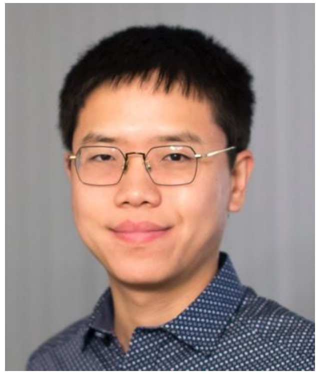
Adaptively Secure BLS Threshold Signatures from DDH and co-CDH



Sourav Das

Ling Ren

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

souravd2@Illinois.edu

Boneh-Lynn-Sacham (BLS) Signatures

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\mathrm{pk} \coloneqq g^s$$

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\mathrm{pk} \coloneqq g^s$$

Signing

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} := s \leftarrow \mathbb{F}$$
$$\mathrm{pk} := g^s$$

Signing

$$\sigma := \mathrm{H}(m)^s$$

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$
$\mathrm{pk} \coloneqq g^s$

Signing

$\sigma \coloneqq \mathrm{H}(m)^s$

Verification

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\mathrm{pk} \coloneqq g^s$$

Signing

$$\sigma \coloneqq \mathrm{H}(m)^s$$

Verification

$$e\big(\mathrm{pk}, \mathrm{H}(m)\big) = e(g, \sigma)$$

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\mathrm{pk} \coloneqq g^s$$

Signing

$$\sigma \coloneqq \mathrm{H}(m)^s$$

Verification

$$e\big(\mathrm{pk}, \mathrm{H}(m)\big) = e(g, \sigma)$$

Correctness:

- LHS: $e\big(\mathrm{pk}, \mathrm{H}(m)\big) = e\big(g, \mathrm{H}(m)\big)^s$
- RHS: $e(g, \sigma) = e\big(g, \mathrm{H}(m)\big)^s$

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\mathrm{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\mathrm{pk} \coloneqq g^s$$

Signing

$$\sigma \coloneqq \mathrm{H}(m)^s$$

Verification

$$e\big(\mathrm{pk}, \mathrm{H}(m)\big) = e(g, \sigma)$$

Correctness:

- LHS: $e\big(\mathrm{pk}, \mathrm{H}(m)\big) = e\big(g, \mathrm{H}(m)\big)^s$
- RHS: $e(g, \sigma) = e\big(g, \mathrm{H}(m)\big)^s$

Security:

Boneh-Lynn-Sacham (BLS) Signatures

Bilinear pairing based signature scheme

Key generation

$$\text{sk} \coloneqq s \leftarrow \mathbb{F}$$
$$\text{pk} \coloneqq g^s$$

Signing

$$\sigma \coloneqq \mathrm{H}(m)^s$$

Verification

$$e\big(\text{pk}, \mathrm{H}(m)\big) = e(g, \sigma)$$

Correctness:

- LHS: $e\big(\text{pk}, \mathrm{H}(m)\big) = e\big(g, \mathrm{H}(m)\big)^s$
- RHS: $e(g, \sigma) = e\big(g, \mathrm{H}(m)\big)^s$

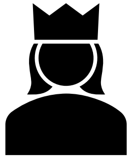
Security:

- Hardness of CDH in the random oracle model (ROM)

Background

(n, t) Threshold Secret Sharing

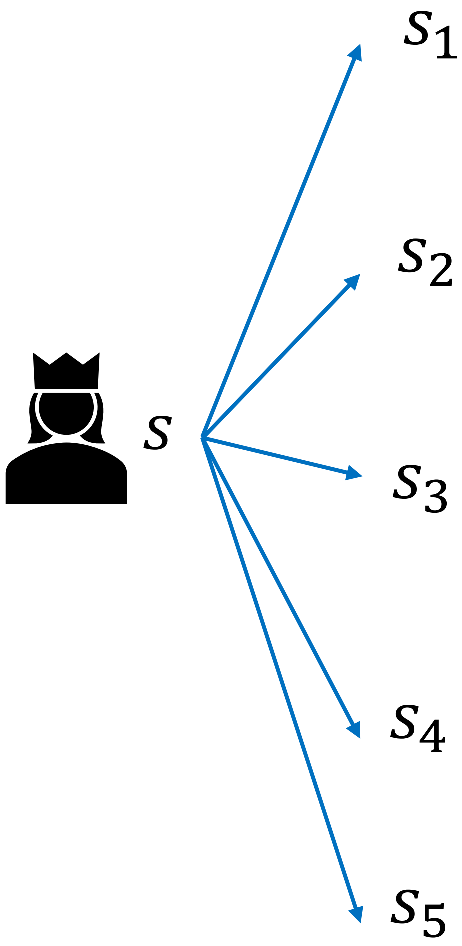
(n, t) Threshold Secret Sharing



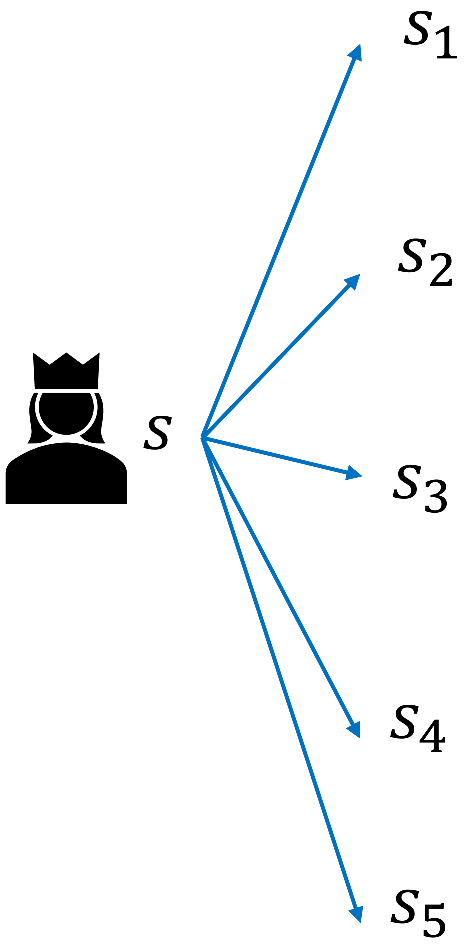
s

(n, t) Threshold Secret Sharing

- A mechanism to share a secret s into n shares

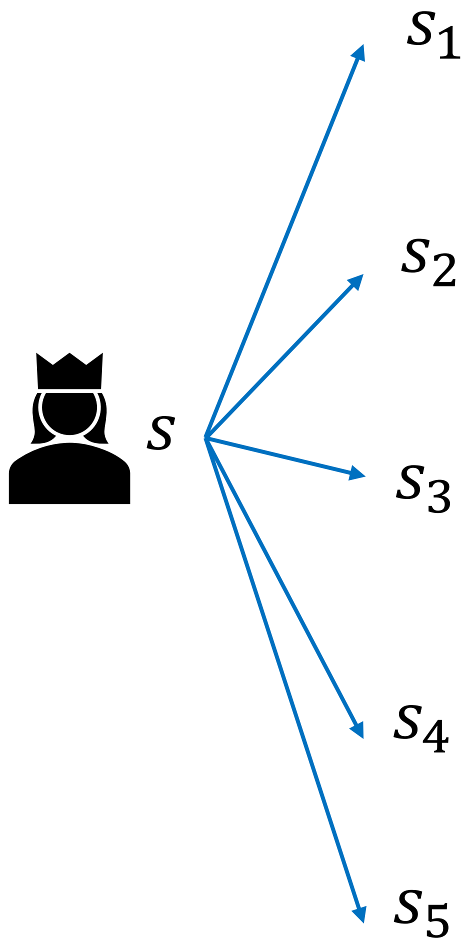


(n, t) Threshold Secret Sharing



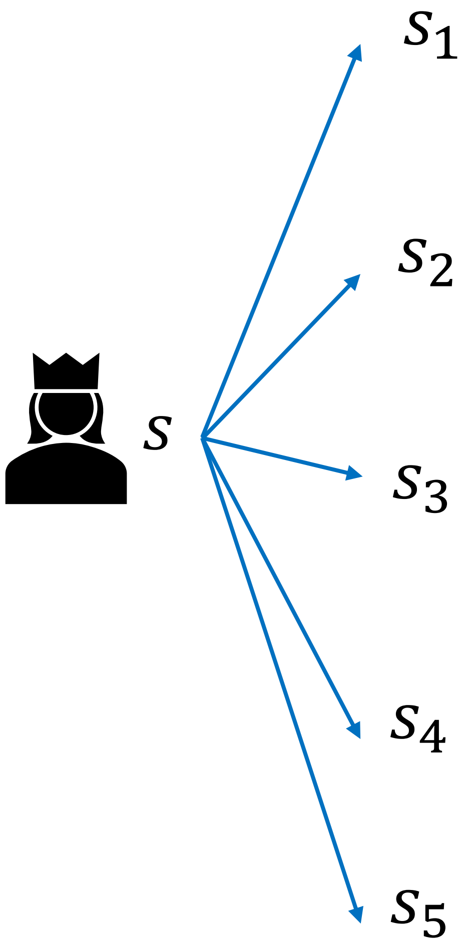
- A mechanism to share a secret s into n shares
- Any subset of $t + 1$ shares reveal the secret

(n, t) Threshold Secret Sharing



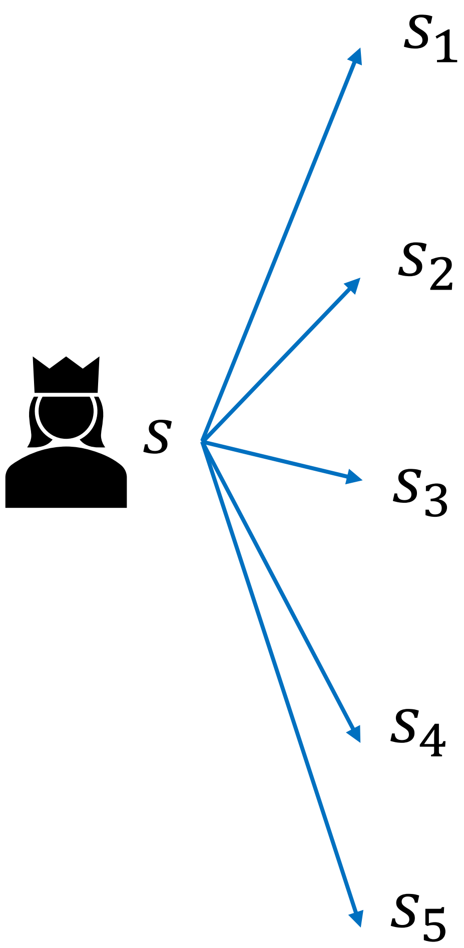
- A mechanism to share a secret s into n shares
- Any subset of $t + 1$ shares reveal the secret
- Any subset of t or less shares reveal nothing about s

(n, t) Threshold Secret Sharing

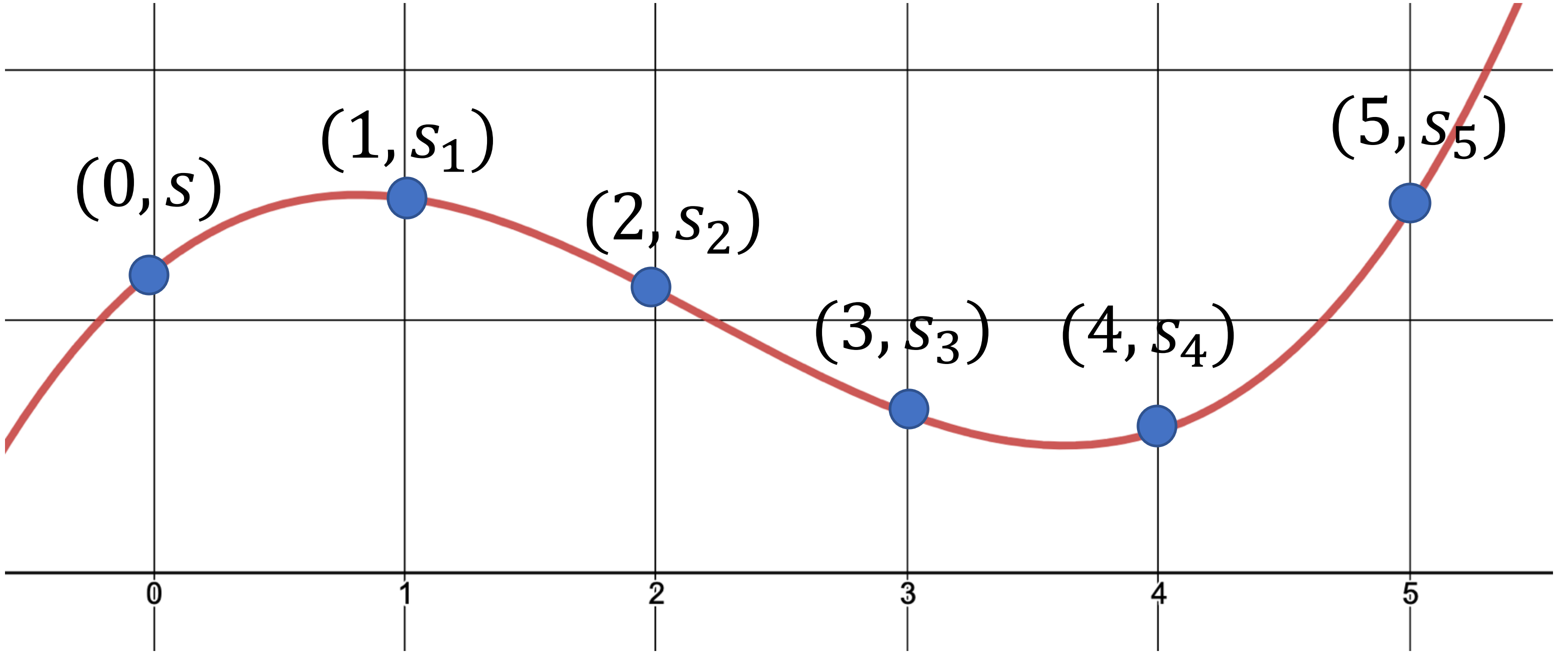


- A mechanism to share a secret s into n shares
- Any subset of $t + 1$ shares reveal the secret
- Any subset of t or less shares reveal nothing about s
- An example of $(5,3)$ threshold secret sharing scheme

(n, t) Threshold Secret Sharing



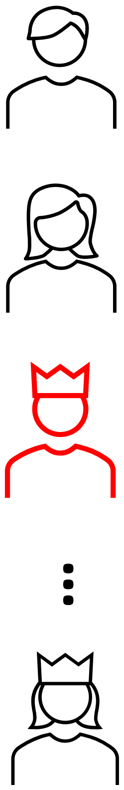
- A mechanism to share a secret s into n shares
- Any subset of $t + 1$ shares reveal the secret
- Any subset of t or less shares reveal nothing about s
- An example of $(5,3)$ threshold secret sharing scheme



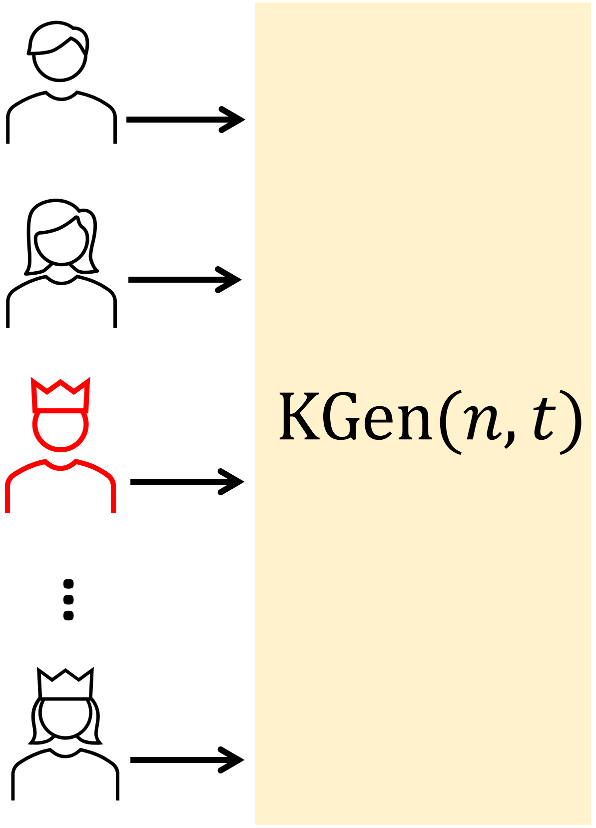
BLS Threshold signature [Boldyreva’03]

BLS Threshold signature [Boldyreva’03]: Key Generation

BLS Threshold signature [Boldyreva’03]: Key Generation

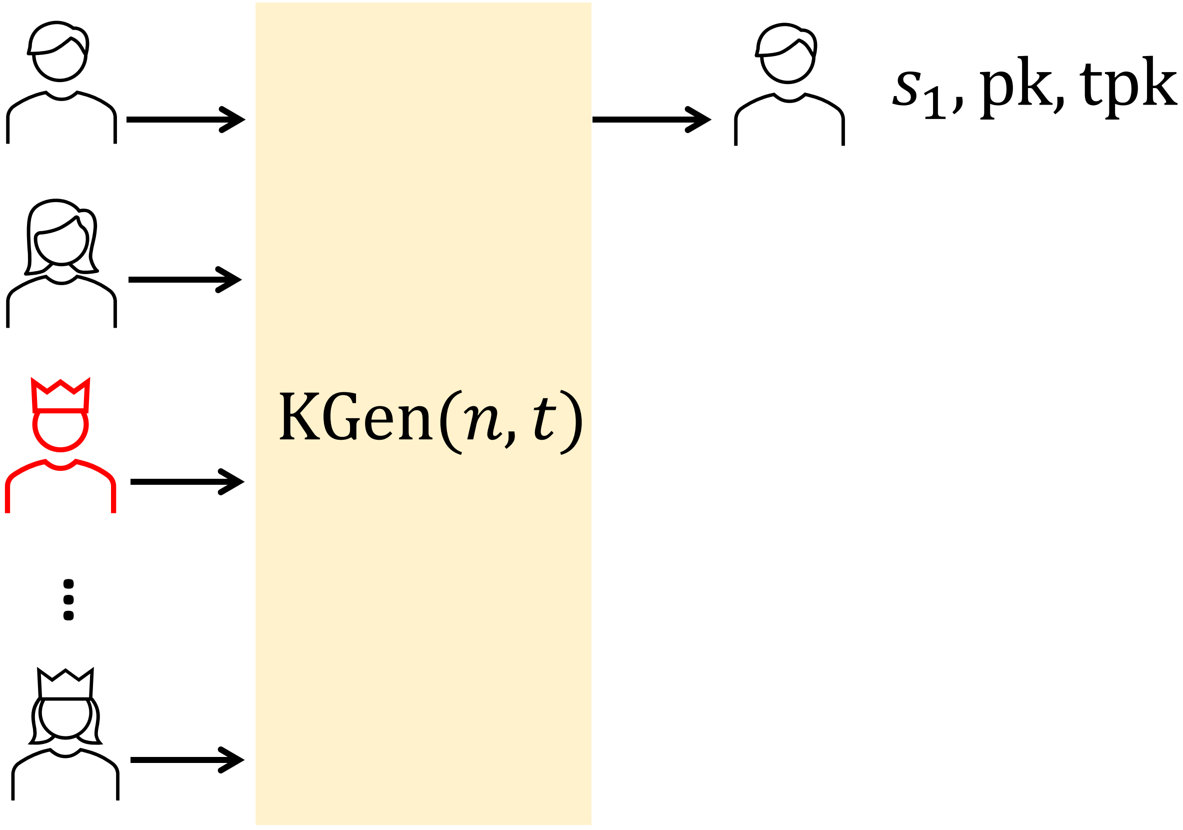


BLS Threshold signature [Boldyreva'03]: Key Generation



$\mathrm{KGen}(n, t)$

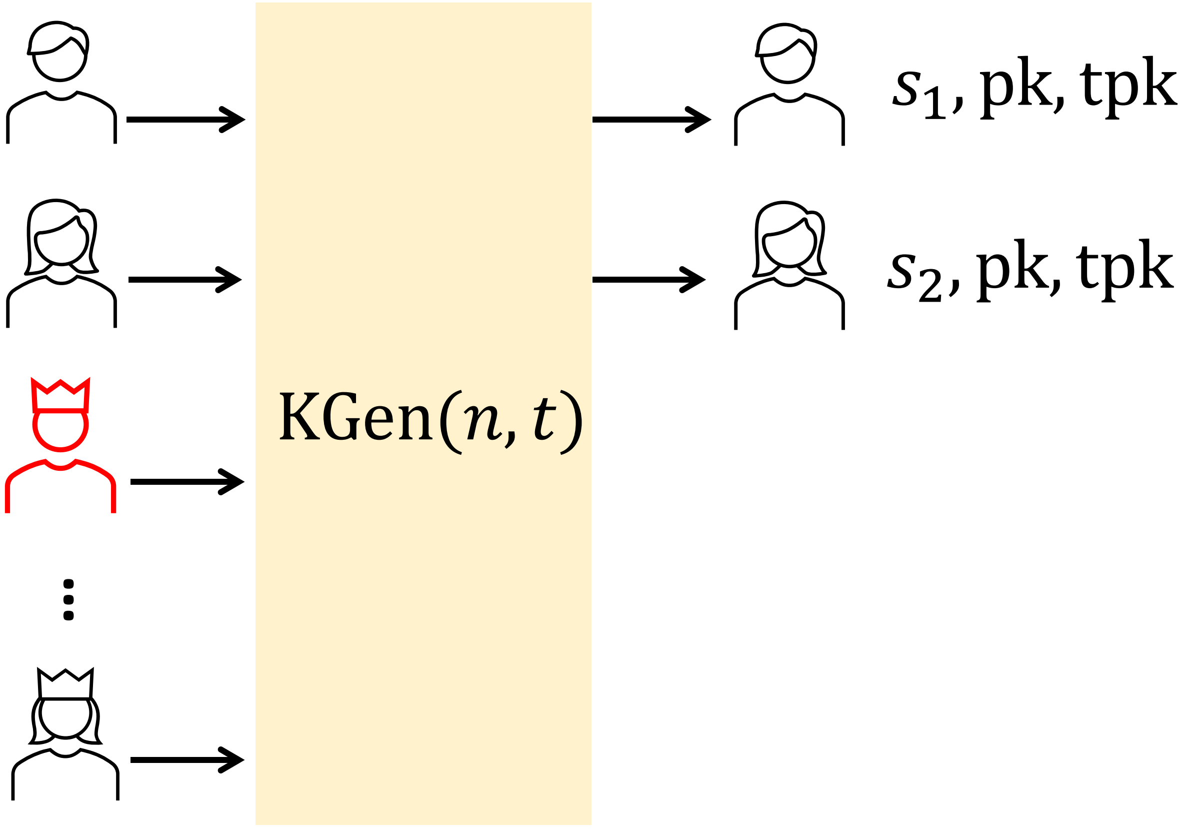
BLS Threshold signature [Boldyreva'03]: Key Generation



$\mathrm{KGen}(n, t)$

$s_1, \mathrm{pk}, \mathrm{tpk}$

BLS Threshold signature [Boldyreva’03]: Key Generation

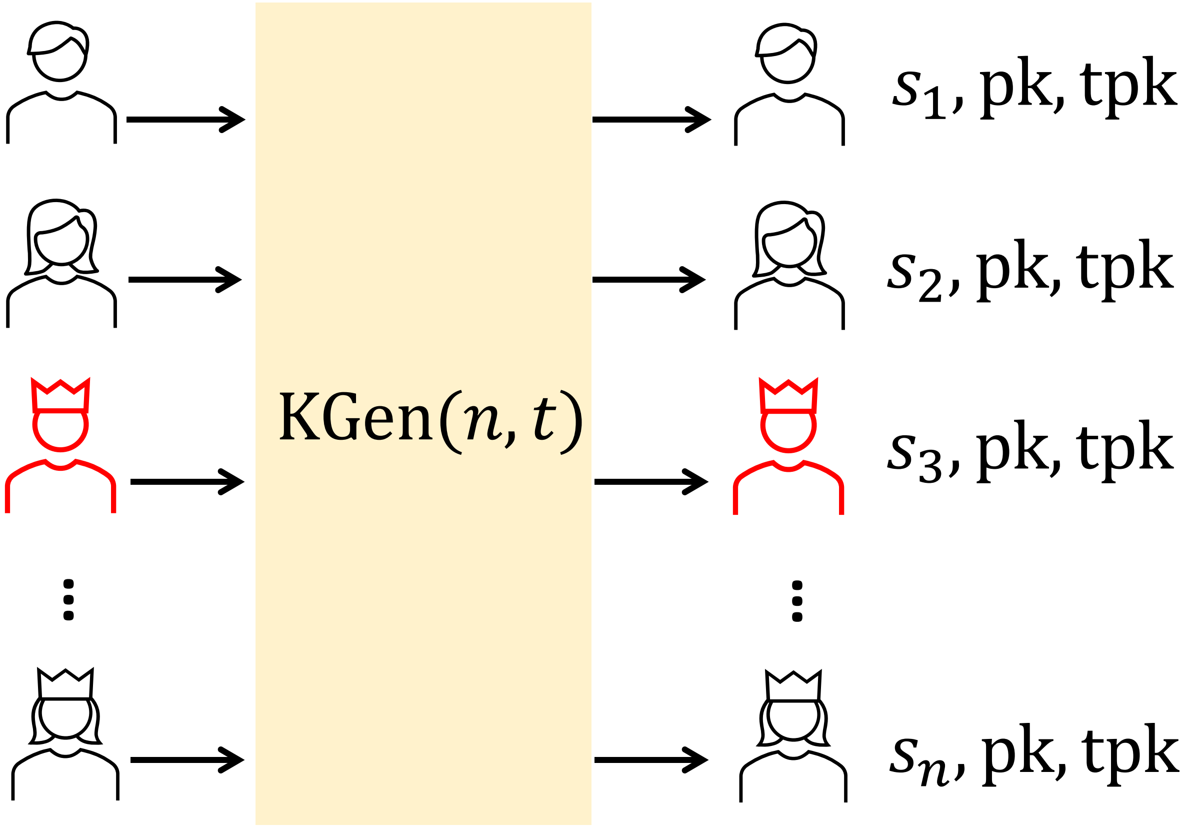


$\mathrm{KGen}(n, t)$

$s_1, \mathrm{pk}, \mathrm{tpk}$

$s_2, \mathrm{pk}, \mathrm{tpk}$

BLS Threshold signature [Boldyreva'03]: Key Generation



$\mathrm{KGen}(n, t)$

$s_1, \mathrm{pk}, \mathrm{tpk}$

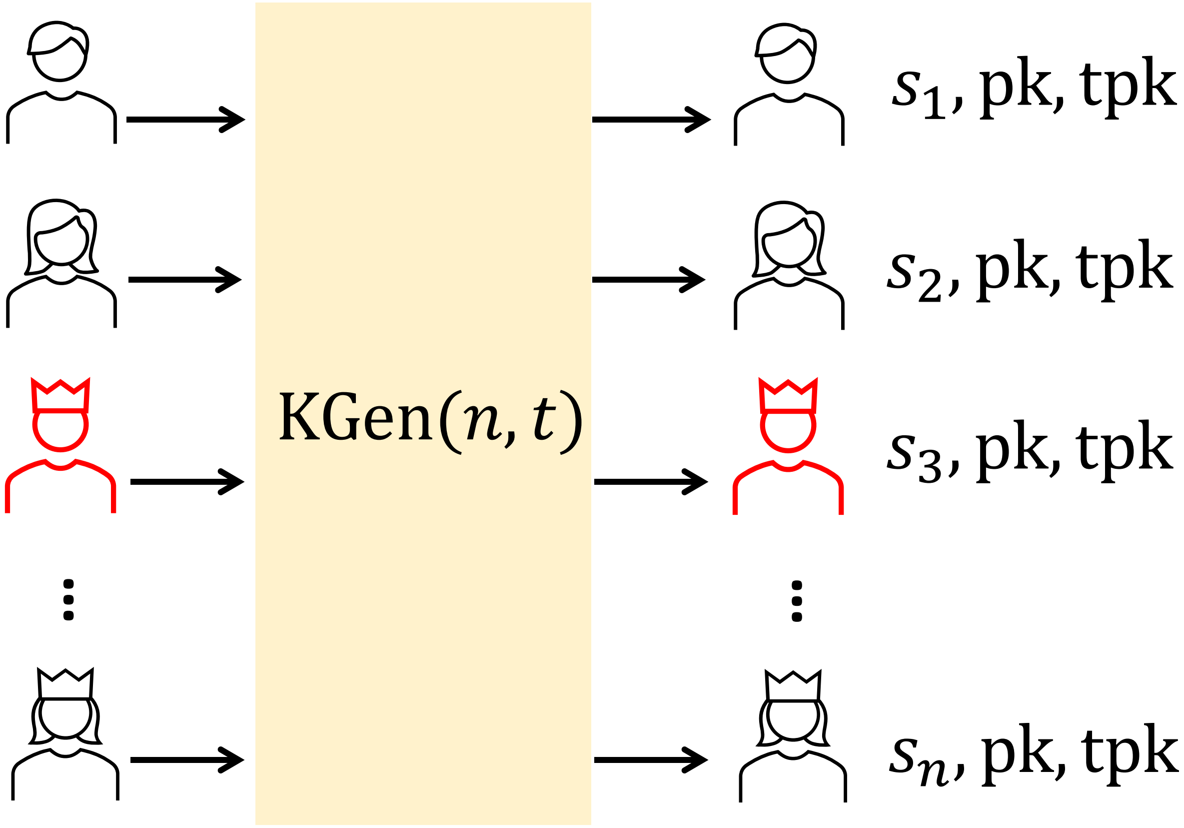
$s_2, \mathrm{pk}, \mathrm{tpk}$

$s_3, \mathrm{pk}, \mathrm{tpk}$

⋮

$s_n, \mathrm{pk}, \mathrm{tpk}$

BLS Threshold signature [Boldyreva'03]: Key Generation



$\mathrm{KGen}(n, t)$

$s_1, \mathrm{pk}, \mathrm{tpk}$

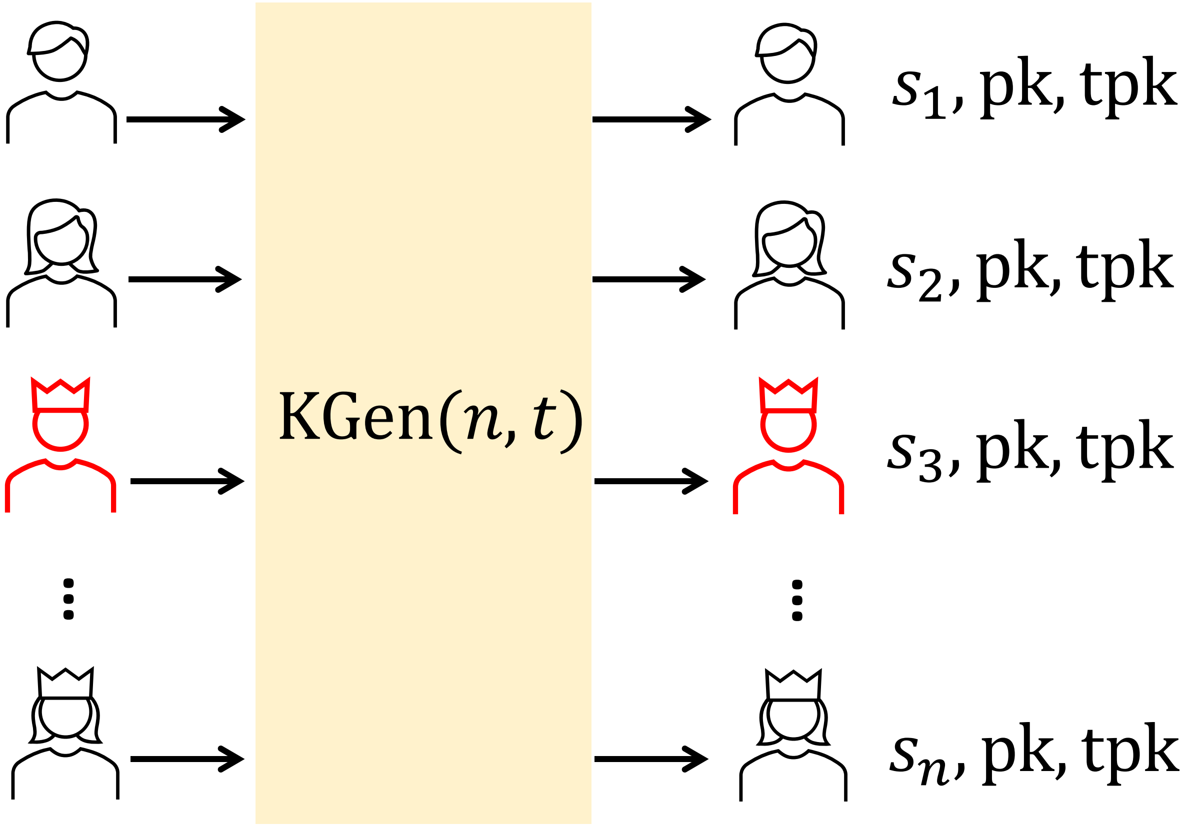
$s_2, \mathrm{pk}, \mathrm{tpk}$

$s_3, \mathrm{pk}, \mathrm{tpk}$

$s_n, \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\mathrm{pk} = g^s$$
$$\mathrm{tpk} = \{g^{s_1}, g^{s_2}, \dots, g^{s_n}\}$$

BLS Threshold signature [Boldyreva'03]: Key Generation



KGen(n, t)

s_1, pk, tpk

s_2, pk, tpk

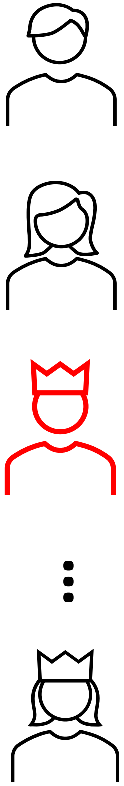
s_3, pk, tpk

s_n, pk, tpk

$$\{s_1, \dots, s_n\} \leftarrow \text{Share}(s)$$
$$\text{pk} = g^s$$
$$\text{tpk} = \{g^{s_1}, g^{s_2}, \dots, g^{s_n}\}$$

Can also use Distributed Key Generation (DKG).
[GJKR-JoC'07, DYXMKR-SP'22, DXKR-USENIX'23]

BLS Threshold signature: Signing



s_1

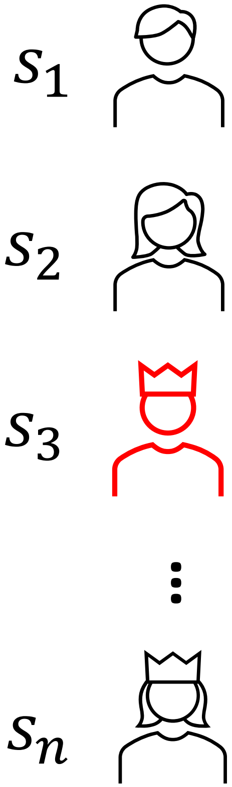
s_2

s_3

⋮

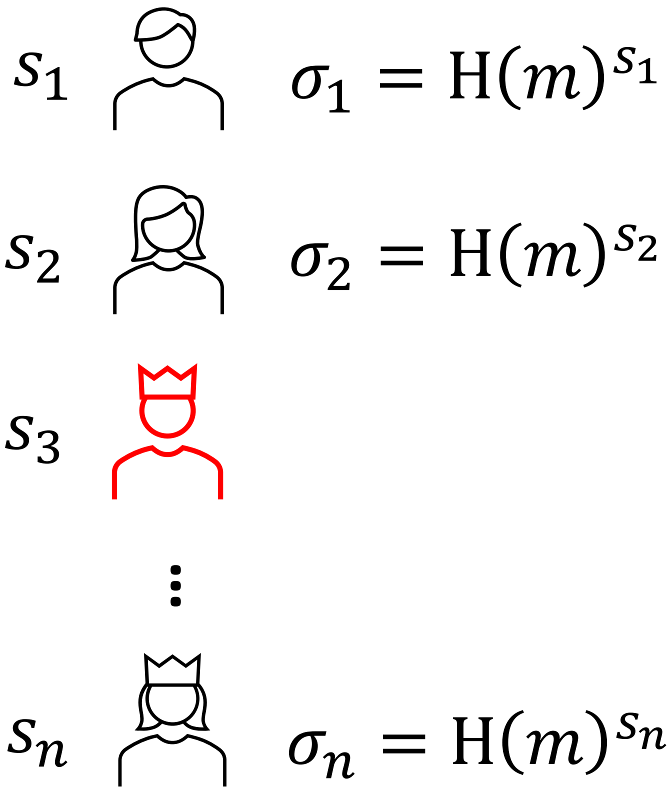
s_n

BLS Threshold signature: Signing



$$\sigma_1 = \mathrm{H}(m)^{s_1}$$

BLS Threshold signature: Signing



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

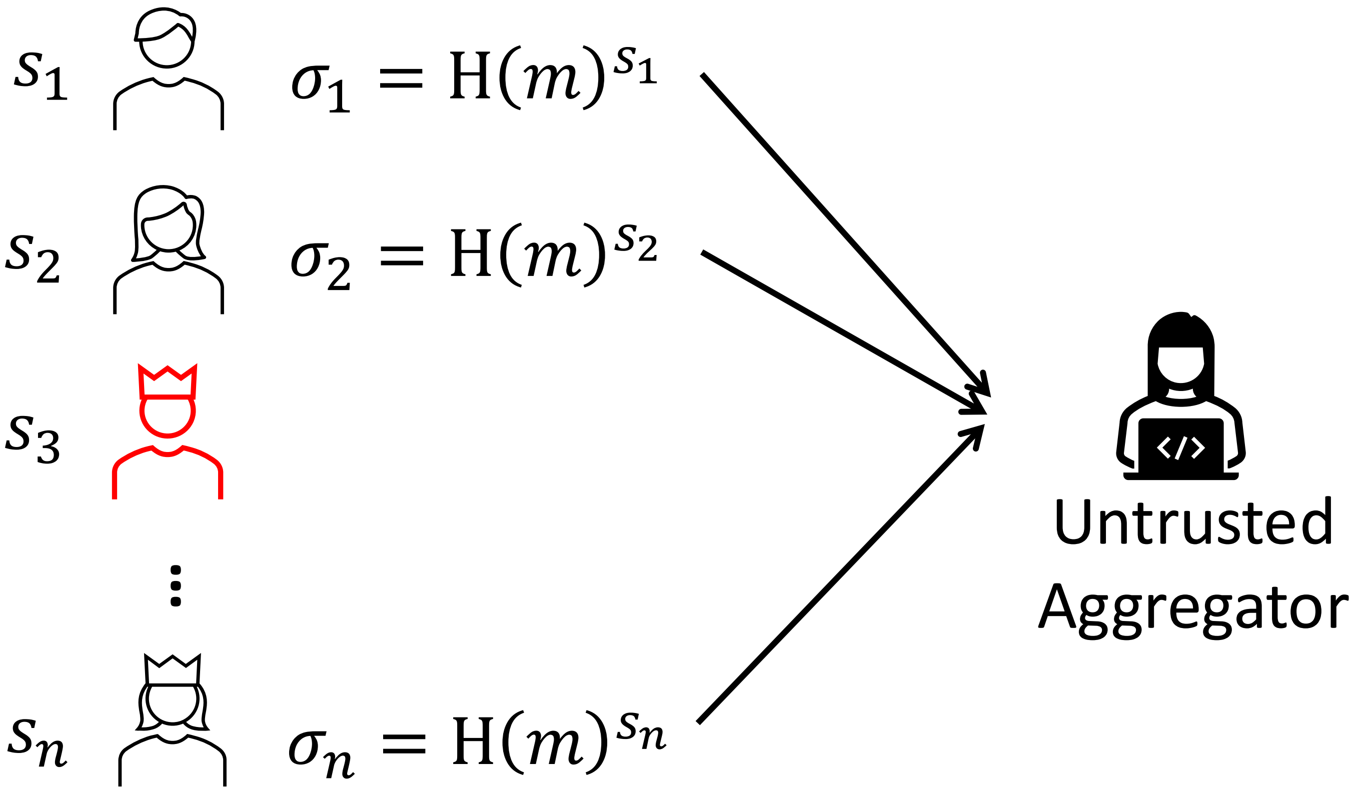
s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

BLS Threshold signature: Signing



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

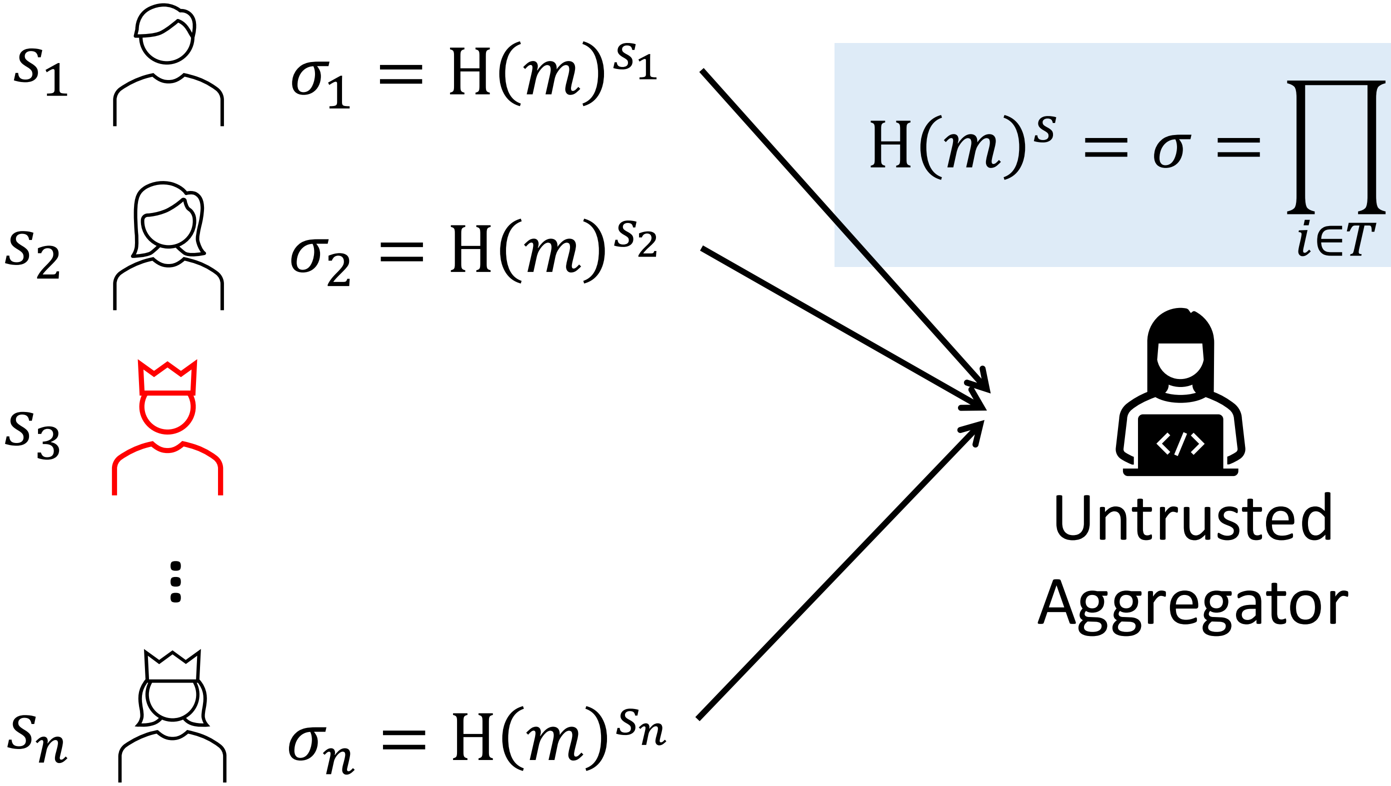
⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

Untrusted Aggregator

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\quad \sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\quad \sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

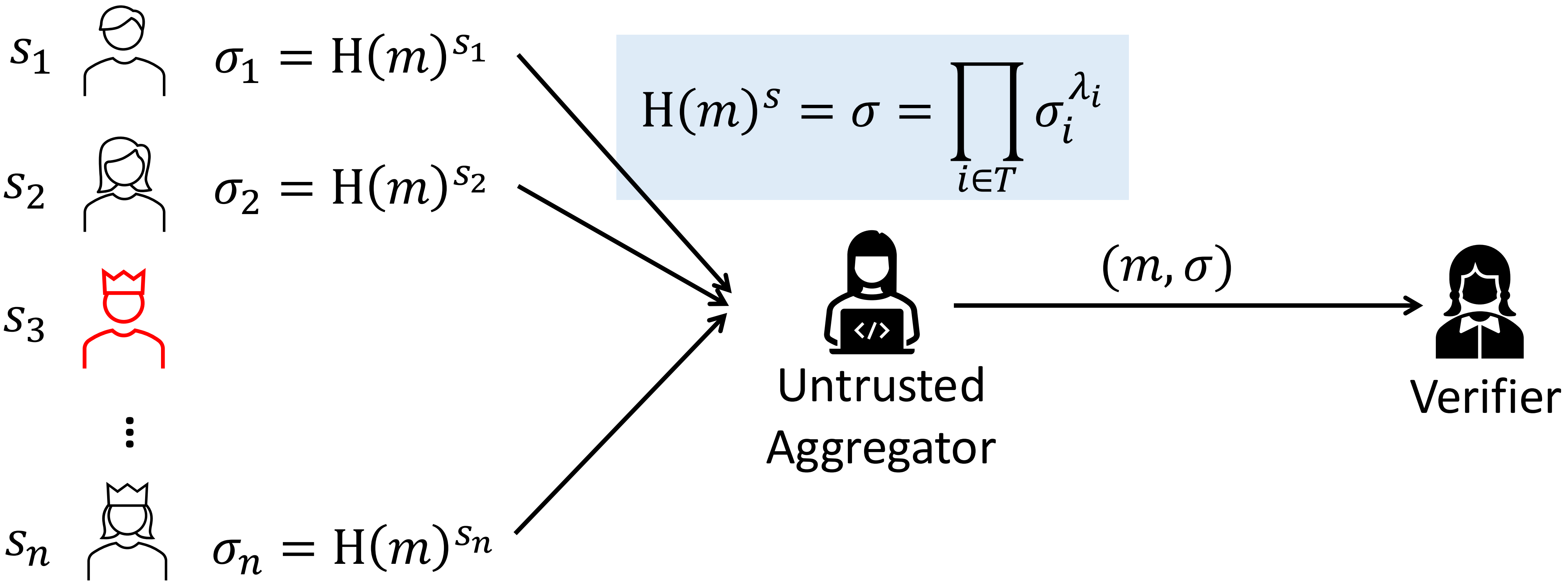
s_n $\quad \sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

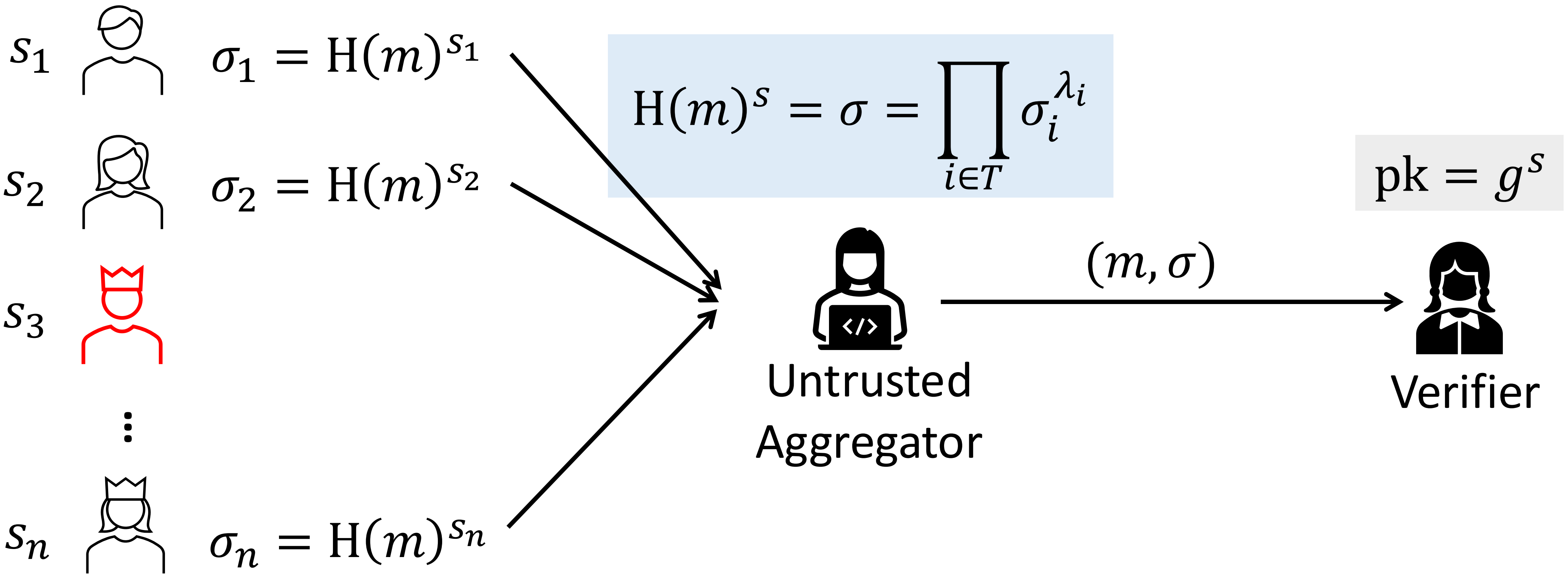
(m, σ)

