Client-Server Homomorphic Secret Sharing in the CRS Model

NTT CIS Seminar

Damiano Abram

Geoffroy Couteau

Lalita Devadas

Aditya Hegde

Abhishek Jain

Lawrence Roy

Sacha Servan-Schreiber

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

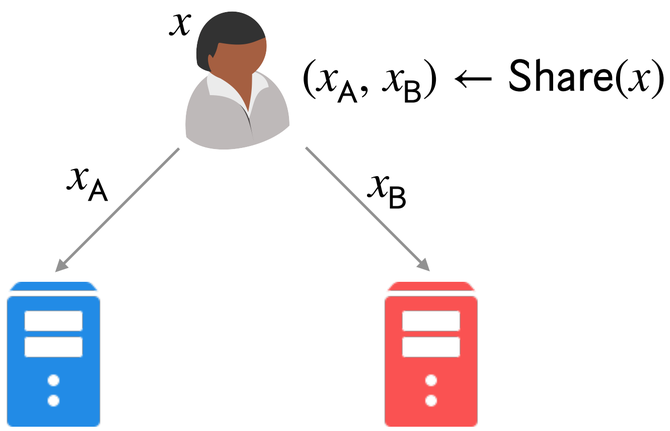
Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

$(x_A, x_B) \leftarrow \mathsf{Share}(x)$

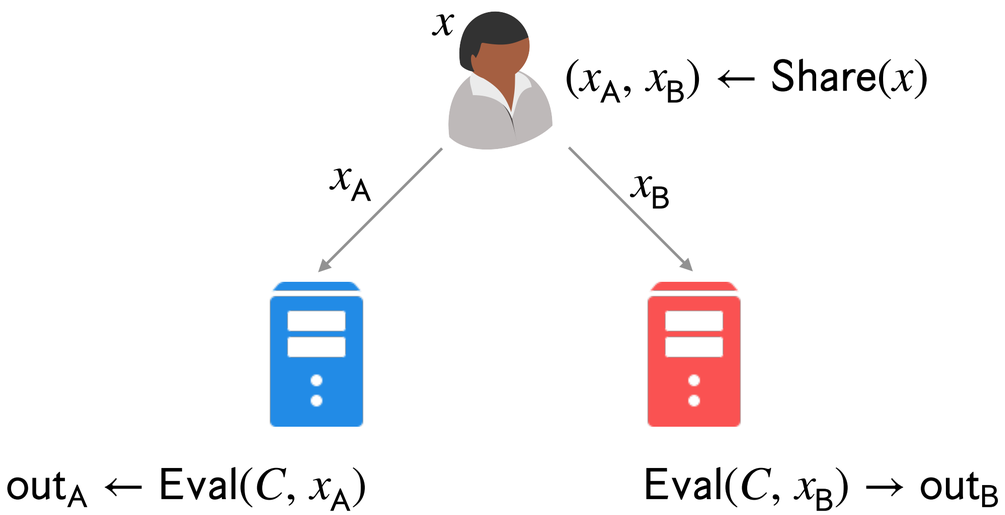
Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



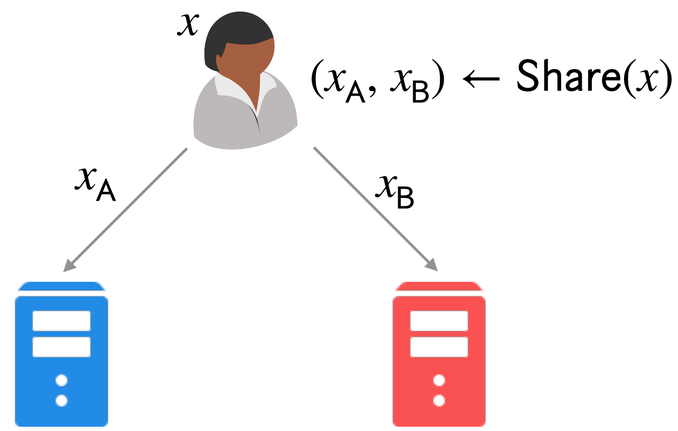
Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

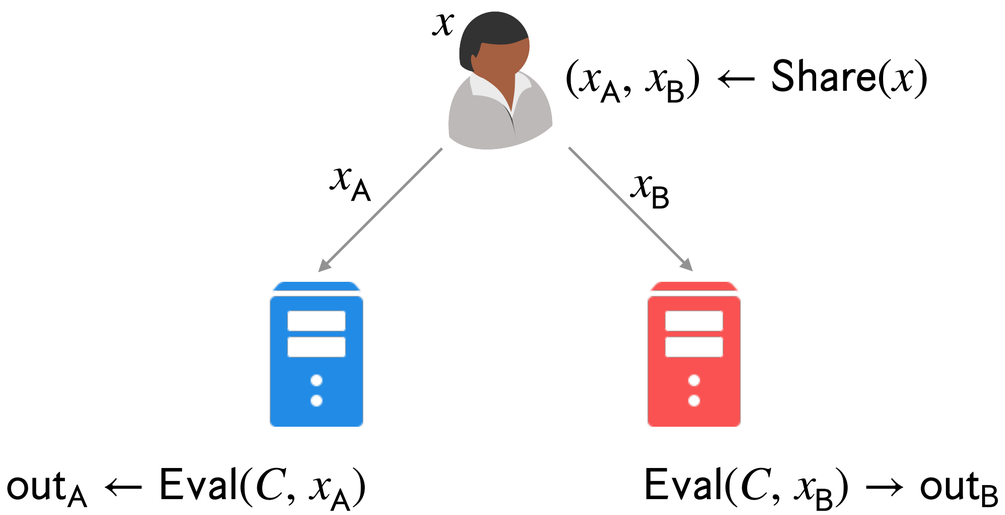


$\text{out}_A \leftarrow \text{Eval}(C, x_A)$ $\quad\quad$ $\text{Eval}(C, x_B) \rightarrow \text{out}_B$

Correctness: $\quad \text{out}_A + \text{out}_B = C(x)$

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



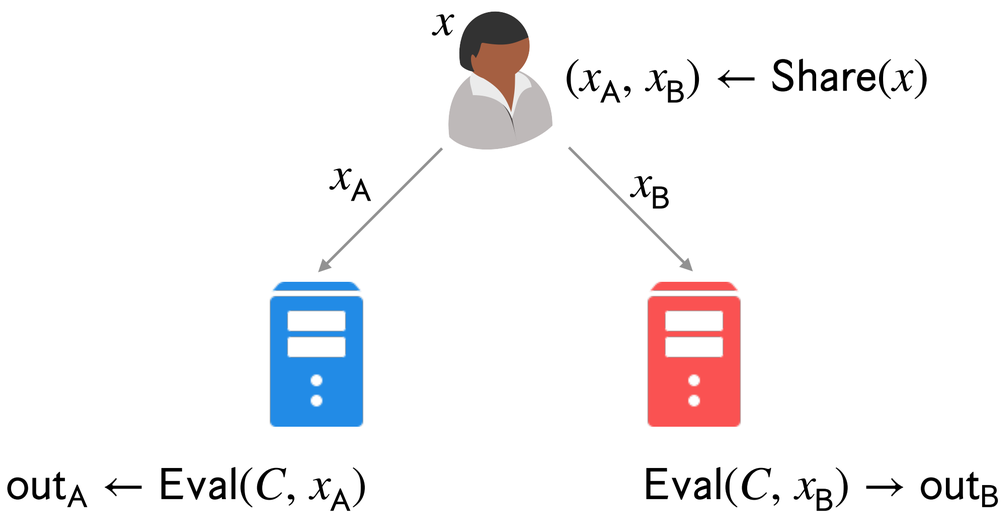
Correctness: $\text{out}_A + \text{out}_B = C(x)$

Security: x_A ensures privacy of x

x_B ensures privacy of x

Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]



Correctness: $\text{out}_A + \text{out}_B = C(x)$

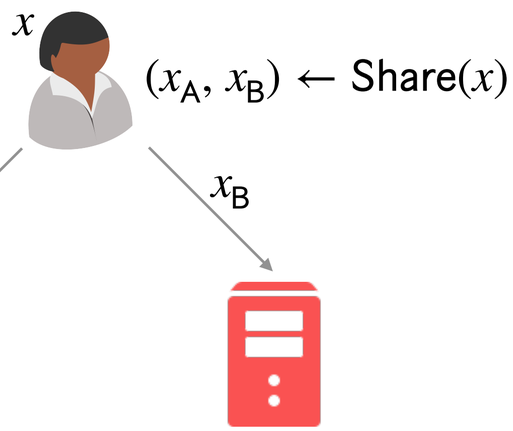
Security: x_A ensures privacy of x
x_B ensures privacy of x

Succinctness: Size of x_A and x_B are independent of C

Homomorphic Secret Sharing (HSS)

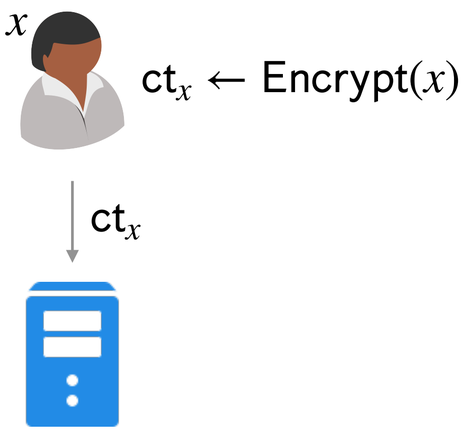
[Boyle-Gilboa-Ishai'16]

HSS



$\text{out}_A \leftarrow \text{Eval}(C, x_A)$ $\quad$ $\text{Eval}(C, x_B) \rightarrow \text{out}_B$

FHE



$\text{ct}_{C(x)} \leftarrow \text{Eval}(C, \text{ct}_x)$

Correctness: $\text{out}_A + \text{out}_B = C(x)$

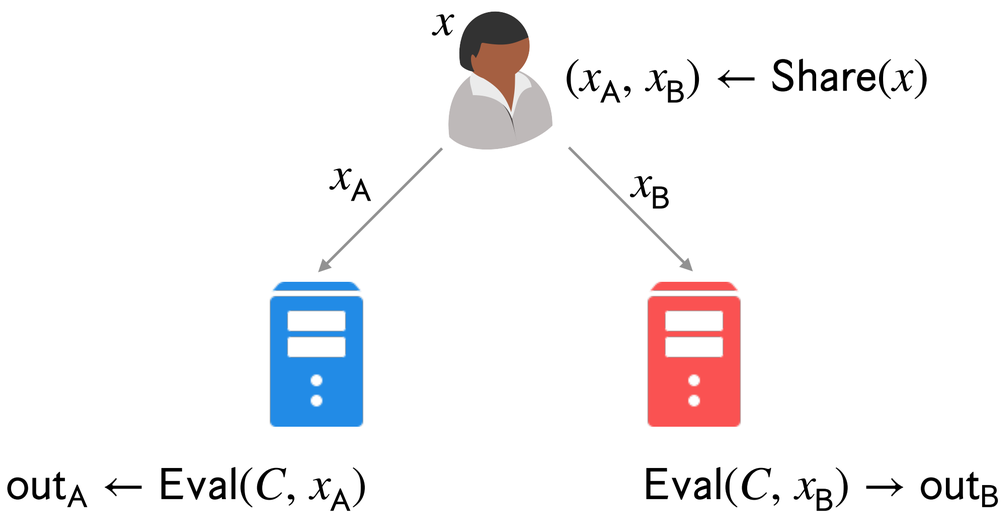
Security: x_A ensures privacy of x
x_B ensures privacy of x

Succinctness: Size of x_A and x_B are independent of C

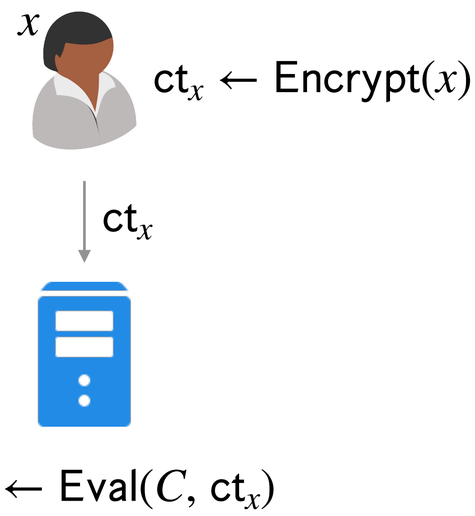
Homomorphic Secret Sharing (HSS)

[Boyle-Gilboa-Ishai'16]

HSS



FHE



Correctness: $\text{out}_A + \text{out}_B = C(x)$

Security: x_A ensures privacy of x
x_B ensures privacy of x

Succinctness: Size of x_A and x_B are independent of C

HSS is known from assumptions not known to imply FHE

Client-Server HSS

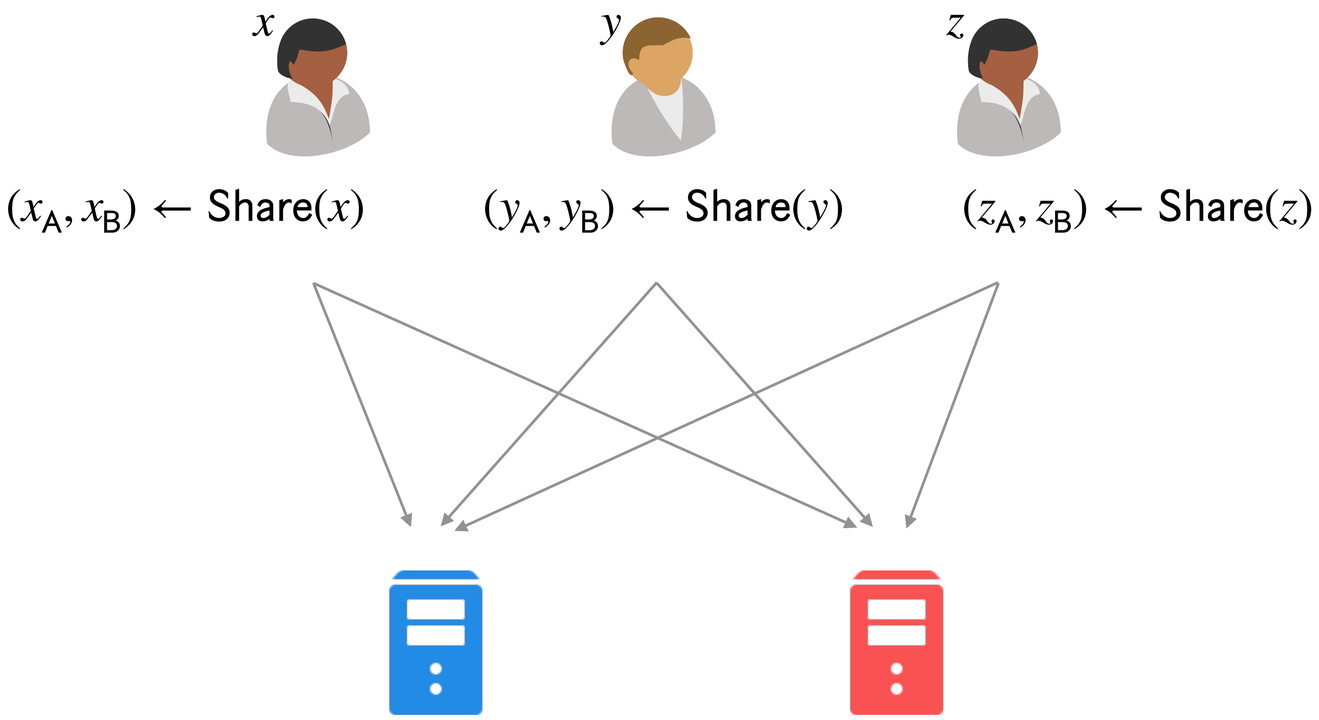
Client-Server HSS

$(x_A, x_B) \leftarrow \text{Share}(x)$

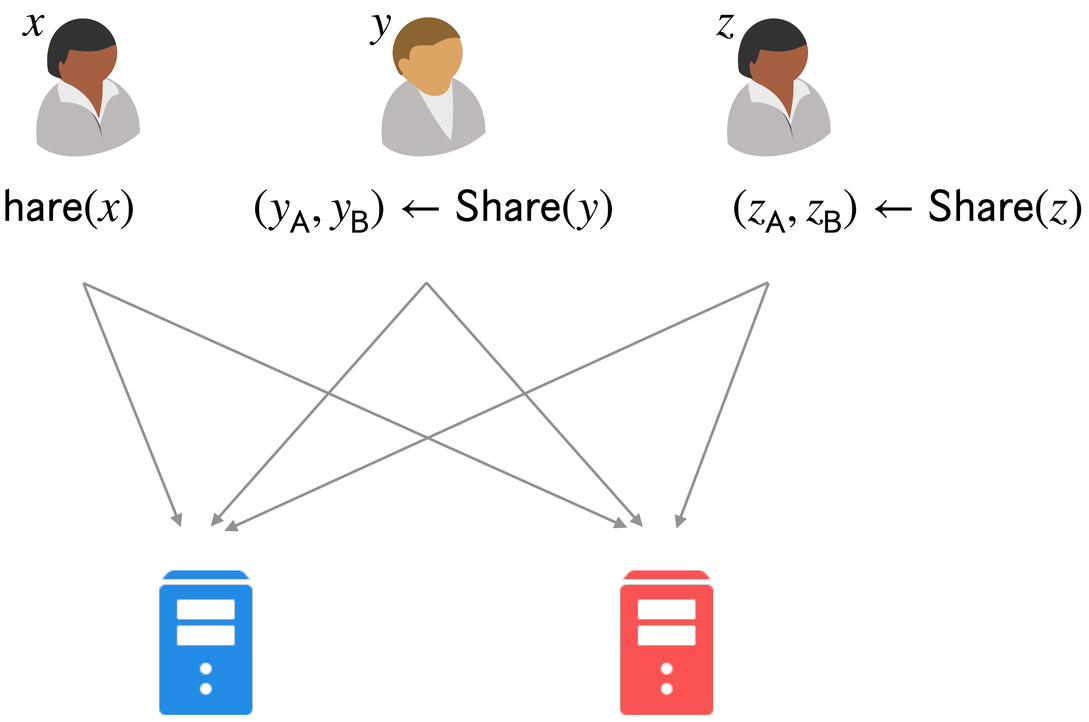
$(y_A, y_B) \leftarrow \text{Share}(y)$

$(z_A, z_B) \leftarrow \text{Share}(z)$

Client-Server HSS

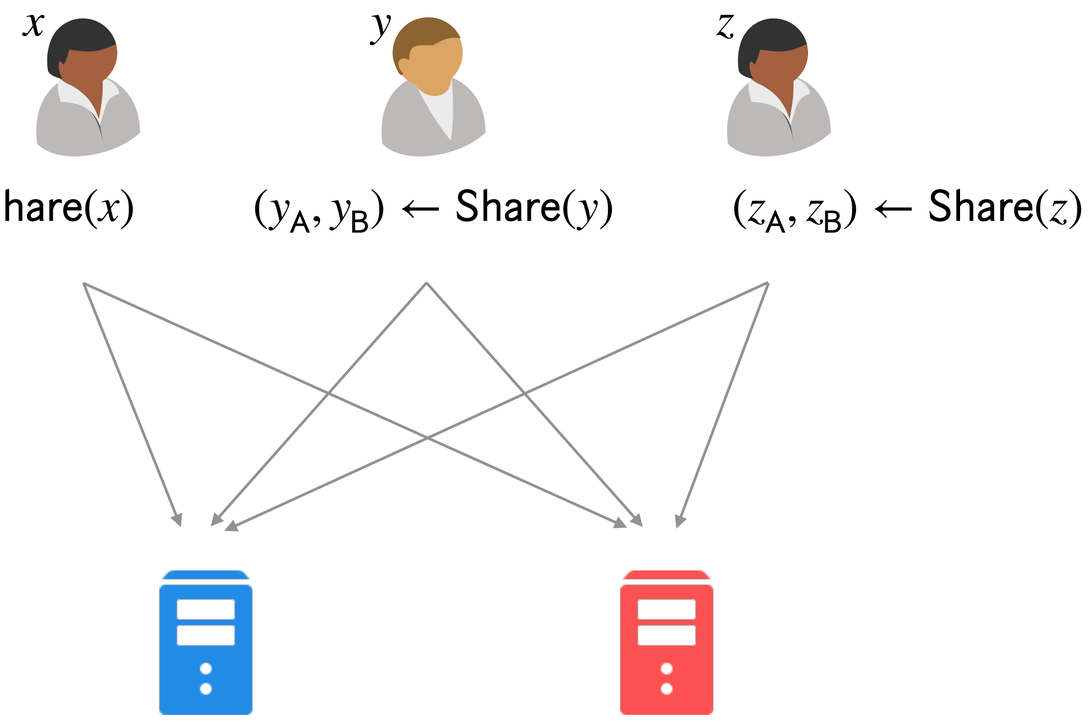


Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

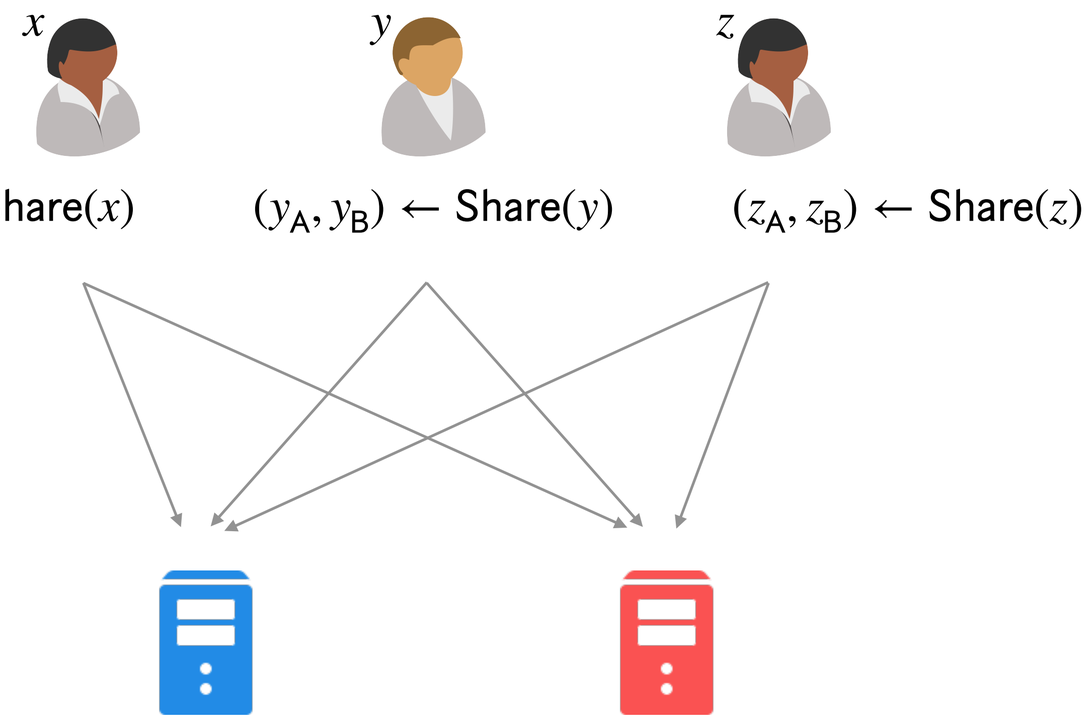
Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

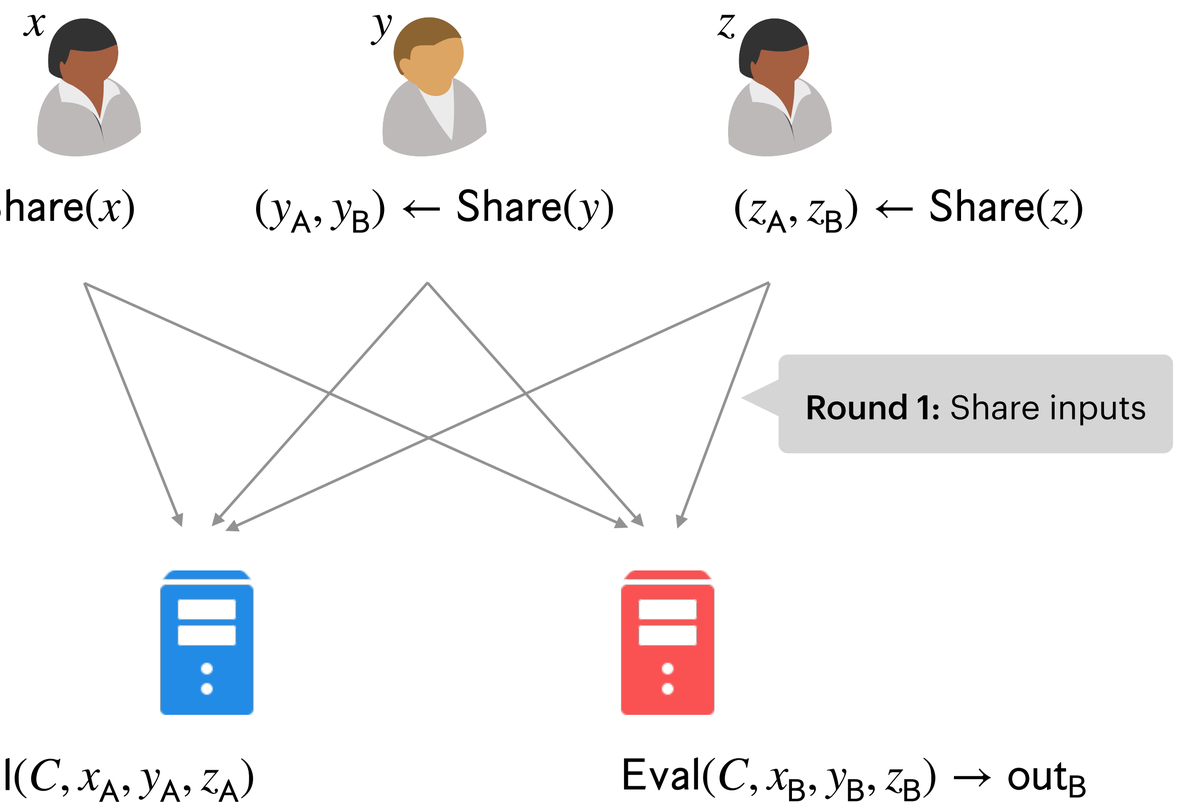
Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

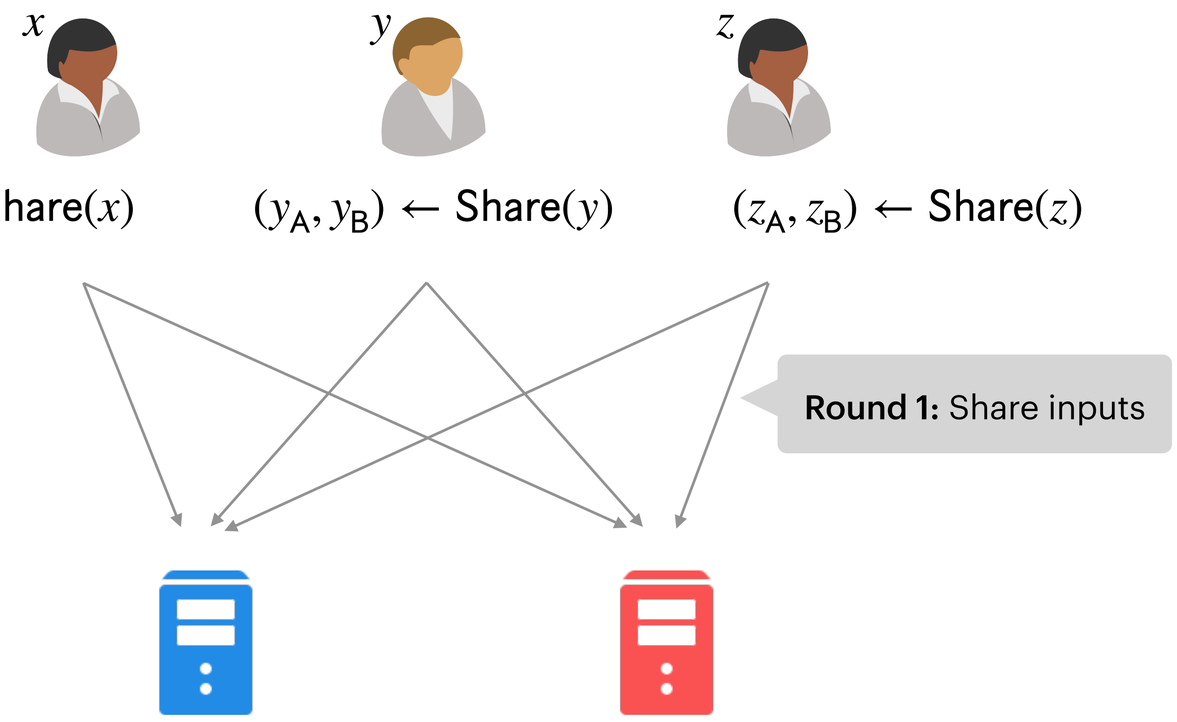
Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Client-Server HSS



$\text{out}_A + \text{out}_B = C(x, y, z)$ **Round 2:** Reconstruct output

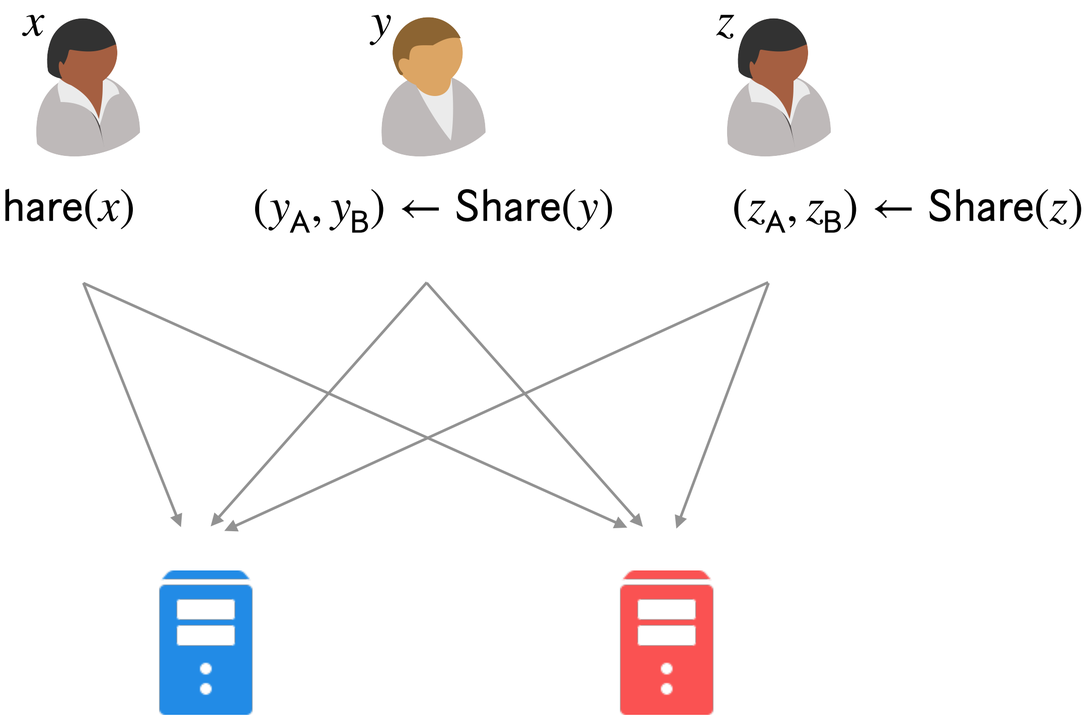
Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

Applications

Two-round succinct MPC

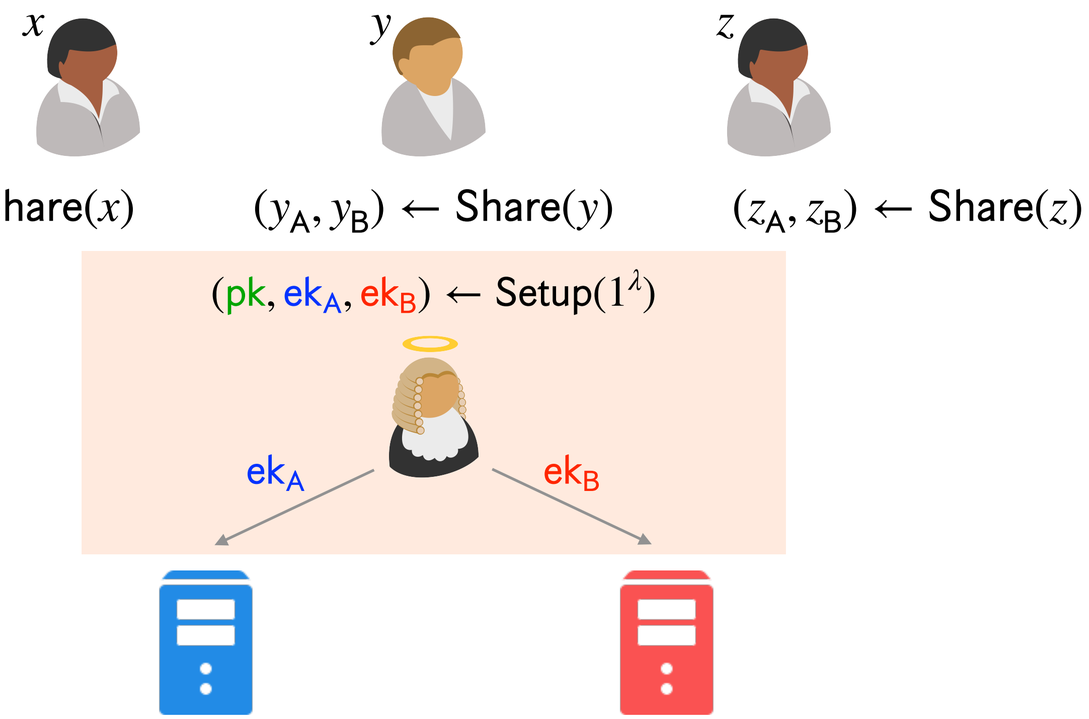
Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS
require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19]
[Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21]
[Abram-Damgård-Orlandi-Scholl'22]

Client-Server HSS



$$\text{out}_A + \text{out}_B = C(x, y, z)$$

Applications

Two-round succinct MPC

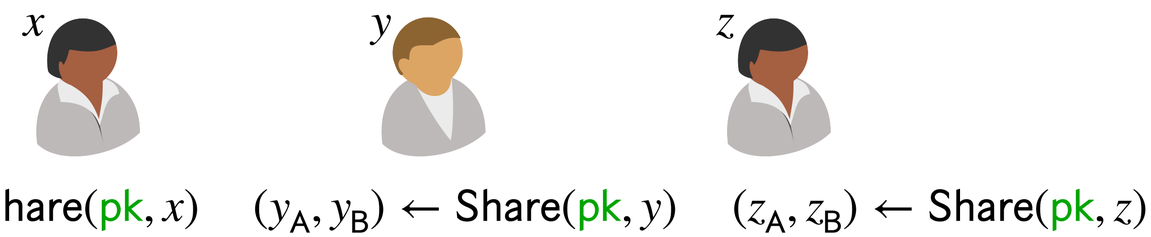
Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS
require correlated setup

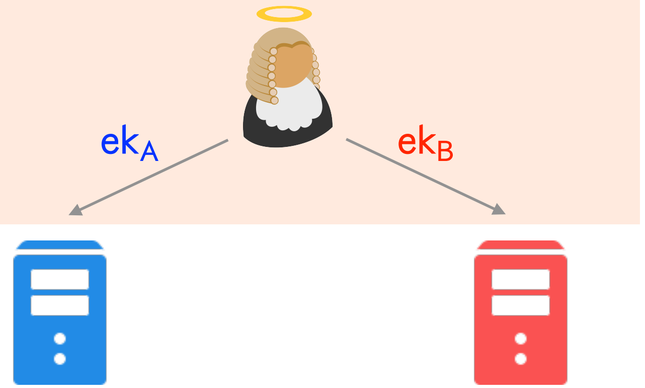
[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19]
[Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21]
[Abram-Damgård-Orlandi-Scholl'22]

Client-Server HSS



$(x_A, x_B) \leftarrow \text{Share}(\text{pk}, x)$ $(y_A, y_B) \leftarrow \text{Share}(\text{pk}, y)$ $(z_A, z_B) \leftarrow \text{Share}(\text{pk}, z)$

$(\text{pk}, \text{ek}_A, \text{ek}_B) \leftarrow \text{Setup}(1^\lambda)$



$\text{out}_A \leftarrow \text{Eval}(C, x_A, y_A, z_A)$ $\text{Eval}(C, x_B, y_B, z_B) \rightarrow \text{out}_B$

$\text{out}_A + \text{out}_B = C(x, y, z)$

Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS
require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19]
[Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21]
[Abram-Damgård-Orlandi-Scholl'22]

Client-Server HSS

$(x_A, x_B) \leftarrow \mathsf{Share}(\mathsf{pk}, x)$ $(y_A, y_B) \leftarrow \mathsf{Share}(\mathsf{pk}, y)$ $(z_A, z_B) \leftarrow \mathsf{Share}(\mathsf{pk}, z)$

$(\mathsf{pk}, \mathsf{ek}_A, \mathsf{ek}_B) \leftarrow \mathsf{Setup}(1^\lambda)$

ek_A ek_B

$\mathsf{out}_A \leftarrow \mathsf{Eval}(\mathsf{ek}_A, C, x_A, y_A, z_A)$ $\mathsf{Eval}(\mathsf{ek}_B, C, x_B, y_B, z_B) \rightarrow \mathsf{out}_B$

$\mathsf{out}_A + \mathsf{out}_B = C(x, y, z)$

Applications

Two-round succinct MPC

Private Information Retrieval

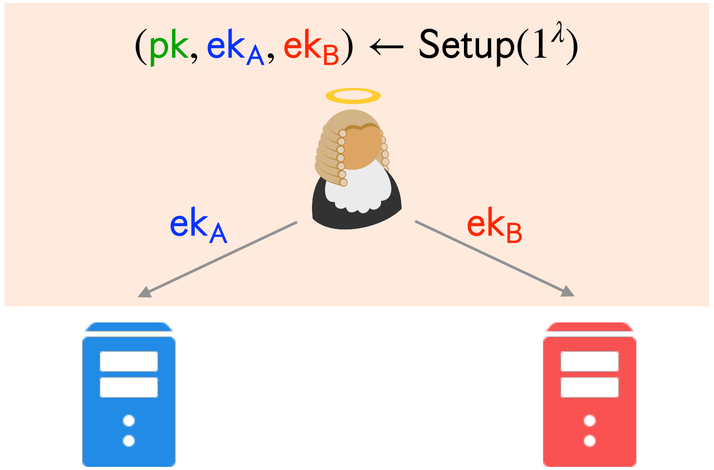
Pseudorandom Correlation Generators

Existing client-server HSS
require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19]
[Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21]
[Abram-Damgård-Orlandi-Scholl'22]

Client-Server HSS

$(x_A, x_B) \leftarrow \text{Share}(\text{pk}, x)$ $(y_A, y_B) \leftarrow \text{Share}(\text{pk}, y)$ $(z_A, z_B) \leftarrow \text{Share}(\text{pk}, z)$



$\text{out}_A \leftarrow \text{Eval}(\text{ek}_A, C, x_A, y_A, z_A)$ $\text{Eval}(\text{ek}_B, C, x_B, y_B, z_B) \rightarrow \text{out}_B$

$\text{out}_A + \text{out}_B = C(x, y, z)$

Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19] [Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21] [Abram-Damgård-Orlandi-Scholl'22]

Multi-key FHE: Multi-input evaluation in the CRS model

[López-Alt–Tromer–Vaikuntanathan'12] [Wichs-Mukherjee'16]

Client-Server HSS

$(x_A, x_B) \leftarrow \text{Share}(\text{crs}, x)$ $(y_A, y_B) \leftarrow \text{Share}(\text{crs}, y)$ $(z_A, z_B) \leftarrow \text{Share}(\text{crs}, z)$

Common Reference String

$\text{out}_A \leftarrow \text{Eval}(\text{crs}, C, x_A, y_A, z_A)$

$\text{Eval}(\text{crs}, C, x_B, y_B, z_B) \rightarrow \text{out}_B$

$\text{out}_A + \text{out}_B = C(x, y, z)$

Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS
require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19]
[Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21]
[Abram-Damgård-Orlandi-Scholl'22]

Multi-key FHE: Multi-input evaluation
in the CRS model

[López-Alt–Tromer–Vaikuntanathan'12]
[Wichs-Mukherjee'16]

Client-Server HSS

$(x_A, x_B) \leftarrow \text{Share}(\text{crs}, x)$ $(y_A, y_B) \leftarrow \text{Share}(\text{crs}, y)$ $(z_A, z_B) \leftarrow \text{Share}(\text{crs}, z)$

