Monotone-Policy Aggregate Signatures

Maya Farber Brodsky
Tel Aviv University

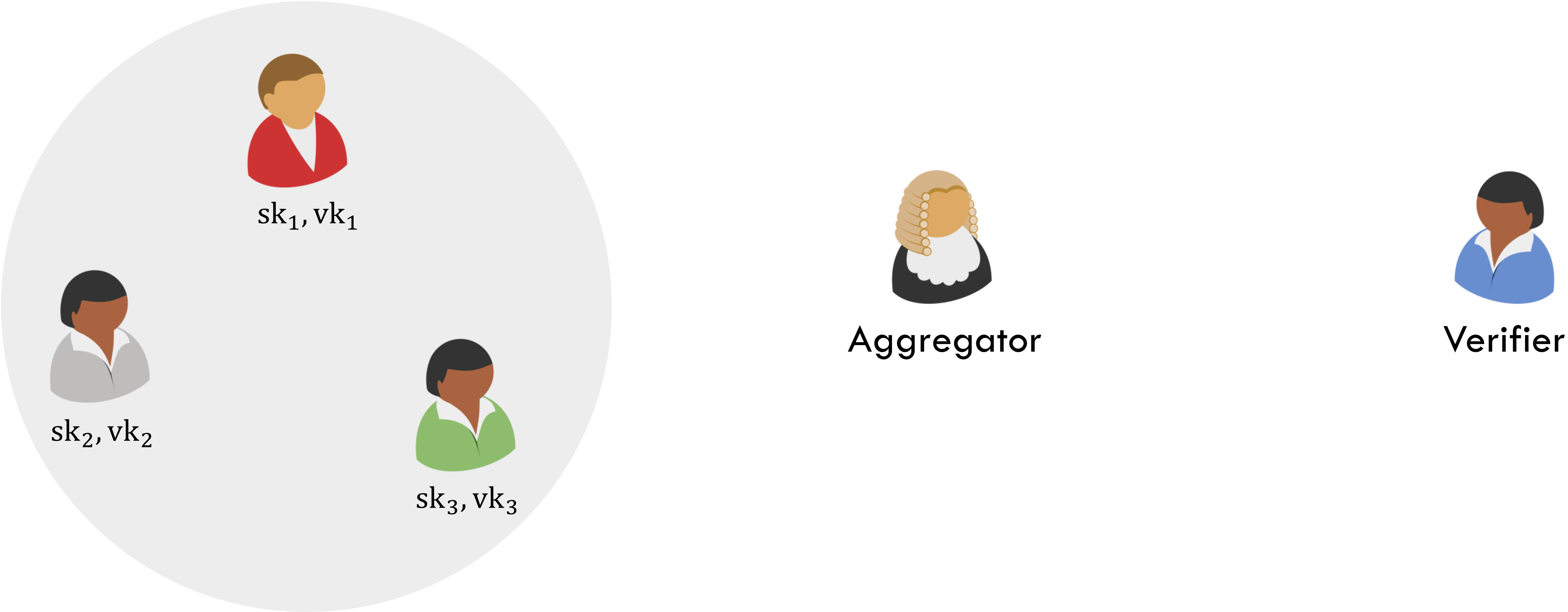
Arka Rai Choudhuri
NTT Research

Abhishek Jain
Johns Hopkins University
and NTT Research

Omer Paneth
Tel Aviv University

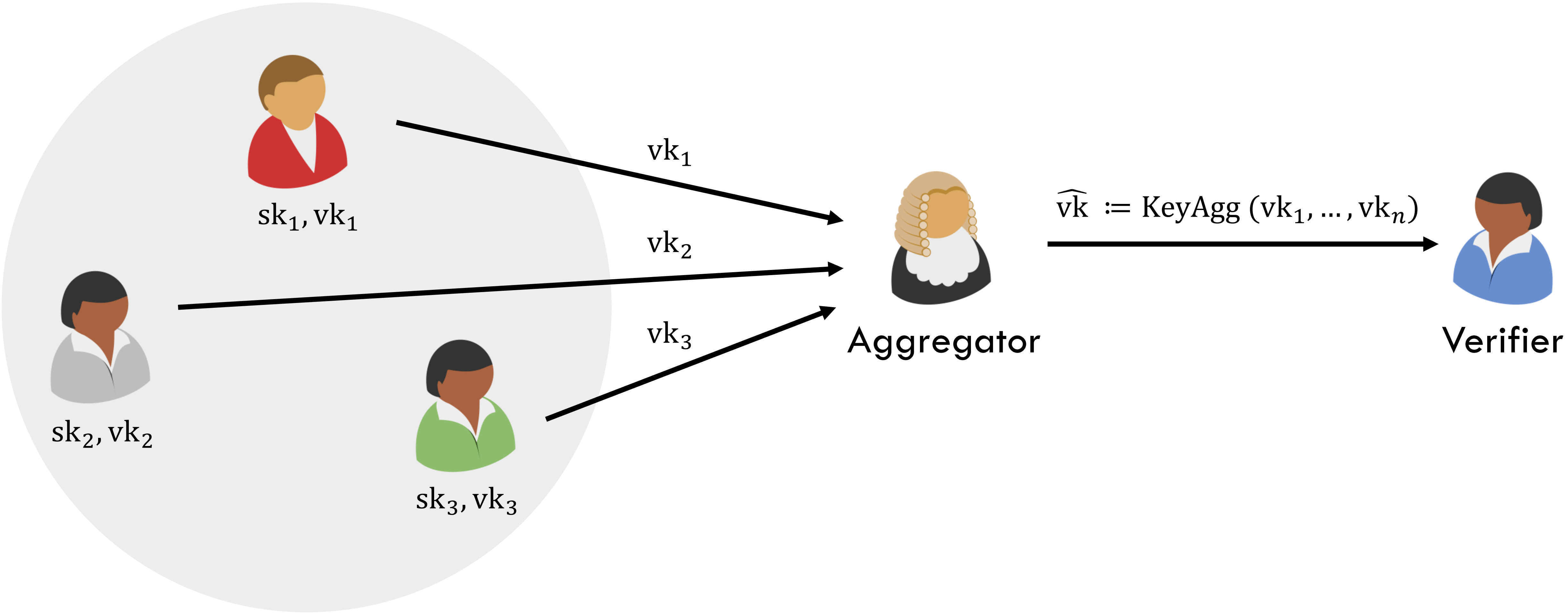
Aggregate Signatures

[Itakura-Nakmura'83, Boneh-Gentry-Lynn-Shacham'03]



Aggregate Signatures

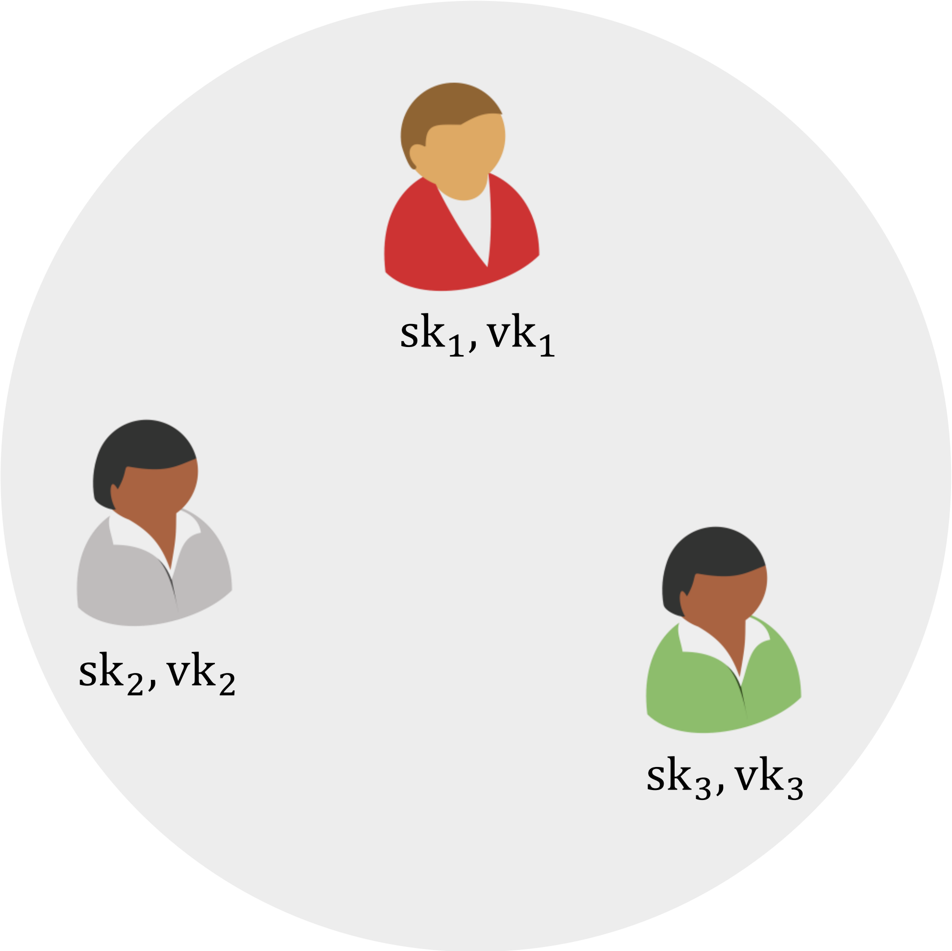
[Itakura-Nakmura'83, Boneh-Gentry-Lynn-Shacham'03]



Aggregate Signatures

[Itakura-Nakmura'83, Boneh-Gentry-Lynn-Shacham'03]

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$



Aggregator

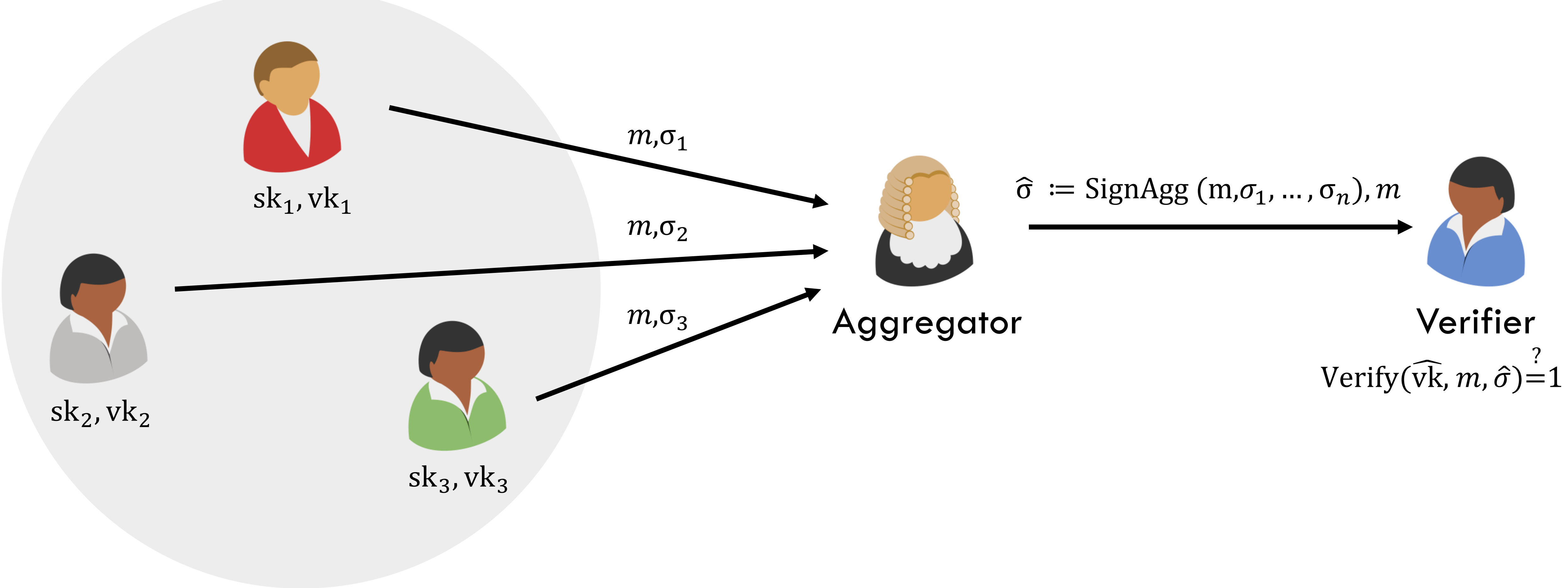
Verifier

$\mathrm{Verify}(\widehat{\mathrm{vk}}, m, \hat{\sigma}) \stackrel{?}{=} 1$

Aggregate Signatures

[Itakura-Nakmura’83, Boneh-Gentry-Lynn-Shacham’03]

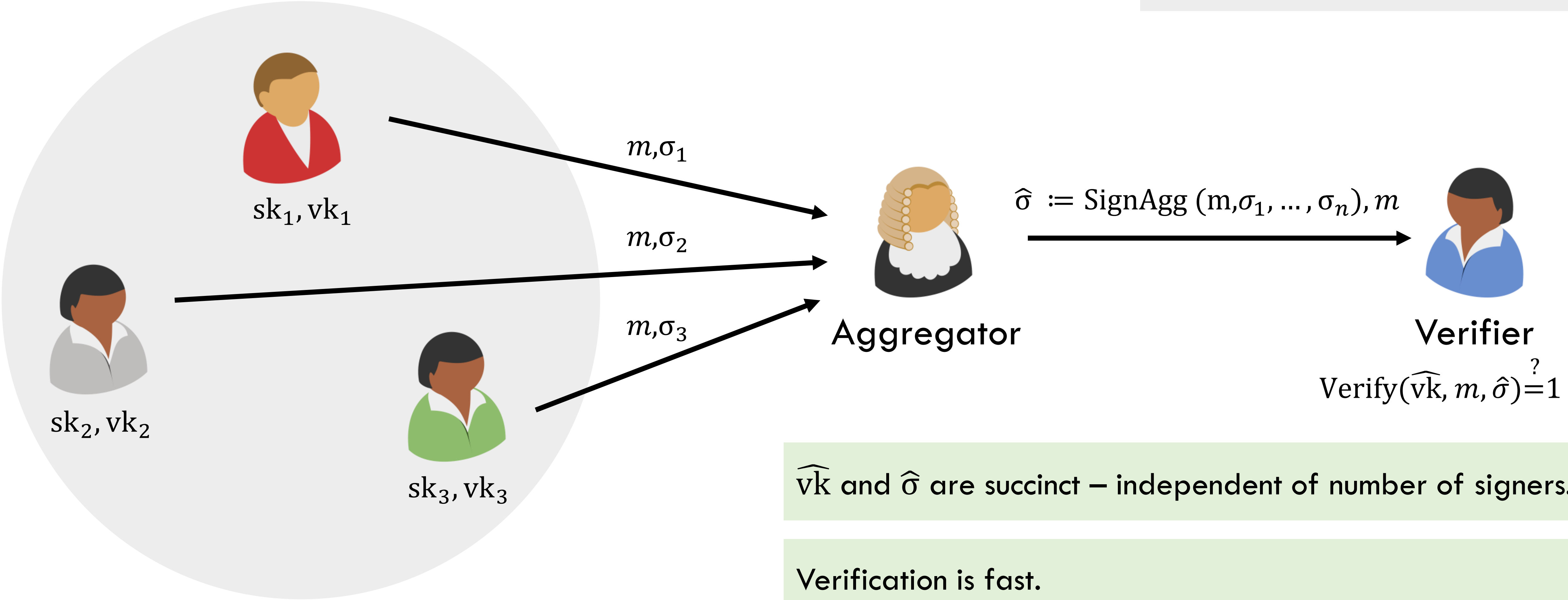
$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$



