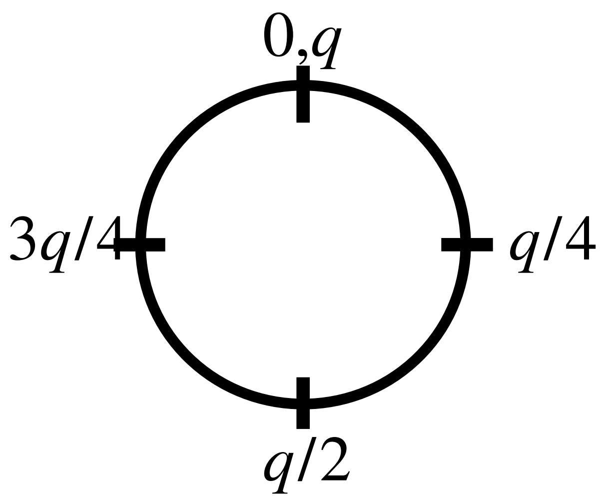
LWE-based public key encryption in a nutshell

- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

$$(\mathbf{A}, \mathbf{b}) \longleftarrow \left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z}+\mathbf{e}\right)$$

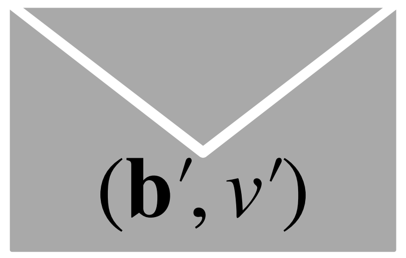
$$\mathbf{b}' = \mathbf{A}^T\mathbf{z}'+\mathbf{e}'$$

$$v' = \mathbf{b}^T\mathbf{z}'+\mathbf{e}''+\frac{q}{2}m$$



message

1-bit encryption



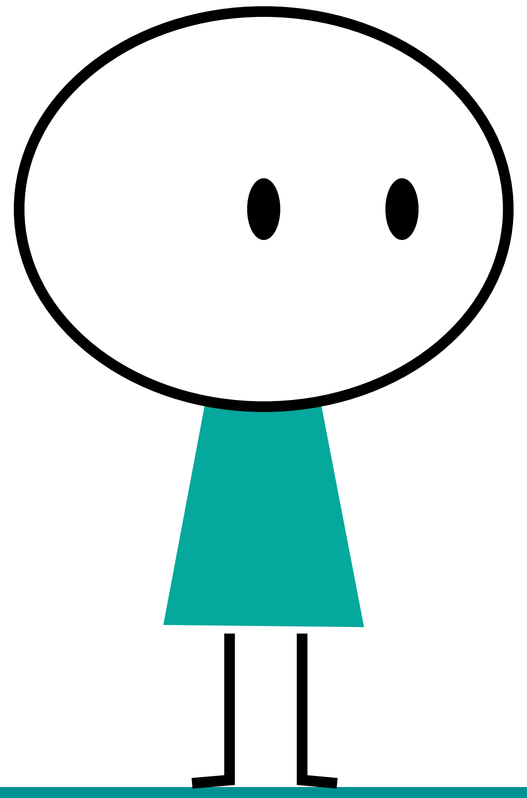
$$m' = \left\lfloor \frac{2}{q}\left(v'-\mathbf{z}^T\mathbf{b}'\right) \right\rceil$$

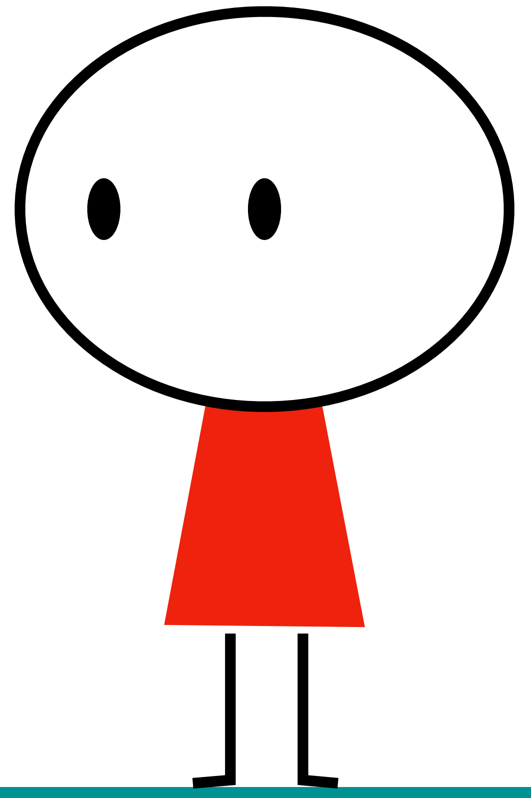
$$m' \approx m + \underbrace{\left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}'+\mathbf{e}''-\mathbf{z}^T\mathbf{e}'\right) \right\rceil}$$

message

1-bit decryption

Noise smaller than $1/2$





- J. Ding, X. Xie and X. Lin EUROCRYPT'14
- C. Peikert PQCRYPTO'14
- J. W. Bos, C. Costello, M. Naehrig and D. Stebila S&P'15
- E. Alkim, L. Ducas, T. Pöppelmann and P. Schwabe USENIX'16

LWE-based public key encryption in a nutshell

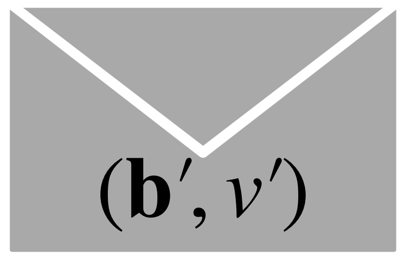
$$(\mathbf{A}, \mathbf{b})$$

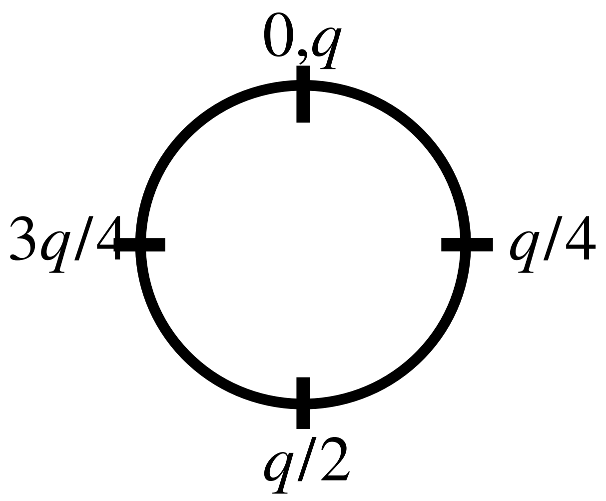
$$\mathbf{b}' = \mathbf{A}^T\mathbf{z}' + \mathbf{e}'$$

$$\left(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{z} + \mathbf{e}\right)$$

$$v' = \mathbf{b}^T\mathbf{z}' + \mathbf{e}'' + \frac{q}{2}m$$

$$m' = \left\lfloor \frac{2}{q}\left(v' - \mathbf{z}^T\mathbf{b}'\right) \right\rceil$$





message

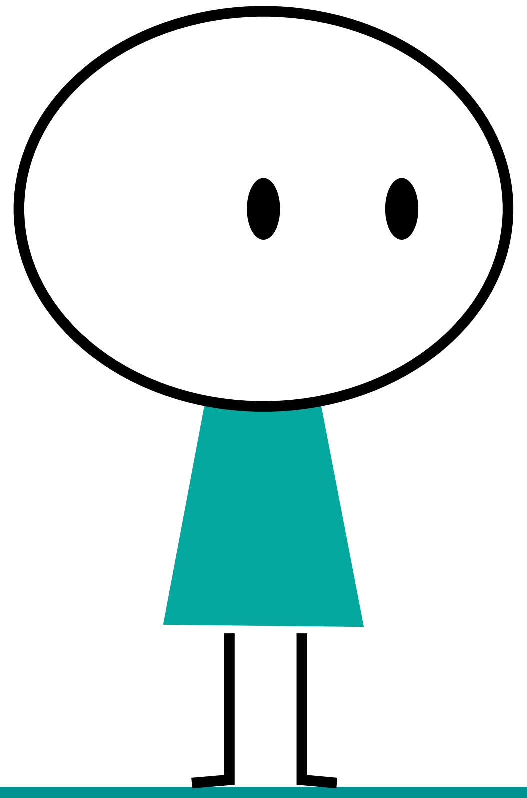
1-bit encryption

$$m' \approx m + \underbrace{\left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil}$$

message

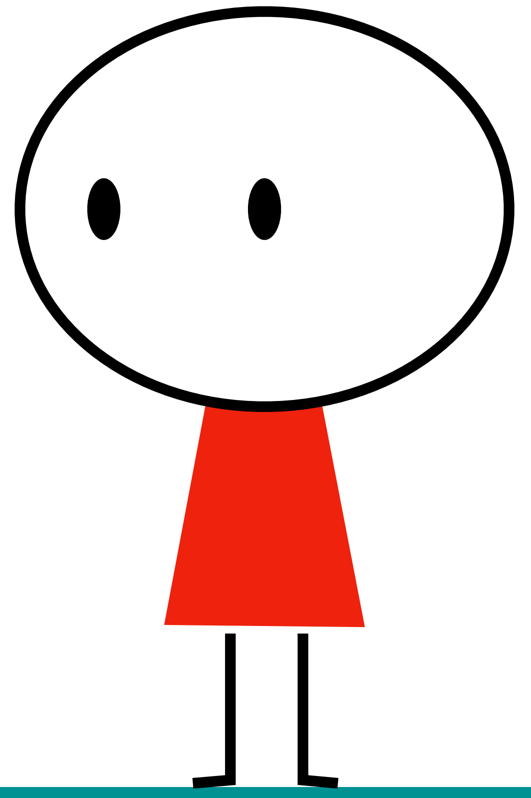
1-bit decryption

Noise smaller than $1/2$



High level idea behind

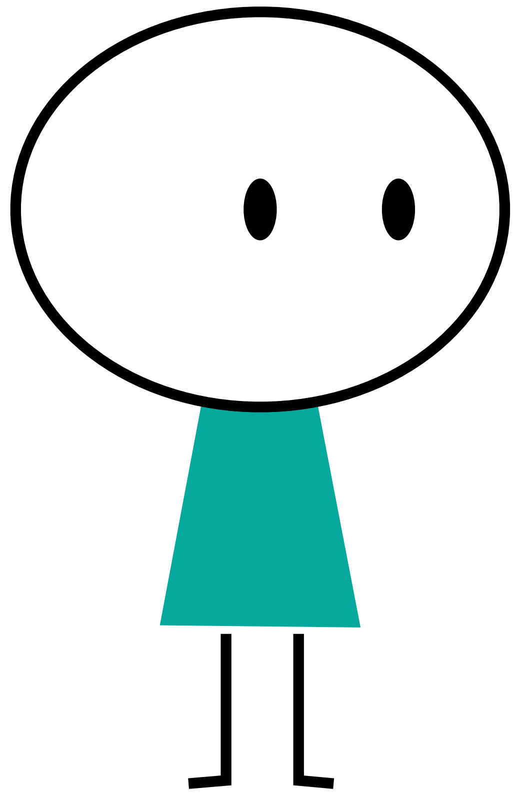
Crystals-Kyber (NIST standard)
Frodo, Saber and NewHope

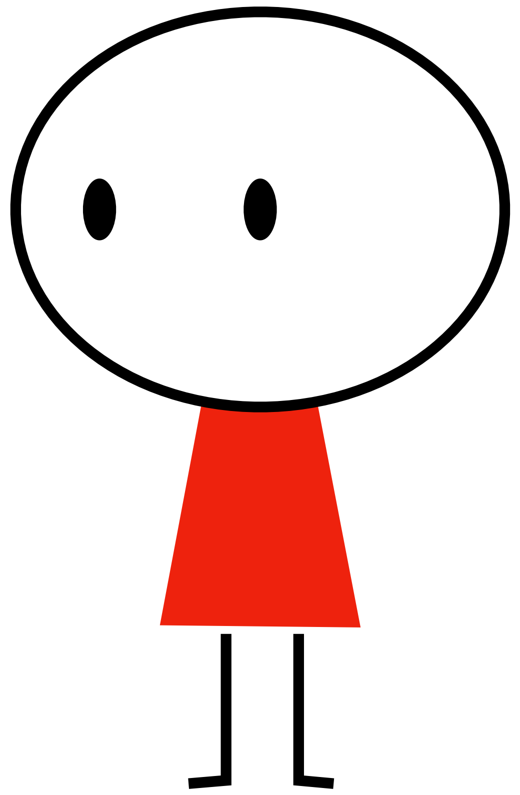


A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)



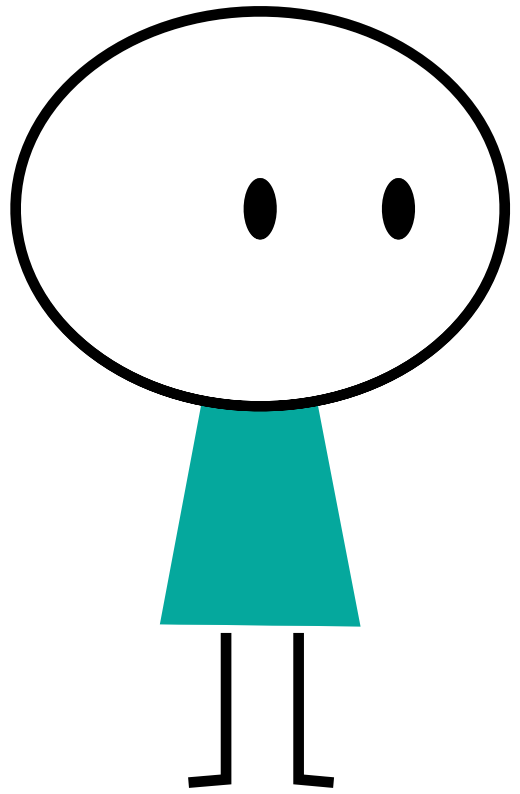


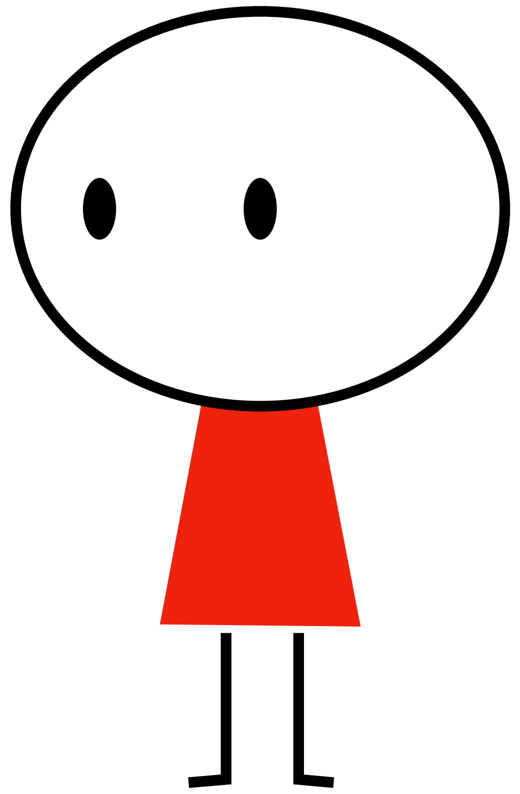
A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$$\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$$



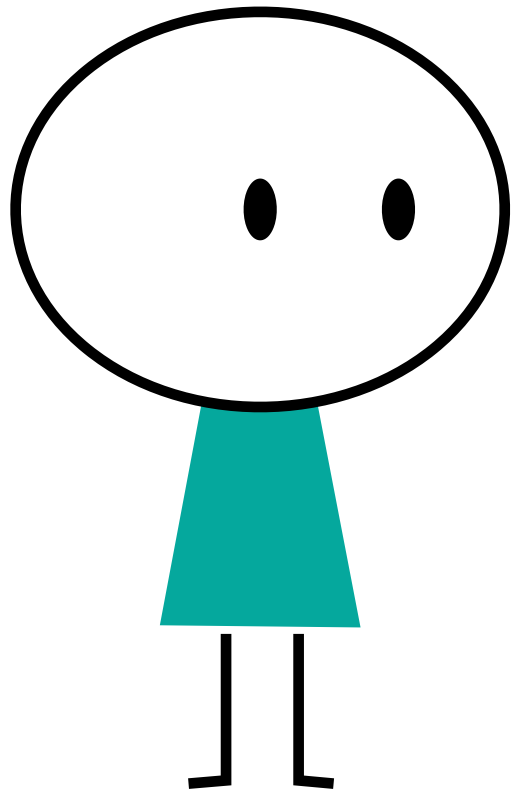


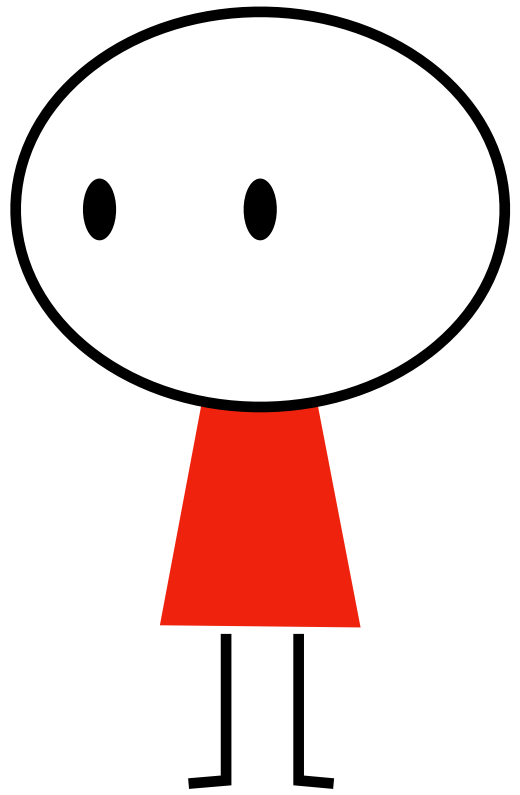
A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$$(\mathbf{A}, \mathbf{t}) \longleftarrow \left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$$





A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$$(\mathbf{A}, \mathbf{t}) \longleftarrow \left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$$

Signature algorithm:

message

1: **do**

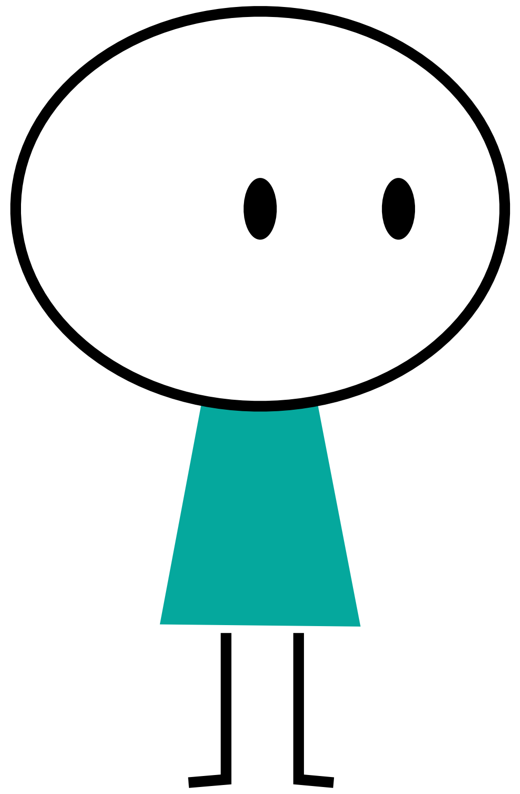
2: $\quad \mathbf{y} \overset{\$}{\leftarrow} Y$

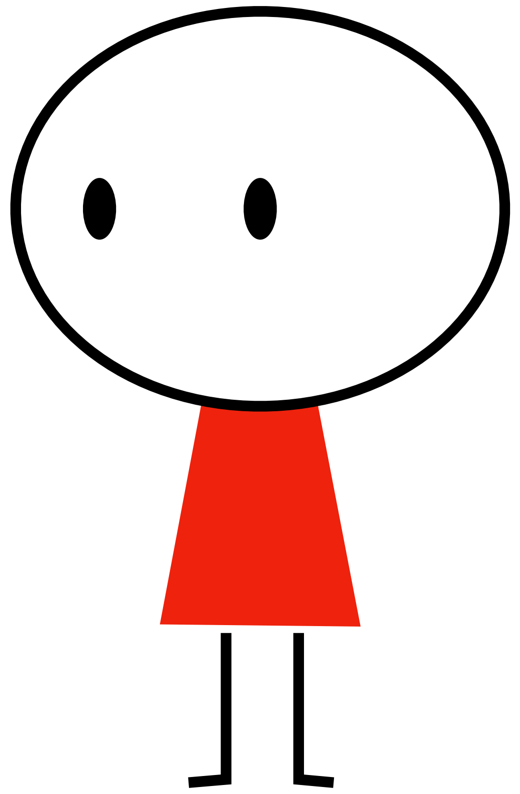
3: $\quad c \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** $(\mathbf{z}, c)$





A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$$(\mathbf{A}, \mathbf{t}) \longleftarrow \left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$$

message $(\mathbf{z}, c) \longleftarrow$

Signature algorithm: message

1: **do**

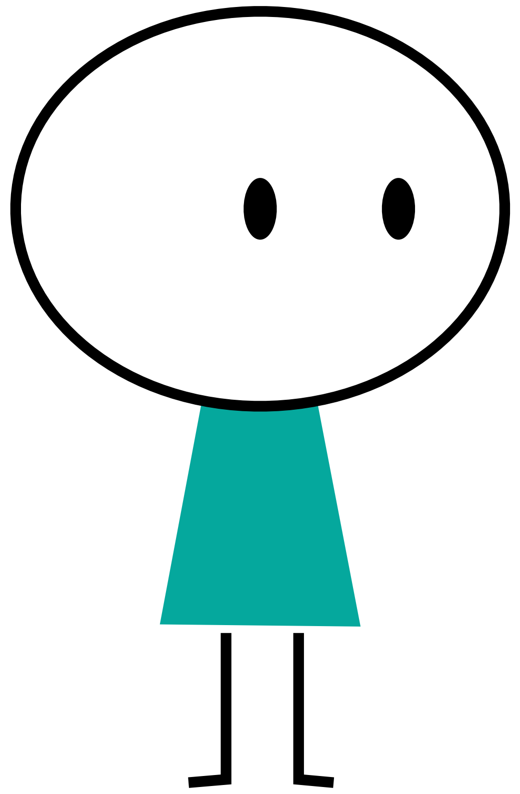
2: $\quad \mathbf{y} \xleftarrow{\$} Y$

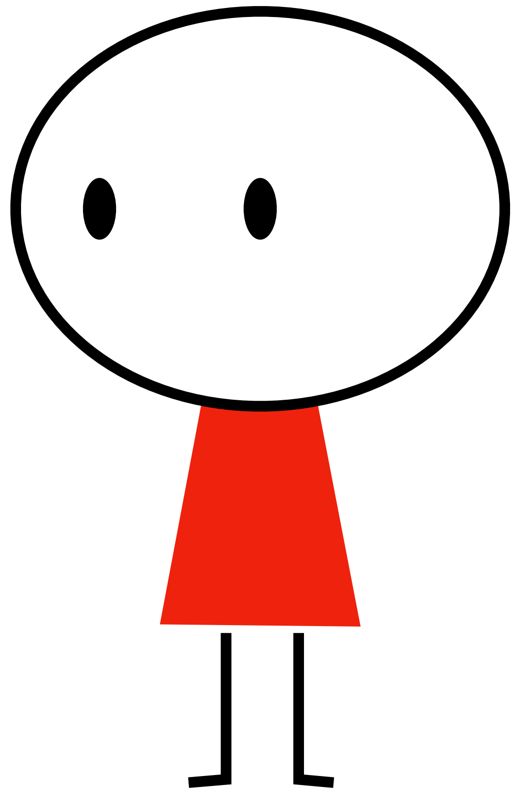
3: $\quad c \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** $(\mathbf{z}, c)$





A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$(\mathbf{A}, \mathbf{t})$ $\longleftarrow$ $\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

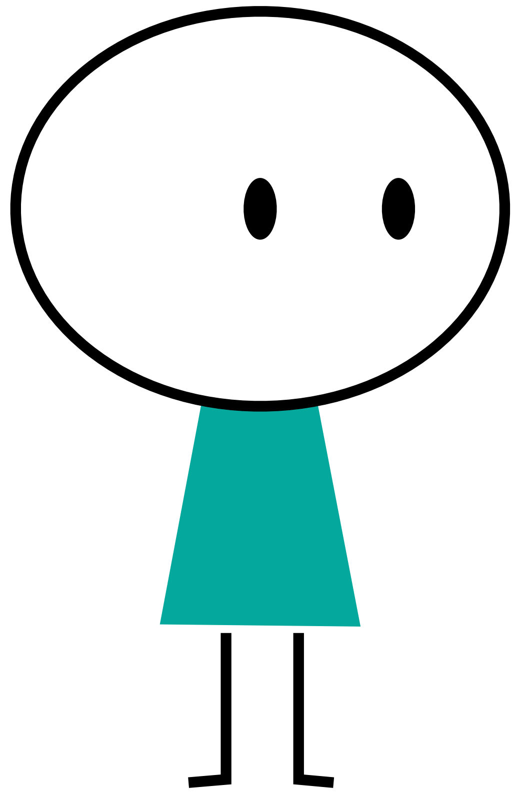
message $(\mathbf{z}, c)$ $\longleftarrow$

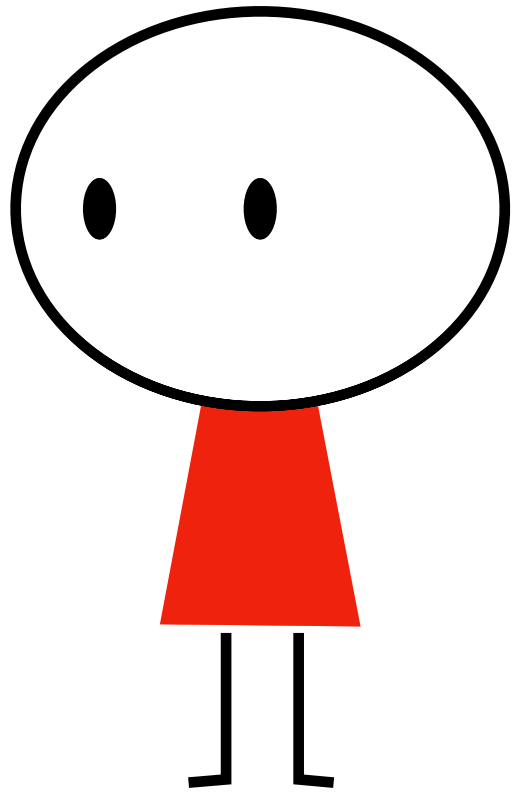
Verification:

1: $\mathbf{r} \leftarrow \mathbf{A} \cdot \mathbf{z} - \mathbf{t} \cdot c$
2: $c' \leftarrow H(\mathbf{r}, m)$
3: if $c' = c$ and $\mathbf{z}$ is small enough:
4: return Valid
5: else:
6: return Invalid

Signature algorithm: message

1: **do**
2: $\mathbf{y} \xleftarrow{\$} Y$
3: $c \leftarrow H(\mathbf{A}\mathbf{y}, m)$
4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)
6: **return** $(\mathbf{z}, c)$





A Fiat-Shamir with aborts signature in a nutshell

- V. Lyubashevsky EUROCRYPT'12
- L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13
- S. Bai and D. Galbraith CT-RSA'14

Short Integer Solution (SIS)

$(\mathbf{A}, \mathbf{t})$ $\leftarrow$ $\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

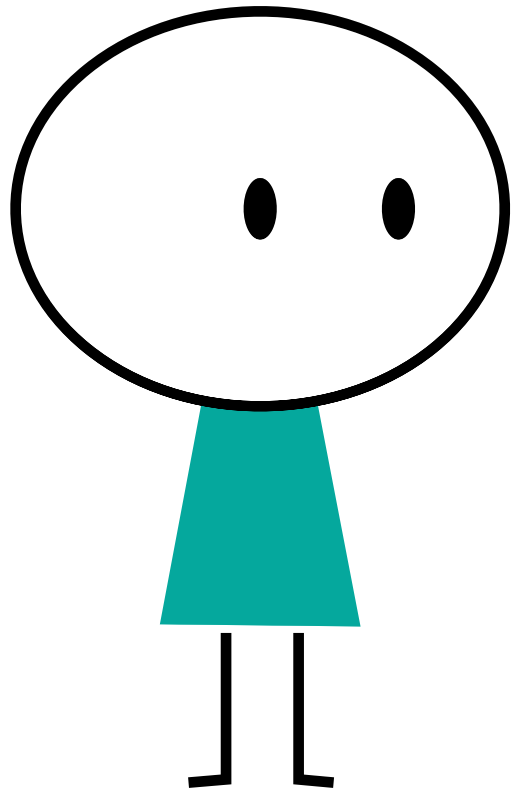
message $(\mathbf{z}, c)$ $\leftarrow$

Verification:

1: $\mathbf{r} \leftarrow \mathbf{A} \cdot \mathbf{z} - \mathbf{t} \cdot c$
2: $c' \leftarrow H(\mathbf{r}, m)$
3: if $c' = c$ and $\mathbf{z}$ is small enough:
4: return Valid
5: else:
6: return Invalid

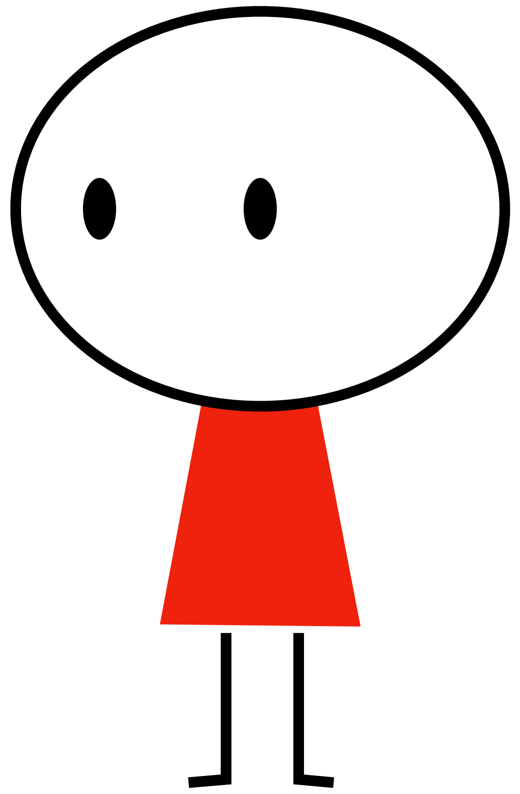
Signature algorithm: message

1: do
2: $\mathbf{y} \xleftarrow{\$} Y$
3: $c \leftarrow H(\mathbf{A}\mathbf{y}, m)$
4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: while Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)
6: return $(\mathbf{z}, c)$



High level idea behind

Crystals-Dilithium (NIST standard)
BLISS, GLP, BG

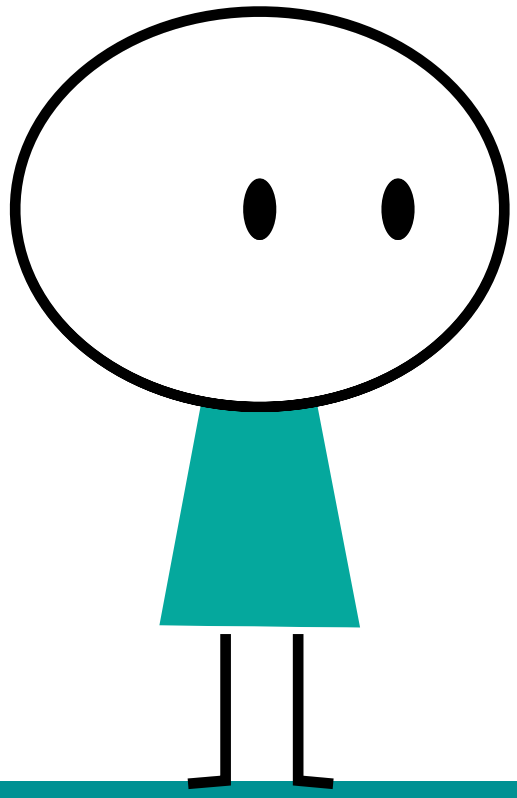


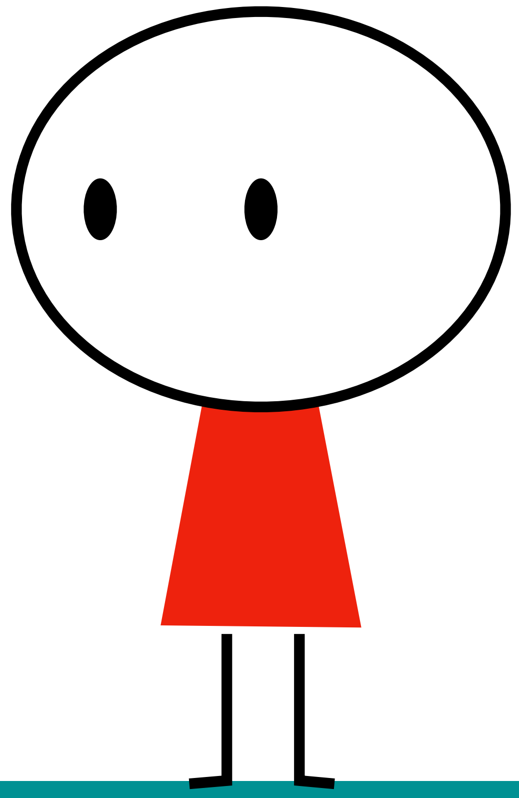
A Hash and sign in a nutshell

- C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08

Generate matrices $\mathbf{A}, \mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$





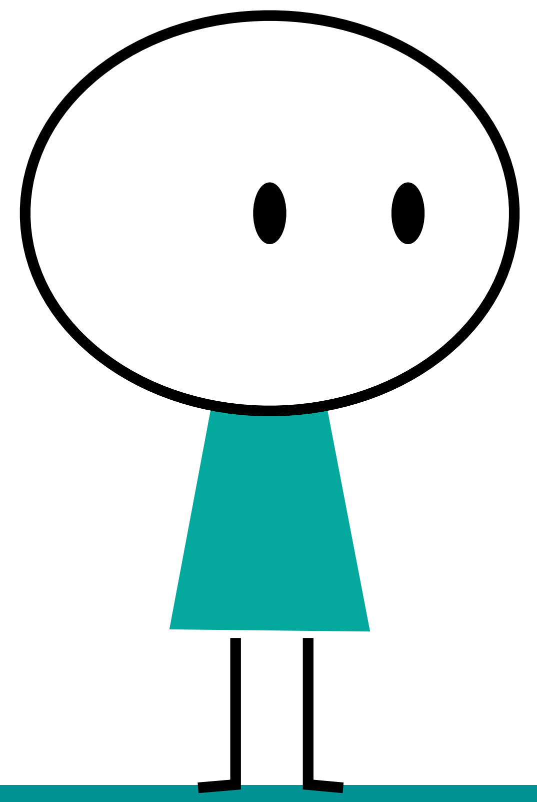
A Hash and sign in a nutshell

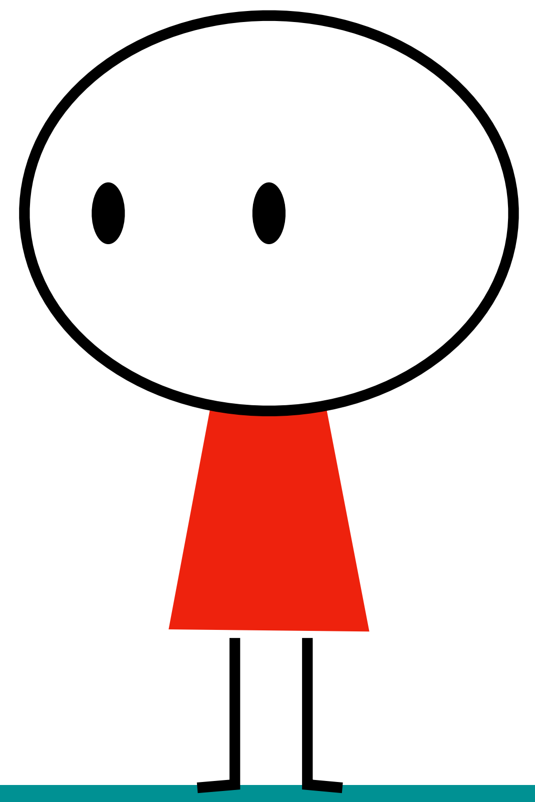
▸ C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08

$\mathbf{A}$

Generate matrices $\mathbf{A}$, $\mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$





A Hash and sign in a nutshell

- C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08

$\mathbf{A}$

Generate matrices $\mathbf{A}, \mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$

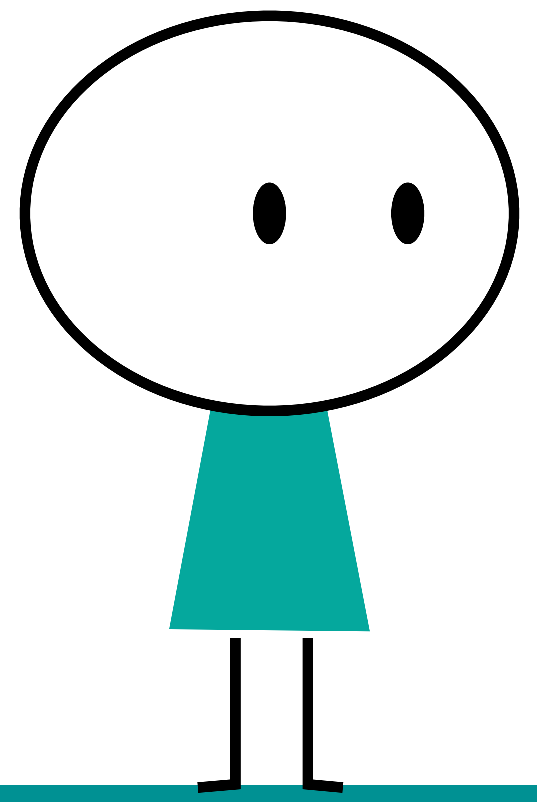
message

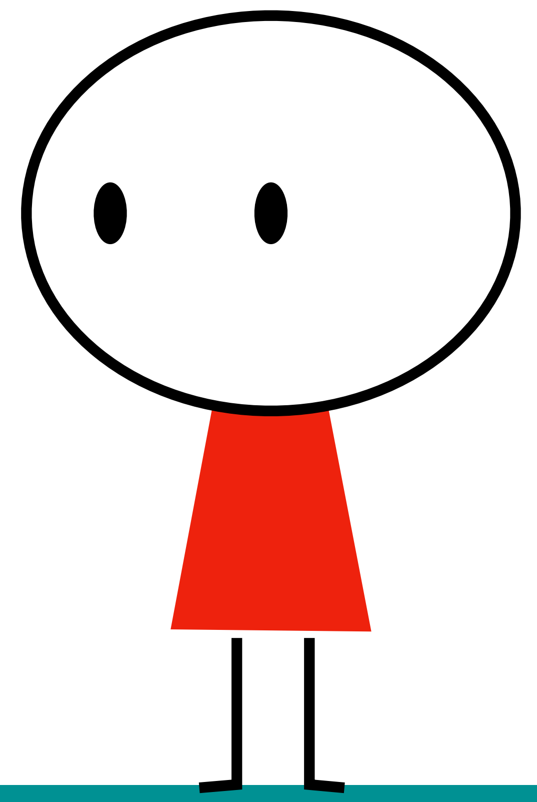
Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{c}\mathbf{A} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

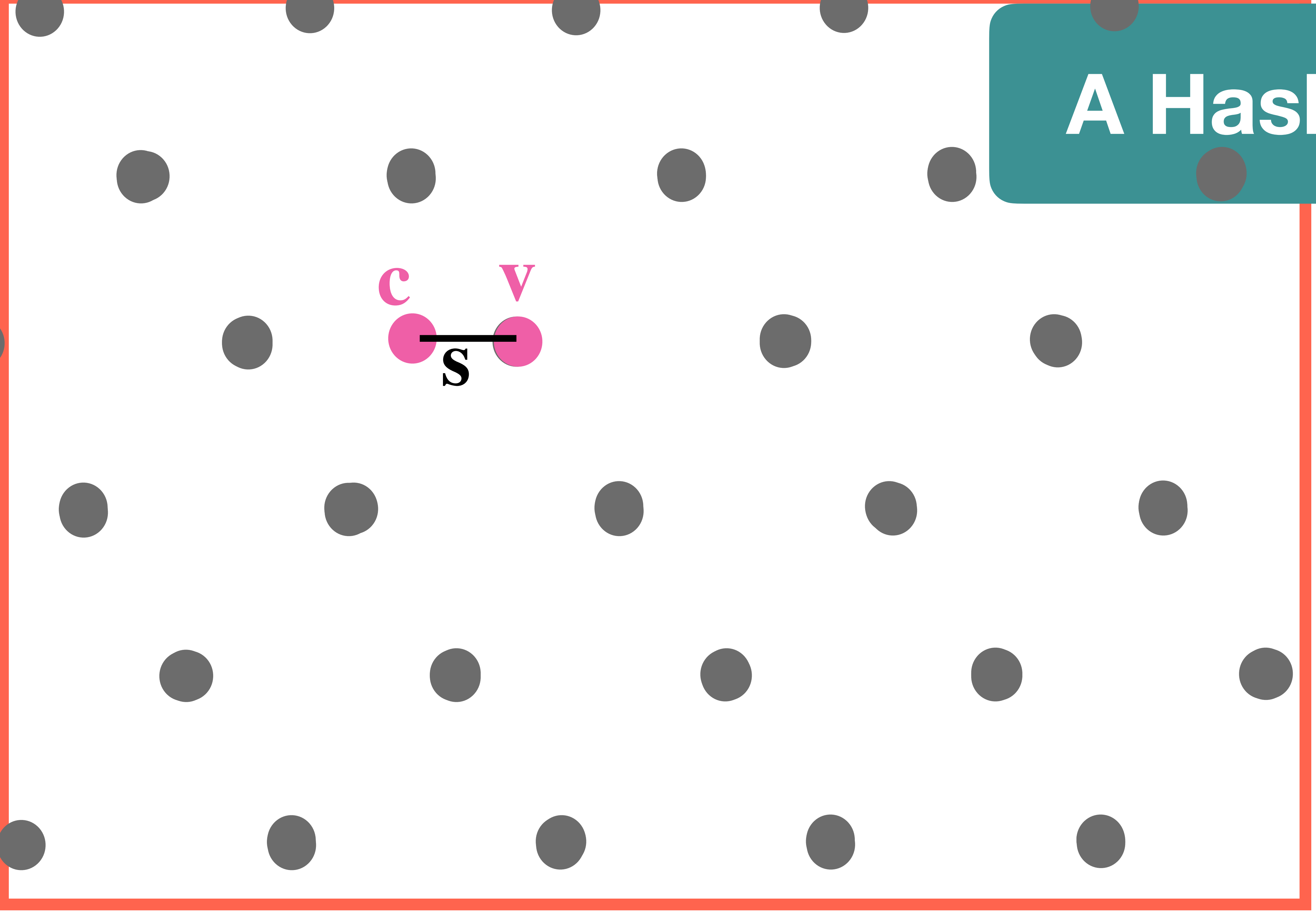
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$





