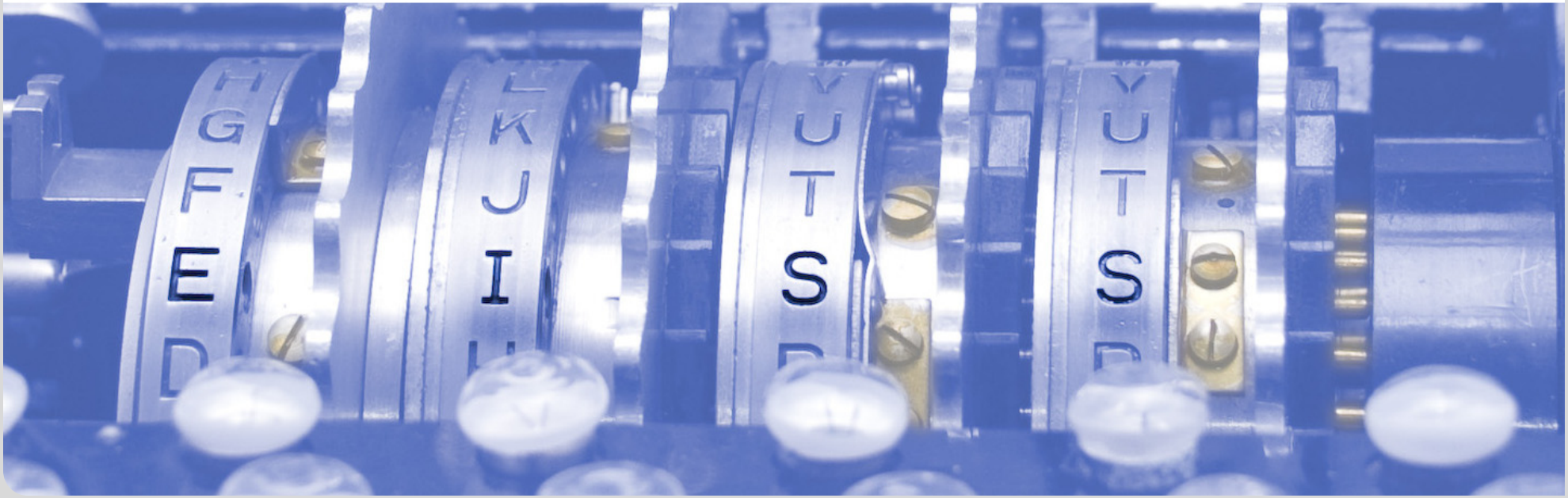
Karlsruhe Institute of Technology

Fault-Tolerant Aggregate Signatures

March, 7th 2016

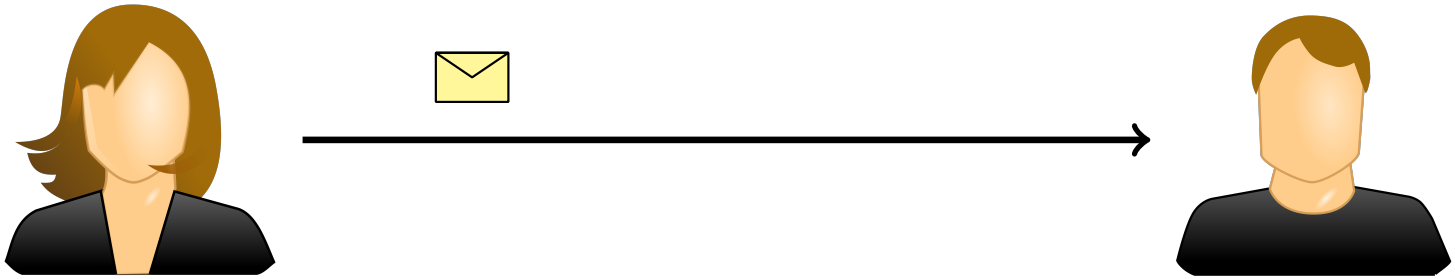
Gunnar Hartung Björn Kaidel Alexander Koch Jessica Koch Andy Rupp

INSTITUTE OF THEORETICAL COMPUTER SCIENCE



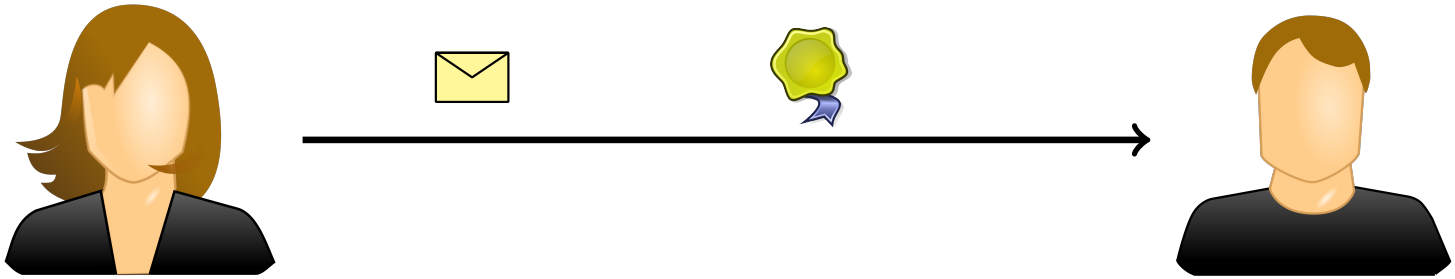
Outline

Introduction: Aggregate Signatures



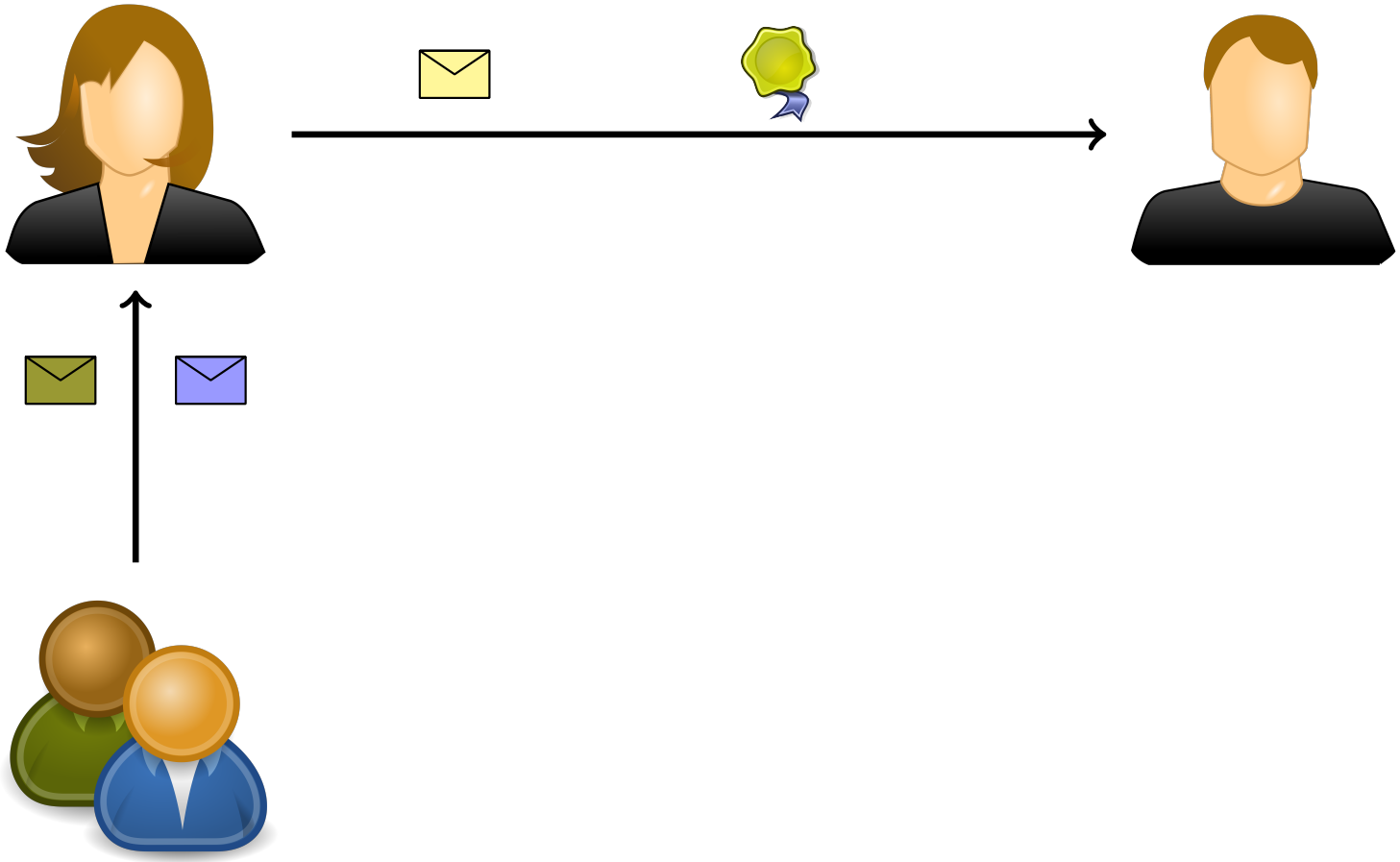
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



