New Tools for On-the-Fly Secure Computation

Sacha Servan-Schreiber

Thesis Defense

Advisor: Srini Devadas

Committee: Yael Tauman Kalai (MIT), Geoffroy Couteau (IRIF)

This thesis: A toolbox for secure computation

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**'24], [CDDK**SS**'24]

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

- **Background** on secure computation

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**'24], [CDDK**SS**'24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

- **Background** on secure computation
- **On-the-fly** secure computation

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

- **Background** on secure computation
- **On-the-fly** secure computation
- **Current** landscape

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

- **Background** on secure computation
- **On-the-fly** secure computation
- **Current** landscape
- **New results**

This thesis: A toolbox for secure computation

Part I: New practical tools and applications [**SS**’24], [CDDK**SS**’24]

Part II: New theoretical tools and applications [CDHJ**SS**'25], [BD**SS**'25]

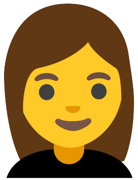
Part III: Expanding the frontier [BJ**SS**S'25]

Overview of this talk

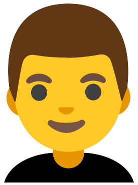
- **Background** on secure computation
- **On-the-fly** secure computation
- **Current** landscape
- **New results**
- **Conclusion**

Secure Computation

Secure Computation



Alice

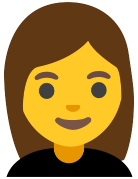


Bob

Secure Computation

Secure Computation

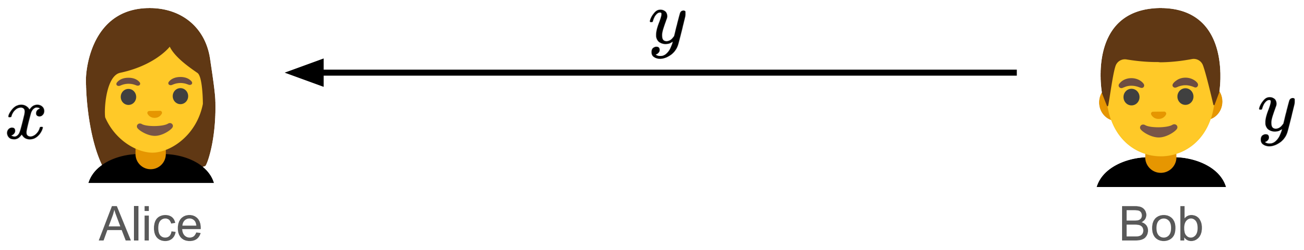
Private Inputs

x  Alice

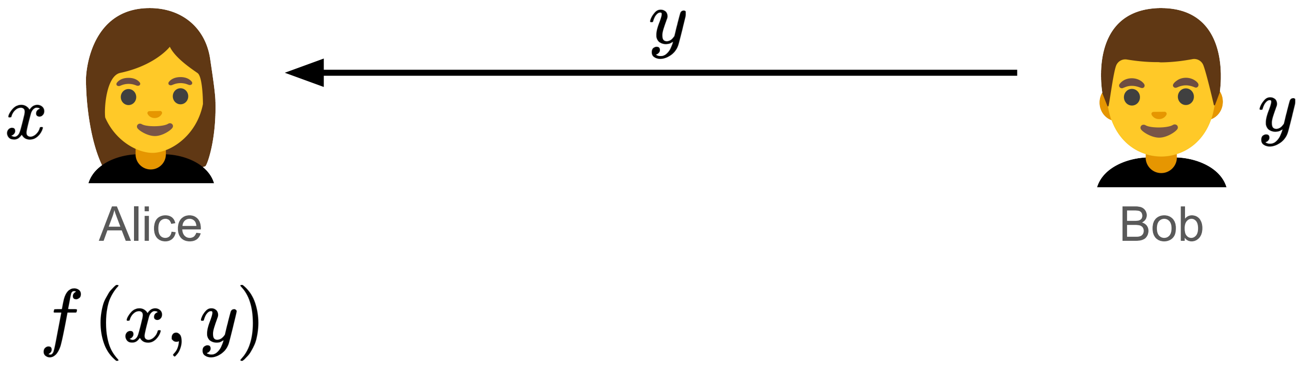
Want: $f(x, y)$

 Bob y

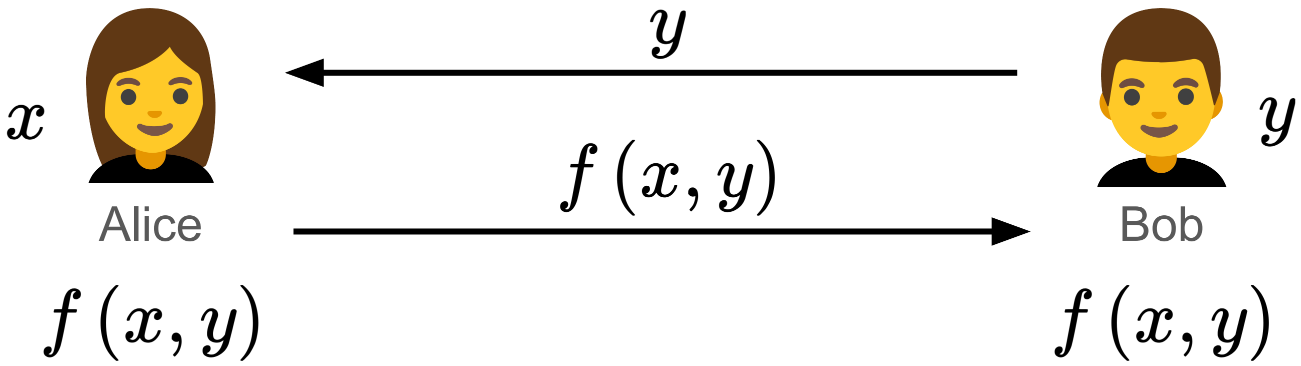
Secure Computation



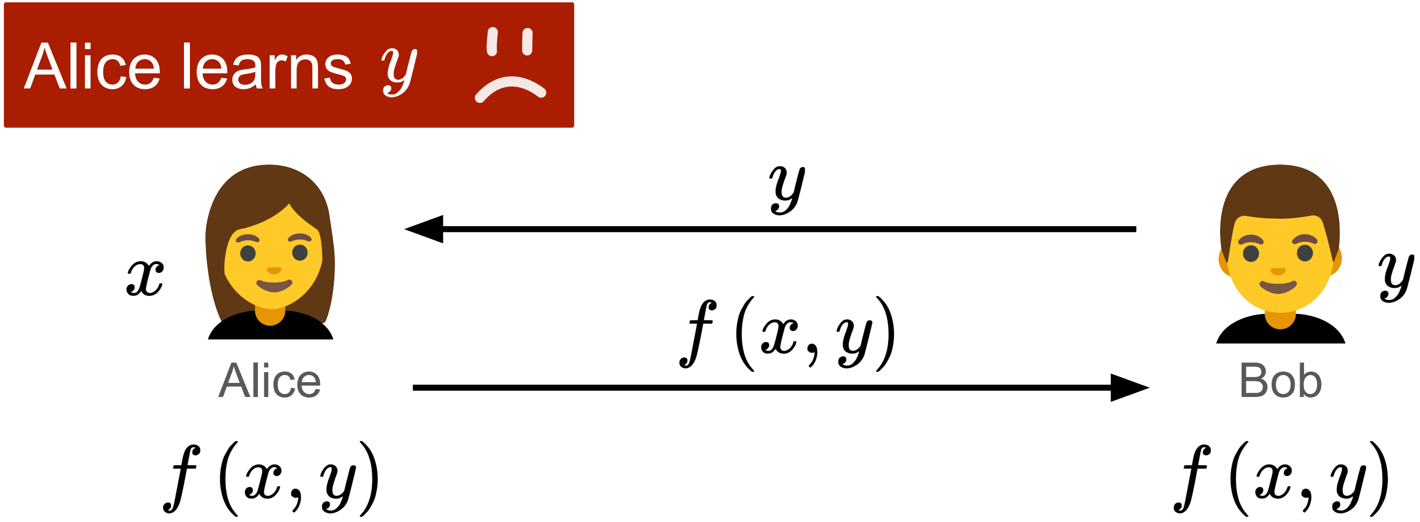
Secure Computation



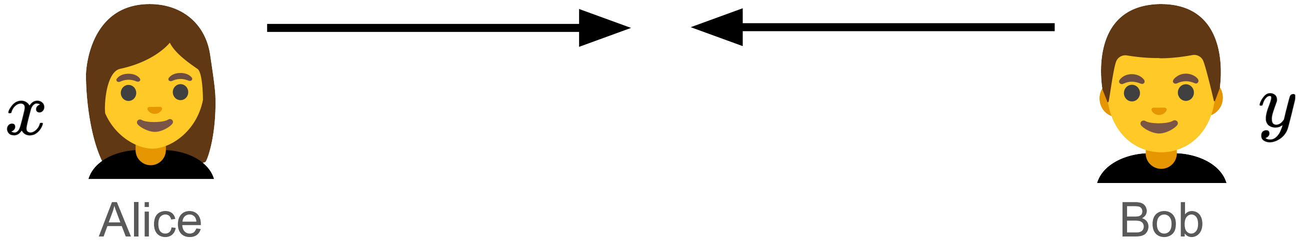
Secure Computation



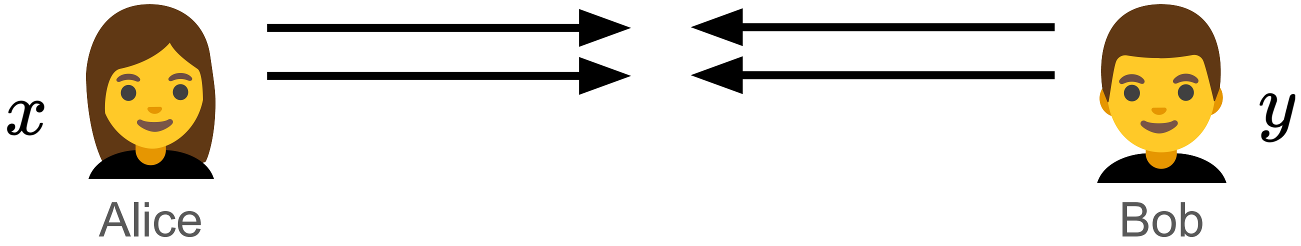
Secure Computation



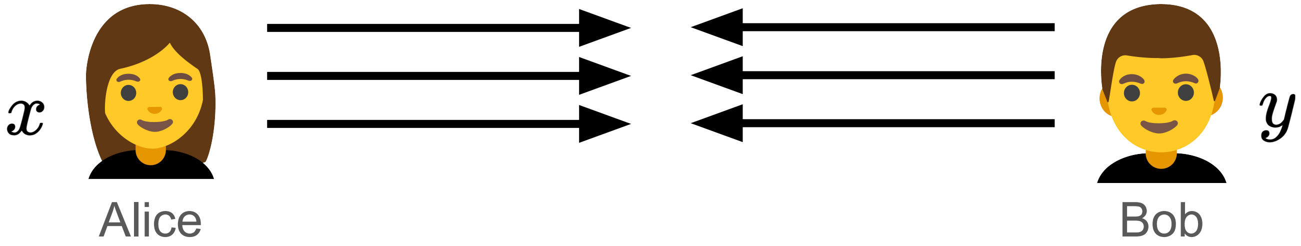
Secure Computation



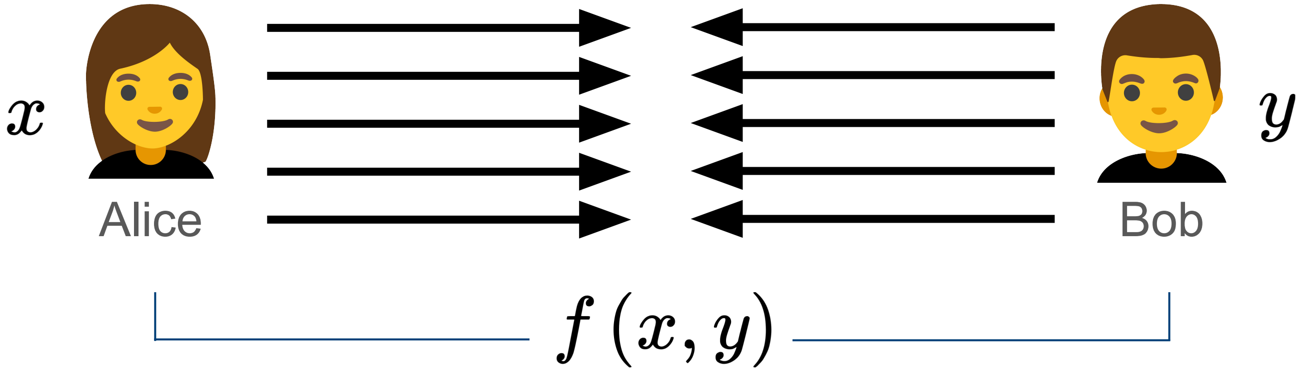
Secure Computation



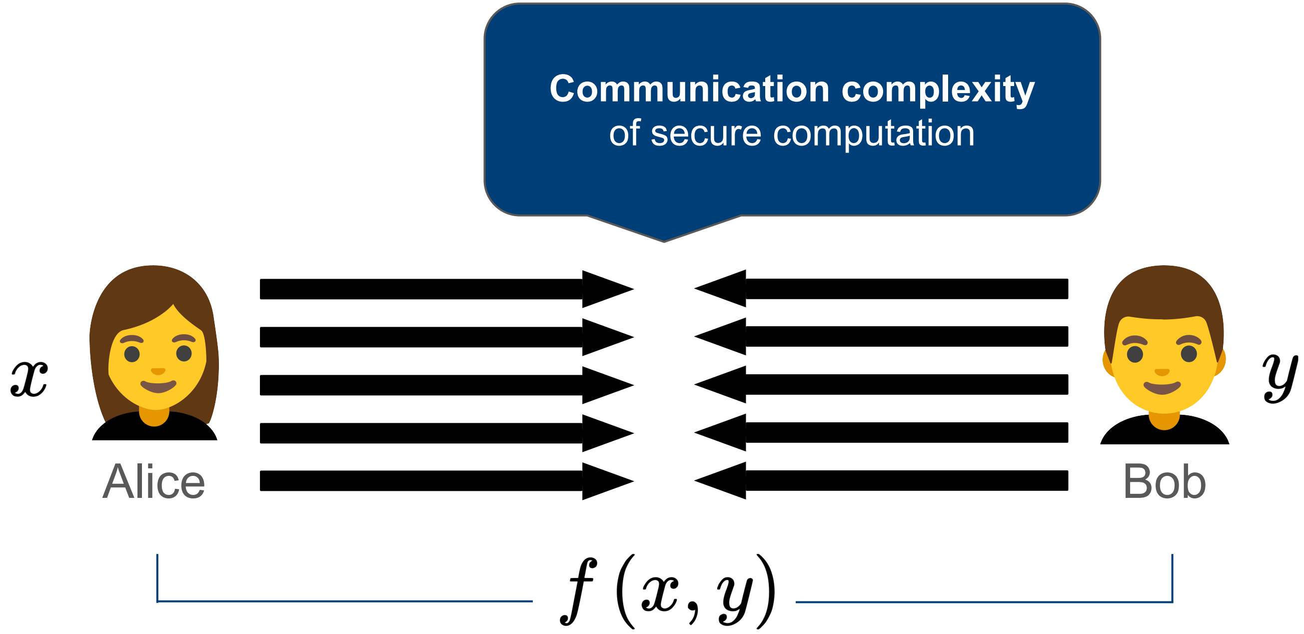
Secure Computation



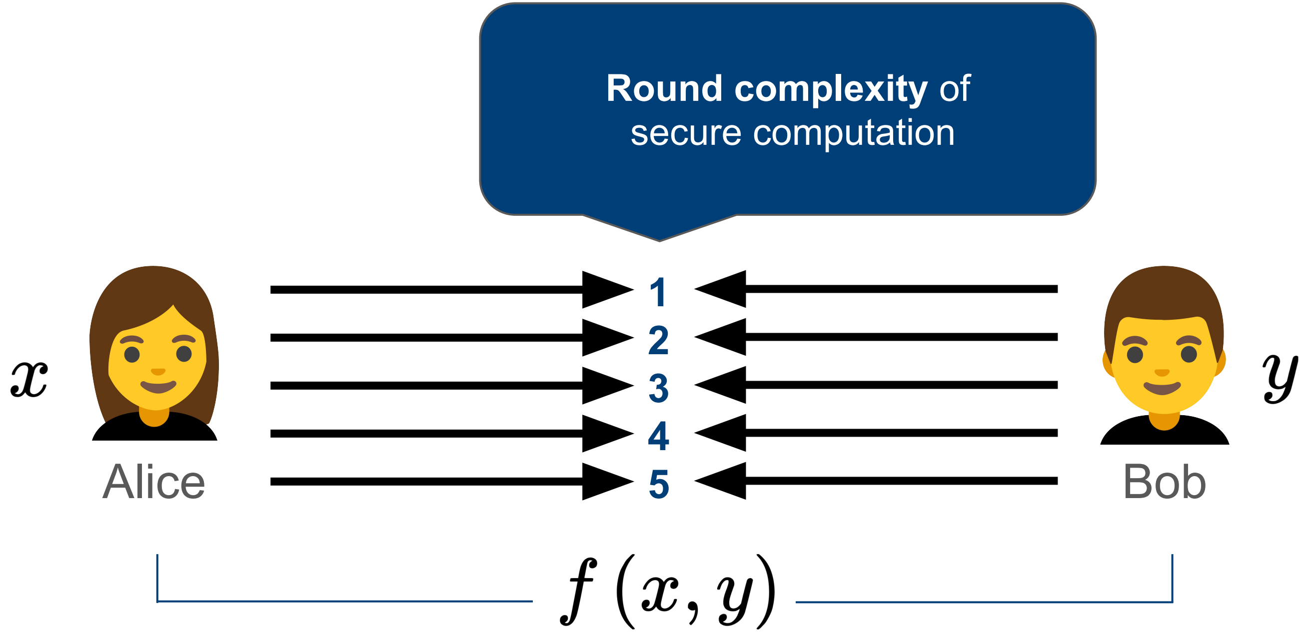
Secure Computation



Secure Computation

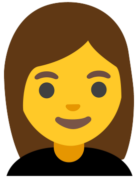


Secure Computation



What can we dream of?