A Hash and sign in a nutshell

- C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08



$\mathbf{A}$ ←

Generate matrices $\mathbf{A}, \mathbf{B}$ such that

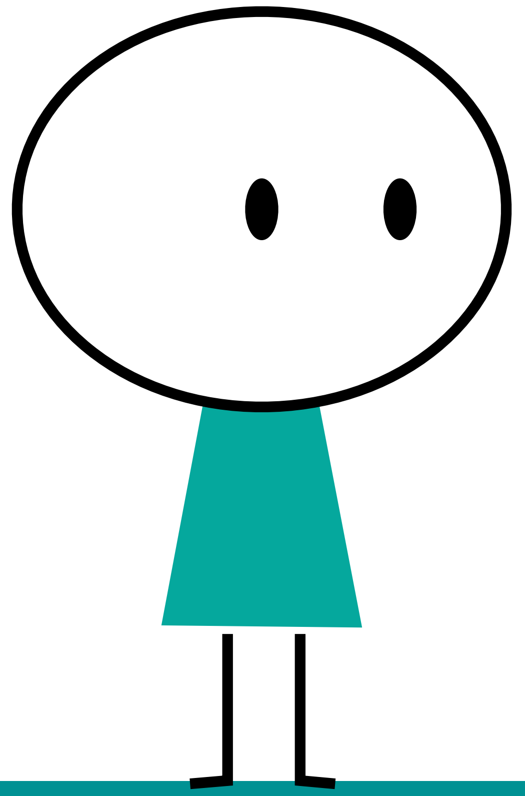
$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$

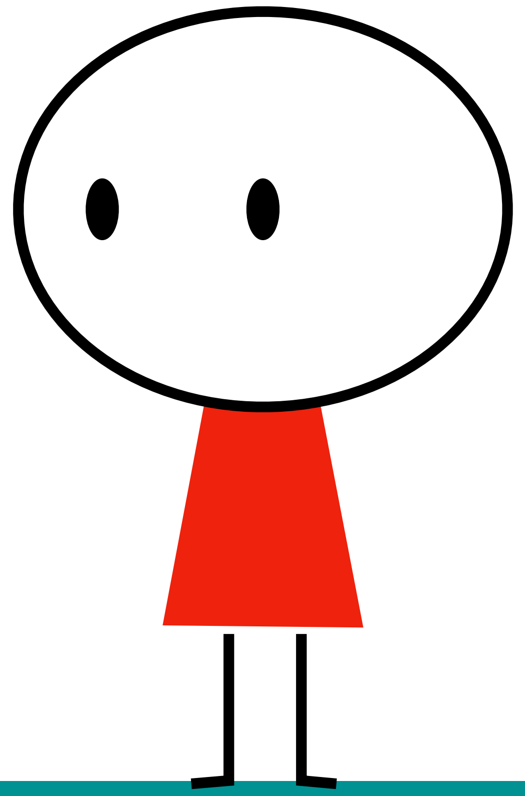
Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{c}\mathbf{A} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

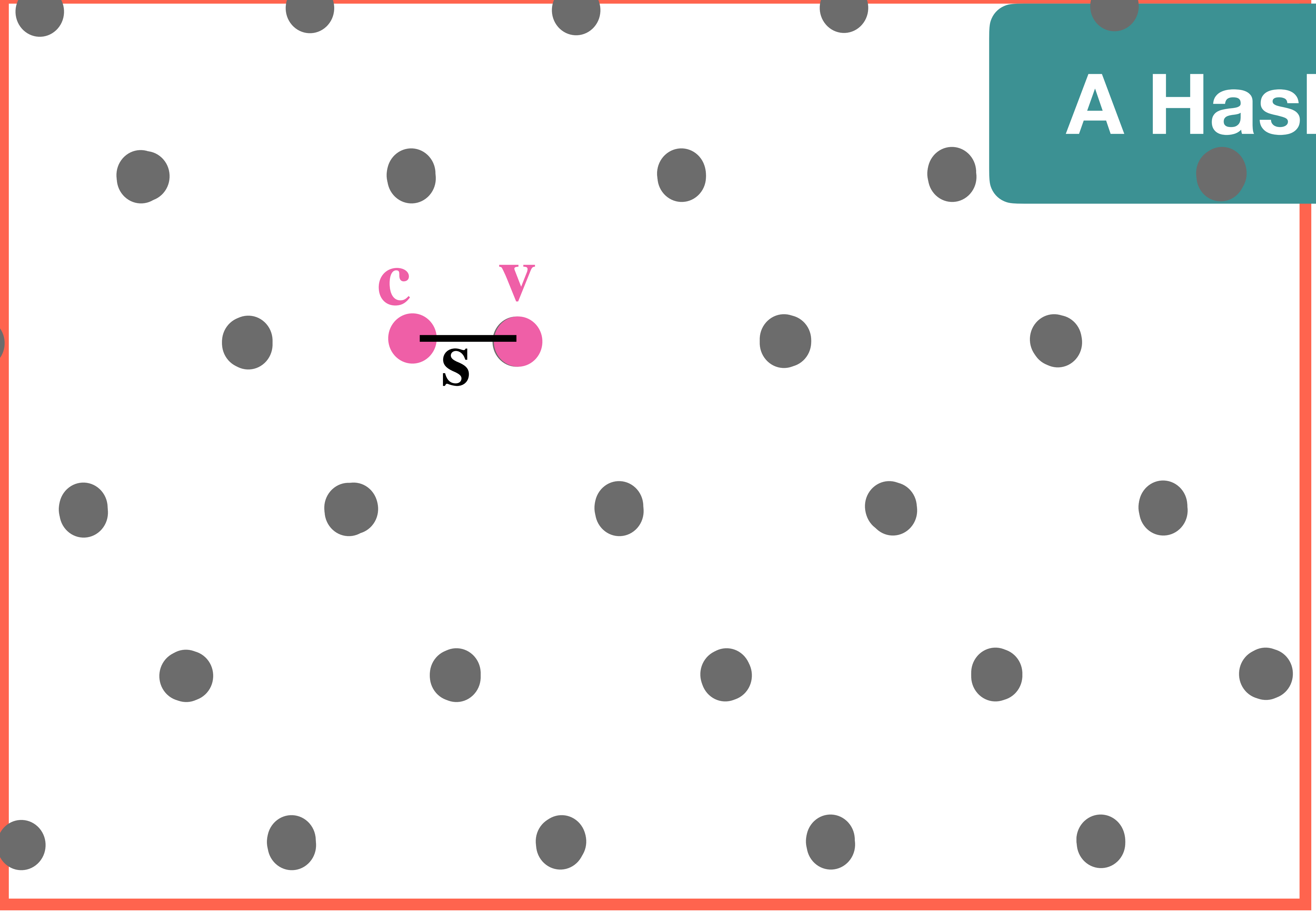
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$





A Hash and sign in a nutshell

▸ C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08



$\mathbf{A}$

Generate matrices $\mathbf{A}, \mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$

message $\mathbf{s}$

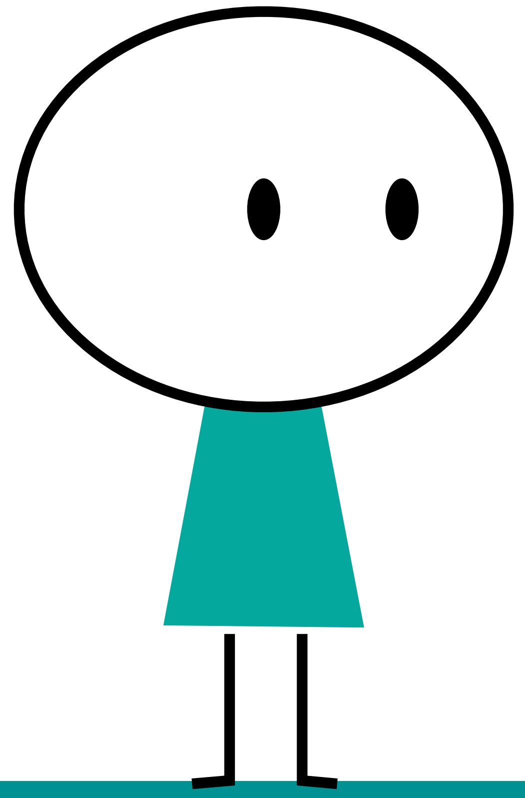
message

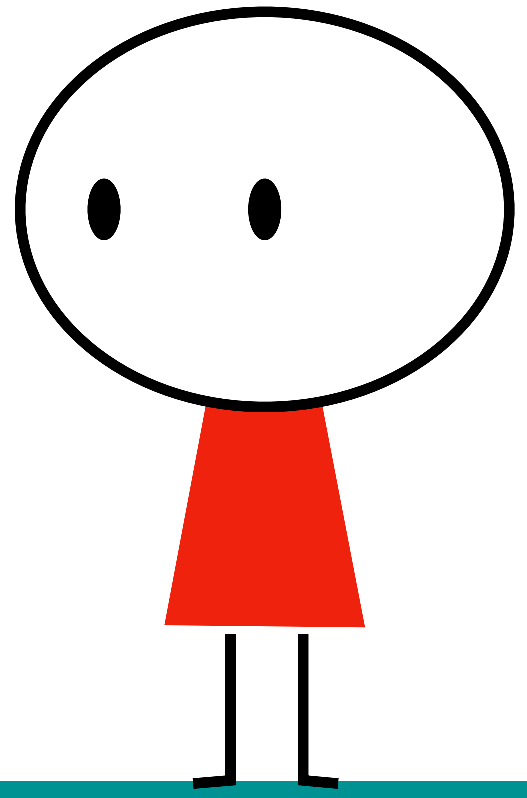
Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{c}\mathbf{A} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

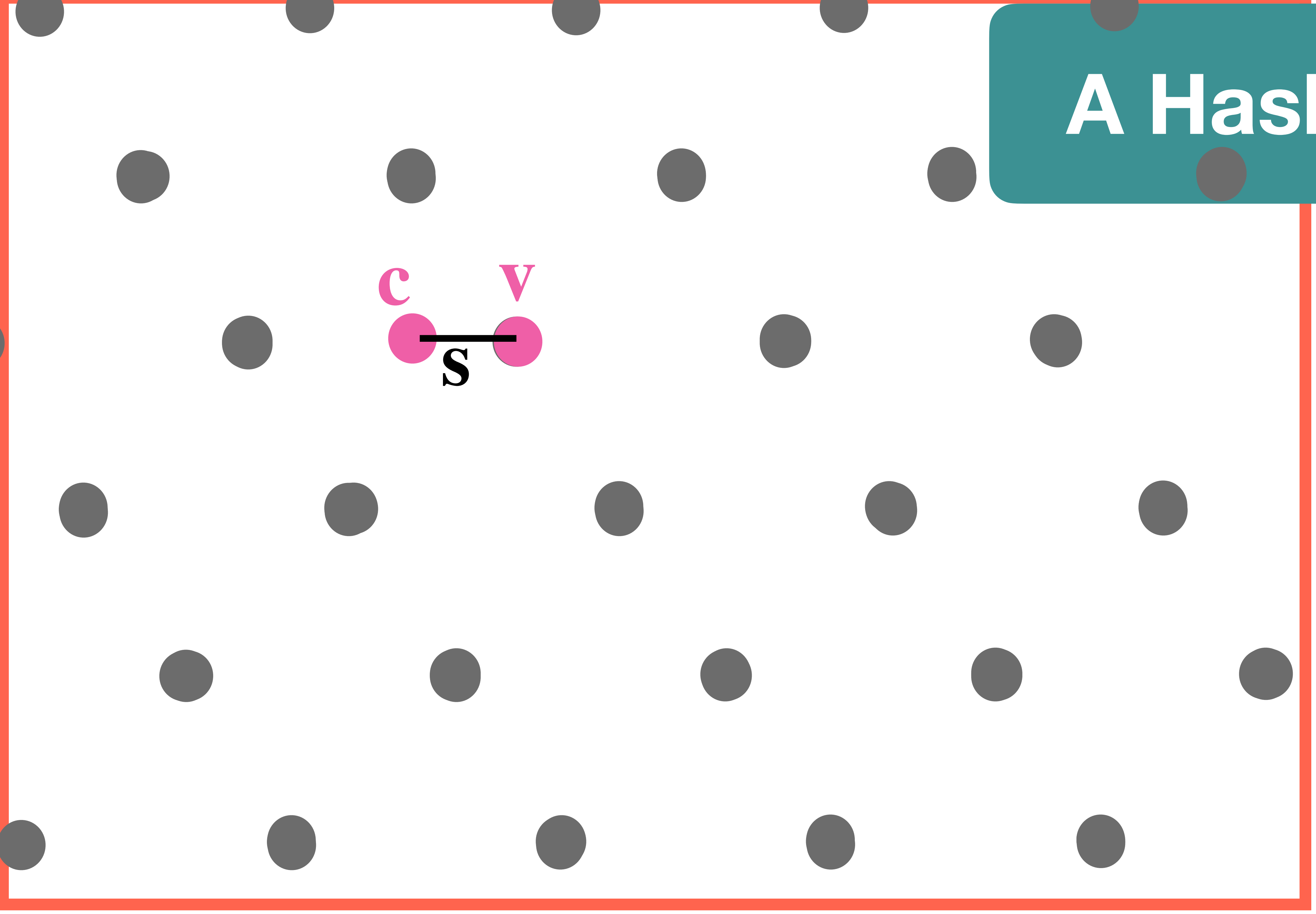
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$





A Hash and sign in a nutshell

▸ C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08



$\mathbf{A}$

Generate matrices $\mathbf{A}, \mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$

message $\mathbf{s}$

Signature algorithm:

message

1: compute $\mathbf{c}$ such that $\mathbf{c}\mathbf{A} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

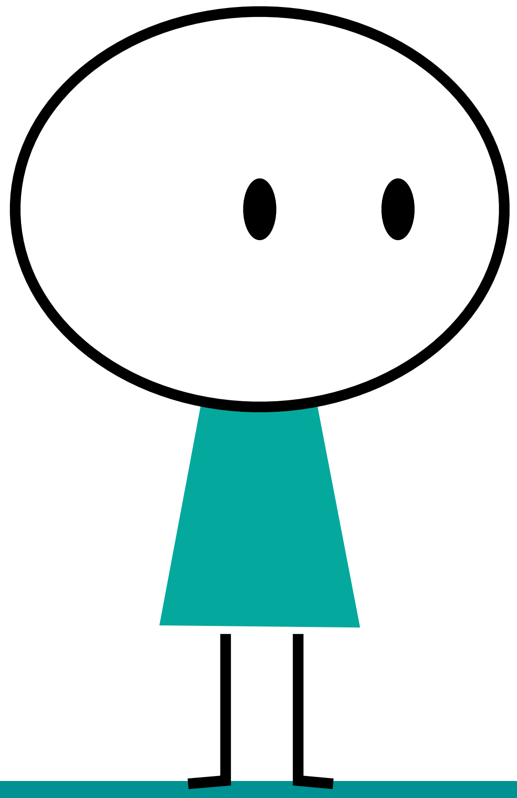
Verification:

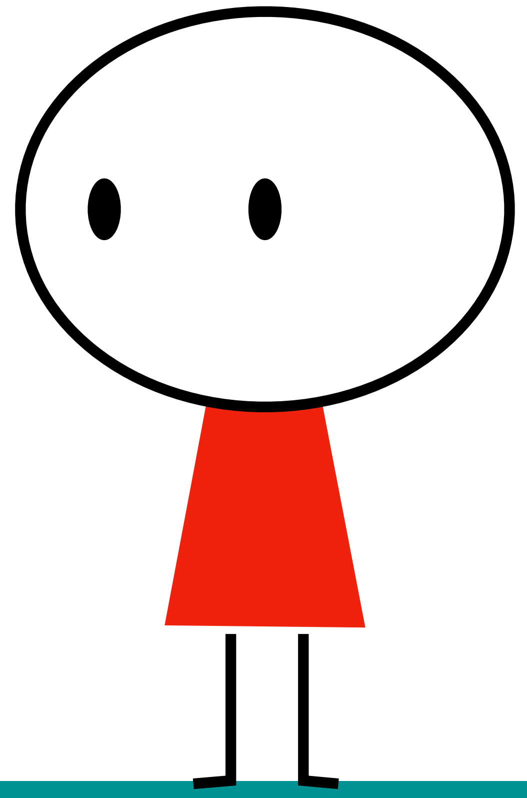
1: if $\mathbf{s}$ is short and $\mathbf{s}\mathbf{A} = H(m)$

2: return Valid

3: else:

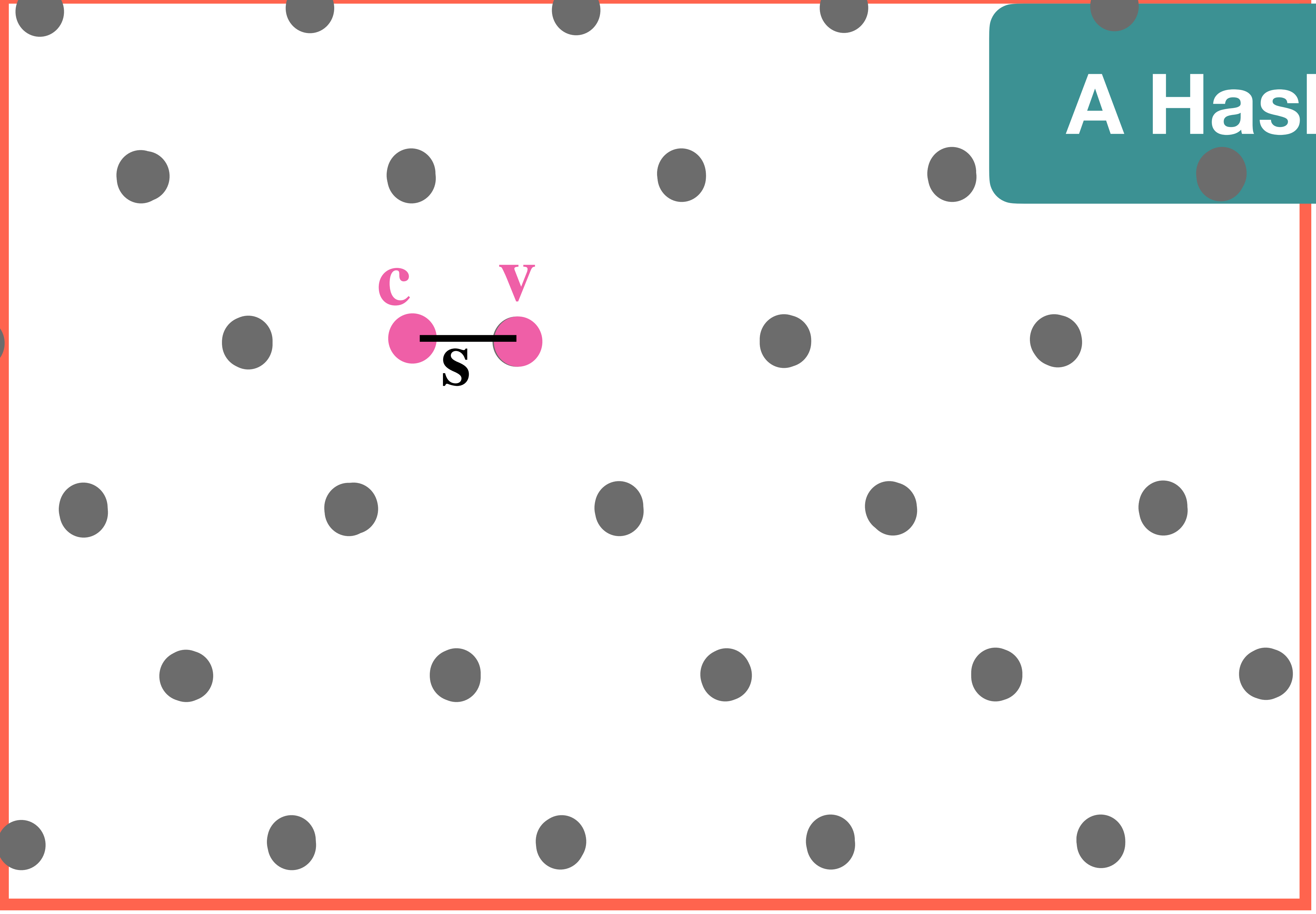
4: return Invalid





A Hash and sign in a nutshell

▸ C.Gentry, C. Peikert and V. Vaikuntanathan STOC'08



$\mathbf{A}$ ←

Generate matrices $\mathbf{A}$, $\mathbf{B}$ such that

$$\begin{cases} \mathbf{B}\mathbf{A} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$$

message $\mathbf{s}$ ←

Signature algorithm: (message)

1: compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

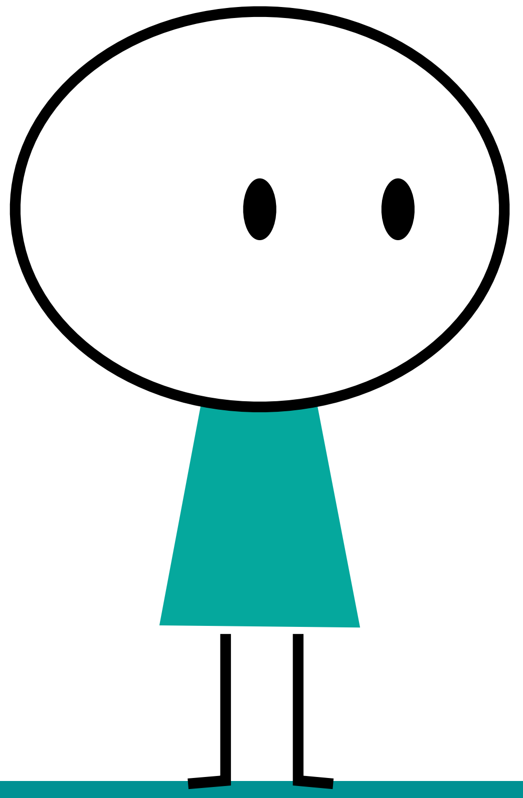
Verification:

1: if $\mathbf{s}$ is short and $\mathbf{sA} = H(m)$

2: return Valid

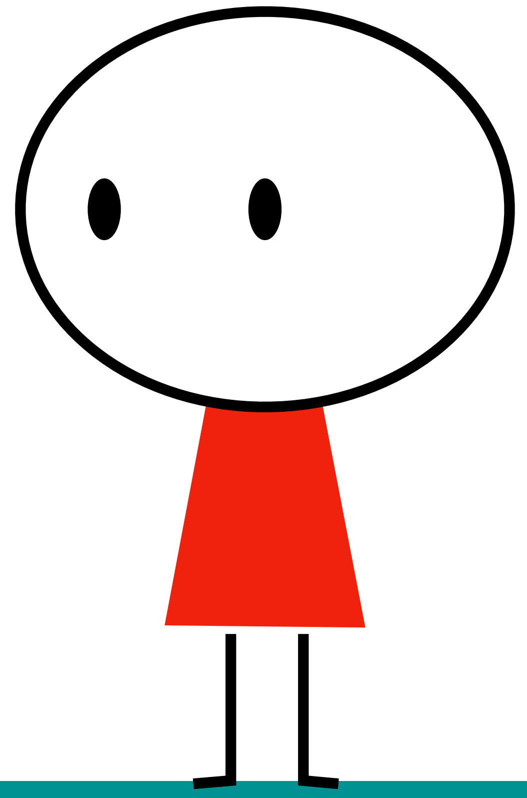
3: else:

4: return Invalid



High level idea behind

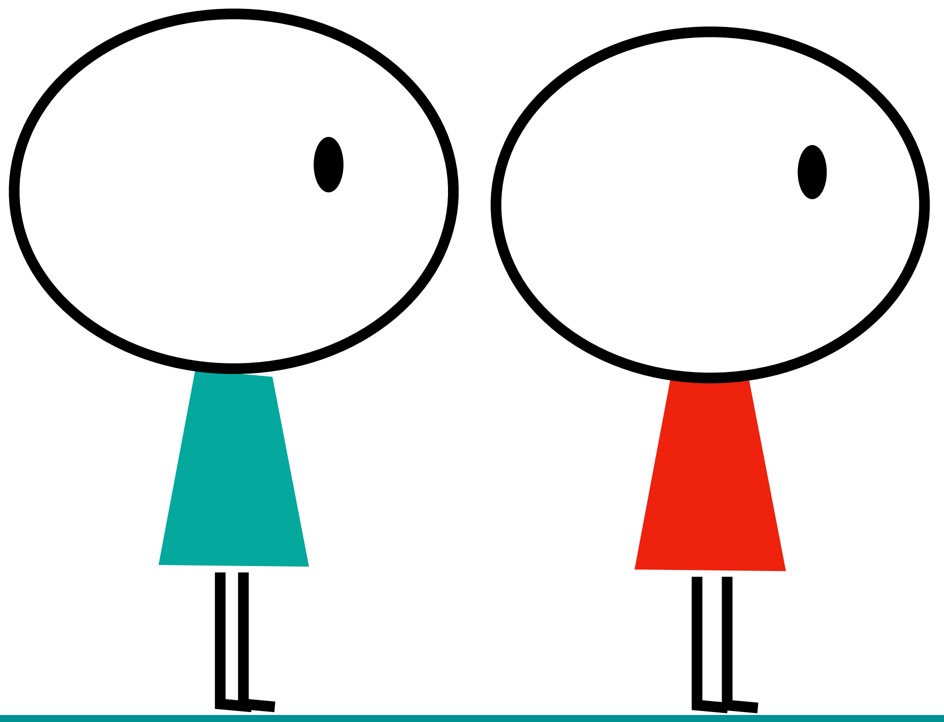
Falcon (NIST standard)
GPV, Mitaka



Lattice-based algorithms

Signature schemes

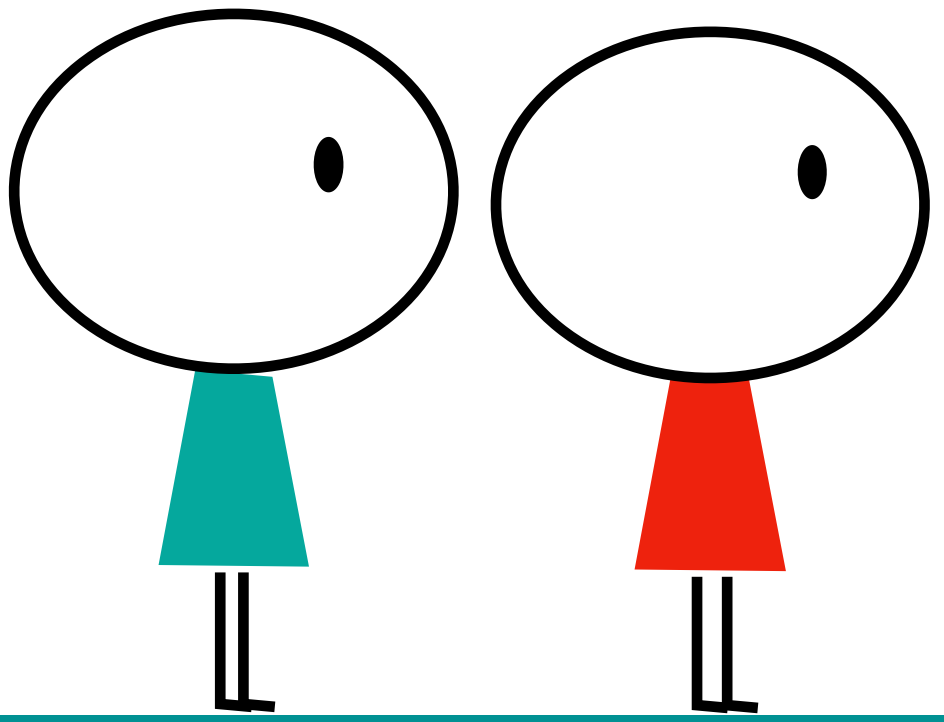
Public key encryption schemes

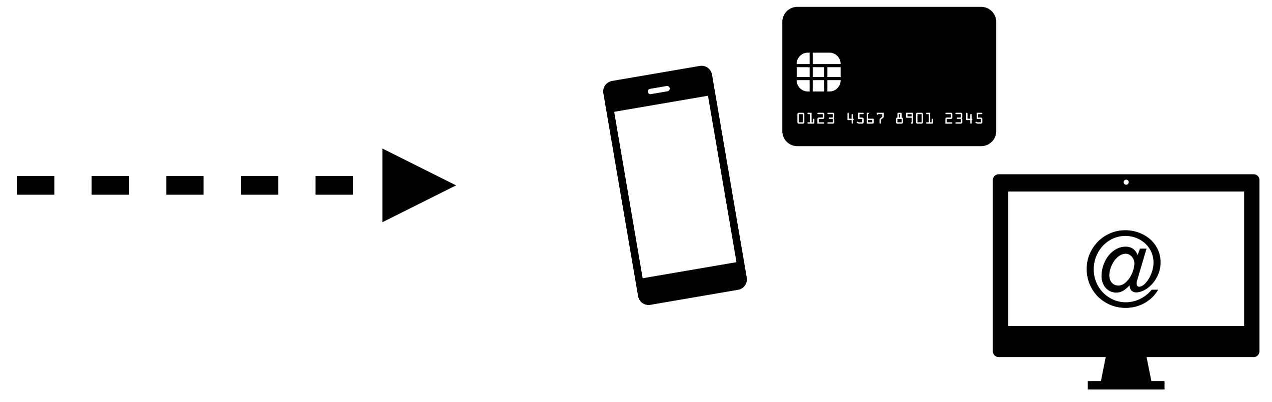


Lattice-based algorithms

Signature schemes

Public key encryption schemes





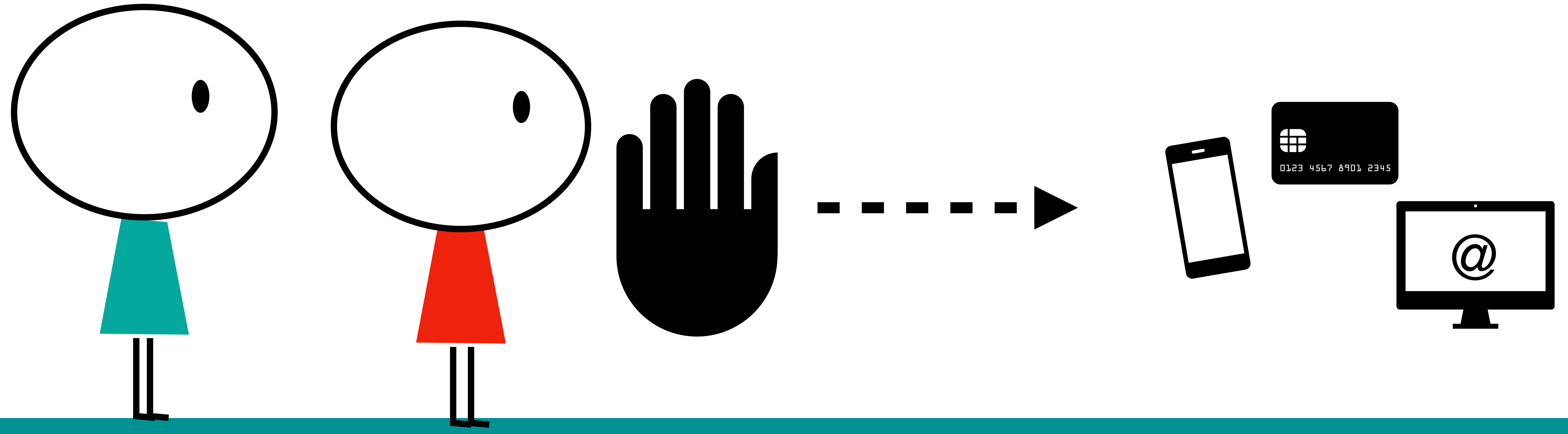
Lattice-based algorithms

Signature schemes

Public key encryption schemes

Are you secure for real-world development ?

Are you timing resistant ?
Are you secure against physical attacks?
Are you misuse resistant ?
Are you decryption-failure resistant?
… by how much ?



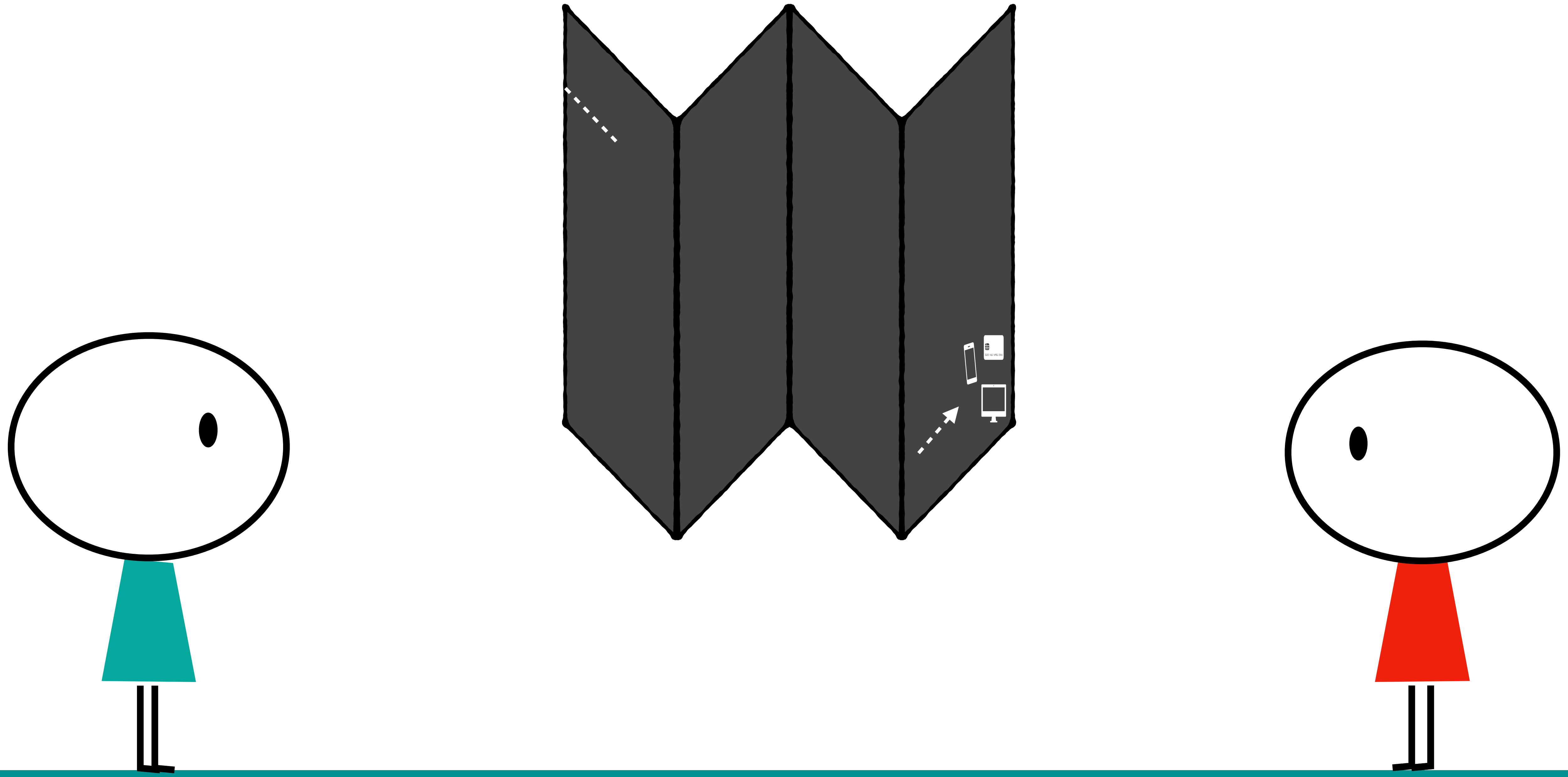
Lattice-based algorithms

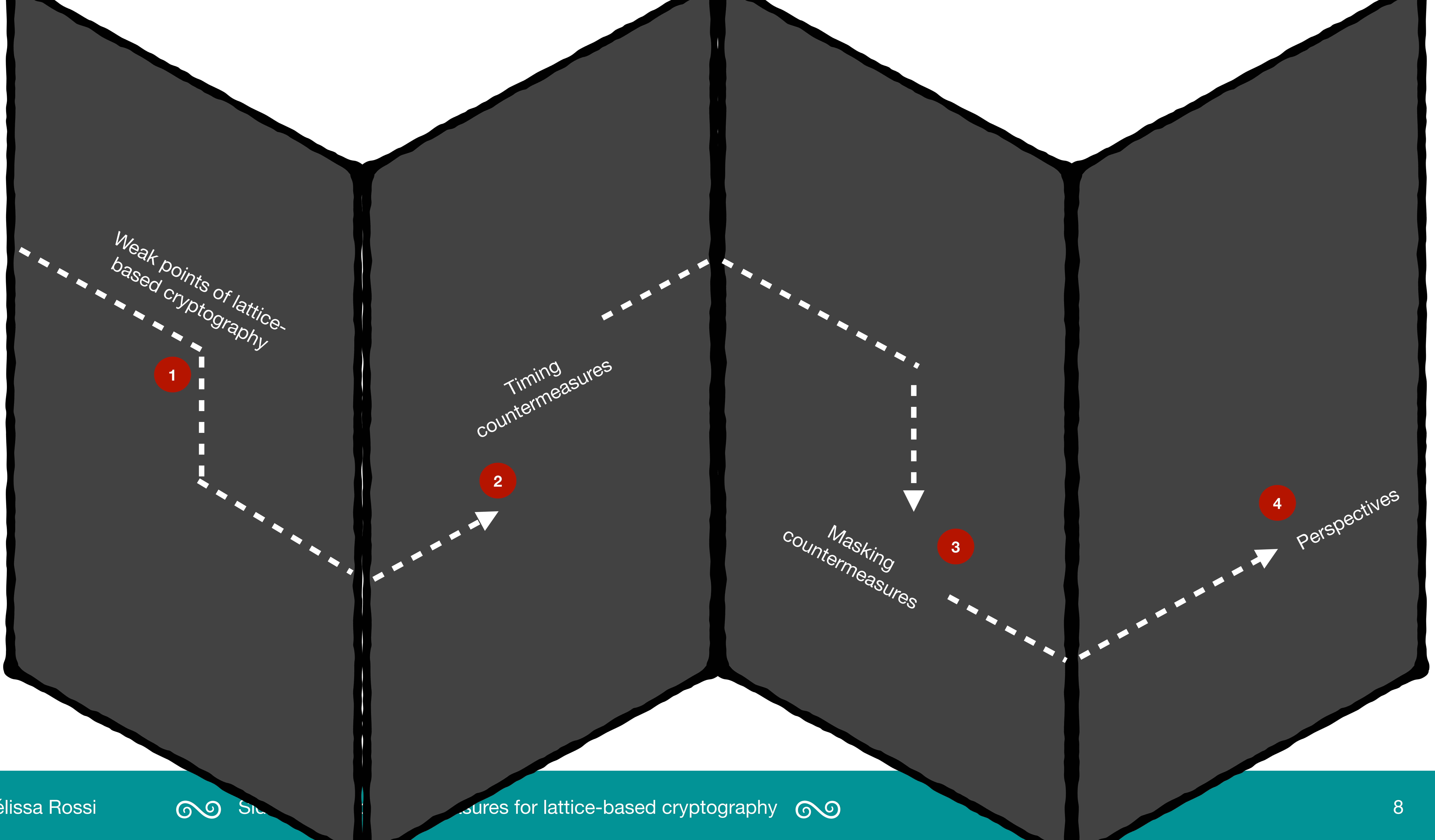
Signature schemes

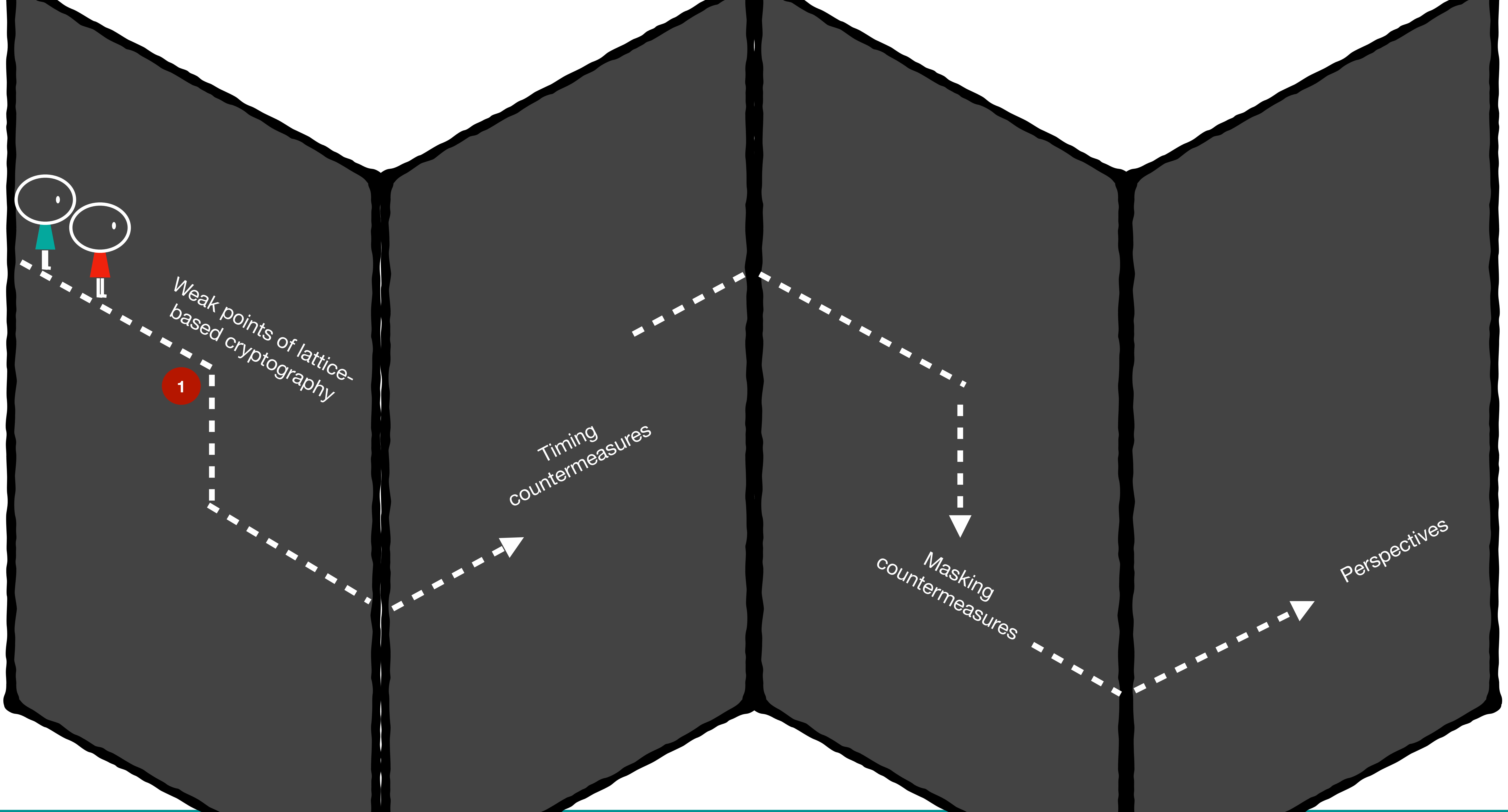
Public key encryption schemes

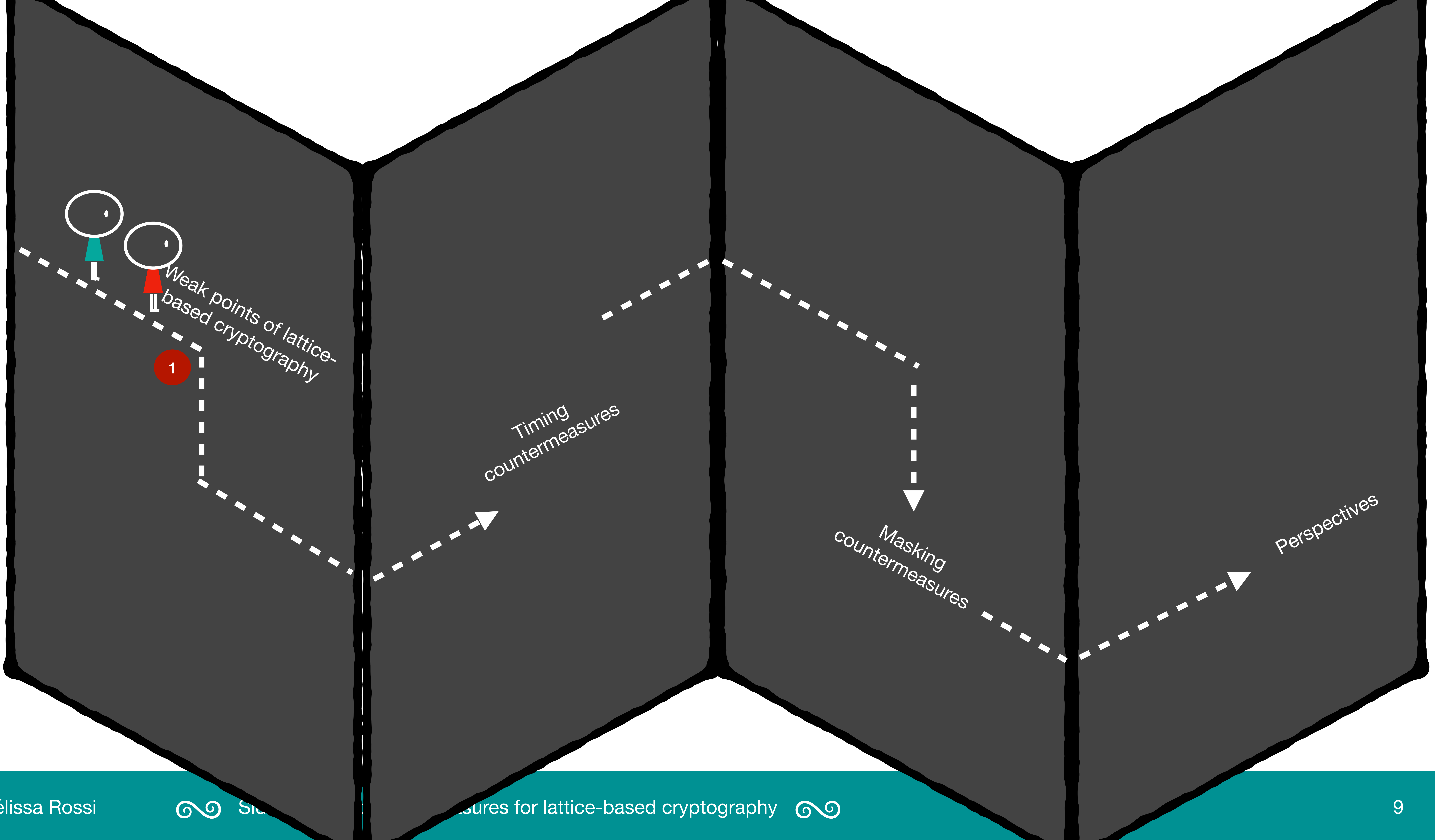
Are you secure for real-world development ?