Untrusted Aggregator

Verifier

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

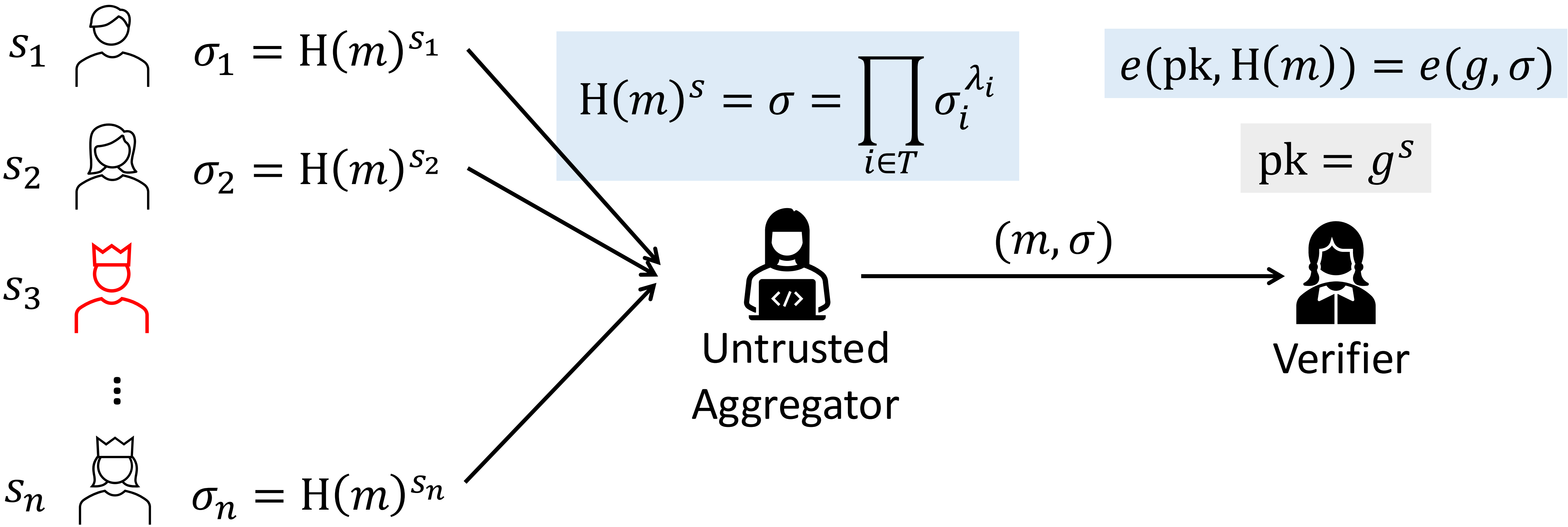
(m, σ)

$\mathrm{pk} = g^s$

Verifier

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

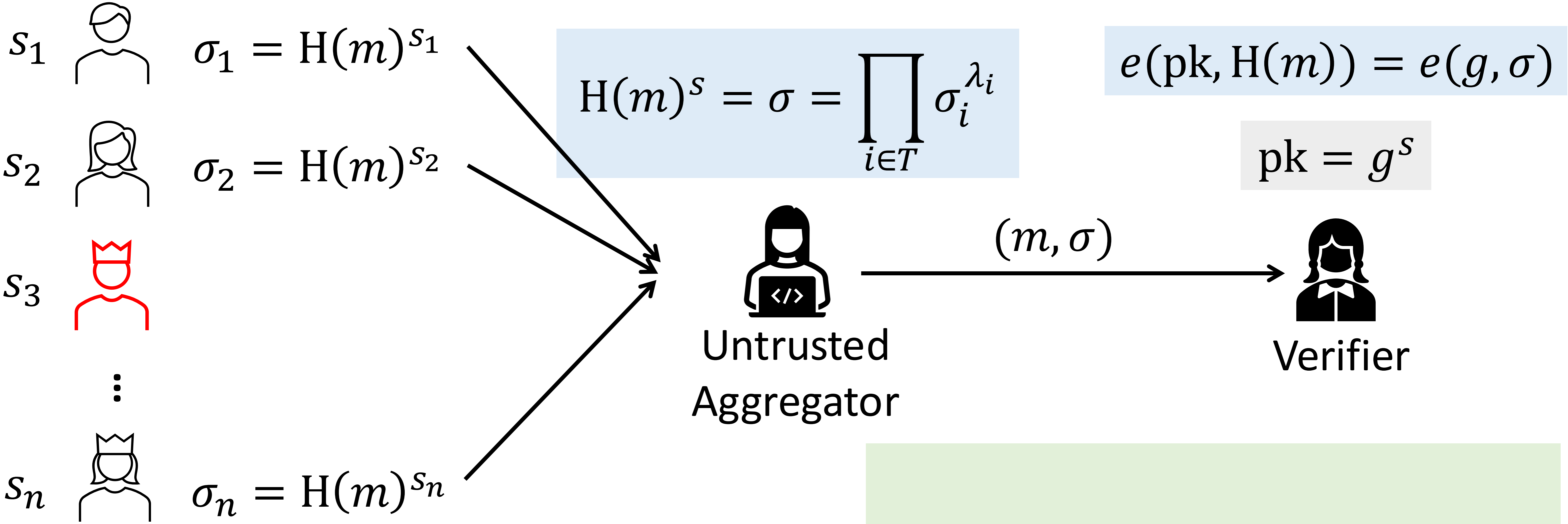
(m, σ)

Untrusted Aggregator

Verifier

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\quad \sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\quad \sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\quad \sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

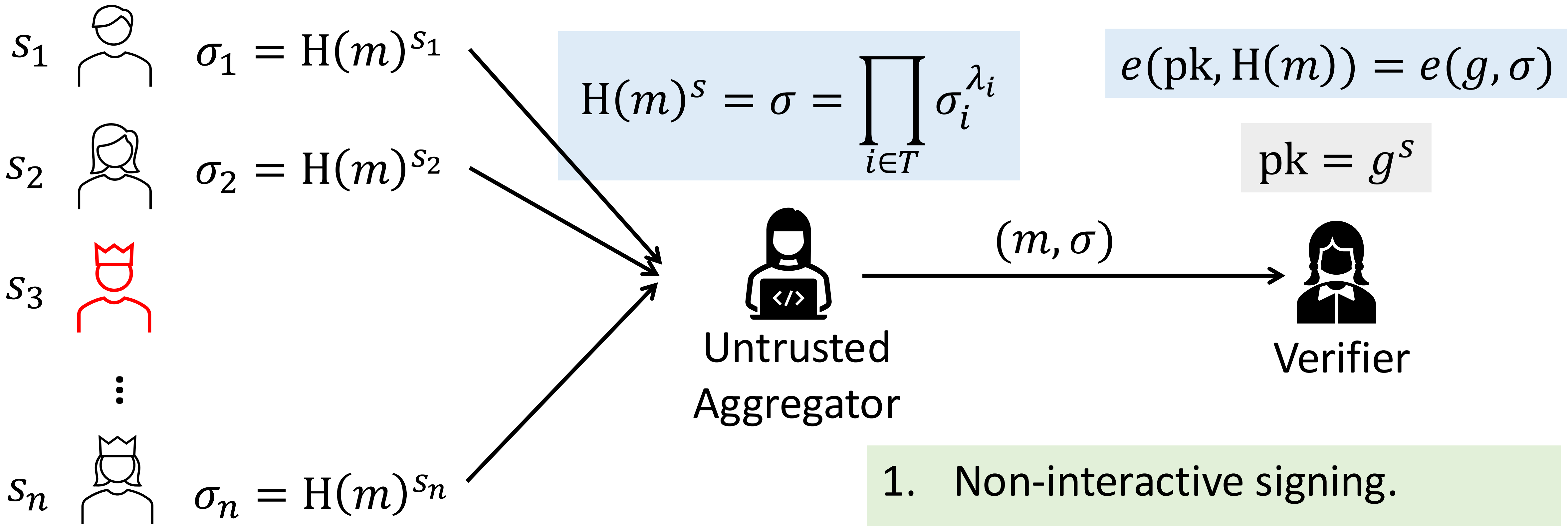
(m, σ)

Untrusted Aggregator

Verifier

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

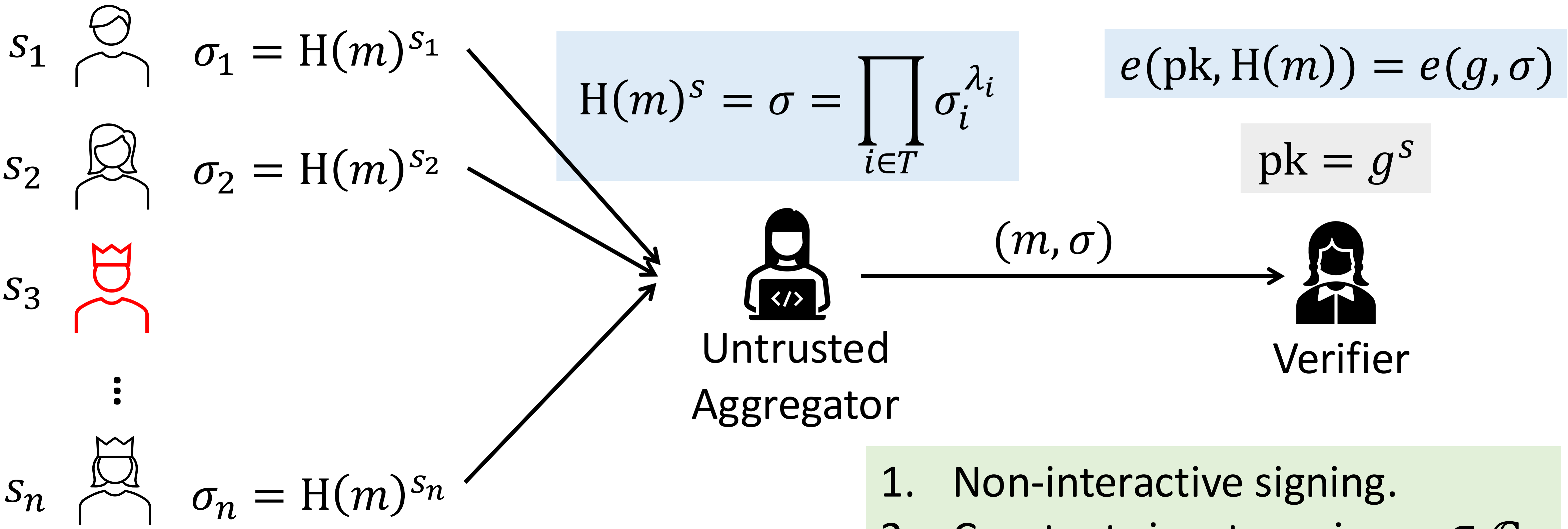
Untrusted Aggregator

Verifier

1. Non-interactive signing.

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

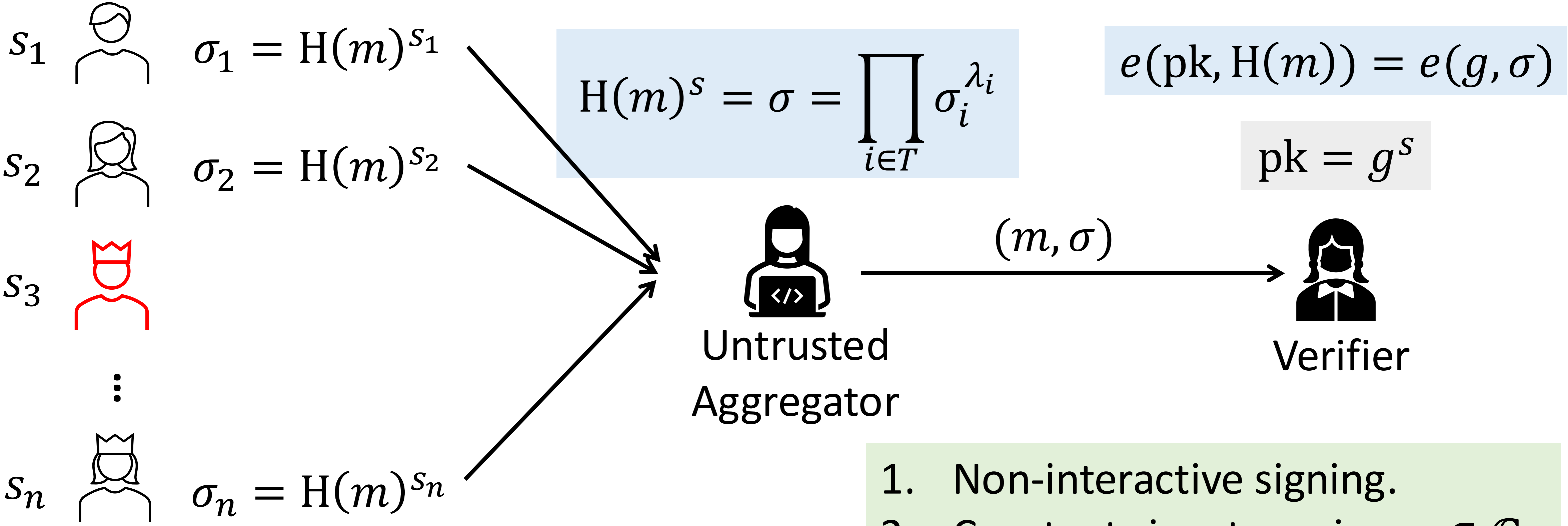
Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.

BLS Threshold signature: Signing

λ_i are the Lagrange coefficients.



s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$\mathrm{pk} = g^s$

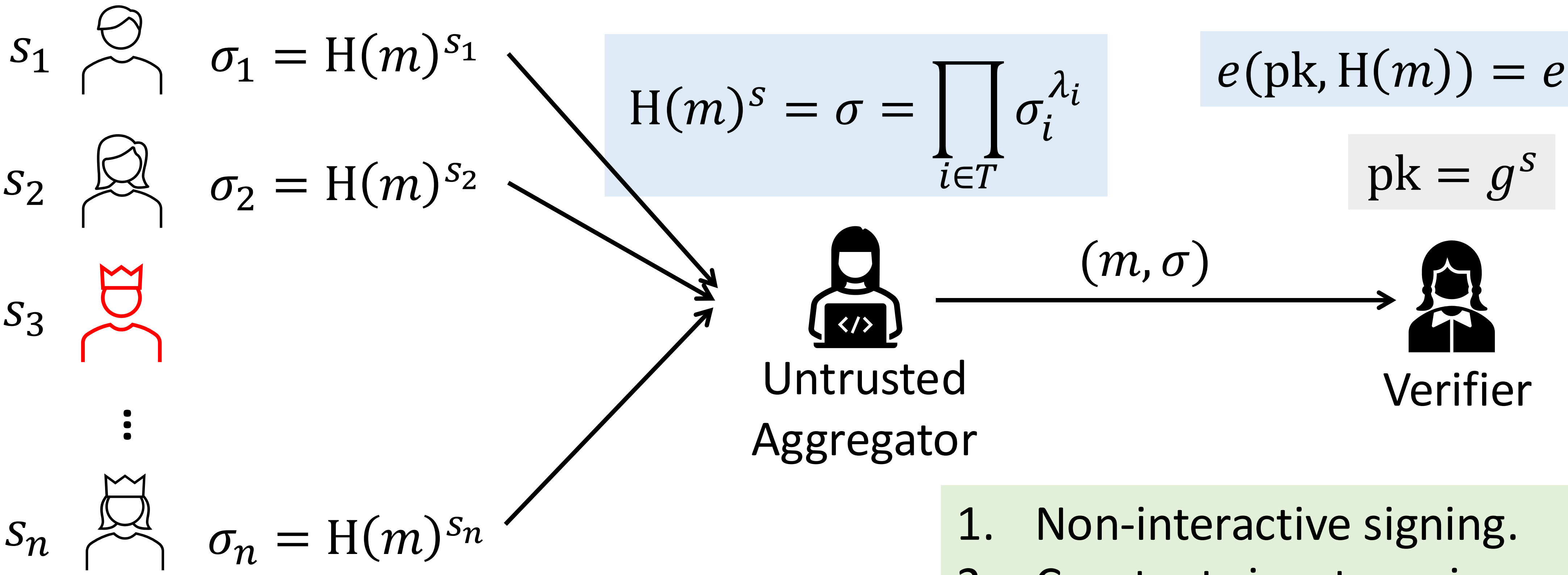
(m, σ)

Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.
3. Verification cost: two pairings.

BLS Threshold signature: Signing



λ_i are the Lagrange coefficients.

s_1 $\sigma_1 = \mathrm{H}(m)^{s_1}$

s_2 $\sigma_2 = \mathrm{H}(m)^{s_2}$

s_3

⋮

s_n $\sigma_n = \mathrm{H}(m)^{s_n}$

$$\mathrm{H}(m)^s = \sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

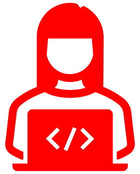
Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.
3. Verification cost: two pairings.
4. Unique signatures.

Static vs Adaptive Security

Static vs Adaptive Security



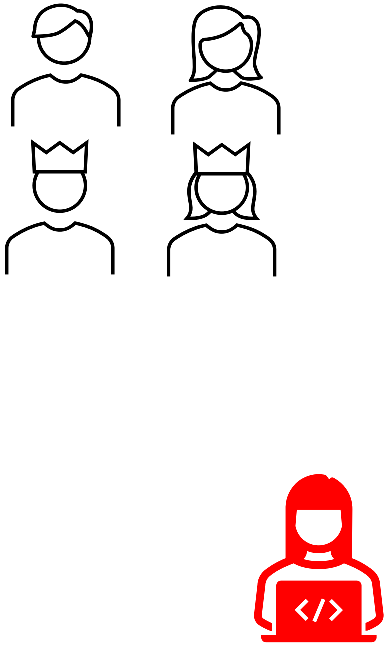
Static $\mathcal{A}_{\mathrm{th}}$

Static vs Adaptive Security

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

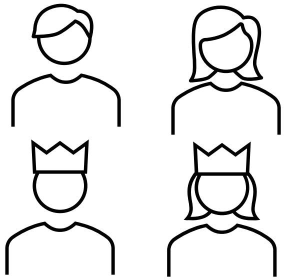
Static vs Adaptive Security

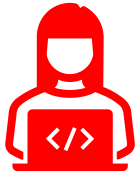


Static $\mathcal{A}_{\mathrm{th}}$

Needs to decide who to corrupt before the protocol begins.

Static vs Adaptive Security



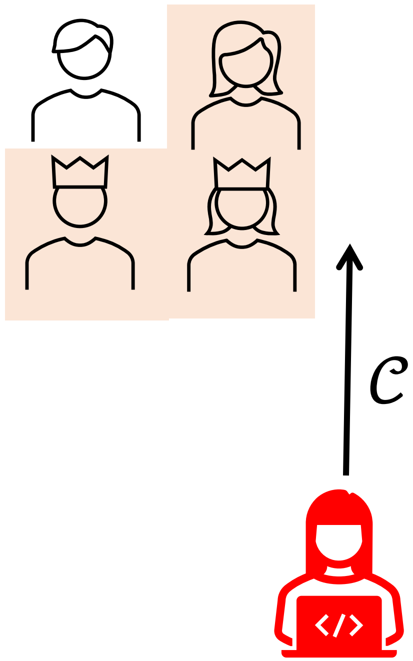


$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

Static vs Adaptive Security



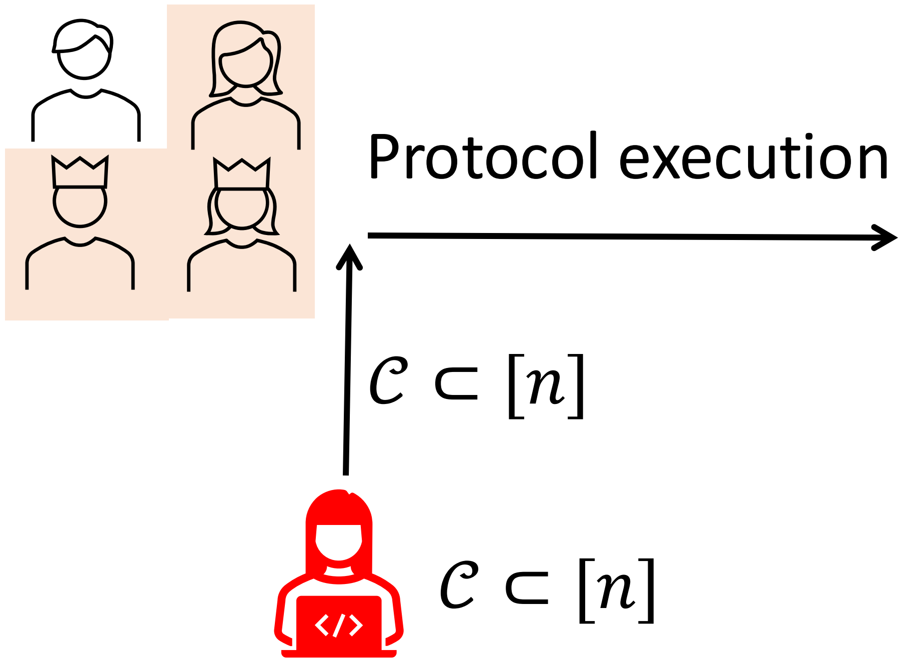
$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

Static vs Adaptive Security



Protocol execution

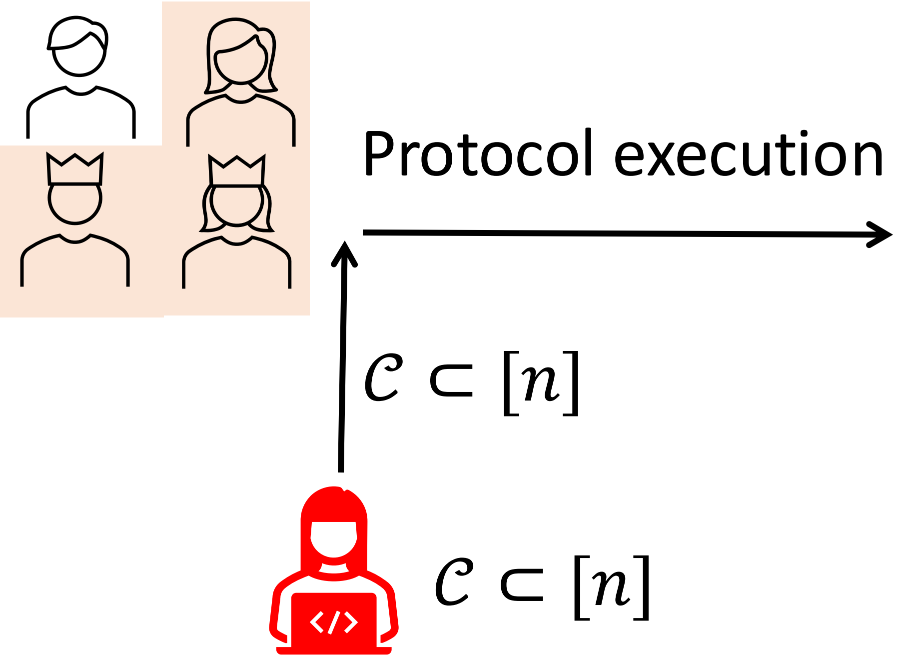
$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

Static vs Adaptive Security

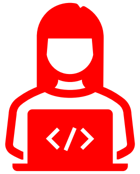


Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

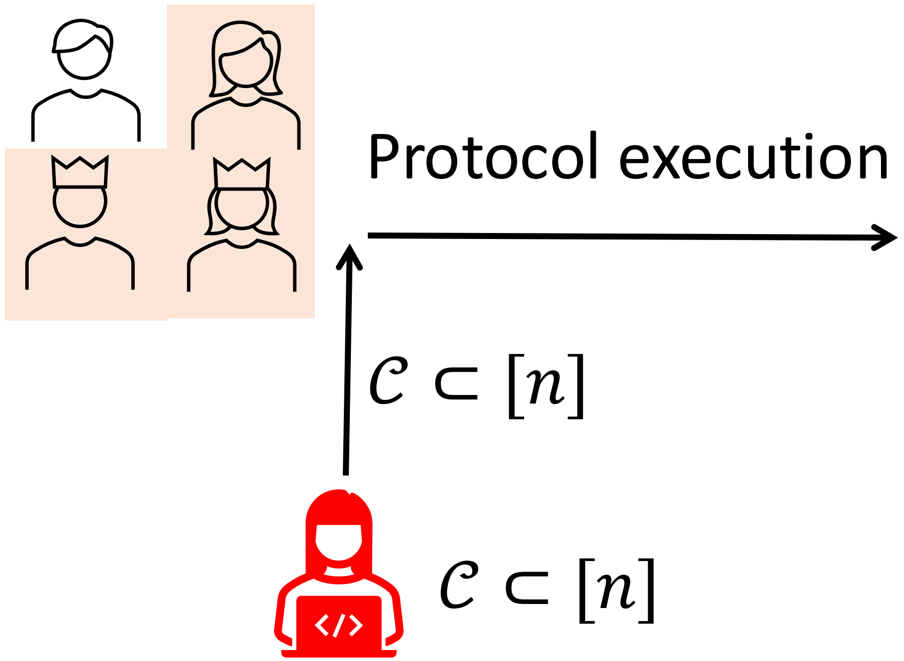
Static $\mathcal{A}_{\text{th}}$



Adaptive $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

Static vs Adaptive Security



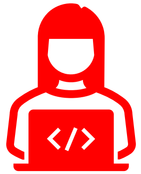
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

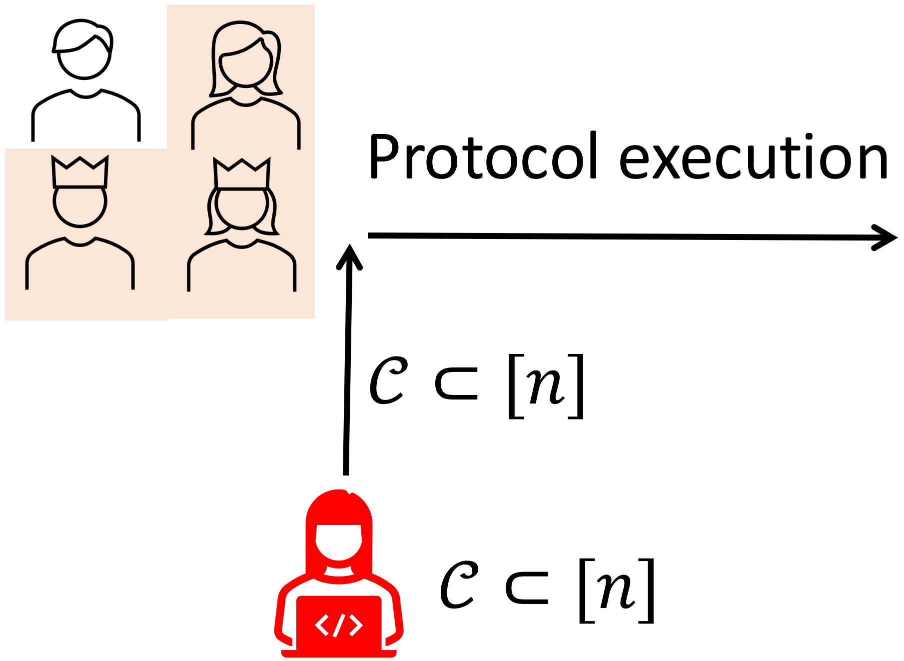
Needs to decide who to corrupt before the protocol begins.



Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



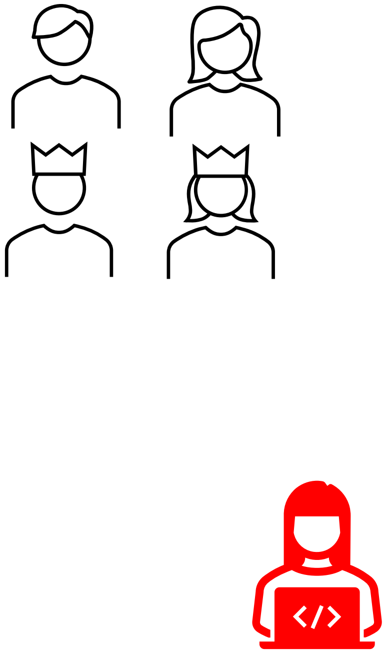
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

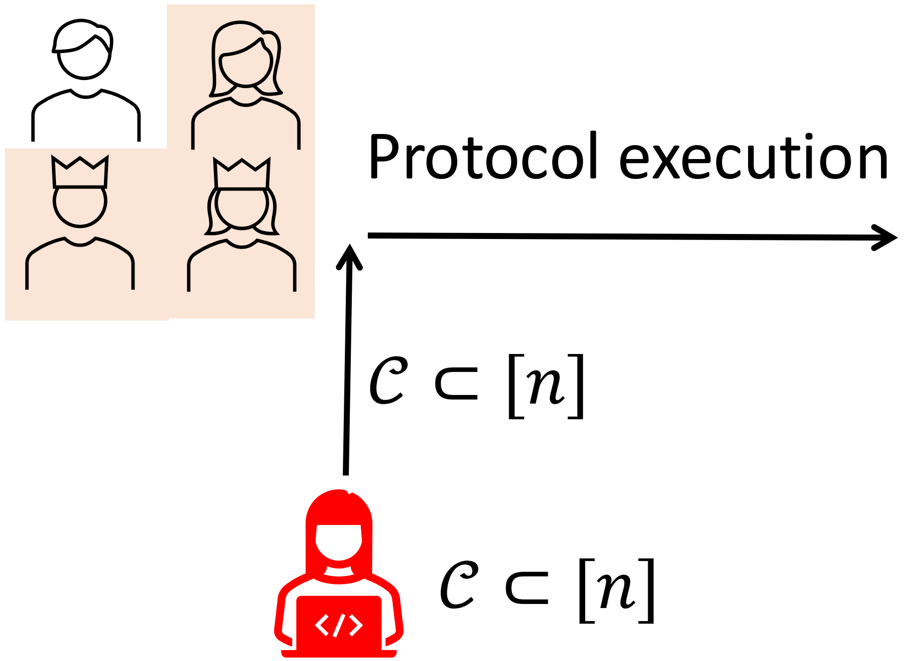
Needs to decide who to corrupt before the protocol begins.



Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



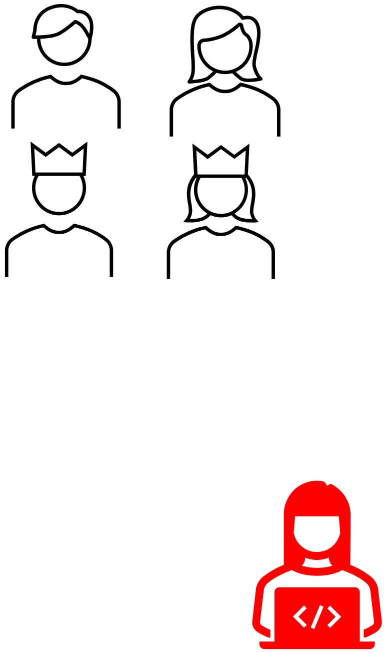
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.

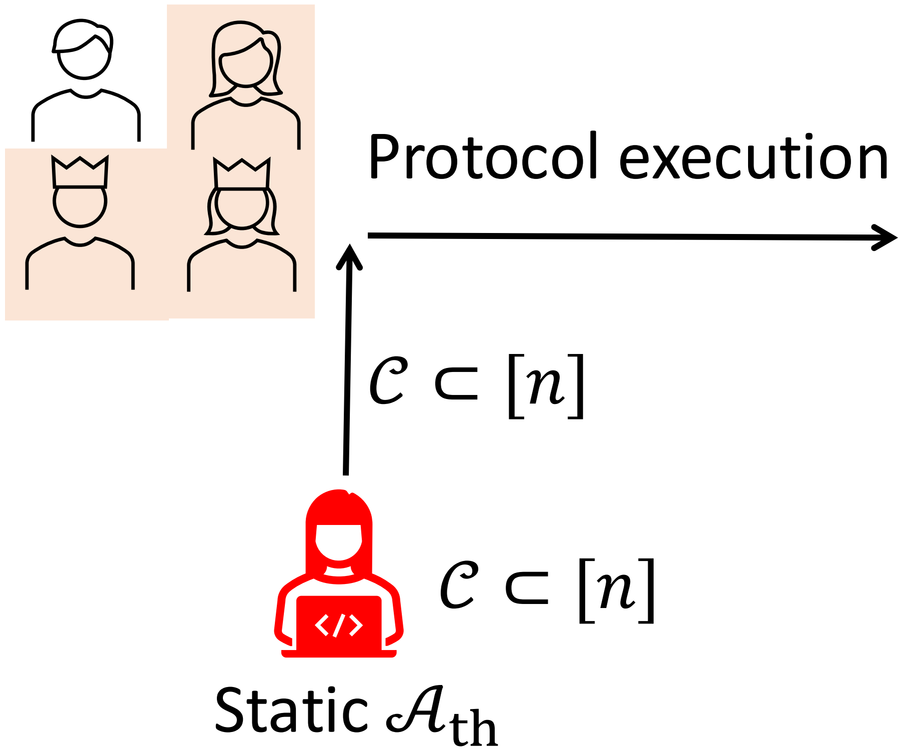


$\mathcal{C} = \{3\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



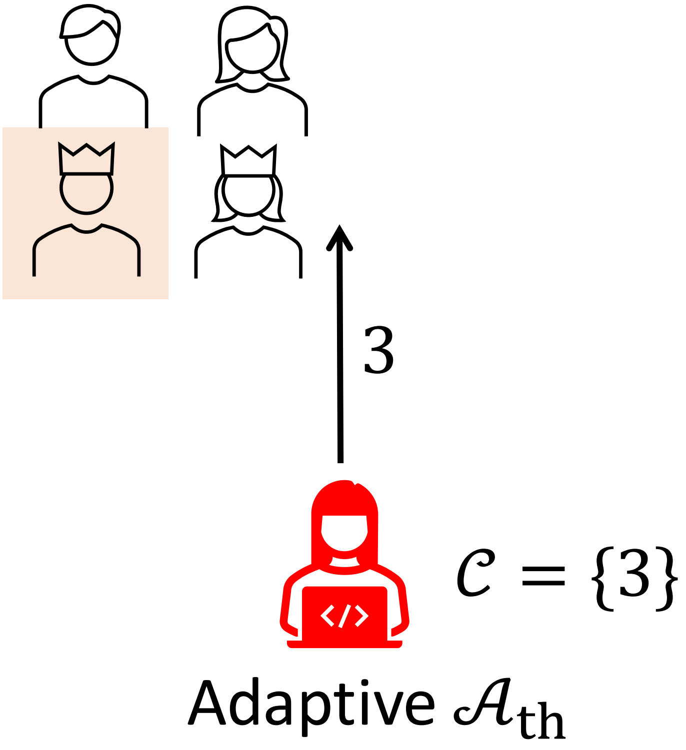
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.



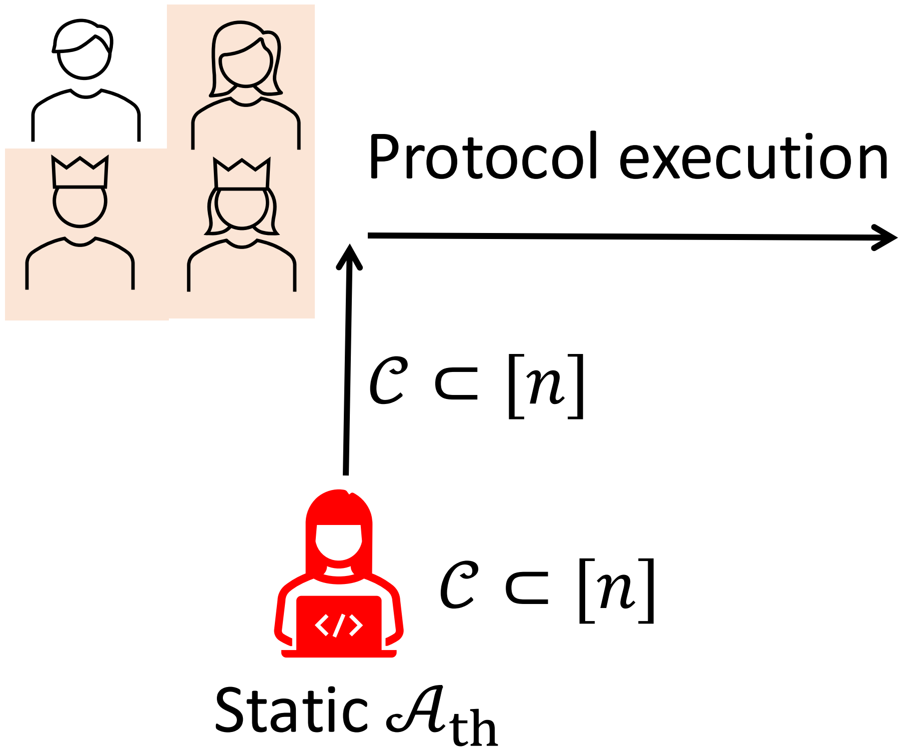
3

$\mathcal{C} = \{3\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security

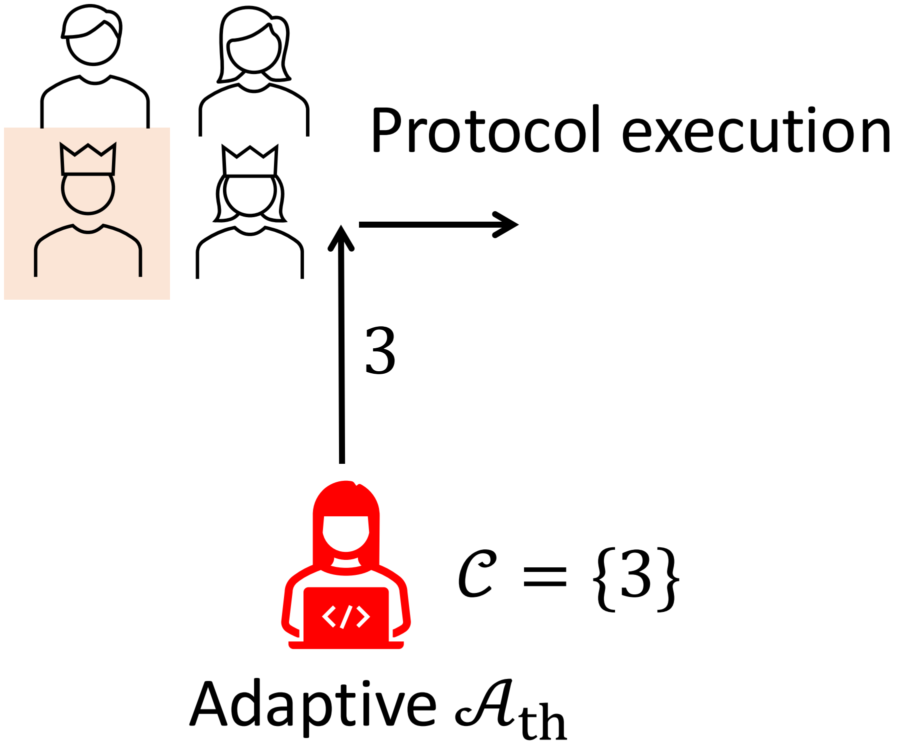

Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.


Protocol execution

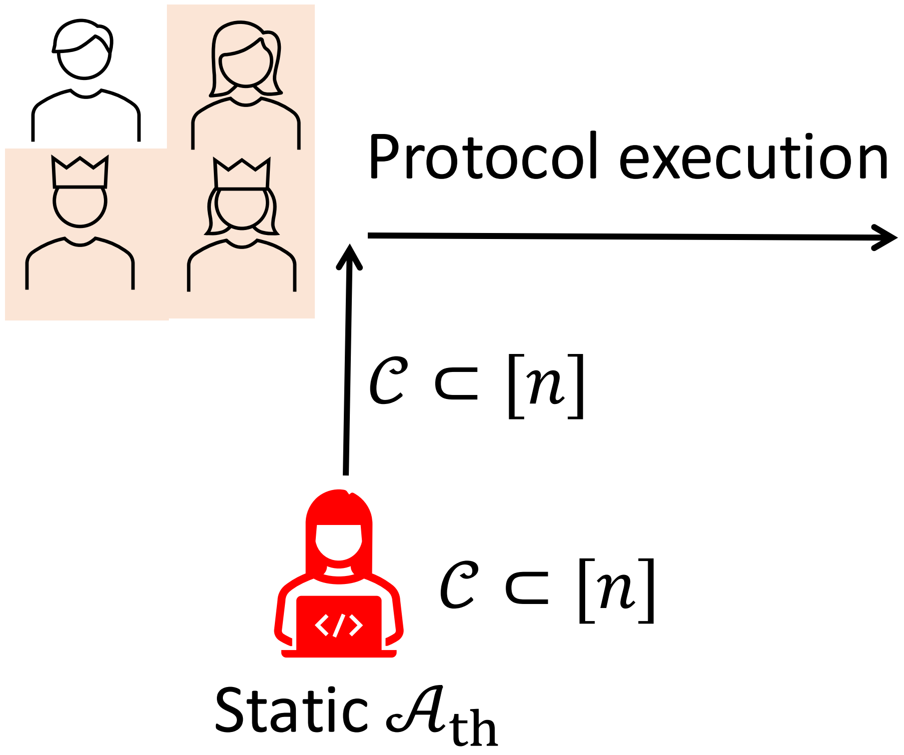
3

$\mathcal{C} = \{3\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



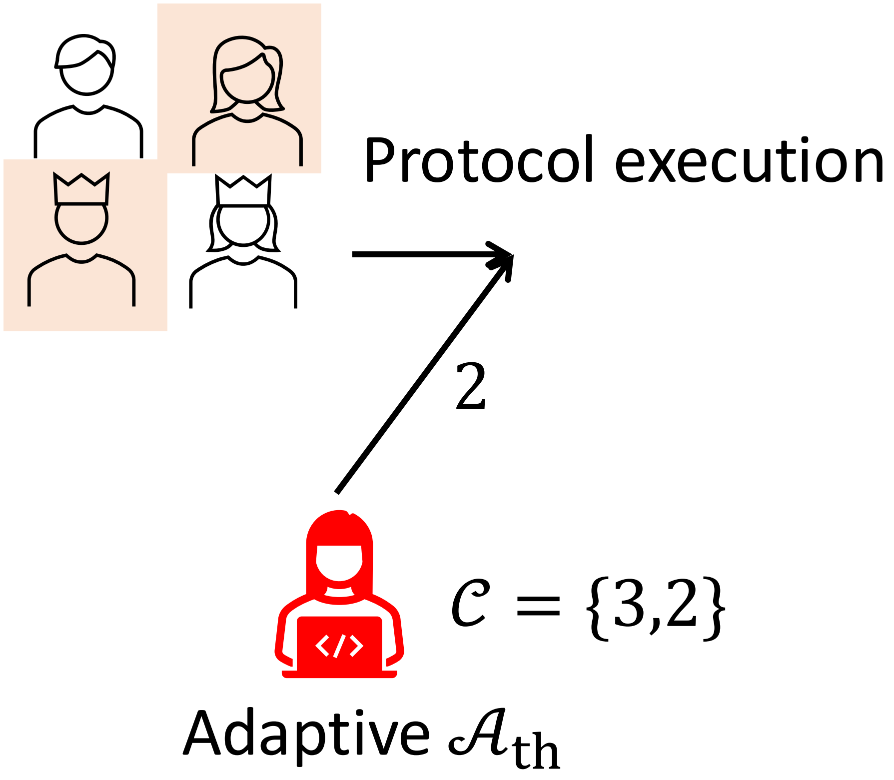
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.



Protocol execution

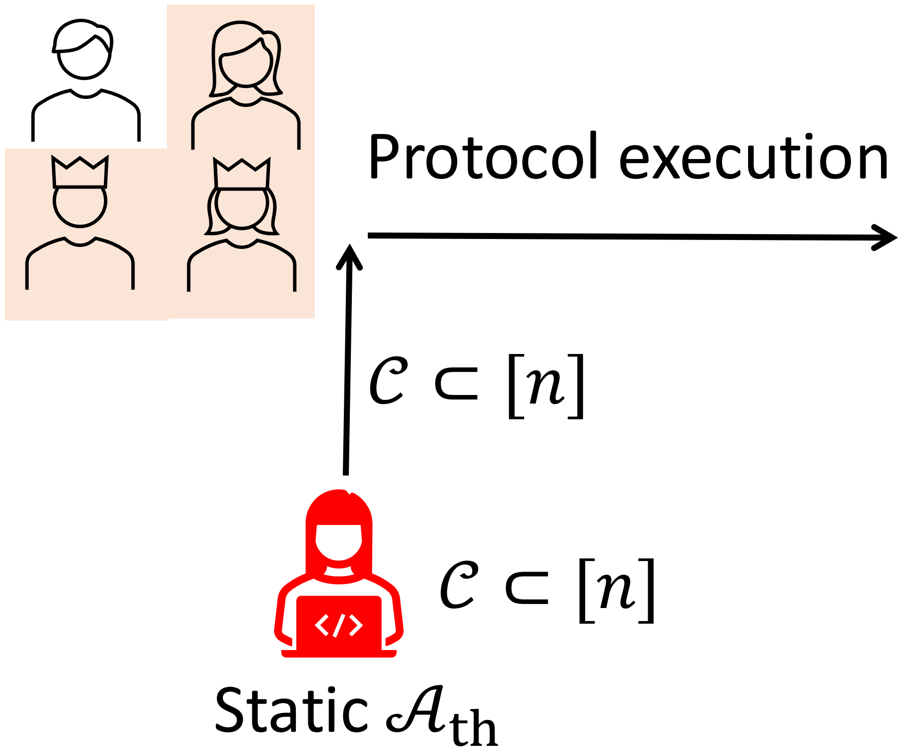
2

$\mathcal{C} = \{3,2\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



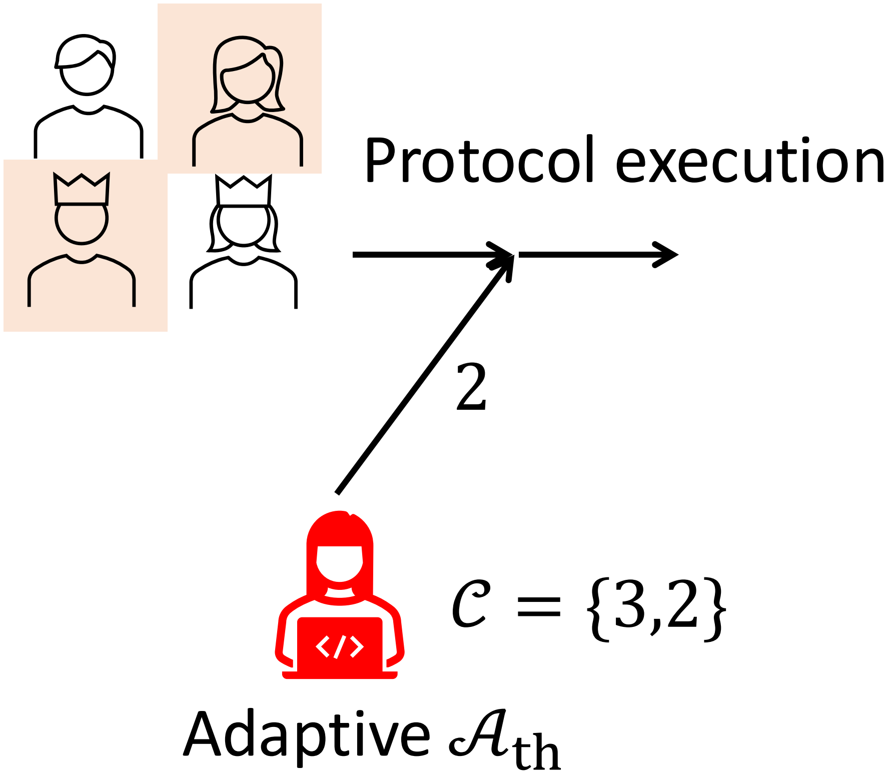
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.



Protocol execution

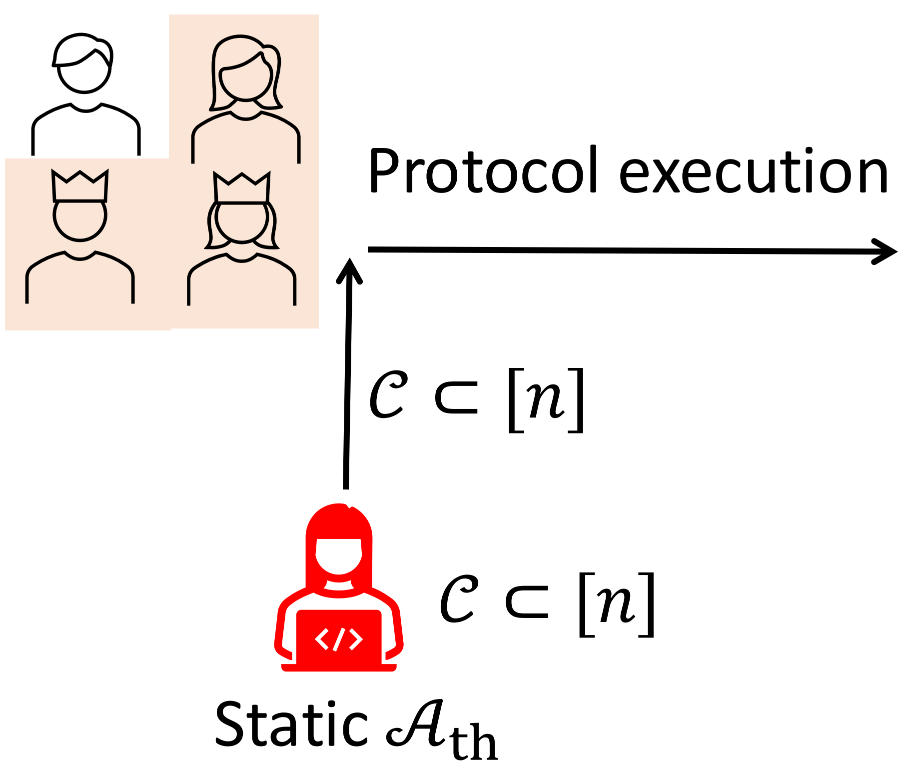
2

$\mathcal{C} = \{3,2\}$

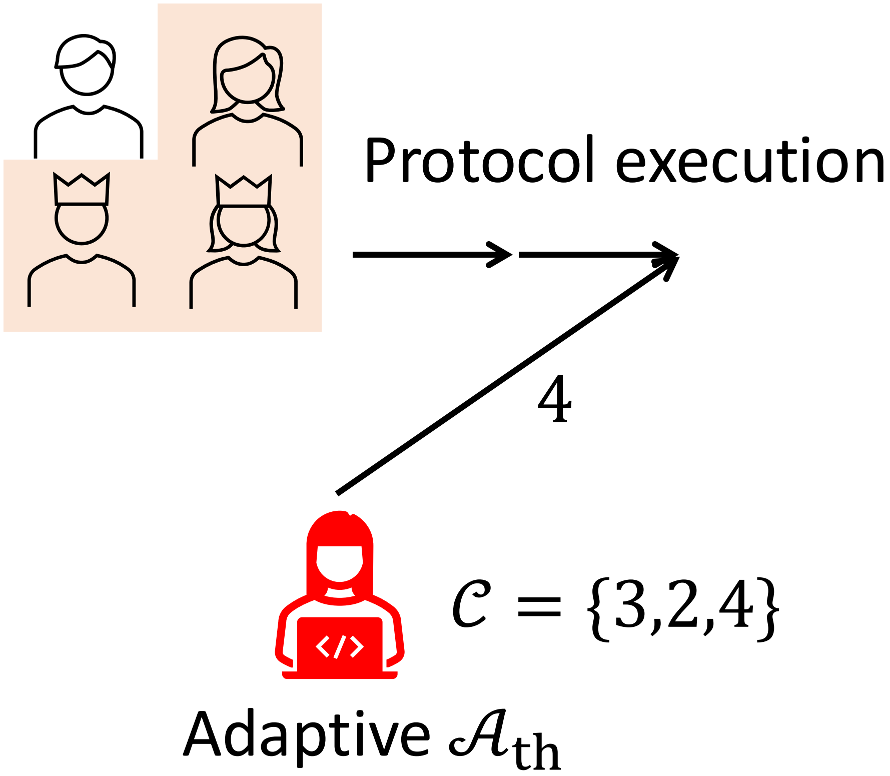
Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security

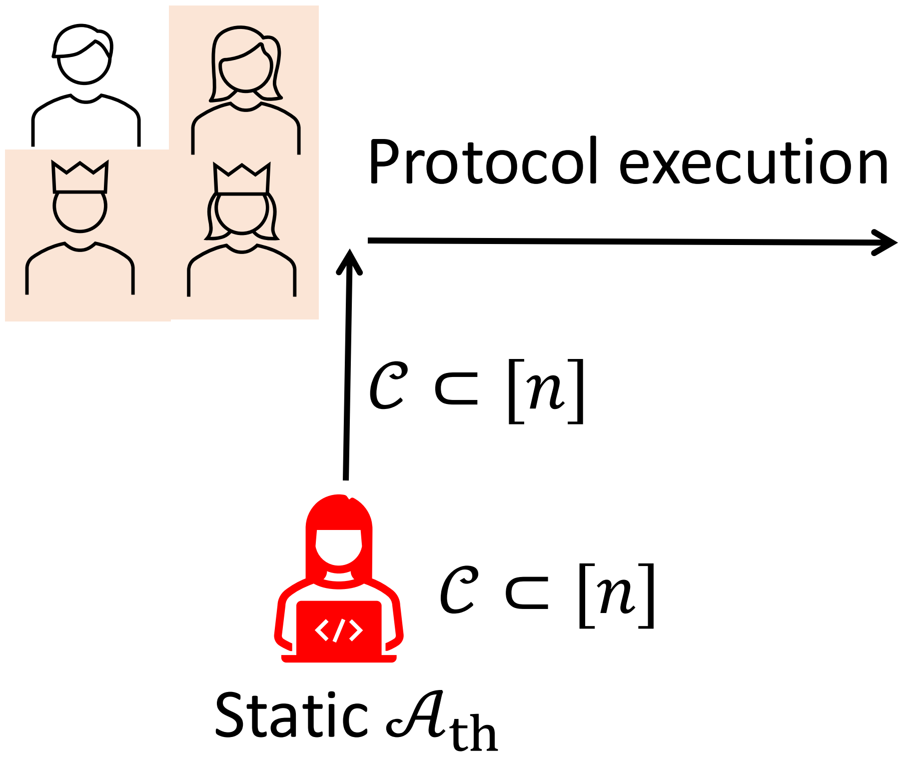


Needs to decide who to corrupt before the protocol begins.