What can we dream of?

x 

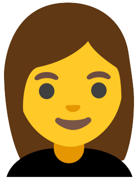
Alice

 y

Bob

What can we dream of?

x



Alice

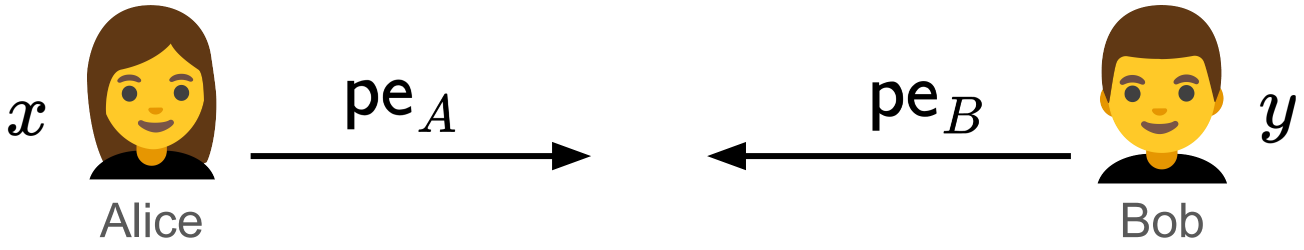
$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

y

Bob

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

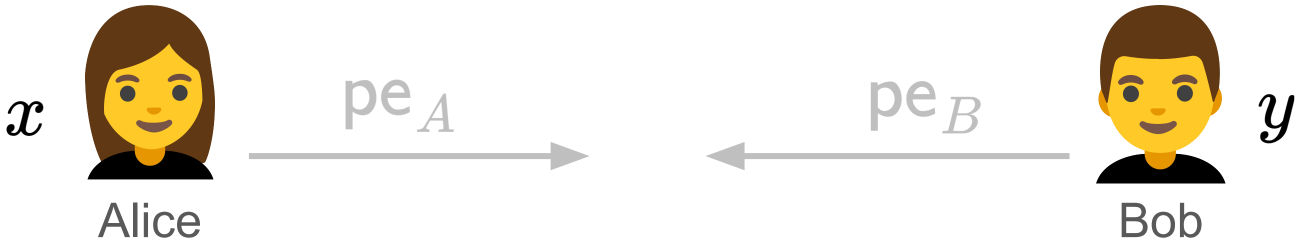
What can we dream of?



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

What can we dream of?

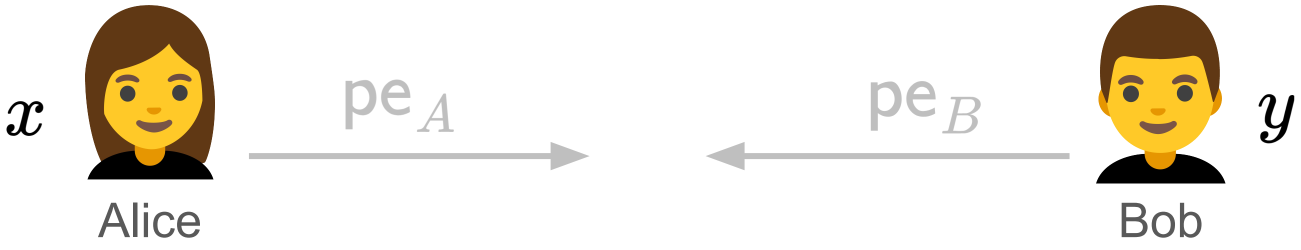


$$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x) \qquad \mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$$

$$f(x,y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B) \qquad f(x,y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP’11]



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

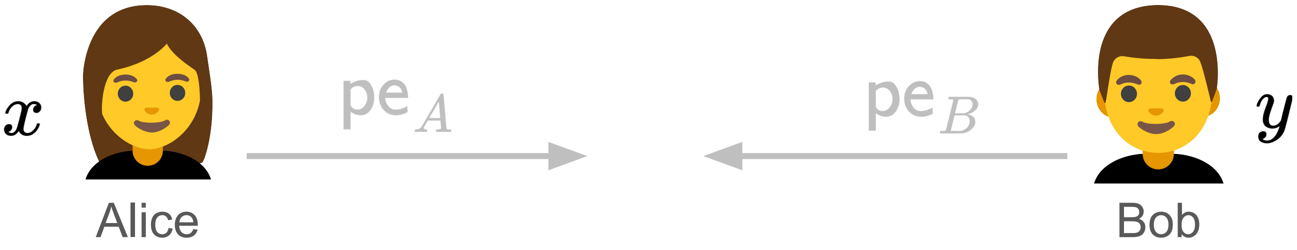
$f(x, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP'11]

Attack: Compute $f(x_1, y), f(x_2, y) \ldots, f(x_n, y)$ using the same pe_B



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

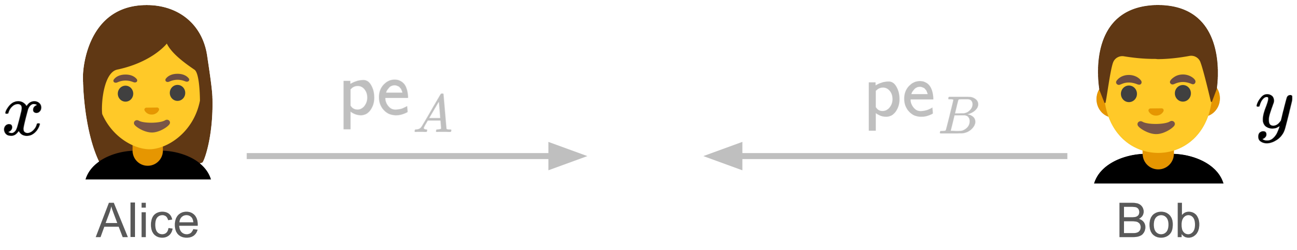
$f(x, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP'11]

Attack: Compute $f(x_1, y), f(x_2, y) \ldots, f(x_n, y)$ using the same pe_B



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

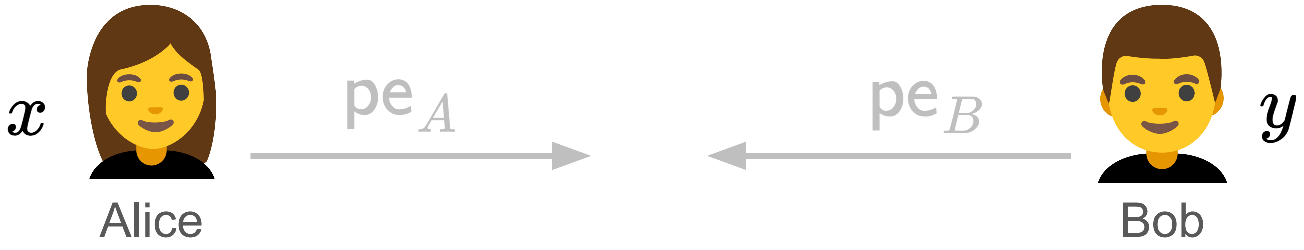
$f(x_1, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP'11]

Attack: Compute $f(x_1, y), f(x_2, y) \ldots, f(x_n, y)$ using the same pe_B



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

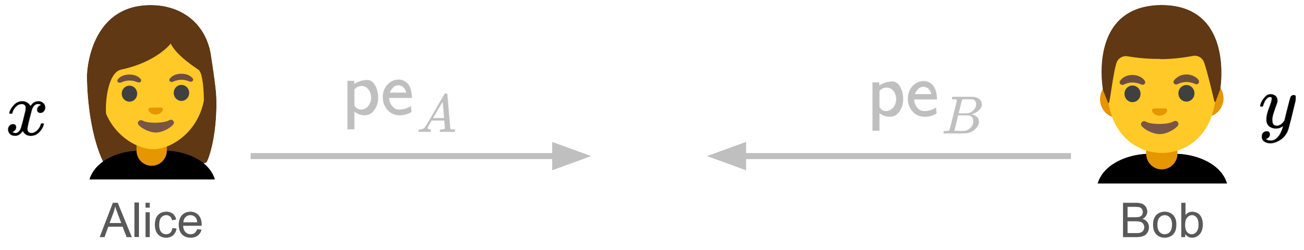
$f(x_2, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP'11]

Attack: Compute $f(x_1, y), f(x_2, y) \ldots, f(x_n, y)$ using the same pe_B



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

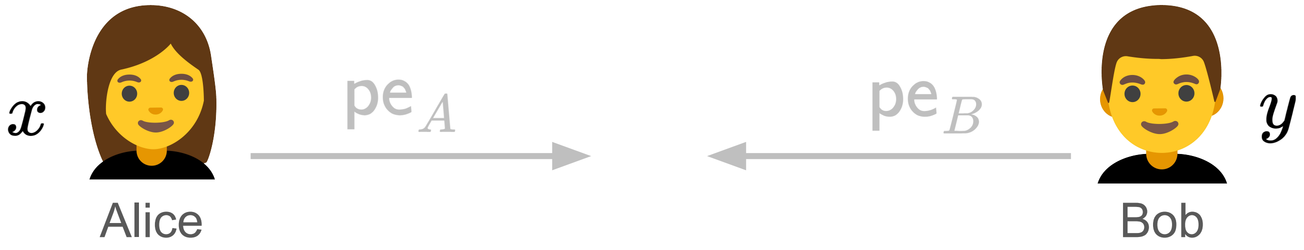
$f(x_3, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Impossible for arbitrary functions

Two-round lower-bound for two party computation [HLP'11]

Attack: Compute $f(x_1, y), f(x_2, y) \dots, f(x_n, y)$ using the same pe_B



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

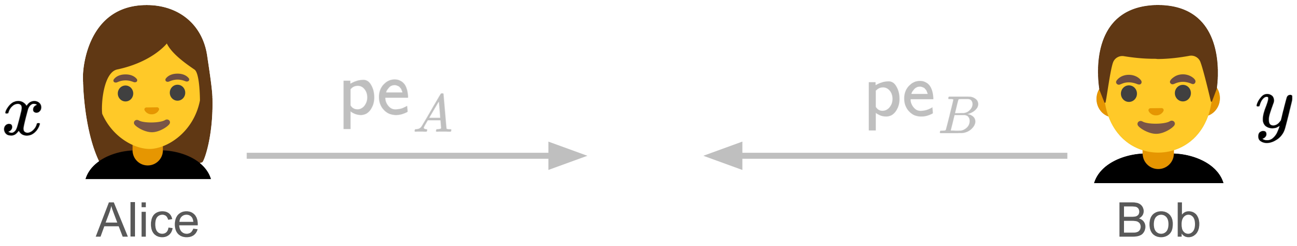
$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

$f(x_3, y) \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$f(x, y) \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Attack: Alice learns more than just $f(x, y)$

Possible for “secret shared” outputs



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

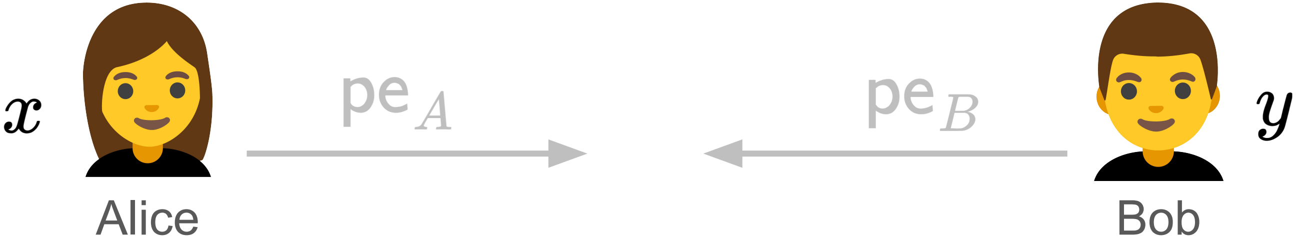
$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

$z_A \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$z_B \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Possible for "secret shared" outputs

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



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

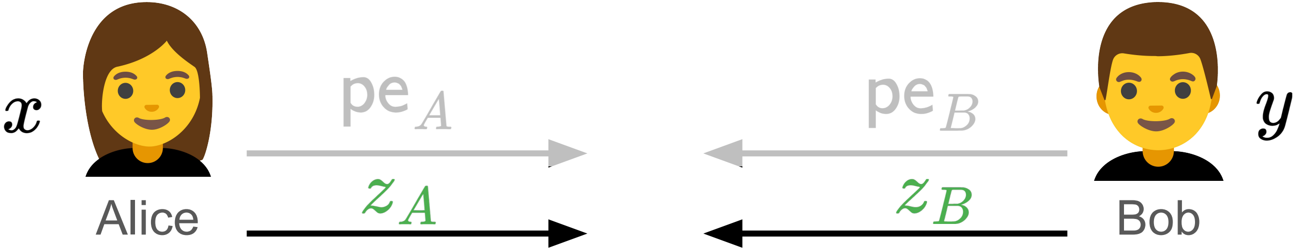
$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

$z_A \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$z_B \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Possible for "secret shared" outputs

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



$\mathsf{pe}_A \leftarrow \mathsf{Encode}_A(x)$

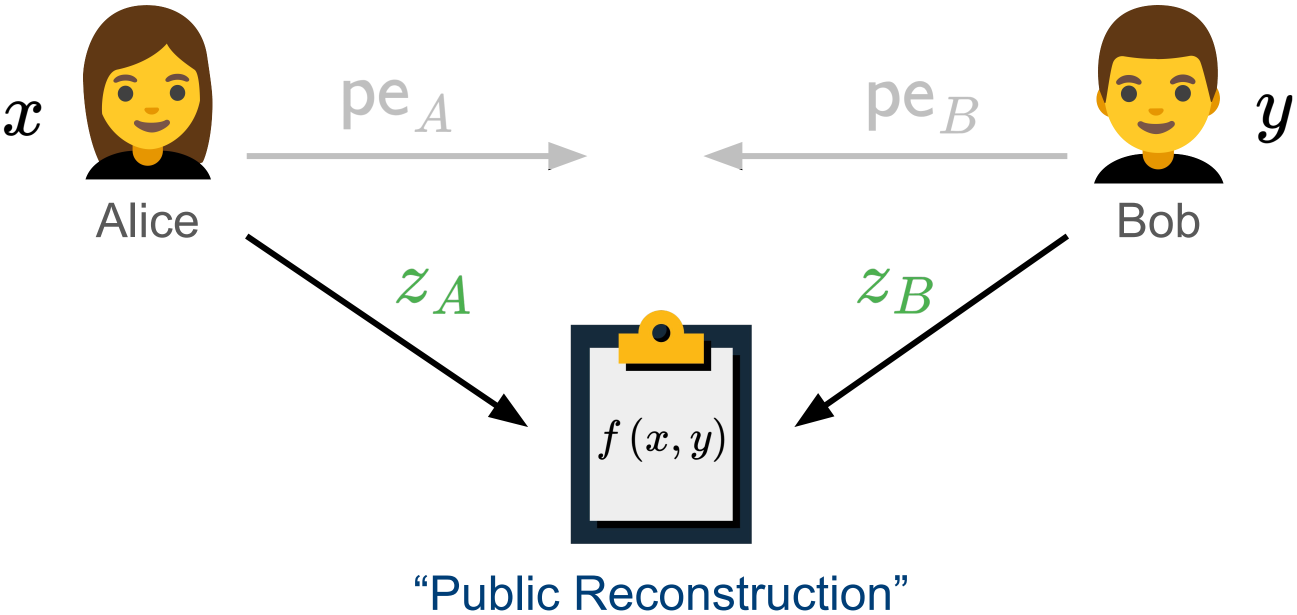
$\mathsf{pe}_B \leftarrow \mathsf{Encode}_B(y)$

$z_A \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

$z_B \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

Possible for "secret shared" outputs

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



A history of secure computation

A history of (two-round) secure computation

Garbled Circuits [Yao'86]

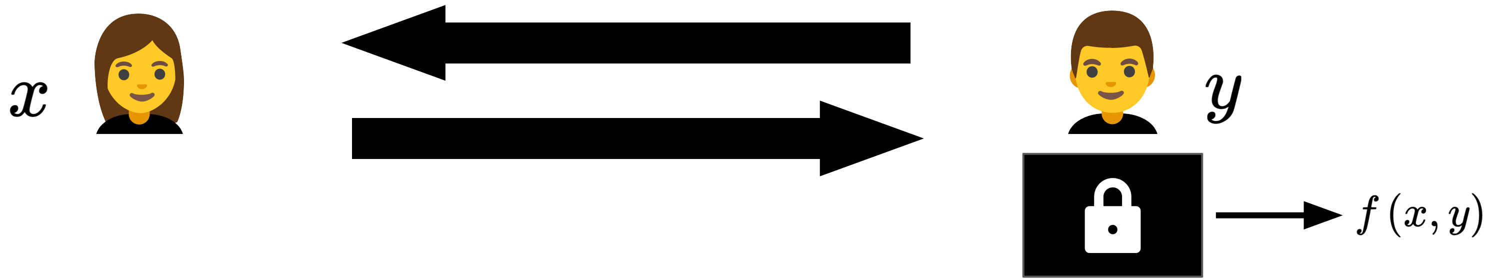
x 👩 👨 y

A history of (two-round) secure computation

Garbled Circuits [Yao’86]

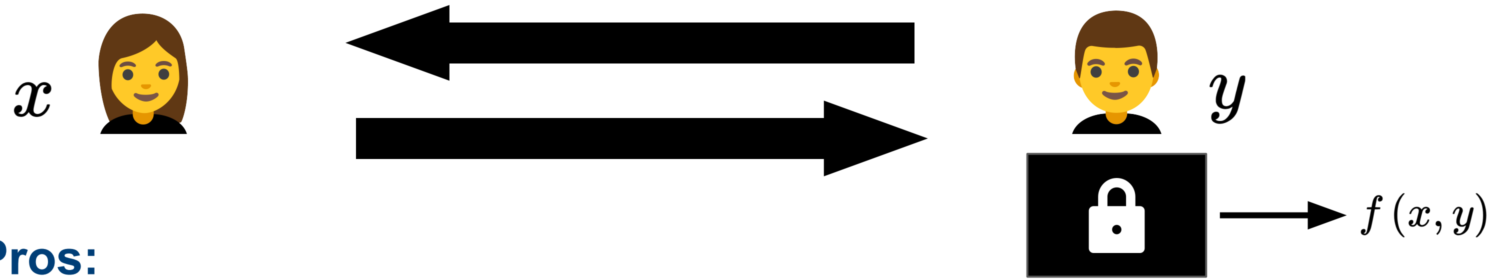
A history of (two-round) secure computation

Garbled Circuits [Yao'86]



A history of (two-round) secure computation

Garbled Circuits [Yao'86]

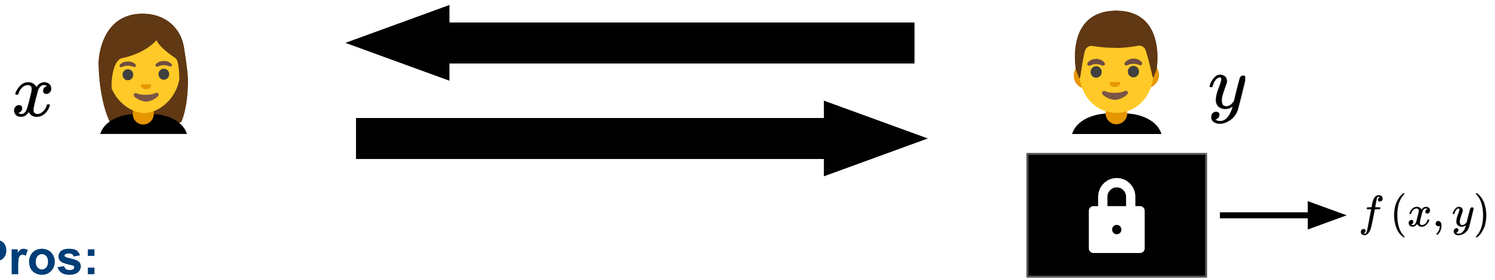


Pros:

- Two rounds (assuming two-round OT) ✓
- Requires minimal assumptions ✓

A history of (two-round) secure computation

Garbled Circuits [Yao'86]



Pros:

- Two rounds (assuming two-round OT) ✓
- Requires minimal assumptions ✓

Cons:

- Linear communication in the circuit size
- No public reconstruction

A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]

x 👩 👨 y

A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]

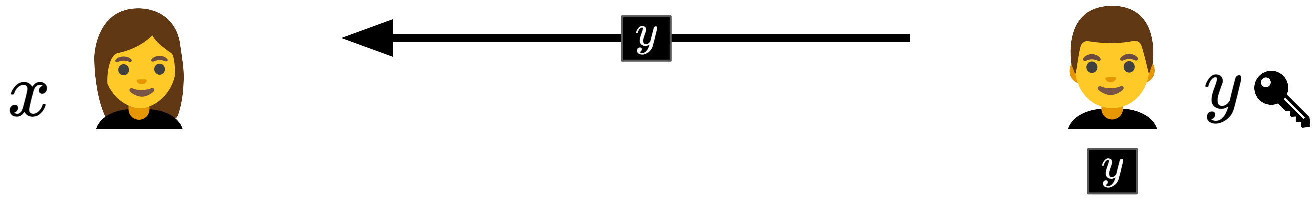
x 👩 👨 y 🔑

A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]

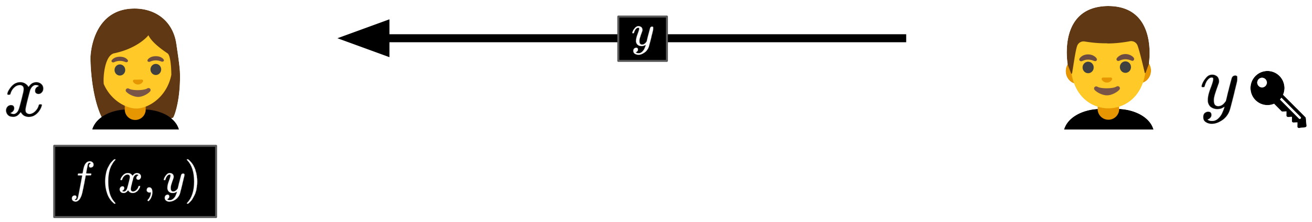
A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]



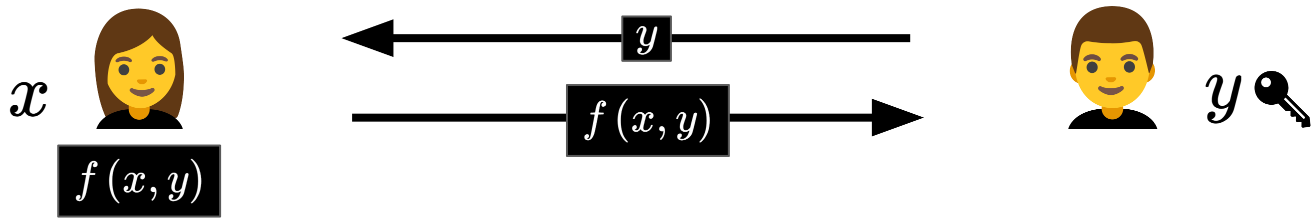
A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]



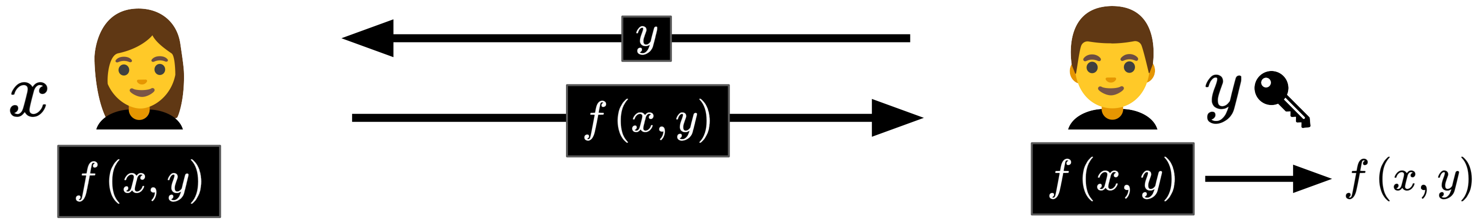
A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]



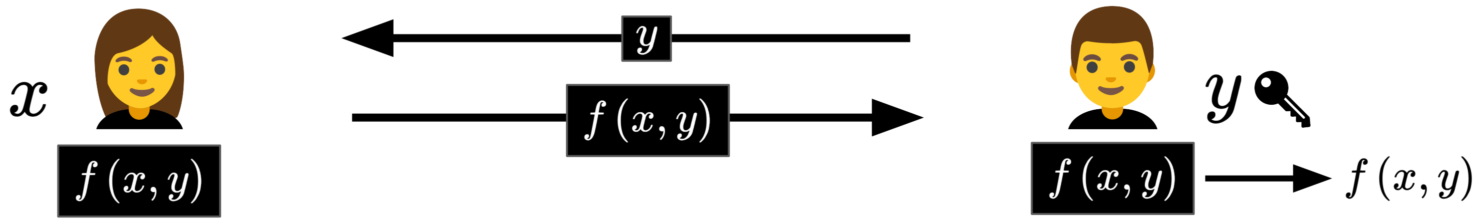
A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry’09]



A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]

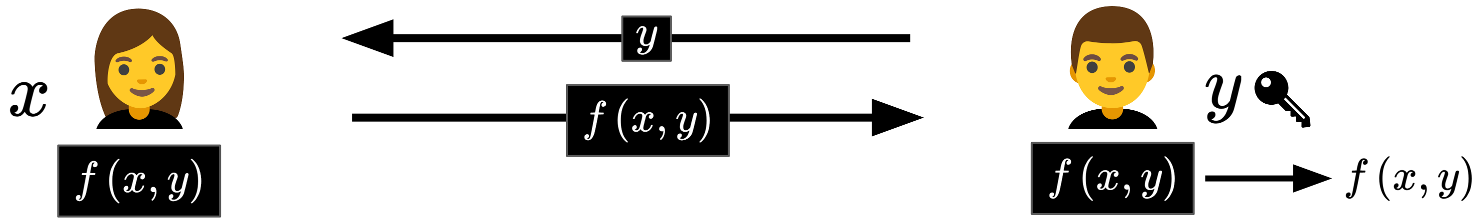


Pros:

- Two rounds ✓
- Sublinear communication in the circuit size ✓

A history of (two-round) secure computation

Fully Homomorphic Encryption [Gentry'09]



Pros:

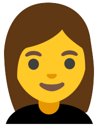
- Two rounds ✓
- Sublinear communication in the circuit size ✓

Cons:

- No public reconstruction

A history of (two-round) secure computation

Spooky Encryption [DHRW'16]

x 

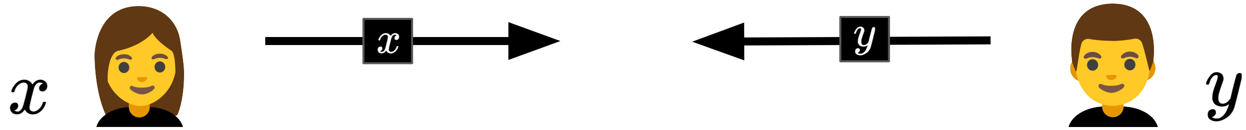
 y

A history of (two-round) secure computation

Spooky Encryption [DHRW'16]

A history of (two-round) secure computation

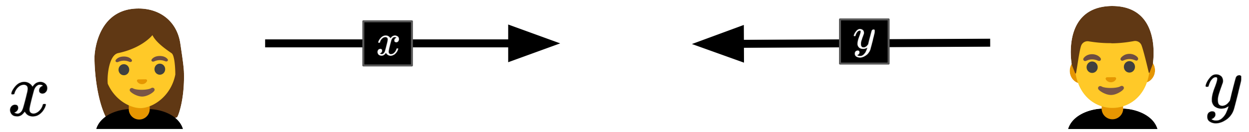
Spooky Encryption [DHRW'16]



$z_A \leftarrow \mathrm{Decode}_A\,(f, x, \boxed{y})$ $z_B \leftarrow \mathrm{Decode}_B\,(f, \boxed{x}, y)$

A history of (two-round) secure computation

Spooky Encryption [DHRW'16]



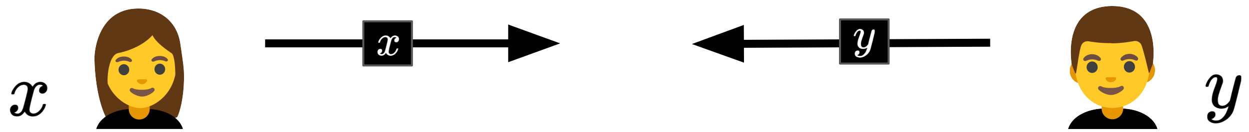
$z_A \leftarrow \mathsf{Decode}_A(f, x, \boxed{y})$ $z_B \leftarrow \mathsf{Decode}_B(f, \boxed{x}, y)$

Pros:

- Two rounds ✓
- Sublinear communication in the circuit size ✓
- Public reconstruction ✓

A history of (two-round) secure computation

Spooky Encryption [DHRW'16]



$z_A \leftarrow \text{Decode}_A(f, x, \boxed{y})$ $z_B \leftarrow \text{Decode}_B(f, \boxed{x}, y)$

Pros:

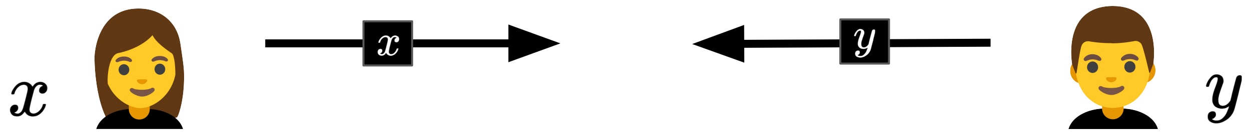
- Two rounds ✓
- Sublinear communication in the circuit size ✓
- Public reconstruction ✓

Cons:

- Only one approach is known

A history of (two-round) secure computation

Spooky Encryption [DHRW'16]



$z_A \leftarrow \text{Decode}_A(f, x, \boxed{y})$ $z_B \leftarrow \text{Decode}_B(f, \boxed{x}, y)$

Pros:

- Two rounds ✓
- Sublinear communication in the circuit size ✓
- Public reconstruction ✓

Spooky encryption gives us one-the-fly secure computation!

A history of (two-round) secure computation

Sacha

Geoffroy

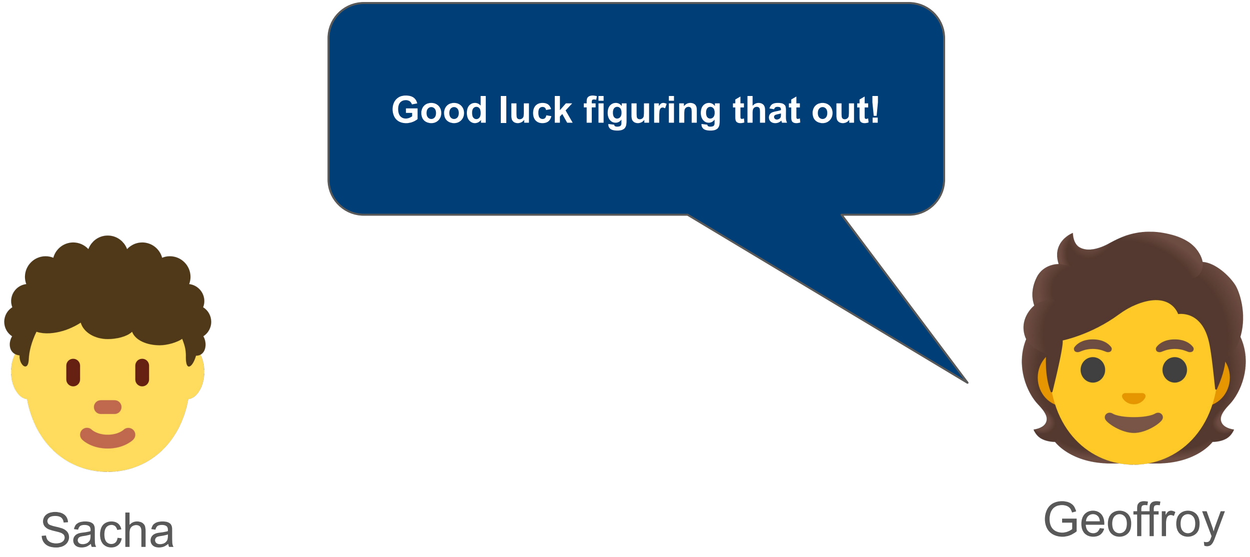
A history of (two-round) secure computation

Is spooky encryption necessary for on-the-flyness?

Sacha

Geoffroy

A history of (two-round) secure computation



Why be spooked by spooky encryption?

Why be spooked by spooky encryption?

Reason 1 (practice): Spooky encryption is a heavy hammer and unlikely to lead concretely efficient protocols.

Why be spooked by spooky encryption?

Reason 1 (practice): Spooky encryption is a heavy hammer and unlikely to lead concretely efficient protocols.

Reason 2 (diversity): Not having all our eggs in one basket (in terms of cryptographic assumptions) is important.

Why be spooked by spooky encryption?

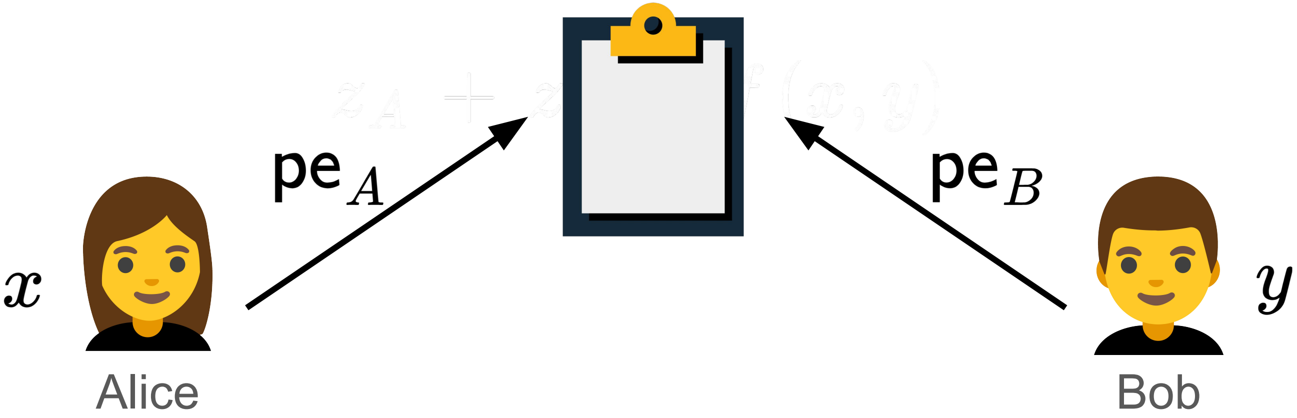
Reason 1 (practice): Spooky encryption is a heavy hammer and unlikely to lead concretely efficient protocols.

Reason 2 (diversity): Not having all our eggs in one basket (in terms of cryptographic assumptions) is important.

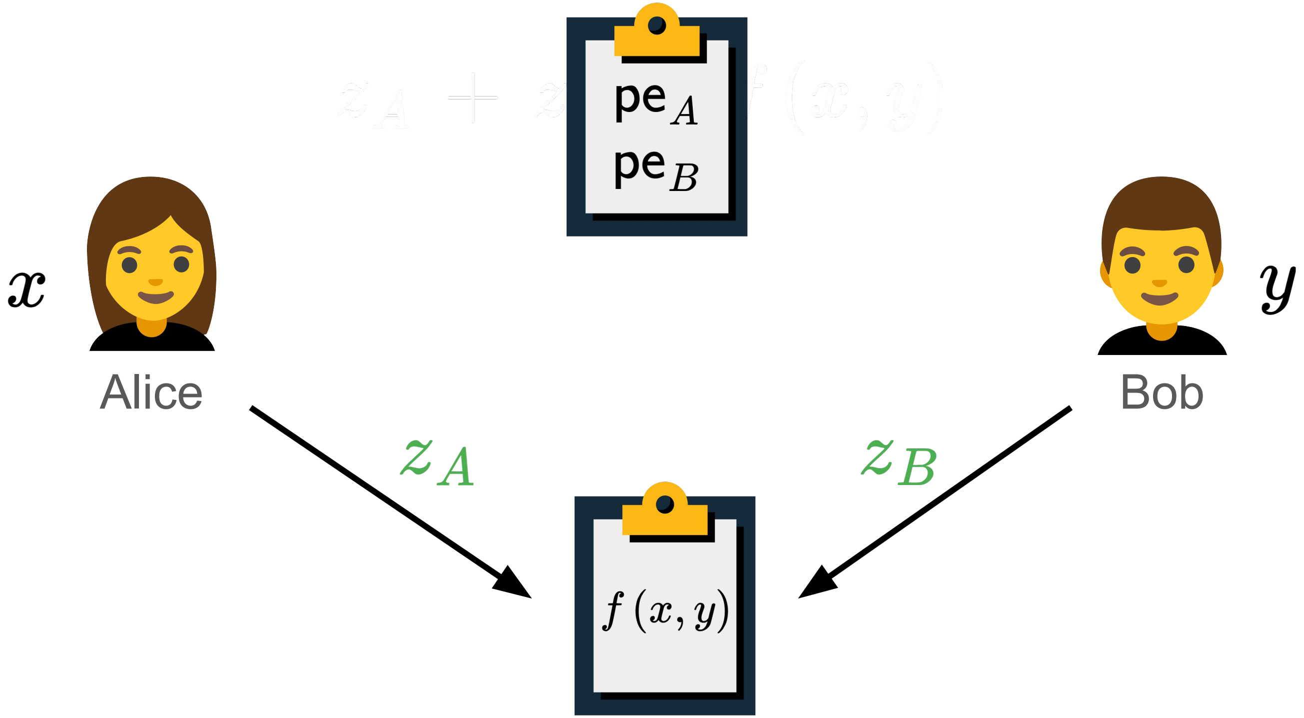
Reason 3 (theory): Finding alternative ways of building something unlocks new insights about the original approach and why it works.