Common Reference String

$\text{out}_A \leftarrow \text{Eval}(\text{crs}, C, x_A, y_A, z_A)$

$\text{Eval}(\text{crs}, C, x_B, y_B, z_B) \rightarrow \text{out}_B$

$$\text{out}_A + \text{out}_B = C(x, y, z)$$

Applications

Two-round succinct MPC

Private Information Retrieval

Pseudorandom Correlation Generators

Existing client-server HSS require correlated setup

[Boyle-Gilboa-Ishai'16] [Boyle-Kohl-Scholl'19] [Roy-Singh'21] [Orlandi-Scholl-Yakoubov'21] [Abram-Damgård-Orlandi-Scholl'22]

Multi-key FHE: Multi-input evaluation in the CRS model

[López-Alt–Tromer–Vaikuntanathan'12] [Wichs-Mukherjee'16]

Goal: Client-server HSS in the CRS model from assumptions not known to imply FHE

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

Our Results

Unbounded polynomial number of clients

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

Our Results

Contains NC^1

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

DDH

DCR

Class groups

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

DDH

DCR

Class groups

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

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

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

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

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

Inverse polynomial correctness error

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

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

Our Results

Multi-client two-server HSS in the CRS model for evaluating RMS Programs

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

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

Outline

Barriers to Removing Correlated Setup

Our Approach

Extensions

Outline

Barriers to Removing Correlated Setup

Our Approach

Extensions

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

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

Client-Server HSS with **Correlated Setup**

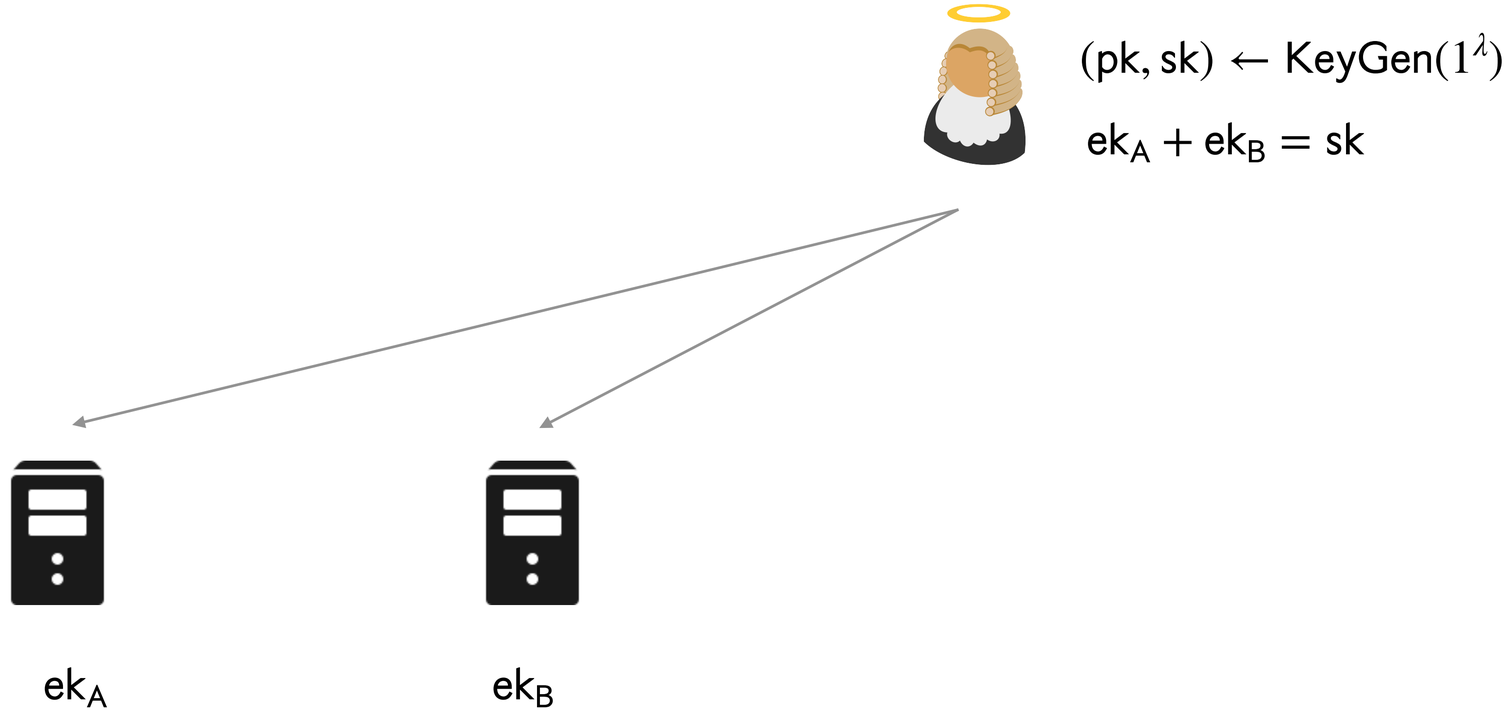
[Boyle-Gilboa-Ishai'16]

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

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

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]



Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

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

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

ek_A

ek_B

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

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

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

ek_A

ek_B

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

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

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

x

y

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

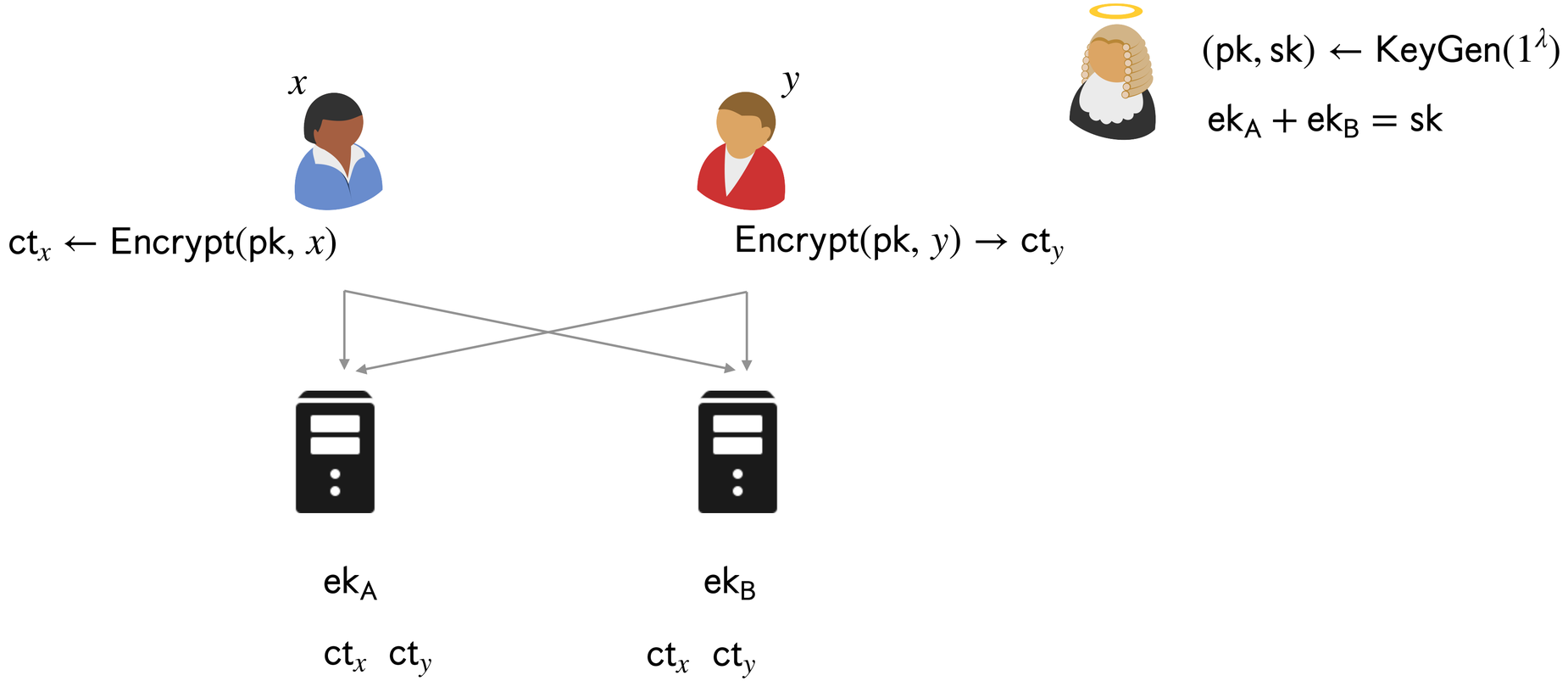
$\mathsf{Encrypt}(\mathsf{pk}, y) \rightarrow \mathsf{ct}_y$

ek_A

ek_B

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]



Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]

x

y

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

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

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

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

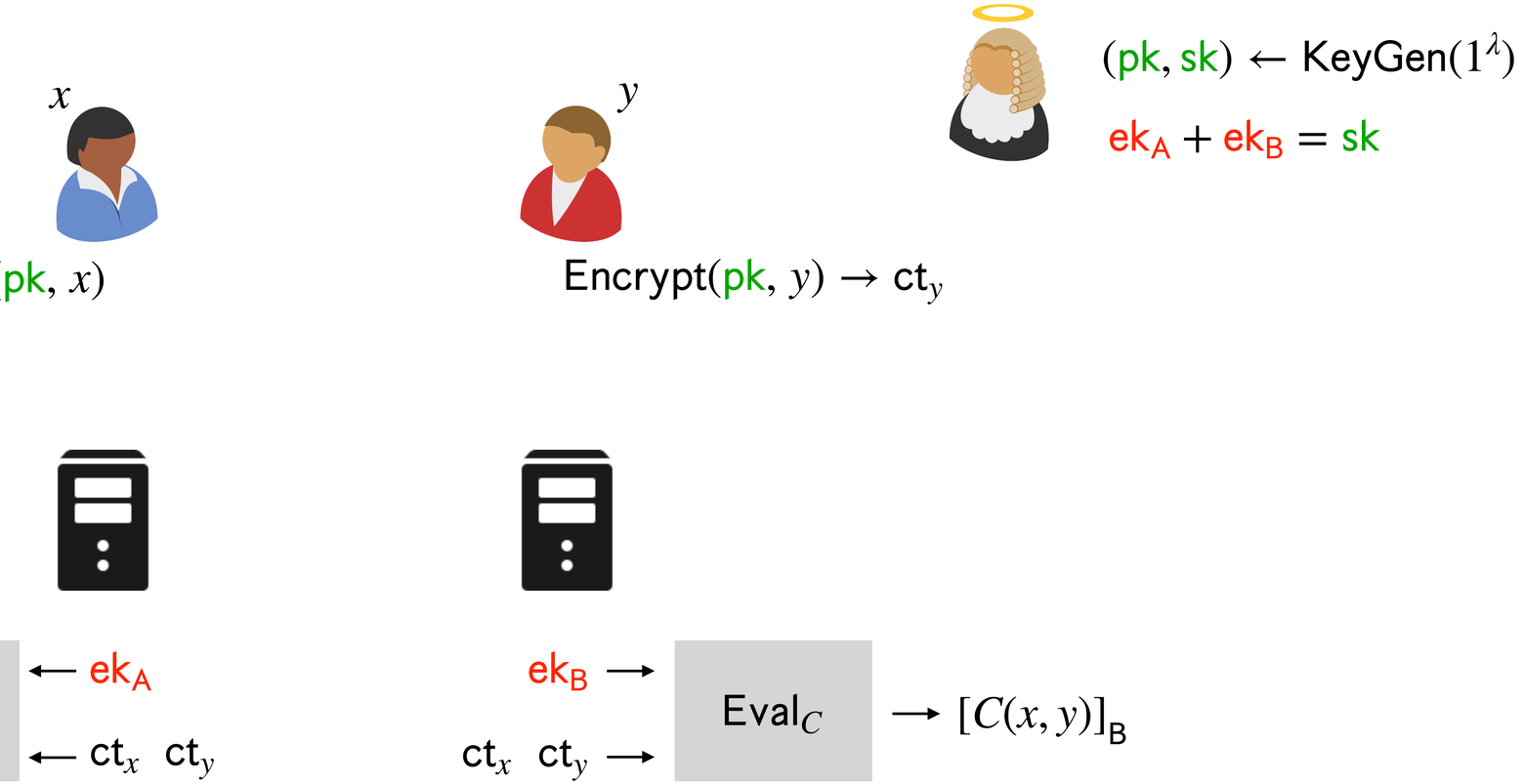
$[C(x,y)]_A \leftarrow \boxed{\mathsf{Eval}_C} \leftarrow \mathsf{ek}_A,\ \mathsf{ct}_x\ \mathsf{ct}_y$

Any RMS program

$\mathsf{ek}_B \rightarrow,\ \mathsf{ct}_x\ \mathsf{ct}_y \rightarrow \boxed{\mathsf{Eval}_C} \rightarrow [C(x,y)]_B$

Client-Server HSS with **Correlated Setup**

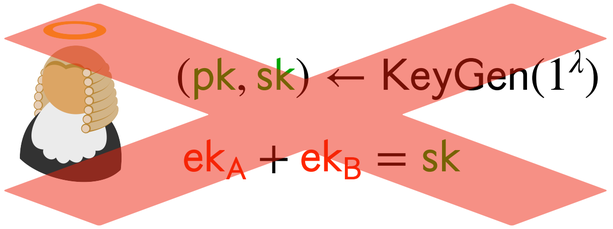
[Boyle-Gilboa-Ishai'16]



Barrier to Removing Correlated Setup: All inputs must be encrypted under a common key

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



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

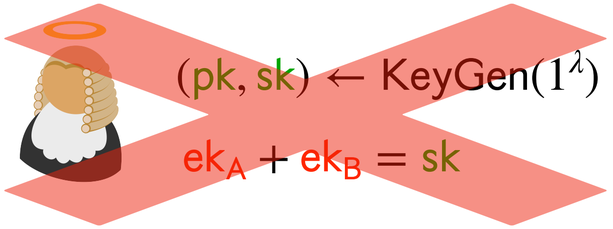
$\mathsf{Encrypt}(\mathsf{pk}, y) \rightarrow \mathsf{ct}_y$

$[C(x,y)]_A \leftarrow \boxed{\mathsf{Eval}_C} \leftarrow \mathsf{ek}_A,\ \mathsf{ct}_x\ \mathsf{ct}_y$

$\mathsf{ek}_B \rightarrow,\ \mathsf{ct}_x\ \mathsf{ct}_y \rightarrow \boxed{\mathsf{Eval}_C} \rightarrow [C(x,y)]_B$

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



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

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

Approach

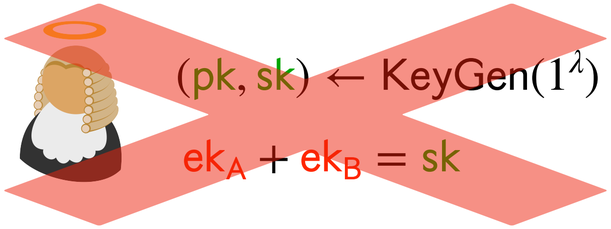
Modify input encoding to use the same evaluation algorithm

$[C(x,y)]_A \leftarrow \mathsf{Eval}_C \leftarrow \mathsf{ek}_A,\ \mathsf{ct}_x,\ \mathsf{ct}_y$

$\mathsf{ek}_B,\ \mathsf{ct}_x,\ \mathsf{ct}_y \rightarrow \mathsf{Eval}_C \rightarrow [C(x,y)]_B$

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

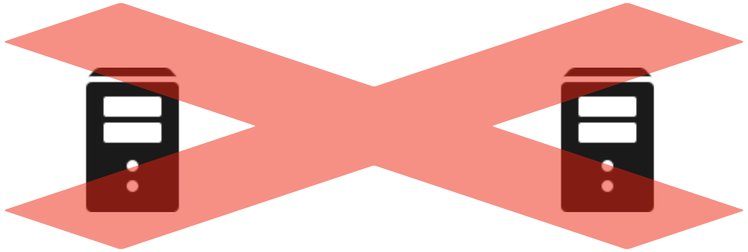


$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

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

Approach

Modify input encoding to use the same evaluation algorithm



$[C(x,y)]_A \leftarrow \mathsf{Eval}_C \leftarrow \mathsf{ek}_A$

$\leftarrow \mathsf{ct}_x \quad \mathsf{ct}_y$

$\mathsf{ek}_B \rightarrow \mathsf{Eval}_C \rightarrow [C(x,y)]_B$

$\mathsf{ct}_x \quad \mathsf{ct}_y \rightarrow$

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

Common Reference String

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

Common Reference String

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

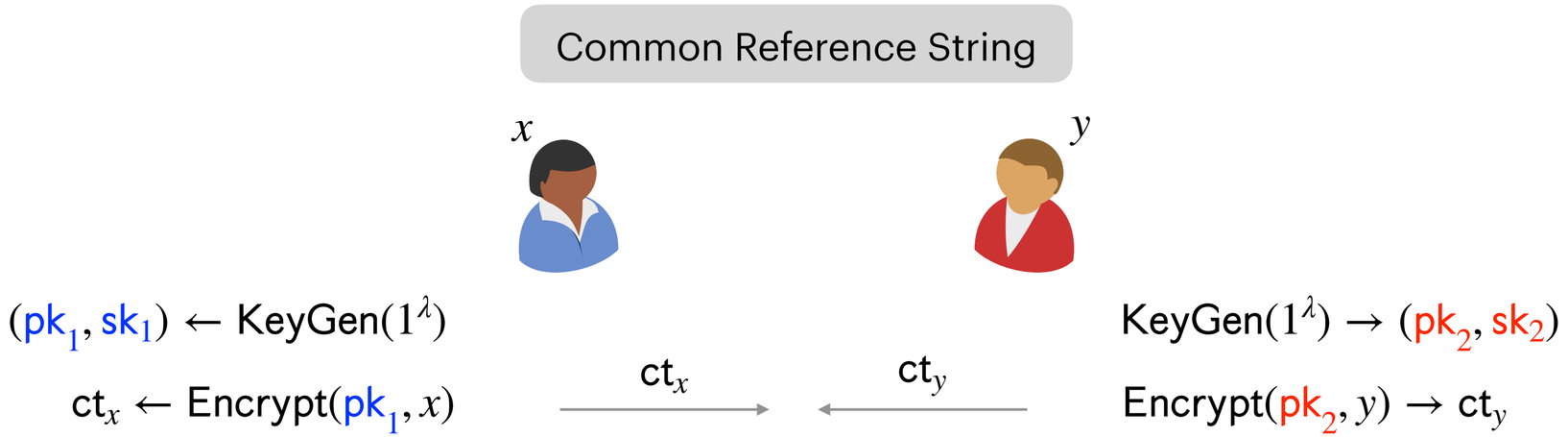
$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, x)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Encrypt}(\mathsf{pk}_2, y) \rightarrow \mathsf{ct}_y$

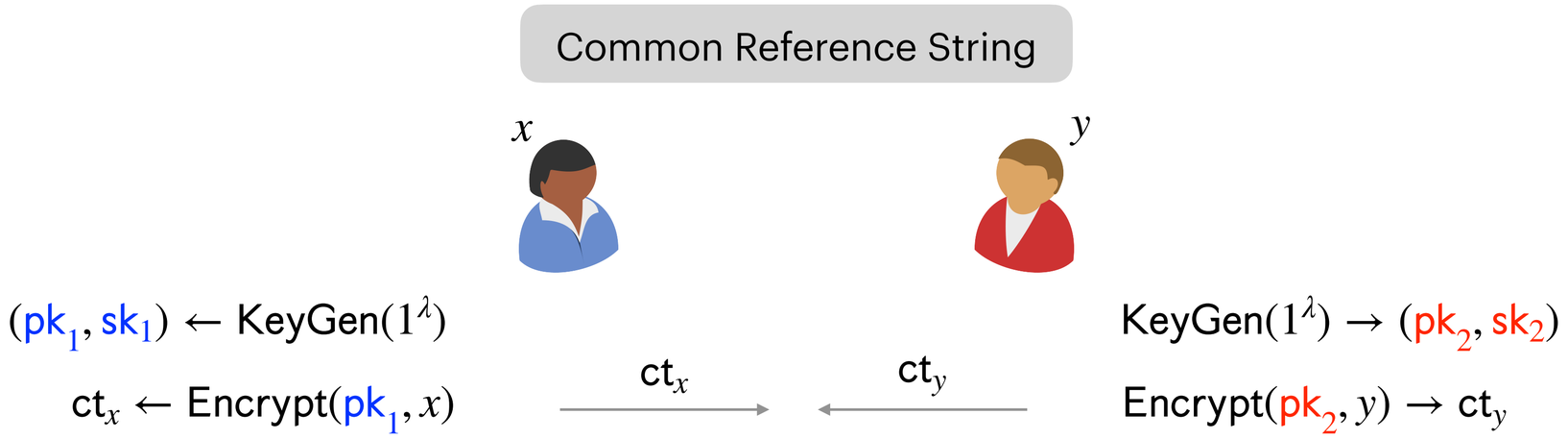
Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

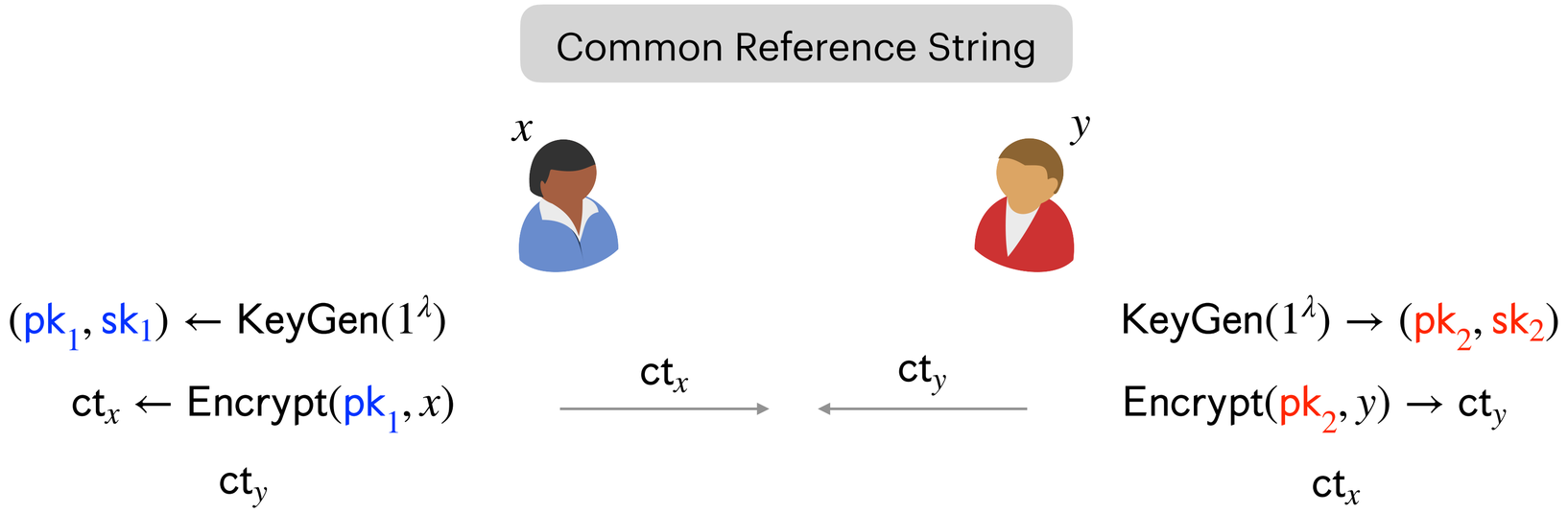


$\mathsf{Encrypt}(\mathsf{pk}_1, x)$
$\mathsf{Encrypt}(\mathsf{pk}_2, y)$
$\rightarrow$ Eval_C

Inputs encrypted under different keys

Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

Common Reference String

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{ct}_x \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, x)$

ct_y

ct_x →

← ct_y

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

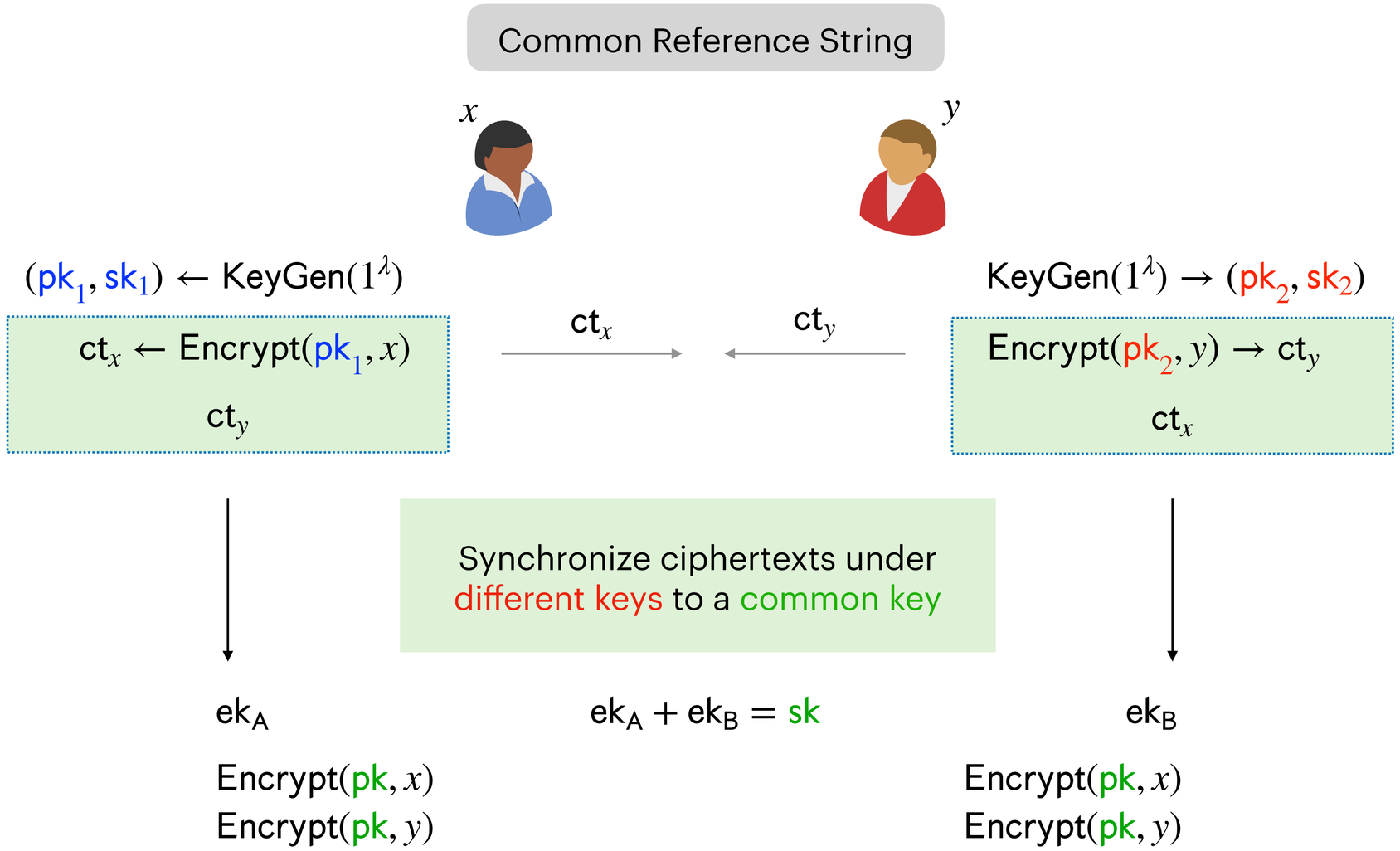
$\mathsf{Encrypt}(\mathsf{pk}_2, y) \rightarrow \mathsf{ct}_y$

ct_x

Synchronize ciphertexts under different keys to a common key

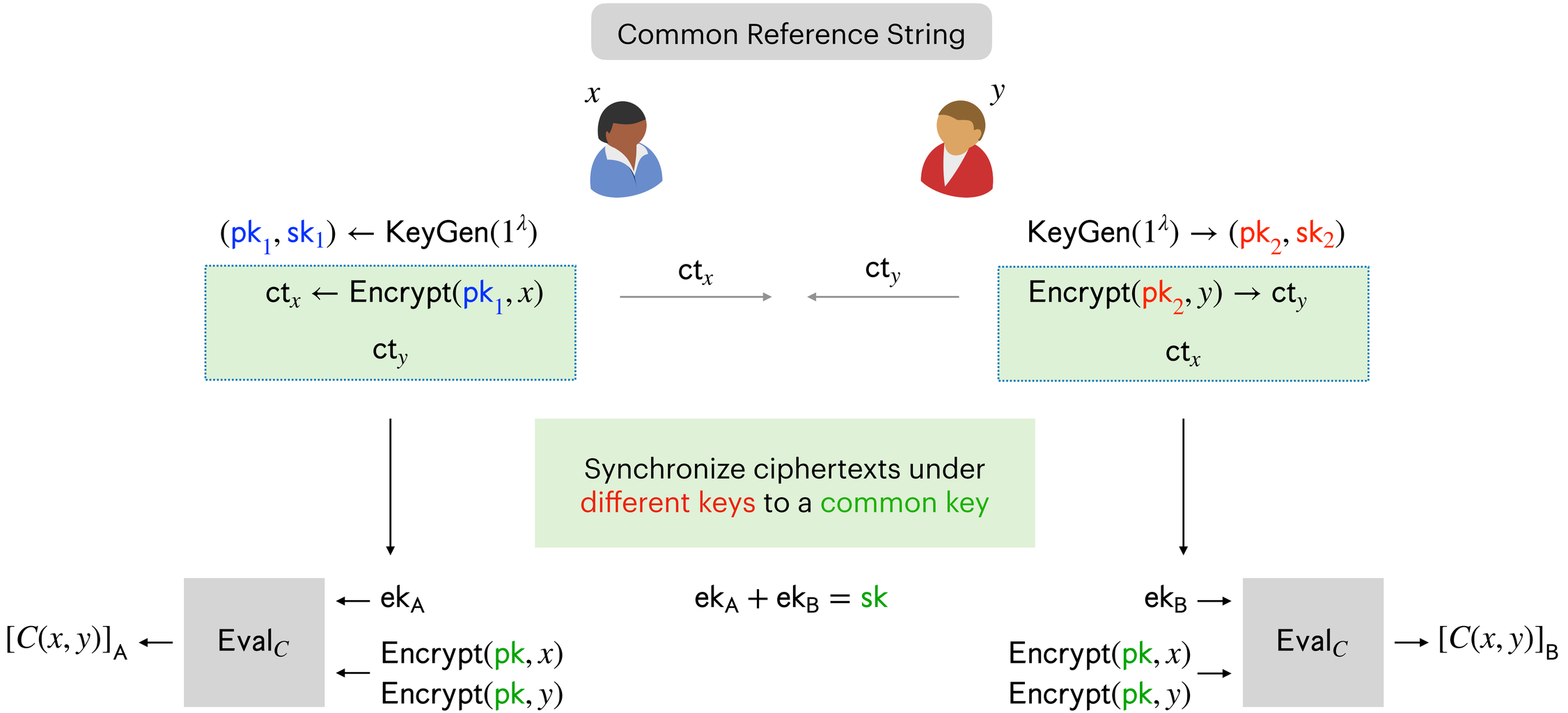
Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



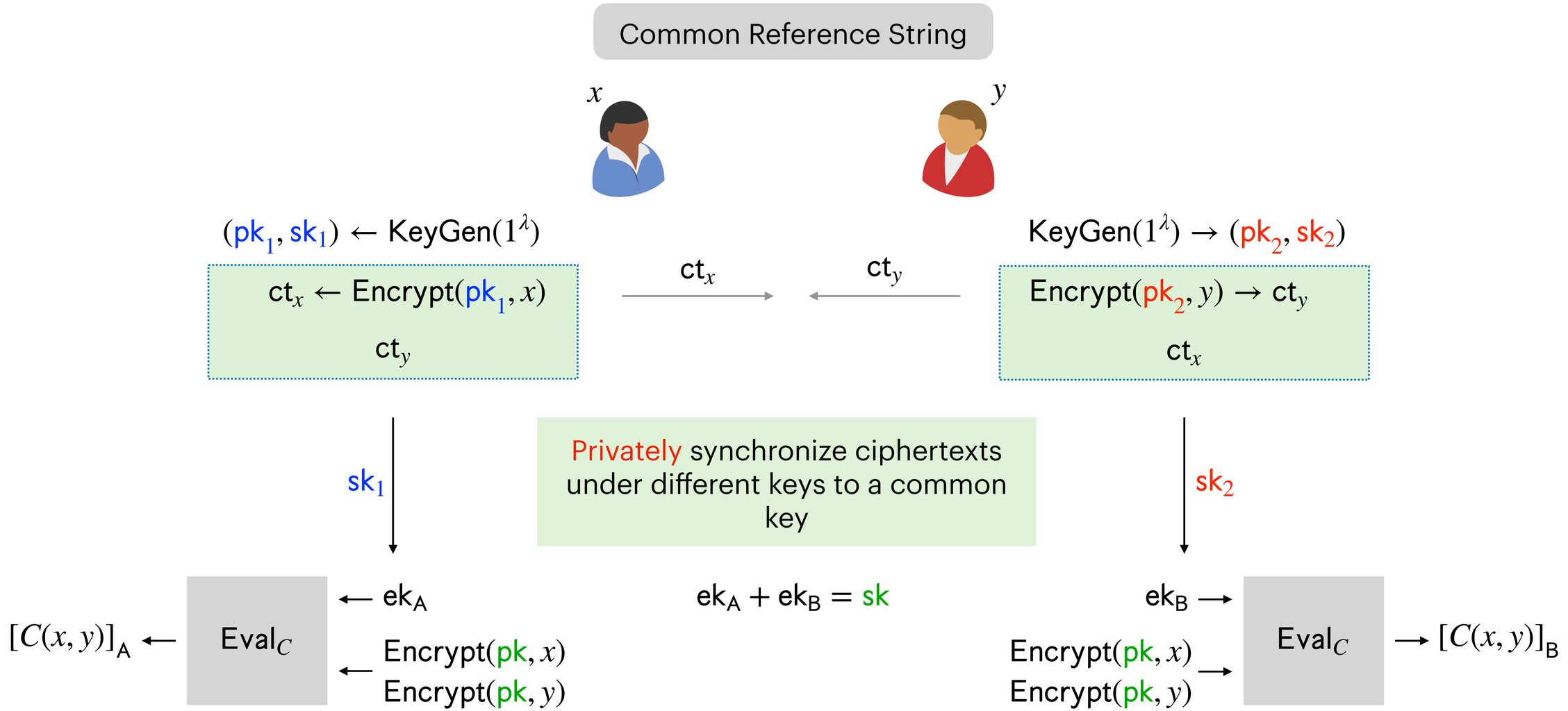
Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



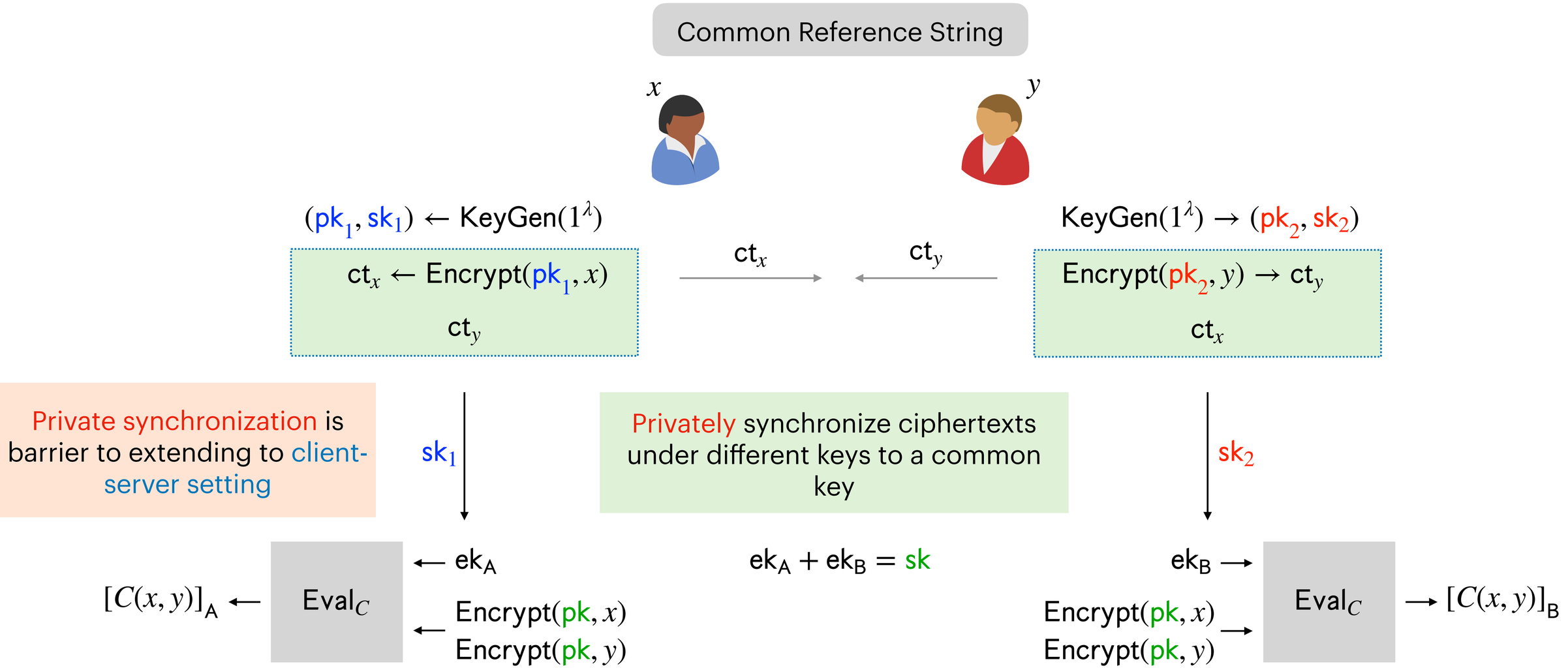
Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]

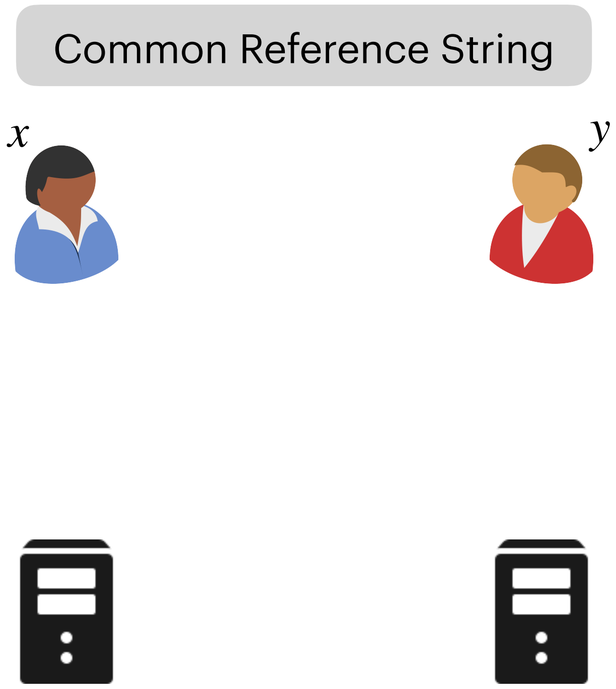


Two-Key HSS

[Couteau–Devadas–H–Jain–Servan-Schreiber'25]



