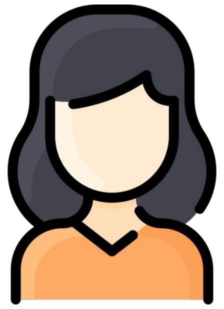
Round-Optimal and Communication-Efficient Multiparty Computation

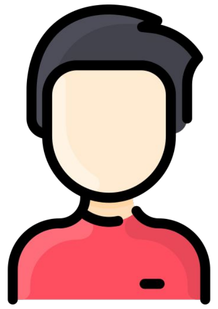
Michele Ciampi, Rafail Ostrovsky,
Hendrik Waldner, Vassilis Zikas

Multiparty Computation (MPC)

Multiparty Computation (MPC)

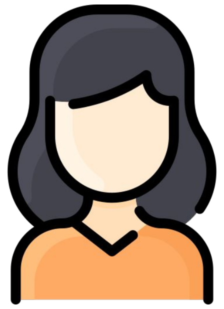


x_1

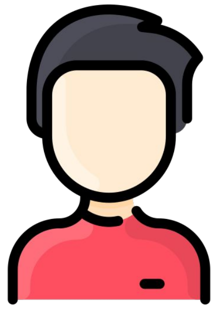


x_2

Multiparty Computation (MPC)



x_1

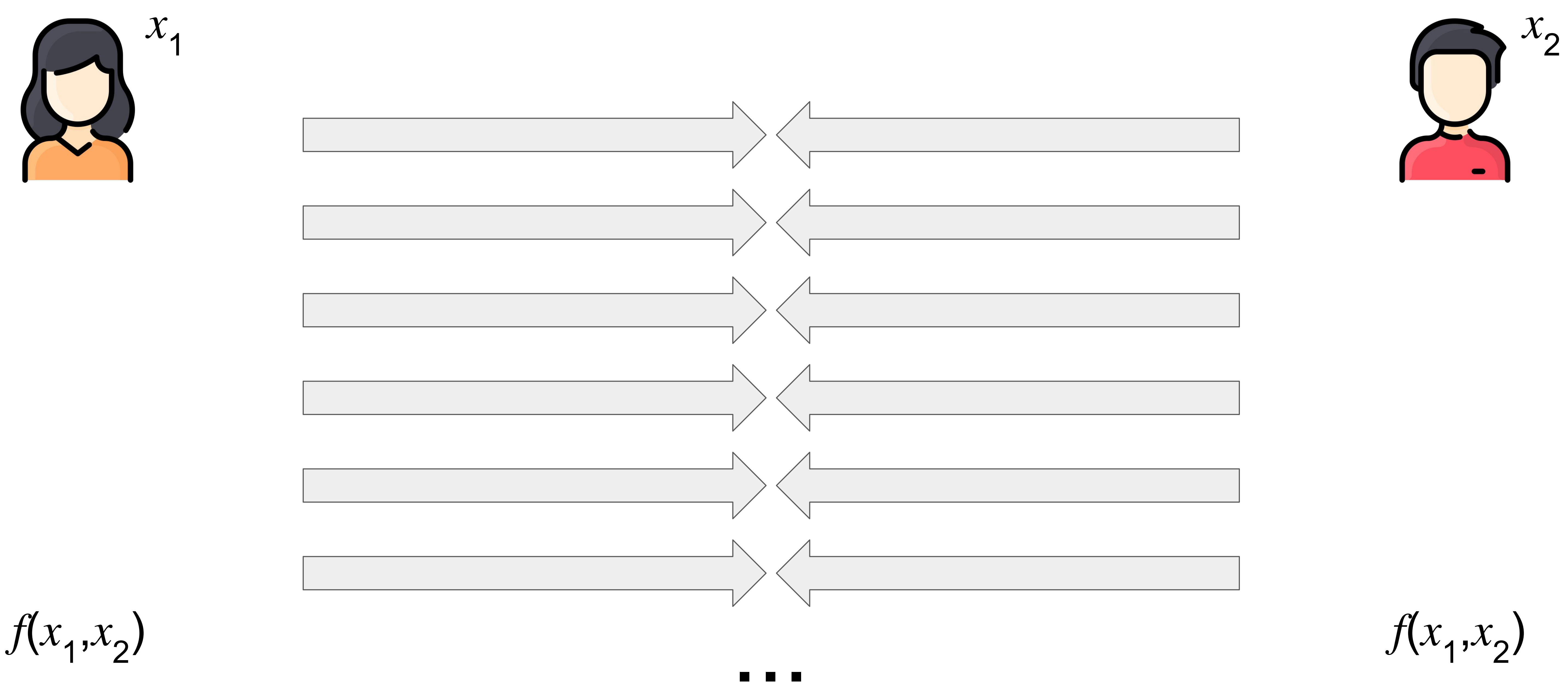


x_2

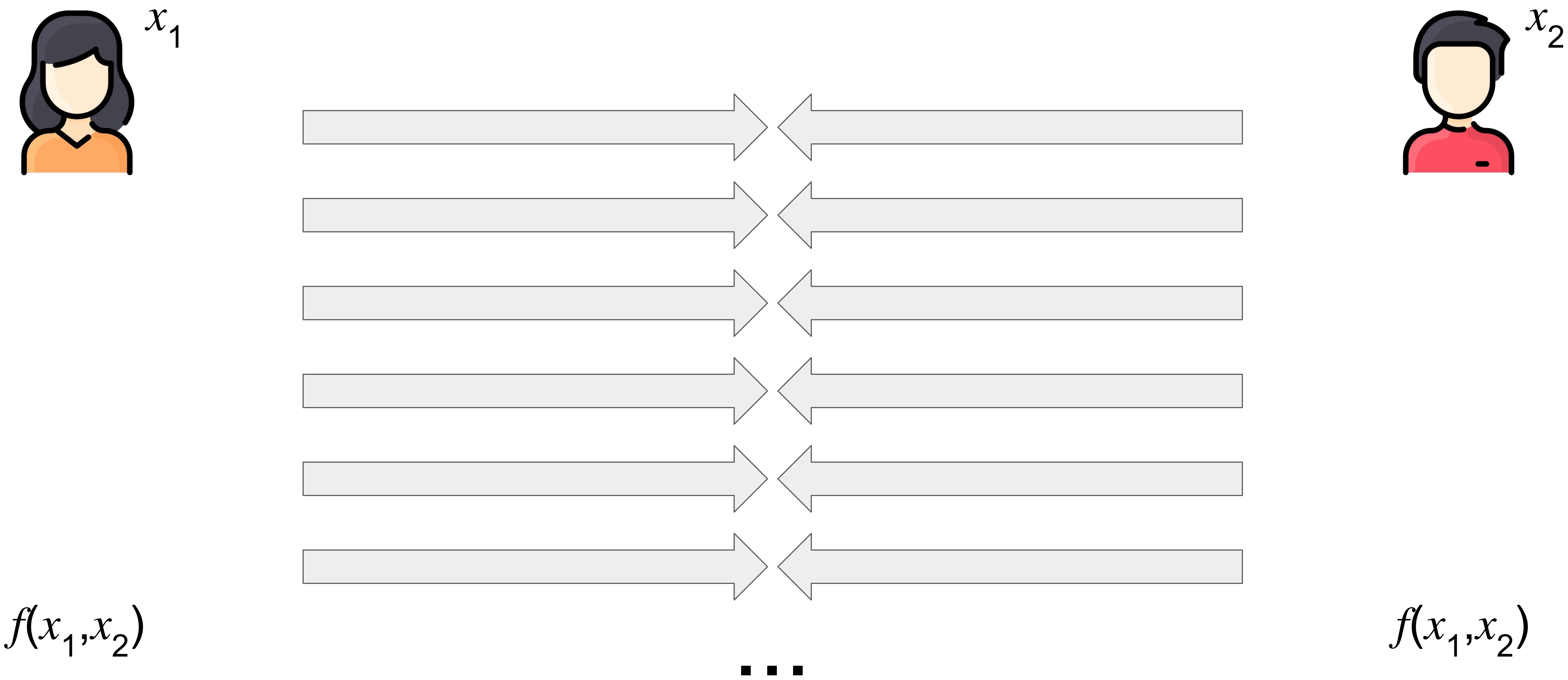
$f(x_1, x_2)$

$f(x_1, x_2)$

Multiparty Computation (MPC)

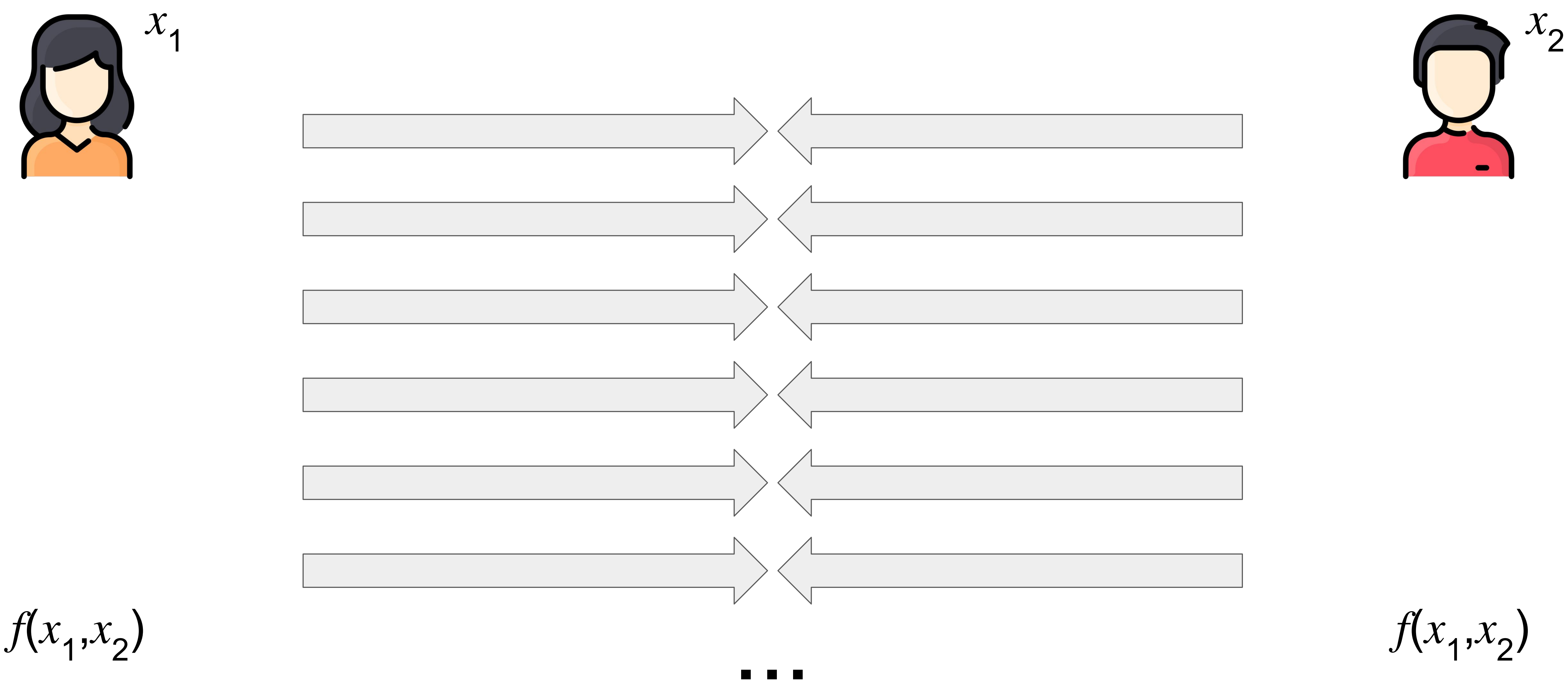


Multiparty Computation (MPC)



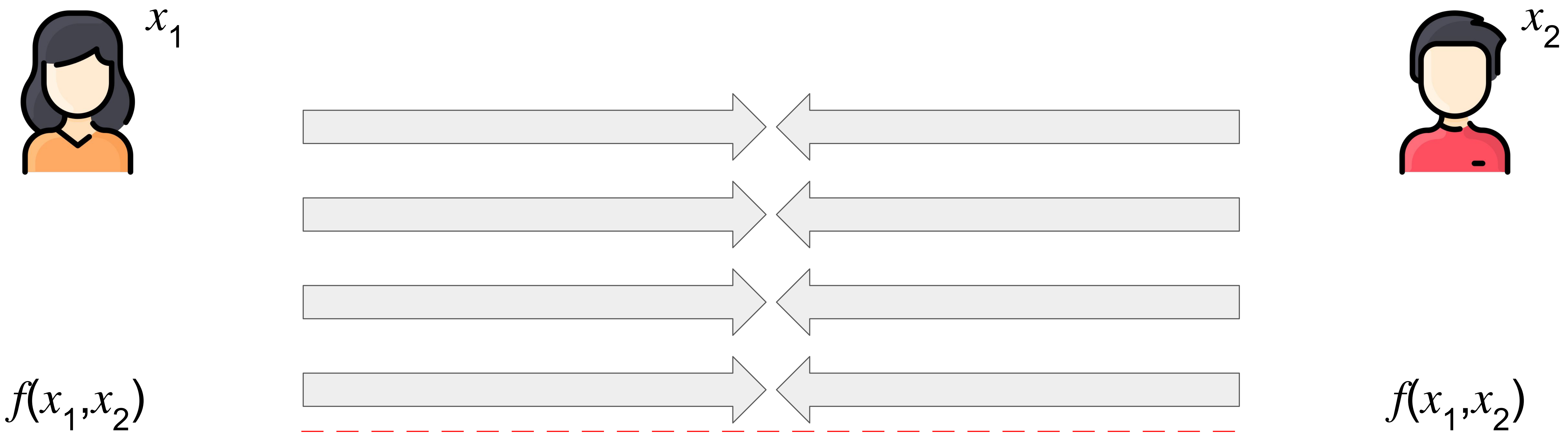
1. Number of Messages

Multiparty Computation (MPC)



1. Number of Messages
2. Size of Messages

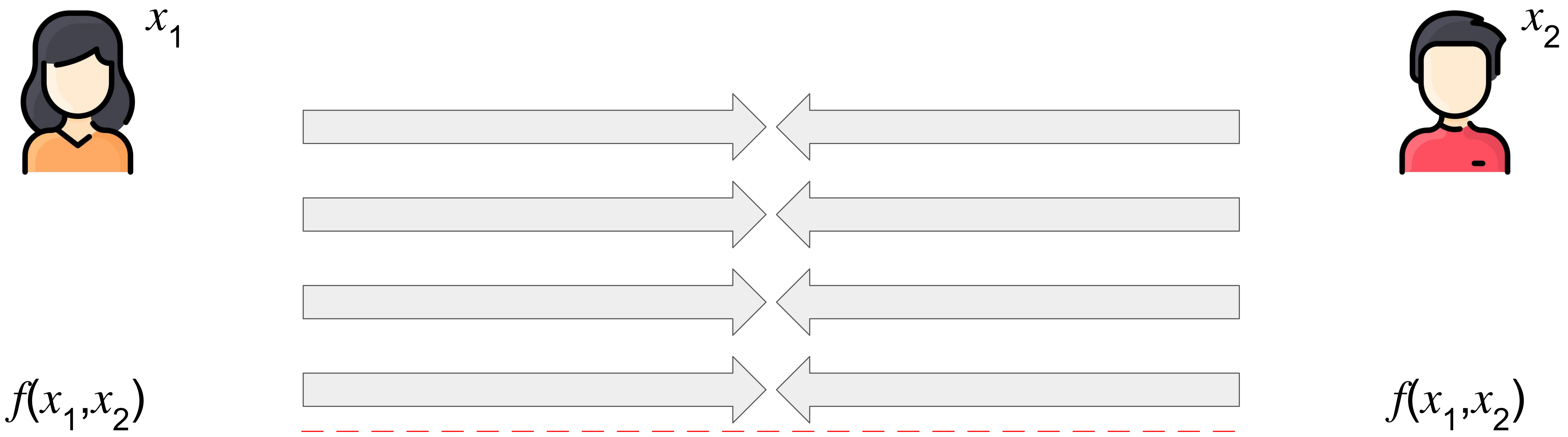
Multiparty Computation (MPC)



Four rounds are necessary [GMPP16] and sufficient [CCG+19,BGJ+18, HHPV18]

1. Number of Messages
2. Size of Messages

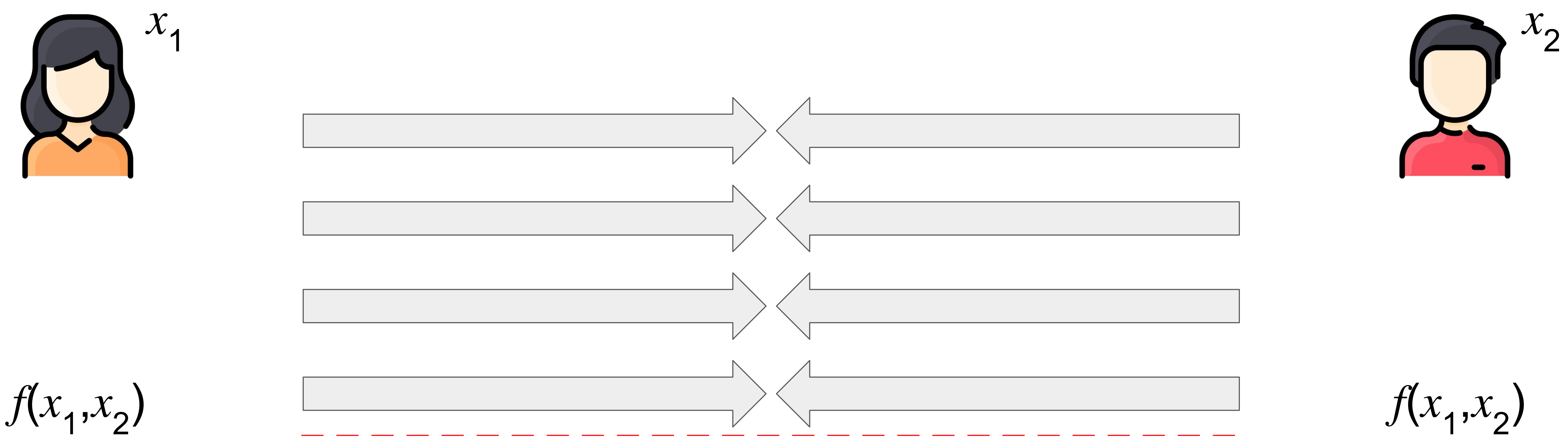
Multiparty Computation (MPC)



Four rounds are necessary [GMPP16] and sufficient [CCG+19,BGJ+18, HHPV18]

1. Number of Messages ✓
2. Size of Messages

Multiparty Computation (MPC)



Four rounds are necessary [GMPP16] and sufficient [CCG+19,BGJ+18, HHPV18]

1. Number of Messages ✓
2. Size of Messages ←

Prior Work

Prior Work

	Semi-Honest		
	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\lvert f \rvert$

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\lvert f \rvert$	[BGJ+18]	DDH/Q-N Res.	$\lvert f \rvert$
				[CCG+20]	OT	

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)			

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)			
	[AJJM20]	R-LWE, DSPR & OT	L_{in} & L_{out}			

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)	This Work		
	[AJJM20]	R-LWE, DSPR & OT	L_{in} & L_{out}			

Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)	This Work	LWE	depth(f)
	[AJJM20]	R-LWE, DSPR & OT	L_{in} & L_{out}			

Prior Work

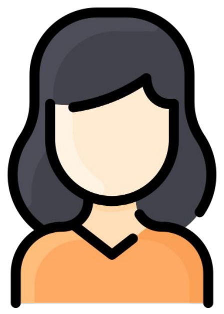
	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\lvert f \rvert$	[BGJ+18]	DDH/Q-N Res.	$\lvert f \rvert$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)	This Work	LWE	depth(f)
	[AJJM20]	R-LWE, DSPR & OT	L_{in} & L_{out}		R-LWE, DSPR & OT	L_{in} & L_{out}

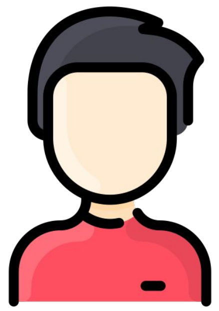
Prior Work

	Semi-Honest			Malicious		
	Work	Assumptions	Comm. Compl.	Work	Assumptions	Comm. Compl.
Round-Optimal Protocols	[BL18] [GS18]	OT	$\|f\|$	[BGJ+18]	DDH/Q-N Res.	$\|f\|$
				[CCG+20]	OT	
With Improved Comm. Compl.	[ABJ+19] [QWW18]	LWE	depth(f)	This Work	LWE	depth(f)
	[AJJM20]	R-LWE, DSPR & OT	L_{in} & L_{out}		R-LWE, DSPR & OT	L_{in} & L_{out}

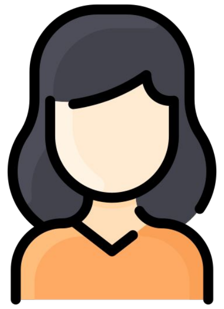
⇒ Start from the work of Ananth et al. and Functional Encryption Combiners

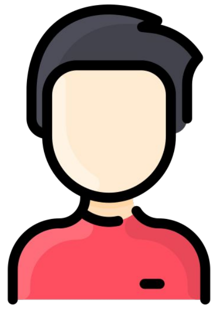
Functional Encryption [BSW11]





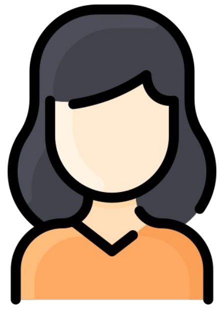
Functional Encryption [BSW11]

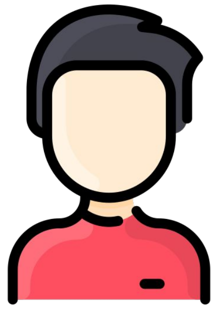




 ← Setup

Functional Encryption [BSW11]

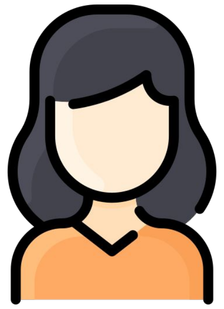


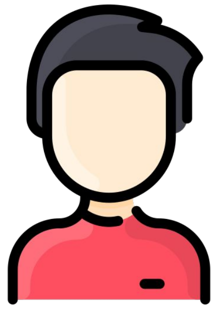


 $\leftarrow$ Setup

$_f$ $\leftarrow$ Keygen(, f)

Functional Encryption [BSW11]



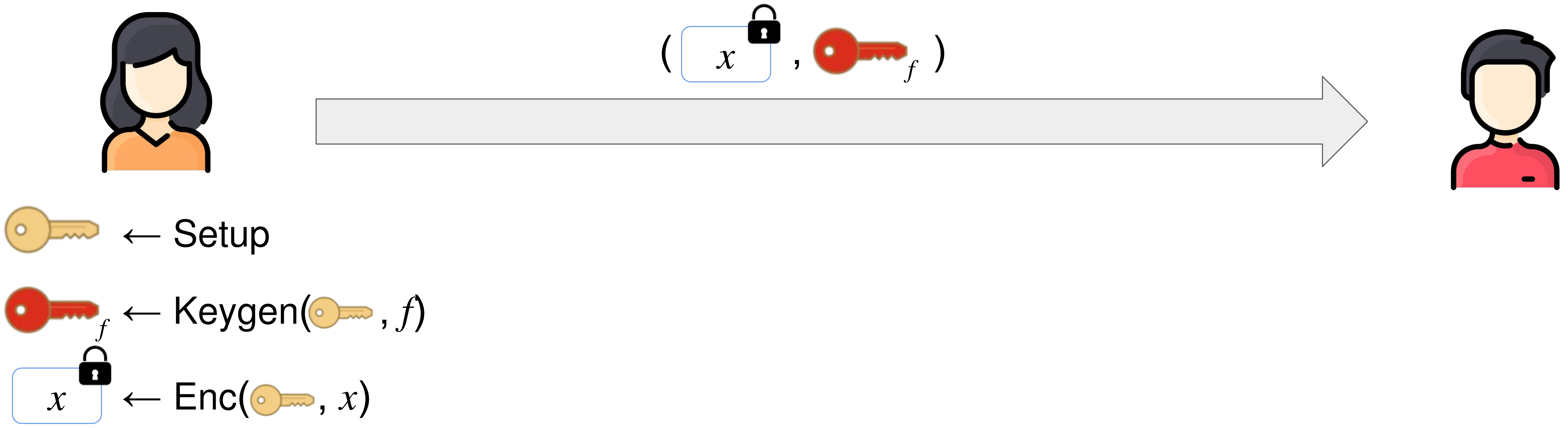


$\leftarrow$ Setup

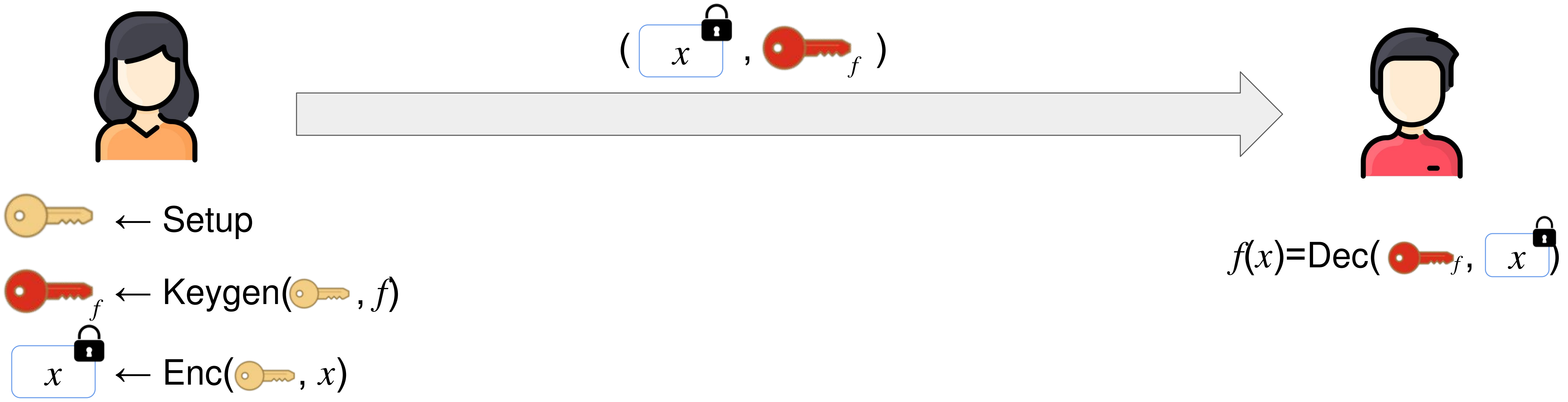
$_f \leftarrow$ Keygen(, f)

$\boxed{x} \leftarrow$ Enc(, x)

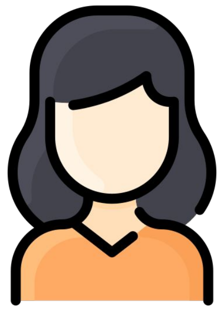
Functional Encryption [BSW11]

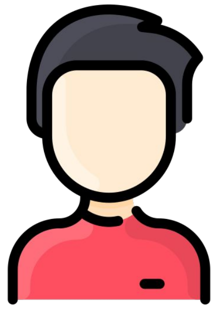


Functional Encryption [BSW11]

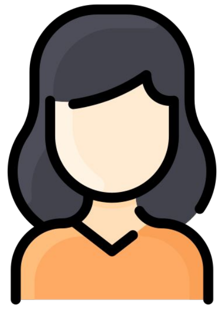


(Decomposable) Functional Encryption Combiner [ABJ+19]