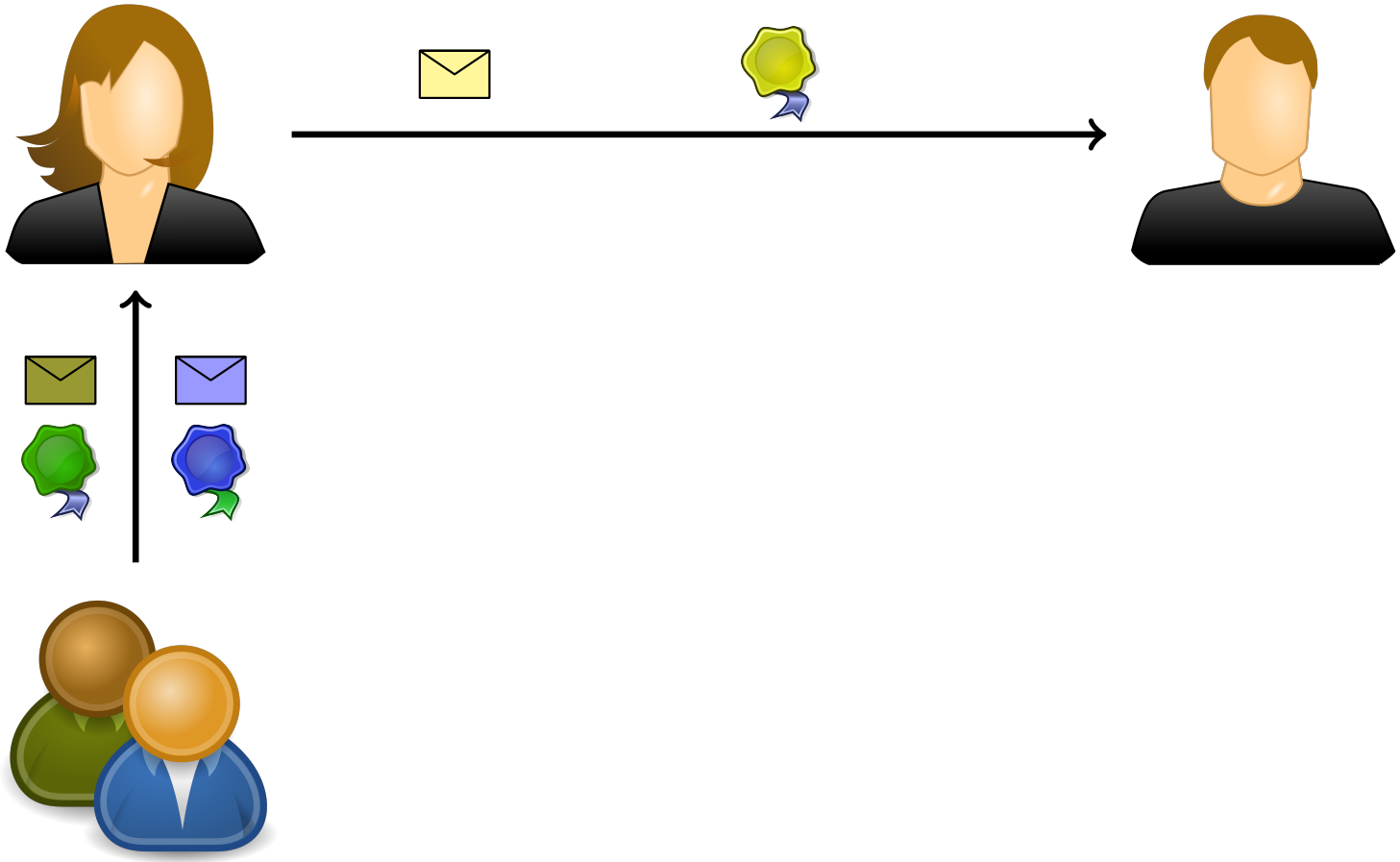
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



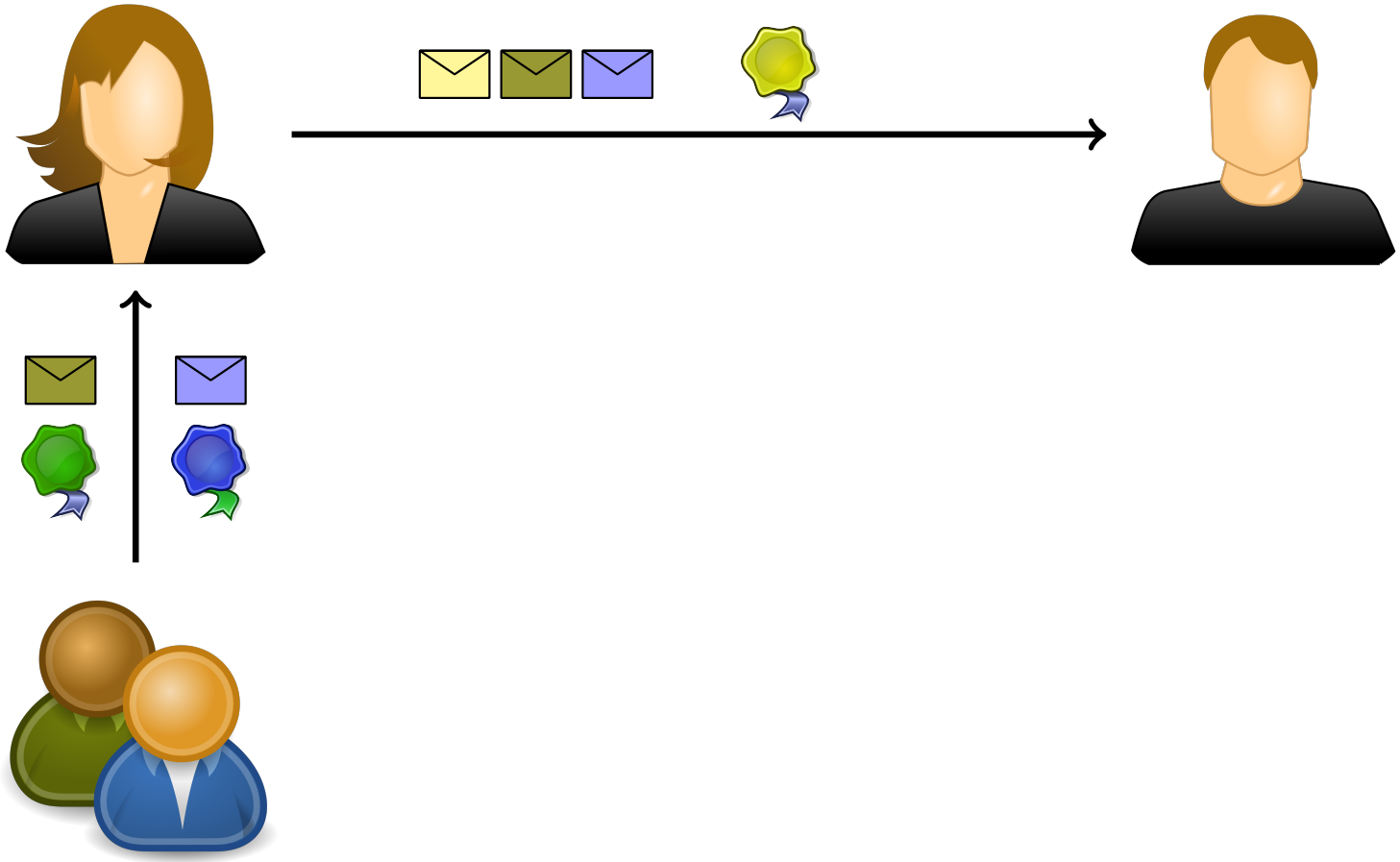
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