Barriers to **Delegating** Two-key HSS



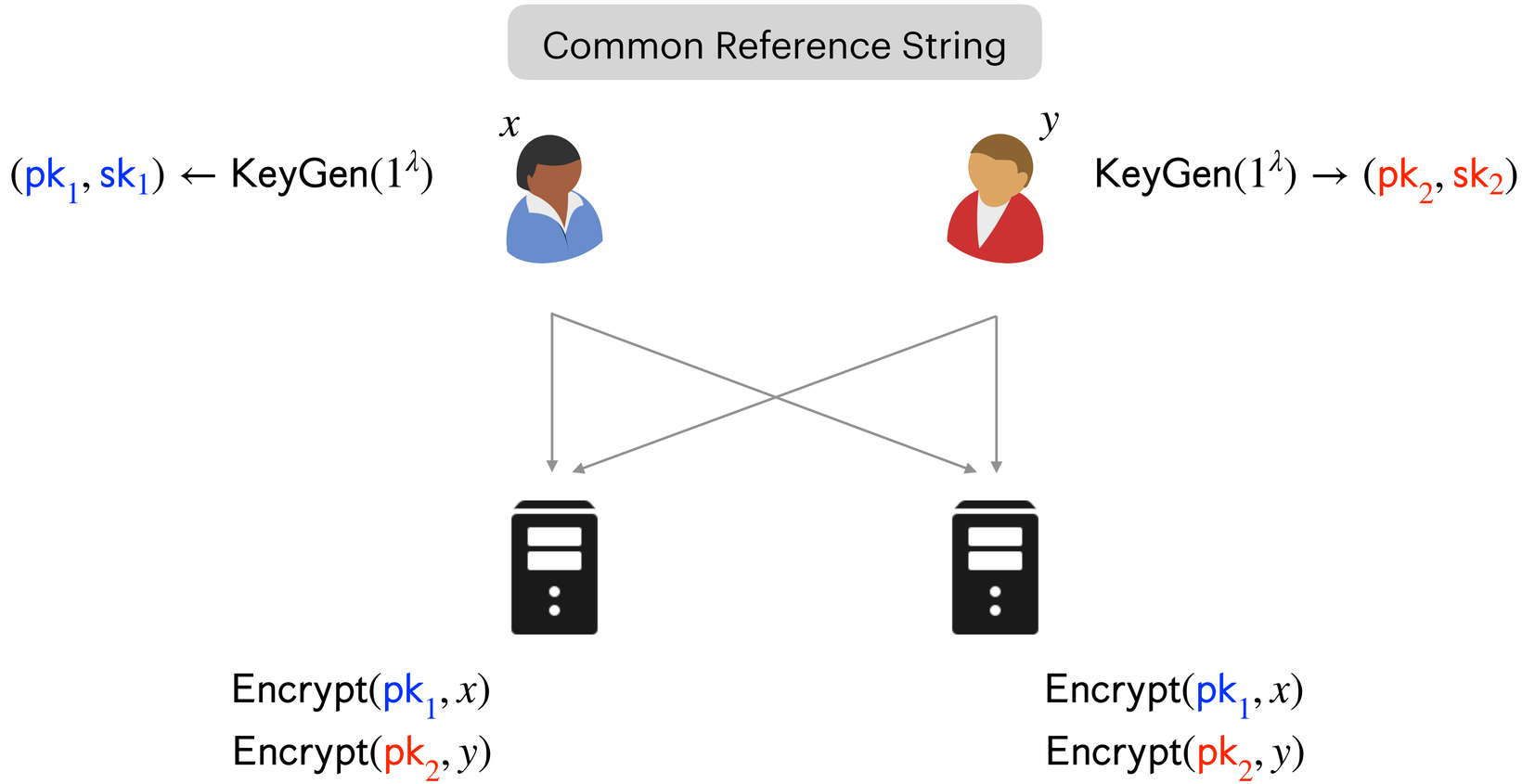
Barriers to **Delegating** Two-key HSS

Common Reference String

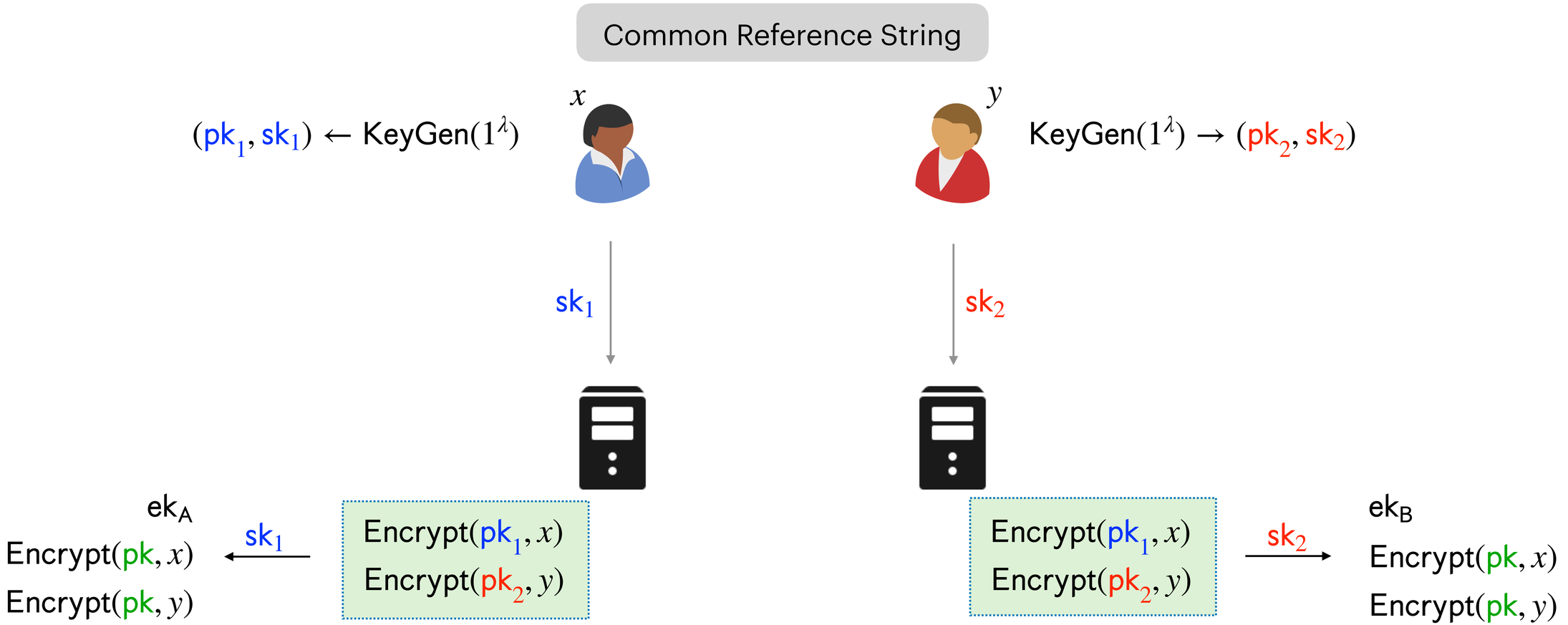
$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

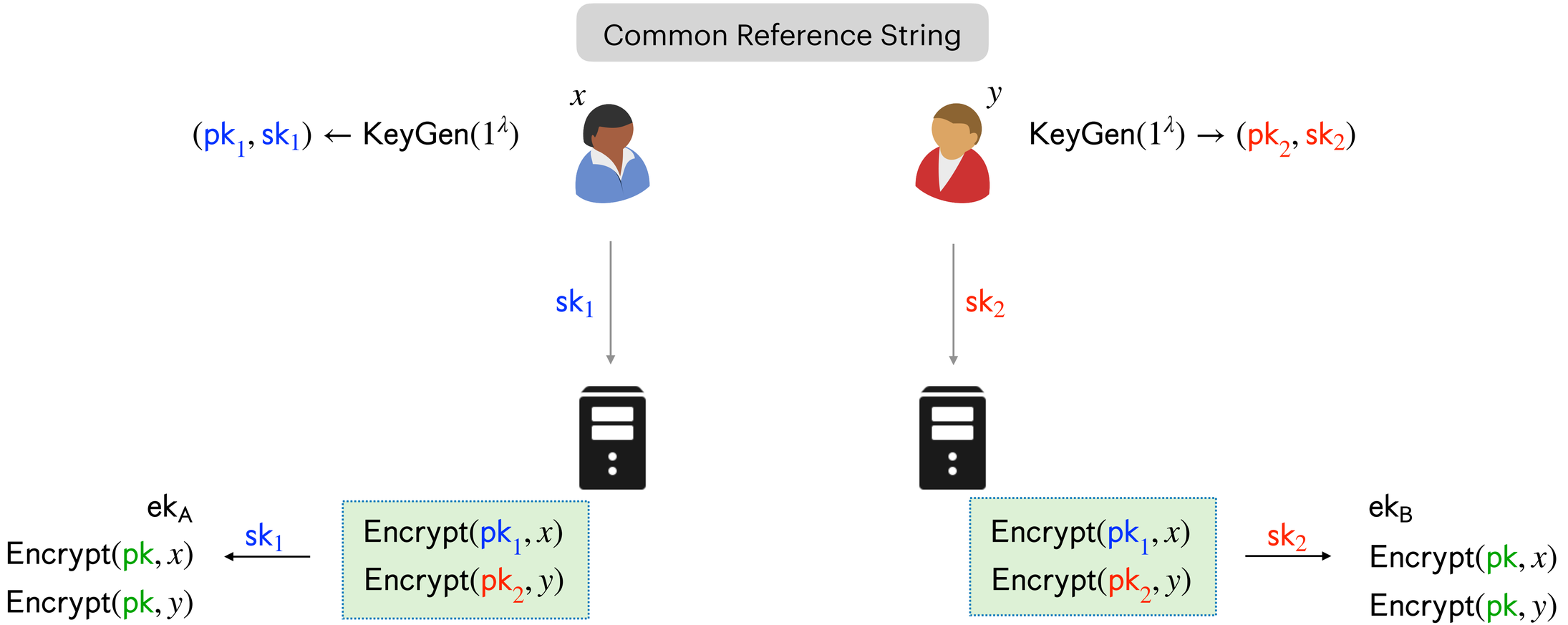
Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS



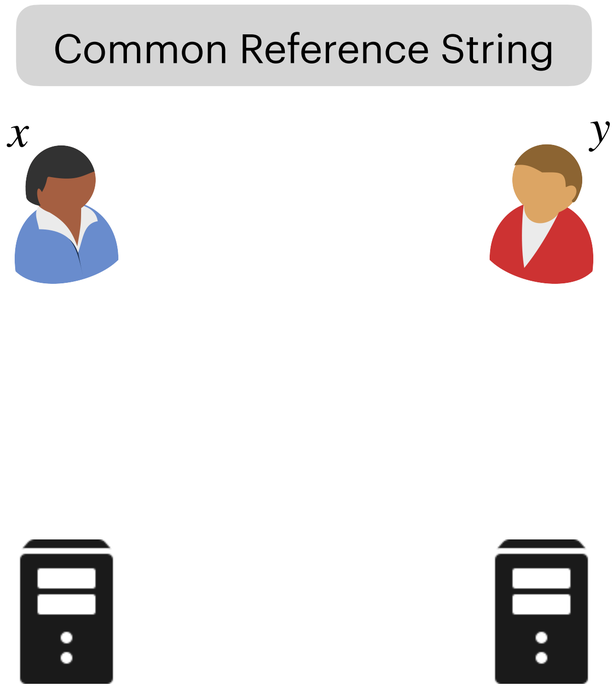
Barriers to **Delegating** Two-key HSS



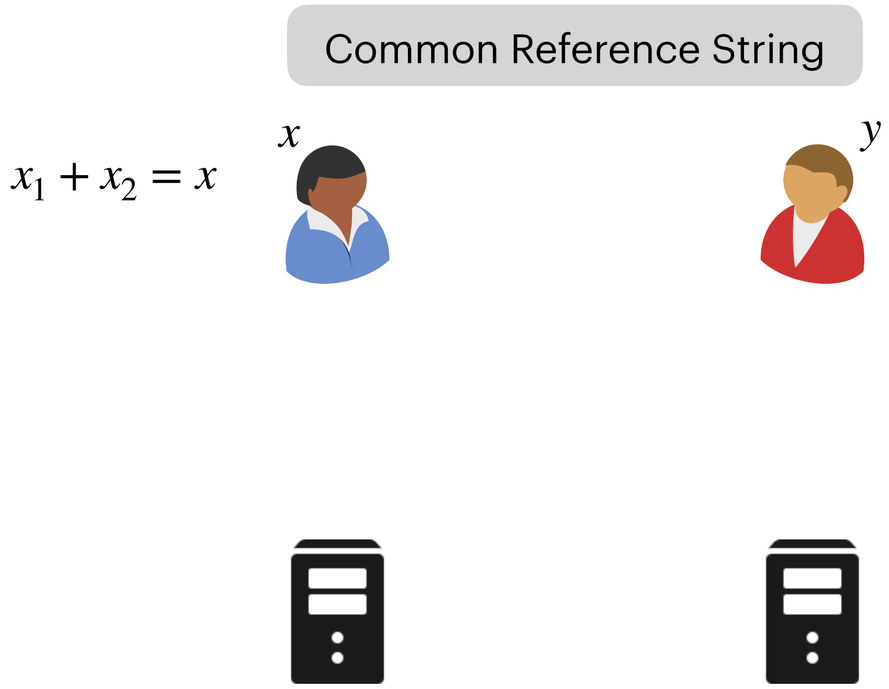
Revealing sk_1 compromises privacy of x

Revealing sk_2 compromises privacy of y

Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS

Common Reference String

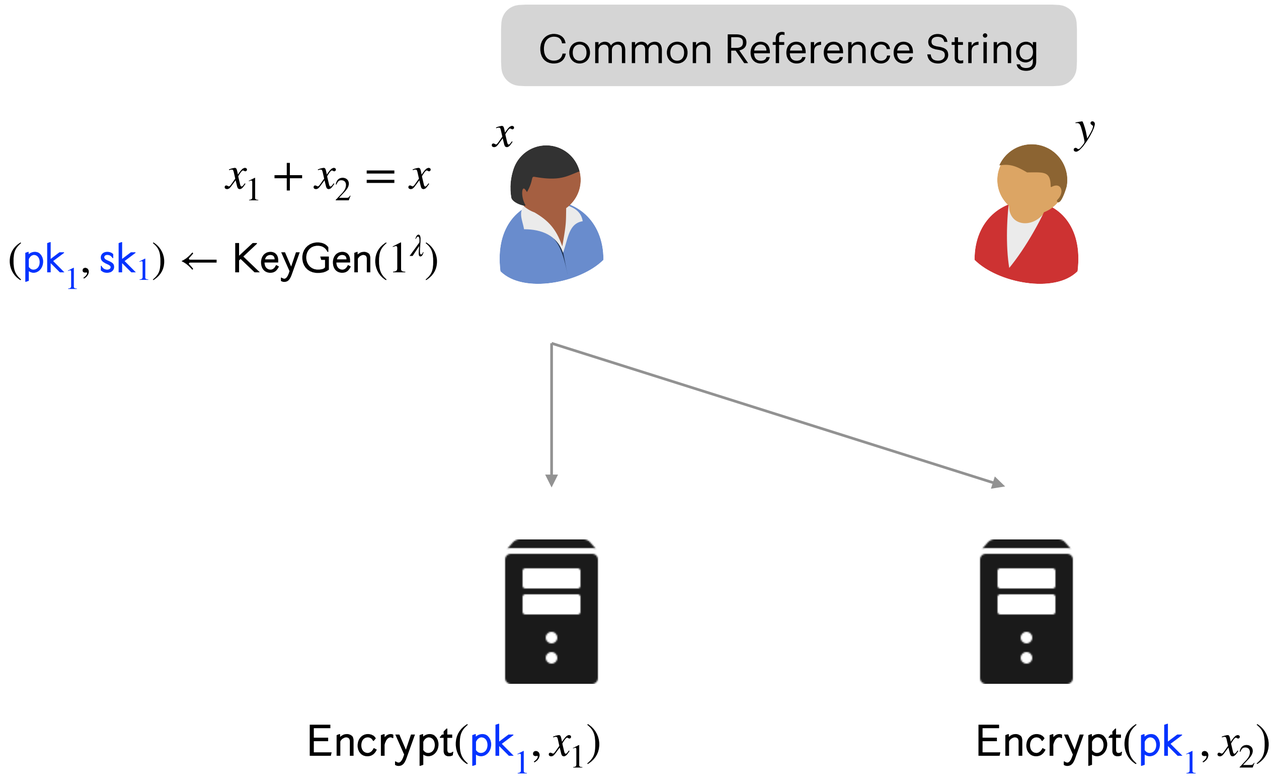
x

y

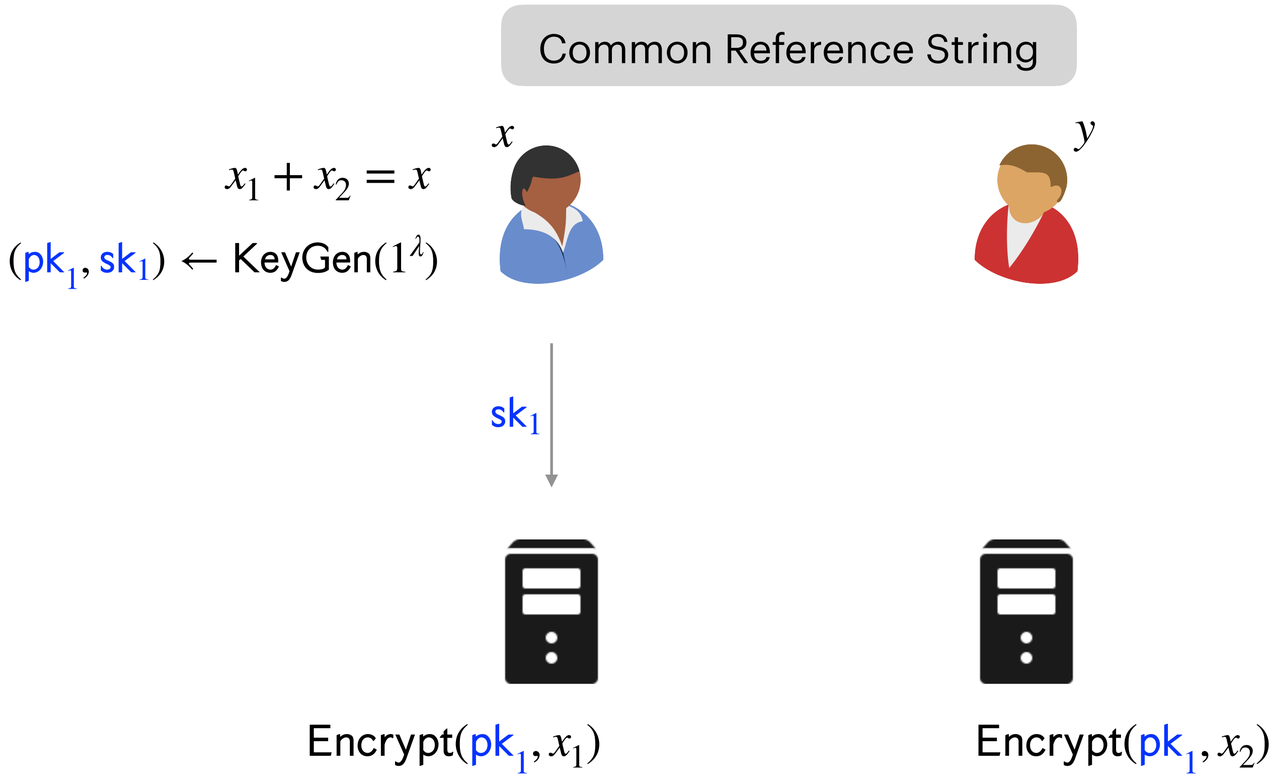
$x_1 + x_2 = x$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

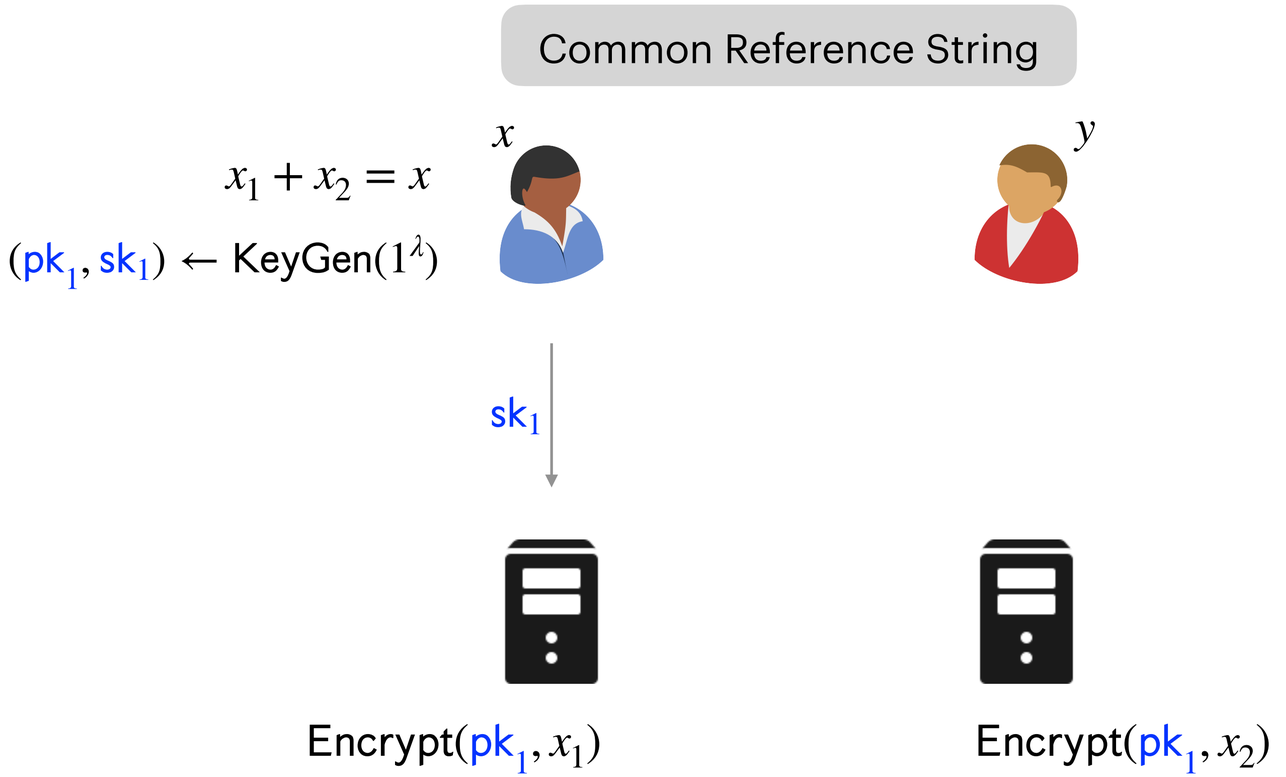
Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS

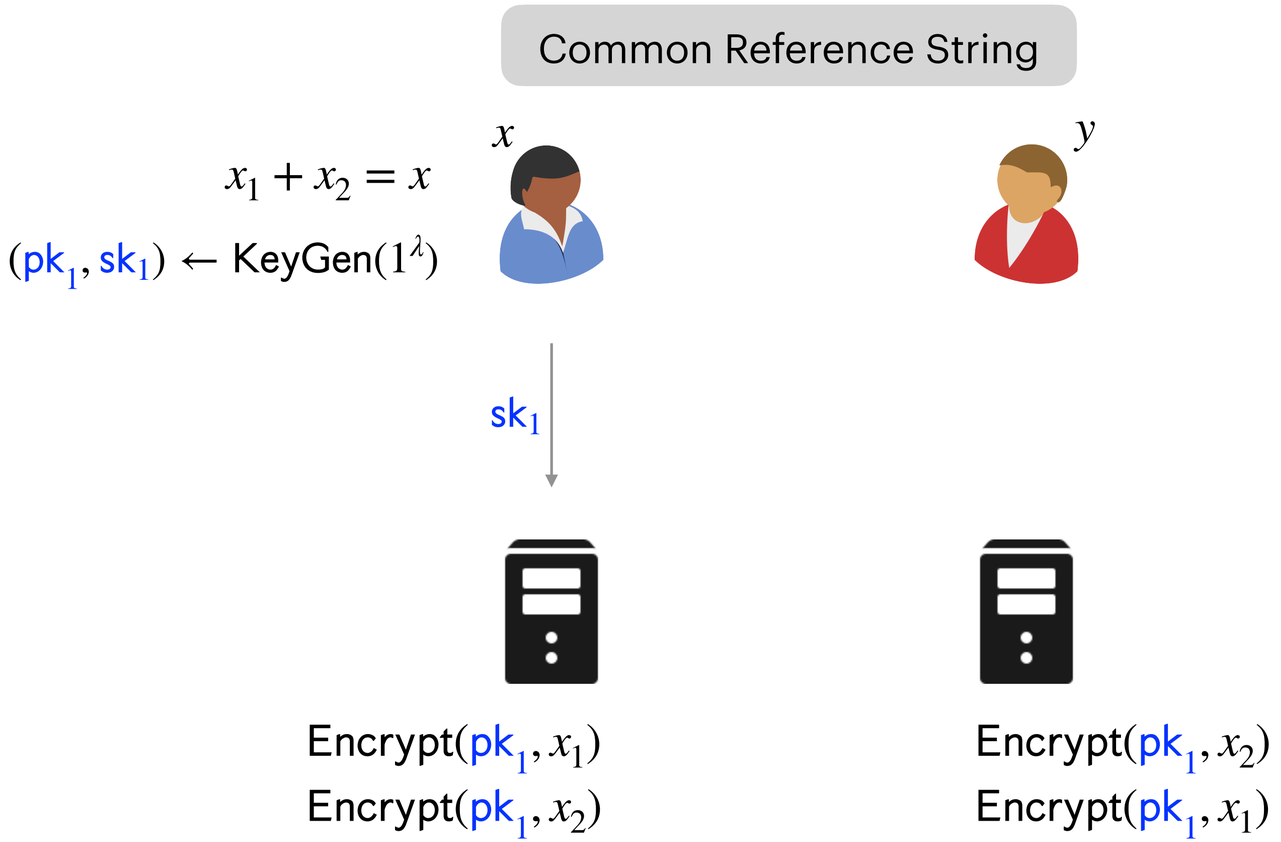


Barriers to **Delegating** Two-key HSS



Evaluation requires encryptions of all input

Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS

Common Reference String

x

y

$x_1 + x_2 = x$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

Barriers to **Delegating** Two-key HSS

Common Reference String

x

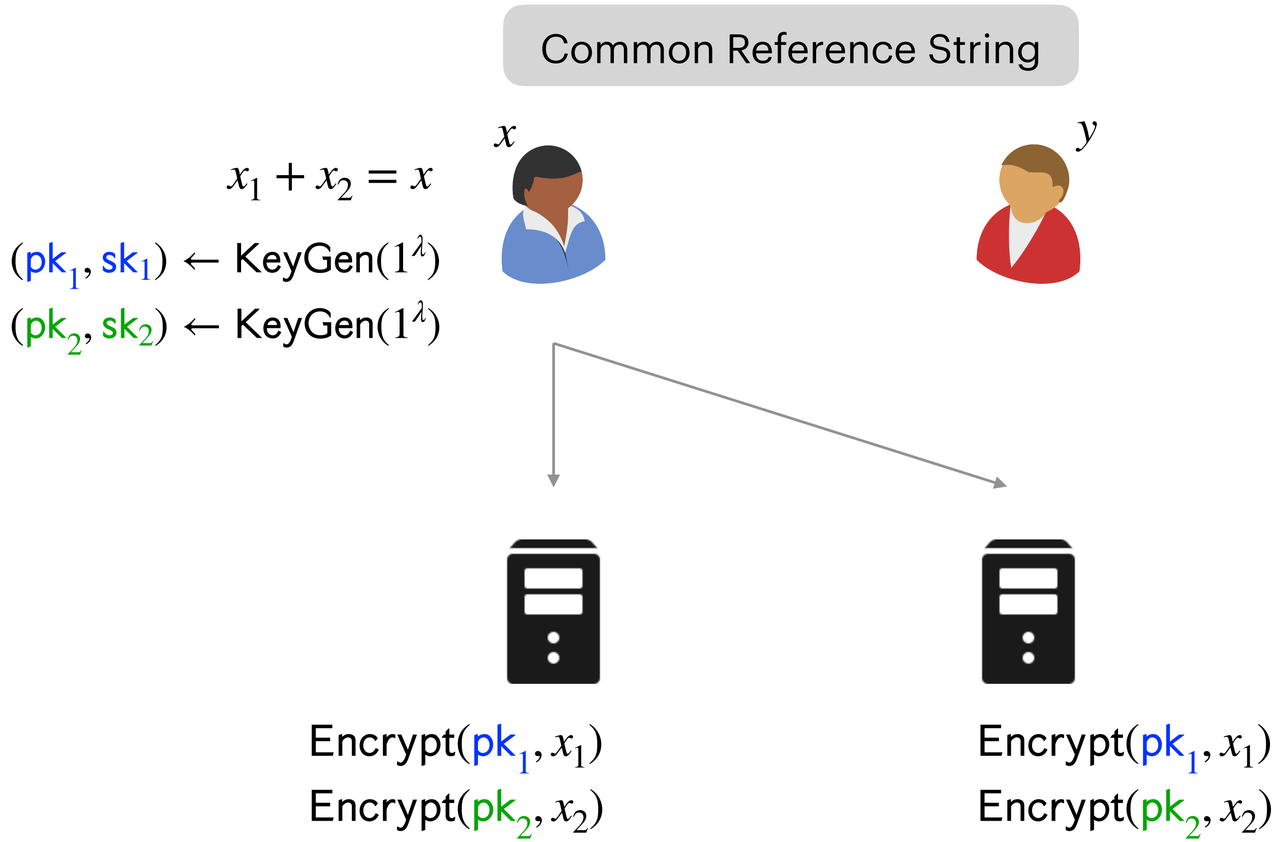
y

$x_1 + x_2 = x$

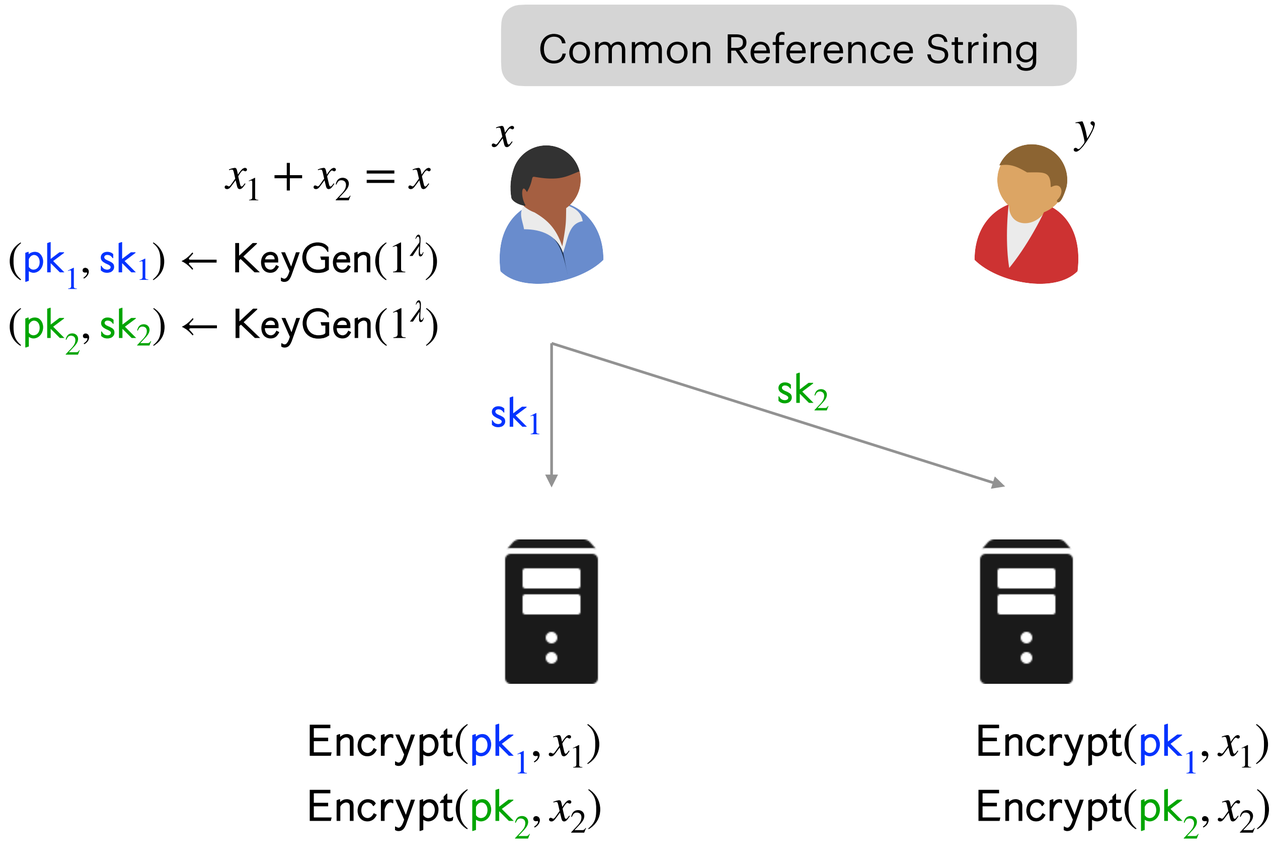
$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathsf{KeyGen}(1^\lambda)$

Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS

Common Reference String

x

$x_1 + x_2 = x$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\mathsf{Encrypt}(\mathsf{pk}_1, x_1)$

$\mathsf{Encrypt}(\mathsf{pk}_2, x_2)$

y

$y_1 + y_2 = y$

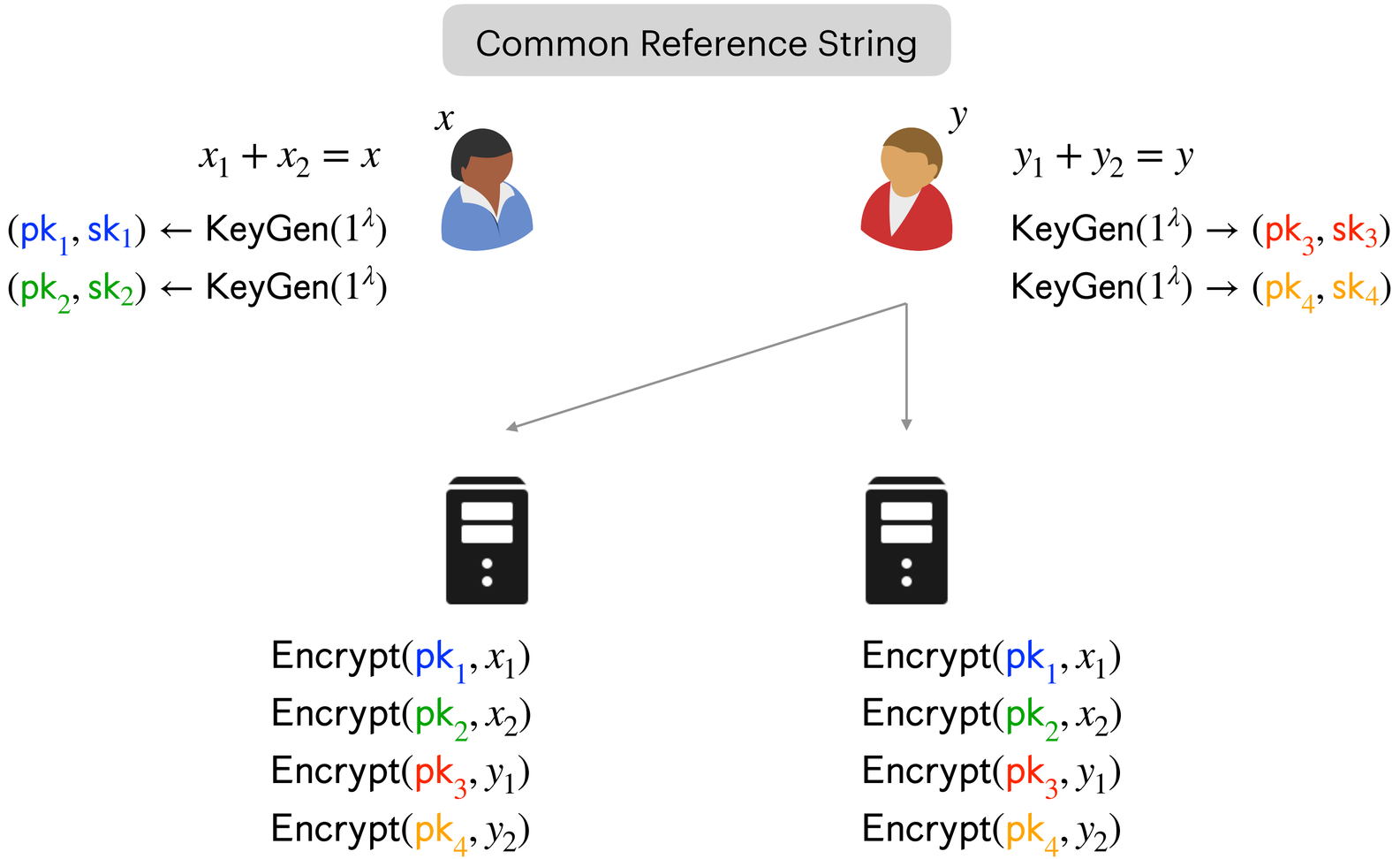
$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_3, \mathsf{sk}_3)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_4, \mathsf{sk}_4)$

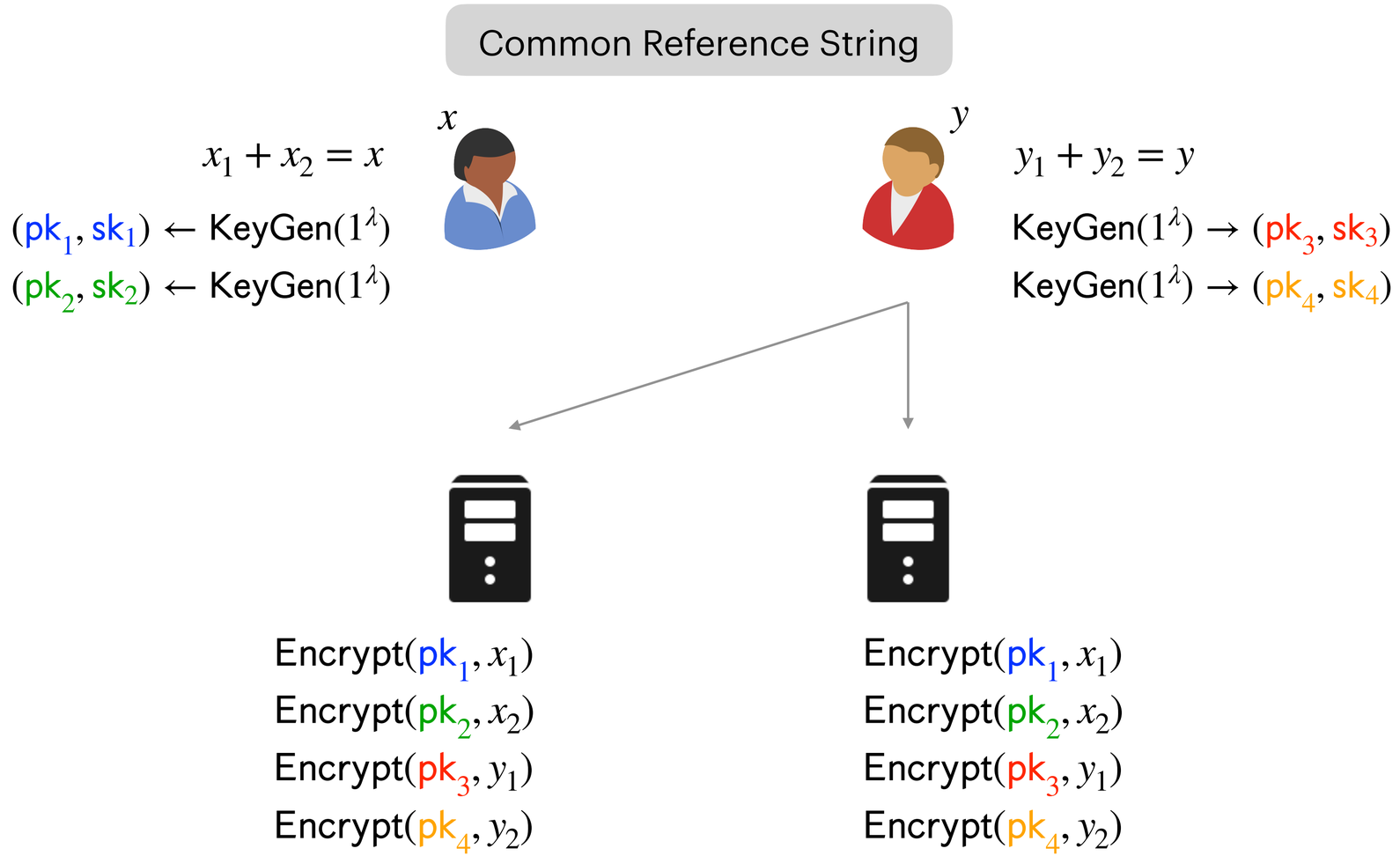
$\mathsf{Encrypt}(\mathsf{pk}_1, x_1)$

$\mathsf{Encrypt}(\mathsf{pk}_2, x_2)$

Barriers to **Delegating** Two-key HSS



Barriers to **Delegating** Two-key HSS



Requires evaluation under
four keys!