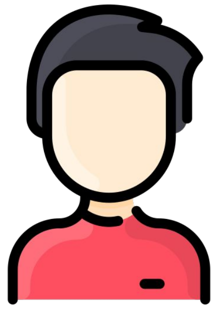




(Decomposable) Functional Encryption Combiner [ABJ+19]

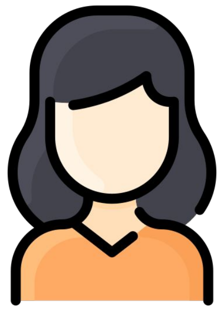


 $_1 \leftarrow$ Setup



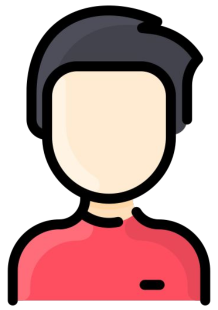
 $_2 \leftarrow$ Setup

(Decomposable) Functional Encryption Combiner [ABJ+19]



$_1 \leftarrow$ Setup

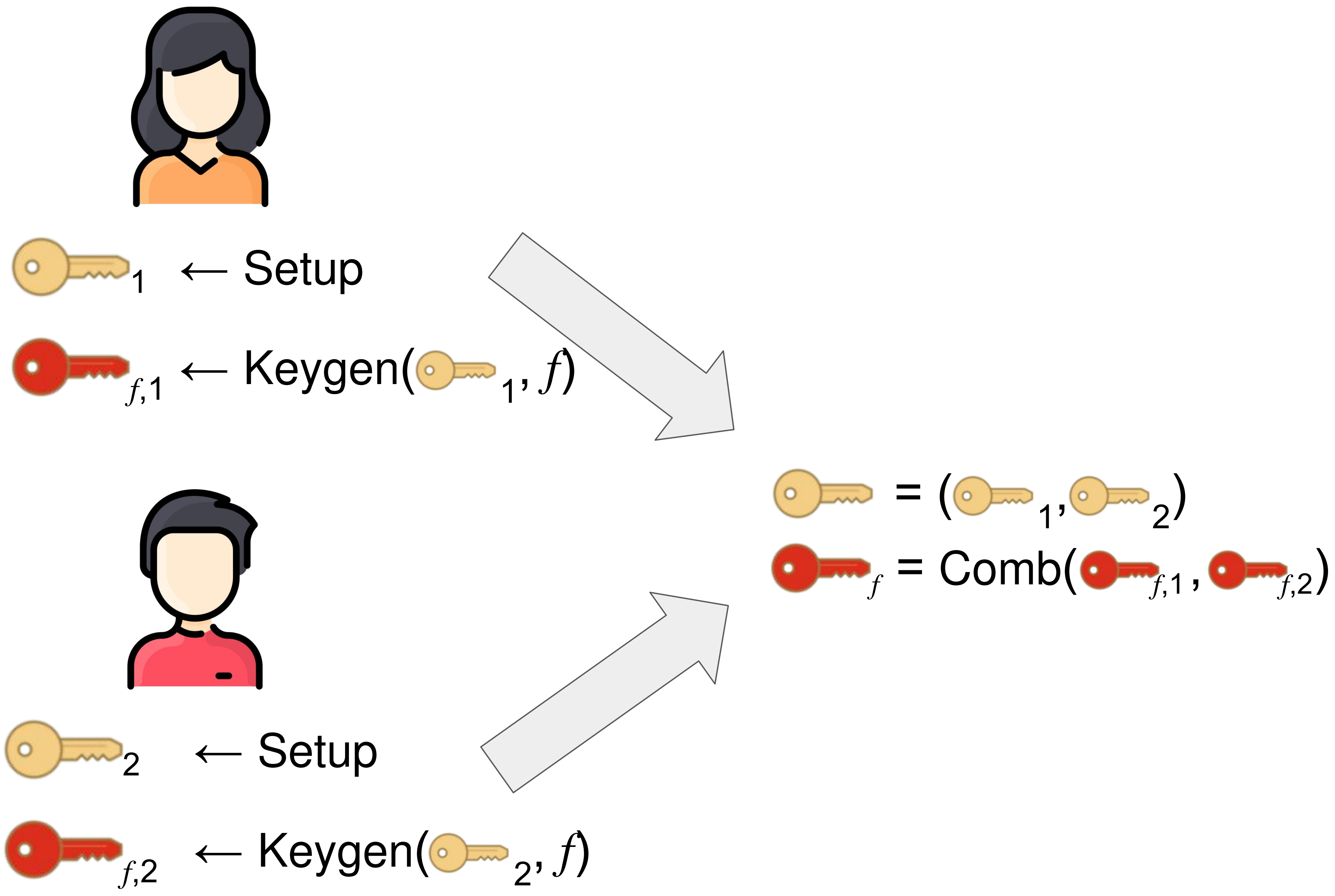
$_{f,1} \leftarrow$ Keygen($_1$, f)



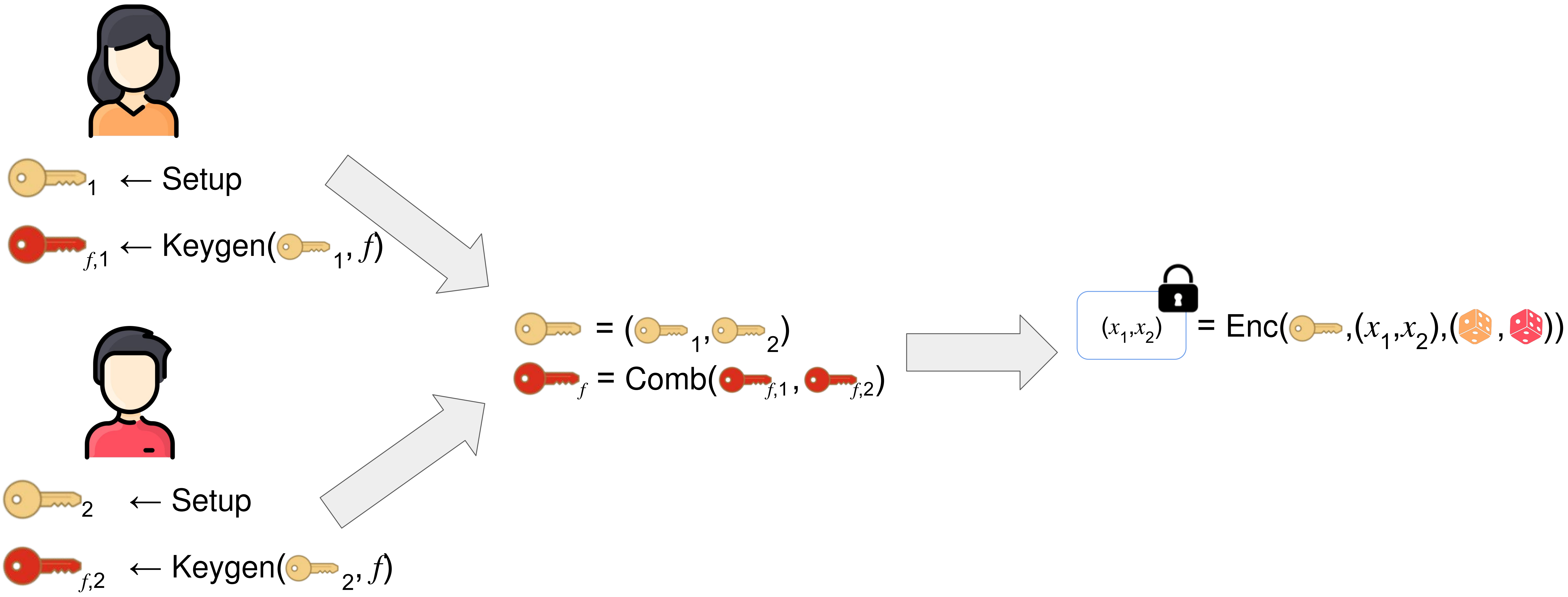
$_2 \leftarrow$ Setup

$_{f,2} \leftarrow$ Keygen($_2$, f)

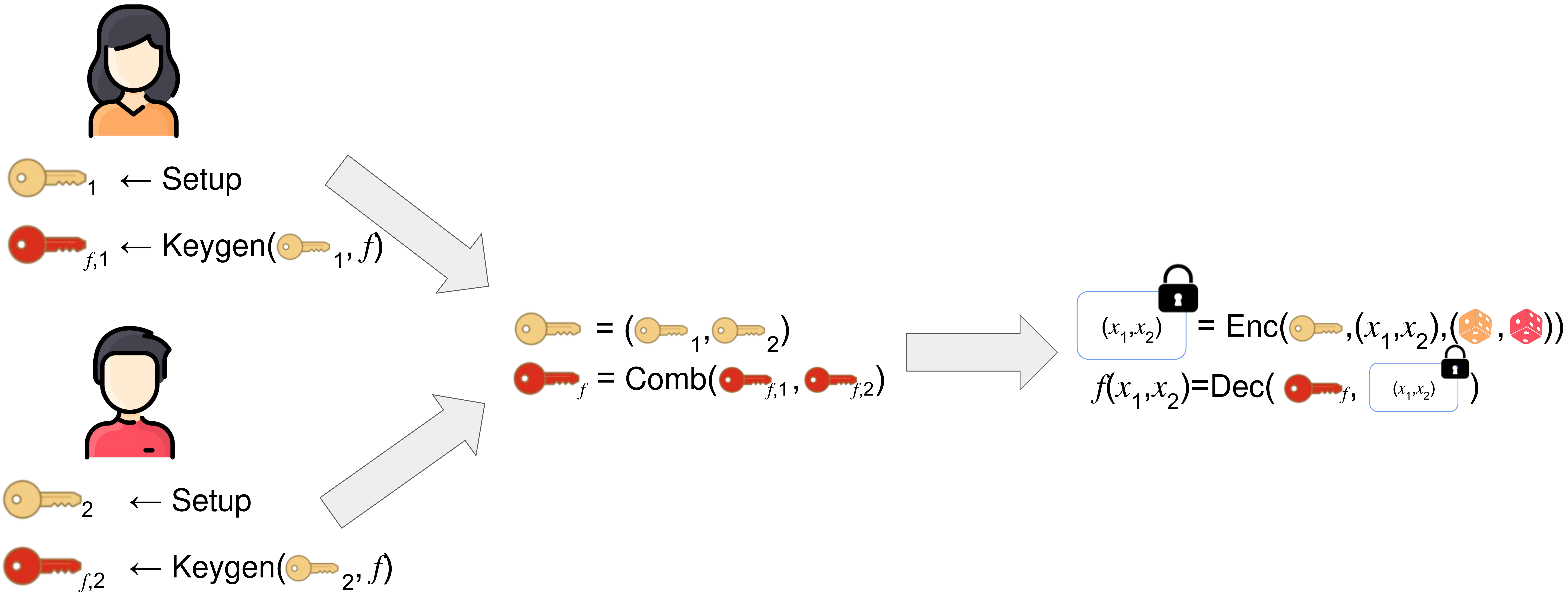
(Decomposable) Functional Encryption Combiner [ABJ+19]



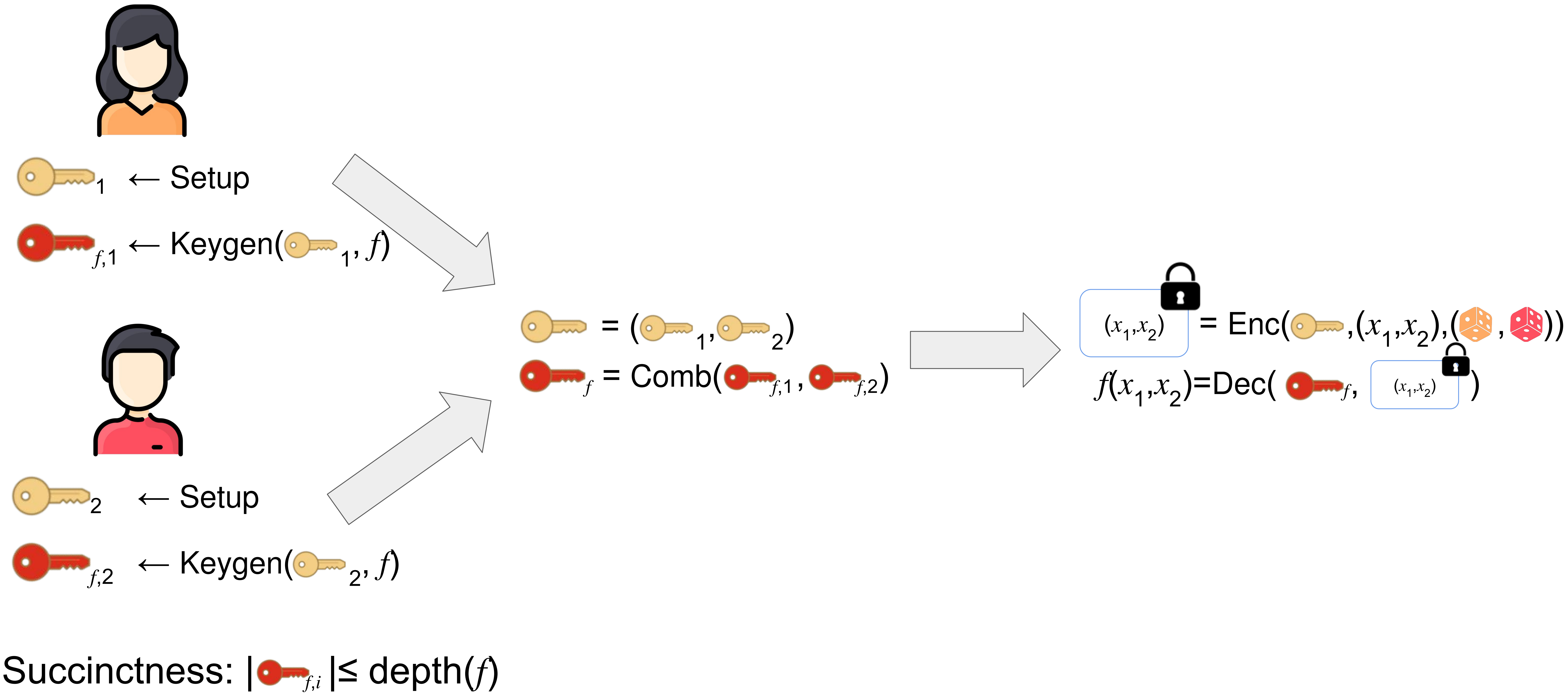
(Decomposable) Functional Encryption Combiner [ABJ+19]



(Decomposable) Functional Encryption Combiner [ABJ+19]

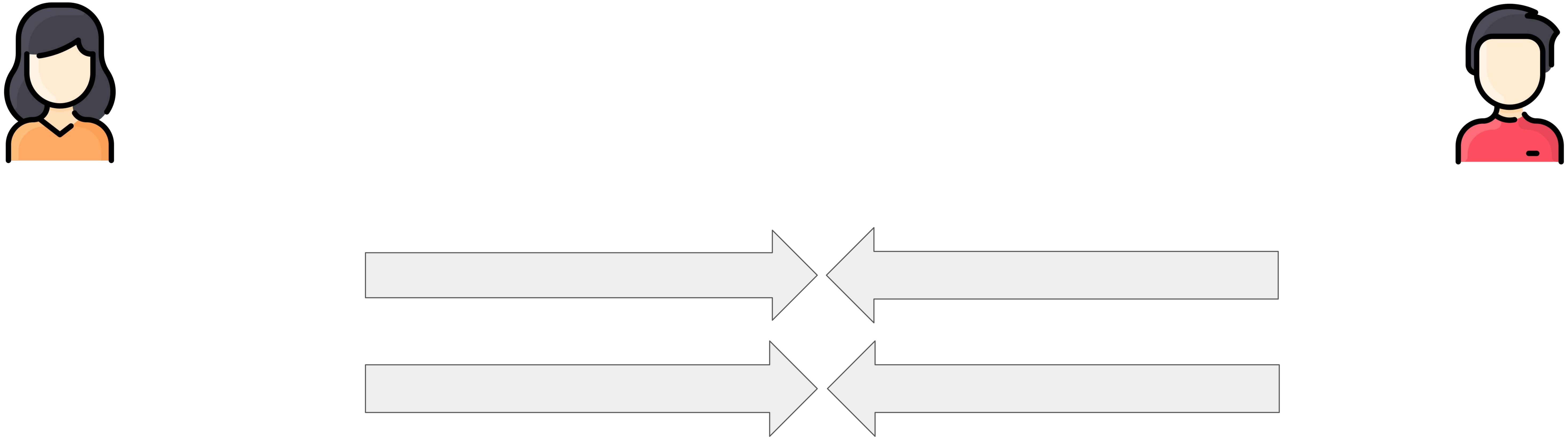


(Decomposable) Functional Encryption Combiner [ABJ+19]

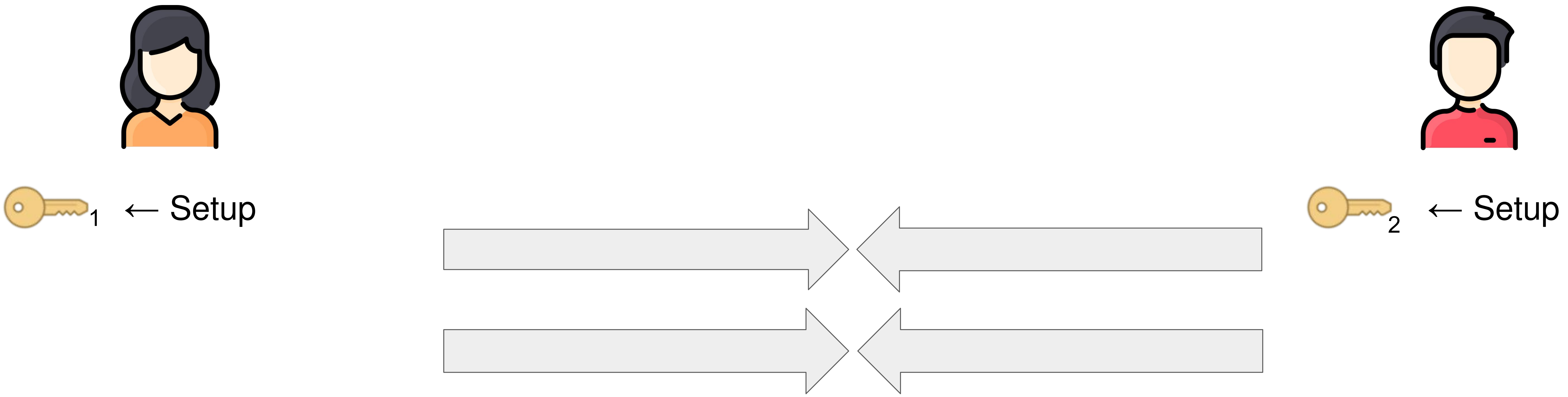


Protocol of Ananth et al. [ABJ+19]

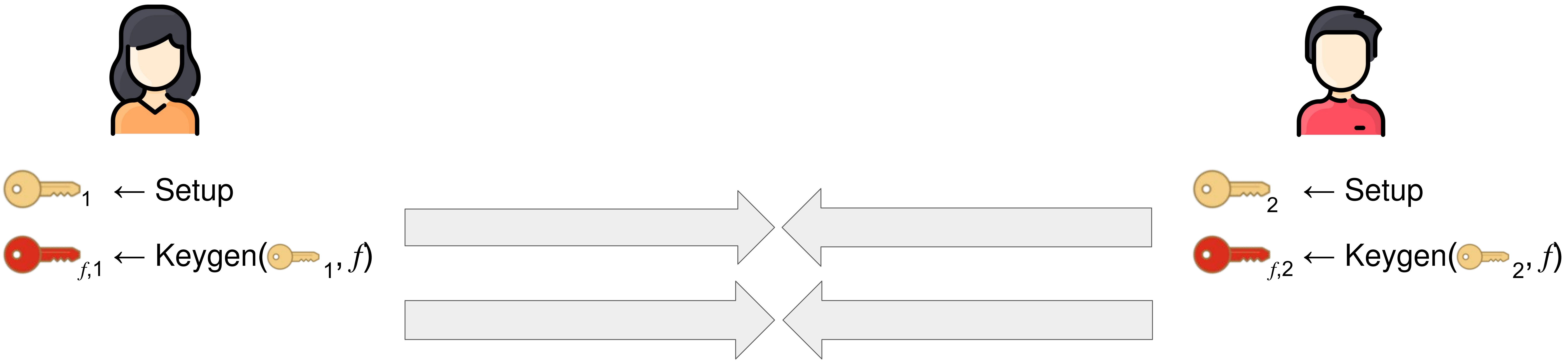
Protocol of Ananth et al. [ABJ+19]



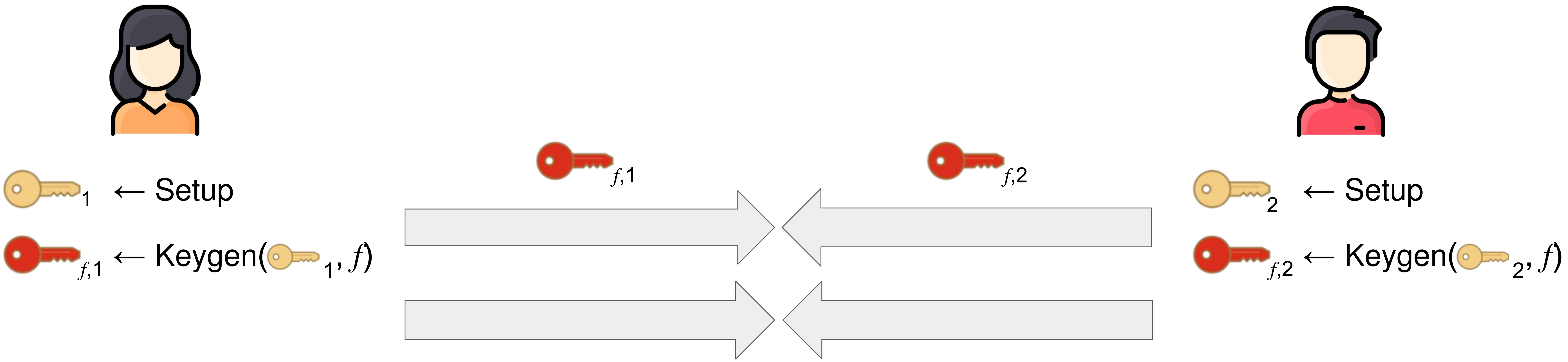
Protocol of Ananth et al. [ABJ+19]



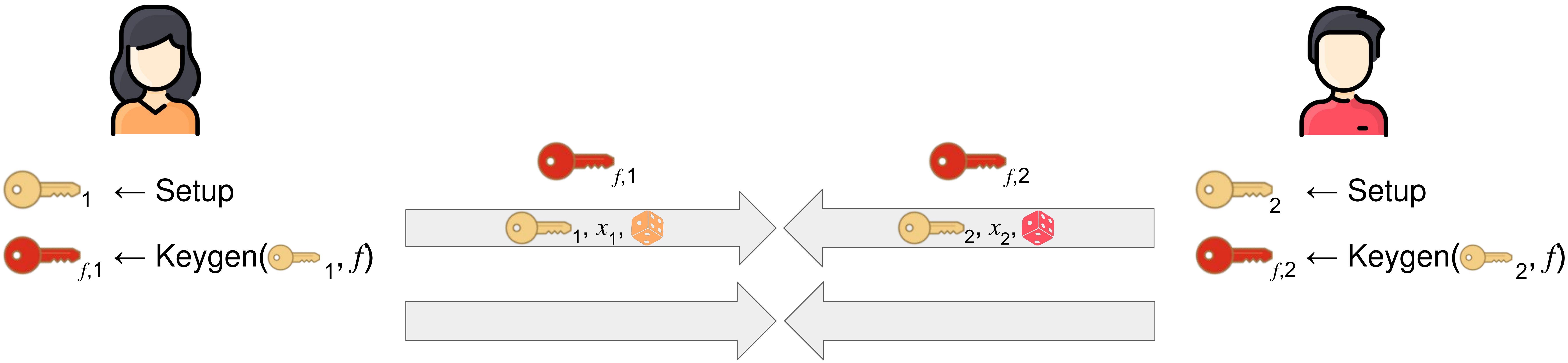
Protocol of Ananth et al. [ABJ+19]



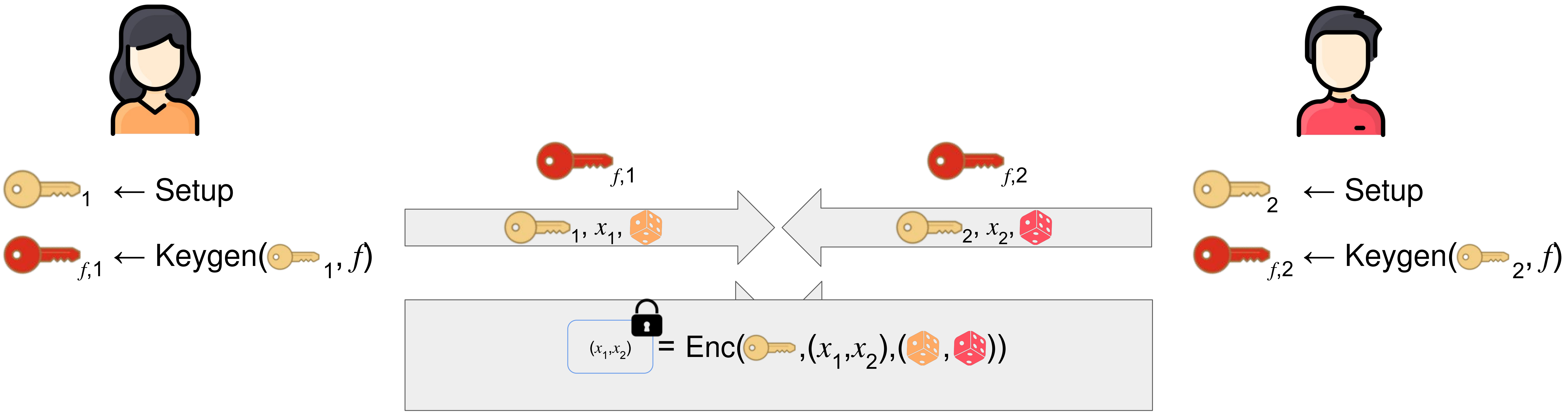
Protocol of Ananth et al. [ABJ+19]



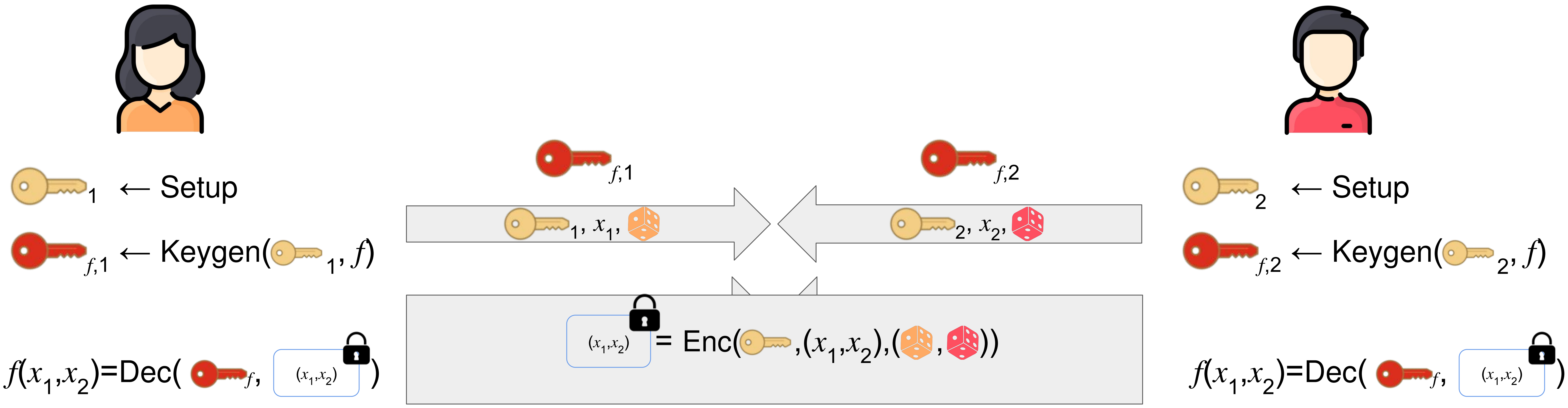
Protocol of Ananth et al. [ABJ+19]