Are you timing resistant ?
Are you secure against physical attacks?
Are you misuse resistant ?
Are you decryption-failure resistant?
… by how much ?




Weak points of lattice-based cryptography
1
Timing countermeasures
2
Masking countermeasures
3
4
Perspectives


Weak points of lattice-based cryptography
1
Timing countermeasures
Masking countermeasures
Perspectives


Weak points of lattice-based cryptography
1
Timing countermeasures
Masking countermeasures
Perspectives

Lattice-based crypto has many side-channel weak points

$\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

$\mathbf{y} \xleftarrow{\$} Y$

while Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

Lattice-based crypto has many side-channel weak points

$\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

$\mathbf{y} \xleftarrow{\$} Y$

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

<u>Usual suspects:</u>

A. Multiplication with the secret: known $\times$ $\mathbf{s}$
B. Complex internal sampling distributions (Cumulative Distribution Tables)
C. Fujisaki-Okamoto transform
D. NTT, message encoding

Lattice-based crypto has many side-channel weak points

$\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{y} \xleftarrow{\$} Y$

while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

<u>Usual suspects:</u>

A. Multiplication with the secret: known $\times$ $\mathbf{s}$
B. Complex internal sampling distributions (Cumulative Distribution Tables)
C. Fujisaki-Okamoto transform
D. NTT, message encoding

Lattice-based crypto has many side-channel weak points

$\left(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q\right)$

$\mathbf{y} \xleftarrow{\$} Y$

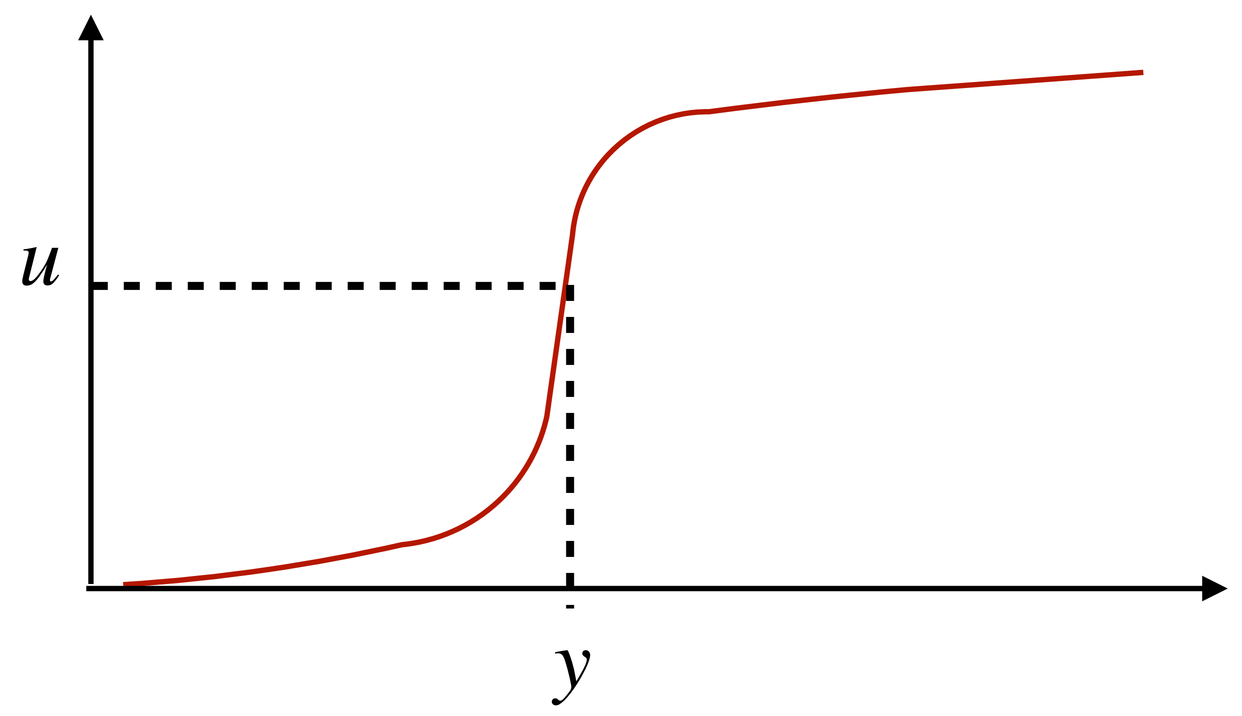
while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

<u>Usual suspects:</u>

A. Multiplication with the secret: known $\times$ $\mathbf{s}$
B. Complex internal sampling distributions (Cumulative Distribution Tables)
C. Fujisaki-Okamoto transform
D. NTT, message encoding



Lattice-based crypto has many side-channel weak points

$(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q)$

$\mathbf{y} \xleftarrow{\$} Y$

while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

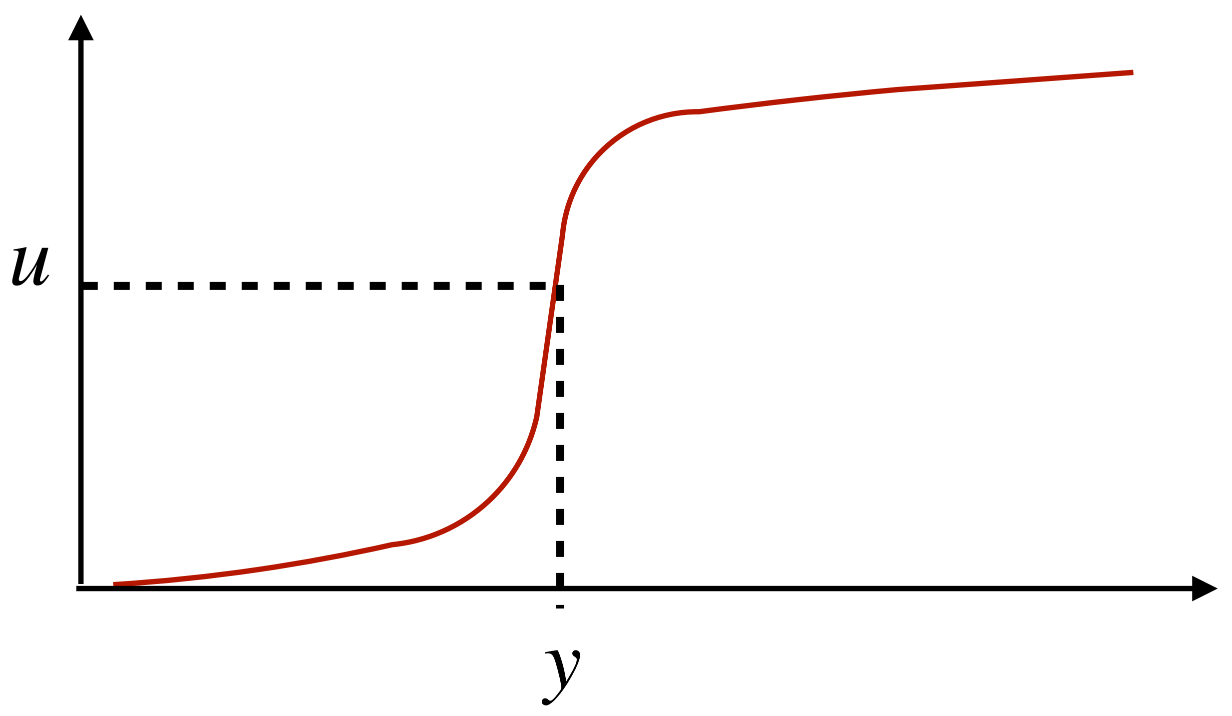
nb?

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

<u>Usual suspects:</u>

A. Multiplication with the secret: known $\times$ $\mathbf{s}$
B. Complex internal sampling distributions (Cumulative Distribution Tables)
C. Fujisaki-Okamoto transform
D. NTT, message encoding



Lattice-based crypto has many side-channel weak points

$(\mathbf{A}, \mathbf{t} = \mathbf{A}\mathbf{S} \bmod q)$

$\mathbf{y} \xleftarrow{\$} Y$

while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

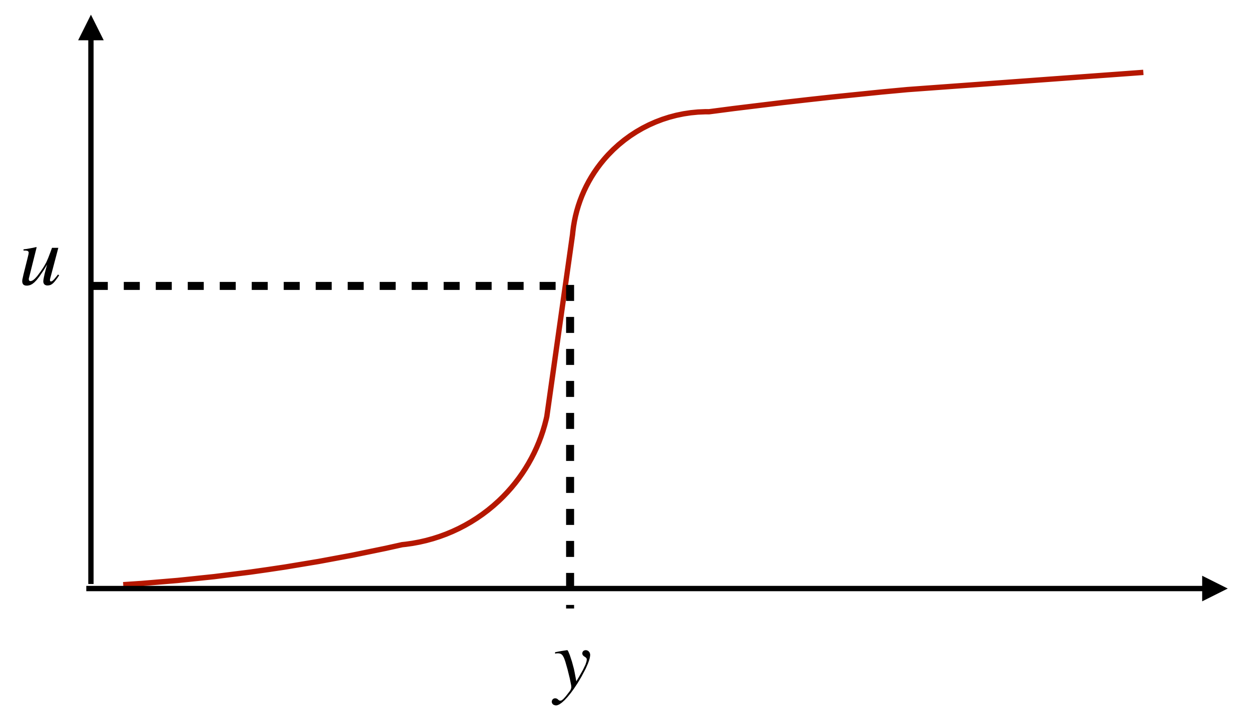
nb?

$\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

$\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

<u>Usual suspects:</u>

A. Multiplication with the secret: known $\times$ $\mathbf{s}$
B. Complex internal sampling distributions (Cumulative Distribution Tables)
C. Fujisaki-Okamoto transform
D. NTT, message encoding



We will give three examples related to B and C

Timing attacks on internal distributions

<u>Timing attack:</u> the attacker knows the time that the algorithm takes e.g. the number of iterations.

Timing attacks on internal distributions

<u>Timing attack:</u> the attacker knows the time that the algorithm takes e.g. the number of iterations.

In lattice-based schemes, we always to sample small coefficients.

Gaussians are often used for two reasons:

Performance

Security reductions

Timing attacks on internal distributions

<u>Timing attack:</u> the attacker knows the time that the algorithm takes e.g. the number of iterations.

In lattice-based schemes, we always to sample small coefficients.

Gaussians are often used for two reasons:

Performance | **Security reductions**

It implies computing transcendental functions $\exp(\,.\,)$ and $\cosh(\,.\,)$ → Hard to compute efficiently in constant time!

Timing attacks on internal distributions

<u>Timing attack:</u> the attacker knows the time that the algorithm takes e.g. the number of iterations.

In lattice-based schemes, we always to sample small coefficients.

Gaussians are often used for two reasons:

Performance

Security reductions

It implies computing transcendental functions $\exp(\,.\,)$ and $\cosh(\,.\,)$ → Hard to compute efficiently in constant time!

Many timing attacks
targeting Gaussian distributions in lattice-based signature schemes

- L. Groot Bruinderink, A. Hülsing, T. Lange, and Y. Yarom. CHES'2016
- T. Espitau, P.-A. Fouque, B. Gérard, M. Tibouchi. SAC'2016
- P. Pessl, L. Groot Bruinderink, and Y. Yarom. ACM-CCS'2017
- T. Espitau, P.-A. Fouque, B. Gérard and M. Tibouchi. ACM-CCS'2017
- J. Bootle, C. Delaplace, T. Espitau, P.-A. Fouque and M. Tibouchi. ASIACRYPT'2018
- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi. ACM-CCS'2019
- P.-A. Fouque, P. Kirchner,,M. Tibouchi, A. Wallet, and Y. Yu. EUROCRYPT'2020

Timing attacks on internal distributions

Timing attack: the attacker knows the time that the algorithm takes e.g. the number of iterations.

In lattice-based schemes, we always to sample small coefficients.

Gaussians are often used for two reasons:

Performance

Security reductions

It implies computing transcendental functions $\exp(\,.\,)$ and $\cosh(\,.\,)$ → Hard to compute efficiently in constant time!

Many timing attacks
targeting Gaussian distributions in lattice-based signature schemes

- L. Groot Bruinderink, A. Hülsing, T. Lange, and Y. Yarom. CHES'2016
- T. Espitau, P.-A. Fouque, B. Gérard, M. Tibouchi. SAC'2016
- P. Pessl, L. Groot Bruinderink, and Y. Yarom. ACM-CCS'2017
- T. Espitau, P.-A. Fouque, B. Gérard and M. Tibouchi. ACM-CCS'2017
- J. Bootle, C. Delaplace, T. Espitau, P.-A. Fouque and M. Tibouchi. ASIACRYPT'2018

An example presented in the next slide →

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi. ACM-CCS'2019
- P.-A. Fouque, P. Kirchner,,M. Tibouchi, A. Wallet, and Y. Yu. EUROCRYPT'2020

An example of timing attack on BLISS signature scheme

Signature algorithm:

1: **do**

2: $\quad \mathbf{y} \xleftarrow{\$} Y$

3: $\quad \mathbf{c} \leftarrow H(\mathbf{Ay}, m)$

4: $\quad \mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** ($\mathbf{z}$, $\mathbf{c}$)

An example of timing attack on BLISS signature scheme

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $\mathbf{c} \leftarrow H(\mathbf{Ay}, m)$

4: $\mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** ($\mathbf{z}$, $\mathbf{c}$)

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}$$

An example of timing attack on BLISS signature scheme

Signature algorithm:

do

$\quad \mathbf{y} \xleftarrow{\$} Y$

$\quad \mathbf{c} \leftarrow H(\mathbf{Ay}, m)$

$\quad \mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

return ($\mathbf{z}, \mathbf{c}$)

$$\boxed{\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}}$$

➡ computed by sampling two Bernouilli $\mathscr{B}_{1/\cosh(x)}$ and $\mathscr{B}_{\exp(-x)}$

An example of timing attack on BLISS signature scheme

Signature algorithm:

do

$\quad \mathbf{y} \xleftarrow{\$} Y$

$\quad \mathbf{c} \leftarrow H(\mathbf{A}\mathbf{y}, m)$

$\quad \mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

while Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$

return $(\mathbf{z}, \mathbf{c})$

$$\boxed{\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{S}\mathbf{c}||^2}{2\sigma^2}\right)}}$$

➡ computed by sampling two Bernouilli $\boxed{\mathscr{B}_{1/\cosh(x)}}$ and $\mathscr{B}_{\exp(-x)}$

$$\frac{1}{\cosh(x)} = \frac{\exp(-|x|)}{1/2 + 1/2\exp(-2|x|)}$$

An example of timing attack on BLISS signature scheme

Signature algorithm:

1: do

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $\mathbf{c} \leftarrow H(\mathbf{Ay}, m)$

4: $\mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

5: while Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

6: return ($\mathbf{z}, \mathbf{c}$)

$$\boxed{\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}}$$

➡ computed by sampling two Bernouilli $\boxed{\mathscr{B}_{1/\cosh(x)}}$ and $\mathscr{B}_{\exp(-x)}$

$$\frac{1}{\cosh(x)} = \frac{\exp(-|x|)}{1/2 + 1/2\exp(-2|x|)}$$

Sampling a Bernoulli with parameter 1/cosh(x) :$\mathscr{B}_{1/\cosh(x)}$

1: $x \leftarrow |x|$

2: $a \leftarrow \mathscr{B}_{\exp(-x)}$

3: $b \leftarrow \mathscr{B}_{1/2}$

4: $c \leftarrow \mathscr{B}_{\exp(-x)}$

5: if $\bar{a} \wedge (b \vee c)$ then restart

6: return a

An example of timing attack on BLISS signature scheme

Signature algorithm:

1: do

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $\mathbf{c} \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\mathbf{z} \leftarrow \mathbf{c} \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

6: **return** ($\mathbf{z}, \mathbf{c}$)

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{S}\mathbf{c}||^2}{2\sigma^2}\right)}$$

➡ computed by sampling two Bernouilli $\boxed{\mathscr{B}_{1/\cosh(x)}}$ and $\mathscr{B}_{\exp(-x)}$

$$\frac{1}{\cosh(x)} = \frac{\exp(-|x|)}{1/2 + 1/2\exp(-2|x|)}$$

Correctness

The distribution of a is indeed $\mathscr{B}_{1/\cosh(x)}$.

‣ L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky CRYPTO'13

Sampling a Bernoulli with parameter 1/cosh(x) : $\mathscr{B}_{1/\cosh(x)}$

1: $x \leftarrow |x|$

2: $a \leftarrow \mathscr{B}_{\exp(-x)}$

3: $b \leftarrow \mathscr{B}_{1/2}$

4: $c \leftarrow \mathscr{B}_{\exp(-x)}$

5: **if** $\bar{a} \wedge (b \vee c)$ **then restart**

6: **return** a

An example of timing attack on BLISS signature scheme

Sampling a Bernoulli with parameter cosh(x) : $\mathcal{B}_{1/\text{cosh(x)}}$

1: $x \leftarrow |x|$
2: $a \leftarrow \mathcal{B}_{\exp(-x)}$
3: $b \leftarrow \mathcal{B}_{1/2}$
4: $c \leftarrow \mathcal{B}_{\exp(-x)}$
5: if $\bar{a} \wedge (b \vee c)$ **then restart**
6: return a

An example of timing attack on BLISS signature scheme

Sampling a Bernoulli with parameter cosh(x) : $\mathcal{B}_{1/\text{cosh(x)}}$

1: $x \leftarrow |x|$
2: $a \leftarrow \mathcal{B}_{\exp(-x)}$
3: $b \leftarrow \mathcal{B}_{1/2}$
4: $c \leftarrow \mathcal{B}_{\exp(-x)}$
5: if $\bar{a} \wedge (b \vee c)$ then restart
6: return a

Even if every Bernoulli sampling is constant time, there is still **timing attack!**

An example of timing attack on BLISS signature scheme

Sampling a Bernoulli with parameter cosh(x) :$\mathcal{B}_{1/\cosh(x)}$

1: $x \leftarrow |x|$
2: $a \leftarrow \mathcal{B}_{\exp(-x)}$
3: $b \leftarrow \mathcal{B}_{1/2}$
4: $c \leftarrow \mathcal{B}_{\exp(-x)}$
5: if $\bar{a} \wedge (b \vee c)$ then restart
6: return a

Even if every Bernoulli sampling is constant time, there is still **timing attack!**

➡ Probability of going from step 5 to step 6:

$$\begin{aligned}\mathbb{P}(\overline{\bar{a} \wedge (b \vee c)}) &= 1 - \mathbb{P}(\bar{a}) \cdot \mathbb{P}(b \vee c) \\ &= 1 - (1 - \mathbb{P}(a)) \cdot (1 - \mathbb{P}(\bar{b} \wedge \bar{c})) \\ &= 1 - (1 - \exp(-x))\left(1 - \frac{1 - \exp(-x)}{2}\right) \\ &= \frac{1 + \exp(-2x)}{2}\end{aligned}$$

An example of timing attack on BLISS signature scheme

Sampling a Bernoulli with parameter cosh(x) :$\mathscr{B}_{1/\cosh(x)}$

1: $x \leftarrow |x|$
2: $a \leftarrow \mathscr{B}_{\exp(-x)}$
3: $b \leftarrow \mathscr{B}_{1/2}$
4: $c \leftarrow \mathscr{B}_{\exp(-x)}$
5: **if** $\bar{a} \wedge (b \vee c)$ **then restart**
6: **return** a

Even if every Bernoulli sampling is constant time, there is still **timing attack!**

➡ Probability of going from step 5 to step 6:

$$\begin{aligned}\mathbb{P}(\overline{\bar{a} \wedge (b \vee c)}) &= 1 - \mathbb{P}(\bar{a}) \cdot \mathbb{P}(b \vee c) \\ &= 1 - (1 - \mathbb{P}(a)) \cdot (1 - \mathbb{P}(\bar{b} \wedge \bar{c})) \\ &= 1 - (1 - \exp(-x))\left(1 - \frac{1 - \exp(-x)}{2}\right) \\ &= \frac{1 + \exp(-2x)}{2}\end{aligned}$$

Depends on the input!

An example of timing attack on BLISS signature scheme

Sampling a Bernoulli with parameter cosh(x) :$\mathcal{B}_{1/\cosh(x)}$

1: $x \leftarrow |x|$
2: $a \leftarrow \mathcal{B}_{\exp(-x)}$
3: $b \leftarrow \mathcal{B}_{1/2}$
4: $c \leftarrow \mathcal{B}_{\exp(-x)}$
5: **if** $\bar{a} \wedge (b \vee c)$ **then restart**
6: **return** a

Even if every Bernoulli sampling is constant time, there is still timing attack!

➡ **Probability of going from step 5 to step 6:**

$$\begin{aligned}\mathbb{P}(\overline{\bar{a} \wedge (b \vee c)}) &= 1 - \mathbb{P}(\bar{a}) \cdot \mathbb{P}(b \vee c) \\ &= 1 - (1 - \mathbb{P}(a)) \cdot (1 - \mathbb{P}(\bar{b} \wedge \bar{c})) \\ &= 1 - (1 - \exp(-x))\left(1 - \frac{1 - \exp(-x)}{2}\right) \\ &= \frac{1 + \exp(-2x)}{2}\end{aligned}$$

Depends on the input!

Idea of the attack

Here $x = -|\langle z, \mathbf{S}c \rangle|$

We select the signatures (z, c) corresponding to **one iteration inside the Bernouilli sampling.**

It means that $\dfrac{1 + \exp(-2|\langle z, \mathbf{S}c \rangle|)}{2}$ is large.

Then, $|\langle z, \mathbf{S}c \rangle|$ is close to 0.

➡ Can be solved with a phase retrieval algorithm (machine learning).

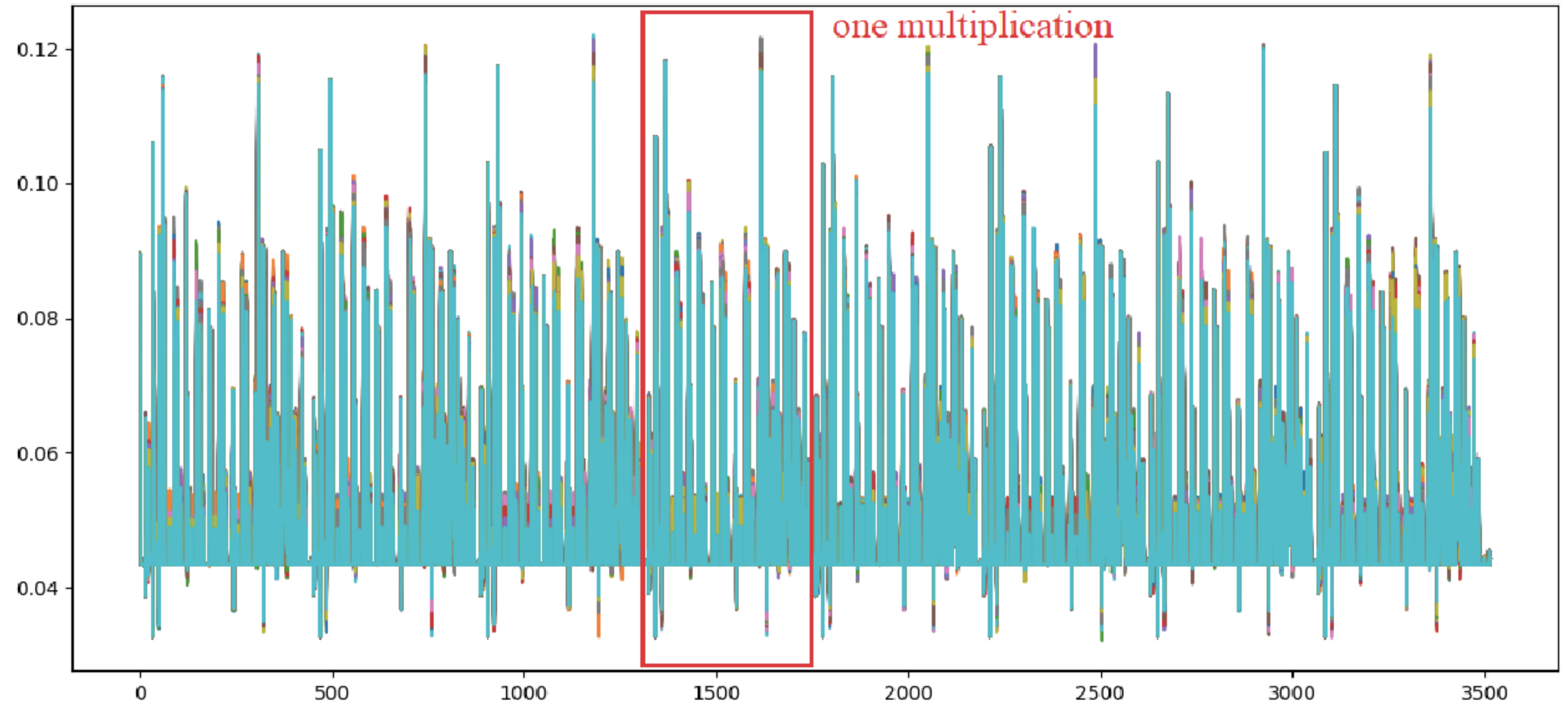
Full key recovery in an average of 40h on a powerful personal computer

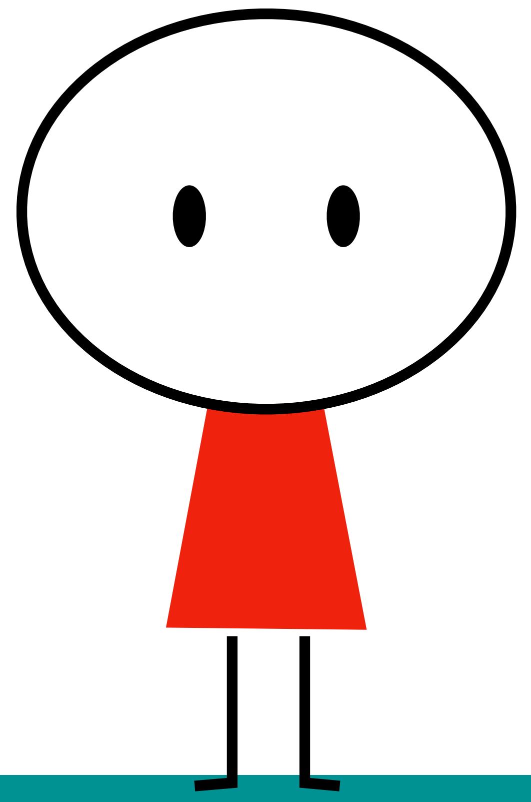
Power consumption attacks

<u>Power consumption attack:</u> the attacker knows the power consumption of the device executing the algorithm. He has access to « traces ».

Many attacks as well

- R. Primas, P. Pessl, S. Magnard. CHES'2017
- S. Bhasin, J.-P. D'Anvers, D. Heinz, T. Pöppelmann, M. Van Beirendonck. TCHES'2021
- B.-Y Sim, J. Kwon, J. Lee, I.-J. Kim, T. Lee, J. Han, H. Yoon, J. Choo, D.-G. Han. IEEE-ACESS'2020
- B.-Y. Sim, A. Park. eprint'2021
- P. Ravi, S. Sinha Roy, A. Chattopadhyay, S. Bhasin. CHES'2020
- E. Karabulut, A. Aysu. DAC'2021
- M. Guerreau, A. Martinellli, T. Ricosset, M. Rossi. TCHES'2022



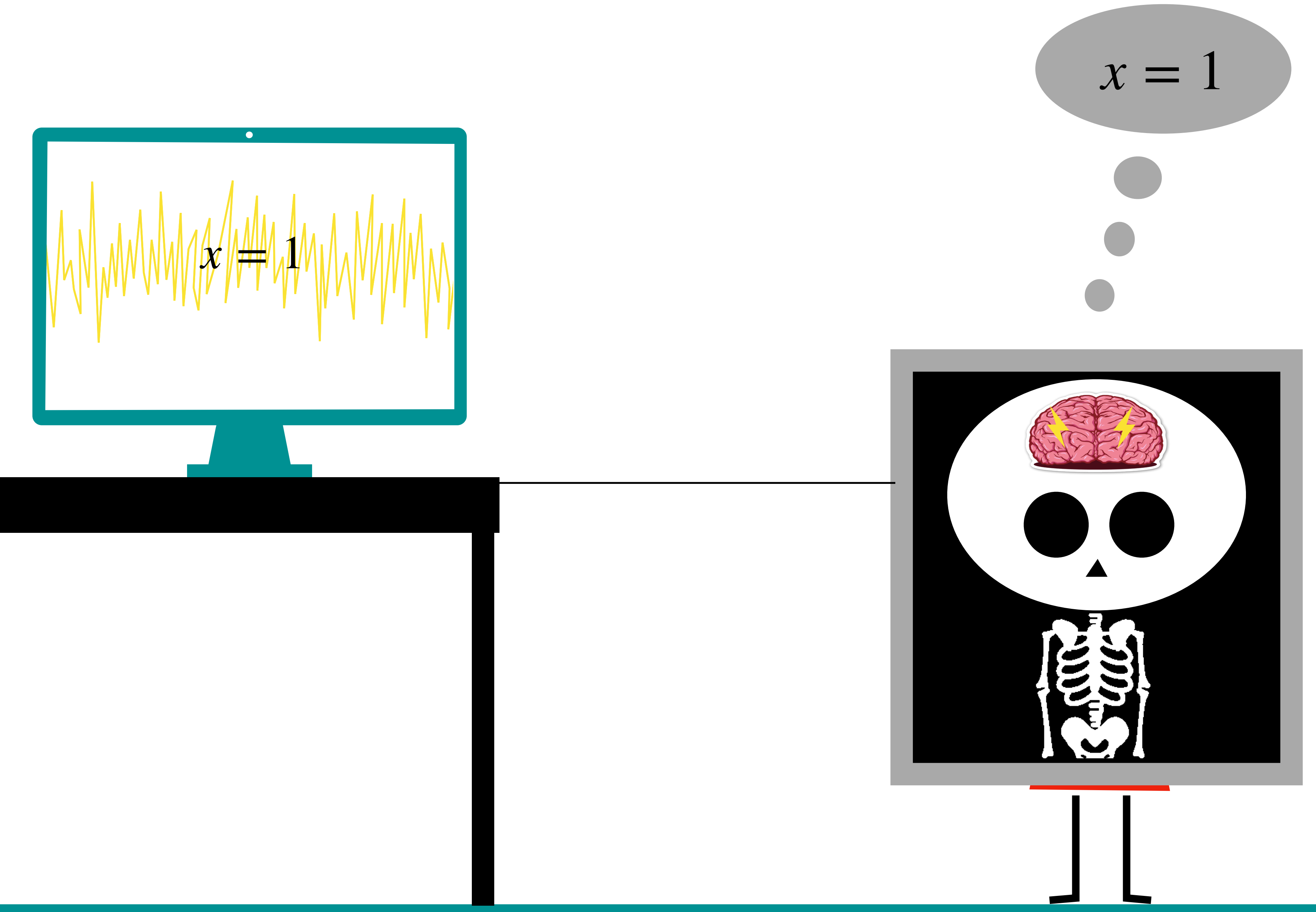


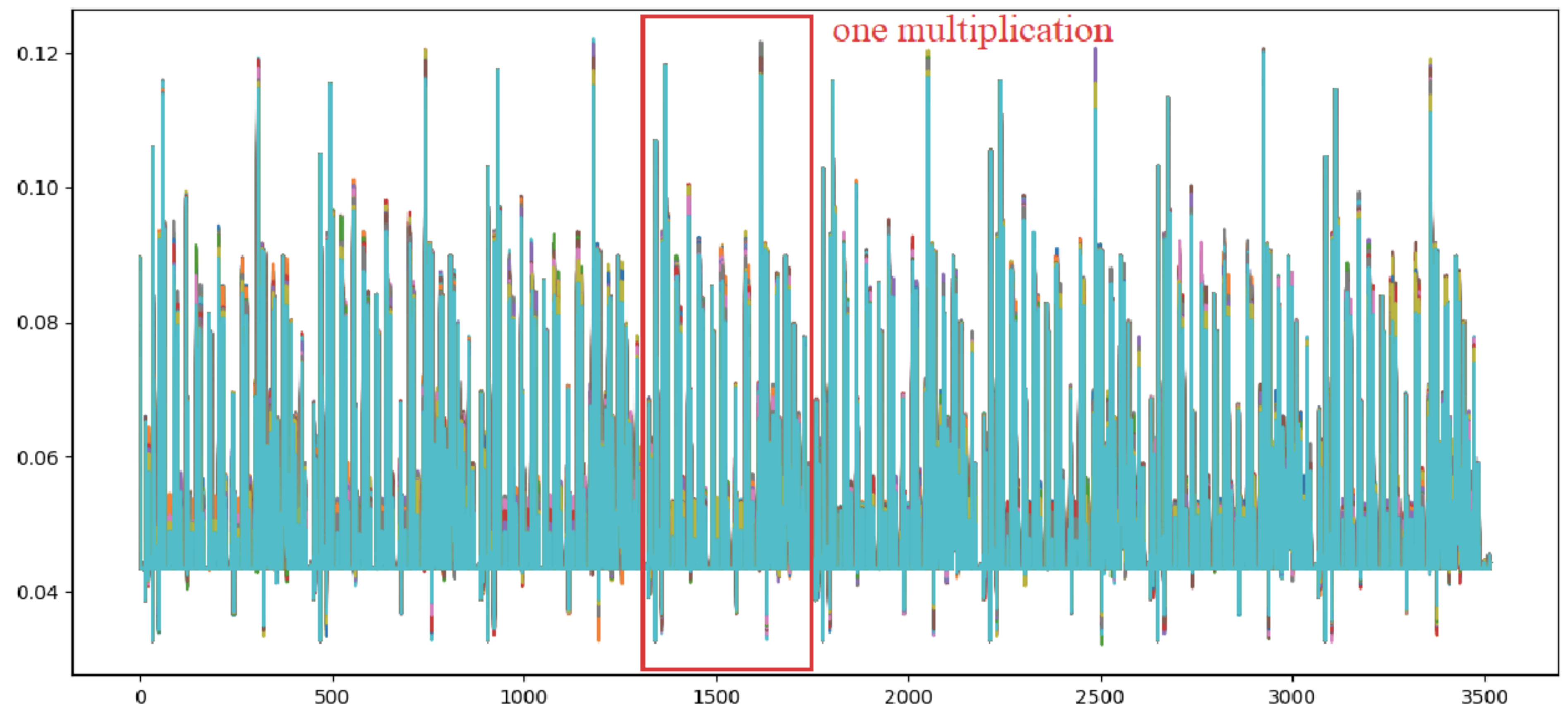
Power consumption attacks

Power consumption attack: the attacker knows the power consumption of the device executing the algorithm. He has access to « traces ».