Barriers to **Delegating** Two-key HSS

Common Reference String

$x_1 + x_2 = x$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$y_1 + y_2 = y$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_3, \mathsf{sk}_3)$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_4, \mathsf{sk}_4)$

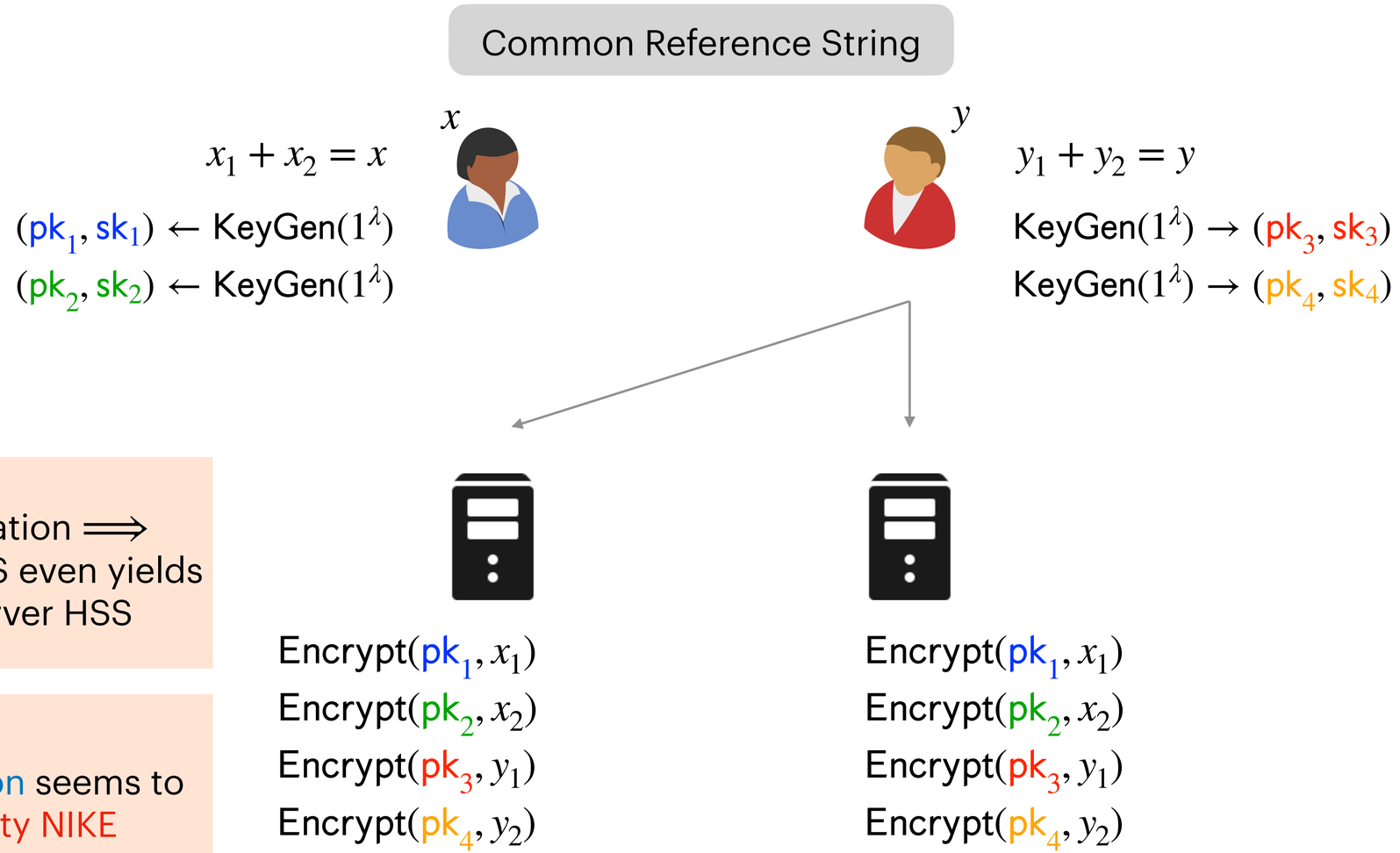
$\mathsf{Encrypt}(\mathsf{pk}_1, x_1)$
$\mathsf{Encrypt}(\mathsf{pk}_2, x_2)$
$\mathsf{Encrypt}(\mathsf{pk}_3, y_1)$
$\mathsf{Encrypt}(\mathsf{pk}_4, y_2)$

$\mathsf{Encrypt}(\mathsf{pk}_1, x_1)$
$\mathsf{Encrypt}(\mathsf{pk}_2, x_2)$
$\mathsf{Encrypt}(\mathsf{pk}_3, y_1)$
$\mathsf{Encrypt}(\mathsf{pk}_4, y_2)$

Private synchronization $\Longrightarrow$
Unclear if two-key HSS even yields two-client two-server HSS

Requires evaluation under four keys!

Barriers to **Delegating** Two-key HSS



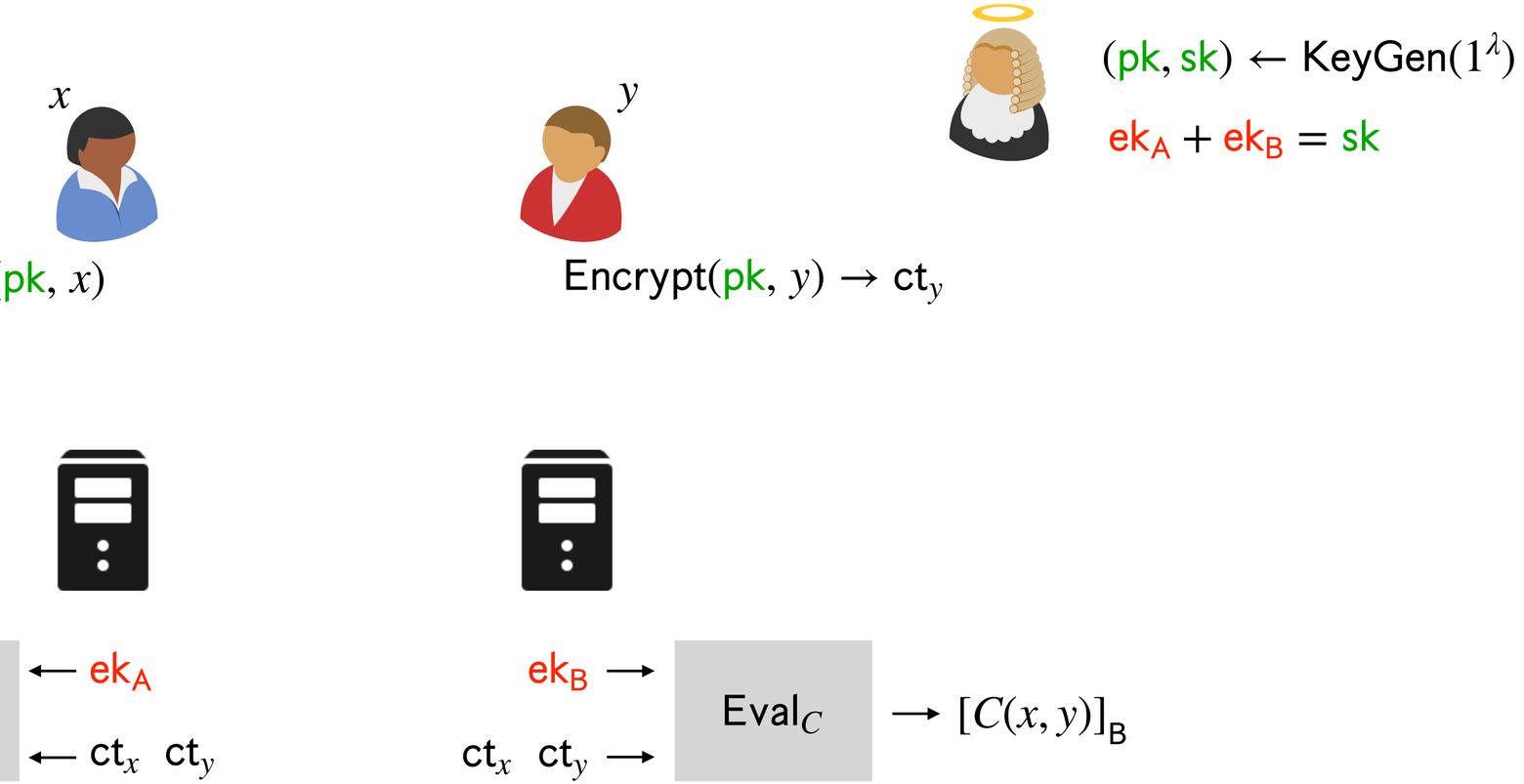
Private synchronization $\Longrightarrow$
Unclear if two-key HSS even yields two-client two-server HSS

Public synchronization seems to require three-party NIKE

Requires evaluation under four keys!

Client-Server HSS with **Correlated Setup**

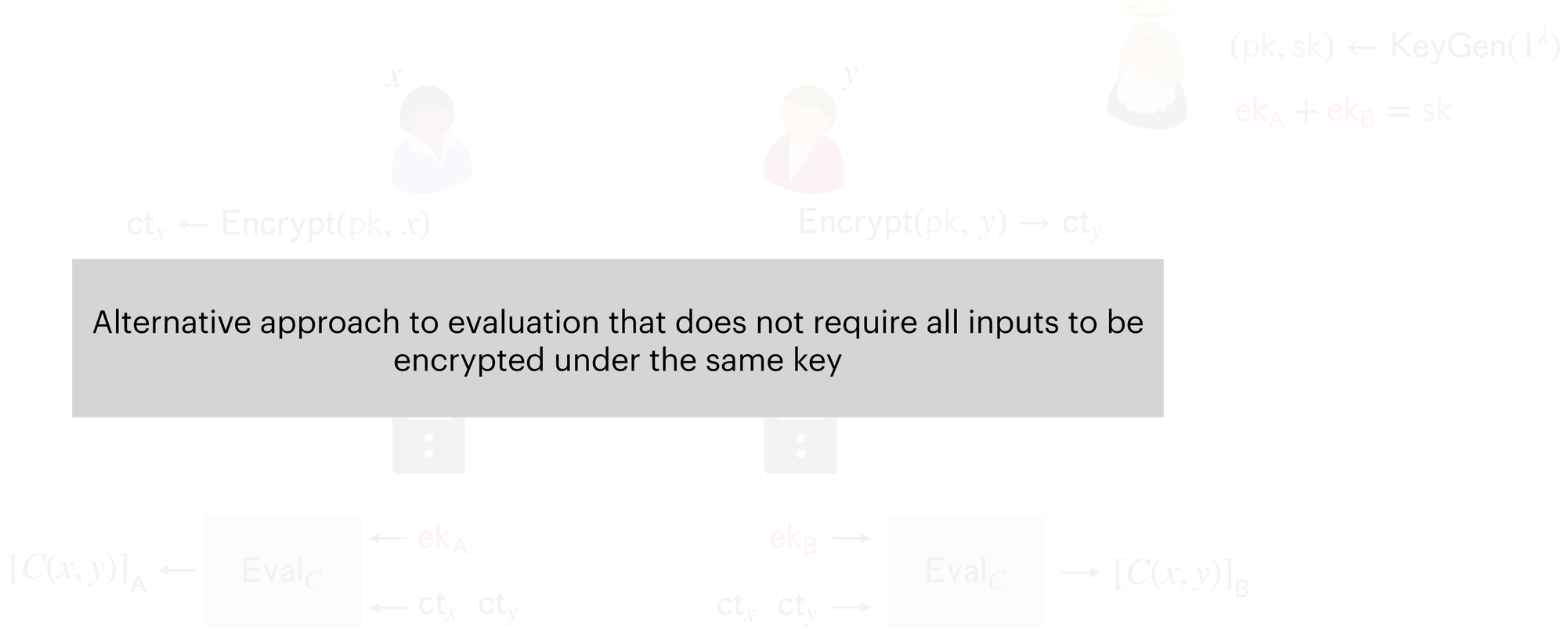
[Boyle-Gilboa-Ishai'16]



Barrier to Removing Correlated Setup: All inputs must be encrypted under a common key

Client-Server HSS with **Correlated Setup**

[Boyle-Gilboa-Ishai'16]



Barrier to Removing Correlated Setup: All inputs must be encrypted under a common key

Outline

Barriers to Removing Correlated Setup

Our Approach

Extensions

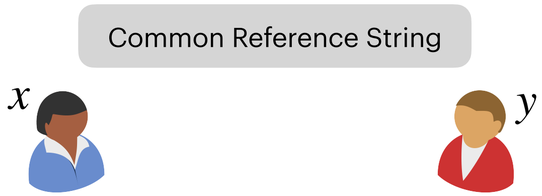
HSS for **Multiplication** is All You Need

HSS for **Multiplication** is All You Need

Two-party HSS for
multiplication in the CRS model

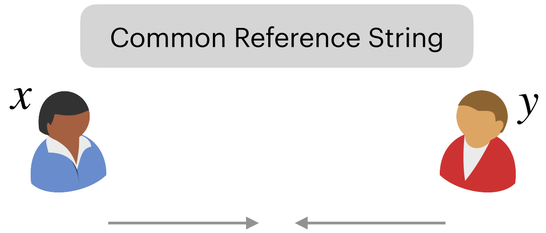
HSS for **Multiplication** is All You Need

Two-party HSS for
multiplication in the CRS model



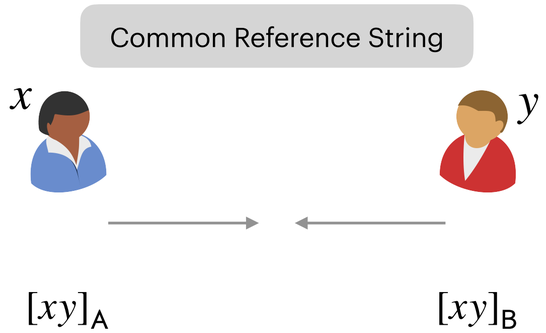
HSS for **Multiplication** is All You Need

Two-party HSS for
multiplication in the CRS model



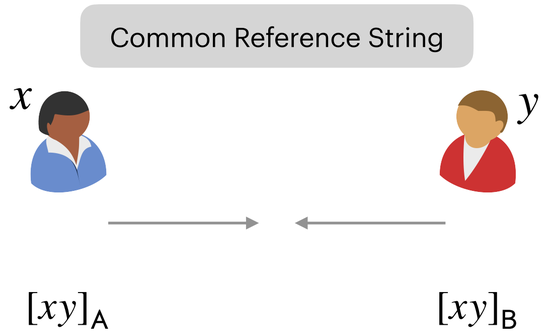
HSS for **Multiplication** is All You Need

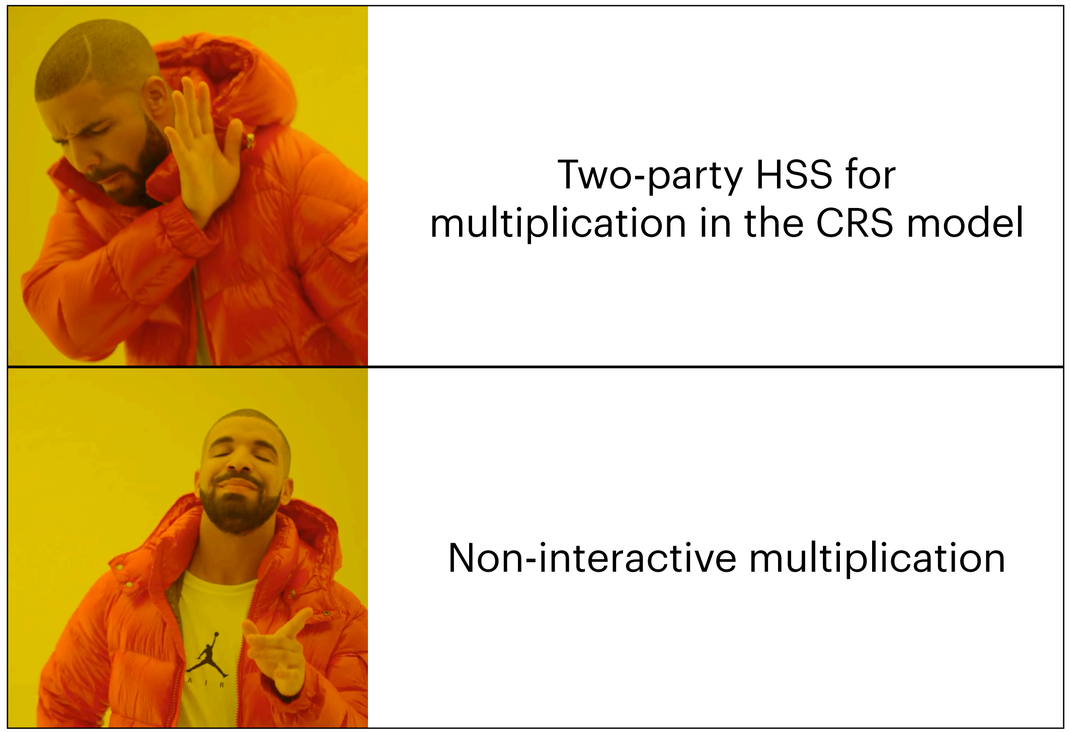
Two-party HSS for
multiplication in the CRS model



HSS for **Multiplication** is All You Need

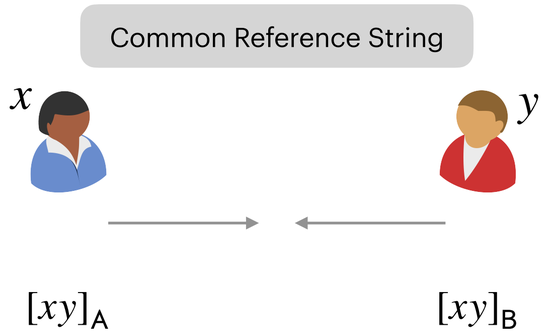
Two-party HSS for
multiplication in the CRS model

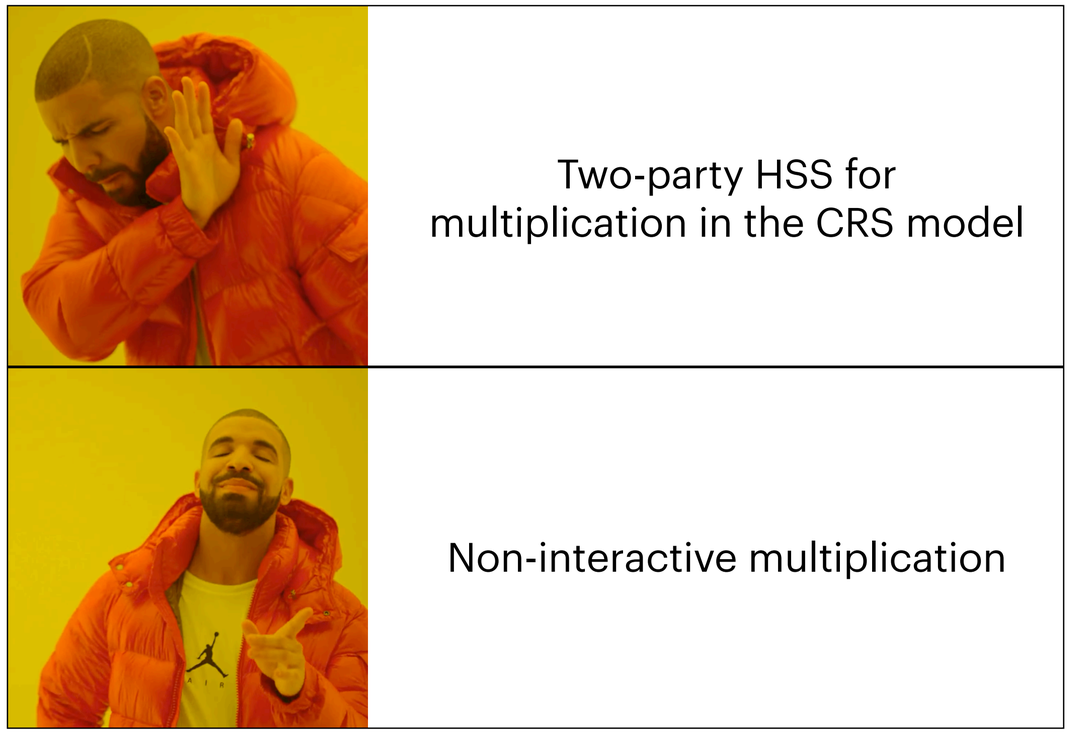




HSS for **Multiplication** is All You Need

Non-interactive multiplication

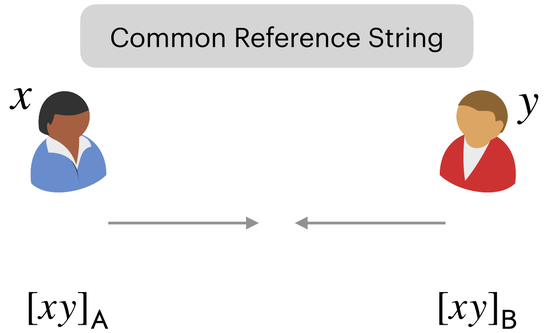




HSS for **Multiplication** is All You Need

Non-interactive multiplication

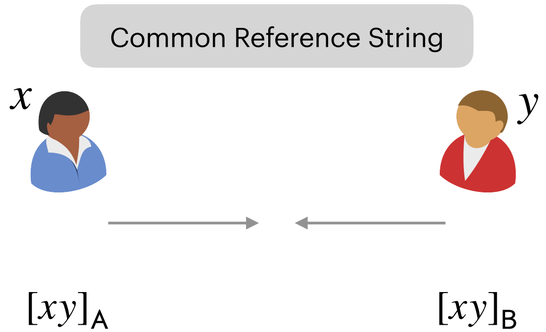
Two-client two-server HSS for multiplication in the CRS model

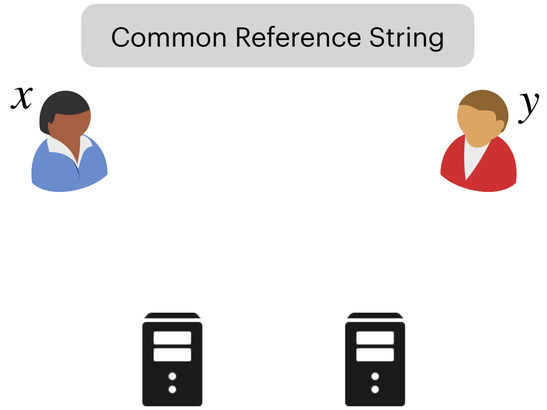


HSS for **Multiplication** is All You Need

Non-interactive multiplication

Two-client two-server HSS for multiplication in the CRS model

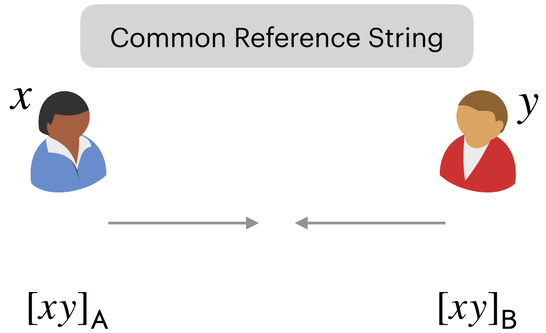


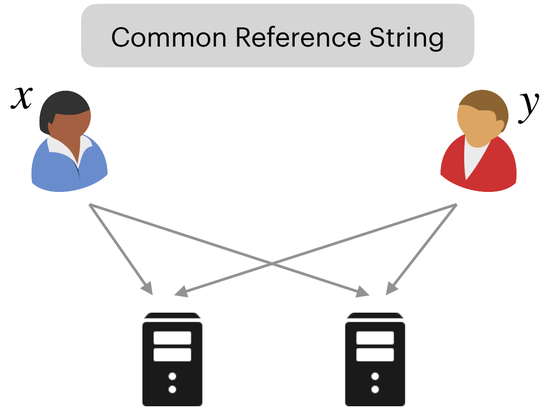


HSS for **Multiplication** is All You Need

Non-interactive multiplication

Two-client two-server HSS for multiplication in the CRS model

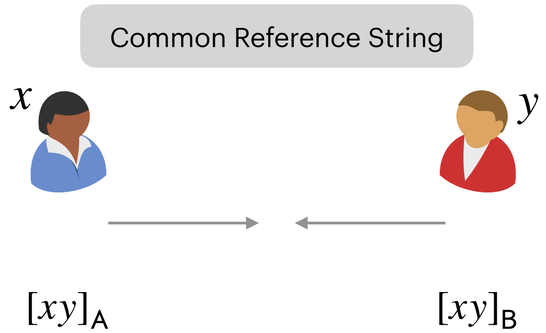


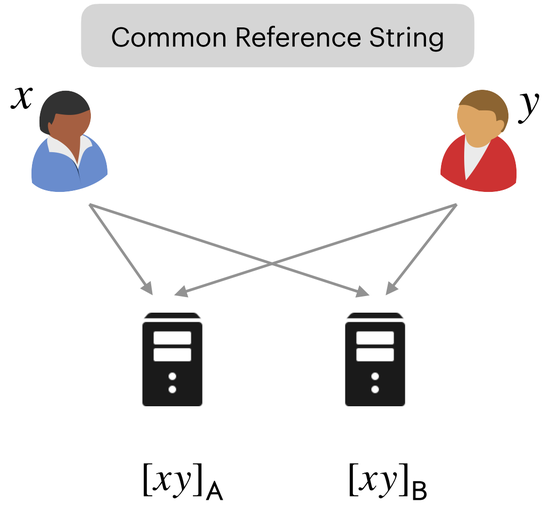


HSS for **Multiplication** is All You Need

Non-interactive multiplication

Two-client two-server HSS for multiplication in the CRS model

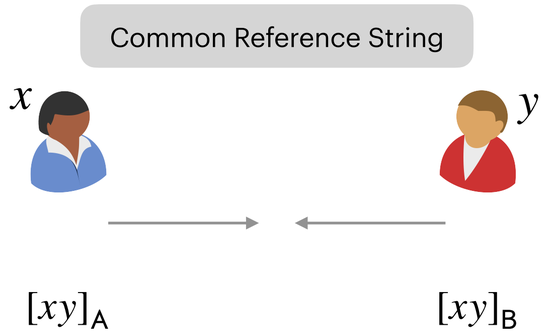


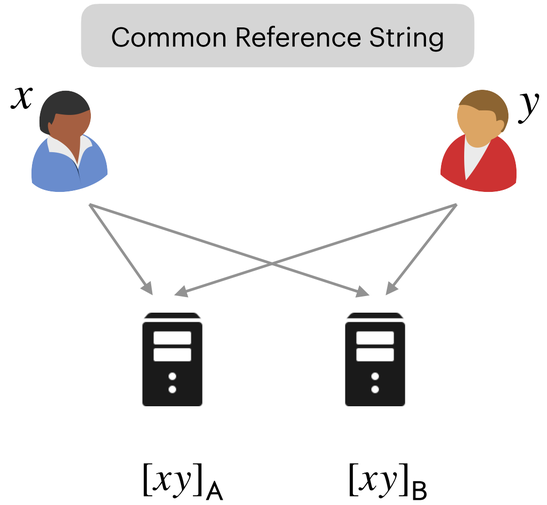


HSS for **Multiplication** is All You Need

Non-interactive multiplication

Delegatable non-interactive multiplication



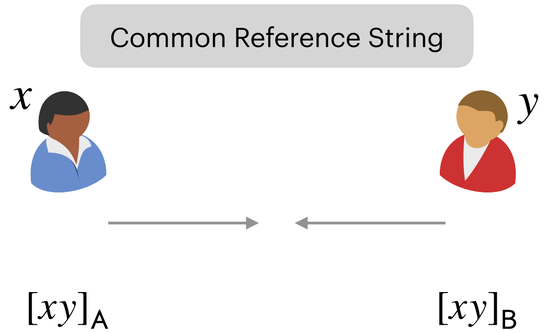


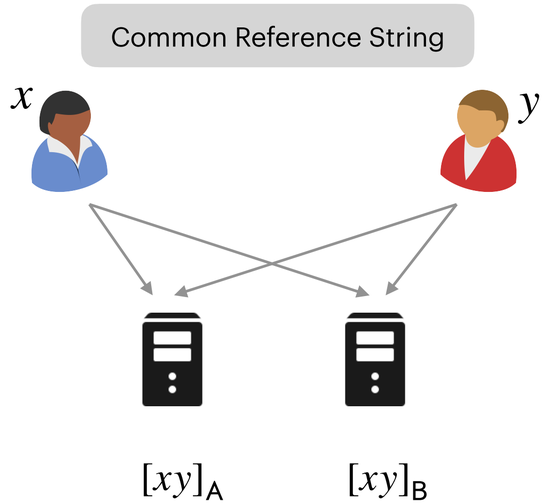
HSS for **Multiplication** is All You Need

Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs



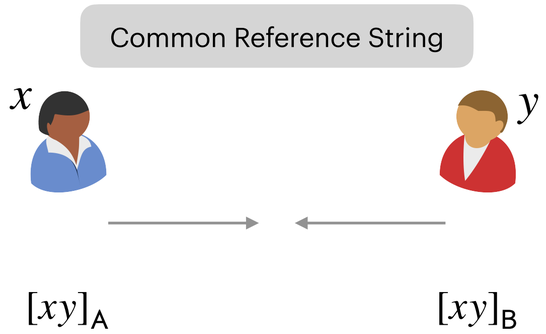


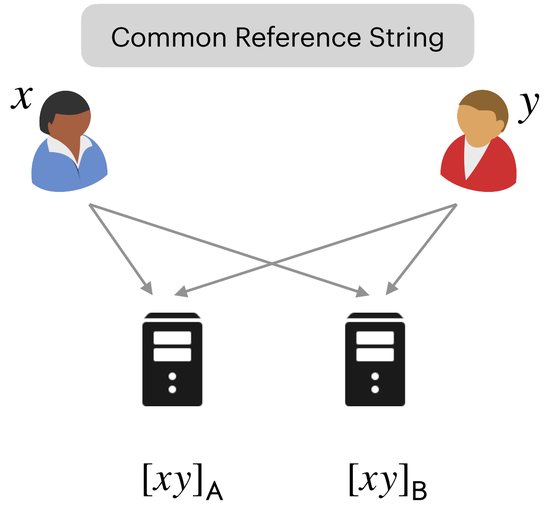
HSS for **Multiplication** is All You Need

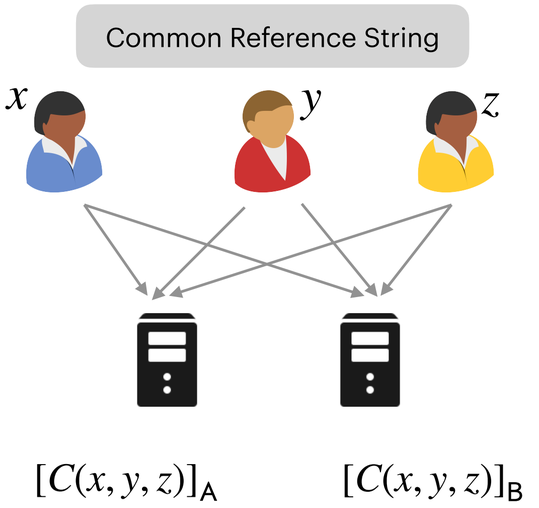
Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs





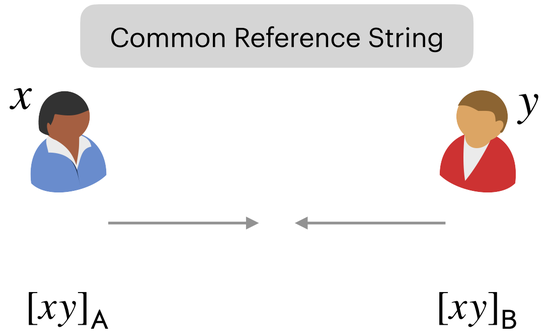


HSS for **Multiplication** is All You Need

Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs



Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\widehat{x} = (h^r, g^r \cdot g^x)$

Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

x

$r \leftarrow \mathbb{Z}_p$

$\widehat{x} = (h^r, g^r \cdot g^x)$

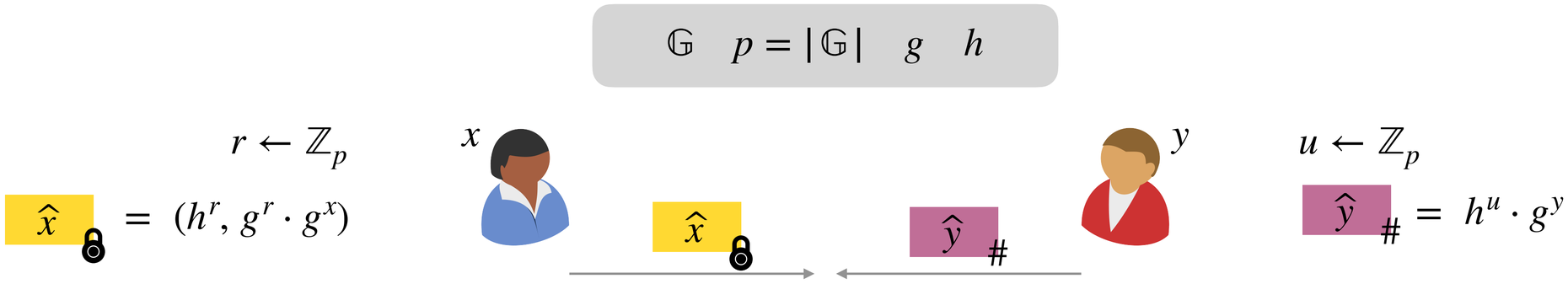
y

$u \leftarrow \mathbb{Z}_p$

$\widehat{y}_{\#} = h^u \cdot g^y$

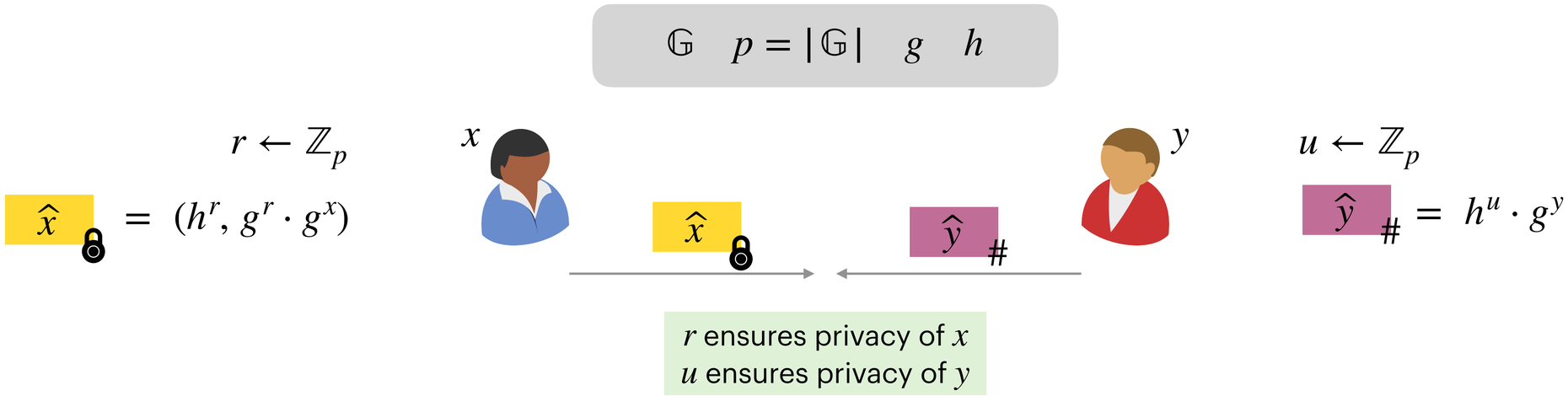
Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]



Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]



Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

x

y

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$\hat{x}$

$\hat{y}_{\#}$

$\gamma_{\mathsf{A}} = \left(h^u \cdot g^y\right)^r$

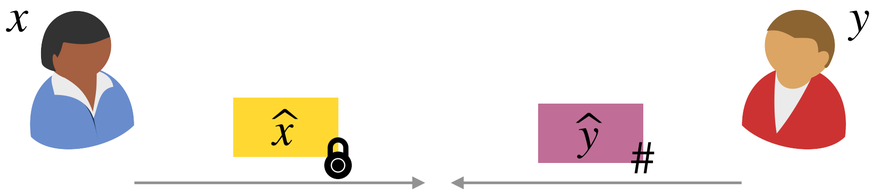
Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$\gamma_A = \left(h^u \cdot g^y\right)^r$

$\gamma_B = \left(h^r\right)^u \cdot \left(g^r \cdot g^x\right)^y$

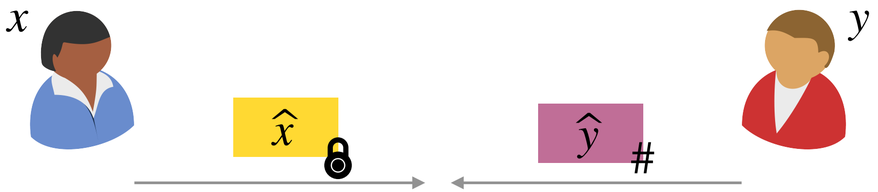
Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y} = h^u \cdot g^y$

$\gamma_A = \left(h^u \cdot g^y\right)^r$

$\gamma_B = \left(h^r\right)^u \cdot \left(g^r \cdot g^x\right)^y$

$$\frac{\gamma_B}{\gamma_A} = \frac{h^{r \cdot u} \cdot g^{r \cdot y} \cdot g^{x \cdot y}}{h^{u \cdot r} \cdot g^{y \cdot r}} = g^{xy}$$

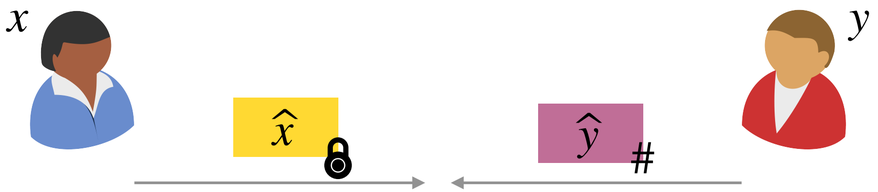
Non-Interactive Multiplication

[Döttling-Garg-Ishai-Malavolta-Mour-Ostrovsky'19] [Abram-Roy-Scholl'24]

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y} = h^u \cdot g^y$

$\gamma_A = (h^u \cdot g^y)^r$

DDLog ↓

$[xy]_A$

$$\frac{\gamma_B}{\gamma_A} = \frac{h^{r \cdot u} \cdot g^{r \cdot y} \cdot g^{x \cdot y}}{h^{u \cdot r} \cdot g^{y \cdot r}} = g^{xy}$$

$\gamma_B = (h^r)^u \cdot (g^r \cdot g^x)^y$

DDLog ↓

$[xy]_B$

Distributed Discrete Log (DDLog): Non-interactively convert divisive shares into additive shares

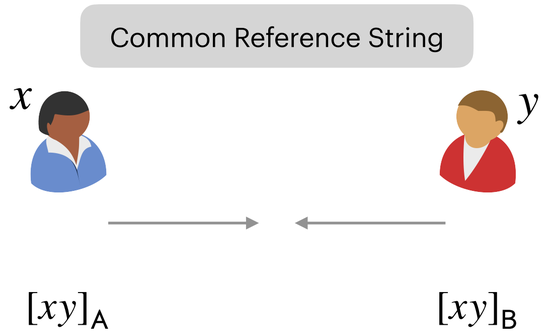
[Boyle-Gilboa-Ishai'16]

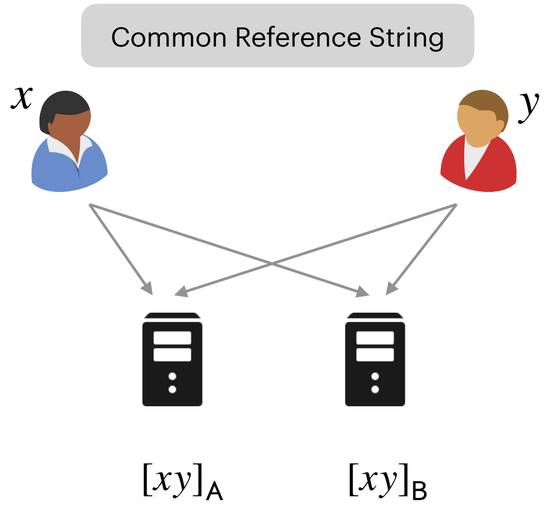
HSS for **Multiplication** is All You Need

Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs



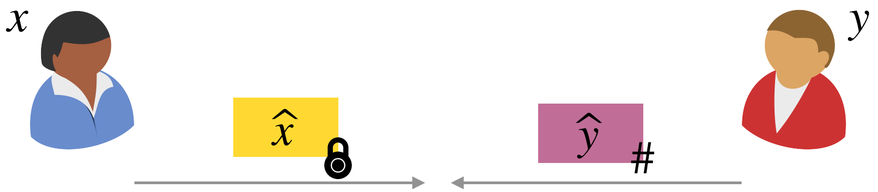


Delegating Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$\gamma_A = (h^u \cdot g^y)^r$

DDLog ↓

$[xy]_A$

$$\frac{\gamma_B}{\gamma_A} = \frac{(h^r)^u \cdot (g^r \cdot g^x)^y}{(h^u \cdot g^y)^r} = g^{xy}$$

$\gamma_B = (h^r)^u \cdot (g^r \cdot g^x)^y$

DDLog ↓

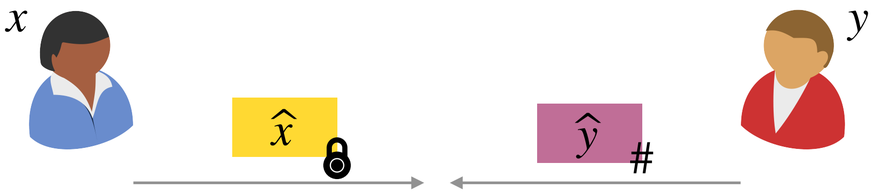
$[xy]_B$

Delegating Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$\gamma_A = (h^u \cdot g^y)^r$