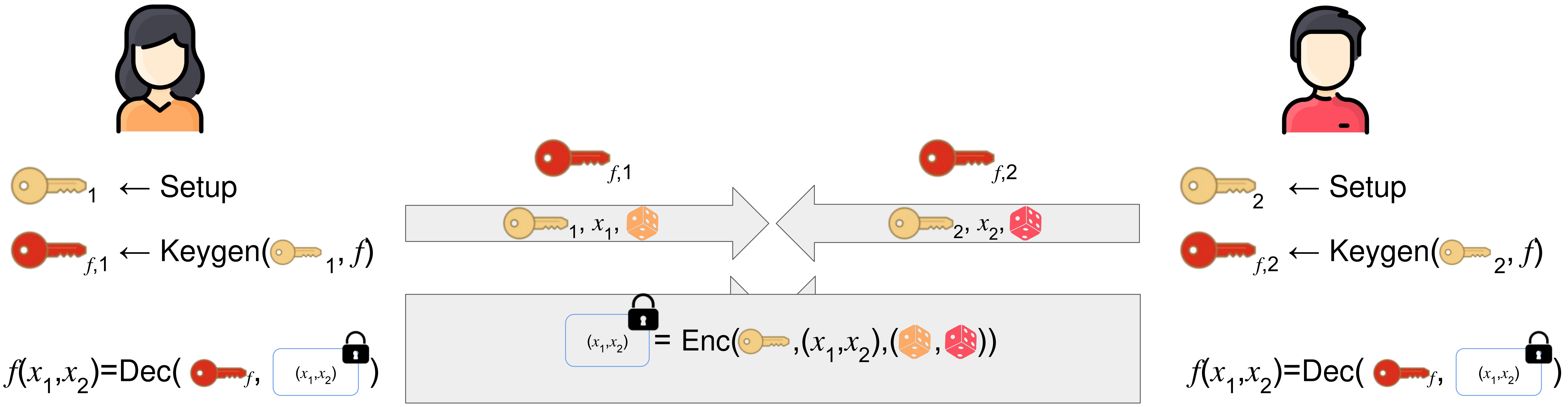
Protocol of Ananth et al. [ABJ+19]



Protocol of Ananth et al. [ABJ+19]

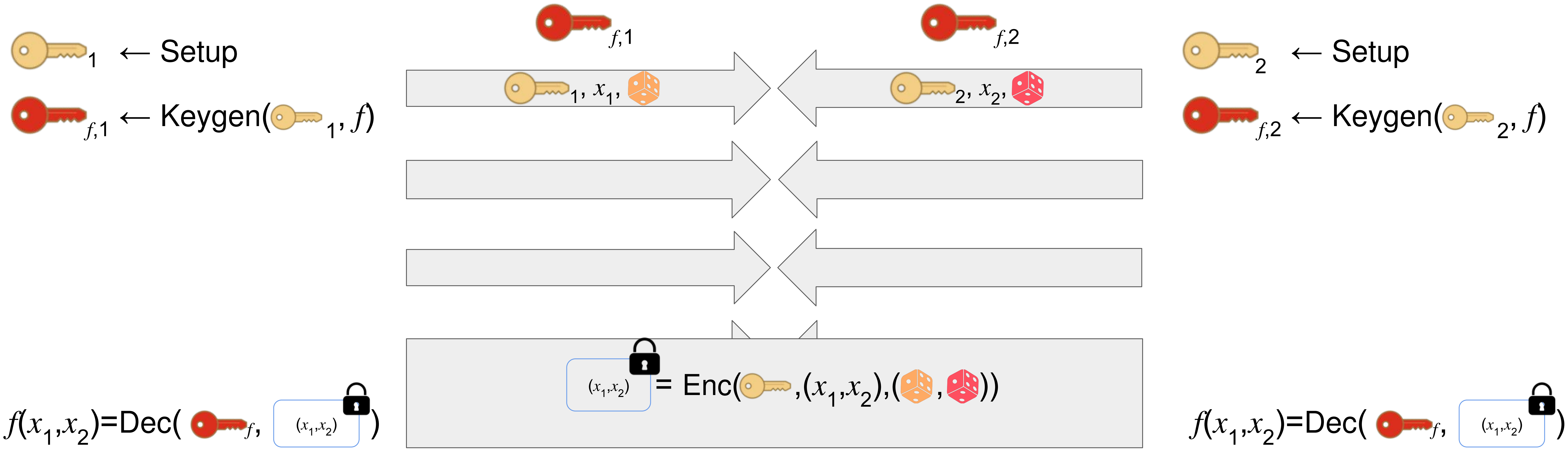


Protocol of Ananth et al. [ABJ+19]

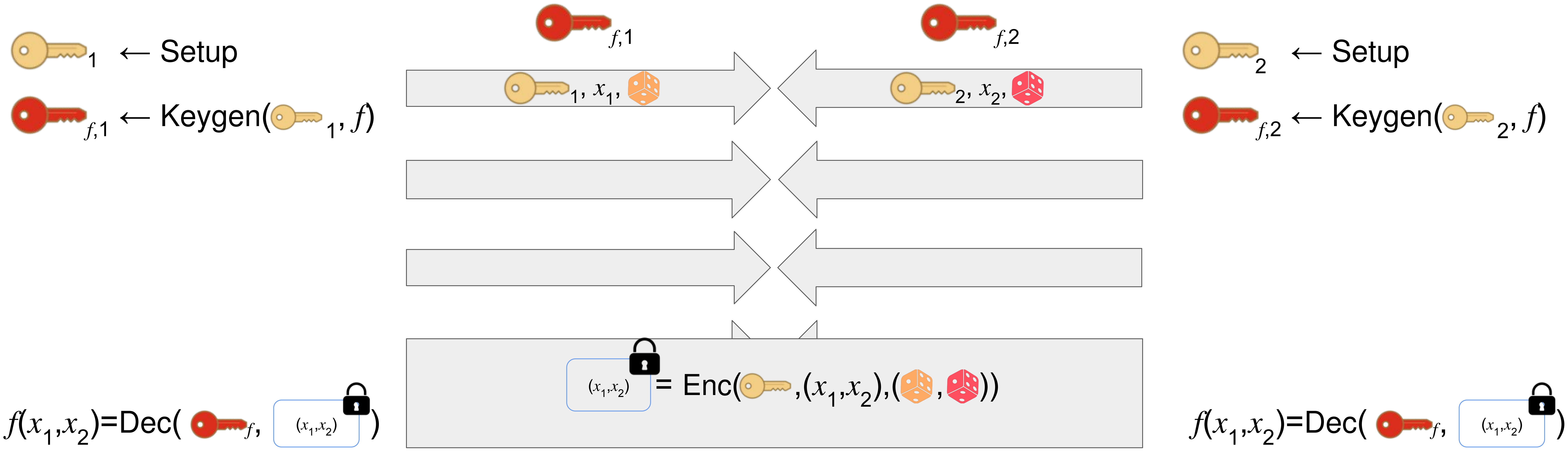


⇒ Replace semi-honest protocol with maliciously secure protocol

First Approach

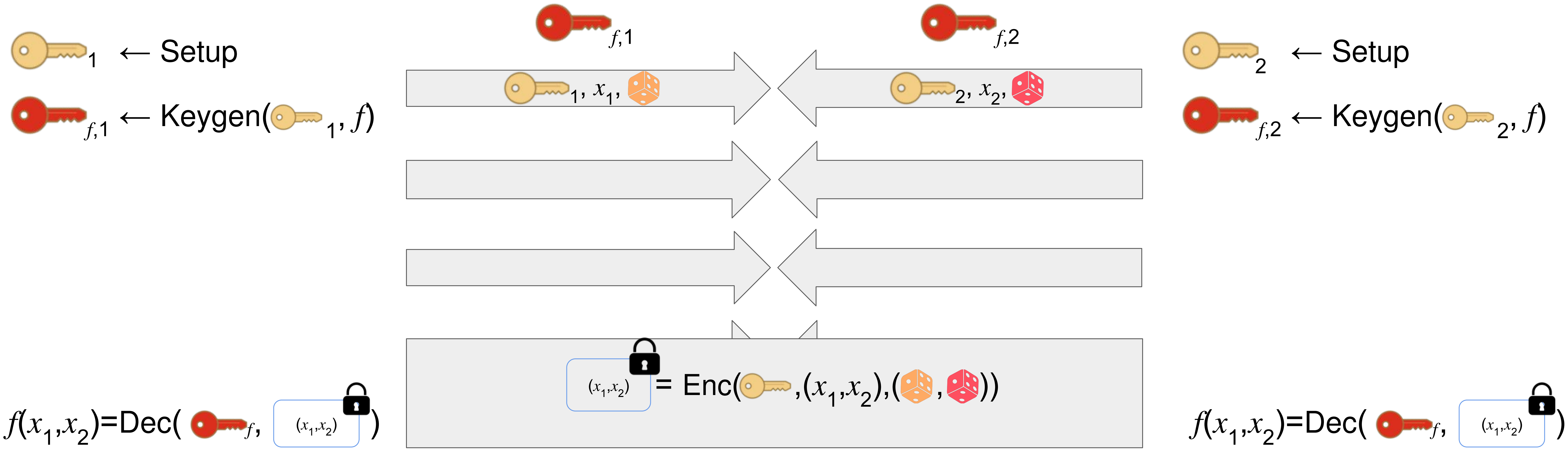


First Approach



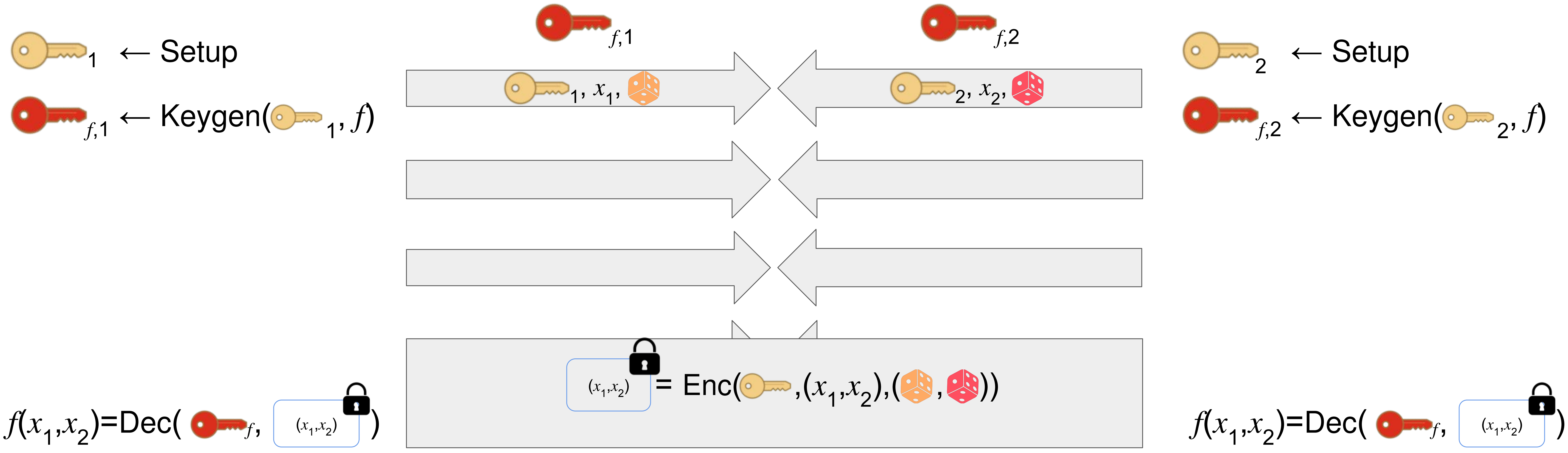
1. ${}_{f,i}$ can be generated maliciously

First Approach



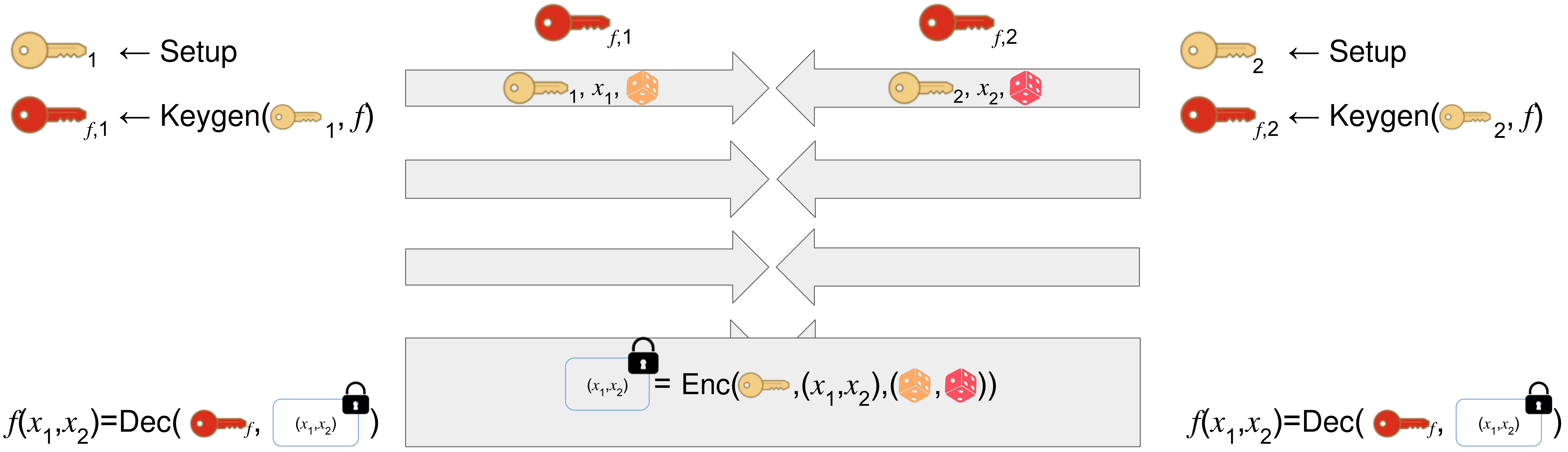
1. $_{f,i}$ can be generated maliciously
2. (,) used for encryption can be “bad”

First Approach



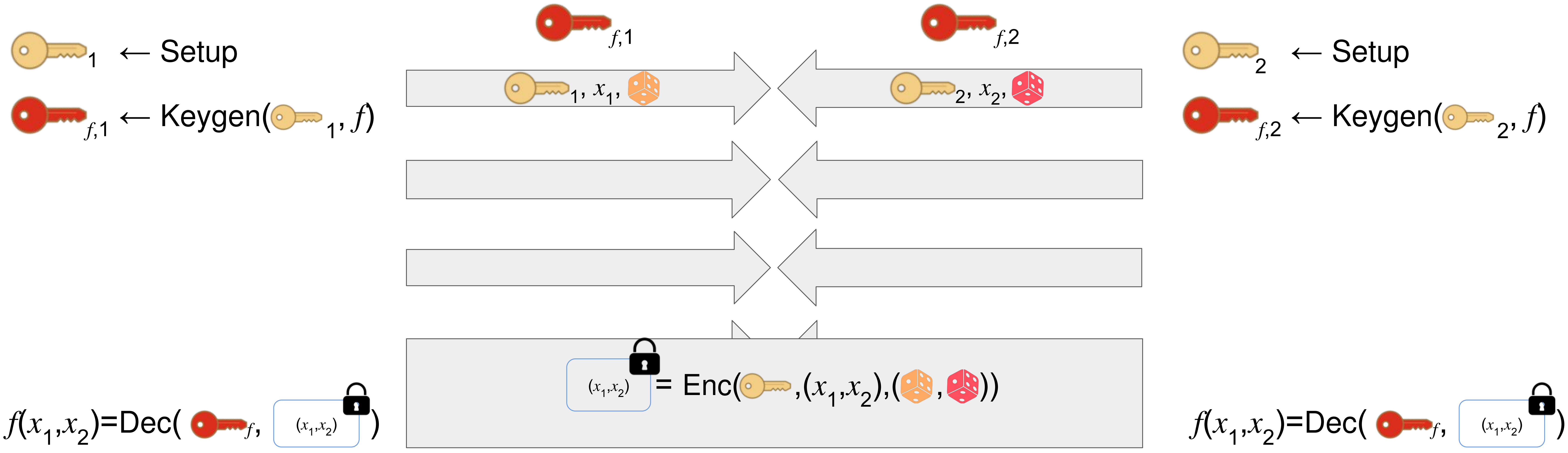
1. $_{f,i}$ can be generated maliciously
2. (,) used for encryption can be "bad"
3. $_i$ can be generated arbitrarily

First Approach



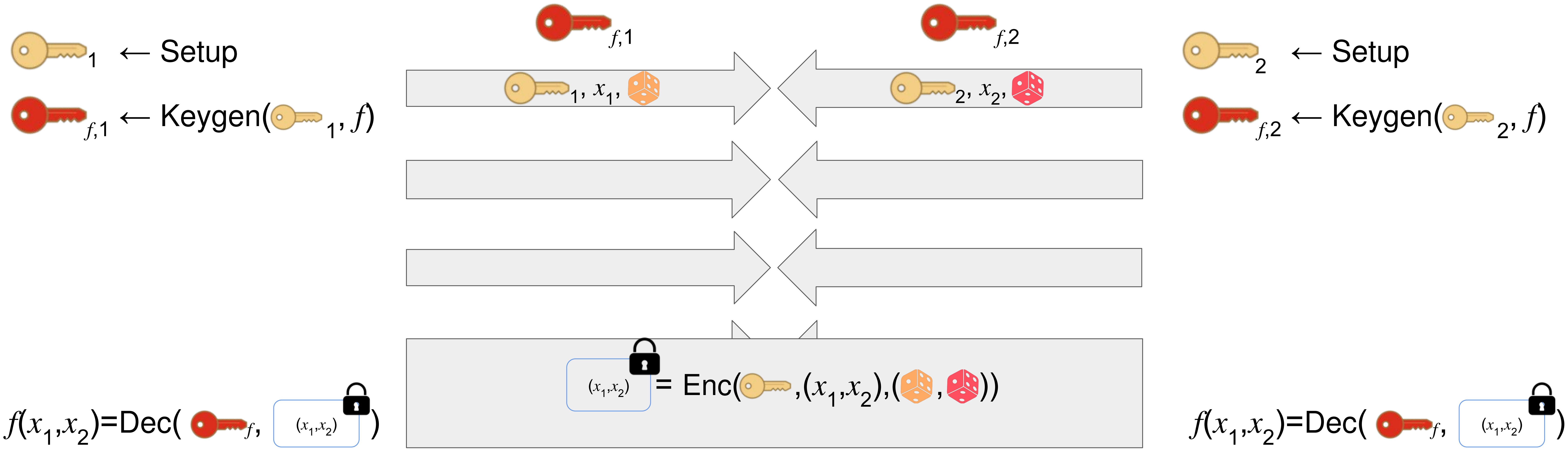
1. $_{f,i}$ can be generated maliciously

First Approach



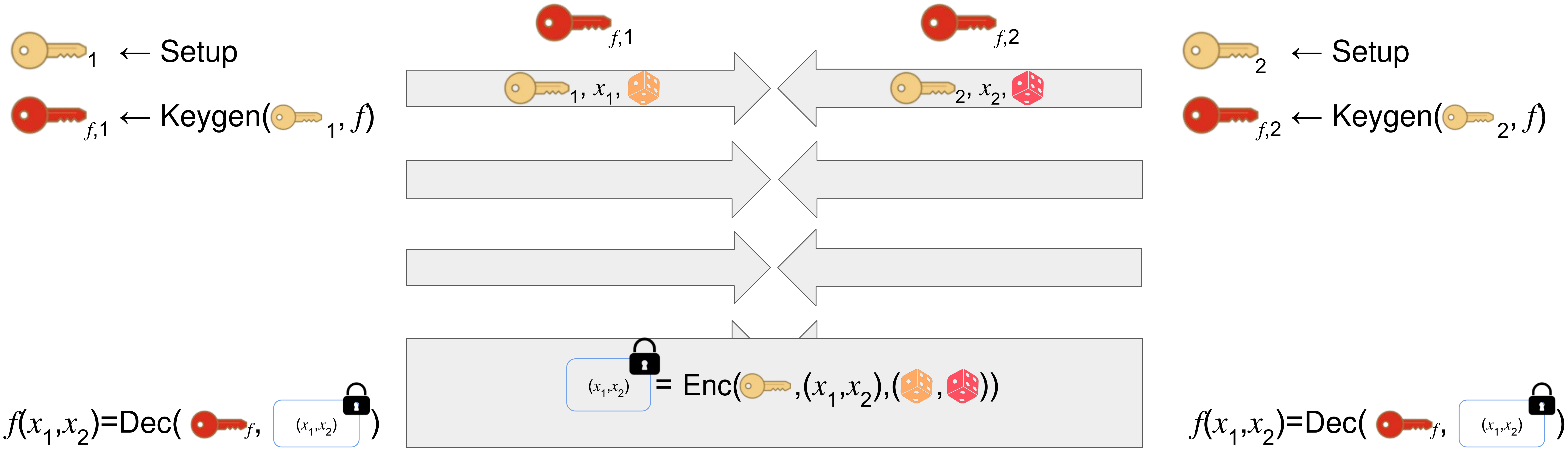
1. $_{f,i}$ can be generated maliciously
   → Privacy with Knowledge of Outputs

First Approach



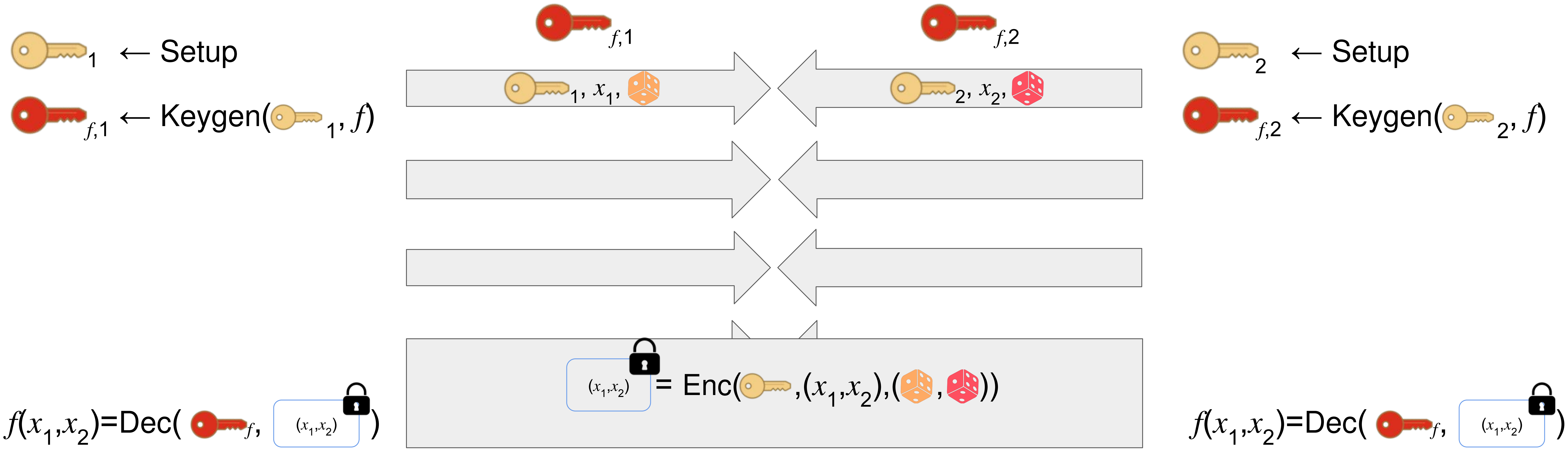
1. $_{f,i}$ can be generated maliciously
   → Privacy with Knowledge of Outputs
   → Can be lifted using [IKP10,PC12]

First Approach



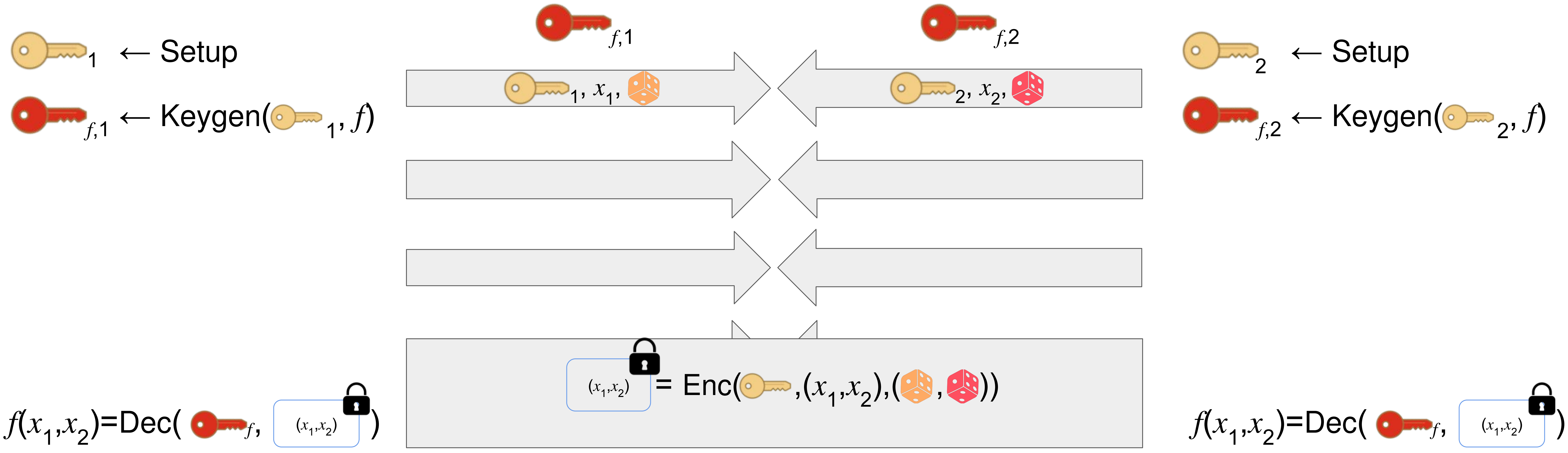
1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be "bad"
3. $_{i}$ can be generated arbitrarily

First Approach



2.(,) used for encryption can be “bad”