Many attacks as well

- R. Primas, P. Pessl, S. Magnard. CHES'2017
- S. Bhasin, J.-P. D'Anvers, D. Heinz, T. Pöppelmann, M. Van Beirendonck. TCHES'2021
- B.-Y Sim, J. Kwon, J. Lee, I.-J. Kim, T. Lee, J. Han, H. Yoon, J. Choo, D.-G. Han. IEEE-ACESS'2020
- B.-Y. Sim, A. Park. eprint'2021
- P. Ravi, S. Sinha Roy, A. Chattopadhyay, S. Bhasin. CHES'2020
- E. Karabulut, A. Aysu. DAC'2021
- M. Guerreau, A. Martinellli, T. Ricosset, M. Rossi. TCHES'2022





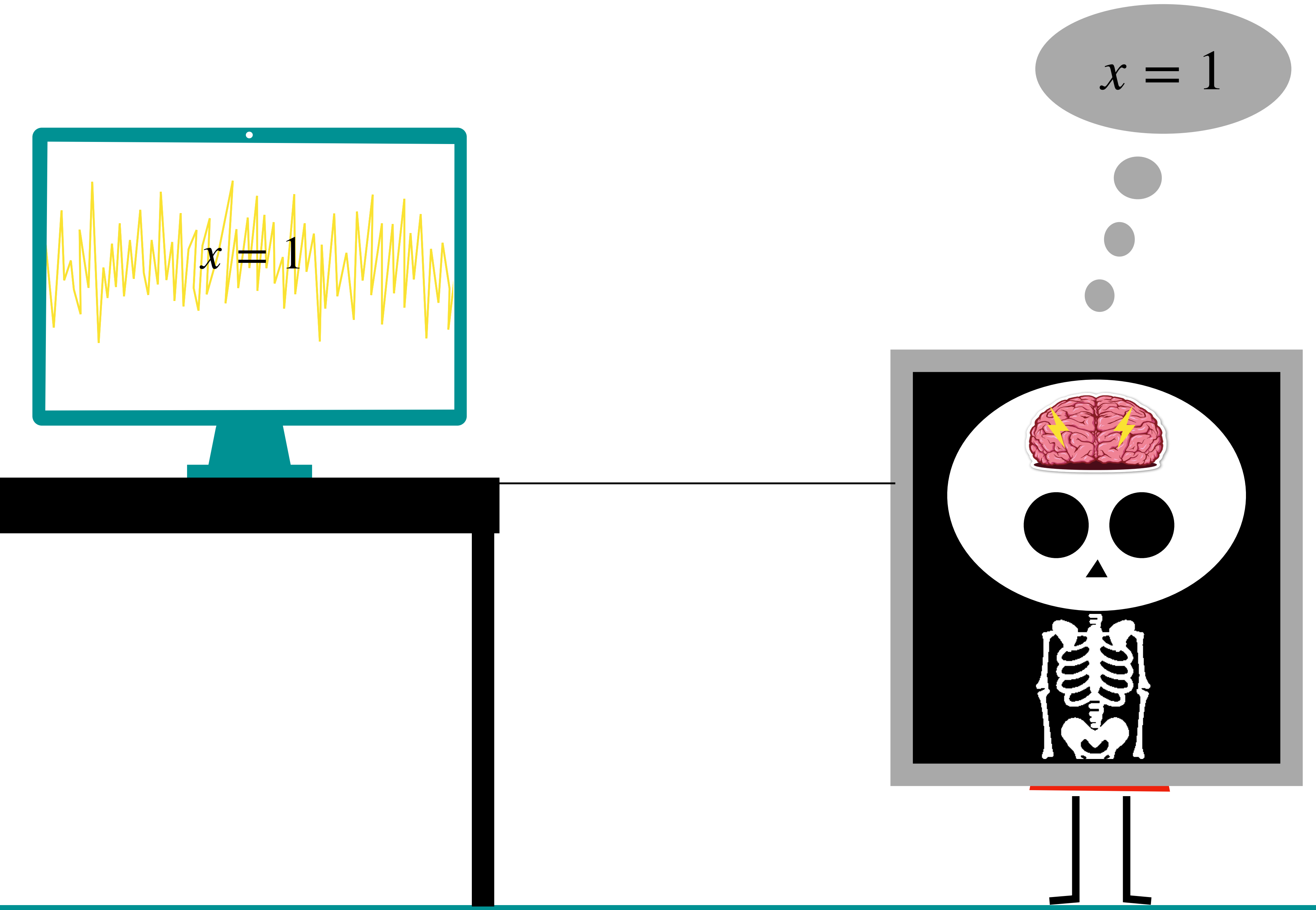
Power consumption attacks

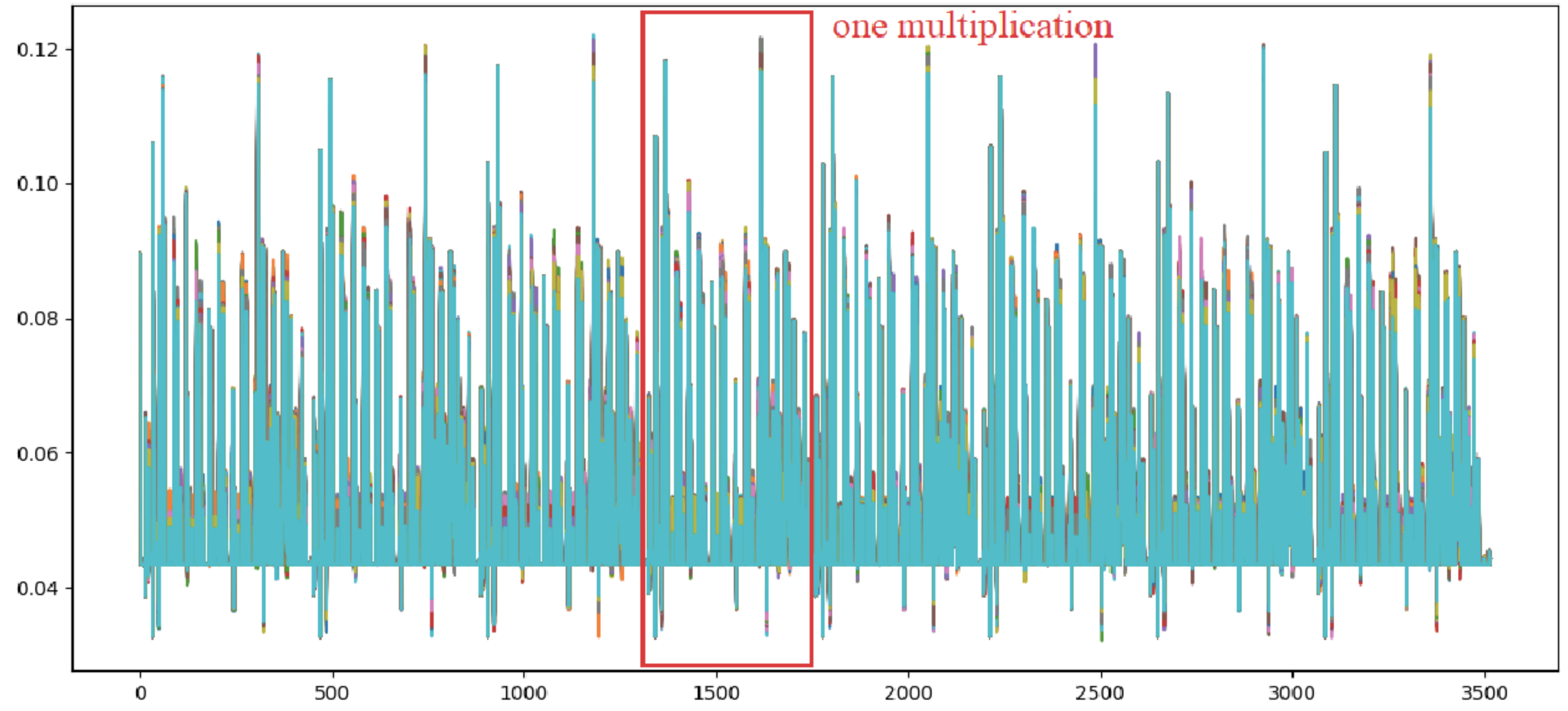
Power consumption attack: the attacker knows the power consumption of the device executing the algorithm. He has access to « traces ».

Many attacks as well

- R. Primas, P. Pessl, S. Magnard. CHES'2017
- S. Bhasin, J.-P. D'Anvers, D. Heinz, T. Pöppelmann, M. Van Beirendonck. TCHES'2021
- B.-Y Sim, J. Kwon, J. Lee, I.-J. Kim, T. Lee, J. Han, H. Yoon, J. Choo, D.-G. Han. IEEE-ACESS'2020
- B.-Y. Sim, A. Park. eprint'2021
- P. Ravi, S. Sinha Roy, A. Chattopadhyay, S. Bhasin. CHES'2020
- E. Karabulut, A. Aysu. DAC'2021
- M. Guerreau, A. Martinellli, T. Ricosset, M. Rossi. TCHES'2022

An example presented in the next slides





Falcon signature scheme

Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{c}\mathbf{A} = H(m)$

2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$

3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Falcon signature scheme

Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Take **a close** vector but not the closest.

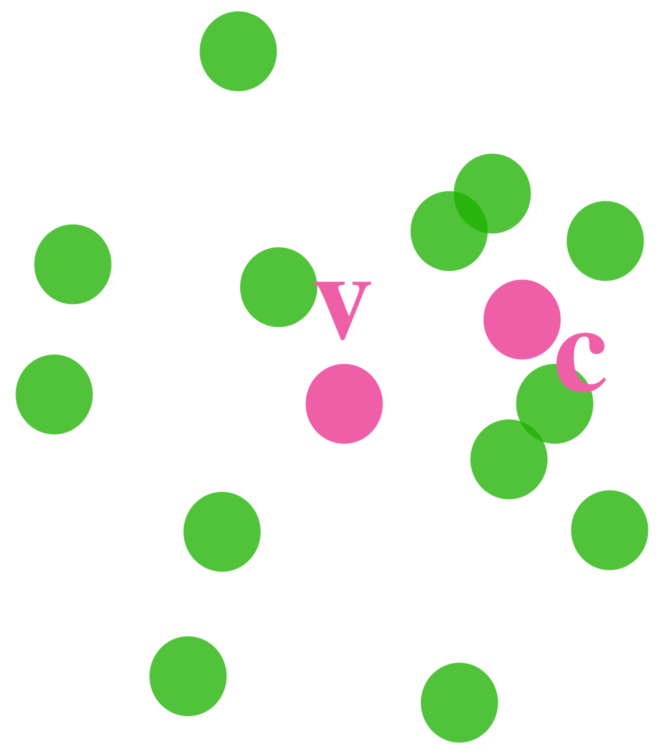
Falcon signature scheme

Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Take **a close** vector but not the closest.

Take the closest vector
Add a Gaussian random shift z_0

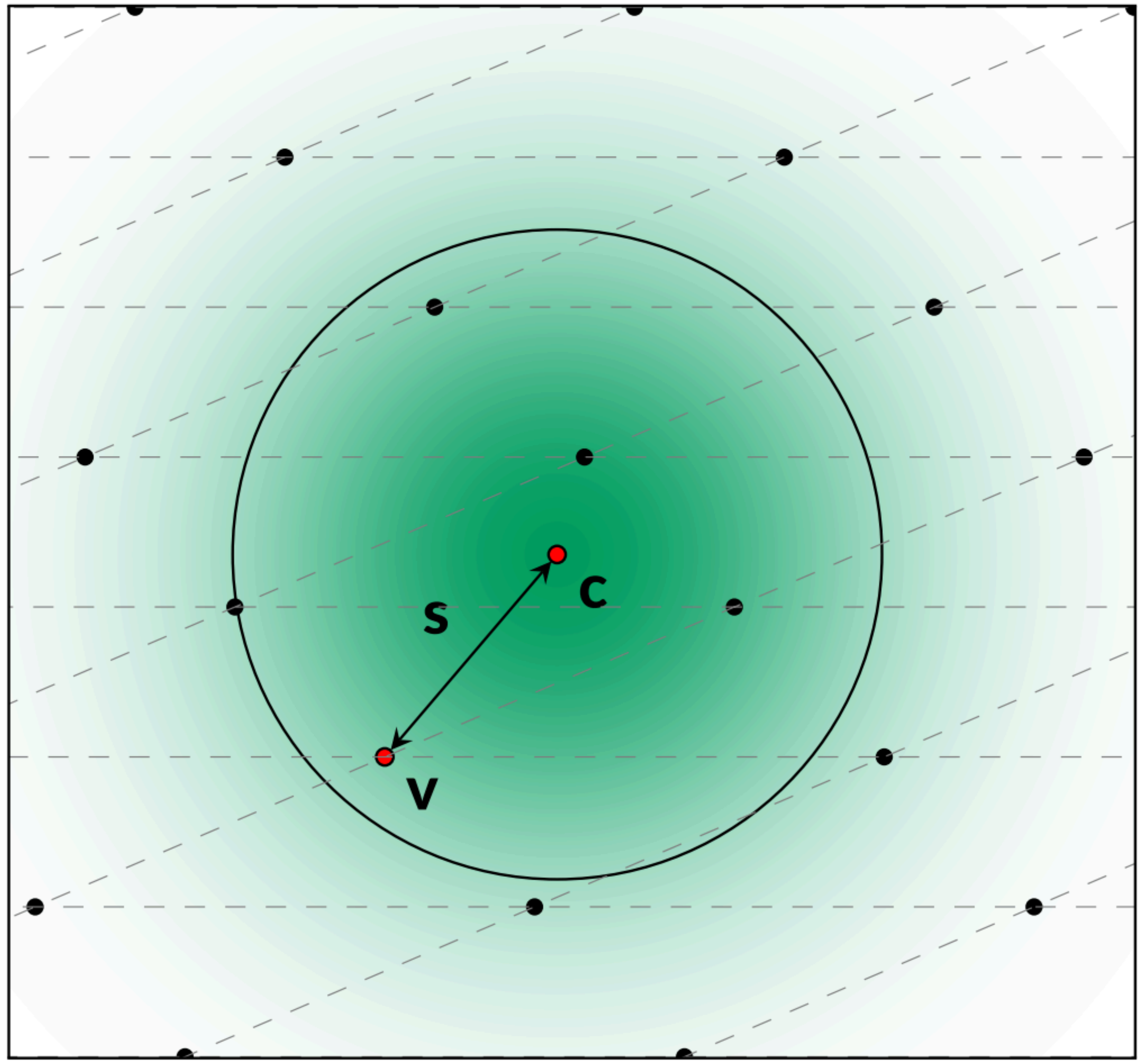


Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Take **a close** vector but not the closest.

Take the closest vector
Add a Gaussian random shift z_0



Falcon signature scheme

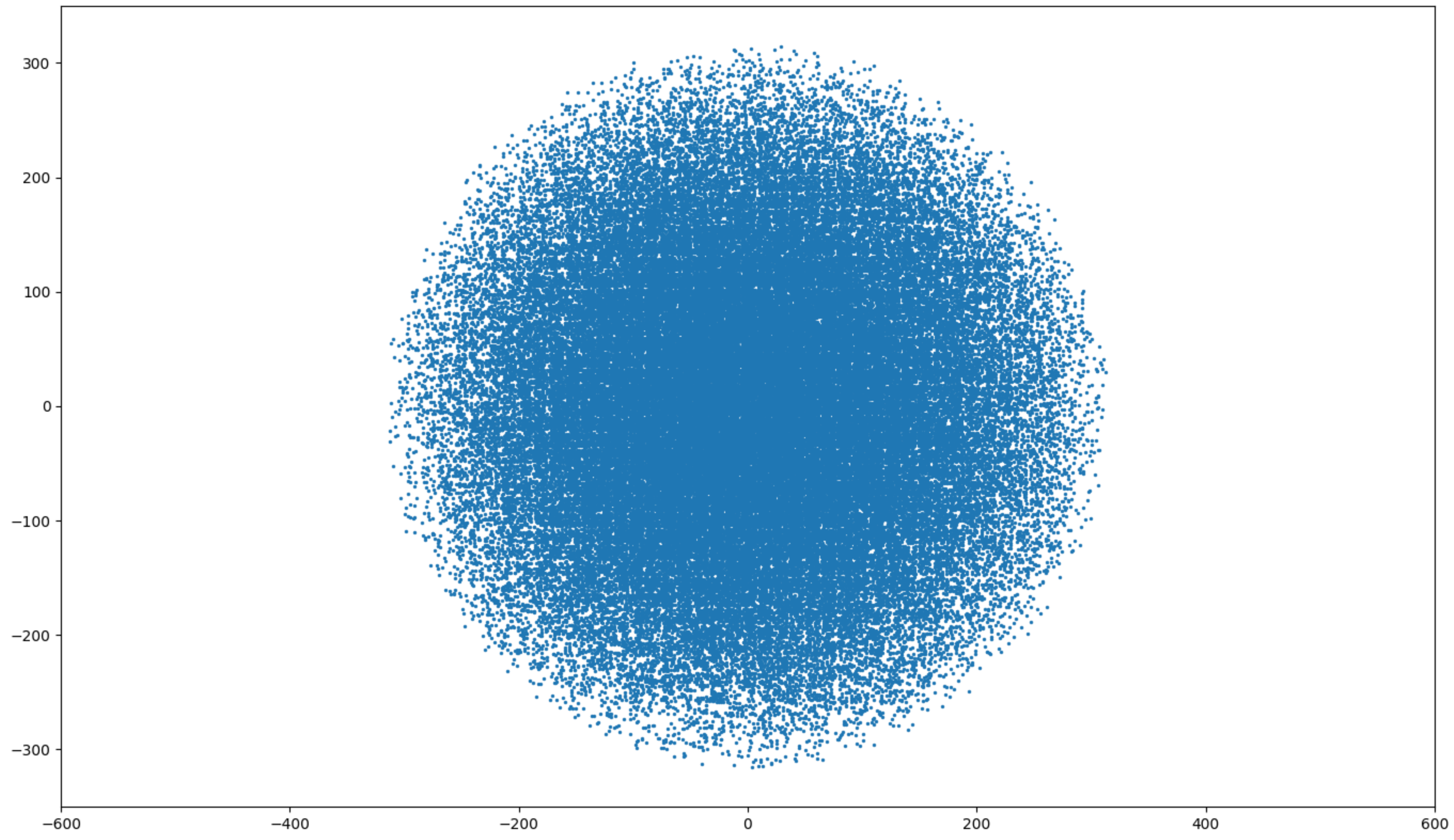
Signature algorithm:

1: compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
2: $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
3: return $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Take **a close** vector but not the closest.

Take the closest vector
Add a Gaussian random shift z_0

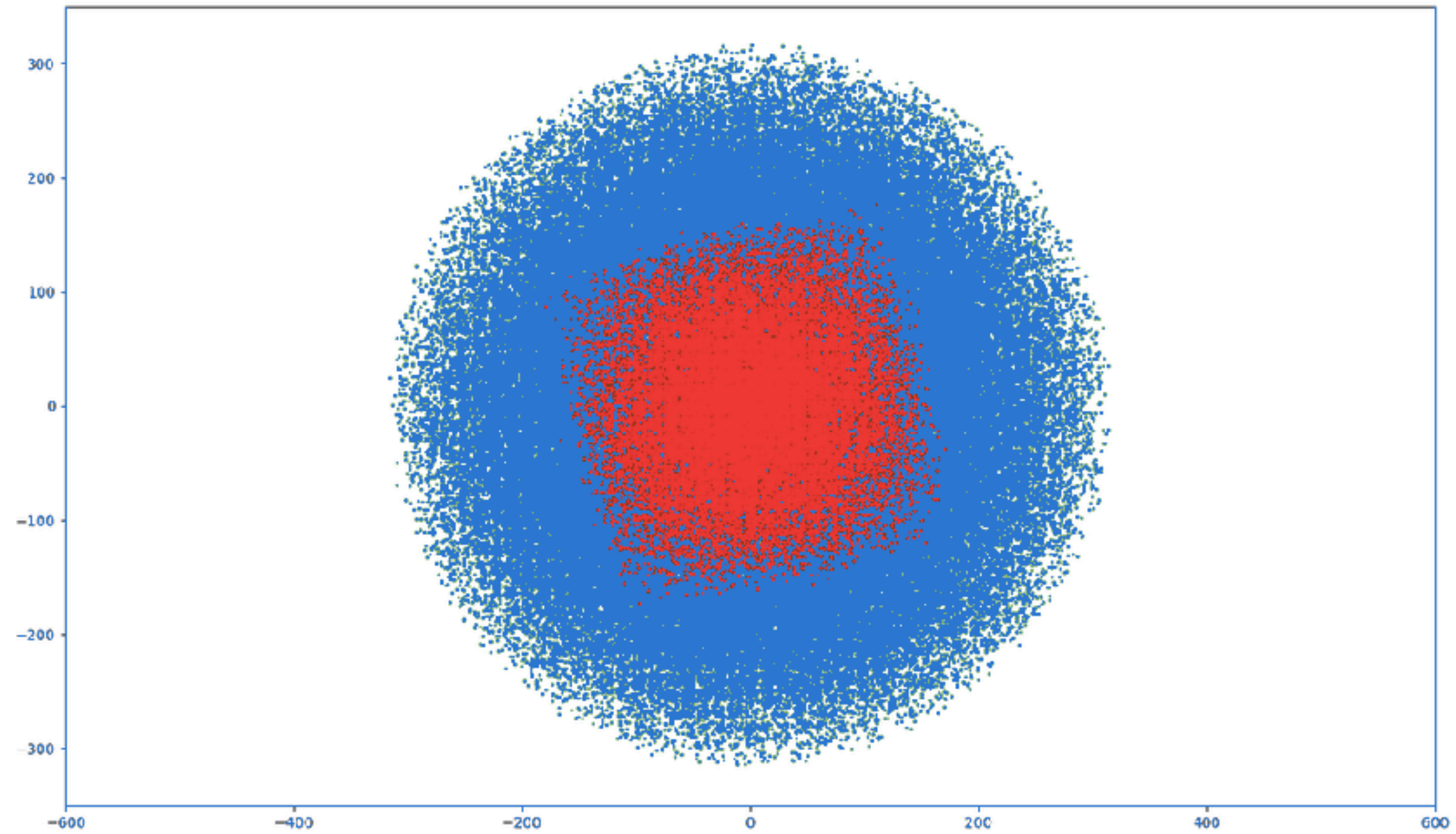
Distribution of signatures



Intuition of the power analysis attack of Falcon

Intuition of the attack

If we select the inputs such that the **Gaussian shift** is zero, we can “see” the hidden basis.



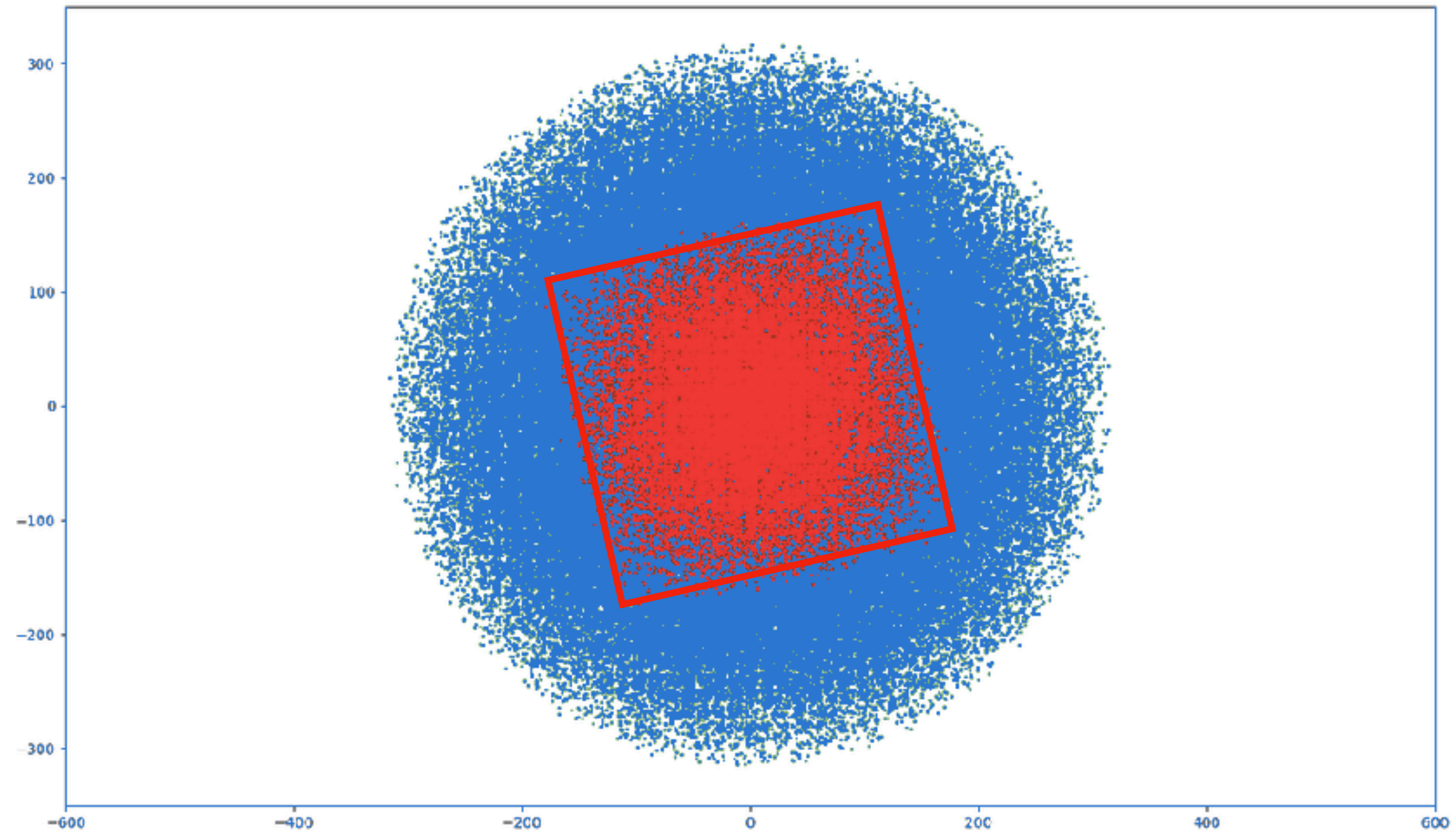
Non shifted signatures in red

What about high dimensions? There is a negligible amount of zero-shift in *all* 512 dimensions.

Intuition of the power analysis attack of Falcon

Intuition of the attack

If we select the inputs such that the **Gaussian shift** is zero, we can "see" the hidden basis.



Non shifted signatures in red

What about high dimensions? There is a negligible amount of zero-shift in *all* 512 dimensions.

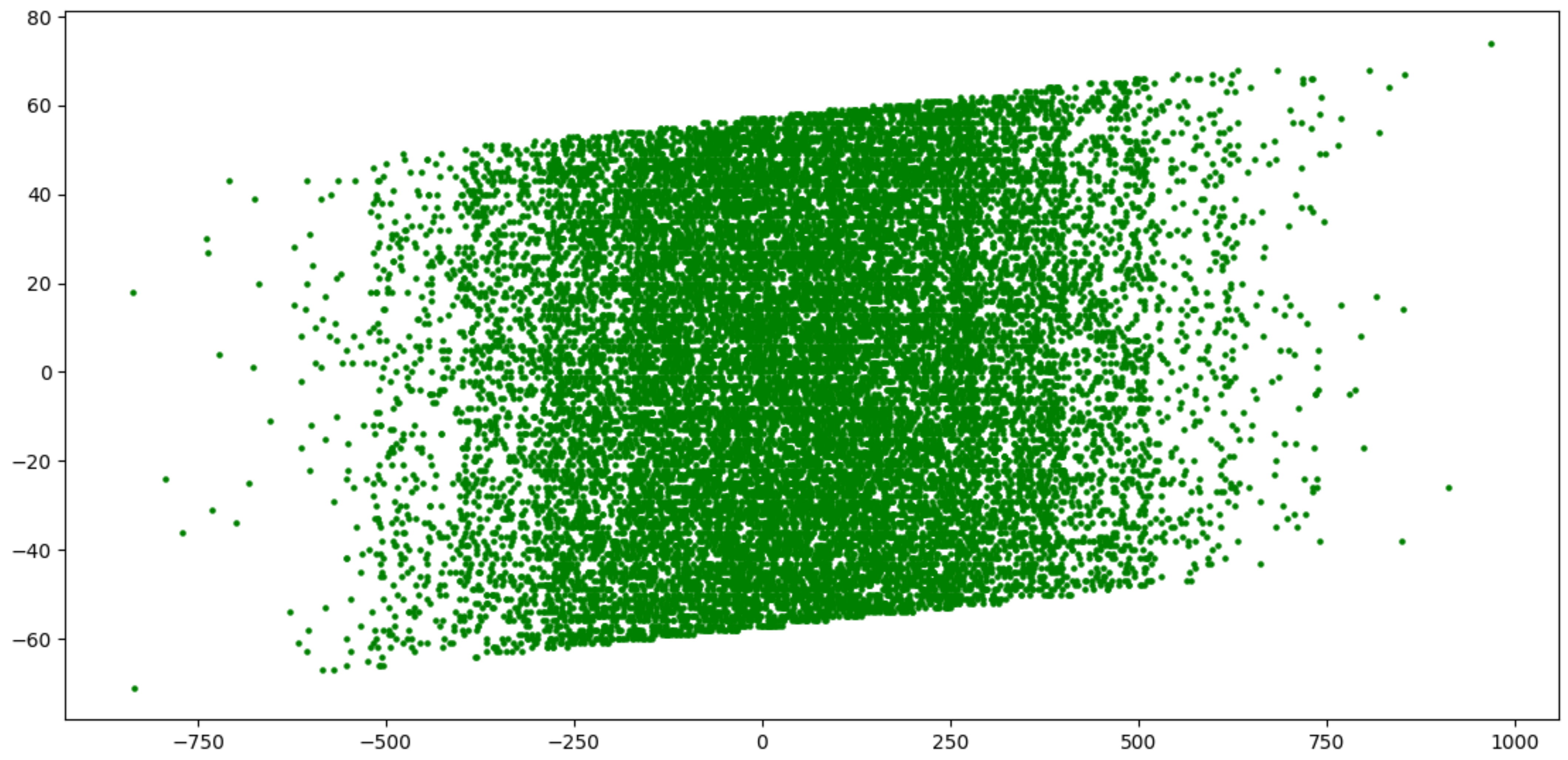
Single trace power analysis of Falcon

We focus on one dimension.
A **single trace analysis** can provide the information: shift $= 0$ or $\neq 0$.

Single trace power analysis of Falcon

We focus on one dimension.
A **single trace analysis** can provide the information: shift $= 0$ or $\neq 0$.

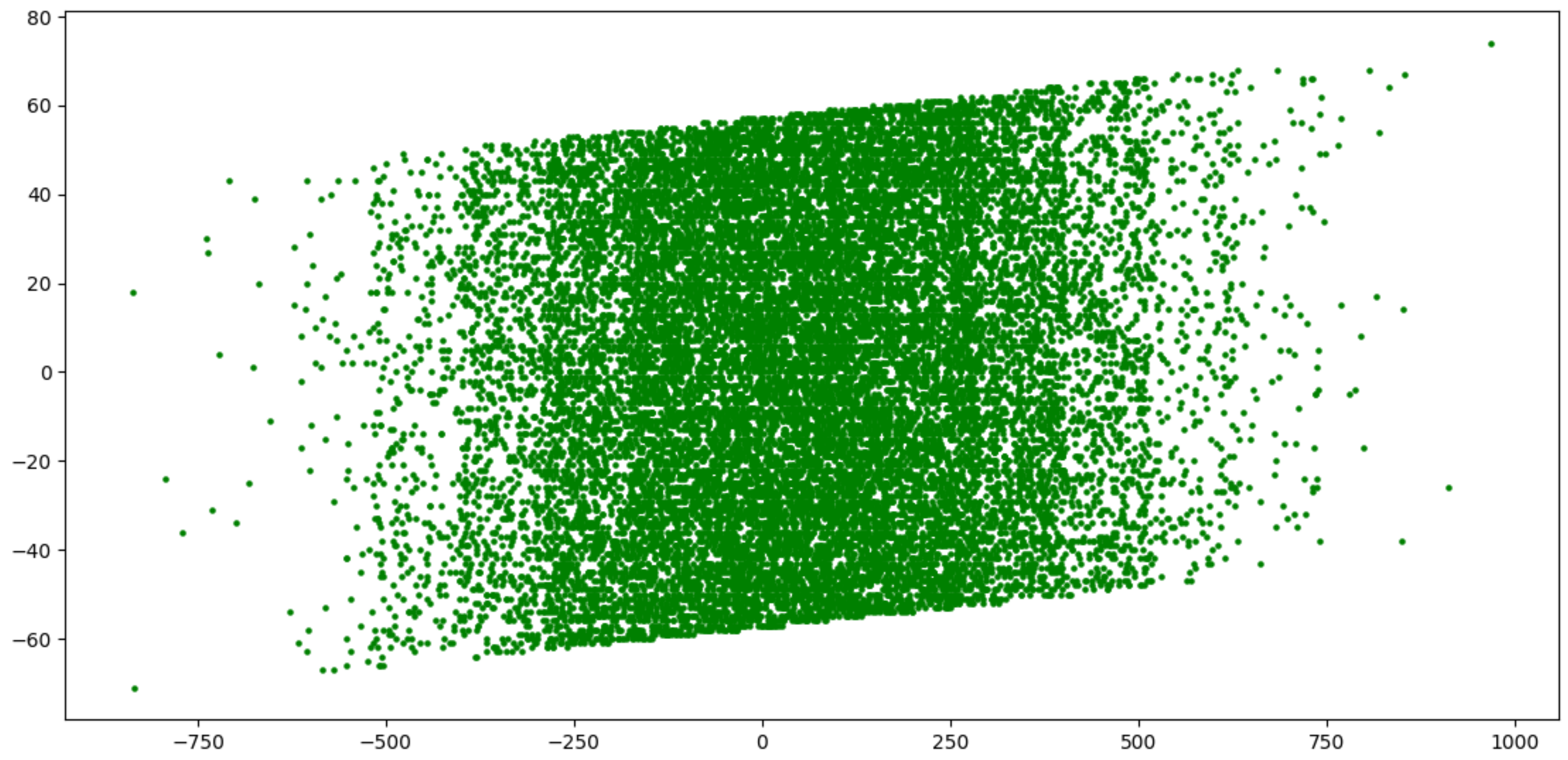
Signatures for which shift $= 0$ in the first coordinate



Single trace power analysis of Falcon

We focus on one dimension.
A **single trace analysis** can provide the information: shift $= 0$ or $\neq 0$.

Signatures for which shift $= 0$ in the first coordinate



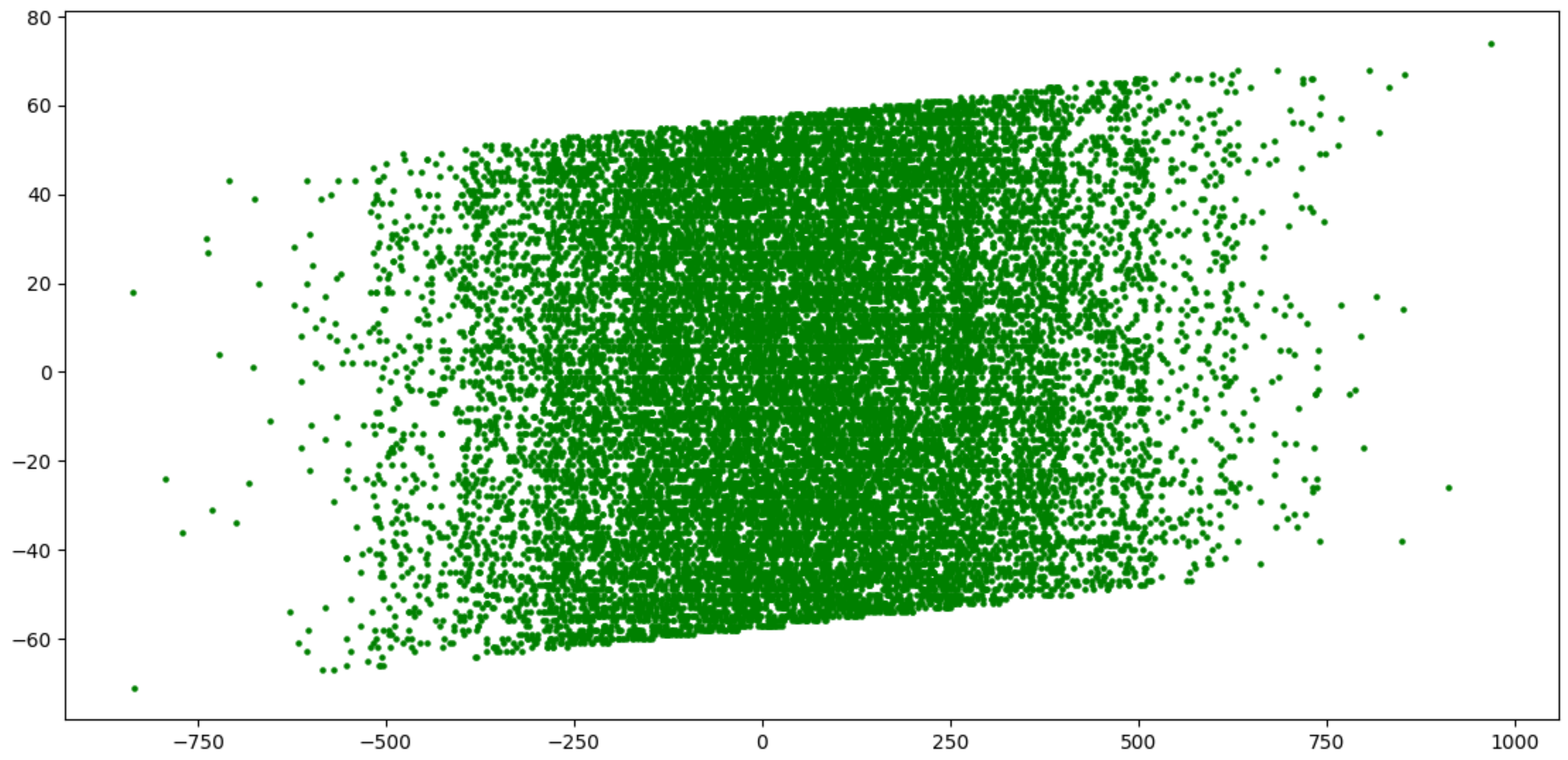
➡ It is possible to apply **a partial hidden parallelepiped recovery.**

- P. Nguyen, O. Regev Eurocrypt'2006
- L. Ducas, P. Nguyen Asiacrypt'2012

Single trace power analysis of Falcon

We focus on one dimension.
A **single trace analysis** can provide the information: shift $= 0$ or $\neq 0$.

Signatures for which shift $= 0$ in the first coordinate



➡ It is possible to apply **a partial hidden parallelepiped recovery.**

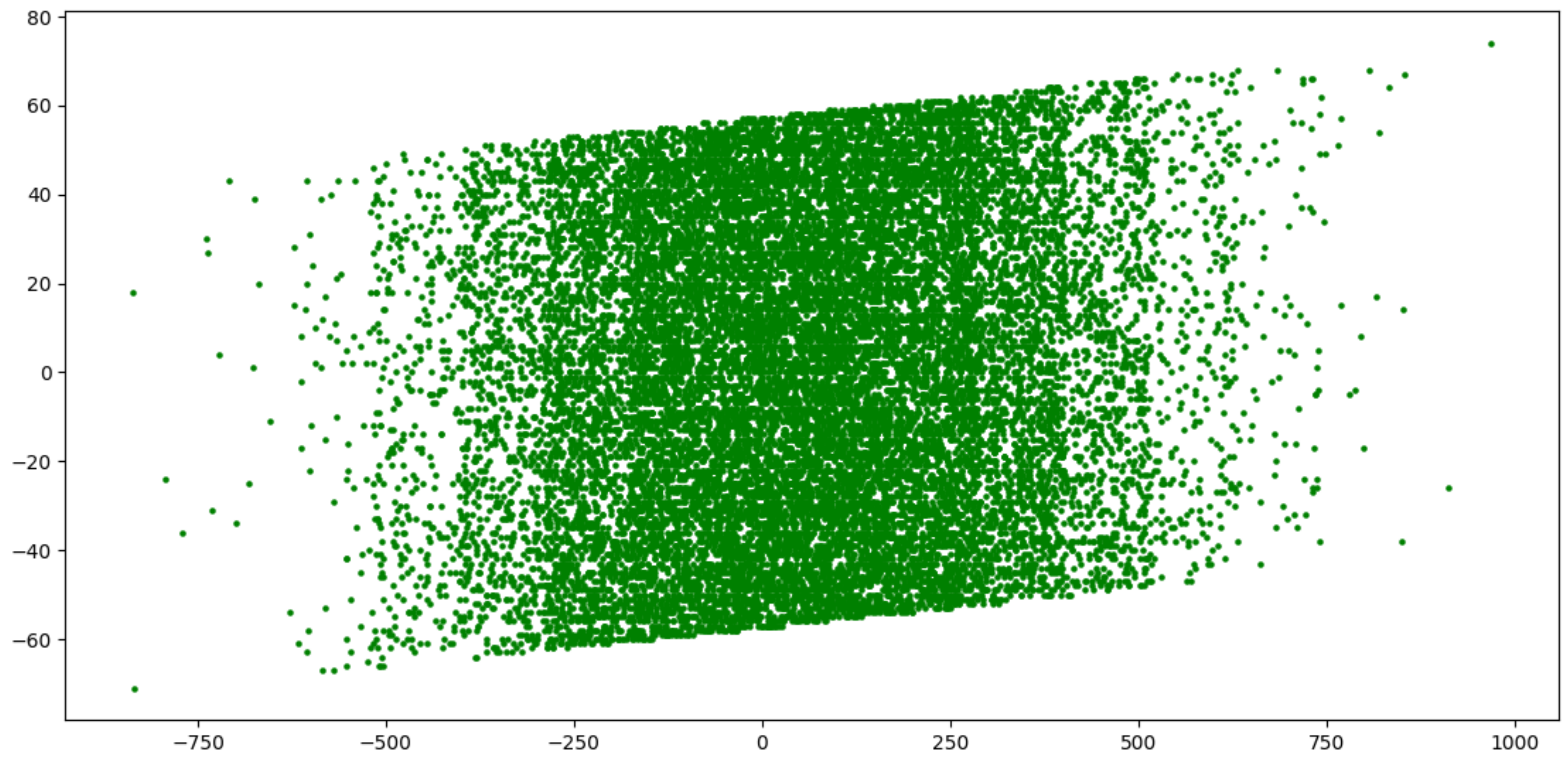
- P. Nguyen, O. Regev Eurocrypt'2006
- L. Ducas, P. Nguyen Asiacrypt'2012

We recover one vector of the basis, this is enough to **recover the full basis** thanks to the structure of the private key.

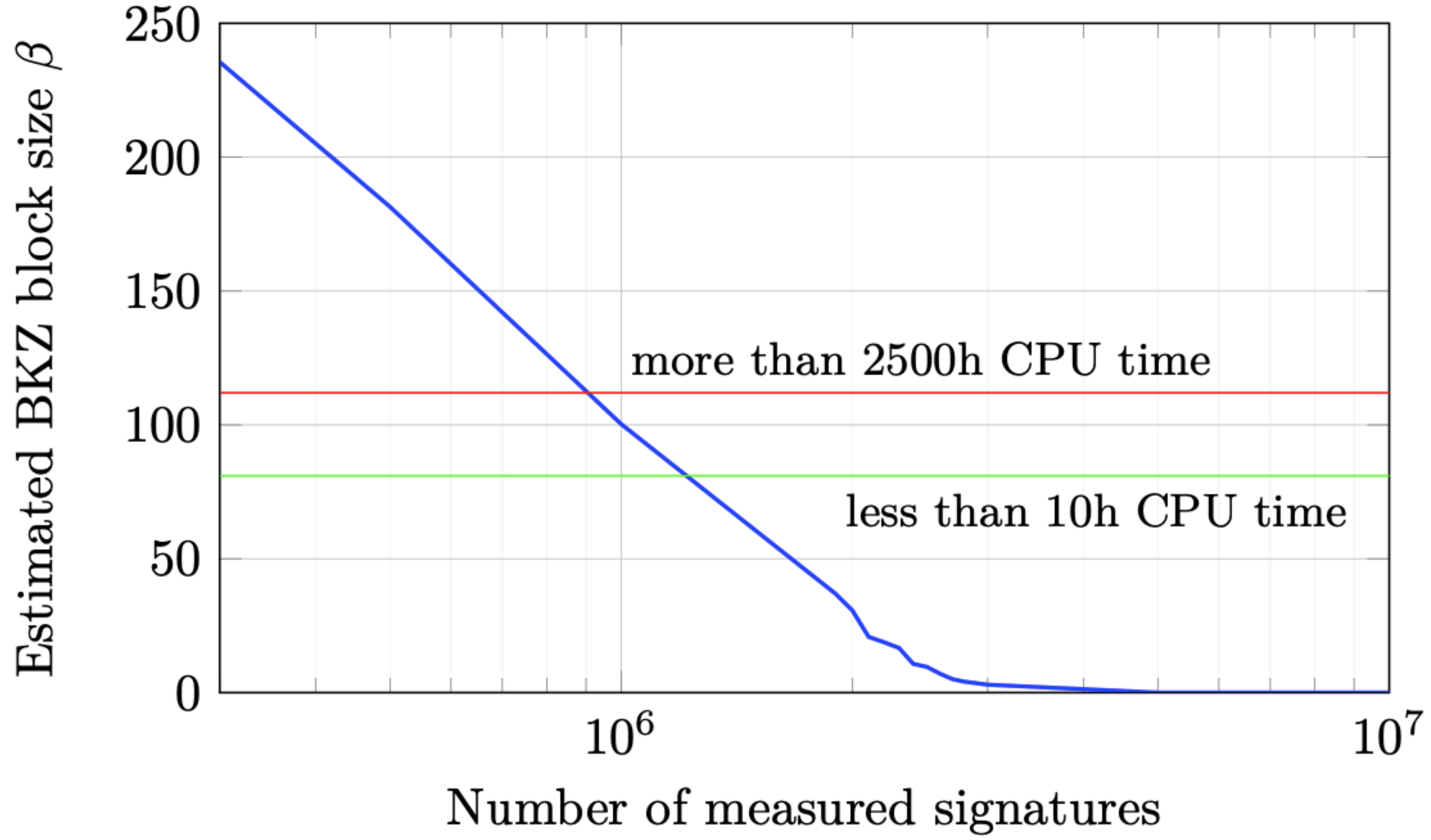
Single trace power analysis of Falcon

We focus on one dimension.
A **single trace analysis** can provide the information: shift $= 0$ or $\neq 0$.