No restriction corruption timing

Static vs Adaptive Security



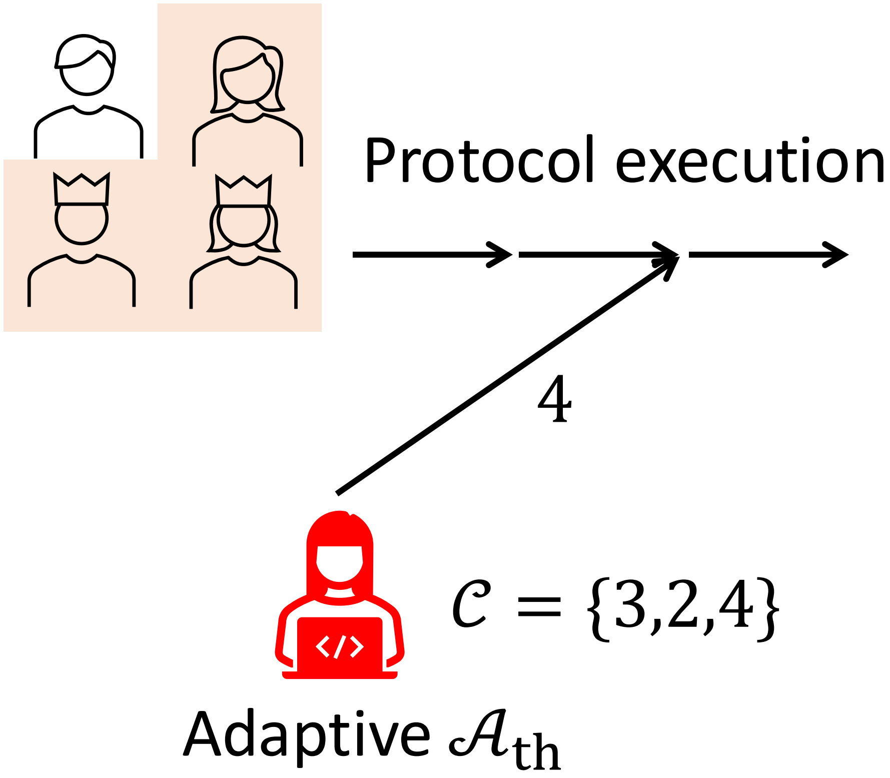
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.



Protocol execution

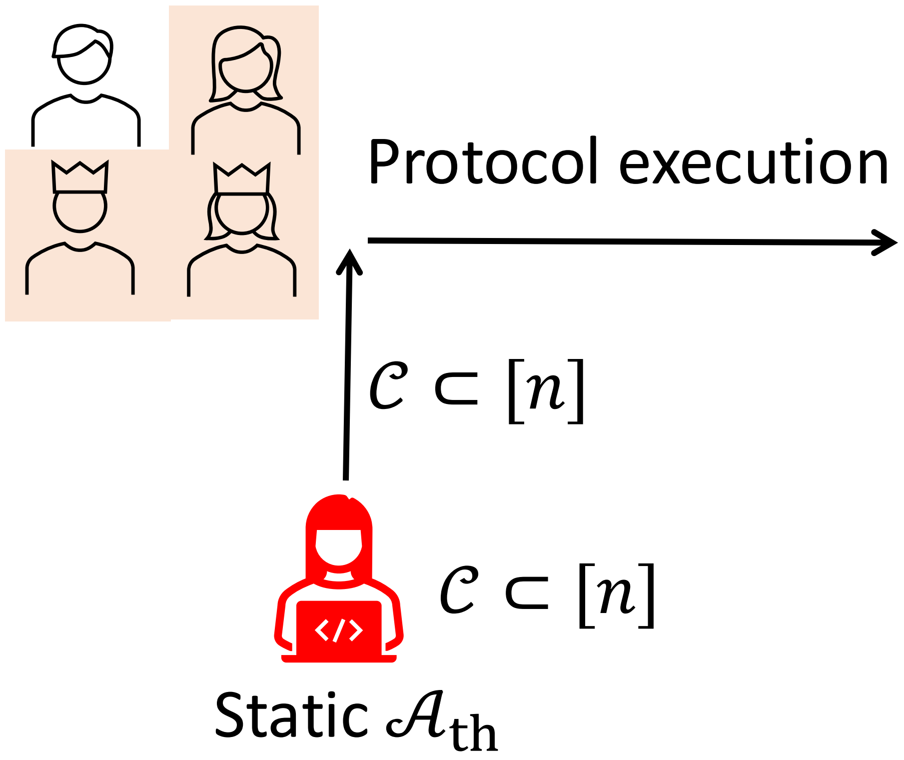
4

$\mathcal{C} = \{3,2,4\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Static vs Adaptive Security



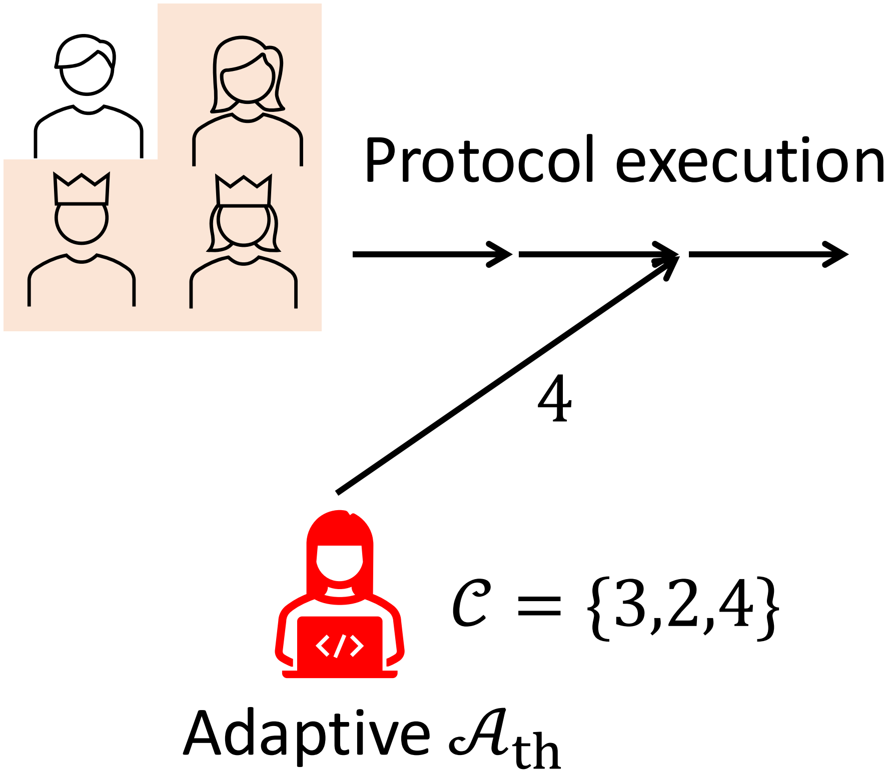
Protocol execution

$\mathcal{C} \subset [n]$

$\mathcal{C} \subset [n]$

Static $\mathcal{A}_{\text{th}}$

Needs to decide who to corrupt before the protocol begins.



Protocol execution

4

$\mathcal{C} = \{3,2,4\}$

Adaptive $\mathcal{A}_{\text{th}}$

No restriction corruption timing

Adaptive corruption is more natural

Why Static Security?

Why Static Security?

Has to do with known techniques of security reduction.

Why Static Security?

Has to do with known techniques of security reduction. More details later.

Why Static Security?

Has to do with known techniques of security reduction. More details later.

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
    - One-More Discrete Logarithm (OMDL) and Algebraic Group Model

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
  - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
    - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
    - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
    - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
  - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH
- Signatures are not compatible with BLS

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
    - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH
- Signatures are not compatible with BLS

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
  - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH
- Signatures are not compatible with BLS

Our Approach:

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
  - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH
- Signatures are not compatible with BLS

Our Approach:

- Minor modification to [Boldyreva03] while ensuring compatibility

Why Static Security?

Has to do with known techniques of security reduction. More details later.

[Bacho-Loss 2022]

- Adaptive security of [Boldyreva03]
  - One-More Discrete Logarithm (OMDL) and Algebraic Group Model
- OMDL is necessary for [Boldyreva03]

[Libert-Joye-Yung 2016]

- Adaptively secure non-interactive signatures with DDH
- Signatures are not compatible with BLS

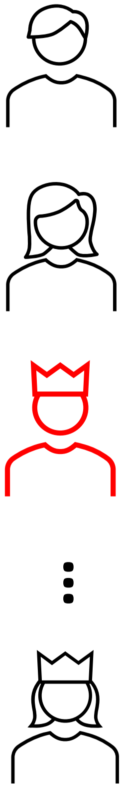
Our Approach:

- Minor modification to [Boldyreva03] while ensuring compatibility
- New proof techniques

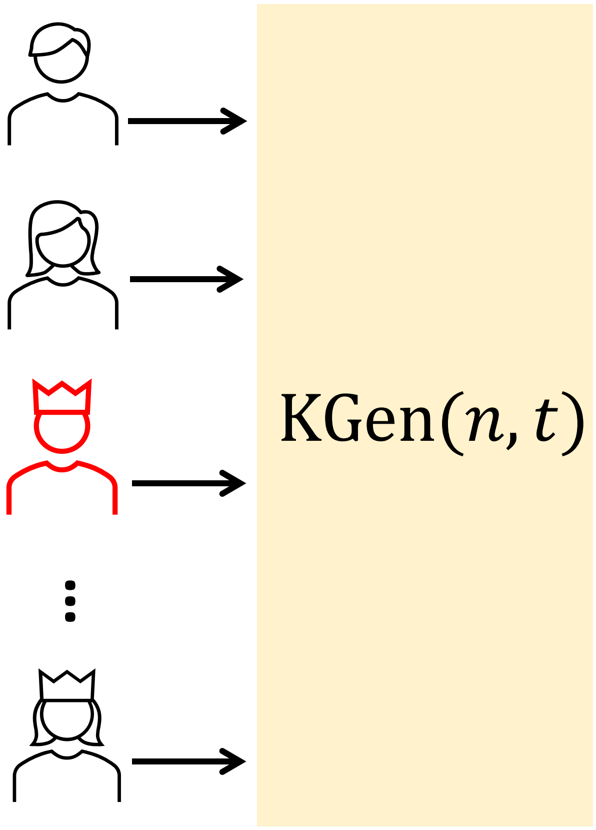
Our Protocol with Idealized Key Generation

Our Approach: Key Generation

Our Approach: Key Generation

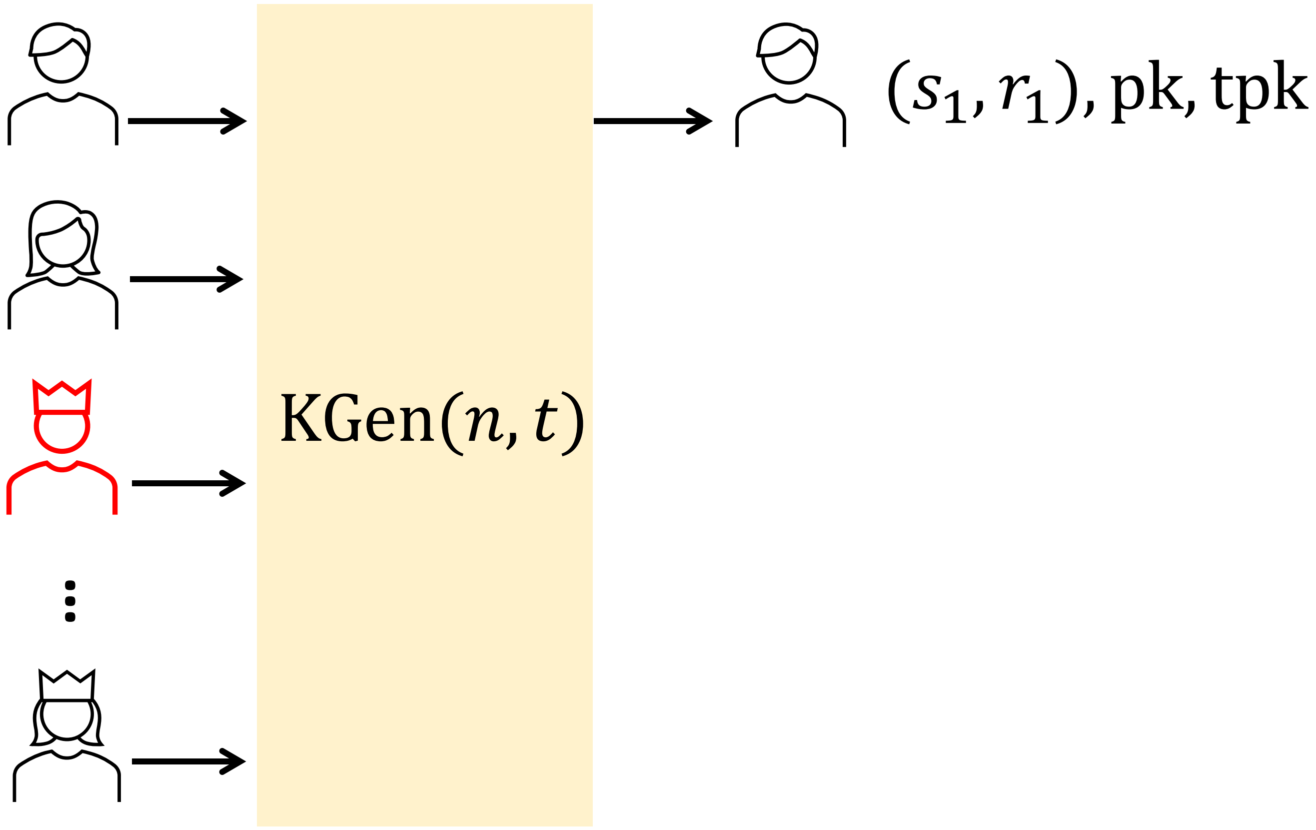


Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

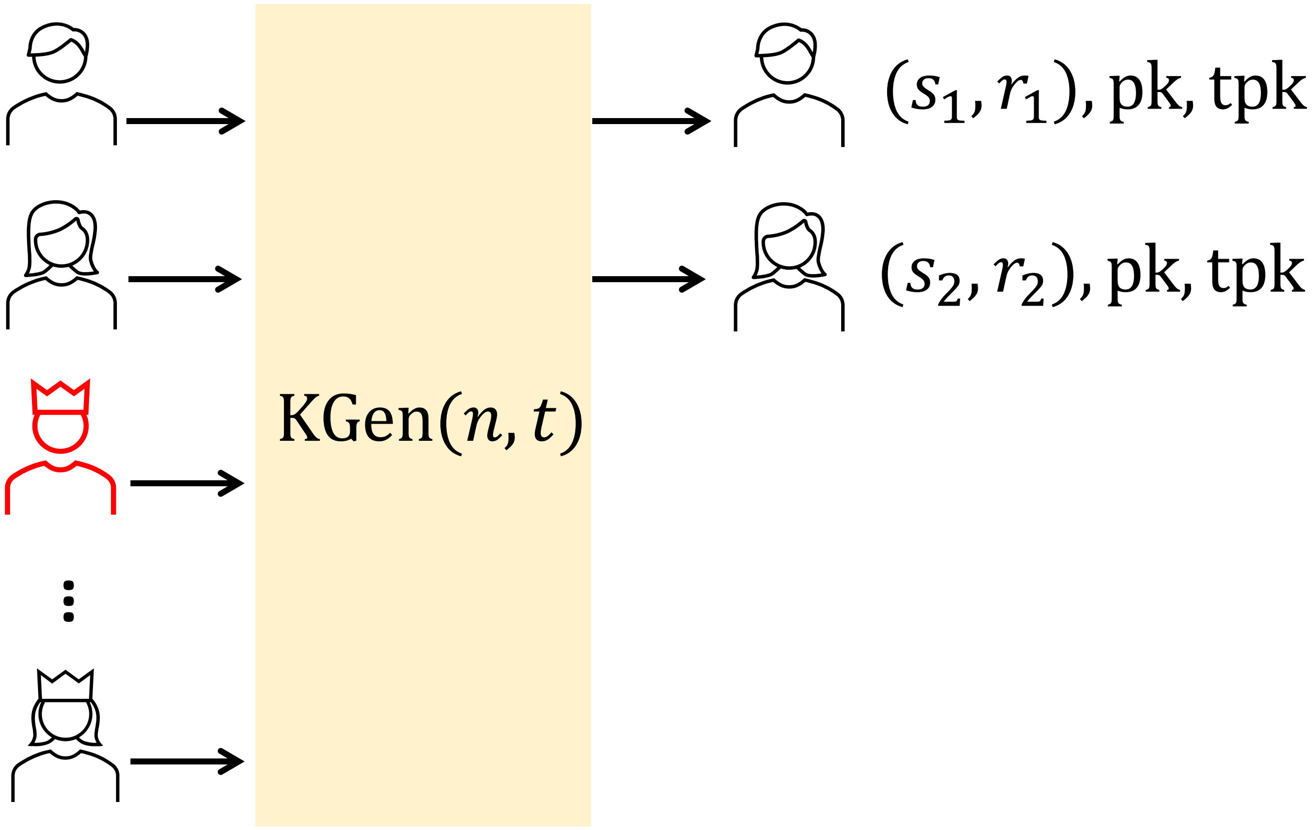
Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

Our Approach: Key Generation

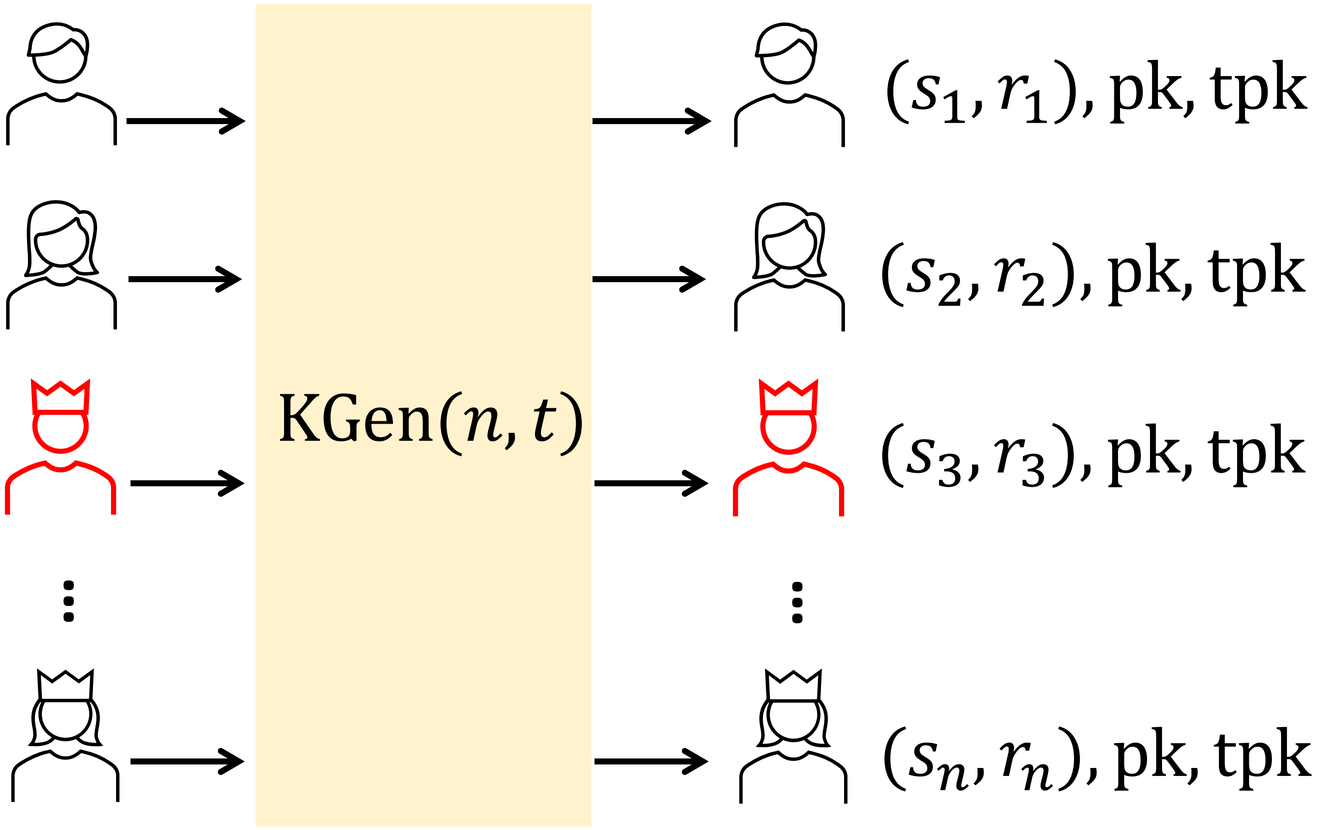


$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

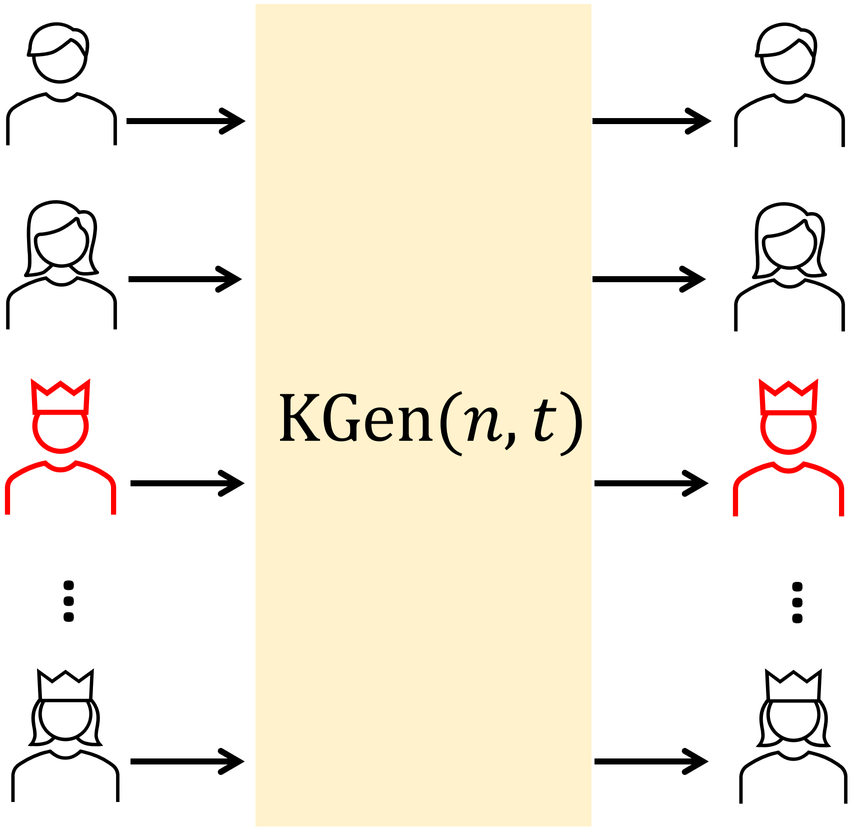
$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n), \mathrm{pk}, \mathrm{tpk}$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

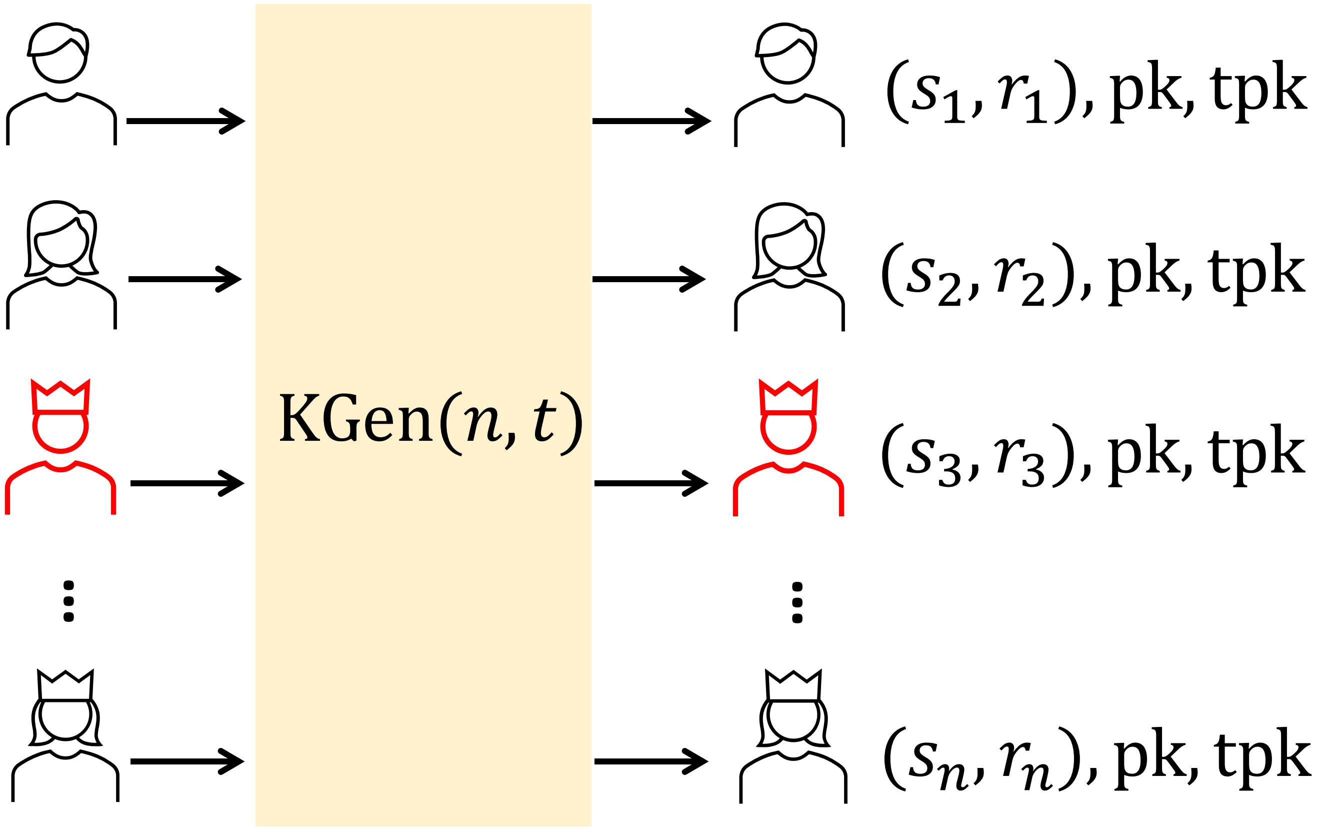
$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n), \mathrm{pk}, \mathrm{tpk}$

$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

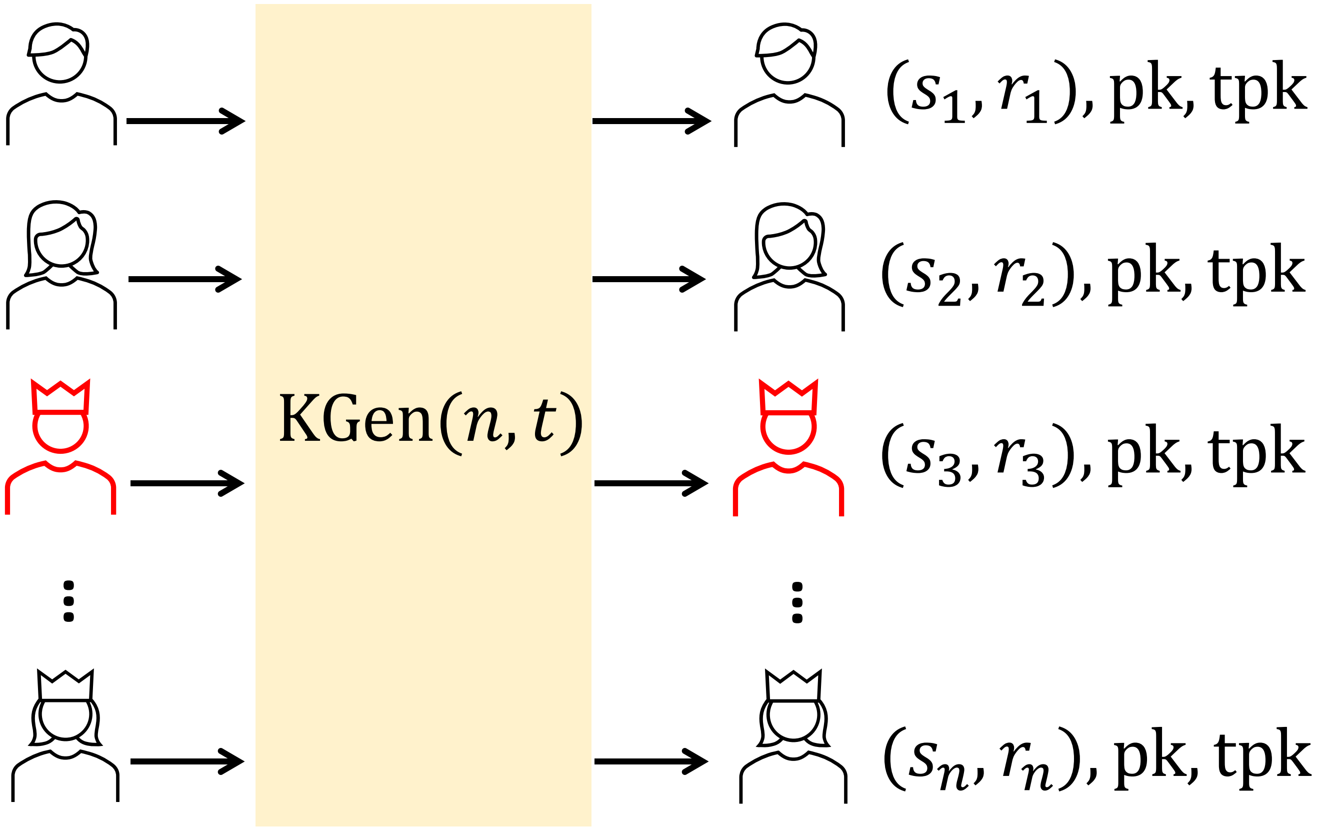
$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n), \mathrm{pk}, \mathrm{tpk}$

$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$
$\{r_1, \ldots, r_n\} \leftarrow \mathrm{Share}(0)$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

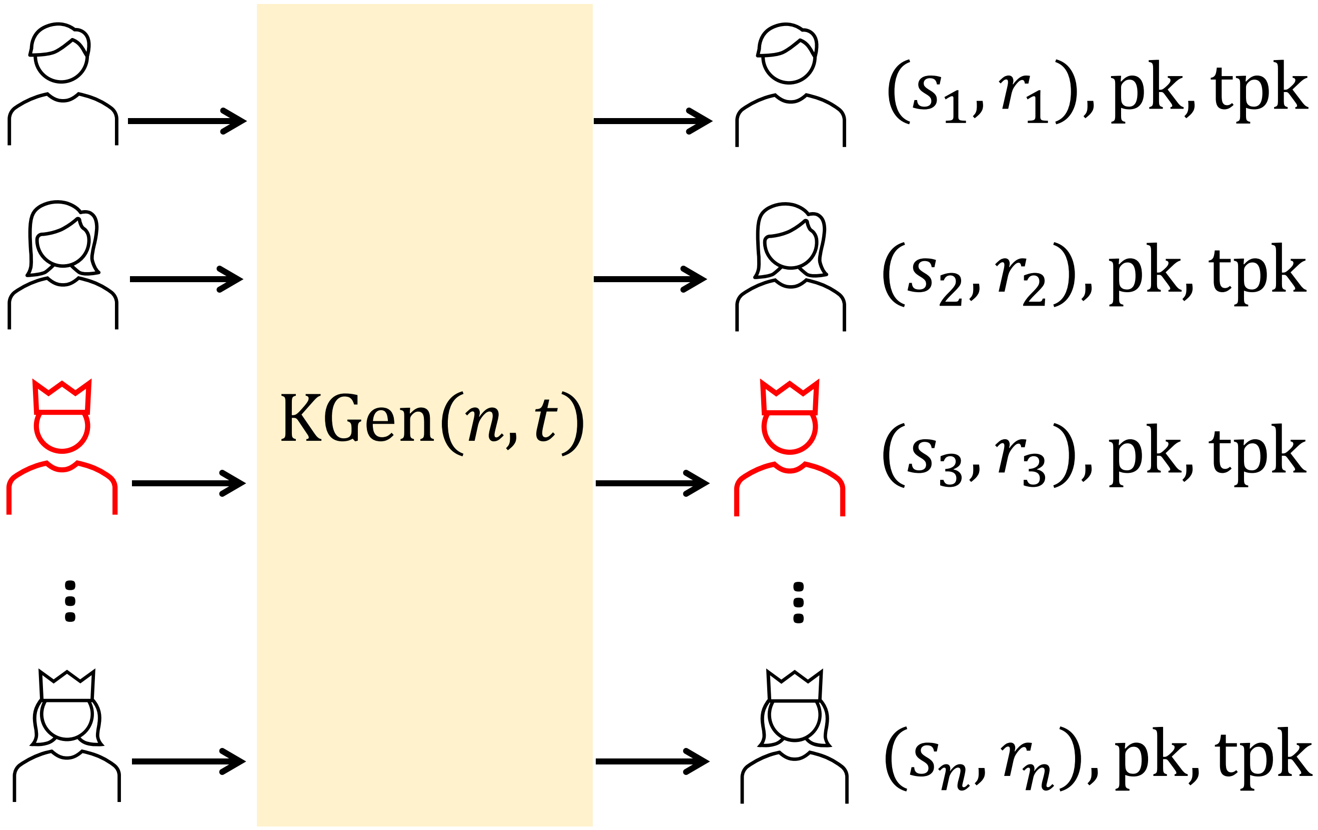
$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n), \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)$$
$$\mathrm{pk} = g^s$$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

(s_1, r_1), pk, tpk

(s_2, r_2), pk, tpk

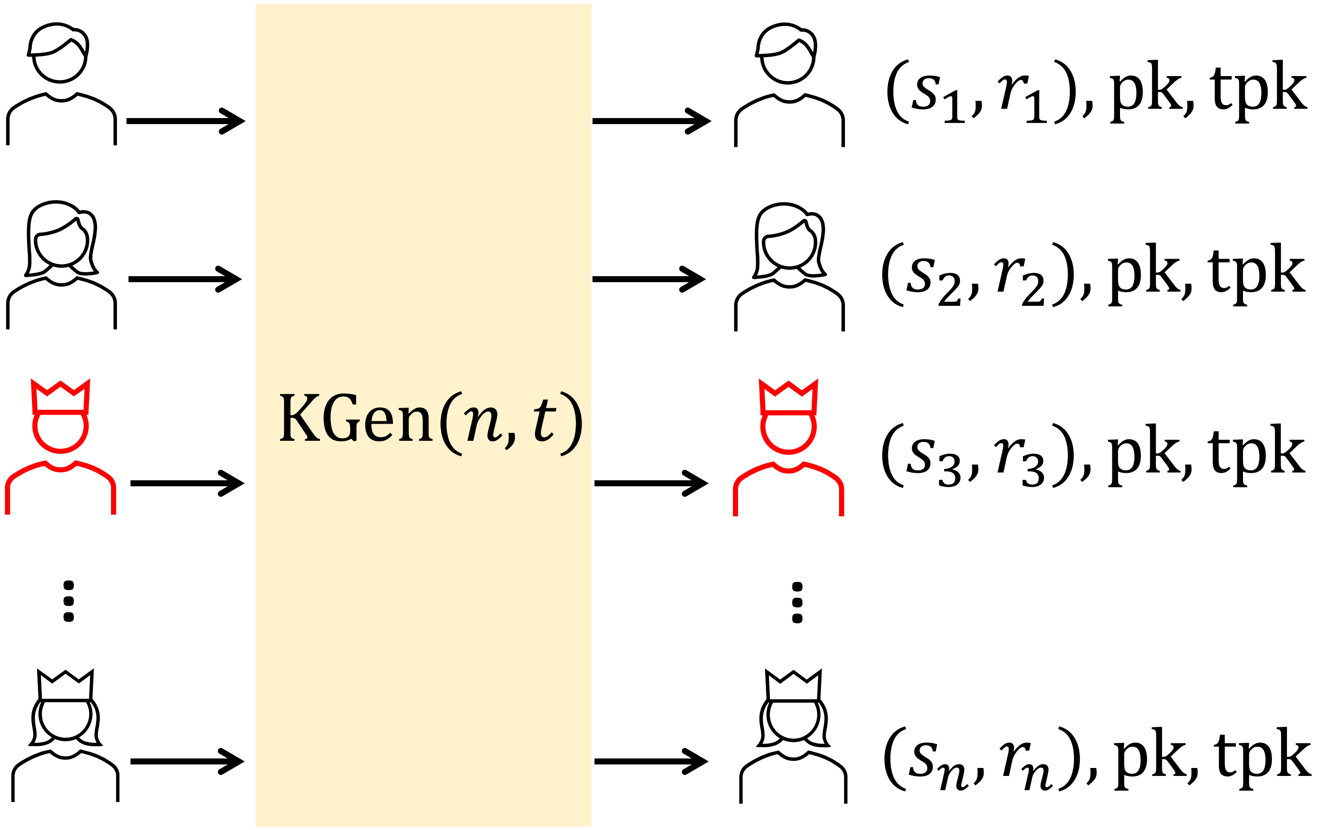
(s_3, r_3), pk, tpk

⋮

(s_n, r_n), pk, tpk

$$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\{r_1, \ldots, r_n\} \leftarrow \mathrm{Share}(0)$$
$$\mathrm{pk} = g^s$$
$$\mathrm{tpk} = \{g^{s_1}h^{r_1} \ldots, g^{s_n}h^{r_n}\}$$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, r_1), \mathrm{pk}, \mathrm{tpk}$

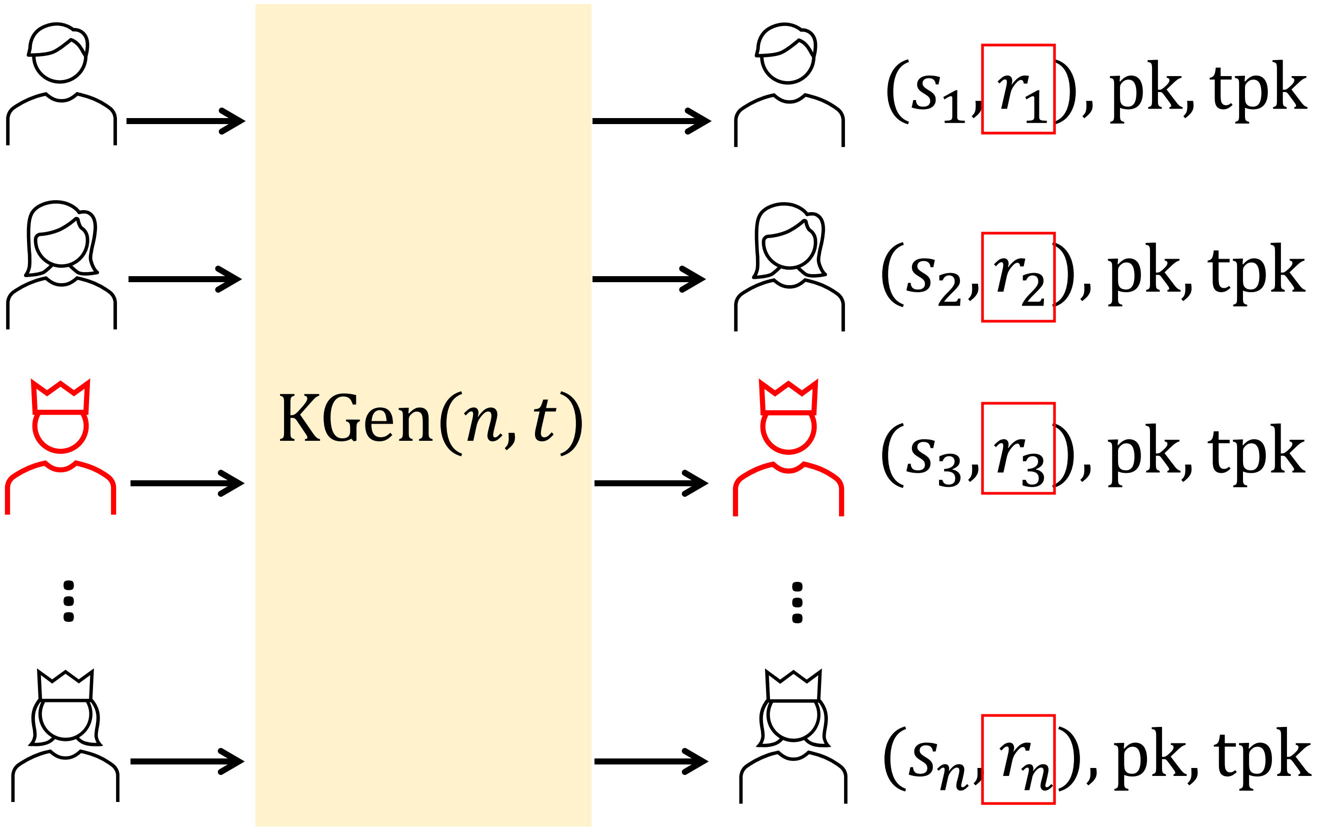
$(s_2, r_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n), \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\{r_1, \ldots, r_n\} \leftarrow \mathrm{Share}(0)$$
$$\mathrm{pk} = g^s$$
$$\mathrm{tpk} = \{g^{s_1} h^{r_1} \ldots, g^{s_n} h^{r_n}\}$$
$$g, h \in \mathbb{G}$$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3}), \mathrm{pk}, \mathrm{tpk}$

⋮

$(s_n, \boxed{r_n}), \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$

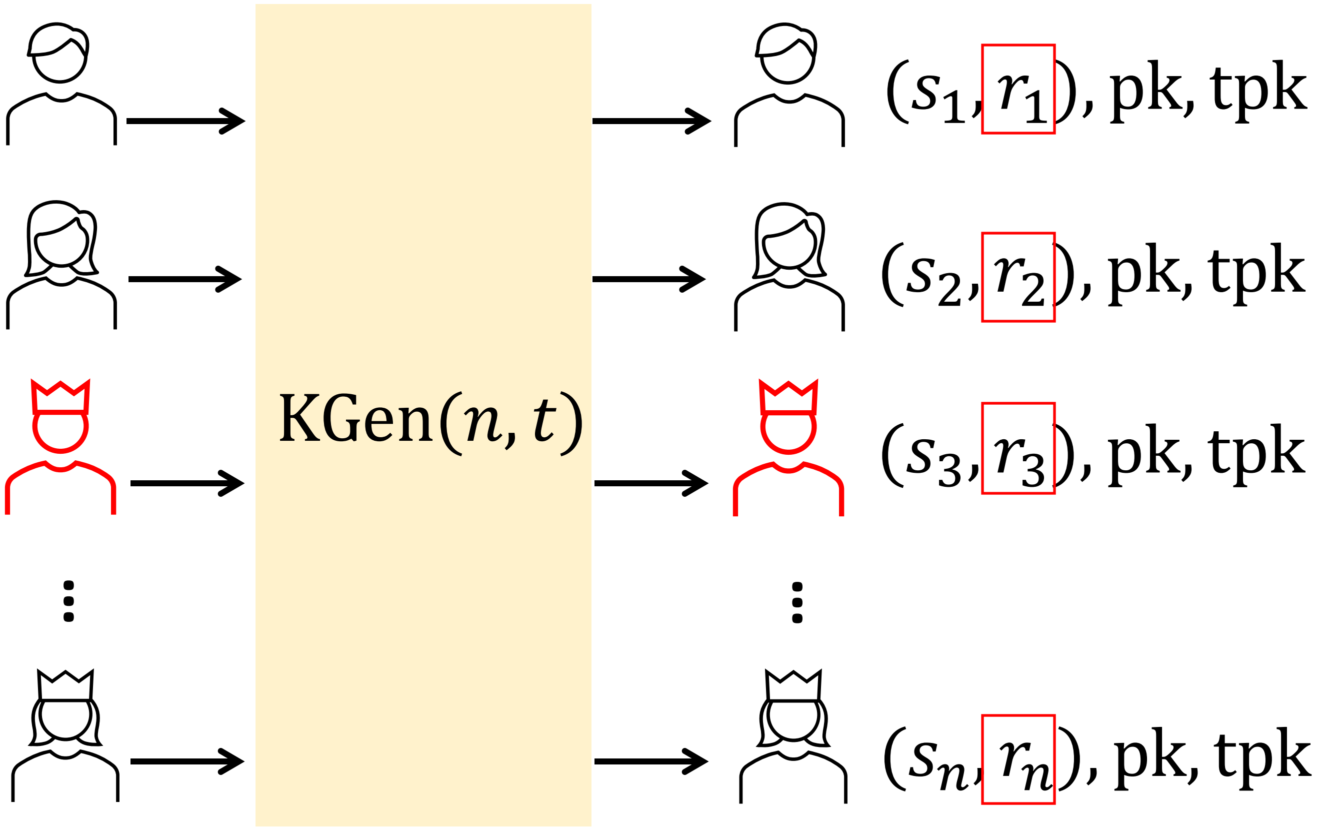
$$\boxed{\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)}$$

$$\mathrm{pk} = g^s$$

$$\mathrm{tpk} = \{g^{s_1}h^{r_1} \dots, g^{s_n}h^{r_n}\}$$

$$g, h \in \mathbb{G}$$

Our Approach: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3}), \mathrm{pk}, \mathrm{tpk}$

$(s_n, \boxed{r_n}), \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$

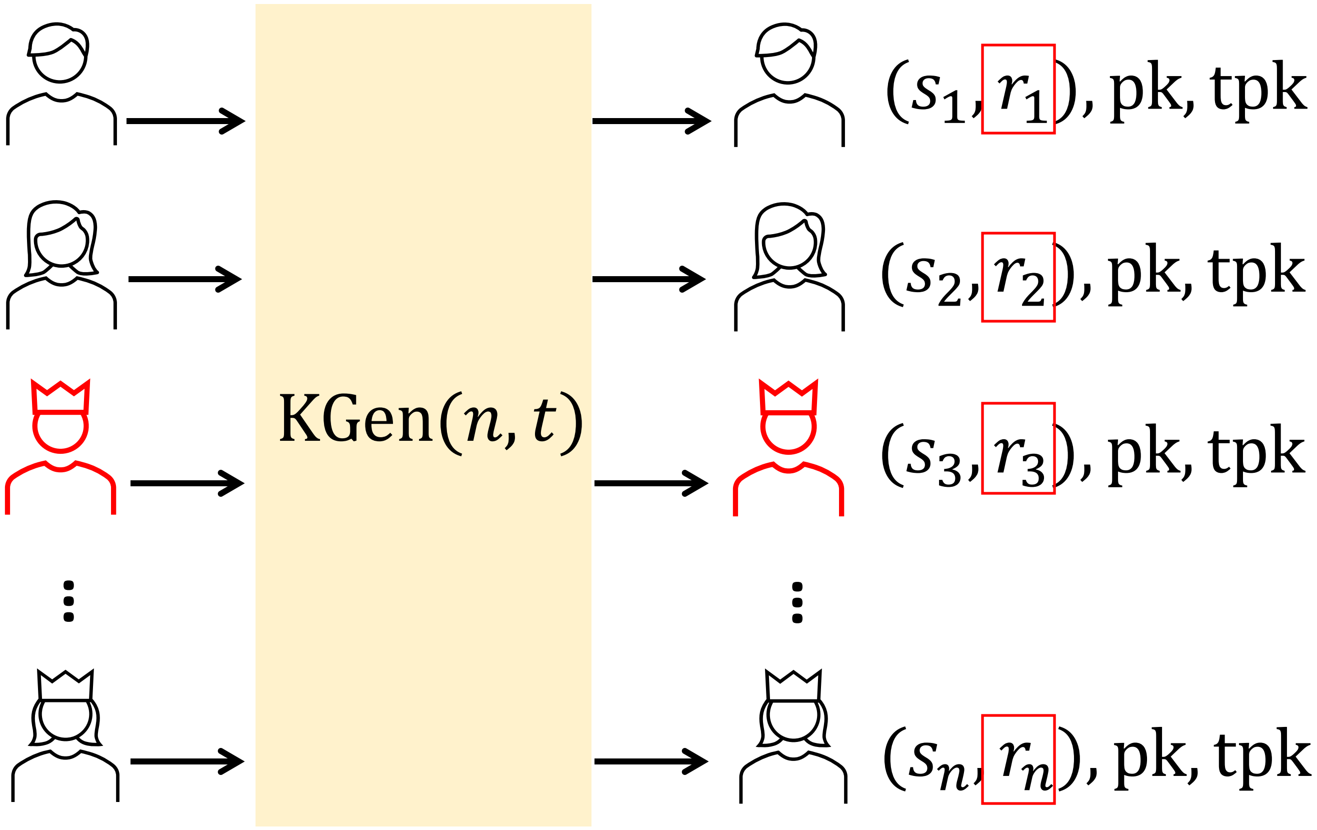
$$\boxed{\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)}$$

$$\mathrm{pk} = g^s$$

$$\mathrm{tpk} = \{g^{s_1} h^{r_1} \dots, g^{s_n} h^{r_n}\}$$

$$g, h \in \mathbb{G}$$

Our Approach: Key Generation



$\mathsf{KGen}(n, t)$

$(s_1, \boxed{r_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3}), \mathrm{pk}, \mathrm{tpk}$

$(s_n, \boxed{r_n}), \mathrm{pk}, \mathrm{tpk}$

$$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$$

$$\boxed{\{r_1, \ldots, r_n\} \leftarrow \mathrm{Share}(0)}$$

$$\mathrm{pk} = g^s$$

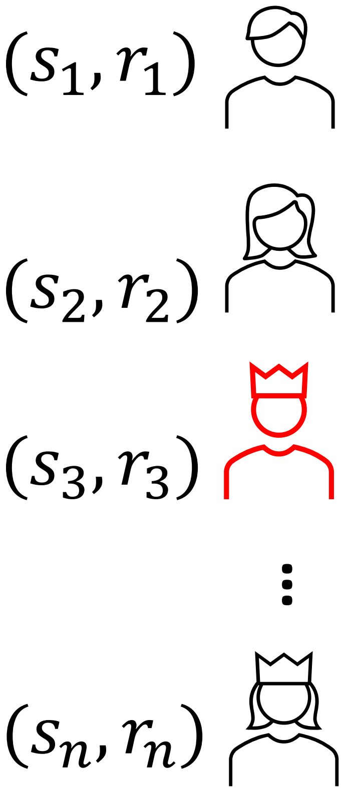
$$\mathrm{tpk} = \{g^{s_1} h^{r_1} \ldots, g^{s_n} h^{r_n}\}$$

$$g, h \in \mathbb{G}$$

We describe a Distributed Key Generation (DKG) in the paper.

