Multi-Key Homomorphic Secret Sharing

EUROCRYPT 2025

Geoffroy Couteau

CNRS, IRIF

Université Paris Cité

Lalita Devadas

MIT

Aditya Hegde

JHU

Sacha Servan-Schreiber

MIT

Abhishek Jain

NTT Research

JHU

Homomorphic Secret Sharing (HSS)

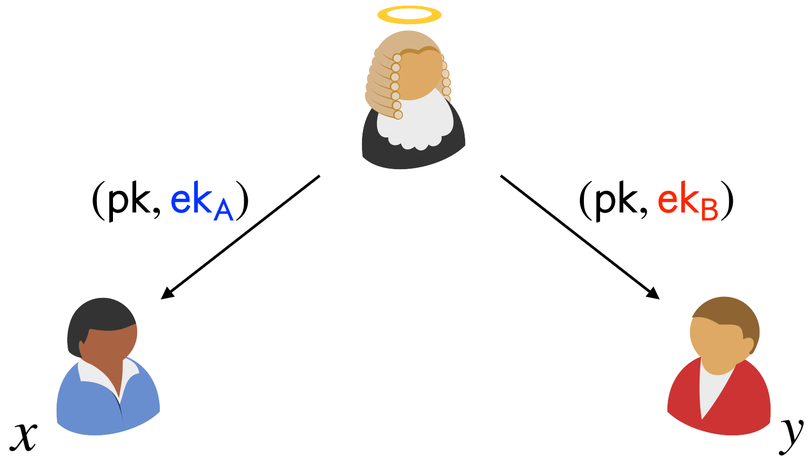
[Boyle-Gilboa-Ishai'16]

x

y

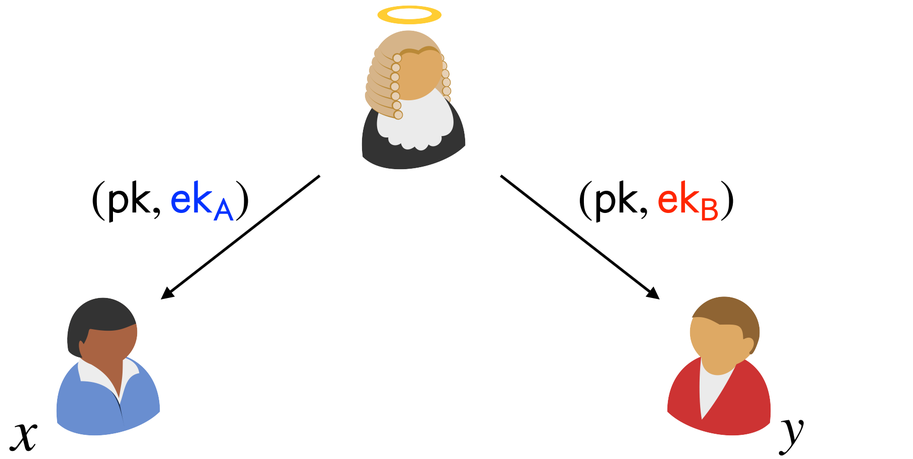
Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

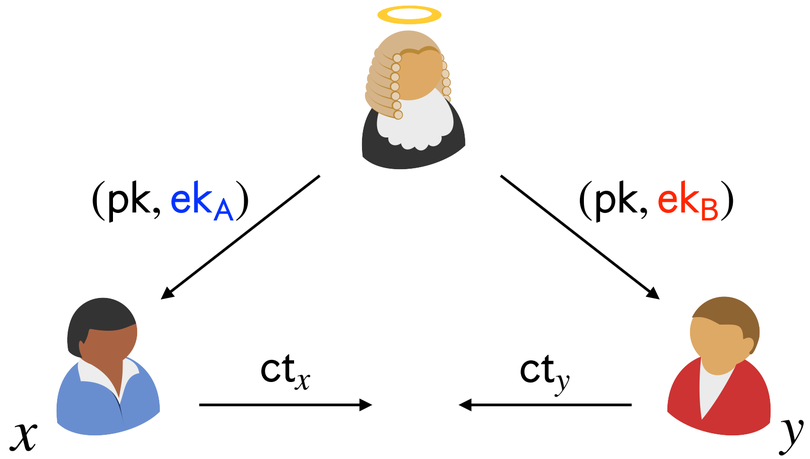


$ct_x \leftarrow \text{Encrypt}(pk, x)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

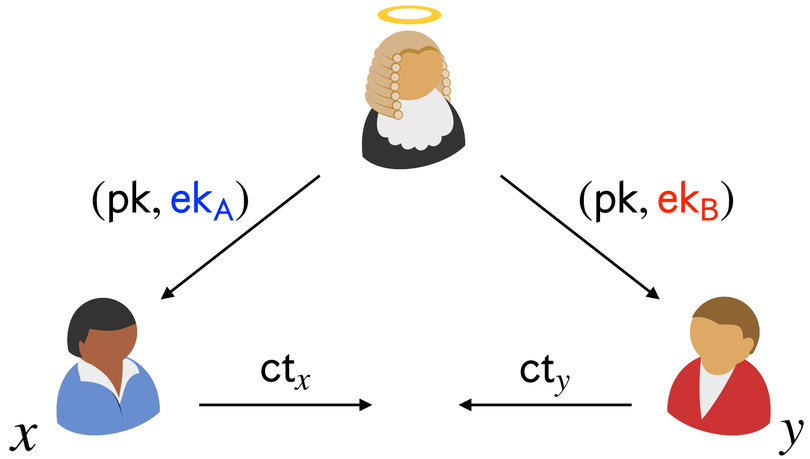


$ct_x \leftarrow \text{Encrypt}(pk, x)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \text{Encrypt}(pk, x)$

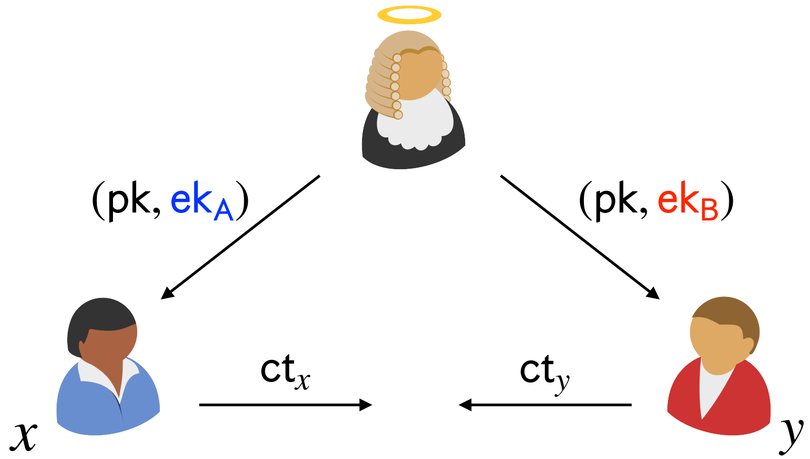
$\text{Encrypt}(pk, y) \rightarrow ct_y$

$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \text{Encrypt}(pk, x)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

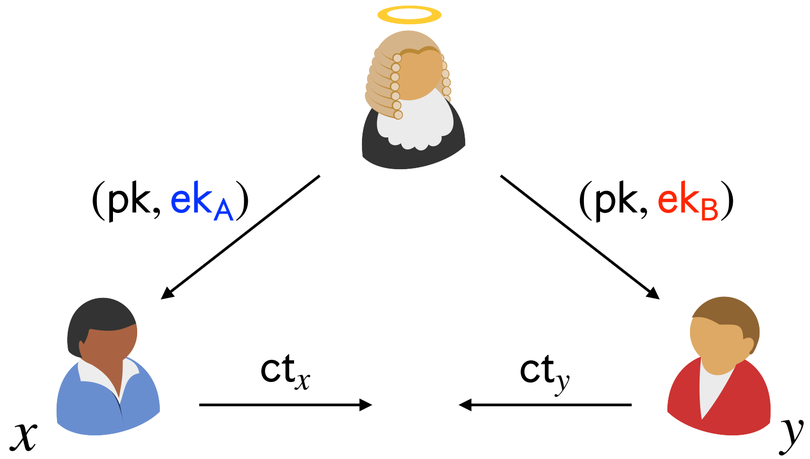
$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Correctness $\quad z_A + z_B = C(x, y)$

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \text{Encrypt}(pk, x)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

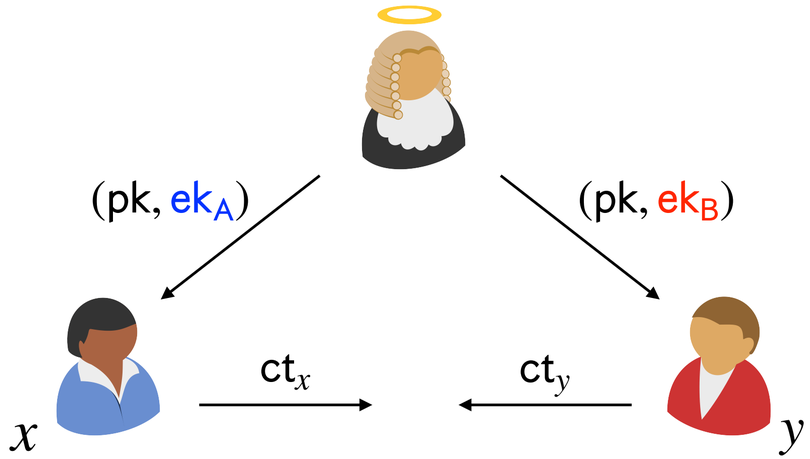
$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Correctness	$z_A + z_B = C(x, y)$
Security	ct_x ensures privacy of x
	ct_y ensures privacy of y

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \text{Encrypt}(pk, x)$

$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Analogue of Threshold FHE

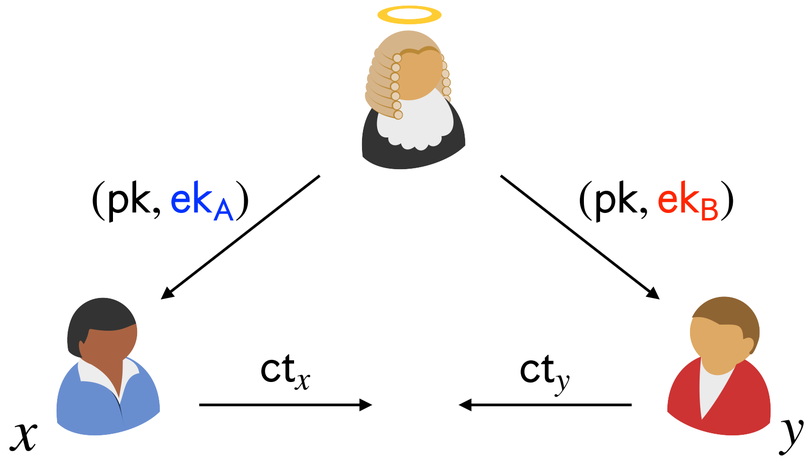
Correctness $z_A + z_B = C(x, y)$

Security ct_x ensures privacy of x

ct_y ensures privacy of y

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \mathsf{Encrypt}(pk, x)$

$\mathsf{Encrypt}(pk, y) \rightarrow ct_y$

$z_A \leftarrow \mathsf{Eval}(ek_A, C, ct_x, ct_y)$

$\mathsf{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Analogue of Threshold FHE

Only known from Lattice assumptions

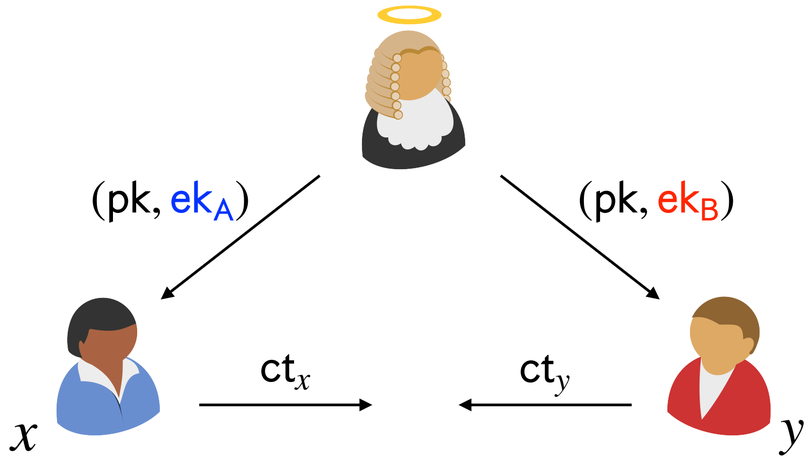
Correctness $z_A + z_B = C(x, y)$

Security ct_x ensures privacy of x

ct_y ensures privacy of y

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



$ct_x \leftarrow \text{Encrypt}(pk, x)$

$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Correctness $z_A + z_B = C(x, y)$

Security ct_x ensures privacy of x

ct_y ensures privacy of y

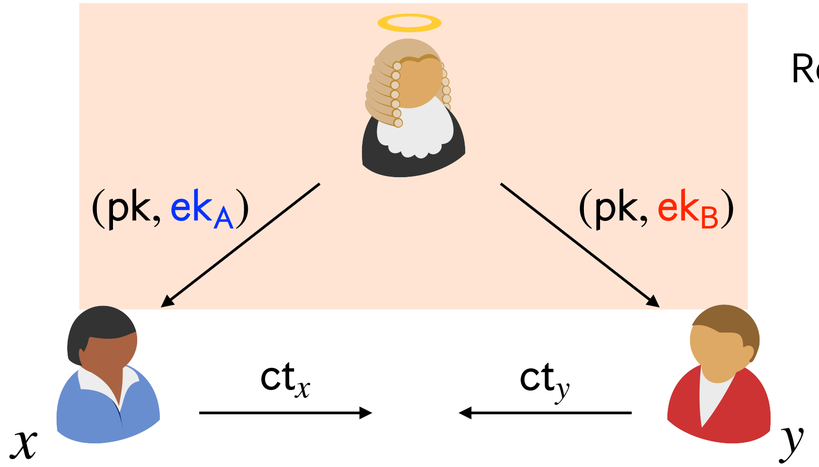
Analogue of Threshold FHE

Only known from Lattice assumptions

HSS can be constructed from group-based assumptions

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



Replace correlated setup with CRS

$ct_x \leftarrow \text{Encrypt}(pk, x)$

$z_A \leftarrow \text{Eval}(ek_A, C, ct_x, ct_y)$

$\text{Encrypt}(pk, y) \rightarrow ct_y$

$\text{Eval}(ek_B, C, ct_x, ct_y) \rightarrow z_B$

Analogue of Threshold FHE

Only known from Lattice assumptions

Correctness $z_A + z_B = C(x, y)$

Security ct_x ensures privacy of x

ct_y ensures privacy of y

HSS can be constructed from group-based assumptions

Multi-Key Homomorphic Secret Sharing

Common Reference String

x

y

Multi-Key Homomorphic Secret Sharing

Common Reference String

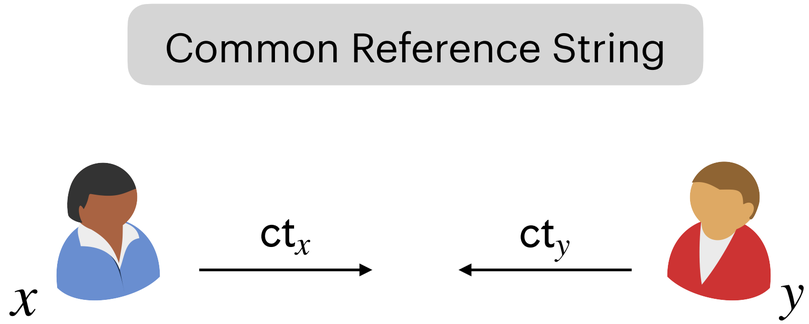
x

y

$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

Multi-Key Homomorphic Secret Sharing

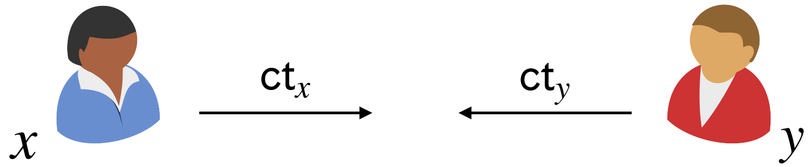


$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

Multi-Key Homomorphic Secret Sharing

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

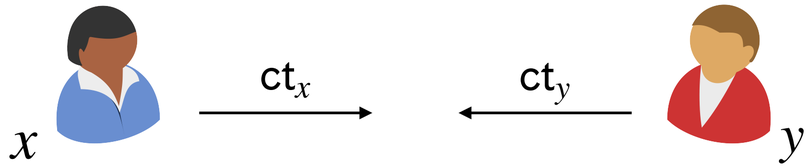
$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Multi-Key Homomorphic Secret Sharing