Signatures for which shift $= 0$ in the first coordinate



Performance of the attack



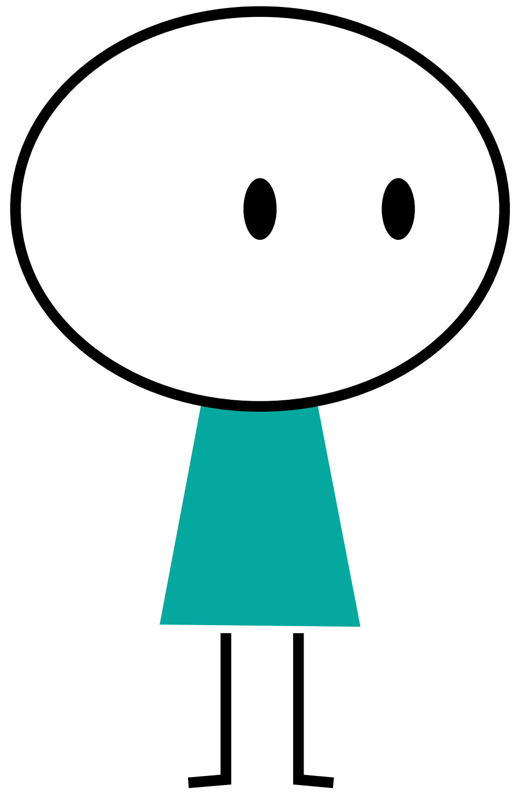
➡ It is possible to apply **a partial hidden parallelepiped recovery.**

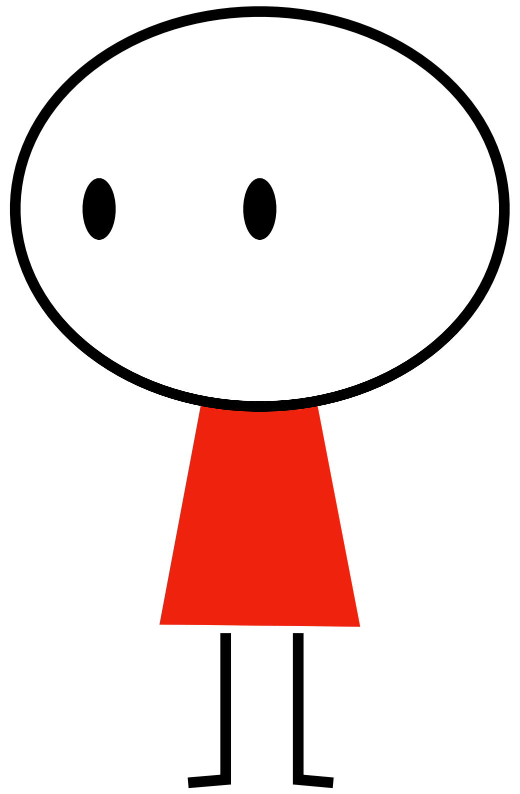
- P. Nguyen, O. Regev Eurocrypt'2006
- L. Ducas, P. Nguyen Asiacrypt'2012

We recover one vector of the basis, this is enough to **recover the full basis** thanks to the structure of the private key.

Decryption failure attacks

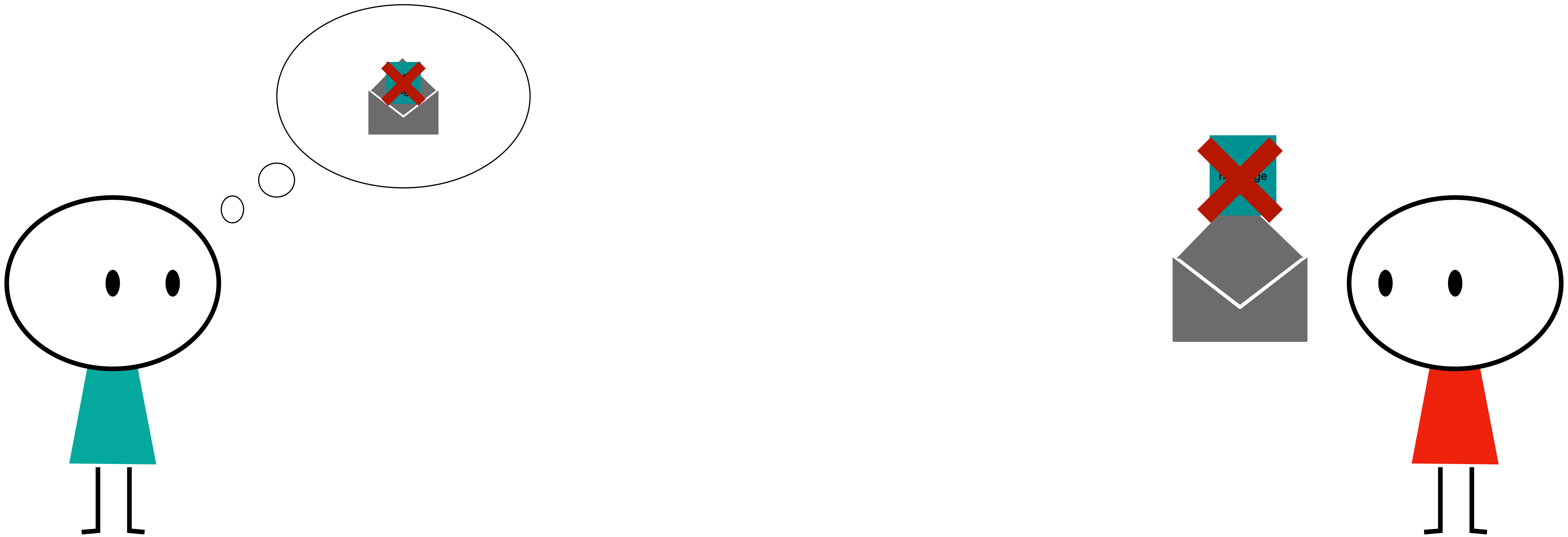
- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.





Decryption failure attacks

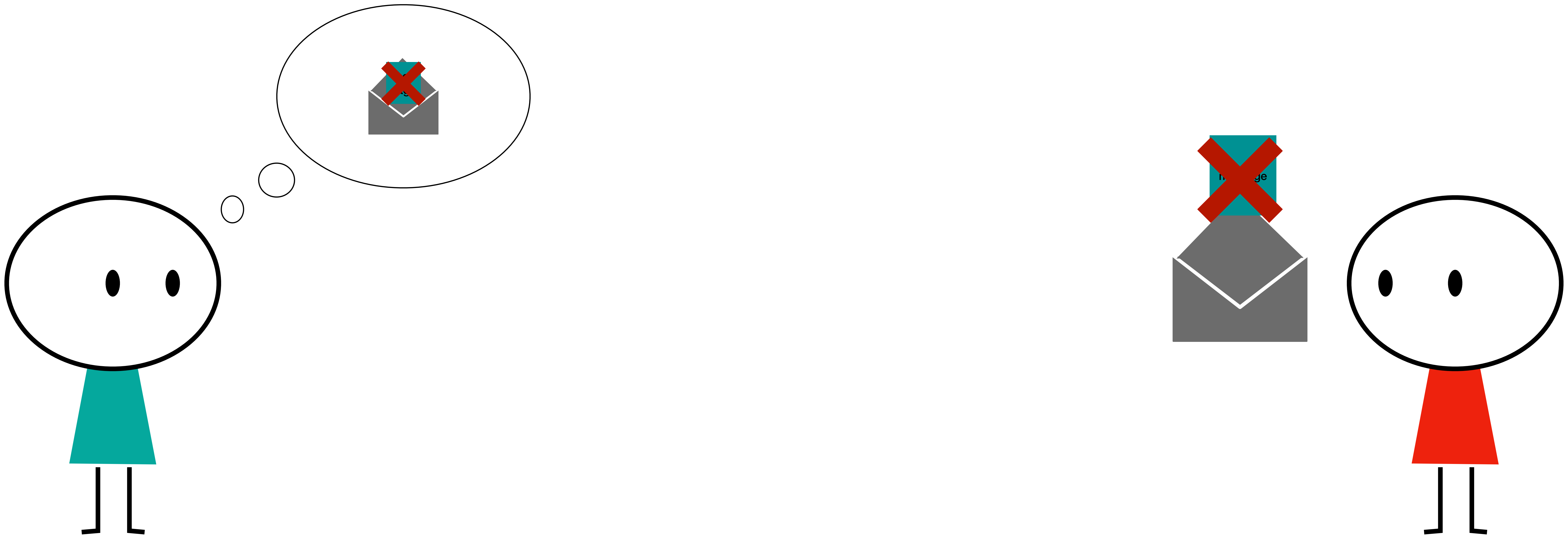
- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.



Decryption failure attacks

- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

Recall that $m' \approx m + \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil$  $\iff \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil \geq \frac{1}{2}$



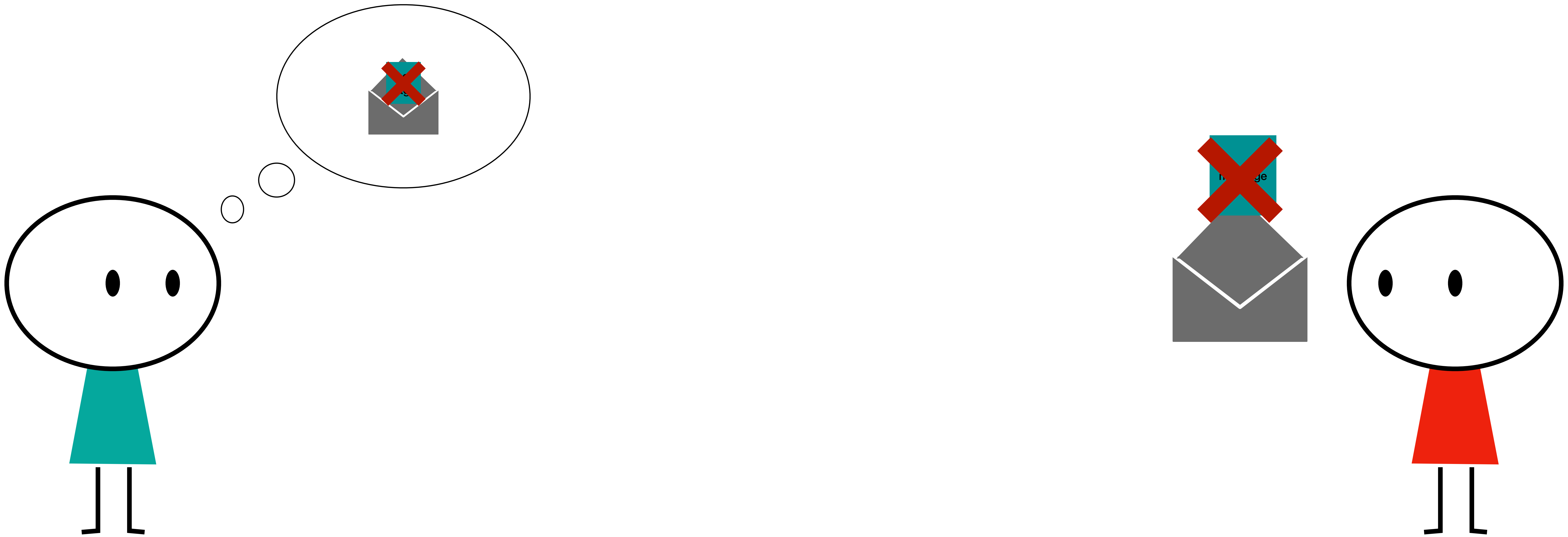
Decryption failure attacks

- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

Recall that $m' \approx m + \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil$

 $\iff \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil \geq \frac{1}{2}$

 $: \left| \mathbf{s}^T \mathbf{w} \right| \geq \frac{q}{4}$



Decryption failure attacks

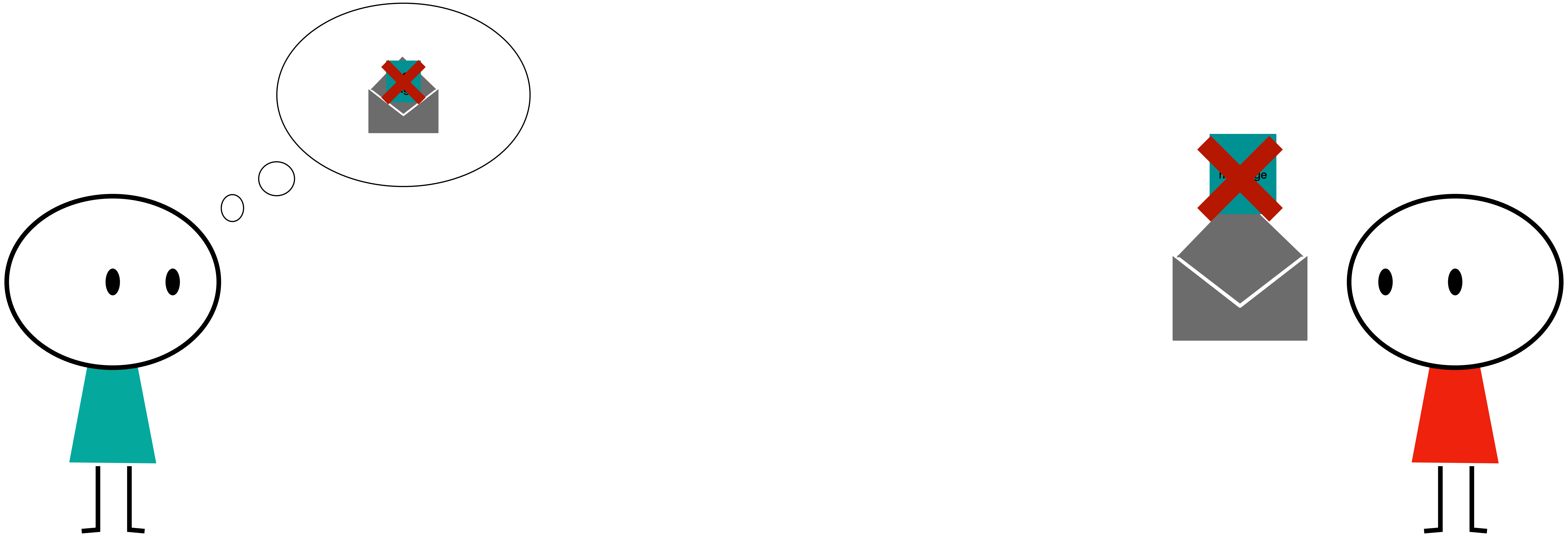
- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

Recall that $m' \approx m + \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}'+\mathbf{e}''-\mathbf{z}^T\mathbf{e}'\right) \right\rceil$

 $\iff \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}'+\mathbf{e}''-\mathbf{z}^T\mathbf{e}'\right) \right\rceil \geq \frac{1}{2}$

 $: \left|\mathbf{s}^T\mathbf{w}\right| \geq \frac{q}{4}$

 $: \mathbf{s}^T\mathbf{w} \geq \frac{q}{4}$



Decryption failure attacks

- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

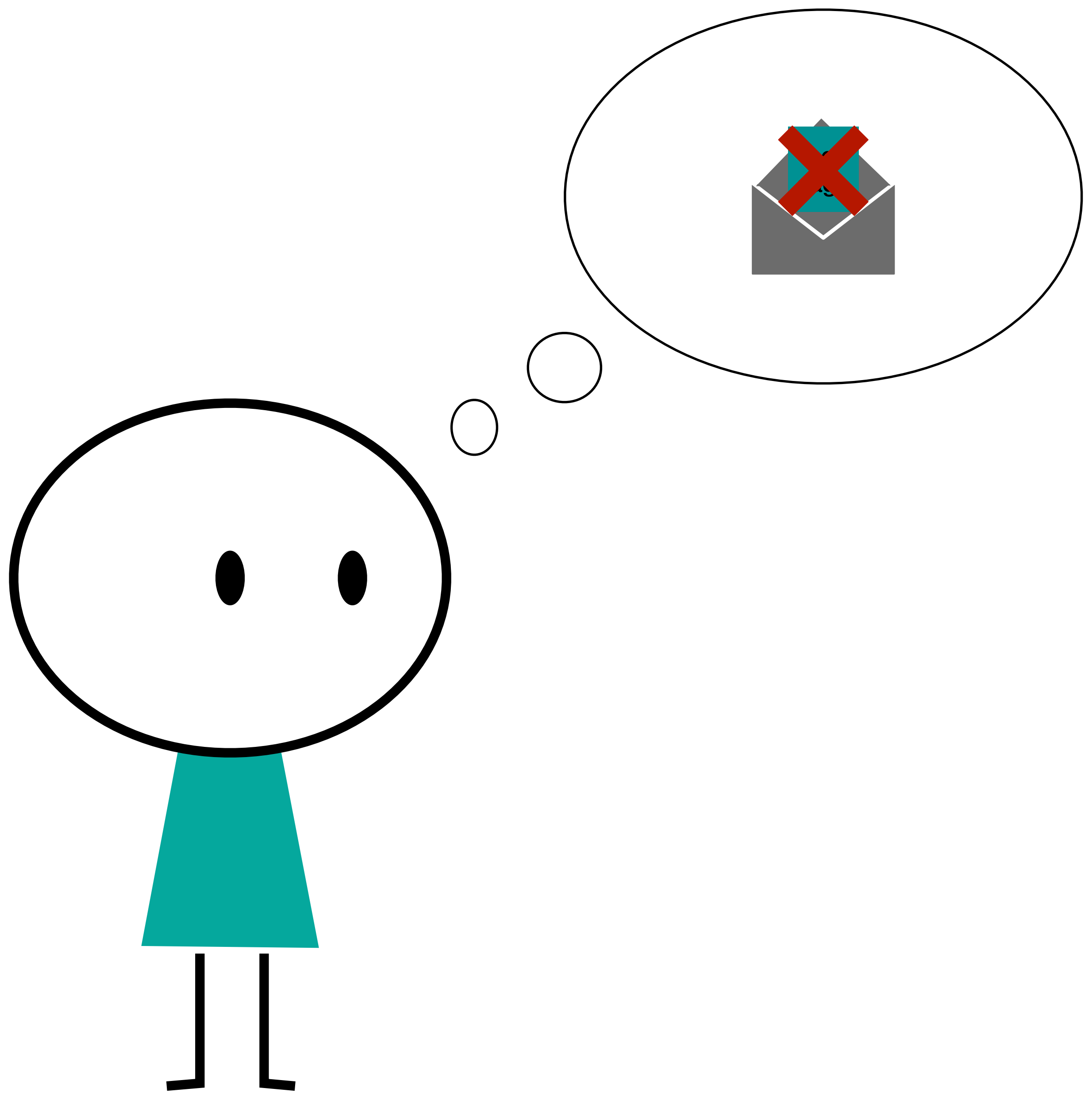
Recall that $m' \approx m + \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil$

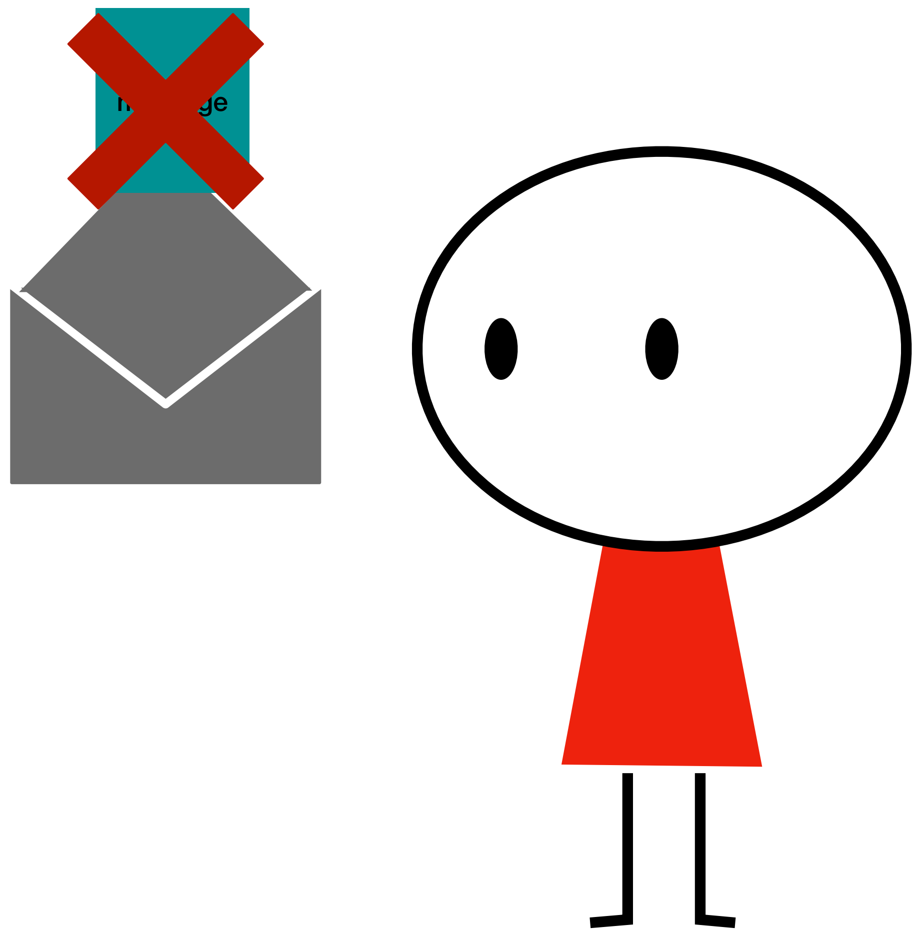
 $\iff \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil \geq \frac{1}{2}$

 $: \left|\mathbf{s}^T\mathbf{w}\right| \geq \frac{q}{4}$

 $: \mathbf{s}^T\mathbf{w} \geq \frac{q}{4}$

 $: \mathbf{s} \approx k \cdot \mathbf{w}$ $\quad (\mathbf{s} = k \cdot \mathbf{w} + \epsilon)$





Decryption failure attacks

- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

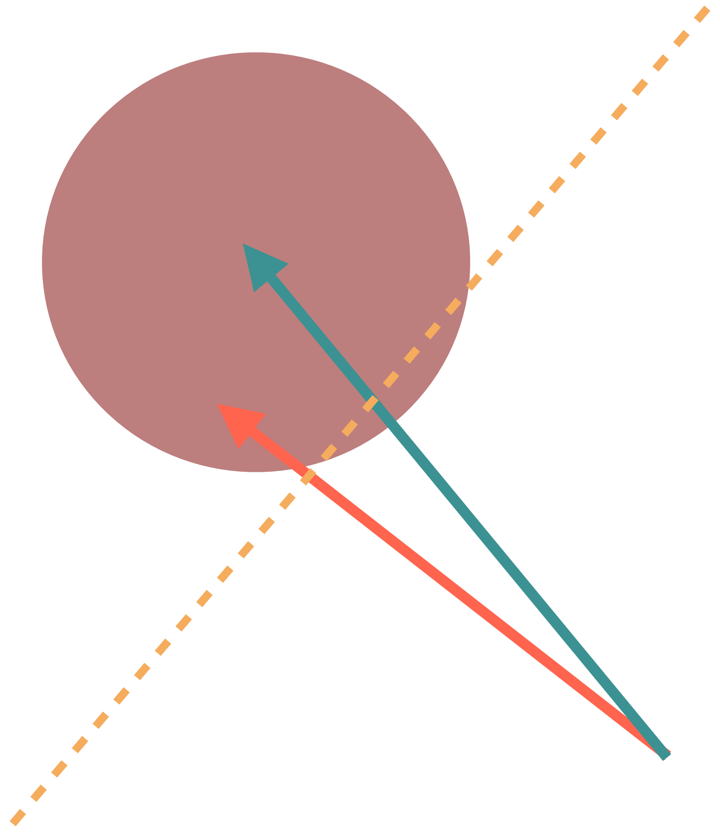
Recall that $m' \approx m + \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil$

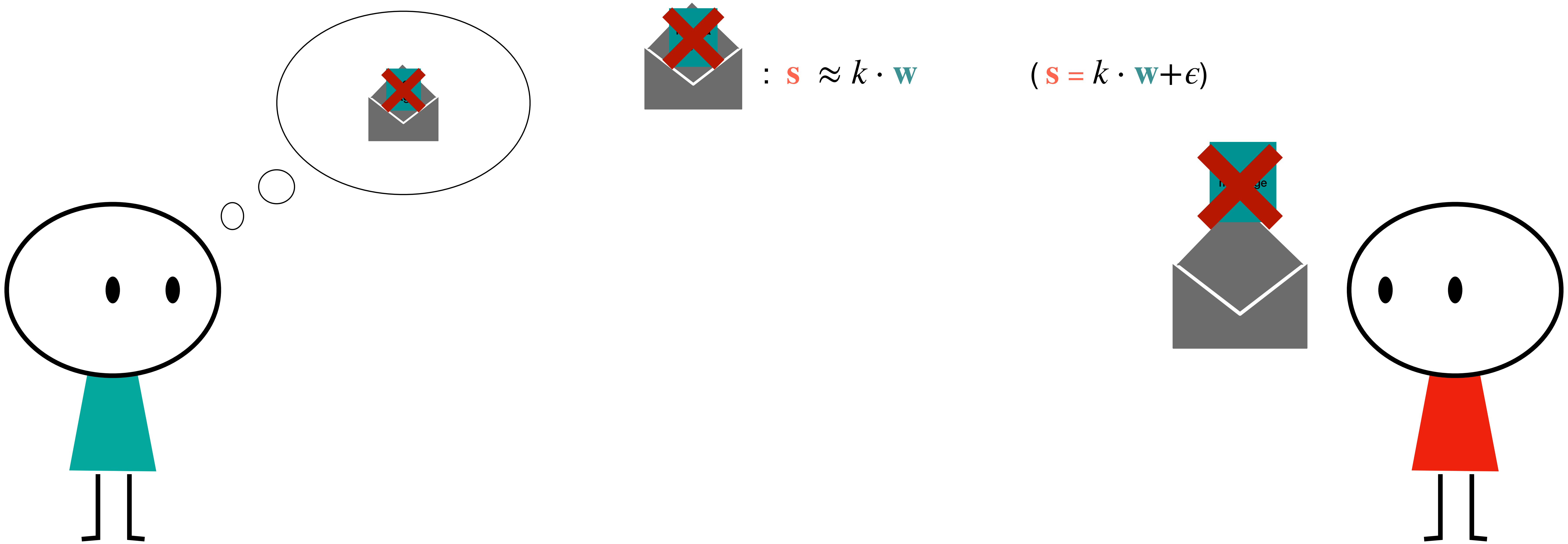
$\Longleftrightarrow \left\lfloor \frac{2}{q}\left(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}'\right) \right\rceil \geq \frac{1}{2}$

$: \left|\mathbf{s}^T \mathbf{w}\right| \geq \frac{q}{4}$

$: \mathbf{s}^T \mathbf{w} \geq \frac{q}{4}$

$: \mathbf{s} \approx k \cdot \mathbf{w}$ $\quad (\mathbf{s} = k \cdot \mathbf{w} + \epsilon)$





Decryption failure attacks

- J.-P. D'Anvers, Q. Guo, T. Johansson, A. Nilsson, F. Vercauteren, and I. Verbauwhede. PKC'19
- Dachman-Soled, L. Ducas, H. Gong and M. Rossi. CRYPTO'2020.

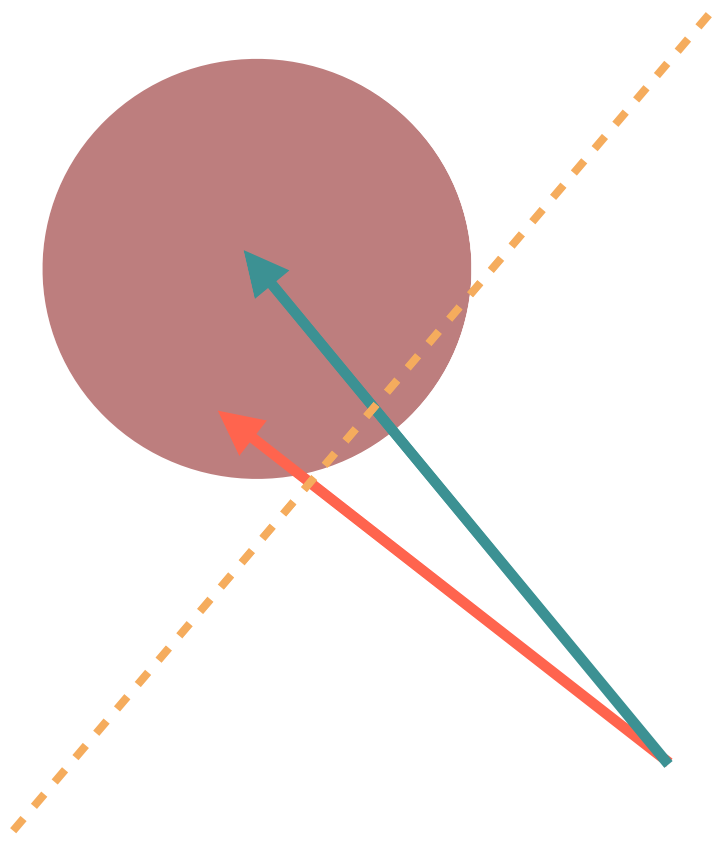
Recall that $m' \approx m + \left\lfloor \frac{2}{q}(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}') \right\rceil$

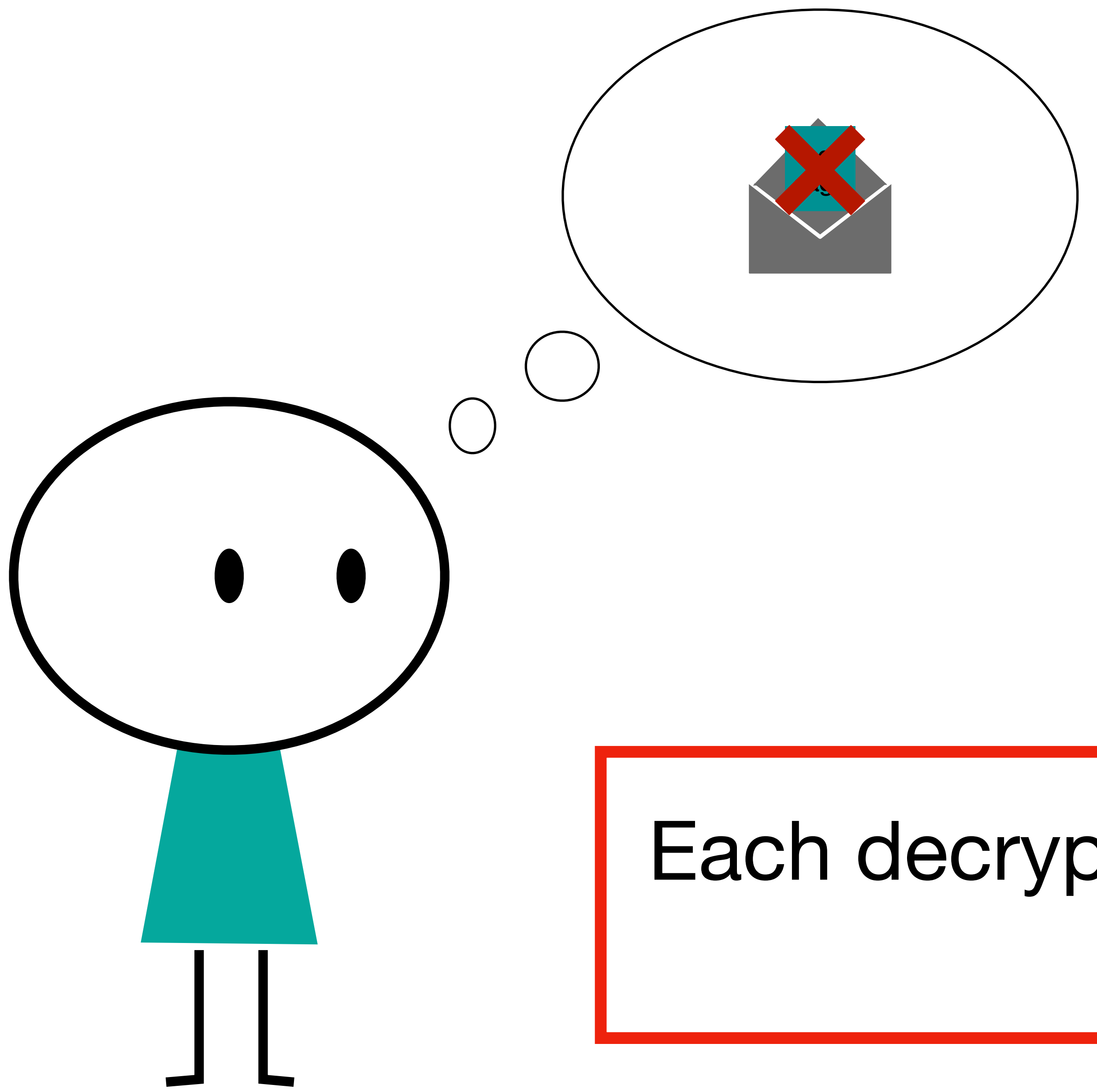
$\iff \left\lfloor \frac{2}{q}(\mathbf{e}^T\mathbf{z}' + \mathbf{e}'' - \mathbf{z}^T\mathbf{e}') \right\rceil \geq \frac{1}{2}$

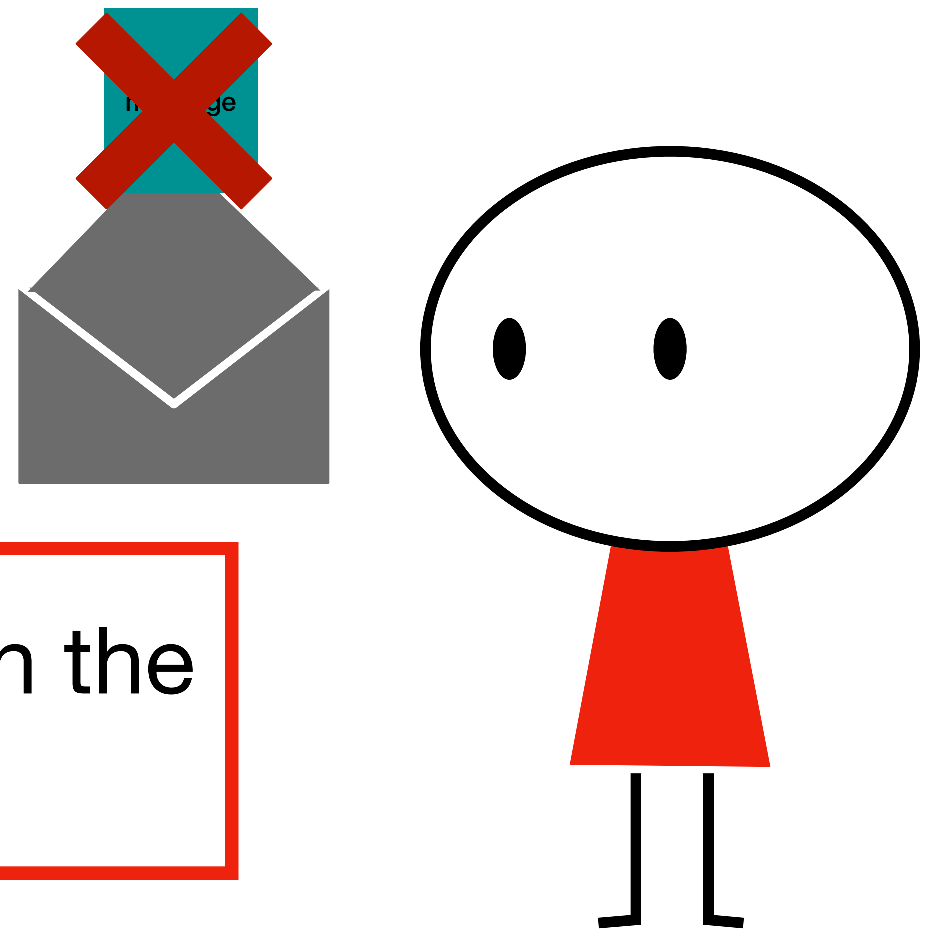
$: \left| \mathbf{s}^T \mathbf{w} \right| \geq \frac{q}{4}$

$: \mathbf{s}^T \mathbf{w} \geq \frac{q}{4}$

$: \mathbf{s} \approx k \cdot \mathbf{w} \qquad (\mathbf{s} = k \cdot \mathbf{w} + \epsilon)$







Each decryption failure provides an equation on the direction of the secret

Decryption failure attacks

➡ Failure probability in IND-CCA setting

0	NTRU, NTRU Prime
2^{-216}	NewHope
2^{-206}	Three Bears
2^{-199}	FrodoKEM
2^{-164}	Kyber
2^{-142}	LAC
2^{-136}	Saber
2^{-117}	Round5

Extremely unlikely in IND-CCA setting (protected by FO transform)

Decryption failure attacks

This transform consists in recovering the encryption's random coins inside the decryption and checking honest generation by re-encryption.

➡ Failure probability in IND-CCA setting

0	NTRU, NTRU Prime
2^{-216}	NewHope
2^{-206}	Three Bears
2^{-199}	FrodoKEM
2^{-164}	Kyber
2^{-142}	LAC
2^{-136}	Saber
2^{-117}	Round5

Extremely unlikely in IND-CCA setting (protected by FO transform)

Decryption failure attacks

This transform consists in recovering the encryption's random coins inside the decryption and checking honest generation by re-encryption.

➡ Failure probability in IND-CCA setting

0	NTRU, NTRU Prime
2^{-216}	NewHope
2^{-206}	Three Bears
2^{-199}	FrodoKEM
2^{-164}	Kyber
2^{-142}	LAC
2^{-136}	Saber
2^{-117}	Round5

Extremely unlikely in IND-CCA setting (protected by FO transform)

The Fujisaki Okamoto transform can be bypassed :

- with timing measurement e.g ‣ J.-P. D'Anvers, M. Tiepelt, F. Vercauteren, I. Verbauwhede. TIS'2019
- with power analysis e.g. ‣ R. Ueno, K. Xagawa, Y. Tanaka, A. Ito, J. Takahashi, N. Homma. TCHES'2022

Decryption failure attacks

This transform consists in recovering the encryption's random coins inside the decryption and checking honest generation by re-encryption.

➡ Failure probability in IND-CCA setting

0	NTRU, NTRU Prime
2^{-216}	NewHope
2^{-206}	Three Bears
2^{-199}	FrodoKEM
2^{-164}	Kyber
2^{-142}	LAC
2^{-136}	Saber
2^{-117}	Round5

Extremely unlikely in IND-CCA setting (protected by FO transform)

The Fujisaki Okamoto transform can be bypassed :

- with timing measurement e.g ‣ J.-P. D'Anvers, M. Tiepelt, F. Vercauteren, I. Verbauwhede. TIS'2019
- with power analysis e.g. ‣ R. Ueno, K. Xagawa, Y. Tanaka, A. Ito, J. Takahashi, N. Homma. TCHES'2022

➡ Open the door to crafting ciphertexts in order to create failures with high probability.

Likely in an IND-CPA setting (where the FO transform is bypassed)

Decryption failure attacks

➡ Failure probability in IND-CCA setting

0	NTRU, NTRU Prime
2^{-216}	NewHope
2^{-206}	Three Bears
2^{-199}	FrodoKEM
2^{-164}	Kyber
2^{-142}	LAC
2^{-136}	Saber
2^{-117}	Round5

This transform consists in recovering the encryption's random coins inside the decryption and checking honest generation by re-encryption.

Extremely unlikely in IND-CCA setting (protected by FO transform)

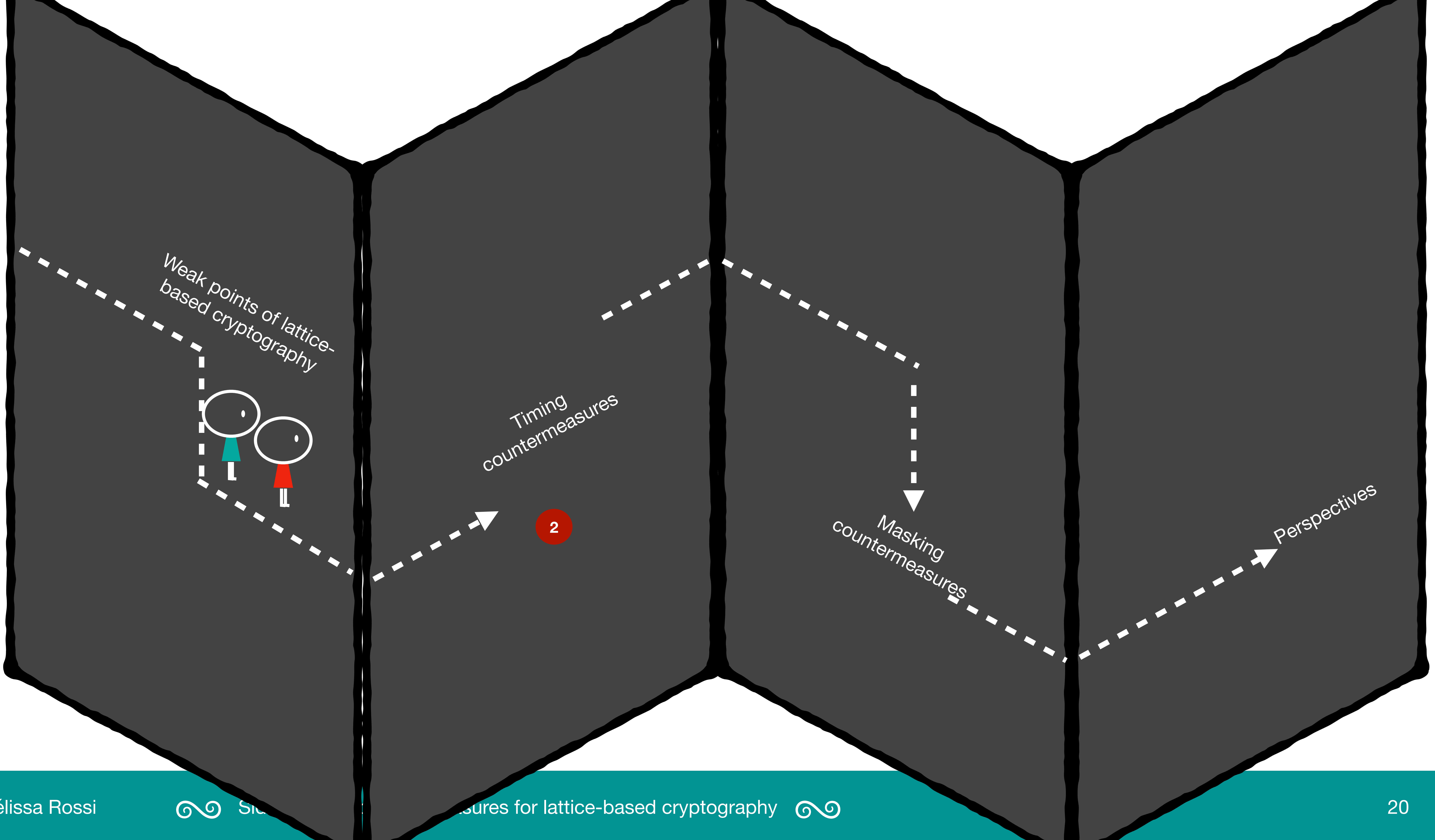
The Fujisaki Okamoto transform can be bypassed :

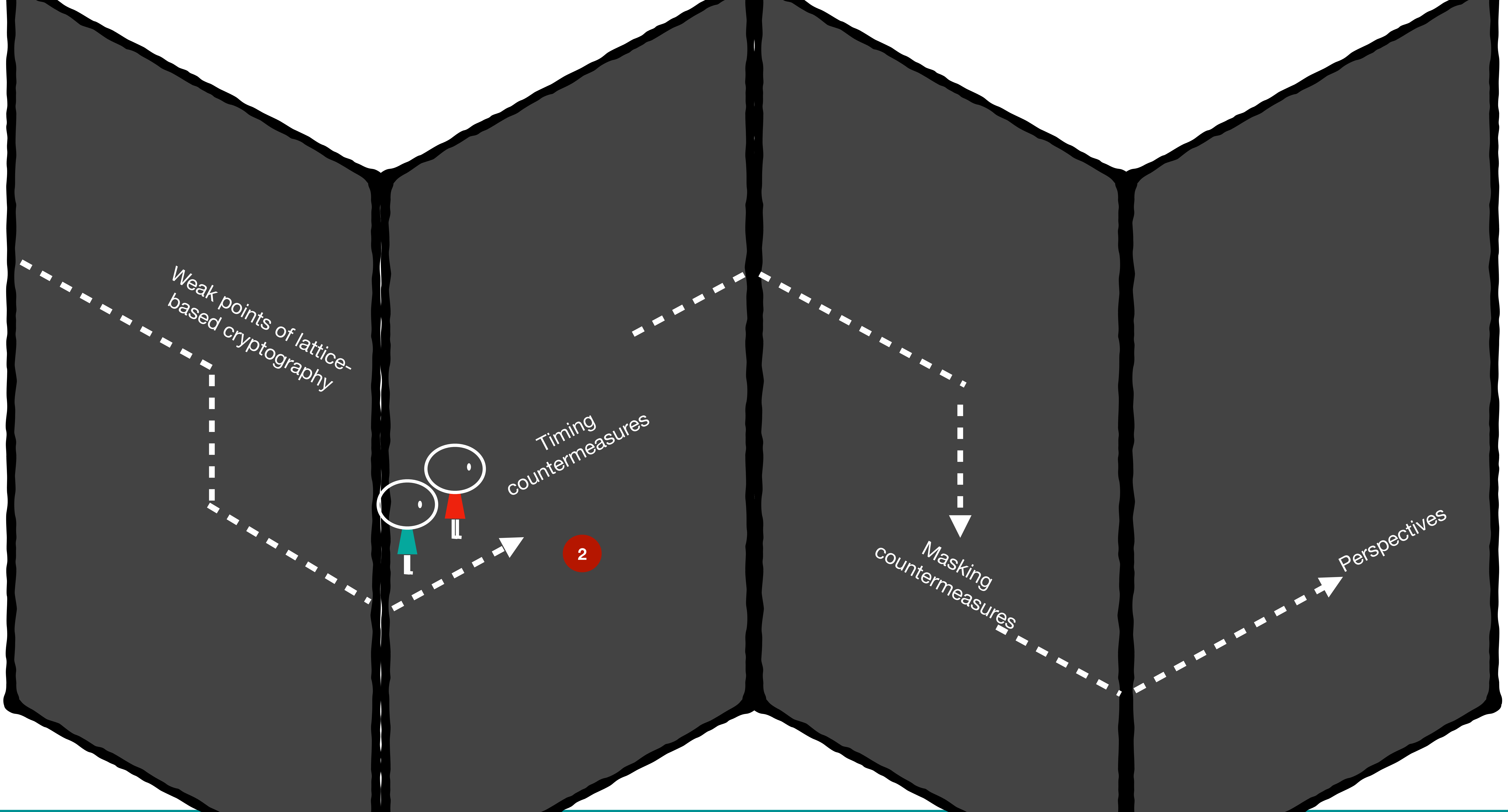
- with timing measurement e.g ‣ J.-P. D'Anvers, M. Tiepelt, F. Vercauteren, I. Verbauwhede. TIS'2019
- with power analysis e.g. ‣ R. Ueno, K. Xagawa, Y. Tanaka, A. Ito, J. Takahashi, N. Homma. TCHES'2022

➡ Open the door to crafting ciphertexts in order to create failures with high probability.