DDLog

$[xy]_A$

$$\frac{\gamma_B}{\gamma_A} = \frac{(h^r)^u \cdot (g^r \cdot g^x)^y}{(h^u \cdot g^y)^r} = g^{xy}$$

$\gamma_B = (h^r)^u \cdot (g^r \cdot g^x)^y$

DDLog

$[xy]_B$

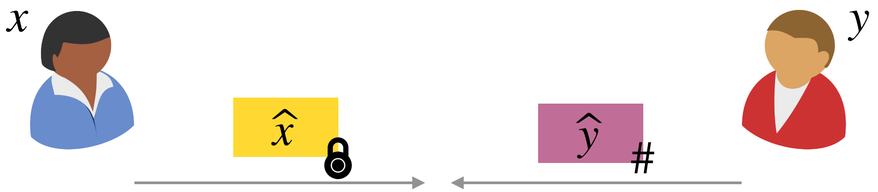
Observation: Decoding is "linear" in the received messages

Delegating Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$



$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$\gamma_A = (h^u \cdot g^y)^r$

DDLog

$[xy]_A$

$$\frac{\gamma_B}{\gamma_A} = \frac{(h^r)^u \cdot (g^r \cdot g^x)^y}{(h^u \cdot g^y)^r} = g^{xy}$$

$\gamma_B = (h^r)^u \cdot (g^r \cdot g^x)^y$

DDLog

$[xy]_B$

Observation: Decoding is "linear" in the received messages

Using shares of r, u, and y to decode gives divisive shares of xy

Delegatable Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

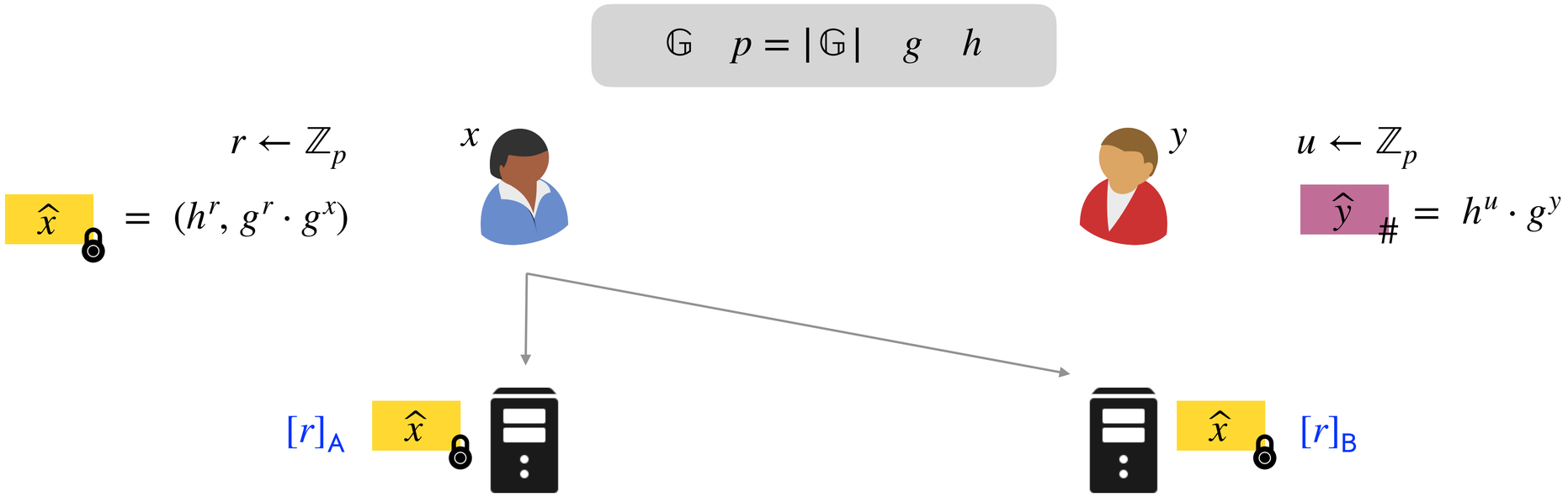
x

y

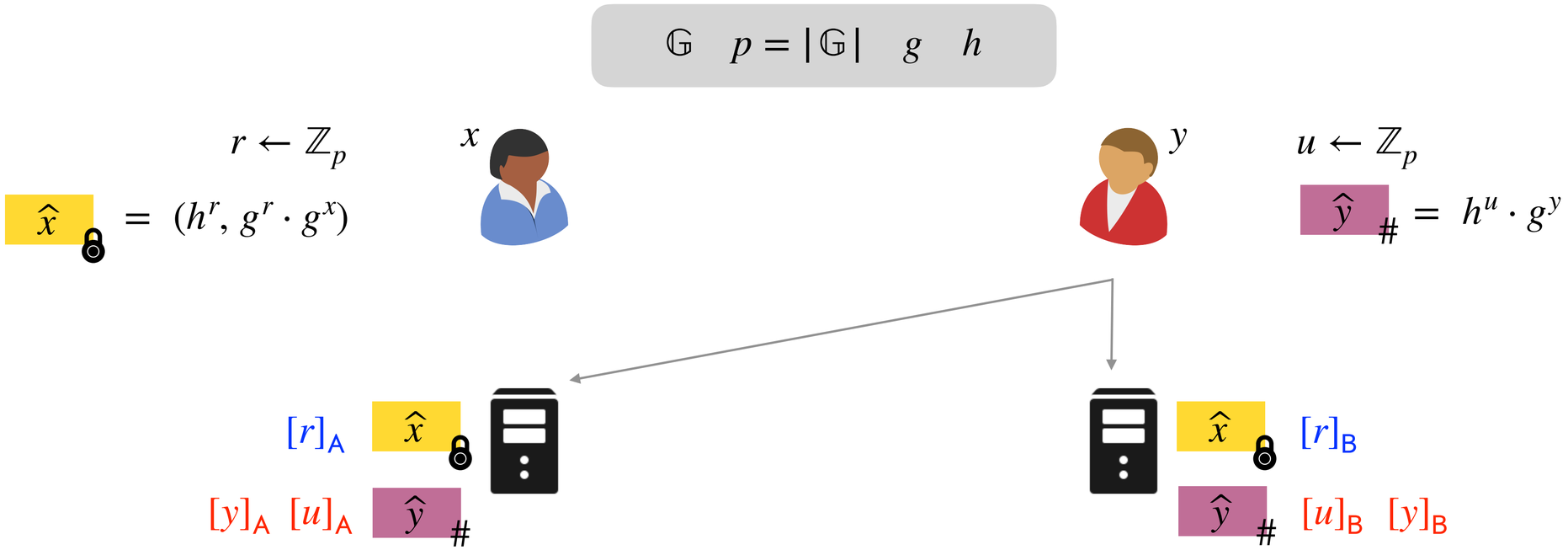
$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

Delegatable Non-Interactive Multiplication



Delegatable Non-Interactive Multiplication



Delegatable Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\widehat{x} = (h^r, g^r \cdot g^x)$

x

y

$u \leftarrow \mathbb{Z}_p$

$\widehat{y}_{\#} = h^u \cdot g^y$

$[r]_A \quad \widehat{x}$

$[y]_A \quad [u]_A \quad \widehat{y}_{\#}$

$\widehat{x} \quad [r]_B$

$\widehat{y}_{\#} \quad [u]_B \quad [y]_B$

$$g^{-[xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[r]_A}}{\left(h^r\right)^{[u]_A} \cdot \left(g^r \cdot g^x\right)^{[y]_A}}$$

Delegatable Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\widehat{x} = (h^r, g^r \cdot g^x)$

x

y

$u \leftarrow \mathbb{Z}_p$

$\widehat{y}_{\#} = h^u \cdot g^y$

$[r]_A \quad \widehat{x}$

$[y]_A \quad [u]_A \quad \widehat{y}_{\#}$

$\widehat{x} \quad [r]_B$

$\widehat{y}_{\#} \quad [u]_B \quad [y]_B$

$$g^{-[xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[r]_A}}{\left(h^r\right)^{[u]_A} \cdot \left(g^r \cdot g^x\right)^{[y]_A}}$$

$$\frac{\left(h^r\right)^{[u]_B} \cdot \left(g^r \cdot g^x\right)^{[y]_B}}{\left(h^u \cdot g^y\right)^{[r]_B}} = g^{[xy]_B}$$

Delegatable Non-Interactive Multiplication

$$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$$

$r \leftarrow \mathbb{Z}_p$

$\widehat{x} = (h^r, g^r \cdot g^x)$

$u \leftarrow \mathbb{Z}_p$

$\widehat{y}_{\#} = h^u \cdot g^y$

$[r]_A$ $\widehat{x}$

$[y]_A$ $[u]_A$ $\widehat{y}_{\#}$

$\widehat{x}$ $[r]_B$

$\widehat{y}_{\#}$ $[u]_B$ $[y]_B$

$$g^{-[xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[r]_A}}{\left(h^r\right)^{[u]_A} \cdot \left(g^r \cdot g^x\right)^{[y]_A}}$$

$$\frac{g^{[xy]_B}}{g^{-[xy]_A}} = g^{xy}$$

$$\frac{\left(h^r\right)^{[u]_B} \cdot \left(g^r \cdot g^x\right)^{[y]_B}}{\left(h^u \cdot g^y\right)^{[r]_B}} = g^{[xy]_B}$$

Delegatable Non-Interactive Multiplication

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[r]_A \quad \hat{x}$

$[y]_A \quad [u]_A \quad \hat{y}_{\#}$

$\hat{x} \quad [r]_B$

$\hat{y}_{\#} \quad [u]_B \quad [y]_B$

$$[xy]_A \xleftarrow{\text{DDLog}} g^{-[xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[r]_A}}{\left(h^r\right)^{[u]_A} \cdot \left(g^r \cdot g^x\right)^{[y]_A}}$$

$$\frac{g^{[xy]_B}}{g^{-[xy]_A}} = g^{xy}$$

$$\frac{\left(h^r\right)^{[u]_B} \cdot \left(g^r \cdot g^x\right)^{[y]_B}}{\left(h^u \cdot g^y\right)^{[r]_B}} = g^{[xy]_B} \xrightarrow{\text{DDLog}} [xy]_B$$

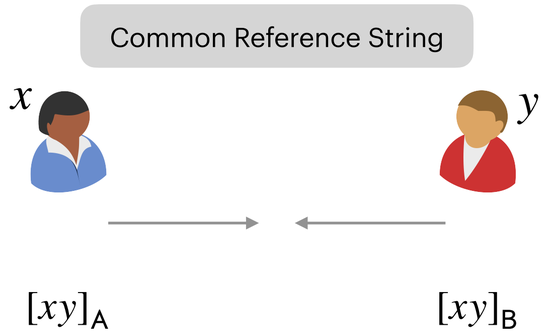
HSS for **Multiplication** is All You Need

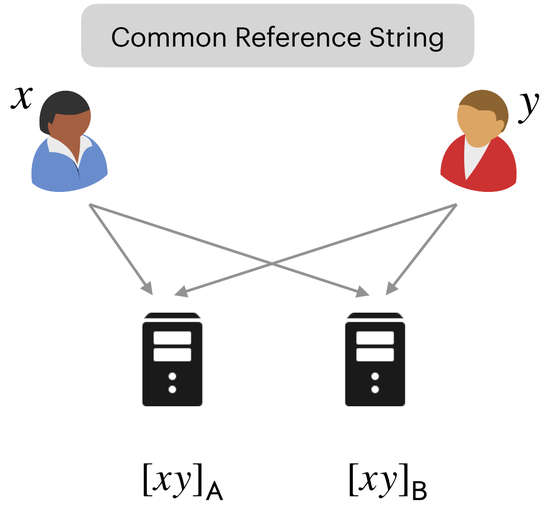
Linear Decoding

Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs





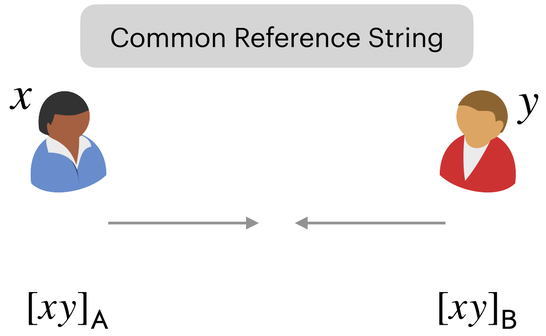
HSS for **Multiplication** is All You Need

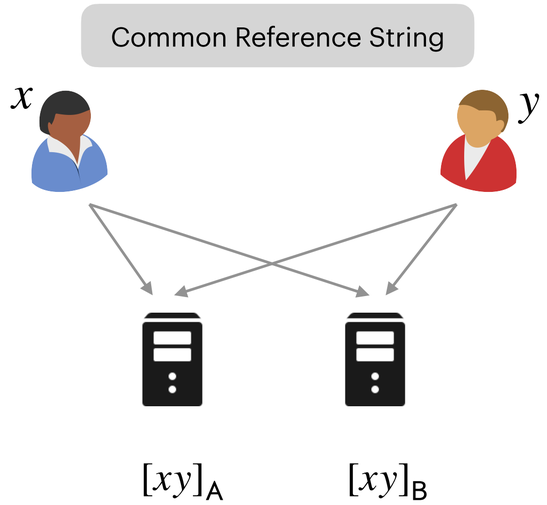
Linear Decoding

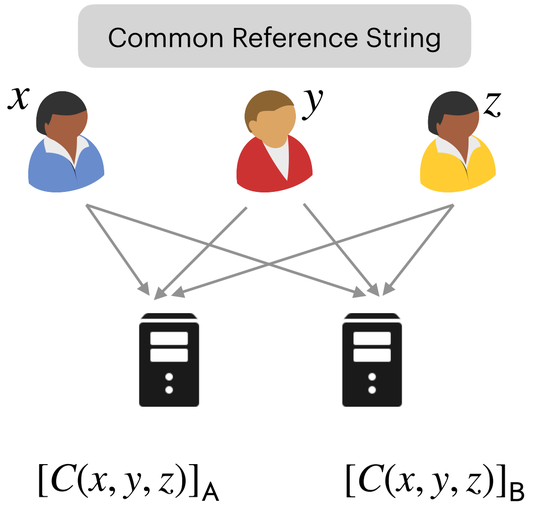
Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs







Towards Evaluating RMS Programs

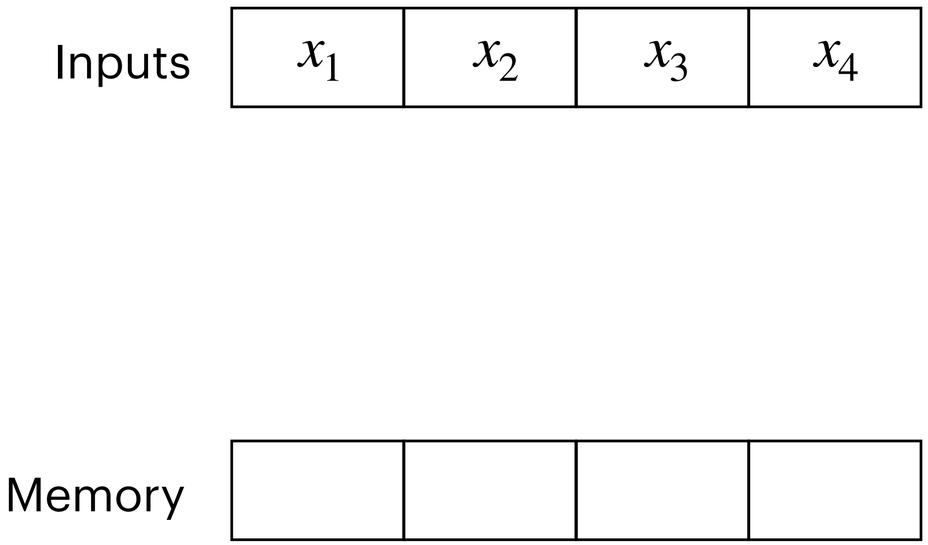
Multiplying inputs with intermediate values of the computation suffices to evaluate RMS programs

NIM can be used to multiply inputs with intermediate values

Restricted Multiplication Straight-line (RMS) Programs

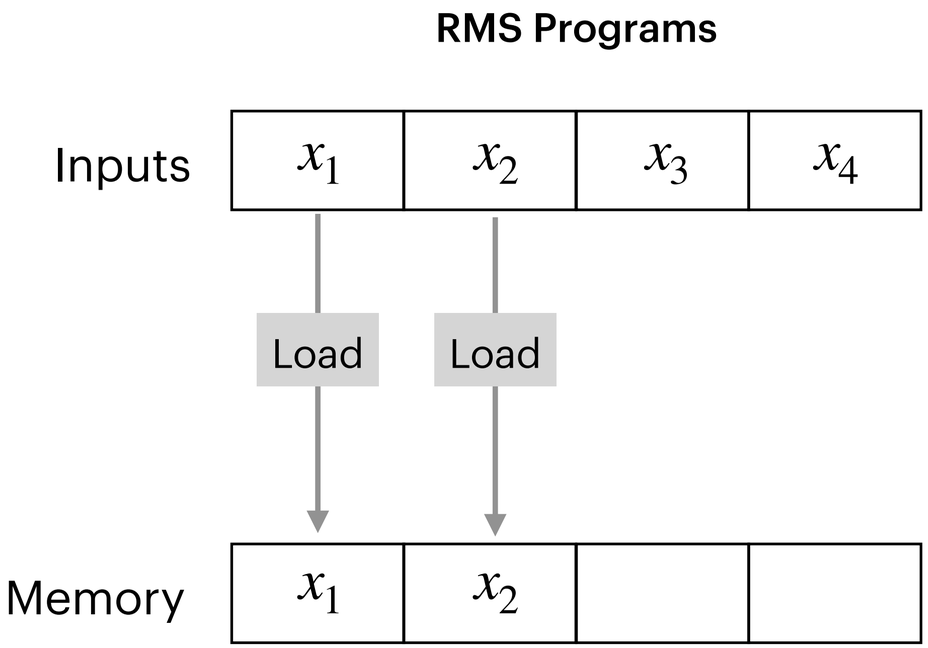
[Boyle-Gilboa-Ishai'16]

RMS Programs



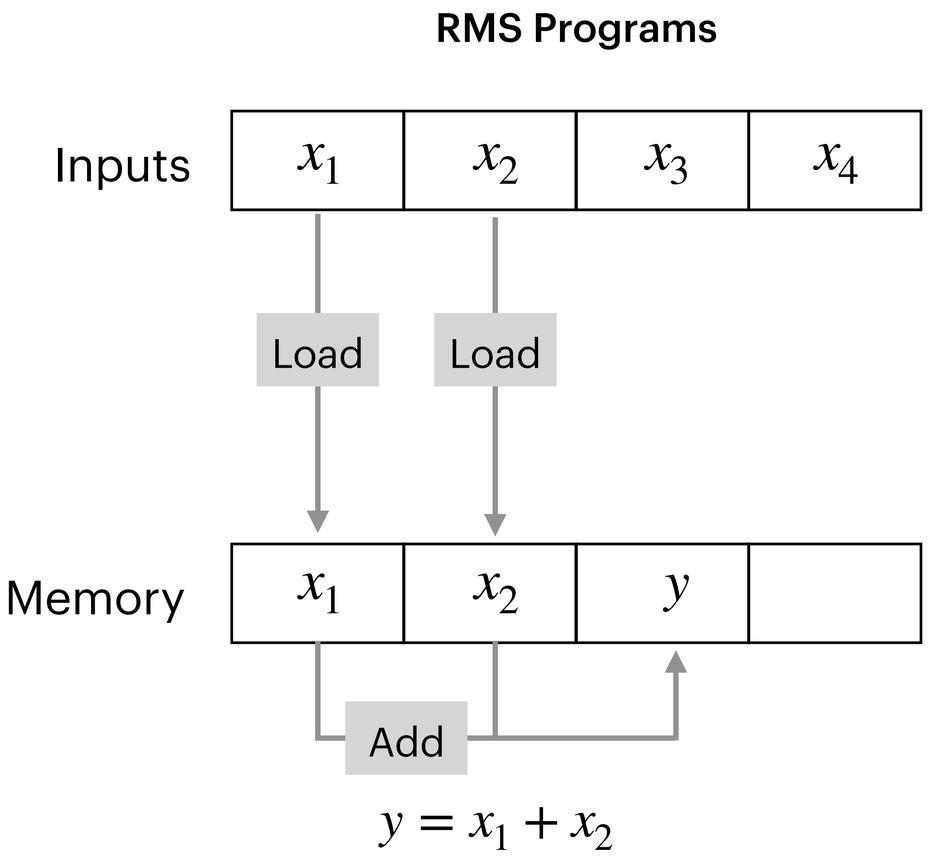
Restricted Multiplication Straight-line (RMS) Programs

[Boyle-Gilboa-Ishai'16]



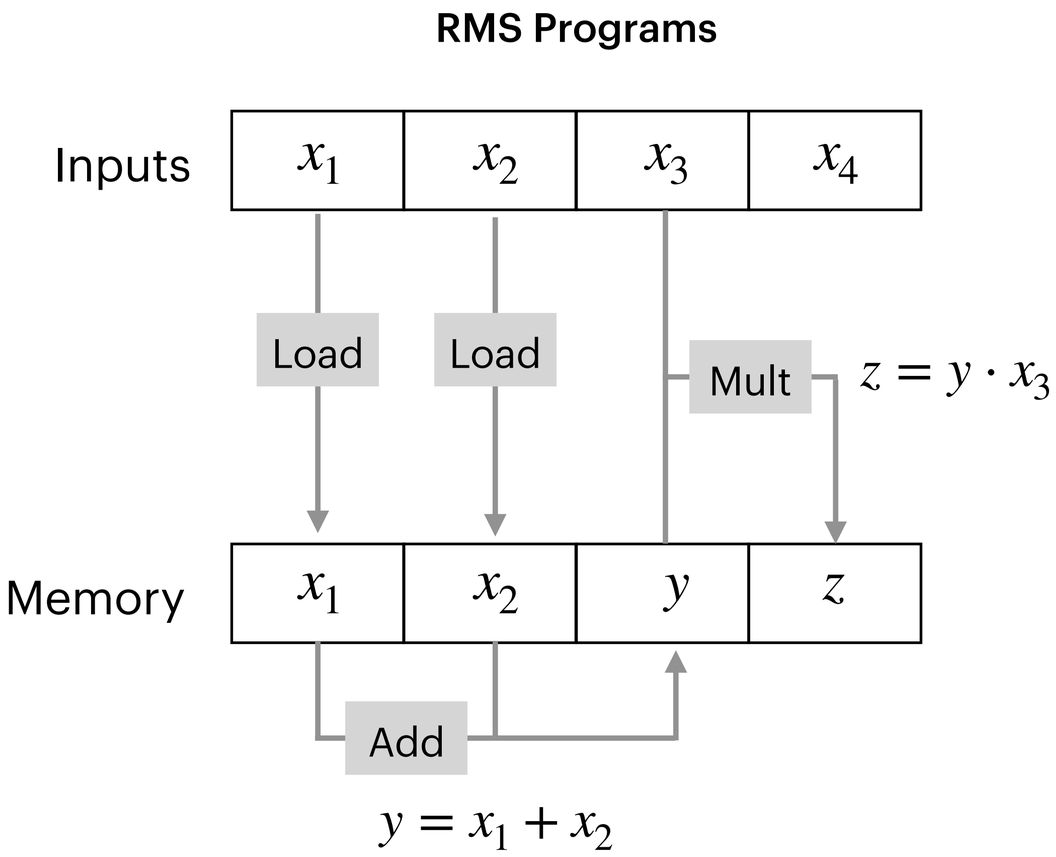
Restricted Multiplication Straight-line (RMS) Programs

[Boyle-Gilboa-Ishai'16]



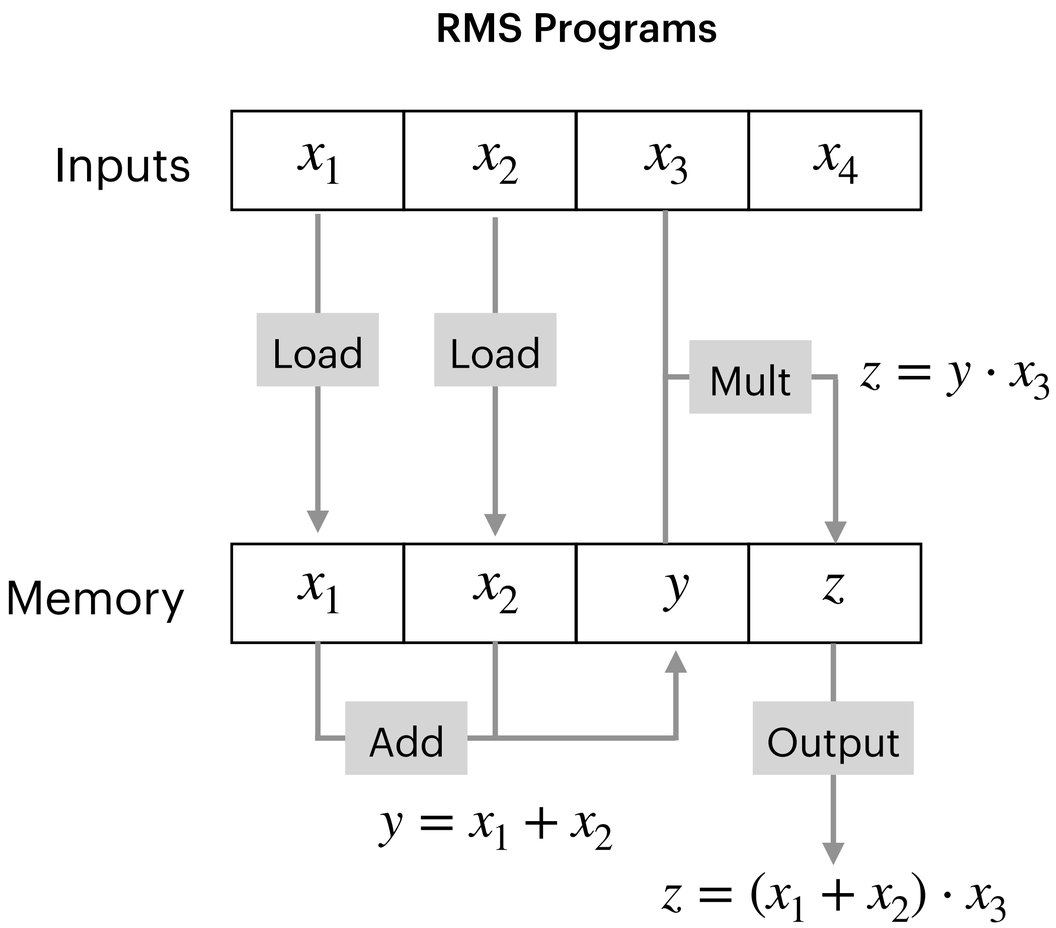
Restricted Multiplication Straight-line (RMS) Programs

[Boyle-Gilboa-Ishai'16]



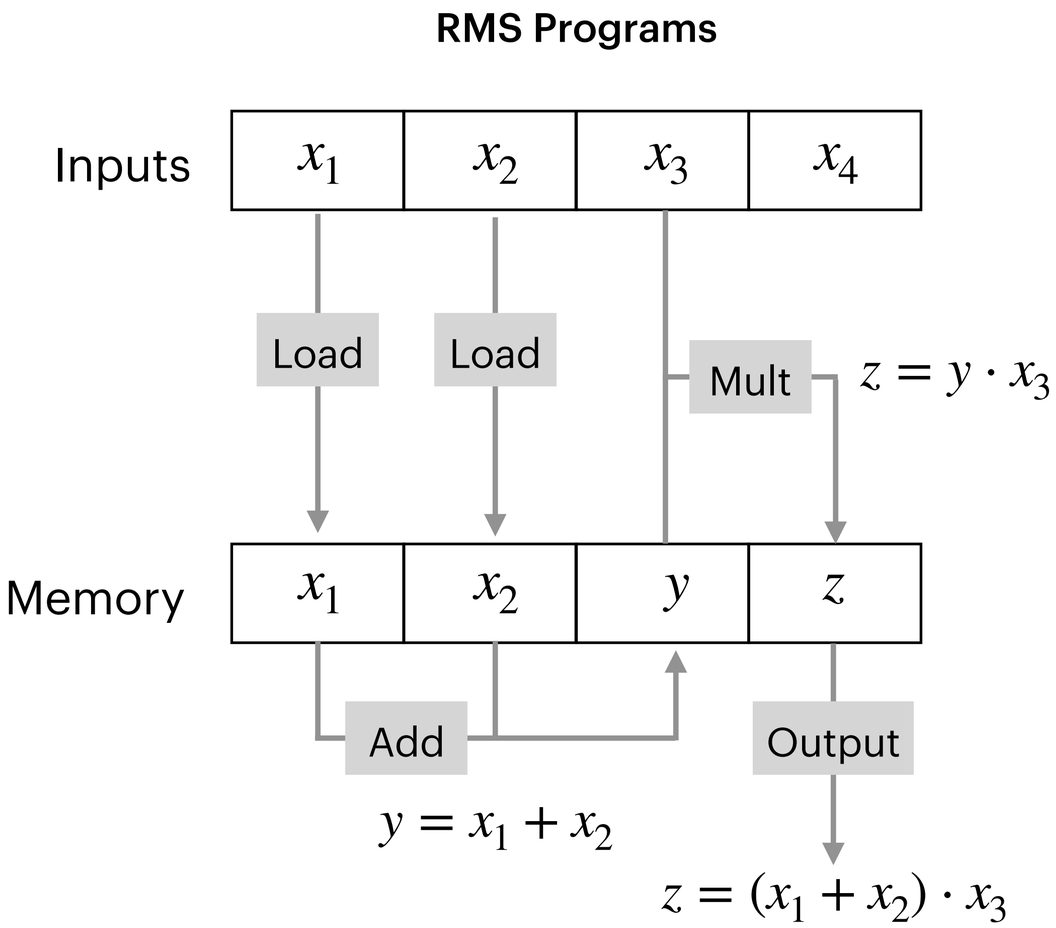
Restricted Multiplication Straight-line (RMS) Programs

[Boyle-Gilboa-Ishai'16]



Restricted Multiplication Straight-line (RMS) Programs

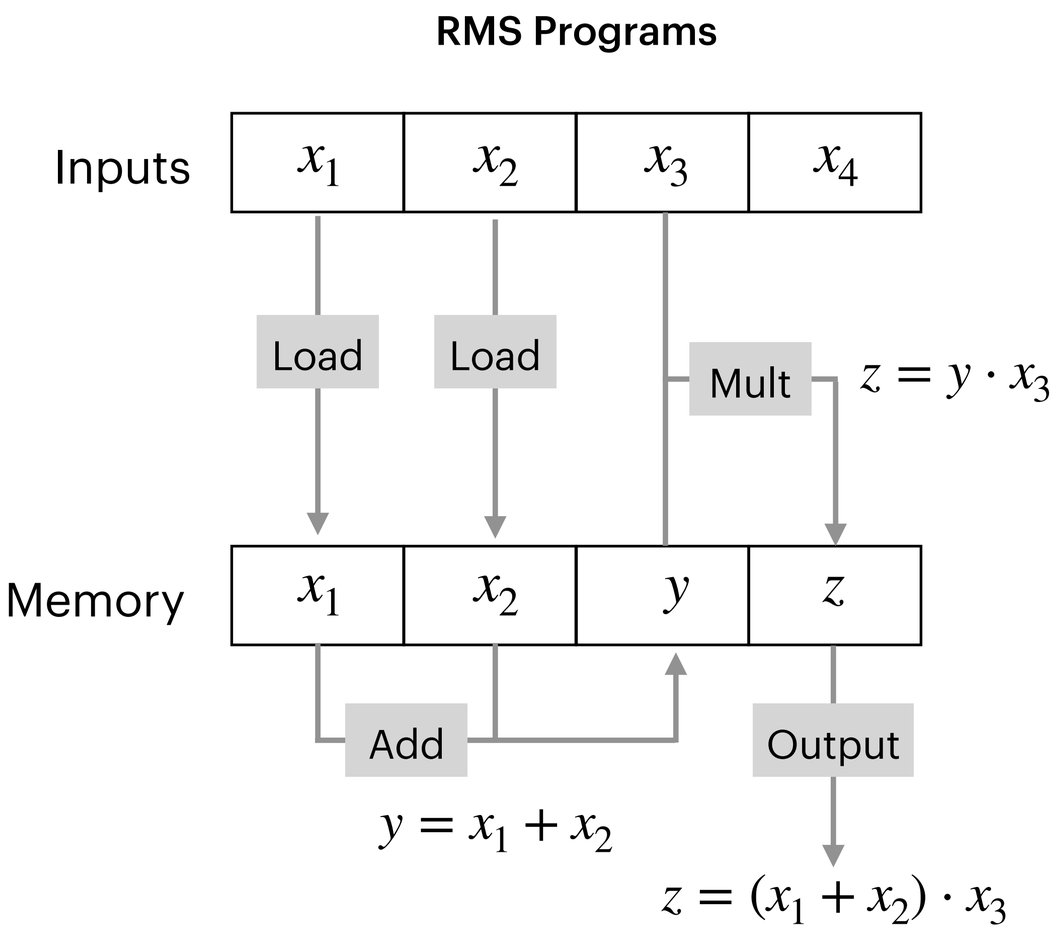
[Boyle-Gilboa-Ishai'16]

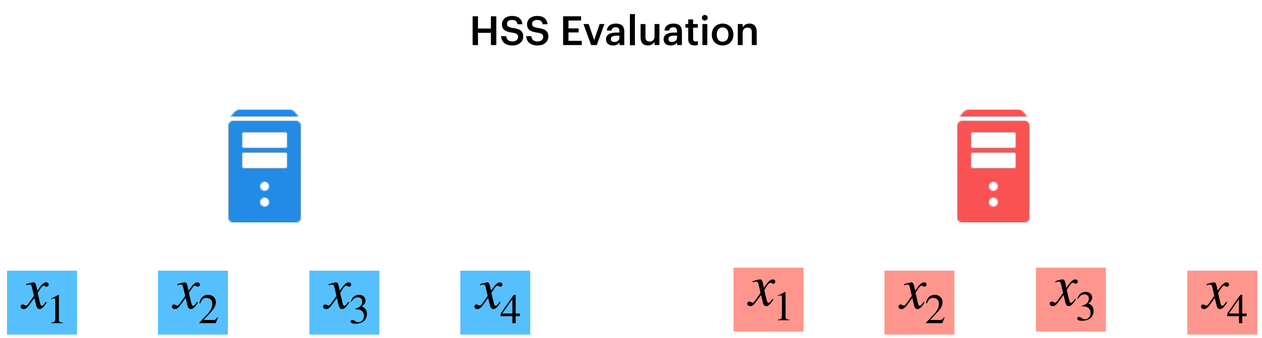


Cannot multiply two memory values

Distributed Evaluation of RMS Programs

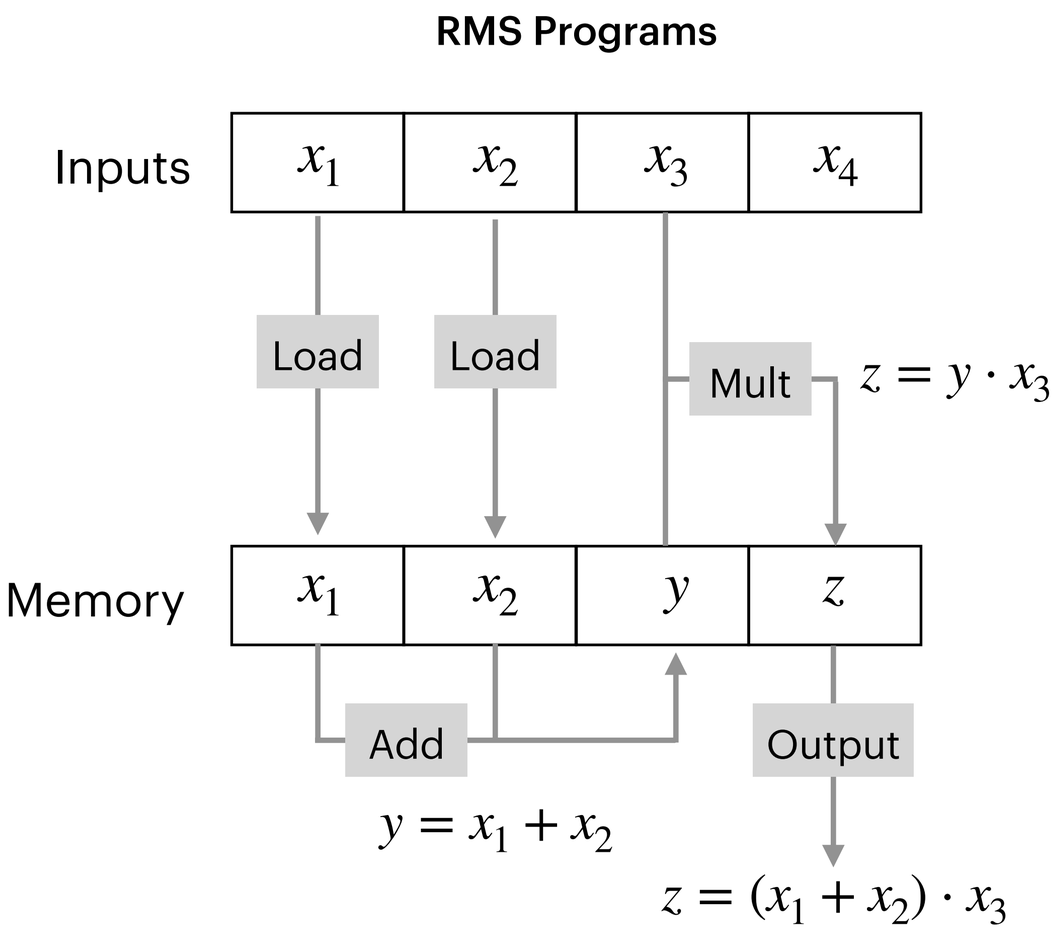
[Boyle-Gilboa-Ishai'16]