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Correctness $\quad z_A + z_B = C(x, y)$

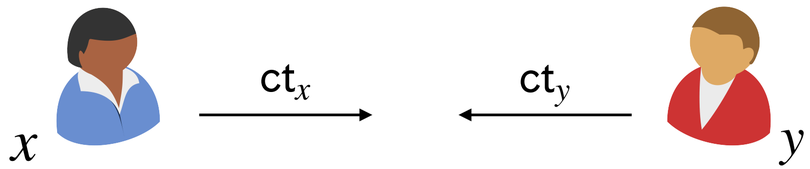
Security

ct_x ensures privacy of x

ct_y ensures privacy of y

Multi-Key Homomorphic Secret Sharing

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Analogue of Multi-Key FHE

Correctness $z_A + z_B = C(x, y)$

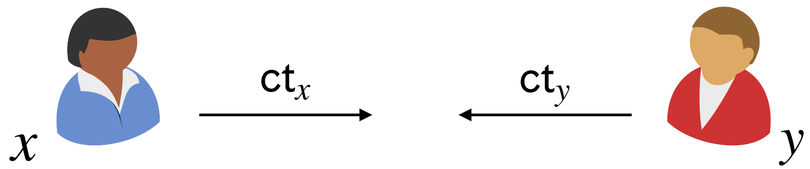
Security

ct_x ensures privacy of x

ct_y ensures privacy of y

Multi-Key Homomorphic Secret Sharing

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Analogue of Multi-Key FHE

Only known from Lattice assumptions

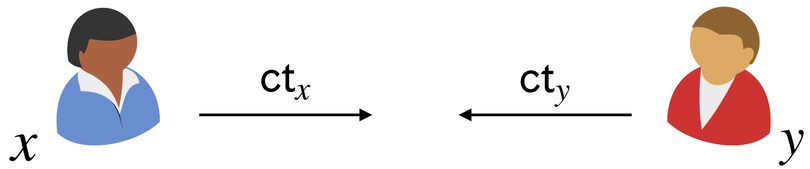
Correctness $z_A + z_B = C(x, y)$

Security ct_x ensures privacy of x

ct_y ensures privacy of y

Multi-Key Homomorphic Secret Sharing

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Analogue of Multi-Key FHE

Only known from Lattice assumptions

Correctness $z_A + z_B = C(x, y)$

Security

ct_x ensures privacy of x

ct_y ensures privacy of y

Multi-Key HSS can be constructed from group-based assumptions

Multi-Key Homomorphic Secret Sharing

Two round secure computation with succinct communication

Public-key pseudorandom correlation functions

MPC with offline phase communication linear in number of parties

Attribute-based key exchange

Before Multi-Key HSS: From lattice-based assumptions

After Multi-Key HSS: From group-based assumptions

Outline

Applications

Our Results

Constructing Multi-Key HSS

Outline

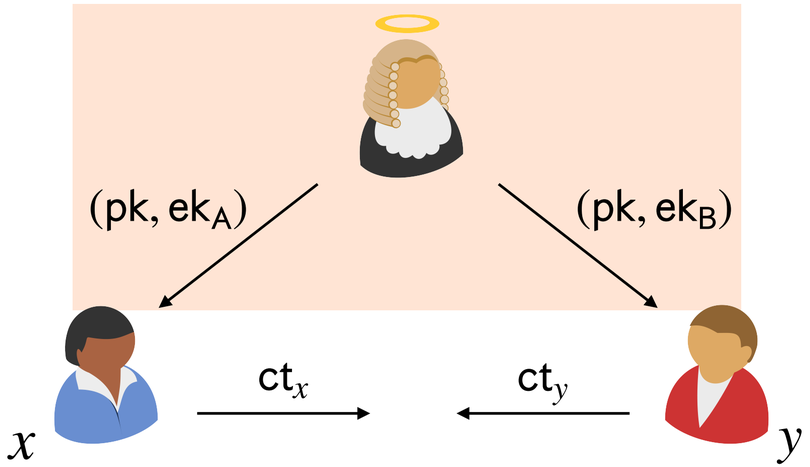
Applications

Our Results

Constructing Multi-Key HSS

Key Properties of Multi-Key HSS

HSS



$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

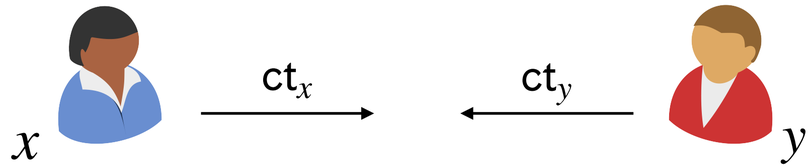
$z_A \leftarrow \mathsf{Eval}(\mathsf{ek}_A, C, \mathsf{ct}_x, \mathsf{ct}_y)$

$\mathsf{Encrypt}(\mathsf{pk}, y) \rightarrow \mathsf{ct}_y$

$\mathsf{Eval}(\mathsf{ek}_B, C, \mathsf{ct}_x, \mathsf{ct}_y) \rightarrow z_B$

Multi-Key HSS

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

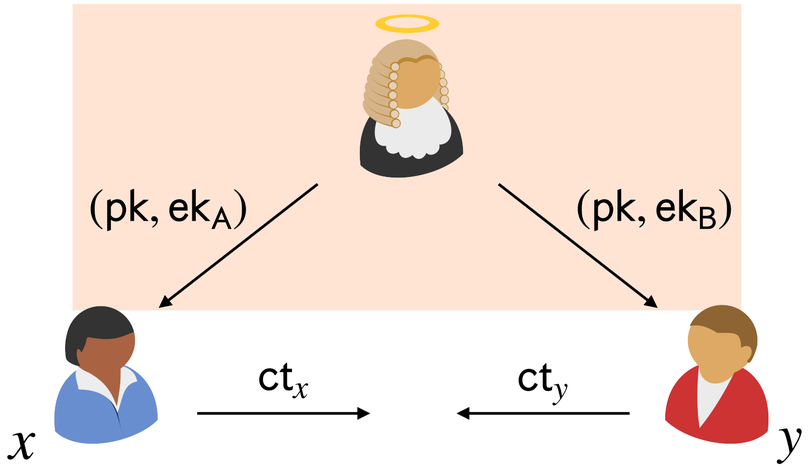
$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Reduces round complexity by avoiding correlated setup

Key Properties of Multi-Key HSS

HSS



$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

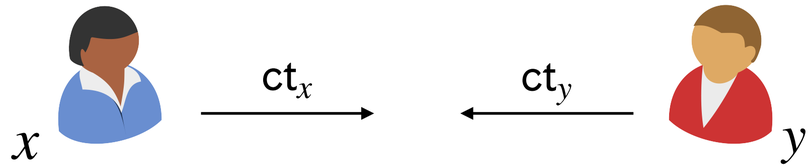
$z_A \leftarrow \mathsf{Eval}(\mathsf{ek}_A, C, \mathsf{ct}_x, \mathsf{ct}_y)$

$\mathsf{Encrypt}(\mathsf{pk}, y) \rightarrow \mathsf{ct}_y$

$\mathsf{Eval}(\mathsf{ek}_B, C, \mathsf{ct}_x, \mathsf{ct}_y) \rightarrow z_B$

Multi-Key HSS

Common Reference String



$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

Reduces round complexity by avoiding correlated setup

Reusability of input encodings

Key Properties of Multi-Key HSS

HSS

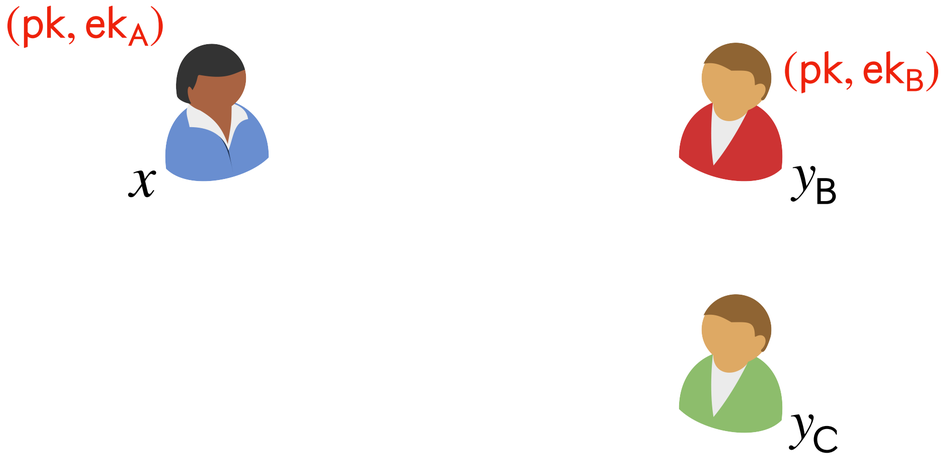
Multi-Key HSS

Reduces round complexity by avoiding correlated setup

Reusability of input encodings

Key Properties of Multi-Key HSS

HSS



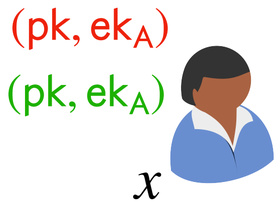
Multi-Key HSS

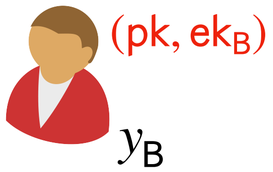
Reduces round complexity by avoiding correlated setup

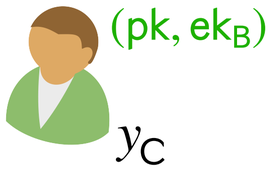
Reusability of input encodings

Key Properties of Multi-Key HSS

HSS







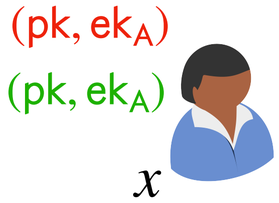
Multi-Key HSS

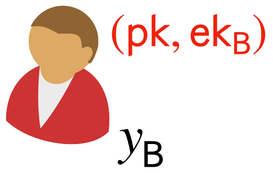
Reduces round complexity by avoiding correlated setup

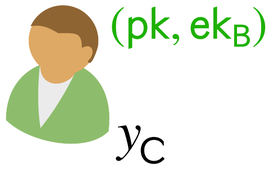
Reusability of input encodings

Key Properties of Multi-Key HSS

HSS







$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

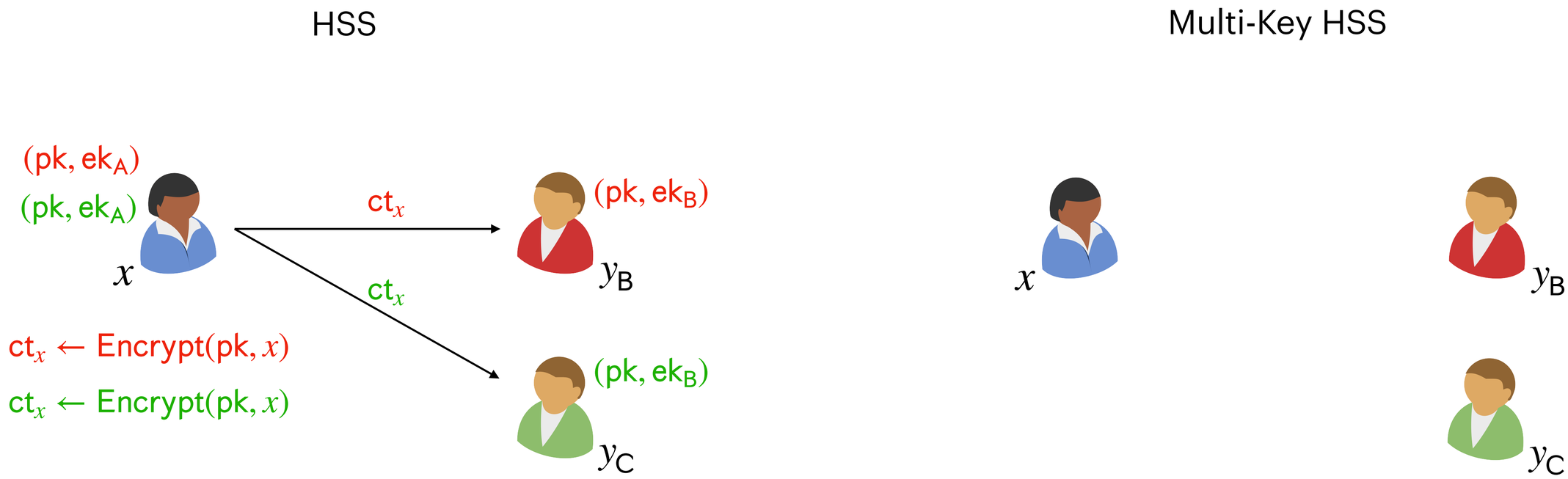
$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