Likely in an IND-CPA setting (where the FO transform is bypassed)

Countermeasure are very important to avoid these side-channel assisted decryption failure attacks


Weak points of lattice-based cryptography
Timing countermeasures
2
Masking countermeasures
Perspectives


Weak points of lattice-based cryptography
Timing countermeasures
2
Masking countermeasures
Perspectives

How to remove timing attacks entry points

The entry points include:

- ✦ computer-science unfriendly distributions like Gaussians.
- ✦ secret-dependent internal distributions.
- ✦ numerous operations with the secret.
- ✦ nonzero failure probability.

How to remove timing attacks entry points

The entry points include:

- ✦ computer-science unfriendly distributions like Gaussians.
- ✦ secret-dependent internal distributions.
- ✦ numerous operations with the secret.
- ✦ nonzero failure probability.

Isochrony the execution time can vary but its distribution should be independent from any sensitive data.

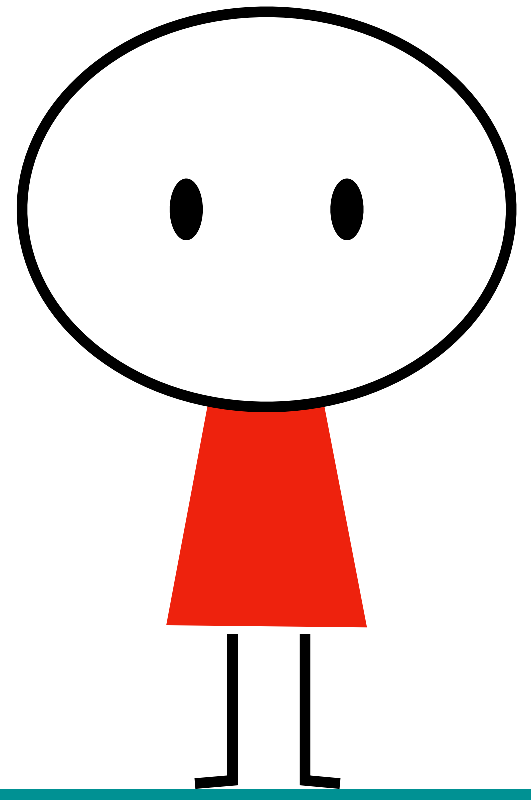
How to remove timing attacks entry points

The entry points include:

- ✦ computer-science unfriendly distributions like Gaussians.
- ✦ secret-dependent internal distributions.
- ✦ numerous operations with the secret.
- ✦ nonzero failure probability.

Isochrony the execution time can vary but its distribution should be independent from any sensitive data.

We want proofs of isochrony!



How to remove timing attacks entry points

The entry points include:

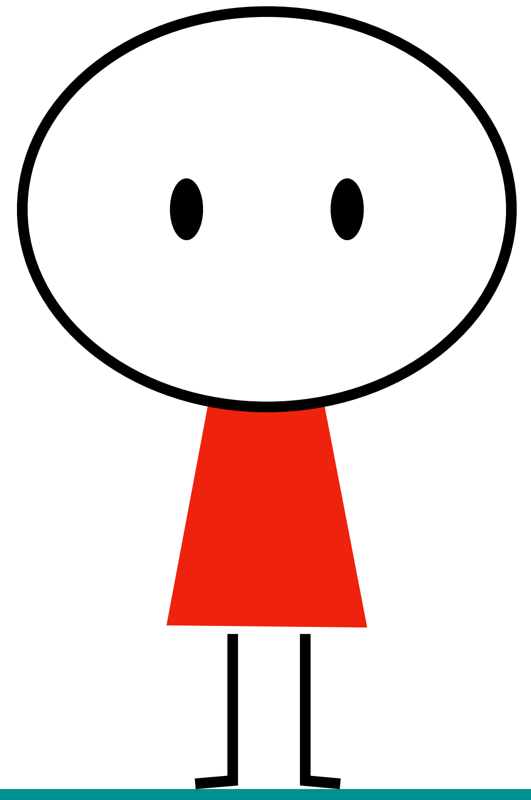
- ✦ computer-science unfriendly distributions like Gaussians.
- ✦ secret-dependent internal distributions.
- ✦ numerous operations with the secret.
- ✦ nonzero failure probability.

Isochrony the execution time can vary but its distribution should be independent from any sensitive data.

Here are some provable countermeasure techniques:

1. Renyi divergence arguments
2. Polynomial approximations

We want proofs of isochrony!



1) Rényi divergence arguments

- S. Bai, A. Langlois, T. Lepoint, D. Stehlé, R. Steinfeld ASIACRYPT'15
- T. Prest ASIACRYPT'17

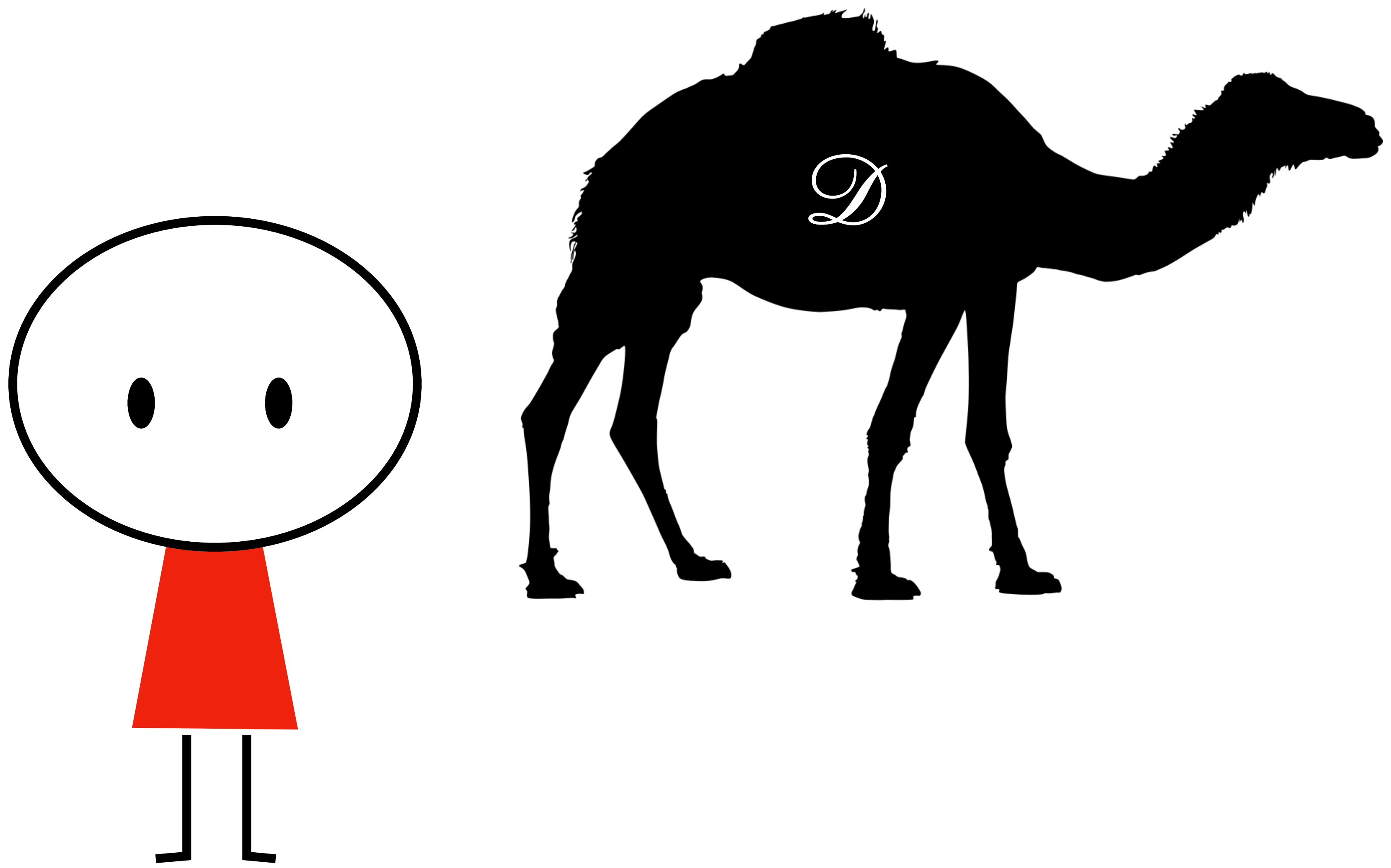
Distributions may be approximated/simplified because of the limited number of queries

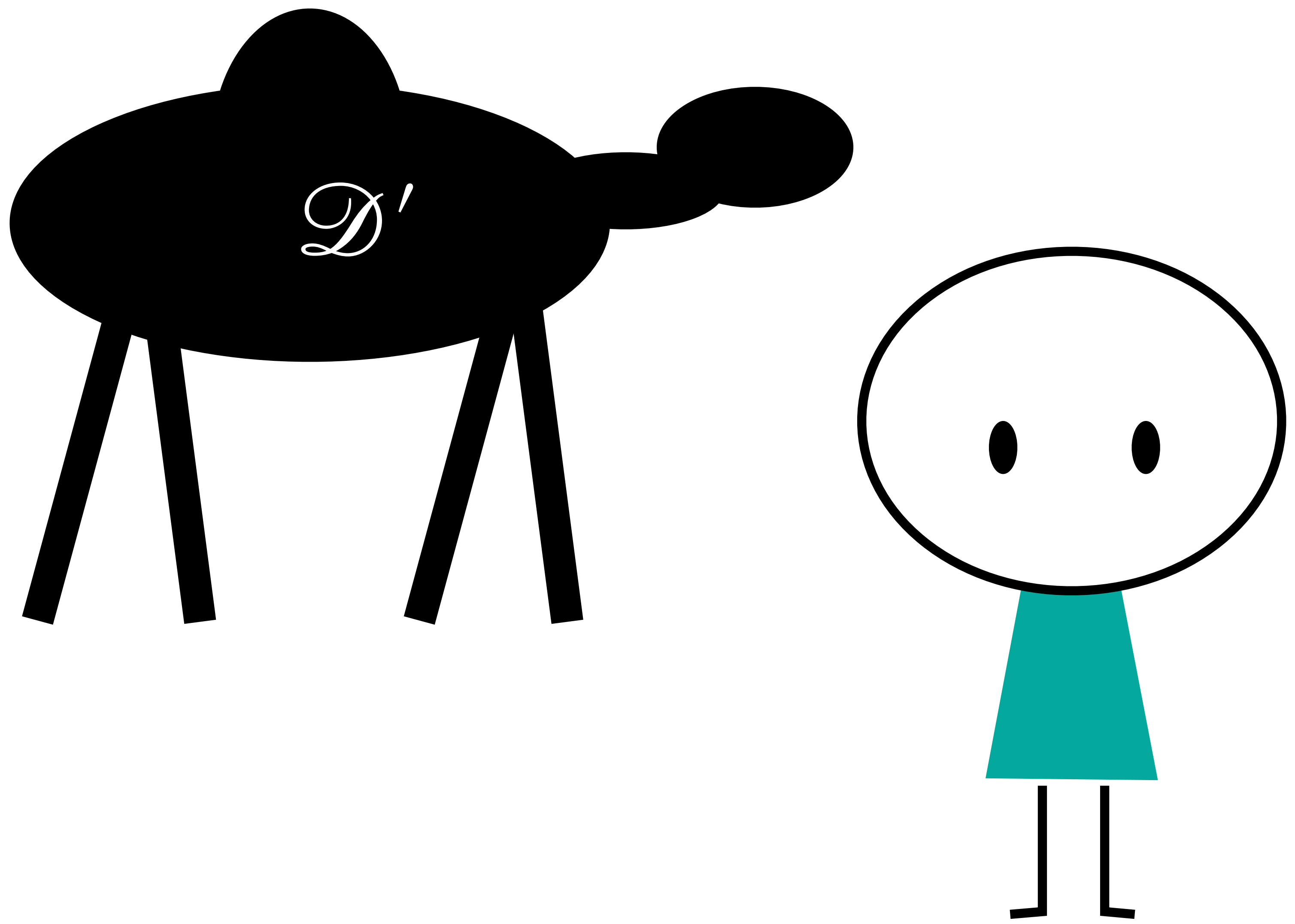
Take two cryptographic schemes
- One with distribution $\mathscr{D}$
- One with an approximate distribution $\mathscr{D}'$ with the same support

Suppose that :

1. $\mathscr{D}$ and $\mathscr{D}'$ are close enough : $\left\| 1 - \frac{\mathscr{D}'}{\mathscr{D}} \right\|_\infty \leq 2^{-K}$
2. the number of sample queries is bounded

Then, the **bit security will remain almost the same.**





1) Rényi divergence arguments

- S. Bai, A. Langlois, T. Lepoint, D. Stehlé, R. Steinfeld ASIACRYPT'15
- T. Prest ASIACRYPT'17

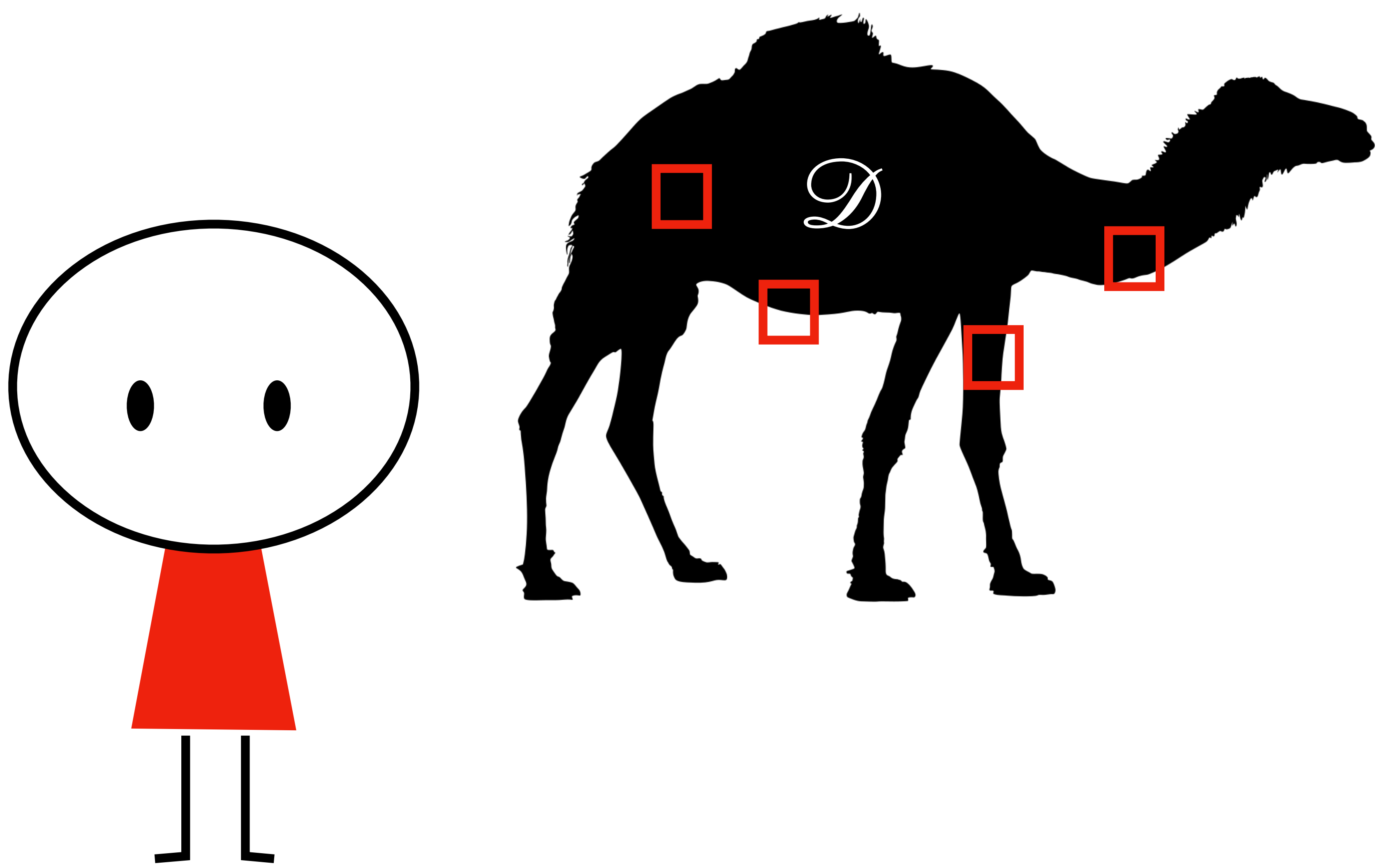
Distributions may be approximated/simplified because of the limited number of queries

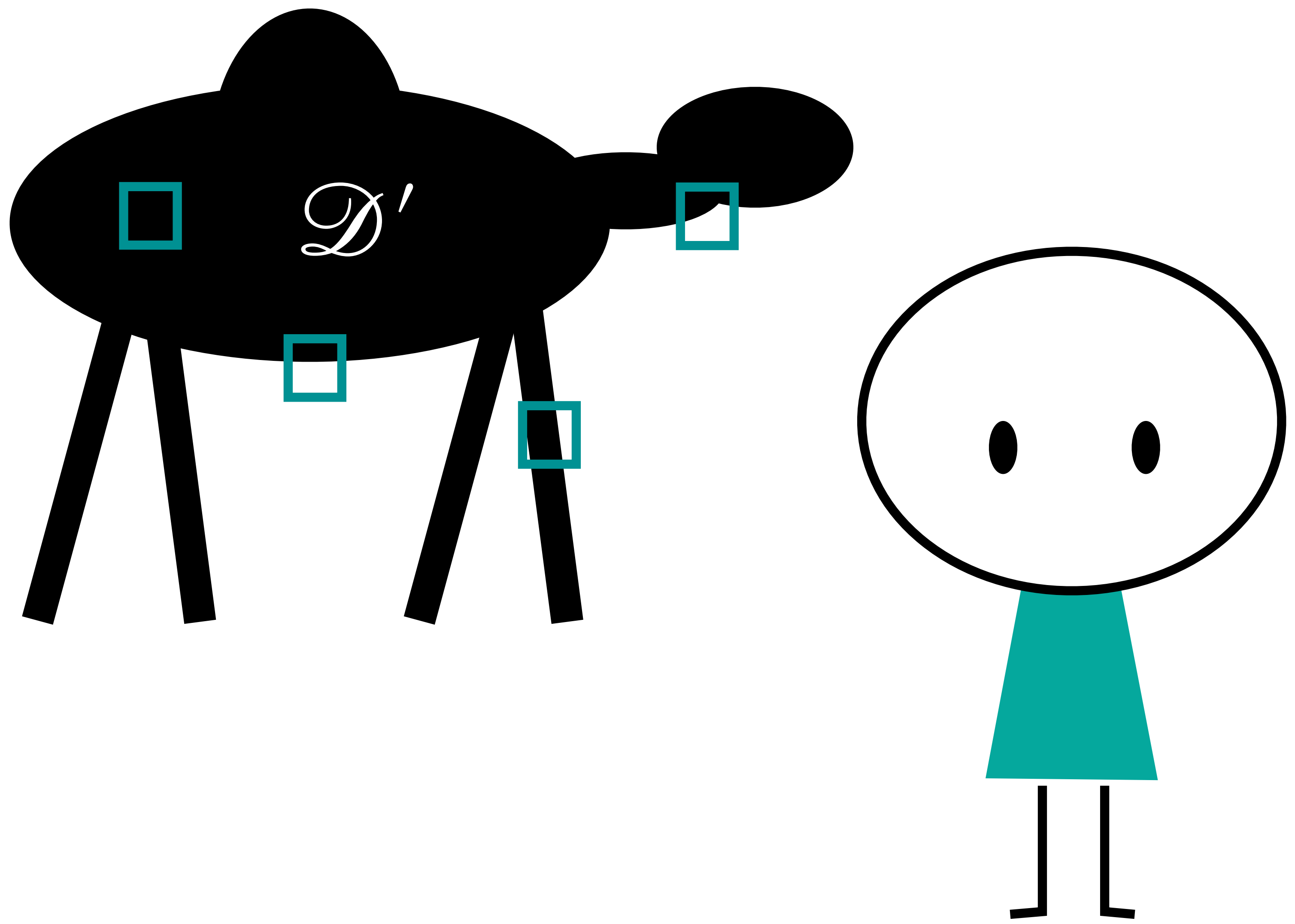
Take two cryptographic schemes
- One with distribution $\mathscr{D}$
- One with an approximate distribution $\mathscr{D}'$ with the same support

Suppose that :

1. $\mathscr{D}$ and $\mathscr{D}'$ are close enough : $\left\| 1 - \frac{\mathscr{D}'}{\mathscr{D}} \right\|_\infty \leq 2^{-K}$
2. the number of sample queries is bounded

Then, the **bit security will remain almost the same.**





1) Rényi divergence arguments

- S. Bai, A. Langlois, T. Lepoint, D. Stehlé, R. Steinfeld ASIACRYPT'15
- T. Prest ASIACRYPT'17

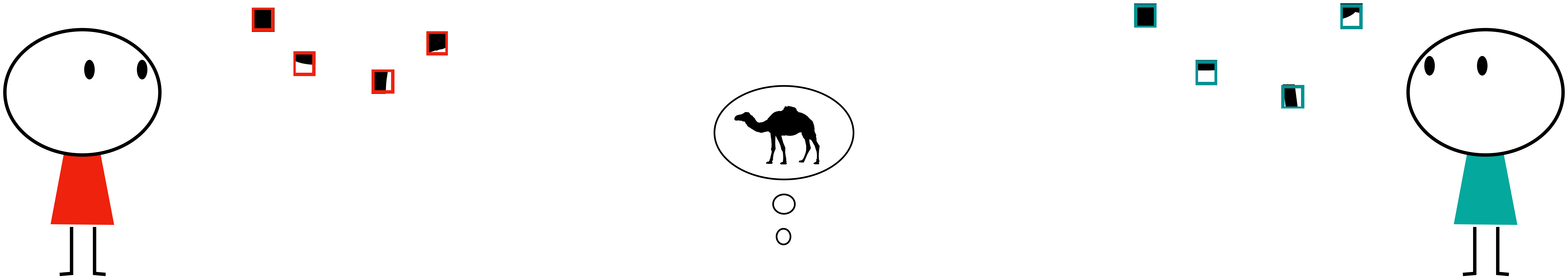
Distributions may be approximated/simplified because of the limited number of queries

Take two cryptographic schemes
- One with distribution $\mathscr{D}$
- One with an approximate distribution $\mathscr{D}'$ with the same support

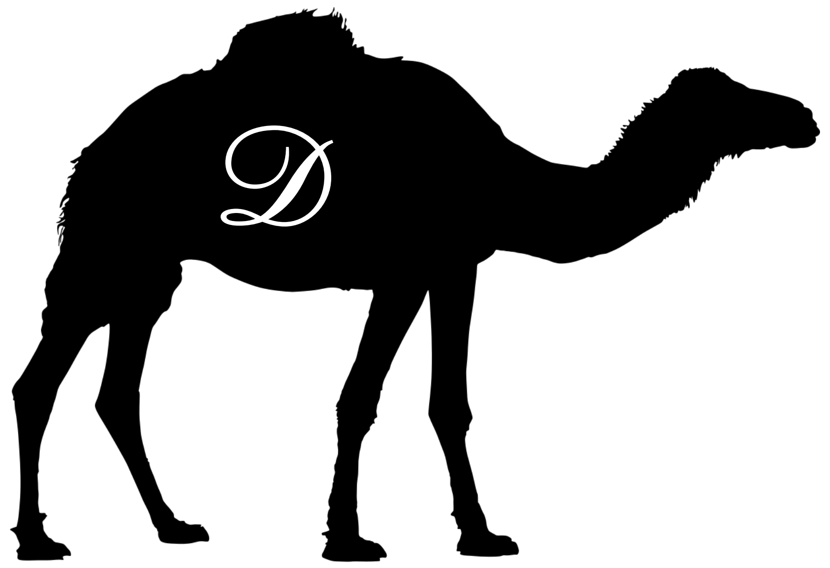
Suppose that :

1. $\mathscr{D}$ and $\mathscr{D}'$ are close enough : $\left\| 1 - \frac{\mathscr{D}'}{\mathscr{D}} \right\|_\infty \leq 2^{-K}$
2. the number of sample queries is bounded

Then, the **bit security will remain almost the same.**



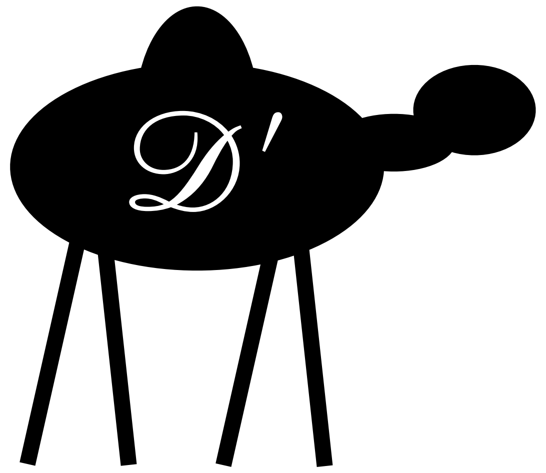
2) Polynomial approximation for Gaussians



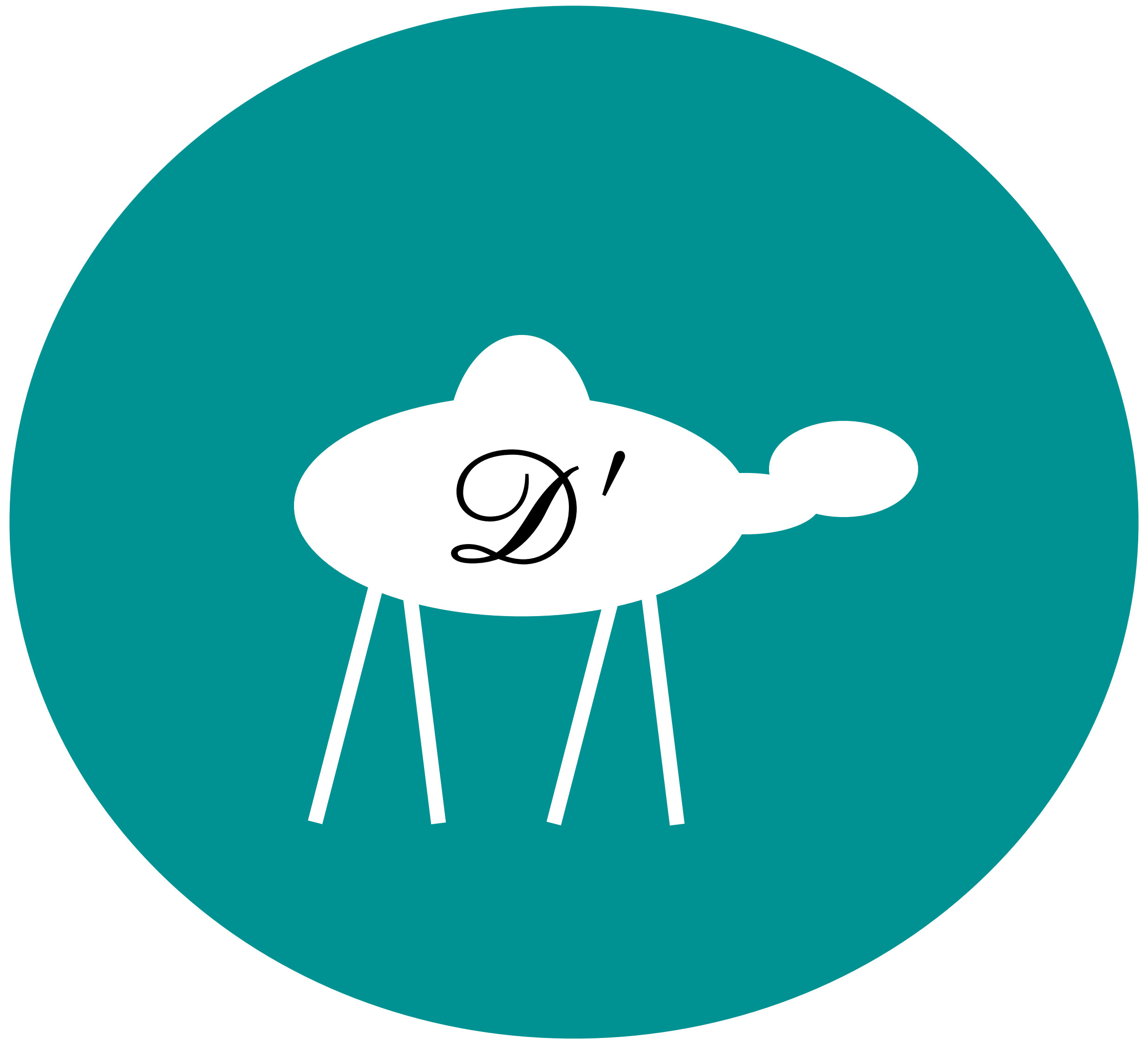
Transcendental
function

Polynomial approximation

Degree d polynomial in $\mathbb{Z}[x]$
with small coefficients



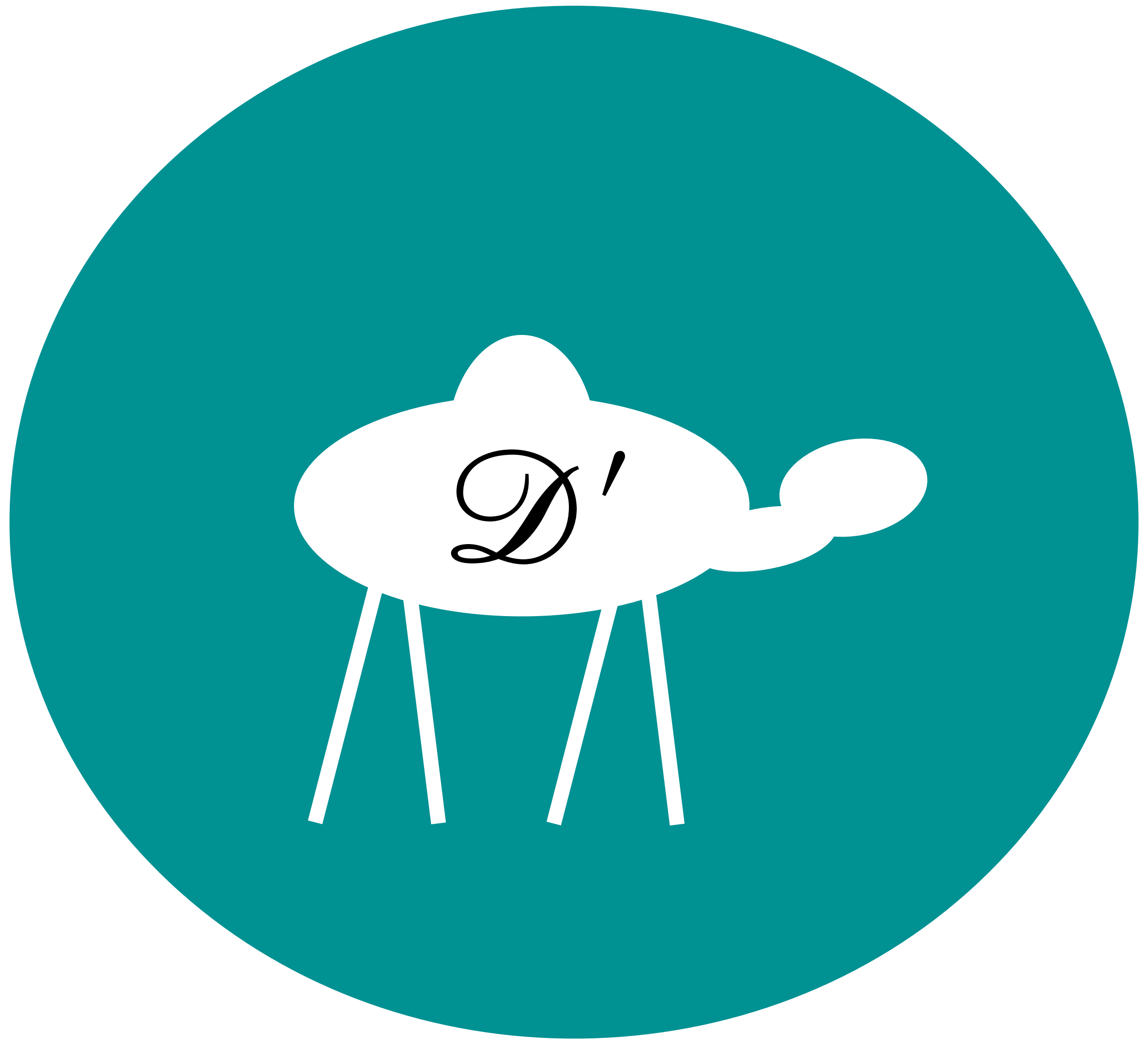
2) Polynomial approximation for Gaussians



2) Polynomial approximation for Gaussians

- **Taylor expansion** $\mathscr{D}'(x) = \mathscr{D}(0) + \mathscr{D}^{(1)}(0) \cdot x + \cdots + \dfrac{\mathscr{D}^{(d)}(0)}{d!} \cdot x^d$

2) Polynomial approximation for Gaussians



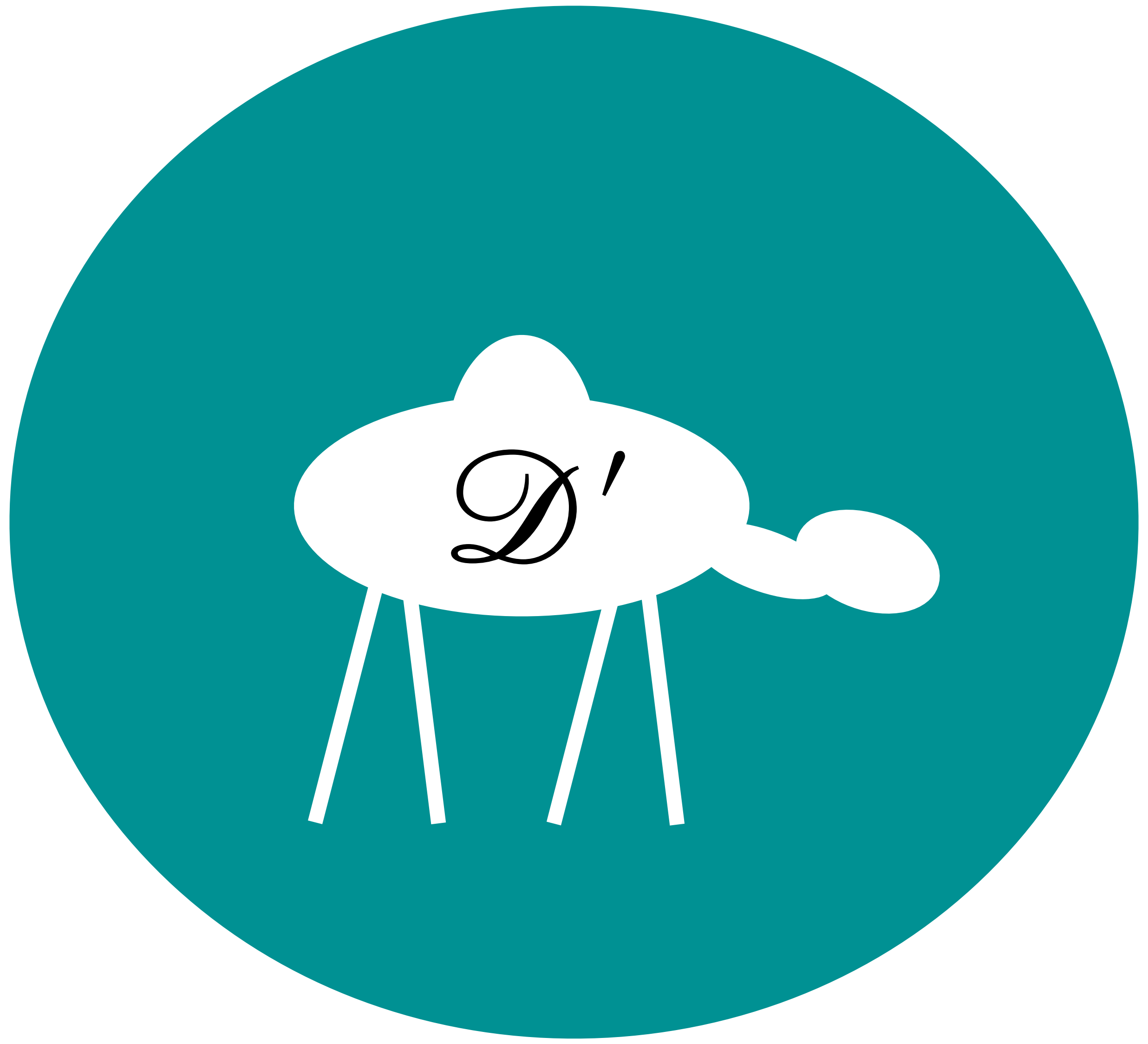
- **Taylor expansion** $\mathscr{D}'(x) = \mathscr{D}(0) + \mathscr{D}^{(1)}(0) \cdot x + \cdots + \frac{\mathscr{D}^{(d)}(0)}{d!} \cdot x^d$

- **Padé approximants** (rational function approximation)

  - T. Prest ASIACRYPT'17

  Two polynomials, higher degrees $\mathscr{D}'(x) = \frac{P(x)}{Q(x)}$

2) Polynomial approximation for Gaussians



- **Taylor expansion** $\mathscr{D}'(x) = \mathscr{D}(0) + \mathscr{D}^{(1)}(0) \cdot x + \cdots + \frac{\mathscr{D}^{(d)}(0)}{d!} \cdot x^d$

- **Padé approximants** (rational function approximation)