Aggregate Signatures

[Itakura-Nakmura'83, Boneh-Gentry-Lynn-Shacham'03]

$\widehat{\mathrm{vk}} := \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

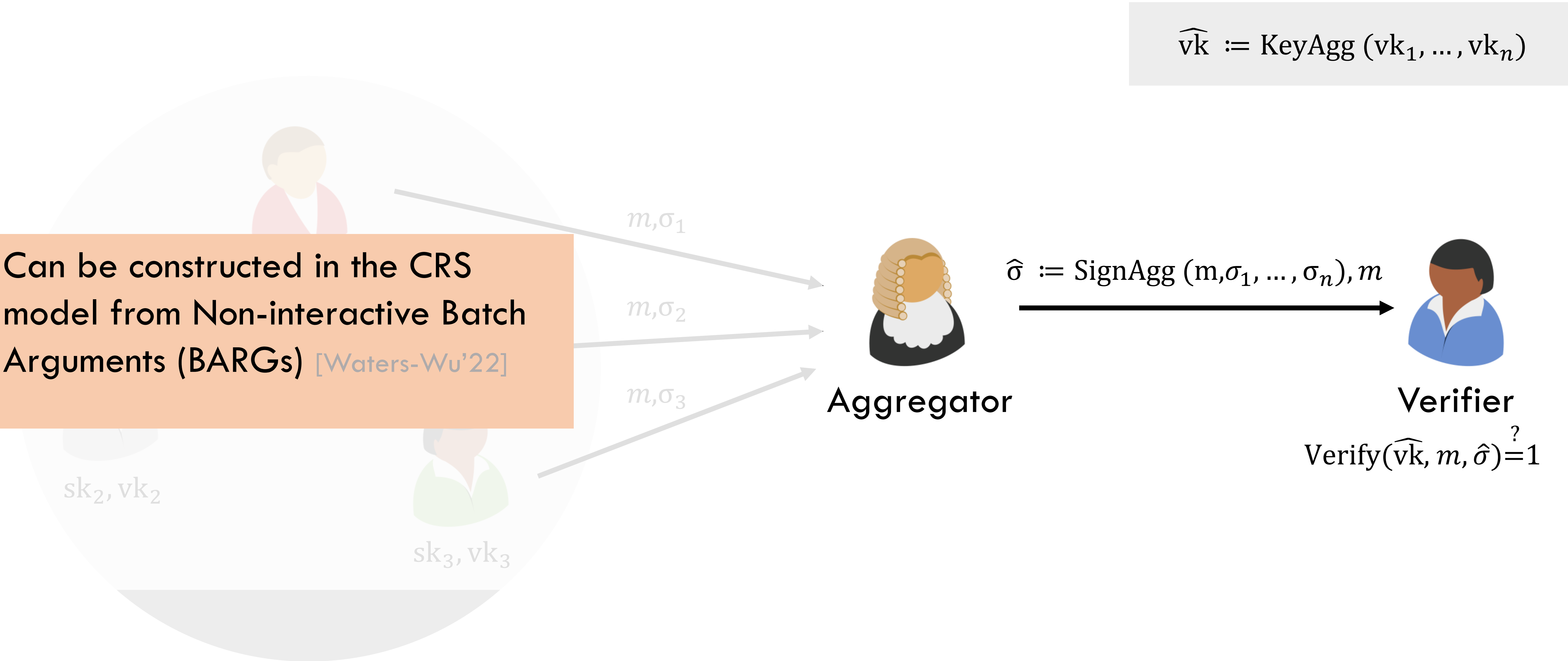


$\widehat{\mathrm{vk}}$ and $\hat{\sigma}$ are succinct – independent of number of signers.

Verification is fast.

Aggregate Signatures

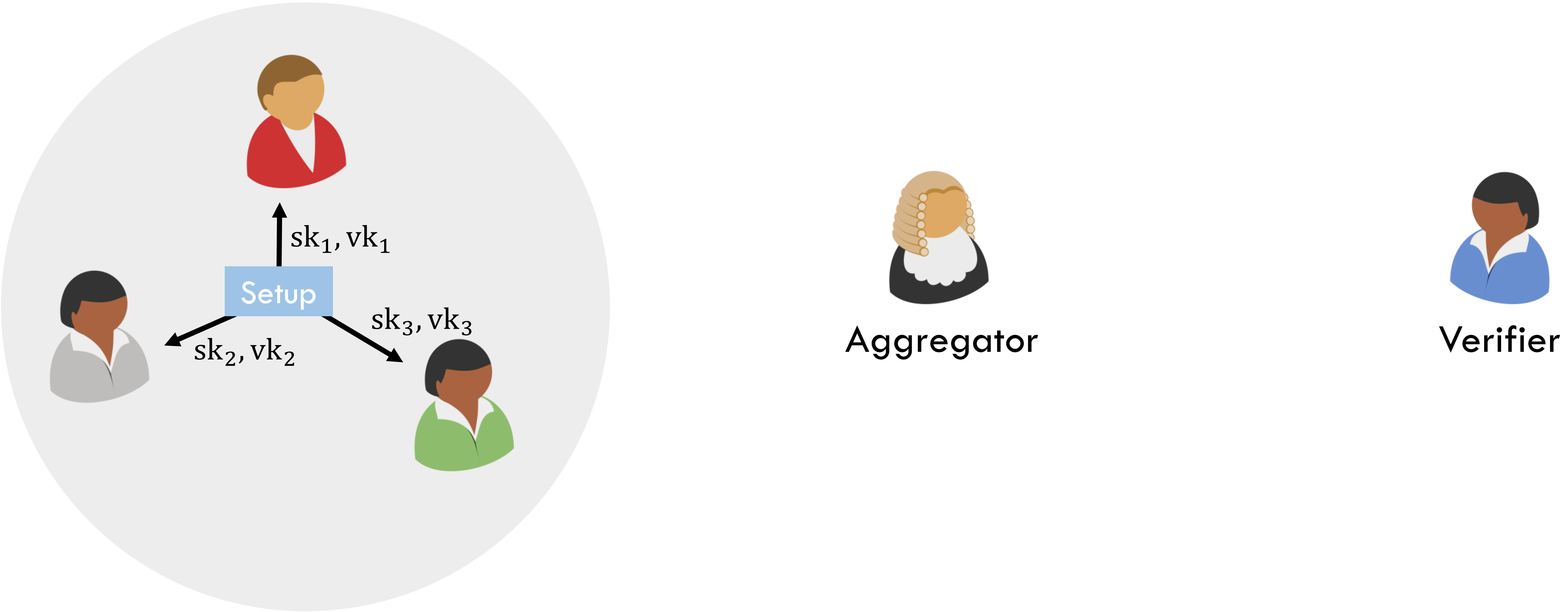
[Itakura-Nakmura’83, Boneh-Gentry-Lynn-Shacham’03]



Can be constructed in the CRS model from Non-interactive Batch Arguments (BARGs) [Waters-Wu’22]

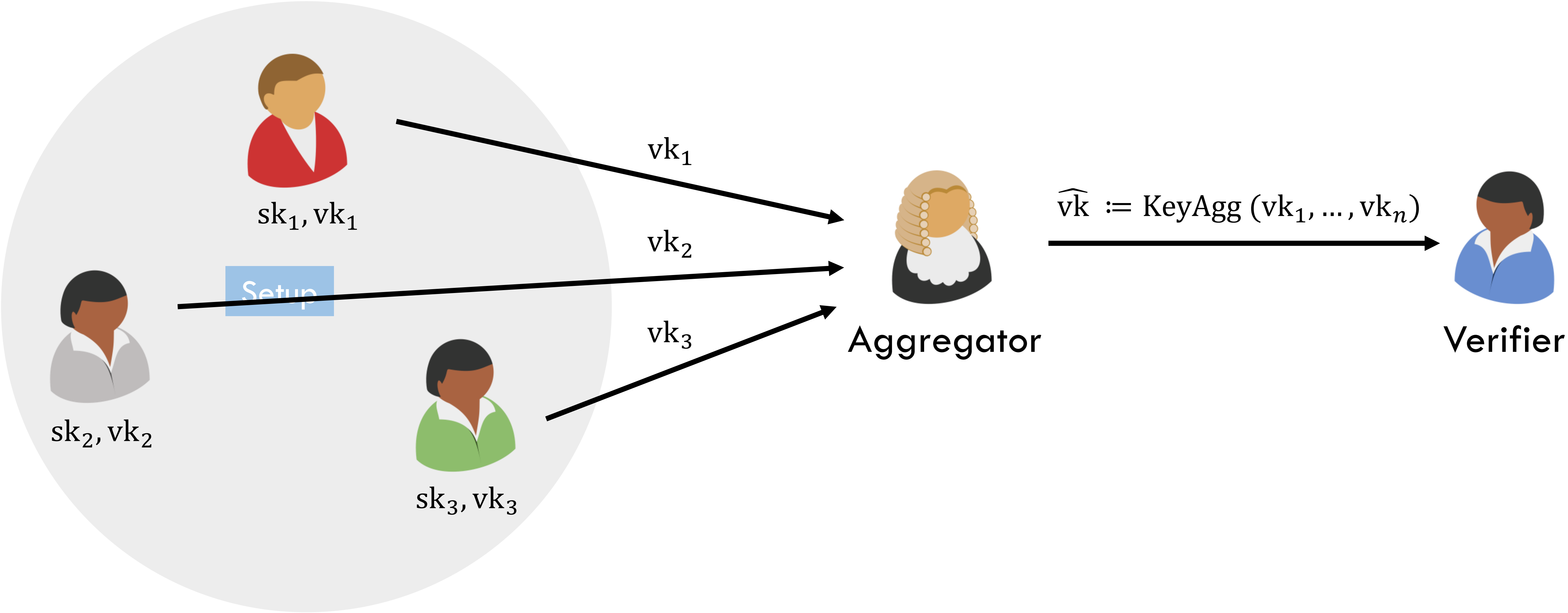
Threshold Signatures

[Desmedt-Frankel'89]



Threshold Signatures

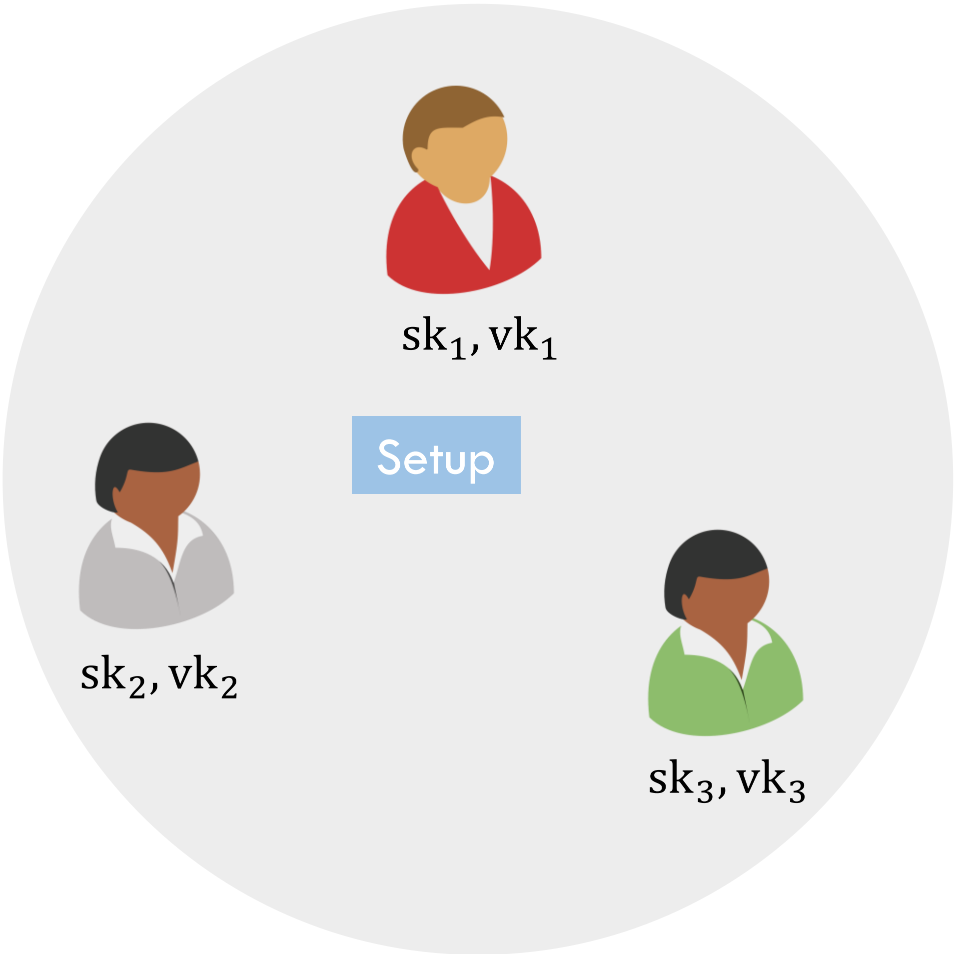
[Desmedt-Frankel'89]



Threshold Signatures

[Desmedt-Frankel'89]