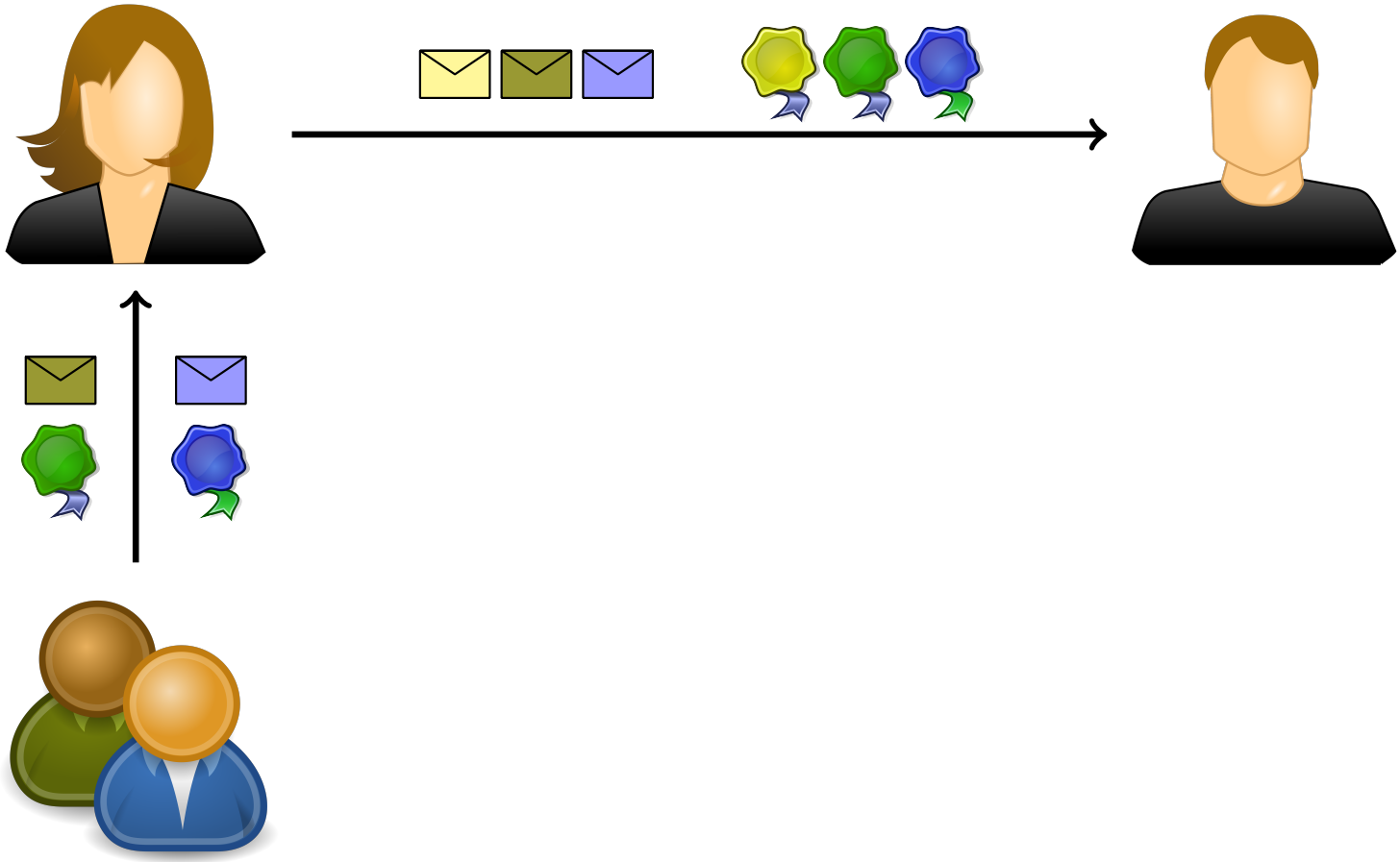
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



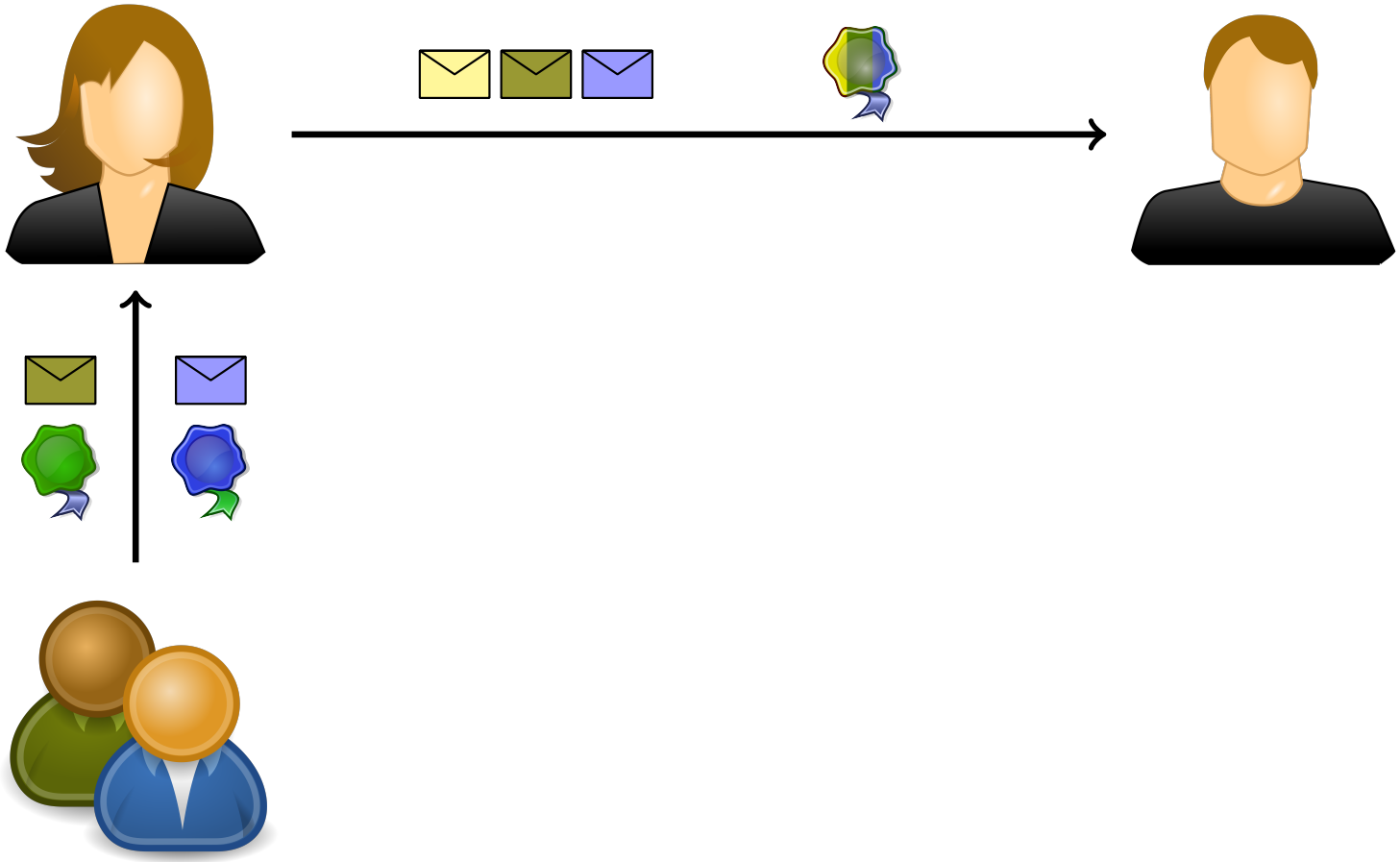
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