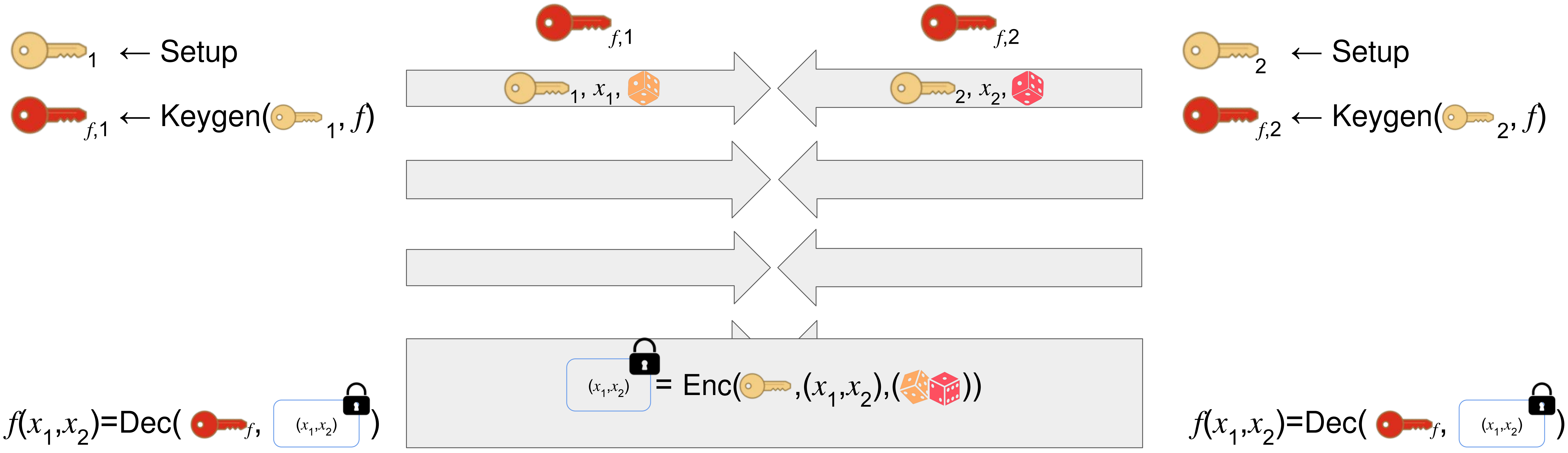
First Approach



2.(,) used for encryption can be “bad”

→ Use XOR instead of concatenation

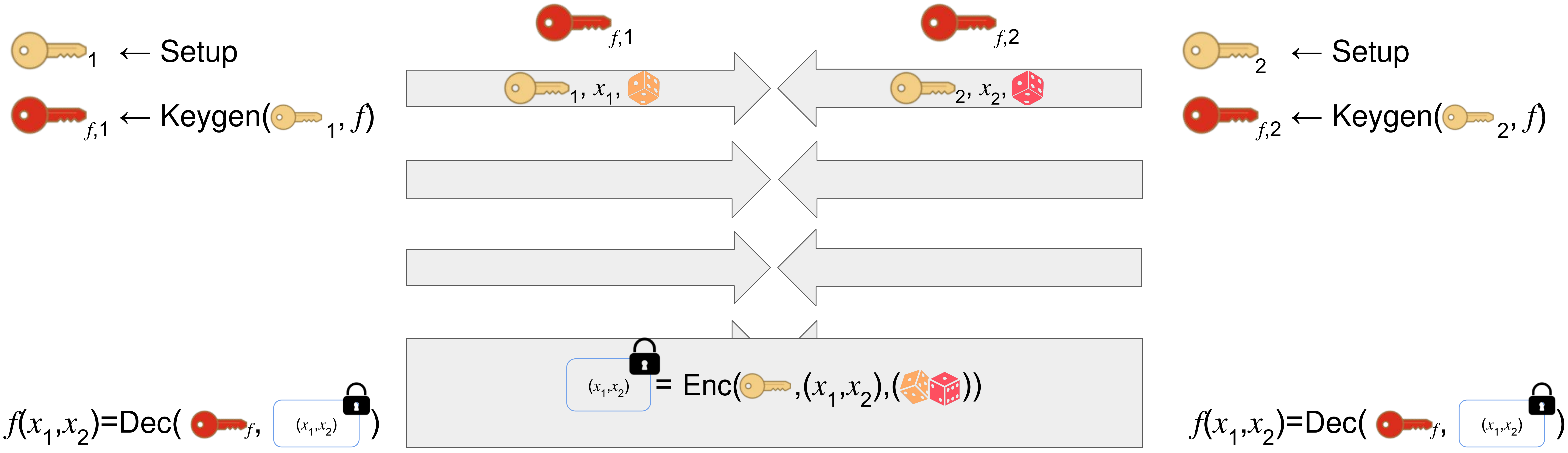
First Approach



2.(,) used for encryption can be “bad”

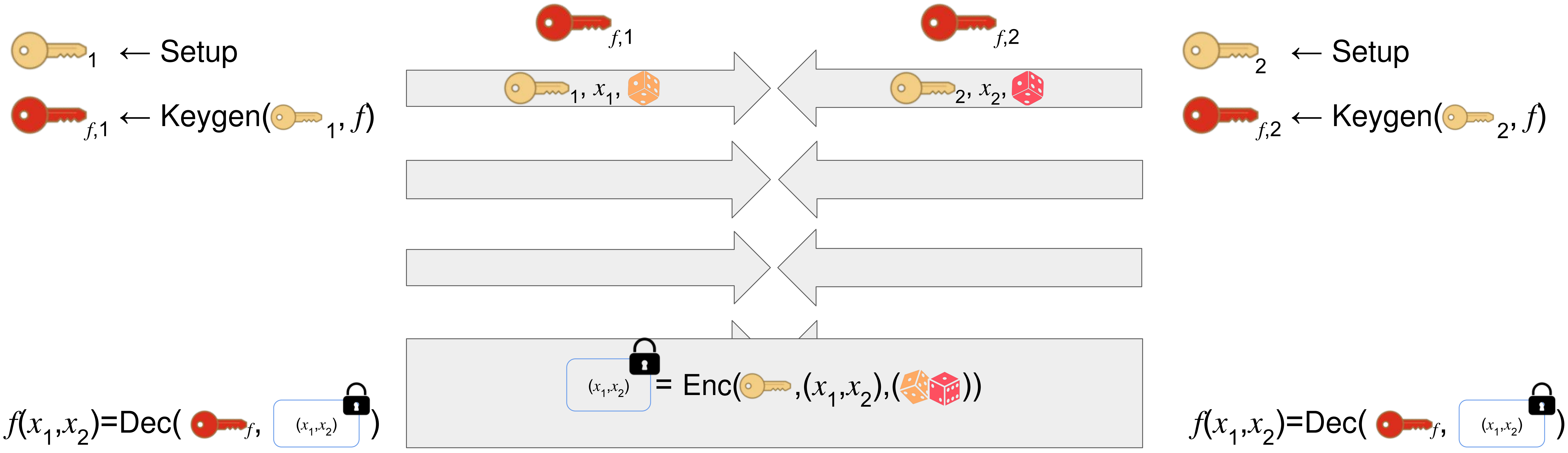
→ Use XOR instead of concatenation

First Approach



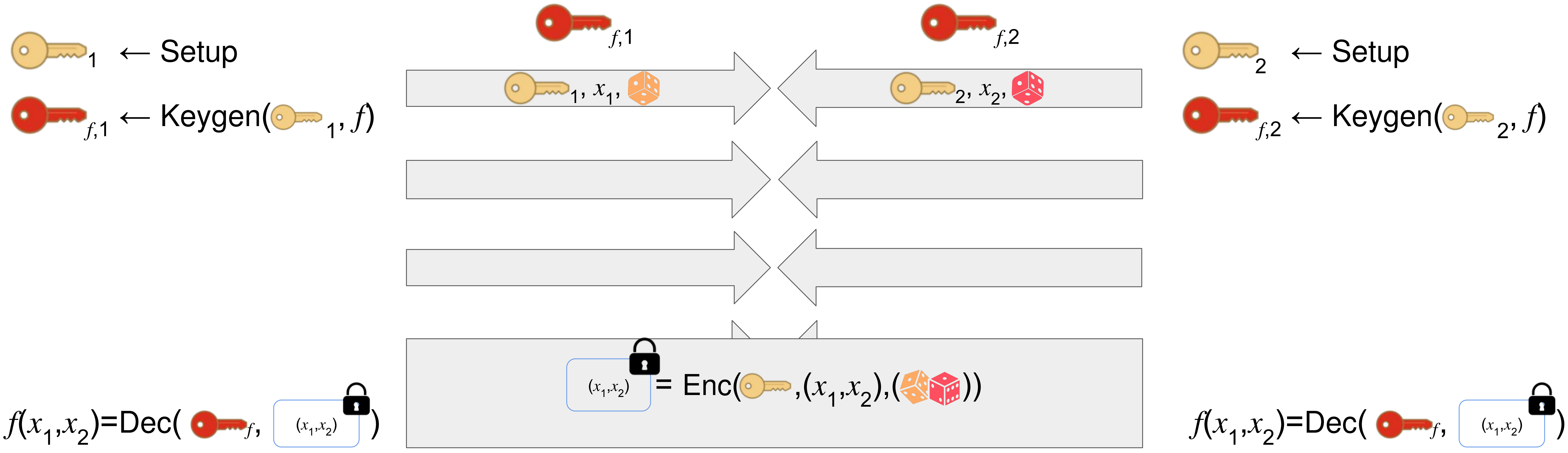
1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be “bad” ✓
3. $_i$ can be generated arbitrarily

First Approach



3. $_i$ can be generated arbitrarily

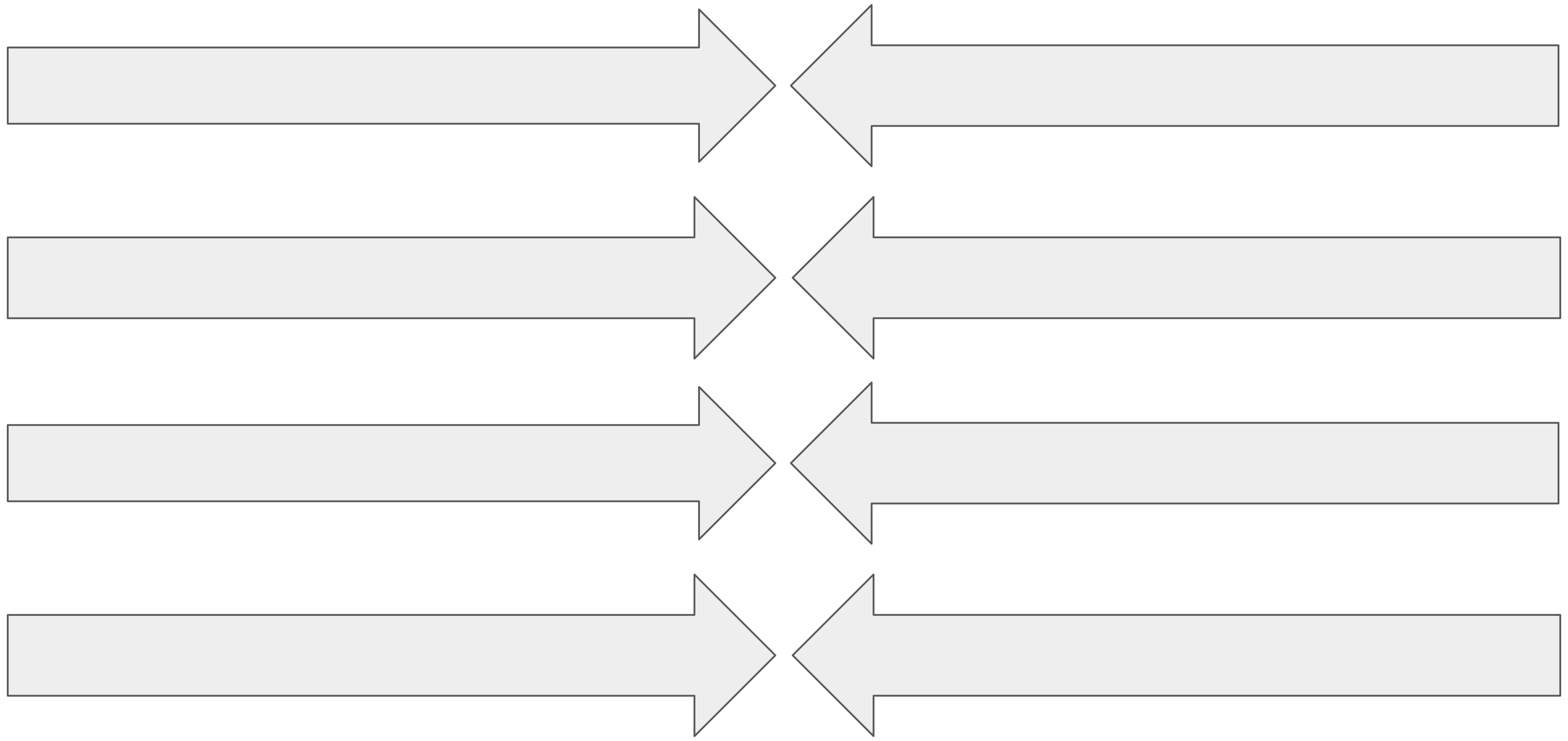
First Approach



3.  $_i$ can be generated arbitrarily

→ Solve as 2.

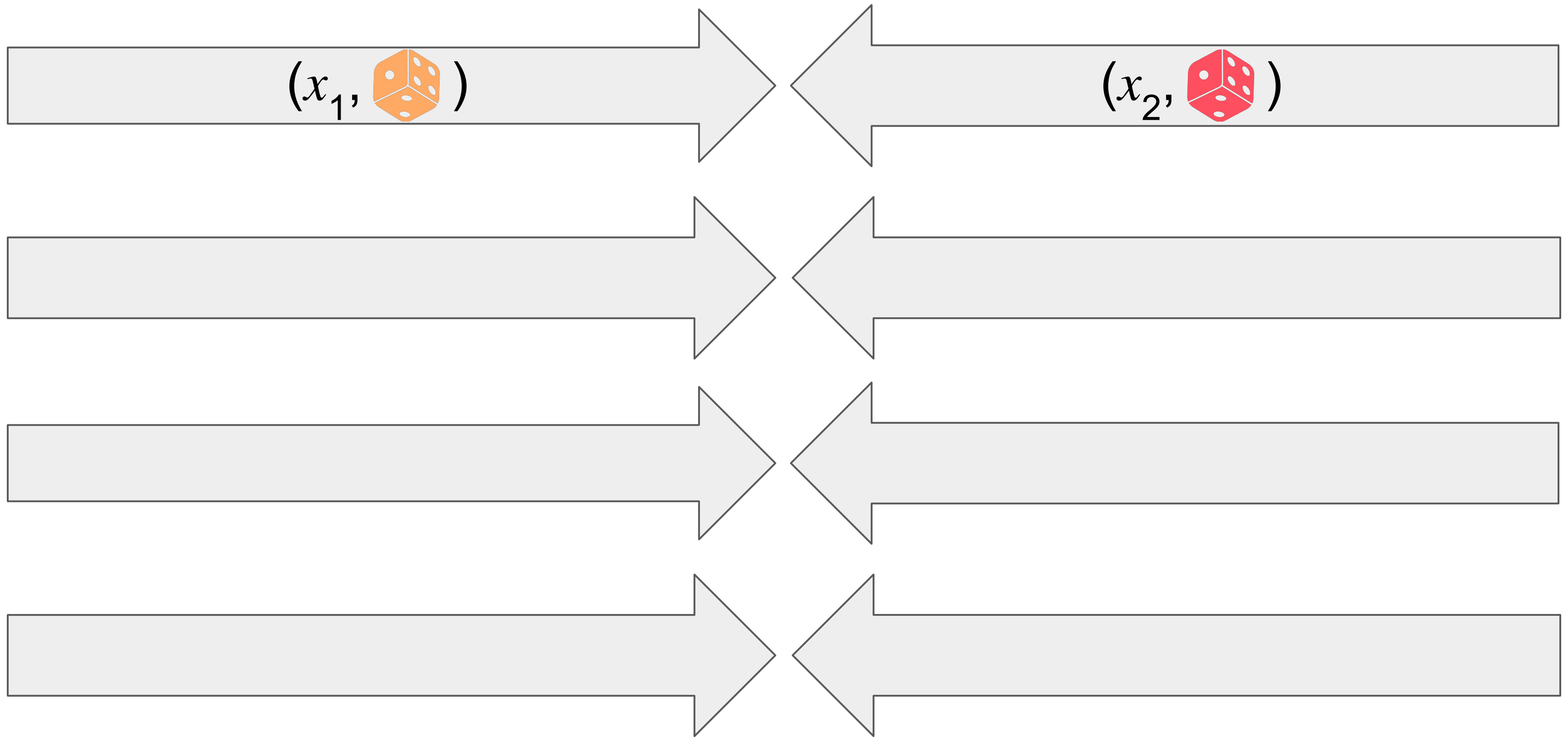
First Approach



3.  $_i$ can be generated arbitrarily

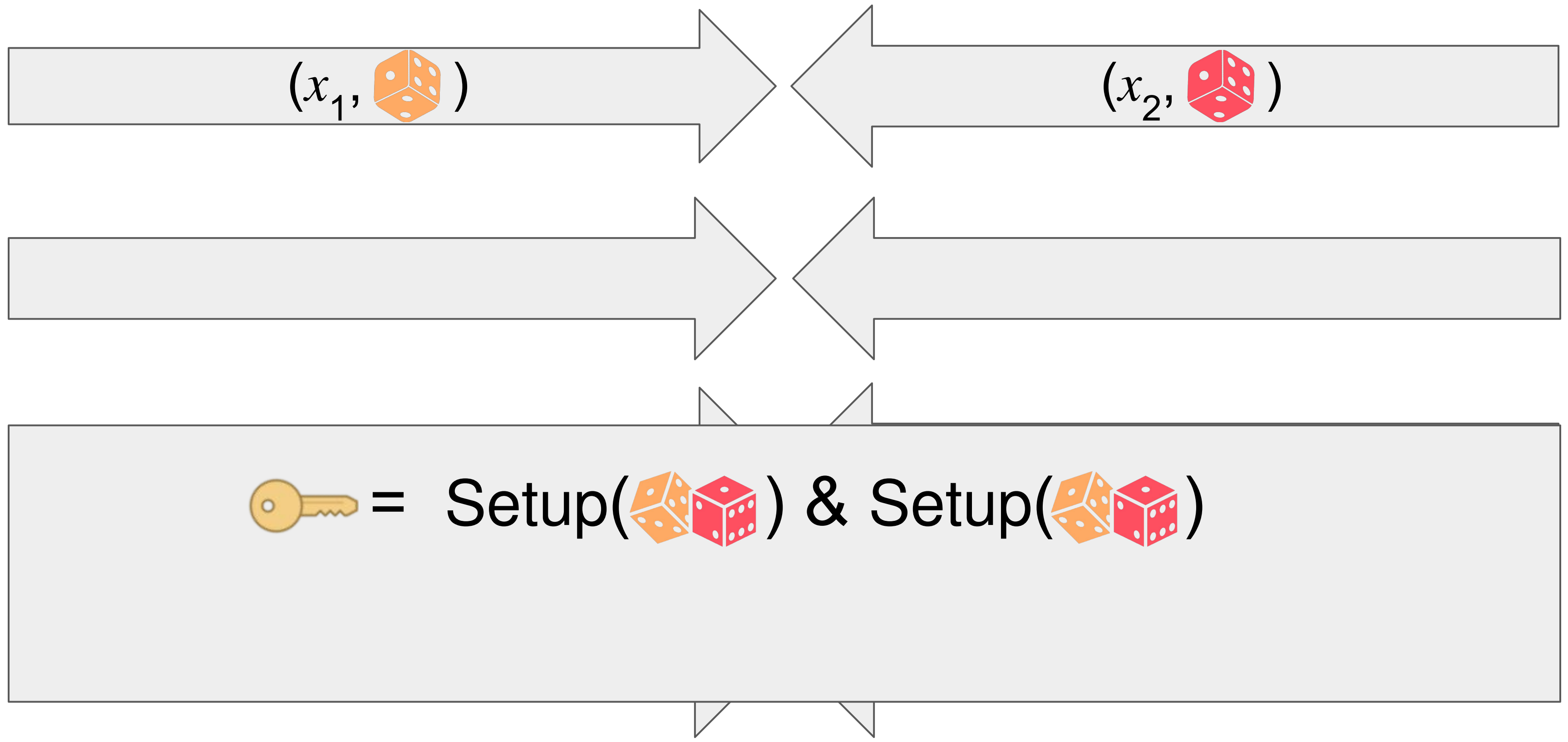
→ Solve as 2.

First Approach



3.  $_i$ can be generated arbitrarily
   → Solve as 2.

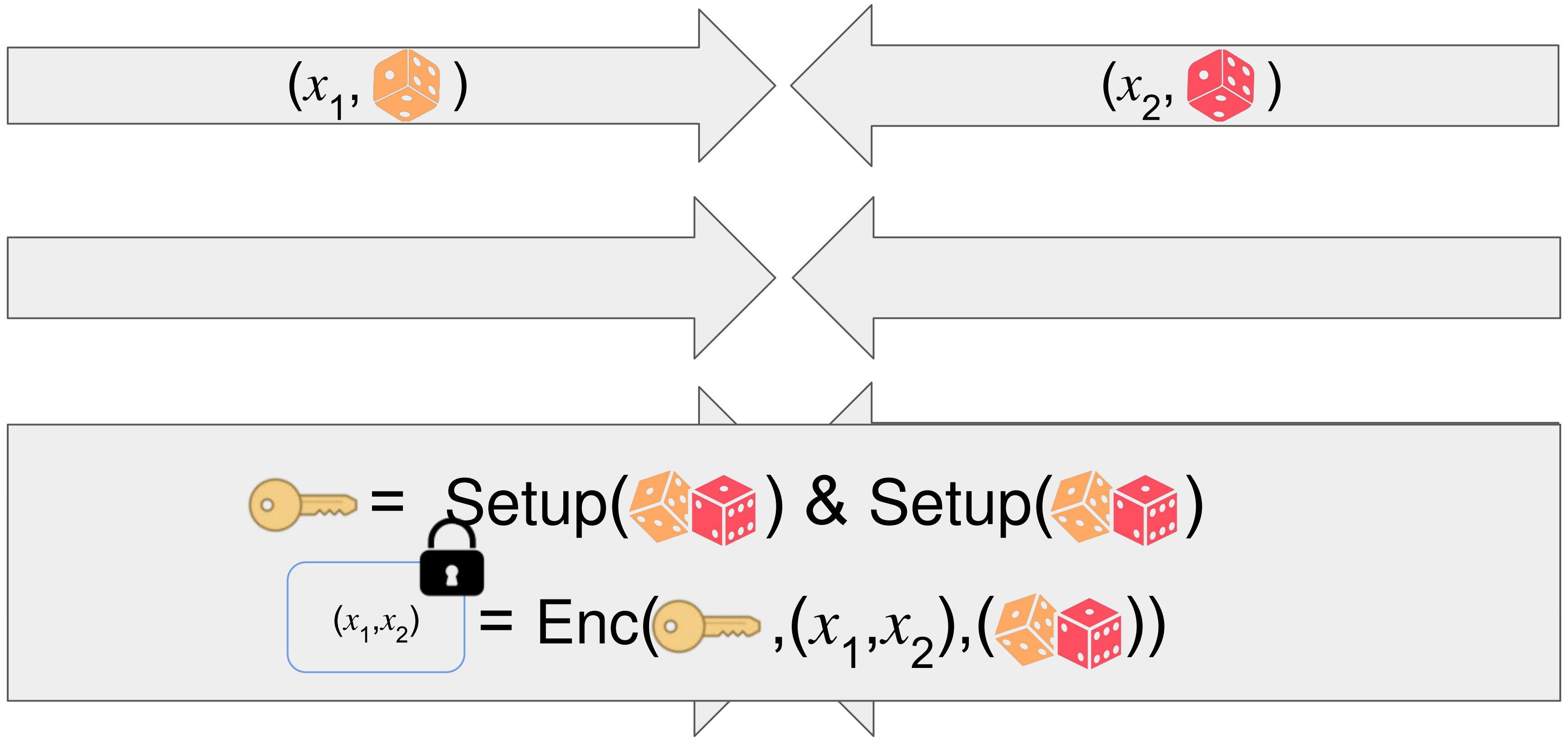
First Approach



3. ${}_i$ can be generated arbitrarily

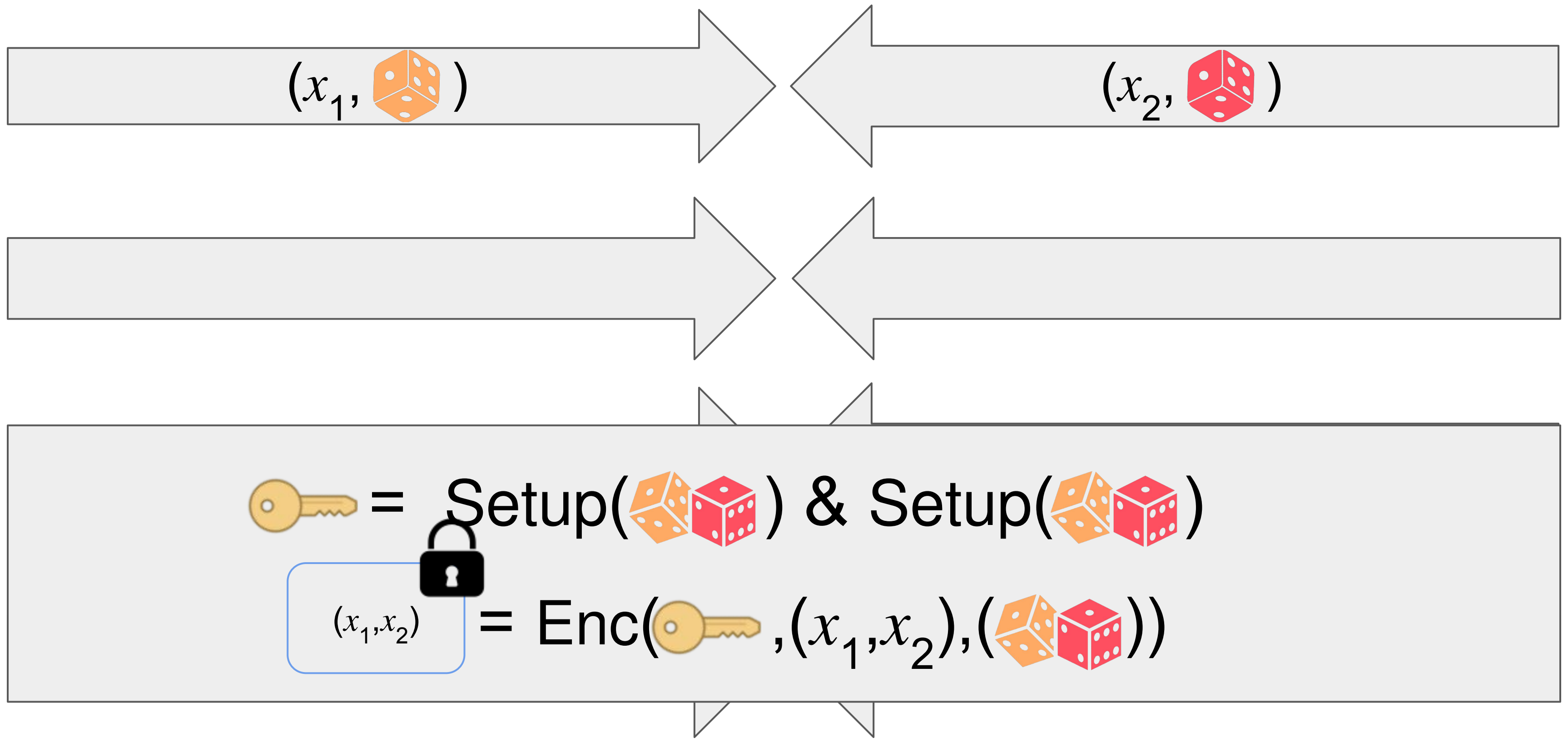
$\rightarrow$ Solve as 2.

First Approach



3. $_i$ can be generated arbitrarily
   → Solve as 2.