Our Approach: Signing

Our Approach: Signing



(s_1, r_1)

(s_2, r_2)

(s_3, r_3)

$\vdots$

(s_n, r_n)

Our Approach: Signing

(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

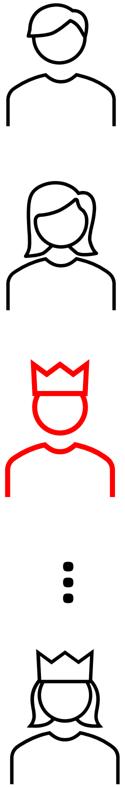
(s_2, r_2)

(s_3, r_3)

⋮

(s_n, r_n)

Our Approach: Signing



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2)

(s_3, r_3)

⋮

(s_n, r_n)

Our Approach: Signing

(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

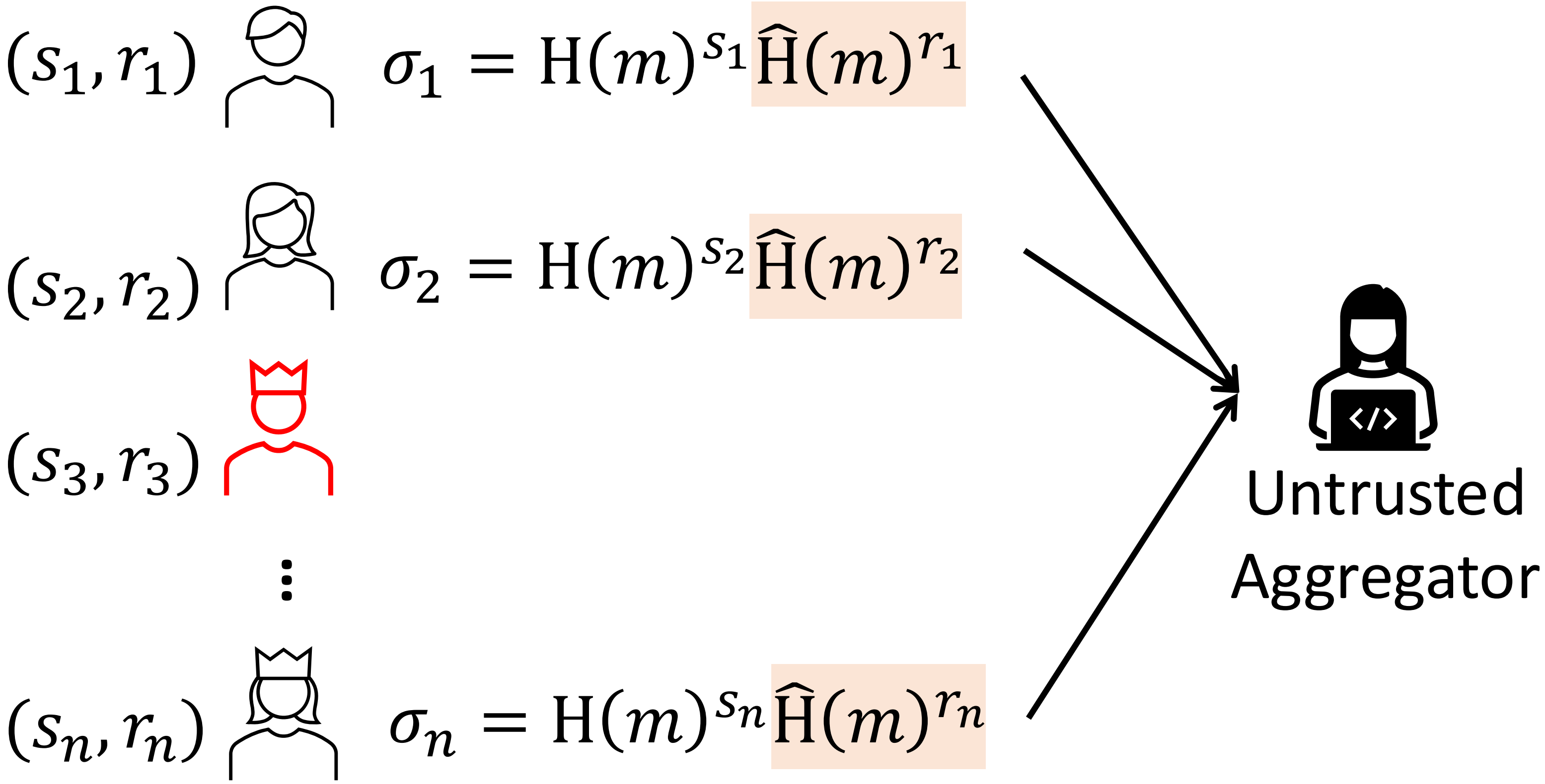
(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

Our Approach: Signing



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

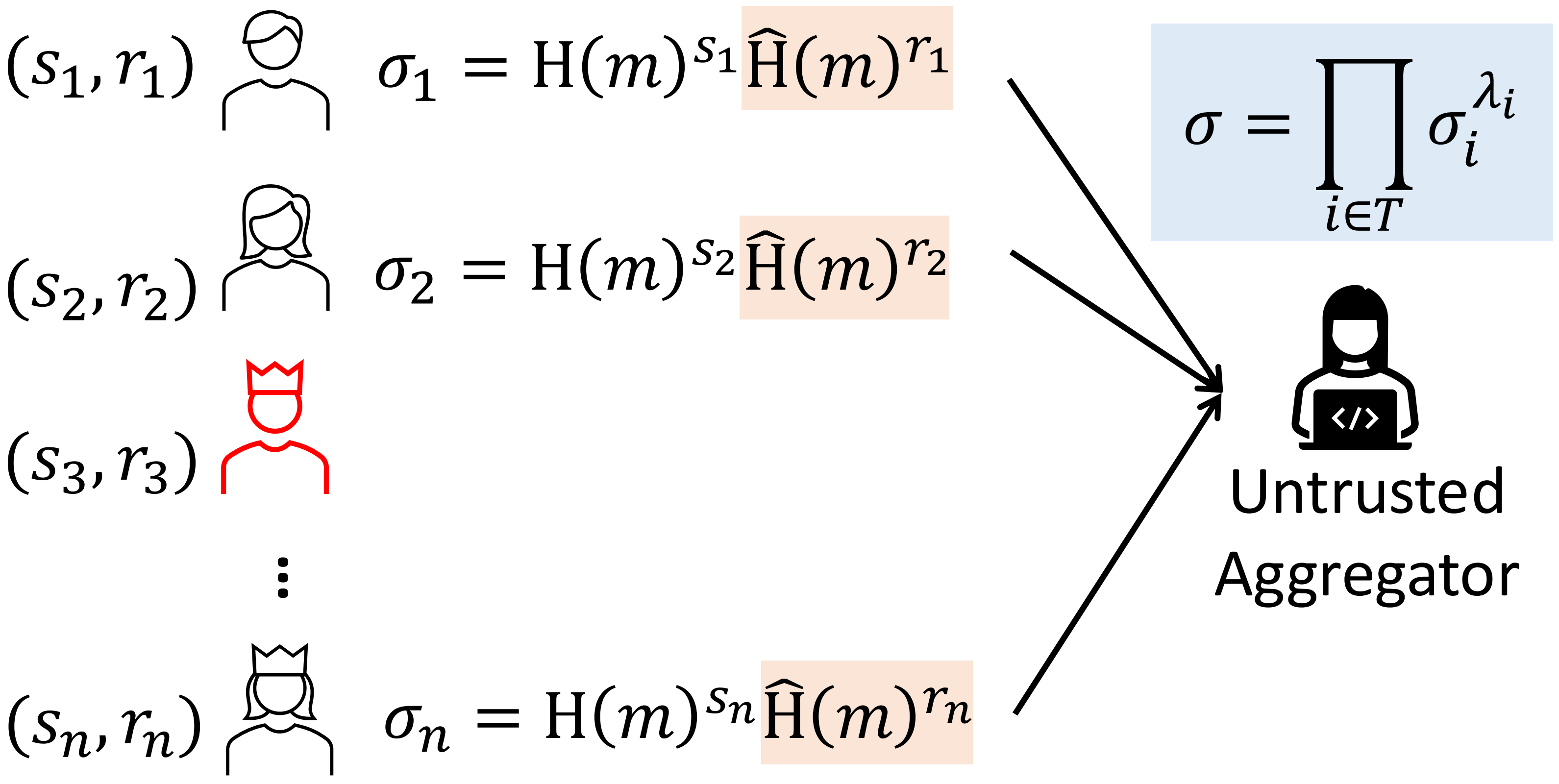
(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

Untrusted Aggregator

Our Approach: Signing



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

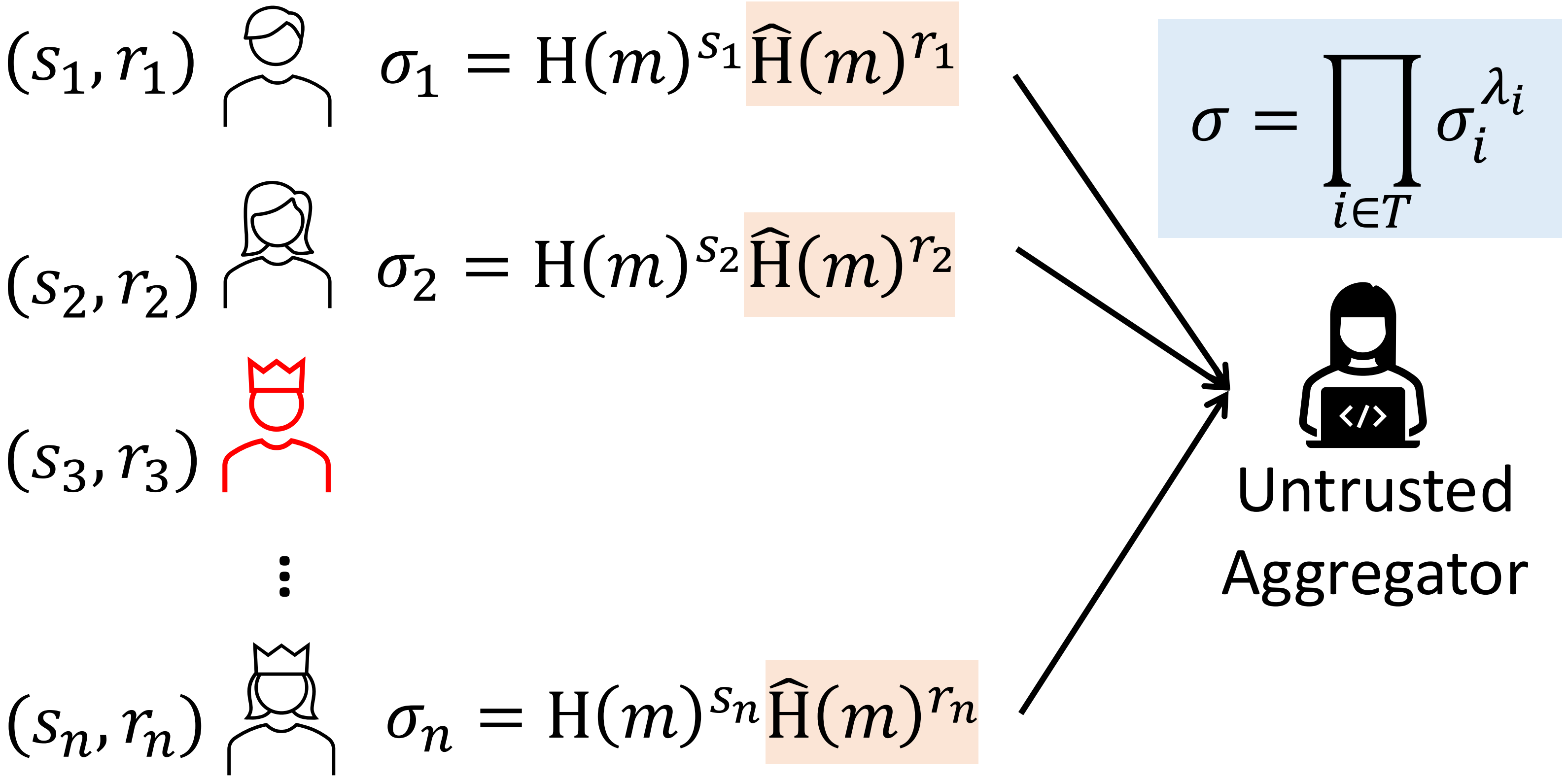
(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

Our Approach: Signing

λ_i are the Lagrange coefficients.



(s_1, r_1) $\quad \sigma_1 = \mathrm{H}(m)^{s_1} \widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\quad \sigma_2 = \mathrm{H}(m)^{s_2} \widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

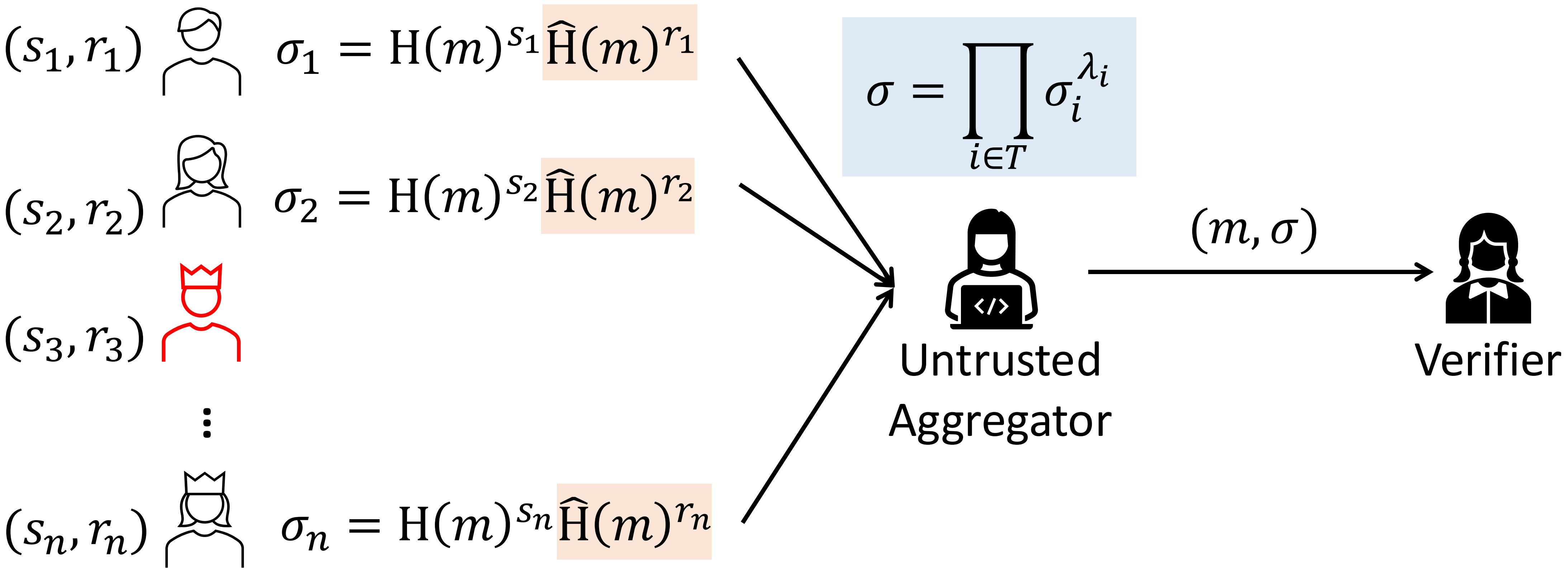
(s_n, r_n) $\quad \sigma_n = \mathrm{H}(m)^{s_n} \widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

Our Approach: Signing

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

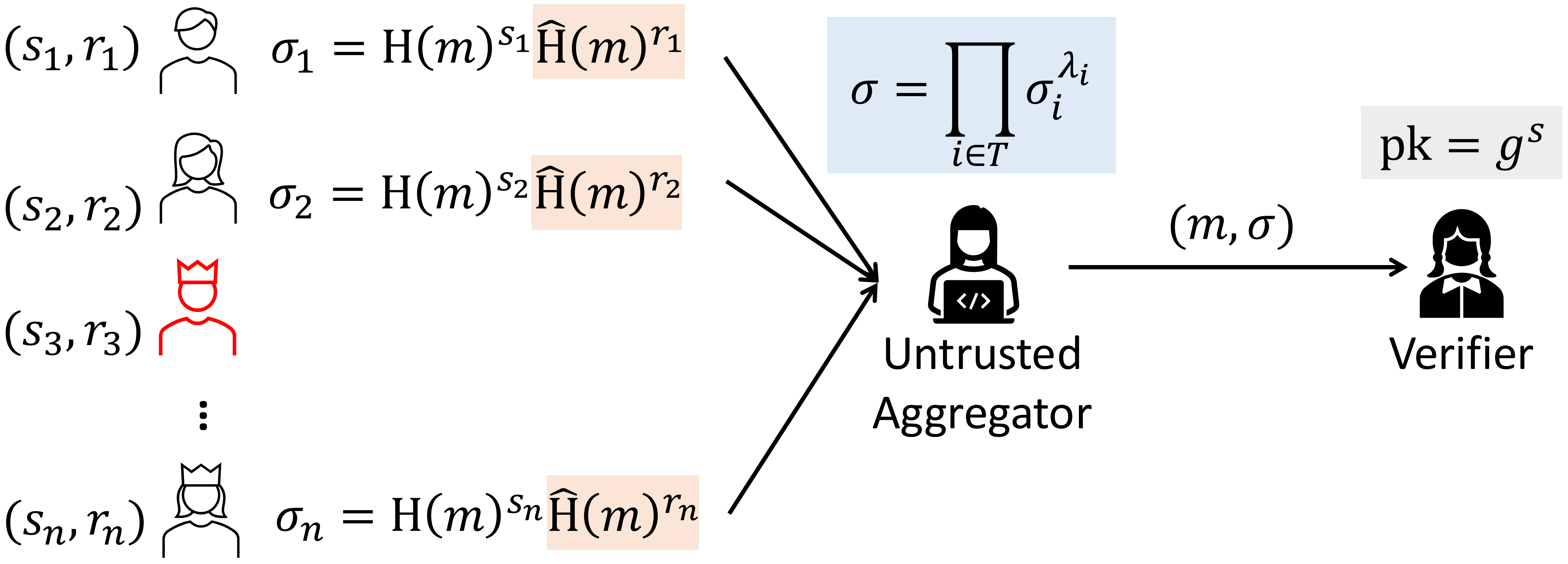
(m, σ)

Untrusted Aggregator

Verifier

Our Approach: Signing

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$\mathrm{pk} = g^s$

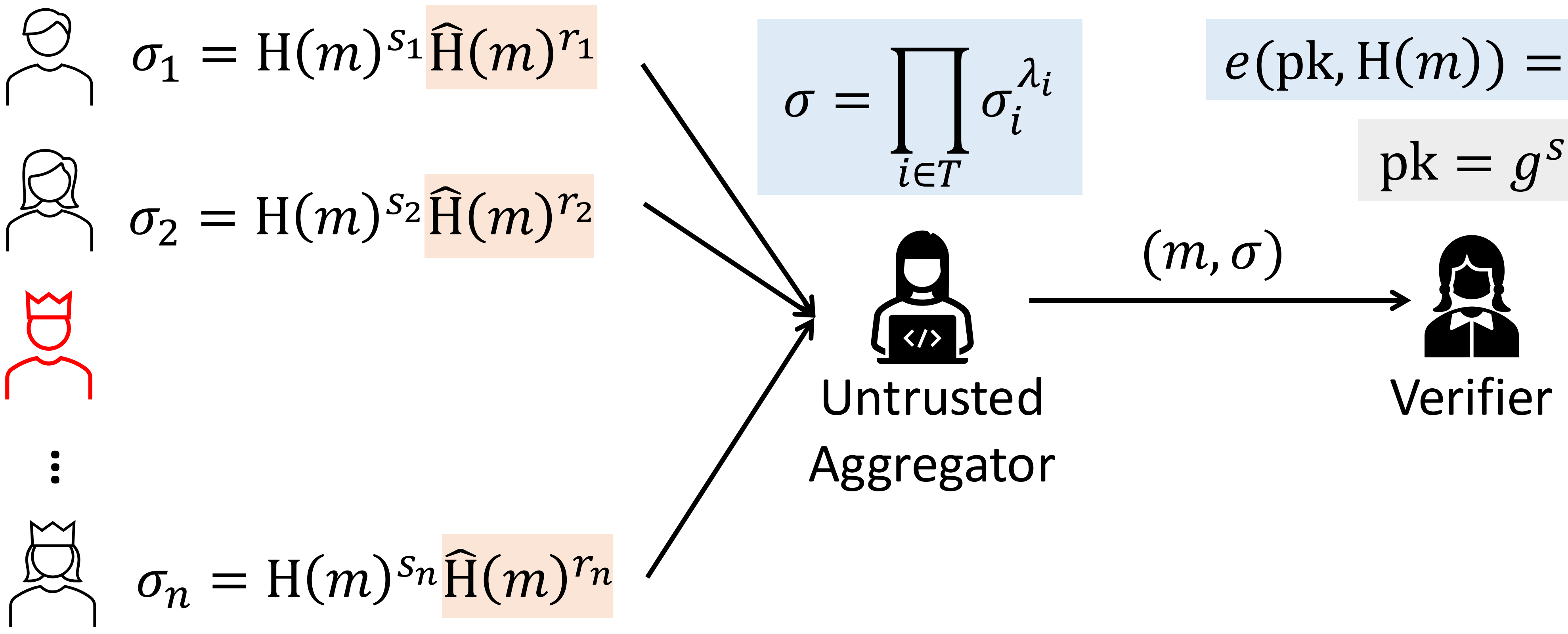
(m, σ)

Untrusted Aggregator

Verifier

Our Approach: Signing

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

(m, σ)

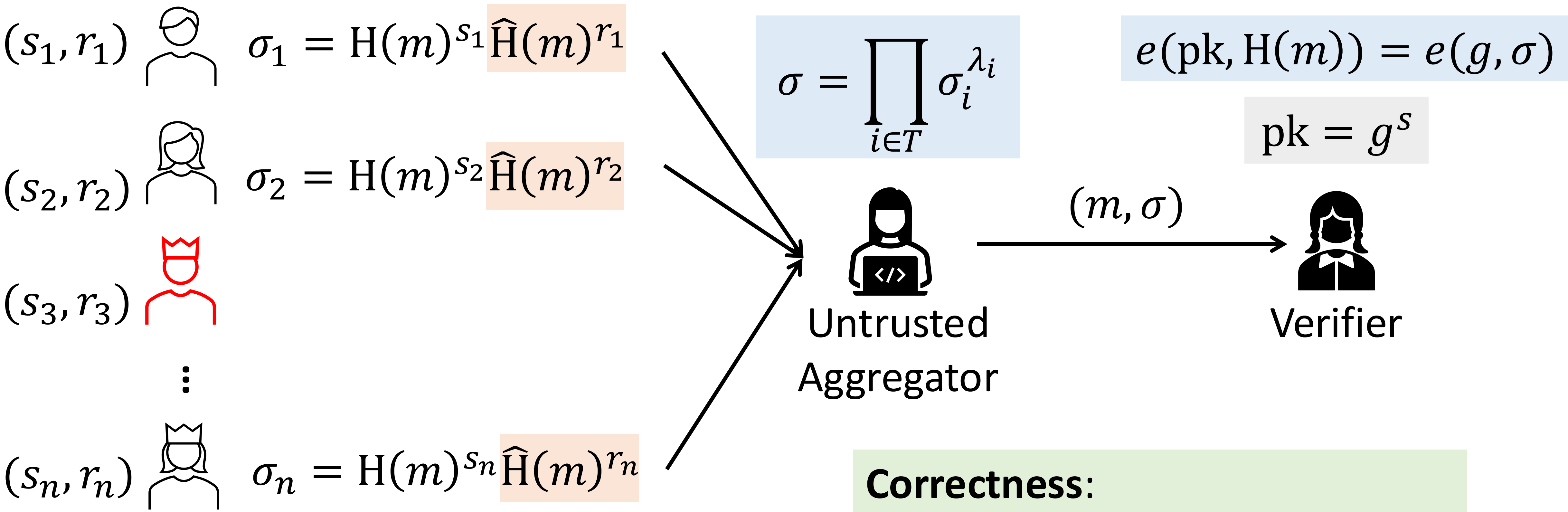
$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

Verifier

Our Approach: Signing

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

Untrusted Aggregator

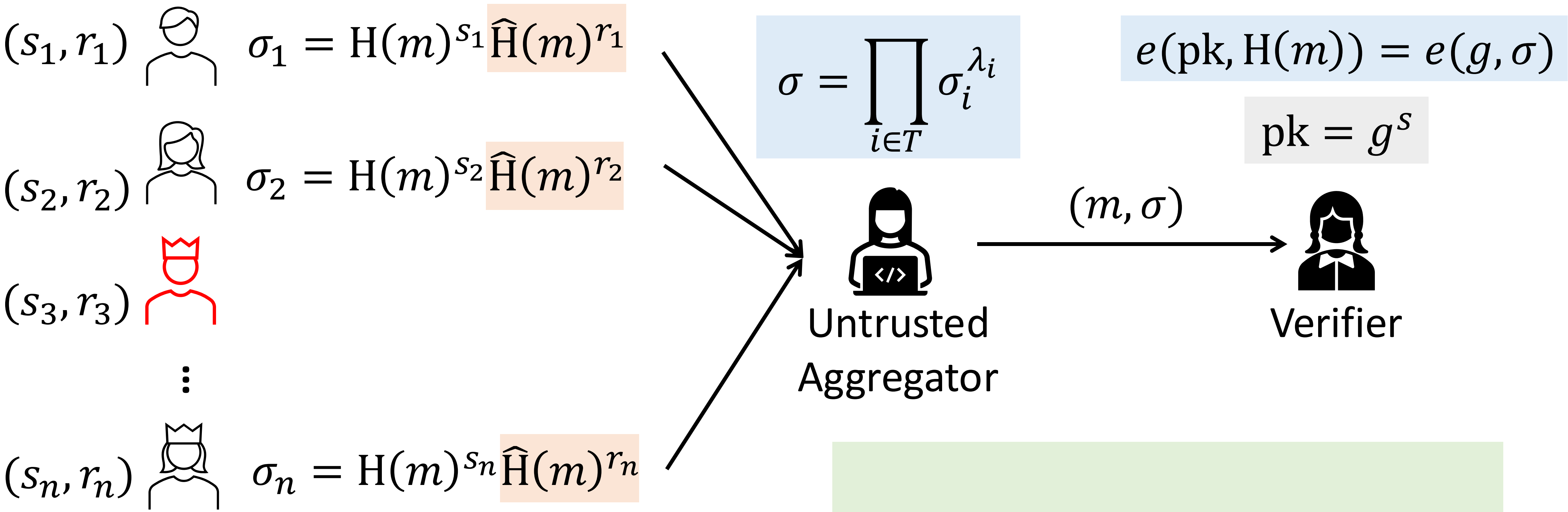
Verifier

Correctness:

$$\sigma = \mathrm{H}(m)^s\widehat{\mathrm{H}}(m)^{r=0} = \mathrm{H}(m)^s$$

Our Approach: Properties

λ_i are the Lagrange coefficients.



(s_1, r_1) $\quad \sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\quad \sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\quad \sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

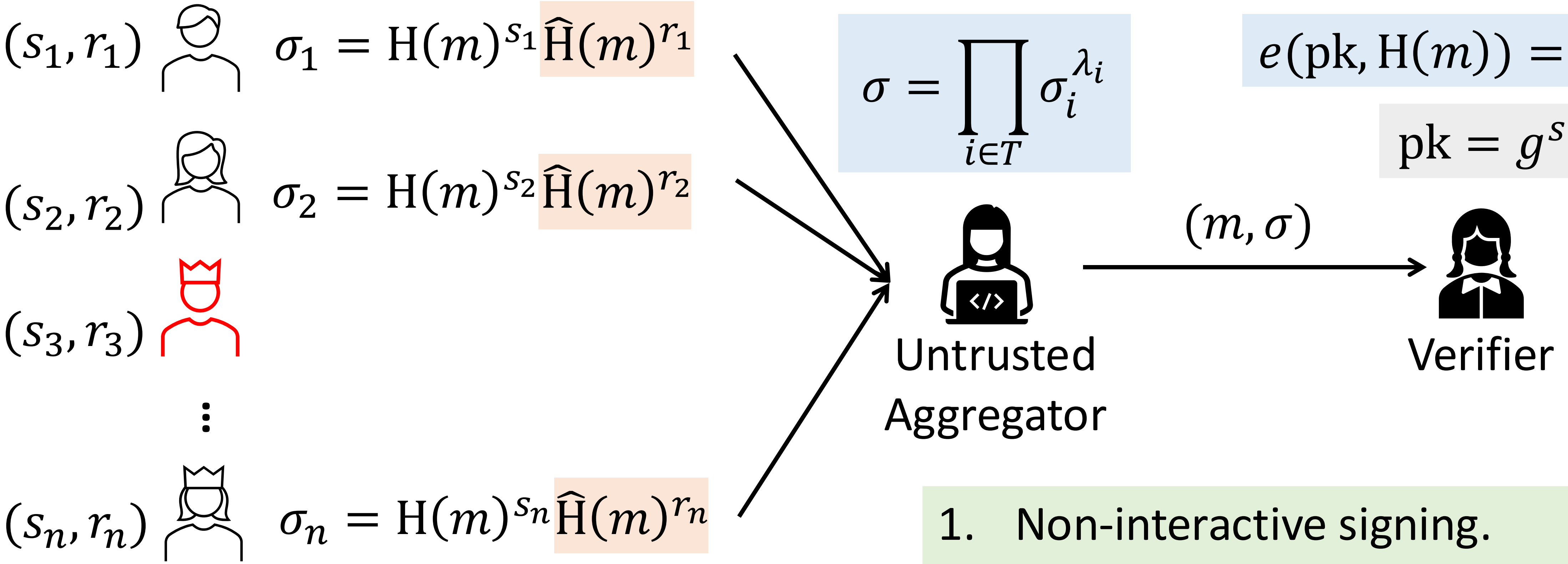
(m, σ)

Untrusted Aggregator

Verifier

Our Approach: Properties

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

(m, σ)

Untrusted Aggregator

Verifier

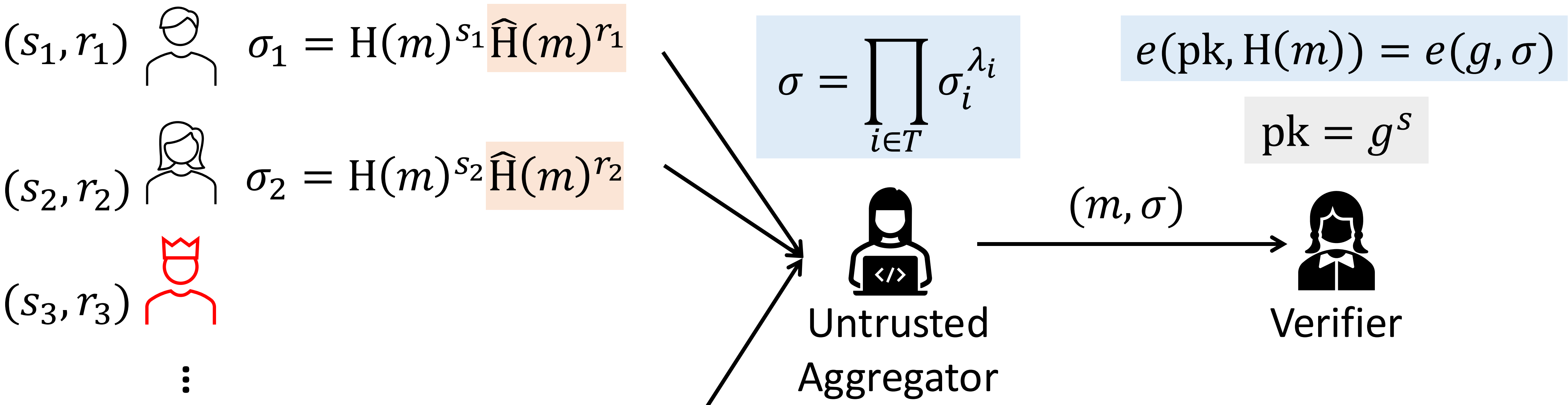
$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

1. Non-interactive signing.

Our Approach: Properties

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

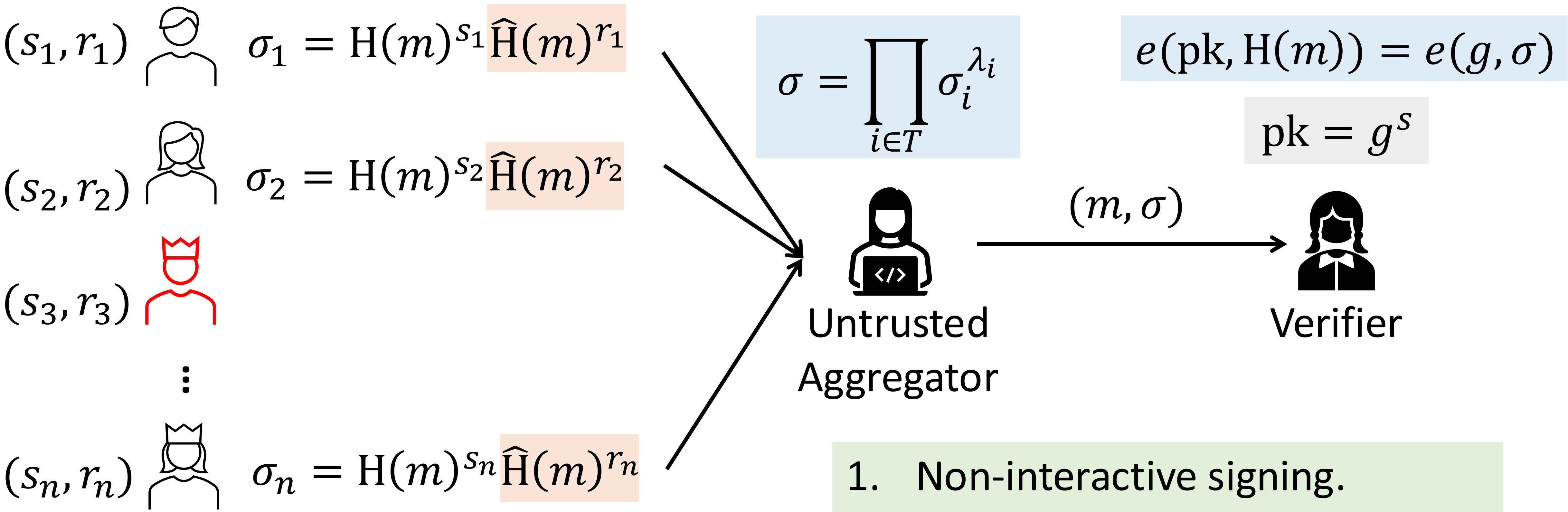
Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.

Our Approach: Properties

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\hat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\hat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\hat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

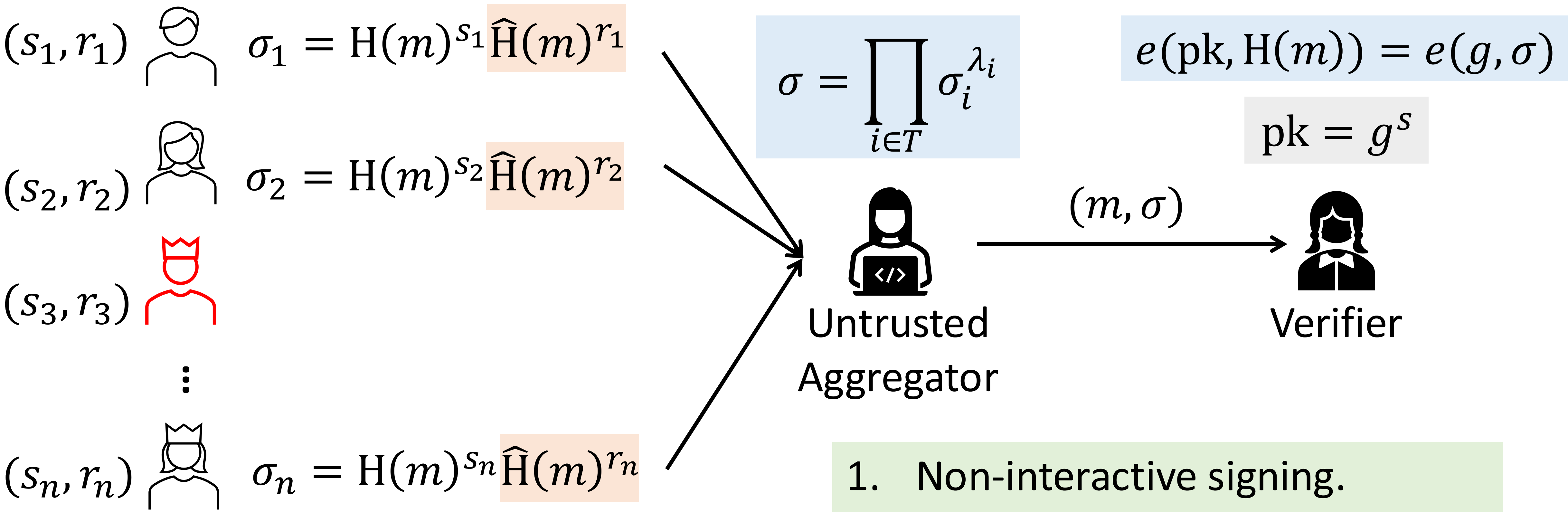
Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.
3. Verification cost: two pairings.

Our Approach: Properties

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

(m, σ)

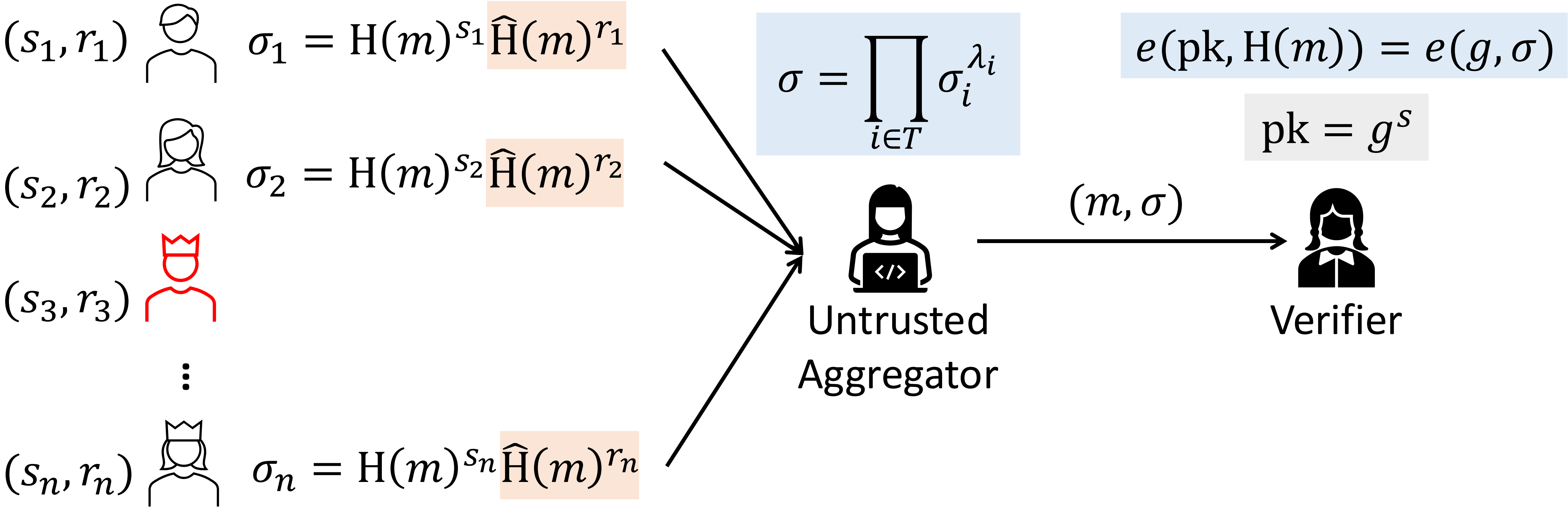
Untrusted Aggregator

Verifier

1. Non-interactive signing.
2. Constant signature size: $\sigma \in \mathbb{G}$.
3. Verification cost: two pairings.
4. Unique signature.

Our Approach: Main Idea

λ_i are the Lagrange coefficients.



(s_1, r_1) $\sigma_1 = \mathrm{H}(m)^{s_1}\widehat{\mathrm{H}}(m)^{r_1}$

(s_2, r_2) $\sigma_2 = \mathrm{H}(m)^{s_2}\widehat{\mathrm{H}}(m)^{r_2}$

(s_3, r_3)

⋮

(s_n, r_n) $\sigma_n = \mathrm{H}(m)^{s_n}\widehat{\mathrm{H}}(m)^{r_n}$

$$\sigma = \prod_{i \in T} \sigma_i^{\lambda_i}$$

Untrusted Aggregator

(m, σ)

$$e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$

$$\mathrm{pk} = g^s$$

Verifier

NOTE: We use Σ-protocol to verify partial signatures.