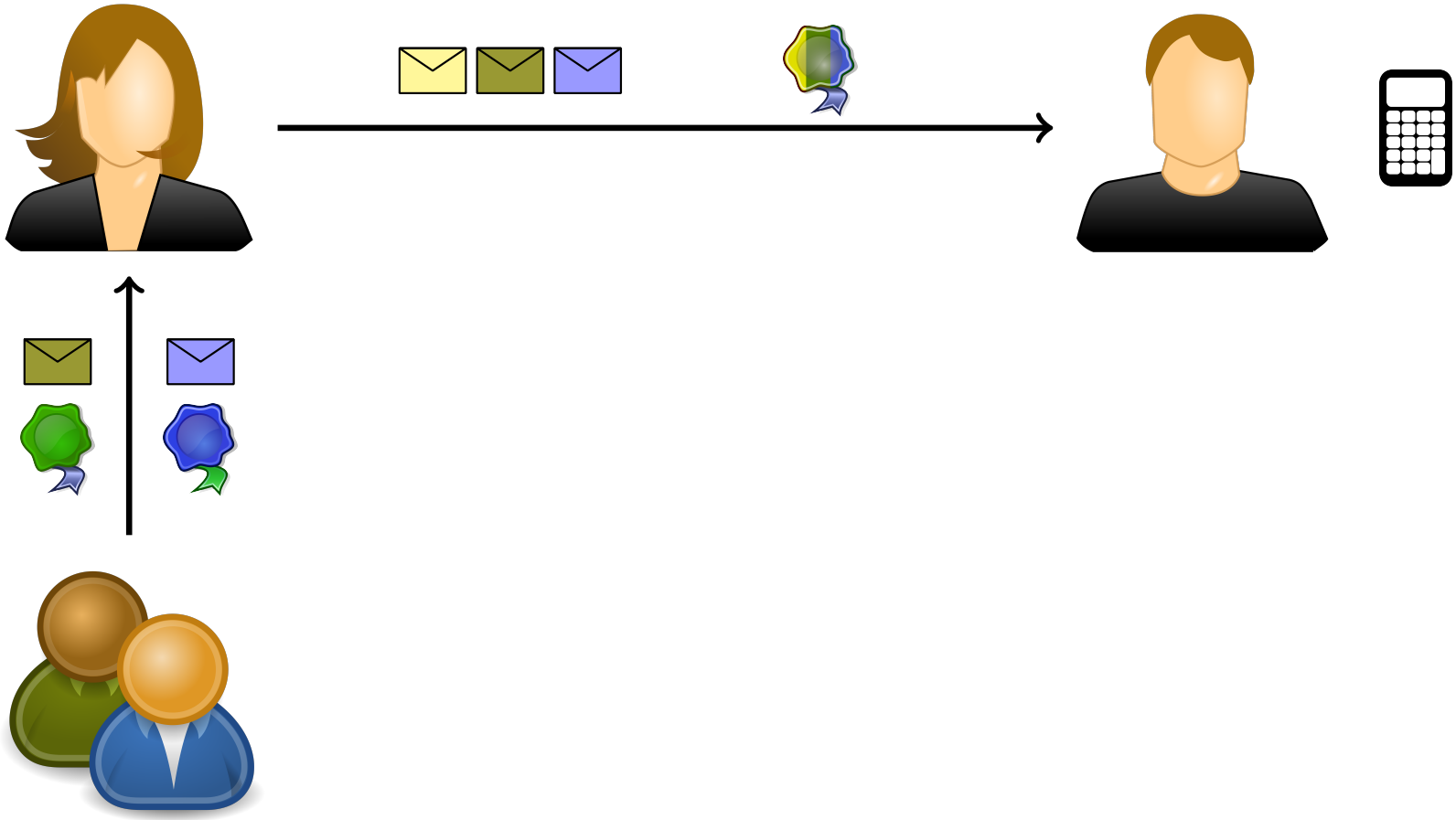
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



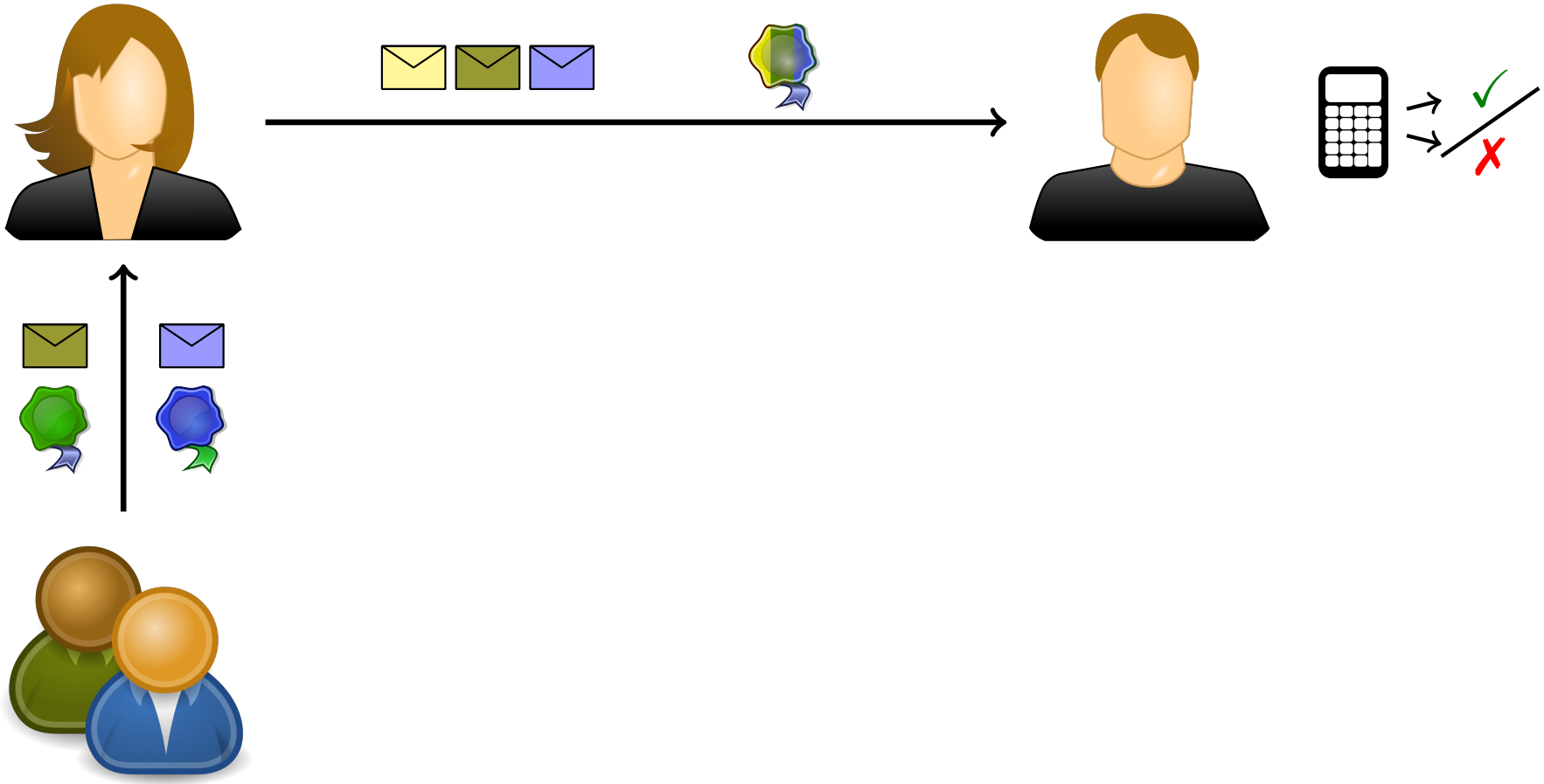
Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Aggregate Signatures

What happens if Verification fails?

Aggregate Signatures

What happens if Verification fails?

Bob doesn't know...

Aggregate Signatures

What happens if Verification fails?

Bob doesn't know...

- which messages are invalid

Aggregate Signatures

What happens if Verification fails?

Bob doesn't know...

- which messages are invalid
- how many messages are invalid

Aggregate Signatures

What happens if Verification fails?

Bob doesn't know...

- which messages are invalid
- how many messages are invalid
- which messages are still valid

Aggregate Signatures

What happens if Verification fails?

Bob doesn't know...

- which messages are invalid
- how many messages are invalid
- which messages are still valid

$\Longrightarrow$ Bob can not trust any message.