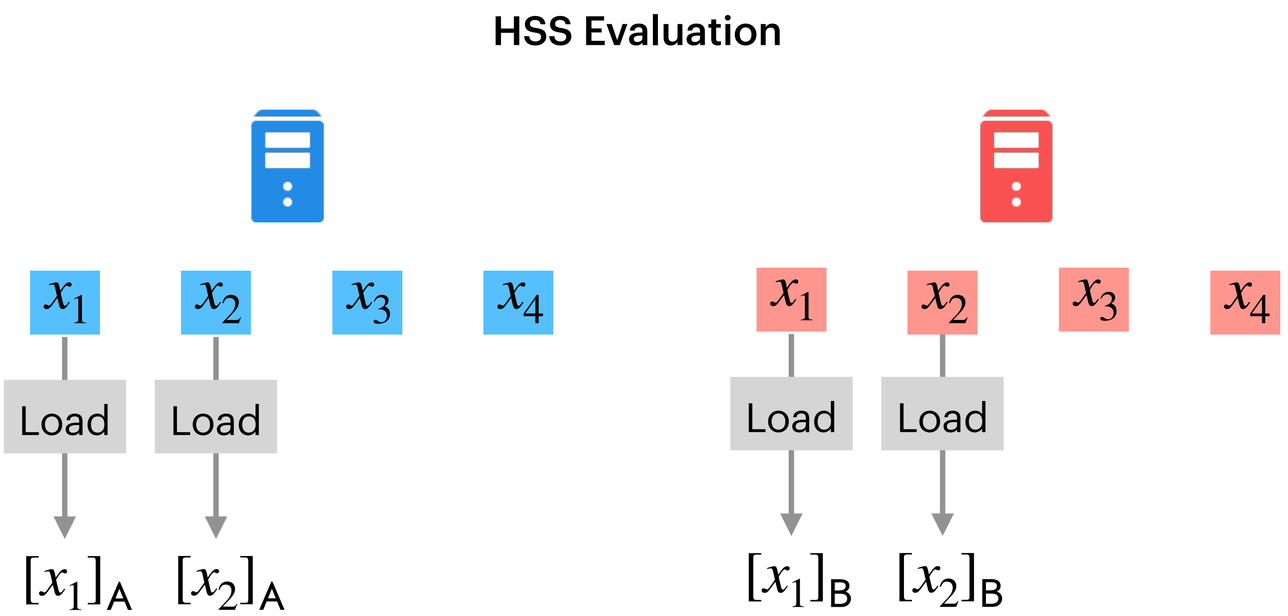




Distributed Evaluation of RMS Programs

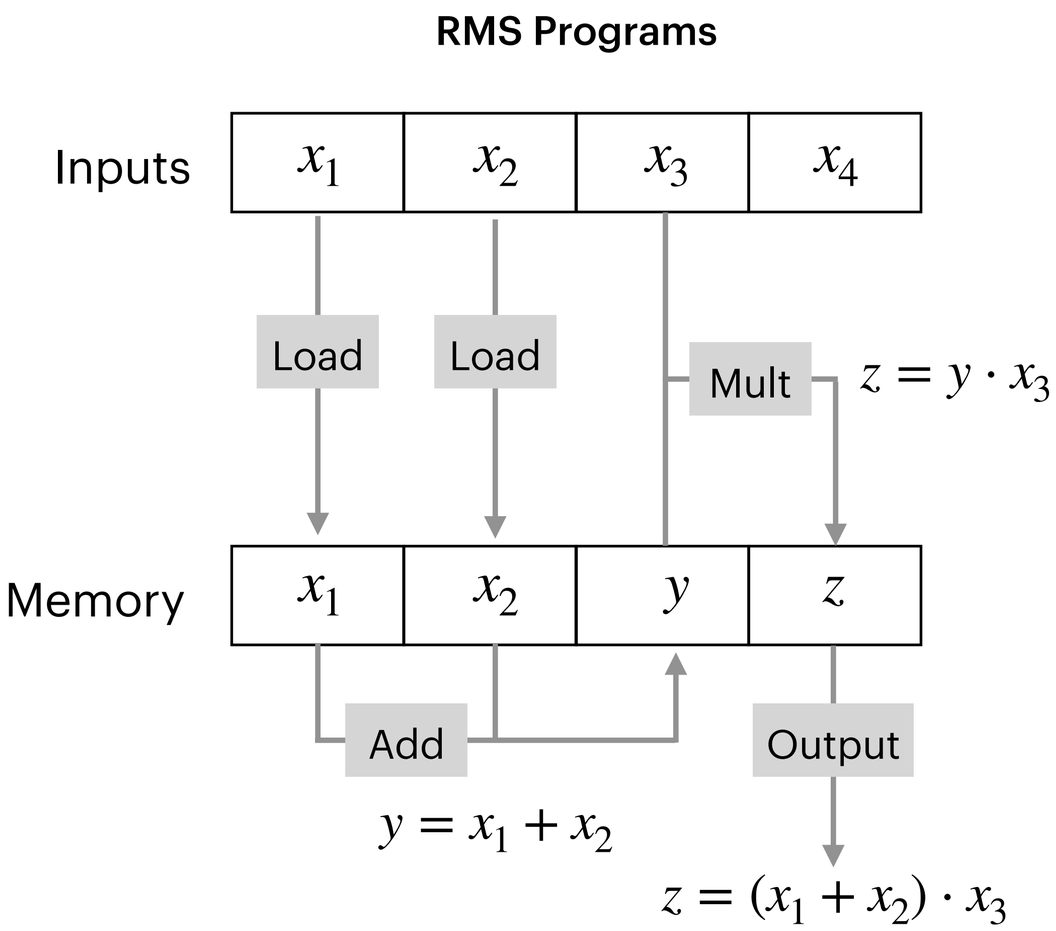
[Boyle-Gilboa-Ishai'16]

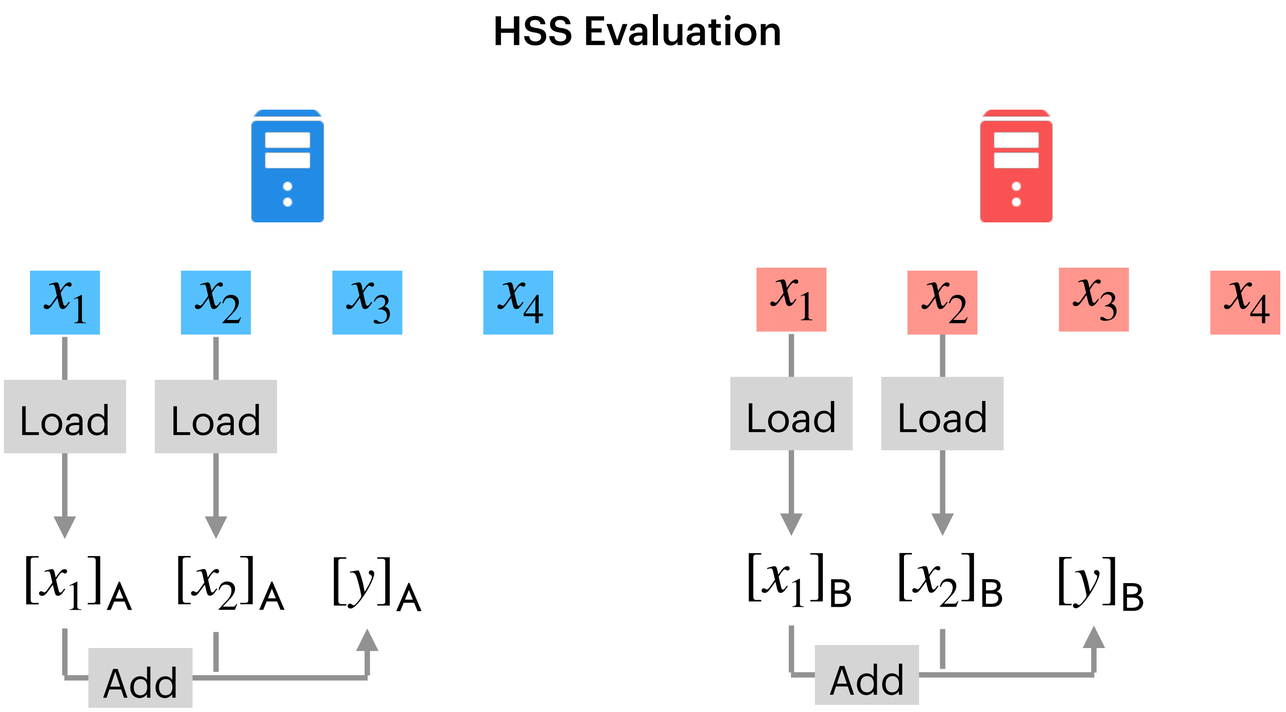




Distributed Evaluation of RMS Programs

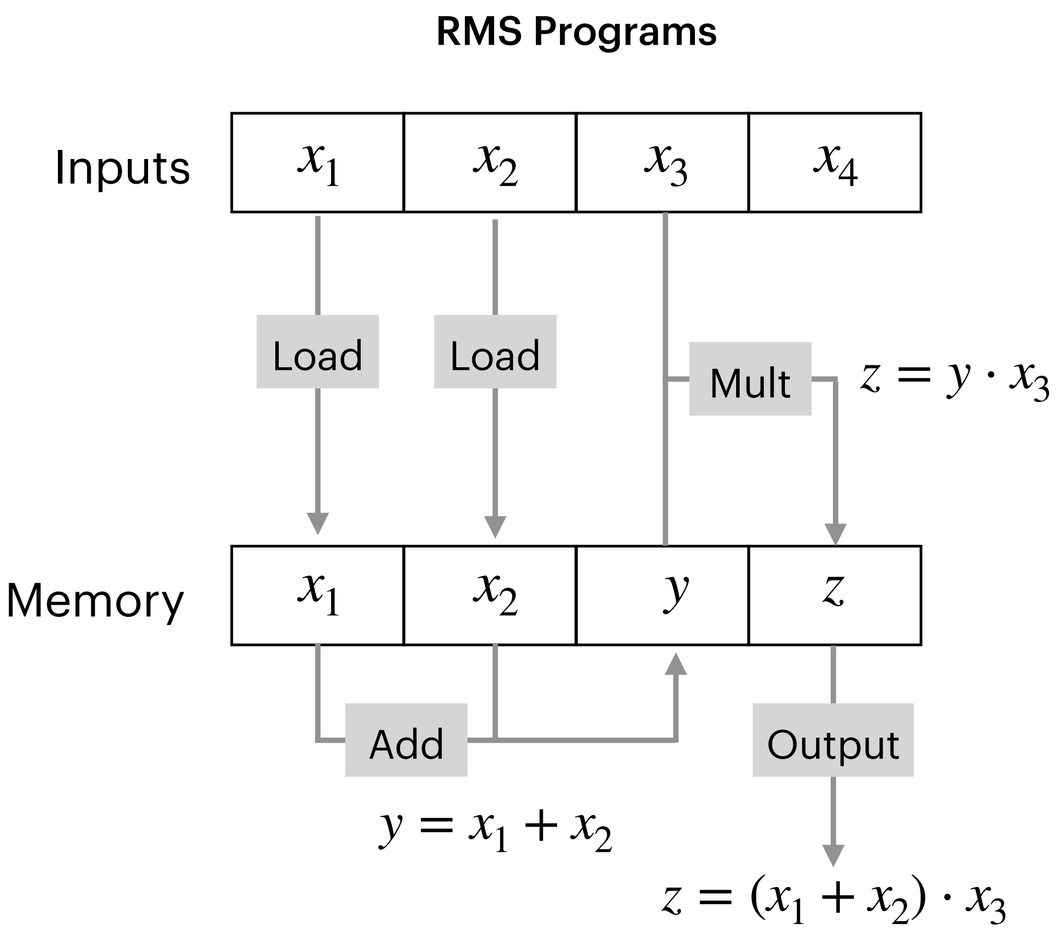
[Boyle-Gilboa-Ishai'16]

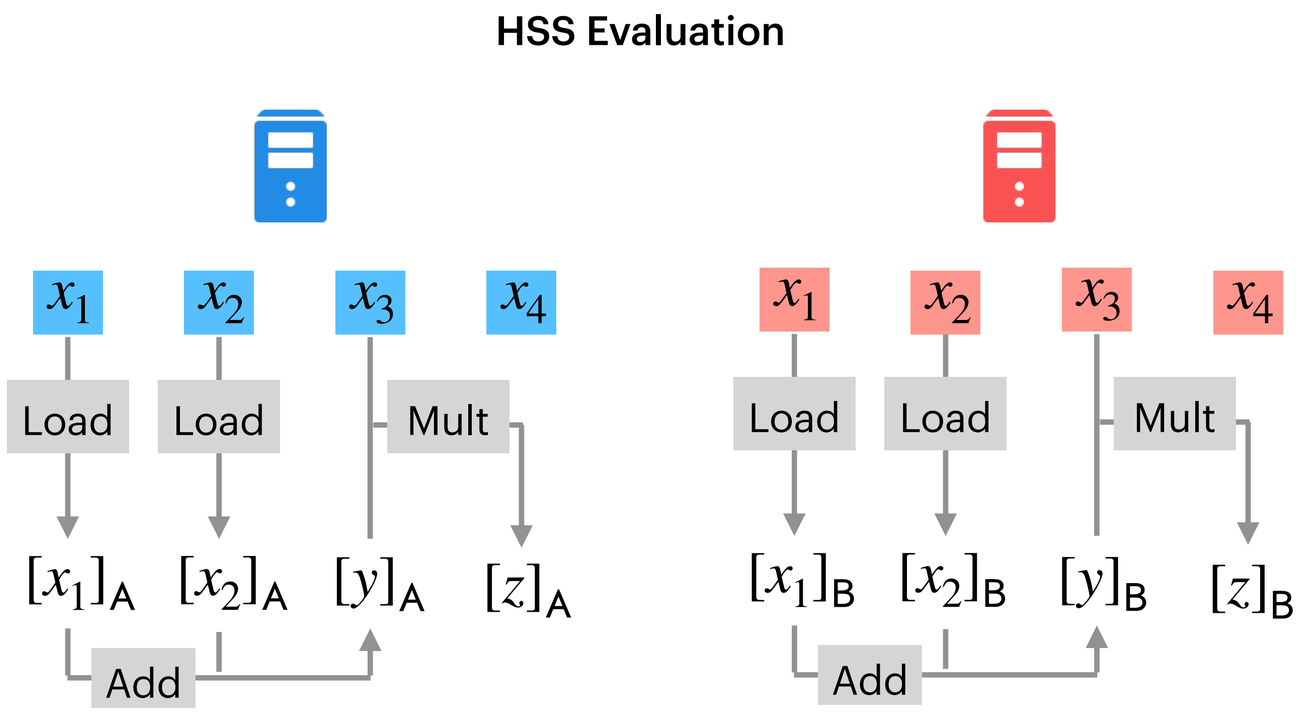




Distributed Evaluation of RMS Programs

[Boyle-Gilboa-Ishai'16]

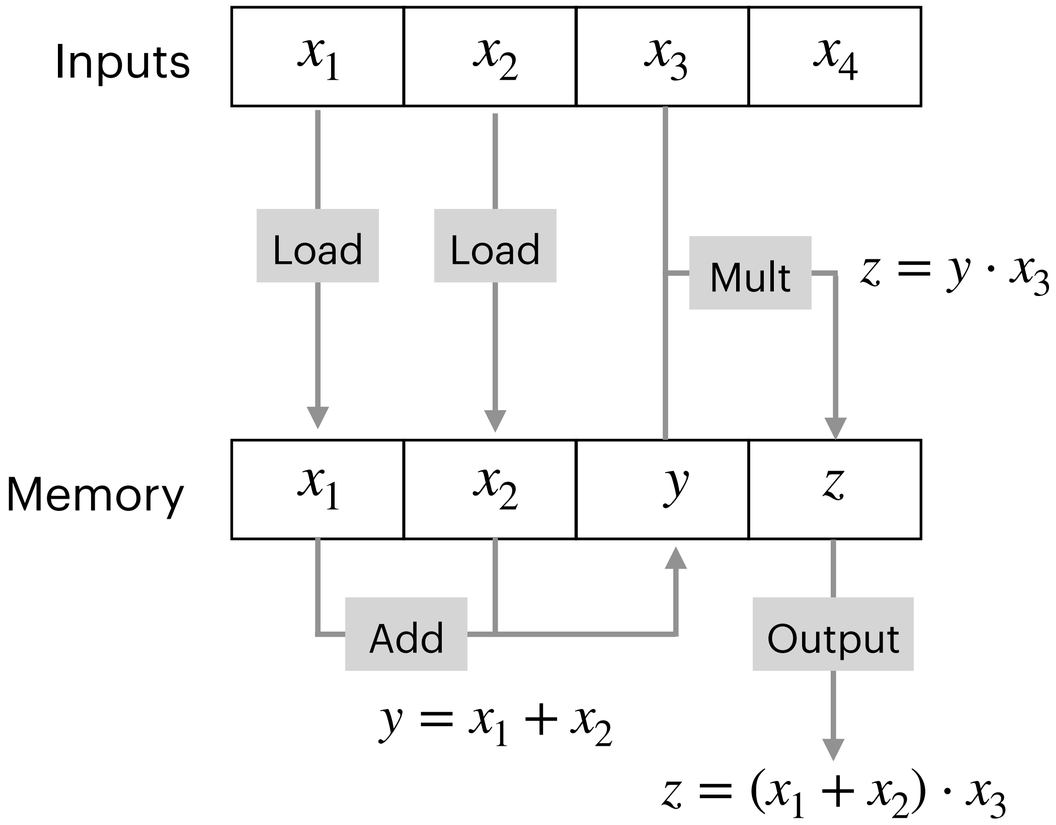




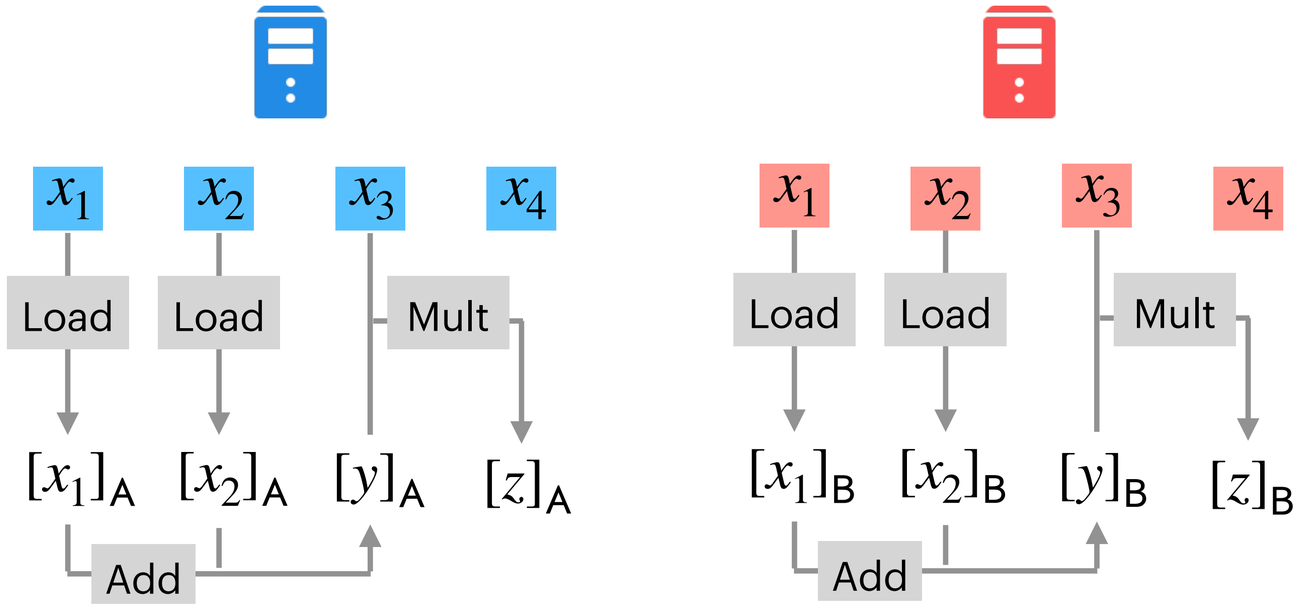
Distributed Evaluation of RMS Programs

[Boyle-Gilboa-Ishai'16]

RMS Programs



HSS Evaluation



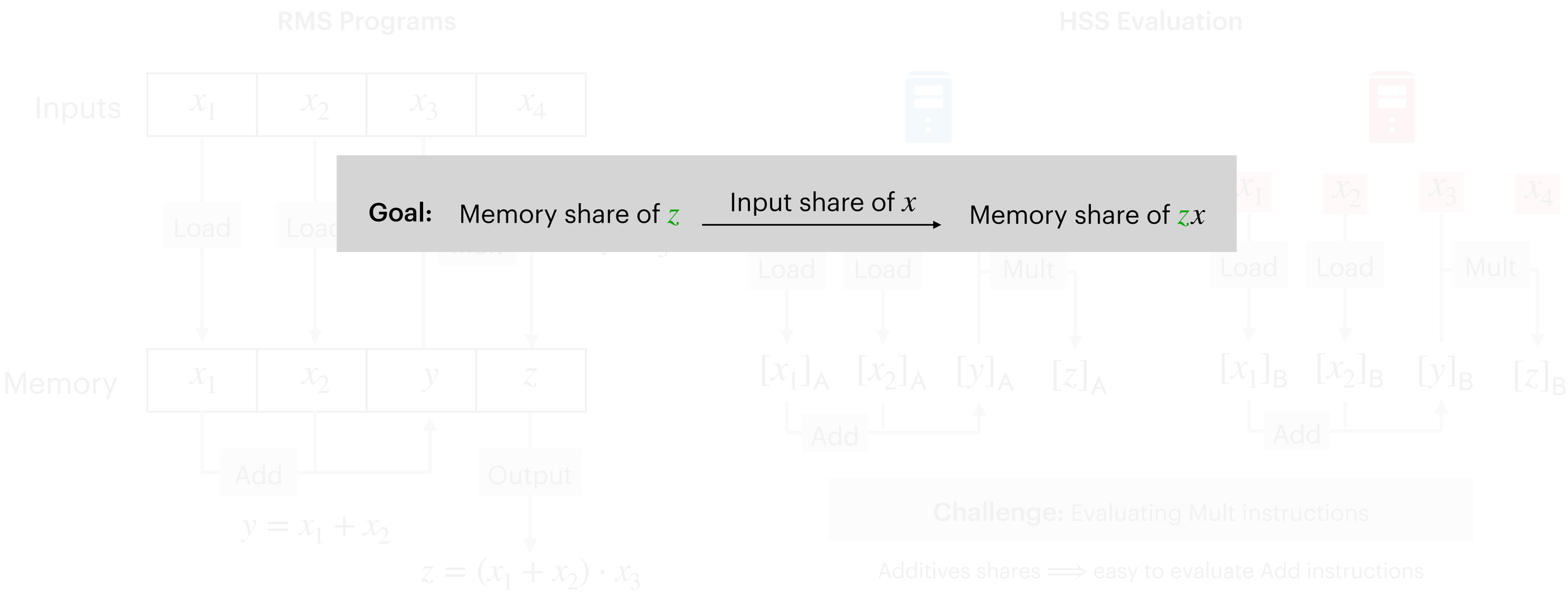
Challenge: Evaluating Mult instructions

Additives shares $\Longrightarrow$ easy to evaluate Add instructions

Load instructions will follow from Mult instructions

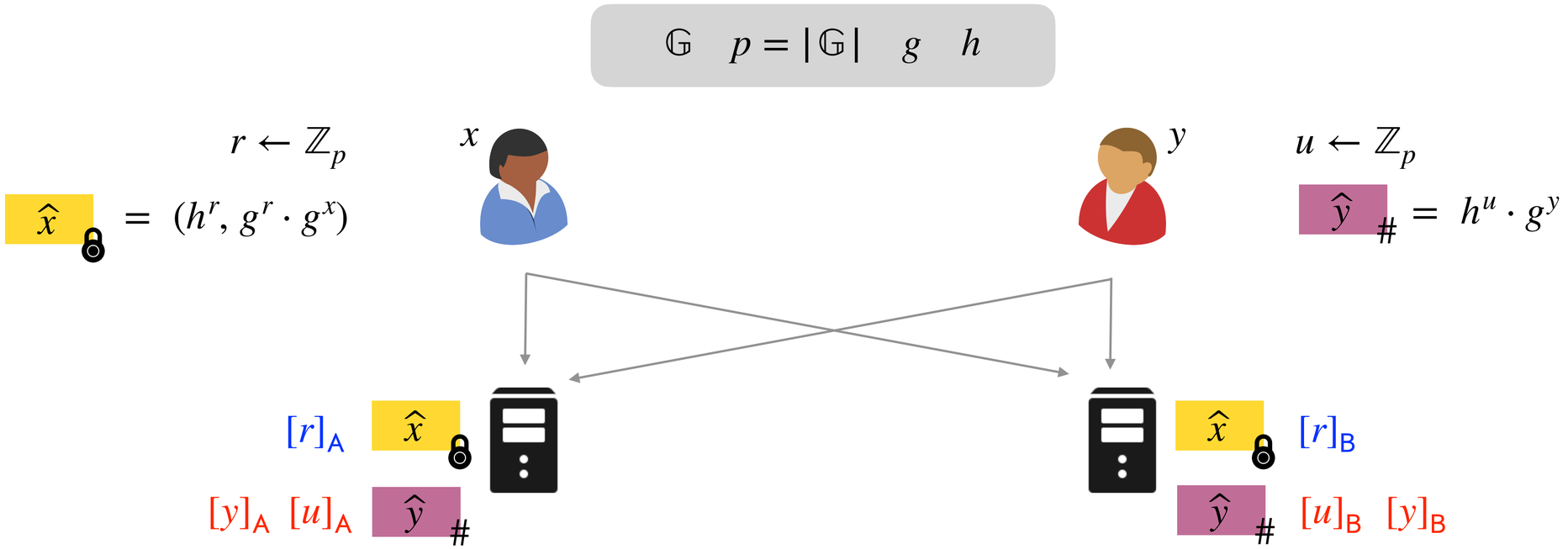
Distributed Evaluation of RMS Programs

[Boyle-Gilboa-Ishai'16]



Goal: Memory share of z —Input share of x→ Memory share of zx

Delegatable Non-Interactive Multiplication



$$[xy]_A \xleftarrow{\text{DDLog}} g^{-[xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[r]_A}}{\left(h^r\right)^{[u]_A} \cdot \left(g^r \cdot g^x\right)^{[y]_A}} \qquad \frac{g^{[xy]_B}}{g^{-[xy]_A}} = g^{xy} \qquad \frac{\left(h^r\right)^{[u]_B} \cdot \left(g^r \cdot g^x\right)^{[y]_B}}{\left(h^u \cdot g^y\right)^{[r]_B}} = g^{[xy]_B} \xrightarrow{\text{DDLog}} [xy]_B$$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[z \cdot r]_A$ $\hat{x}$

$[z \cdot y]_A$ $[z \cdot u]_A$ $\hat{y}_{\#}$

$\hat{x}$ $[z \cdot r]_B$

$\hat{y}_{\#}$ $[z \cdot u]_B$ $[z \cdot y]_B$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

x

y

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[z \cdot r]_A \quad \hat{x}$

$[z \cdot y]_A \quad [z \cdot u]_A \quad \hat{y}_{\#}$

$\hat{x} \quad [z \cdot r]_B$

$\hat{y}_{\#} \quad [z \cdot u]_B \quad [z \cdot y]_B$

$$g^{-[z \cdot xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[z \cdot r]_A}}{\left(h^r\right)^{[z \cdot u]_A} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_A}}$$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

y

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[z \cdot r]_A \quad \hat{x}$

$[z \cdot y]_A \quad [z \cdot u]_A \quad \hat{y}_{\#}$

$\hat{x} \quad [z \cdot r]_B$

$\hat{y}_{\#} \quad [z \cdot u]_B \quad [z \cdot y]_B$

$$g^{-[z \cdot xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[z \cdot r]_A}}{\left(h^r\right)^{[z \cdot u]_A} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_A}}$$

$$\frac{\left(h^r\right)^{[z \cdot u]_B} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_B}}{\left(h^u \cdot g^y\right)^{[z \cdot r]_B}} = g^{[z \cdot xy]_B}$$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$r \leftarrow \mathbb{Z}_p$

$\hat{x} = (h^r, g^r \cdot g^x)$

$u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[z \cdot r]_A \quad \hat{x}$

$[z \cdot y]_A \quad [z \cdot u]_A \quad \hat{y}_{\#}$

$\hat{x} \quad [z \cdot r]_B$

$\hat{y}_{\#} \quad [z \cdot u]_B \quad [z \cdot y]_B$

$$[z \cdot xy]_A \xleftarrow{\text{DDLog}} g^{-[z \cdot xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[z \cdot r]_A}}{\left(h^r\right)^{[z \cdot u]_A} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_A}}$$

$$\frac{\left(h^r\right)^{[z \cdot u]_B} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_B}}{\left(h^u \cdot g^y\right)^{[z \cdot r]_B}} = g^{[z \cdot xy]_B} \xrightarrow{\text{DDLog}} [z \cdot xy]_B$$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$\hat{x}$ $r \leftarrow \text{Encode}(x)$  x

 y $u \leftarrow \mathbb{Z}_p$

$\hat{y}_{\#} = h^u \cdot g^y$

$[z \cdot r]_A$ $\hat{x}$ 

$[z \cdot y]_A$ $[z \cdot u]_A$ $\hat{y}_{\#}$

 $\hat{x}$ $[z \cdot r]_B$

$\hat{y}_{\#}$ $[z \cdot u]_B$ $[z \cdot y]_B$

$$[z \cdot xy]_A \xleftarrow{\text{DDLog}} g^{-[z \cdot xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[z \cdot r]_A}}{\left(h^r\right)^{[z \cdot u]_A} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_A}}$$

$$\frac{\left(h^r\right)^{[z \cdot u]_B} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_B}}{\left(h^u \cdot g^y\right)^{[z \cdot r]_B}} = g^{[z \cdot xy]_B} \xrightarrow{\text{DDLog}} [z \cdot xy]_B$$

Extending Delegatable NIM

$\mathbb{G} \quad p = |\mathbb{G}| \quad g \quad h$

$\hat{x}$ $r \leftarrow \text{Encode}(x)$  x

 y $\text{Hash}(y) \rightarrow u$ $\hat{y}$ #

$[z \cdot r]_A$ $\hat{x}$

$[z \cdot y]_A$ $[z \cdot u]_A$ $\hat{y}$ #

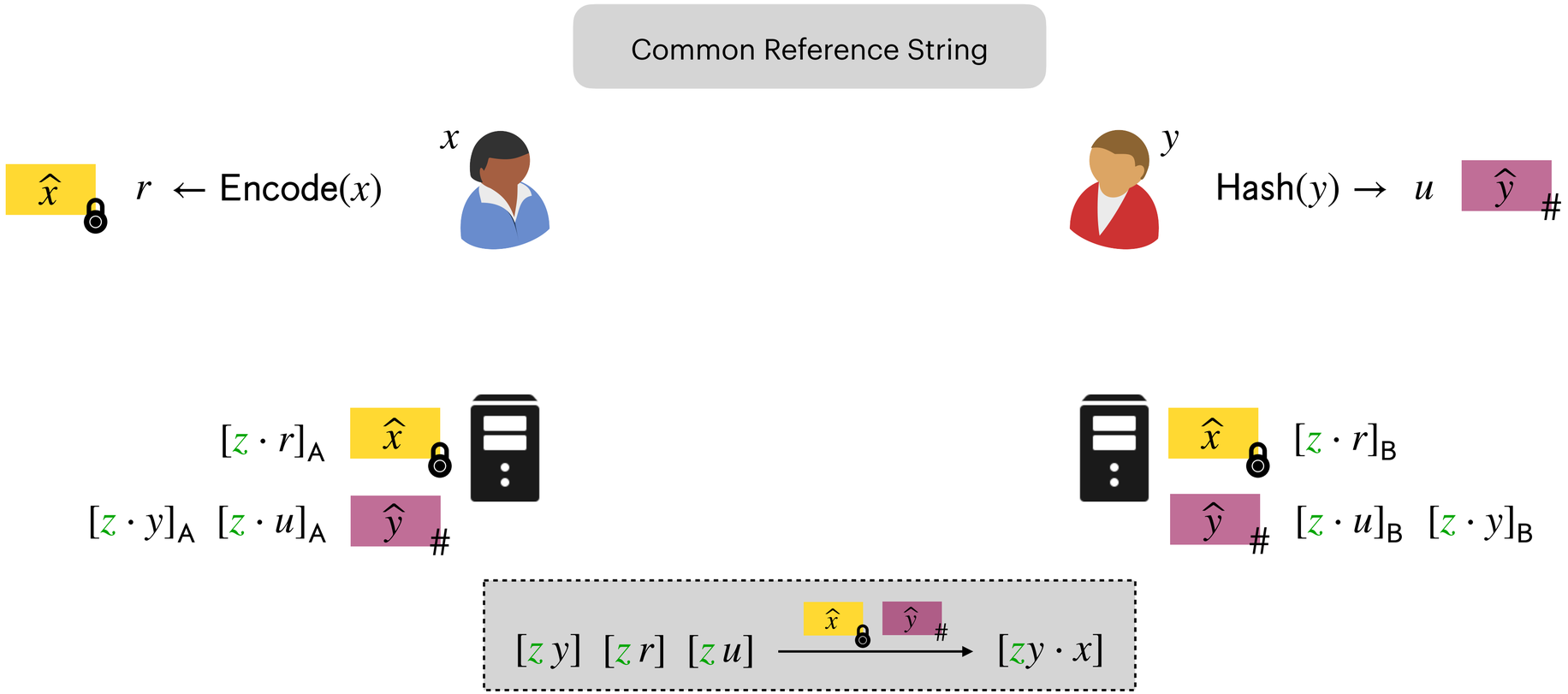
$\hat{x}$ $[z \cdot r]_B$

$\hat{y}$ # $[z \cdot u]_B$ $[z \cdot y]_B$

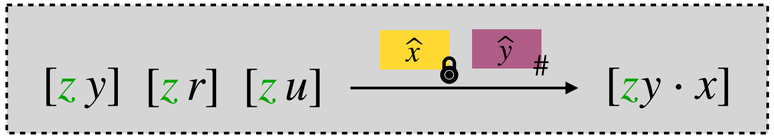
$$[z \cdot xy]_A \xleftarrow{\text{DDLog}} g^{-[z \cdot xy]_A} = \frac{\left(h^u \cdot g^y\right)^{[z \cdot r]_A}}{\left(h^r\right)^{[z \cdot u]_A} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_A}}$$

$$\frac{\left(h^r\right)^{[z \cdot u]_B} \cdot \left(g^r \cdot g^x\right)^{[z \cdot y]_B}}{\left(h^u \cdot g^y\right)^{[z \cdot r]_B}} = g^{[z \cdot xy]_B} \xrightarrow{\text{DDLog}} [z \cdot xy]_B$$

Extending Delegatable NIM



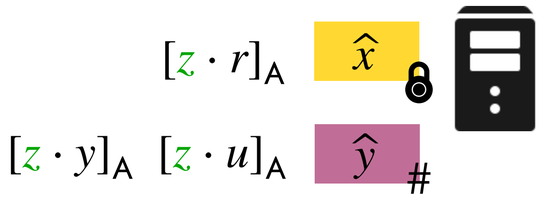
Extending Delegatable NIM

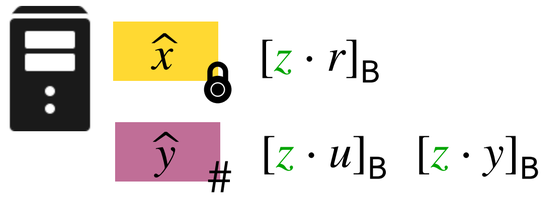


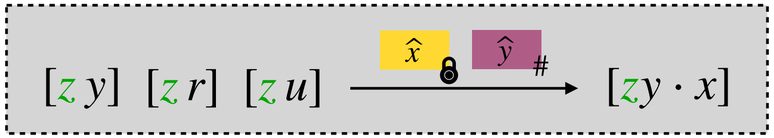
Common Reference String

$\hat{x}$ $r \leftarrow \text{Encode}(x)$

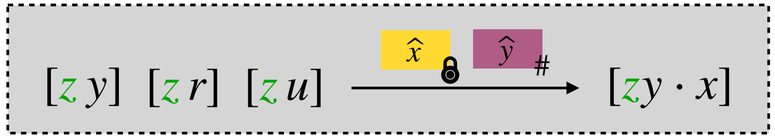
$\text{Hash}(y) \rightarrow u$ $\hat{y}$ #



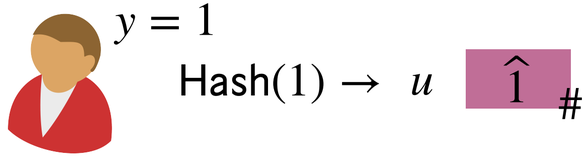


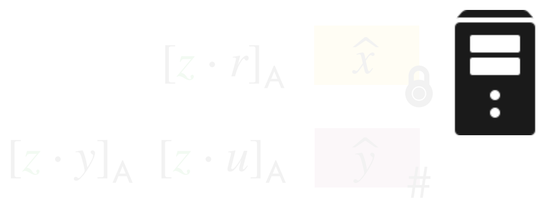


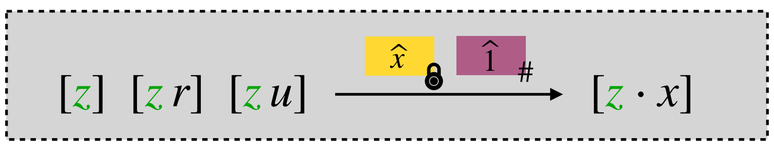
Extending Delegatable NIM



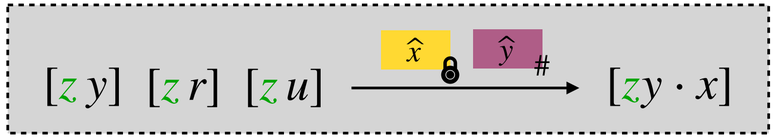
Common Reference String

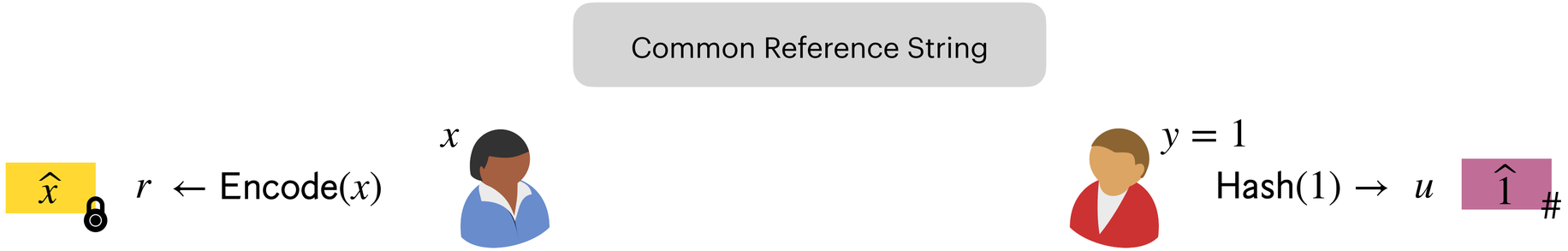


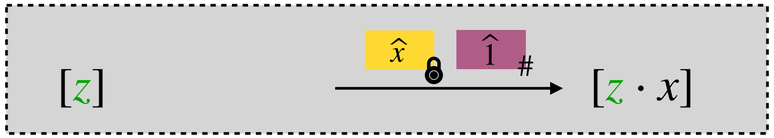




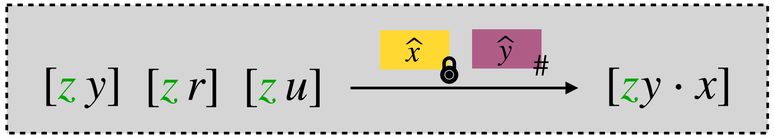
Extending Delegatable NIM



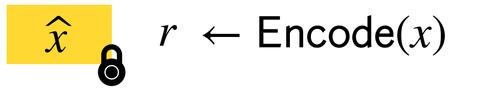


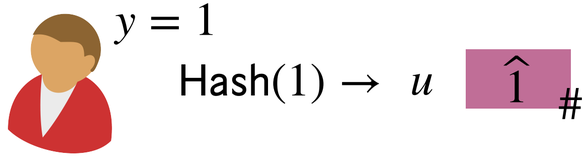


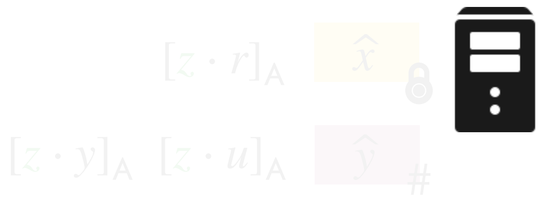
Extending Delegatable NIM

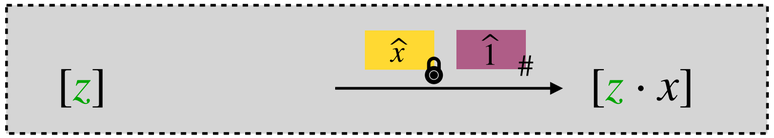


Common Reference String

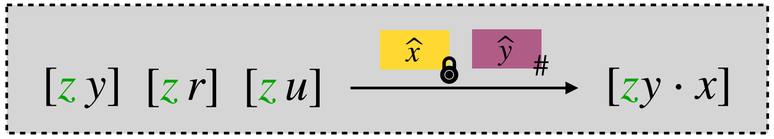






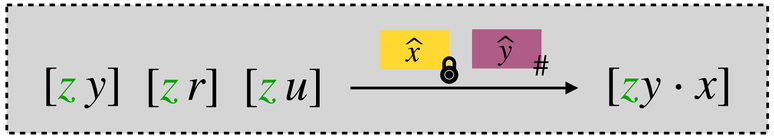


Goal: Memory share of z —Input share of x→ Memory share of zx



Attempt at Evaluating RMS Programs

Common Reference String

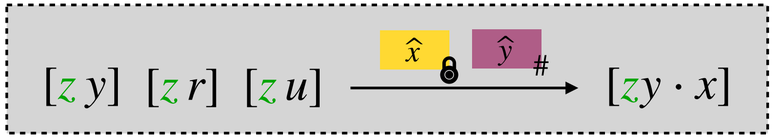


Attempt at Evaluating RMS Programs

Common Reference String

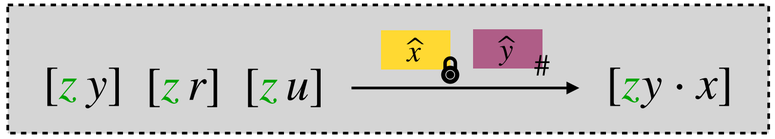
Defined in CRS

$\text{Hash}(1) \rightarrow u$ $\hat{1}$ #



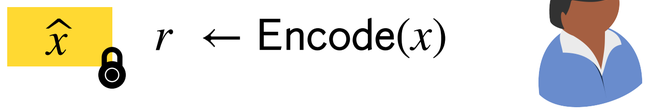
Attempt at Evaluating RMS Programs

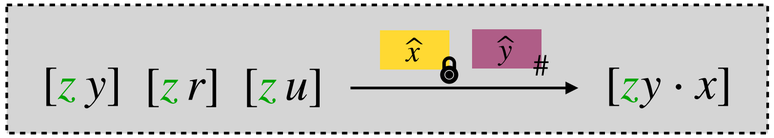
Common Reference String



Attempt at Evaluating RMS Programs

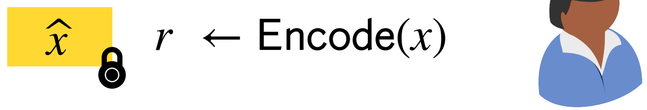
Common Reference String



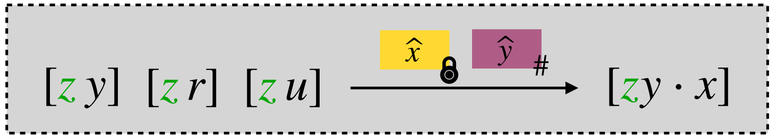


Attempt at Evaluating RMS Programs

Common Reference String

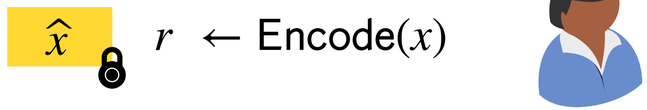


Memory share of z: $[z]$ $[z \cdot r]$

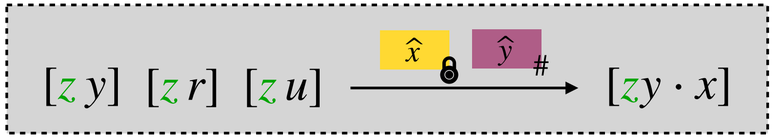


Attempt at Evaluating RMS Programs

Common Reference String

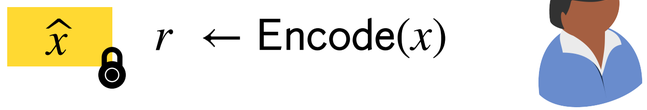


Memory share of z: $[z]$ $[z \cdot r]$

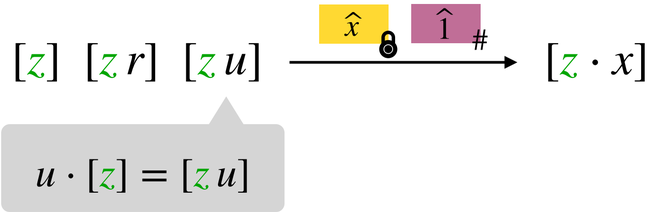


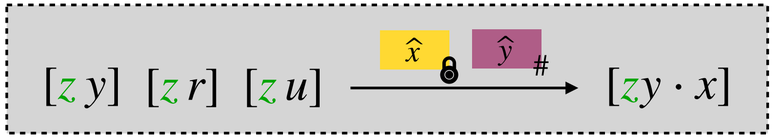
Attempt at Evaluating RMS Programs

Common Reference String



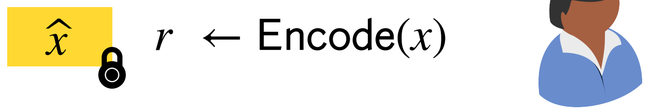
Memory share of z: $[z]$ $[z \cdot r]$





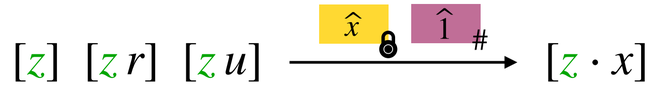
Attempt at Evaluating RMS Programs

Common Reference String

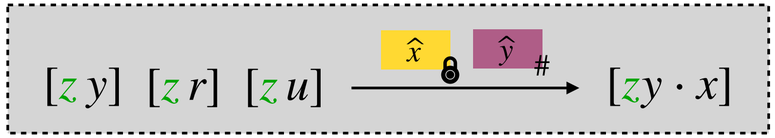


Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$



$u \cdot [z] = [z\,u]$

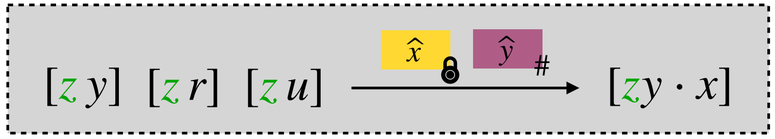


Attempt at Evaluating RMS Programs

Common Reference String

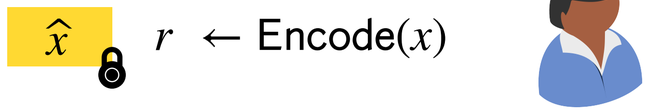
Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$