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$



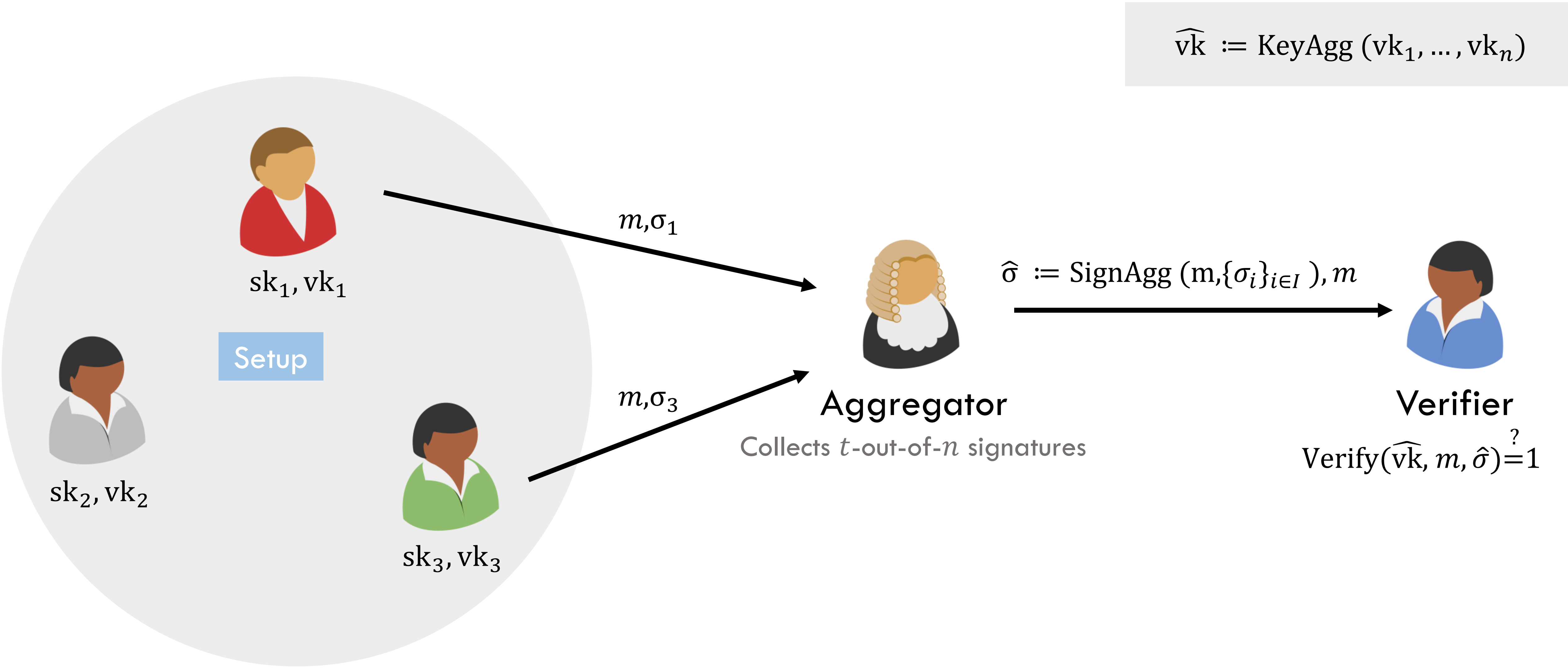
Aggregator

Verifier

$\mathrm{Verify}(\widehat{\mathrm{vk}}, m, \hat{\sigma}) \stackrel{?}{=} 1$

Threshold Signatures

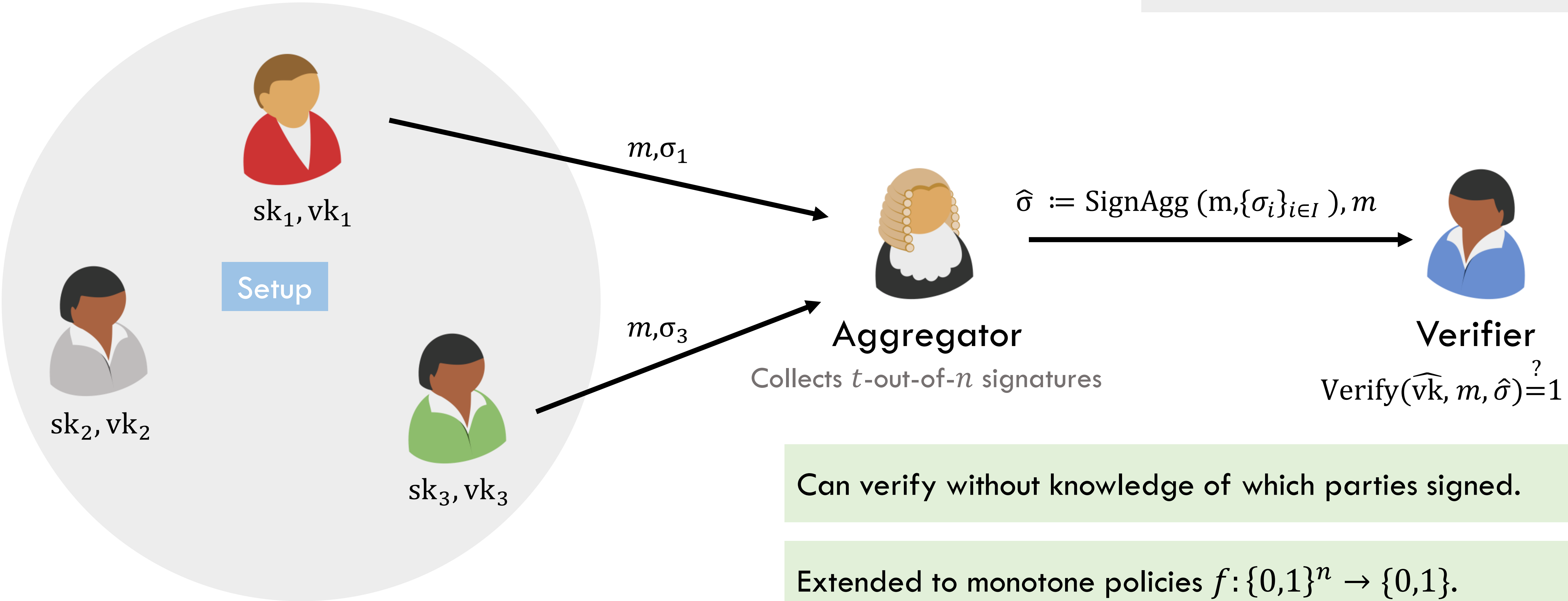
[Desmedt-Frankel'89]



Threshold Signatures

[Desmedt-Frankel'89]

$\widehat{\mathrm{vk}} := \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$



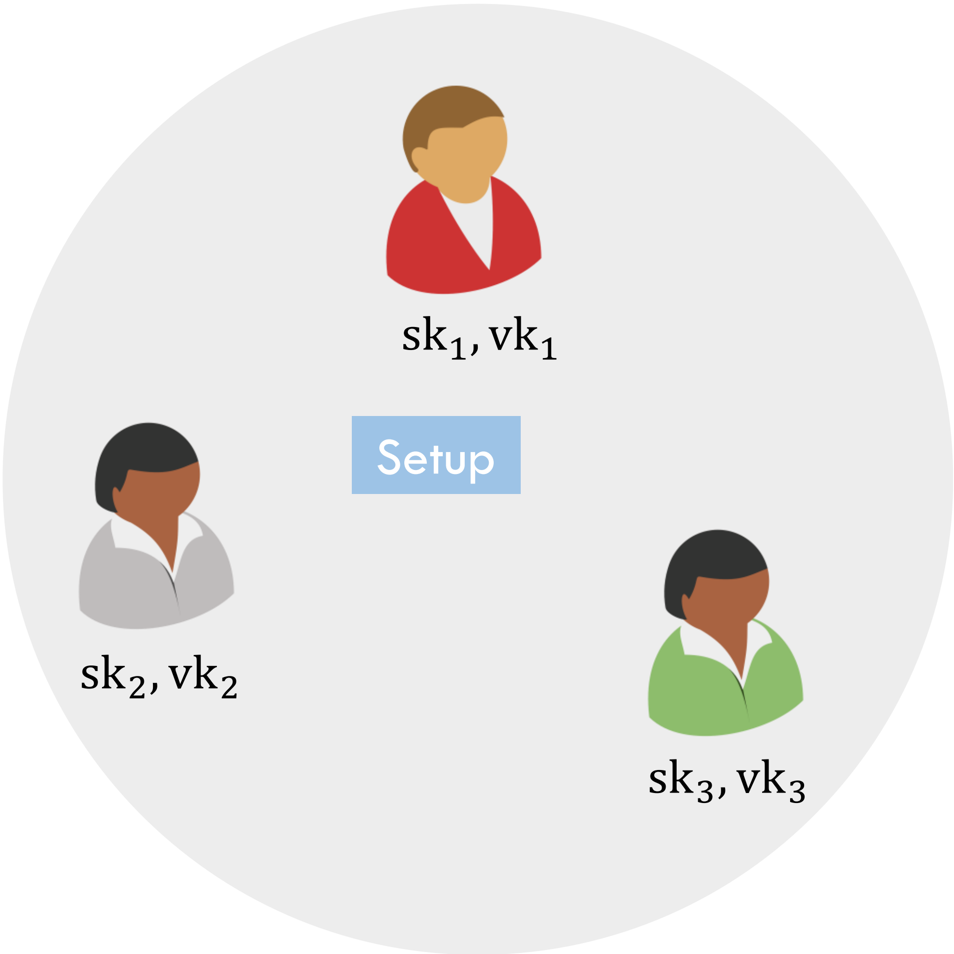
Can verify without knowledge of which parties signed.

Extended to monotone policies $f\colon \{0,1\}^n \to \{0,1\}$.

Threshold Signatures

[Desmedt-Frankel'89]

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$



m, σ_1

m, σ_3

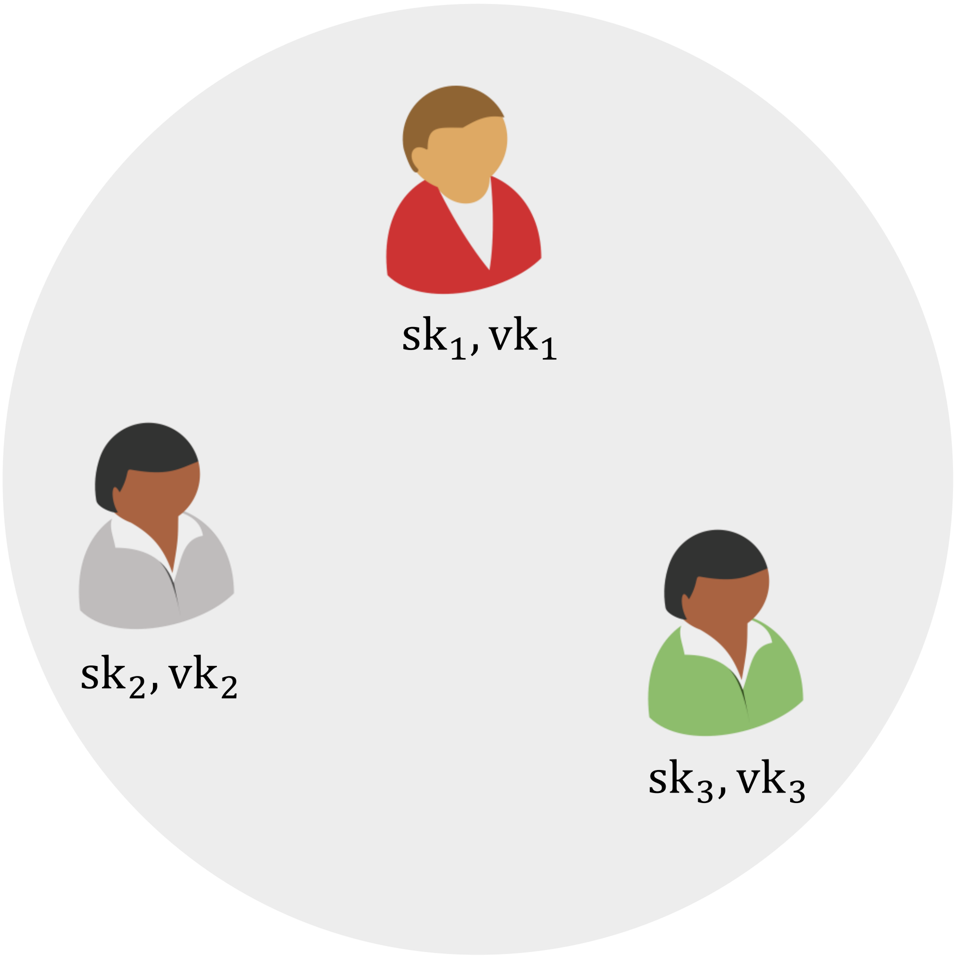
Recent works [Garg-Jain-Mukherjee-Sinha-Wang-Zhang'24, Das-Camacho-Xiang-Nieto-Bünz-Ren'23] remove the need for an interactive "Setup" for specific schemes, but in idealized models.

$\mathrm{Verify}(\mathrm{vk}, m, \hat{\sigma}) = 1$

Can verify without knowledge of which parties signed.

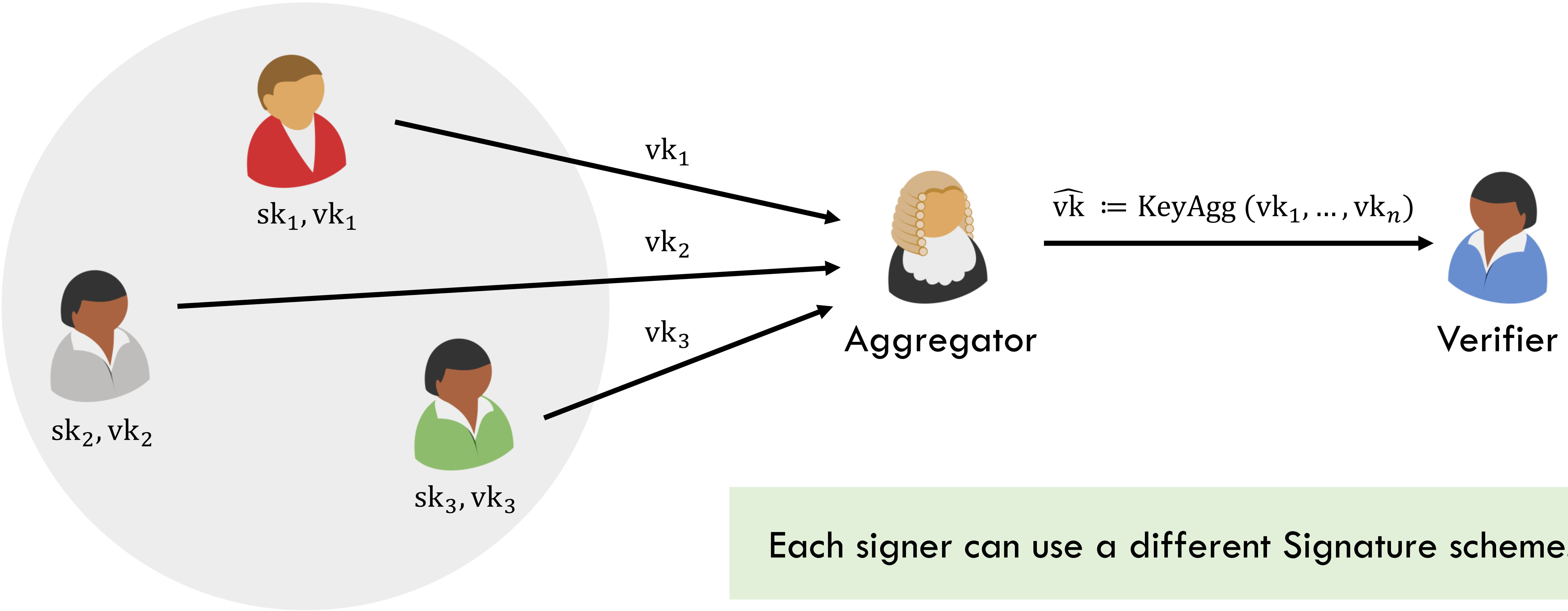
Extended to monotone policies $f : \{0,1\}^n \to \{0,1\}$.

Monotone Policy Aggregate Signatures



Each signer can use a different Signature scheme.