Multi-Key HSS

Reduces round complexity by avoiding correlated setup

Reusability of input encodings

Key Properties of Multi-Key HSS

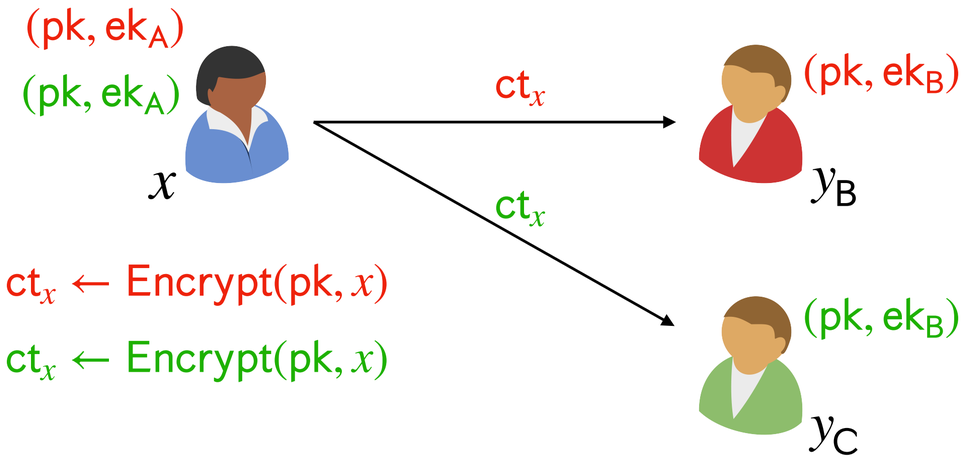


Reduces round complexity by avoiding correlated setup

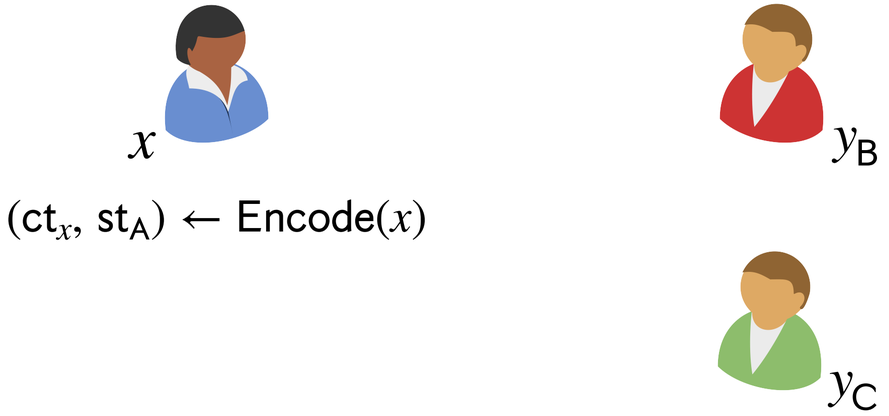
Reusability of input encodings

Key Properties of Multi-Key HSS

HSS



Multi-Key HSS

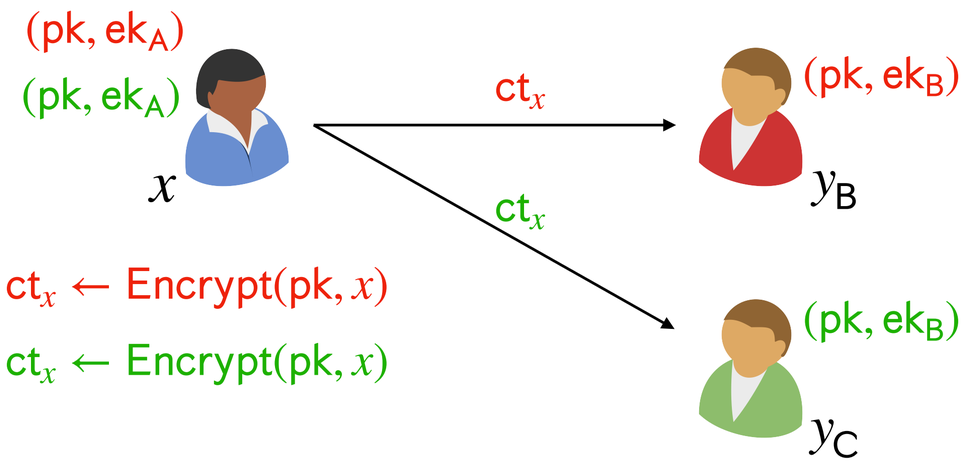


Reduces round complexity by avoiding correlated setup

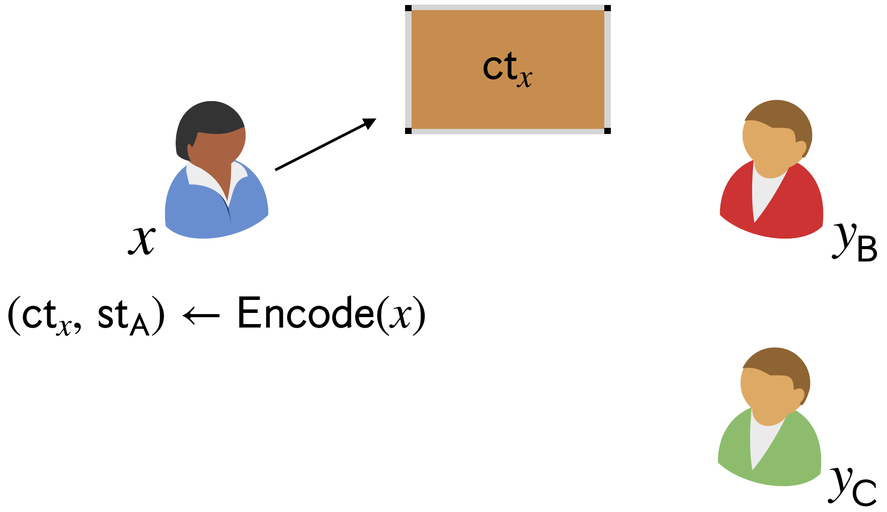
Reusability of input encodings

Key Properties of Multi-Key HSS

HSS



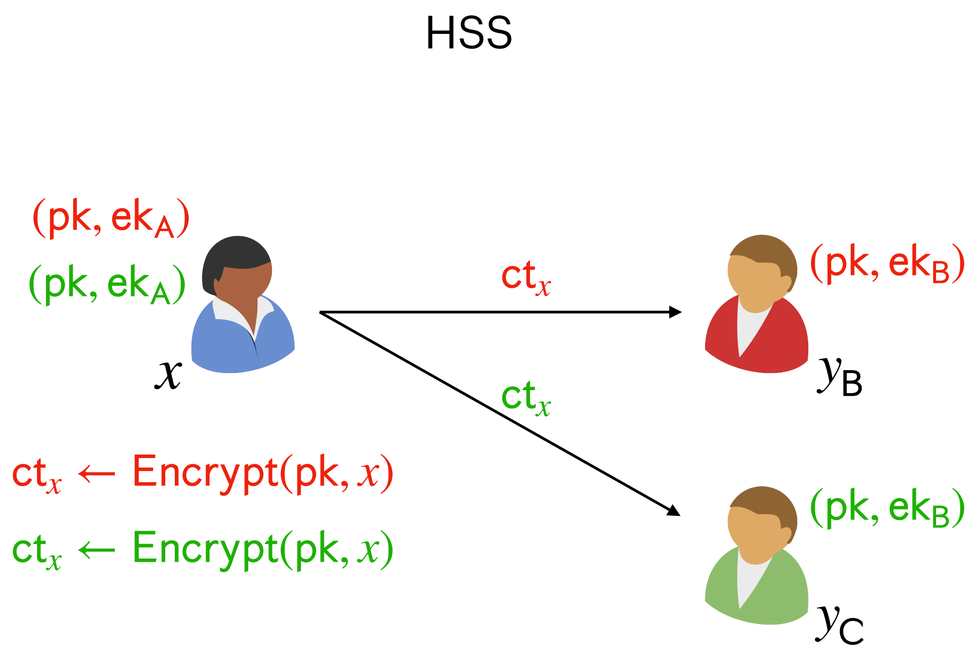
Multi-Key HSS

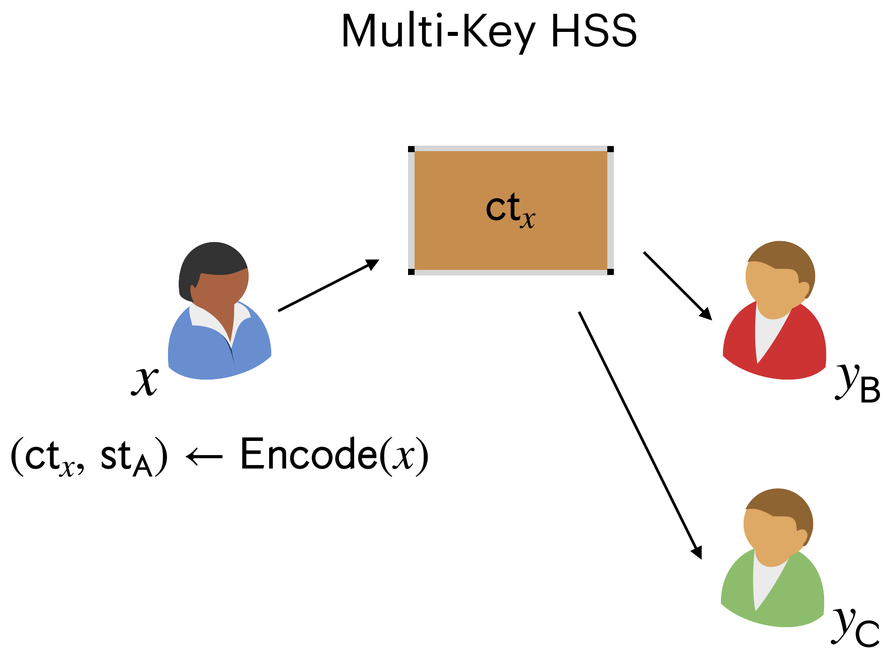


Reduces round complexity by avoiding correlated setup

Reusability of input encodings

Key Properties of Multi-Key HSS





Reduces round complexity by avoiding correlated setup

Reusability of input encodings

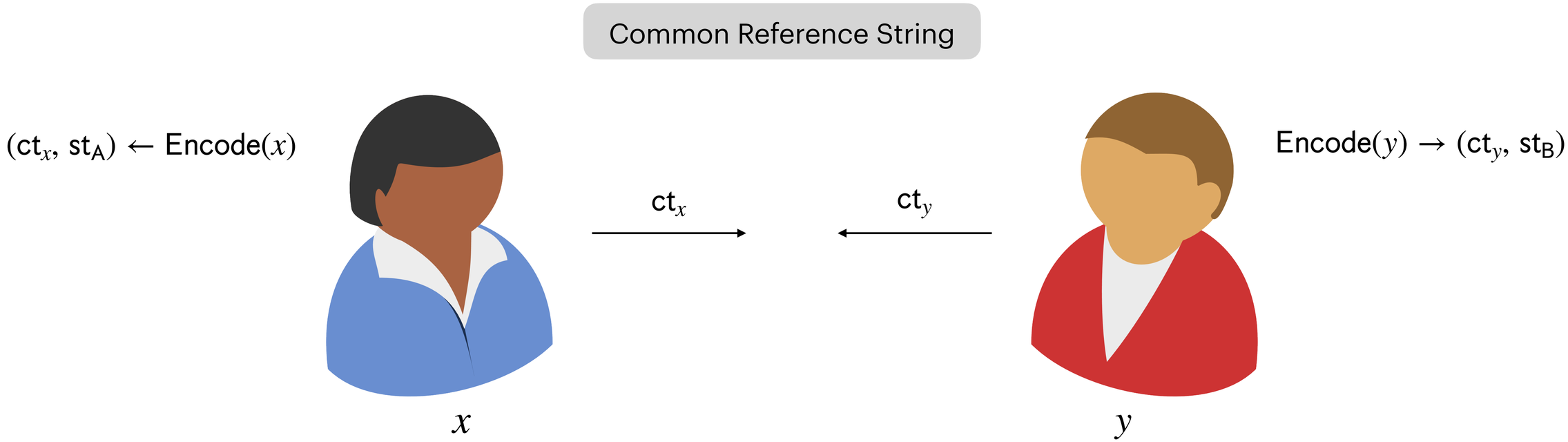
Application 1: Succinct Two-round Secure Computation

Common Reference String

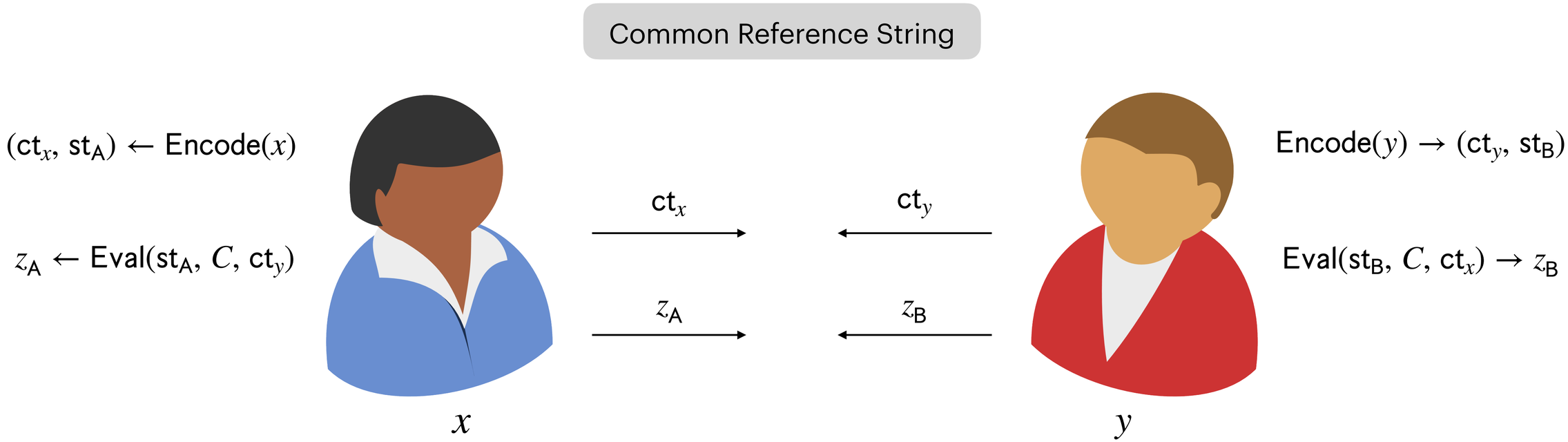
x

y

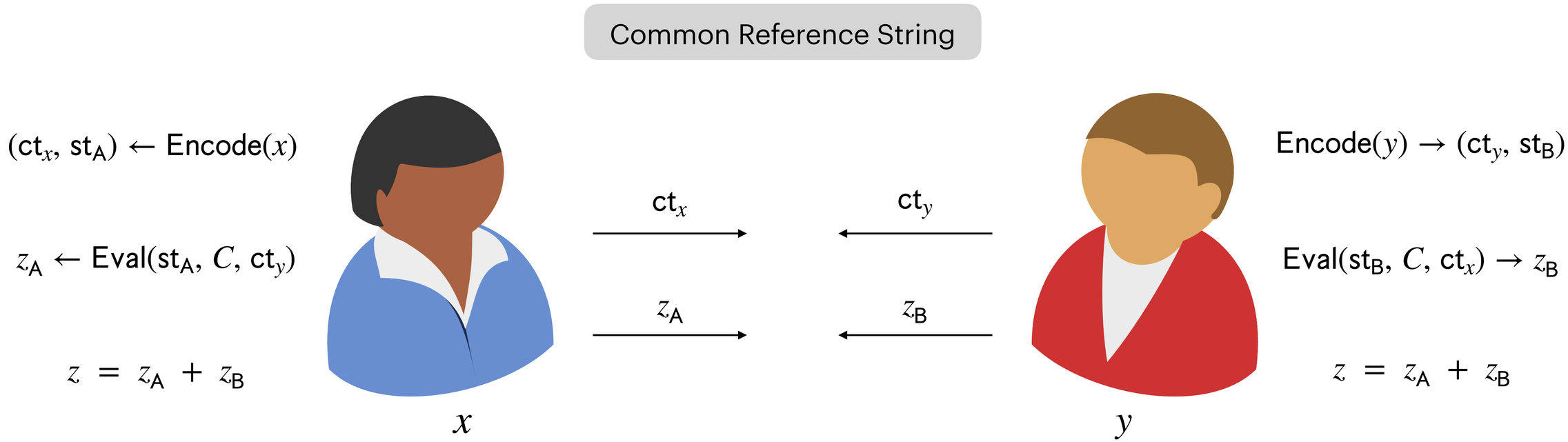
Application 1: Succinct Two-round Secure Computation



Application 1: Succinct Two-round Secure Computation



Application 1: Succinct Two-round Secure Computation



Application 1: Succinct Two-round Secure Computation

Common Reference String

$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$z = z_A + z_B$

x

ct_x →

z_A →

← ct_y

← z_B

y

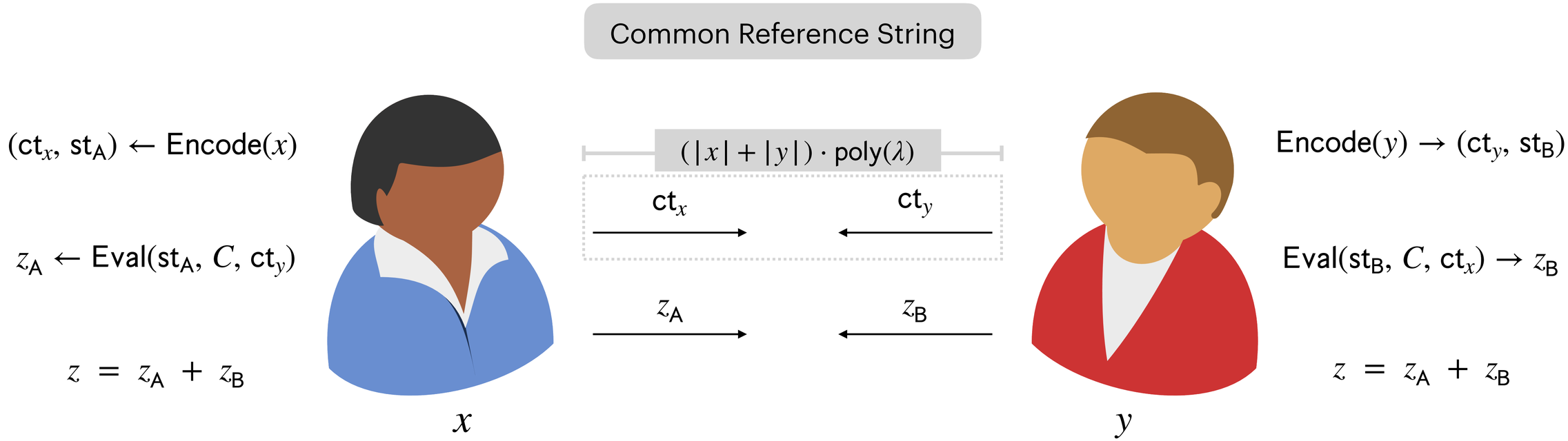
$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