First Approach



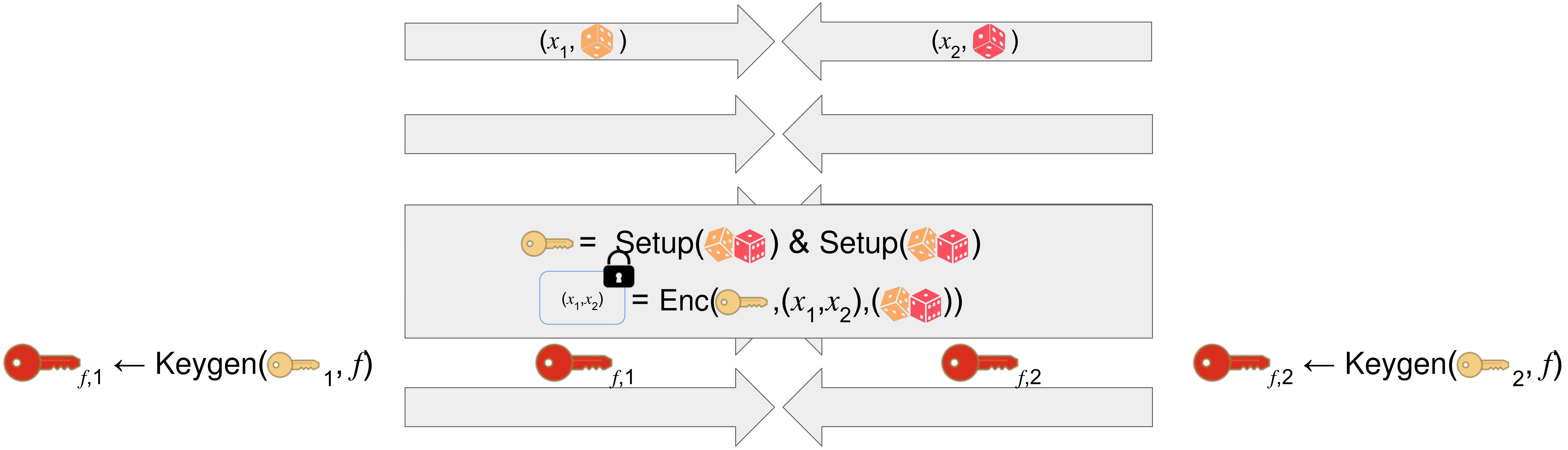
$_{f,1} \leftarrow$ Keygen($_1$, f)

$_{f,2} \leftarrow$ Keygen($_2$, f)

3. $_i$ can be generated arbitrarily

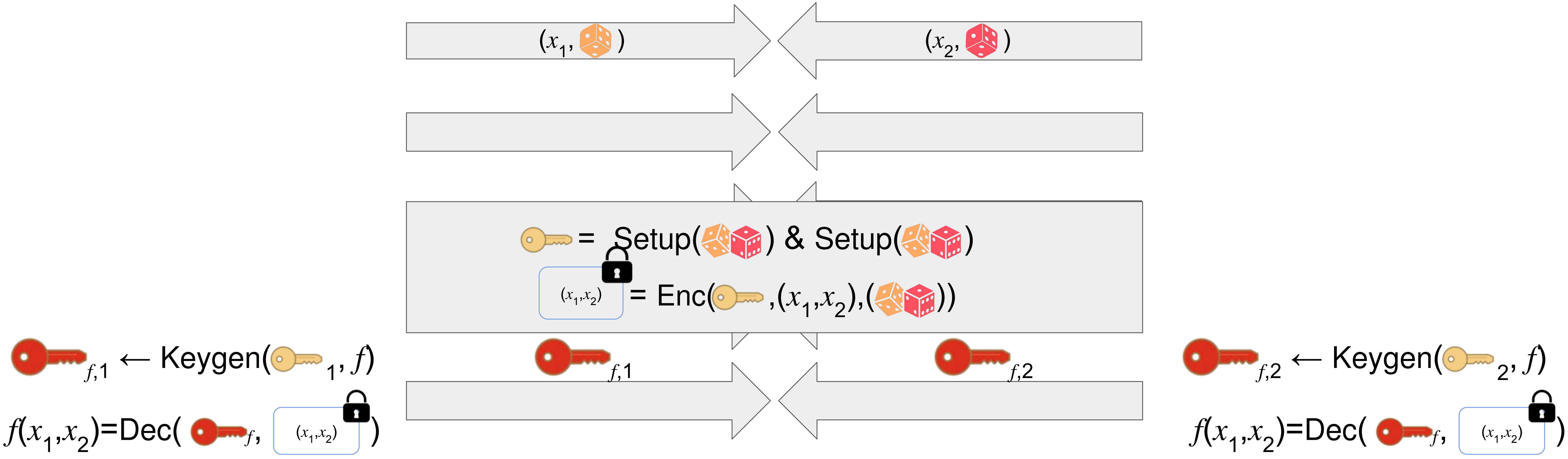
$\rightarrow$ Solve as 2.

First Approach



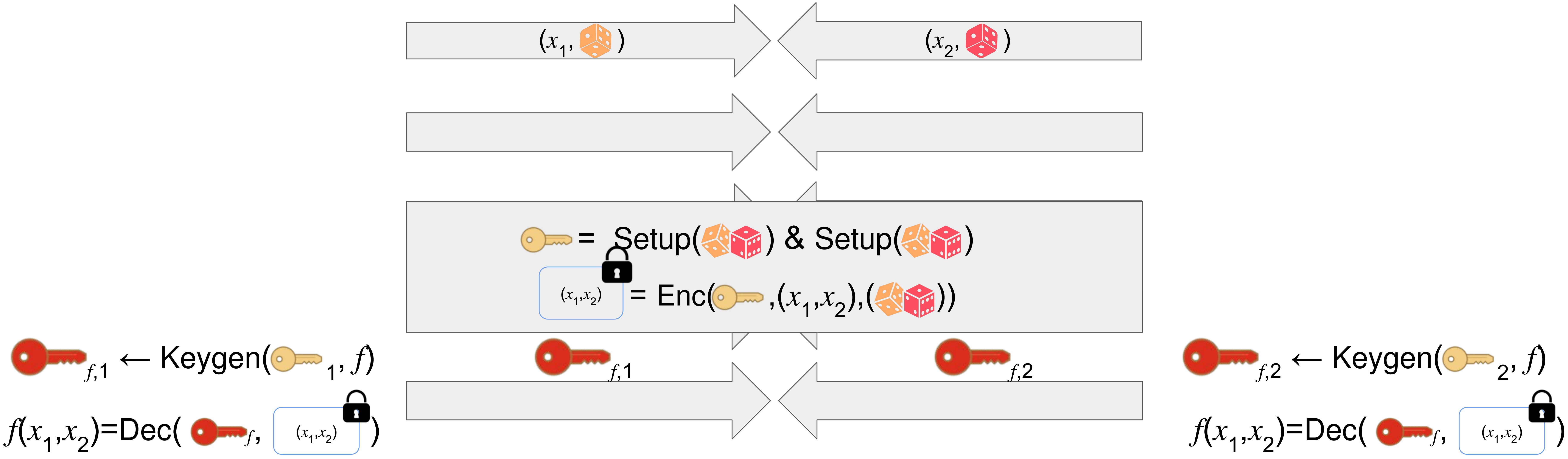
3. $_i$ can be generated arbitrarily
→ Solve as 2.

First Approach



3. $_i$ can be generated arbitrarily
$\rightarrow$ Solve as 2.

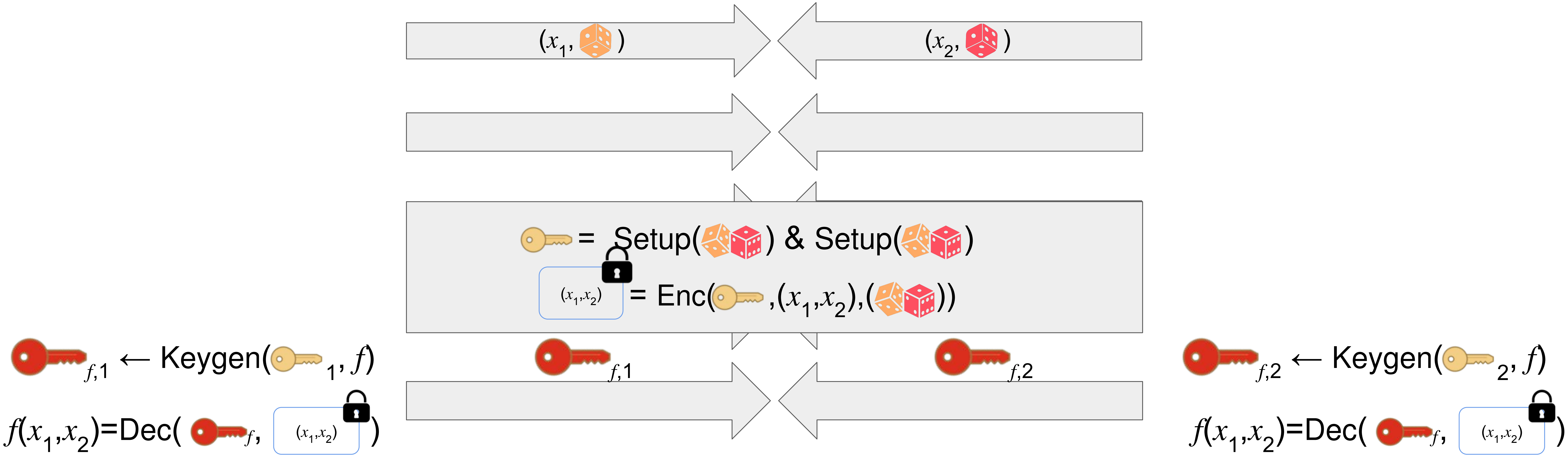
First Approach



3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

First Approach

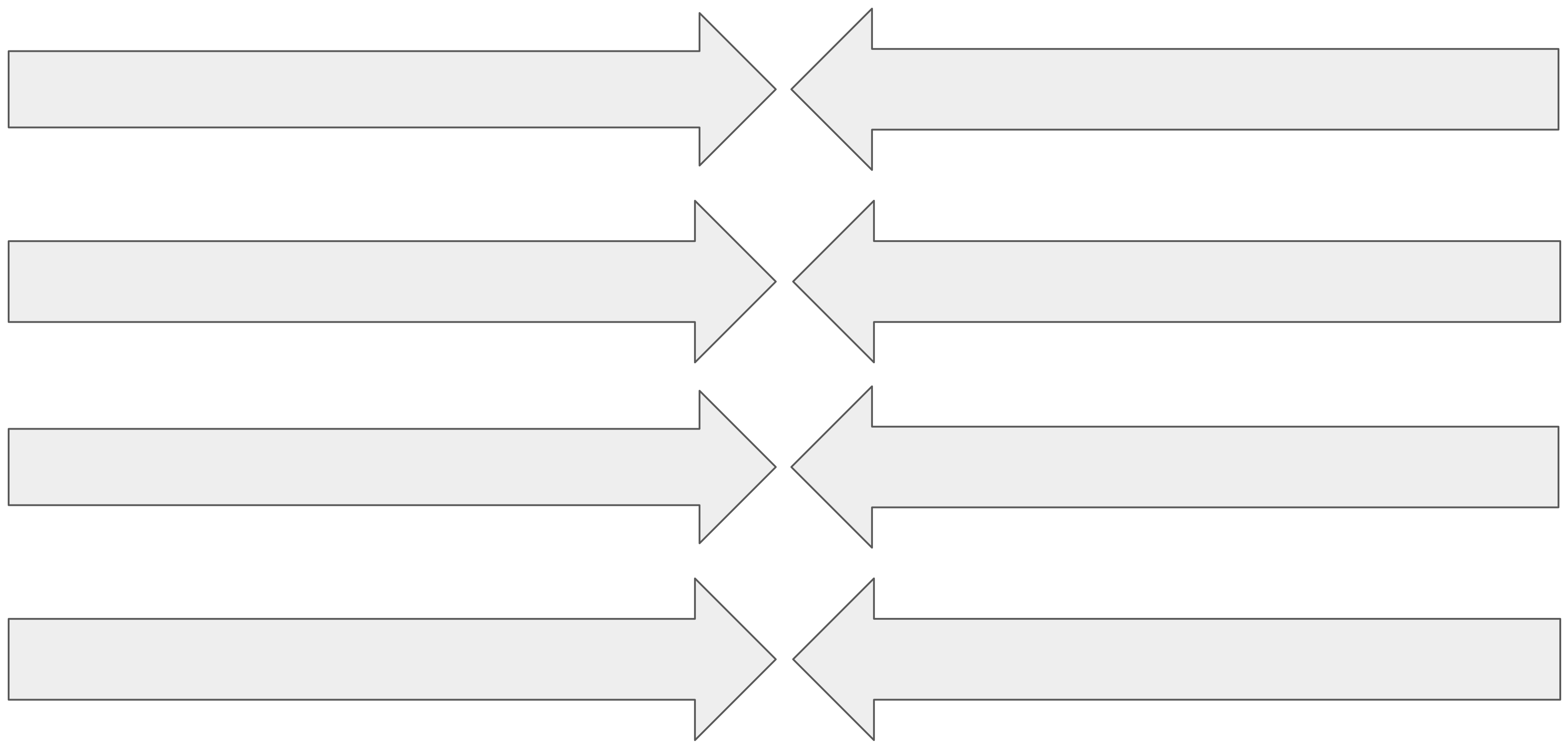


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

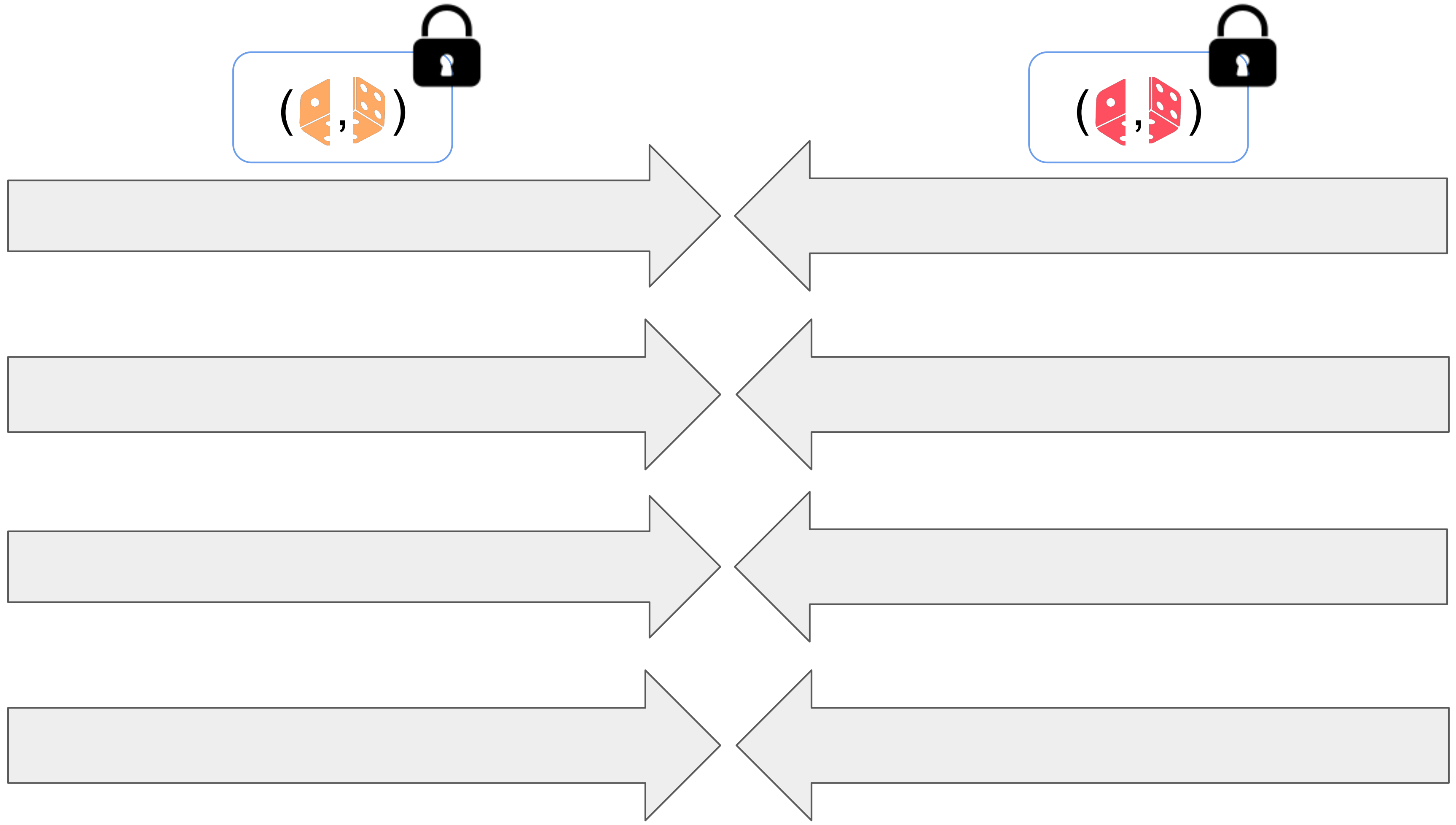


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

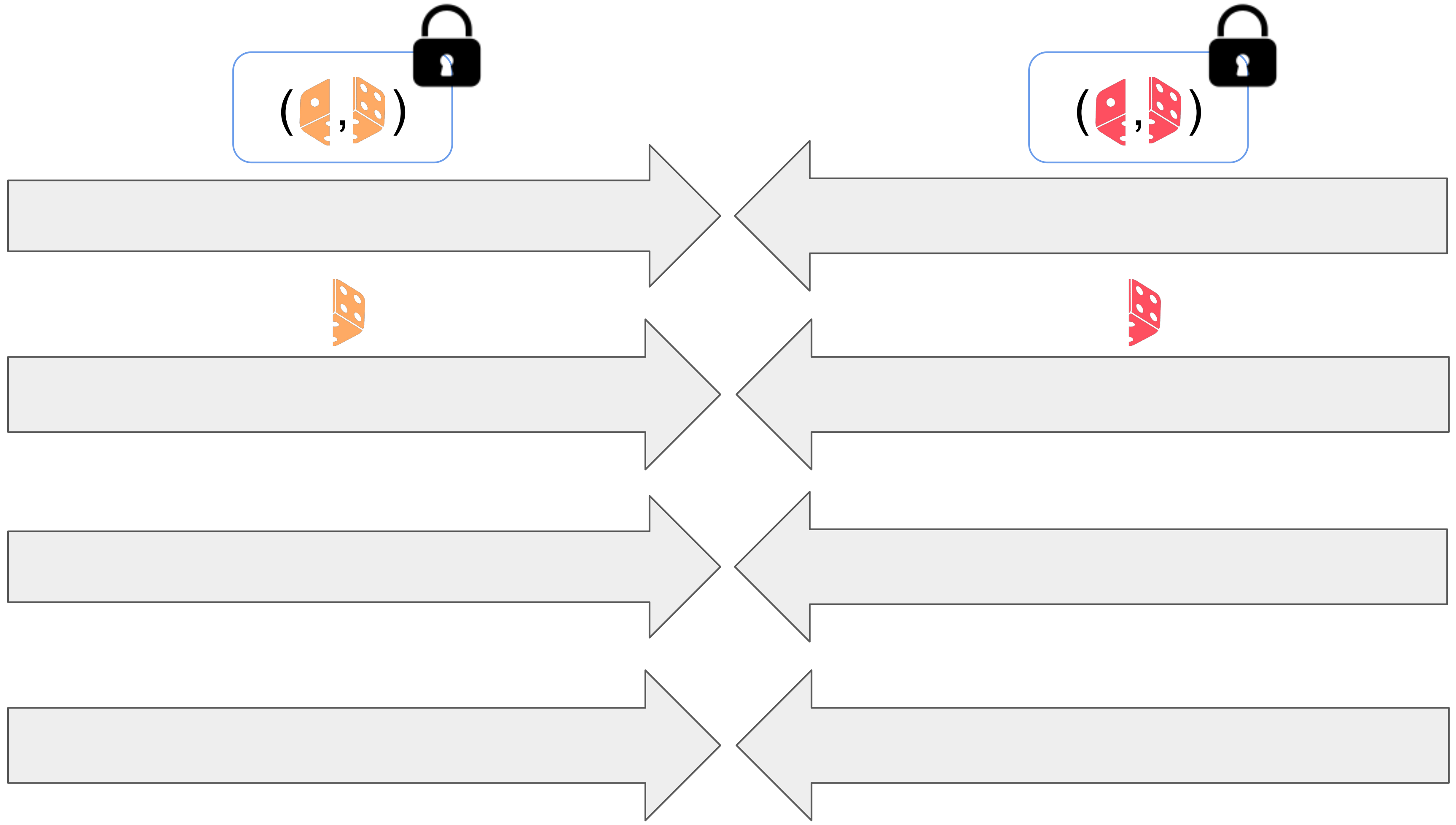


3.  $_i$ can be generated arbitrarily

   → Solve as 2. ✗ → Adds an additional round

   → Do Coin Flipping outside of protocol

Final Approach

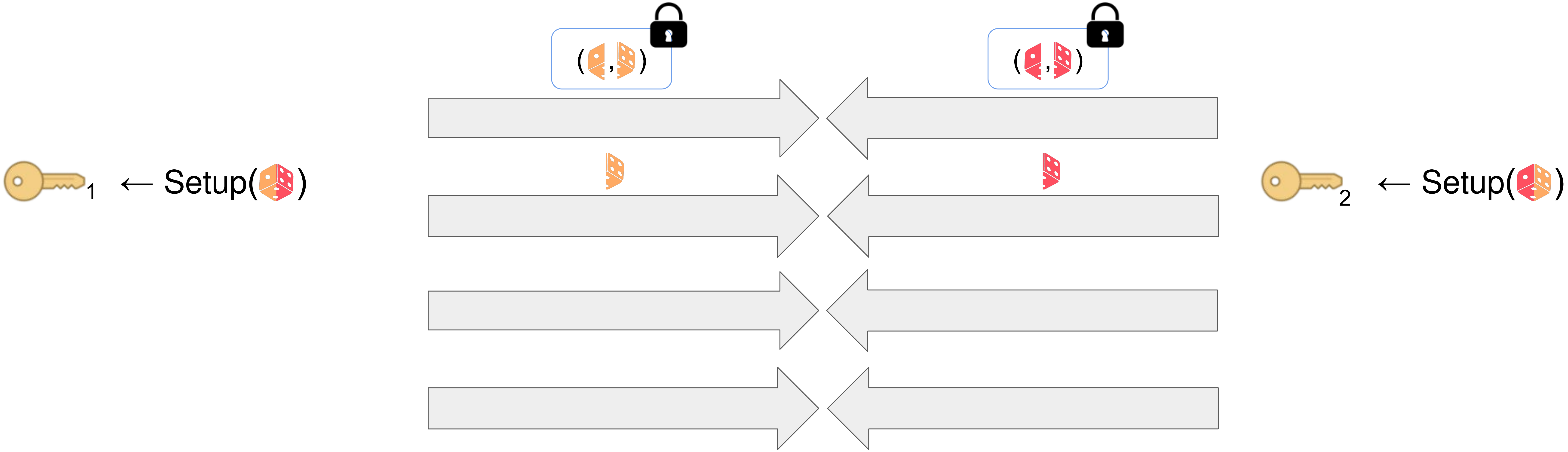


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach



3. $_i$ can be generated arbitrarily

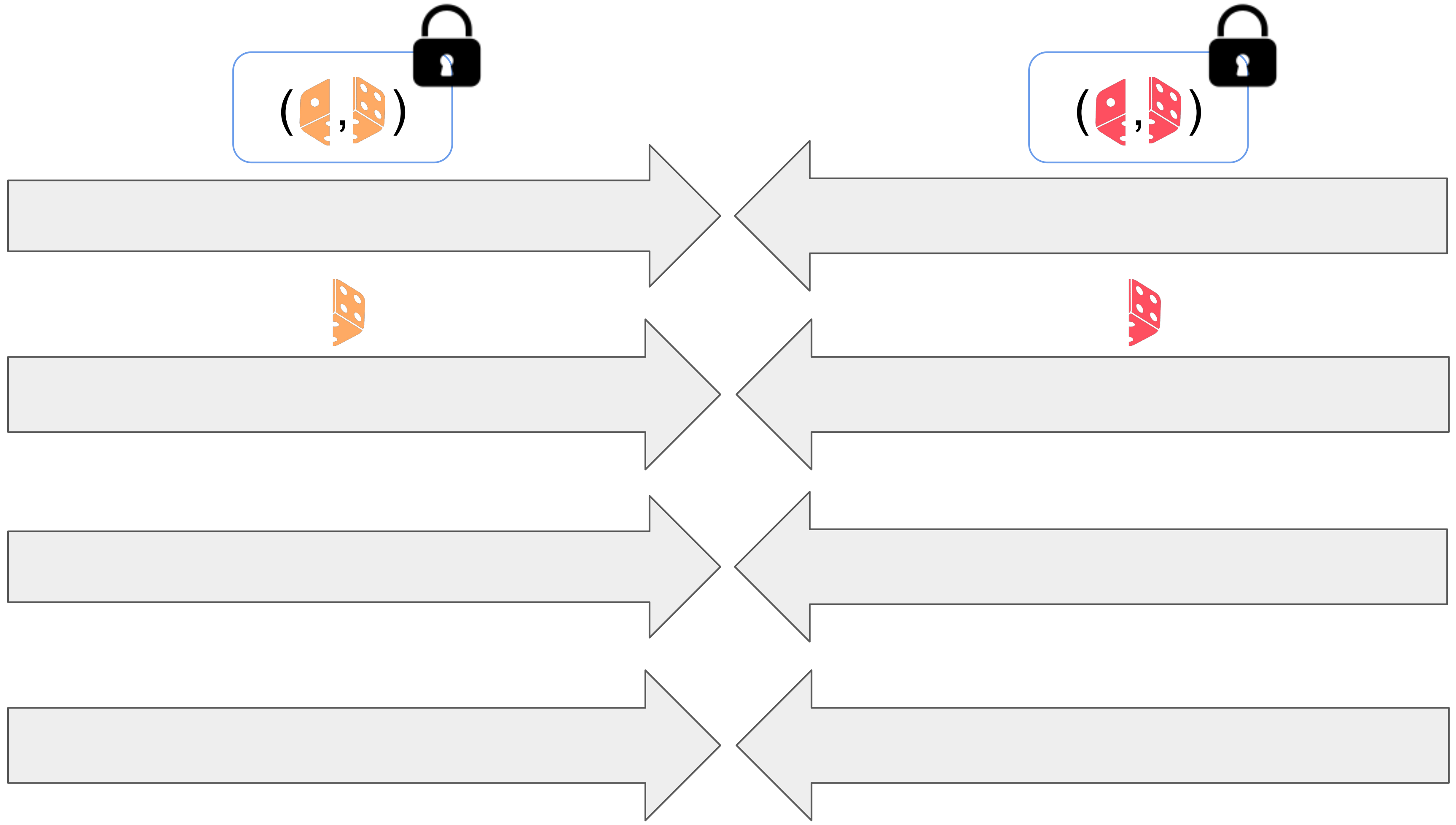
→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

$_1 \leftarrow$ Setup( )

$_{f,1} \leftarrow$ Keygen($_1$, f)



$_2 \leftarrow$ Setup( )