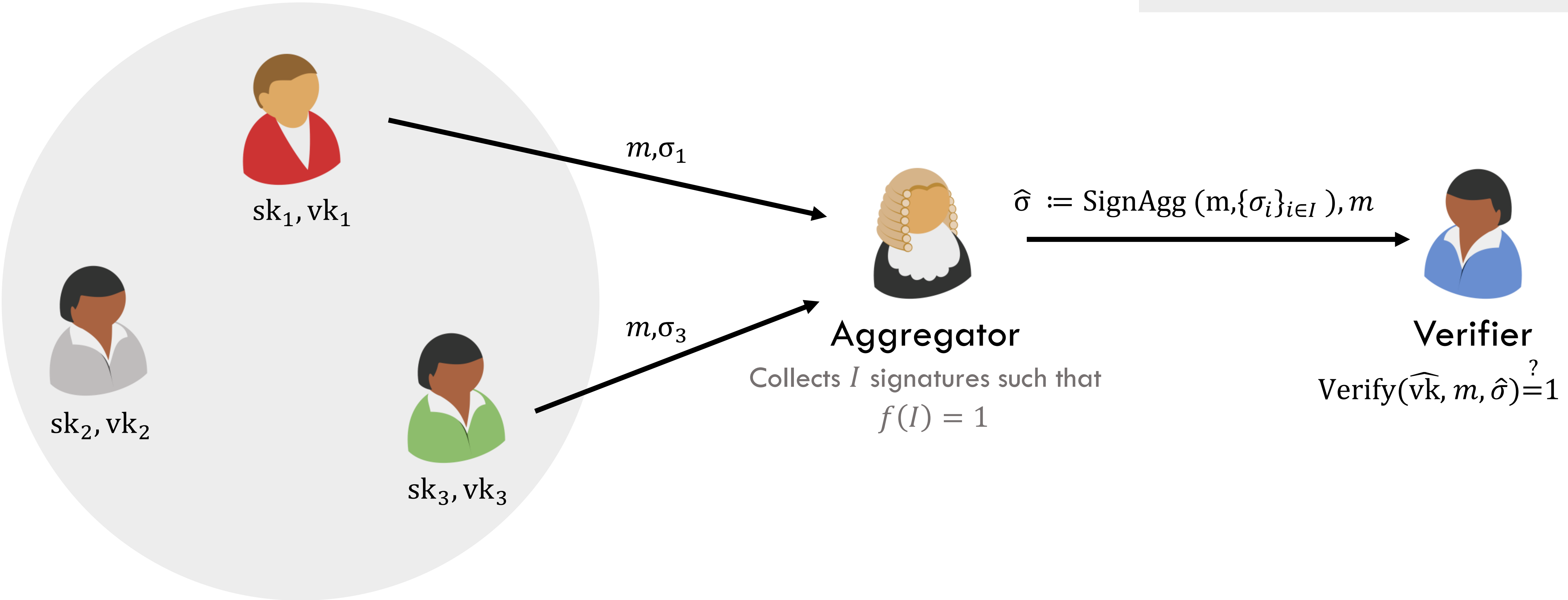
Monotone Policy Aggregate Signatures



Each signer can use a different Signature scheme.

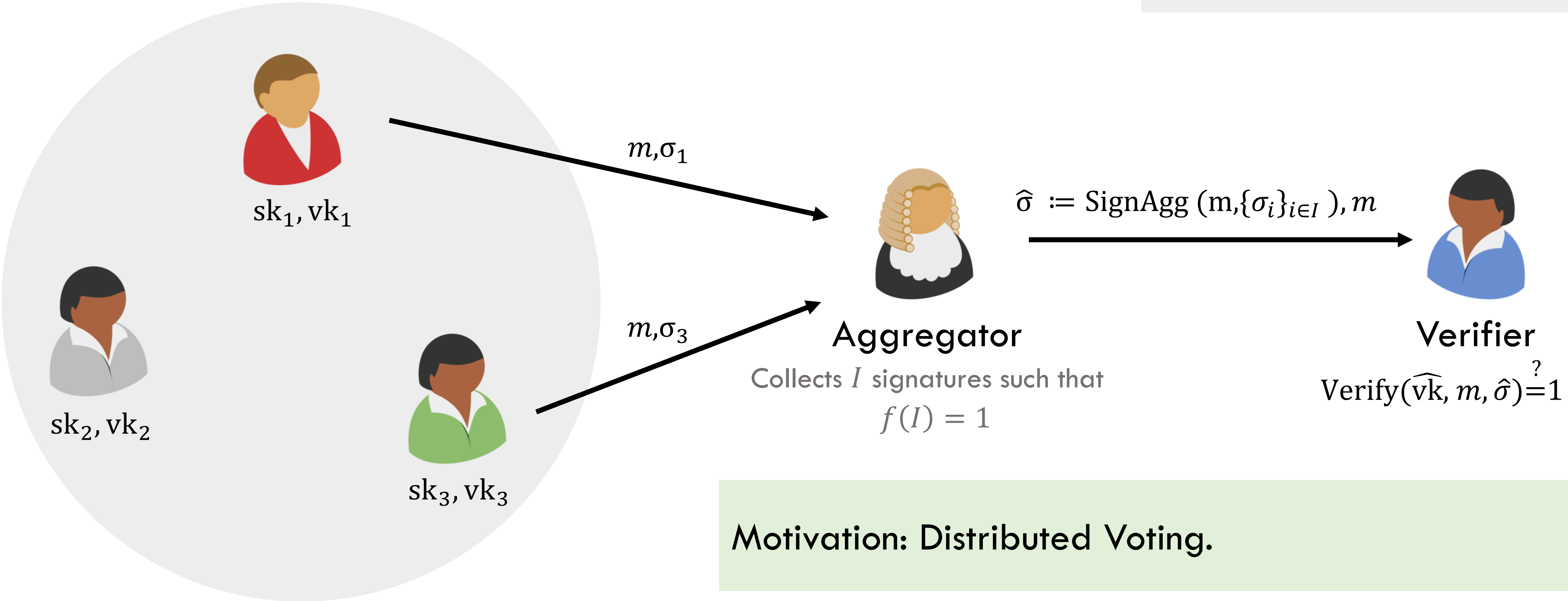
Monotone Policy Aggregate Signatures

$\widehat{\mathrm{vk}} := \mathrm{KeyAgg}(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$



Monotone Policy Aggregate Signatures

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$



Motivation: Distributed Voting.

Security Monotone Policy Aggregate Signatures

Challenger

Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

key

vk_i

Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

key

vk_i

$\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}_i, m)$

sign vk_i, m

σ

Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

key

vk_i

$\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}_i, m)$

sign vk_i, m

σ

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

key

vk_i

$\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}_i, m)$

sign vk_i, m

σ

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

key

vk_i

$\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}_i, m)$

sign vk_i, m

σ

vk$_i$ generated by

sign vk_i, m^* queried

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

Security Monotone Policy Aggregate Signatures

Challenger

$sk_i, vk_i \leftarrow \text{KeyGen}()$

$\sigma \leftarrow \text{Sign}(sk_i, m)$

wins if

1. $\text{Verify}(\widehat{vk}, m^*, \hat{\sigma})=1$
2. $f(\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}) \neq 1$

vk_i generated by

sign vk_i, m^* queried

$vk_1, \ldots, vk_n$

$m^*, \hat{\sigma}$

$\widehat{vk} := \text{KeyAgg}(vk_1, \ldots, vk_n)$

$vk_1 \quad \cdots \quad vk_i \quad \cdots \quad vk_n$

Weak Security Monotone Policy Aggregate Signatures

Challenger

$\mathrm{sk}_i, \mathrm{vk}_i \leftarrow \mathrm{KeyGen}()$

$\sigma \leftarrow \mathrm{Sign}(\mathrm{sk}_i, m)$

 wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. $f($ 0 1 0 1 0 0 0 $) \neq 1$
3. No ▢

▢ vk_i generated by 

▢ sign vk_i, m^* queried

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

Our Results

Non-interactive batch arguments. → Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

Our Results

BARGs → Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

Our Results

BARGs → Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

Examples: threshold, weighted threshold.

Aggregation time in threshold grows with threshold t (not n).

Construction of BARGs

LWE

[C-Jain-Jin'21]

DLIN

[Waters-Wu'22]

Sub-exp DDH
+ QR

[C-Jain-Jin'21a, Hulett-Jawale-Khurana-Srinivasan'22]

Sub-exp DDH

[C-Garg-Jain-Jin-Zhang'23]

BARGs

QR – Quadratic residuosity, LWE – Learning with Error, DDH – Decisional Diffie-Hellman, DLIN – Decisional Linear Assumption over Bilinear Groups.

Our Results

BARGs $\rightarrow$ Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

BARGs $\xrightarrow{+\ \text{FHE}}$ Weakly Secure monotone Policy Aggregate Signatures for poly size monotone circuits.

Our Results

BARGs

Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

+ FHE

Weakly Secure monotone Policy Aggregate Signatures for poly size monotone circuits.

Recently [Nassar-Waters-Wu'24] construct static secure monotone Policy Aggregate Signatures for poly size monotone circuits challenge message is declared first.

Non-Interactive Batch Arguments (or BARGs)

CRS

$x_1, \cdots, x_n$

$w_1, \cdots, w_n$

$x_1, \cdots, x_n$

$L = \{x \mid \exists w \ s.t. \ R_L(x, w) = 1\}$

$\forall i \in [n], x_i \in L$

Non-Interactive Batch Arguments (or BARGs)

CRS

Π

$x_1, \cdots, x_n$

$w_1, \cdots, w_n$

$x_1, \cdots, x_n$

Π is publicly verifiable

$L = \{x \mid \exists w \; s.t. \; R_L(x, w) = 1\}$

$\forall i \in [n], x_i \in L$

Non-Interactive Batch Arguments (or BARGs)

CRS

$x_1, \cdots, x_n$

$w_1, \cdots, w_n$

$|w| \cdot \mathrm{poly}(\lambda)$

Π

$x_1, \cdots, x_n$

Π is publicly verifiable

$L = \{x \mid \exists w \ s.t.\ R_L(x, w) = 1\}$

$\forall i \in [n], x_i \in L$

Non-Interactive Batch Arguments (or BARGs)

CRS

$x_1, \cdots, x_n$

$w_1, \cdots, w_n$

$|w| \cdot \mathrm{poly}(\lambda)$

Π

$x_1, \cdots, x_n$

Π is publicly verifiable

$L = \{x \mid \exists w \ s.t. \ R_L(x, w) = 1\}$

$\forall i \in [n], x_i \in L$

No PPT  can produce accepting Π if

$\exists i^* \in [n], x_{i^*} \cancel{\in} L$

somewhere extractable BARGs (seBARGs)

[C-Jain-Jin’21]

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$

somewhere extractable BARGs (seBARGs)

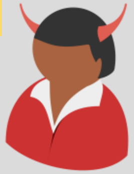
[C-Jain-Jin'21]

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS

$\approx_c$

CRS_j



Π

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$

CRS_j hides j

Aggregate Signatures from seBARGs

[Waters-Wu'22]

Aggregate Signatures from seBARGs

[Waters-Wu'22]

$\underline{\text{KeyAgg}\ (\text{vk}_1, \dots, \text{vk}_n)}$

$$\widehat{\text{vk}} = \text{MerkleRoot}(\text{vk}_1, \dots, \text{vk}_n)$$

Aggregate Signatures from seBARGs

[Waters-Wu'22]

$\text{KeyAgg}\ (\text{vk}_1, \ldots, \text{vk}_n)$

$\widehat{\text{vk}} = \text{MerkleRoot}(\text{vk}_1, \ldots, \text{vk}_n)$

$\text{SignAgg}\ (m, \sigma_1, \ldots, \sigma_n)$

$x_i = i, m, \widehat{\text{vk}}$

$w_i = \text{vk}_i, \sigma_i, \pi_i$

$\hat{\sigma} = \text{BARG}(x_1 \ldots, x_n, w_1, \ldots, w_n)$

Statement: $i, m, \widehat{\text{vk}}$

Witness: $\text{vk}_i, \sigma_i, \pi_i$

s.t. π_i is a valid opening to vk_i AND $\text{Verify}(\text{vk}_i, m, \sigma_i) = 1$

Aggregate Signatures from seBARGs

[Waters-Wu'22]

KeyAgg $(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$

$\widehat{\mathrm{vk}} = \mathrm{MerkleRoot}(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$

SignAgg $(m, \sigma_1, \dots, \sigma_n)$

$x_i = i, m, \widehat{\mathrm{vk}}$

$w_i = \mathrm{vk}_i, \sigma_i, \pi_i$

$\hat{\sigma} = \mathrm{BARG}(x_1 \dots, x_n, w_1, \dots, w_n)$

Statement: $i, m, \widehat{\mathrm{vk}}$

Witness: $\mathrm{vk}_i, \sigma_i, \pi_i$

The n statements have a succinct representation - $m, \widehat{\mathrm{vk}}$, n and thus the BARG proof can be verified quickly without dependence on n.

s.t. π_i is a valid opening to vk_i AND $\mathrm{Verify}(\mathrm{vk}_i, m, \sigma_i) = 1$

Security Monotone Policy Aggregate Signatures

Challenger

CRS

key

sign

vk_i generated by

sign vk_i, m^* queried

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} := \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. $f(\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}) \neq 1$

Security ~~Monotone Policy~~ Aggregate Signatures

Challenger

CRS

key

sign

vk$_i$ generated by 

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$

$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

 wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. $f(\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{1}\boxed{1}) \neq 1$