$z = z_A + z_B$

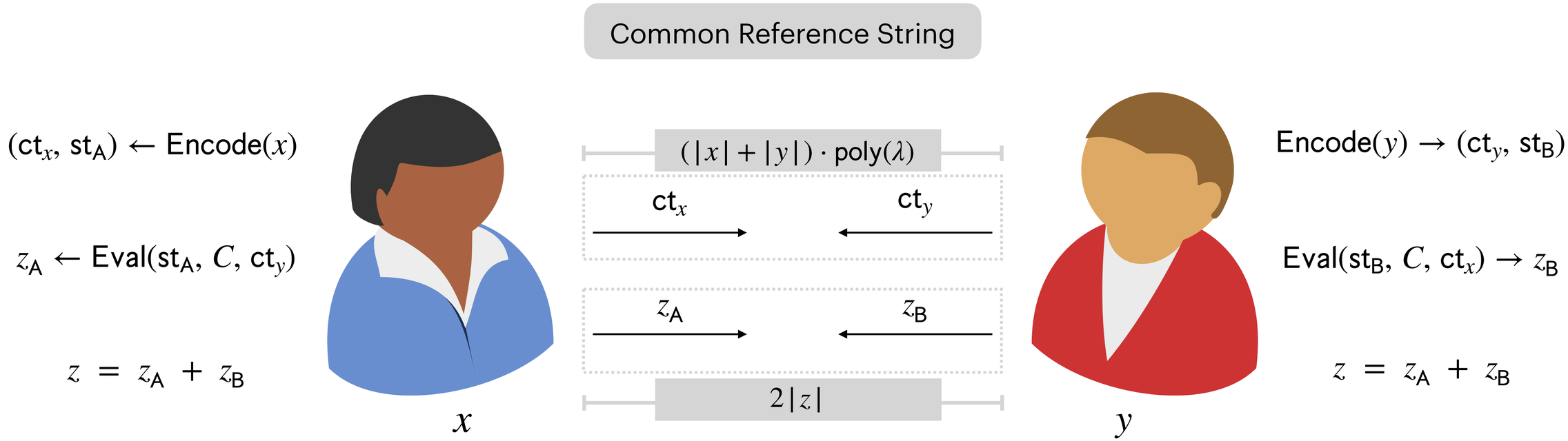
Communication is independent of evaluated circuit

Application 1: Succinct Two-round Secure Computation



Communication is independent of evaluated circuit

Application 1: Succinct Two-round Secure Computation



Communication is independent of evaluated circuit

Application 1: Succinct Two-round Secure Computation

Common Reference String

$(\mathsf{ct}_x, \mathsf{st}_A) \leftarrow \mathsf{Encode}(x)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{st}_A, C, \mathsf{ct}_y)$

$z = z_A + z_B$

x

$(|x| + |y|) \cdot \mathsf{poly}(\lambda)$

ct_x →

← ct_y

z_A →

← z_B

$2|z|$

y

$\mathsf{Encode}(y) \rightarrow (\mathsf{ct}_y, \mathsf{st}_B)$

$\mathsf{Eval}(\mathsf{st}_B, C, \mathsf{ct}_x) \rightarrow z_B$

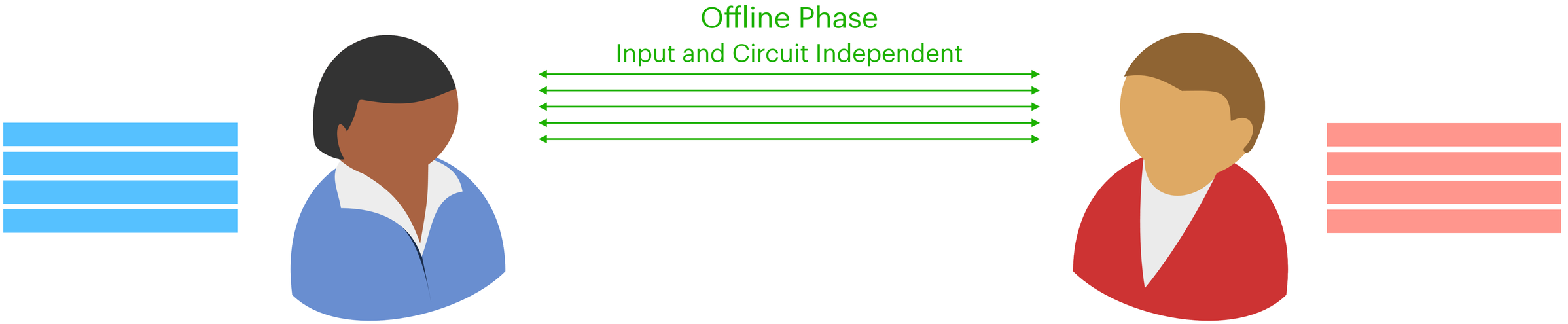
$z = z_A + z_B$

Communication is independent of evaluated circuit

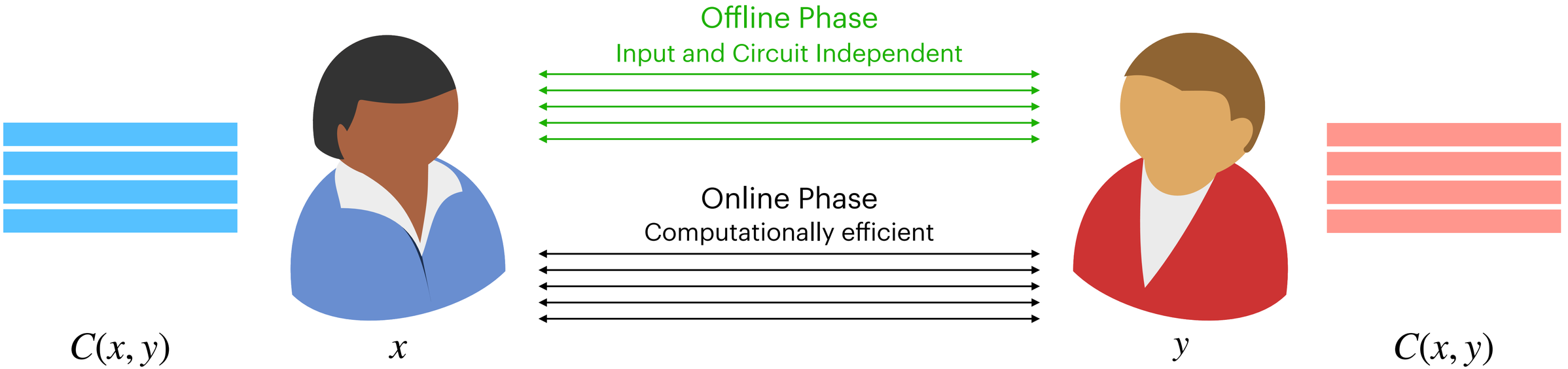
Two-round protocol in the CRS model

Preprocessing Model for Secure Computation

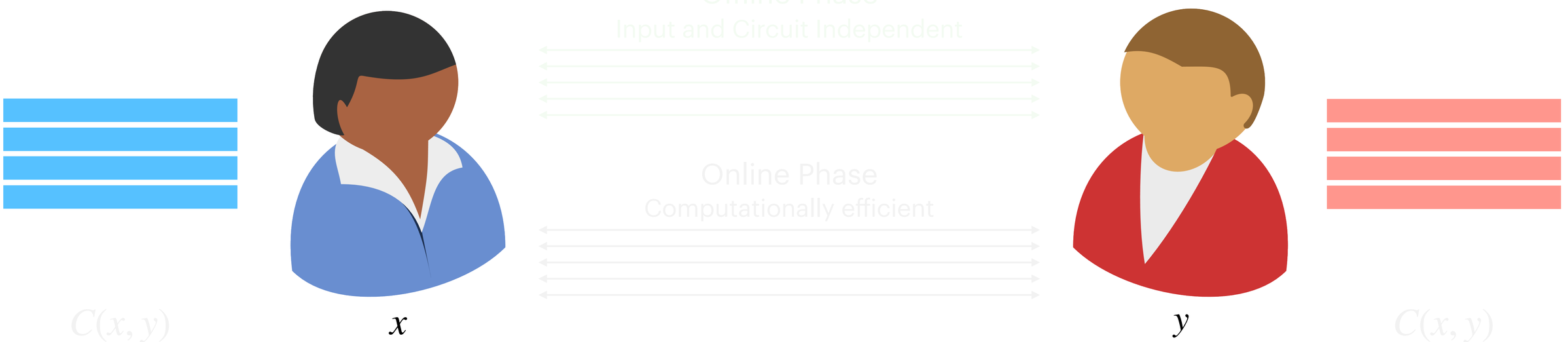
Preprocessing Model for Secure Computation



Preprocessing Model for Secure Computation

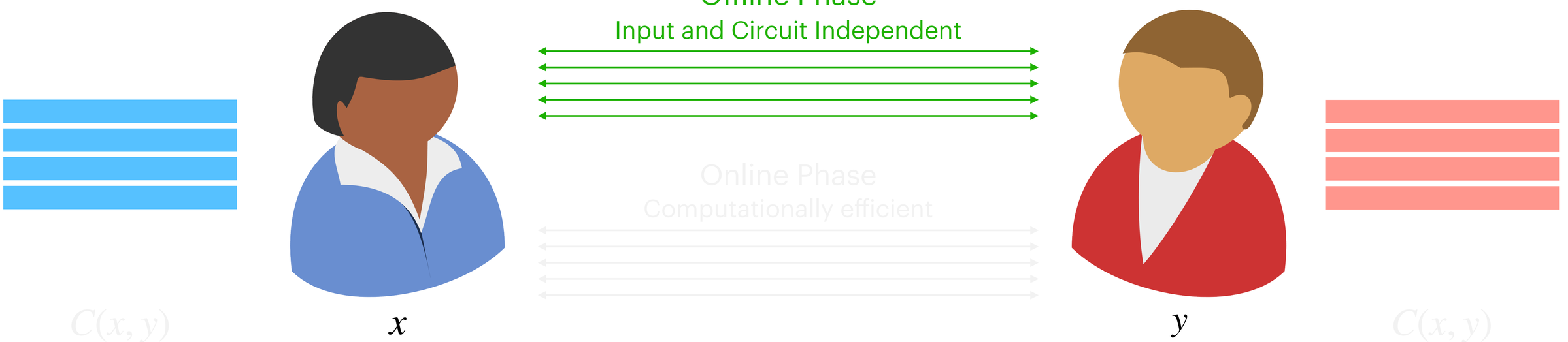


Preprocessing Model for Secure Computation



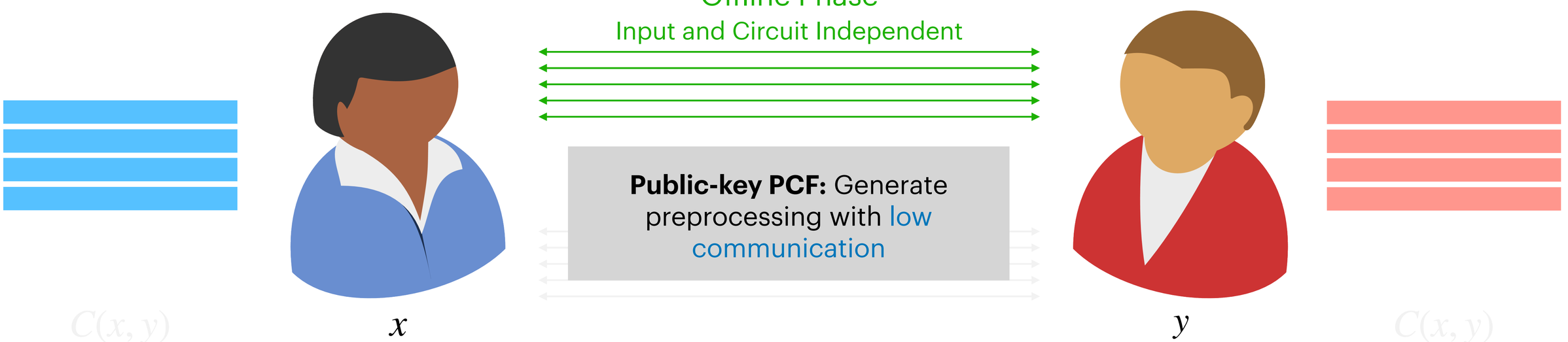
Size of preprocessing is linear in the size of circuit evaluated $|C|$

Preprocessing Model for Secure Computation



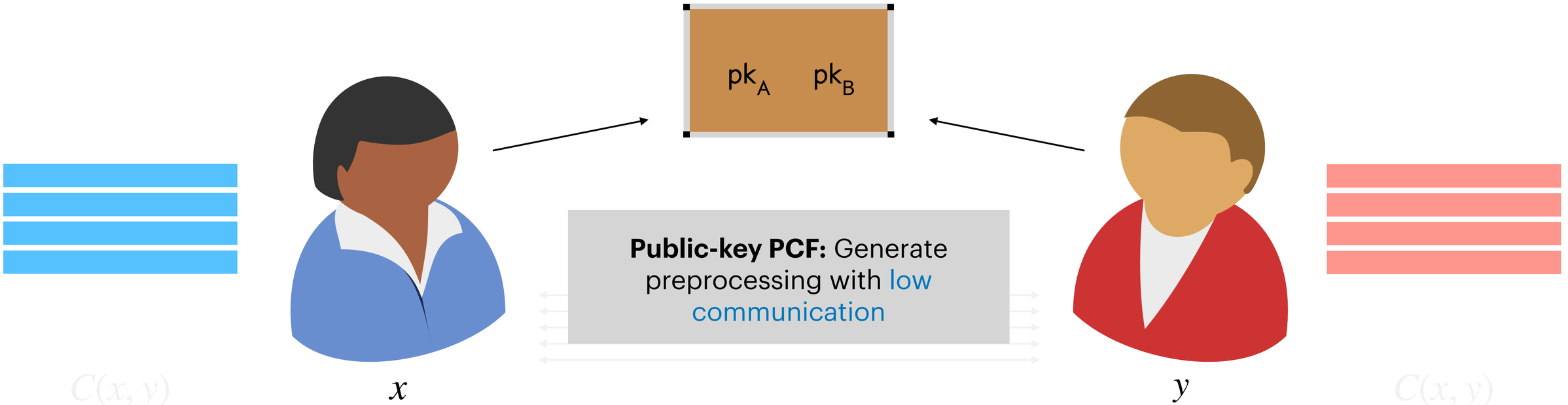
Size of preprocessing is linear in the size of circuit evaluated $|C|$

Preprocessing Model for Secure Computation



Size of preprocessing is linear in the size of circuit evaluated $|C|$

Preprocessing Model for Secure Computation



Public-key PCF: Generate preprocessing with low communication

Size of preprocessing is linear in the size of circuit evaluated $|C|$

Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

Public-Key Pseudorandom Correlation Function (PCF)

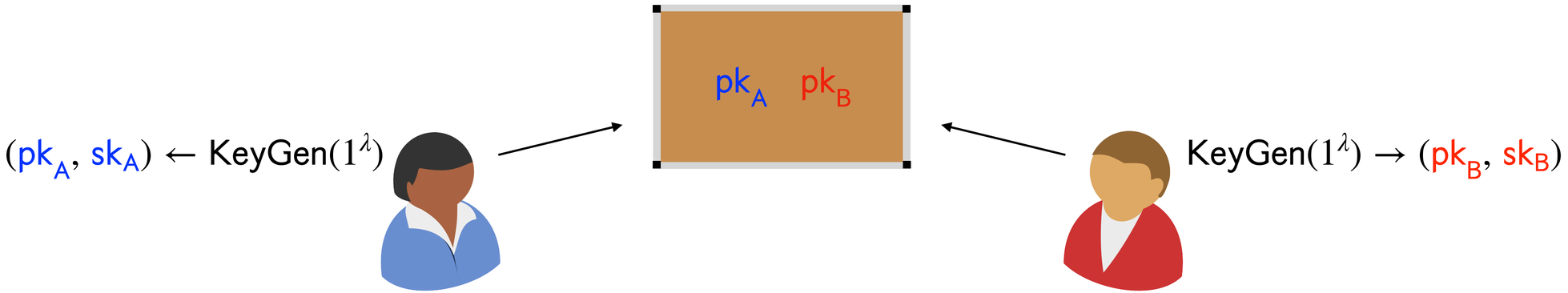
[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