Attempt at Evaluating RMS Programs

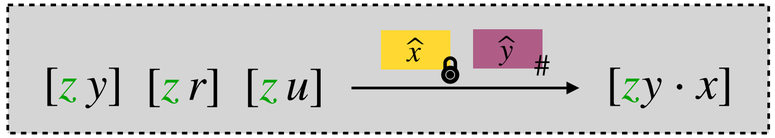
Common Reference String



Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$

Need $[zx \cdot r]$ for subsequent multiplications

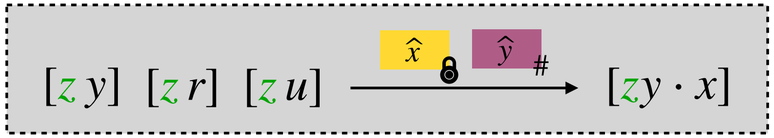


Attempt at Evaluating RMS Programs

Common Reference String

Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$



Attempt at Evaluating RMS Programs

Common Reference String

$\widehat{x}$ $r \leftarrow \text{Encode}(x)$

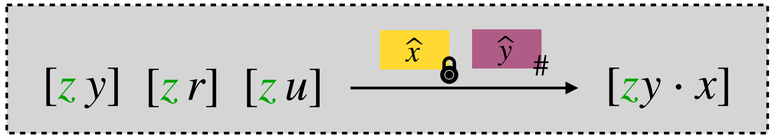
$\widehat{r}$ # $s \leftarrow \text{Hash}(r)$

u $\widehat{1}$ #

$\widehat{1}$ # u

Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$



Attempt at Evaluating RMS Programs

Common Reference String

$\widehat{x}$ $r \leftarrow \text{Encode}(x)$

$\widehat{r}$ # $s \leftarrow \text{Hash}(r)$

Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$

Attempt at Evaluating RMS Programs

$[z\,y]\ \ [z\,r]\ \ [z\,u] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

Common Reference String

$\widehat{x}$ $r \leftarrow \text{Encode}(x)$

$\widehat{r}_{\#}$ $s \leftarrow \text{Hash}(r)$

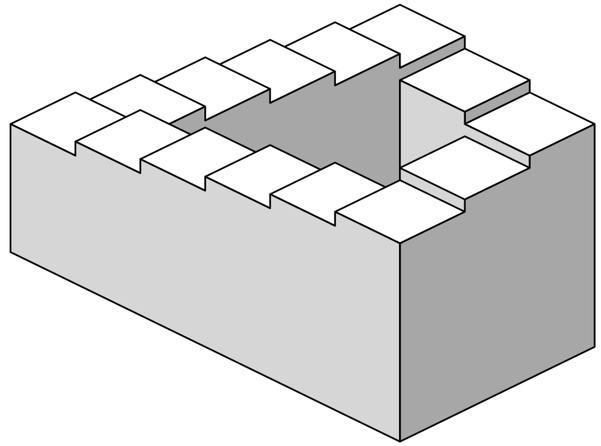
u $\widehat{1}_{\#}$

$\widehat{1}_{\#}$ u

Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$

$[z\,r]\ \ [z\,r]\ \ [z\,s] \xrightarrow{\widehat{x}\ \ \widehat{r}\ \#} [zr \cdot x]$



$[z\,y]\ \ [z\,r]\ \ [z\,u] \xrightarrow{\widehat{x}\ \ \widehat{y}^{\#}} [zy \cdot x]$

Attempt at Evaluating RMS Programs

Common Reference String

$\widehat{x}$ $r \leftarrow \text{Encode}(x)$

$\widehat{r}^{\#}$ $s \leftarrow \text{Hash}(r)$

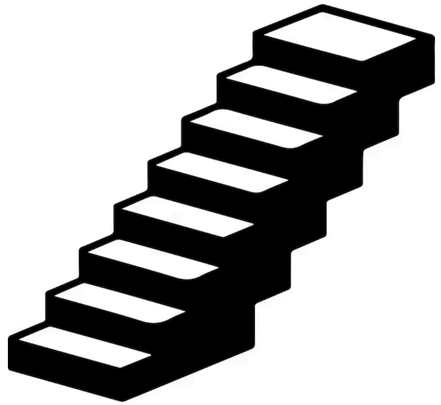
u $\widehat{1}^{\#}$

$\widehat{1}^{\#}$ u

Memory share of z: $[z]$ $[z \cdot r]$

Memory share of zx: $[zx]$ $[zx \cdot r]$

$[z\,r]\ \ [z\,r]\ \ [z\,s] \xrightarrow{\widehat{x}\ \ \widehat{r}^{\#}} [zr \cdot x]$



Solution: Encryption scheme with linear decryption

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}}$ $\mathsf{sk} \leftarrow \mathbb{Z}_p$

x

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}} \qquad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right) \qquad r \leftarrow \mathbb{Z}_p$

x

Encryption with Linear Decryption

$\mathbb{G}$ $p = |\mathbb{G}|$ g

$\mathsf{pk} = g^{-\mathsf{sk}}$ $\quad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right)$ $\quad r \leftarrow \mathbb{Z}_p$

Decryption is "linear": $\left(g^r\right)^{\mathsf{sk}} \cdot \mathsf{pk}^r \cdot g^x = g^{r \cdot \mathsf{sk}} \cdot g^{-r \cdot \mathsf{sk}} \cdot g^x = g^x$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}} \qquad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right) \qquad r \leftarrow \mathbb{Z}_p$

Decryption is "linear": $\left(g^r\right)^{\mathsf{sk}} \cdot \mathsf{pk}^r \cdot g^x = g^{r \cdot \mathsf{sk}} \cdot g^{-r \cdot \mathsf{sk}} \cdot g^x = g^x$

$[z]_A \quad [z \cdot \mathsf{sk}]_A \quad \mathsf{ct}_x$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_B \quad [z]_B$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}} \qquad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right) \qquad r \leftarrow \mathbb{Z}_p$

Decryption is "linear": $\left(g^r\right)^{\mathsf{sk}} \cdot \mathsf{pk}^r \cdot g^x = g^{r \cdot \mathsf{sk}} \cdot g^{-r \cdot \mathsf{sk}} \cdot g^x = g^x$

$[z]_\mathsf{A} \quad [z \cdot \mathsf{sk}]_\mathsf{A} \quad \mathsf{ct}_x$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_\mathsf{B} \quad [z]_\mathsf{B}$

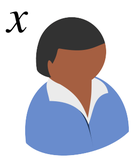
$g^{[z \cdot x]_\mathsf{A}} = \left(g^r\right)^{[z \cdot \mathsf{sk}]_\mathsf{A}} \cdot \left(\mathsf{pk}^r \cdot g^x\right)^{[z]_\mathsf{A}}$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}} \qquad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right) \qquad r \leftarrow \mathbb{Z}_p$



Decryption is "linear": $\left(g^r\right)^{\mathsf{sk}} \cdot \mathsf{pk}^r \cdot g^x = g^{r \cdot \mathsf{sk}} \cdot g^{-r \cdot \mathsf{sk}} \cdot g^x = g^x$

$[z]_\mathsf{A} \quad [z \cdot \mathsf{sk}]_\mathsf{A} \quad \mathsf{ct}_x$

$g^{[z \cdot x]_\mathsf{A}} = \left(g^r\right)^{[z \cdot \mathsf{sk}]_\mathsf{A}} \cdot \left(\mathsf{pk}^r \cdot g^x\right)^{[z]_\mathsf{A}}$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_\mathsf{B} \quad [z]_\mathsf{B}$

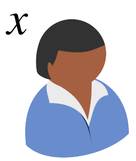
$\left(g^r\right)^{-[z \cdot \mathsf{sk}]_\mathsf{B}} \cdot \left(\mathsf{pk}^r \cdot g^x\right)^{-[z]_\mathsf{B}} = g^{-[z \cdot x]_\mathsf{B}}$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

$\mathsf{pk} = g^{-\mathsf{sk}} \qquad \mathsf{sk} \leftarrow \mathbb{Z}_p$

$\mathsf{ct}_x = \left(g^r, \mathsf{pk}^r \cdot g^x\right) \qquad r \leftarrow \mathbb{Z}_p$



Decryption is "linear": $\left(g^r\right)^{\mathsf{sk}} \cdot \mathsf{pk}^r \cdot g^x = g^{r \cdot \mathsf{sk}} \cdot g^{-r \cdot \mathsf{sk}} \cdot g^x = g^x$

$[z]_\mathsf{A} \quad [z \cdot \mathsf{sk}]_\mathsf{A} \quad \mathsf{ct}_x$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_\mathsf{B} \quad [z]_\mathsf{B}$

$[z \cdot x]_\mathsf{A} \xleftarrow{\text{DDLog}} g^{[z \cdot x]_\mathsf{A}} = \left(g^r\right)^{[z \cdot \mathsf{sk}]_\mathsf{A}} \cdot \left(\mathsf{pk}^r \cdot g^x\right)^{[z]_\mathsf{A}}$

$\left(g^r\right)^{-[z \cdot \mathsf{sk}]_\mathsf{B}} \cdot \left(\mathsf{pk}^r \cdot g^x\right)^{-[z]_\mathsf{B}} = g^{-[z \cdot x]_\mathsf{B}} \xrightarrow{\text{DDLog}} [z \cdot x]_\mathsf{B}$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

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

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

$[z]_\mathsf{A} \quad [z \cdot \mathsf{sk}]_\mathsf{A} \quad \mathsf{ct}_x$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_\mathsf{B} \quad [z]_\mathsf{B}$

$[z] \quad [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Encryption with Linear Decryption

$\mathbb{G} \quad p = |\mathbb{G}| \quad g$

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

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

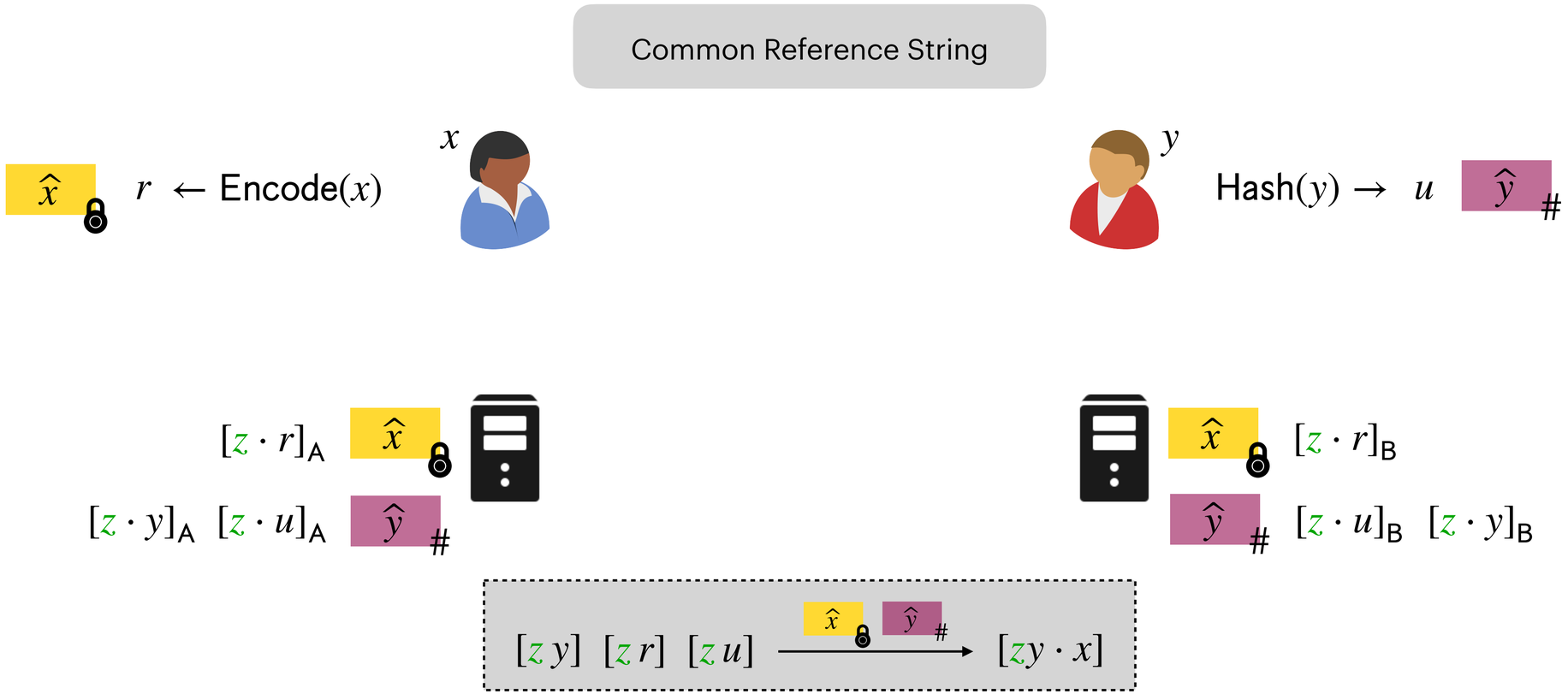
$[z]_A \quad [z \cdot \mathsf{sk}]_A \quad \mathsf{ct}_x$

$\mathsf{ct}_x \quad [z \cdot \mathsf{sk}]_B \quad [z]_B$

$[z] \quad [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Switch from $[z\,\mathsf{sk}]$ to $[z\,x]$

A Simplification of Delegatable NIM



A Simplification of Delegatable NIM

Simplification: Hash random values or public constants

Common Reference String

$\hat{x}$ $r \leftarrow \text{Encode}(x)$ x

y $\text{Hash}(y) \rightarrow u$ $\hat{y}$ #

$[z \cdot r]_A$ $\hat{x}$

$[z \cdot y]_A$ $[z \cdot u]_A$ $\hat{y}$ #

$\hat{x}$ $[z \cdot r]_B$

$\hat{y}$ # $[z \cdot u]_B$ $[z \cdot y]_B$

$[z\,y]$ $[z\,r]$ $[z\,u]$ $\xrightarrow{\hat{x}\ \ \hat{y}\ \#}$ $[zy \cdot x]$

A Simplification of Delegatable NIM

Simplification: Hash random values or public constants

Common Reference String

$\hat{x}$ $r \leftarrow \text{Encode}(x)$ x

y $\text{Hash}(y) \rightarrow$ ~~u~~ $\hat{y}$ #

$[z \cdot r]_A$ $\hat{x}$

$[z \cdot y]_A$ $[z \cdot u]_A$ $\hat{y}$ #

$\hat{x}$ $[z \cdot r]_B$

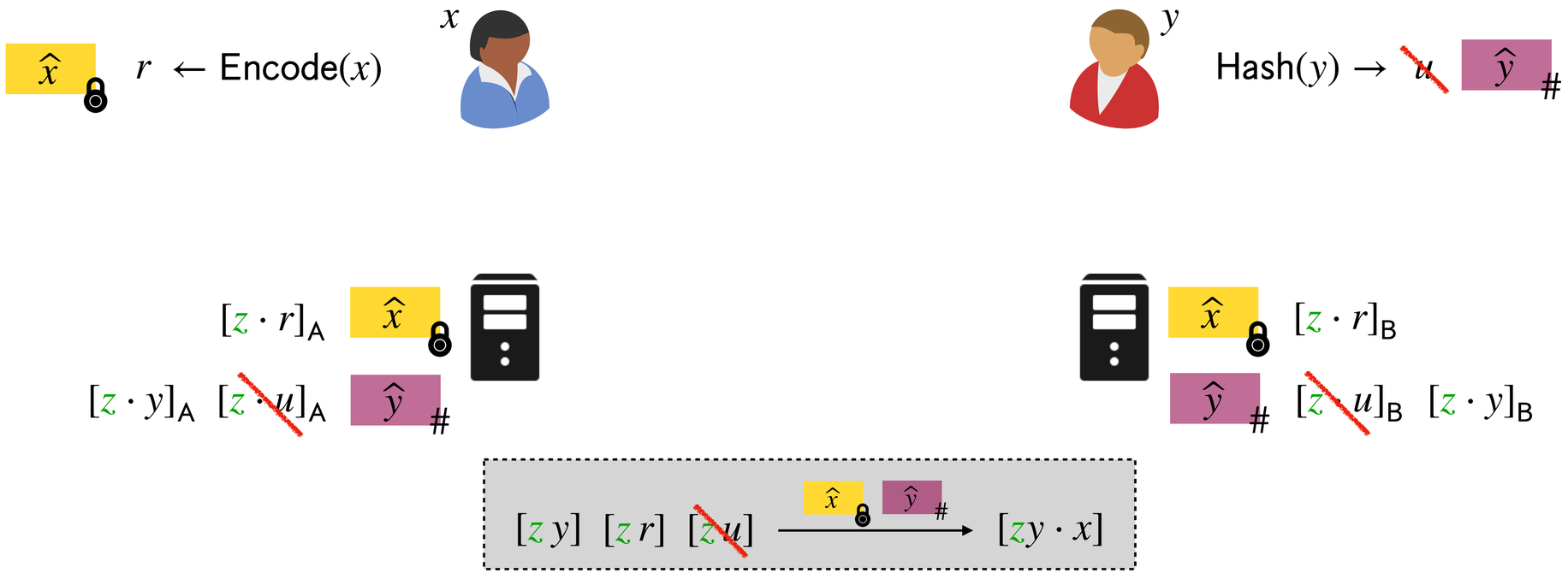
$\hat{y}$ # $[z \cdot u]_B$ $[z \cdot y]_B$

$[z\,y]$ $[z\,r]$ $[z\,u]$ $\xrightarrow{\hat{x}\ \hat{y}\ \#}$ $[zy \cdot x]$

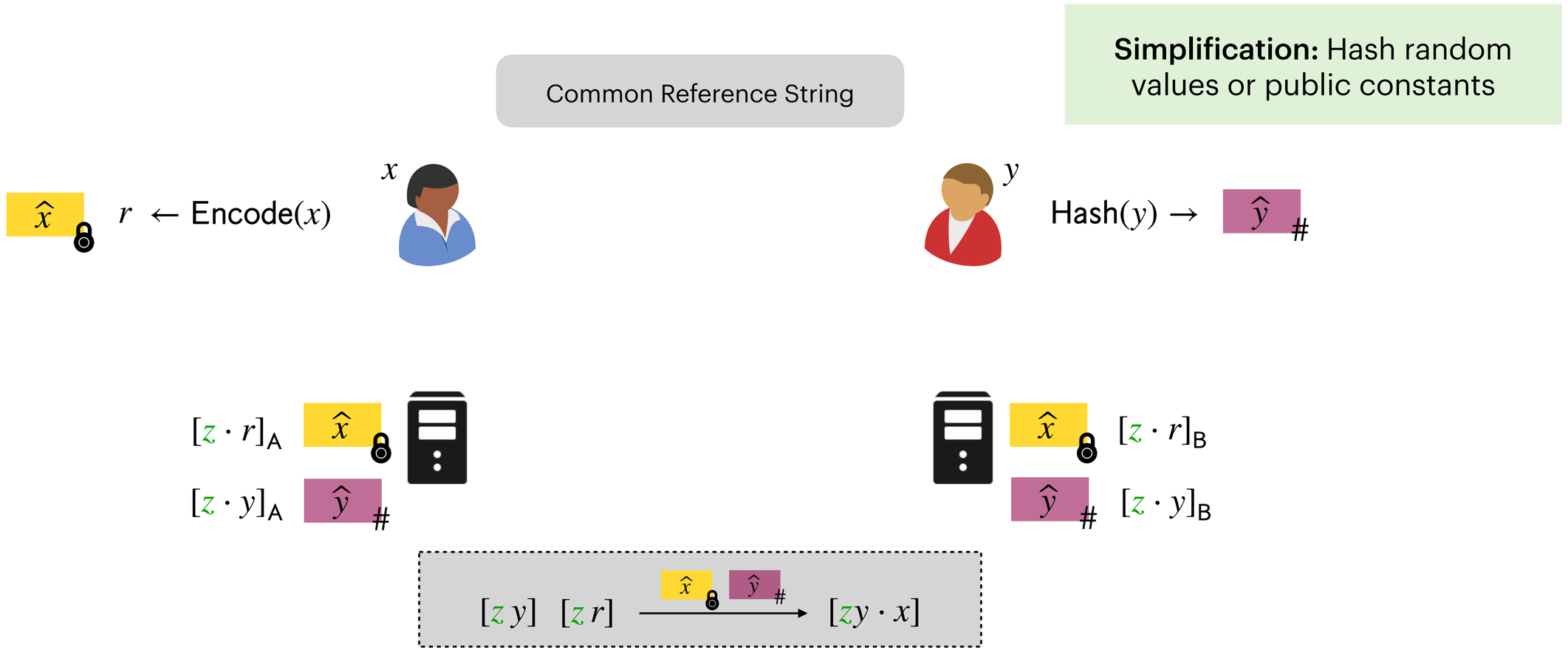
A Simplification of Delegatable NIM

Simplification: Hash random values or public constants

Common Reference String



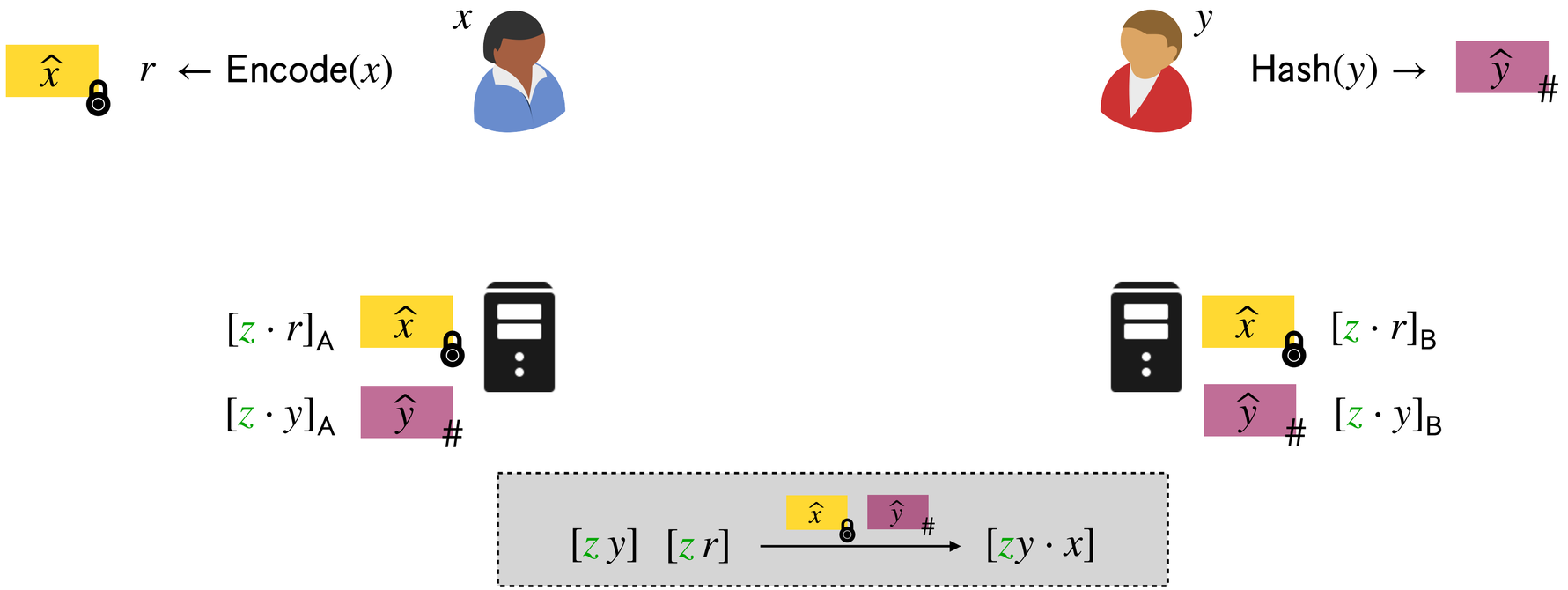
A Simplification of Delegatable NIM



A Simplification of Delegatable NIM

Simplification: Hash random values or public constants

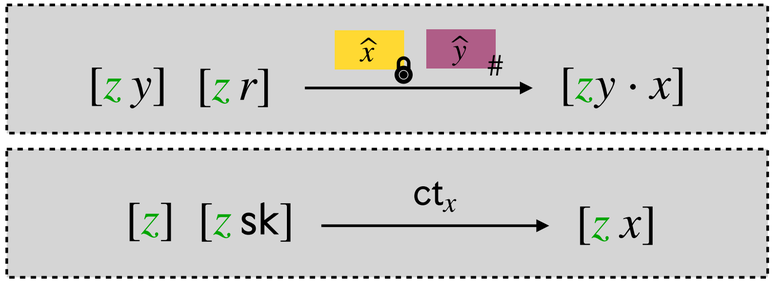
Common Reference String



Multiply $[z\,y]$ and x

(using $[z\,r]$)

Evaluating RMS Programs



Common Reference String

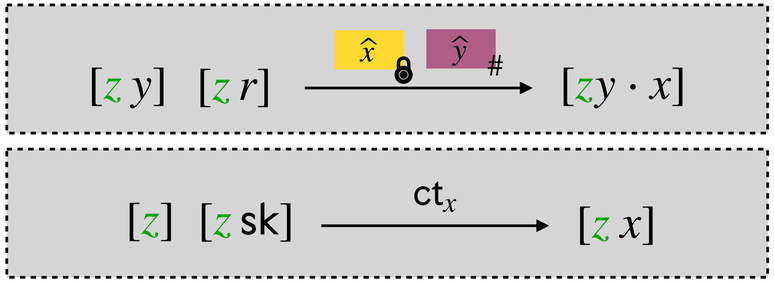
Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\hat{x}\ \ \hat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

Evaluating RMS Programs



Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

Evaluating RMS Programs

$[z\ y]\ \ [z\ r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\ \mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\ x]$

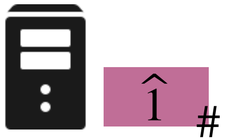
Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$



Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

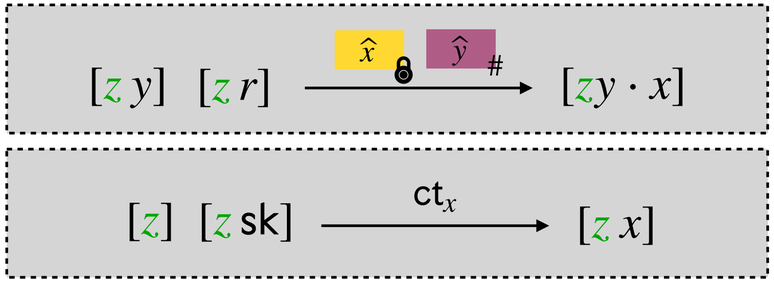
$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Evaluating RMS Programs



Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}$ # $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

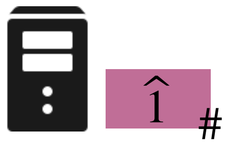
$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

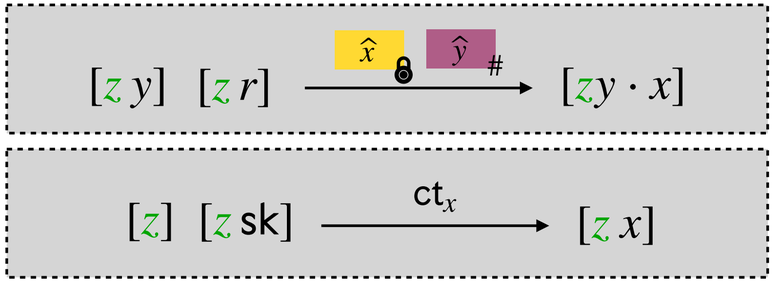
$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow$ $\widehat{\mathsf{sk}_2}$ #

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$



Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Evaluating RMS Programs



Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}_{\#} \leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}_{\#}$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$:

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

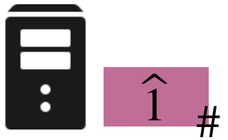
$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$



Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$:

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

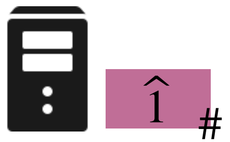
$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$



Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

2) Multiply $[z]$ with x_1: $[z]\ \ [z\,r_1] \xrightarrow{\widehat{x_1}\ \ \widehat{1}\ \#} [z \cdot x_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

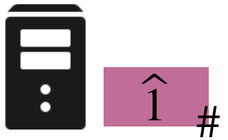
$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$



Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$ $[zx_1\,\mathsf{sk}_1]$

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

3) Multiply $[z\,\mathsf{sk}_1]$ with x_1: $[z\,\mathsf{sk}_1]\ \ [z\,r_1] \xrightarrow{\widehat{x_1}\ \ \widehat{\mathsf{sk}_1}\ \#} [z\,\mathsf{sk}_1 \cdot x_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

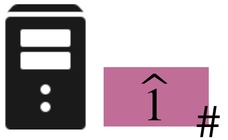
$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$



Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$ $[zx_1\,\mathsf{sk}_1]$

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$ $[zx_1\,\mathsf{sk}_1]$ $[zx_1\,\mathsf{sk}_2]$

1) Switch to $[z\,r_1]$: $[z]\ \ [z\,\mathsf{sk}_1] \xrightarrow{\mathsf{ct}_{r_1}} [z\,r_1]$

4) Multiply $[z\,\mathsf{sk}_2]$ with x_1: $[z\,\mathsf{sk}_2]\ \ [z\,r_1] \xrightarrow{\widehat{x_1}\ \ \widehat{\mathsf{sk}_2}\ \#} [z\,\mathsf{sk}_2 \cdot x_1]$

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \# \leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$ $[zx_1\,\mathsf{sk}_1]$ $[zx_1\,\mathsf{sk}_2]$

Invariant preserved!

Evaluating RMS Programs

$[z\,y]\ \ [z\,r] \xrightarrow{\widehat{x}\ \ \widehat{y}\ \#} [zy \cdot x]$

$[z]\ \ [z\,\mathsf{sk}] \xrightarrow{\mathsf{ct}_x} [z\,x]$

Common Reference String

$\widehat{x_1}$ $r_1 \leftarrow \mathsf{Encode}(x_1)$

$(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widehat{\mathsf{sk}_1}\ \#$ $\leftarrow \mathsf{Hash}(\mathsf{sk}_1)$

$\mathsf{ct}_{r_1} \leftarrow \mathsf{Encrypt}(\mathsf{pk}_1, r_1)$

$\mathsf{Encode}(x_2) \rightarrow r_2$ $\widehat{x_2}$

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}_2, \mathsf{sk}_2)$

$\mathsf{Hash}(\mathsf{sk}_2) \rightarrow \widehat{\mathsf{sk}_2}\ \#$

$\mathsf{Encrypt}(\mathsf{pk}_2, r_2) \rightarrow \mathsf{ct}_{r_2}$

Memory share of z: $[z]$ $[z\,\mathsf{sk}_1]$ $[z\,\mathsf{sk}_2]$

Memory share of $z\,x_1$: $[zx_1]$ $[zx_1\,\mathsf{sk}_1]$ $[zx_1\,\mathsf{sk}_2]$

Invariant preserved!