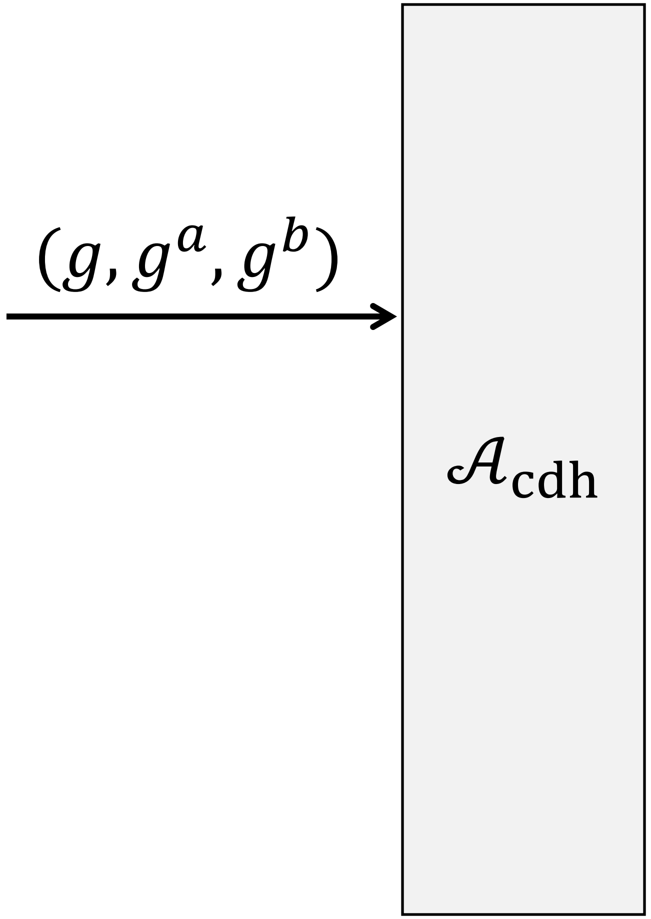
Proof Technique

Proving Security of a Signature Scheme

Proving Security of a Signature Scheme

$\mathcal{A}_{\mathrm{cdh}}$

Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

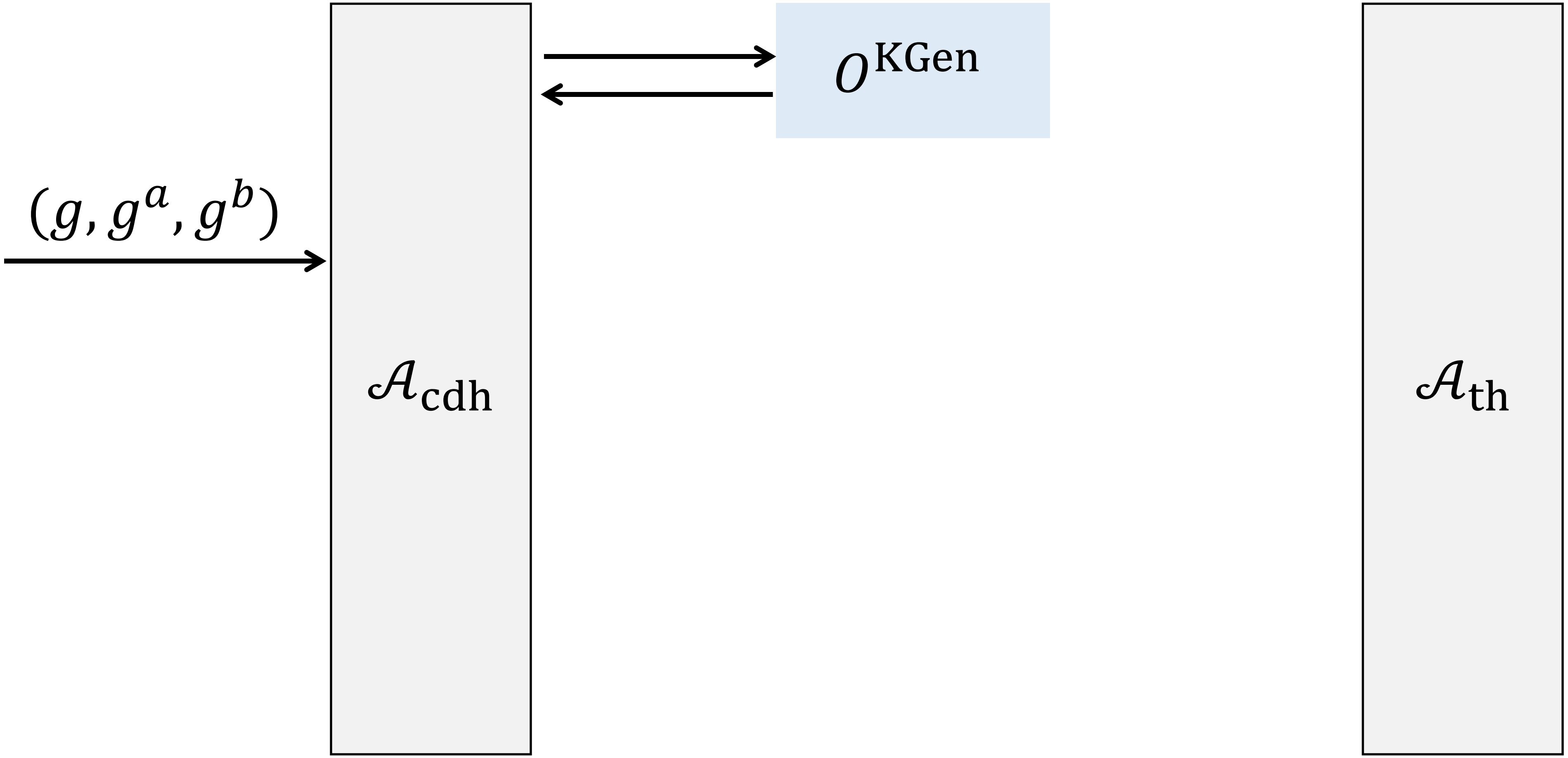
Proving Security of a Signature Scheme

(g, g^a, g^b)

$\mathcal{A}_{\text{cdh}}$

$\mathcal{A}_{\text{th}}$

Proving Security of a Signature Scheme



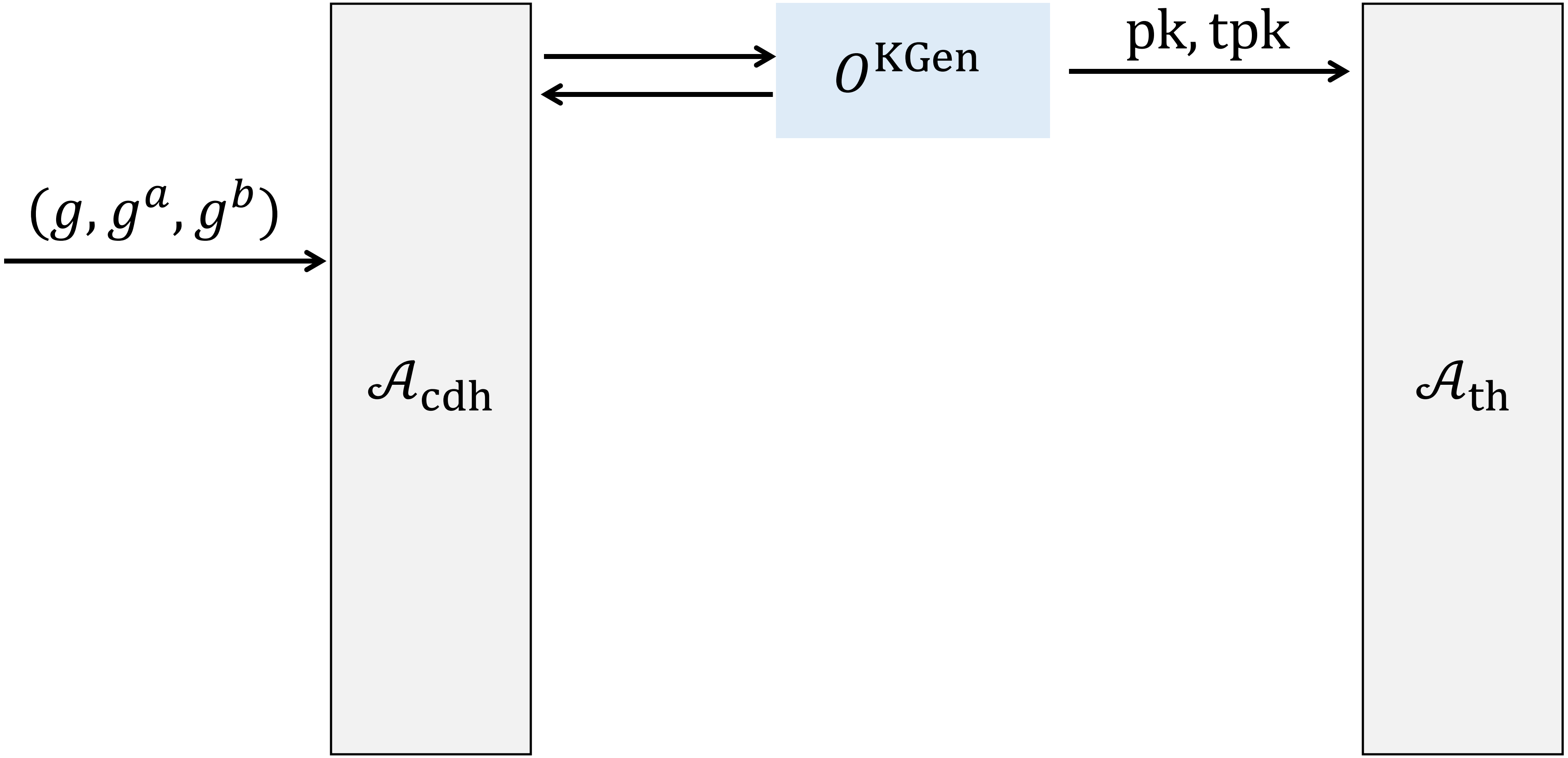
O^{KGen}

(g, g^a, g^b)

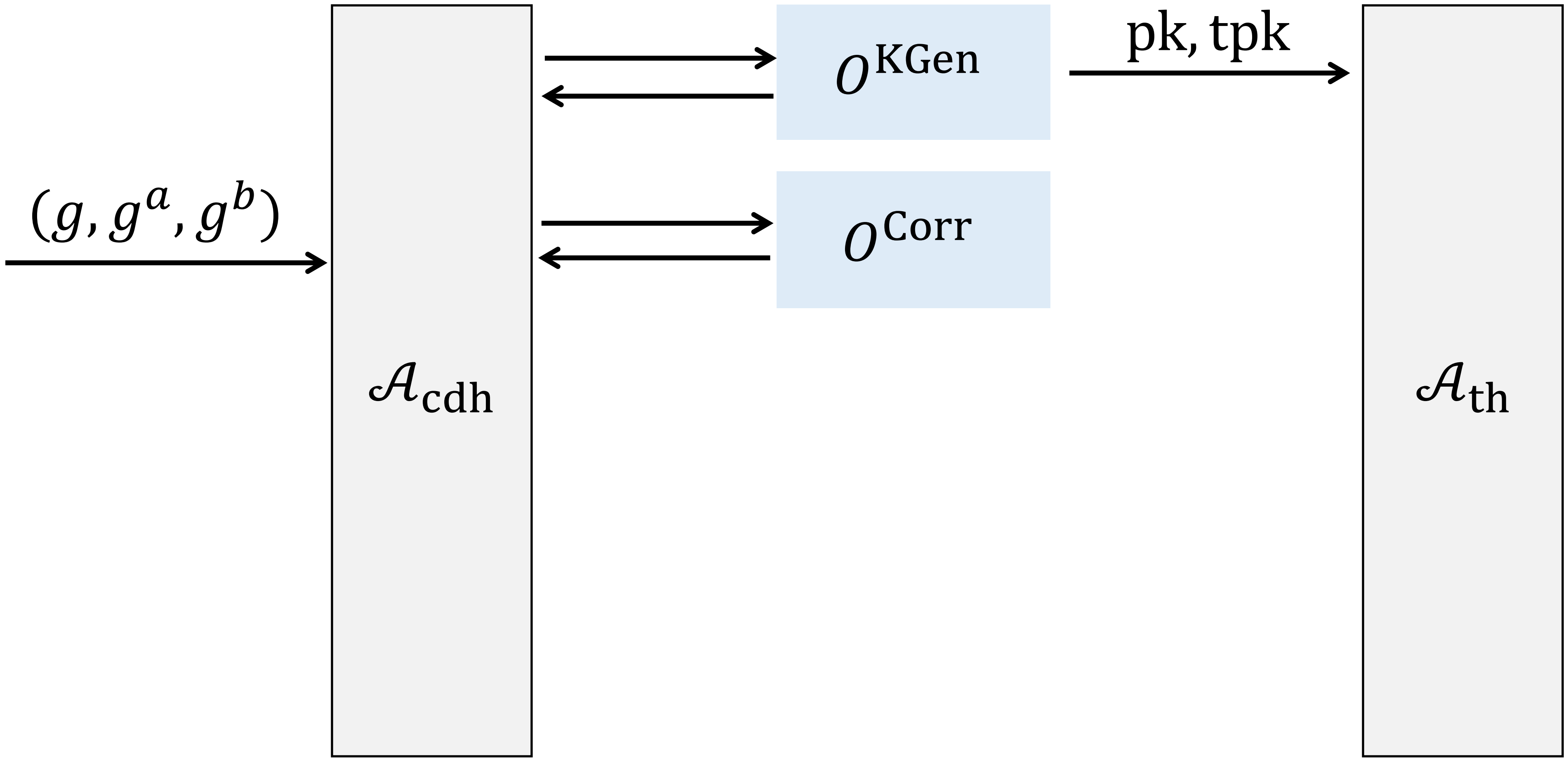
$\mathcal{A}_{\text{cdh}}$

$\mathcal{A}_{\text{th}}$

Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

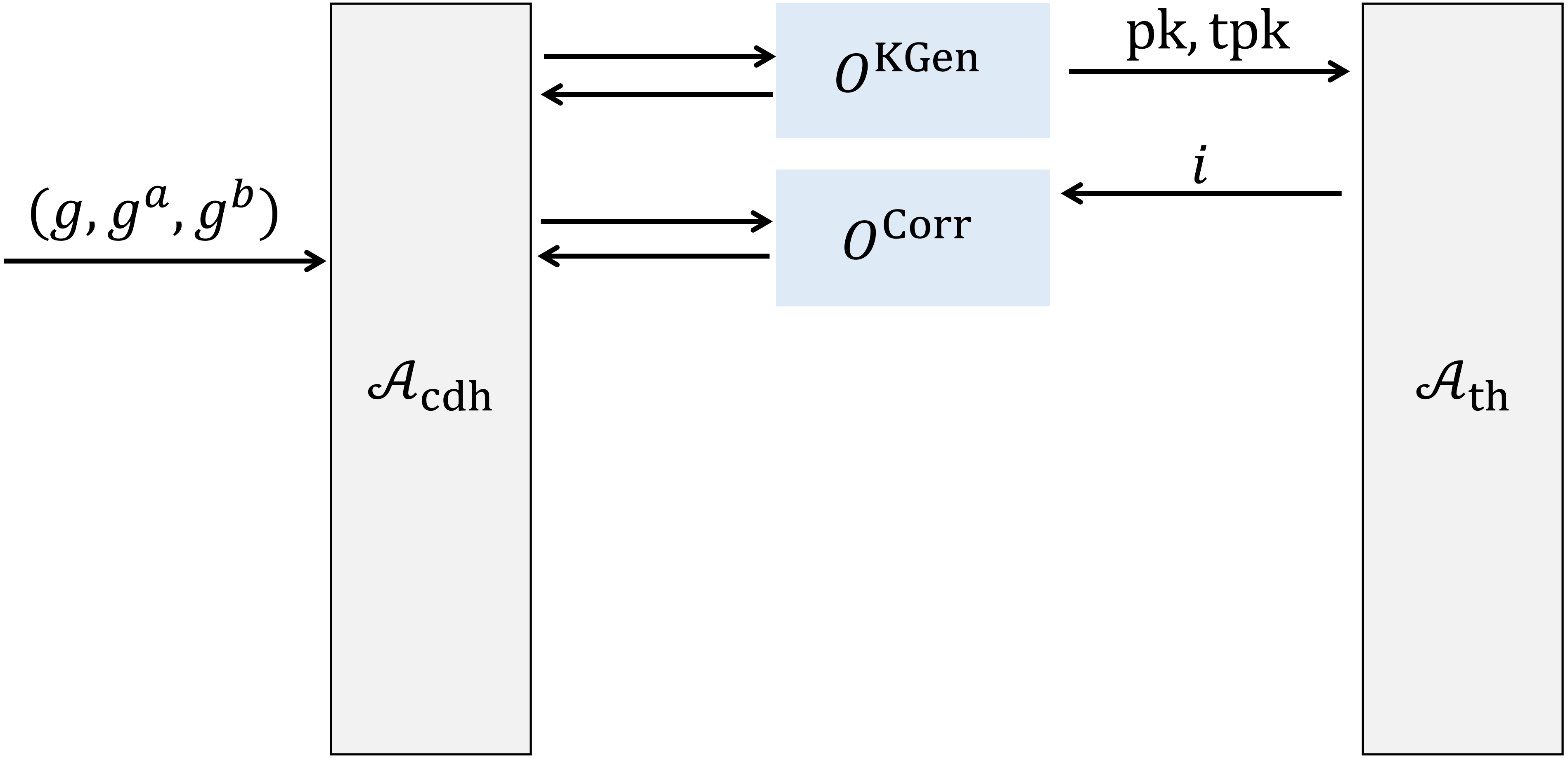
O^{KGen}

O^{Corr}

$\mathrm{pk}, \mathrm{tpk}$

$\mathcal{A}_{\mathrm{th}}$

Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

O^{KGen}

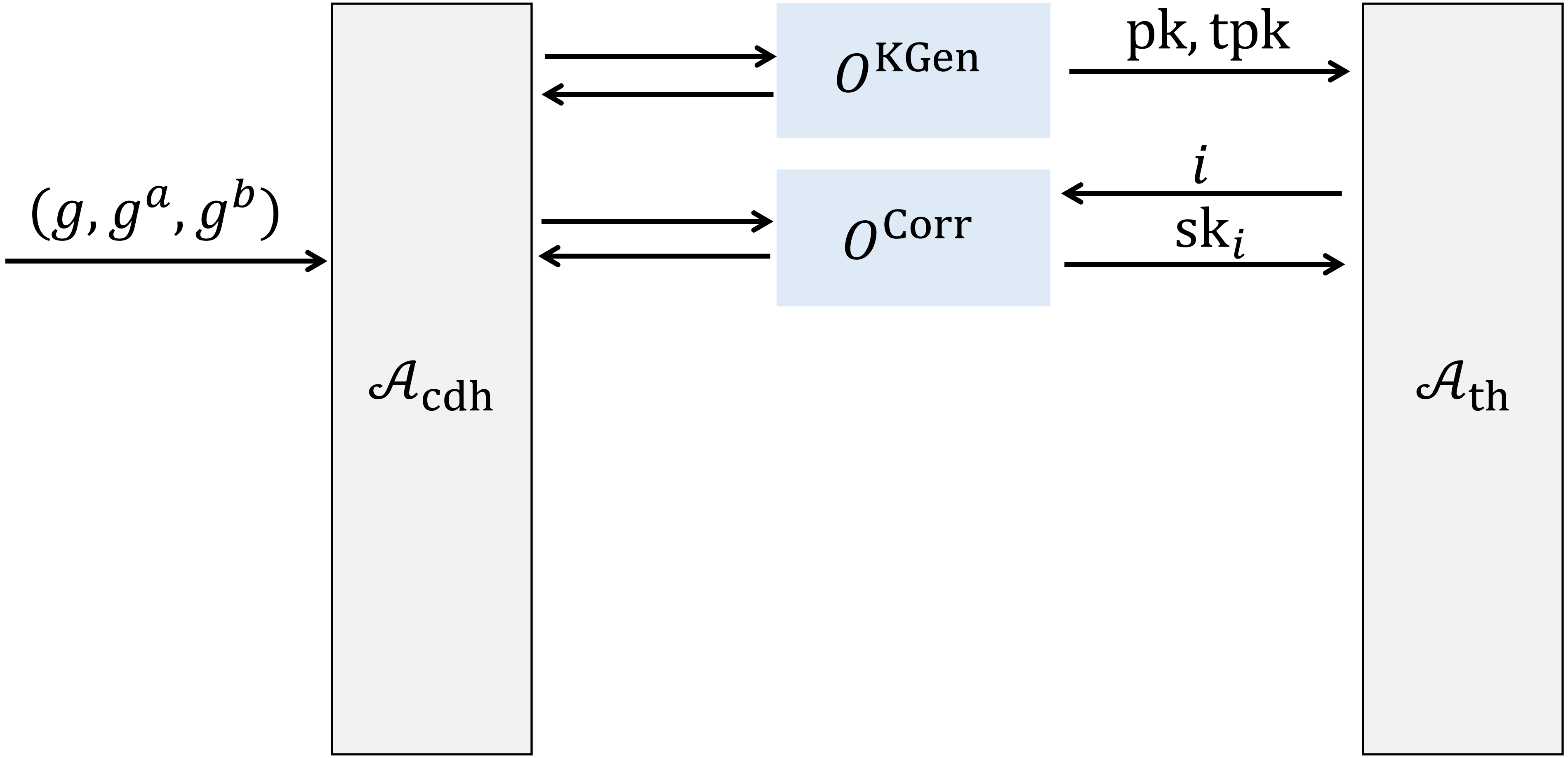
O^{Corr}

$\mathrm{pk}, \mathrm{tpk}$

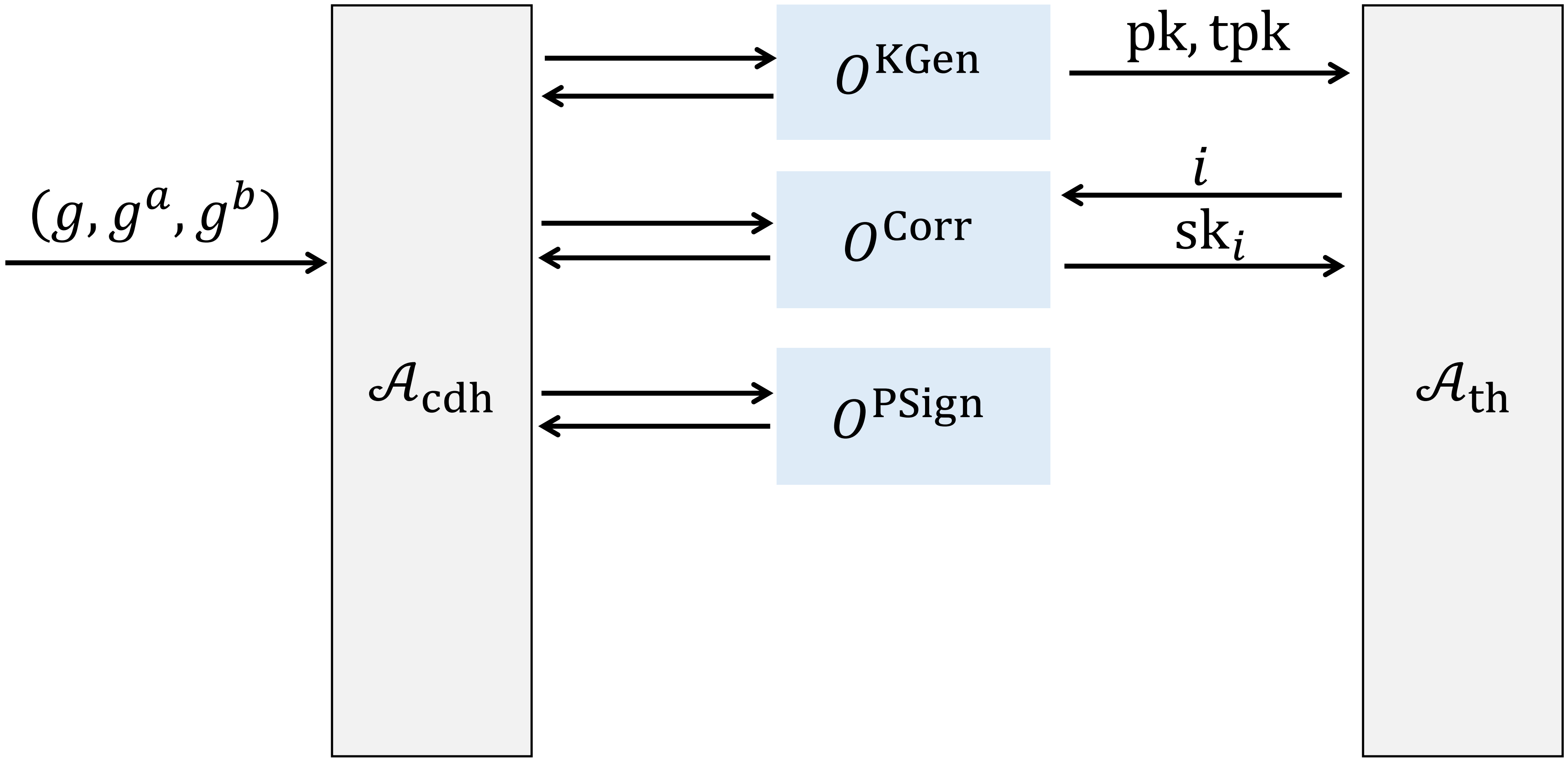
i

$\mathcal{A}_{\mathrm{th}}$

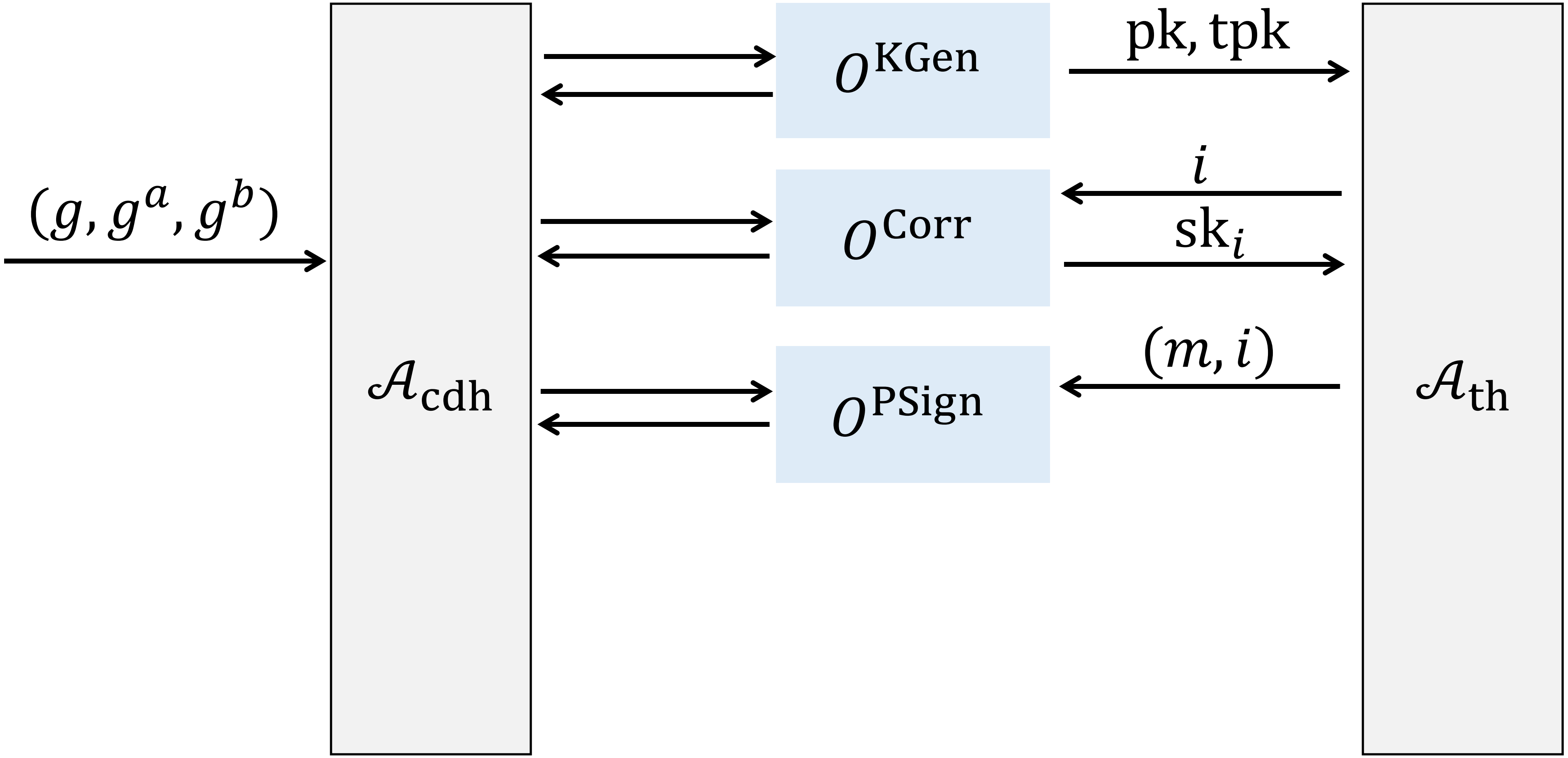
Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

$\mathcal{O}^{\mathrm{KGen}}$

$\mathcal{O}^{\mathrm{Corr}}$

$\mathcal{O}^{\mathrm{PSign}}$

$\mathrm{pk}, \mathrm{tpk}$

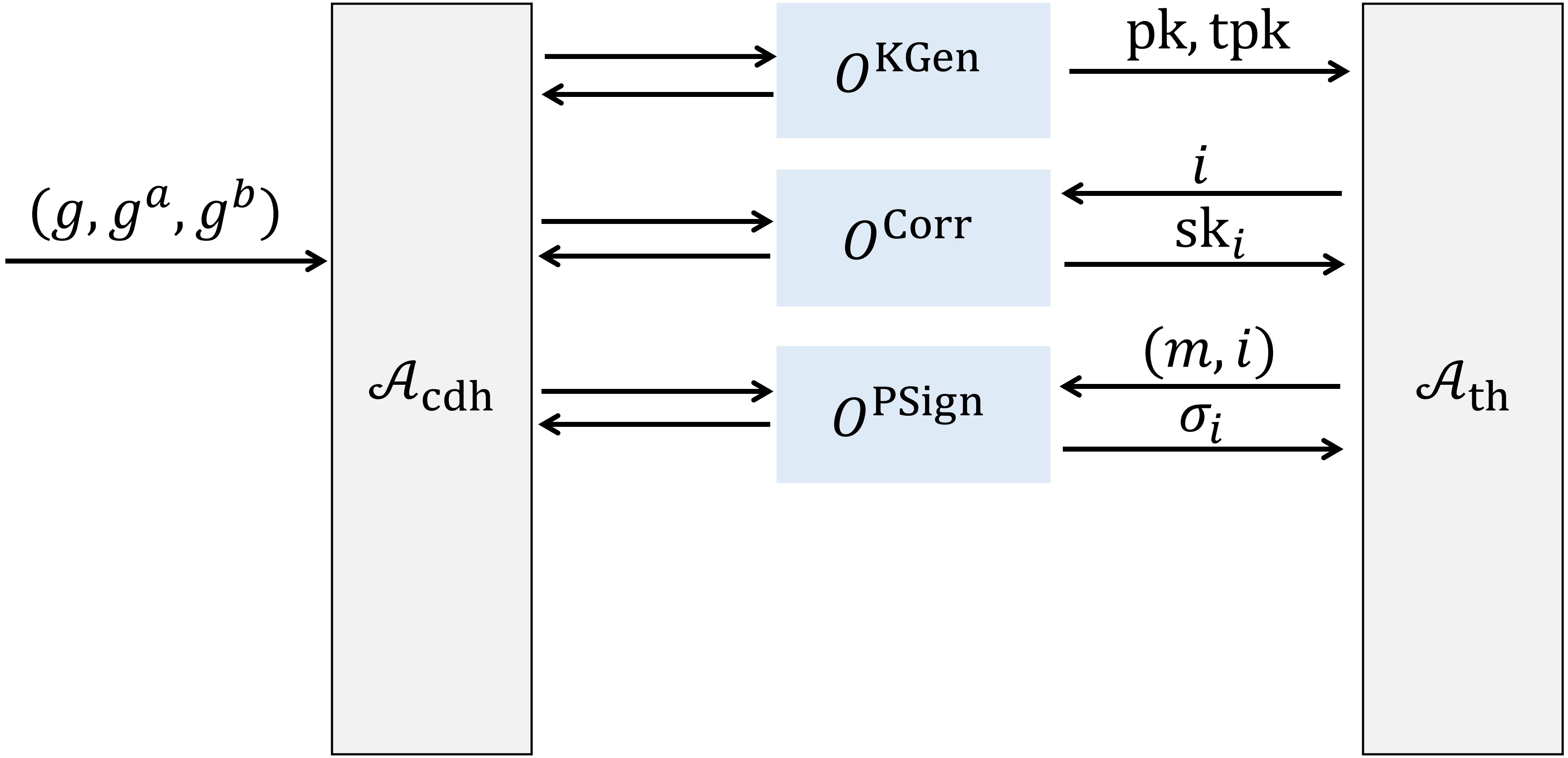
i

sk_i

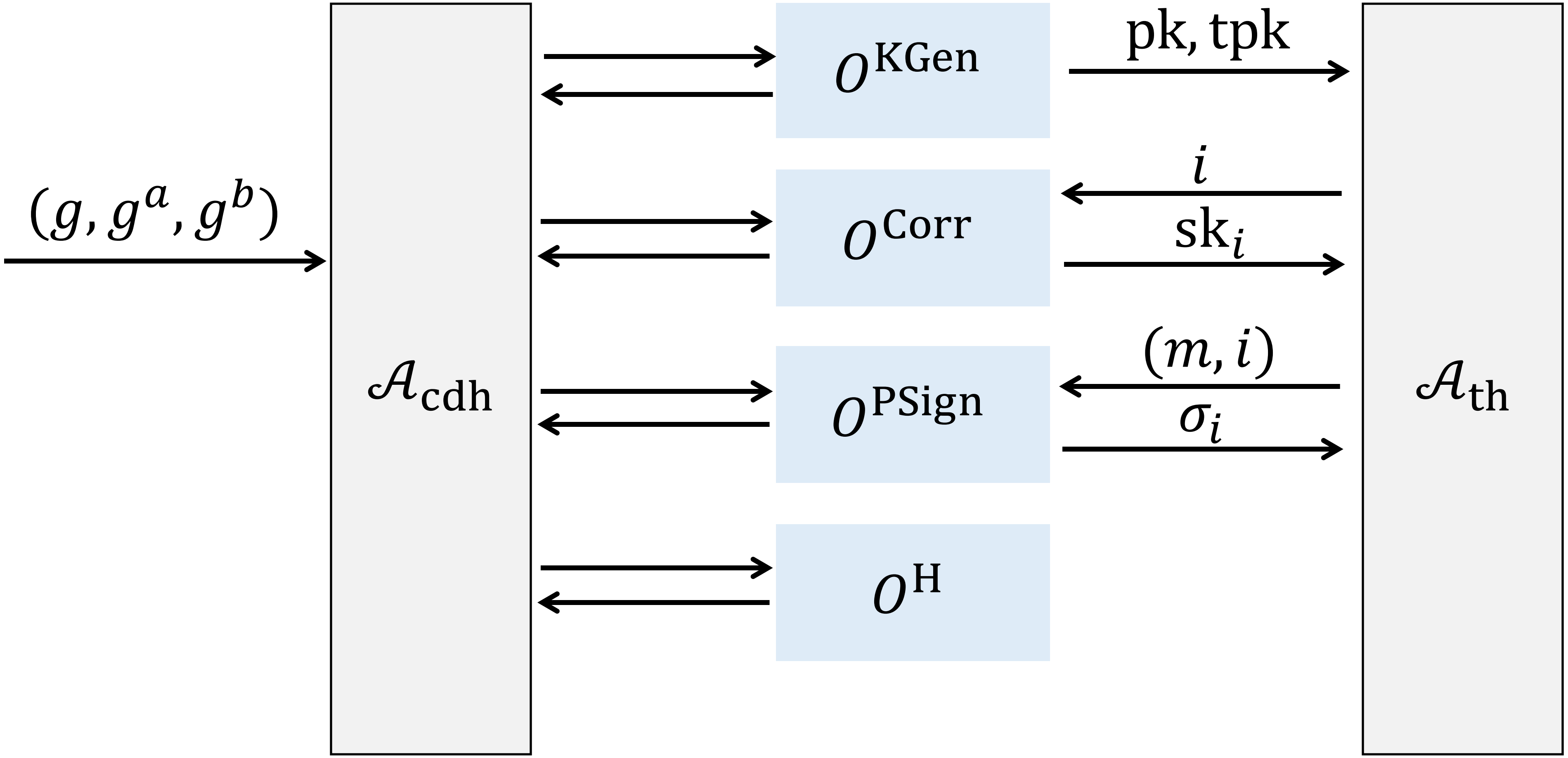
(m, i)

$\mathcal{A}_{\mathrm{th}}$

Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

O^{KGen}

O^{Corr}

O^{PSign}

O^{H}

$\mathrm{pk}, \mathrm{tpk}$

i

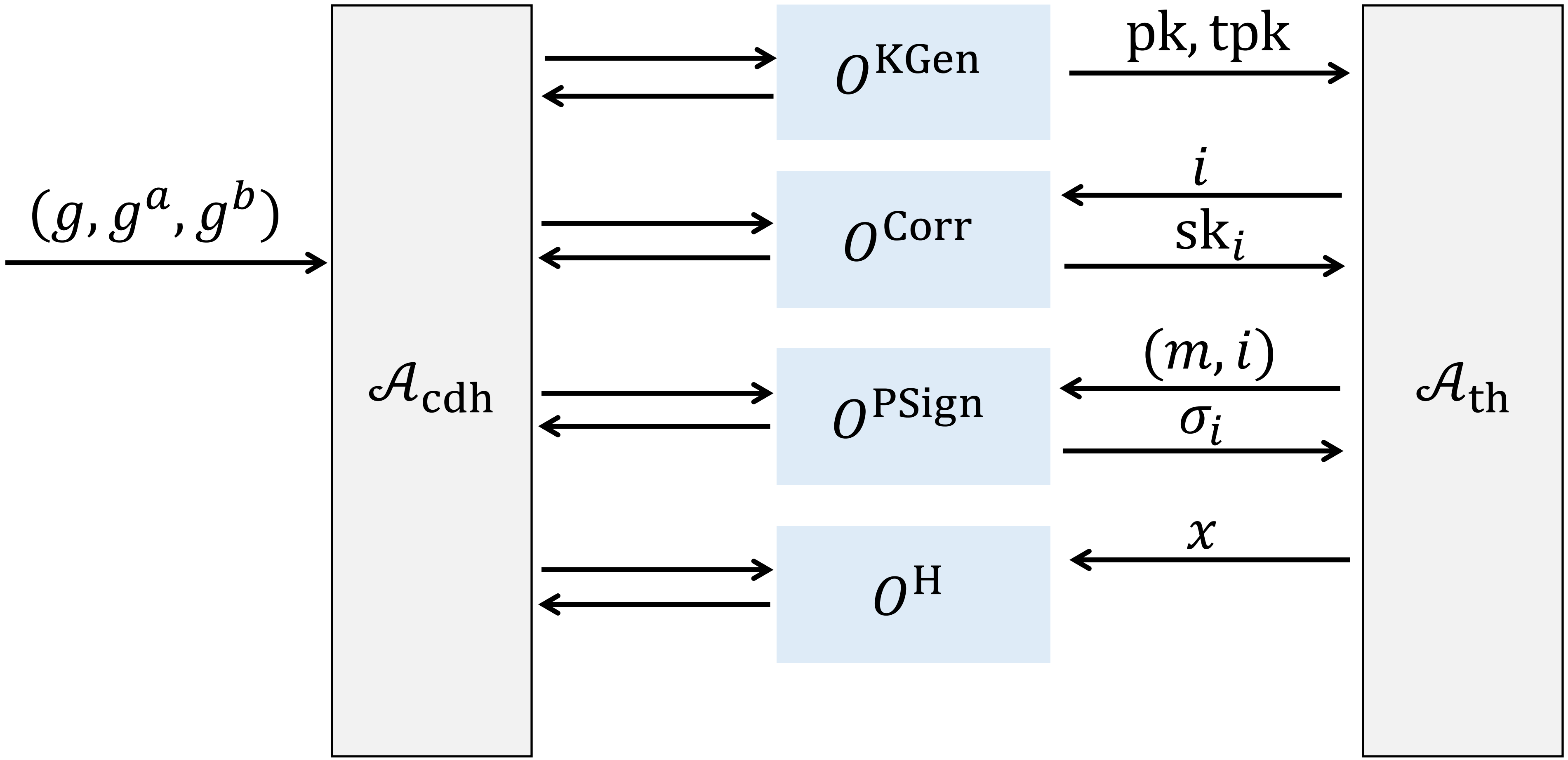
sk_i

(m, i)

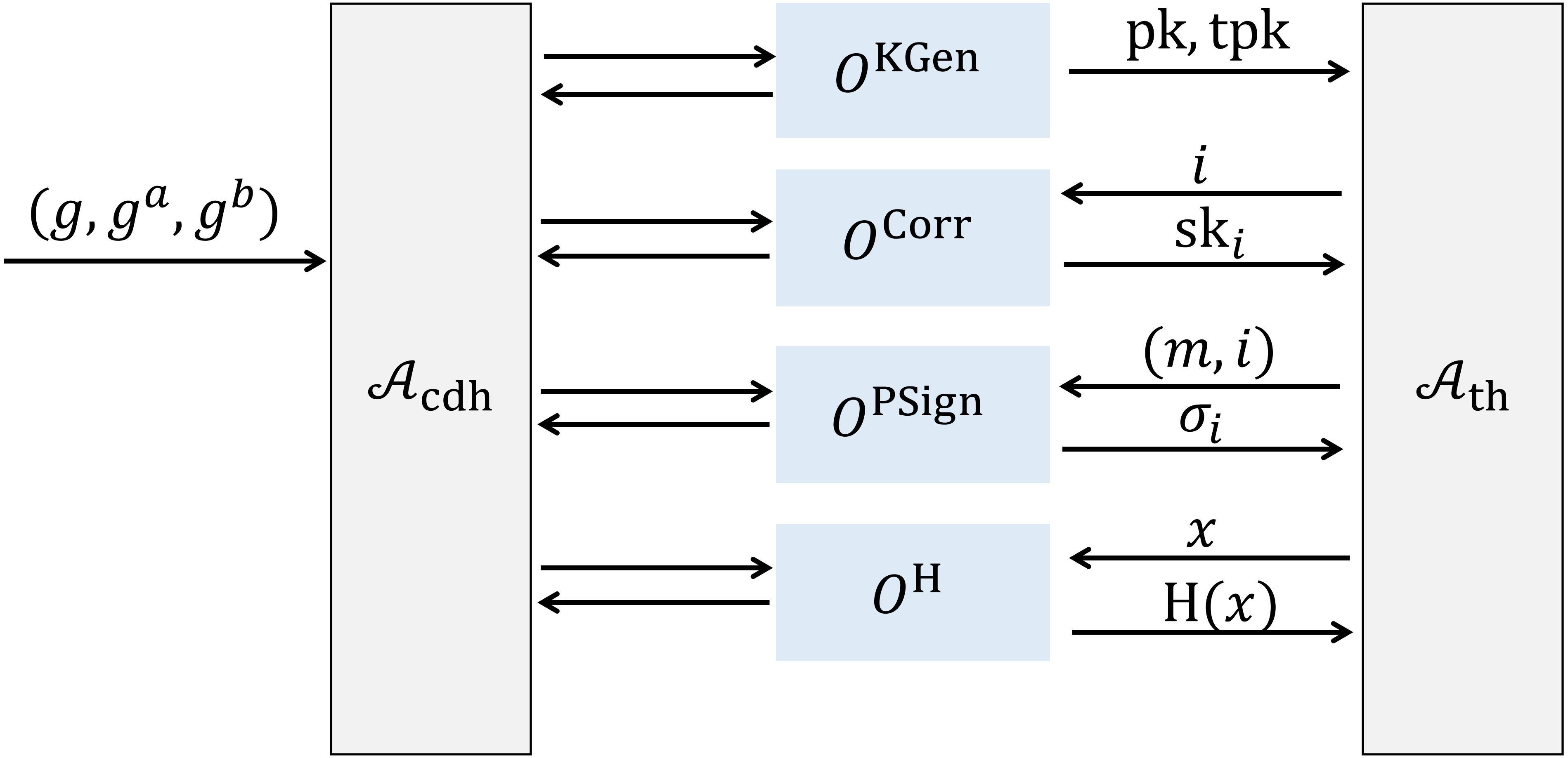
σ_i

$\mathcal{A}_{\mathrm{th}}$

Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



(g, g^a, g^b)

$\mathcal{A}_{\mathrm{cdh}}$

O^{KGen}

$\mathrm{pk}, \mathrm{tpk}$

i

sk_i

O^{Corr}

(m, i)

σ_i

O^{PSign}

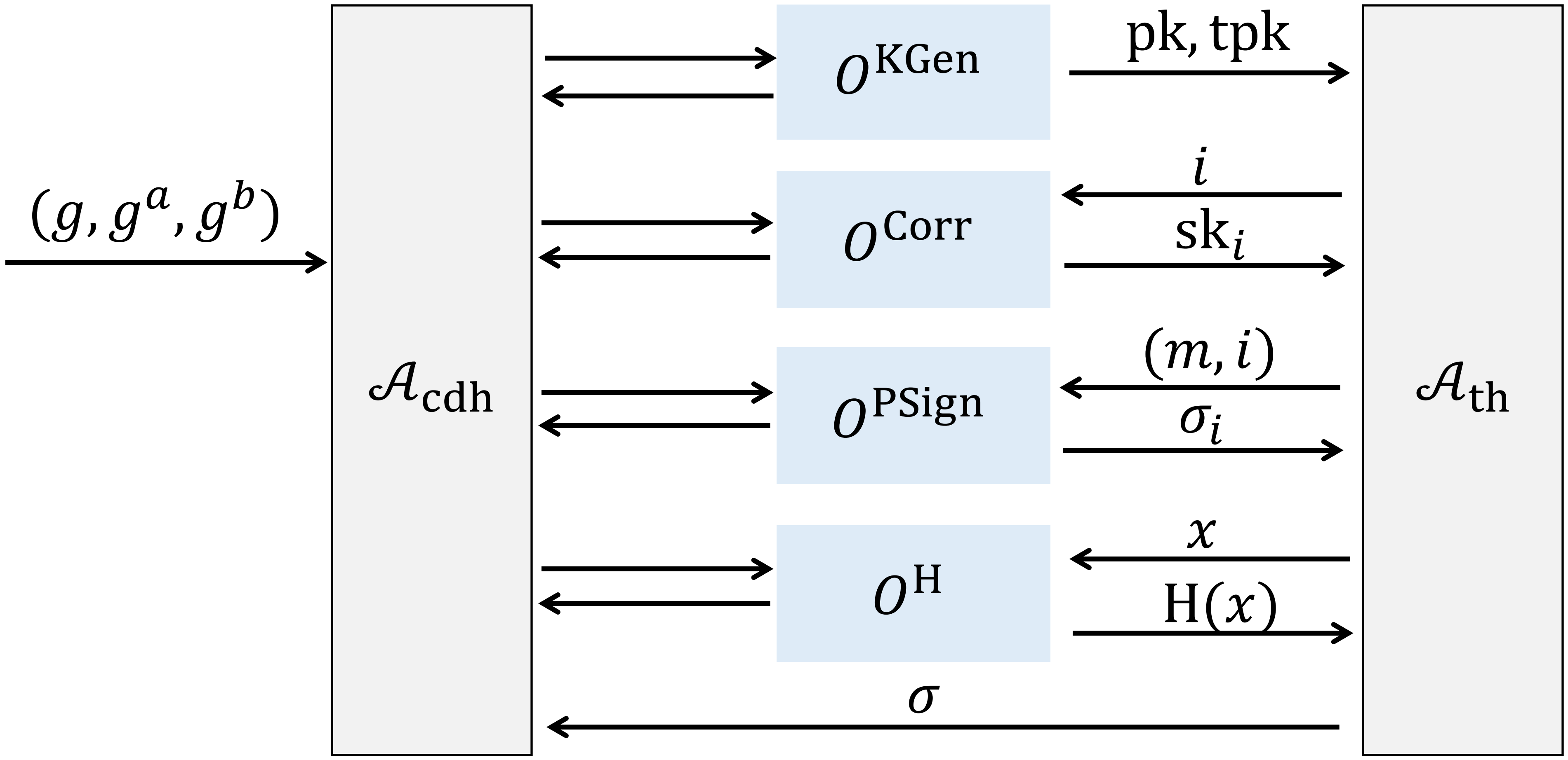
x

$\mathrm{H}(x)$

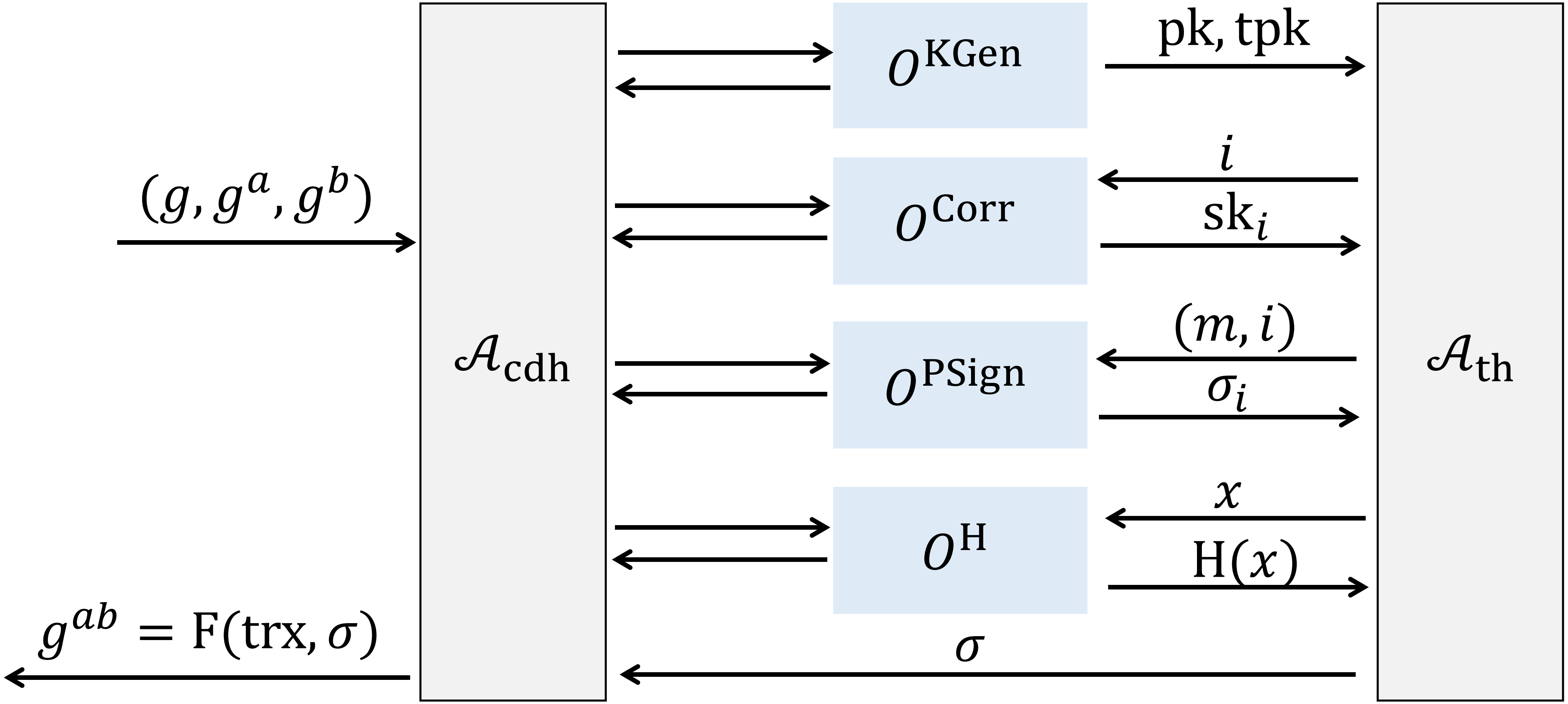
O^{H}

$\mathcal{A}_{\mathrm{th}}$

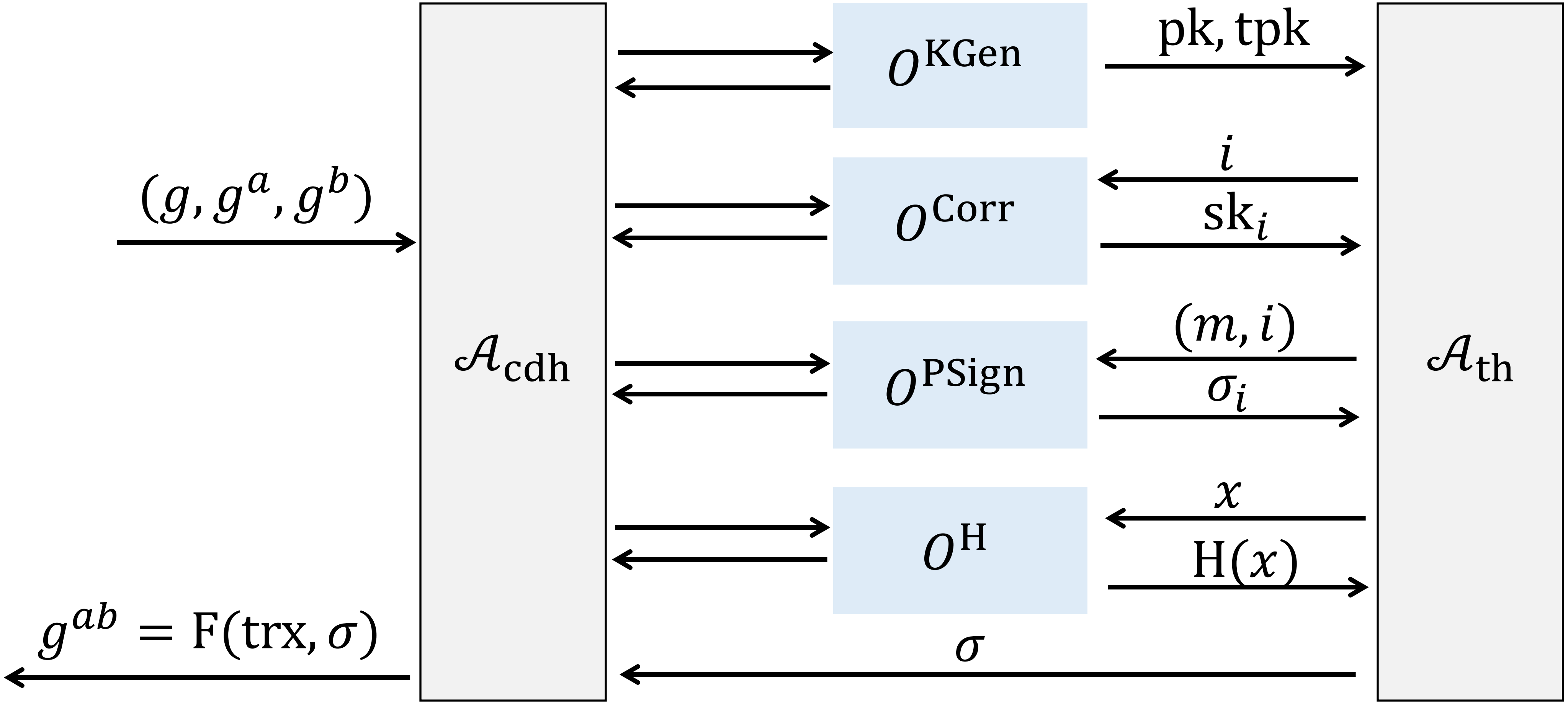
Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



Proving Security of a Signature Scheme



O^{Corr} is the trickiest to simulate. Next, we will see why.