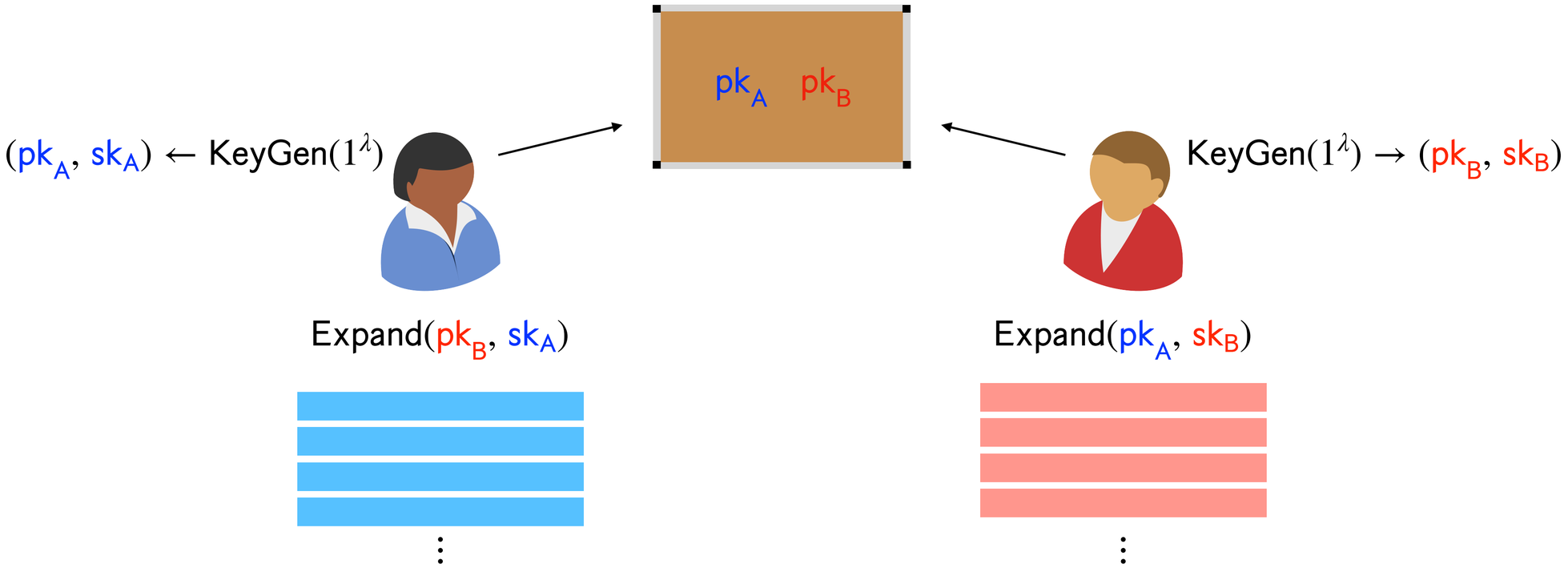
Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]



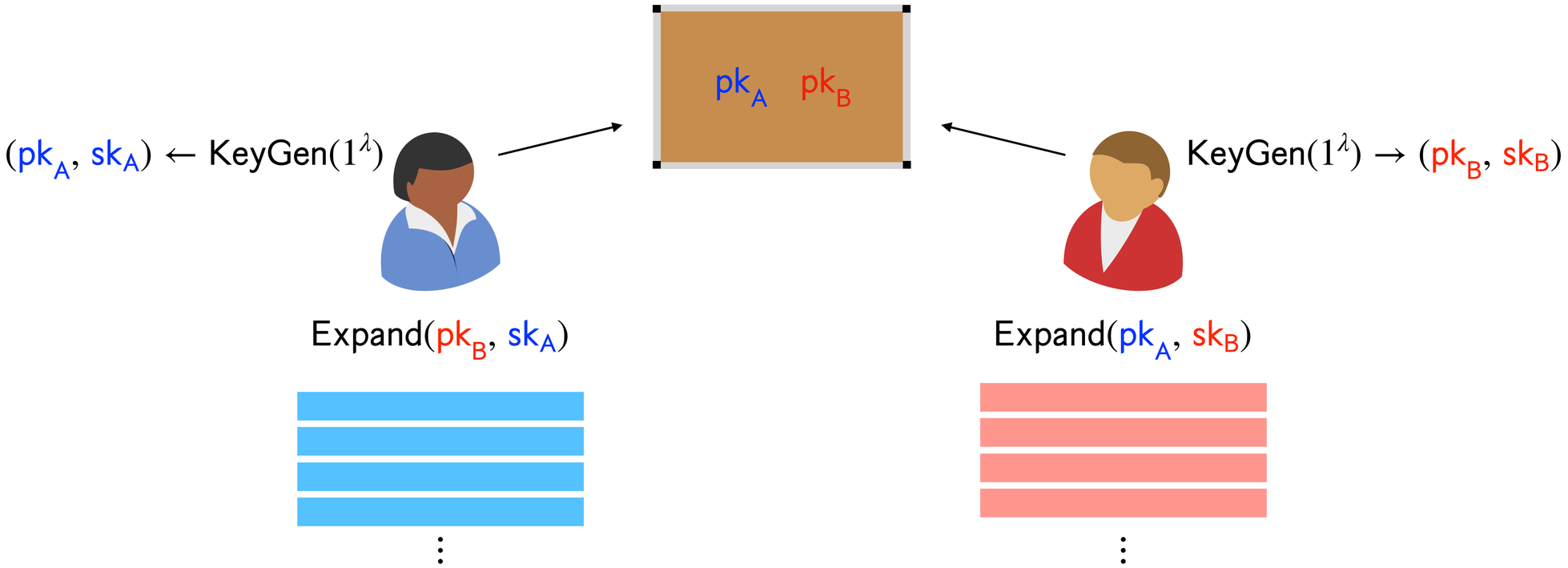
Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

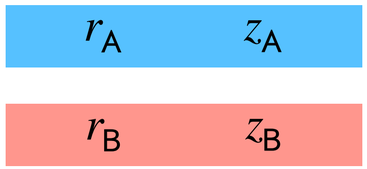


Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

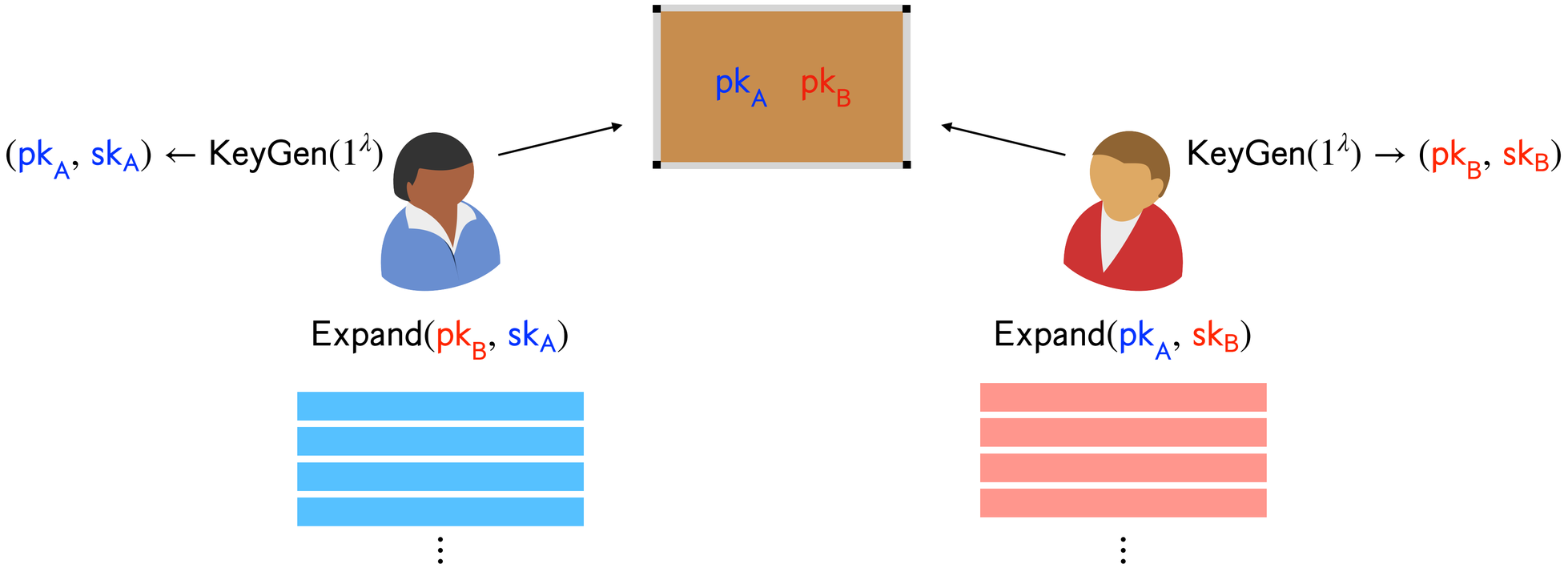


Additive Correlations:



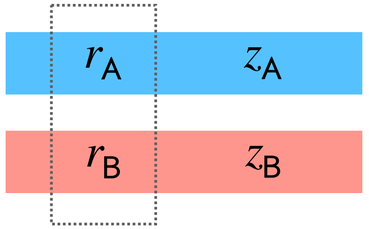
Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]



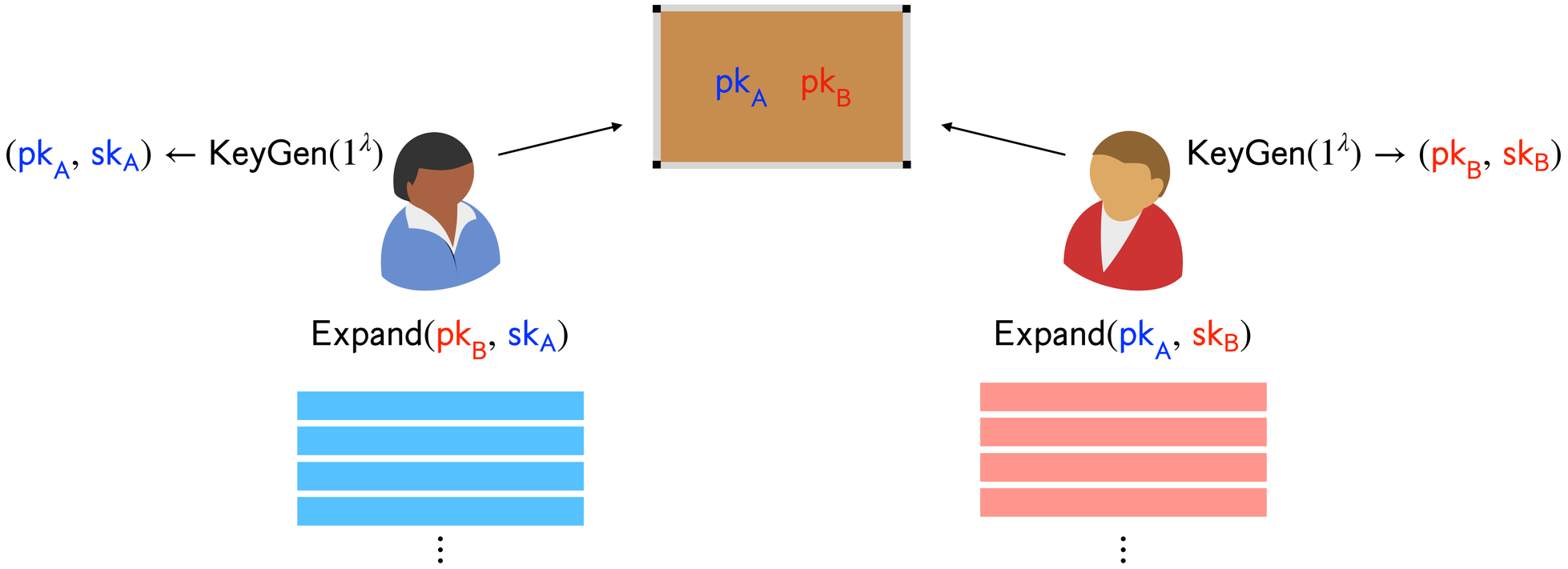
Private pseudorandom value

Additive Correlations:

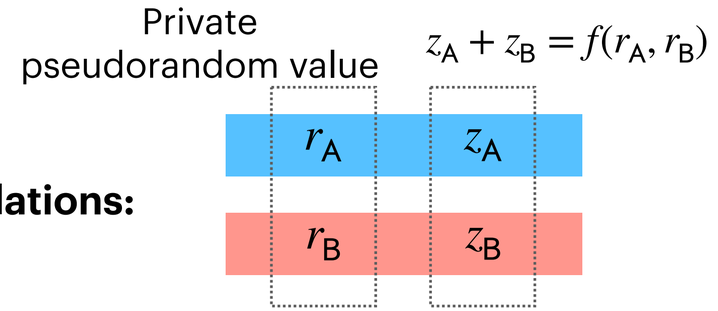


Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

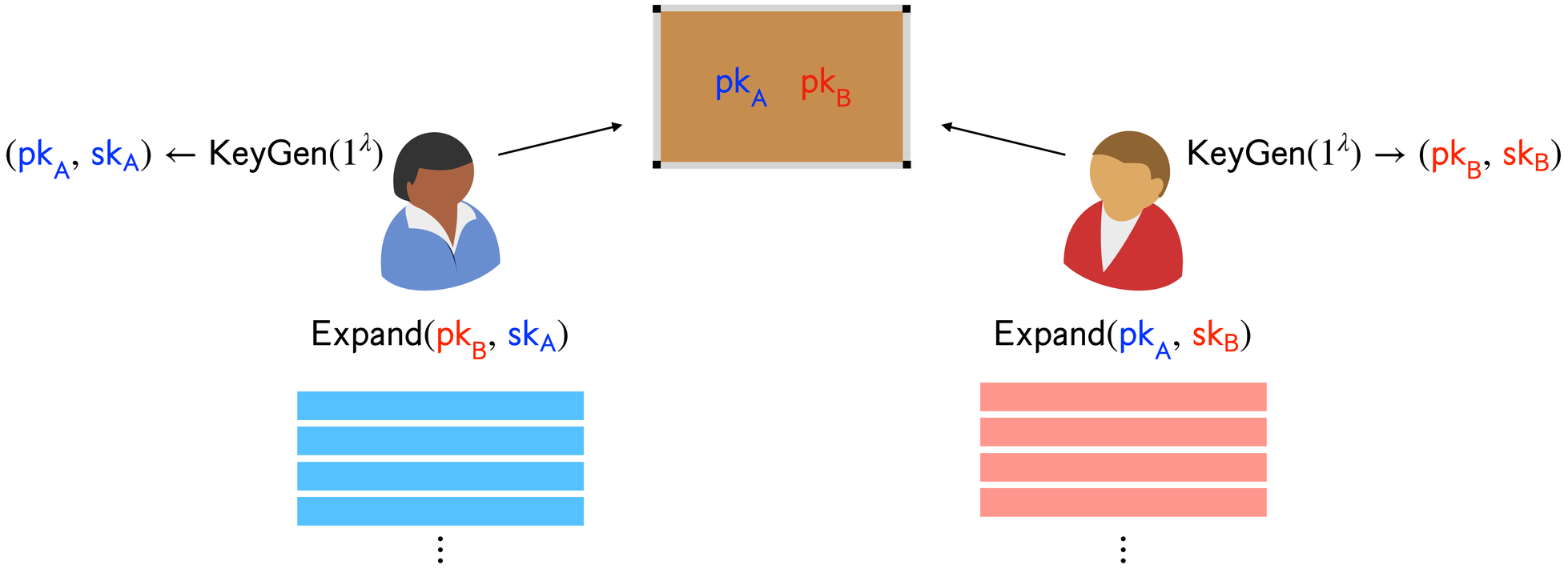


Additive Correlations:



Public-Key Pseudorandom Correlation Function (PCF)

[Orlandi-Scholl-Yakoubov'21] [Bui-Couteau-Meyer-Passalègue-Riahinia'24]

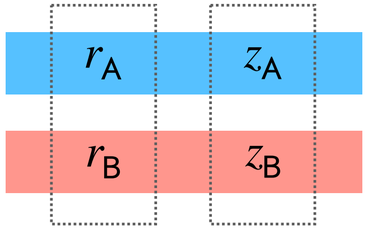


Captures most MPC correlations
e.g., correlated OT, OLE, and
Beaver triples

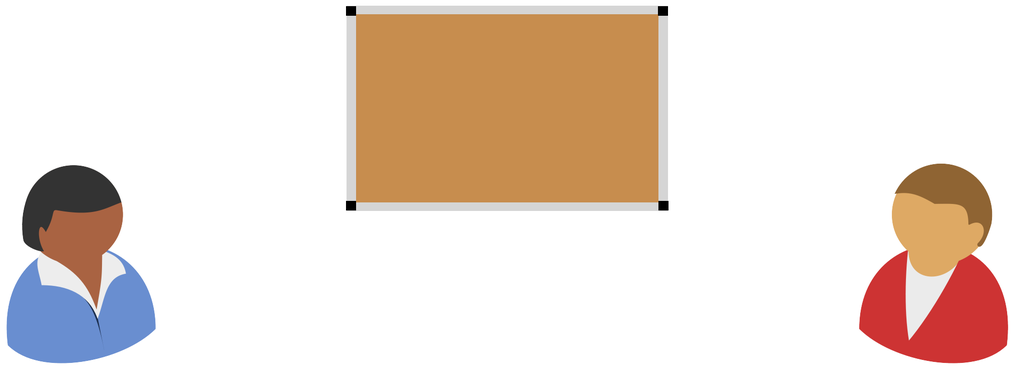
Additive Correlations:

Private
pseudorandom value

$z_A + z_B = f(r_A, r_B)$



Application 2: Public-Key PCFs for Additive Correlations



Application 2: Public-Key PCFs for Additive Correlations

$k_A \leftarrow \{0,1\}^\lambda$

$k_B \leftarrow \{0,1\}^\lambda$

Application 2: Public-Key PCFs for Additive Correlations

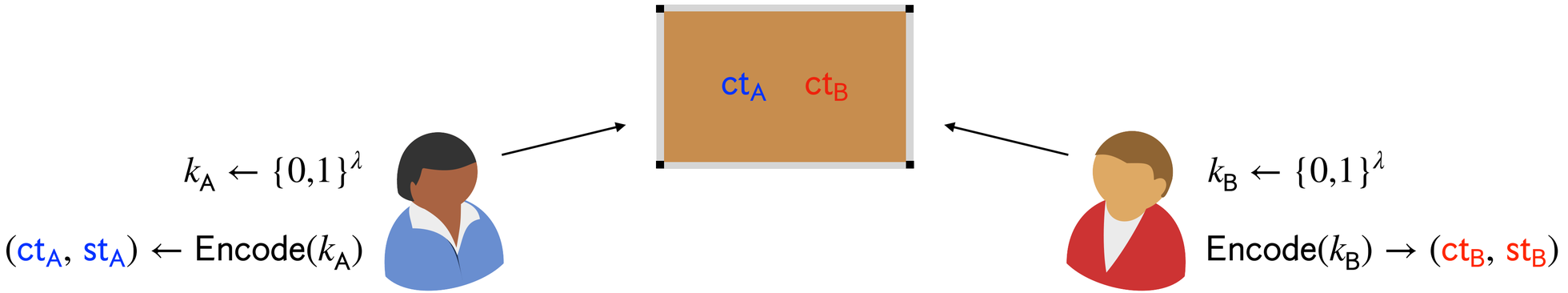
$k_A \leftarrow \{0,1\}^\lambda$

$(ct_A, st_A) \leftarrow \text{Encode}(k_A)$

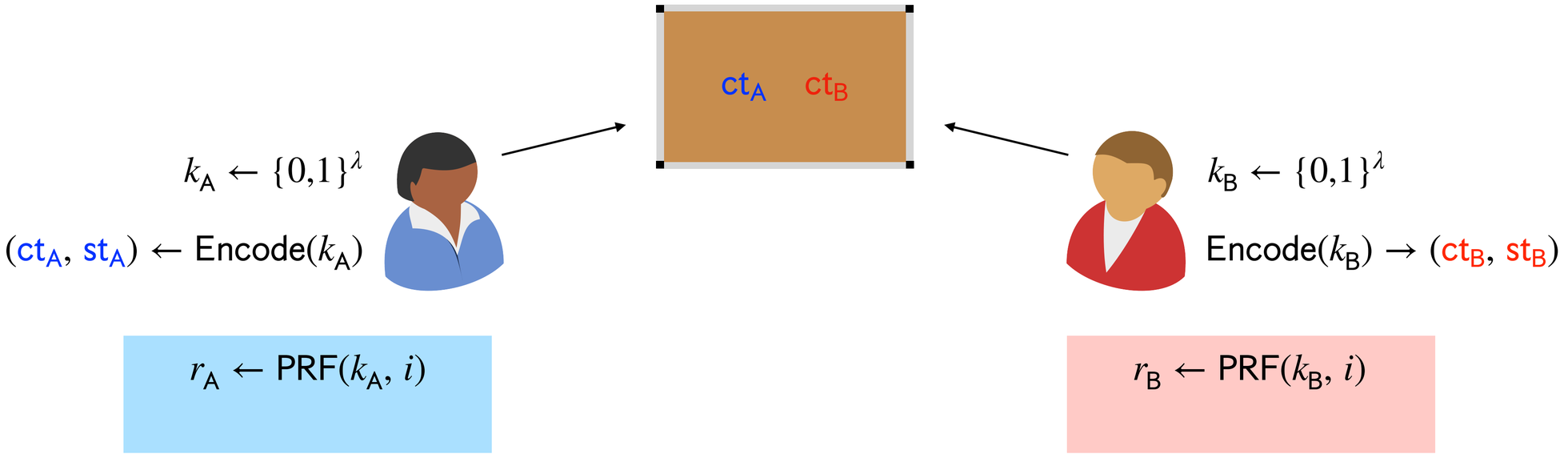
$k_B \leftarrow \{0,1\}^\lambda$

$\text{Encode}(k_B) \rightarrow (ct_B, st_B)$

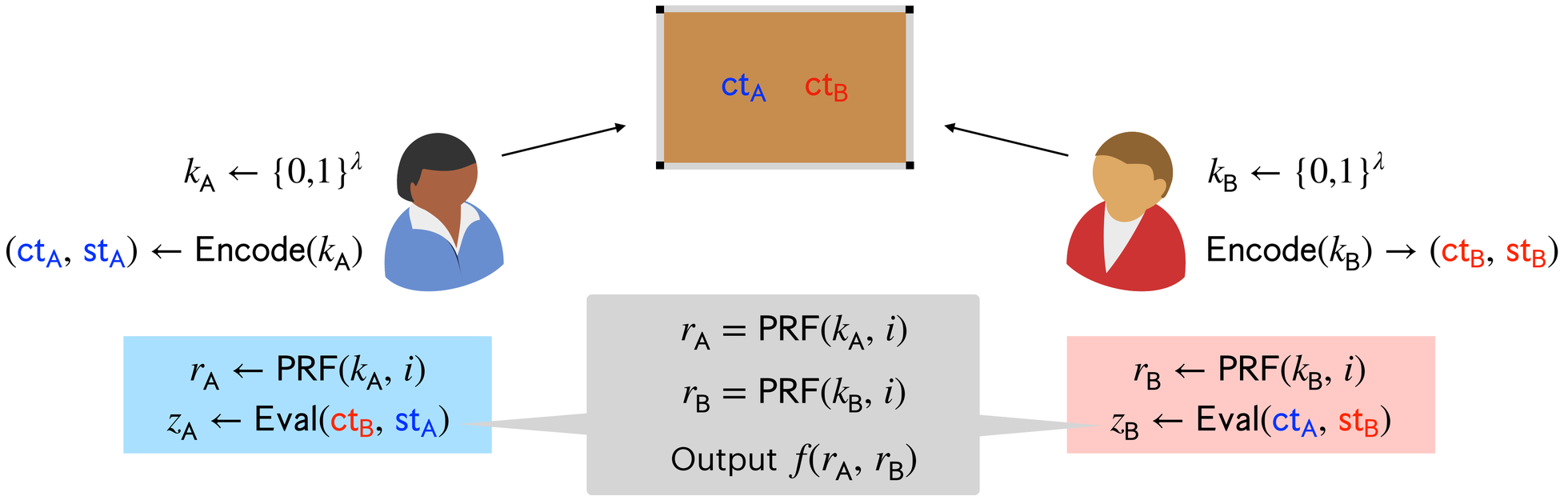
Application 2: Public-Key PCFs for Additive Correlations



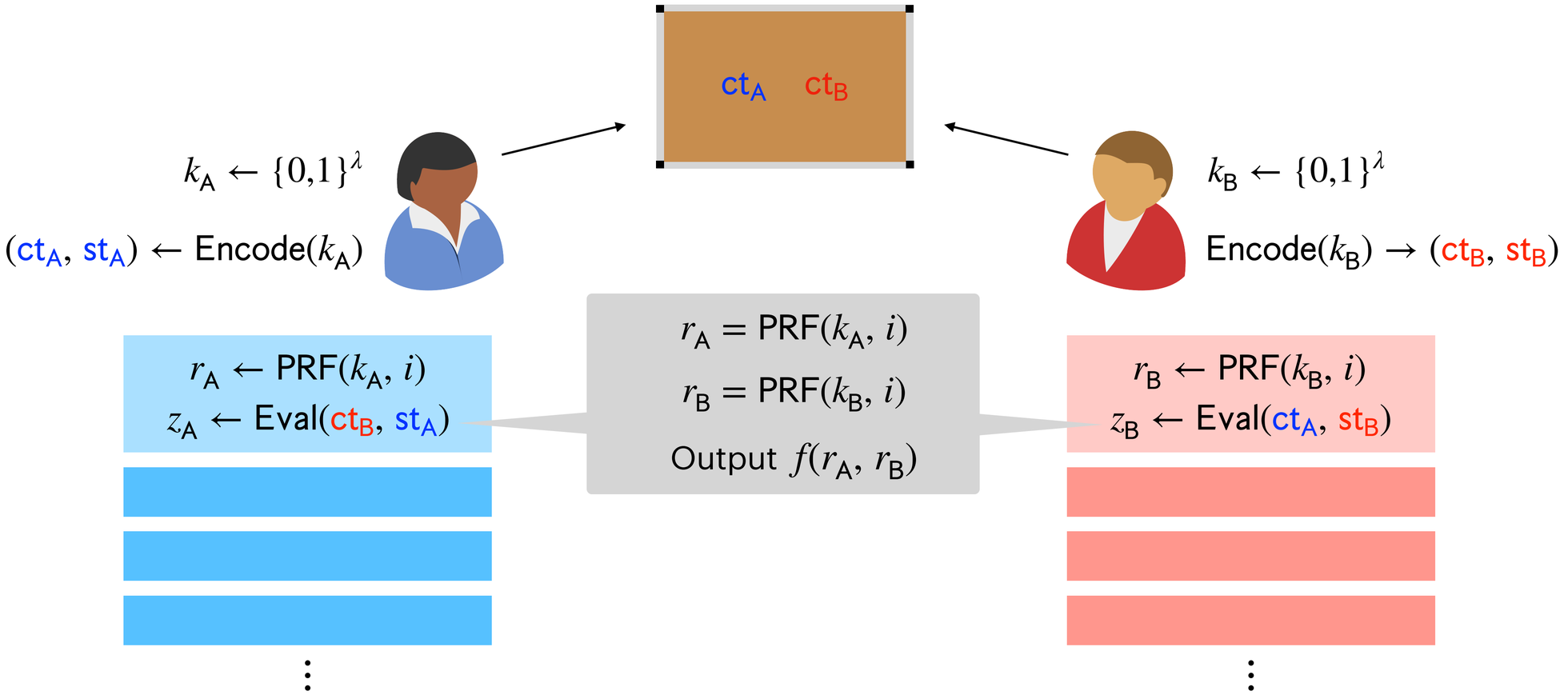
Application 2: Public-Key PCFs for Additive Correlations



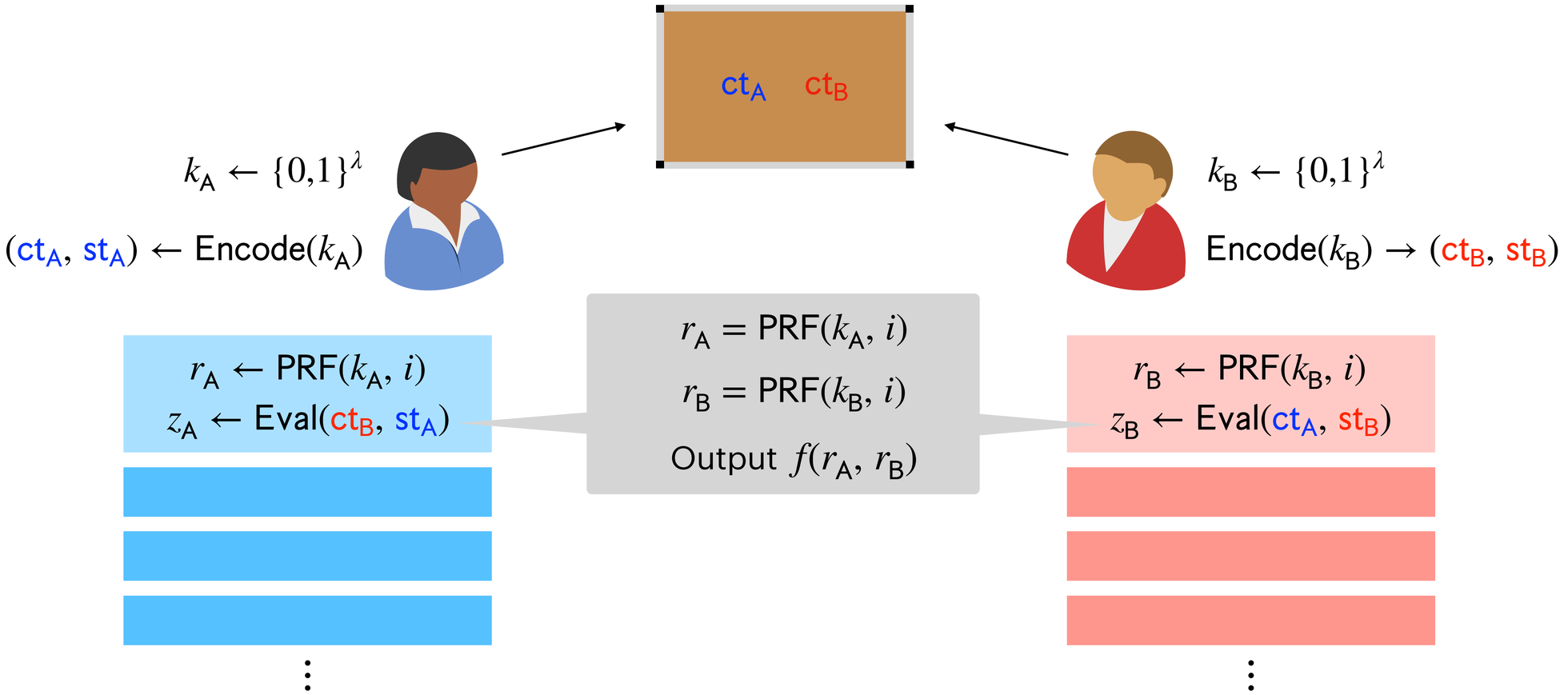
Application 2: Public-Key PCFs for Additive Correlations



Application 2: Public-Key PCFs for Additive Correlations



Application 2: Public-Key PCFs for Additive Correlations



Reusability of input encodings $\Longrightarrow$ public-key can be used by multiple parties

Outline

Applications

Our Results

Constructing Multi-Key HSS

Our Results: Multi-Key HSS

Two-party multi-key HSS schemes for evaluating NC^1 functions

DDH

DCR

DDH over class groups

Previously known only from LWE and $i\mathcal{O}$ + DDH [Dodis-Halevi-Rothblum-Wichs'16]

Our Results: Multi-Key HSS

Two-party multi-key HSS schemes for evaluating NC^1 functions

	HSS Schemes from Prior Works (Require Correlated Setup)
DDH	[Boyle-Gilboa-Ishai'16]
DCR	[Orlandi-Scholl-Yakoubov'21] [Roy-Singh'21]
DDH over class groups	[Abram-Damgård-Orlandi-Scholl'22]

Previously known only from LWE and $i\mathcal{O}$ + DDH [Dodis-Halevi-Rothblum-Wichs'16]

Our Results: Multi-Key HSS

Two-party multi-key HSS schemes for evaluating NC^1 functions

Inverse polynomial correctness error

	HSS Schemes from Prior Works (Require Correlated Setup)
DDH	[Boyle-Gilboa-Ishai'16]
DCR	[Orlandi-Scholl-Yakoubov'21] [Roy-Singh'21]
DDH over class groups	[Abram-Damgård-Orlandi-Scholl'22]

Previously known only from LWE and $i\mathcal{O}$ + DDH [Dodis-Halevi-Rothblum-Wichs'16]