- Similar approach to multiply with x_2
- Extends naturally to arbitrary number of clients

HSS for **Multiplication** is All You Need

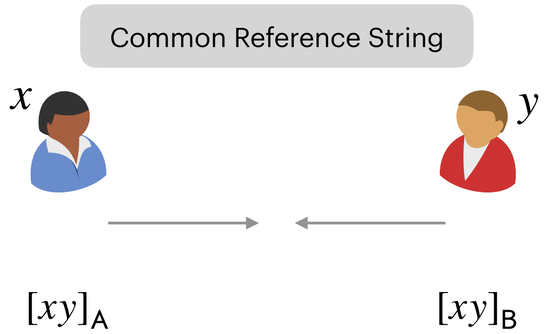
Linear Decoding

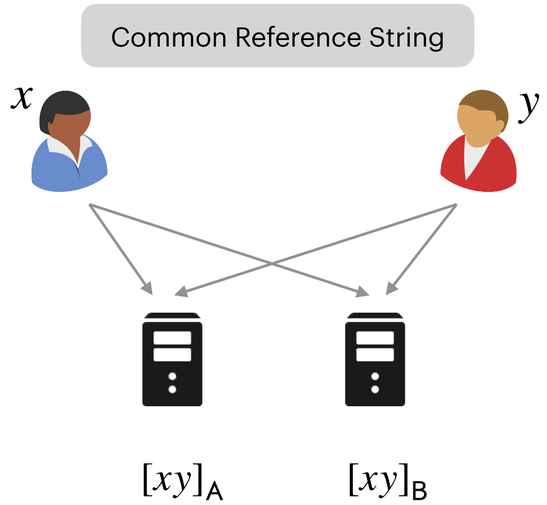
Encryption with
Linear Decryption

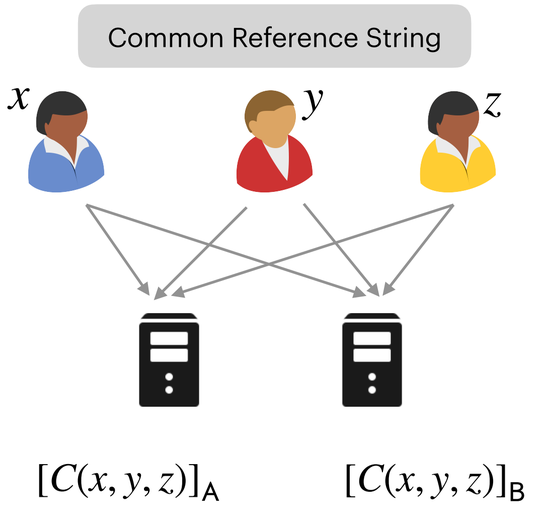
Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs







HSS for **Multiplication** is All You Need

Linear Decoding

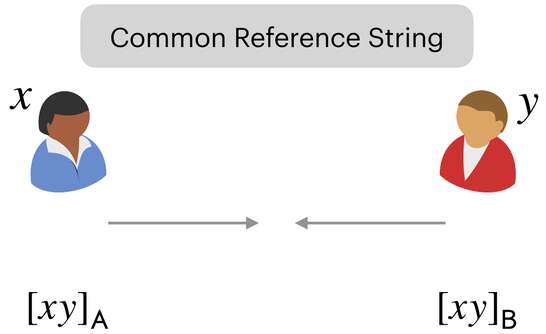
Encryption with Linear Decryption

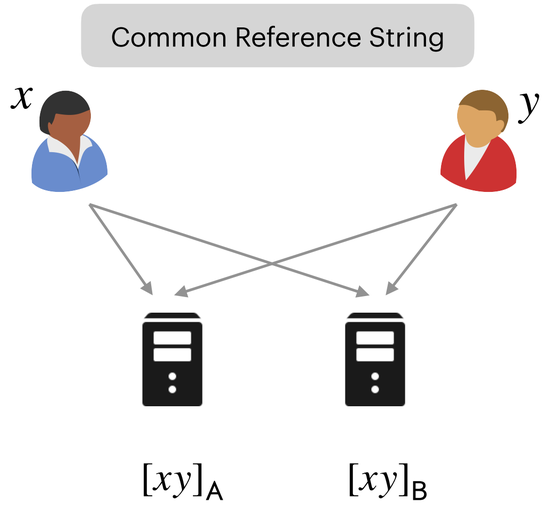
Implied by NIM with linear decoding

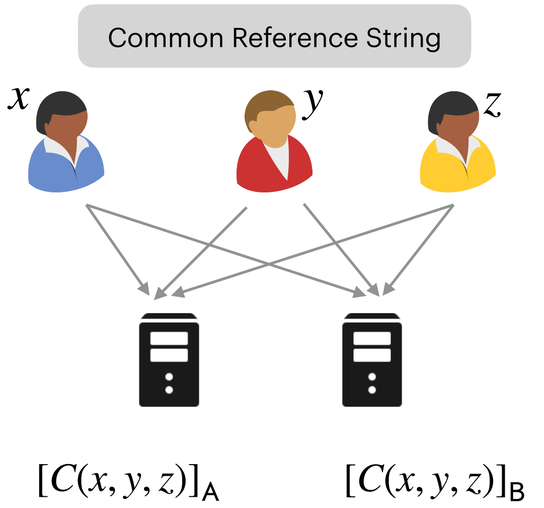
Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs







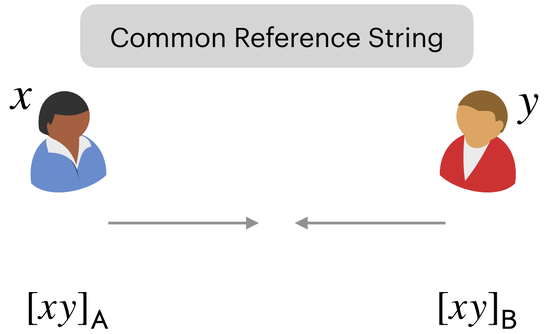
NIM with Linear Decoding is All You Need

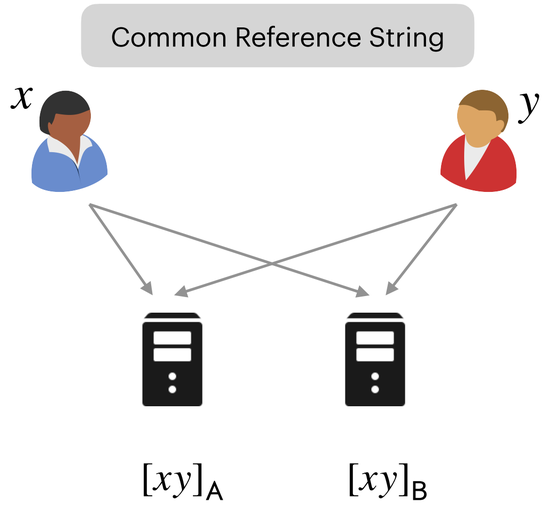
Linear Decoding

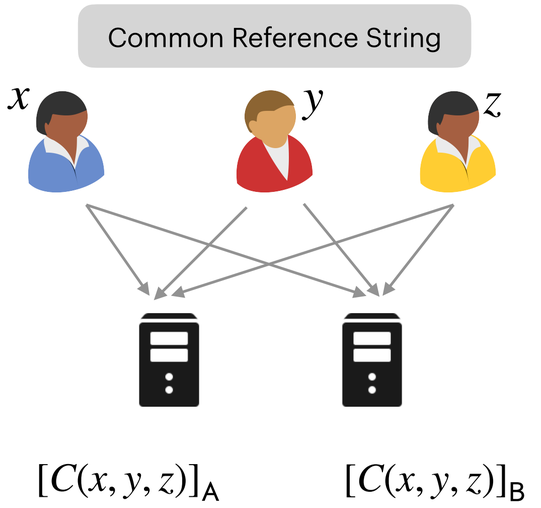
Non-interactive multiplication

Delegatable non-interactive multiplication

Client-server HSS for RMS programs







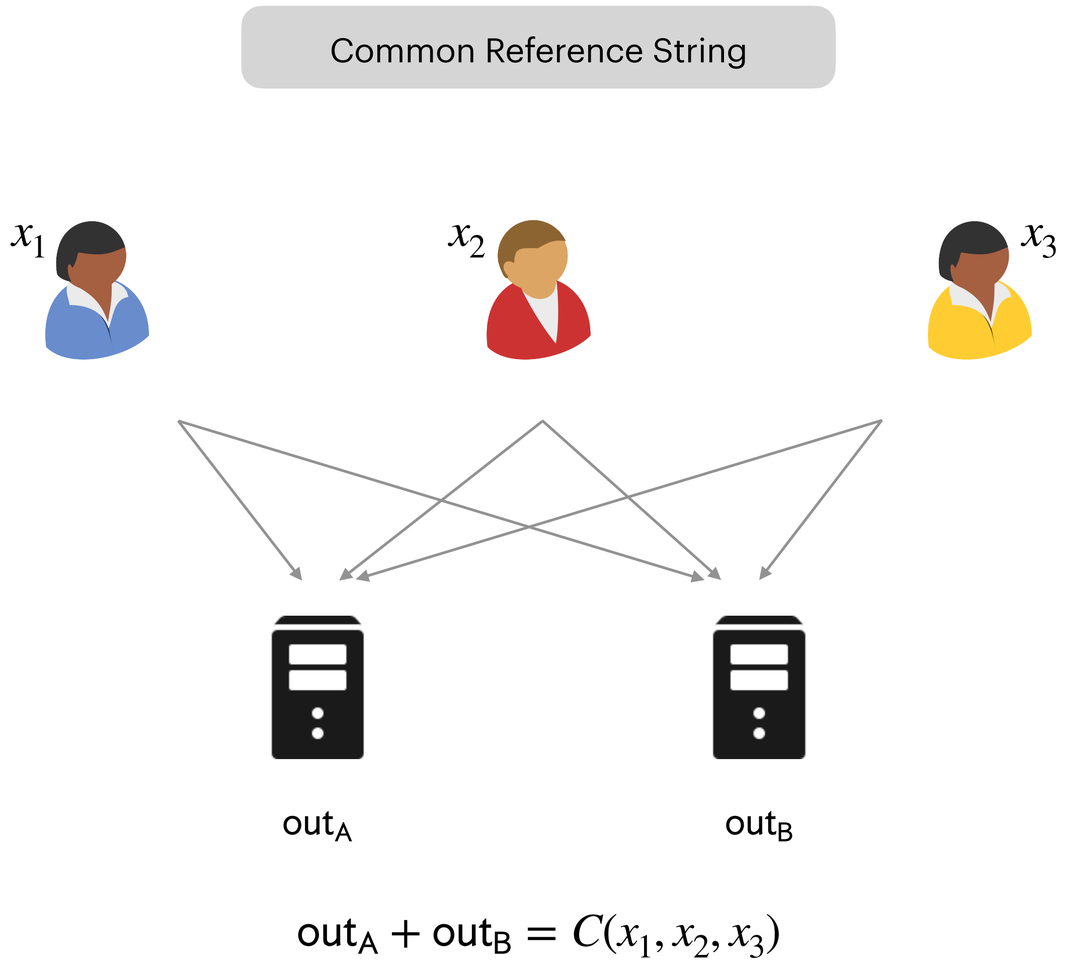
Outline

Barriers to Removing Correlated Setup

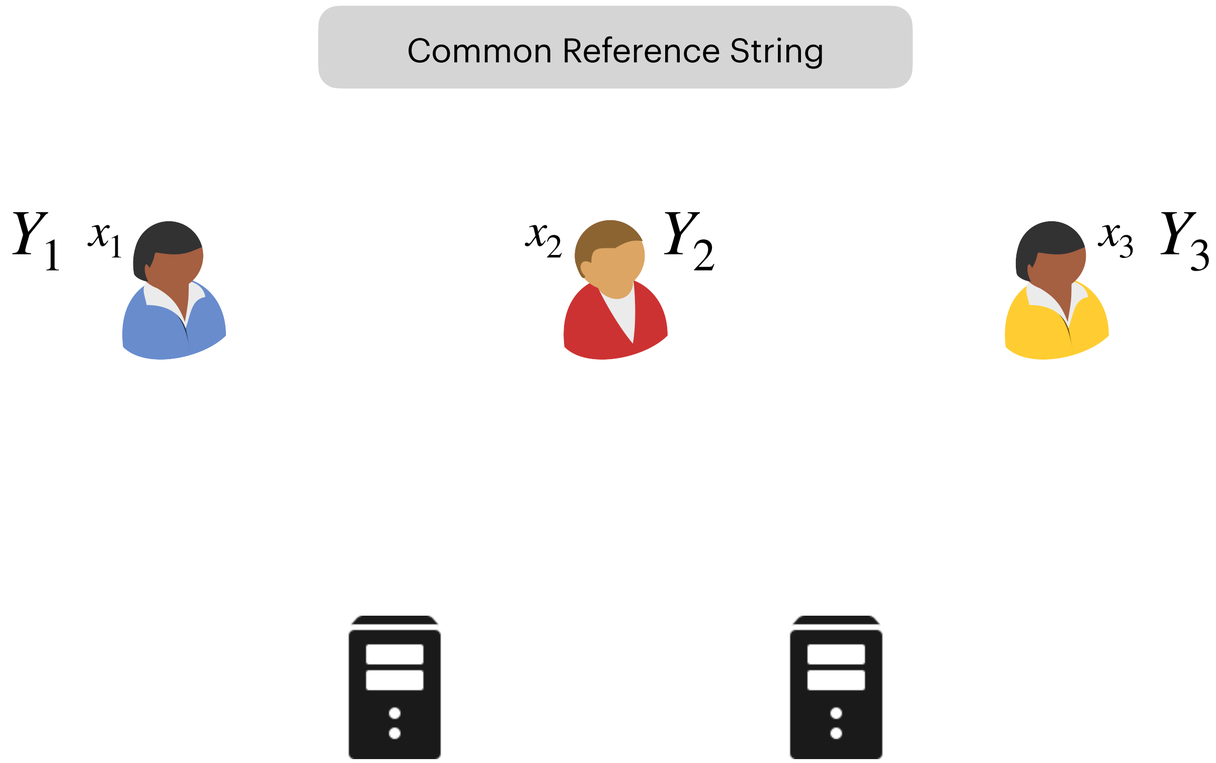
Our Approach

Extensions

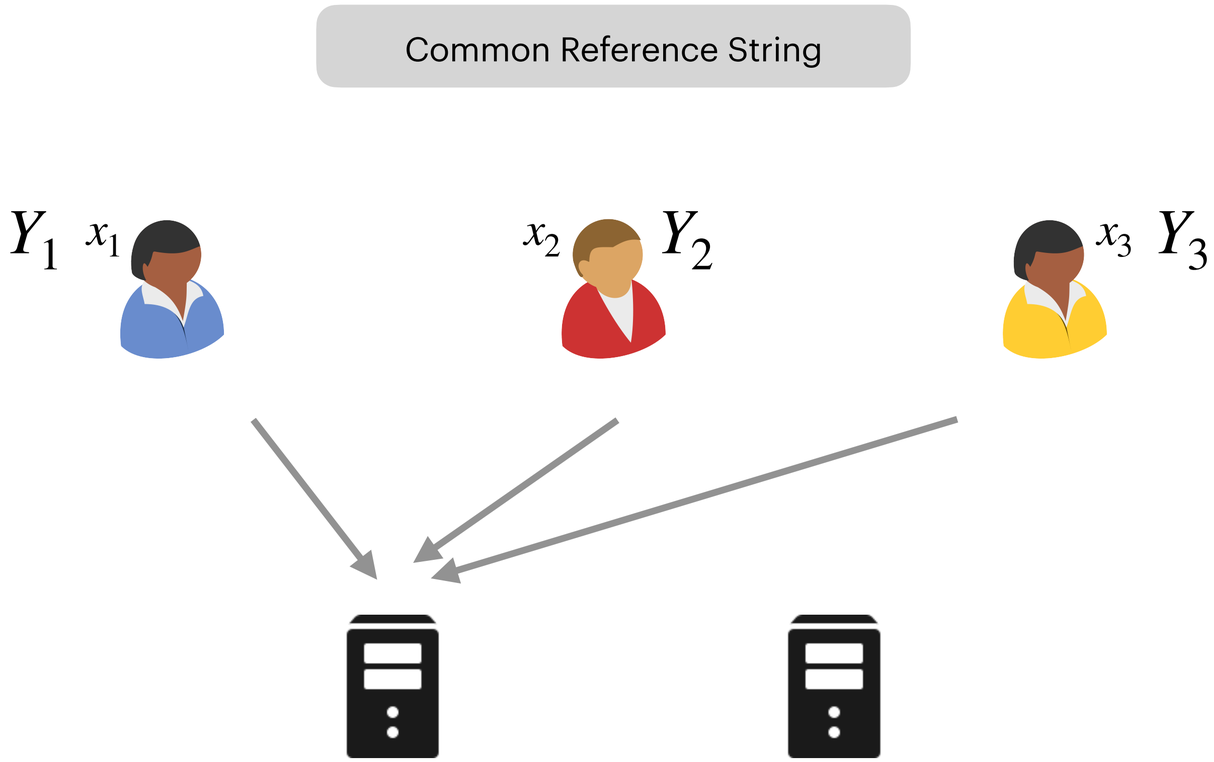
Succinct Client-Server HSS



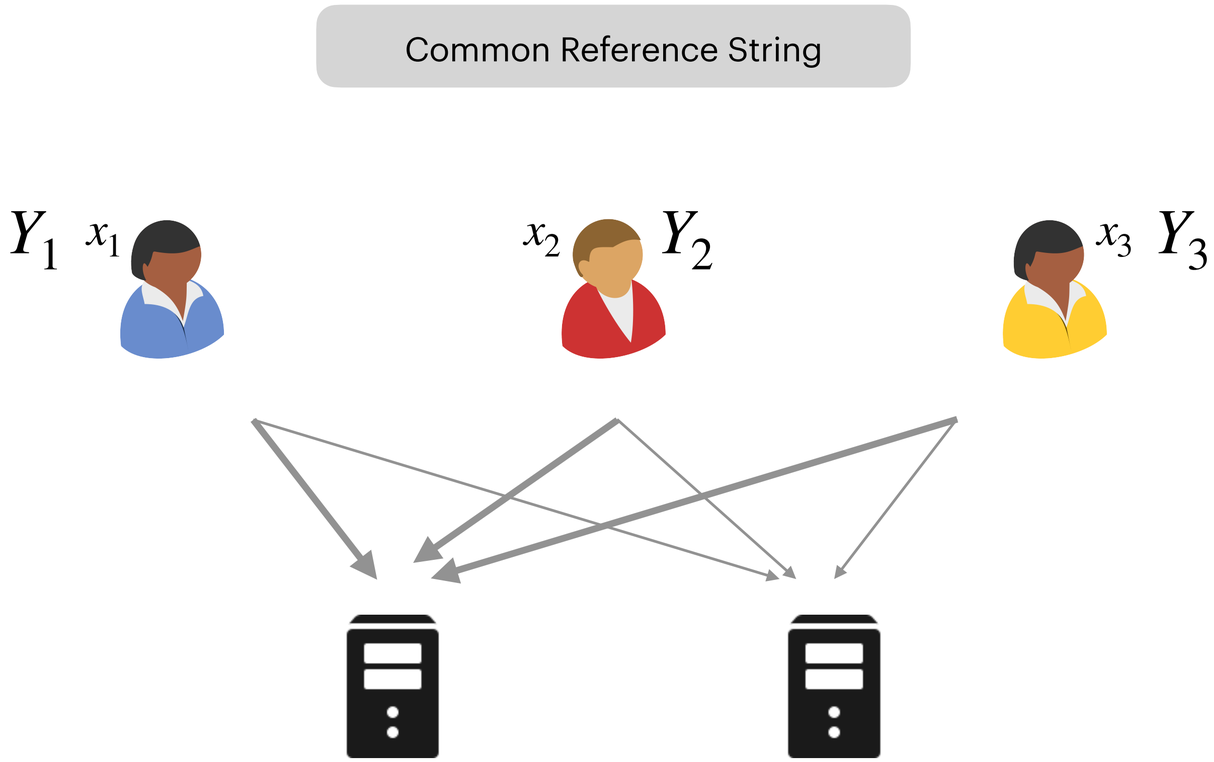
Succinct Client-Server HSS



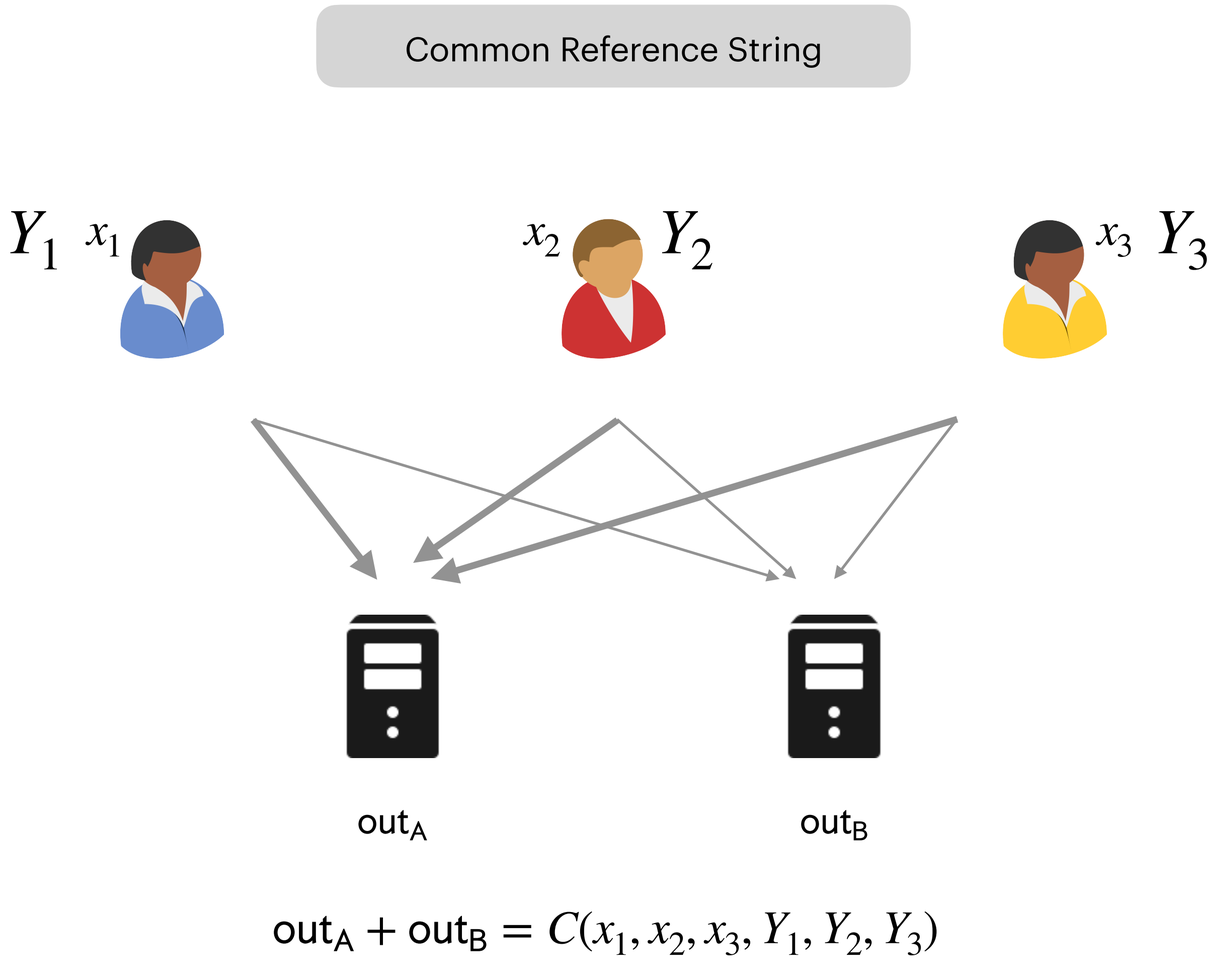
Succinct Client-Server HSS



Succinct Client-Server HSS

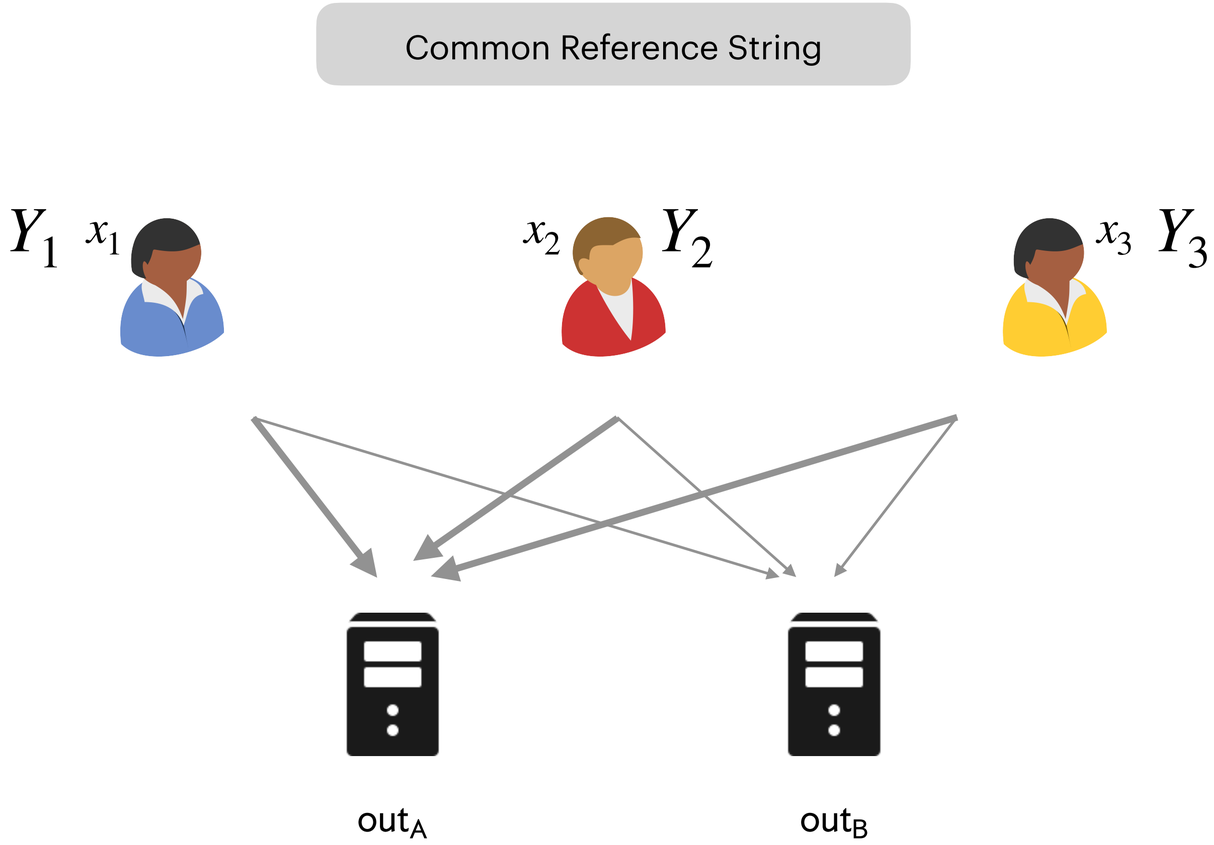


Succinct Client-Server HSS



Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs



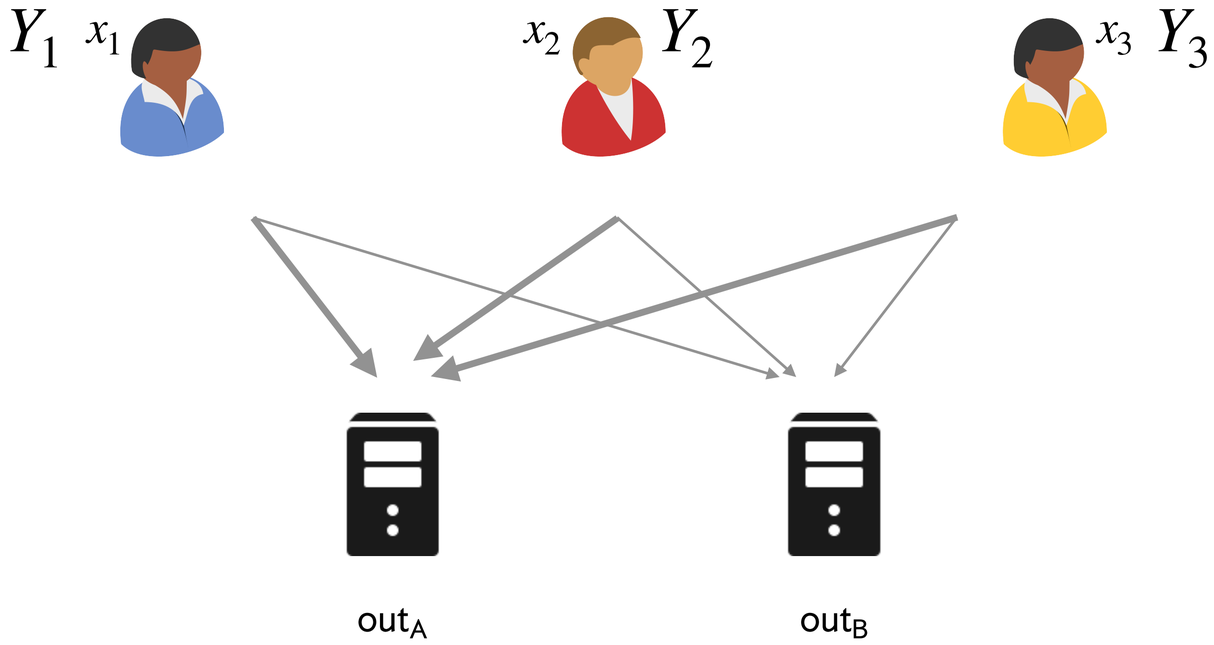
$$\text{out}_A + \text{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

Common Reference String



$$\text{out}_A + \text{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

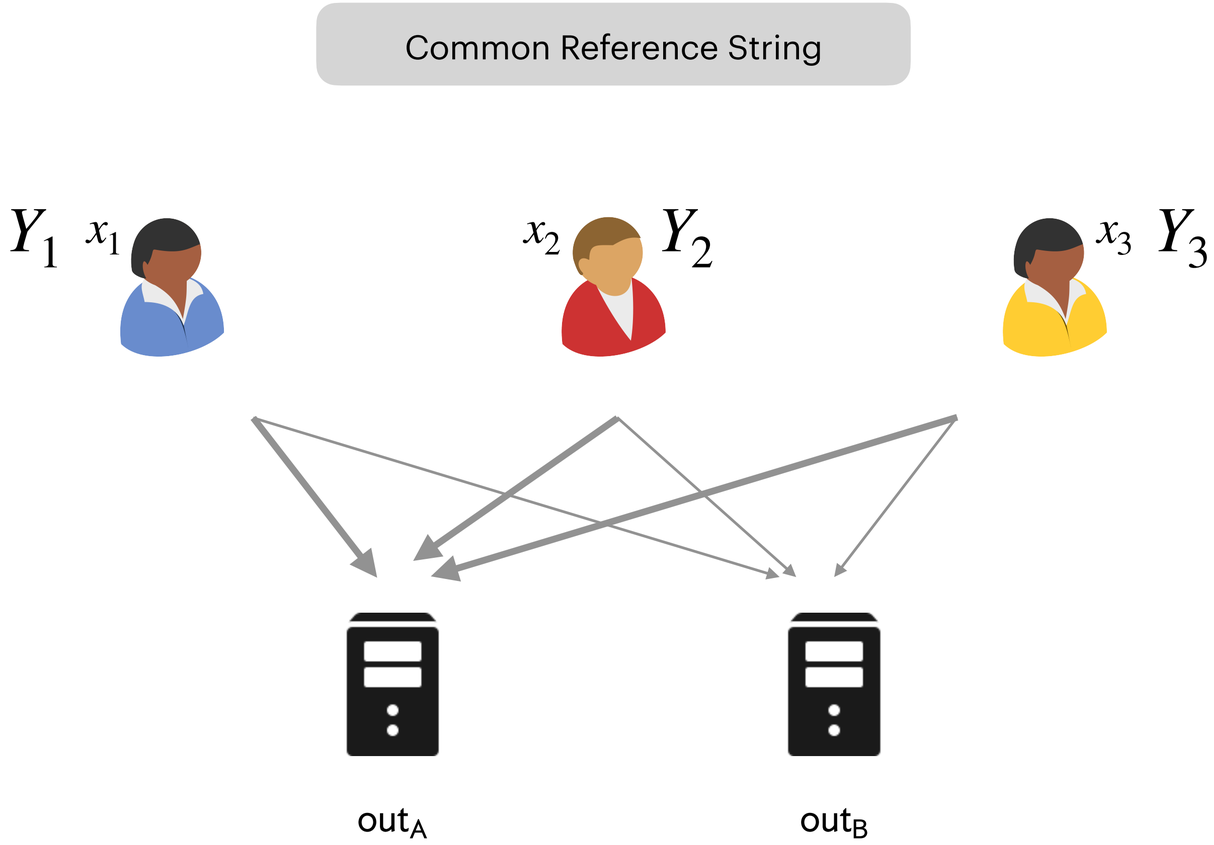
Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

Previously required correlated setup or supported only two parties

[Abram-Roy-Scholl'24]
[Couteau-H-Pu'24]



$$\text{out}_A + \text{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

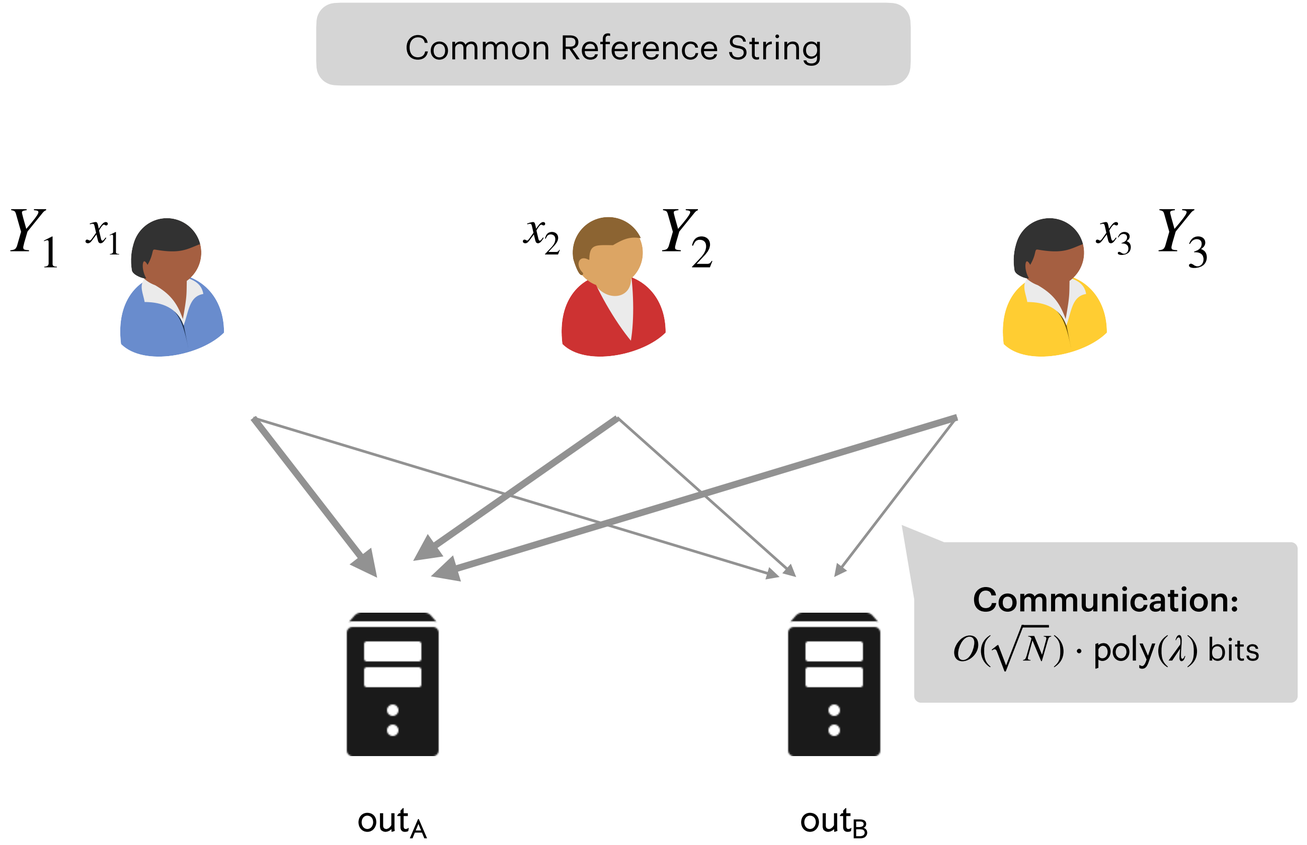
Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

Previously required correlated setup or supported only two parties

[Abram-Roy-Scholl'24]
[Couteau-H-Pu'24]



$$\text{out}_A + \text{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

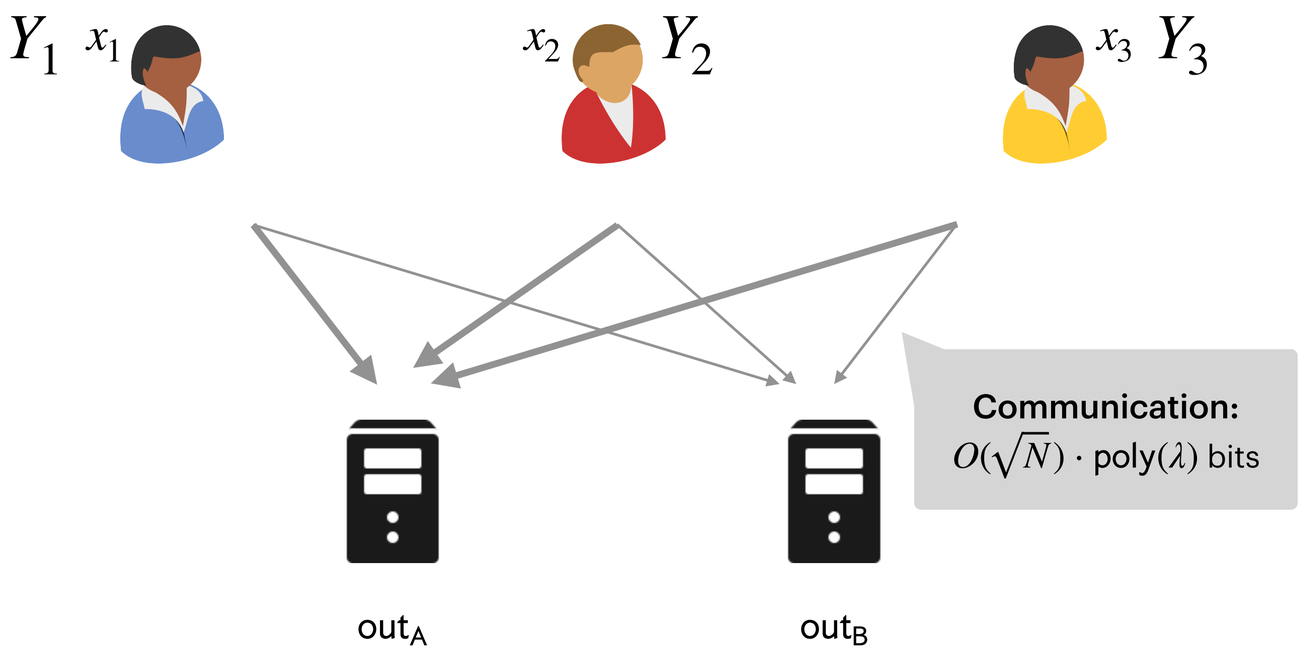
Previously required correlated setup or supported only two parties

[Abram-Roy-Scholl'24]
[Couteau-H-Pu'24]

Private long inputs

$$C \equiv \sum_{i,j} \mathsf{RMS}(x_1, \ldots, x_m) \cdot Y_i^{(j)}$$

Common Reference String



Communication:
$O(\sqrt{N}) \cdot \mathsf{poly}(\lambda)$ bits

$$\mathsf{out}_A + \mathsf{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

Previously required correlated setup or supported only two parties

[Abram-Roy-Scholl'24]
[Couteau-H-Pu'24]

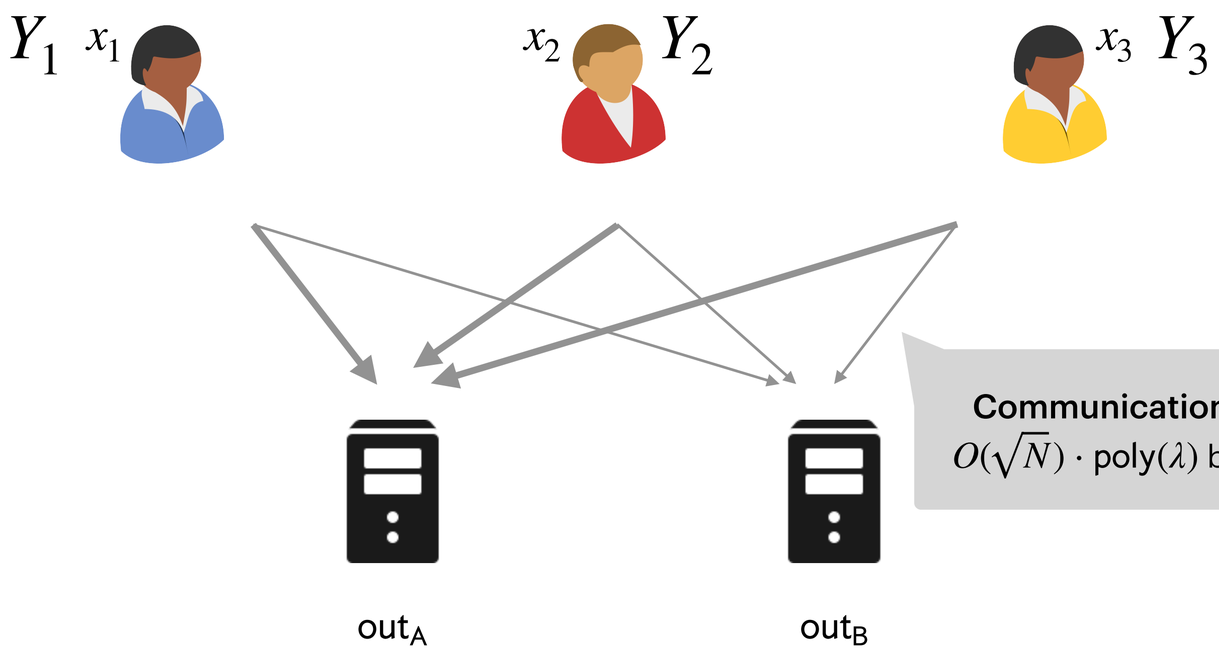
Private long inputs

$$C \equiv \sum_{i,j} \mathsf{RMS}(x_1, \ldots, x_m) \cdot Y_i^{(j)}$$

Public long inputs

$$C \equiv \mathsf{RMS}(x_1, \ldots, x_m) \cdot \mathsf{P/poly}(Y_1, \ldots, Y_m)$$

Common Reference String



Communication:
$O(\sqrt{N}) \cdot \mathsf{poly}(\lambda)$ bits

$$\mathsf{out}_A + \mathsf{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

Succinct Client-Server HSS

Succinct multi-client two-server HSS in the CRS model for RMS programs

DDH, DCR, and class groups

Previously required correlated setup or supported only two parties

[Abram-Roy-Scholl'24]
[Couteau-H-Pu'24]

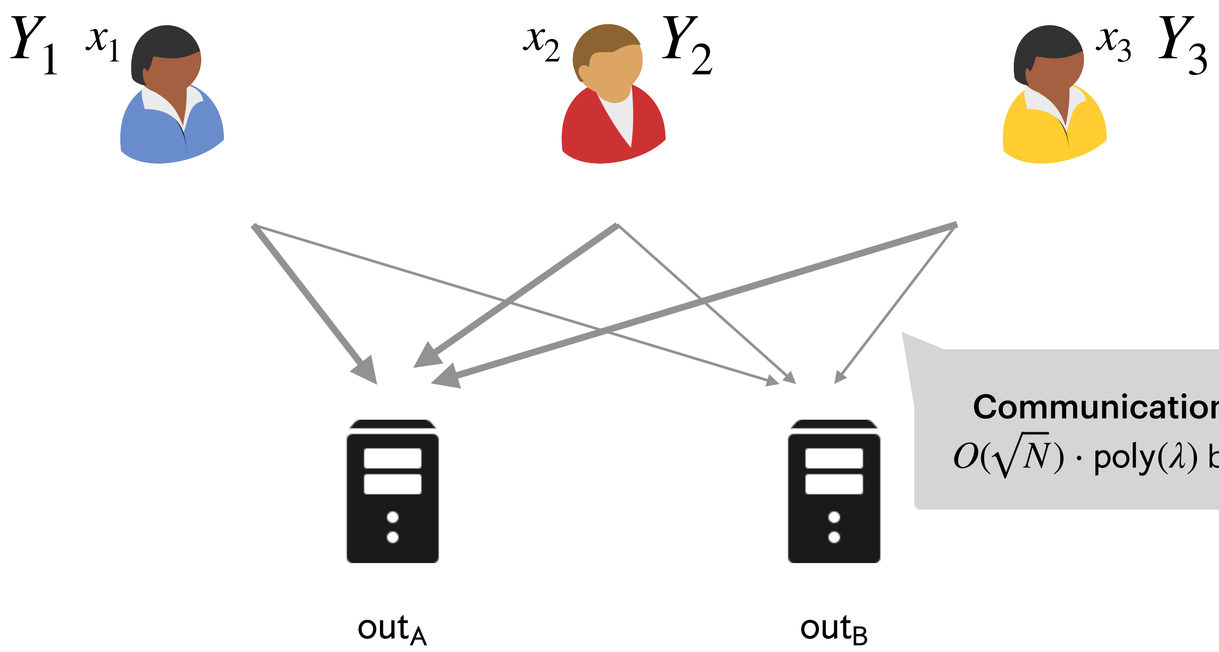
Private long inputs

$$C \equiv \sum_{i,j} \mathsf{RMS}(x_1, \ldots, x_m) \cdot Y_i^{(j)}$$

Public long inputs

$$C \equiv \mathsf{RMS}(x_1, \ldots, x_m) \cdot \mathsf{P/poly}(Y_1, \ldots, Y_m)$$

Common Reference String



Communication: $O(\sqrt{N}) \cdot \mathsf{poly}(\lambda)$ bits

$$\mathsf{out}_A + \mathsf{out}_B = C(x_1, x_2, x_3, Y_1, Y_2, Y_3)$$

Key Ingredient: Combine delegation and input-succinctness properties of NIM

Thank You