Existing Proof Techniques: Breaking CDH

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\mathrm{cdh}}$

$\mathcal{A}_{\mathrm{th}}$

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

$\mathcal{A}_{\mathrm{th}}$

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\mathrm{cdh}}$

$\mathcal{A}_{\mathrm{th}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

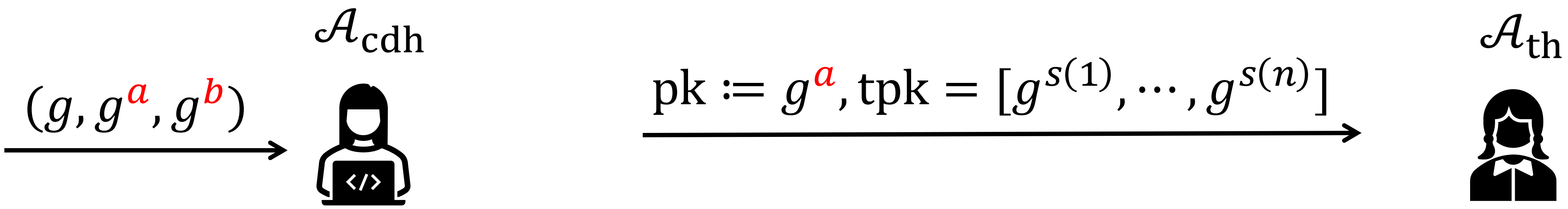
$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$

Existing Proof Techniques: Breaking CDH

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\mathrm{sk} \coloneqq a = s(0); \quad \mathrm{sk}_i = s(i)$$

Existing Proof Techniques: Breaking CDH



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

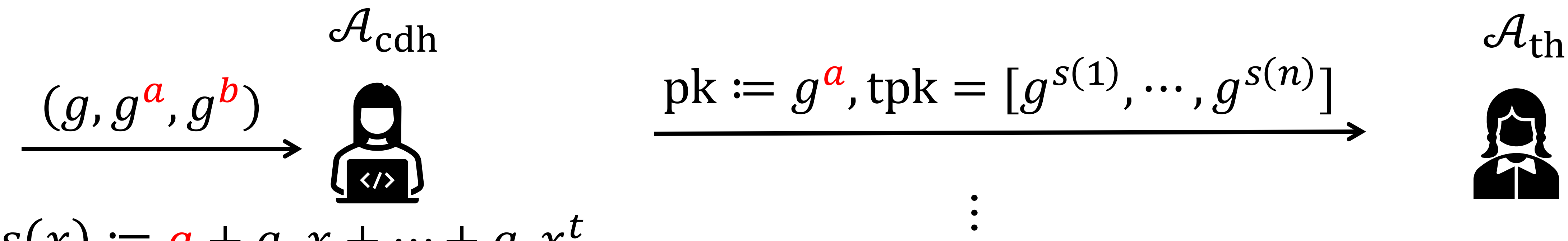
$\text{pk} := g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\mathcal{A}_{\text{th}}$

$$s(x) := a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} := a = s(0); \quad \text{sk}_i = s(i)$$

Existing Proof Techniques: Breaking CDH



$\mathcal{A}_{\text{cdh}}$

$(g, g^{a}, g^{b}) \longrightarrow$

$\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}] \longrightarrow$

$\vdots$

$\mathcal{A}_{\text{th}}$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$

$$s(x) := a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} := a = s(0); \quad \text{sk}_i = s(i)$$

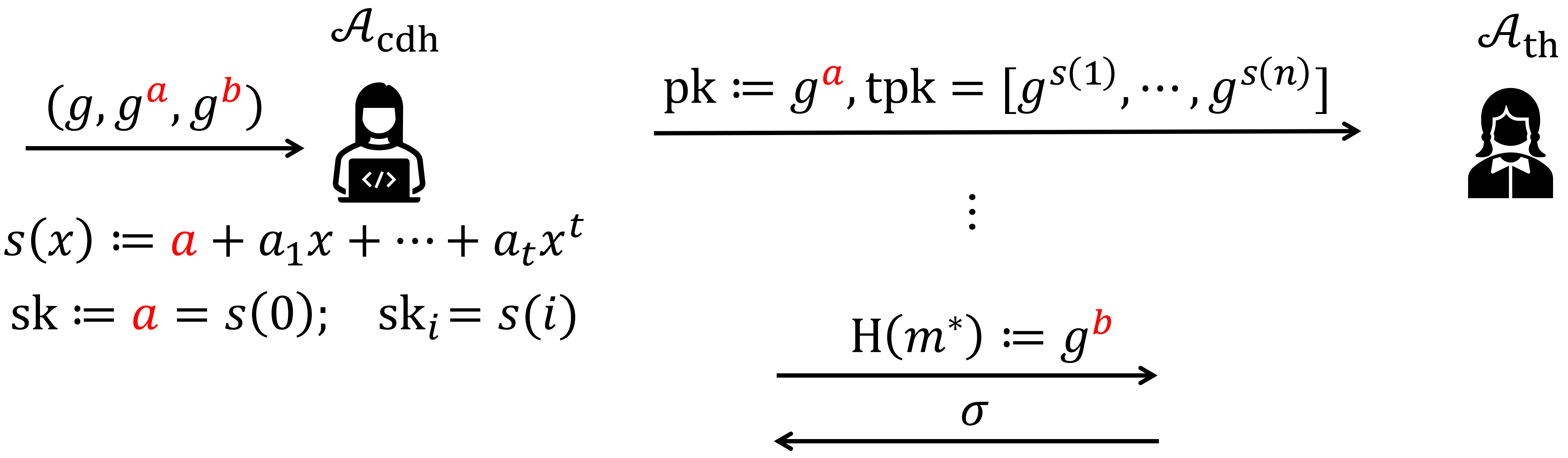
$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} := g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$$

$$\vdots$$

$$\xrightarrow{\text{H}(m^*) := g^{b}}$$

Existing Proof Techniques: Breaking CDH



$\mathcal{A}_{\text{cdh}}$

$\mathcal{A}_{\text{th}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

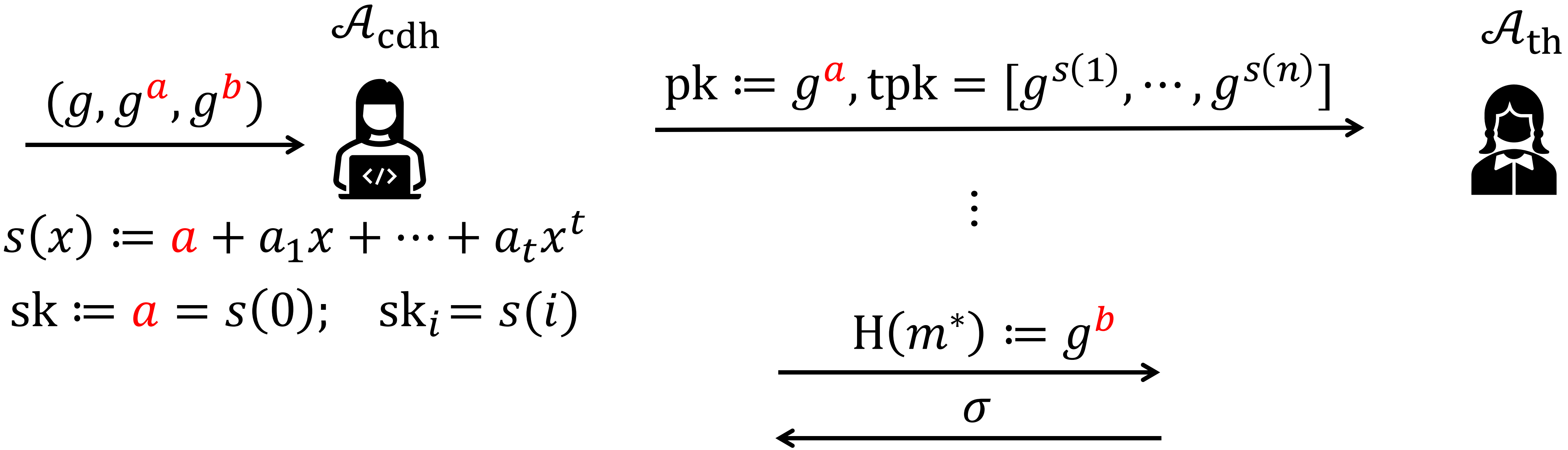
$$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$$

$$\vdots$$

$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

Existing Proof Techniques: Breaking CDH



$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

$\mathcal{A}_{\text{th}}$

$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$

$\vdots$

$\xrightarrow{\mathrm{H}(m^*) \coloneqq g^{b}}$

$\xleftarrow{\sigma}$

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

$$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \dots, g^{s(n)}]}$$

$\mathcal{A}_{\text{th}}$

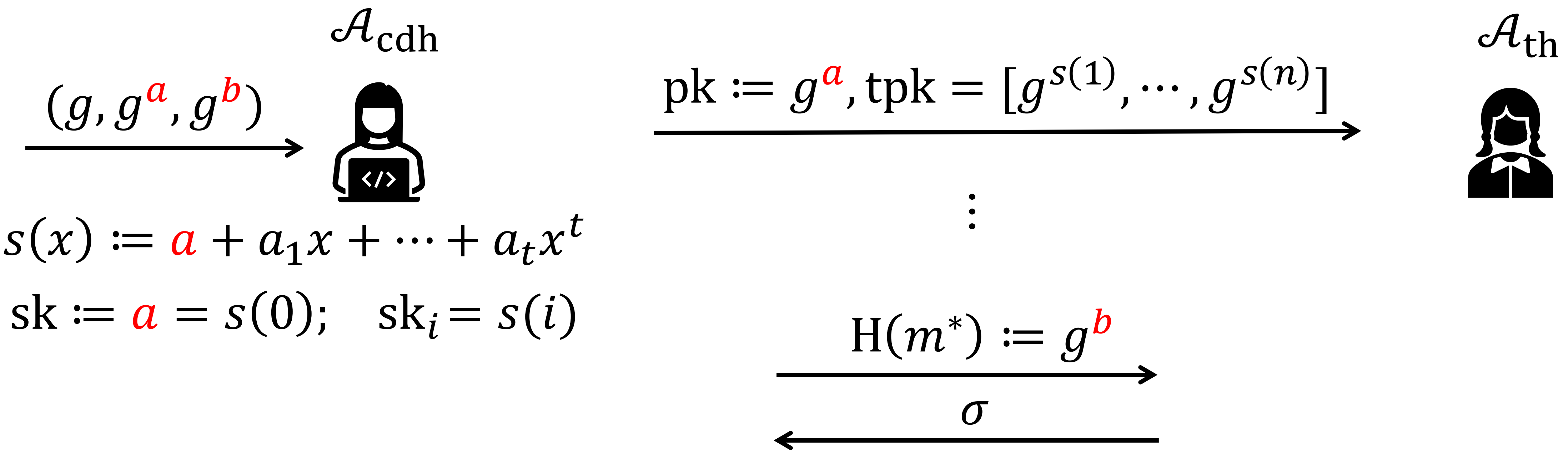
$$\vdots$$

$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

$$e\big(\text{pk}, \text{H}(m^*)\big) = e(g, \sigma)$$

Existing Proof Techniques: Breaking CDH



$\mathcal{A}_{\text{cdh}}$

$\mathcal{A}_{\text{th}}$

(g, g^{a}, g^{b})

$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$

$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$

$\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\vdots$

$\text{H}(m^*) \coloneqq g^{b}$

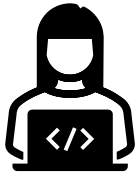
σ

$$e\big(\text{pk}, \text{H}(m^*)\big) = e(g, \sigma)$$
$$\Rightarrow e\big(g^{a}, g^{b}\big) = e(g, \sigma) \Rightarrow \sigma = g^{ab}$$

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\text{cdh}}$

$$(g, g^{a}, g^{b}) \xrightarrow{\qquad\qquad}$$



$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$$

$$\vdots$$

$$\xrightarrow{\mathrm{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

$$e\big(\text{pk}, \mathrm{H}(m^*)\big) = e(g, \sigma)$$
$$\Rightarrow e\big(g^{a}, g^{b}\big) = e(g, \sigma) \Rightarrow \sigma = g^{ab}$$

How does $\mathcal{A}_{\text{cdh}}$ compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$?

Existing Proof Techniques: Breaking CDH

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$$

$$\vdots$$

$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

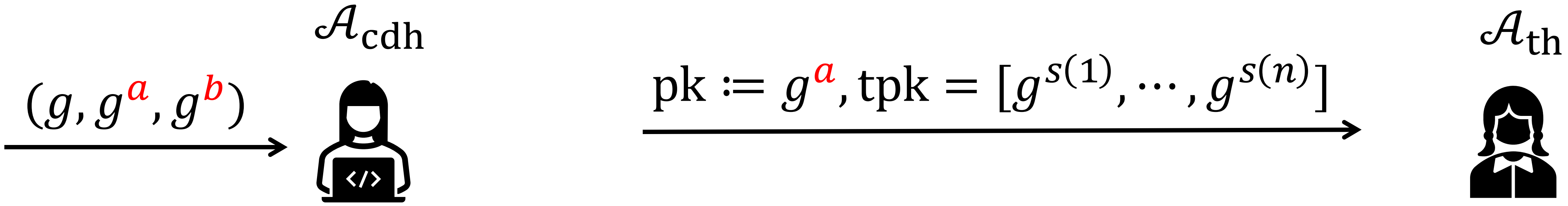
$$e\big(\text{pk}, \text{H}(m^*)\big) = e(g, \sigma)$$
$$\Rightarrow e\big(g^{a}, g^{b}\big) = e(g, \sigma) \Rightarrow \sigma = g^{ab}$$

How does $\mathcal{A}_{\text{cdh}}$ compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$?

How does $\mathcal{A}_{\text{cdh}}$ respond to corruption queries?

Existing Proof Techniques: Corruption queries

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\text{cdh}}$

(g, g^a, g^b)

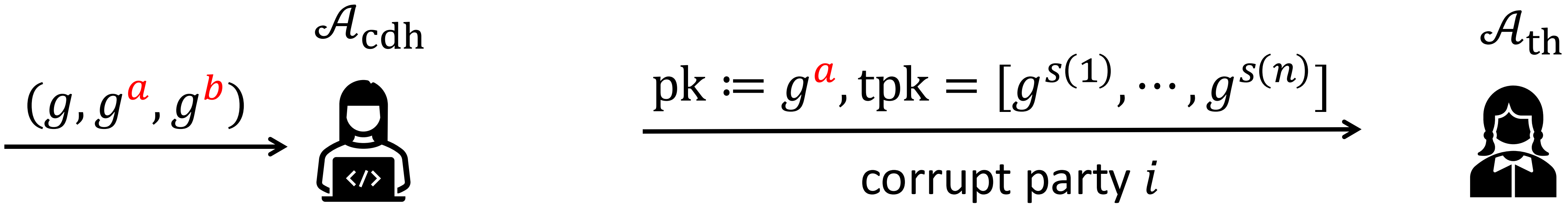
$\text{pk} \coloneqq g^a, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\mathcal{A}_{\text{th}}$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

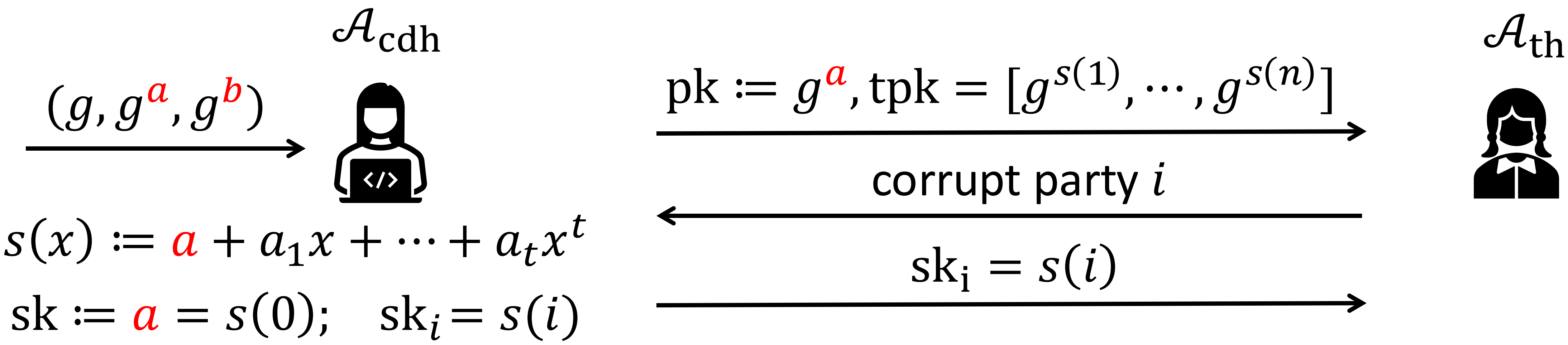
corrupt party i

$\mathcal{A}_{\text{th}}$

$$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$$

$$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$$

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$

$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$

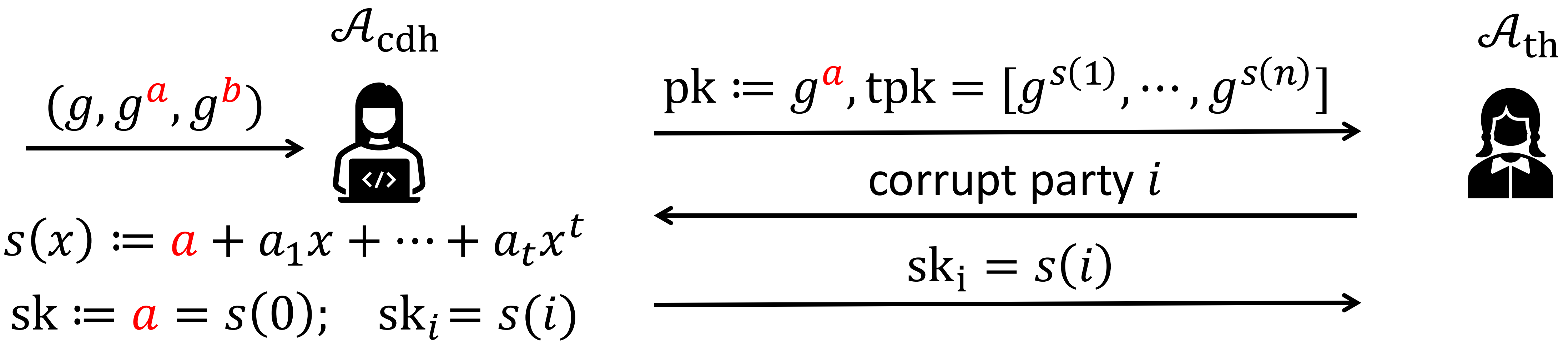
$\mathcal{A}_{\text{th}}$

$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \dots, g^{s(n)}]}$

$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\text{sk}_\text{i} = s(i)}$

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$

$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$

$\mathcal{A}_{\text{th}}$

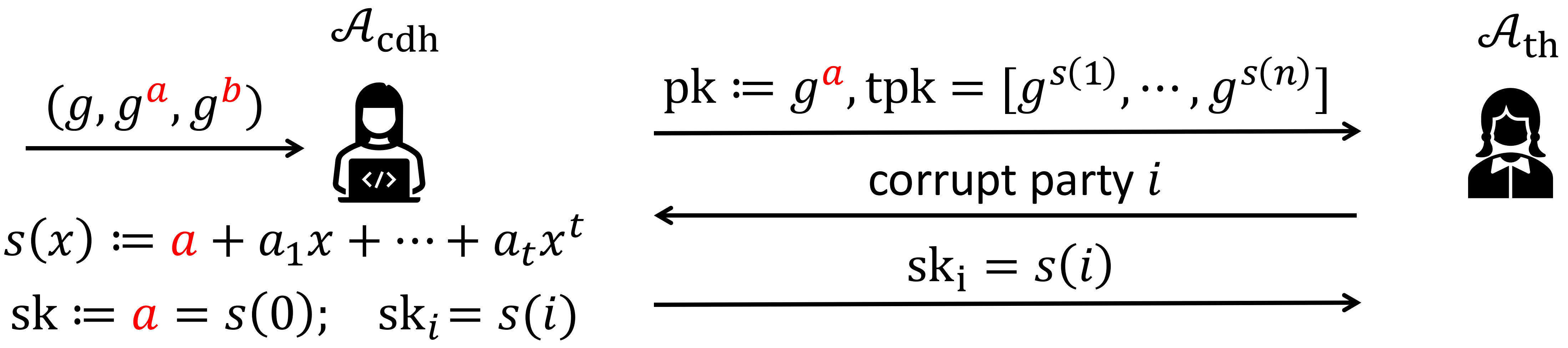
$\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

corrupt party i

$\text{sk}_\text{i} = s(i)$

How?

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$s(x) \coloneqq a + a_1 x + \cdots + a_t x^t$

$\text{sk} \coloneqq a = s(0); \quad \text{sk}_i = s(i)$

$\mathcal{A}_{\text{th}}$

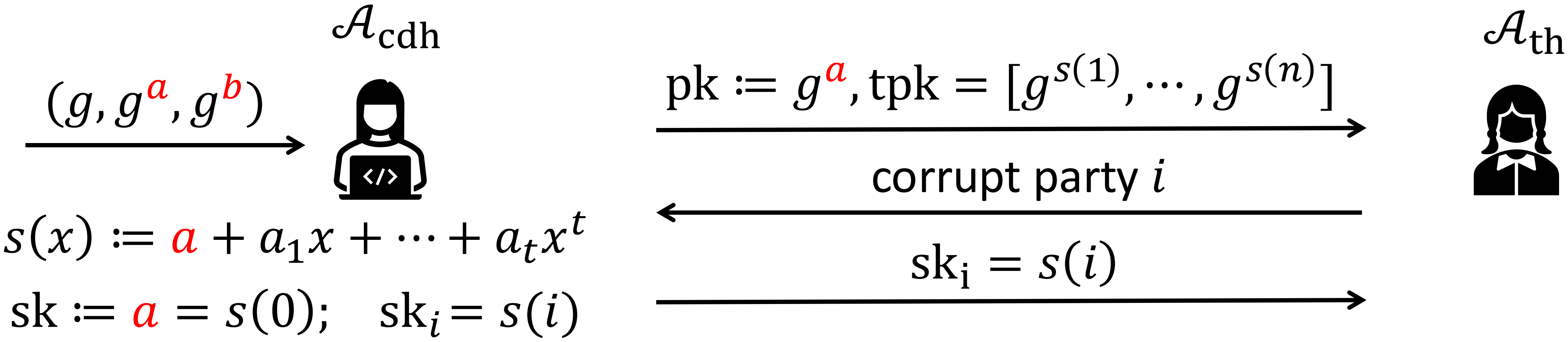
$\xrightarrow{\text{pk} \coloneqq g^{a}, \text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]}$

$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\text{sk}_\text{i} = s(i)}$

How? $\mathcal{A}_{\text{cdh}}$ does not know $s(x)$

Existing Proof Techniques: Corruption queries



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

$s(x) := a + a_1 x + \cdots + a_t x^t$

$\mathrm{sk} := a = s(0); \quad \mathrm{sk}_i = s(i)$

$\mathcal{A}_{\mathrm{th}}$

$\mathrm{pk} := g^{a}, \mathrm{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

corrupt party i

$\mathrm{sk_i} = s(i)$

How? $\mathcal{A}_{\mathrm{cdh}}$ does not know $s(x)$

This is why we need to restrict $\mathcal{A}_{\mathrm{th}}$ to be static.

Existing Proof Techniques: Static Security

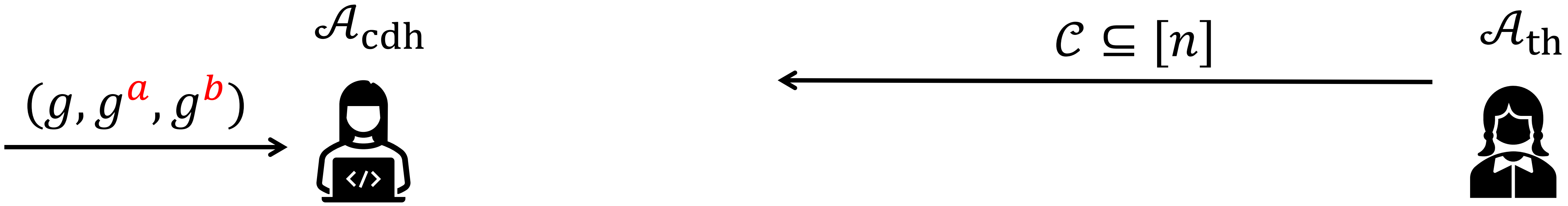
Existing Proof Techniques: Static Security

$\mathcal{A}_{\mathrm{cdh}}$

$\mathcal{A}_{\mathrm{th}}$

$(g, g^{a}, g^{b}) \longrightarrow$

Existing Proof Techniques: Static Security



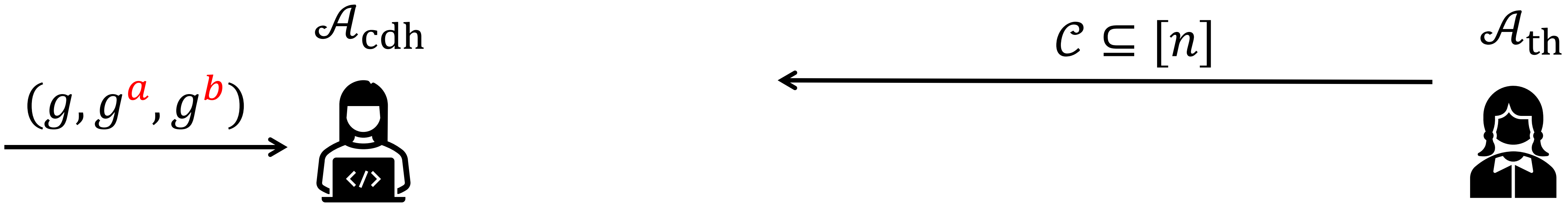
(g, g^{a}, g^{b})

$\mathcal{A}_{\text{cdh}}$

$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\text{th}}$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\mathrm{cdh}}$

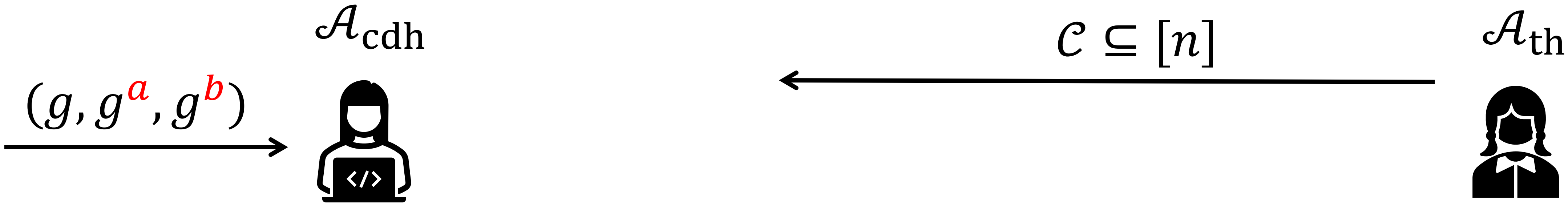
(g, g^{a}, g^{b})

$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\mathrm{th}}$

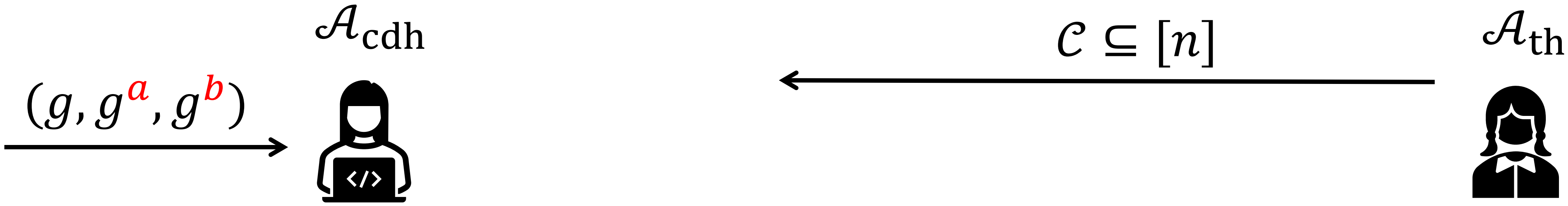
- Let $\mathcal{C} = \{1,2, \ldots, t\}$

Existing Proof Techniques: Static Security



- Let $\mathcal{C} = \{1,2, \dots, t\}$
- Sample $s(1), \dots, s(t) \leftarrow \mathbb{F}$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\text{cdh}}$

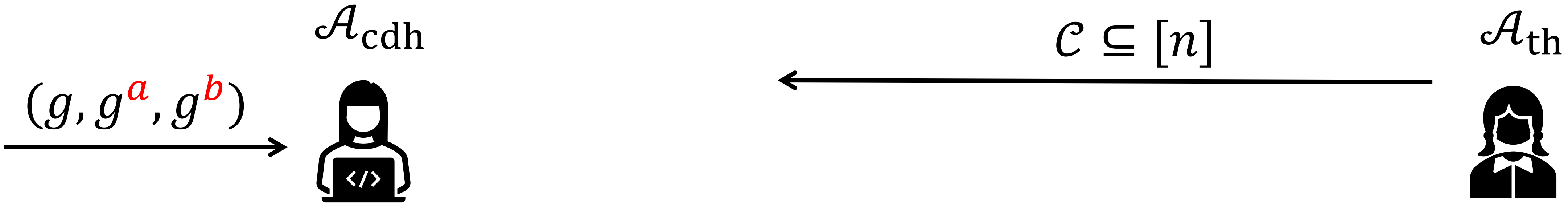
(g, g^{a}, g^{b})

$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2,\dots,t\}$
- Sample $s(1),\dots,s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\text{cdh}}$

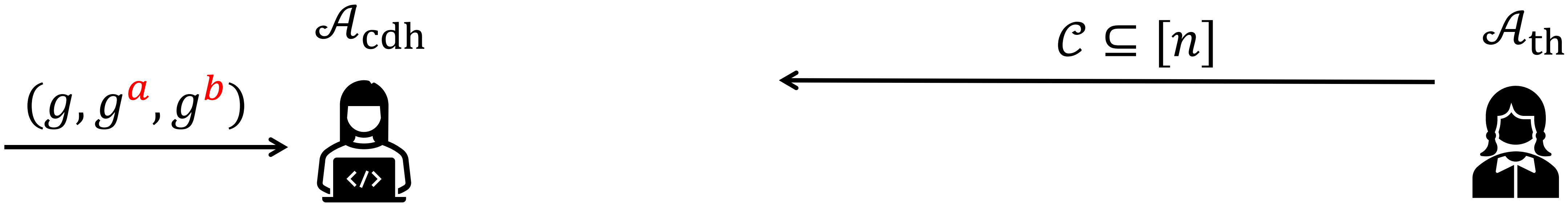
(g, g^{a}, g^{b})

$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2, \ldots, t\}$
- Sample $s(1), \ldots, s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$
- Interpolate $\{g^{a}, g^{s(1)}, \cdots, g^{s(t)}\}$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\text{cdh}}$

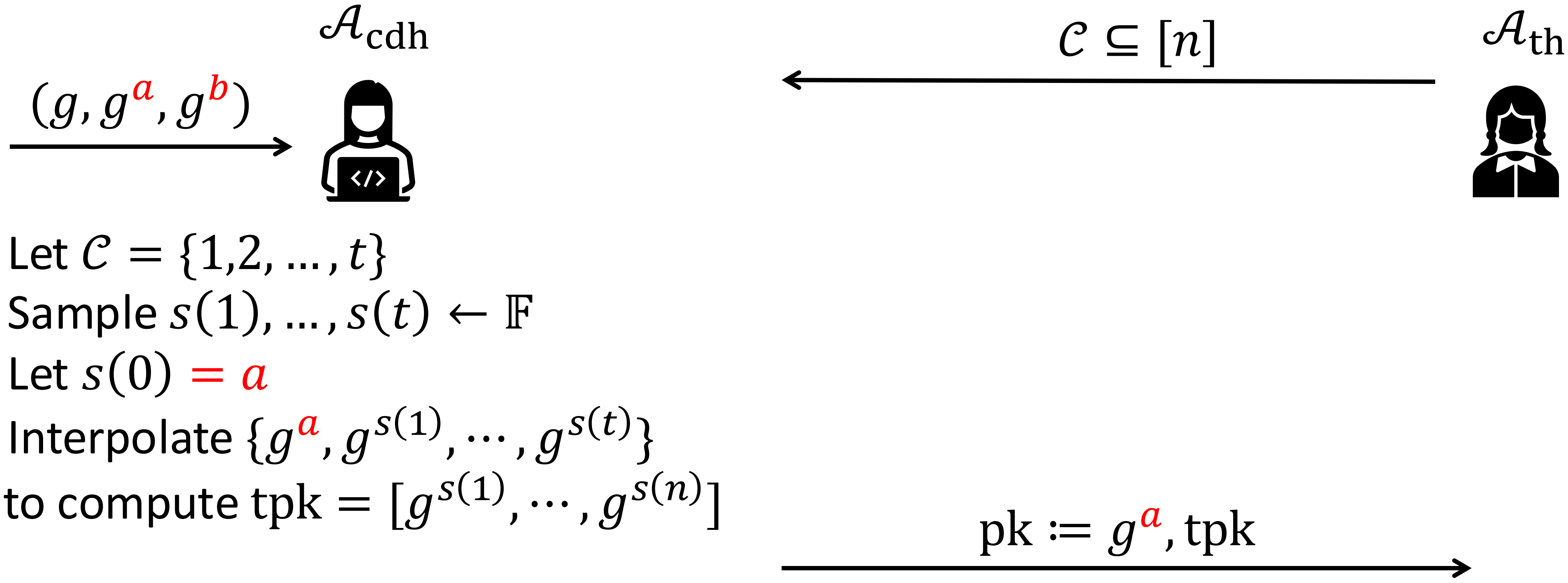
(g, g^{a}, g^{b})

$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2, \dots, t\}$
- Sample $s(1), \dots, s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$
- Interpolate $\{g^{a}, g^{s(1)}, \cdots, g^{s(t)}\}$ to compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

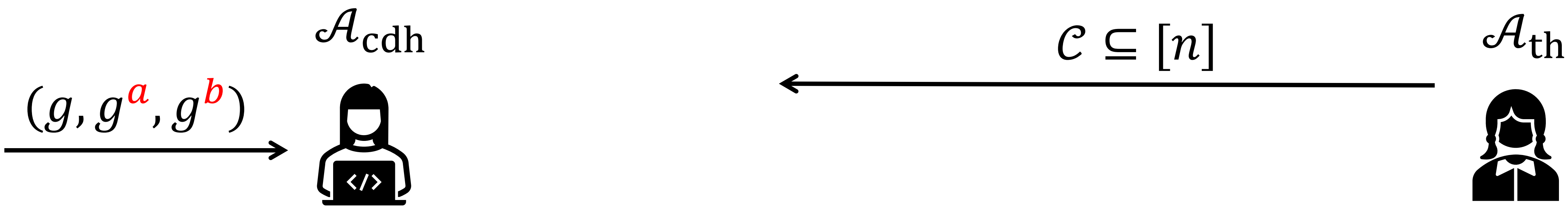
$\mathcal{C} \subseteq [n]$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2,\ldots,t\}$
- Sample $s(1),\ldots,s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$
- Interpolate $\{g^{a}, g^{s(1)}, \cdots, g^{s(t)}\}$ to compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\text{pk} \coloneqq g^{a}, \text{tpk}$

Existing Proof Techniques: Static Security



$\mathcal{A}_{\text{cdh}}$

$(g, g^{a}, g^{b}) \longrightarrow$

$\xleftarrow{\mathcal{C} \subseteq [n]}$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2, \ldots, t\}$
- Sample $s(1), \ldots, s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$
- Interpolate $\{g^{a}, g^{s(1)}, \cdots, g^{s(t)}\}$ to compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\xrightarrow{\text{pk} := g^{a}, \text{tpk}}$

$\xrightarrow{\{\text{sk}_i = s(i)\}_{i \in \mathcal{C}}}$

Existing Proof Techniques: Static Security

$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^a, g^b)}$

$\xleftarrow{\mathcal{C} \subseteq [n]}$

$\mathcal{A}_{\text{th}}$

- Let $\mathcal{C} = \{1,2, \ldots, t\}$
- Sample $s(1), \ldots, s(t) \leftarrow \mathbb{F}$
- Let $s(0) = a$
- Interpolate $\{g^a, g^{s(1)}, \cdots, g^{s(t)}\}$ to compute $\text{tpk} = [g^{s(1)}, \cdots, g^{s(n)}]$

$\xrightarrow{\text{pk} := g^a, \text{tpk}}$

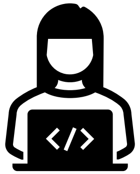
$\xrightarrow{\{\text{sk}_i = s(i)\}_{i \in \mathcal{C}}}$

$\vdots$

Our Proof: Rigged Public Key

Our Proof: Rigged Public Key

$\mathcal{A}_{\mathrm{cdh}}$



$\mathcal{A}_{\mathrm{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$\mathcal{A}_{\text{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$h := g^{a}$

$\mathcal{A}_{\text{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$(g, g^{a}, g^{b}) \longrightarrow$

$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$

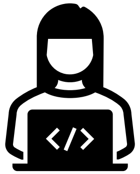
$h \coloneqq g^{a}$

$\mathcal{A}_{\text{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$



$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$
$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

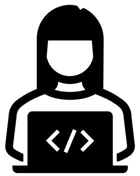
$$\boxed{h \coloneqq g^{a}}$$

$\mathcal{A}_{\text{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$



$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$

$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$\boxed{r \neq 0}$$

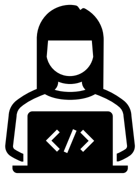
$$\boxed{h \coloneqq g^{a}}$$

$\mathcal{A}_{\text{th}}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$



$\boxed{h := g^{a}}$

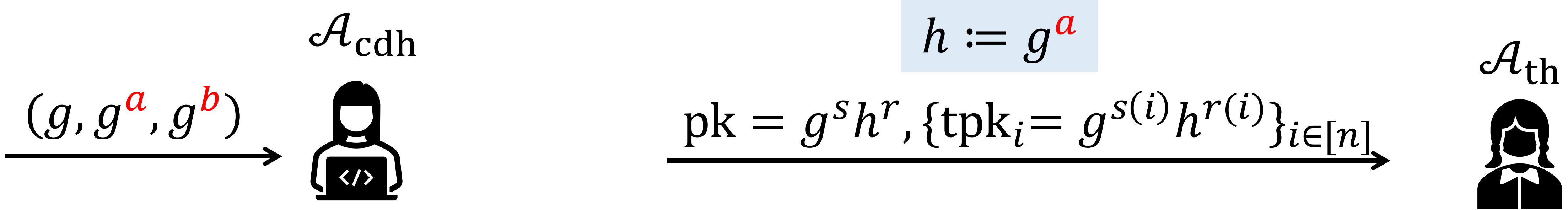
$\mathcal{A}_{\text{th}}$

$$s(x) := s + s_1 x + \cdots + s_t x^t$$
$$r(x) := r + r_1 x + \cdots + r_t x^t$$

$$\boxed{r \neq 0}$$

$$\text{sk} := (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\text{cdh}}$

$$(g, g^{a}, g^{b}) \longrightarrow$$

$$h \coloneqq g^{a}$$

$$\xrightarrow{\text{pk} = g^{s}h^{r}, \{\text{tpk}_i = g^{s(i)}h^{r(i)}\}_{i\in[n]}}$$

$\mathcal{A}_{\text{th}}$

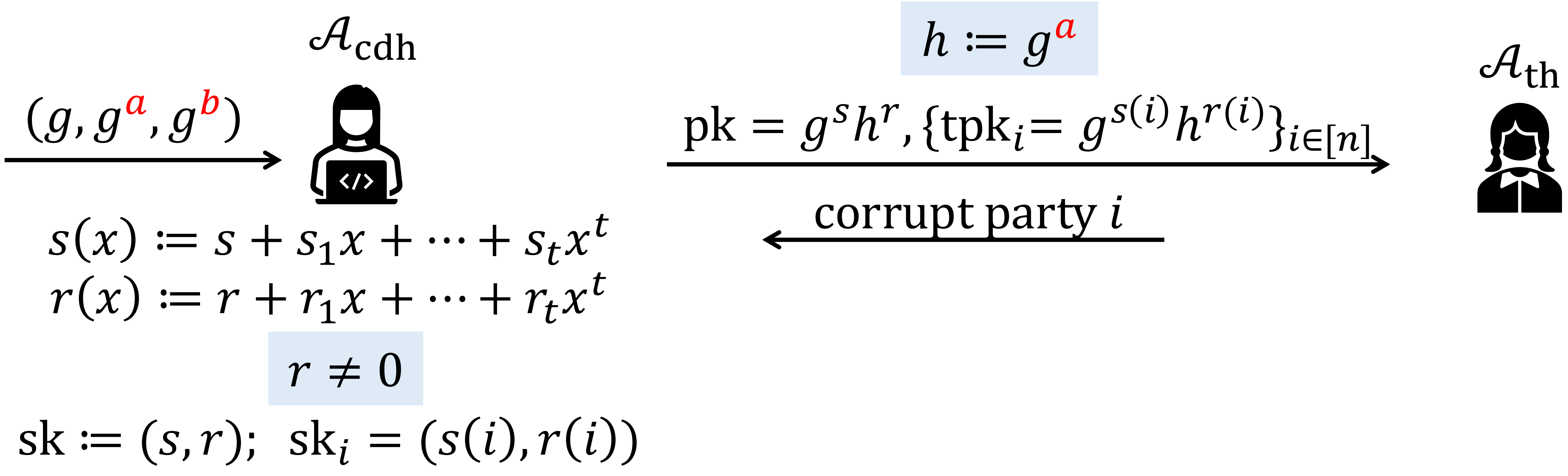
$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$

$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

$h := g^{a}$

$\mathrm{pk} = g^{s} h^{r}, \{\mathrm{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}$

corrupt party i

$\mathcal{A}_{\mathrm{th}}$

$$s(x) := s + s_1 x + \cdots + s_t x^t$$
$$r(x) := r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\mathrm{sk} := (s, r);\;\; \mathrm{sk}_i = (s(i), r(i))$$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$$s(x) := s + s_1 x + \cdots + s_t x^t$$

$$r(x) := r + r_1 x + \cdots + r_t x^t$$

$$\boxed{r \neq 0}$$

$$\text{sk} := (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

$$\boxed{h := g^{a}}$$

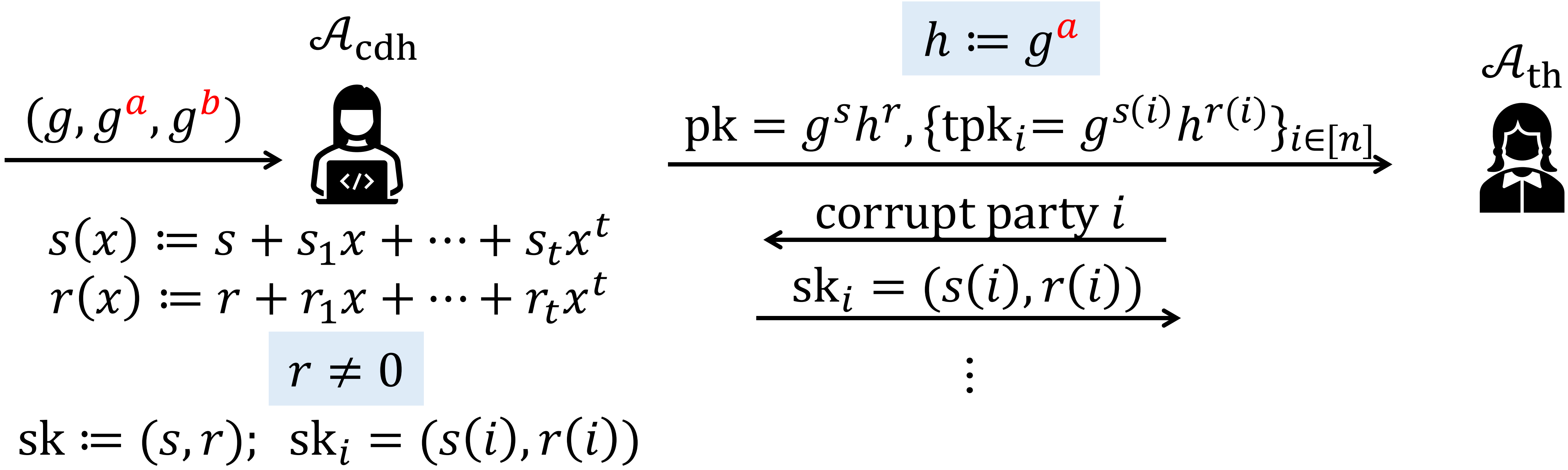
$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$$

$$\xleftarrow{\text{corrupt party } i}$$

$$\xrightarrow{\text{sk}_i = (s(i), r(i))}$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^a, g^b)}$

$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$

$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$

$r \neq 0$

$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$

$h \coloneqq g^a$

$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$

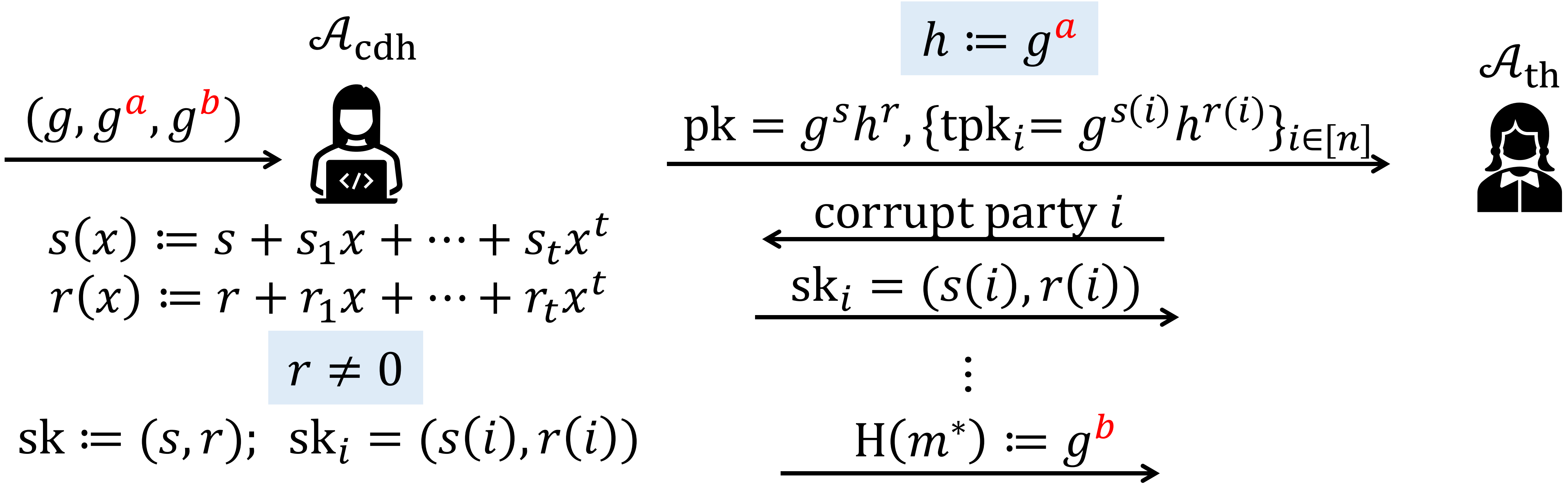
$\mathcal{A}_{\text{th}}$

$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\text{sk}_i = (s(i), r(i))}$

$\vdots$

Our Proof: Rigged Public Key



$\mathcal{A}_{\text{cdh}}$

$(g, g^{a}, g^{b}) \rightarrow$

$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$

$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$\boxed{r \neq 0}$$

$$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

$$\boxed{h \coloneqq g^{a}}$$

$\mathcal{A}_{\text{th}}$

$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$

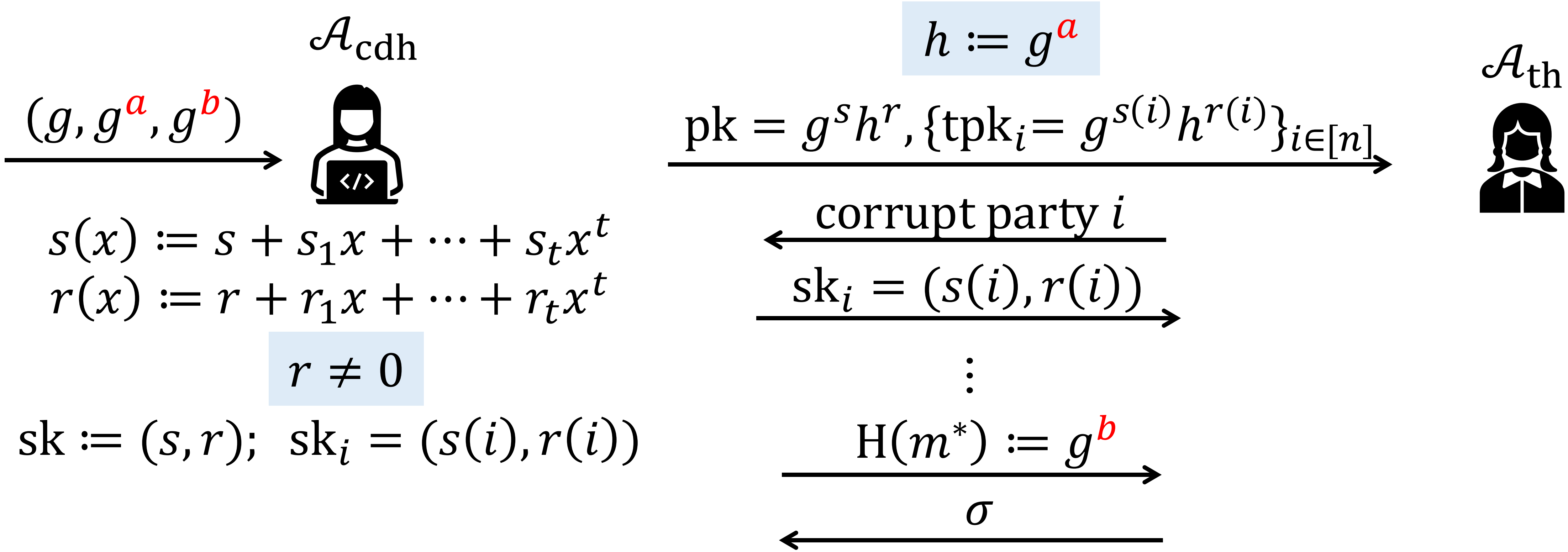
$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\text{sk}_i = (s(i), r(i))}$

$\vdots$

$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$

Our Proof: Rigged Public Key



$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$
$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$\boxed{r \neq 0}$

$$\text{sk} \coloneqq (s, r); \;\; \text{sk}_i = (s(i), r(i))$$

$\boxed{h \coloneqq g^{a}}$

$\mathcal{A}_{\text{th}}$

$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$

$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\text{sk}_i = (s(i), r(i))}$

$\vdots$

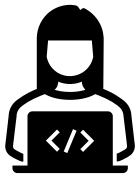
$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$