Our Results: Multi-Key HSS

Two-party multi-key HSS schemes for evaluating NC^1 functions

		HSS Schemes from Prior Works (Require Correlated Setup)
Transparent setup	DDH	[Boyle-Gilboa-Ishai'16]
	DCR	[Orlandi-Scholl-Yakoubov'21] [Roy-Singh'21]
Transparent setup	DDH over class groups	[Abram-Damgård-Orlandi-Scholl'22]

Previously known only from LWE and $i\mathcal{O}$ + DDH [Dodis-Halevi-Rothblum-Wichs'16]

Our Results: **Applications** of Multi-Key HSS

Two-round succinct 2PC for NC^1 circuits in the CRS model

DDH

DCR

DDH over class groups

Previously known only from multi-key FHE [Mukherjee-Wichs'16]

Our Results: **Applications** of Multi-Key HSS

Two-round succinct 2PC for NC^1 circuits in the CRS model

DDH

DCR

DDH over class groups

Previously from group-based assumptions

3 round protocol in the CRS model

Previously known only from multi-key FHE [Mukherjee-Wichs'16]

Our Results: **Applications** of Multi-Key HSS

Two-round succinct 2PC for NC^1 circuits in the CRS model

DDH | DCR | DDH over class groups

Previously from group-based assumptions

3 round protocol in the CRS model

Previously known only from multi-key FHE [Mukherjee-Wichs'16]

Attribute-based NIKE supporting NC^1 predicates

DCR | DDH over class groups

Our Results: **Applications** of Multi-Key HSS

Public-key PCFs for NC^1 additive correlations

DCR

DDH over class groups

Our Results: **Applications** of Multi-Key HSS

Public-key PCFs for NC^1 additive correlations

Includes Beaver triples, correlated OT, OLE etc.,

DCR

DDH over class groups

Our Results: **Applications** of Multi-Key HSS

Public-key PCFs for NC^1 additive correlations

DCR

DDH over class groups

Previously from group-based assumptions

Public-key PCFs for OT and Vector-OLE correlations

Our Results: **Applications** of Multi-Key HSS

Public-key PCFs for NC^1 additive correlations

DCR

DDH over class groups

Previously from group-based assumptions

Public-key PCFs for OT and Vector-OLE correlations

n-party secure computation protocol in the preprocessing model with communication complexity

- Offline phase: $\mathsf{poly}(\lambda) \cdot n$
- Online phase: $O(|C| \cdot n)$

DCR

DDH over class groups

Previously from group-based assumptions

Offline communication complexity $\mathsf{poly}(\lambda) \cdot n^2$

Outline

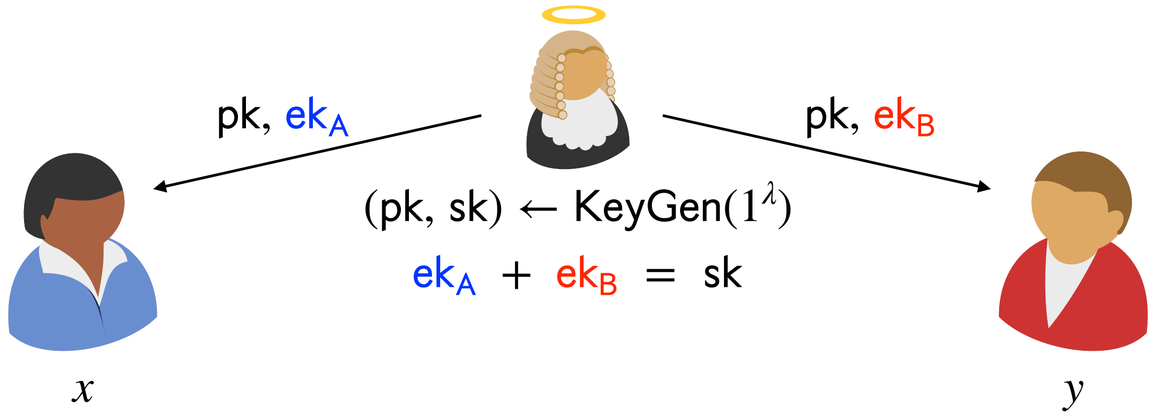
Applications

Our Results

Constructing Multi-Key HSS

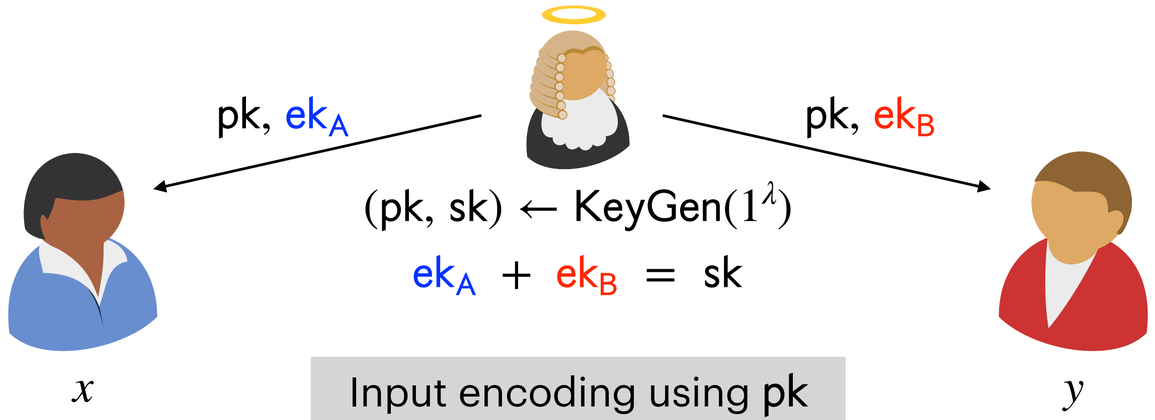
Group-based HSS Schemes

[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]



Group-based HSS Schemes

[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]

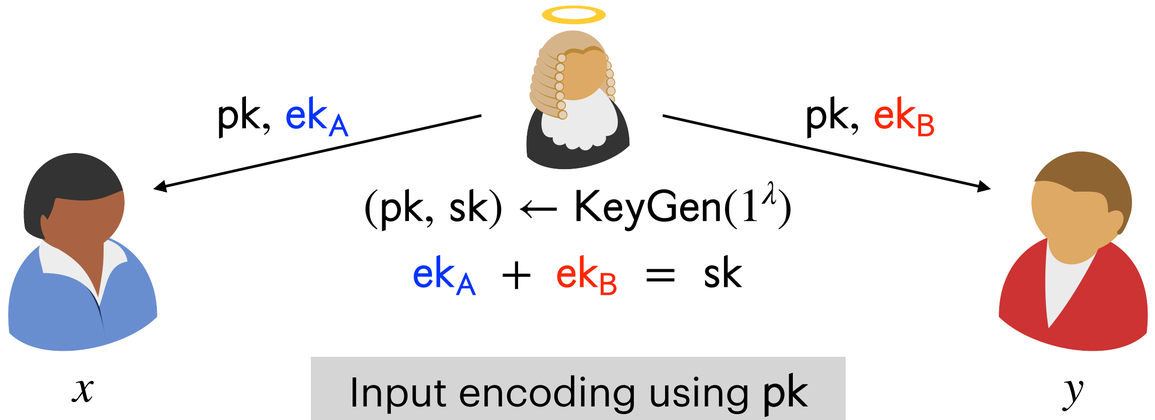


Input encoding using pk

$$ct_x = \begin{pmatrix} Enc(pk, x) \\ Enc(pk, sk \cdot x) \end{pmatrix}$$

Group-based HSS Schemes

[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]



Input encoding using pk

$$ct_x = \begin{pmatrix} Enc(pk, x) \\ Enc(pk, sk \cdot x) \end{pmatrix}$$

$Enc(pk, x)$
$Enc(pk, sk \cdot x)$

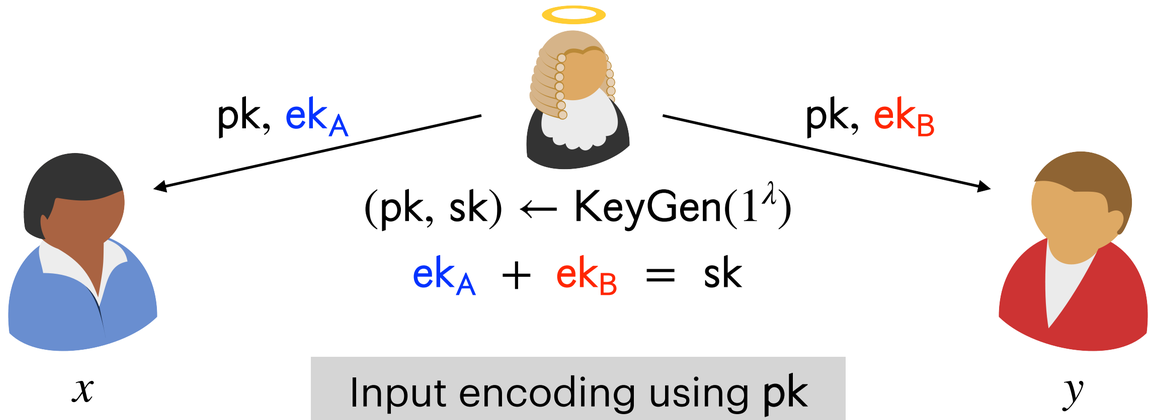
$Enc(pk, y)$
$Enc(pk, sk \cdot y)$

$Enc(pk, x)$
$Enc(pk, sk \cdot x)$

$Enc(pk, y)$
$Enc(pk, sk \cdot y)$

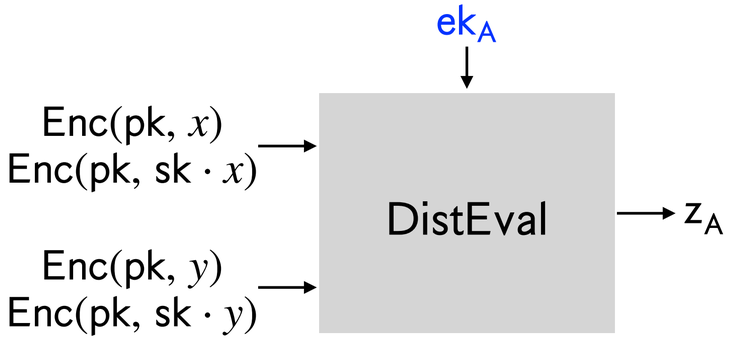
Group-based HSS Schemes

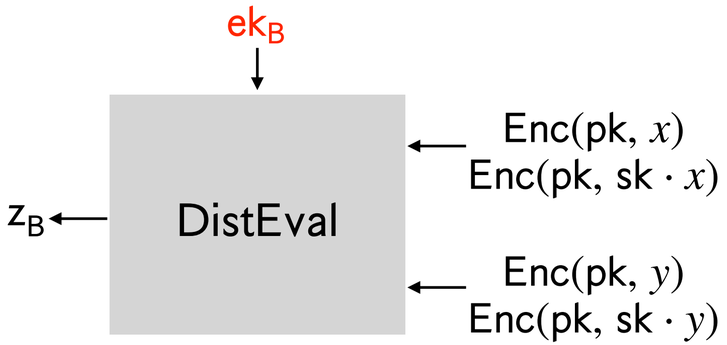
[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]



Input encoding using pk

$$ct_x = \begin{pmatrix} Enc(pk, x) \\ Enc(pk, sk \cdot x) \end{pmatrix}$$



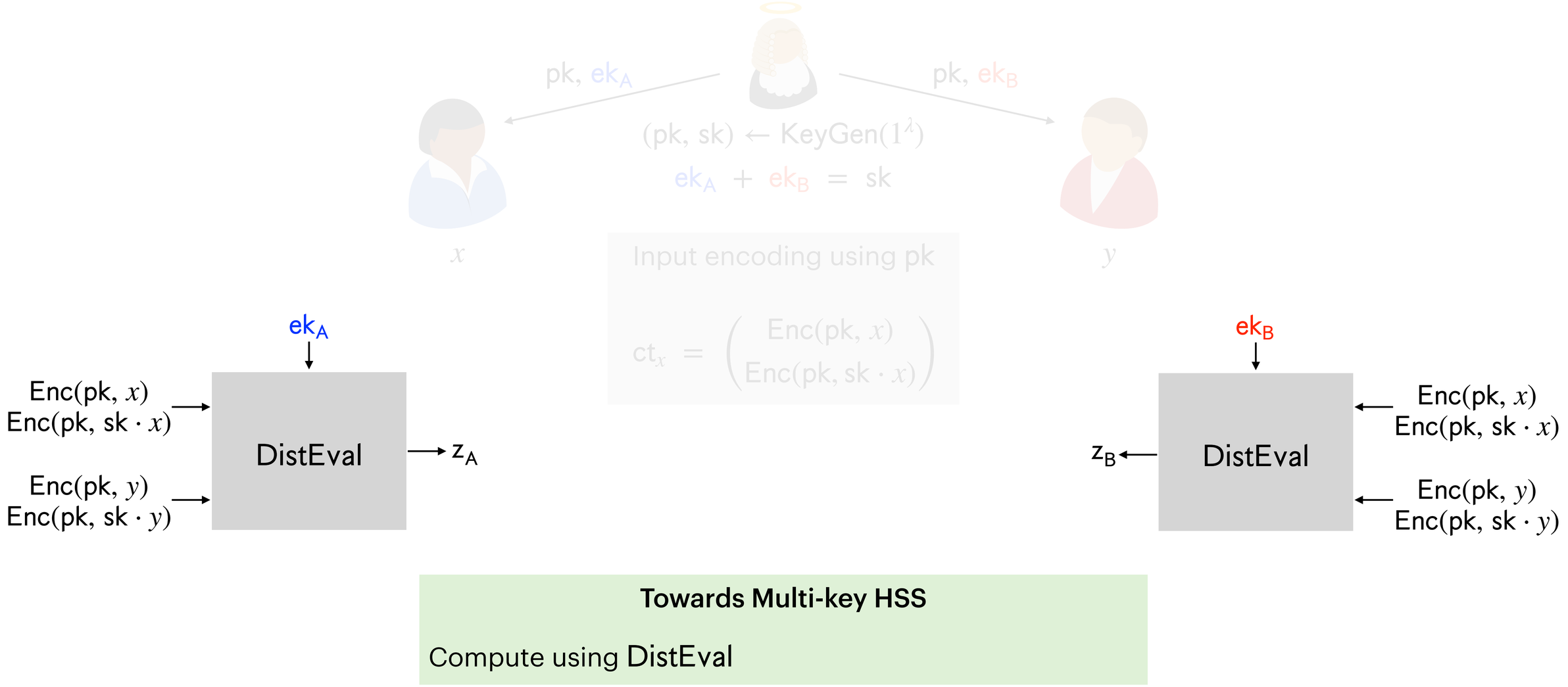


$$z_A + z_B = C(x, y)$$

Any NC^1 circuit

Group-based HSS Schemes

[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]



Towards Multi-key HSS

Compute using DistEval

Group-based HSS Schemes

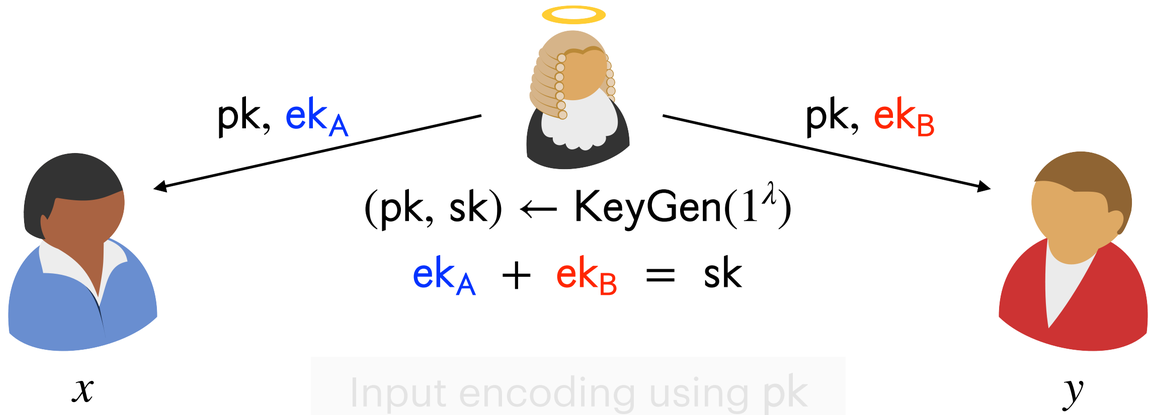
[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]