Security Proof Aggregate Signatures

CRS

key

sign

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

Security Proof Aggregate Signatures

Signature Scheme

Challenger

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

CRS_{i^*}

`key`

`sign`

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

Security Proof Aggregate Signatures

Signature Scheme

Challenger

vk^*

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

CRS_{i^*}

vk^*

key

sign

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

Security Proof Aggregate Signatures

Signature Scheme

Challenger

vk^*

m^*, σ^*

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

CRS_{i^*}

vk^*

key

sign

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

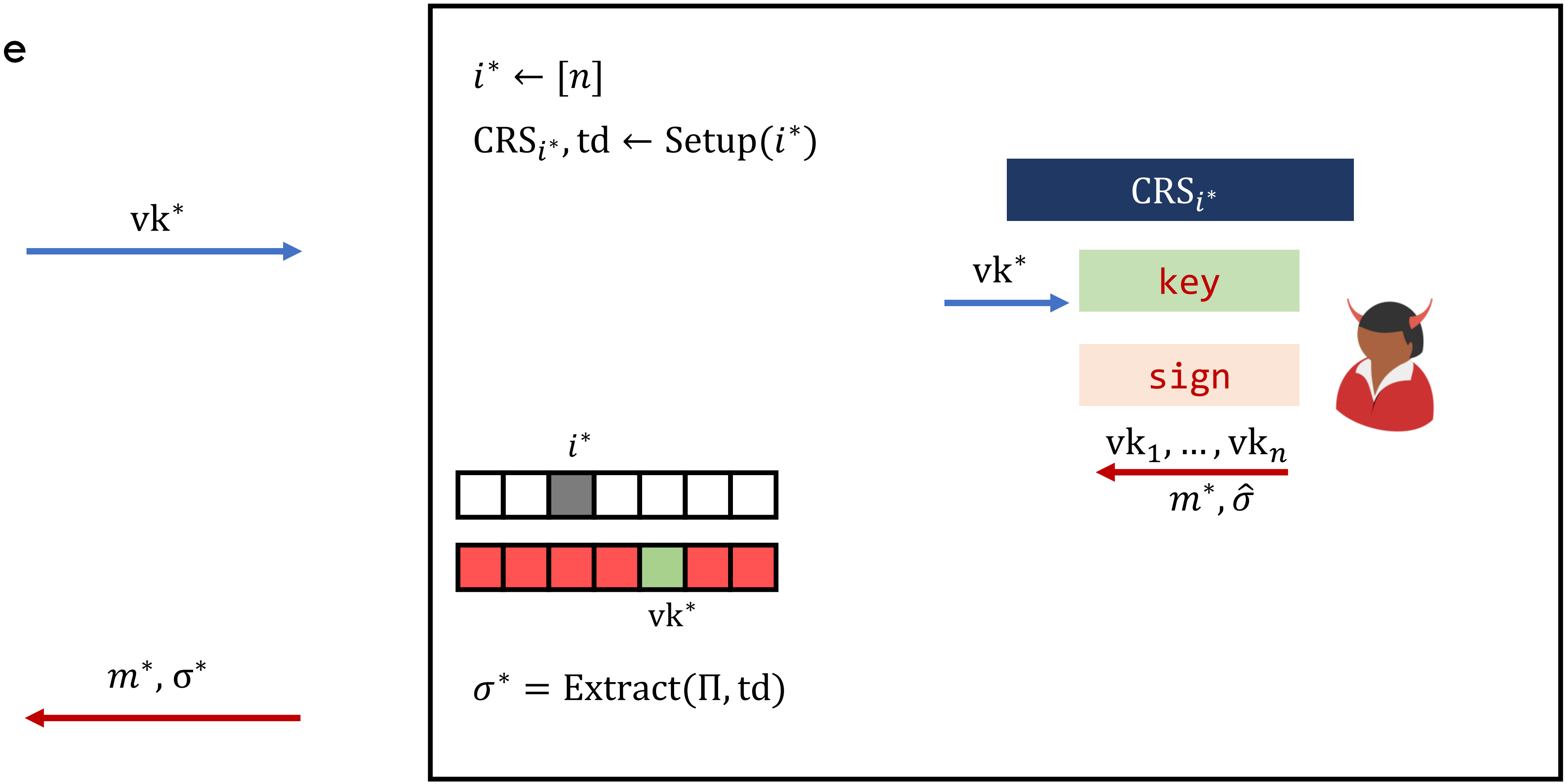
vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$

Security Proof Aggregate Signatures

Signature Scheme

Challenger



Security Proof Aggregate Signatures

Signature Scheme

Challenger

vk^*

m^*, σ^*

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

i^*

vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$

CRS_{i^*}

vk^*

`key`

`sign`

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

Security Proof Aggregate Signatures

Signature Scheme

Challenger

vk^*

Can  keep avoiding i^*?

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

CRS_{i^*}

vk^*

key

sign

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

i^*

vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$

Security Proof Aggregate Signatures

Signature Scheme

Challenger

vk^*

$i^* \leftarrow [n]$

$\mathrm{CRS}_{i^*}, \mathrm{td} \leftarrow \mathrm{Setup}(i^*)$

CRS_{i^*}

vk^*

key

sign

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

i^*

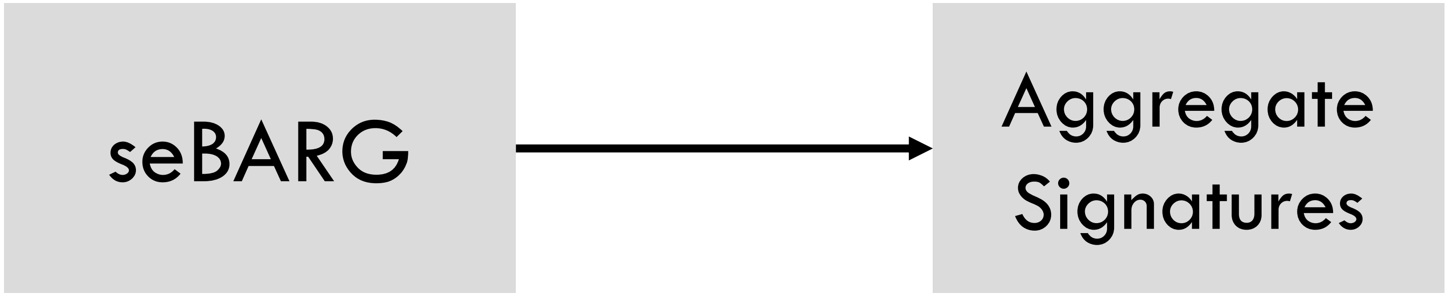
vk^*

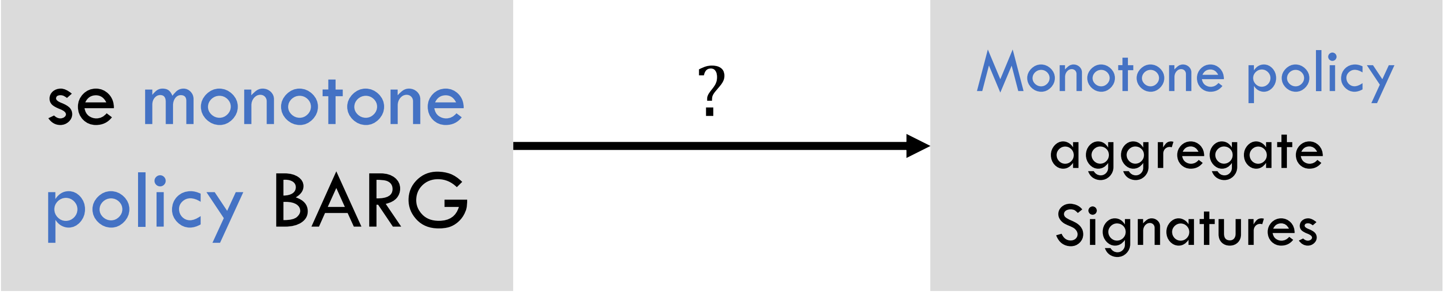
$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$

Can  keep avoiding i^*?

No! Avoidance can be checked from i^* and

Would break CRS indistinguishability.


seBARG
Aggregate Signatures


se monotone policy BARG
?
Monotone policy aggregate Signatures

Monotone Policy BARGs (pBARGs)

Monotone Policy
$f: \{0,1\}^n \to \{0,1\}$

CRS

$x_1, \cdots, x_n$
$\{w_i\}_{i \in I}$

$|w| \cdot \mathrm{poly}(\lambda)$

Π

$x_1, \cdots, x_n$

Π is publicly verifiable

$$L = \{x \mid \exists w \ s.t. \ R_L(x, w) = 1\}$$

$f(b_1, \ldots, b_n) = 1$ where $b_i = 1$ iff $i \in I$

somewhere extractable pBARG

[Brakerski-Brodsky-Kalai-Lombardi-Paneth'23]

Monotone Policy
$f: \{0,1\}^n \rightarrow \{0,1\}$

Example:

For t-out-of-n threshold, any set J of size $n - t + 1$ is necessary.

$J \subset [n]$ is necessary for f if $f(b_1, \ldots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

somewhere extractable pBARG

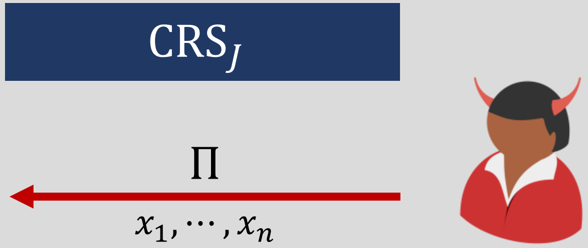
[Brakerski-Brodsky-Kalai-Lombardi-Paneth'23]

Monotone Policy

$f: \{0,1\}^n \rightarrow \{0,1\}$

For any necessary set J

$\mathrm{CRS}_J, \mathrm{td} \leftarrow \mathrm{Setup}(J)$



$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$ for some $j \in J$

$J \subset [n]$ is necessary for f if $f(b_1, \dots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

somewhere extractable pBARG

[Brakerski-Brodsky-Kalai-Lombardi-Paneth’23]

Monotone Policy
$f: \{0,1\}^n \rightarrow \{0,1\}$

For any necessary set J

$\mathrm{CRS}_J, \mathrm{td} \leftarrow \mathrm{Setup}(J)$

CRS

$\approx_c$

CRS_J

Π

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$ for some $j \in J$

$J \subset [n]$ is necessary for f if $f(b_1, \dots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

somewhere extractable pBARG

[Brakerski-Brodsky-Kalai-Lombardi-Paneth'23]

Monotone Policy
$f: \{0,1\}^n \to \{0,1\}$

For any necessary set J

$\mathrm{CRS}_J, \mathrm{td} \leftarrow \mathrm{Setup}(J)$

CRS_J

Π

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$ for some $j \in J$

$J \subset [n]$ is necessary for f if $f(b_1, \ldots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

Aggregate Signatures from pBARGs

KeyAgg $(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

$\widehat{\mathrm{vk}} = \mathrm{MerkleRoot}(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

SignAgg $(m, \{\sigma_i\}_{i \in I})$

$x_i = i, m, \widehat{\mathrm{vk}}$

$w_i = \mathrm{vk}_i, \sigma_i, \pi_i$ or $\perp$ if $i \notin I$

$\hat{\sigma} = \mathrm{pBARG}(x_1 \ldots, x_n, w_1, \ldots, w_n)$

Statement: $i, m, \widehat{\mathrm{vk}}$

Witness: $\mathrm{vk}_i, \sigma_i, \pi_i$

s.t. π_i is a valid opening to vk_i AND $\mathrm{Verify}(\mathrm{vk}_i, m, \sigma_i) = 1$

Security Monotone Policy Aggregate Signatures

Challenger

CRS

key

sign

vk_i generated by 

sign vk_i, m^* queried

$$\xleftarrow[m^*,\hat{\sigma}]{\mathrm{vk}_1,\ldots,\mathrm{vk}_n}$$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1,\ldots,\mathrm{vk}_n)$

$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

 wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. $f(\boxed{0}\boxed{1}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}) \neq 1$

Security Monotone Policy Aggregate Signatures

Challenger

CRS

key

sign

vk_i generated by 

sign vk_i, m^* queried

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \ldots, \mathrm{vk}_n)$

$J = \begin{array}{ccccc} \mathrm{vk}_1 & \cdots & \mathrm{vk}_i & \cdots & \mathrm{vk}_n \end{array}$

 wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. $f(\,0\;1\;1\;1\;0\;0\;0\,) \neq 1$

Security Monotone Policy Aggregate Signatures

Challenger

CRS

key

sign

vk_i generated by

sign vk_i, m^* queried

$\mathrm{vk}_1, \dots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

$\widehat{\mathrm{vk}} \coloneqq \mathrm{KeyAgg}\,(\mathrm{vk}_1, \dots, \mathrm{vk}_n)$

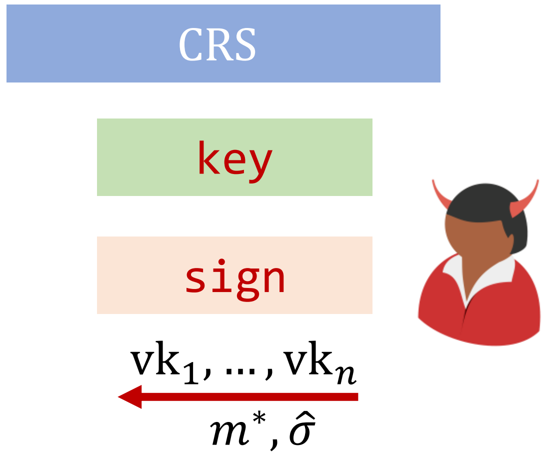
$\mathrm{vk}_1 \quad \cdots \quad \mathrm{vk}_i \quad \cdots \quad \mathrm{vk}_n$

$J =$

wins if

1. $\mathrm{Verify}(\widehat{\mathrm{vk}}, m^*, \hat{\sigma}) = 1$
2. J is necessary.

Security Proof Monotone Policy Aggregate Signatures



Security Proof Monotone Policy Aggregate Signatures

Signature Scheme

Challenger

vk^* →

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

vk^* → `key`

`sign`

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$ ←

$m^*, \hat{\sigma}$

J^*

J vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

m^*, σ^* ←

Security Proof Monotone Policy Aggregate Signatures

Signature Scheme

Challenger

vk^* →

← m^*, σ^*

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

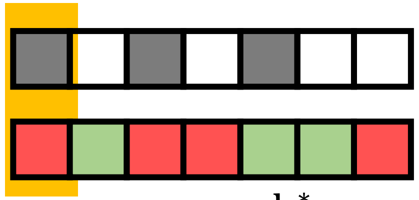
vk^* →

key

sign

← $\mathrm{vk}_1, \dots, \mathrm{vk}_n$, $m^*, \hat{\sigma}$

J^*



J vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

Security Proof Monotone Policy Aggregate Signatures

Signature Scheme
Challenger

vk^*

Fails unless guessed J^* is the same as adaptively chosen J.

m^*, σ^*

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

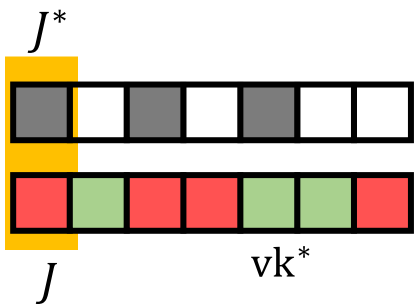
vk^*

key

sign

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$



$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

Security Proof Monotone Policy Aggregate Signatures

Signature Scheme
Challenger

vk^*

Fails unless guessed J^* is the same as adaptively chosen J.