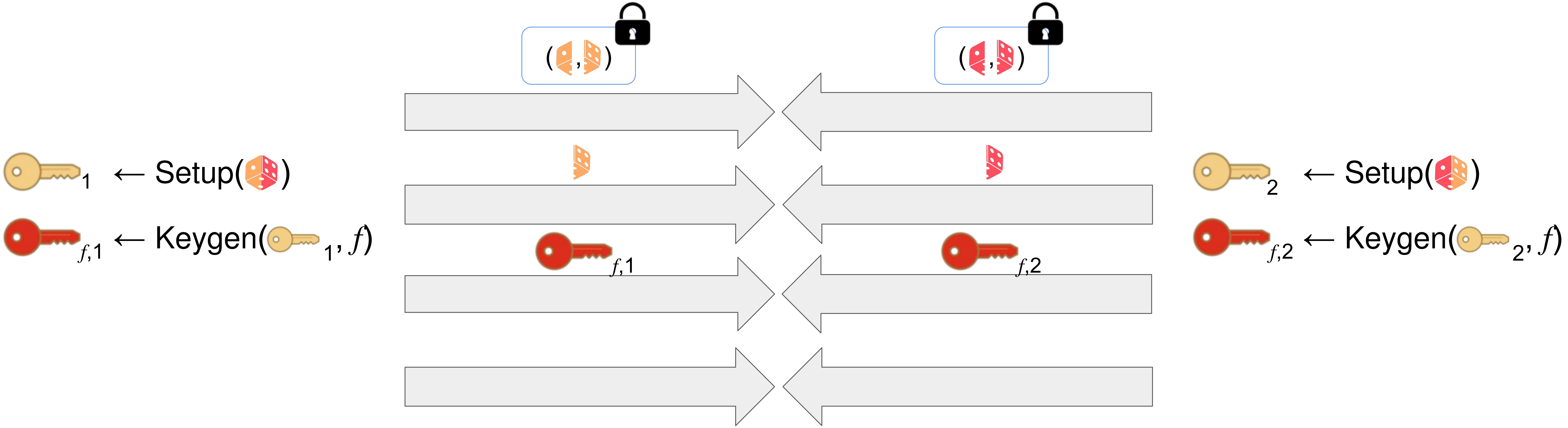
$_{f,2} \leftarrow$ Keygen($_2$, f)

3. $_i$ can be generated arbitrarily

$\rightarrow$ Solve as 2. ✗ $\rightarrow$ Adds an additional round

$\rightarrow$ Do Coin Flipping outside of protocol

Final Approach

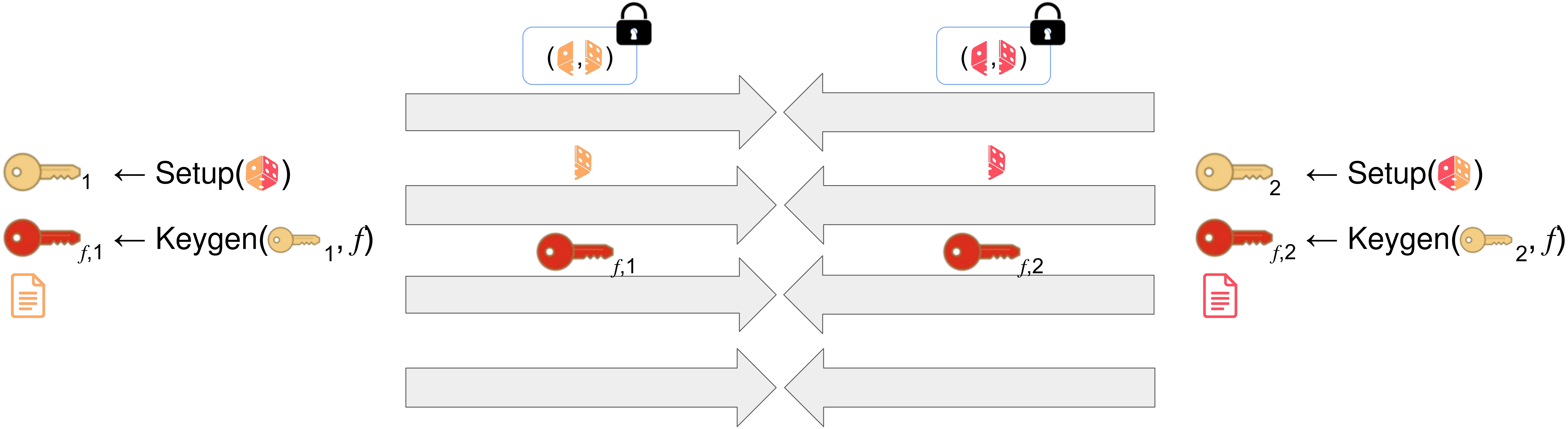


3.  $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

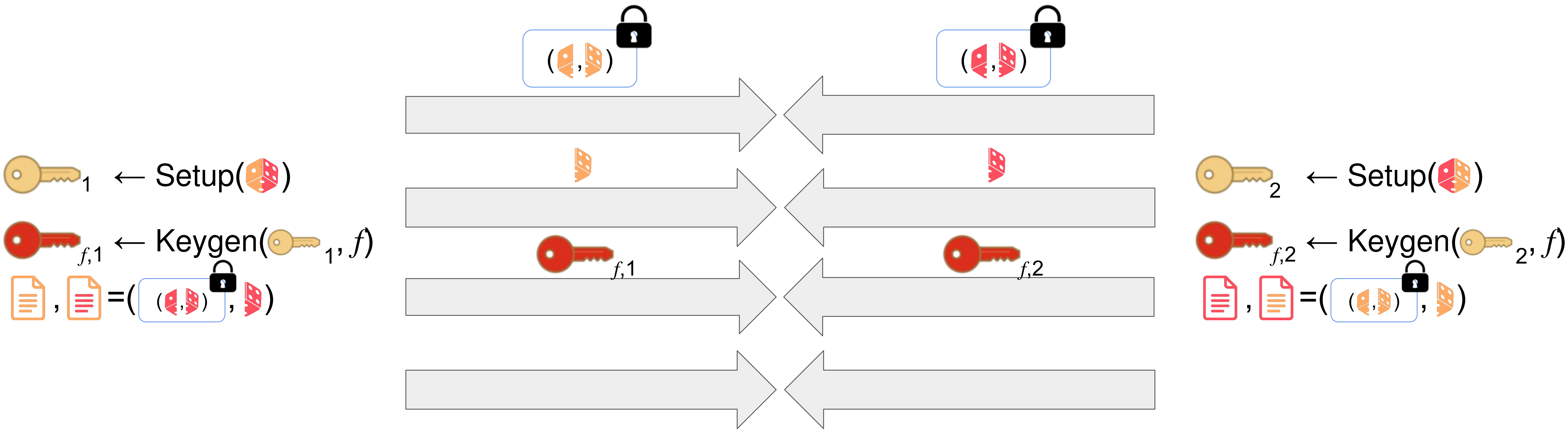


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

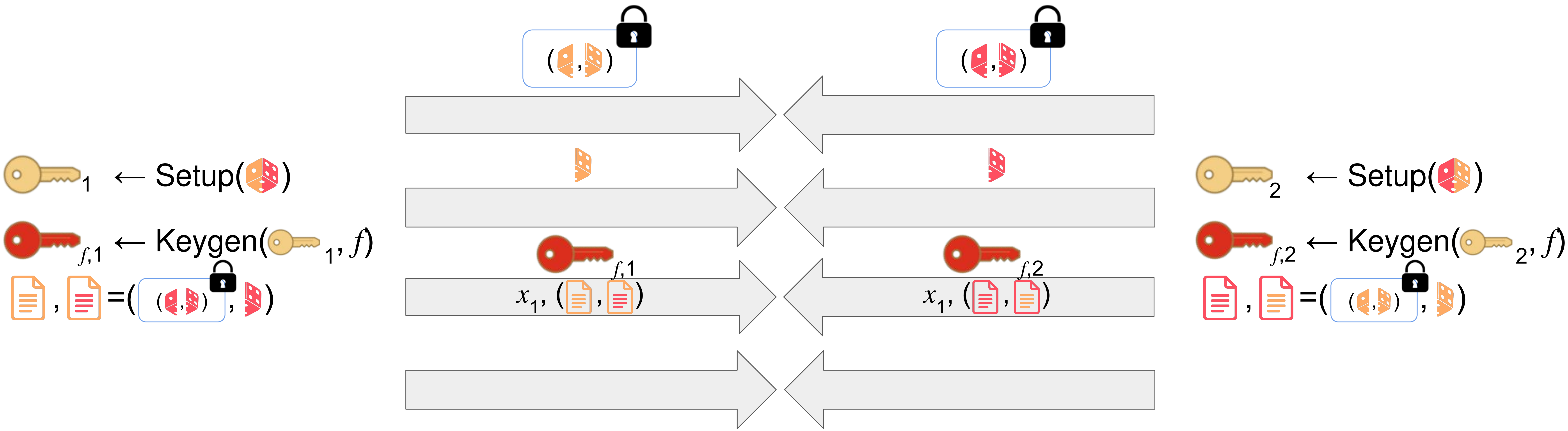


3.  $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

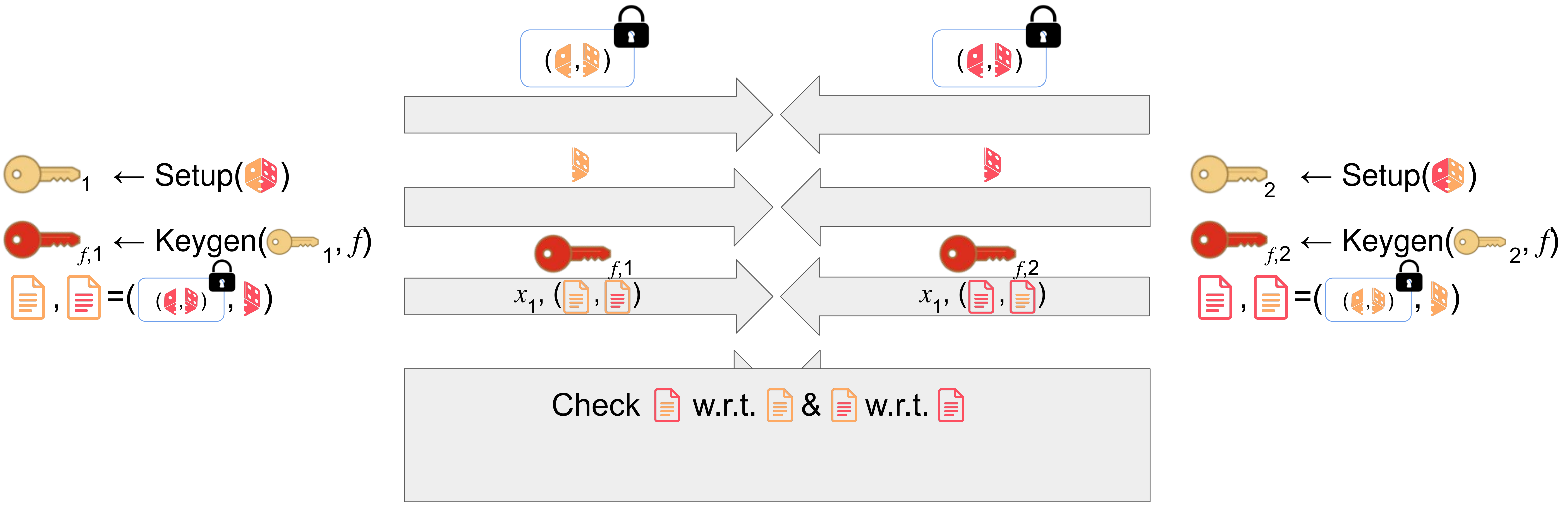


3. i can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

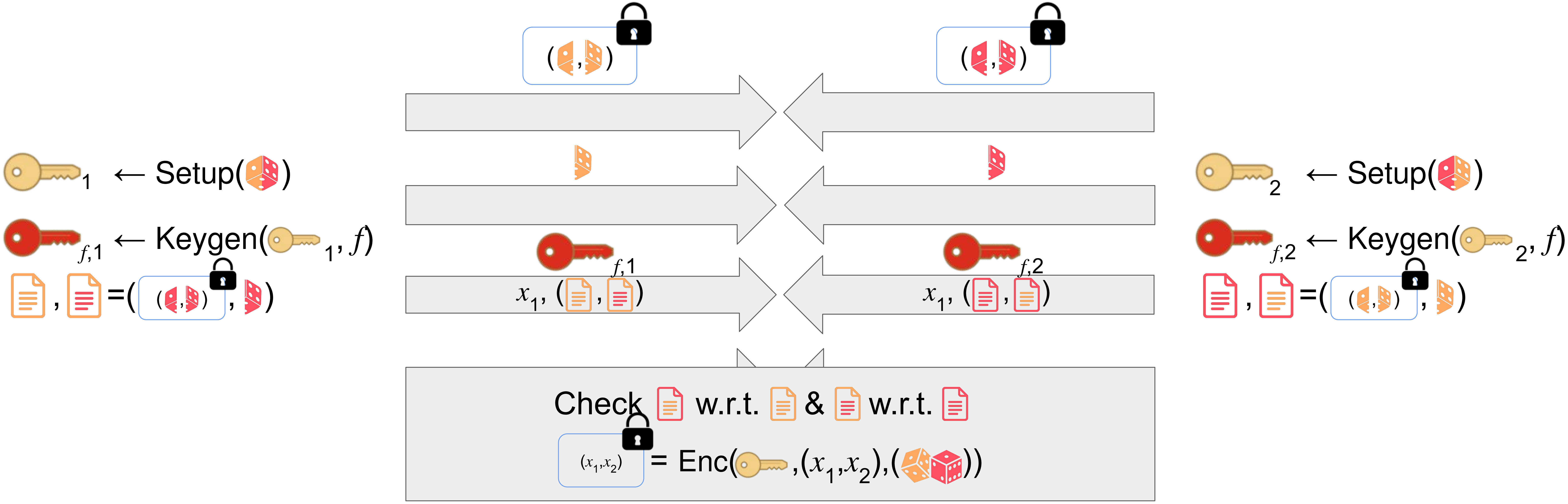


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

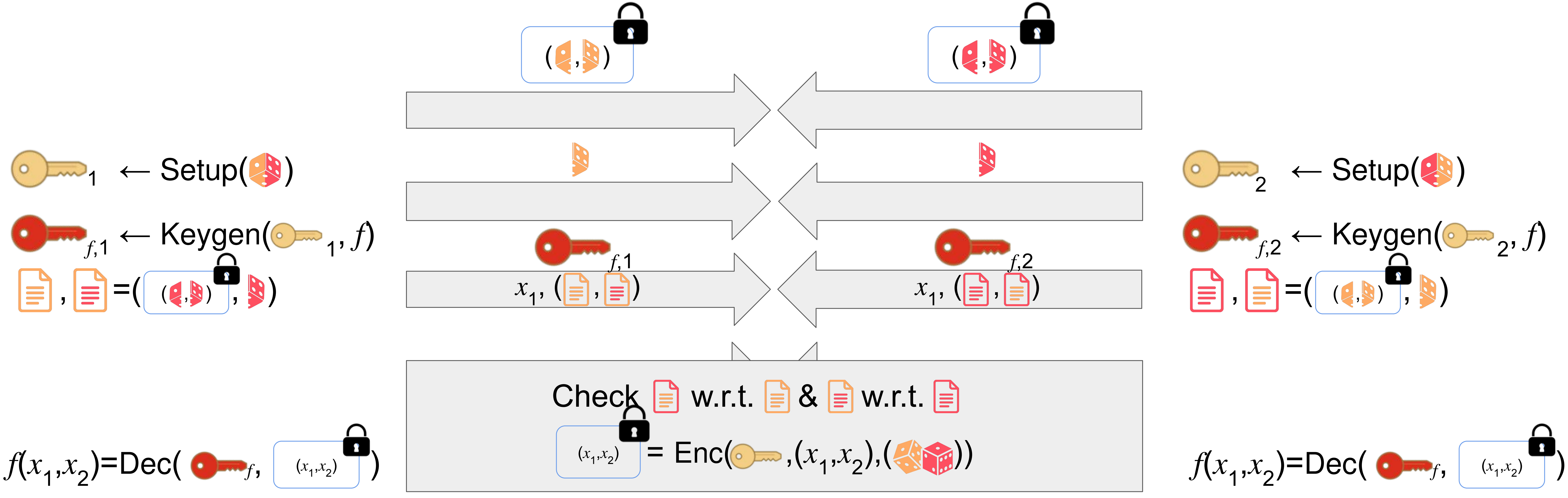


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

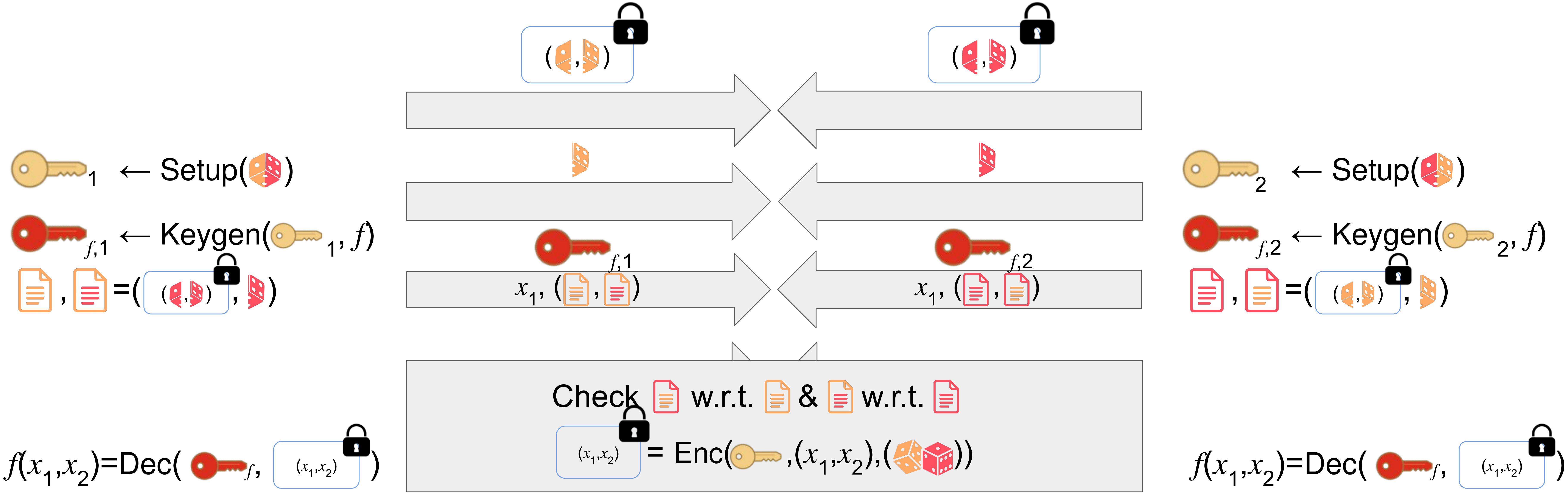


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol

Final Approach

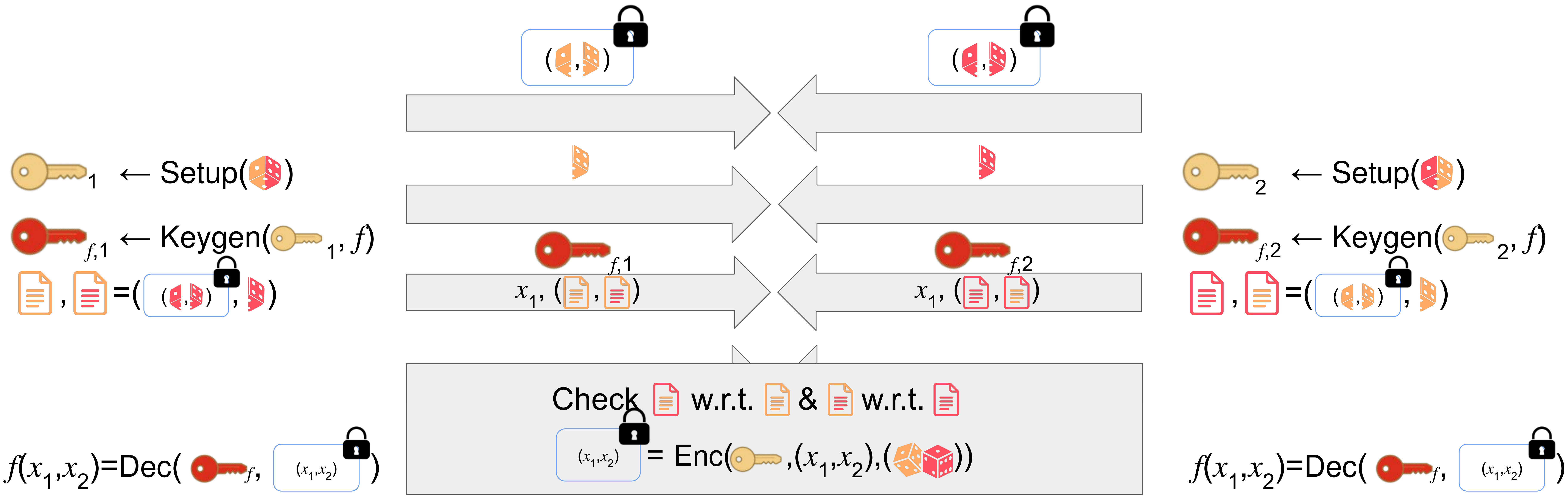


3. ${}_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol→ How to allow for honest behavior check?

Final Approach



3. key_i can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol→ How to allow for honest behavior check?

⇒ k-Delayed-Input Function MPC

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input MPC:

1. The input is needed in round k

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input MPC:

1. The input is needed in round k

2. but needs to be fixed before the protocol execution

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input MPC:

1. The input is needed in round k
2. but needs to be fixed before the protocol execution

k-Delayed-Input Function MPC:

1. The input is needed in round k

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input MPC:

1. The input is needed in round k
2. but needs to be fixed before the protocol execution

k-Delayed-Input Function MPC:

1. The input is needed in round k
2. and is partially decided during the protocol execution

k-Delayed-Input vs. k-Delayed-Input Function MPC

k-Delayed-Input MPC:

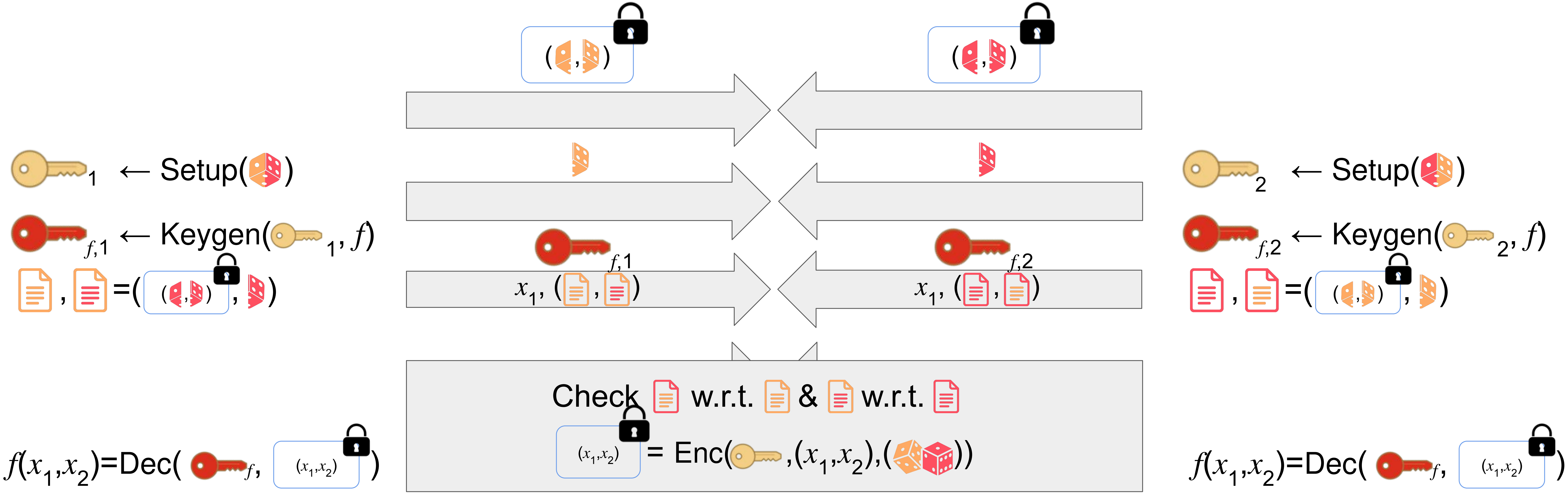
1. The input is needed in round k
2. but needs to be fixed before the protocol execution

k-Delayed-Input Function MPC:

1. The input is needed in round k
2. and is partially decided during the protocol execution

$\Rightarrow$ 2n-Party k-delayed-input MPC protocol + information-theoretic MAC

Final Approach

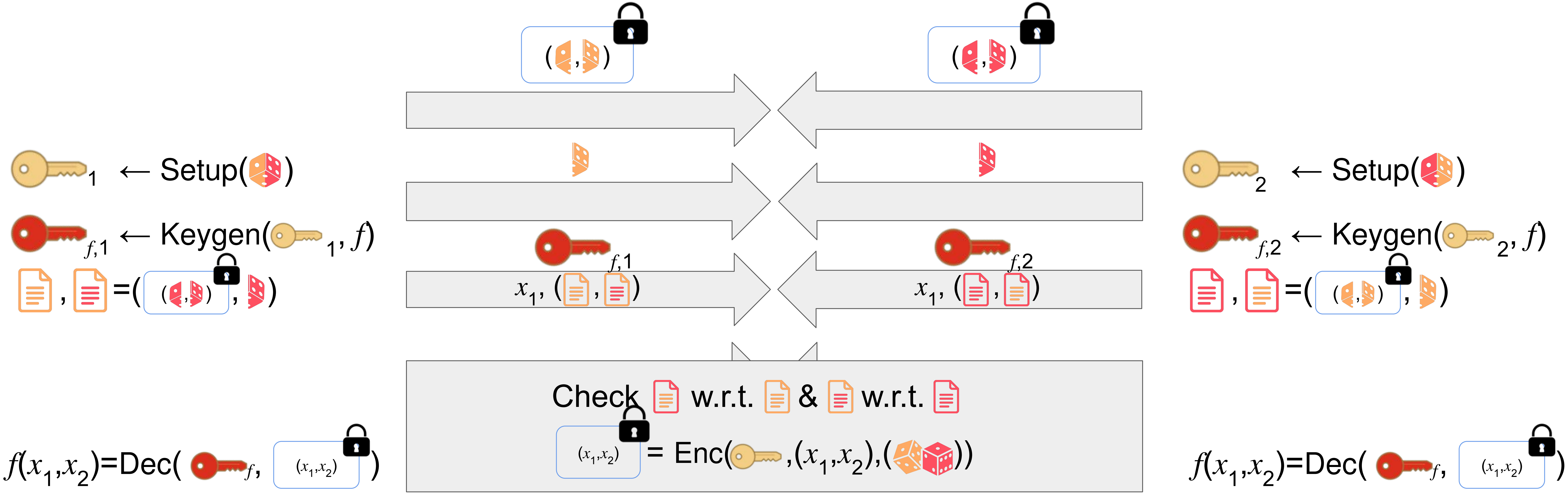


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

→ Do Coin Flipping outside of protocol→ How to allow for honest behavior check?