$\xleftarrow{\sigma}$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$



$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$

$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

$$h \coloneqq g^{a}$$

$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} = g^{s} h^{r}, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$$

$$\xleftarrow{\text{corrupt party } i}$$

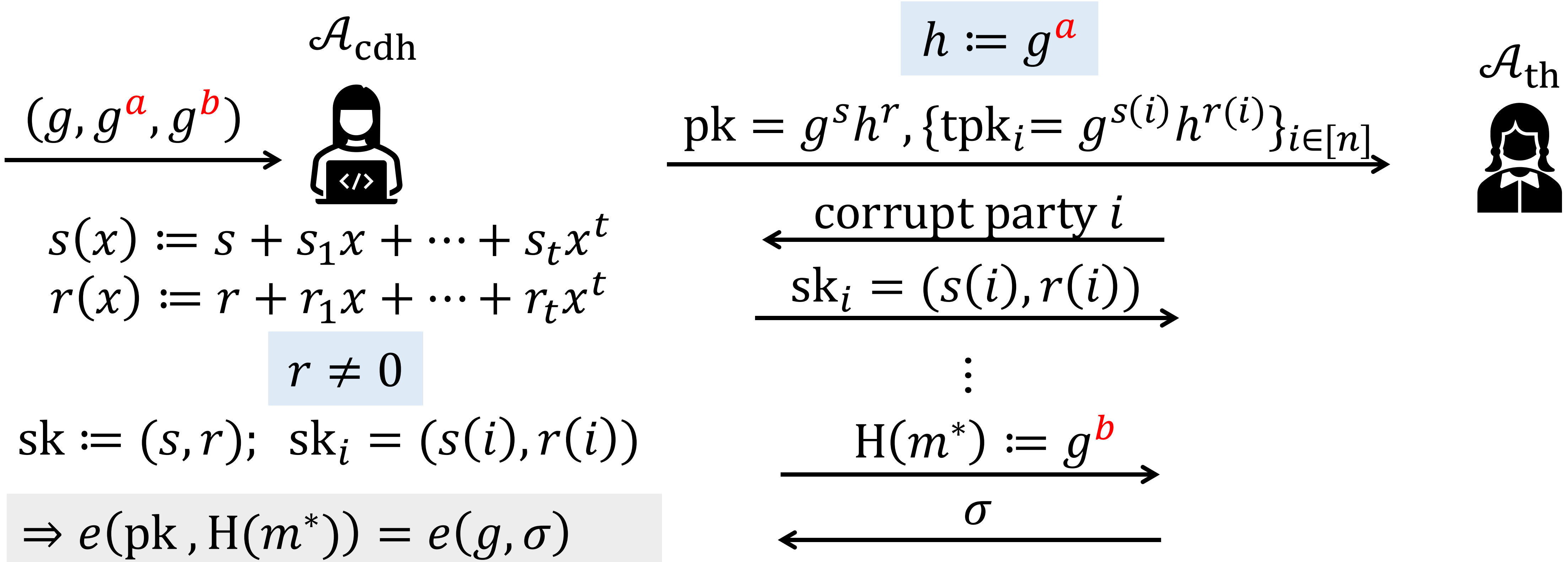
$$\xrightarrow{\text{sk}_i = (s(i), r(i))}$$

$$\vdots$$

$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\mathrm{cdh}}$

$(g, g^{a}, g^{b}) \longrightarrow$

$h \coloneqq g^{a}$

$\mathcal{A}_{\mathrm{th}}$

$\xrightarrow{\mathrm{pk} = g^{s}h^{r}, \{\mathrm{tpk}_i = g^{s(i)}h^{r(i)}\}_{i\in[n]}}$

$\xleftarrow{\text{corrupt party } i}$

$\xrightarrow{\mathrm{sk}_i = (s(i), r(i))}$

$\vdots$

$\xrightarrow{\mathrm{H}(m^*) \coloneqq g^{b}}$

$\xleftarrow{\sigma}$

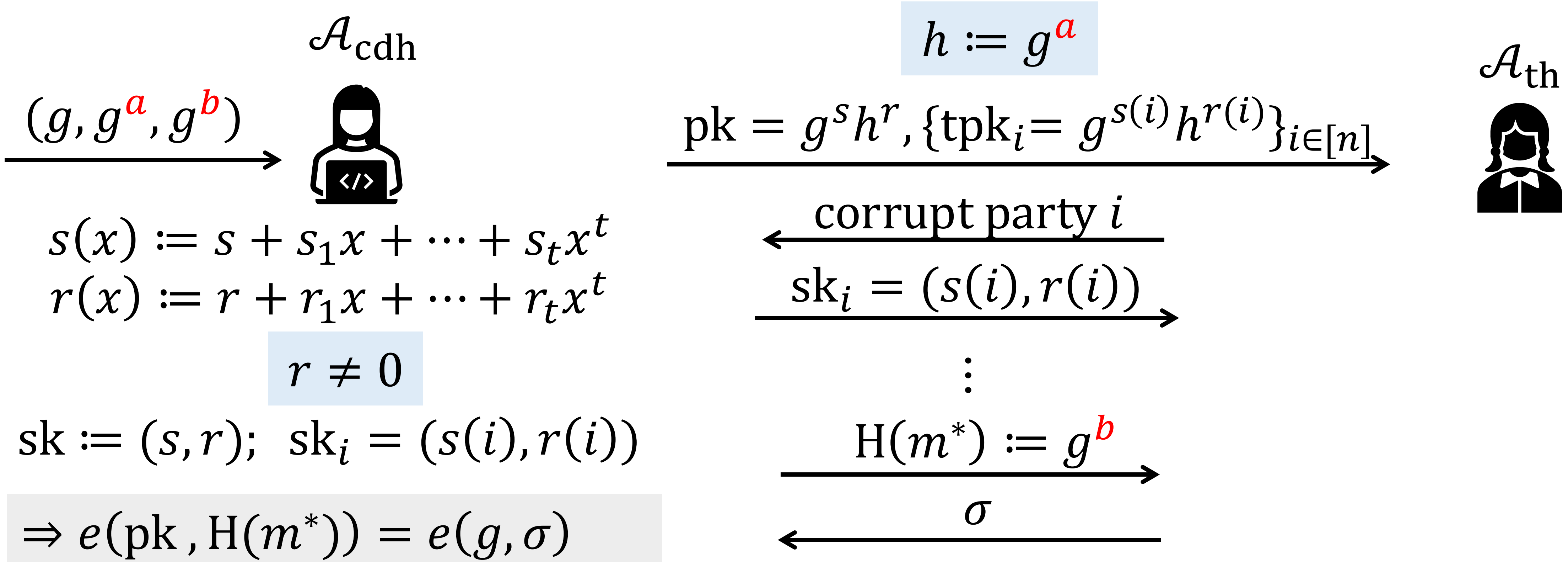
$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$
$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\mathrm{sk} \coloneqq (s, r);\ \ \mathrm{sk}_i = (s(i), r(i))$$

$$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m^*)) = e(g, \sigma)$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\mathrm{cdh}}$

$(g, g^{a}, g^{b}) \rightarrow$

$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$

$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$\boxed{r \neq 0}$$

$$\mathrm{sk} \coloneqq (s, r);\ \ \mathrm{sk}_i = (s(i), r(i))$$

$$\boxed{h \coloneqq g^{a}}$$

$\mathcal{A}_{\mathrm{th}}$

$\mathrm{pk} = g^{s} h^{r}, \{\mathrm{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]} \rightarrow$

$\leftarrow$ corrupt party i

$\mathrm{sk}_i = (s(i), r(i)) \rightarrow$

$\vdots$

$\mathrm{H}(m^*) \coloneqq g^{b} \rightarrow$

$\leftarrow \sigma$

$$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m^*)) = e(g, \sigma)$$

$$\Rightarrow e\left(g^{s} h^{r}, g^{b}\right) = e(g, \sigma)$$

Our Proof: Rigged Public Key

$\mathcal{A}_{\text{cdh}}$

$$\xrightarrow{(g, g^{a}, g^{b})}$$

$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$
$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

$$\Rightarrow e(\text{pk}, \text{H}(m^*)) = e(g, \sigma)$$
$$\Rightarrow e\left(g^s h^r, g^{b}\right) = e(g, \sigma)$$
$$\Rightarrow e\left(g^s g^{ar}, g^{b}\right) = e(g, \sigma)$$

$$h \coloneqq g^{a}$$

$\mathcal{A}_{\text{th}}$

$$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$$

$$\xleftarrow{\text{corrupt party } i}$$

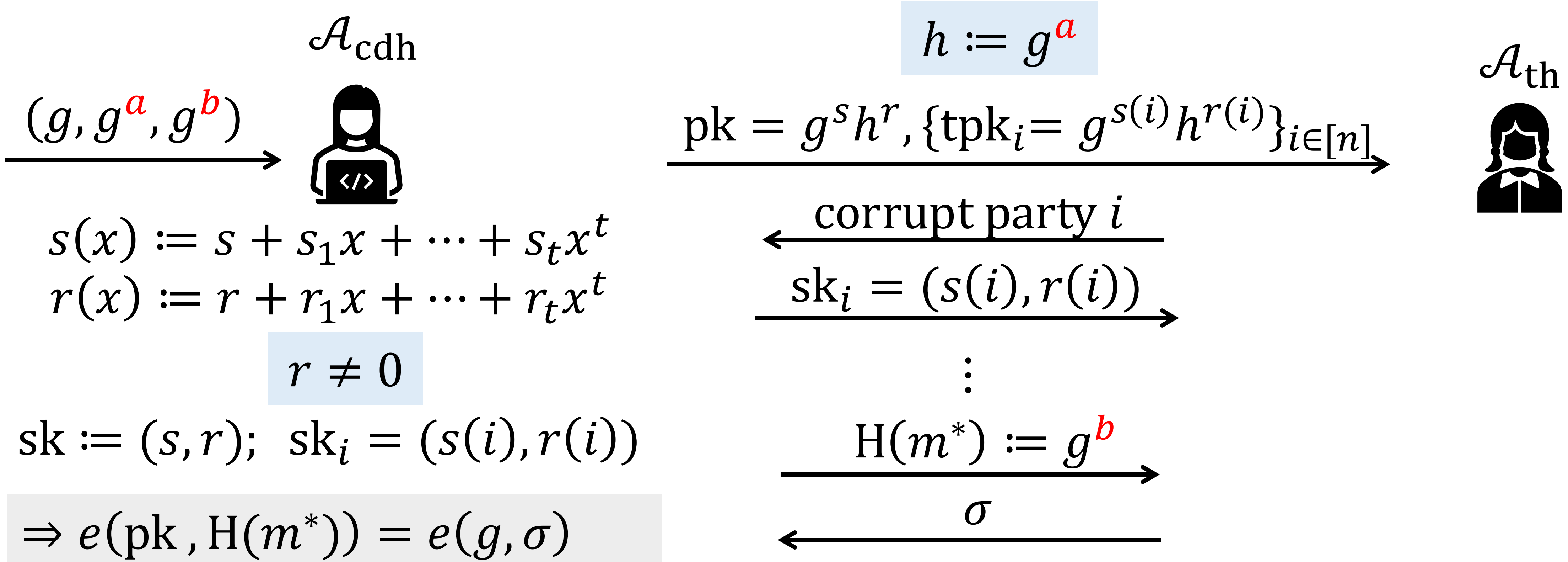
$$\xrightarrow{\text{sk}_i = (s(i), r(i))}$$

$$\vdots$$

$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\text{cdh}}$

$$(g, g^{a}, g^{b}) \longrightarrow$$

$$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$$
$$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$$

$$r \neq 0$$

$$\text{sk} \coloneqq (s, r);\ \ \text{sk}_i = (s(i), r(i))$$

$$h \coloneqq g^{a}$$

$$\xrightarrow{\text{pk} = g^s h^r, \{\text{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}}$$

$$\xleftarrow{\text{corrupt party } i}$$

$$\xrightarrow{\text{sk}_i = (s(i), r(i))}$$

$$\vdots$$

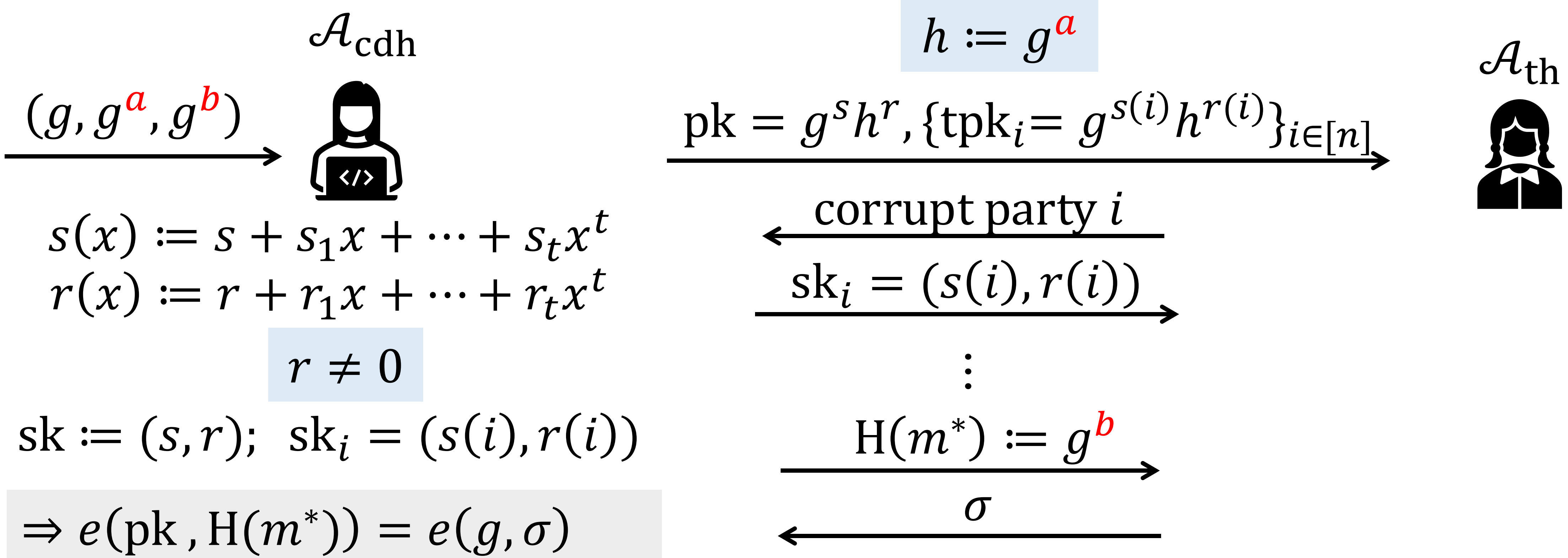
$$\xrightarrow{\text{H}(m^*) \coloneqq g^{b}}$$

$$\xleftarrow{\sigma}$$

$\mathcal{A}_{\text{th}}$

$$\Rightarrow e(\text{pk}, \text{H}(m^*)) = e(g, \sigma)$$
$$\Rightarrow e\left(g^s h^r, g^{b}\right) = e(g, \sigma)$$
$$\Rightarrow e\left(g^s g^{ar}, g^{b}\right) = e(g, \sigma)$$
$$\Rightarrow g^{ab} = \sigma^{r^{-1}} \cdot g^{-sbr^{-1}}$$

Our Proof: Rigged Public Key



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

$s(x) \coloneqq s + s_1 x + \cdots + s_t x^t$

$r(x) \coloneqq r + r_1 x + \cdots + r_t x^t$

$r \neq 0$

$\mathrm{sk} \coloneqq (s, r);\ \ \mathrm{sk}_i = (s(i), r(i))$

$h \coloneqq g^{a}$

$\mathcal{A}_{\mathrm{th}}$

$\mathrm{pk} = g^s h^r, \{\mathrm{tpk}_i = g^{s(i)} h^{r(i)}\}_{i \in [n]}$

corrupt party i

$\mathrm{sk}_i = (s(i), r(i))$

$\vdots$

$\mathrm{H}(m^*) \coloneqq g^{b}$

σ

$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m^*)) = e(g, \sigma)$

$\Rightarrow e\left(g^s h^r, g^{b}\right) = e(g, \sigma)$

$\Rightarrow e\left(g^s g^{ar}, g^{b}\right) = e(g, \sigma)$

$\Rightarrow g^{ab} = \sigma^{r^{-1}} \cdot g^{-sbr^{-1}}$

$\mathcal{A}_{\mathrm{cdh}}$ knows both $s(x)$ and $r(x)$, hence s and r

Simulating Signing Queries

Simulating Signing Queries

$\mathcal{A}_{\text{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

$h \coloneqq g^{a}$

$\mathcal{A}_{\text{th}}$

Simulating Signing Queries

$\mathcal{A}_{\mathrm{cdh}}$

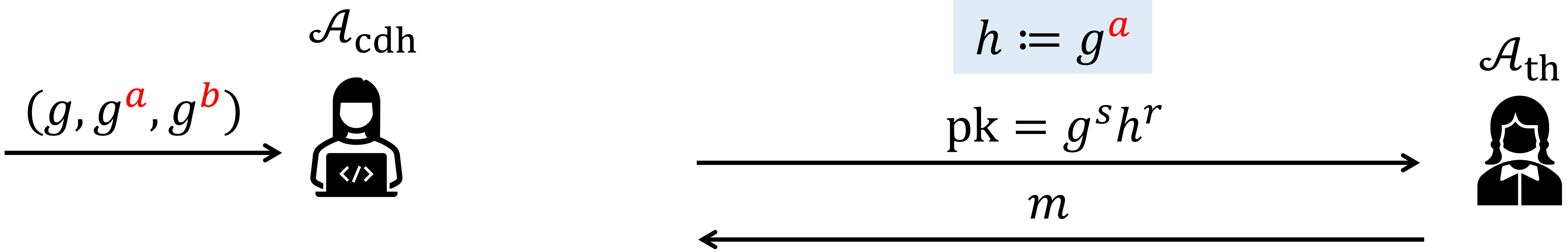
(g, g^{a}, g^{b})

$h := g^{a}$

$\mathrm{pk} = g^{s} h^{r}$

$\mathcal{A}_{\mathrm{th}}$

Simulating Signing Queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$h \coloneqq g^{a}$

$\mathrm{pk} = g^{s}h^{r}$

m

$\mathcal{A}_{\text{th}}$

Simulating Signing Queries

$\xrightarrow{(g, g^{a}, g^{b})}$ $\mathcal{A}_{\mathrm{cdh}}$

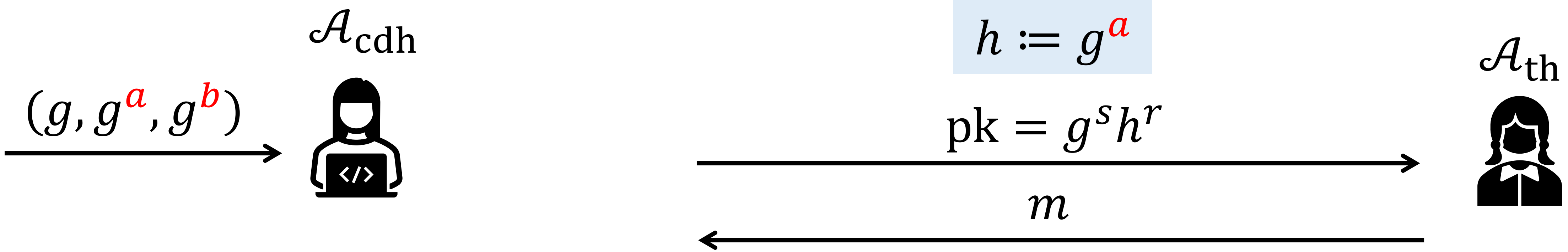
$h := g^{a}$

$\xrightarrow{\mathrm{pk} = g^{s} h^{r}}$ $\mathcal{A}_{\mathrm{th}}$

$\xleftarrow{m}$

1. Program $\mathrm{H}(m) = g^{x}$

Simulating Signing Queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$h \coloneqq g^{a}$

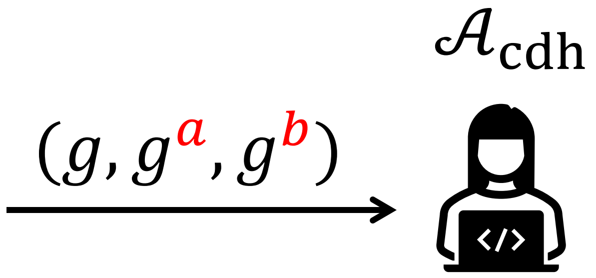
$\mathcal{A}_{\text{th}}$

$\text{pk} = g^{s}h^{r}$

m

1. Program $\text{H}(m) = g^{x}$
2. Let $\sigma = \text{H}(m)^{s}$

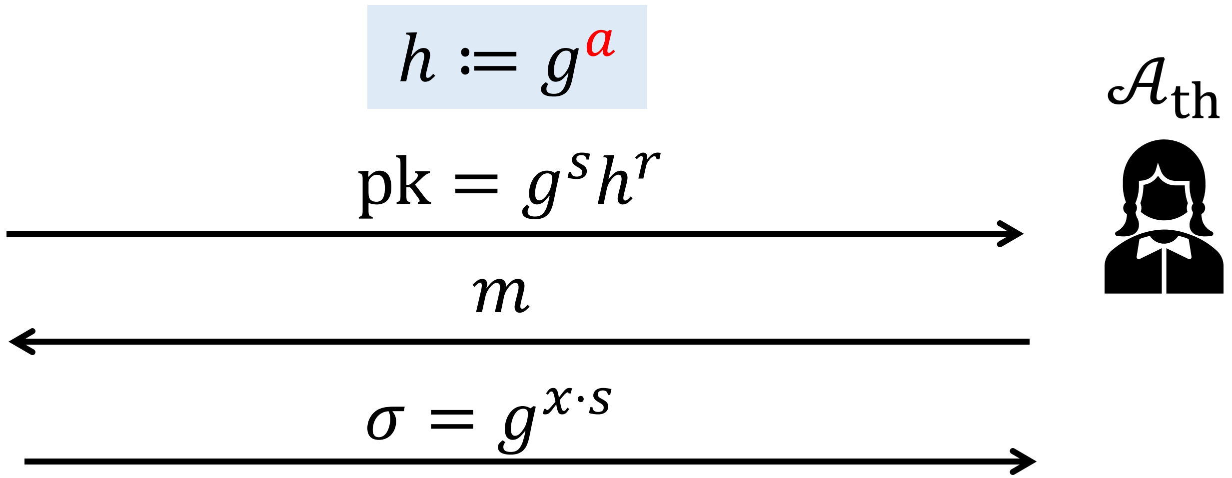
Simulating Signing Queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

1. Program $\text{H}(m) = g^{x}$
2. Let $\sigma = \text{H}(m)^{s}$



$h \coloneqq g^{a}$

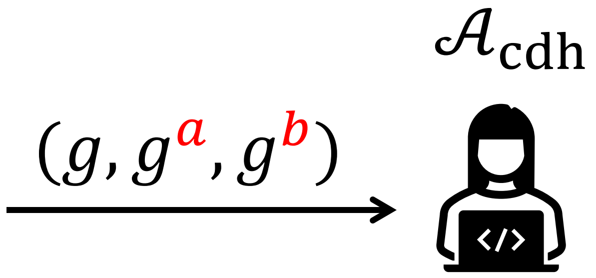
$\mathcal{A}_{\text{th}}$

$\text{pk} = g^{s}h^{r}$

m

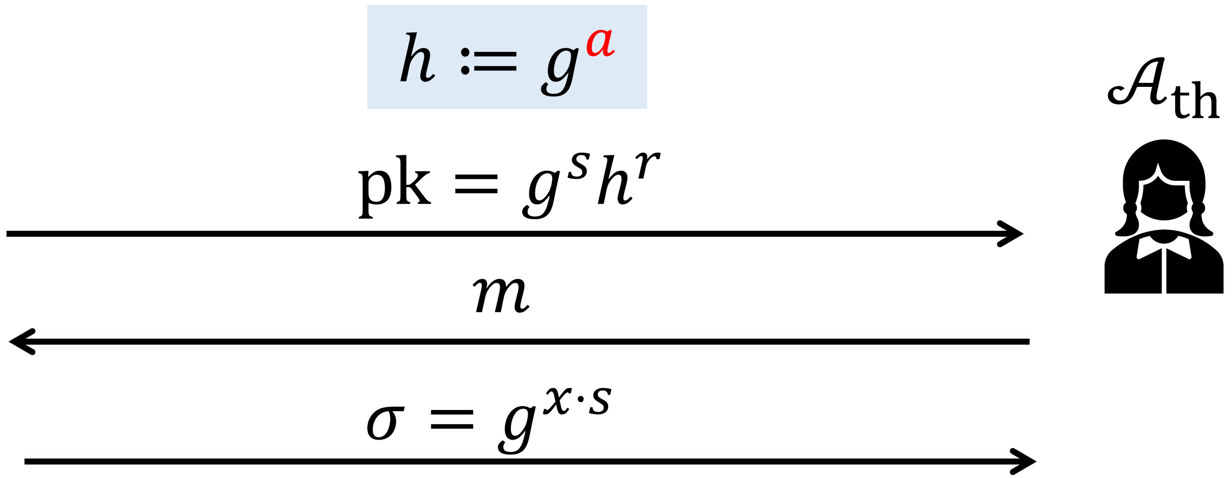
$\sigma = g^{x \cdot s}$

Simulating Signing Queries



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})



$h := g^{a}$

$\mathcal{A}_{\mathrm{th}}$

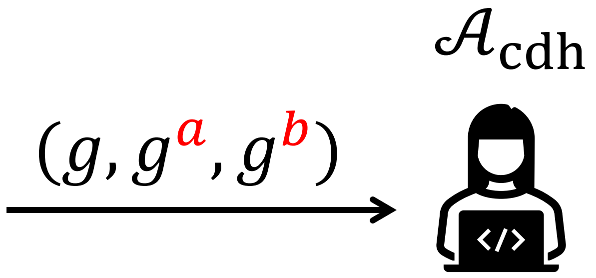
$\mathrm{pk} = g^{s}h^{r}$

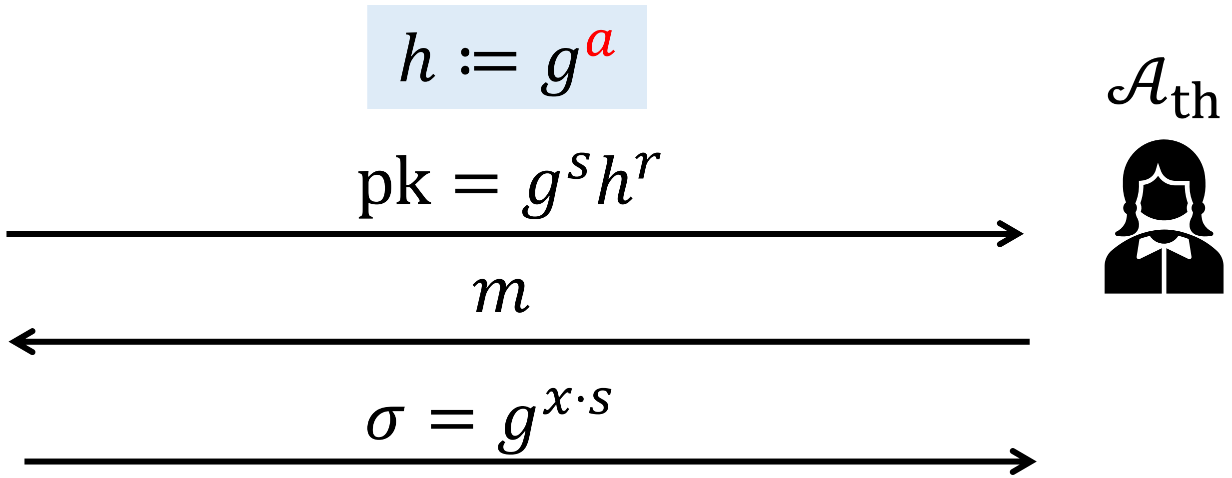
m

$\sigma = g^{x \cdot s}$

1. Program $\mathrm{H}(m) = g^{x}$
2. Let $\sigma = \mathrm{H}(m)^{s}$

Simulating Signing Queries





1. Program $\mathrm{H}(m) = g^{x}$
2. Let $\sigma = \mathrm{H}(m)^{s}$

$\Rightarrow e(\text{pk}, \mathrm{H}(m)) = e(g, \sigma)$

Simulating Signing Queries

$\mathcal{A}_{\mathrm{cdh}}$

$\xrightarrow{(g, g^{a}, g^{b})}$

1. Program $\mathrm{H}(m) = g^{x}$
2. Let $\sigma = \mathrm{H}(m)^{s}$

$h := g^{a}$

$\mathcal{A}_{\mathrm{th}}$

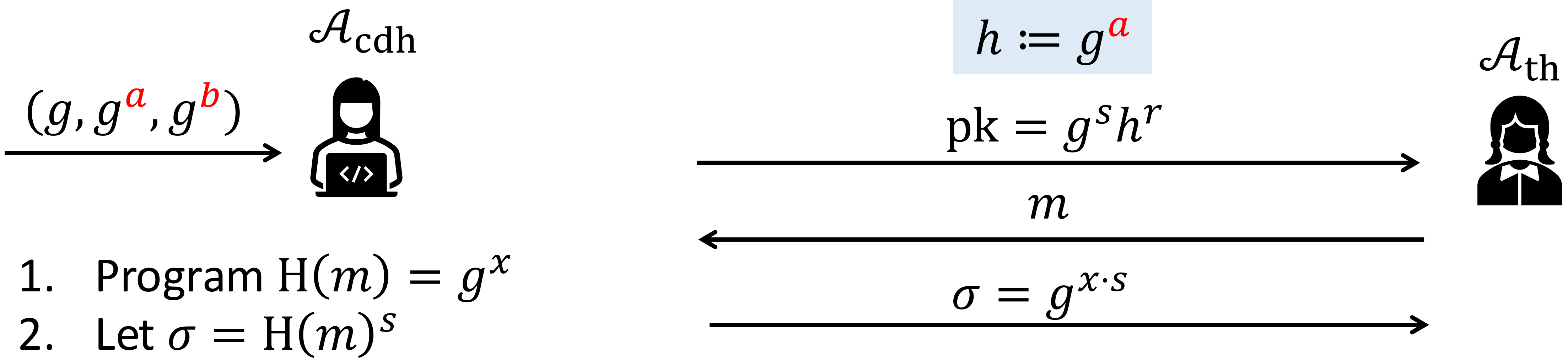
$\xrightarrow{\mathrm{pk} = g^{s} h^{r}}$

$\xleftarrow{m}$

$\xrightarrow{\sigma = g^{x \cdot s}}$

$$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$
$$\Rightarrow e(g^{s+ar}, g^{x}) = e(g, g^{x \cdot s})$$

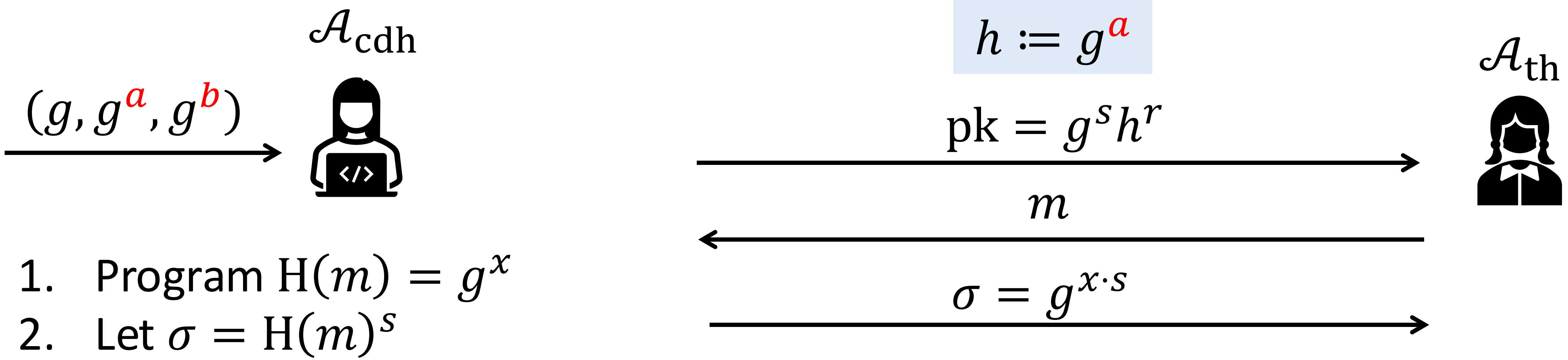
Simulating Signing Queries



1. Program $\text{H}(m) = g^{x}$
2. Let $\sigma = \text{H}(m)^{s}$

$$\Rightarrow e(\text{pk}, \text{H}(m)) = e(g, \sigma)$$
$$\Rightarrow e(g^{s+ar}, g^{x}) = e(g, g^{x \cdot s})$$
$$\Rightarrow r = 0$$

Simulating Signing Queries



$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$h \coloneqq g^{a}$

$\mathcal{A}_{\text{th}}$

$\text{pk} = g^{s}h^{r}$

m

$\sigma = g^{x \cdot s}$

1. Program $\text{H}(m) = g^{x}$
2. Let $\sigma = \text{H}(m)^{s}$

$\Rightarrow e(\text{pk}, \text{H}(m)) = e(g, \sigma)$

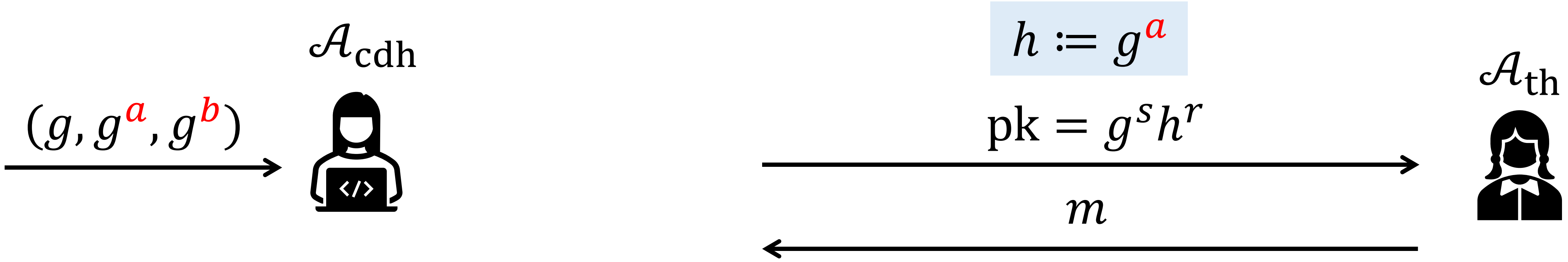
$\Rightarrow e(g^{s+ar}, g^{x}) = e(g, g^{x \cdot s})$

$\Rightarrow r = 0$

We will need new ideas

Simulating Signing Queries

Simulating Signing Queries



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

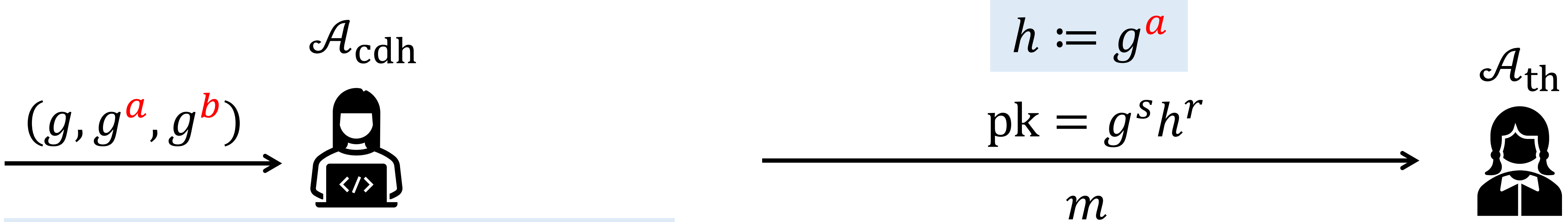
$h := g^{a}$

$\mathrm{pk} = g^{s} h^{r}$

m

$\mathcal{A}_{\mathrm{th}}$

Simulating Signing Queries


$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

$h := g^{a}$

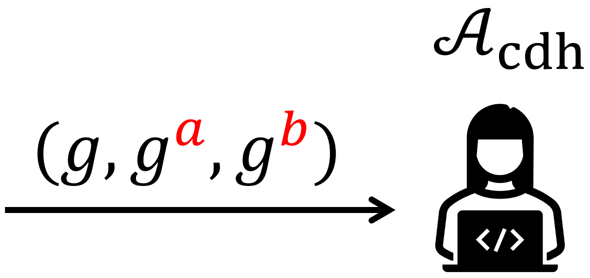
$\mathcal{A}_{\text{th}}$

$\text{pk} = g^{s}h^{r}$

m

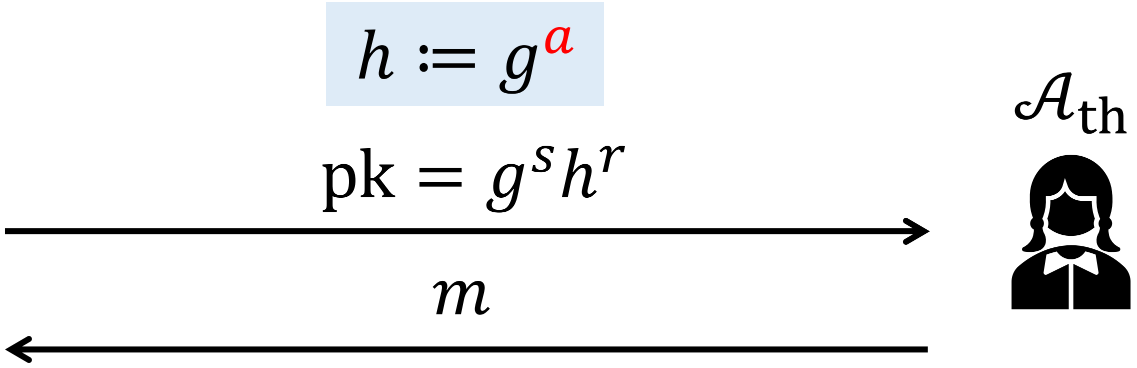
Idea: Use **two** random oracles!

Simulating Signing Queries



$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})



$h := g^{a}$

$\mathcal{A}_{\mathrm{th}}$

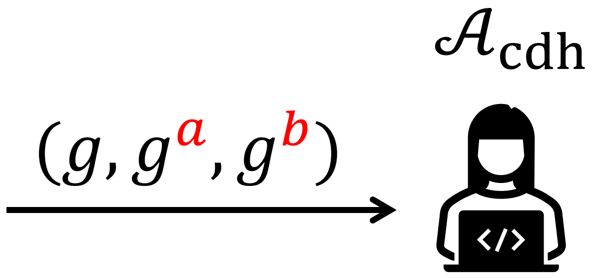
$\mathrm{pk} = g^{s}h^{r}$

m

Idea: Use **two** random oracles!

$\mathrm{H}(m) = g^{x}$ $\qquad \widehat{\mathrm{H}}(m) = g^{y}$

Simulating Signing Queries

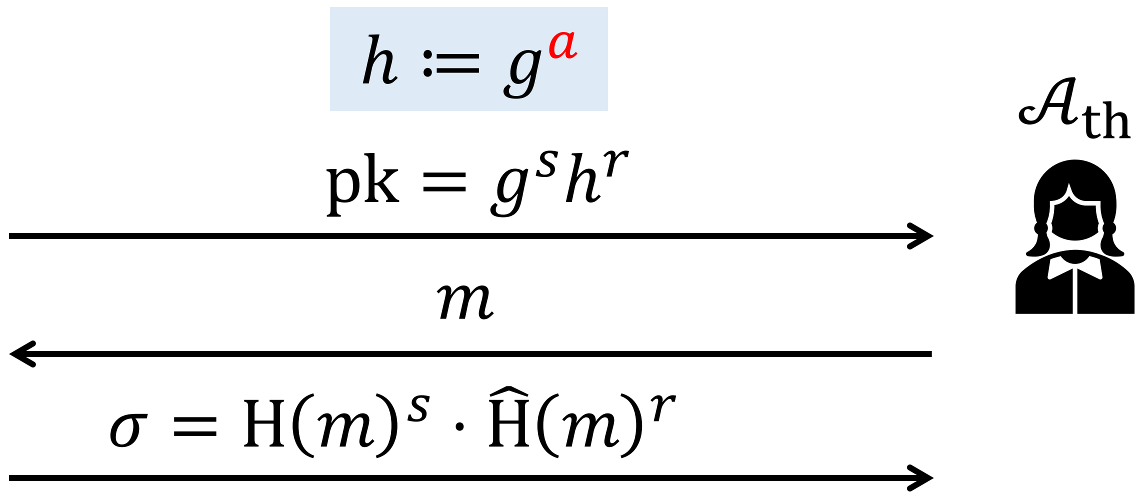


$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

Idea: Use **two** random oracles!

$\mathrm{H}(m) = g^{x}$ $\qquad$ $\widehat{\mathrm{H}}(m) = g^{y}$



$h \coloneqq g^{a}$

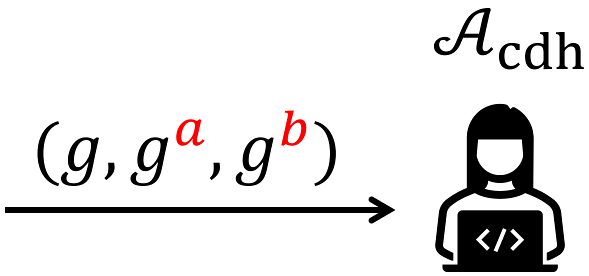
$\mathcal{A}_{\text{th}}$

$\mathrm{pk} = g^{s} h^{r}$

m

$\sigma = \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}$

Simulating Signing Queries

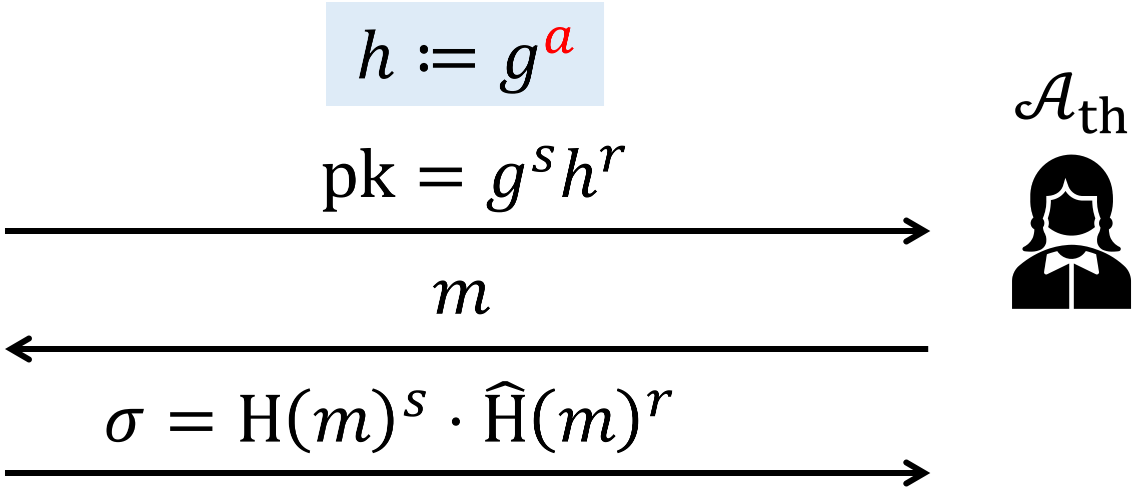


$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

Idea: Use **two** random oracles!

$\mathrm{H}(m) = g^{x}$ $\widehat{\mathrm{H}}(m) = g^{y}$



$h := g^{a}$

$\mathcal{A}_{\text{th}}$

$\text{pk} = g^{s}h^{r}$

m

$\sigma = \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}$

Simulating Signing Queries

$\mathcal{A}_{\text{cdh}}$

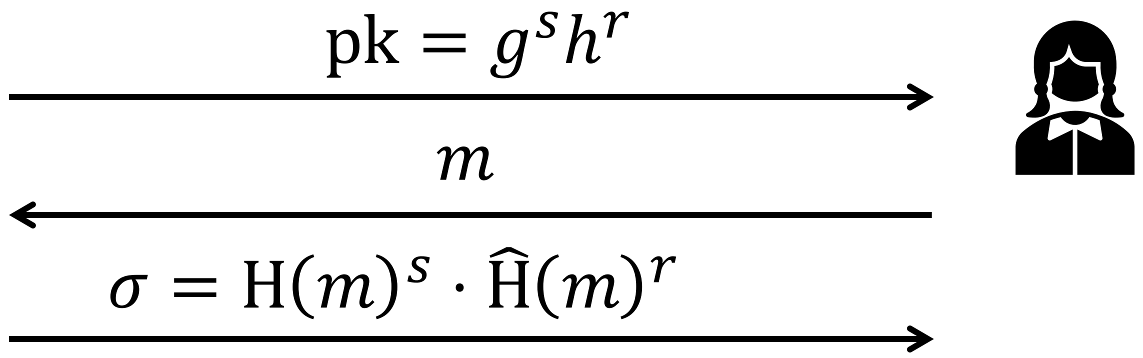
$\xrightarrow{(g, g^{a}, g^{b})}$

Idea: Use **two** random oracles!

$\mathrm{H}(m) = g^{x} \qquad \widehat{\mathrm{H}}(m) = g^{y}$

$h \coloneqq g^{a}$

$\mathcal{A}_{\text{th}}$



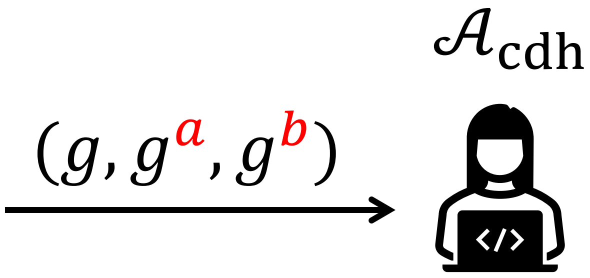
$\xrightarrow{\mathrm{pk} = g^{s} h^{r}}$

$\xleftarrow{m}$

$\xrightarrow{\sigma = \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}}$

$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$

Simulating Signing Queries

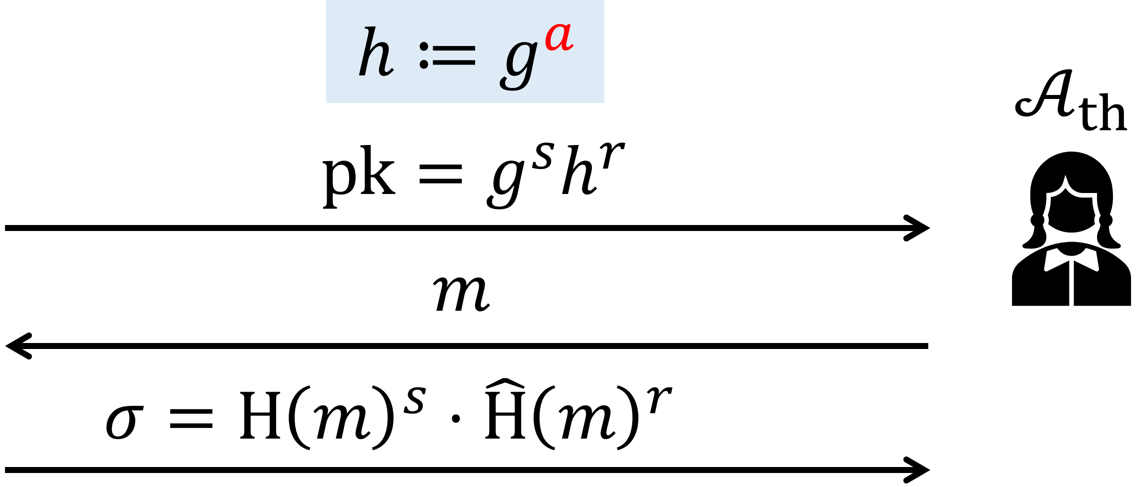


$\mathcal{A}_{\text{cdh}}$

(g, g^{a}, g^{b})

Idea: Use **two** random oracles!

$\text{H}(m) = g^{x} \qquad \widehat{\text{H}}(m) = g^{y}$



$h := g^{a}$

$\mathcal{A}_{\text{th}}$

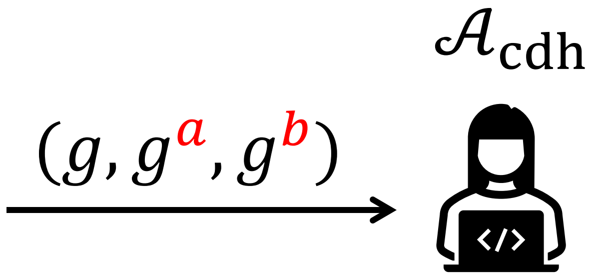
$\xrightarrow{\text{pk} = g^{s}h^{r}}$

$\xleftarrow{m}$

$\xrightarrow{\sigma = \text{H}(m)^{s} \cdot \widehat{\text{H}}(m)^{r}}$

$$\Rightarrow e(\text{pk}, \text{H}(m)) = e(g, \sigma)$$
$$\Rightarrow e(g^{s+ar}, g^{x}) = e\left(g, \text{H}(m)^{s} \cdot \widehat{\text{H}}(m)^{r}\right)$$

Simulating Signing Queries

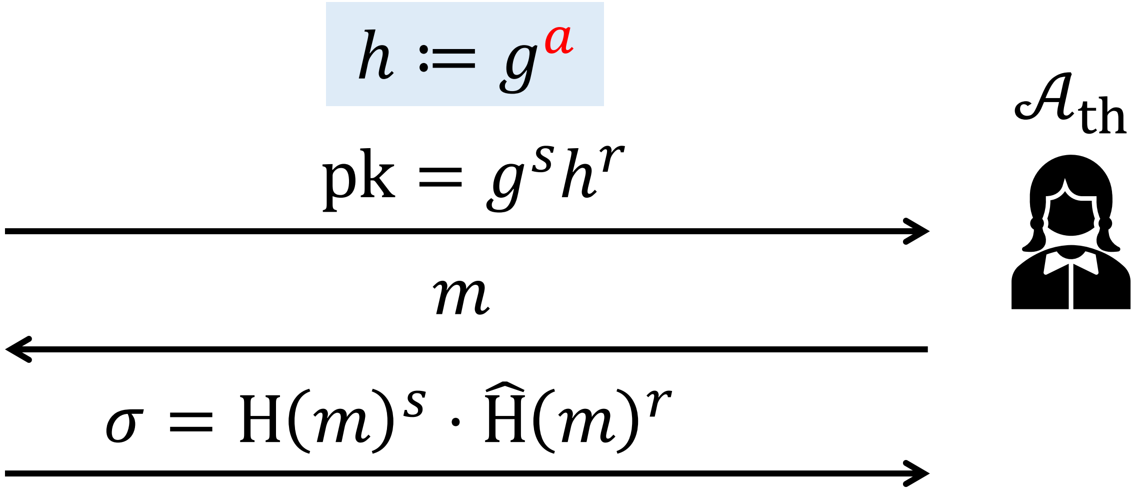


$\mathcal{A}_{\mathrm{cdh}}$

(g, g^{a}, g^{b})

Idea: Use **two** random oracles!

$$\mathrm{H}(m) = g^{x} \qquad \widehat{\mathrm{H}}(m) = g^{y}$$



$h := g^{a}$

$\mathcal{A}_{\mathrm{th}}$

$\mathrm{pk} = g^{s} h^{r}$

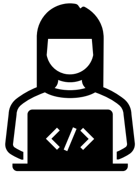
m

$\sigma = \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}$

$$\Rightarrow e(\mathrm{pk}, \mathrm{H}(m)) = e(g, \sigma)$$
$$\Rightarrow e(g^{s+ar}, g^{x}) = e\left(g, \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}\right)$$
$$\Rightarrow ax = y$$

Simulating Signing Queries

$\mathcal{A}_{\text{cdh}}$



$(g, g^{a}, g^{b}) \longrightarrow$

Idea: Use **two** random oracles!