May be exponentially many J^*.

m^*, σ^*

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

vk^*

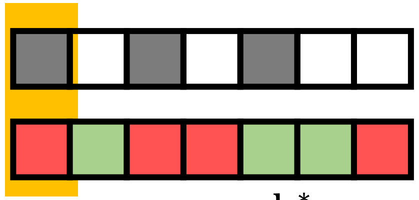
key

sign

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

J^*



J

vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

somewhere extractable pBARG

[Brakerski-Brodsky-Kalai-Lombardi-Paneth'23]

Monotone Policy
$f: \{0,1\}^n \rightarrow \{0,1\}$

For any necessary set J

$\mathrm{CRS}_J, \mathrm{td} \leftarrow \mathrm{Setup}(J)$

CRS_J

Π

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

If Π verifies, $R_L(x_j, w) = 1$ for some $j \in J$

$J \subset [n]$ is necessary for f if $f(b_1, \dots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

pBARG with Adaptive Subset Extraction

Monotone Policy
$f: \{0,1\}^n \rightarrow \{0,1\}$

$\text{CRS}, \text{td} \leftarrow \text{Setup}()$

CRS

Π, J

$x_1, \cdots, x_n$

$w \leftarrow \text{Extract}(\Pi, \text{td})$

If Π verifies, $R_L(x_j, w) = 1$ for some $j \in J$ with probability $\approx 1/n$

$J \subset [n]$ is necessary for f if $f(b_1, \ldots, b_n) = 0$ where $b_i = 0$ iff $i \in J$

Security Proof Monotone Policy Aggregate Signatures

Signature Scheme

Challenger

vk^*

m^*, σ^*

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

vk^*

key

sign

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

J^*

J

vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

Security Proof Monotone Policy Aggregate Signatures

Signature Scheme

Challenger

vk^*

$\mathrm{CRS}, \mathrm{td} \leftarrow \mathrm{Setup}()$

CRS

vk^*

key

sign

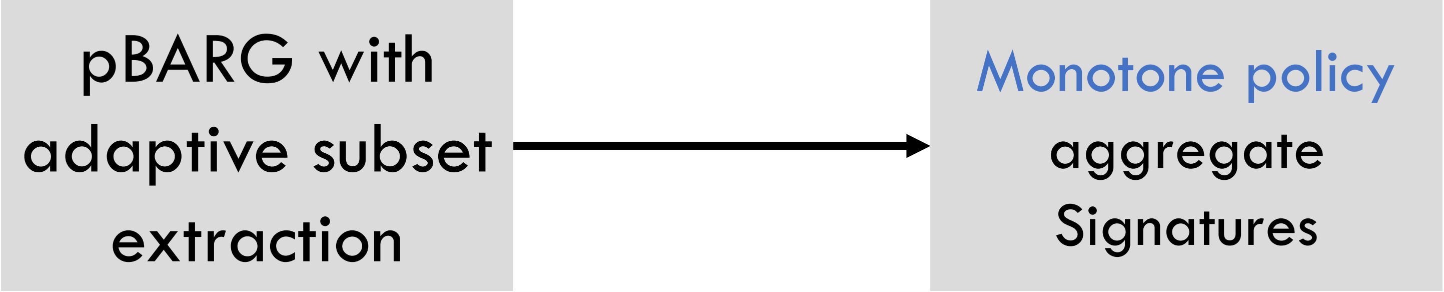
$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

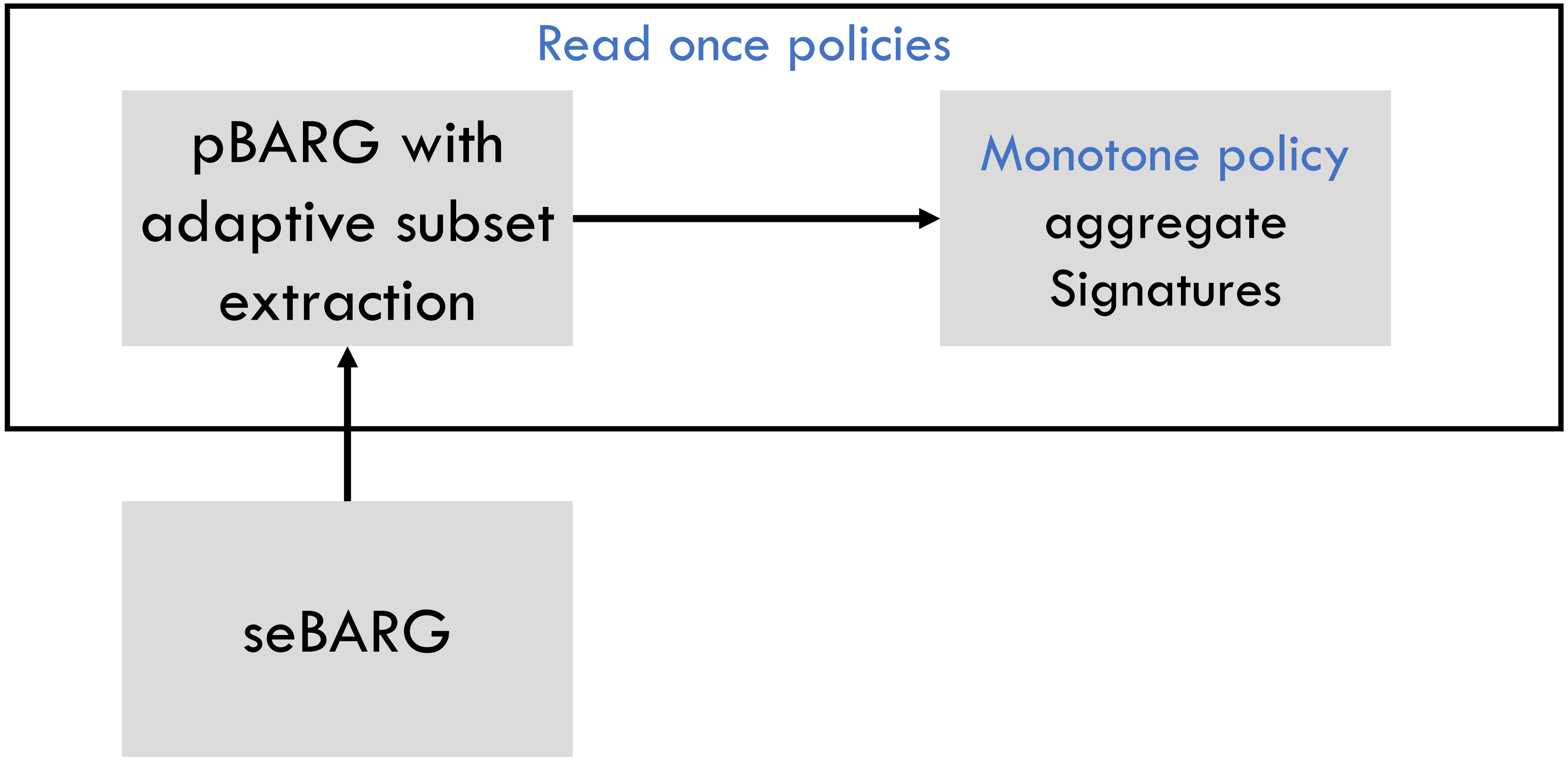
$m^*, \hat{\sigma}$

J vk^*

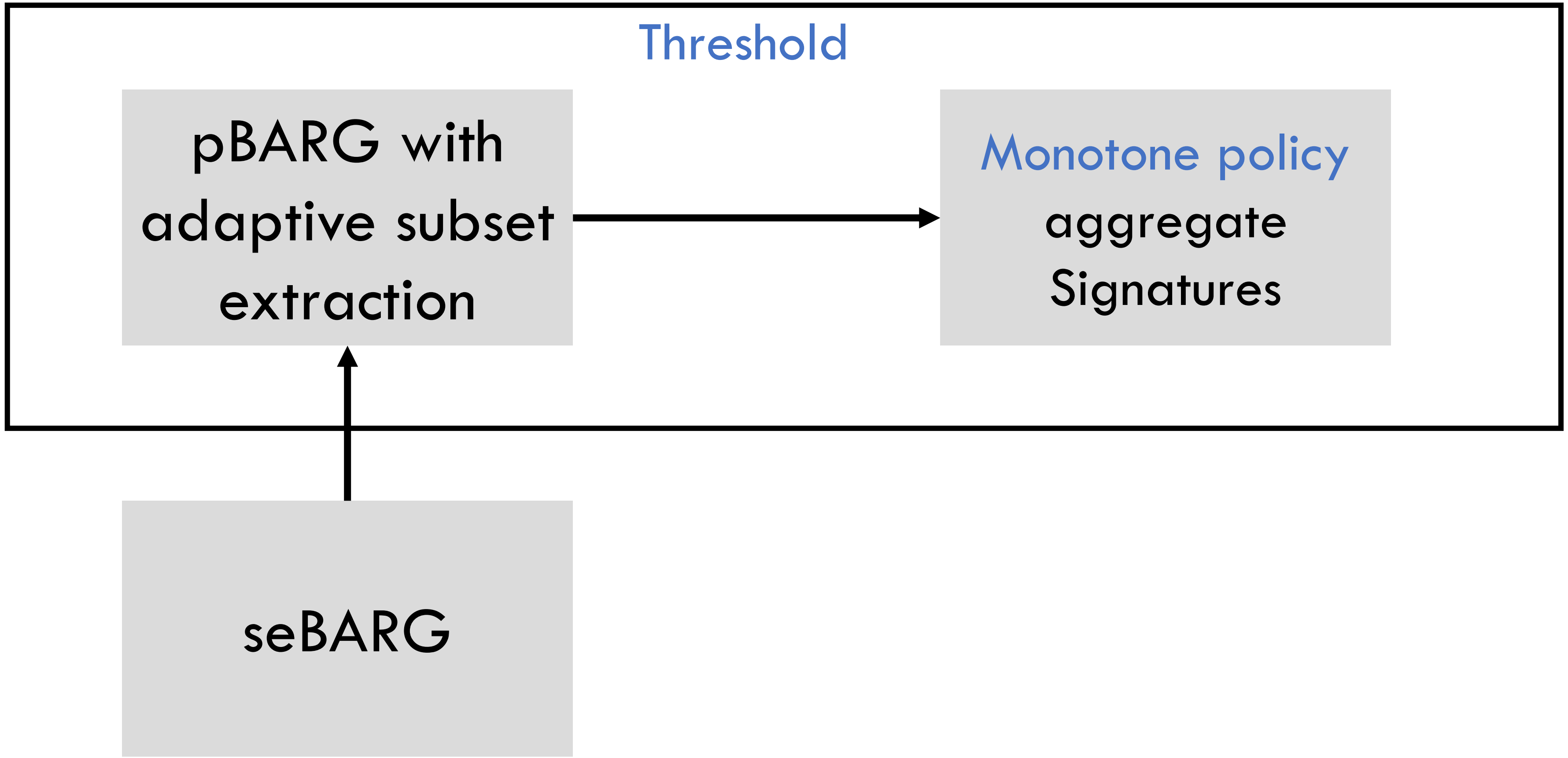
$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J$ with probability $\approx 1/n$

m^*, σ^*


pBARG with adaptive subset extraction
Monotone policy aggregate Signatures


Read once policies
pBARG with adaptive subset extraction
Monotone policy aggregate Signatures
seBARG

Today


Threshold
pBARG with adaptive subset extraction
Monotone policy aggregate Signatures
seBARG

Previous Attempt

vk^*

Fails unless guessed J^* is the same as adaptively chosen J.

May be exponentially many J^*.

m^*, σ^*

Random necessary set J^*

$\mathrm{CRS}_{J^*}, \mathrm{td} \leftarrow \mathrm{Setup}(J^*)$

CRS_{J^*}

vk^*

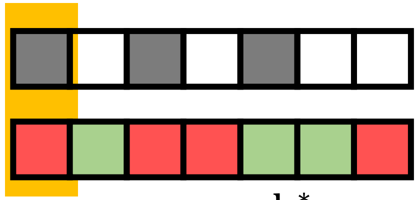
key

sign

$\mathrm{vk}_1, \ldots, \mathrm{vk}_n$

$m^*, \hat{\sigma}$

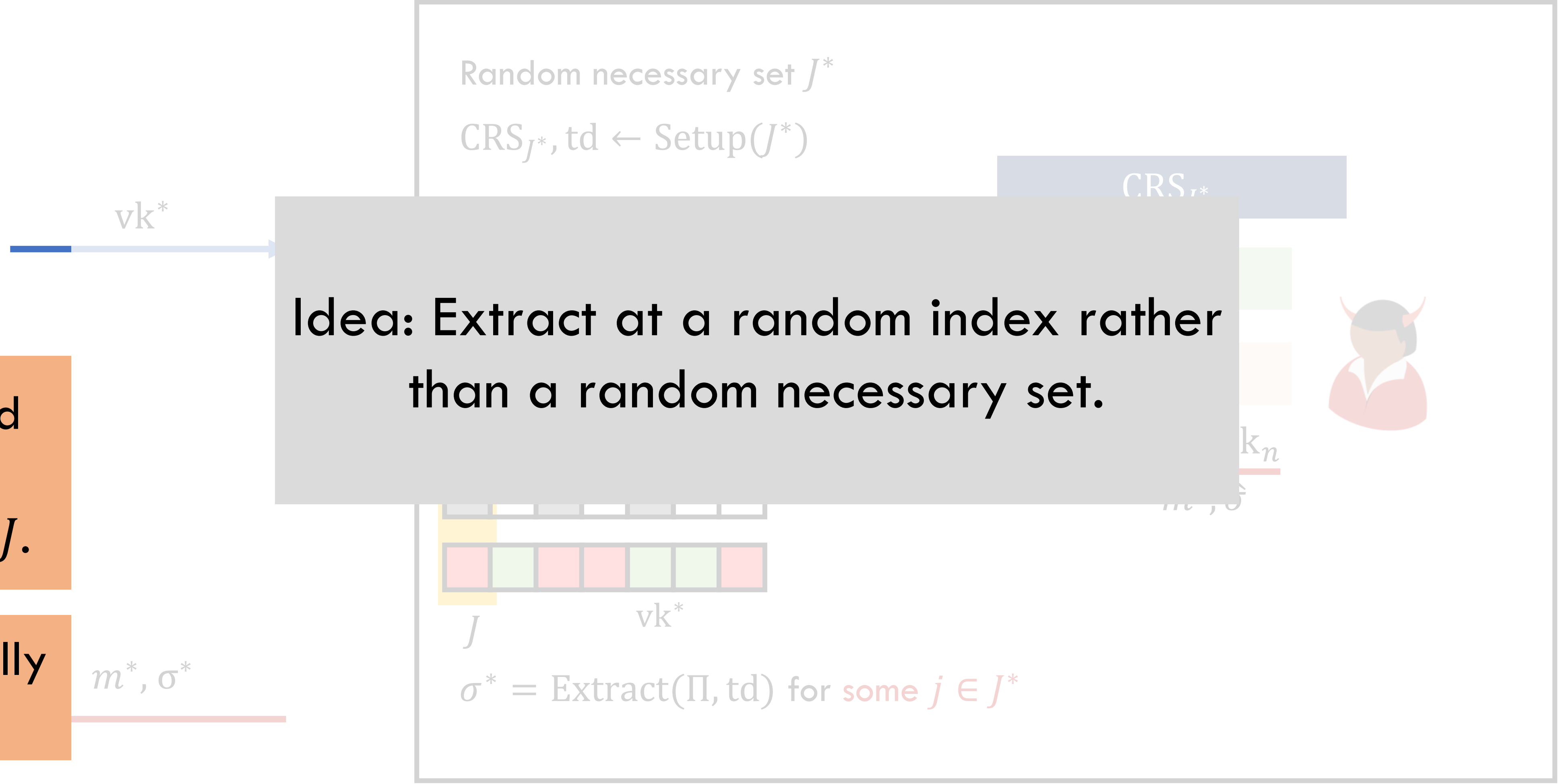
J^*



J vk^*

$\sigma^* = \mathrm{Extract}(\Pi, \mathrm{td})$ for some $j \in J^*$

Previous Attempt



Idea: Extract at a random index rather than a random necessary set.

Fails unless guessed J^* is the same as adaptively chosen J.

May be exponentially many J^*.

Threshold BARG with Adaptive Subset Extraction

x_1 w_1

$\vdots$

x_i w_i

$\vdots$

x_n w_n

At least t witnesses w_i such that $w_i \neq \perp$.