Aggregate Signatures

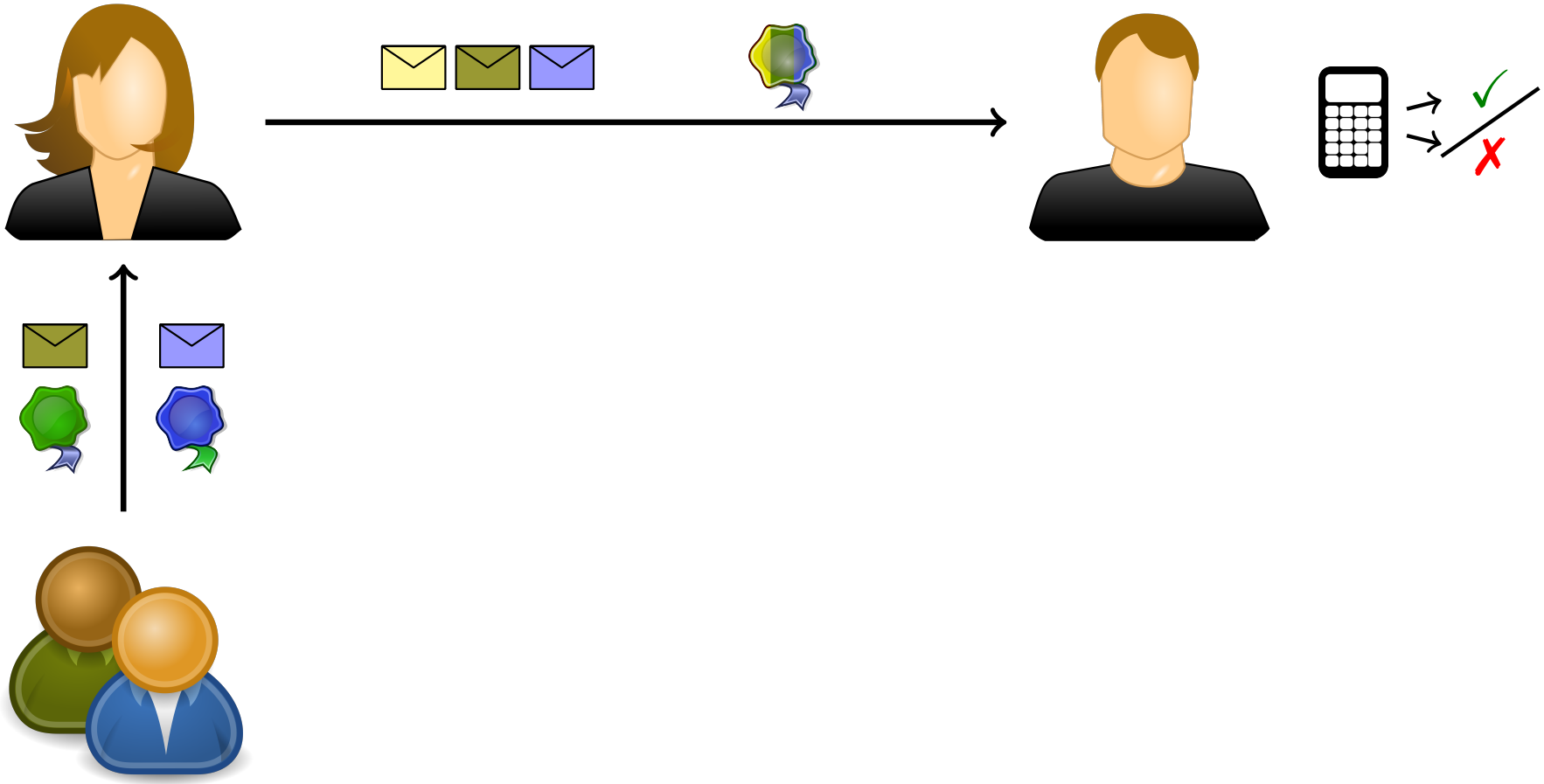
What happens if Verification fails?

Bob doesn't know...

- which messages are invalid
- how many messages are invalid
- which messages are still valid

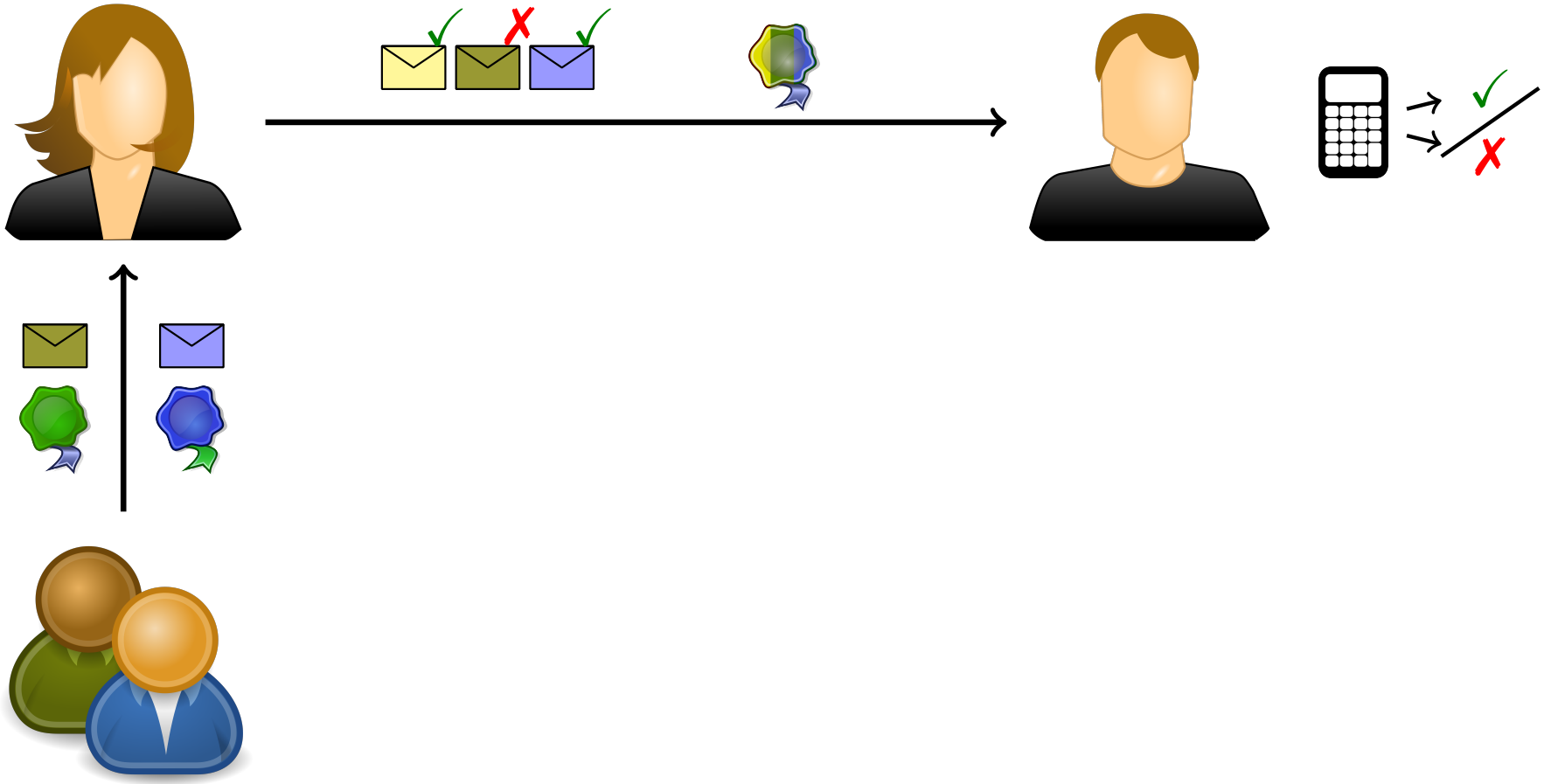
$\Longrightarrow$ Bob can not trust any message.
$\Longrightarrow$ resend data and/or agg. signature

Introduction: Aggregate Signatures



Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Introduction: Aggregate Signatures



Images: CC-0 by dagobert83 via openclipart, Public Domain by computating, warszawianka, and cinemacookie

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)
An aggregate signature scheme is *d-fault-tolerant*, iff

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathrm{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathrm{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$
- with up to d invalid triples,

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathsf{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$
- with up to d invalid triples,
- letting $\tau := \mathsf{Agg}(\sigma_1, \ldots, \sigma_n)$

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathsf{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$
- with up to d invalid triples,
- letting $\tau := \mathsf{Agg}(\sigma_1, \ldots, \sigma_n)$
- $\mathsf{Verify}(\{(\mathsf{pk}_i, m_i)\}, \tau)$ outputs at least all valid messages.

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathsf{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$
- with up to d invalid triples,
- letting $\tau := \mathsf{Agg}(\sigma_1, \ldots, \sigma_n)$
- $\mathsf{Verify}(\{(\mathsf{pk}_i, m_i)\}, \tau)$ outputs at least all valid messages.

- 0-fault-tolerance $\hat{=}$ correctness

Fault-Tolerance

- The verification algorithm outputs a list of valid messages.

Definition: Fault-Tolerance (informal)

An aggregate signature scheme is *d-fault-tolerant*, iff

- given $(\mathsf{pk}_i, m_i, \sigma_i)$ for $i = 1, \ldots, n$
- with up to d invalid triples,
- letting $\tau := \mathsf{Agg}(\sigma_1, \ldots, \sigma_n)$
- $\mathsf{Verify}(\{(\mathsf{pk}_i, m_i)\}, \tau)$ outputs at least all valid messages.

- 0-fault-tolerance $\hat{=}$ correctness
- generalization of correctness

Our Result

Our Result

Fault-Tolerant Aggregate Signature

Our Result

Fault-Tolerant Aggregate Signature

- We construct an aggregate signature scheme, which can detect a fixed number d of faults

Our Result

Fault-Tolerant Aggregate Signature

- We construct an aggregate signature scheme, which can detect a fixed number d of faults
- via a black-box transformation from an arbitrary aggregate signature scheme,

Our Result

Fault-Tolerant Aggregate Signature

- We construct an aggregate signature scheme, which can detect a fixed number d of faults
- via a black-box transformation from an arbitrary aggregate signature scheme,
- using Cover-Free-Families [KRS99].