On-the-Fly Secure Computation

On-the-Fly Secure Computation



On-the-Fly Secure Computation

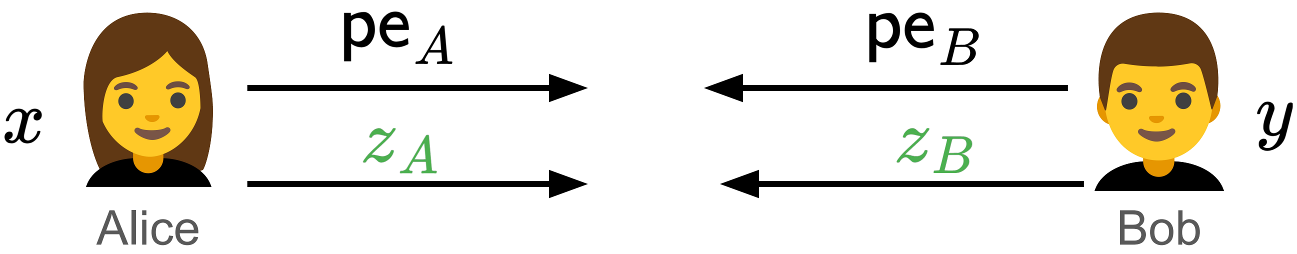


On-the-Fly Secure Computation

- **Implies two-round secure computation**

On-the-Fly Secure Computation

- **Implies two-round secure computation**



On-the-Fly Secure Computation

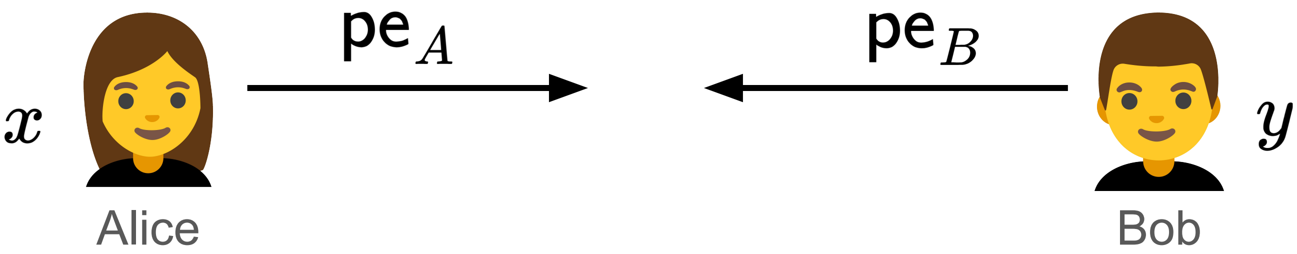
- Implies two-round secure computation
- **Implies key agreement (and so it is black-box separable from OT [GKM+00])**

On-the-Fly Secure Computation

- Implies two-round secure computation
- **Implies key agreement (and so it is black-b ...)**

Alice and Bob get the same pseudorandom "share" i.e., key

$$z_A - z_B = 0 \cdot f(x, y) \implies z_A = z_B$$



$z_A \leftarrow \mathsf{Decode}_A(\mathsf{pe}_B)$

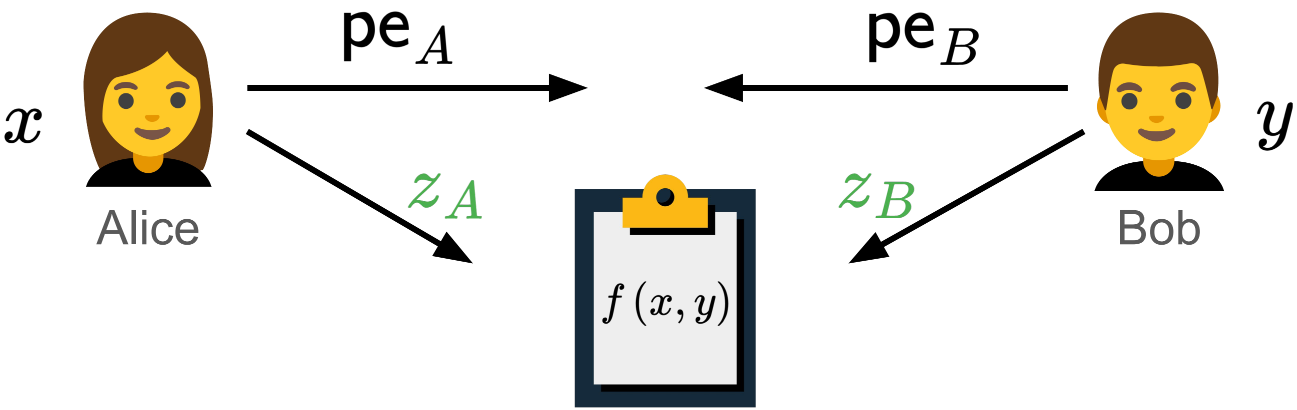
$z_B \leftarrow \mathsf{Decode}_B(\mathsf{pe}_A)$

On-the-Fly Secure Computation

- Implies two-round secure computation
- Implies key agreement (and so it is black-box separable from OT [GKM+00])
- **Public reconstruction: Alice and Bob can publicly disclose the output**

On-the-Fly Secure Computation

- Implies two-round secure computation
- Implies key agreement (and so it is black-box separable from OT [GKM+00])
- **Public reconstruction: Alice and Bob can publicly disclose the output**

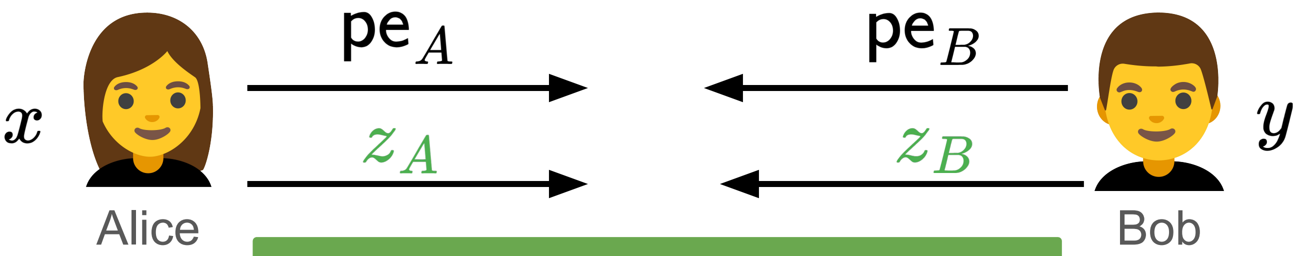


On-the-Fly Secure Computation

- Implies two-round secure computation
- Implies key agreement (and so it is black-box separable from OT [GKM+00])
- Public reconstruction: Alice and Bob can publicly disclose the output
- **Optimal output size (same size as the function output)**

On-the-Fly Secure Computation

- Implies two-round secure computation
- Implies key agreement (and so it is black-box separable from OT [GKM+00])
- Public reconstruction: Alice and Bob can publicly disclose the output
- **Optimal output size (same size as the function output)**



$$|z_A| = |z_B| = |f(x,y)|$$

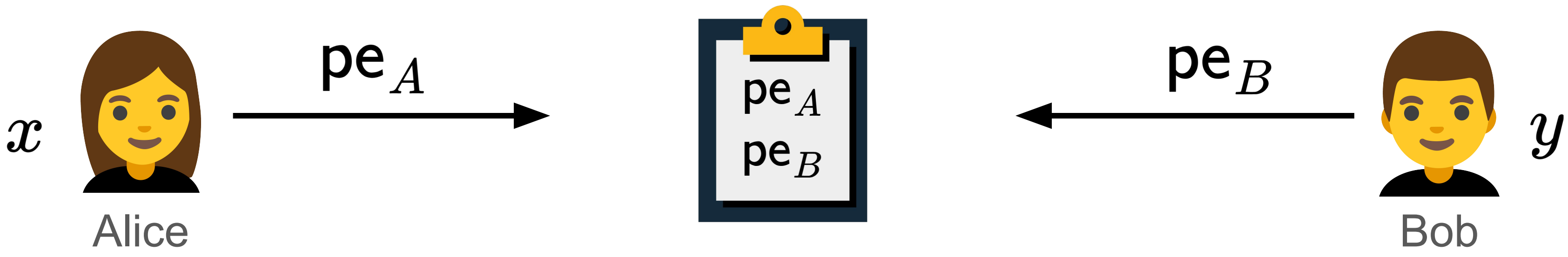
Why is

Sublinearity + Two-Rounds + Public Reconstruction

the gold standard?

Minimal Requirements <> Easy Deployments

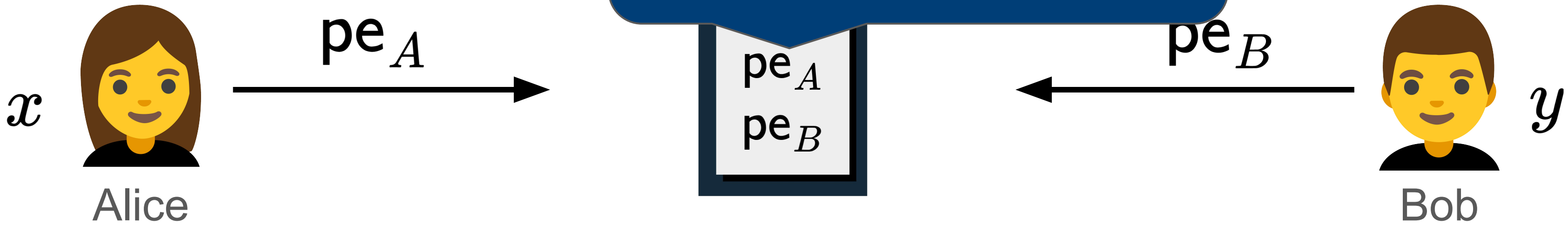
- Easy to deploy protocols that don't depend on people
- Truly "asynchronous" model of communication



Minimal Requirements <> Easy Deployments

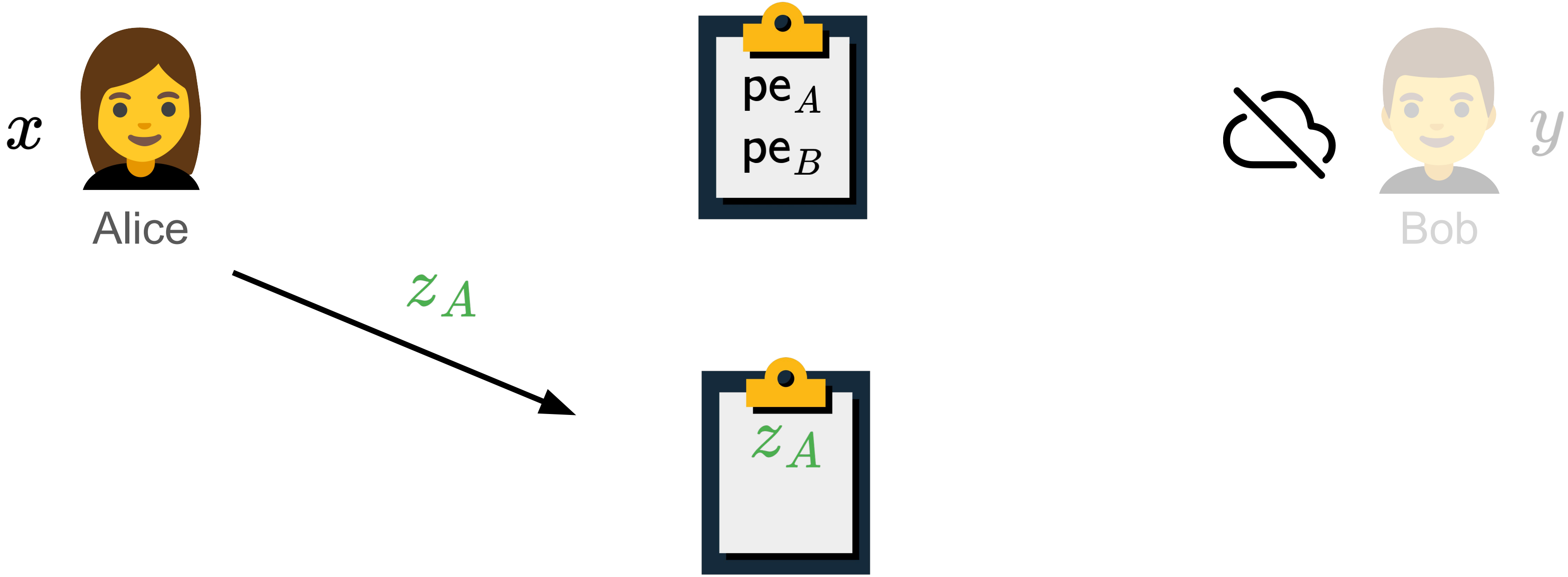
- Easy to deploy protocols
- Truly “asynchronous” mo

These are “succinct” with respect to the function description…



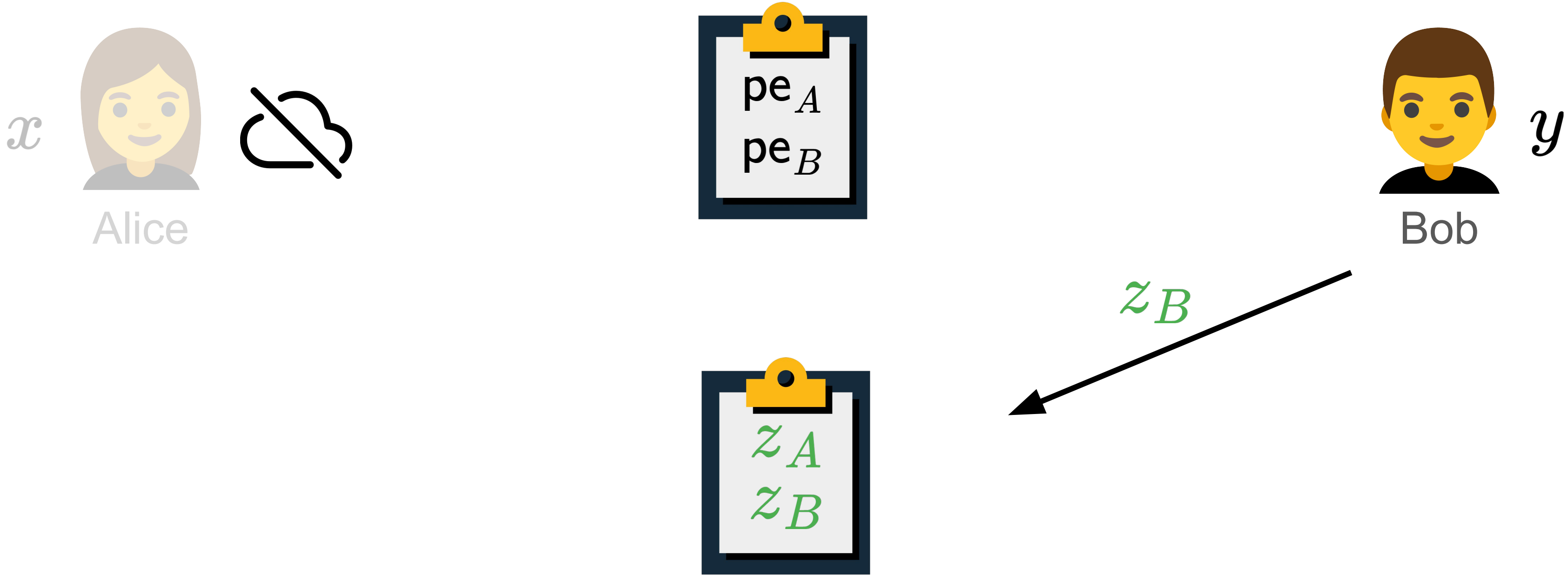
Minimal Requirements <> Easy Deployments

- Easy to deploy protocols that don't depend on people
- Truly "asynchronous" model of communication



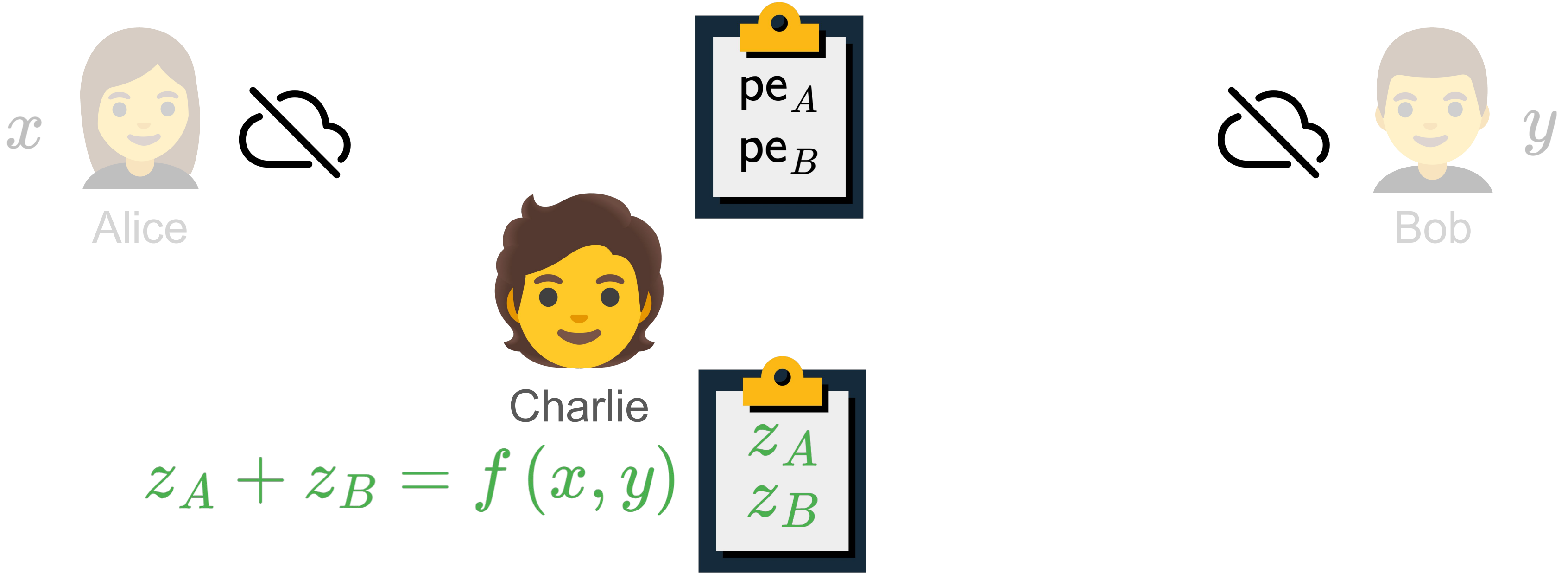
Minimal Requirements <> Easy Deployments

- Easy to deploy protocols that don't depend on people
- Truly "asynchronous" model of communication



Minimal Requirements <> Easy Deployments

- Easy to deploy protocols that don't depend on people
- Truly "asynchronous" model of communication



The current landscape

Building On-the-Fly Secure Computation

Constant-Degree Polynomials [BM’89, CZ’23, BCMPR’24, ARS’24]

From DCR, LPN, DDH

Building On-the-Fly Secure Computation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

Building On-the-Fly Secure Computation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Can we build anything here?