Towards Multi-key HSS

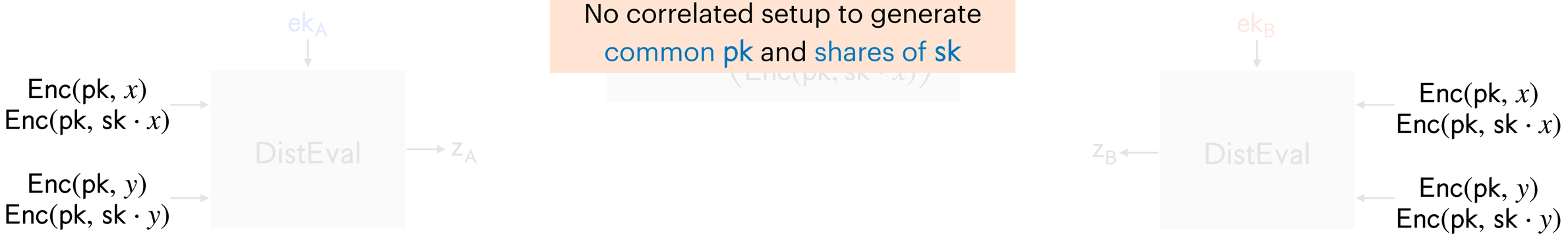
Compute using DistEval $\Longrightarrow$ compatible input encoding

Group-based HSS Schemes

[Boyle-Gilboa-Ishai'16][Orlandi-Scholl-Yakoubov'21][Roy-Singh'21][Abram-Damgård-Orlandi-Scholl'22]



No correlated setup to generate common pk and shares of sk



Towards Multi-key HSS

Compute using DistEval $\Longrightarrow$ compatible input encoding

Constructing Multi-Key HSS

Common Reference String

x

y

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

x

y

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

x

y

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

x

$\mathsf{ct}_x, \mathsf{pk}_A$

$\mathsf{ct}_y, \mathsf{pk}_B$

y

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

x

$\mathsf{ct}_x, \mathsf{pk}_A$ →

← $\mathsf{ct}_y, \mathsf{pk}_B$

y

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_y, \mathsf{pk}_B$

$\mathsf{ct}_x, \mathsf{pk}_A$

Cannot be used with DistEval since inputs are encrypted under different keys

$\mathsf{Enc}(\mathsf{pk}_A, x)$ →
$\mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x)$

$\mathsf{Enc}(\mathsf{pk}_B, y)$ →
$\mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y)$

DistEval

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

x

$\mathsf{ct}_x, \mathsf{pk}_A$

$\mathsf{ct}_y, \mathsf{pk}_B$

y

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Key Ingredient: Synchronize ciphertexts under different keys to a common public key pk

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

$\mathsf{ct}_y, \mathsf{pk}_B$

x

$\mathsf{ct}_x, \mathsf{pk}_A$ →

← $\mathsf{ct}_y, \mathsf{pk}_B$

y

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Key Ingredient: Synchronize ciphertexts under different keys to a common public key pk

↓

$\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

$\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

↓

$\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

$\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

$\mathsf{ct}_y, \mathsf{pk}_B$

x

$\mathsf{ct}_x, \mathsf{pk}_A$ →

← $\mathsf{ct}_y, \mathsf{pk}_B$

y

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Key Ingredient: Synchronize ciphertexts under different keys to a common public key pk

ek_A $\quad$ $\mathsf{ek}_A + \mathsf{ek}_B = \mathsf{sk}$ $\quad$ ek_B

$\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

$\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

$\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

$\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

$\mathsf{ct}_y, \mathsf{pk}_B$

x

$\mathsf{ct}_x, \mathsf{pk}_A$ →

← $\mathsf{ct}_y, \mathsf{pk}_B$

y

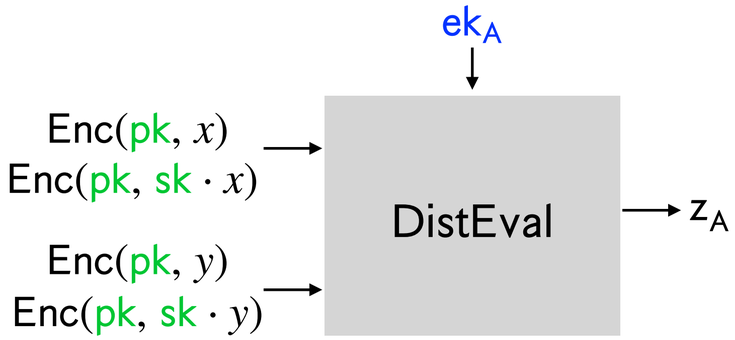
$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

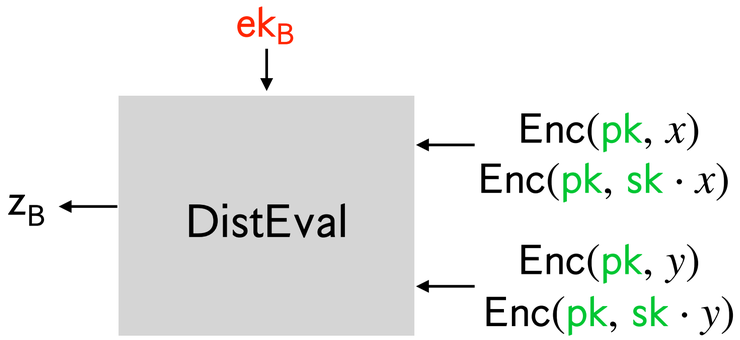
$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Key Ingredient: Synchronize ciphertexts under different keys to a common public key pk

$\mathsf{ek}_A + \mathsf{ek}_B = \mathsf{sk}$





Constructing Multi-Key HSS

Common Reference String

$(\mathsf{pk}_A, \mathsf{sk}_A) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$$\mathsf{ct}_x = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_A, x) \\ \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_A \cdot x) \end{pmatrix}$$

$\mathsf{ct}_y, \mathsf{pk}_B$

x

$\mathsf{ct}_x, \mathsf{pk}_A$ →

← $\mathsf{ct}_y, \mathsf{pk}_B$

y

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_B, \mathsf{sk}_B)$

$$\mathsf{ct}_y = \begin{pmatrix} \mathsf{Enc}(\mathsf{pk}_B, y) \\ \mathsf{Enc}(\mathsf{pk}_B, \mathsf{sk}_B \cdot y) \end{pmatrix}$$

$\mathsf{ct}_x, \mathsf{pk}_A$

Key Ingredient: Synchronize ciphertexts under different keys to a common public key pk

$\mathsf{ek}_A + \mathsf{ek}_B = \mathsf{sk}$

ek_A

$\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

$\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

DistEval → z_A

ek_B

z_B ← DistEval

← $\mathsf{Enc}(\mathsf{pk}, x)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x)$

← $\mathsf{Enc}(\mathsf{pk}, y)$
$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot y)$

Constructing Multi-Key HSS

Structure of Synchronized Key

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Constructing Multi-Key HSS

Structure of Synchronized Key

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Inspired by Multi-Key FHE
[Mukherjee-Wichs'16]

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

Constructing Multi-Key HSS

Structure of Synchronized Key

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Inspired by Multi-Key FHE
[Mukherjee-Wichs'16]

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$ $\quad$ $\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x) \in \mathbb{G}^{2\times 3} =$

Constructing Multi-Key HSS

Structure of Synchronized Key

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Inspired by Multi-Key FHE
[Mukherjee-Wichs'16]

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$ $\quad$ $\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

Decryption with sk

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x) \in \mathbb{G}^{2\times 3} = \begin{pmatrix} & & \\ & & \end{pmatrix} \begin{pmatrix} \mathsf{sk}_A \\ \mathsf{sk}_B \\ 1 \end{pmatrix} = \begin{pmatrix} g^{\mathsf{sk}_A \cdot x} \\ g^{\mathsf{sk}_B \cdot x} \end{pmatrix}$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x},\ \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x},\ g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x) \in \mathbb{G}^{2\times 3} = \begin{pmatrix} & & \\ & & \end{pmatrix} \begin{pmatrix} \mathsf{sk}_A \\ \mathsf{sk}_B \\ 1 \end{pmatrix} = \begin{pmatrix} g^{\mathsf{sk}_A \cdot x} \\ g^{\mathsf{sk}_B \cdot x} \end{pmatrix}$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

How to synchronize to an encryption of $\mathsf{sk}_B \cdot x$ under sk?

Decryption with sk

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x) \in \mathbb{G}^{2\times 3} = \begin{pmatrix} & & \\ & & \end{pmatrix} \begin{pmatrix} \mathsf{sk}_A \\ \mathsf{sk}_B \\ 1 \end{pmatrix} = \begin{pmatrix} g^{\mathsf{sk}_A \cdot x} \\ g^{\mathsf{sk}_B \cdot x} \end{pmatrix}$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x},\ \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x},\ g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk_B: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_B} = g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk_B: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_B} = g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk_B: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_B} = g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}, g$

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x},\ \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x},\ g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk_B: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_B} = g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$

Convert Enc(pk_A, r) into Enc(pk_A, $\mathsf{sk}_B \cdot r$)

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk · x)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

Flipped ElGamal Encryption:

[Abram-Damgård-Orlandi-Scholl'22]

$$\mathsf{FlipEG}(\mathsf{pk}_A, x) = \left(g^{-r} \cdot g^{x}, \mathsf{pk}_A^{r}\right) = \left(g^{-r} \cdot g^{x}, g^{\mathsf{sk}_A \cdot r}\right)$$

Decryption with sk_A: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_A} \cdot g^{\mathsf{sk}_A \cdot r} = g^{\mathsf{sk}_A \cdot x}$

Decryption with sk_B: $\left(g^{-r} \cdot g^{x}\right)^{\mathsf{sk}_B} = g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$

From key synchronization to homomorphism

Convert $\mathsf{Enc}(\mathsf{pk}_A, r)$ into $\mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk $\cdot\, x$)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

$\mathsf{FlipEG}(\mathsf{pk}_A, x; r), \quad \mathsf{EG}(\mathsf{pk}_A, r) = \left(g^r \cdot g^x, g^{-\mathsf{sk}_A \cdot r}\right), \quad \left(g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r\right)$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk · x)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

$\mathsf{FlipEG}(\mathsf{pk}_A, x; r), \quad \mathsf{EG}(\mathsf{pk}_A, r) = (g^r \cdot g^x, g^{-\mathsf{sk}_A \cdot r}), \quad (g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r)$

Convert $\mathsf{EG}(\mathsf{pk}_A, r)$**:**

Raise each component to sk_B

$$(g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r)^{\mathsf{sk}_B}$$

$$= (g^{u \cdot \mathsf{sk}_B}, g^{-\mathsf{sk}_A \cdot u \cdot \mathsf{sk}_B} \cdot g^{\mathsf{sk}_B \cdot r})$$

$$= (g^{u'}, g^{-\mathsf{sk}_A \cdot u'} \cdot g^{\mathsf{sk}_B \cdot r})$$

$$= \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk · x)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

$\mathsf{FlipEG}(\mathsf{pk}_A, x; r), \quad \mathsf{EG}(\mathsf{pk}_A, r) = (g^r \cdot g^x, g^{-\mathsf{sk}_A \cdot r}), \quad (g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r)$

Convert EG(pk_A, r):

Re-encrypt r using pk_B

$$\left(\mathsf{pk}_B^u, \mathsf{pk}_B^{-\mathsf{sk}_A \cdot u} \cdot \mathsf{pk}_B^r\right)$$

$$= \left(g^{u \cdot \mathsf{sk}_B}, g^{-\mathsf{sk}_A \cdot u \cdot \mathsf{sk}_B} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \left(g^{u'}, g^{-\mathsf{sk}_A \cdot u'} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$$

Convert EG(pk_A, r):

Raise each component to sk_B

$$\left(g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r\right)^{\mathsf{sk}_B}$$

$$= \left(g^{u \cdot \mathsf{sk}_B}, g^{-\mathsf{sk}_A \cdot u \cdot \mathsf{sk}_B} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \left(g^{u'}, g^{-\mathsf{sk}_A \cdot u'} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk · x)

Common Reference String: $\mathbb{G}$, g

$\mathsf{sk}_A \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_A = g^{\mathsf{sk}_A}$

x

$\mathsf{sk} = (\mathsf{sk}_A, \mathsf{sk}_B)$

$\mathsf{pk} = (g^{\mathsf{sk}_A}, g^{\mathsf{sk}_B})$

$\mathsf{sk}_B \leftarrow \mathbb{Z}_p$

$\mathsf{pk}_B = g^{\mathsf{sk}_B}$

$\mathsf{FlipEG}(\mathsf{pk}_A, x; r), \quad \mathsf{EG}(\mathsf{pk}_A, r) = (g^r \cdot g^x, g^{-\mathsf{sk}_A \cdot r}), \quad (g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r)$

Convert EG(pk_A, r):

Re-encrypt r using pk_B

$$\left(\mathsf{pk}_B^u, \mathsf{pk}_B^{-\mathsf{sk}_A \cdot u} \cdot \mathsf{pk}_B^r\right)$$

$$= \left(g^{u \cdot \mathsf{sk}_B}, g^{-\mathsf{sk}_A \cdot u \cdot \mathsf{sk}_B} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \left(g^{u'}, g^{-\mathsf{sk}_A \cdot u'} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$= \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$

$$\begin{pmatrix} \mathsf{sk}_A \\ \mathsf{sk}_B \\ 1 \end{pmatrix}$$

Convert EG(pk_A, r):

Raise each component to sk_B

$$\left(g^u, g^{-\mathsf{sk}_A \cdot u} \cdot g^r\right)^{\mathsf{sk}_B}$$

$$= \left(g^{u \cdot \mathsf{sk}_B}, g^{-\mathsf{sk}_A \cdot u \cdot \mathsf{sk}_B} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$$= \left(g^{u'}, g^{-\mathsf{sk}_A \cdot u'} \cdot g^{\mathsf{sk}_B \cdot r}\right)$$

$= \mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r)$

Constructing Multi-Key HSS

Synchronizing to Enc(pk, sk · x)

Decryption with sk

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{sk} \cdot x) \in \mathbb{G}^{2\times 3} = \begin{pmatrix} \mathsf{sk}_A & \mathsf{sk}_B & 1 \end{pmatrix} = \begin{pmatrix} g^{\mathsf{sk}_A \cdot x} \\ g^{\mathsf{sk}_B \cdot x} \end{pmatrix}$$

$$\left(\mathsf{Enc}(\mathsf{pk}_A, \mathsf{sk}_B \cdot r),\ \mathsf{FlipEG}(\mathsf{pk}_A, x; r)\right) \begin{pmatrix} \mathsf{sk}_A \\ \mathsf{sk}_B \\ 1 \end{pmatrix}$$

$$= g^{\mathsf{sk}_B \cdot r} \cdot g^{\mathsf{sk}_B \cdot x} \cdot g^{-\mathsf{sk}_B \cdot r}$$

$$= g^{\mathsf{sk}_B \cdot x}$$

Conclusion

Conclusion

- Discussed approach works for Multi-Key HSS from class groups

Conclusion

- Discussed approach works for Multi-Key HSS from class groups
- **DDH over prime-order groups:** Extension to synchronize to encryption of "small values"

Synchronized ciphertexts: $\mathsf{Enc}(\mathsf{pk}, \mathsf{s}_0 \cdot x) \quad \ldots \quad \mathsf{Enc}(\mathsf{pk}, \mathsf{s}_\lambda \cdot x)$

$$\mathsf{sk} = \sum_{i=0}^{\lambda} 2^i \cdot s_i$$

Conclusion

- Discussed approach works for Multi-Key HSS from class groups
- **DDH over prime-order groups:** Extension to synchronize to encryption of "small values"

Synchronized ciphertexts: $\mathsf{Enc}(\mathsf{pk}, s_0 \cdot x) \quad \ldots \quad \mathsf{Enc}(\mathsf{pk}, s_\lambda \cdot x)$

$$\mathsf{sk} = \sum_{i=0}^{\lambda} 2^i \cdot s_i$$

- **DCR:** Requires a different synchronization approach

$$\mathsf{pk}_A = g^{\mathsf{sk}_A}$$
$$\mathsf{pk}_B = g^{\mathsf{sk}_B}$$
$$\longrightarrow \quad \mathsf{pk} = g^{\mathsf{sk}_A \cdot \mathsf{sk}_B}$$

Thank You

eprint.iacr.org/2025/094