    - T. Prest ASIACRYPT'17

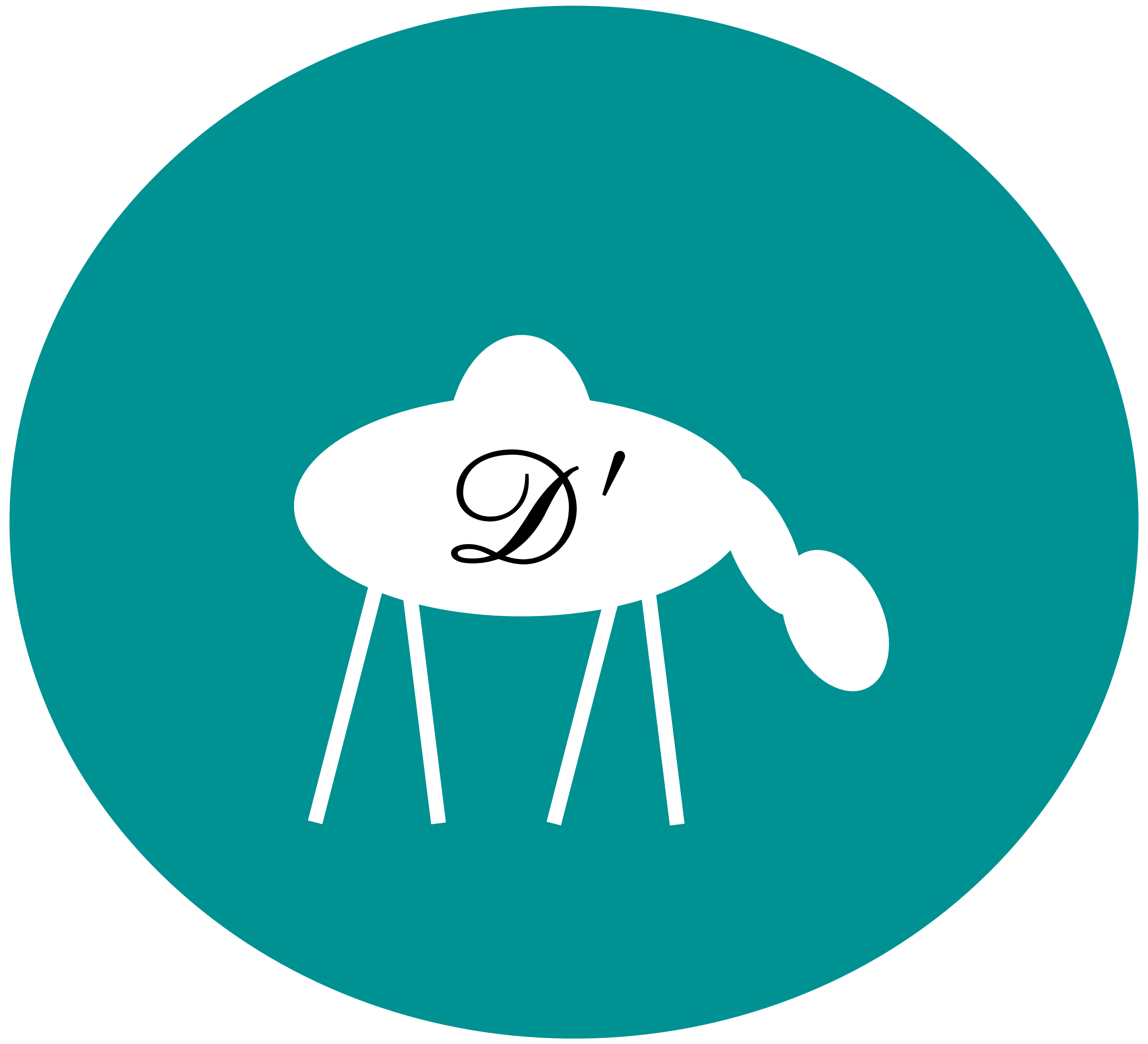
  Two polynomials, higher degrees $\mathscr{D}'(x) = \frac{P(x)}{Q(x)}$

- **Minimax computations** : Sollya software package

    - N. Brisebarre and S. Chevillard IEEE'07
    - S. Chevillard, M. Joldes and C. Q. Lauter ICMS'10
    - R. Zhao, R. Steinfeld and A. Sakzad IEEE'19

  Floating point arithmetics $$\mathscr{D}' = \arg \min_{\deg(P) \leq d} \left(\sup_{x \in I} \left(1 - \frac{P(x)}{\mathscr{D}(x)} \right) \right)$$

2) Polynomial approximation for Gaussians



- **Taylor expansion** $\mathscr{D}'(x) = \mathscr{D}(0) + \mathscr{D}^{(1)}(0) \cdot x + \cdots + \frac{\mathscr{D}^{(d)}(0)}{d!} \cdot x^d$

- **Padé approximants** (rational function approximation)
  - T. Prest ASIACRYPT'17

  Two polynomials, higher degrees $\mathscr{D}'(x) = \frac{P(x)}{Q(x)}$

- **Minimax computations** : Sollya software package
  - N. Brisebarre and S. Chevillard IEEE'07
  - S. Chevillard, M. Joldes and C. Q. Lauter ICMS'10
  - R. Zhao, R. Steinfeld and A. Sakzad IEEE'19

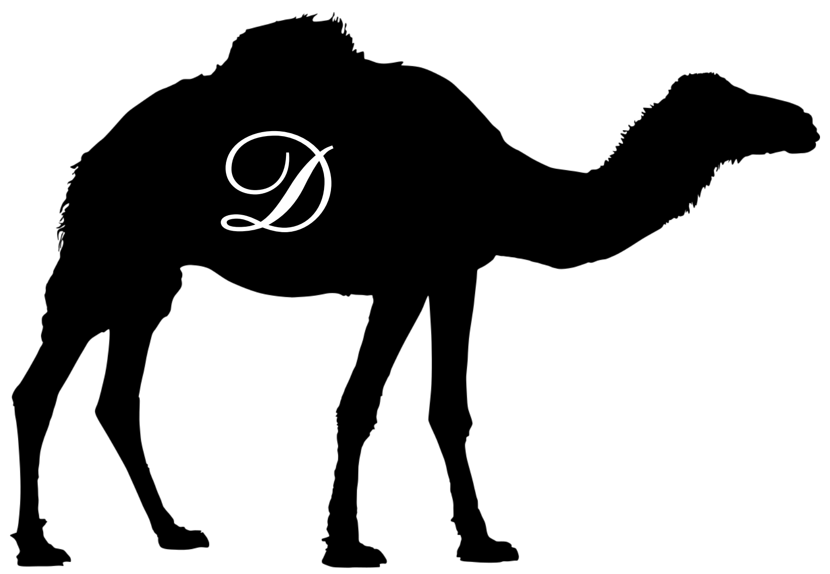
  Floating point arithmetics $$\mathscr{D}' = \arg \min_{\deg(P) \le d} \left(\sup_{x \in I} \left(1 - \frac{P(x)}{\mathscr{D}(x)} \right) \right)$$

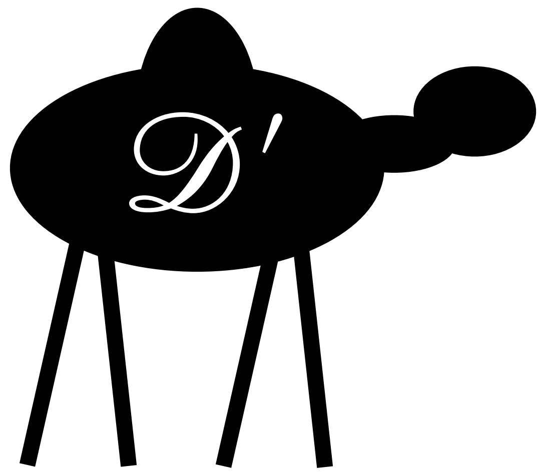
- **Projections with respect to the Sobolev Norm**
  - *GALACTICS [...]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

$$\|f\|_\infty \le \sqrt{2} \cdot \|f\|_S$$

- *GALACTICS [...]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

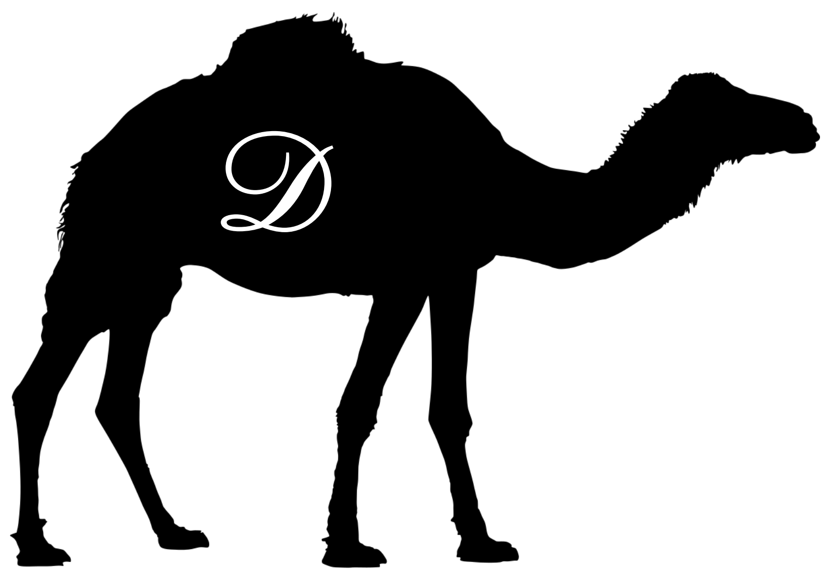
Polynomial approximation





- *GALACTICS […]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

Polynomial approximation



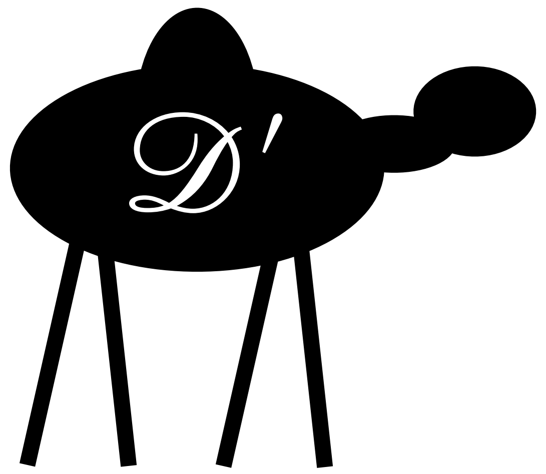
Transcendental function

$\downarrow$

Degree d polynomial in $\mathbb{R}[x]$

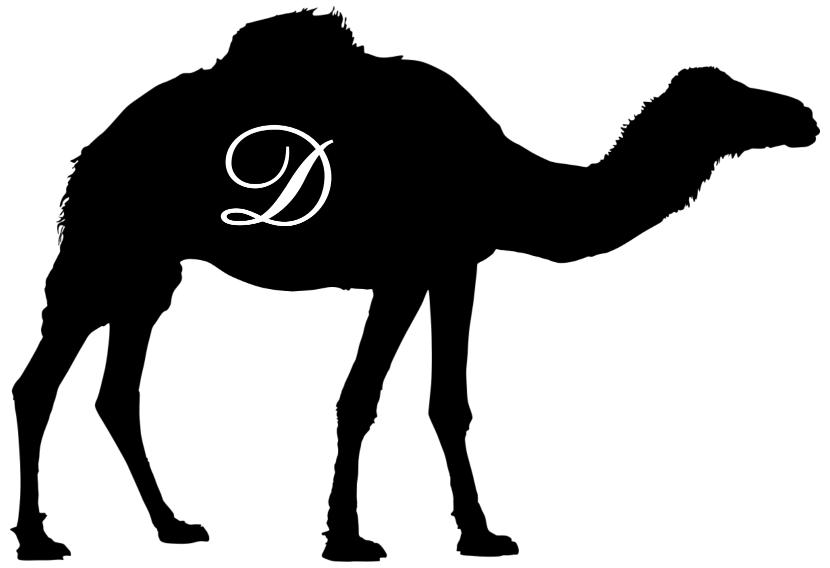
$\downarrow$

Degree d polynomial in $\mathbb{Z}[x]$ with η-bit coefficients



▸ *GALACTICS [...]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

Polynomial approximation



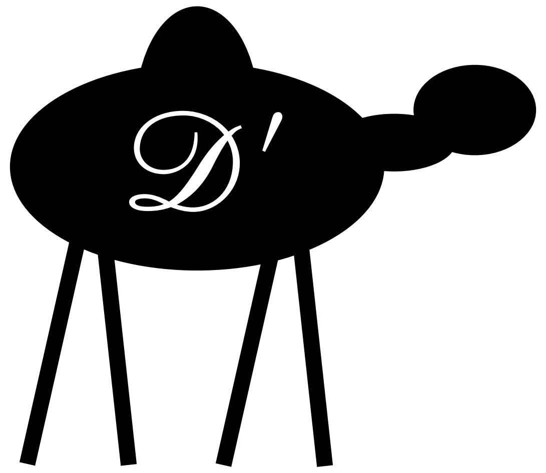
Transcendental function

↓

Degree d polynomial in $\mathbb{R}[x]$

↓

Degree d polynomial in $\mathbb{Z}[x]$ with η-bit coefficients

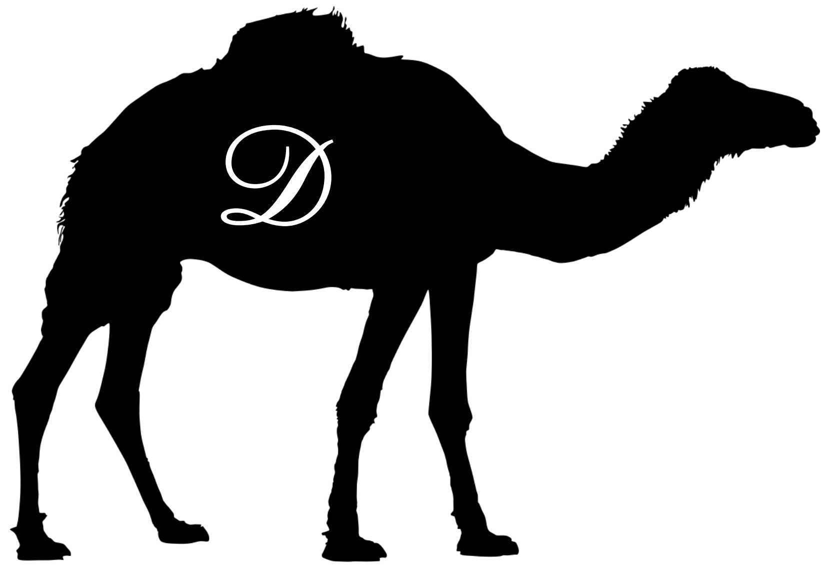


Successive orthogonal projections in a polynomial space wrt the Sobolev norm.

$$\|f\|_\infty \leq \sqrt{2} \cdot \|f\|_S$$

▸ *GALACTICS [...]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

Polynomial approximation



Transcendental function

↓

Successive orthogonal projections in a polynomial space wrt the Sobolev norm.

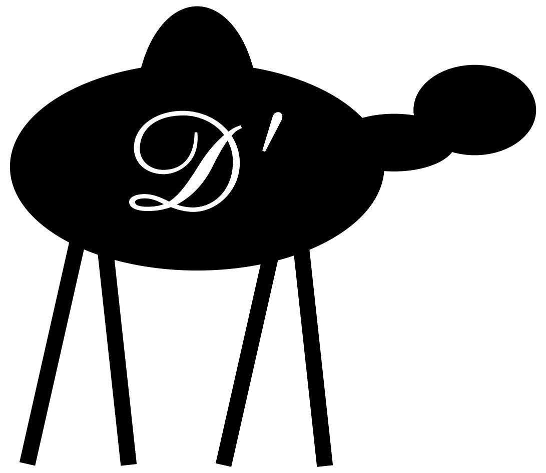
$$\|f\|_\infty \leq \sqrt{2} \cdot \|f\|_S$$

Degree d polynomial in $\mathbb{R}[x]$

↓

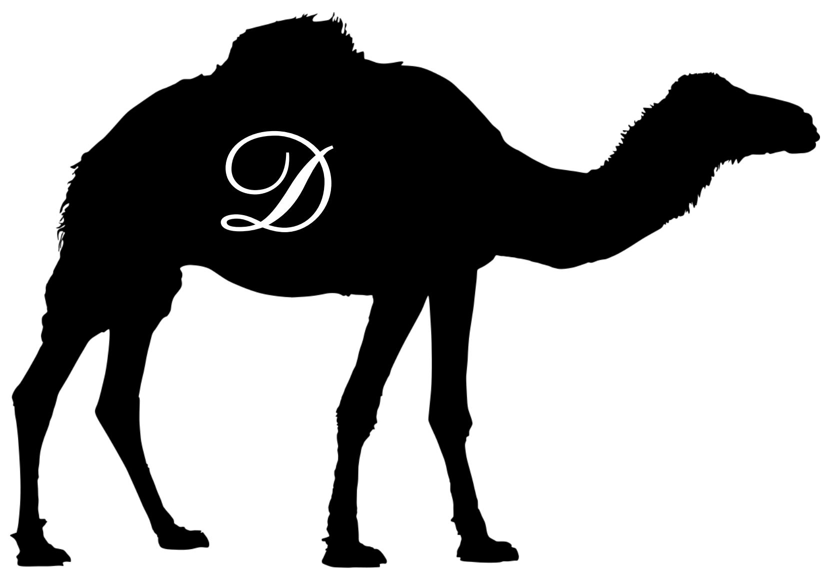
LLL reduction
Babai rounding

Degree d polynomial in $\mathbb{Z}[x]$ with η-bit coefficients



- *GALACTICS [...]* ACM-CCS'2019. G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi.

Polynomial approximation



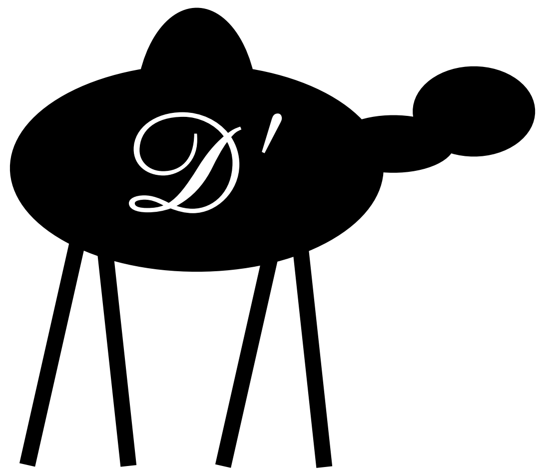
Transcendental function

↓

Degree d polynomial in $\mathbb{R}[x]$

↓

Degree d polynomial in $\mathbb{Z}[x]$ with η-bit coefficients



Successive orthogonal projections in a polynomial space wrt the Sobolev norm.

$$\|f\|_\infty \leq \sqrt{2} \cdot \|f\|_S$$

LLL reduction
Babai rounding

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{Ay}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** ($\mathbf{z}$, c)

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** ($\mathbf{z}$, c)

Constant time

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\quad \mathbf{y} \xleftarrow{\$} Y$

3: $\quad c \leftarrow H(\mathbf{Ay}, m)$

4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{Ay}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)

6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

Secret dependent timing

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{Ay}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$

6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

Secret dependent timing

For BLISS

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}$$

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$

6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

Secret dependent timing

For BLISS

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{S}\mathbf{c}||^2}{2\sigma^2}\right)}$$

Rényi divergence proof

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M} \cdot P_{\cosh^{-1}}\left(\frac{\langle \mathbf{z}, \mathbf{S}\mathbf{c}\rangle}{\sigma^2}\right) \cdot P_{\exp^{-1}}\left(-\frac{||\mathbf{S}\mathbf{c}||^2}{2\sigma^2}\right)$$

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**
2: $\mathbf{y} \xleftarrow{\$} Y$
3: $c \leftarrow H(\mathbf{Ay}, m)$
4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected($\mathbf{z}, \mathbf{c}, \mathbf{S}$)
6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

Secret dependent timing

For BLISS

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}$$

Rényi divergence proof

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M} \cdot P_{\cosh^{-1}}\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot P_{\exp^{-1}}\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)$$

Polynomial evaluation: simple and constant time (multiplications and additions)

An example with Fiat-Shamir with aborts

Signature algorithm:

1: **do**
2: $\mathbf{y} \xleftarrow{\$} Y$
3: $c \leftarrow H(\mathbf{Ay}, m)$
4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)
6: **return** $(\mathbf{z}, c)$

Isochronous

Rejection sampling theorem

Constant time

Secret dependent timing

For BLISS

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M \cdot \cosh\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot \exp\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)}$$

Rényi divergence proof

$$\text{Rejected}(\mathbf{z}, \mathbf{c}, \mathbf{S}) = \frac{1}{M} \cdot P_{\cosh^{-1}}\left(\frac{\langle \mathbf{z}, \mathbf{Sc} \rangle}{\sigma^2}\right) \cdot P_{\exp^{-1}}\left(-\frac{||\mathbf{Sc}||^2}{2\sigma^2}\right)$$

Polynomial evaluation: simple and constant time (multiplications and additions)

➡Would you say that it is more or less efficient?

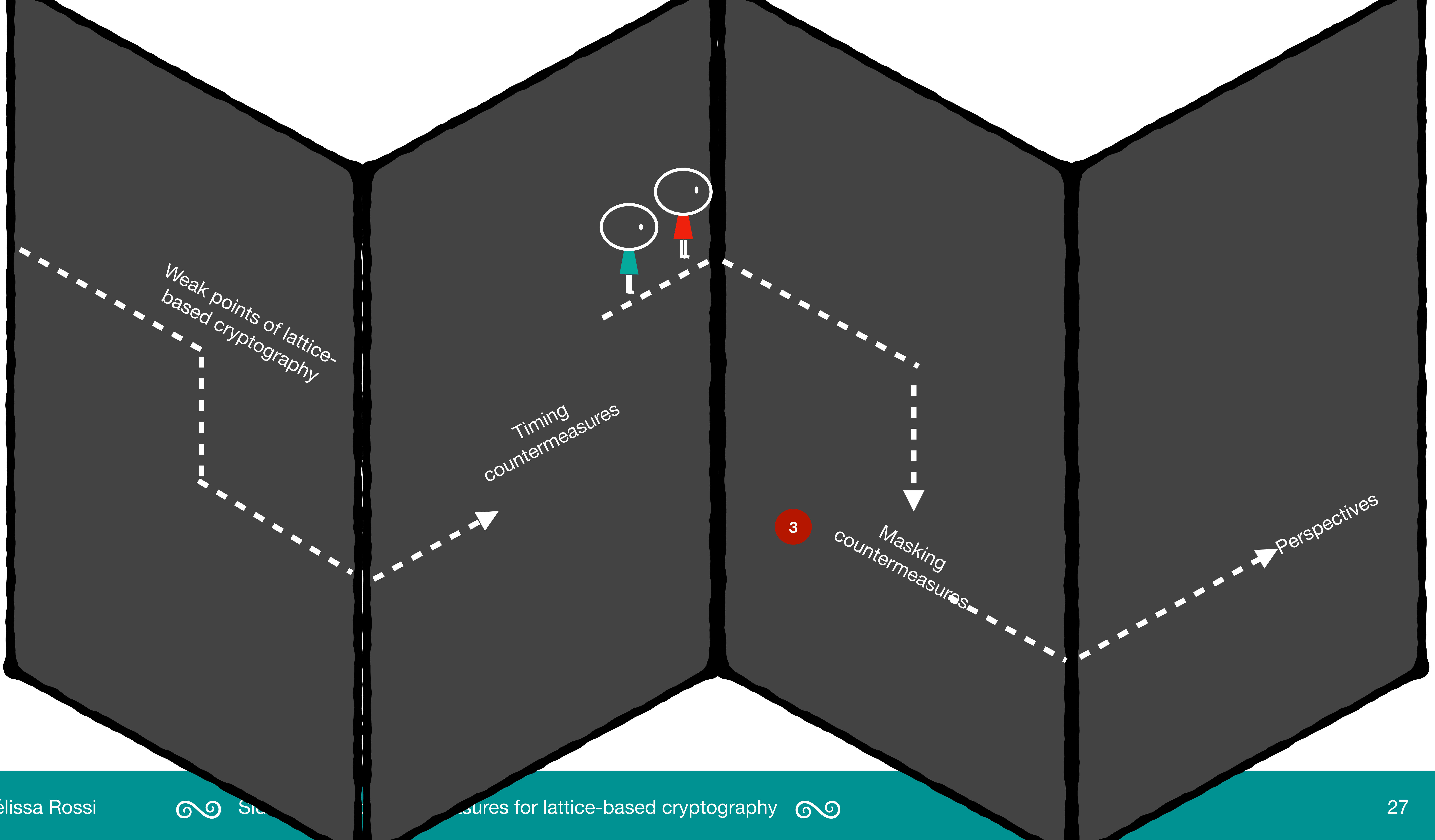
Examples of application for proving isochrony

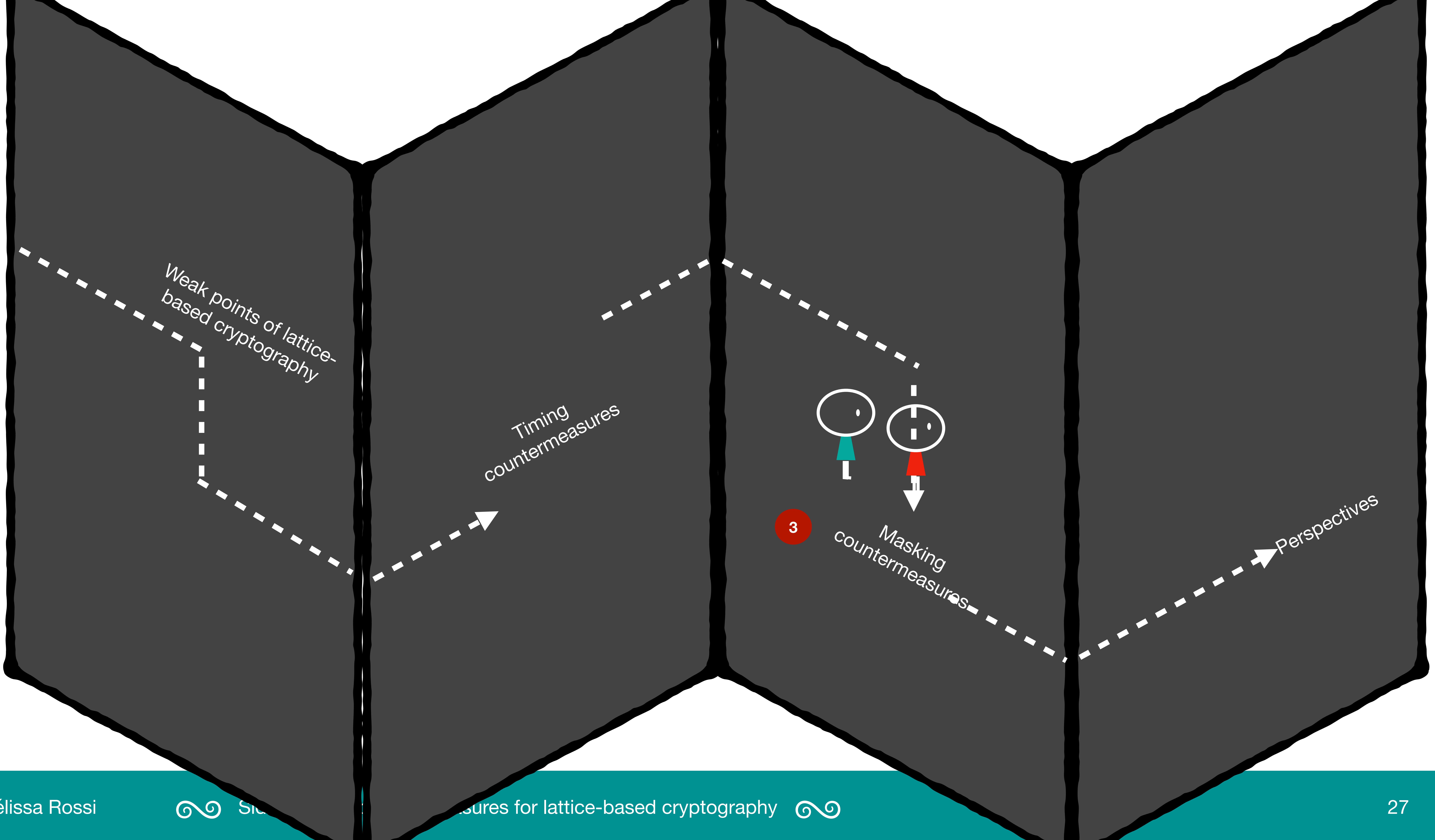
Falcon Performance penalty factor : $+50\,\%$

- J. Howe, T. Prest, T. Ricosset and M. Rossi. PQ-CRYPTO'2020.
- *T. Pornin https://falcon-sign.info/falcon-impl-20190802.pdf*

BLISS Performance penalty factor : $+13\,\%$

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi. ACM-CCS'2019.


Weak points of lattice-based cryptography
Timing countermeasures
3
Masking countermeasures
Perspectives


Weak points of lattice-based cryptography
Timing countermeasures
3
Masking countermeasures
Perspectives

Masking

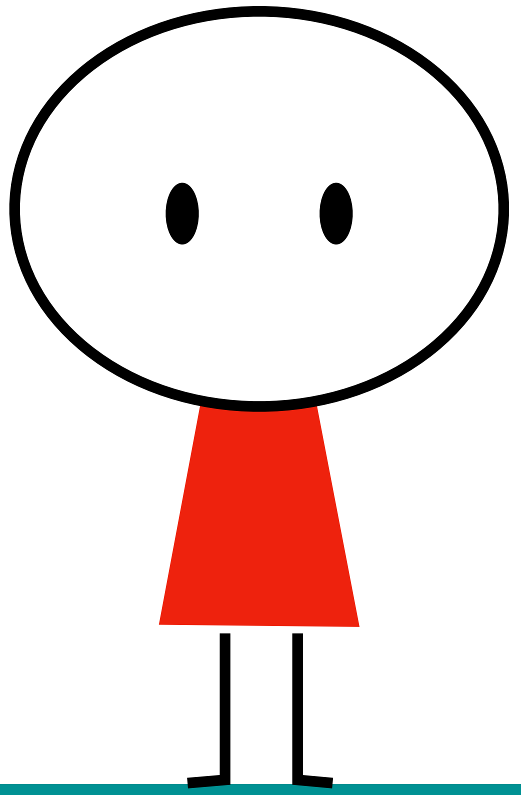
$$x$$

$$x_0 + x_1 + x_2 + x_3 + x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$

Each share looks random.
The only way to recover x is to know all of them.

Masking order : $d = 4.$

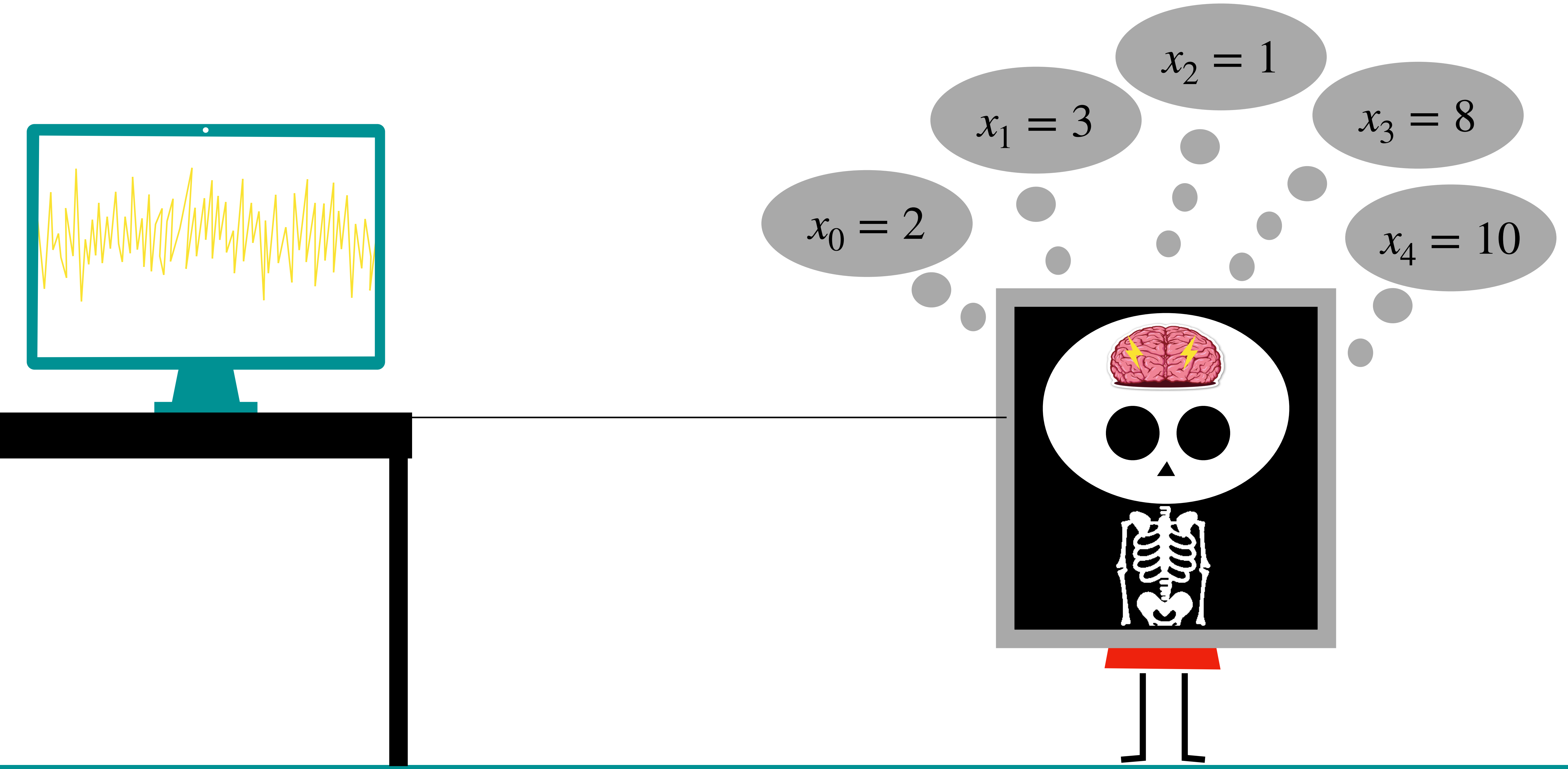


Masking

x

$$x_0 + x_1 + x_2 + x_3 + x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$



Each share looks random.
The only way to recover x is to know all of them.

Masking order : $d = 4$.

The real secret value is
$$x = 2 + 3 + 1 + 8 + 10$$
$$= 24$$

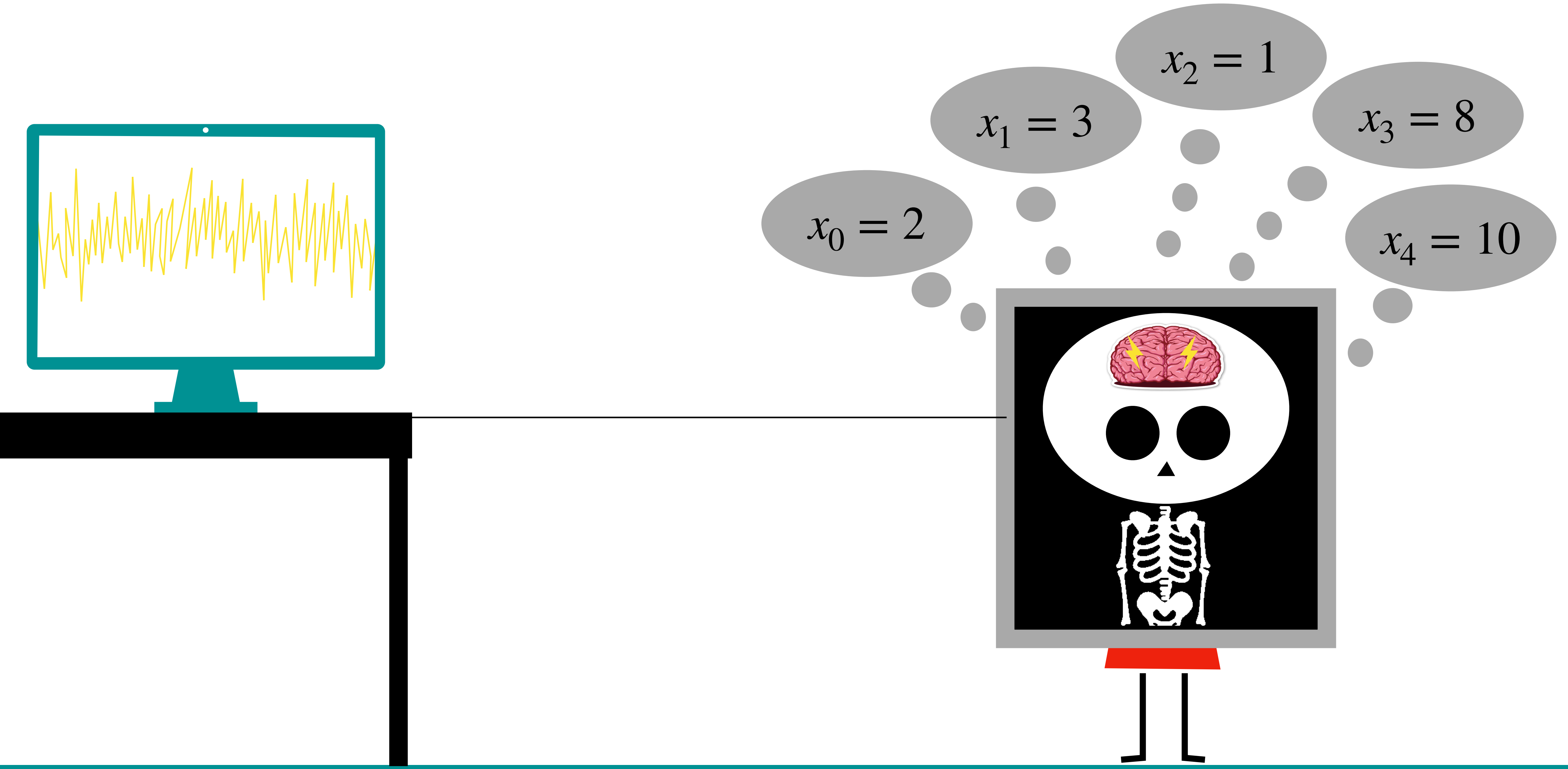
Masking

➡ Increase of the noise: Highly complicates the dependancies between the secret and the measurement

$$x$$

$$x_0 + x_1 + x_2 + x_3 + x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$



Each share looks random.
The only way to recover x is to know all of them.

Masking order : $d = 4$.

The real secret value is
$x = 2 + 3 + 1 + 8 + 10$
$= 24$

Boolean and arithmetic masking

Boolean masking

x

$$x_0 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

Arithmetic masking

x

$$x_0 + x_1 + x_2 + x_3 + x_4 \bmod q$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

Boolean and arithmetic masking

Boolean masking

$$x$$

$$x_0 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

$\mathbb{F}_2$-linear operations:

$$[[x]] \oplus [[y]] = (x_0 \oplus y_0, x_1 \oplus y_1, x_2 \oplus y_2, x_3 \oplus y_3, x_4 \oplus y_4)$$

Arithmetic masking

$$x$$

$$x_0 + x_1 + x_2 + x_3 + x_4 \bmod q$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

$\mathbb{F}_q$-linear operations:

$$[[x]] + [[y]] \bmod q = (x_0 + y_0 \bmod q, \ldots, x_4 + y_4 \bmod q)$$

Boolean and arithmetic masking

Boolean masking

x

$$x_0 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

$\mathbb{F}_2$**-linear operations:**

$$[[x]] \oplus [[y]] = (x_0 \oplus y_0, x_1 \oplus y_1, x_2 \oplus y_2, x_3 \oplus y_3, x_4 \oplus y_4)$$

Arithmetic masking

x

$$x_0 + x_1 + x_2 + x_3 + x_4 \bmod q$$

$$[[x]] = (x_0, x_1, x_2, x_3, x_4)$$
$$[[y]] = (y_0, y_1, y_2, y_3, y_4)$$

$\mathbb{F}_q$**-linear operations:**

$$[[x]] + [[y]] \bmod q = (x_0 + y_0 \bmod q, \ldots, x_4 + y_4 \bmod q)$$

What about non linear operations?

➡Need for extra randomness to mix shares without introducing any biais.

Designs for the multiplication of two shared values
- L. Goubin and J. Patarin CHES'1999
- S. Chari, C. Jutla, J. Rao and P. Rohatgi CRYPTO'1999

More information in J.S. Coron's presentation

How to combine many operations?

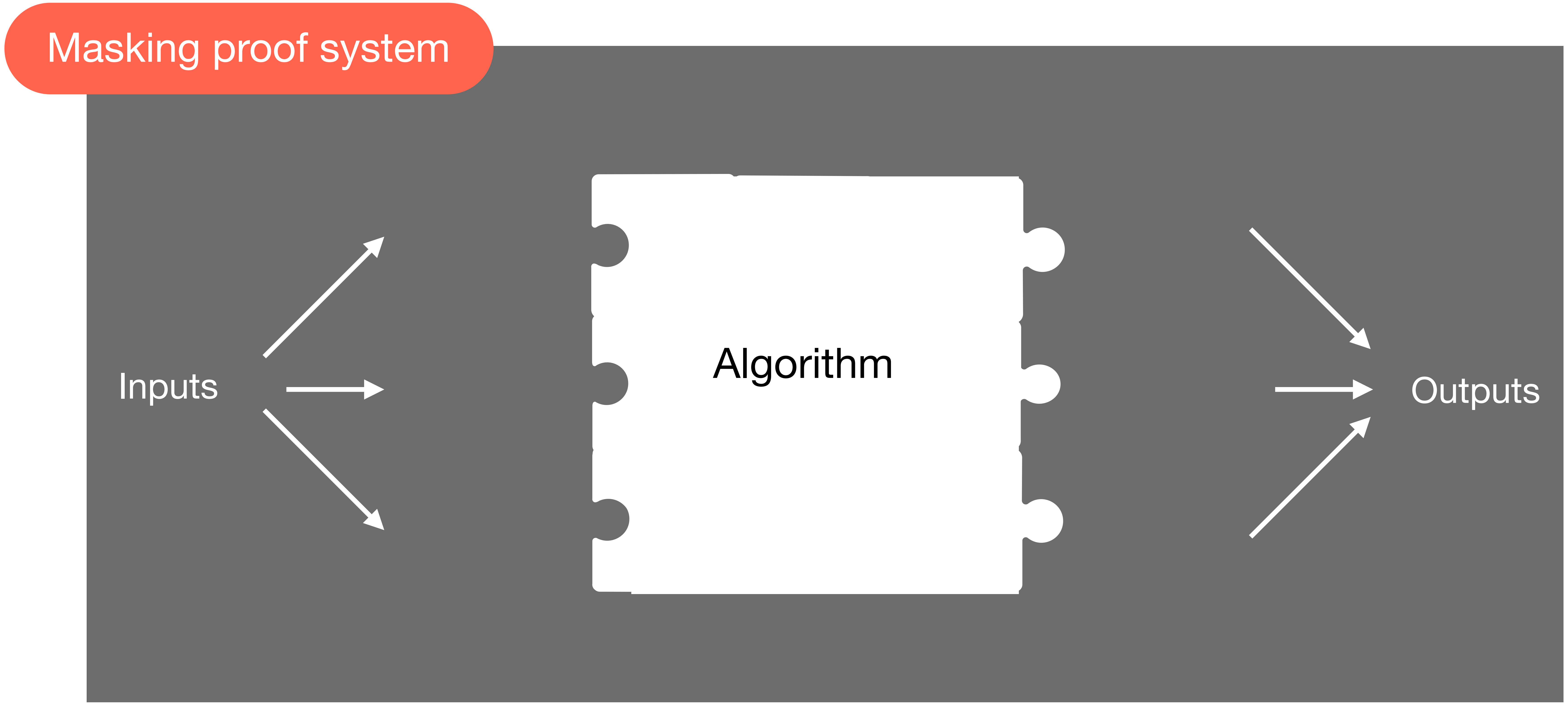
- Y. Ishai, A. Sahai and D. Wagner CRYPTO'2003
- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016

Masking proof system

Algorithm

How to combine many operations?