Threshold BARG with Adaptive Subset Extraction

x_1	w_1
$\vdots$	$\vdots$
x_i	w_i
$\vdots$	
	$\vdots$
x_n	w_n

Threshold BARG with Adaptive Subset Extraction

x_1	w_1	c_1
$\vdots$		
x_i	w_i	c_i
$\vdots$		
x_n	w_n	c_n

$$\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$$

Threshold BARG with Adaptive Subset Extraction

x_1	c_0	w_1	c_1
$\vdots$	c_1		
x_i	c_{i-1}	w_i	c_i
$\vdots$			
x_n	c_{n-1}	w_n	c_n

$$\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$$

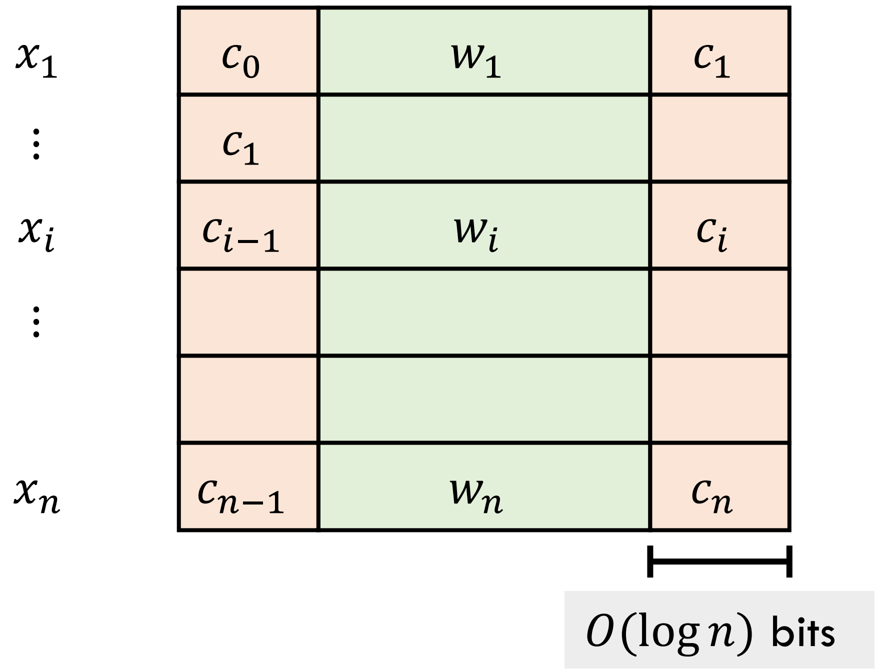
Threshold BARG with Adaptive Subset Extraction

x_1	c_0	w_1	c_1
$\vdots$	c_1		
x_i	c_{i-1}	w_i	c_i
$\vdots$			
x_n	c_{n-1}	w_n	c_n

$O(\log n)$ bits

$$\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$$

Threshold BARG with Adaptive Subset Extraction



$$\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$$

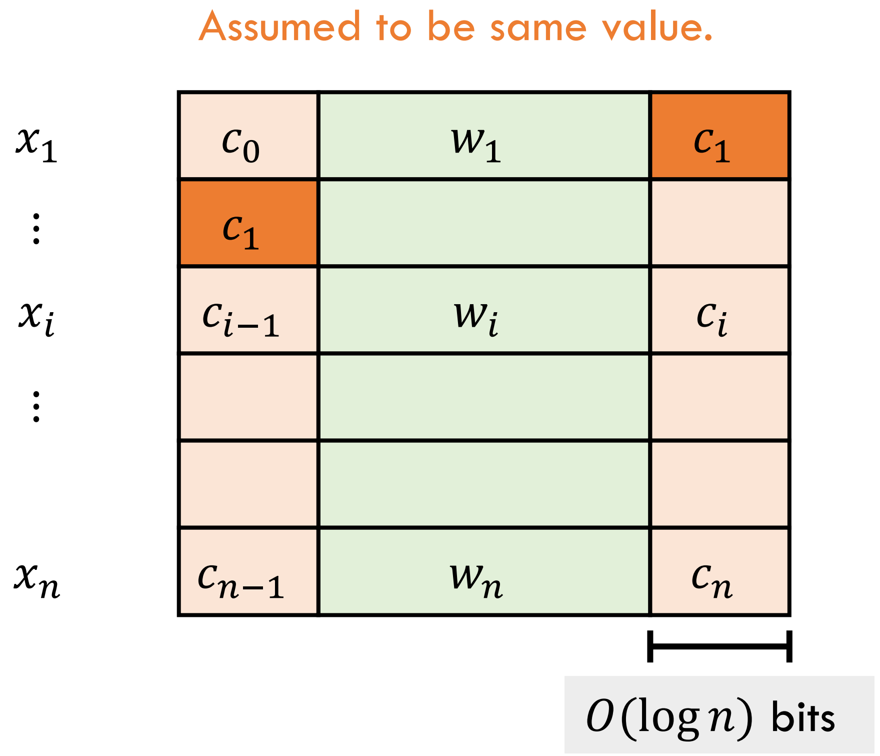
Batch $\forall\, i \in [n]$

Statement: x_i

Witness: c_{i-1}, w_i, c_i

s.t. $\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$

Threshold BARG with Adaptive Subset Extraction



$$\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$$

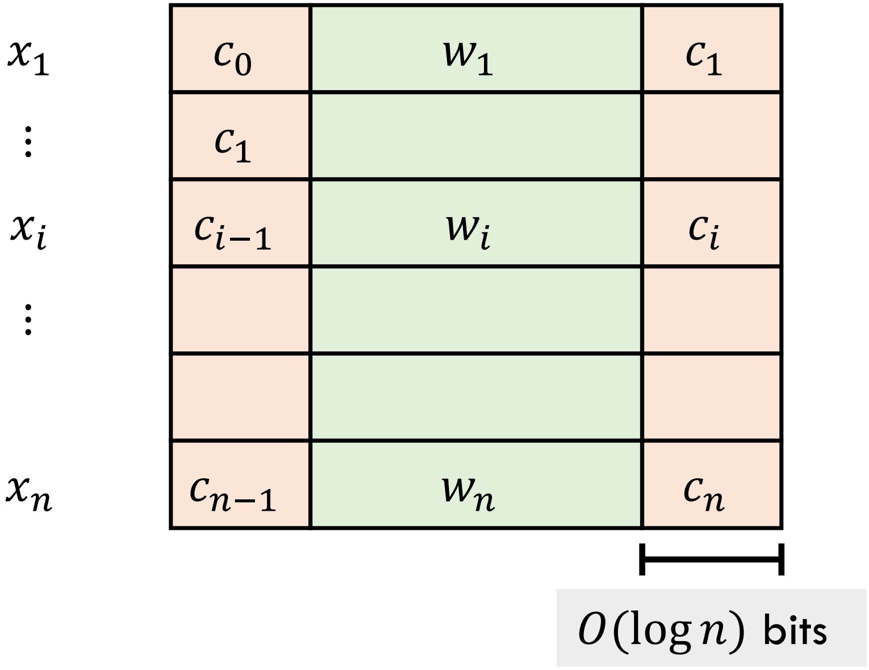
Batch $\forall\, i \in [n]$

Statement: x_i

Witness: c_{i-1}, w_i, c_i

s.t. $\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$

Threshold BARG with Adaptive Subset Extraction



Batch $\forall\, i \in [n]$

Statement: x_i, rt

Witness: $c_{i-1}, \pi_{i-1}, w_i, c_i, \pi_i$

s.t. $c_i = c_{i-1} + R_L(x_i, w_i)$

π_{i-1} valid opening to c_{i-1}

π_i valid opening to c_i

$$\mathrm{rt} = H(\boxed{c_0}\ \boxed{c_1}\ \cdots\ \boxed{c_n})$$

Threshold BARG with Adaptive Subset Extraction

x_1	c_0	w_1	c_1
$\vdots$	c_1		
x_i	c_{i-1}	w_i	c_i
$\vdots$			
x_n	c_{n-1}	w_n	c_n

$O(\log n)$ bits

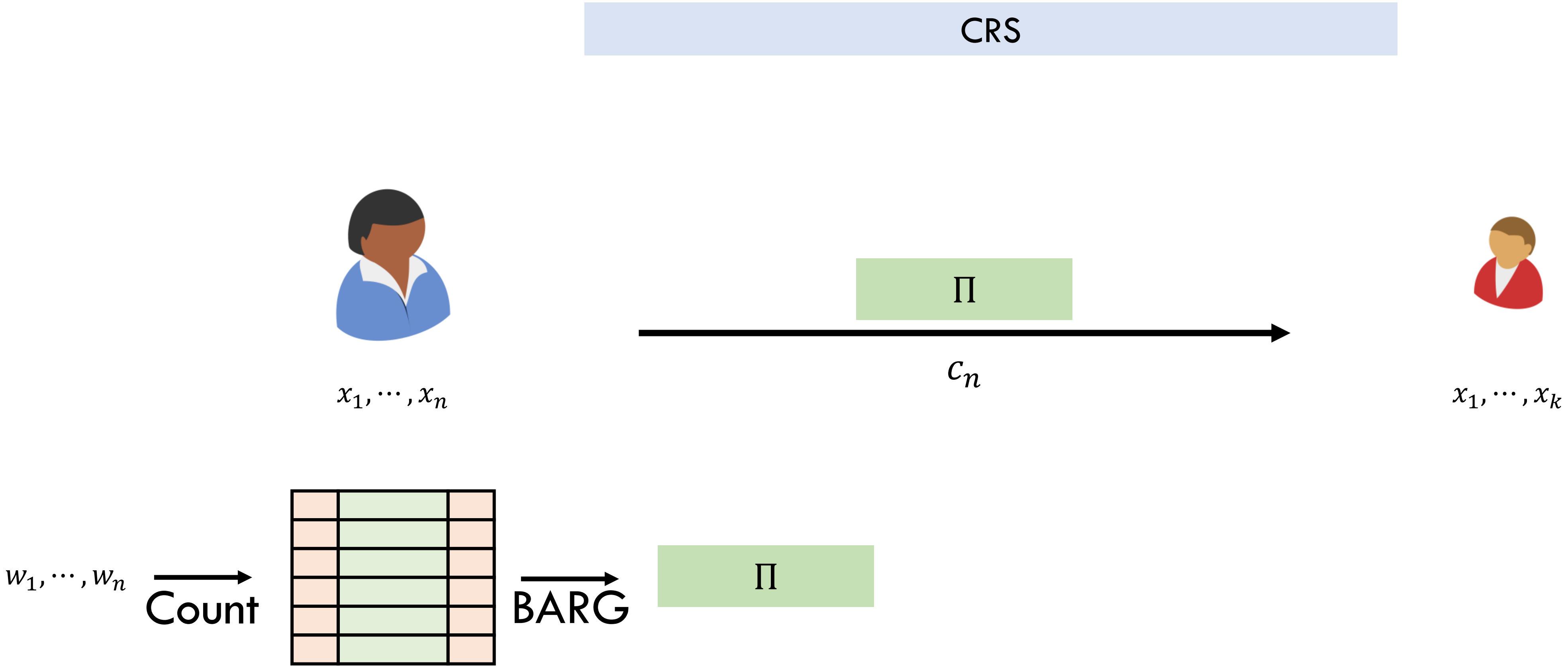
Batch $\forall\, i \in [n]$

Statement: x_i

Witness: c_{i-1}, w_i, c_i

s.t. $\mathrm{c}_i = \mathrm{c}_{i-1} + R_L(x_i, w_i)$

Threshold BARG with Adaptive Subset Extraction



Adaptive Subset Extraction

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

Adaptive Subset Extraction

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

$$\Pr_{j \leftarrow [n]}\left[j \in J \wedge R_L(x_j, w_j) = 1\right] \overset{?}{\approx} 1/n$$

Adaptive Subset Extraction

Can  keep avoiding j by setting $w_j = \perp$?

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

$$\Pr_{j \leftarrow [n]}\left[j \in J \wedge R_L(x_j, w_j) = 1\right] \overset{?}{\approx} 1/n$$

Adaptive Subset Extraction

Can  keep avoiding j by setting $w_j = \perp$?

Checking requires extracting using td $\implies$ cannot rely on CRS hiding.

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

$$\Pr_{j \leftarrow [n]}\left[j \in J \wedge \boxed{R_L\left(x_j, w_j\right) = 1}\right] \overset{?}{\approx} 1/n$$

Adaptive Subset Extraction

Can  keep avoiding j by setting $w_j = \perp$?

Sufficient to determine which statements were true to perform above check.

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

$$\Pr_{j \leftarrow [n]}\left[j \in J \wedge R_L(x_j, w_j) = 1\right] \overset{?}{\approx} 1/n$$

Adaptive Subset Extraction

Can  keep avoiding j by setting $w_j = \perp$?

Sufficient to determine which statements were true to perform above check.

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

$$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

Goal: Determine which t statements proved were true.

Adaptive Subset Extraction

Can  keep avoiding j by setting $w_j = \perp$?

Sufficient to determine which statements were true to perform above check.

$j \leftarrow [n]$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$

$w \leftarrow \mathrm{Extract}(\Pi, \mathrm{td})$

Approximately Simulate

Goal: ~~Determine~~ which t statements proved were true.