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

Building On-the-Fly Secure Computation

All Functions from Spooky Encryption [DHRW16]

Fror ... n

Like all functions in NC^1

Can we build anything here?

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

What about here?

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Can we build anything here?

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

Overview of thesis results

Contributions of this Thesis

Practice

Theory

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

Practice

Theory

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**'24]

QuietOT: Lightweight Oblivious
Transfer with a Public-Key Setup
[CDDK**SS**'24]

Practice

Theory

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**'24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**'24]

} Previously post-quantum constructions only known from Spooky Encryption

---

Theory

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**’24]

Previously post-quantum constructions only known from Spooky Encryption

---

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**’24]

} Previously post-quantum constructions only known from Spooky Encryption

---

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions with a Non-Interactive Setup
[BD**SS**’25]

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**’24]

} Previously post-quantum constructions only known from Spooky Encryption

---

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions with a Non-Interactive Setup
[BD**SS**’25]

Simultaneous Message and Succinct (SMS) Secure Computation
[BJ**SS**S’25]

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**’24]

Previously post-quantum constructions only known from Spooky Encryption

---

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions with a Non-Interactive Setup
[BD**SS**’25]

Previously only known from Spooky Encryption

Simultaneous Message and Succinct (SMS) Secure Computation
[BJ**SS**S’25]

Contributions of this Thesis

Constrained PRFs for Inner-Product Predicates
[**SS**'24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup
[CDDK**SS**'24]

Previously post-quantum constructions only known from Spooky Encryption

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions with a Non-Interactive Setup
[BD**SS**'25]

Previously only known from Spooky Encryption

Simultaneous Message and Succinct (SMS) Secure Computation
[BJ**SS**S'25]

Previously unknown

Building On-the-Fly Secure Computation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Constant-Degree Polynomials [BM’89, CZ’23, BCMPR’24, ARS’24]

From DCR, LPN, DDH

Building On-the-Fly Secure Computation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Multi-Key Homomorphic Secret Sharing [CDHJSS'25]

From DCR

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

Simultaneous-Message and Succinct (SMS) Secure Computation [BJSSS'25]

From LWE or Indistinguishability Obfuscation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Multi-Key Homomorphic Secret Sharing [CDHJSS'25]

From DCR

Constant-Degree Polynomials [BM'89, CZ'23, BCMPR'24, ARS'24]

From DCR, LPN, DDH

Simultaneous-Message and Succinct (SMS) Secure Computation [BJSSS'25]

From LWE or Indistinguishability Obfuscation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Multi-Key Homomorphic Secret Sharing [CDHJSS'25]

From DCR

Non-Interactive Distributed Point Functions [BDSS'25]

From DCR, QR, DXDH, DDH in class groups

Simultaneous-Message and Succinct (SMS) Secure Computation [BJSSS'25]

From LWE or Indistinguishability Obfuscation

All Functions from Spooky Encryption [DHRW16]

From LWE or Indistinguishability Obfuscation

Multi-Key Homomorphic Secret Sharing [CDHJSS'25]

From DCR

Non-Interactive Distributed Point Functions [BDSS'25]

From DCR, QR, DXDH, DDH in class groups

Lightweight, Non-Interactive OT Extension [CDDKSS'24]

From Post-Quantum Assumptions

This Talk

Constrained PRFs for Inner-Product Predicates

[**SS**'24]

Practice

QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup

[CDDK**SS**'24]

Theory

Multi-key Homomorphic Secret Sharing

[CDHJ**SS**'25]

Distributed Point Functions with a Non-Interactive Setup

[BD**SS**'25]

Simultaneous Message and Succinct (SMS) Secure Computation

[BJ**SS**S'25]

This Talk

Constrained PRFs for Inner-Product Predicates
[**SS**'24]

QuietOT: Lightweight Oblivious
Transfer with a Public-Key Setup
[CDDK**SS**'24]

Practice

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions
with a Non-Interactive Setup
[BD**SS**'25]

Simultaneous Message and Succinct (SMS)
Secure Computation
[BJ**SS**S'25]

This Talk

Constrained PRFs for Inner-Product Predicates
[**SS**'24]

Quiet

Practice

Transfe

Just overview of results

Theory

Multi-key Homomorphic Secret Sharing
[CDHJ**SS**'25]

Distributed Point Functions
with a Non-Interactive Setup
[BD**SS**'25]

Simultaneous Message and Succinct (SMS)
Secure Computation
[BJ**SS**S'25]

This Talk

Constrained PRFs for Inner-Product Predicates
[**SS**’24]

QuietOT: Lightweight Oblivious
Transfer with a Public-Key Setup
[CDDK**SS**’24]

Practice

Theory

Multi-key Homomorphic Secret Sharing

Dist
with

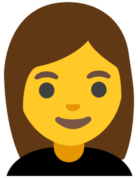
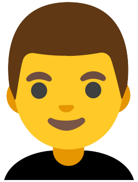
Technical details & construction

Simultaneous Message and Succinct (SMS)
Secure Computation
[BJ**SS**S’25]

Multi-key Homomorphic Secret Sharing

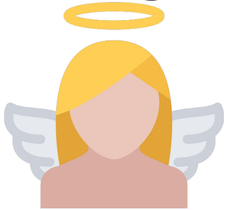
Joint work with
Geoffroy Couteau, Lali Devadas, Aditya Hegde, and Abhishek Jain

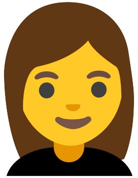
Homomorphic Secret Sharing [BGI'16]

x  y 

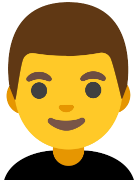
Alice **Bob**

Homomorphic Secret Sharing [BGI'16]



x 

Alice

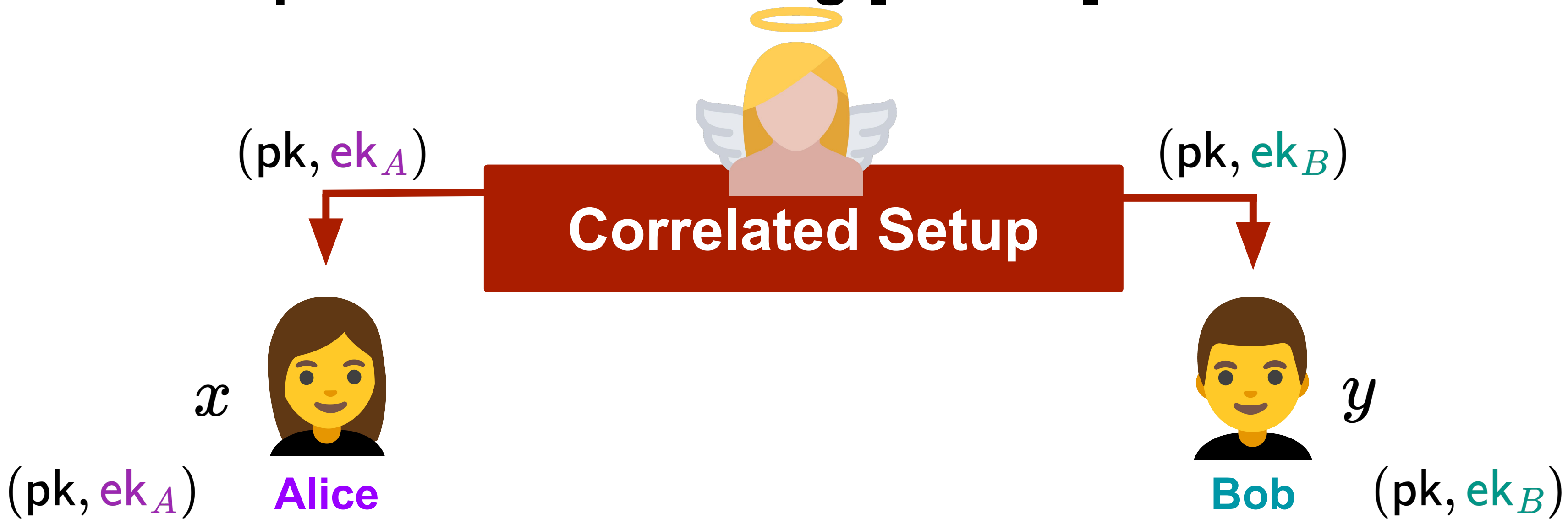
 y

Bob

Homomorphic Secret Sharing [BGI'16]

Homomorphic Secret Sharing [BGI'16]

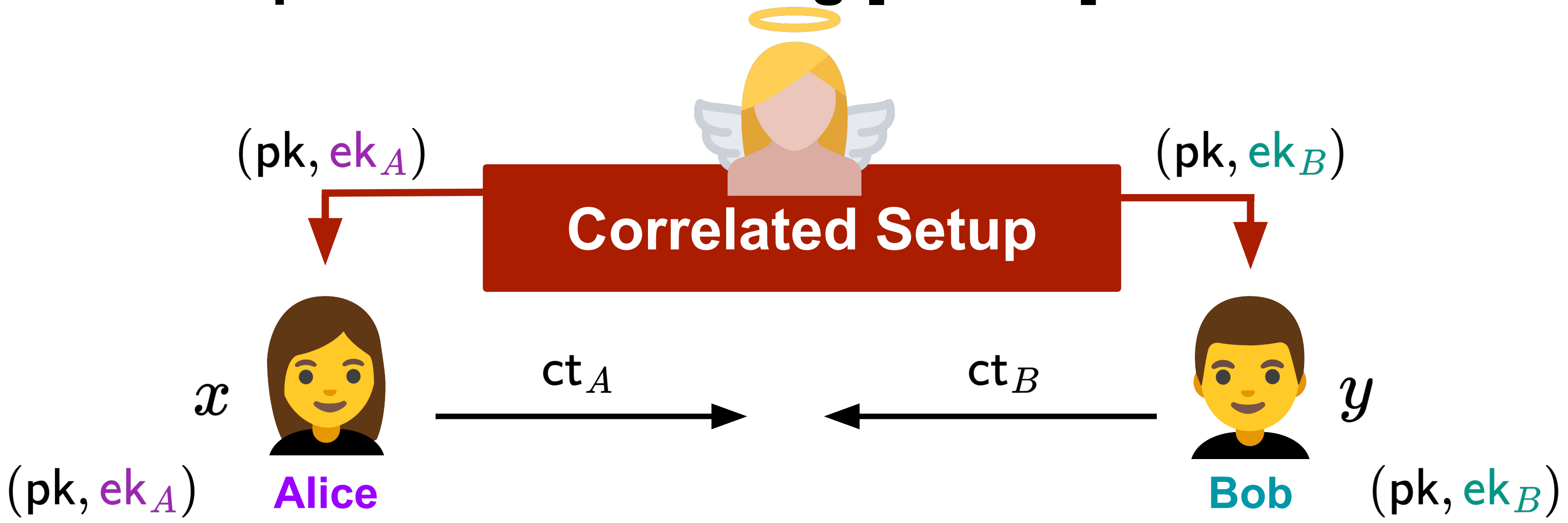
Homomorphic Secret Sharing [BGI'16]



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

$\mathsf{ct}_B \leftarrow \mathsf{Encrypt}(\mathsf{pk}, y)$

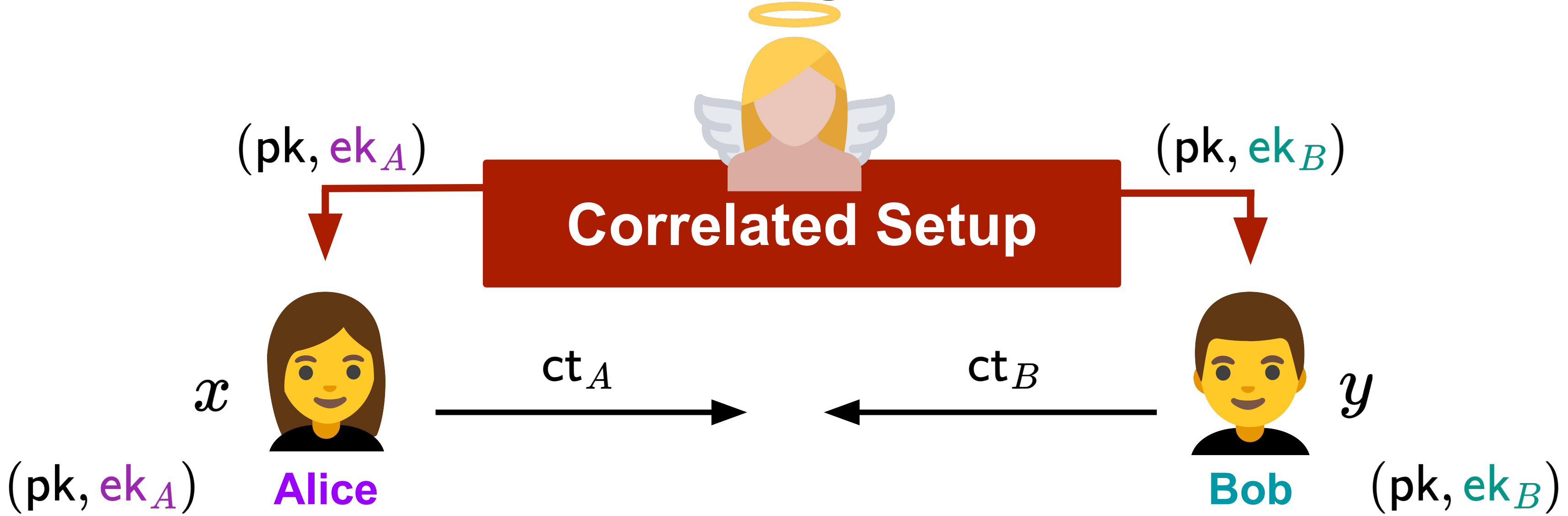
Homomorphic Secret Sharing [BGI'16]



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

$$\mathsf{ct}_B \leftarrow \mathsf{Encrypt}(\mathsf{pk}, y)$$

Homomorphic Secret Sharing [BGI'16]



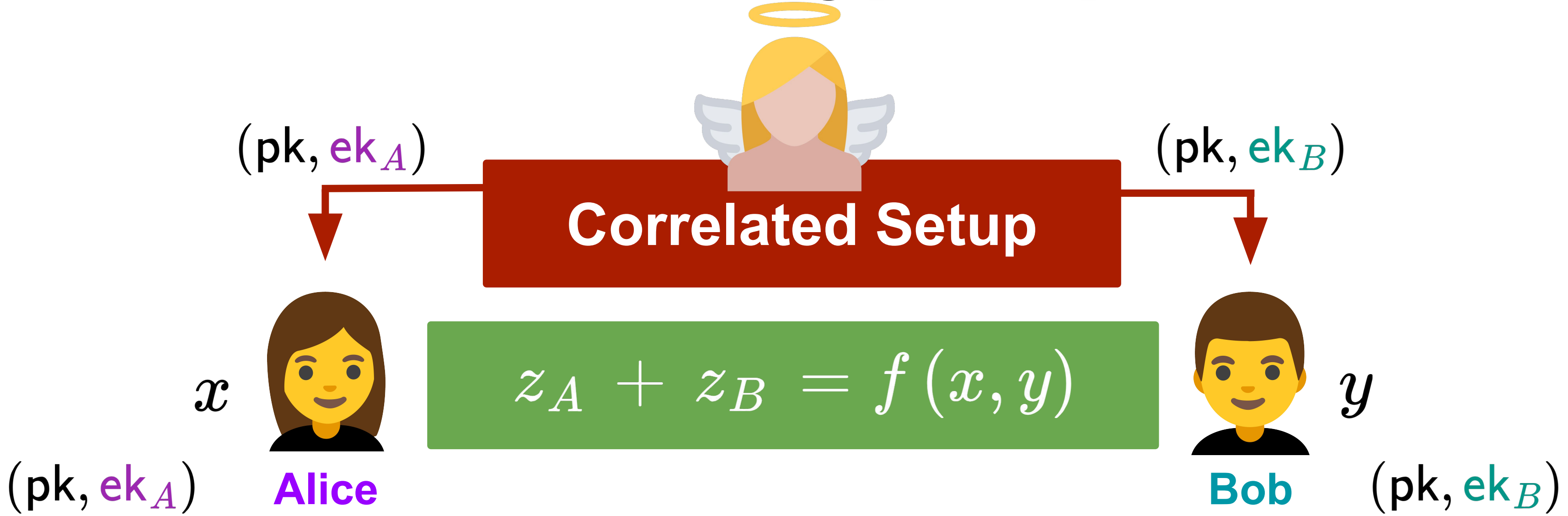
$\mathsf{ct}_A \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$

$\mathsf{ct}_B \leftarrow \mathsf{Encrypt}(\mathsf{pk}, y)$

$z_A \leftarrow \mathsf{Eval}(\mathsf{pk},\ f,\ \mathsf{ek}_A,\ \mathsf{ct}_A, \mathsf{ct}_B)$

$z_B \leftarrow \mathsf{Eval}(\mathsf{pk},\ f,\ \mathsf{ek}_B,\ \mathsf{ct}_A, \mathsf{ct}_B)$

Homomorphic Secret Sharing [BGI'16]



$\mathsf{ct}_A \leftarrow \mathsf{Encrypt}(\mathsf{pk}, x)$ $\quad$ $\mathsf{ct}_B \leftarrow \mathsf{Encrypt}(\mathsf{pk}, y)$

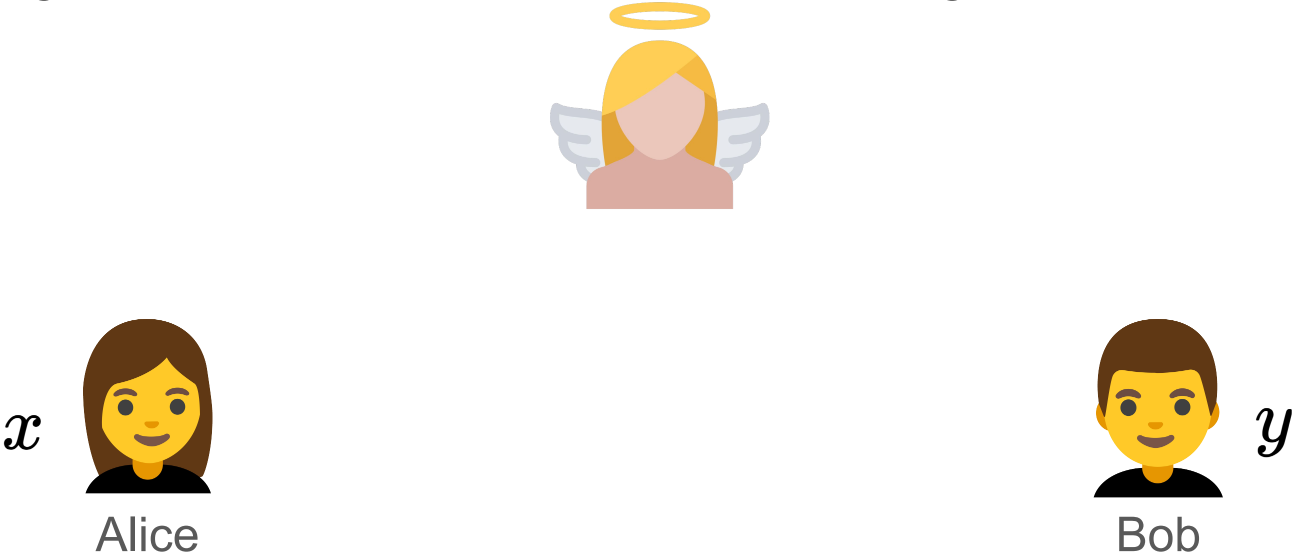
$z_A \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_A, \mathsf{ct}_A, \mathsf{ct}_B)$ $\quad$ $z_B \leftarrow \mathsf{Eval}(\mathsf{pk}, f, \mathsf{ek}_B, \mathsf{ct}_A, \mathsf{ct}_B)$

Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]

x Alice

Bob y

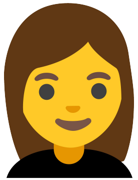
Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]



Multi-Key Homomorphic Secret Sharing [CDHJ**SS**’25]

Multi-Key Homomorphic Secret Sharing [CDHJ**SS**’25]

No Correlated Setup

x 

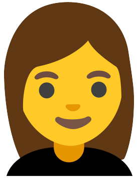
Alice

 y

Bob

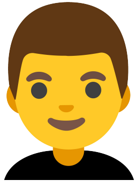
Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]

No Correlated Setup

x 

Alice

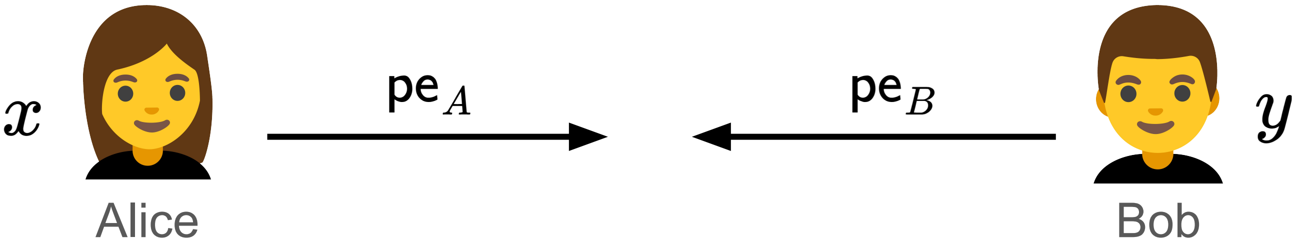
$(\mathsf{pe}_A, \mathsf{st}_A) \leftarrow \mathsf{Encode}_A(x)$

 y

Bob

$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(y)$

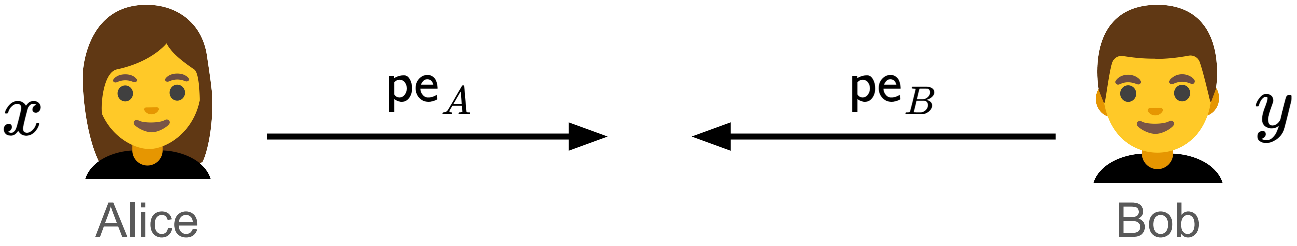
Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]



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

$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(y)$

Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]



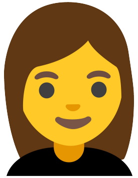
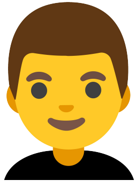
$$(\mathsf{pe}_A, \mathsf{st}_A) \leftarrow \mathsf{Encode}_A(x)$$

$$z_A \leftarrow \mathsf{Eval}_A(f, \mathsf{pe}_B, \mathsf{st}_A)$$

$$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(y)$$

$$z_B \leftarrow \mathsf{Eval}_B(f, \mathsf{pe}_A, \mathsf{st}_B)$$

Multi-Key Homomorphic Secret Sharing [CDHJ**SS**'25]

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

Alice

Bob

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

$z_A \leftarrow \mathsf{Eval}_A(f, \mathsf{pe}_B, \mathsf{st}_A)$

$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(y)$

$z_B \leftarrow \mathsf{Eval}_B(f, \mathsf{pe}_A, \mathsf{st}_B)$

Our results

- **First construction of multi-key HSS for NC^1 from the DCR assumption**

Our results

- First construction of multi-key HSS for NC^1 from the DCR assumption
- Applications include:

Our results

- **First construction of multi-key HSS for NC^1 from the DCR assumption**
- **Applications include:**
    - First construction of sublinear, two-round secure computation from DCR

Our results

- **First construction of multi-key HSS for NC^1 from the DCR assumption**
- **Applications include:**
  - First construction of sublinear, two-round secure computation from DCR
  - Better communication in secure multi-party computation

Our results

- **First construction of multi-key HSS for NC^1 from the DCR assumption**
- **Applications include:**
  - First construction of sublinear, two-round secure computation from DCR
  - Better communication in secure multi-party computation
  - Non-interactive attribute based key exchange in the standard model

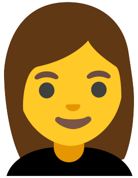
Our results

- **First construction of multi-key HSS for NC^1 from the DCR assumption**
- **Applications include:**
  - First construction of sublinear, two-round secure computation from DCR
  - Better communication in secure multi-party computation
  - Non-interactive attribute based key exchange in the standard model

First construction of these applications without using spooky encryption

Can we go further?

What can we dream of?

x 

Alice

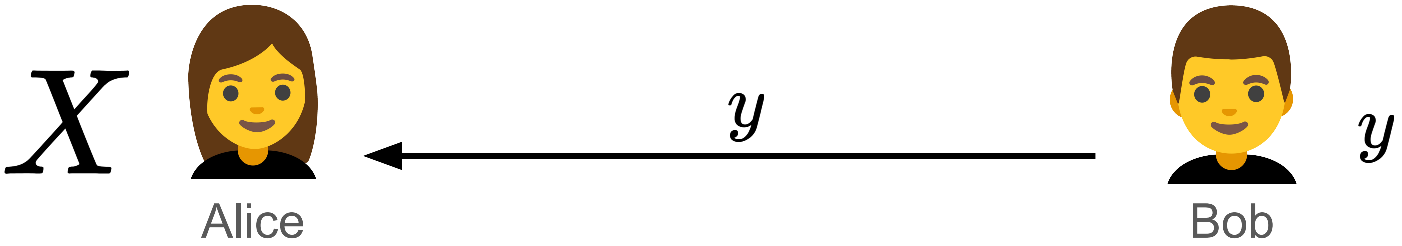
 y

Bob

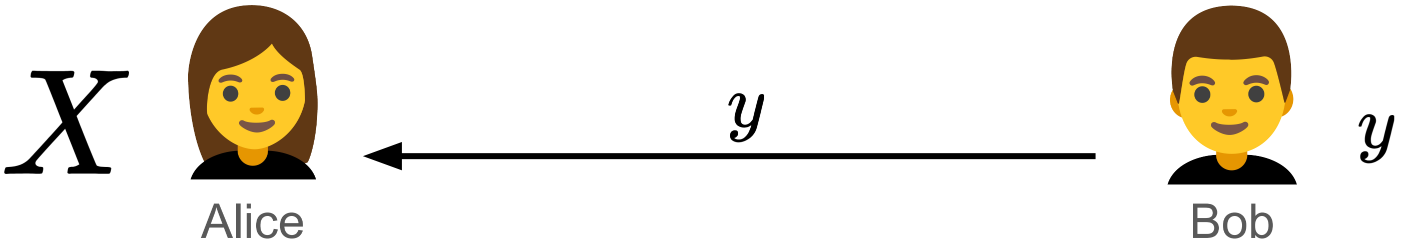
What can we dream of?

What can we dream of?

What can we dream of?



What can we dream of?

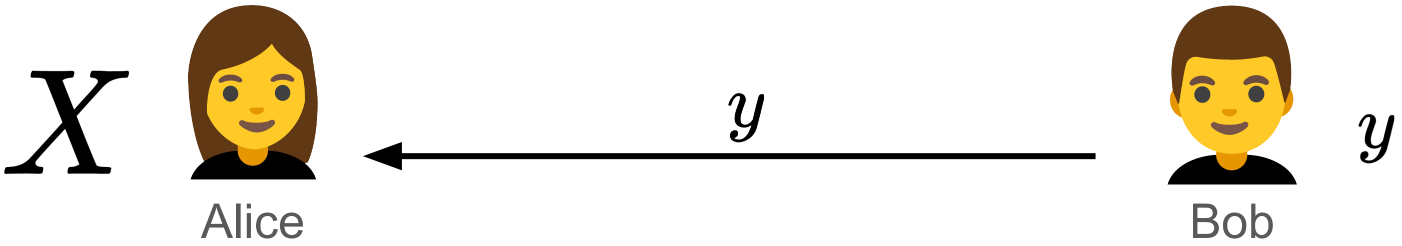


$$f(X, y)$$

What can we dream of?

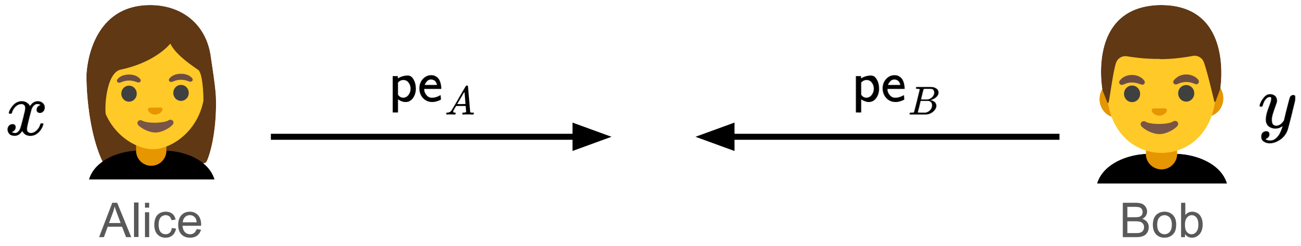
Alice learns y ☹

Optimal communication $|y|$ ☺

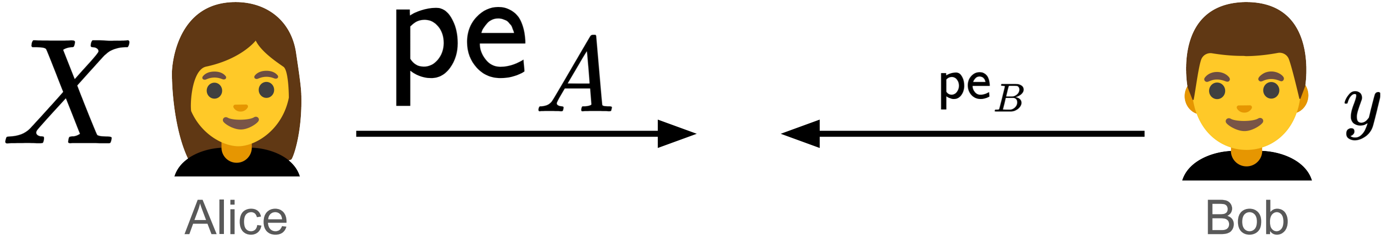


$f\left(X, y\right)$

Use Multi-Key Homomorphic Secret Sharing?

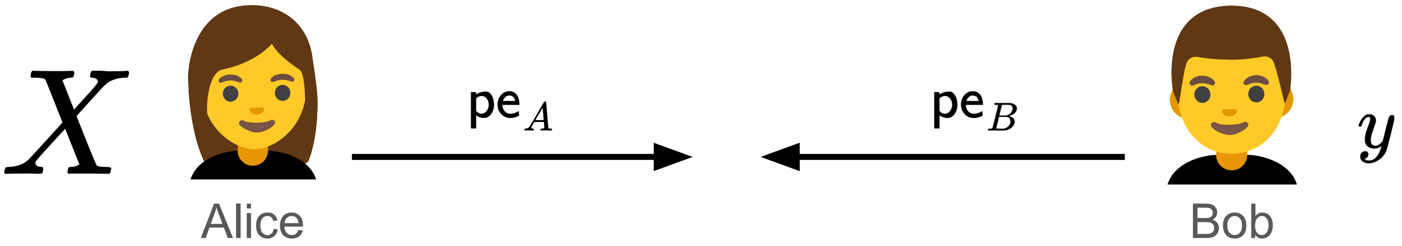


Use Multi-Key Homomorphic Secret Sharing?



Can we get a "*fully succinct*" protocol?

$$|\mathsf{pe}_\sigma| \leq (|X|^\epsilon + |f(X, y)|^\epsilon) \text{ for all } \sigma \in \{A, B\}$$



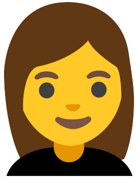
$$(\mathsf{pe}_A, \mathsf{st}_A) \leftarrow \mathsf{Encode}_A(f, X)$$

$$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(f, y)$$

Can we get a "*fully succinct*" protocol?

$$|\mathsf{pe}_\sigma| \leq (|X|^\epsilon + |f(X,y)|^\epsilon) \text{ for all } \sigma \in \{A, B\}$$

"Input and Output succinctness"

X  Alice $\xrightarrow{\mathsf{pe}_A}$ $\xleftarrow{\mathsf{pe}_B}$  Bob y

$(\mathsf{pe}_A, \mathsf{st}_A) \leftarrow \mathsf{Encode}_A(f, X)$

$(\mathsf{pe}_B, \mathsf{st}_B) \leftarrow \mathsf{Encode}_B(f, y)$

Simultaneous-**M**essage and **S**uccinct (SMS) Secure Computation

Joint work with
Elette Boyle, Abhishek Jain, and Akshay Srinivasan

The “magic” scheme

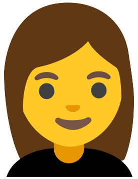
X 👩 Alice

👨 Bob y

The “magic” scheme

$$\text{Hash}\left(X\right) \rightarrow \mathsf{pe}_A$$

X



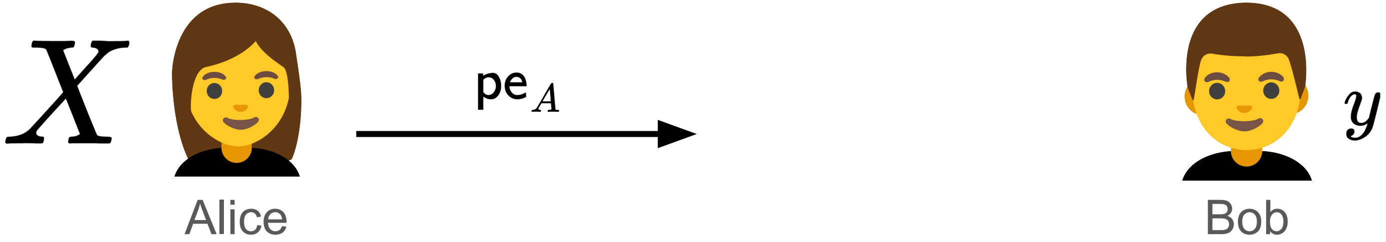
Alice

y

Bob

The “magic” scheme

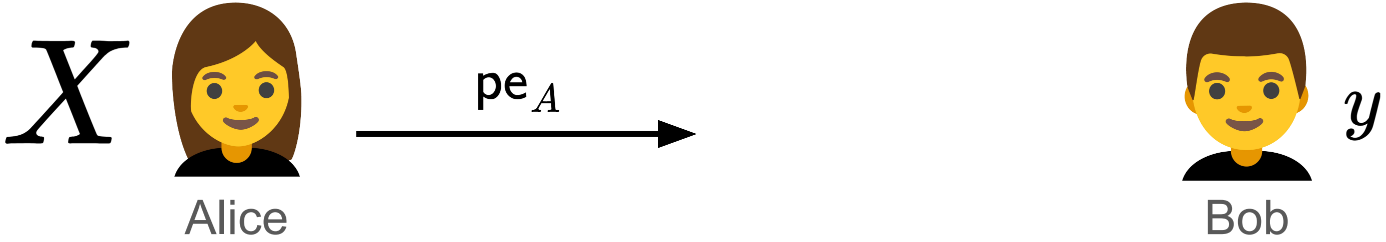
$$\text{Hash}(X) \rightarrow \text{pe}_A$$



The "magic" scheme

$\mathsf{ct}_y := \mathsf{Encrypt}($🔑$, y)$

$\mathsf{Hash}(X) \rightarrow \mathsf{pe}_A$

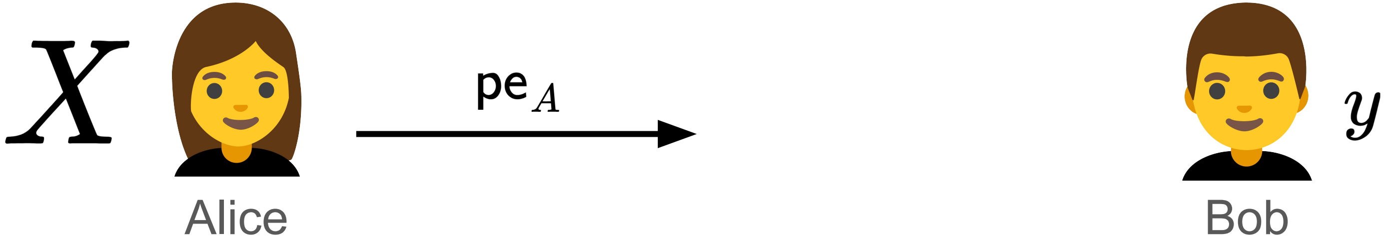


The "magic" scheme

$\mathsf{Hash}(X) \to \mathsf{pe}_A$

$\mathsf{ct}_y := \mathsf{Encrypt}($🔑$, y)$

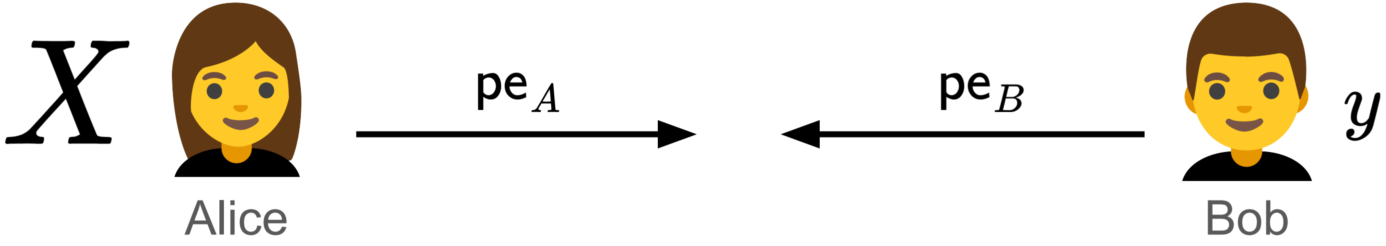
$\mathsf{ct}_{🔑} := \mathsf{Encrypt}($🗝$,$ 🔑$)$