Final Approach

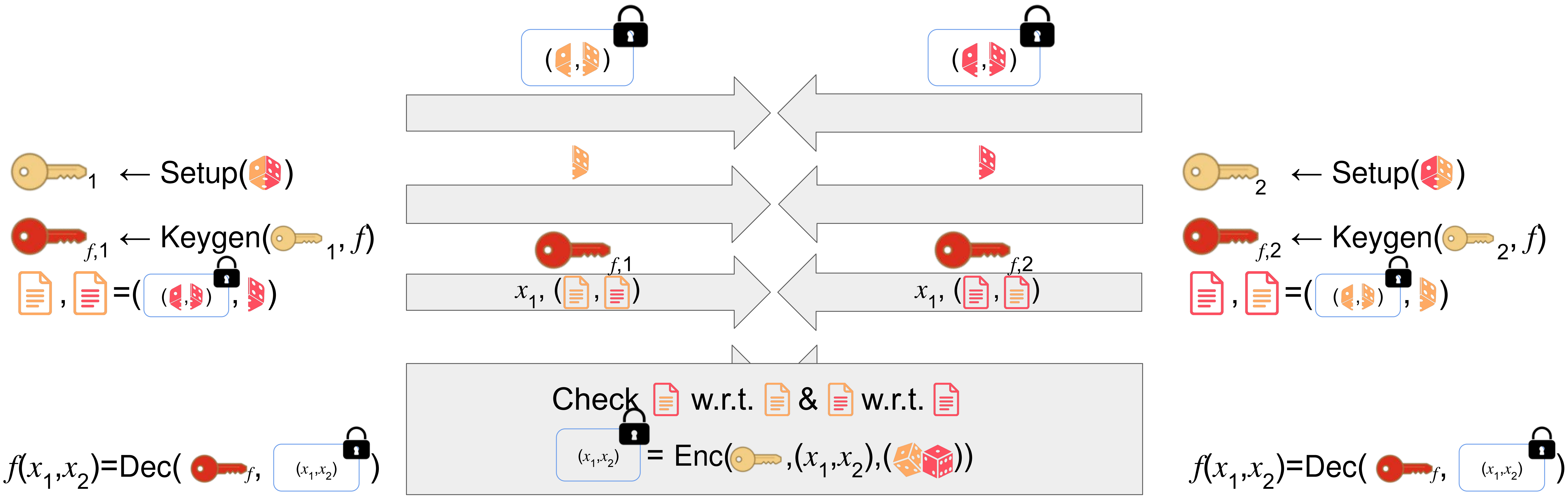


3. $_i$ can be generated arbitrarily

→ Solve as 2. ✗ → Adds an additional round

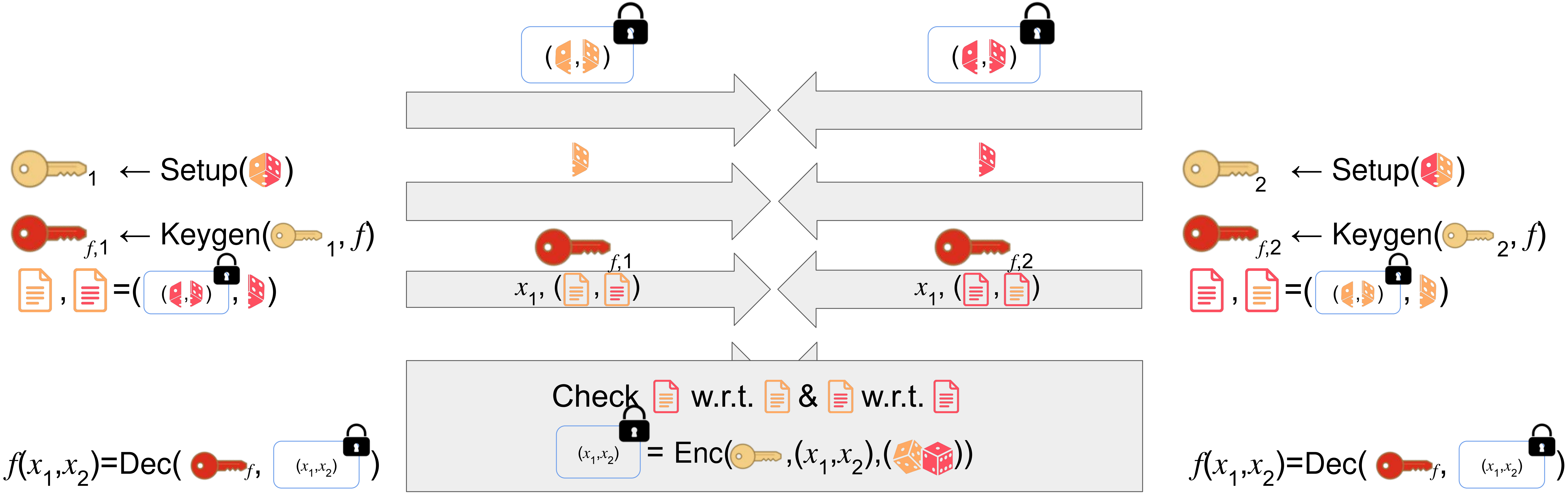
→ Do Coin Flipping outside of protocol→ How to allow for honest behavior check? ✓

Final Approach



1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be “bad” ✓
3. $_i$ can be generated arbitrarily ✓

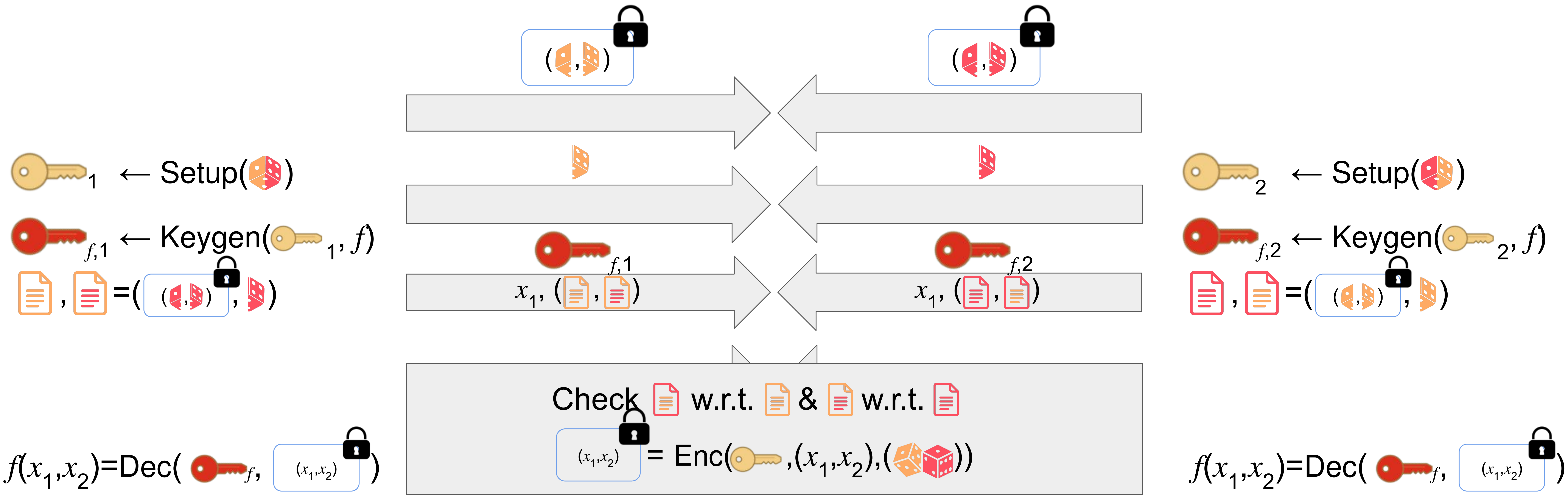
Final Approach



1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be "bad" ✓
3. $_i$ can be generated arbitrarily ✓

⇒ Communication Complexity: depth(f)

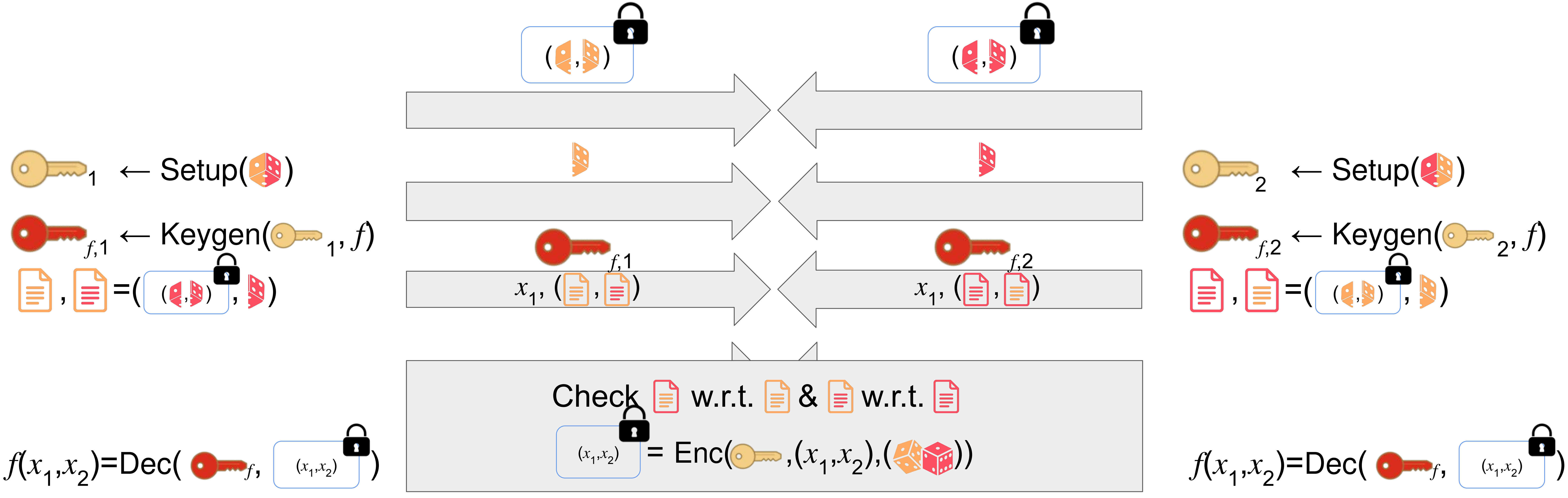
Final Approach



1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be "bad" ✓
3. $_i$ can be generated arbitrarily ✓

⇒ Communication Complexity: depth(f) Can we do better?

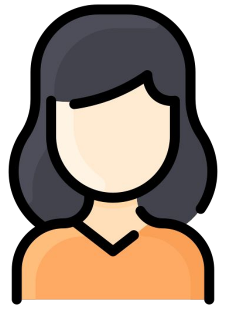
Final Approach

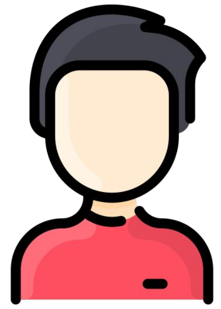


1. $_{f,i}$ can be generated maliciously ✓
2. (,) used for encryption can be “bad” ✓
3. $_i$ can be generated arbitrarily ✓

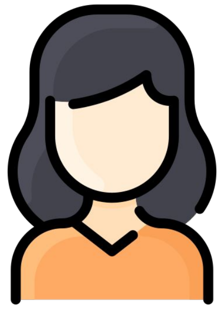
⇒ Communication Complexity: depth(f) Can we do better? Yes!

Multi-Key Fully Homomorphic Encryption [LTV12]

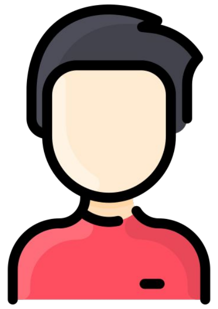




Multi-Key Fully Homomorphic Encryption [LTV12]

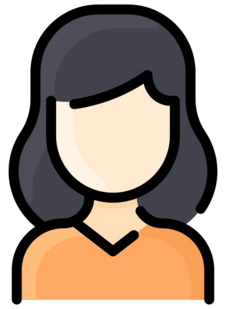


($_1$, $_1$) ← Setup



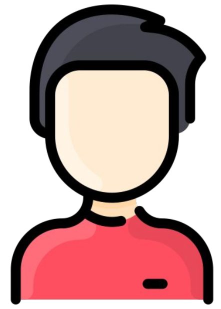
($_2$, $_2$) ← Setup

Multi-Key Fully Homomorphic Encryption [LTV12]



$({}_1, {}_1) \leftarrow$ Setup

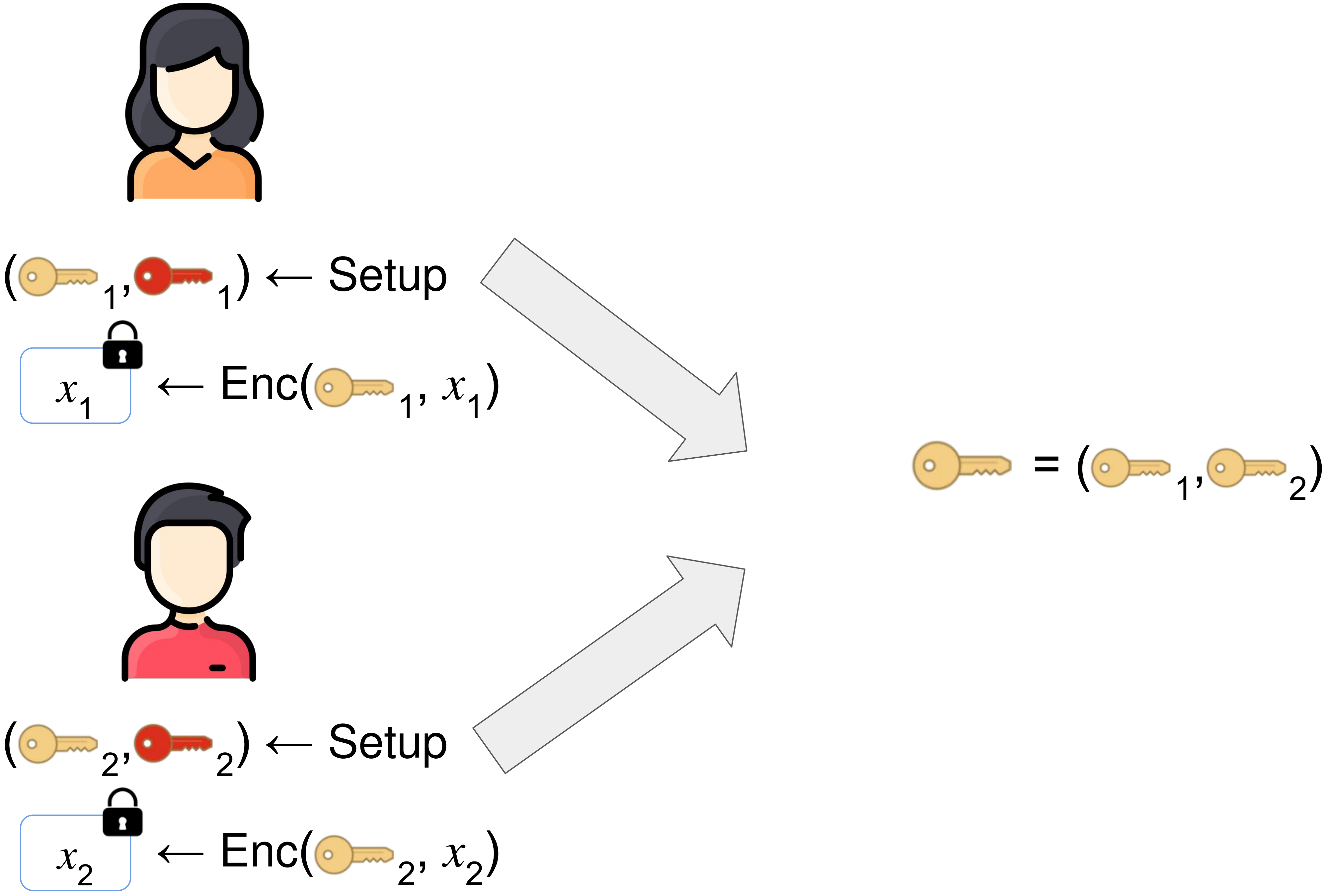
$x_1 \leftarrow \text{Enc}({}_1, x_1)$



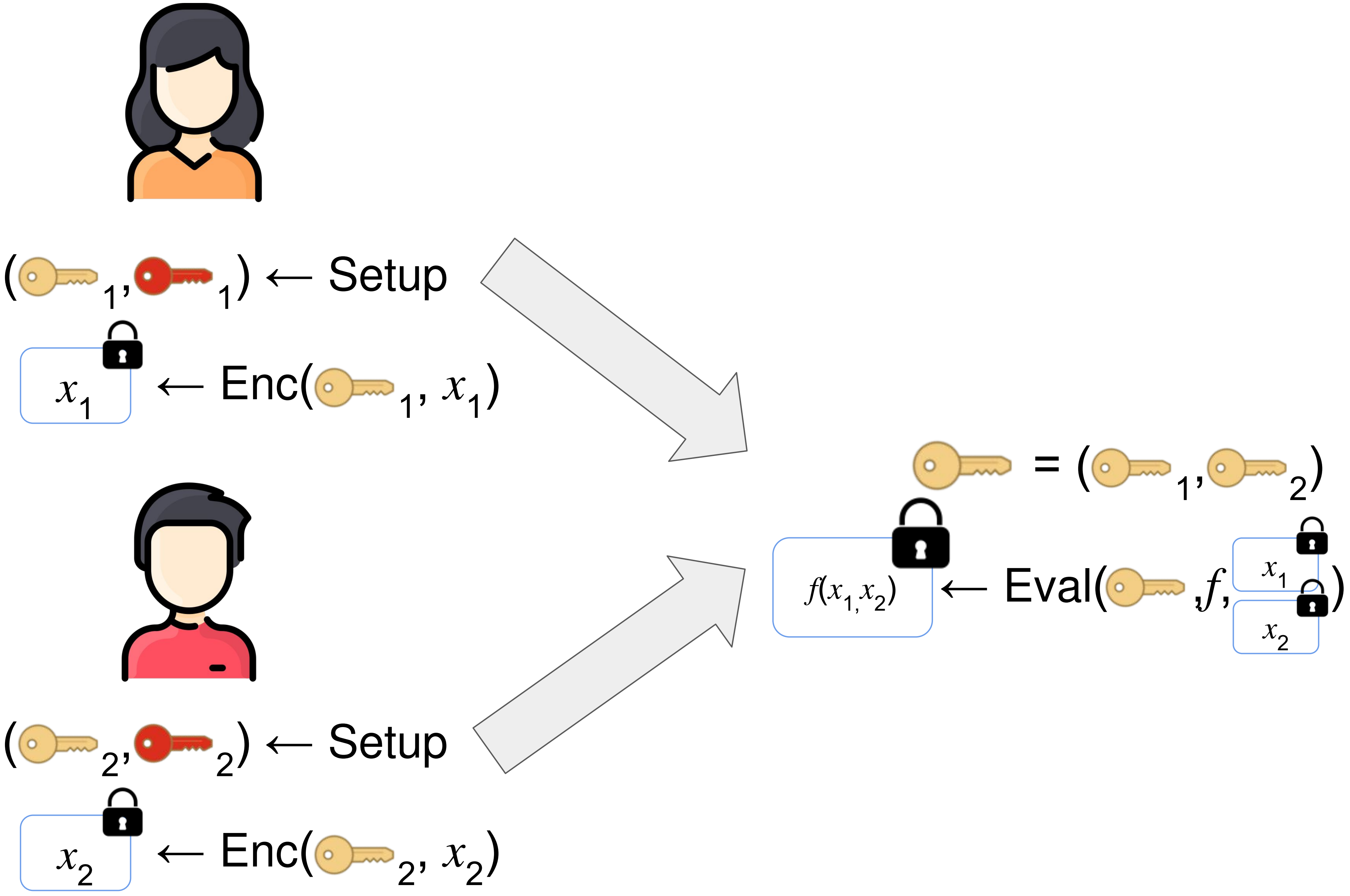
$({}_2, {}_2) \leftarrow$ Setup

$x_2 \leftarrow \text{Enc}({}_2, x_2)$

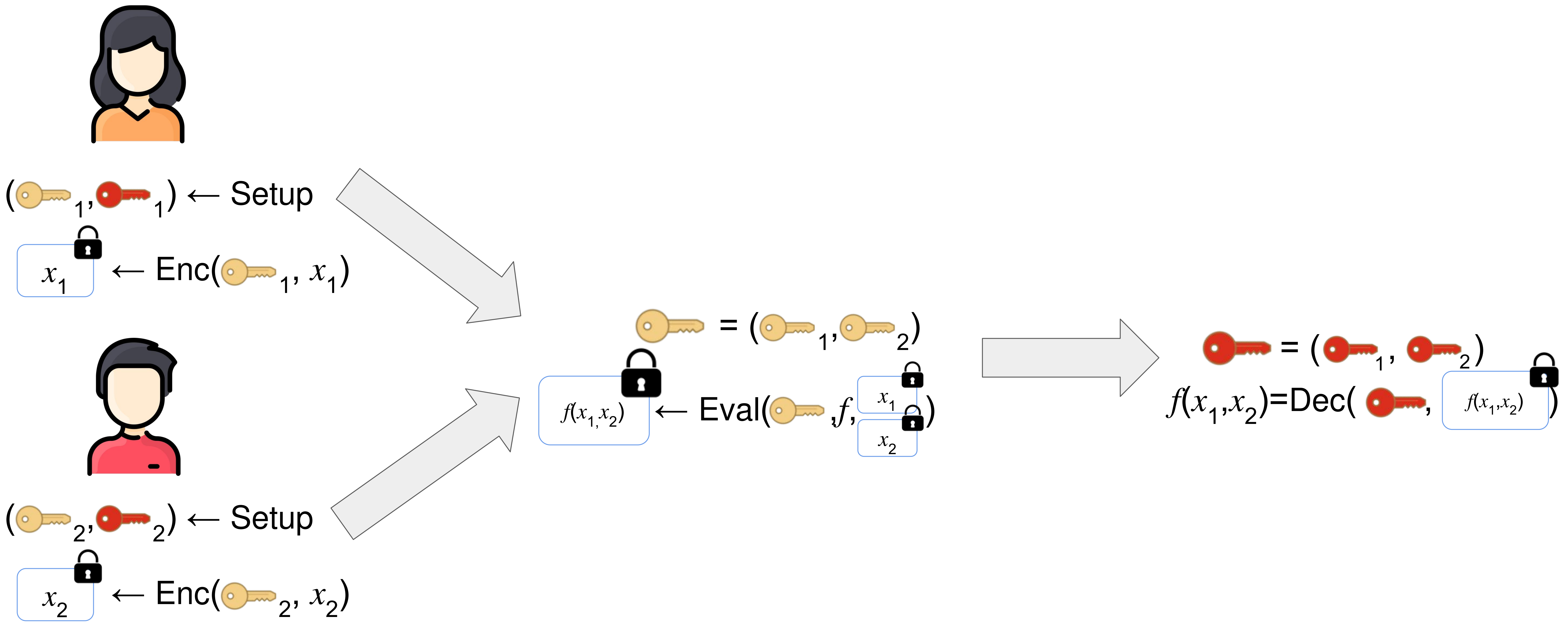
Multi-Key Fully Homomorphic Encryption [LTV12]



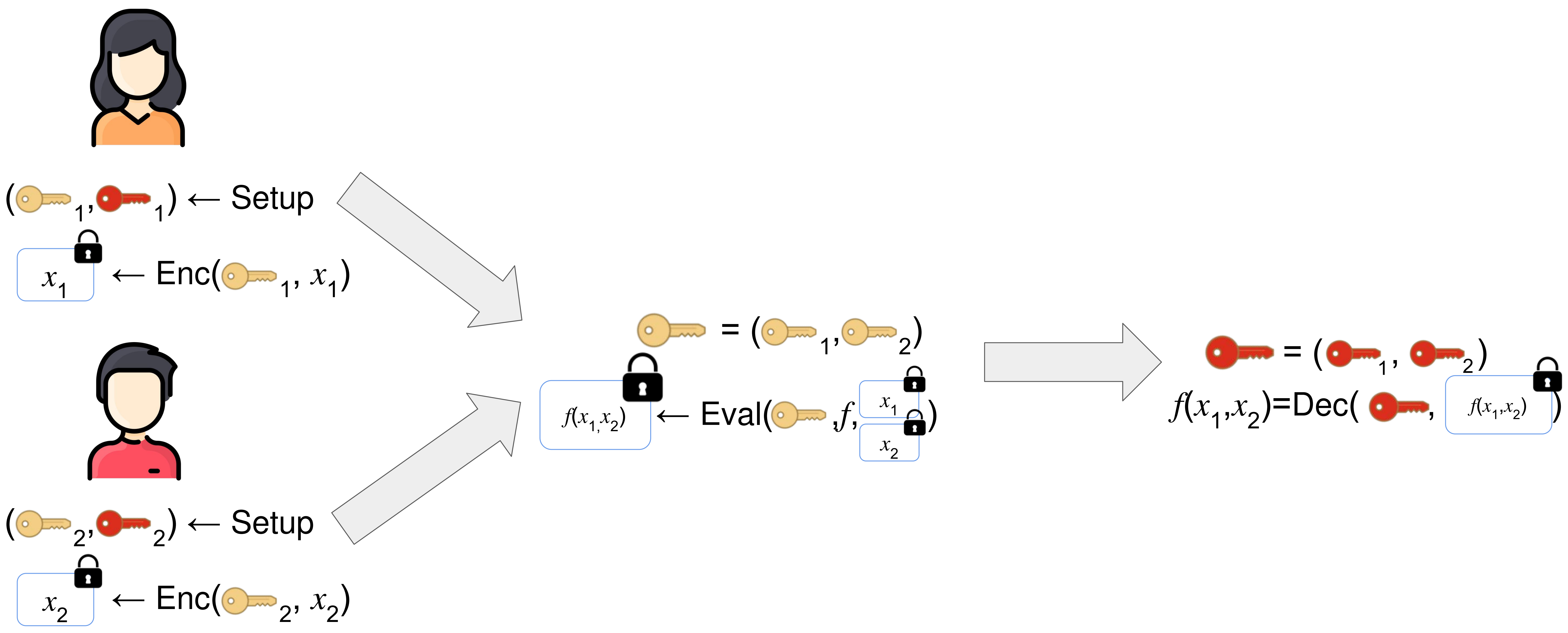
Multi-Key Fully Homomorphic Encryption [LTV12]



Multi-Key Fully Homomorphic Encryption [LTV12]

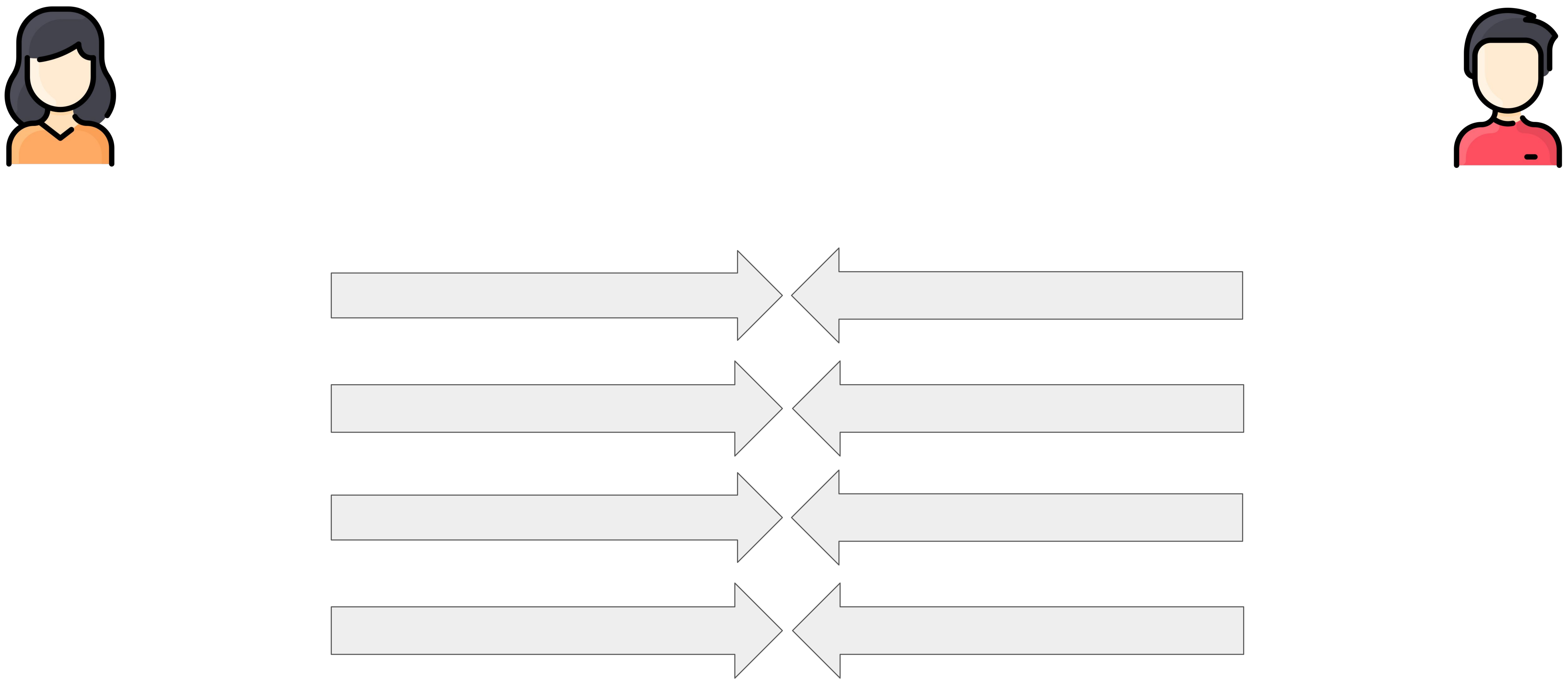


Multi-Key Fully Homomorphic Encryption [LTV12]