$\mathrm{H}(m) = g^{x} \qquad \widehat{\mathrm{H}}(m) = g^{y}$

$h \coloneqq g^{a}$

$\mathcal{A}_{\text{th}}$

$\xrightarrow{\mathrm{pk} = g^{s} h^{r}}$

$\xleftarrow{m}$

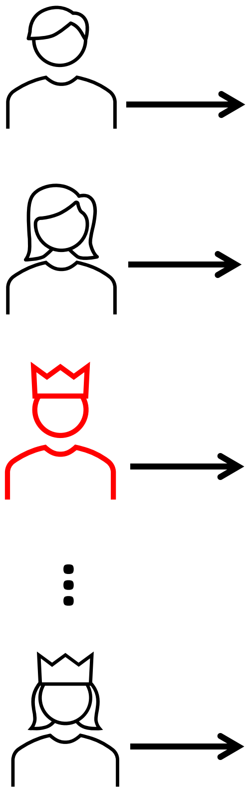
$\xrightarrow{\sigma = \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}}$

$$\Rightarrow e(\mathrm{pk}\,, \mathrm{H}(m)) = e(g, \sigma)$$
$$\Rightarrow e(g^{s+ar}, g^{x}) = e\left(g, \mathrm{H}(m)^{s} \cdot \widehat{\mathrm{H}}(m)^{r}\right)$$
$$\Rightarrow ax = y$$

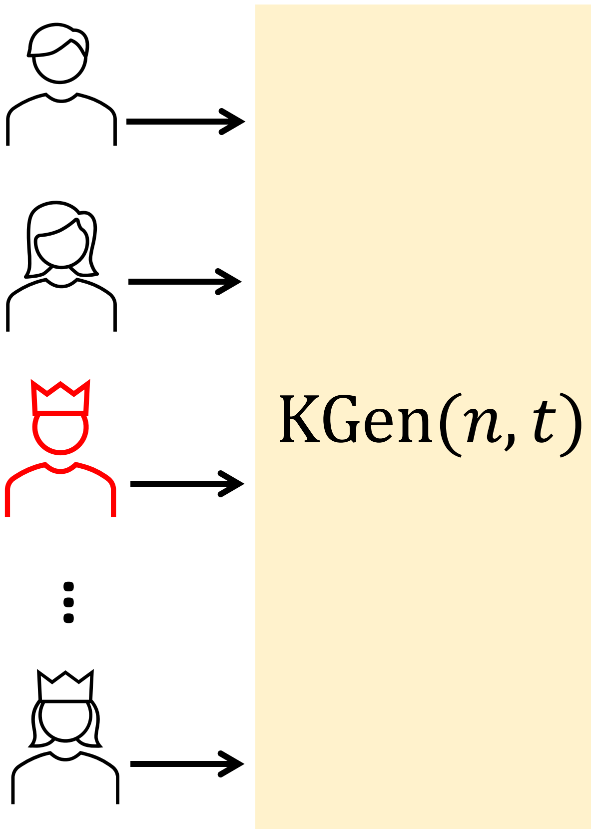
No security proof for this construction :(

Our Final Protocol: Key Generation

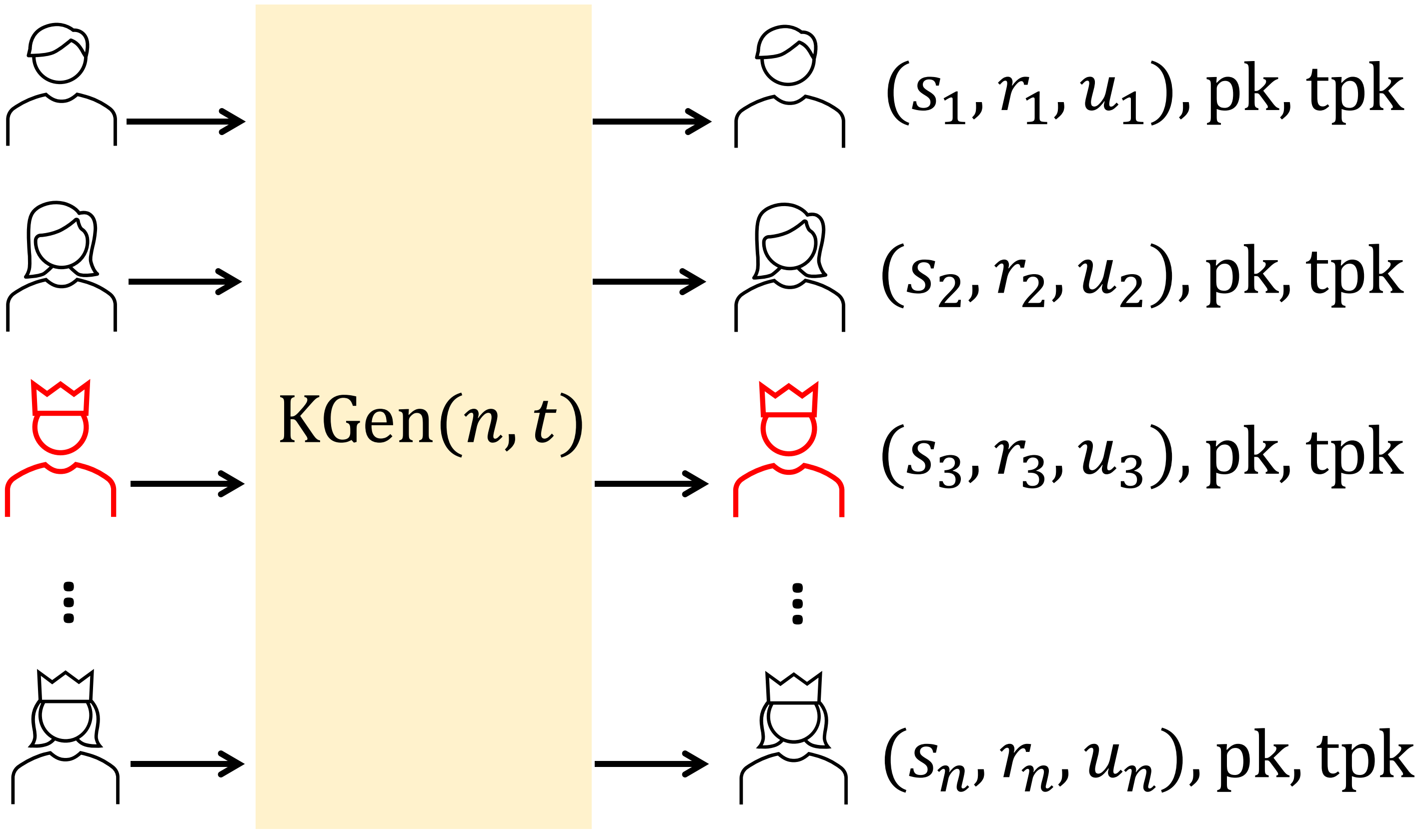
Our Final Protocol: Key Generation



Our Final Protocol: Key Generation



Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

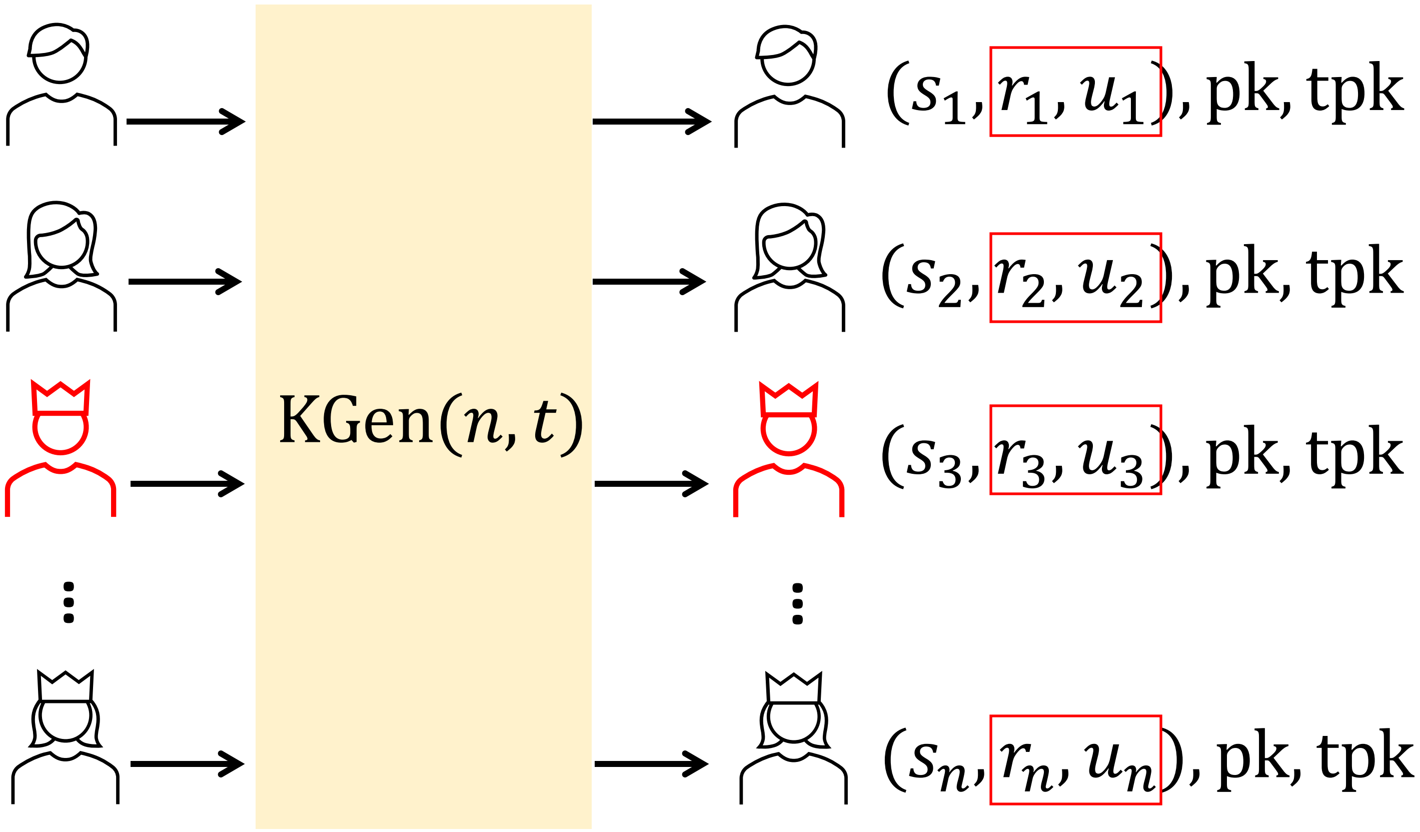
$(s_1, r_1, u_1), \mathrm{pk}, \mathrm{tpk}$

$(s_2, r_2, u_2), \mathrm{pk}, \mathrm{tpk}$

$(s_3, r_3, u_3), \mathrm{pk}, \mathrm{tpk}$

$(s_n, r_n, u_n), \mathrm{pk}, \mathrm{tpk}$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

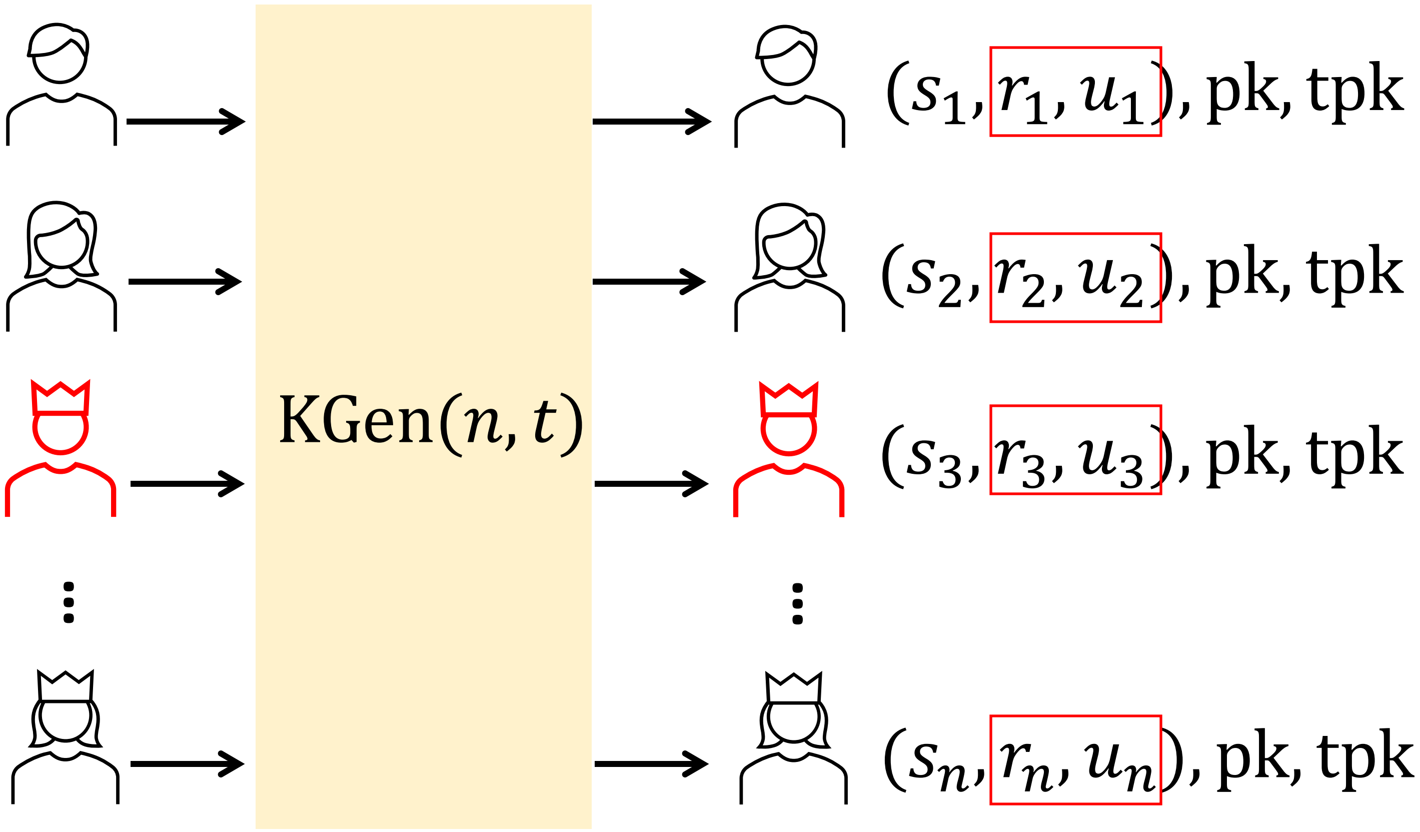
$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

$\vdots$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

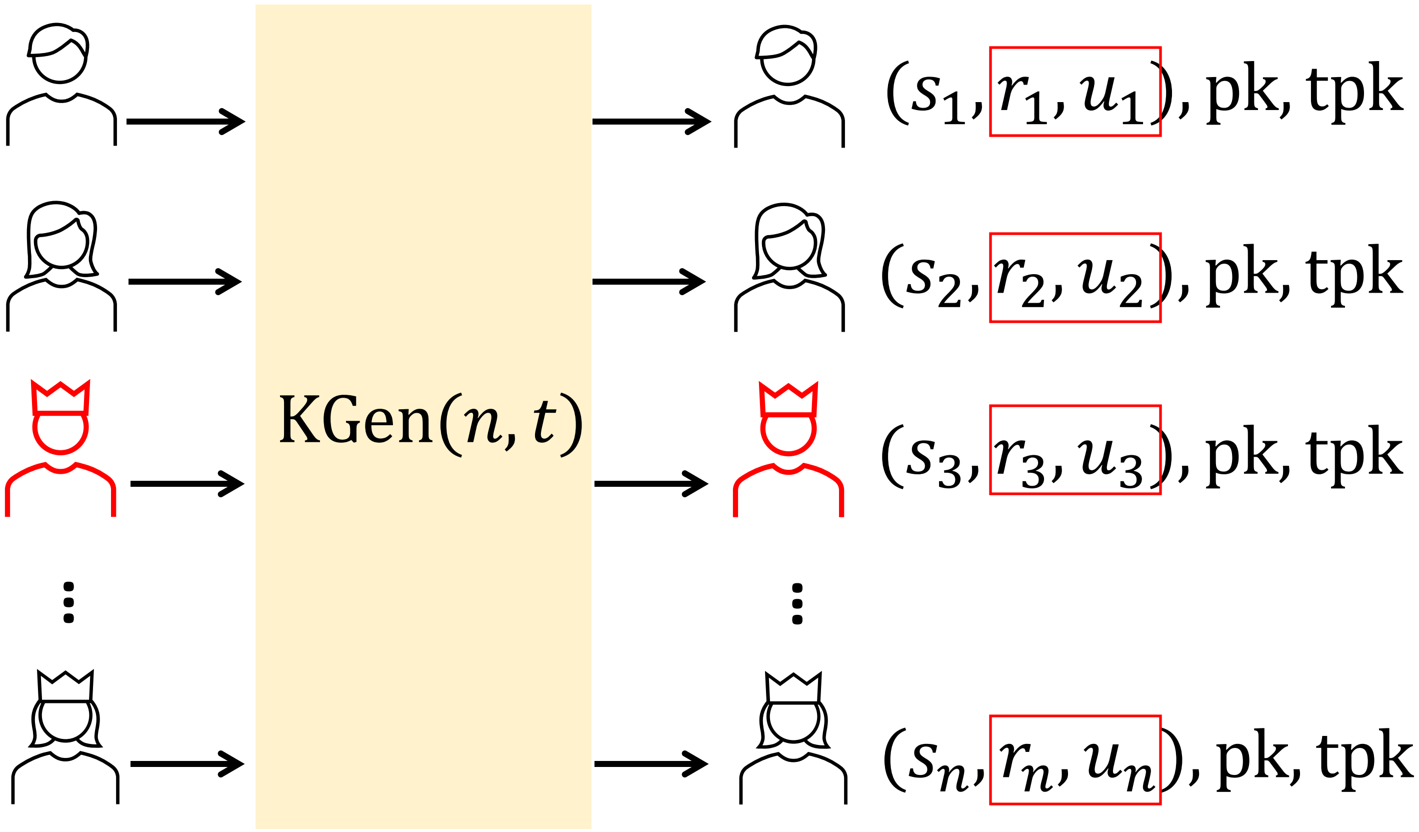
$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

$\vdots$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

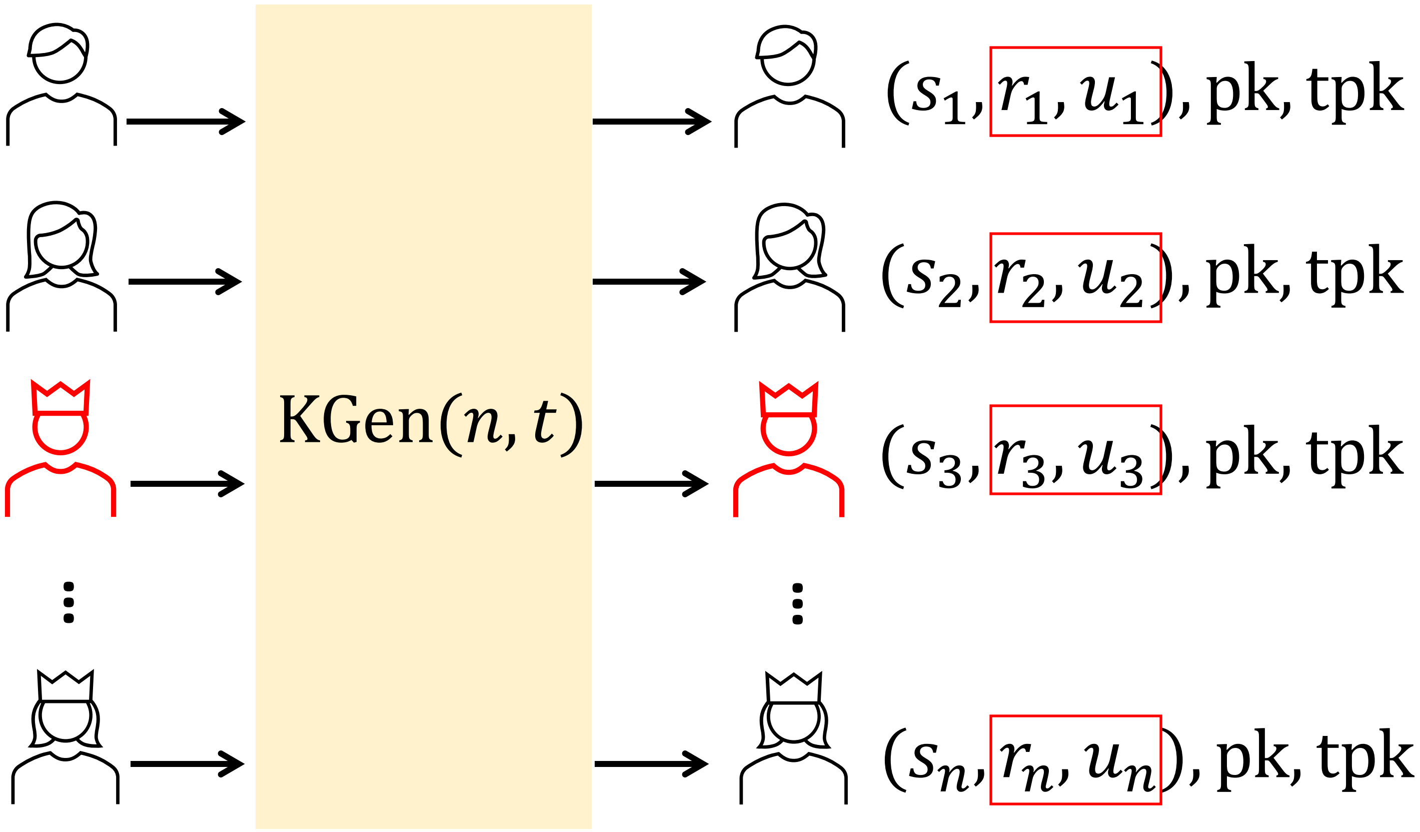
$\vdots$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

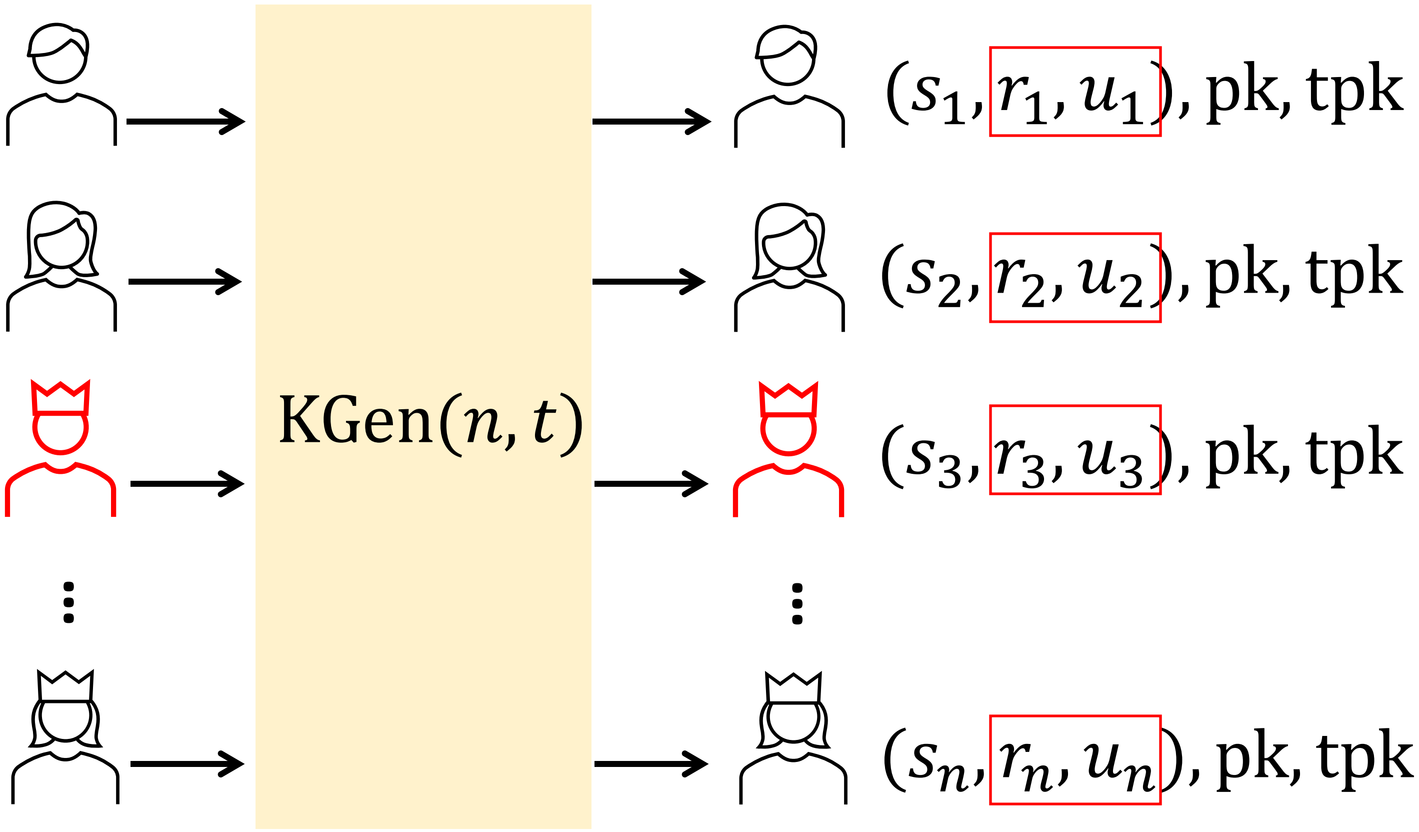
$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$

$\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)$

Our Final Protocol: Key Generation



$\mathsf{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

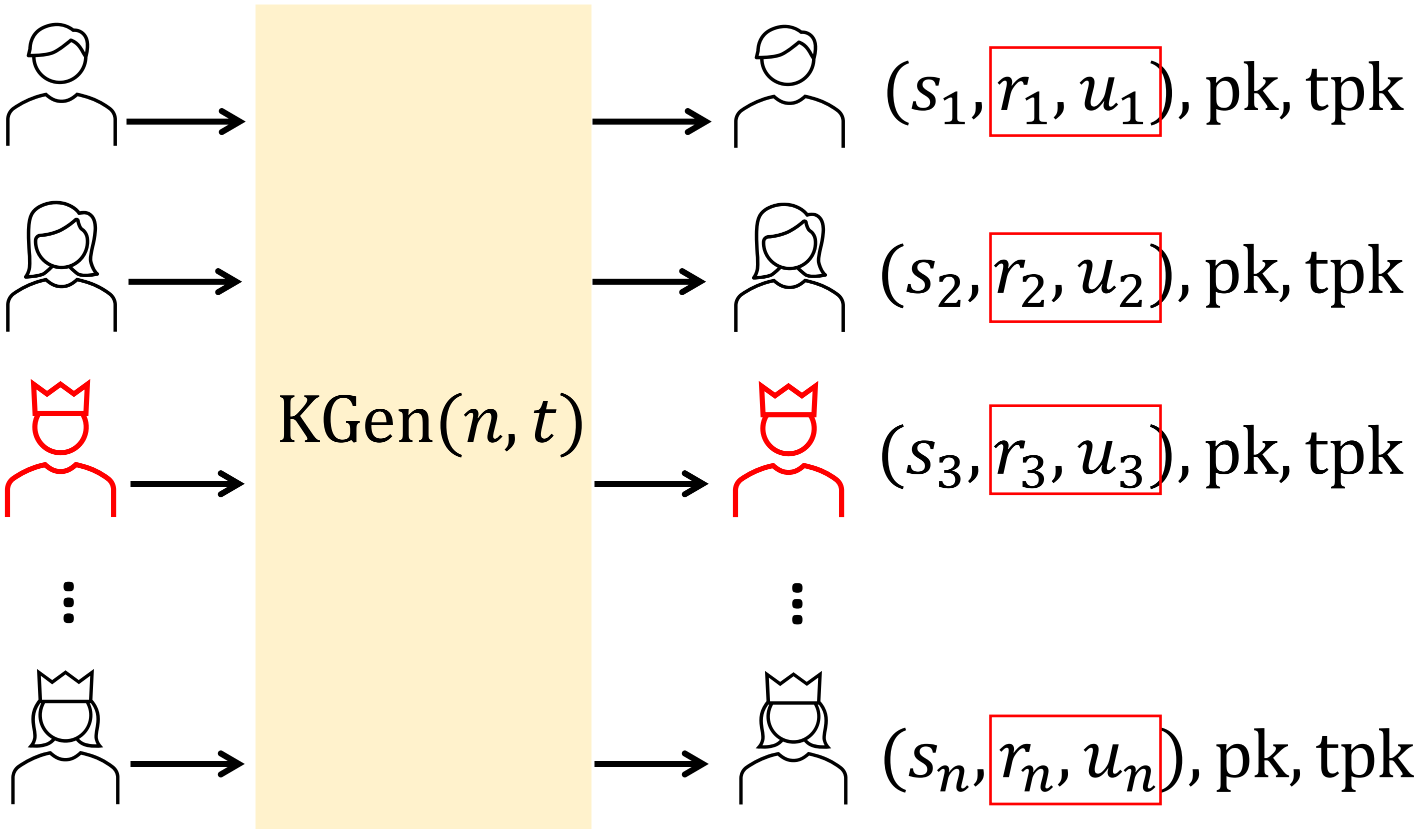
⋮

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$
$\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)$
$\{u_1, \dots, u_n\} \leftarrow \mathrm{Share}(0)$

Our Final Protocol: Key Generation



KGen(n, t)

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

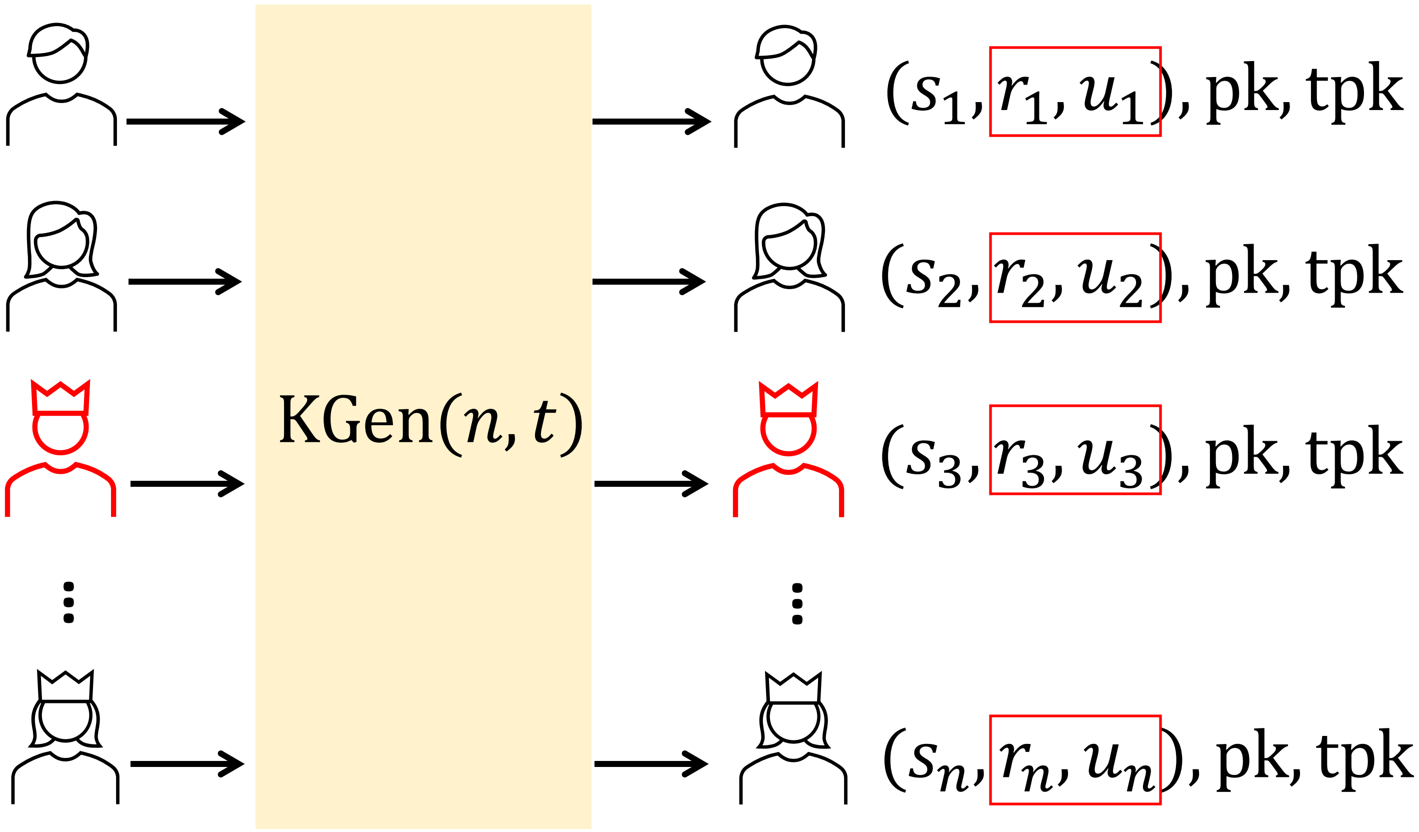
$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0)$$
$$\{u_1, \dots, u_n\} \leftarrow \mathrm{Share}(0)$$
$$\mathrm{pk} = g^s$$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

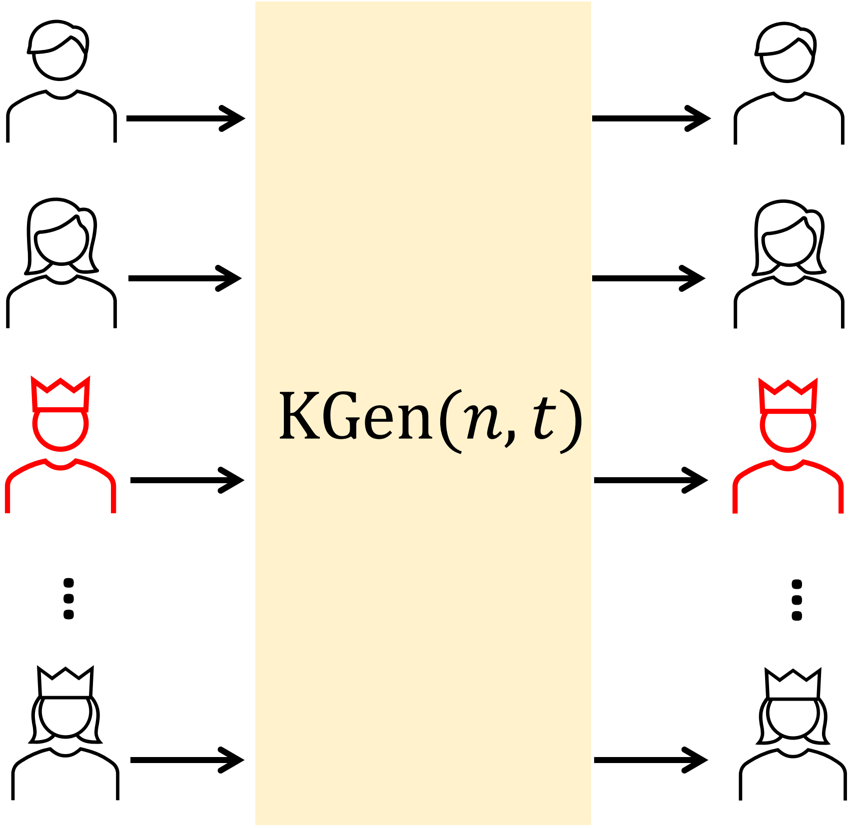
$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$$\{s_1, \ldots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\{r_1, \ldots, r_n\} \leftarrow \mathrm{Share}(0)$$
$$\{u_1, \ldots, u_n\} \leftarrow \mathrm{Share}(0)$$
$$\mathrm{pk} = g^s$$
$$\mathrm{tpk}_i = g^{s_i} h^{r_i} v^{u_i}$$

Our Final Protocol: Key Generation



$\text{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \text{pk}, \text{tpk}$

$(s_2, \boxed{r_2, u_2}), \text{pk}, \text{tpk}$

$(s_3, \boxed{r_3, u_3}), \text{pk}, \text{tpk}$

$(s_n, \boxed{r_n, u_n}), \text{pk}, \text{tpk}$

$(g, h, v) \in \mathbb{G}^3$

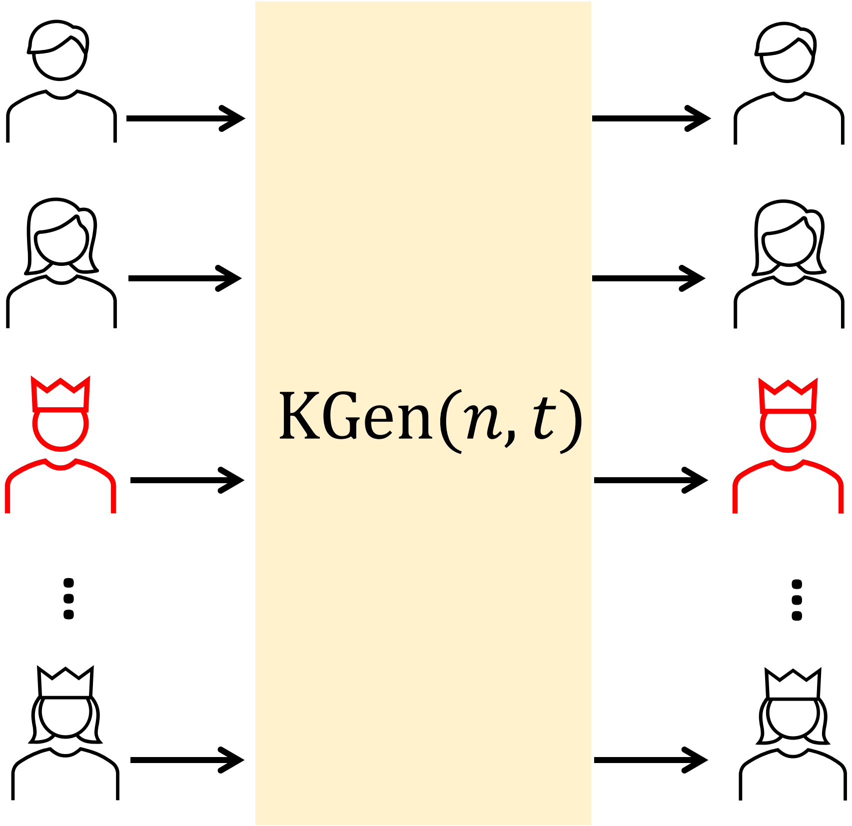
$$\{s_1, \ldots, s_n\} \leftarrow \text{Share}(s)$$

$$\boxed{\begin{gathered}\{r_1, \ldots, r_n\} \leftarrow \text{Share}(0) \\ \{u_1, \ldots, u_n\} \leftarrow \text{Share}(0)\end{gathered}}$$

$$\text{pk} = g^s$$

$$\text{tpk}_i = g^{s_i} h^{r_i} v^{u_i}$$

Our Final Protocol: Key Generation



$\mathsf{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

⋮

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

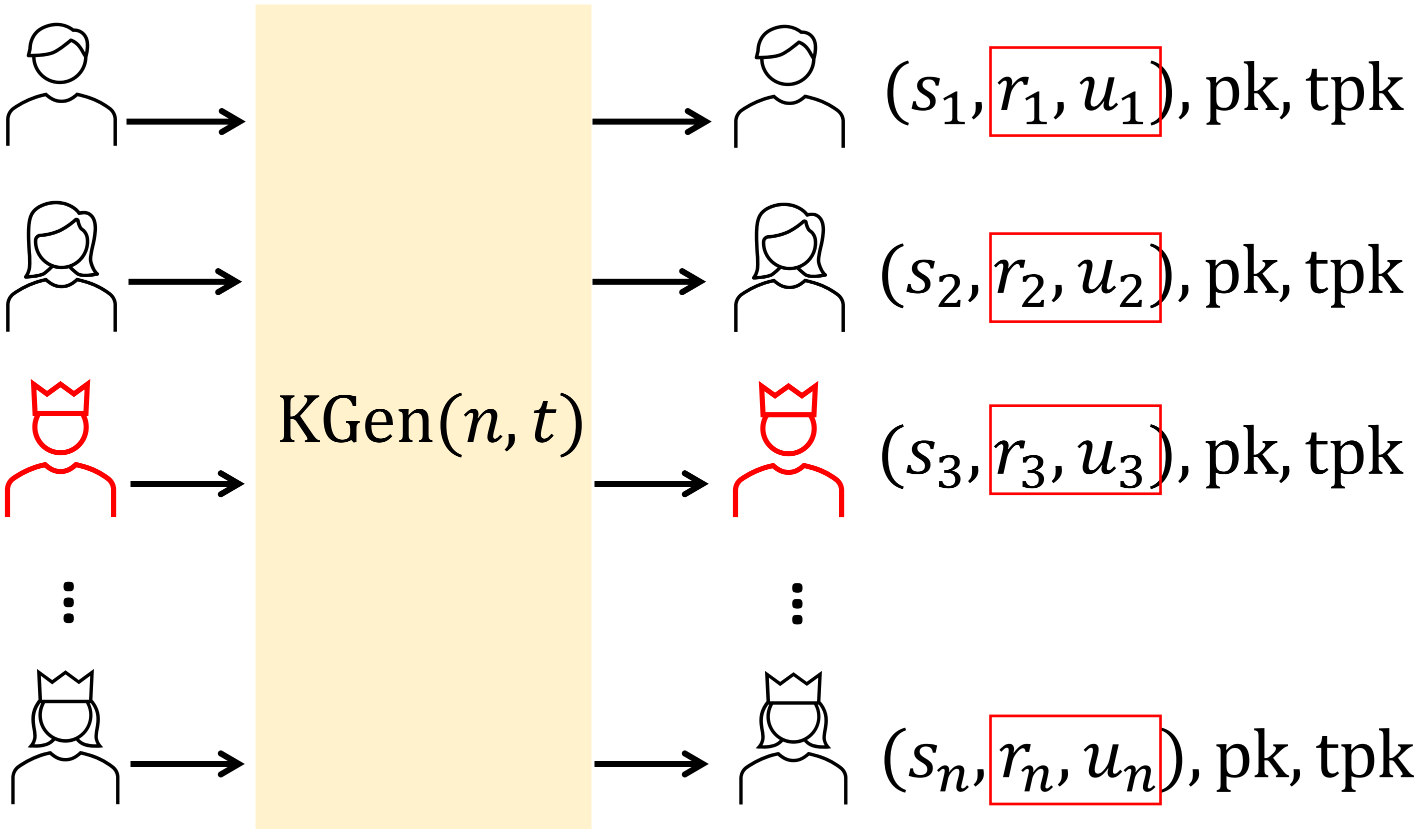
$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$

$$\boxed{\begin{array}{c} \{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0) \\ \{u_1, \dots, u_n\} \leftarrow \mathrm{Share}(0) \end{array}}$$

$$\mathrm{pk} = g^s$$

$$\mathrm{tpk}_i = g^{s_i} h^{r_i} v^{u_i}$$

Our Final Protocol: Key Generation



$\mathrm{KGen}(n, t)$

$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$

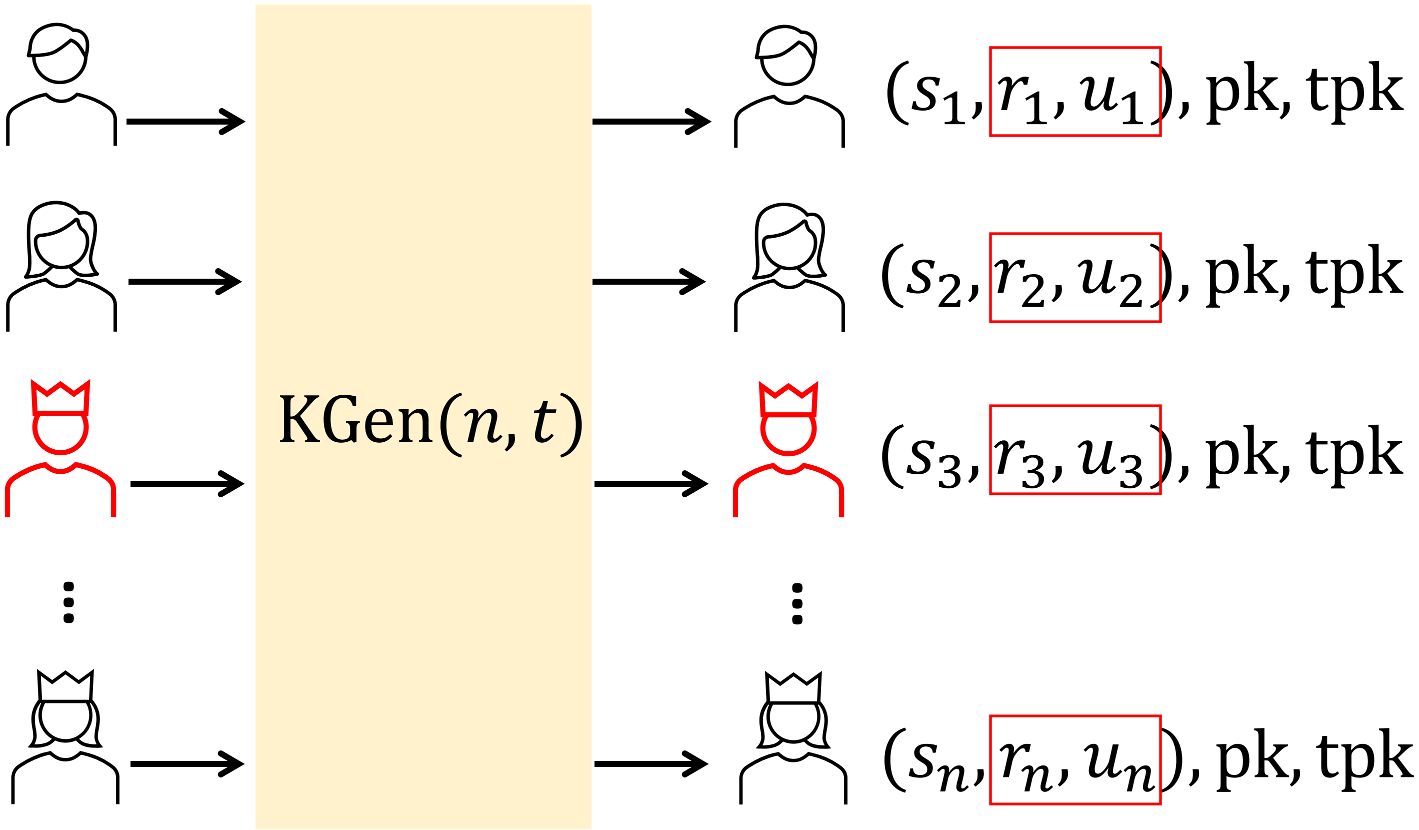
$$\boxed{\begin{array}{c}\{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0) \\ \{u_1, \dots, u_n\} \leftarrow \mathrm{Share}(0)\end{array}}$$

$$\mathrm{pk} = g^s$$

$$\mathrm{tpk}_i = g^{s_i} h^{r_i} v^{u_i}$$

Signing step is same as the two generators protocol

Our Final Protocol: Key Generation



$(s_1, \boxed{r_1, u_1}), \mathrm{pk}, \mathrm{tpk}$

$(s_2, \boxed{r_2, u_2}), \mathrm{pk}, \mathrm{tpk}$

$(s_3, \boxed{r_3, u_3}), \mathrm{pk}, \mathrm{tpk}$

⋮

$(s_n, \boxed{r_n, u_n}), \mathrm{pk}, \mathrm{tpk}$

$(g, h, v) \in \mathbb{G}^3$

$$\{s_1, \dots, s_n\} \leftarrow \mathrm{Share}(s)$$
$$\boxed{\begin{array}{c} \{r_1, \dots, r_n\} \leftarrow \mathrm{Share}(0) \\ \{u_1, \dots, u_n\} \leftarrow \mathrm{Share}(0) \end{array}}$$
$$\mathrm{pk} = g^s$$
$$\mathrm{tpk}_i = g^{s_i} h^{r_i} v^{u_i}$$

Signing step is same as the two generators protocol

See paper for the proof!

Implementation and Evaluation

Evaluation

Evaluation

- gnark-crypto library, bls12-381 curve

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
    - Boldyreva-I: Pairing based partial signature verification

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
    - Boldyreva-I: Pairing based partial signature verification
    - Boldyreva-II: Σ-protocol based partial signature verification

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
    - Boldyreva-I: Pairing based partial signature verification
    - Boldyreva-II: Σ-protocol based partial signature verification
- Multi-paring and multi-exponentiations

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
  - Boldyreva-I: Pairing based partial signature verification
  - Boldyreva-II: Σ-protocol based partial signature verification
- Multi-paring and multi-exponentiations

Scheme	Signing time (ms)	Partial signature verification time (ms)	Partial signature size (bytes)
Boldyreva-I	0.81	1.12	96
Boldyreva-II	1.20	0.76	160
Ours	3.92	2.16	224

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
  - Boldyreva-I: Pairing based partial signature verification
  - Boldyreva-II: Σ-protocol based partial signature verification
- Multi-paring and multi-exponentiations

Scheme	Signing time (ms)	Partial signature verification time (ms)	Partial signature size (bytes)
Boldyreva-I	0.81	1.12	96
Boldyreva-II	1.20	0.76	160
Ours	3.92	2.16	224

Overhead is primarily due to Σ-protocol proof.

Evaluation

- gnark-crypto library, bls12-381 curve
- **Baselines:**
  - Boldyreva-I: Pairing based partial signature verification
  - Boldyreva-II: Σ-protocol based partial signature verification
- Multi-paring and multi-exponentiations

Scheme	Signing time (ms)	Partial signature verification time (ms)	Partial signature size (bytes)
Boldyreva-I	0.81	1.12	96
Boldyreva-II	1.20	0.76	160
Ours	3.92	2.16	224

Overhead is primarily due to Σ-protocol proof.

Common case aggregation time (for t=64) is 7.7 ms for all three schemes!

Summary and Open Problems

Summary and Open Problems

Adaptively secure threshold BLS

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission
- Open Problems

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission
- Open Problems
  - Improve its efficiency

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission
- Open Problems
    - Improve its efficiency
    - Extend to other threshold cryptosystems

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission
- Open Problems
  - Improve its efficiency
  - Extend to other threshold cryptosystems

Paper link

Implementation

Summary and Open Problems

Adaptively secure threshold BLS

- Simple and efficient
- Preparing a NIST submission
- Open Problems
    - Improve its efficiency
    - Extend to other threshold cryptosystems

Paper link

Implementation

My website

Thank you (souravd2@Illinois.edu)