The "magic" scheme

$\mathsf{Hash}(X) \to \mathsf{pe}_A$

ct_y := Encrypt(🔑, y)
$\mathsf{ct}_{🔑}$:= Encrypt(🗝, 🔑)
} pe_B



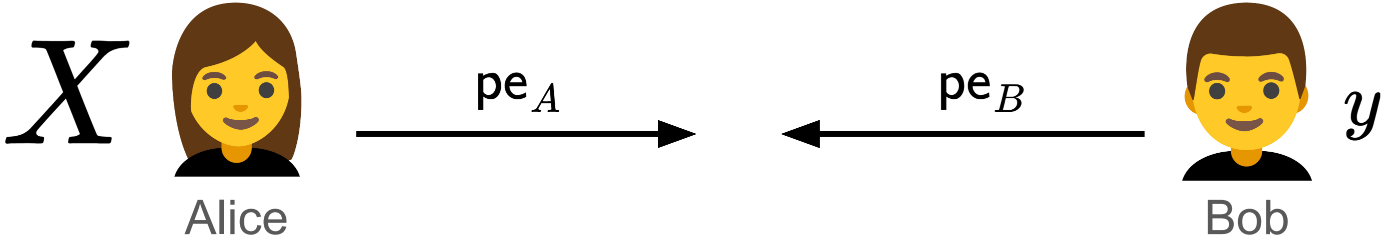
The "magic" scheme

$\mathsf{Hash}(X) \to \mathsf{pe}_A$

$\mathsf{ct}_y := \mathsf{Encrypt}($🔑$, y)$

$\mathsf{ct}_{🔑} := \mathsf{Encrypt}($🗝$,$ 🔑$)$

$\Big\}\ \mathsf{pe}_B$



$\mathsf{ct} \leftarrow \mathsf{Eval}(f, X, \mathsf{ct}_y)$

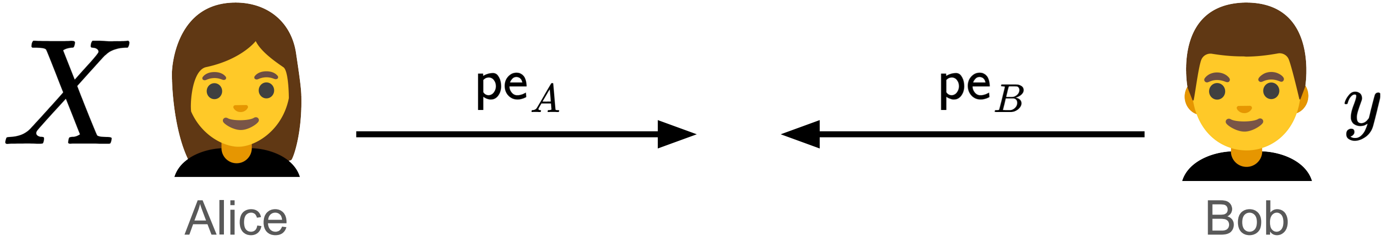
The "magic" scheme

$\mathsf{Hash}(X) \rightarrow \mathsf{pe}_A$

$\mathsf{ct}_y := \mathsf{Encrypt}(🔑, y)$

$\mathsf{ct}_{🔑} := \mathsf{Encrypt}(🗝, 🔑)$

$\Big\}\ \mathsf{pe}_B$



$\mathsf{ct} \leftarrow \mathsf{Eval}(f, X, \mathsf{ct}_y)$

$z_A \leftarrow \mathsf{Magic}(\mathsf{ct}_{🔑}, \mathsf{ct})$

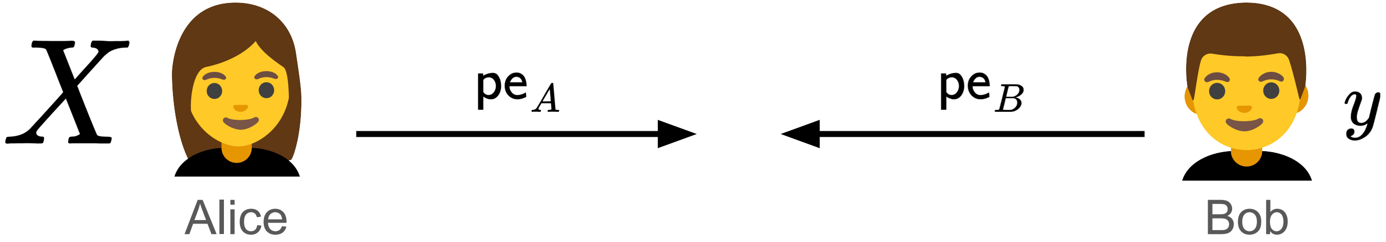
The "magic" scheme

$\mathsf{Hash}(X) \to \mathsf{pe}_A$

$\mathsf{ct}_y := \mathsf{Encrypt}(🔑, y)$

$\mathsf{ct}_{🔑} := \mathsf{Encrypt}(🗝️, 🔑)$

} pe_B



$\mathsf{ct} \leftarrow \mathsf{Eval}(f, X, \mathsf{ct}_y)$

$z_B \leftarrow \mathsf{Magic}(\mathsf{pe}_A, 🗝️)$

$z_A \leftarrow \mathsf{Magic}(\mathsf{ct}_{🔑}, \mathsf{ct})$

Preliminaries

Ingredient I: FHE from LWE with “nice” decryption

Ingredient I: FHE from LWE with "nice" decryption

$$\mathsf{FHE.KeyGen}\left(1^{\lambda}\right) : \mathsf{sk} \overset{R}{\leftarrow} \left(\mathbb{Z}_q^{n-1}, 1\right)$$

Ingredient I: FHE from LWE with "nice" decryption

$$\mathsf{FHE.KeyGen}\left(1^{\lambda}\right) : \mathsf{sk} \overset{R}{\leftarrow} \left(\mathbb{Z}_q^{n-1}, 1\right)$$

$$\mathsf{FHE.Encrypt}\left(\mathsf{sk}, x\right) : \left(-\mathbf{a}, \ \langle \mathbf{a}, \mathsf{sk} \rangle + \frac{q}{p} x + \mathsf{noise}\right)$$

Ingredient I: FHE from LWE with "nice" decryption

$$\mathsf{FHE.KeyGen}\left(1^{\lambda}\right) : \mathsf{sk} \xleftarrow{R} \left(\mathbb{Z}_q^{n-1}, 1\right)$$

$$\mathsf{FHE.Encrypt}\left(\mathsf{sk},\ x\right) : \left(-\mathbf{a},\ \langle \mathbf{a},\ \mathsf{sk}\rangle + \frac{q}{p}x + \mathsf{noise}\right)$$

$$\mathsf{FHE.Decrypt}\left(\mathsf{sk},\ \mathsf{ct}\right) : \lceil \langle \mathsf{ct}, \mathsf{sk}\rangle \rfloor_p$$

Ingredient I: FHE from LWE with "nice" decryption

$$\mathsf{FHE.KeyGen}\left(1^{\lambda}\right) : \mathsf{sk} \overset{R}{\leftarrow} \left(\mathbb{Z}_q^{n-1}, 1\right)$$

$$\mathsf{FHE.Encrypt}\left(\mathsf{sk},\ x\right) : \ \left(-\mathbf{a},\ \langle \mathbf{a}, \mathsf{sk}\rangle + \frac{q}{p}x + \mathsf{noise}\right)$$

$$\mathsf{FHE.Decrypt}\left(\mathsf{sk},\ \mathsf{ct}\right) : \ \lceil \langle \mathsf{ct}, \mathsf{sk} \rangle \rfloor_p$$

$$\langle \mathsf{ct}, \mathsf{sk}\rangle \ = \ \frac{q}{p}x + \mathsf{noise}$$

Ingredient I: FHE from LWE with “nice” decryption

$$\text{FHE.KeyGen}\left(1^\lambda\right) : \text{sk} \xleftarrow{R} \left(\mathbb{Z}_q^{n-1}, 1\right)$$

$$\text{FHE.Encrypt}\left(\text{sk}, x\right) : \left(-\mathbf{a}, \ \langle \mathbf{a}, \text{sk} \rangle + \frac{q}{p} x + \text{noise}\right)$$

$$\text{FHE.Decrypt}\left(\text{sk}, \ \text{ct}\right) : \ \lceil \langle \text{ct}, \text{sk} \rangle \rfloor_p$$

$$\boxed{\langle \text{ct}, \text{sk} \rangle = \frac{q}{p} x + \text{noise}}$$

“Near linear decryption”

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

$\textsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\textsf{crs}, C) \to \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \to \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $\left(\textsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}\right) \to \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

$\textsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\textsf{crs}, C) \rightarrow \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $(\textsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \rightarrow \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

  $$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

  $$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

$\text{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\text{crs}, C) \to \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \to \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $(\text{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \to \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

  $$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

  $$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

$\textsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\textsf{crs}, C) \rightarrow \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $(\textsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \rightarrow \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

  $$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

  $$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

crs $= (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\mathsf{crs}, C) \rightarrow \mathbf{A}_C$.
  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$
  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \rightarrow \mathbf{w}_C$
  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

crs $= (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK (crs , C) $\rightarrow \mathbf{A}_C$.
  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$
  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT (crs , $\mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \rightarrow \mathbf{w}_C$
  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G} \right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

crs $= (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK (crs , C) $\rightarrow \mathbf{A}_C$.
  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$
  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT (crs , $\mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}$) $\rightarrow \mathbf{w}_C$
  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G} \right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

crs $= (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK (crs , C) $\rightarrow \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \rightarrow \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT (crs , $\mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}$) $\rightarrow \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$
$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

LWE ciphertexts encrypted with key

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

Ingredient II: GVW Evaluation Algorithms

Building blocks from [GVW'15]:

$\text{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

common **random** string

- EvalPK $(\text{crs}, C) \to \mathbf{A}_C$.

  Input: CRS and a circuit $C : \{0,1\}^\alpha \to \mathbb{Z}_q^\beta$

  Output: a public matrix $\mathbf{A}_C \in \mathbb{Z}_q^{n \times k}$

- EvalCT $(\text{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \hat{a}) \to \mathbf{w}_C$

  Input: CRS, $\alpha + \beta$ ciphertexts, the circuit C and public input $\hat{a}$ where:

$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \hat{a}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \hat{\mathbf{b}}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$$

  Output: a ciphertext $\mathbf{w}_C = \mathbf{s}^\top \left(\mathbf{A}_C + \left\langle C(\hat{a}), \hat{\mathbf{b}} \right\rangle \cdot \mathbf{G}\right) + \text{noise}$

SMS Secure Computation

SMS Secure Computation

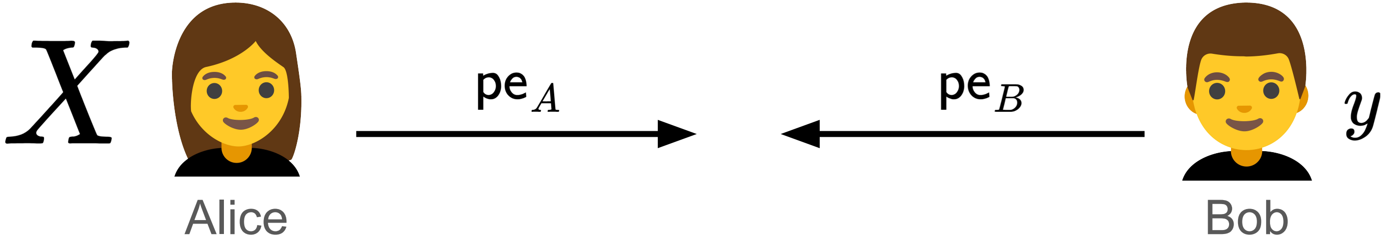
Getting input succinctness

SMS Secure Computation
Getting input succinctness

Output succinctness will come later

Building SMS with Input Succinctness

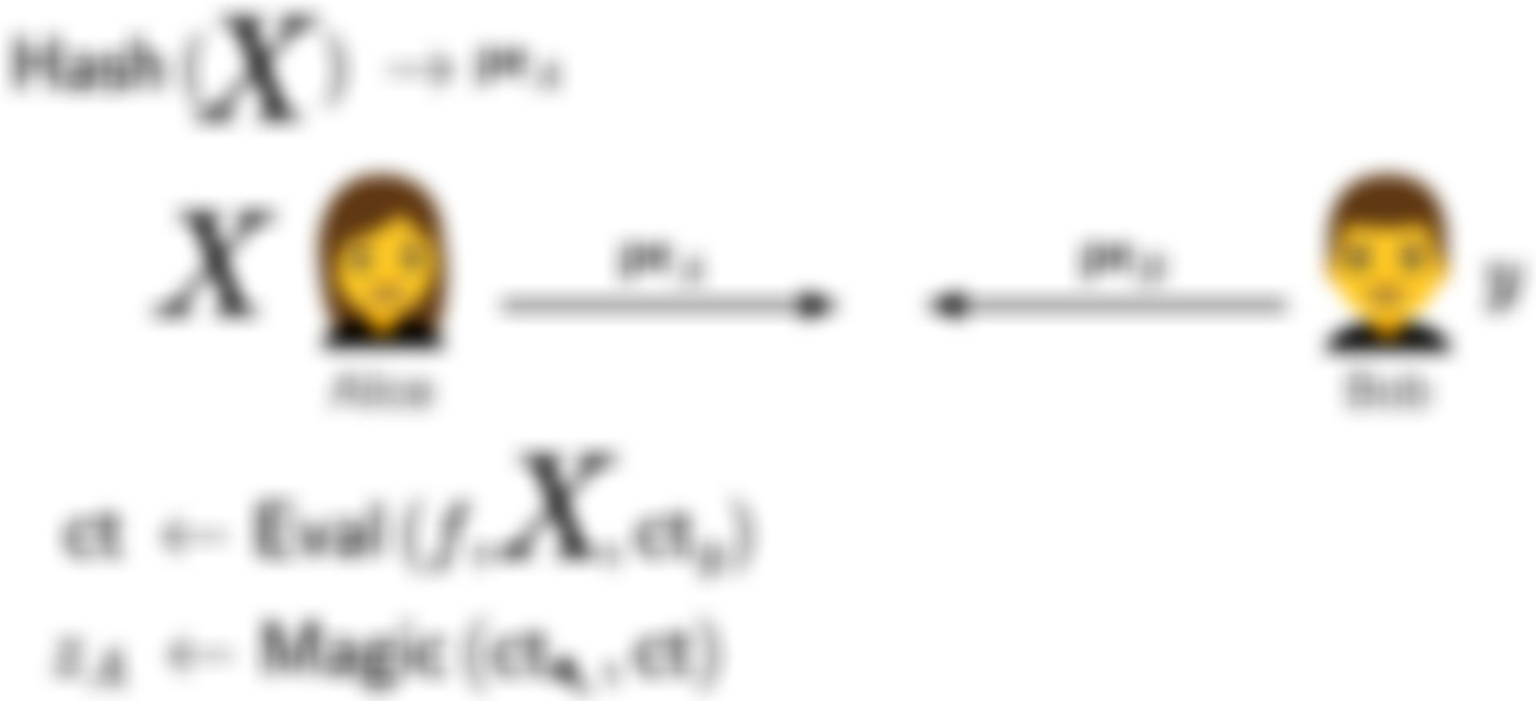
$$\text{Hash}(X) \rightarrow \text{pe}_A$$



$$\text{ct} \leftarrow \text{Eval}(f, X, \text{ct}_y)$$

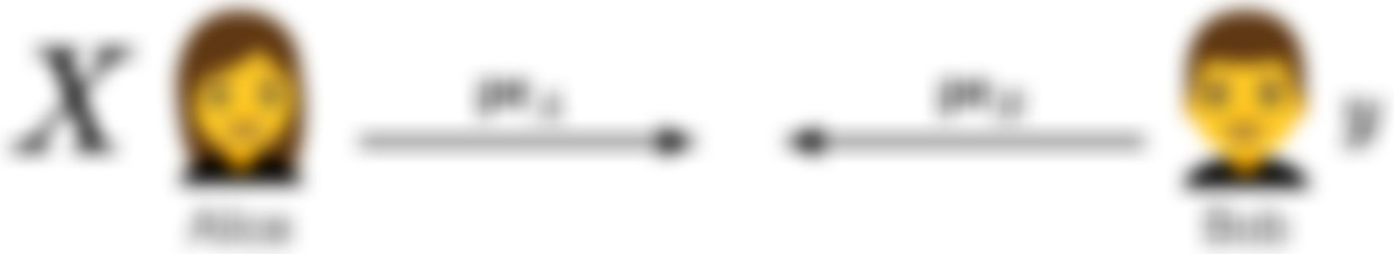
$$z_A \leftarrow \text{Magic}(\text{ct}_{🔑}, \text{ct})$$

Building SMS with Input Succinctness



Building SMS with Input Succinctness

$\mathsf{EvalPK}\,(X) \rightarrow \mathsf{pe}_A$



$\mathsf{ct} \leftarrow \mathsf{EvalCT}\,(f, X, \mathsf{ct}_y)$

Building SMS with Input Succinctness

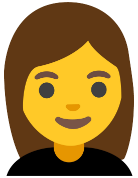
$f : \{0,1\}^{\mathsf{BIG}} \times \{0,1\}^{\mathsf{small}} \to \{0,1\}$

Building SMS with Input Succinctness

$f : \{0,1\}^{\text{BIG}} \times \{0,1\}^{\text{small}} \to \{0,1\}$

Building SMS with Input Succinctness

X



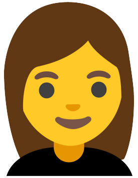
Alice

$f : \{0,1\}^{\mathsf{BIG}} \times \{0,1\}^{\mathsf{small}} \to \{0,1\}$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct
and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

X

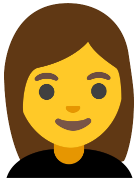


$f : \{0,1\}^{\mathsf{BIG}} \times \{0,1\}^{\mathsf{small}} \rightarrow \{0,1\}$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X 

Alice

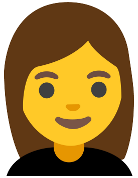
$f : \{0,1\}^{\mathsf{BIG}} \times \{0,1\}^{\mathsf{small}} \to \{0,1\}$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X