Outline

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme

$$\left(\qquad \right)$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \quad \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \\ \\ \\ \\ \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \\ \text{✗} \\ \\ \end{matrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✓} \\ \text{✗} \\ \text{✓} \end{matrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✓} \\ \text{✗} \\ \text{✓} \end{matrix}$$

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✓} \\ \text{✗} \\ \text{✓} \end{matrix}$$

Why does this work?

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✓} \\ \text{✗} \\ \text{✓} \end{matrix}$$

Why does this work?

No single column “covers” any other column.

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✓} \\ \text{✗} \\ \text{✓} \end{matrix}$$

Why does this work?

No single column “covers” any other column. (But 2 do!)

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✗} \\ \text{✗} \\ \text{✗} \end{matrix}$$

Why does this work?

No single column “covers” any other column. (But 2 do!)

Our Approach

- signature $\hat{=}$ *vector* of signatures of underlying scheme
- columns $\hat{=}$ individual signatures

$$\begin{pmatrix} \sigma_1 & 0 & 0 & \sigma_4 & 0 & \sigma_6 \\ \sigma_1 & \sigma_2 & 0 & 0 & \sigma_5 & 0 \\ 0 & \sigma_2 & \sigma_3 & \sigma_4 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \text{✗} \\ \text{✗} \\ \text{✗} \\ \text{✗} \end{matrix}$$

Why does this work?

No single column “covers” any other column. (But 2 do!)

Incidence matrices of cover-free families [KRS99] allow for more invalid signatures.

Outline

Security

Recall: Verify outputs a list of valid messages

Definition (Security, informal)

A fault-tolerant signature scheme is *secure* iff no PPT adversary (with signing oracle) can forge an (aggregate) signature.

Our Construction:

- Verify outputs the union of messages in valid rows

  $\Longrightarrow$ security inherited from the underlying scheme

Compression Ratio

Definition (Compression Ratio)

Let σ^* be a maximum size aggregate signature for n individual signatures. An aggregate signature scheme has *compression ratio* $\rho(n)$ iff

$$\frac{n}{\text{size}(\sigma^*)} \in \Theta(\rho(n)).$$

Compression Ratio

Definition (Compression Ratio)

Let σ^* be a maximum size aggregate signature for n individual signatures. An aggregate signature scheme has *compression ratio* $\rho(n)$ iff

$$\frac{n}{\text{size}(\sigma^*)} \in \Theta(\rho(n)).$$

Example: constant signature size $\rightsquigarrow \rho(n) = n$

Compression Ratio

Definition (Compression Ratio)

Let σ^* be a maximum size aggregate signature for n individual signatures. An aggregate signature scheme has *compression ratio* $\rho(n)$ iff

$$\frac{n}{\text{size}(\sigma^*)} \in \Theta(\rho(n)).$$

Example: constant signature size $\rightsquigarrow \rho(n) = n$

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Compression Ratio

Definition (Compression Ratio)

Let σ^* be a maximum size aggregate signature for n individual signatures. An aggregate signature scheme has *compression ratio* $\rho(n)$ iff

$$\frac{n}{\text{size}(\sigma^*)} \in \Theta(\rho(n)).$$

Example: constant signature size $\rightsquigarrow \rho(n) = n$

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Our scheme achieves this bound.

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix}$$

- check only relevant rows

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \checkmark \\ \\ \checkmark \\ \\ \end{matrix}$$

- check only relevant rows

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \begin{matrix} \checkmark \\ \\ \checkmark \\ \\ \end{matrix}$$

- check only relevant rows
- return early on success

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \checkmark$$

- check only relevant rows
- return early on success

Selective Verification

- Verification of a single message-signature pair (m_j, σ_j)
- more efficiently than full verification

Example:

$$\begin{pmatrix} \sigma_1 & & & \sigma_4 & & \sigma_6 \\ \sigma_1 & \sigma_2 & & & \sigma_5 & \\ & \sigma_2 & \sigma_3 & \sigma_4 & & \\ & & \sigma_3 & & \sigma_5 & \sigma_6 \end{pmatrix} \rightsquigarrow \begin{pmatrix} \mathrm{Agg}(\sigma_1, \sigma_4, \sigma_6) \\ \mathrm{Agg}(\sigma_1, \sigma_2, \sigma_5) \\ \mathrm{Agg}(\sigma_2, \sigma_3, \sigma_4) \\ \mathrm{Agg}(\sigma_3, \sigma_5, \sigma_6) \end{pmatrix} \checkmark$$

- check only relevant rows
- return early on success
- our construction: matrix does not need to be stored, can be generated on demand efficiently

Example Numbers

d	m	n
max.faults	dimension of vector	single signatures
2	25	125
5	121	1331
8	289	4913
4	289	$\approx 1.42 \cdot 10^6$
14	841	24389
26	2809	148877
50	10201	$\approx 1.03 \cdot 10^6$
83	63001	$\approx 3.97 \cdot 10^9$
510	1042441	$\approx 1.06 \cdot 10^9$

Conclusion

- Aggregate signature schemes can “compress” signatures from different signers.

Our Construction:

Pros:

- tolerates a fixed number of errors
- black-box transformation
- asymptotically optimal compression
- selective verification

Cons:

- a-priori upper bound on number of aggregated signatures
- compression-rate limited by fault-tolerance
- requires order among the messages

Thank you.
Questions?

Contact: gunnar.hartung@kit.edu
PGP-Key ID: B1A7 C146
Fingerprint: 4C39 AC36 6FAD 9E52 3144
8352 9E37 381F B1A7 C146
S/MIME Cert: at
https://crypto.iti.kit.edu/?id=hartung&L=2

Security Definition

Definition (Security Experiment)

- *Setup Phase.* Run $(\mathsf{pk}, \mathsf{sk}) := \mathsf{KeyGen}(1^\kappa)$ and start adversary $\mathcal{A}$ with input pk.
- *Query Phase.* $\mathcal{A}$ may adaptively issue signature queries m_i, and receives the responses $\tau_i := \mathsf{Sign}(\mathsf{sk}, m_i)$.
- *Forgery Phase.* $\mathcal{A}$ outputs a claim sequence C^* and a signature τ^*.

Definition (Security)

An aggregate signature scheme with list verification is (t, q, ε)-*secure* if there is no adversary $\mathcal{A}$ running in time at most t, making at most q queries to the signature oracle and winning in the above experiment with probability at least ε.

d-Cover-Free Families

Definition (d-Cover-Free Family [KS64])

A d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ consists of:

d-Cover-Free Families

Definition (d-Cover-Free Family [KS64])

A d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ consists of:

- a set $\mathcal{S} = \{s_1, \ldots, s_m\}$

d-Cover-Free Families

Definition (d-Cover-Free Family [KS64])

A d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ consists of:

- a set $\mathcal{S} = \{s_1, \ldots, s_m\}$
- a set $\mathcal{B} = \{B_1, \ldots, B_n\}$, where $B_i \subset \mathcal{S}$ for all i

d-Cover-Free Families

Definition (d-Cover-Free Family [KS64])

A d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ consists of:

- a set $\mathcal{S} = \{s_1, \ldots, s_m\}$
- a set $\mathcal{B} = \{B_1, \ldots, B_n\}$, where $B_i \subset \mathcal{S}$ for all i

and for any d subsets $B_{i_1}, \ldots, B_{i_d} \in \mathcal{B}$ and all distinct $B \in \mathcal{B}$ it holds:

d-Cover-Free Families

Definition (d-Cover-Free Family [KS64])

A d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ consists of:

- a set $\mathcal{S} = \{s_1, \ldots, s_m\}$
- a set $\mathcal{B} = \{B_1, \ldots, B_n\}$, where $B_i \subset \mathcal{S}$ for all i

and for any d subsets $B_{i_1}, \ldots, B_{i_d} \in \mathcal{B}$ and all distinct $B \in \mathcal{B}$ it holds:

- $|B \setminus \bigcup_{k=1}^{d} B_{i_k}| \geq 1$
  i.e. d subsets do not cover a different single subset.

d-Cover-Free Families

Definition (Incidence Matrix)

For a *d*-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

d-Cover-Free Families

Definition (Incidence Matrix)

For a d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

$$\mathcal{M}[i,j] = \begin{cases} 1, & \text{if } s_i \in B_j, \\ 0, & \text{otherwise.} \end{cases}$$

d-Cover-Free Families

Definition (Incidence Matrix)

For a d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

$$\mathcal{M}[i,j] = \begin{cases} 1, & \text{if } s_i \in B_j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{ccccccc} B_1 & B_2 & \dots & B_j & \dots & B_n \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 1 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 1 \end{pmatrix} \end{array}$$

d-Cover-Free Families

Definition (Incidence Matrix)