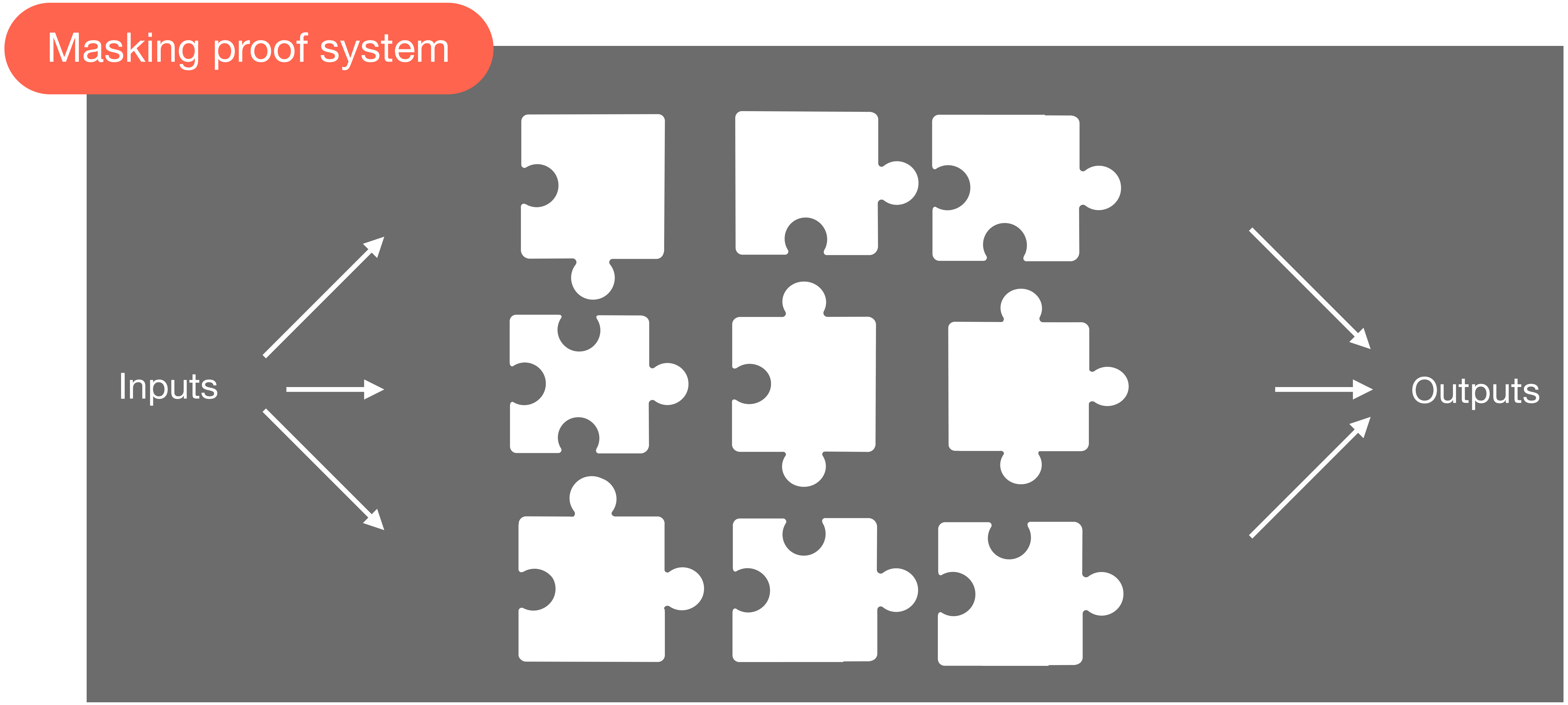
- Y. Ishai, A. Sahai and D. Wagner CRYPTO'2003
- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016



Proofs of masking for each gadget
+
Composition proofs

How to combine many operations?

- Y. Ishai, A. Sahai and D. Wagner CRYPTO'2003
- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016



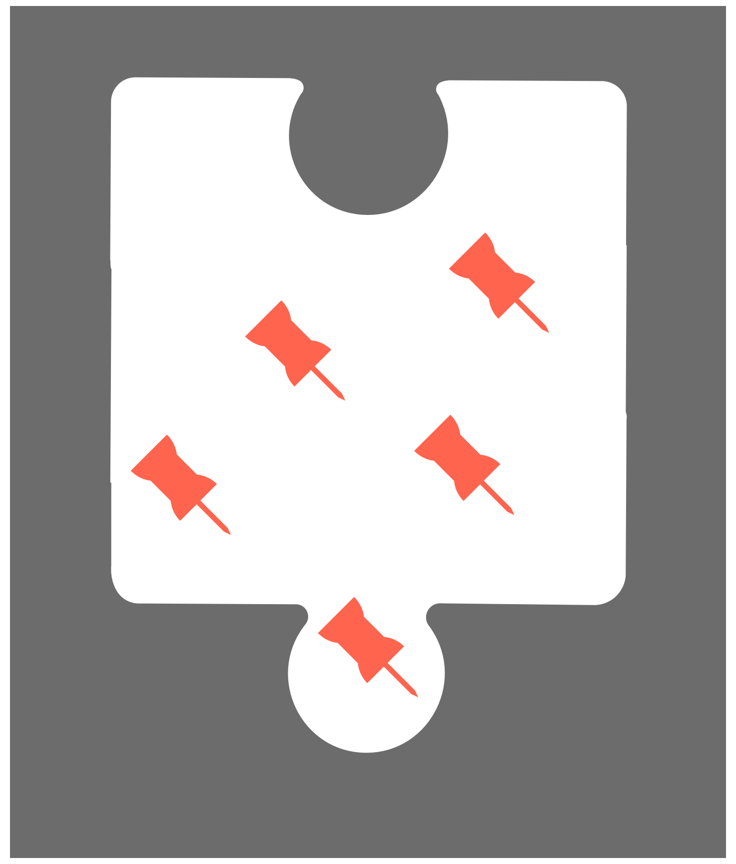
Proofs of masking for each gadget
+
Composition proofs

Non interference

- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016

Non Interference

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input.

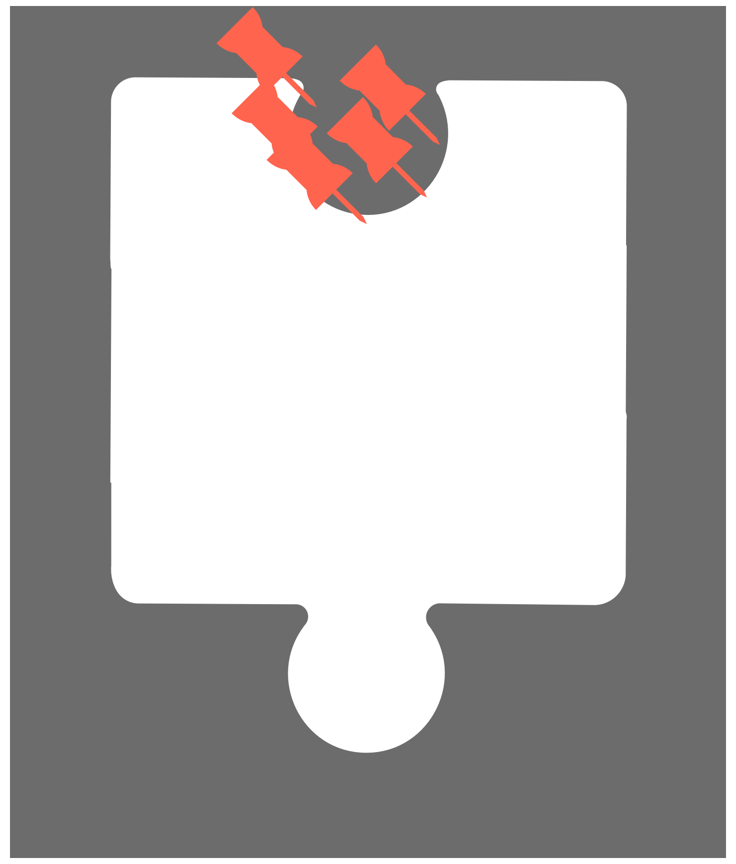


Non interference

‣ G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016

Non Interference

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input.

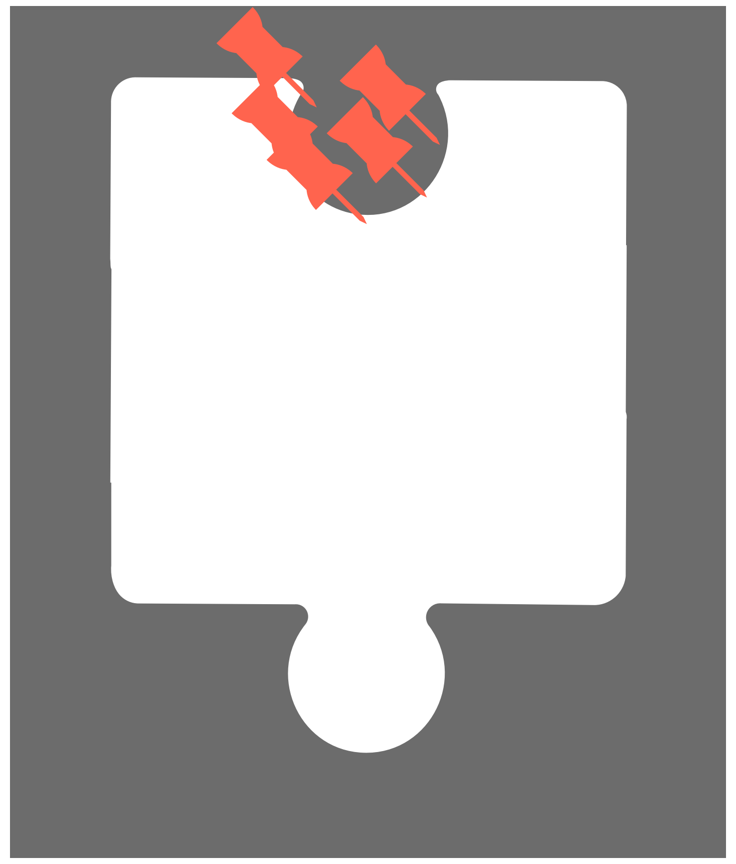


Non interference

- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016

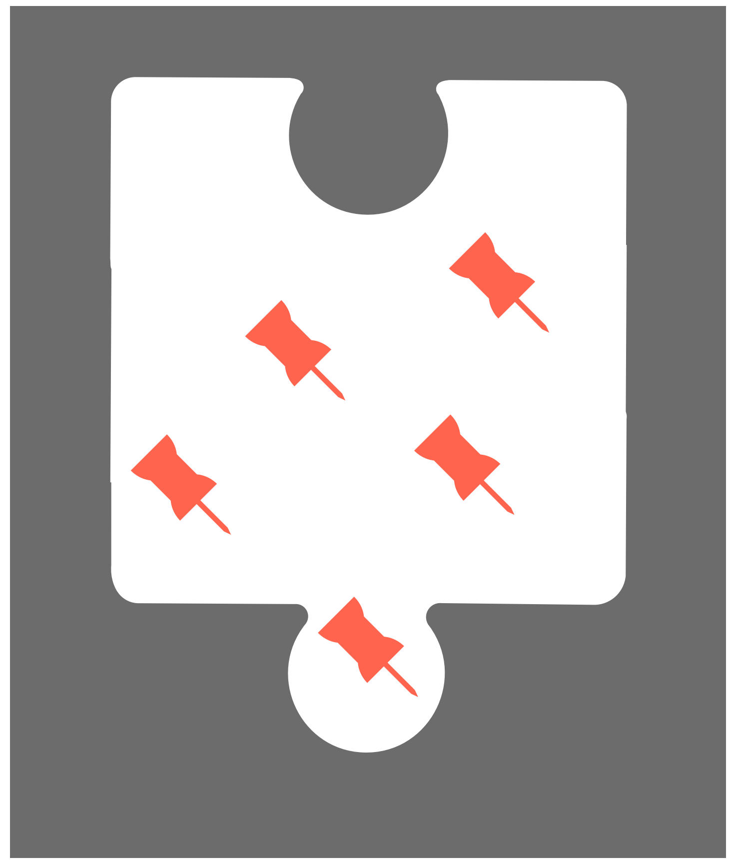
Non Interference

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input.



Strong Non Interference

A gadget is d-strongly-non-interfering (NI) iff any set of at most d observations whose d^{int} observations on the internal data and d^{out} observations on the outputs can be perfectly simulated from at most d^{int} shares of each input.

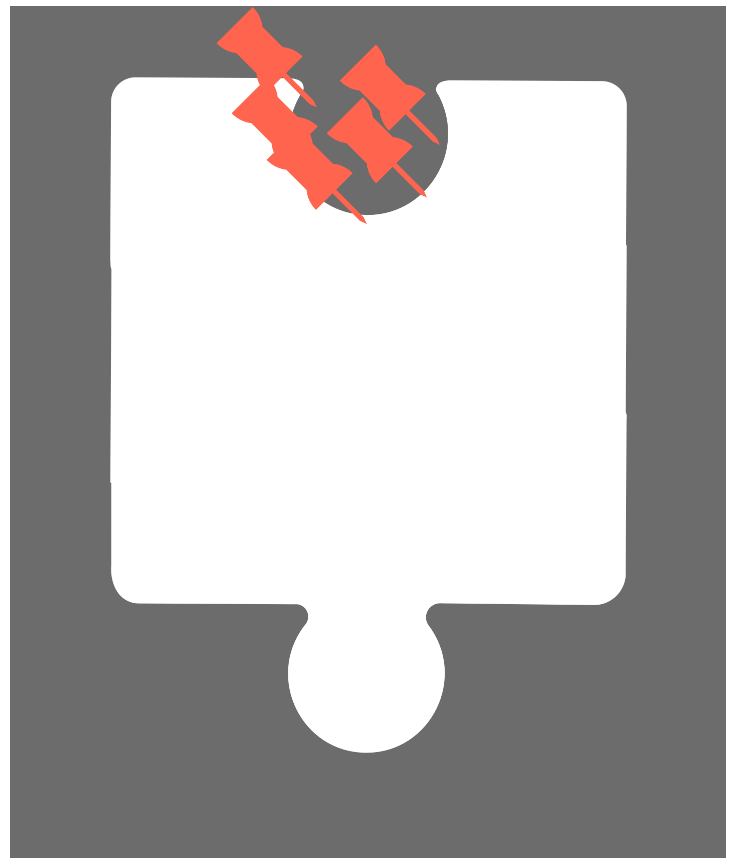


Non interference

- G. Barthe, S. Belaıd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini. ACM-CCS'2016

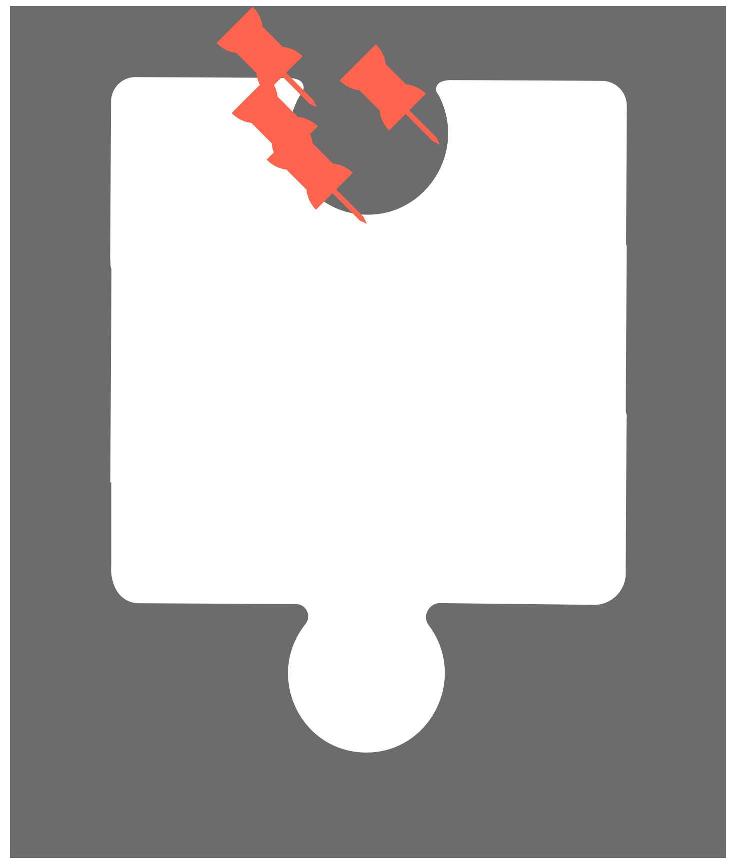
Non Interference

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input.



Strong Non Interference

A gadget is d-strongly-non-interfering (NI) iff any set of at most d observations whose d^{int} observations on the internal data and d^{out} observations on the outputs can be perfectly simulated from at most d^{int} shares of each input.



Non interference with public outputs

Signature algorithm:

1: **do**

2: $\mathbf{y} \xleftarrow{\$} Y$

3: $c \leftarrow H(\mathbf{A}\mathbf{y}, m)$

4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)

6: **return** ($\mathbf{z}$, c)

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

Signature algorithm:

1: **do**

2: $\quad \mathbf{y} \xleftarrow{\$} Y$

3: $\quad c \leftarrow H(\mathbf{Ay}, m)$

4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$

5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$

6: **return** $(\mathbf{z}, c)$

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

2 Besides, by design, the number of iterations may be public.
Thus the bit corresponding to Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$ may be revealed.

Signature algorithm:

1: **do**
2: $\quad \mathbf{y} \xleftarrow{\$} Y$
3: $\quad c \leftarrow H(\mathbf{A}\mathbf{y}, m)$
4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$
6: **return** $(\mathbf{z}, c)$

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

2 Besides, by design, the number of iterations may be public.
Thus the bit corresponding to Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$ may be revealed.

3 In addition, since $\mathbf{Ay} = \mathbf{Az} - \mathbf{t}c \mod q$ the value of $\mathbf{Ay}$ of the final iteration may be revealed.

Signature algorithm:

1: do
2: $\mathbf{y} \xleftarrow{\$} Y$
3: $c \leftarrow H(\mathbf{Ay}, m)$
4: $\mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: while Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$
6: return $(\mathbf{z}, c)$

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

2 Besides, by design, the number of iterations may be public.
Thus the bit corresponding to Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$ may be revealed.

3 In addition, since $\mathbf{Ay} = \mathbf{Az} - \mathbf{t}c \mod q$ the value of $\mathbf{Ay}$ ~~of the final iteration~~ may be revealed.
Under a mild assumption

Signature algorithm:

1: **do**
2: $\quad \mathbf{y} \xleftarrow{\$} Y$
3: $\quad c \leftarrow H(\mathbf{Ay}, m)$
4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$
6: **return** $(\mathbf{z}, c)$

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

2 Besides, by design, the number of iterations may be public.
Thus the bit corresponding to Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$ may be revealed.

3 In addition, since $\mathbf{Ay} = \mathbf{Az} - \mathbf{t}c \mod q$ the value of $\mathbf{Ay}$ ~~of the final iteration~~ may be revealed.
Under a mild assumption

Signature algorithm:

1: **do**
2: $\quad \mathbf{y} \xleftarrow{\$} Y$
3: $\quad c \leftarrow H(\mathbf{Ay}, m)$
4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected$(\mathbf{z}, \mathbf{c}, \mathbf{S})$
6: **return** $(\mathbf{z}, c)$

Non Interference with public outputs

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input and the public outputs.

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, B. Grégoire, M. Rossi and M. Tibouchi. EUROCRYPT'2017.

Non interference with public outputs

1 The signature $(\mathbf{z}, c)$ and the message m are public.

2 Besides, by design, the number of iterations may be public.
Thus the bit corresponding to Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$) may be revealed.

3 In addition, since $\mathbf{Ay} = \mathbf{Az} - \mathbf{t}c \mod q$ the value of $\mathbf{Ay}$ ~~of the final iteration~~ may be revealed.
Under a mild assumption

Signature algorithm:

1: **do**
2: $\quad \mathbf{y} \xleftarrow{\$} Y$
3: $\quad c \leftarrow H(\mathbf{Ay}, m)$
4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)
6: **return** $(\mathbf{z}, c)$

$$\approx$$

Signature algorithm:

1: **do**
2: $\quad \mathbf{y} \xleftarrow{\$} Y$
3: $\quad c \leftarrow H(\mathbf{Ay}, m)$
4: $\quad \mathbf{z} \leftarrow c \cdot \mathbf{S} + \mathbf{y}$
5: **while** Rejected($\mathbf{z}$, $\mathbf{c}$, $\mathbf{S}$)
6: **return** $(\mathbf{z}, c, \boxed{\mathbf{Ay}, \text{nb of iterations}})$

Non Interference with public outputs

A gadget is d-non-interfering (NI) iff any set of at most d observations can be perfectly simulated from at most d shares of each input and the public outputs.

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, B. Grégoire, M. Rossi and M. Tibouchi. EUROCRYPT'2017.

Masking for lattice-based cryptography

Need for lattice adapted gadgets

Small uniform random generation in $\mathbb{Z}/q\mathbb{Z}$

Gaussian generation

Rejection sampling

The constructions must use mask conversions

$$x$$

$$x_0 + x_1 + x_2 + x_3 + x_4$$

- J.-S. Coron, J. Großschädl and P. K. Vadnala CHES'2014
- J.-S. Coron, J. Großschädl, M. Tibouchi, and P. K. Vadnala FSE'2015
- J.-S. Coron CHES'2017

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi. ACM-CCS'2019.

Masking for lattice-based cryptography

Need for lattice adapted gadgets

- Small uniform random generation in $\mathbb{Z}/q\mathbb{Z}$
- Gaussian generation
- Rejection sampling

The constructions must use mask conversions

$$x$$

$$x_0 \oplus x_1 \oplus x_2 \oplus x_3 \oplus x_4$$

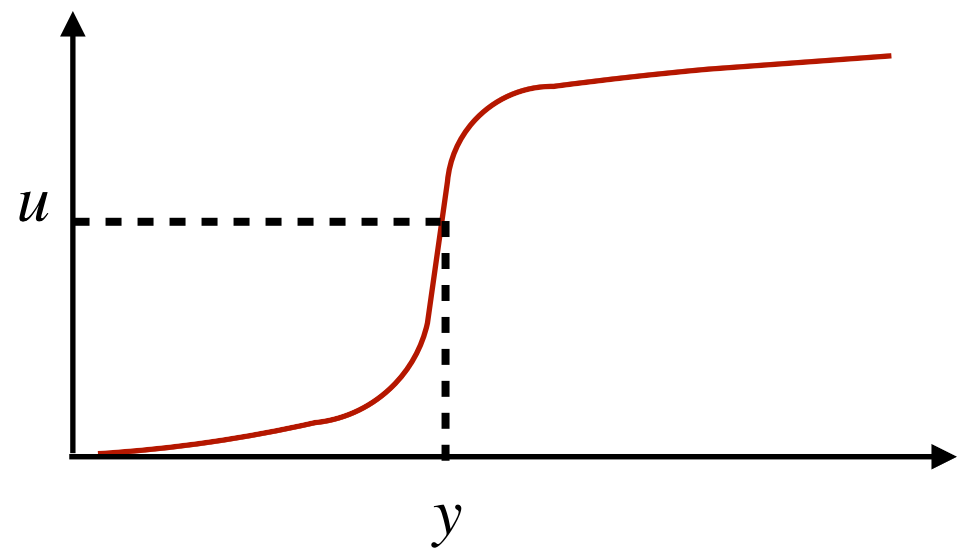
- J.-S. Coron, J. Großschädl and P. K. Vadnala CHES'2014
- J.-S. Coron, J. Großschädl, M. Tibouchi, and P. K. Vadnala FSE'2015
- J.-S. Coron CHES'2017

- G. Barthe, S. Belaïd, T. Espitau, P.-A. Fouque, M. Rossi and M. Tibouchi. ACM-CCS'2019.

Masking Gaussian sampling

How to mask Gaussian generation?

	Fixed center	**Masked variable center**
Fixed standard deviation	Masking the CDT sampling	Mitaka's share by share sampling
Masked variable standard deviation	Mask the existing convolution and rejection sampling techniques.	



$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

? ? ? ?

z_0 z_1 z_2 z_3

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sigma}$

$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

$D_{\mathbb{Z},c_0,\sigma}$ $D_{\mathbb{Z},c_1,\sigma}$ $D_{\mathbb{Z},c_2,\sigma}$ $D_{\mathbb{Z},c_3,\sigma}$

z_0 z_1 z_2 z_3

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sigma}$

$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

$D_{\mathbb{Z},c_0,\sigma}$ $D_{\mathbb{Z},c_1,\sigma}$ $D_{\mathbb{Z},c_2,\sigma}$ $D_{\mathbb{Z},c_3,\sigma}$

z_0 z_1 z_2 z_3

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sqrt{d}\sigma}$

$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

$D_{\frac{1}{B}\mathbb{Z},c_0,\frac{1}{\sqrt{d}}\sigma}$ $\quad\dots\quad$ $D_{\frac{1}{B}\mathbb{Z},c_3,\frac{1}{\sqrt{d}}\sigma}$

$z_0 \quad z_1 \quad z_2 \quad z_3$

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sqrt{d}\sigma}$

$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

$D_{\frac{1}{B}\mathbb{Z},c_0,\frac{1}{\sqrt{d}}\sigma}$ ... $D_{\frac{1}{B}\mathbb{Z},c_3,\frac{1}{\sqrt{d}}\sigma}$

z_0 z_1 z_2 z_3

such that $\sum z_i \mod q \sim D_{\frac{1}{B}\mathbb{Z},\sum c_i \mod q,\sigma}$

$D_{\mathbb{Z},c,\sigma}$

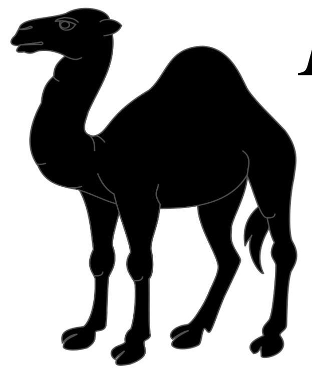
Gaussian share by share sampling

$D_{\frac{1}{B}\mathbb{Z},c_0,\frac{1}{\sqrt{d}}\sigma}$ $\ldots$ $D_{\frac{1}{B}\mathbb{Z},c_3,\frac{1}{\sqrt{d}}\sigma}$

z_0 z_1 z_2 z_3

Reject if $\sum (z_i \mod 1) \neq 0$

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sigma}$



$D_{\mathbb{Z},c,\sigma}$

Gaussian share by share sampling

$D_{\frac{1}{B}\mathbb{Z},c_0,\frac{1}{\sqrt{d}}\sigma}$ $\dots$ $D_{\frac{1}{B}\mathbb{Z},c_3,\frac{1}{\sqrt{d}}\sigma}$

z_0 z_1 z_2 z_3

Reject if $\sum (z_i \mod 1) \neq 0$

such that $\sum z_i \mod q \sim D_{\mathbb{Z},\sum c_i \mod q,\sigma}$

Gauss share by share

This Gaussian share by share sampling is correct and secure.

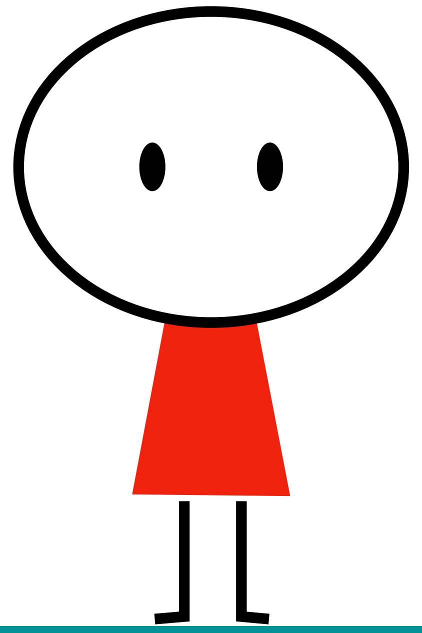
- « Mitaka: A Simpler, Parallelizable, Maskable Variant of Falcon »
  T. Espitau, P.-A. Fouque, F. Gérard, M. Rossi, A. Takahashi, M. Tibouchi, A. Wallet, Y. Yu EUROCRYPT'2022

Example of performance

▸ F. Gérard and M. Rossi. CARDIS'2019.

Examples of overhead on the number of cycles for qTesla signature scheme

Unmasked	Order 1	Order 2	Order 3	Order 4
1	$\times 4$	$\times 21$	$\times 37$	$\times 60$

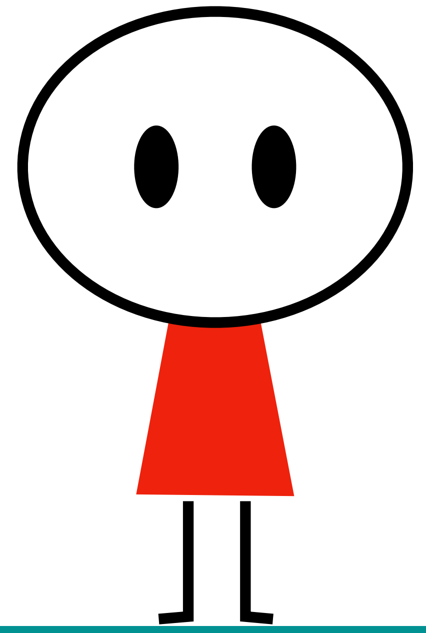


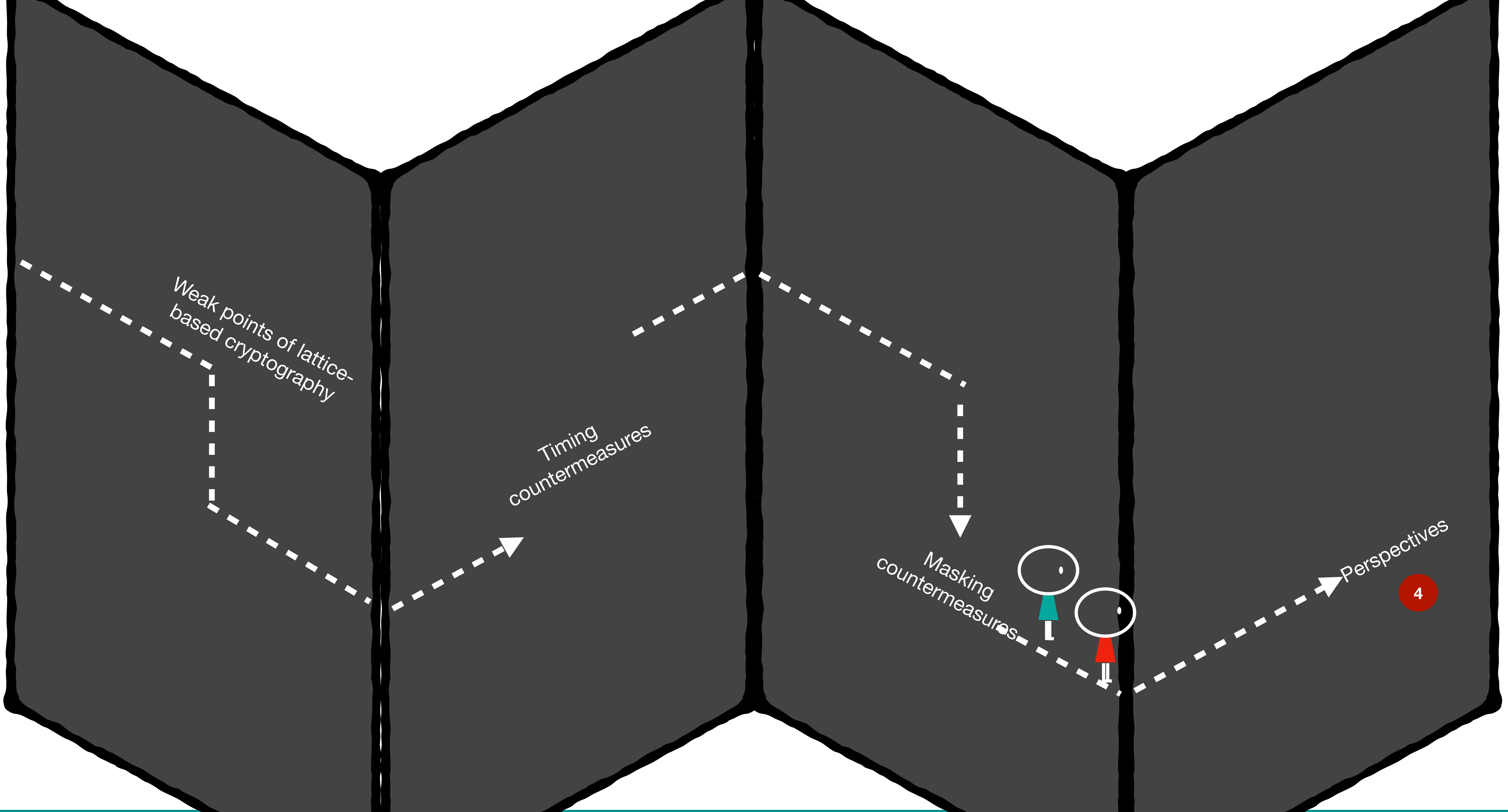
Example of performance

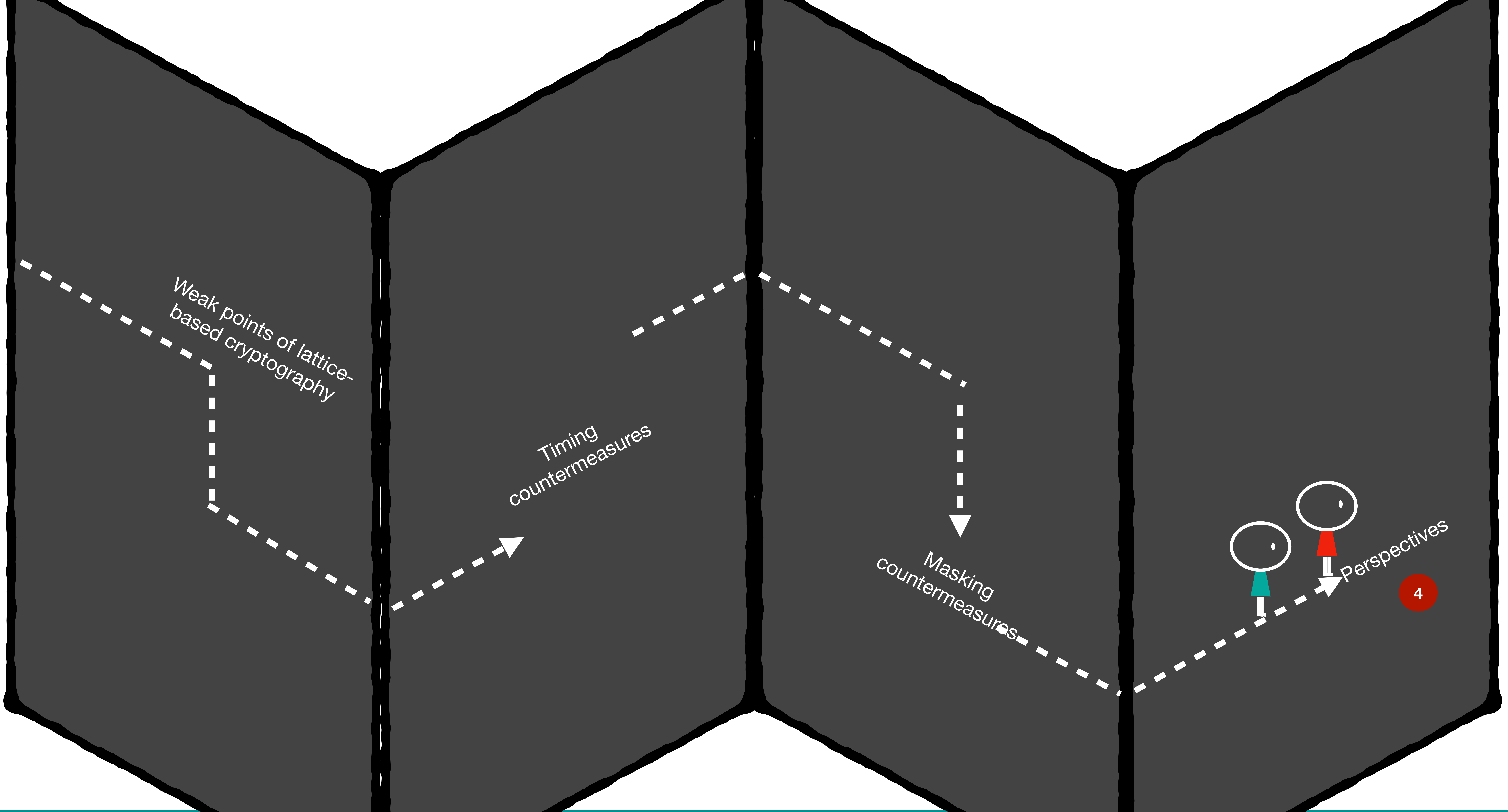
▸ F. Gérard and M. Rossi. CARDIS'2019.

Examples of overhead on the number of cycles for qTesla signature scheme

Unmasked	Order 1	Order 2	Order 3	Order 4
1	$\times 4$	$\times 21$	$\times 37$	$\times 60$




Weak points of lattice-based cryptography
Timing countermeasures
Masking countermeasures
Perspectives
4


Weak points of lattice-based cryptography
Timing countermeasures
Masking countermeasures
Perspectives
4

Automated verification of side-channel protection in the lattice domain

Provable countermeasures are not infaillible

Automated verification of side-channel protection in the lattice domain

Provable countermeasures are not infaillible

Besides proofs, how to verify automatically the isochrony of lattice-based crypto?

Automated verification of side-channel protection in the lattice domain

Provable countermeasures are not infaillible

Besides proofs, how to verify automatically the isochrony of lattice-based crypto?

Existing tools

- B. Rodrigues, F. Magno Quintao Pereira, D. Aranha ACM'2016
- « ct-verif » J. Barcelar Almeida, M. Barbosa, G. Barthe, F. Dupressoir, M. Emmi USENIX'16
- « Dudect » O. Reparaz, J. Balasch, I. Verbauwhede DATE'17

Automated verification of side-channel protection in the lattice domain

Provable countermeasures are not infaillible

Besides proofs, how to verify automatically the isochrony of lattice-based crypto?

Existing tools

- B. Rodrigues, F. Magno Quintao Pereira, D. Aranha ACM'2016
- « ct-verif » J. Barcelar Almeida, M. Barbosa, G. Barthe, F. Dupressoir, M. Emmi USENIX'16
- « Dudect » O. Reparaz, J. Balasch, I. Verbauwhede DATE'17

Intuition:

- generate two random keys
- sign many messages or decrypt many ciphertexts with either of the two keys
- look for statistical differences in the timing among the two keys

Automated verification of side-channel protection in the lattice domain

Provable countermeasures are not infaillible

Besides proofs, how to verify automatically the isochrony of lattice-based crypto?

Existing tools
- B. Rodrigues, F. Magno Quintao Pereira, D. Aranha ACM'2016
- « ct-verif » J. Barcelar Almeida, M. Barbosa, G. Barthe, F. Dupressoir, M. Emmi USENIX'16
- « Dudect » O. Reparaz, J. Balasch, I. Verbauwhede DATE'17

Intuition:
- generate two random keys
- sign many messages or decrypt many ciphertexts with either of the two keys
- look for statistical differences in the timing among the two keys

Challenges for lattice-based crypto :

- [ ] How to handle inherent variable execution time ?
- [ ] The sensitive values are not only the keys but many intermediate randomness are sensitive

Automated verification of side-channel protection in the lattice domain

Besides proofs, how to verify automatically the masking of lattice-based crypto?

Automated verification of side-channel protection in the lattice domain

Besides proofs, how to verify automatically the masking of lattice-based crypto?

Many existing tools for verifying Boolean masking

- B. Gigerl, V. Hadzic, R. Primas, S. Mangard, R. Bloem USENIX Security'21
- R. Bloem, H. Gross, R. Iusupov, B. Könighofer, S. Mangard, J. Winter EUROCRYP'18
- G. Barthe, S. Belaïd, G. Cassiers, P.-A. Fouqe, B. Gregoire, F.-X. Standaret ESORICS'19
- V. Hadzic, R. Bloem FMCAD'21
- D. Knichel, P. Sasdrich, A. Moradi ASIACRYPT'20
- G. Barthe, M. Goujon, B. Grégoire, M. Orlt, C. Paglialonga, L. Porth TCHES'21

Automated verification of side-channel protection in the lattice domain

Besides proofs, how to verify automatically the masking of lattice-based crypto?

Many existing tools for verifying Boolean masking

- B. Gigerl, V. Hadzic, R. Primas, S. Mangard, R. Bloem USENIX Security'21
- R. Bloem, H. Gross, R. Iusupov, B. Könighofer, S. Mangard, J. Winter EUROCRYP'18
- G. Barthe, S. Belaïd, G. Cassiers, P.-A. Fouqe, B. Gregoire, F.-X. Standaret ESORICS'19
- V. Hadzic, R. Bloem FMCAD'21
- D. Knichel, P. Sasdrich, A. Moradi ASIACRYPT'20
- G. Barthe, M. Goujon, B. Grégoire, M. Orlt, C. Paglialonga, L. Porth TCHES'21

Lattice-based crypto is essentially masked in arithmetic form

Automated verification of side-channel protection in the lattice domain

Besides proofs, how to verify automatically the masking of lattice-based crypto?

Many existing tools for verifying Boolean masking

- B. Gigerl, V. Hadzic, R. Primas, S. Mangard, R. Bloem USENIX Security'21
- R. Bloem, H. Gross, R. Iusupov, B. Könighofer, S. Mangard, J. Winter EUROCRYP'18
- G. Barthe, S. Belaïd, G. Cassiers, P.-A. Fouqe, B. Gregoire, F.-X. Standaret ESORICS'19
- V. Hadzic, R. Bloem FMCAD'21
- D. Knichel, P. Sasdrich, A. Moradi ASIACRYPT'20
- G. Barthe, M. Goujon, B. Grégoire, M. Orlt, C. Paglialonga, L. Porth TCHES'21

Lattice-based crypto is essentially masked in arithmetic form

Challenges: Arithmetic and Boolean masking

Conversions

Automated verification of side-channel protection in the lattice domain

Besides proofs, how to verify automatically the masking of lattice-based crypto?

Many existing tools for verifying Boolean masking

- B. Gigerl, V. Hadzic, R. Primas, S. Mangard, R. Bloem USENIX Security'21
- R. Bloem, H. Gross, R. Iusupov, B. Könighofer, S. Mangard, J. Winter EUROCRYP'18
- G. Barthe, S. Belaïd, G. Cassiers, P.-A. Fouqe, B. Gregoire, F.-X. Standaret ESORICS'19
- V. Hadzic, R. Bloem FMCAD'21
- D. Knichel, P. Sasdrich, A. Moradi ASIACRYPT'20
- G. Barthe, M. Goujon, B. Grégoire, M. Orlt, C. Paglialonga, L. Porth TCHES'21

Lattice-based crypto is essentially masked in arithmetic form

Challenges: Arithmetic and Boolean masking

Conversions

Partial resolution:

- « Formal verification of Arithmetic Masking in Hardware and Software » B. Gigerl, R. Primas, S. Magnard eprint.iacr.org/2022/849

Modeling arithmetic expression with Boolean logic

Applied to A2B and B2A

Other perspectives

Masking friendly design

The designs contain many « masking unfriendly » features: Gaussian distributions, uniform small distributions, comparison of sensitive values, rejection, prime modulus…

➡ Schemes designs that minimize the masking overhead at a cost of less efficient unmasked version.

Other perspectives

Masking friendly design

The designs contain many « masking unfriendly » features: Gaussian distributions, uniform small distributions, comparison of sensitive values, rejection, prime modulus…

➡ Schemes designs that minimize the masking overhead at a cost of less efficient unmasked version.

Fujisaki-Okamoto transform

This transform is needed because it protects against active attacks (IND-CCA2 security) but it highly increases the attack surface and introduces new attack entry points.

➡ Is re-encryption (or similar tests) inevitable?
➡ Is it possible to design a fully protected generic Fujisaki-Okamoto transform?