Alice

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$f : \{0,1\}^{\text{BIG}} \times \{0,1\}^{\text{small}} \rightarrow \{0,1\}$

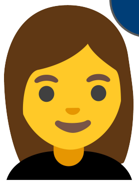
Building SMS with Input Succinctness

$|\mathbf{A}_C| = \text{poly}(\text{depth}(C), \lambda)$

"It's very small"

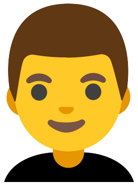
$\text{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X



Alice

$\mathbf{A}_C$ →



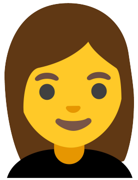
Bob

y

$\mathbf{A}_C \leftarrow \text{EvalPK}(\text{crs}, C)$

Remark: EvalPK does not guarantee hiding of the circuit *C*, so $\mathbf{A}_C$ may leak something about Alice's input. We resolve this using the transformation of Quach et al. [QWW'18].

$$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$$

X  $\xrightarrow{\mathbf{A}_C}$  y

Alice   Bob

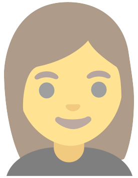
$$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

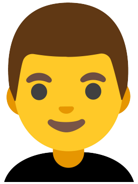
$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X



Alice

$\mathbf{A}_C$ →



Bob

y

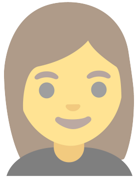
$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X  $\xrightarrow{\mathbf{A}_C}$

Alice

 y

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

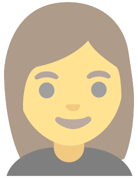
$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes FHE. Eval $(f,\ X,\ \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X  $\xrightarrow{\mathbf{A}_C}$

Alice

 y

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

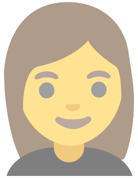
$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$

$\mathbf{s} \leftarrow (1, \mathsf{random}) \in \mathbb{Z}_q^n$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X  $\xrightarrow{\mathbf{A}_C}$

Alice

 y

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$

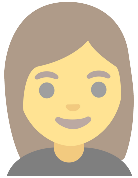
$\mathbf{s} \leftarrow (1, \mathsf{random}) \in \mathbb{Z}_q^n$

$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \mathsf{ct}[i] \cdot \mathbf{G} + \mathsf{noise}, \text{ for all } i \in [\alpha]$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes FHE. Eval (f, X, ct)

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X  $\xrightarrow{\mathbf{A}_C}$

Alice

 y

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$

$\mathbf{s} \leftarrow (1, \mathsf{random}) \in \mathbb{Z}_q^n$

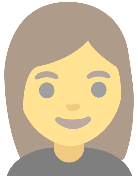
$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \mathsf{ct}[i] \cdot \mathbf{G} + \mathsf{noise}, \text{ for all } i \in [\alpha]$

$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \mathsf{sk}[i] \cdot \mathbf{G} + \mathsf{noise}, \text{ for all } i \in [\beta]$

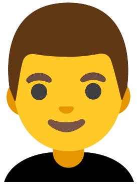
Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes FHE. Eval (f, X, ct)

$$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$$

X  $\xrightarrow{\mathbf{A}_C}$

Alice

 y

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$$

$$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$$

$$\mathbf{s} \leftarrow (1, \mathsf{random}) \in \mathbb{Z}_q^n$$

Nested encryption of y →

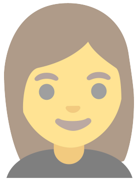
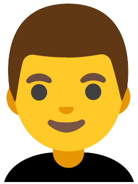
$$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \mathsf{ct}[i] \cdot \mathbf{G} + \mathsf{noise}, \quad \text{for all } i \in [\alpha]$$

$$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \mathsf{sk}[i] \cdot \mathbf{G} + \mathsf{noise}, \quad \text{for all } i \in [\beta]$$

Building SMS with Input Succinctness

C takes as input an FHE ciphertext ct and computes $\mathsf{FHE.Eval}(f, X, \mathsf{ct})$

$\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\alpha, \mathbf{B}_1, \ldots \mathbf{B}_\beta)$

X  $\xrightarrow{\mathbf{A}_C}$  y

Alice

Bob

$\mathbf{A}_C \leftarrow \mathsf{EvalPK}(\mathsf{crs}, C)$

$\mathsf{sk} \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$

$\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{sk}, y)$

$\mathbf{s} \leftarrow (1, \mathsf{random}) \in \mathbb{Z}_q^n$

Encryption of sk

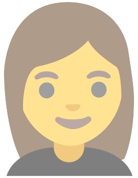
$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \mathsf{ct}[i] \cdot \mathbf{G} + \mathsf{noise}, \text{ for all } i \in [\alpha]$

$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \mathsf{sk}[i] \cdot \mathbf{G} + \mathsf{noise}, \text{ for all } i \in [\beta]$

Building SMS with Input S

C takes as input an FHE ciphertext ct and computes FHE.Eval (f, X, ct)

Size poly $(|y|, \lambda)$
"Independent of $|X|$"

X  $\xrightarrow{\mathbf{A}_C}$ $\xleftarrow{(\text{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)}$  y

Alice

Bob

$\mathbf{A}_C \leftarrow \text{EvalPK}(\text{crs}, C)$

$\text{sk} \leftarrow \text{FHE.KeyGen}(1^\lambda)$

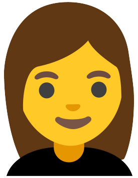
$\text{ct} \leftarrow \text{FHE.Enc}(\text{sk}, y)$

$\mathbf{s} \leftarrow (1, \text{random}) \in \mathbb{Z}_q^n$

$\mathbf{u}_i = \mathbf{s}^\top \mathbf{A}_i + \text{ct}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\alpha]$

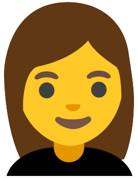
$\mathbf{v}_i = \mathbf{s}^\top \mathbf{B}_i + \text{sk}[i] \cdot \mathbf{G} + \text{noise}, \quad \text{for all } i \in [\beta]$

Building SMS with Input Succinctness



Alice

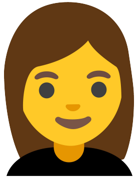
Building SMS with Input Succinctness



Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

Building SMS with Input Succinctness

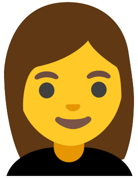


Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

Building SMS with Input Succinctness



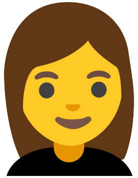
Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

$$\mathbf{w}_C[1] = \mathbf{s}^\top (\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk}\rangle \cdot \mathbf{G})[1] + \mathsf{noise}$$ // correctness of EvalCT

Building SMS with Input Succinctness



Alice

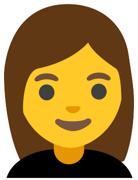
$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

$$\mathbf{w}_C[1] = \mathbf{s}^\top(\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk}\rangle \cdot \mathbf{G})[1] + \mathsf{noise}$$ // correctness of EvalCT

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle C(\mathsf{ct}), \mathsf{sk}\rangle + \mathsf{noise}$$ // because $\mathbf{s}[1] = 1$

Building SMS wit



Alice

$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha$

$$\mathbf{G} = \begin{bmatrix} 1 & 2 & \ldots & 0 \\ \mathbf{0} & & & \\ \vdots & & & \\ \mathbf{0} & & & \end{bmatrix}$$

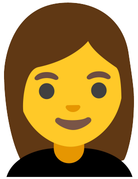
$$(\mathbf{s}^\top \mathbf{G})[1] = \langle \mathbf{s}, (1, 0, \ldots, 0) \rangle = 1$$

$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots,$

$\mathbf{w}_C[1] = \mathbf{s}^\top (\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk} \rangle \cdot \mathbf{G})[1] + \mathsf{noise}$ // correctness of EvalCT

$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle C(\mathsf{ct}), \mathsf{sk} \rangle + \mathsf{noise}$ // because $\mathbf{s}[1] = 1$

Building SMS with Input Succinctness



Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

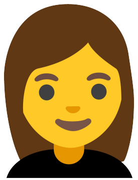
$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

$$\mathbf{w}_C[1] = \mathbf{s}^\top(\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk}\rangle \cdot \mathbf{G})[1] + \mathsf{noise}$$ // correctness of EvalCT

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle C(\mathsf{ct}), \mathsf{sk}\rangle + \mathsf{noise}$$ // because $\mathbf{s}[1] = 1$

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle \mathsf{FHE.Eval}(f, (X, \mathsf{ct})), \mathsf{sk}\rangle + \mathsf{noise}$$

Building SMS with Input Succinctness



Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

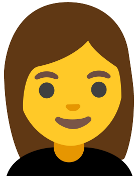
$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

$$\mathbf{w}_C[1] = \mathbf{s}^\top (\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk}\rangle \cdot \mathbf{G})[1] + \text{noise} \quad \text{// correctness of EvalCT}$$

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle C(\mathsf{ct}), \mathsf{sk}\rangle + \text{noise} \quad \text{// because } \mathbf{s}[1] = 1$$

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle \mathsf{FHE.Encrypt}(\mathsf{sk}, f(X, y)), \mathsf{sk}\rangle + \text{noise} \quad \text{// correctness}$$

Building SMS with Input Succinctness



Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

$$\mathbf{w}_C \leftarrow \mathsf{EvalCT}(\mathsf{crs}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta, C, \mathsf{ct})$$

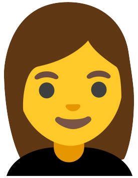
$$\mathbf{w}_C[1] = \mathbf{s}^\top (\mathbf{A}_C + \langle C(\mathsf{ct}), \mathsf{sk}\rangle \cdot \mathbf{G})[1] + \text{noise}$$ // correctness of EvalCT

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle C(\mathsf{ct}), \mathsf{sk}\rangle + \text{noise}$$ // because $\mathbf{s}[1] = 1$

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \langle \mathsf{FHE.Encrypt}(\mathsf{sk}, f(X, y)), \mathsf{sk}\rangle + \text{noise}$$ // correctness

$$= \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \text{noise}$$ // near-linear decryption of FHE

Building SMS with Input Succinctness

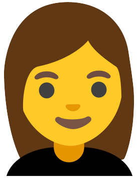


Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

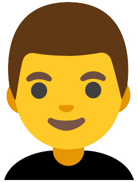
$$z_A := \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \text{noise}$$

Building SMS with Input Succinctness



Alice

$$\left(\mathsf{ct}\,,\ \mathbf{u}_1,\ \ldots,\ \mathbf{u}_\alpha,\ \mathbf{v}_1,\ \ldots,\ \mathbf{v}_\beta\right)$$



Bob

$$\mathbf{s} := (1, \mathsf{random})$$

$$z_A := \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \mathsf{noise}$$

Building SMS with Input Succinctness

Alice

$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$

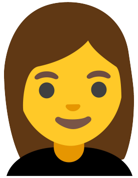
$\mathbf{A}_C$

Bob

$\mathbf{s} := (1, \mathsf{random})$

$$z_A := \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \mathsf{noise}$$

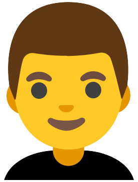
Building SMS with Input Succinctness



Alice

$$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$$

$\mathbf{A}_C$



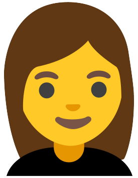
Bob

$\mathbf{s} := (1, \mathsf{random})$

$$z_A := \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \mathsf{noise}$$

$$z_B := -\left(\mathbf{s}^\top \mathbf{A}_C\right)[1]$$

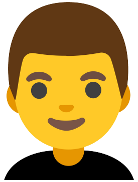
Building SMS with Input Succinctness



Alice

$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$

$\mathbf{A}_C$



Bob

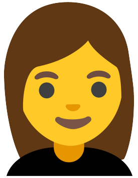
$\mathbf{s} := (1, \mathsf{random})$

$z_A := \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \mathsf{noise}$

$z_B := -(\mathbf{s}^\top \mathbf{A}_C)[1]$

$$z_A + z_B = \frac{q}{p} f(X, y) + \mathsf{noise}$$

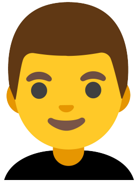
Building SMS with Input Succinctness



Alice

$(\mathsf{ct}, \mathbf{u}_1, \ldots, \mathbf{u}_\alpha, \mathbf{v}_1, \ldots, \mathbf{v}_\beta)$

$\mathbf{A}_C$

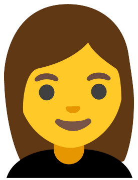


Bob

$\mathbf{s} := (1, \mathsf{random})$

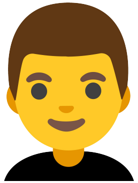
$$z_A := \lceil \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \mathsf{noise} \rfloor_p \qquad z_B := -\lceil (\mathbf{s}^\top \mathbf{A}_C)[1] \rfloor_p$$

Building SMS with Input Succinctness



Alice

Lemma (Rounding of Noisy Shares): Assuming LWE with *superpolynomial modulus-to-noise ratio*, rounding of two noisy shares results in additive shares.



Bob

$$\mathbf{s} := (1, \mathsf{random})$$

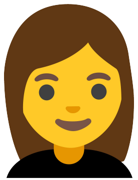
$$z_A := \lceil \mathbf{s}^\top \mathbf{A}_C[1] + \tfrac{q}{p} f(X, y) + \mathsf{noise} \rfloor_p$$

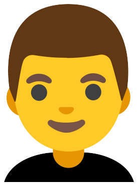
$$= \mathbf{s}^\top \mathbf{A}_C[1] + f(X, y) \pmod{p}$$

$$z_B := -\lceil (\mathbf{s}^\top \mathbf{A}_C)[1] \rfloor_p$$

$$= -(\mathbf{s}^\top \mathbf{A}_C)[1] \pmod{p}$$

Building SMS with Input Succinctness


Alice


Bob

$\mathbf{s} := (1, \text{random})$

$$z_A := \left\lceil \mathbf{s}^\top \mathbf{A}_C[1] + \frac{q}{p} f(X, y) + \text{noise} \right\rfloor_p$$
$$= \mathbf{s}^\top \mathbf{A}_C[1] + f(X, y) \pmod{p}$$

$$z_B := -\left\lceil \left(\mathbf{s}^\top \mathbf{A}_C\right)[1] \right\rfloor_p$$
$$= -\left(\mathbf{s}^\top \mathbf{A}_C\right)[1] \pmod{p}$$

$$z_A + z_B = f(X, y)$$

Long outputs?

Long outputs?

Too long to explain;
Short answer: Use SMS for vector OLE [ARS'24]

Applications of SMS

SMS Secure Computation

Direct applications to

SMS Secure Computation

Direct applications to

1. First construction of trapdoor hashing beyond linear predicates

SMS Secure Computation

Direct applications to

1. First construction of trapdoor hashing beyond linear predicates
2. Generic compiler to correlation-intractable hash functions

SMS Secure Computation

Direct applications to

1. First construction of trapdoor hashing beyond linear predicates
2. Generic compiler to correlation-intractable hash functions
3. Generic compiler to rate-1 fully-homomorphic encryption

SMS Secure Computation

Direct applications to

1. First construction of trapdoor hashing beyond linear predicates
2. Generic compiler to correlation-intractable hash functions
3. Generic compiler to rate-1 fully-homomorphic encryption
4. Hubacek–Wichs [HW'15]-style succinct secure computation (from our iO-based construction of SMS)

Conclusion

This thesis: A toolbox for secure computation

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**
- **New theory connecting**

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**
- **New theory connecting**
    - Rate-1 FHE, succinct computation

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**
- **New theory connecting**
    - Rate-1 FHE, succinct computation
    - Trapdoor and correlation-intractable hash functions

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**
- **New theory connecting**
  - Rate-1 FHE, succinct computation
  - Trapdoor and correlation-intractable hash functions
  - Output-succinct secure computation

This thesis: A toolbox for secure computation

- **New constructions of succinct, two-round secure computation**
- **New constructions of constrained PRFs + implementations**
- **New constructions non-interactive OT extension + implementations**
- **New theory connecting**
  - Rate-1 FHE, succinct computation
  - Trapdoor and correlation-intractable hash functions
  - Output-succinct secure computation
- **and more…**

So Long, and Thanks for All the Fish!

— Douglas Adams

References

[Yao'86]: A. C. Yao. "How to Generate and Exchange Secrets."

[BM'89]: M. Bellare and S. Micali. "Non-Interactive Oblivious Transfer and Applications."

[GKM+'00]: Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. "The Relationship between Public Key Encryption and Oblivious Transfer."

[Gen'09]: C. Gentry. "Fully Homomorphic Encryption Using Ideal Lattices."

[HLP'11]: S. Halevi, Y. Lindell, and B. Pinkas. "Secure Computation on the Web: Computing without Simultaneous Interaction."

[BGG+'14]: D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. "Fully Key-Homomorphic Encryption, Arithmetic Circuit ABE, and Compact Garbled Circuits."

[GVW'15]: S. Gorbunov, V. Vaikuntanathan, and H. Wee. "Predicate Encryption for Circuits from LWE."

References

[HW'15]: P. Hubacek and D. Wichs. "On the Communication Complexity of Secure Function Evaluation with Long Output."

[BGI'16]: E. Boyle, N. Gilboa, and Y. Ishai. "Breaking the Circuit Size Barrier for Secure Computation Under DDH."

[DHRW'16]: D. Dodis, S. Halevi, R. D. Rothblum, and D. Wichs. "Spooky Encryption and Its Applications."

[QWW'18]: W. Quach, H. Wee, and D. Wichs. "Laconic Function Evaluation and Applications."

[OSY'21]: C. Orlandi, P. Scholl, and S. Yakoubov. "The Rise of Paillier: Homomorphic Secret Sharing and Public-Key Silent OT."

[CZ'22]: G. Couteau and M. Zarezadeh. "Non-Interactive Secure Computation of Inner-Product from LPN and LWE."

[ARS'24]: D. Abram, L. Roy, and P. Scholl. "Succinct Homomorphic Secret Sharing."

References

[BCMPR'24]: D. Bui, G. Couteau, P. Meyer, A. Passelègue, and M. Riahinia. "Fast Public-Key Silent OT and More from Constrained Naor-Reingold."

[CDDKSS'24]: G. Couteau, L. Devadas, S. Devadas, A. Koch, and S. Servan-Schreiber. "QuietOT: Lightweight Oblivious Transfer with a Public-Key Setup."

[SS'24]: S. Servan-Schreiber. "Constrained PRFs for Inner-Product Predicates from Weaker Assumptions."

[CDHJSS'25]: G. Couteau, L. Devadas, A. Hegde, A. Jain, and S. Servan-Schreiber. "Multi-Key Homomorphic Secret Sharing."

[BDSS'25]: E. Boyle, L. Devadas, and S. Servan-Schreiber. "Non-Interactive Distributed Point Functions."

[BJSSS'25]: E. Boyle, A. Jain, S. Servan-Schreiber, and A. Srinivasan. "Simultaneous-Message and Succinct Secure Computation."