For a d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

$$\mathcal{M}[i,j] = \begin{cases} 1, & \text{if } s_i \in B_j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{cccccc} B_1 & B_2 & \dots & B_j & \dots & B_n \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 1 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 1 \end{pmatrix} \end{array}$$

$$s_1 \in B_1 \quad \notin B_2 \quad \in B_j \quad \notin B_n$$

d-Cover-Free Families

Definition (Incidence Matrix)

For a d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

$$\mathcal{M}[i,j] = \begin{cases} 1, & \text{if } s_i \in B_j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{ccccccc} B_1 & B_2 & \dots & B_j & \dots & B_n \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 1 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 1 \end{pmatrix} \end{array}$$

$$\begin{array}{llllll} s_1 \in B_1 & \notin B_2 & \in B_j & \notin B_n \\ s_2 \notin B_1 & \in B_2 & \notin B_j & \notin B_n \end{array}$$

d-Cover-Free Families

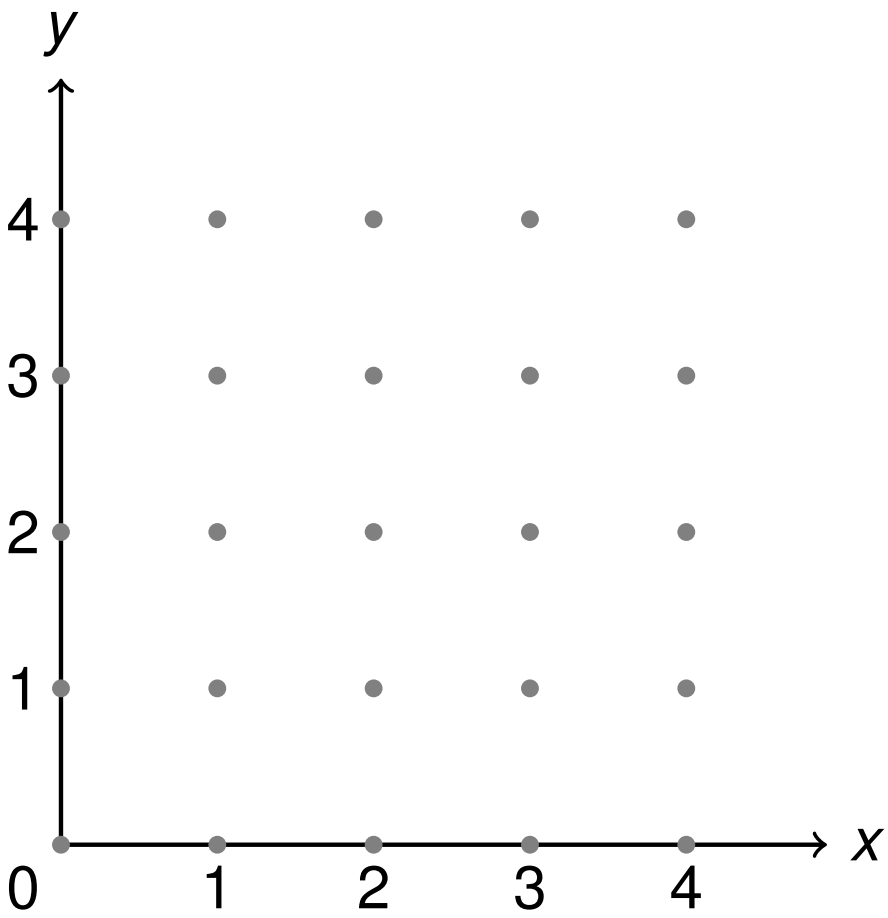
Definition (Incidence Matrix)

For a d-CFF $\mathcal{F} = (\mathcal{S}, \mathcal{B})$ its *incidence matrix* $\mathcal{M}$ is:

$$\mathcal{M}[i,j] = \begin{cases} 1, & \text{if } s_i \in B_j, \\ 0, & \text{otherwise.} \end{cases}$$

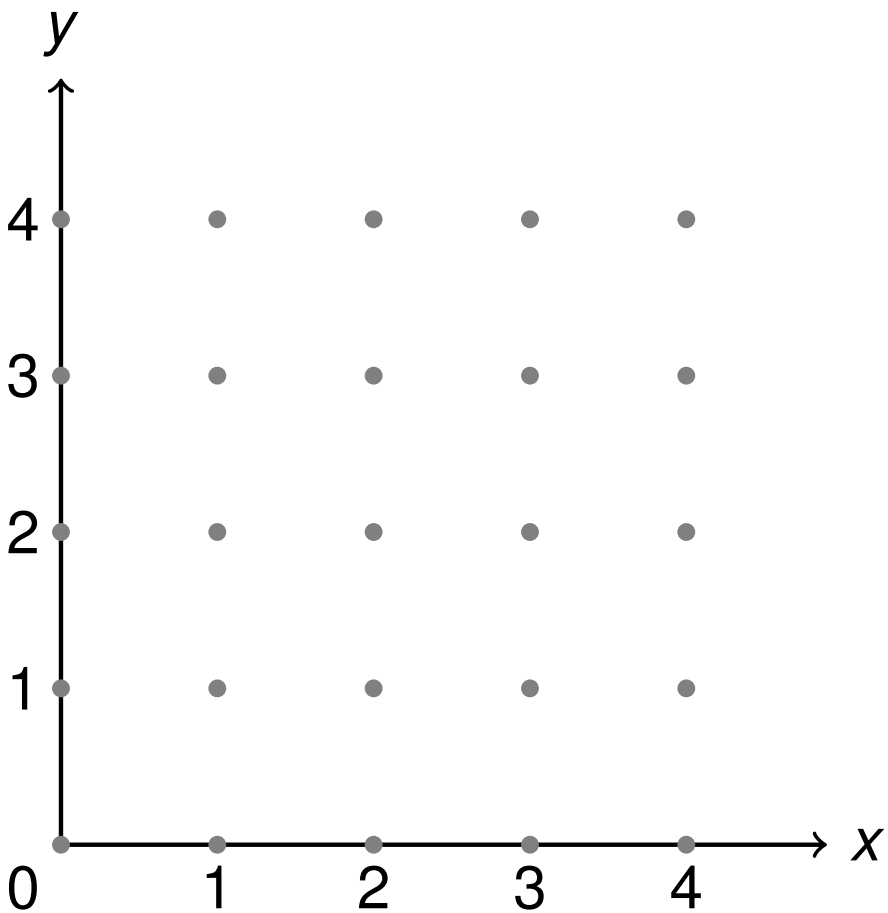
$$\begin{array}{c} \\ s_1 \\ s_2 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{ccccccc} B_1 & B_2 & \dots & B_j & \dots & B_n \end{array} \\ \begin{pmatrix} 1 & 0 & \dots & 1 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & \dots & 1 \end{pmatrix} \end{array} \qquad \begin{array}{cccc} s_1 \in B_1 & \notin B_2 & \in B_j & \notin B_n \\ s_2 \notin B_1 & \in B_2 & \notin B_j & \notin B_n \\ \vdots & \vdots & \vdots & \vdots \\ s_m \notin B_1 & \notin B_2 & \in B_j & \in B_n \end{array}$$

Example



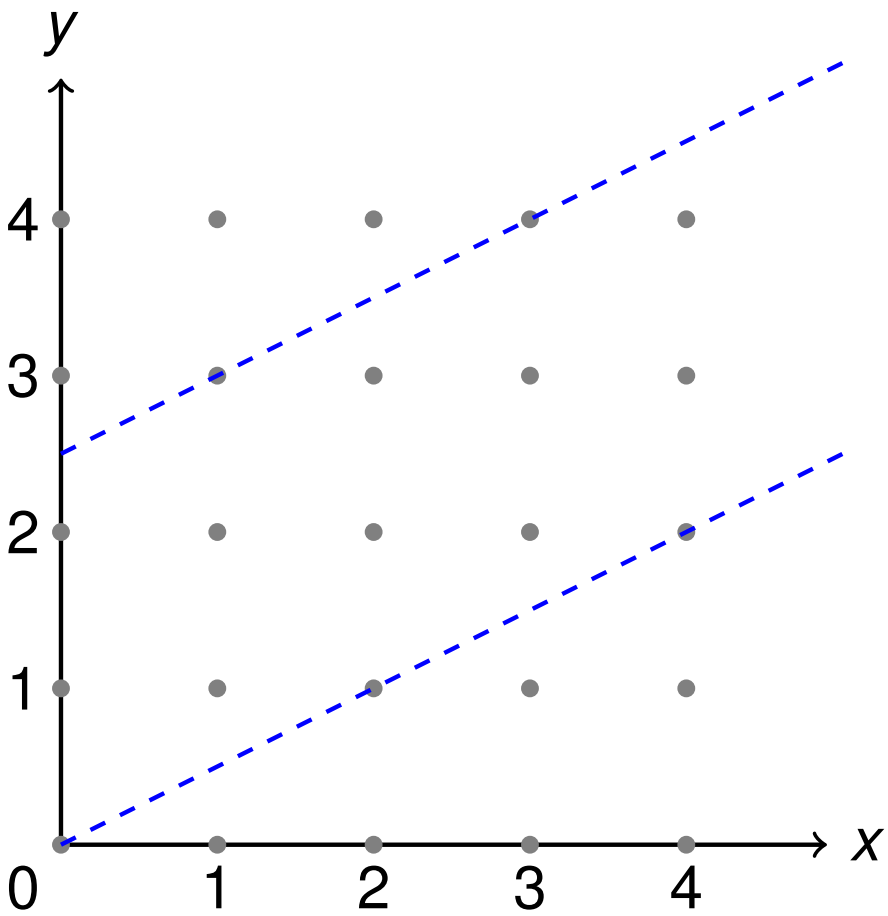
$\mathcal{S} = \mathbb{F}_5^2,$

Example



$\mathcal{S} = \mathbb{F}_5^2, \qquad B_i = \{(x, f(x)) : f \text{ poly over } \mathbb{F}_5 \text{ and } \deg(f) \leq 2\}$