There exists a simulator Sim such that for all $j \in [n]$

Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$\xleftarrow[x_1, \cdots, x_n]{}$

$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$\approx$

Ideal (j)

$x_1, \cdots, x_n, w_1, \cdots, w_n, J \leftarrow \mathrm{Sim}()$

If $|J| \leq k - t$ or $|\{w_i \neq \perp\}| < t$

abort

Output (x_j, w_j)

There exists a simulator Sim such that for all $j \in [n]$

Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

$\boxed{\mathrm{CRS}_j}$

$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$

$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$\approx$

Ideal (j)

$x_1, \cdots, x_n, w_1, \cdots, w_n, J \leftarrow \mathrm{Sim}()$

If $|J| \leq k - t$ or $|\{w_i \neq \perp\}| < t$

abort

Output (x_j, w_j)

$$\Pr_{j \leftarrow [n]}\left[R_L(x_j, w_j) = 1\right] \approx t/n$$

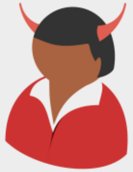
There exists a simulator Sim such that for all $j \in [n]$

Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$



$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$\approx$

Ideal (j)

$x_1, \cdots, x_n, w_1, \cdots, w_n, J \leftarrow \mathrm{Sim}()$

If $|J| \leq k - t$ or $|\{w_i \neq \perp\}| < t$

abort

Output (x_j, w_j)

$$\Pr_{j \leftarrow [n]}[R_L(x_j, w_j) = 1] \approx t/n \quad \Longleftarrow \quad \Pr_{j \leftarrow [n]}[R_L(x_j, w_j) = 1] \approx t/n$$

There exists a simulator Sim such that for all $j \in [n]$

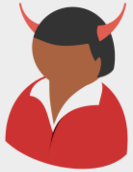
Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$



$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$$\Pr_{j \leftarrow [n]}\left[R_L(x_j, w_j) = 1\right] \approx t/n$$

There exists a simulator Sim such that for all $j \in [n]$

$\mathsf{Real}(j)$

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

$\boxed{\mathrm{CRS}_j}$

Π, J

$\xleftarrow[x_1, \cdots, x_n]{}$

$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$$\Pr_{j \leftarrow [n]}\left[R_L\left(x_j, w_j\right) = 1\right] \approx t/n$$

There exists a simulator Sim such that for all $j \in [n]$

Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$

$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$$\Pr_{j \leftarrow [n]}\left[R_L\left(x_j, w_j\right) = 1\right] \approx t/n$$

$$\Pr_{j \leftarrow [n]}[j \in J] \approx (n - t + 1)/n$$

By CRS indistinguishability and that the size of any necessary set is $(n - t + 1)$

There exists a simulator Sim such that for all $j \in [n]$

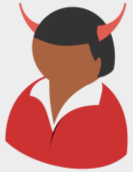
Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$



$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$$\Pr_{j \leftarrow [n]}\left[R_L\left(x_j, w_j\right) = 1\right] \approx t/n$$

$$\Pr_{j \leftarrow [n]}[j \in J] \approx (n - t + 1)/n$$

By CRS indistinguishability and that the size of any necessary set is $(n - t + 1)$

$$\Pr_{j \leftarrow [n]}\left[j \in J \wedge R_L\left(x_j, w_j\right) = 1\right] \approx 1/n$$

There exists a simulator Sim such that for all $j \in [n]$

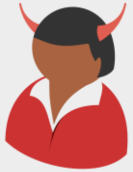
Real(j)

$\mathrm{CRS}_j, \mathrm{td} \leftarrow \mathrm{Setup}(j)$

CRS_j

Π, J

$x_1, \cdots, x_n$



$c_{j-1}, w, c_j \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_j, w)

$\approx$

Ideal (j)

$x_1, \cdots, x_n, w_1, \cdots, w_n, J \leftarrow \mathrm{Sim}()$

If $|J| \leq k - t$ or $|\{w_i \neq \perp\}| < t$

abort

Output (x_j, w_j)

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

CRS_1

$$\xleftarrow[x_1, \cdots, x_n]{\Pi, J}$$

$c_0, w, c_1 \leftarrow \mathrm{Extract}(\mathrm{td})$

Output (x_1, c_0, w, c_1)

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

x_1 

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ $\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

x_1  x'_1 

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

x_1 

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

${x'}_1$ 

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

${x''}_1$ 

…

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ $\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ $\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ …

x_1  x'_1  x''_1 

Simulator Sketch

$\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ $\quad$ $\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$ $\quad$ $\mathrm{CRS}_1, \mathrm{td} \leftarrow \mathrm{Setup}(1)$

x_1 $\quad$ x'_1 $\quad$ x''_1 $\quad$ …

$\mathrm{CRS}_2, \mathrm{td} \leftarrow \mathrm{Setup}(2)$ $\quad$ $\mathrm{CRS}_2, \mathrm{td} \leftarrow \mathrm{Setup}(2)$ $\quad$ $\mathrm{CRS}_2, \mathrm{td} \leftarrow \mathrm{Setup}(2)$

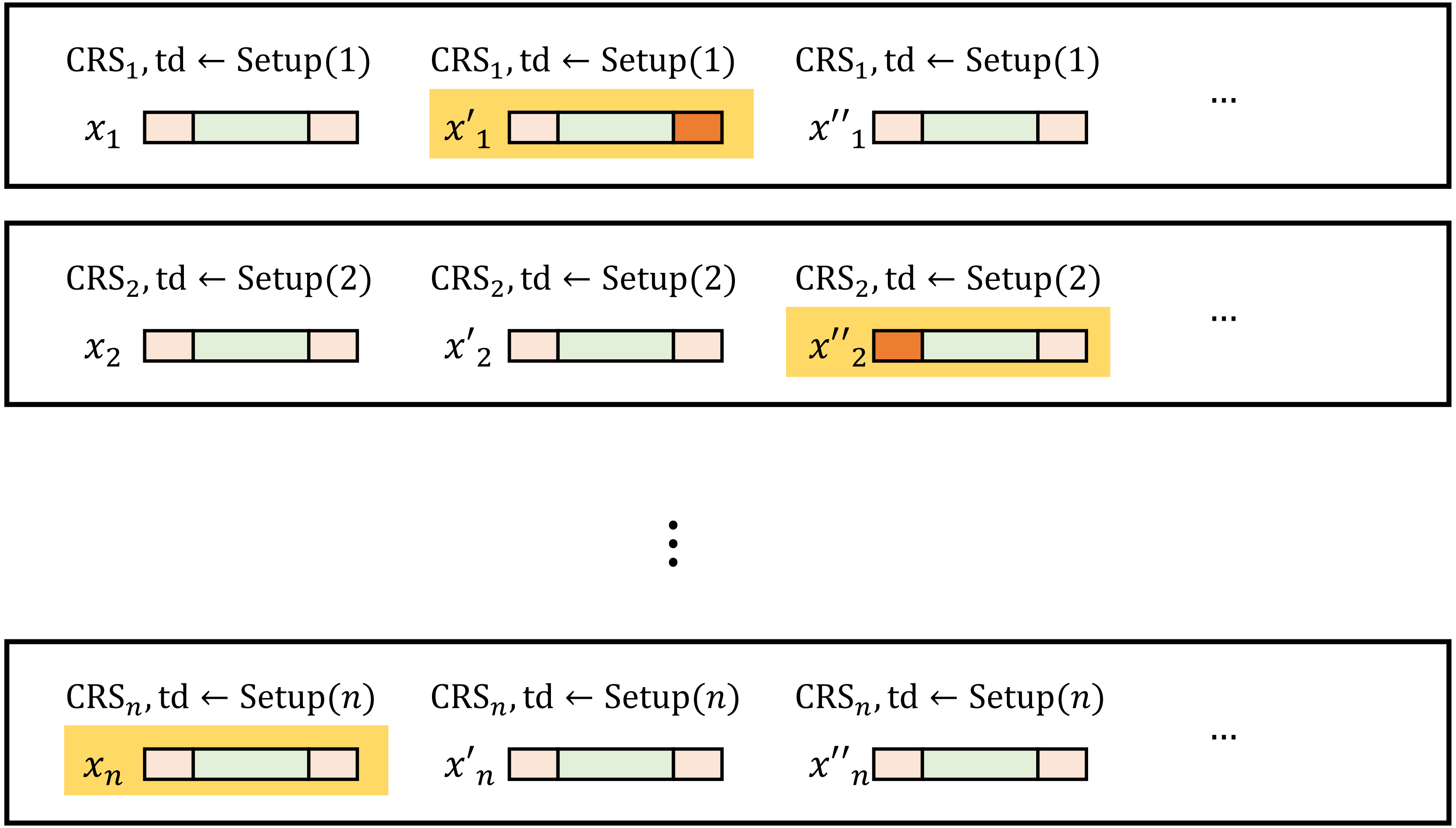
x_2 $\quad$ x'_2 $\quad$ x''_2 $\quad$ …

$\vdots$

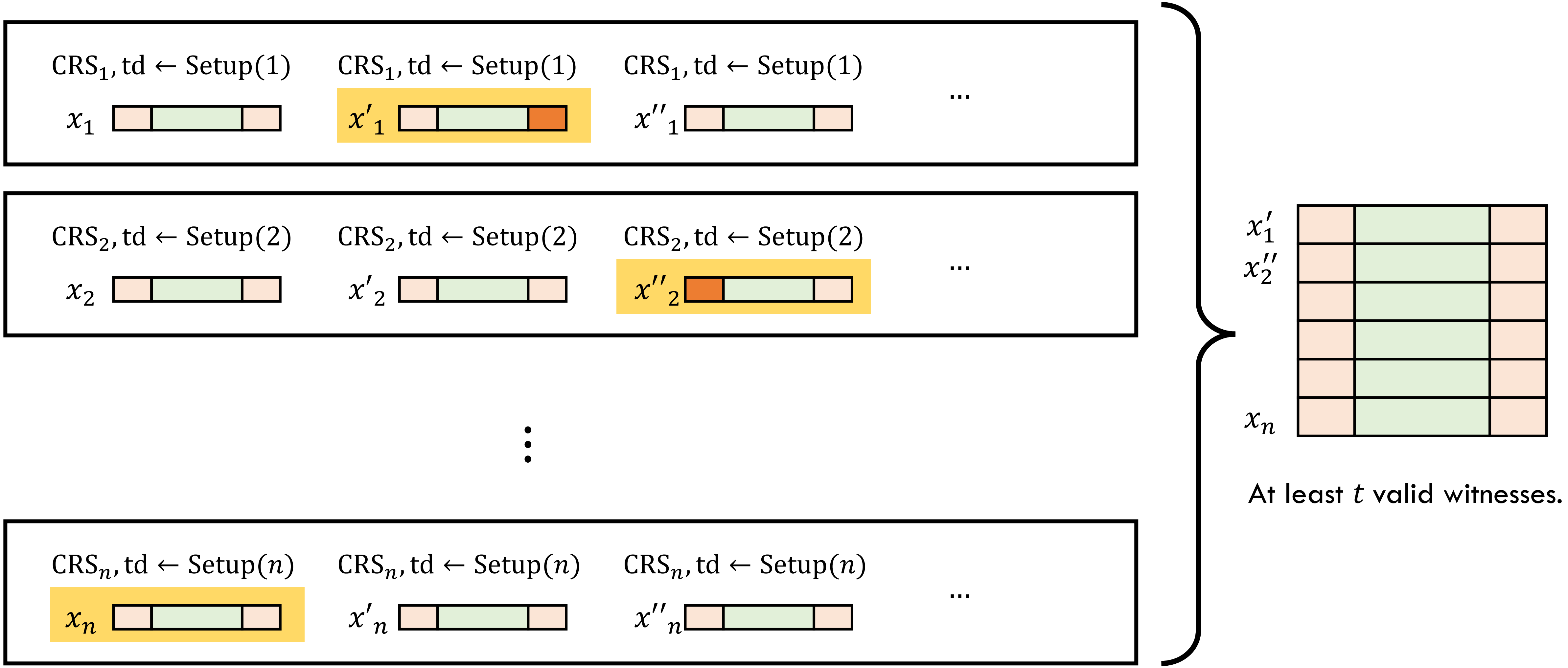
$\mathrm{CRS}_n, \mathrm{td} \leftarrow \mathrm{Setup}(n)$ $\quad$ $\mathrm{CRS}_n, \mathrm{td} \leftarrow \mathrm{Setup}(n)$ $\quad$ $\mathrm{CRS}_n, \mathrm{td} \leftarrow \mathrm{Setup}(n)$

x_n $\quad$ x'_n $\quad$ x''_n $\quad$ …

Simulator Sketch

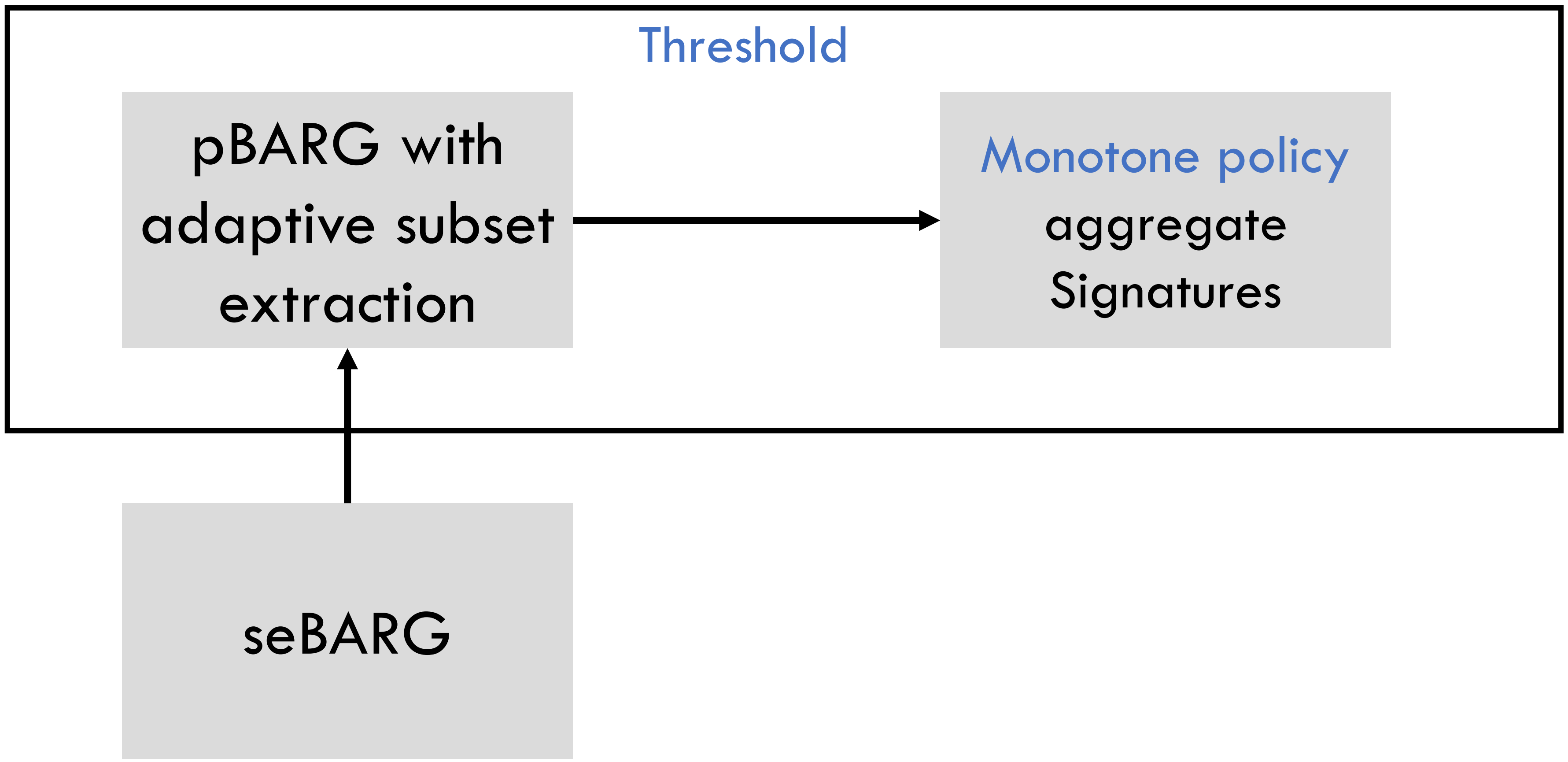


Simulator Sketch



If $|c_i| = O(\log n)$

Today


Threshold
pBARG with adaptive subset extraction
Monotone policy aggregate Signatures
seBARG

Our Results

BARGs → Monotone Policy Aggregate Signatures for read-once $O(\log n)$ space monotone policies.

BARGs + FHE → Weakly Secure monotone Policy Aggregate Signatures for poly size monotone circuits.

Thank you. Questions?

Arka Rai Choudhuri

arkarai.choudhuri@ntt-research.com

Paper on ePrint soon!