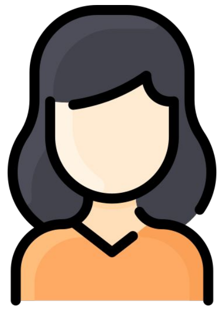


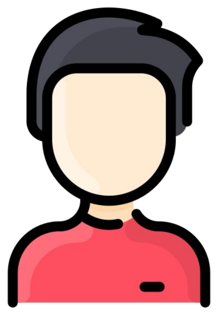
Compactness: | $f(x_1,x_2)$ | independent of f

Round-Optimal and Communication-Efficient MPC



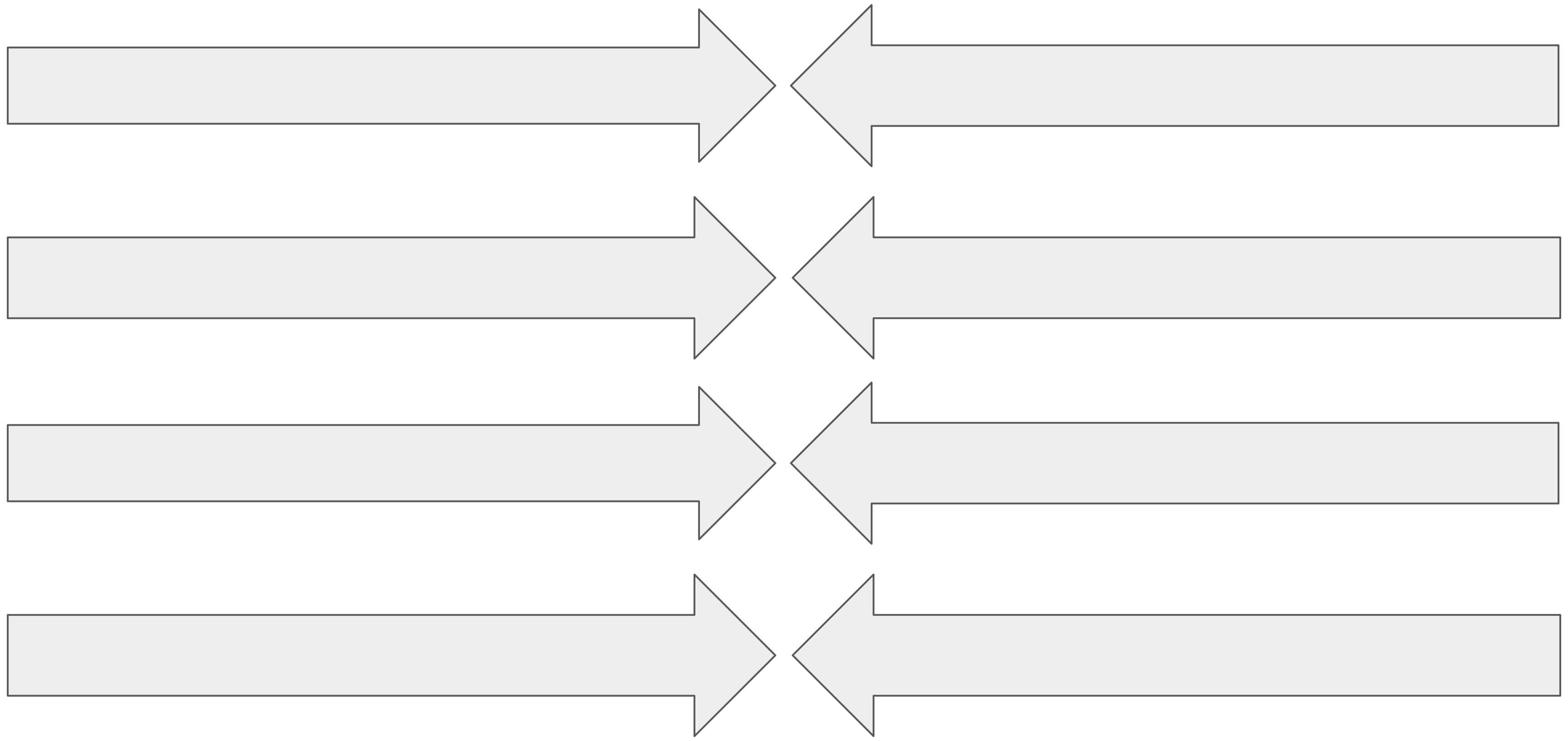
Round-Optimal and Communication-Efficient MPC



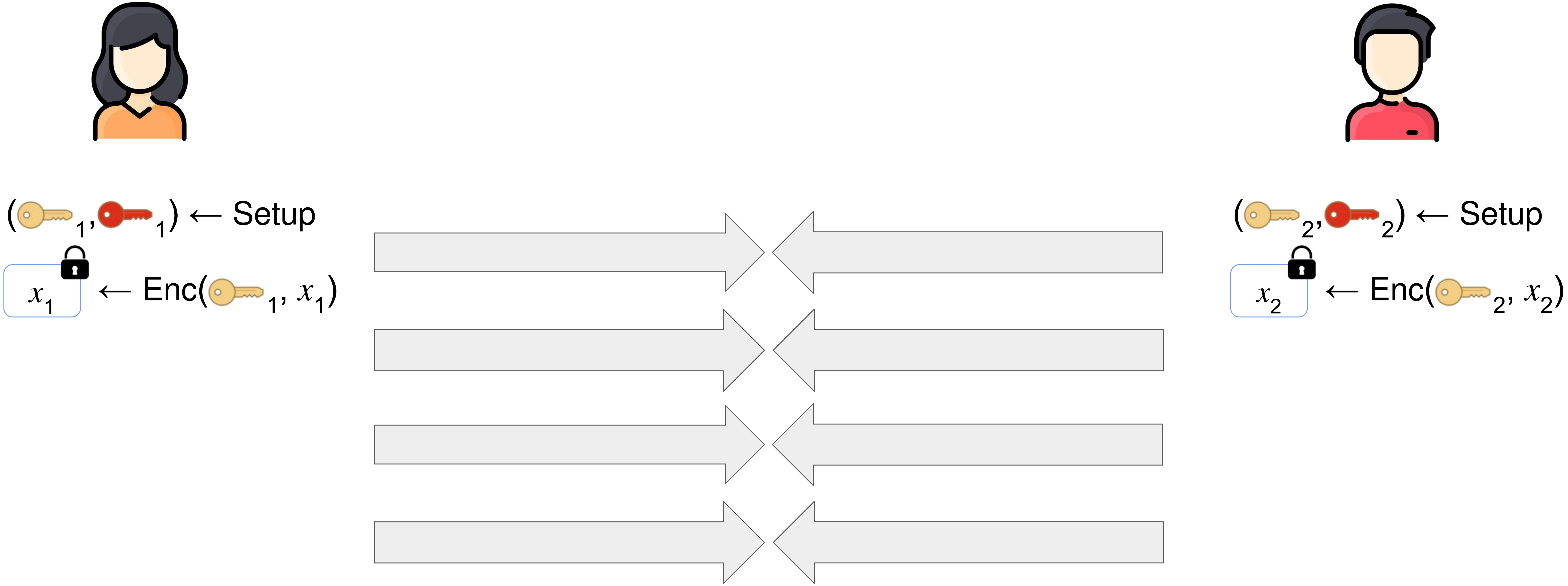


($_1$, $_1$) $\leftarrow$ Setup

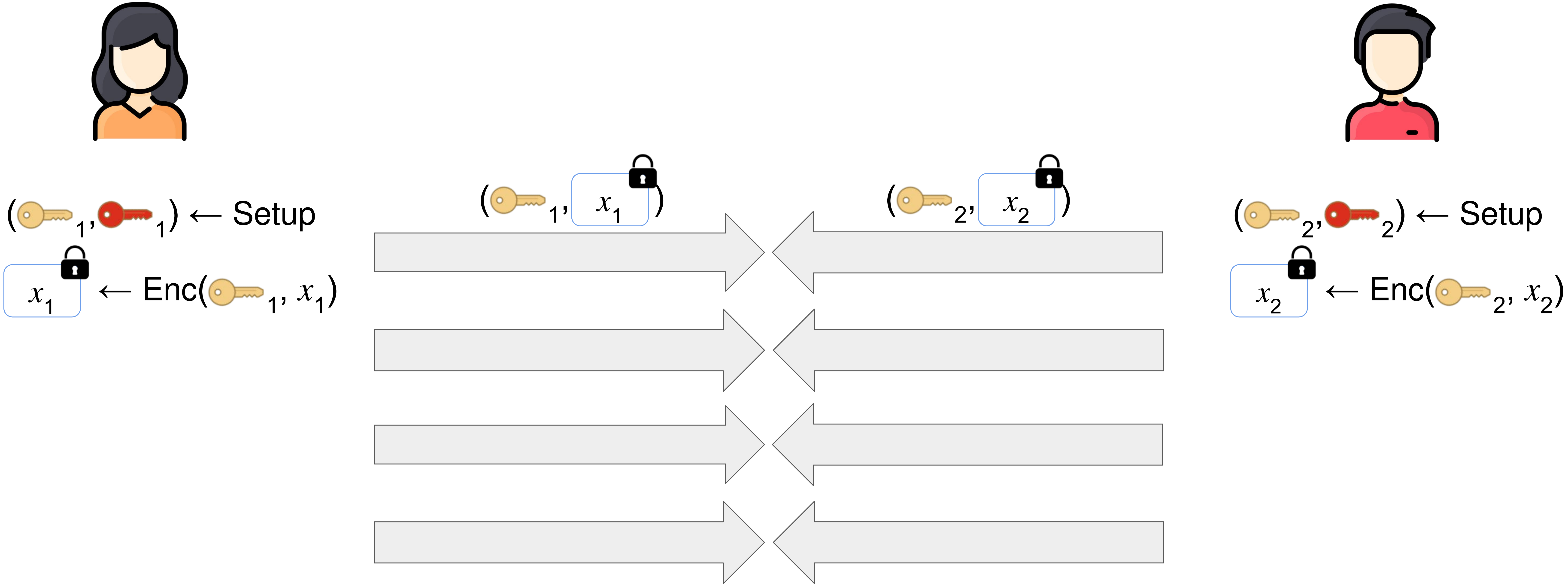
($_2$, $_2$) $\leftarrow$ Setup



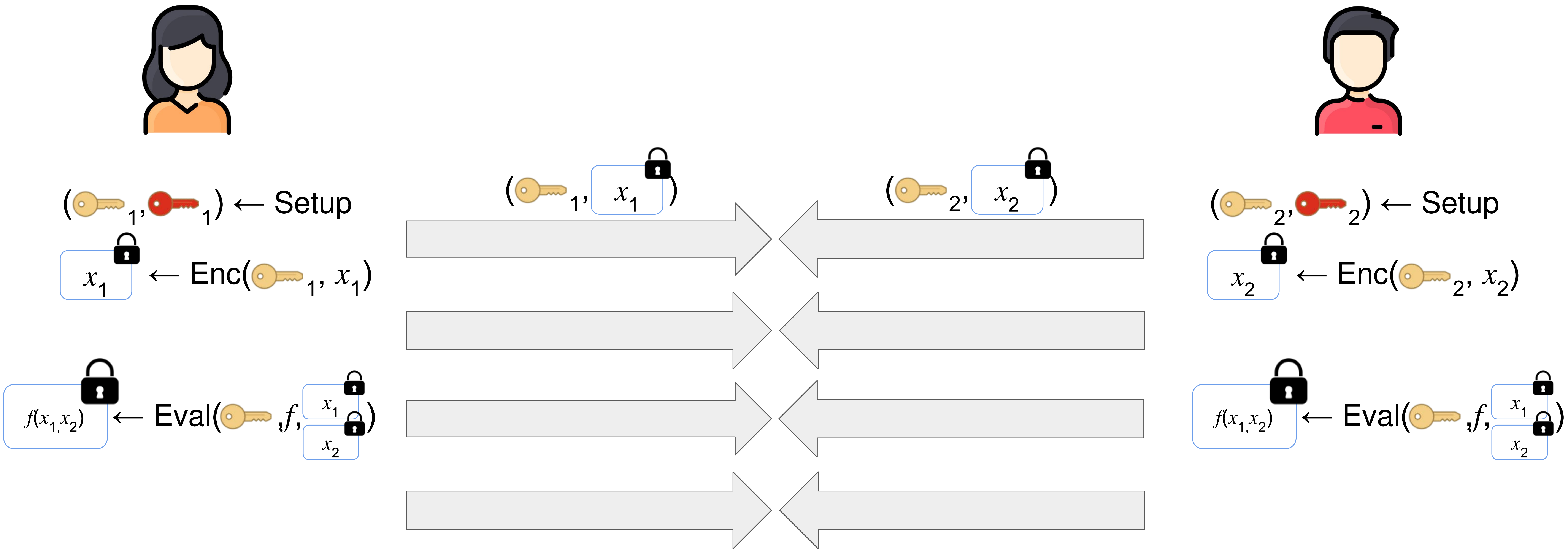
Round-Optimal and Communication-Efficient MPC



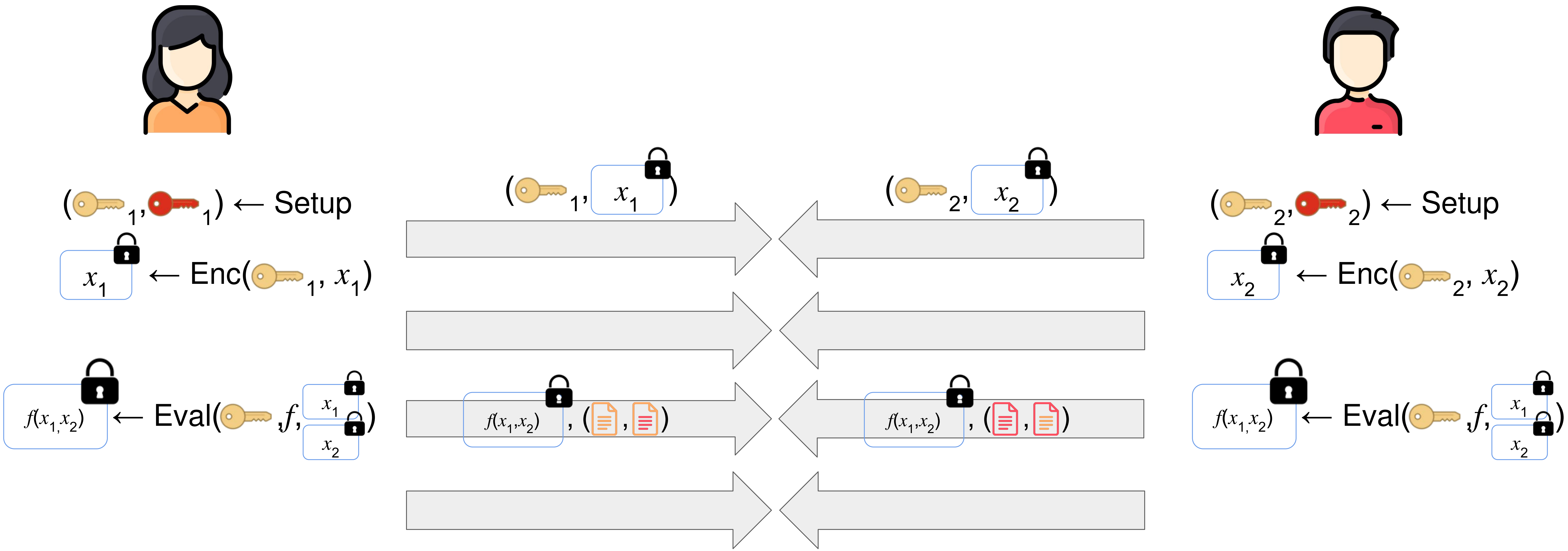
Round-Optimal and Communication-Efficient MPC



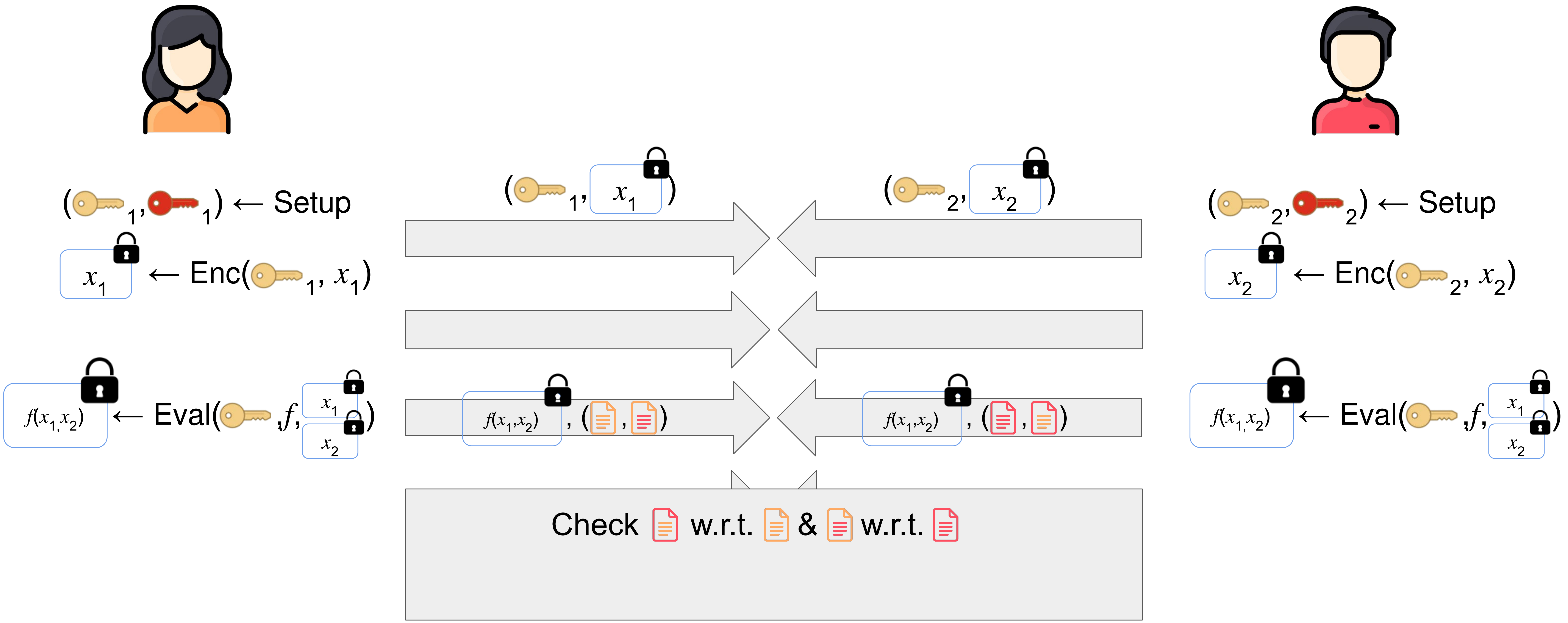
Round-Optimal and Communication-Efficient MPC



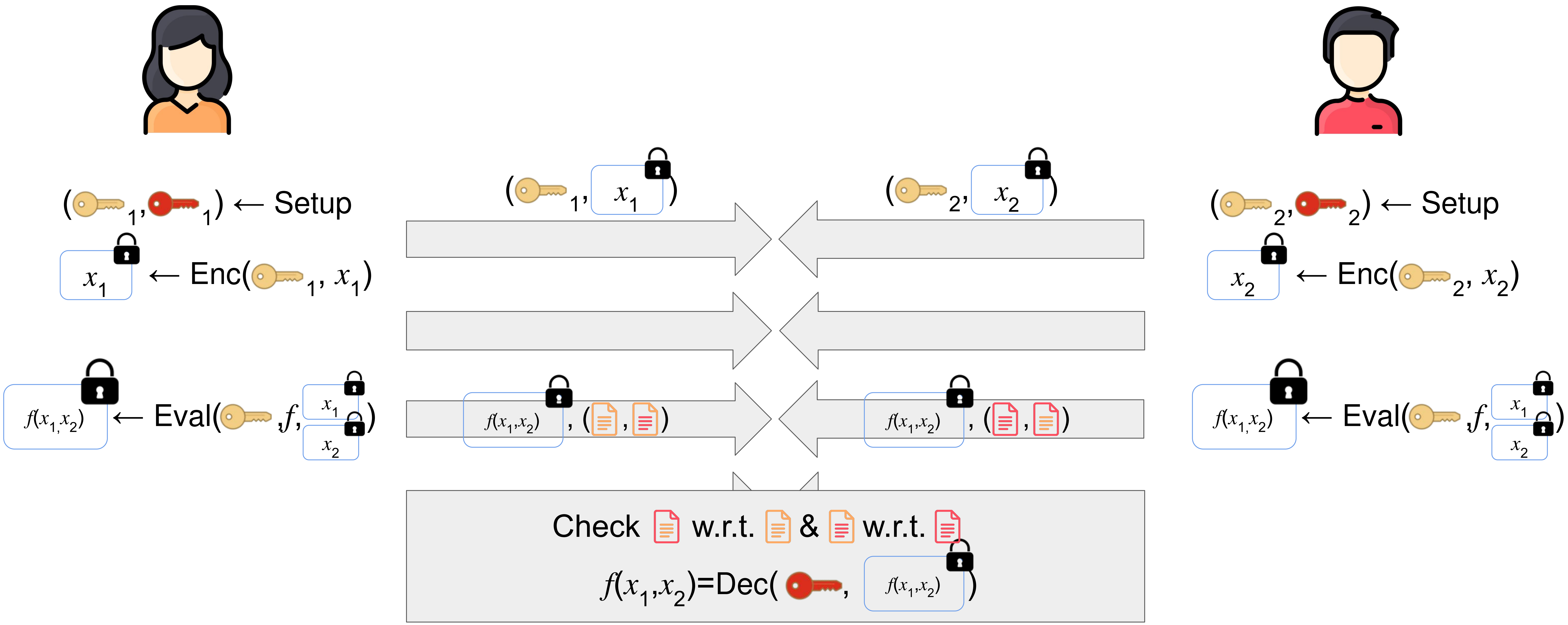
Round-Optimal and Communication-Efficient MPC



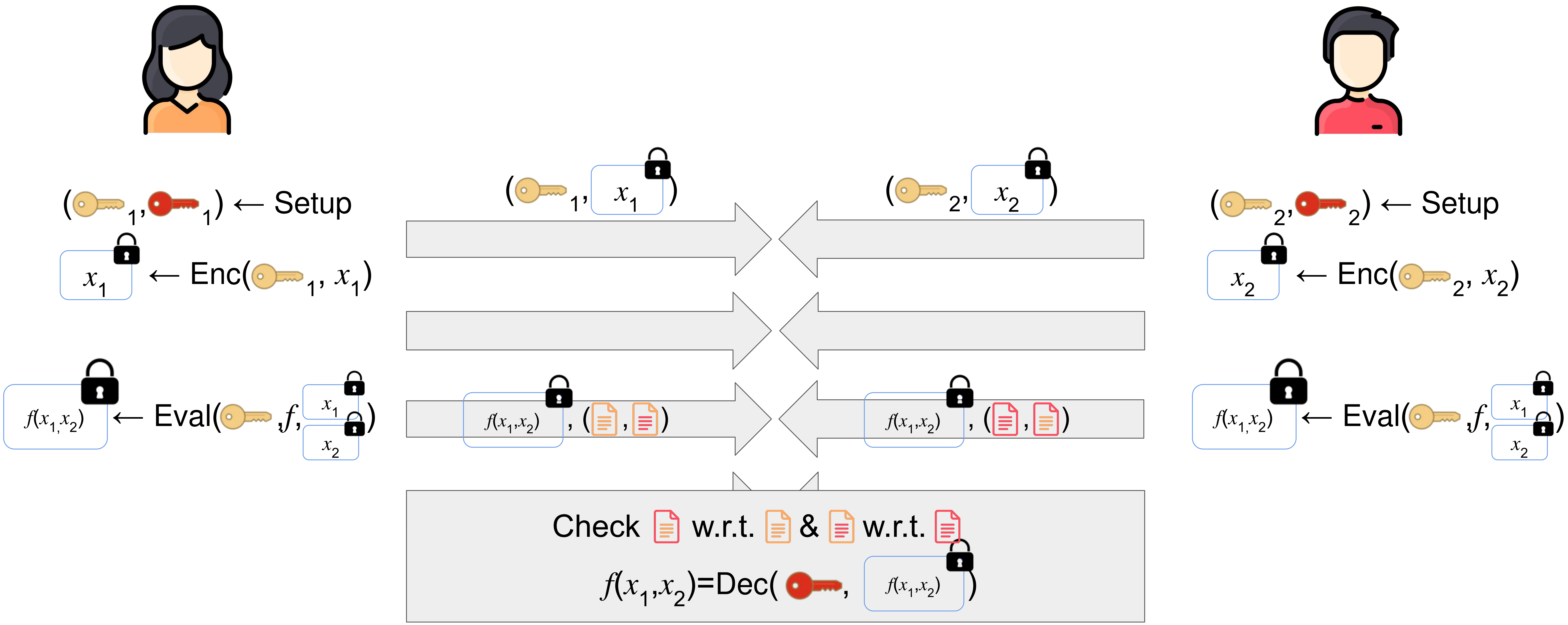
Round-Optimal and Communication-Efficient MPC



Round-Optimal and Communication-Efficient MPC



Round-Optimal and Communication-Efficient MPC



⇒ Communication Complexity: L_{in} & L_{out} independent of f

Conclusion

Conclusion

- Round-Optimal and Communication-Efficient Multiparty Computation

Conclusion

- Round-Optimal and Communication-Efficient Multiparty Computation
    - Protocol with Communication Complexity depth(f) based on Functional Encryption Combiners

Conclusion

- Round-Optimal and Communication-Efficient Multiparty Computation
  - Protocol with Communication Complexity depth(f) based on Functional Encryption Combiners
  - Protocol with Communication Complexity L_{in} & L_{out} based on Multi-Key Fully Homomorphic Encryption

Conclusion

- Round-Optimal and Communication-Efficient Multiparty Computation
  - Protocol with Communication Complexity depth(f) based on Functional Encryption Combiners
  - Protocol with Communication Complexity L_{in} & L_{out} based on Multi-Key Fully Homomorphic Encryption
- k-Delayed-Input Function MPC

Conclusion

- Round-Optimal and Communication-Efficient Multiparty Computation
  - Protocol with Communication Complexity depth(f) based on Functional Encryption Combiners
  - Protocol with Communication Complexity L_{in} & L_{out} based on Multi-Key Fully Homomorphic Encryption
- k-Delayed-Input Function MPC

Thank You!