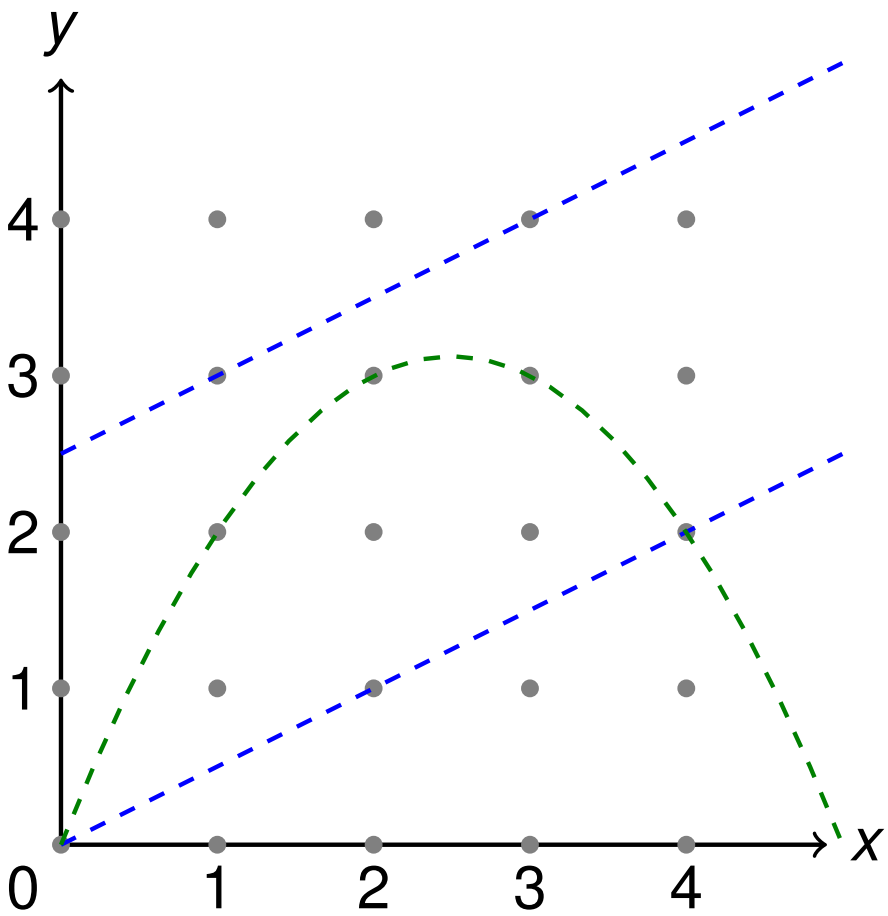
Example



$$f_1(x) = \tfrac{1}{2}x \;\hat{=}\; 3x$$

$$\mathcal{S} = \mathbb{F}_5^2, \qquad B_i = \{(x, f(x)) : f \text{ poly over } \mathbb{F}_5 \text{ and } \deg(f) \leq 2\}$$

Example



$$f_1(x) = \tfrac{1}{2}x \;\hat{=}\; 3x$$

$$f_2(x) = -\tfrac{1}{2}(x-2.5)^2 + 3\tfrac{1}{8} \;\hat{=}\; 2x^2$$

$$\mathcal{S} = \mathbb{F}_5^2, \qquad B_i = \{(x, f(x)) \colon f \text{ poly over } \mathbb{F}_5 \text{ and } \deg(f) \leq 2\}$$

Our Approach

Polynomial-Based Construction of a d-CFF

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$
- distinct $B_f, B_{f_1}, \ldots, B_{f_d} \in \mathcal{B}$:

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$
- distinct $B_f, B_{f_1}, \ldots, B_{f_d} \in \mathcal{B} : |B_f \cap B_{f_i}| \leq k$

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$
- distinct $B_f, B_{f_1}, \ldots, B_{f_d} \in \mathcal{B} : |B_f \cap B_{f_i}| \leq k$
  $\Longrightarrow \left|B_f \setminus \bigcup_{i=1}^{d} B_{f_i}\right| \geq q - d \cdot k$

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$
- distinct $B_f, B_{f_1}, \ldots, B_{f_d} \in \mathcal{B} : |B_f \cap B_{f_i}| \leq k$
  $\Longrightarrow \left|B_f \setminus \bigcup_{i=1}^{d} B_{f_i}\right| \geq q - d \cdot k \geq 1$

Our Approach

Polynomial-Based Construction of a d-CFF

- $\mathcal{S} := \mathbb{F}_q^2 = \{(x_i, x_j) : i, j = 1, \ldots, q\}$
- $f \in \mathbb{F}_q[X]_{\leq k}$, $\deg(f)$ at most k
- $B_f = \{(x_1, f(x_1)), \ldots, (x_q, f(x_q))\} \subset \mathcal{S}$ of size q
- $\Longrightarrow \mathcal{B} := \{B_f : f \in \mathbb{F}_q[X]_{\leq k}\}$
- distinct $B_f, B_{f_1}, \ldots, B_{f_d} \in \mathcal{B} : |B_f \cap B_{f_i}| \leq k$
  $\Longrightarrow \left|B_f \setminus \bigcup_{i=1}^{d} B_{f_i}\right| \geq q - d \cdot k \geq 1 \Leftrightarrow q \geq d \cdot k + 1$

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Proof (Sketch):

- given (pk_i, m_i) for $i = 1, \ldots, n$ with d faults and τ

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Proof (Sketch):

- given (pk_i, m_i) for $i = 1, \ldots, n$ with d faults and τ
- Verify must output a $(n - d)$-subset of $\{(\mathrm{pk}_i, m_i)\}$

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Proof (Sketch):

- given (pk_i, m_i) for $i = 1, \ldots, n$ with d faults and τ
- Verify must output a $(n-d)$-subset of $\{(\mathrm{pk}_i, m_i)\}$
- there are $\binom{n}{n-d}$ such subsets

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Proof (Sketch):

- given (pk_i, m_i) for $i = 1, \ldots, n$ with d faults and τ
- Verify must output a $(n-d)$-subset of $\{(\mathrm{pk}_i, m_i)\}$
- there are $\binom{n}{n-d}$ such subsets
- based on a signature of length ℓ, Verify can distinguish up to 2^ℓ cases

Compression Ratio

Lemma (Informal)

For a fixed number of faults d, the size of a fault-tolerant signature is at least $\Omega(\log_2(n))$.

Proof (Sketch):

- given (pk_i, m_i) for $i = 1, \ldots, n$ with d faults and τ
- Verify must output a $(n-d)$-subset of $\{(\mathrm{pk}_i, m_i)\}$
- there are $\binom{n}{n-d}$ such subsets
- based on a signature of length ℓ, Verify can distinguish up to 2^ℓ cases
- required: $2^\ell \geq \binom{n}{n-d} \underset{\text{math}}{\rightsquigarrow} \ell \in \Omega(\log_2(n))$ $\square$

References I

R. Kumar, S. Rajagopalan, and A. Sahai. “Coding Constructions for Blacklisting Problems without Computational Assumptions”. In: *CRYPTO ’99*. Ed. by M. J. Wiener. Vol. 1666. LNCS. Springer, Aug. 1999, pp. 609–623.

W. H. Kautz and R. C. Singleton. “Nonrandom binary superimposed codes”. In: *IEEE Transactions on Information Theory* 10.4 (1964), pp. 363–377. DOI: 10.1109/TIT.1964.1053689.