Fully-Secure MPC with Minimal Trust

Yuval Ishai, Arpita Patra, Sikhar Patranabis, Divya Ravi, Akshayaram Srinivasan

TCC 2022
Secure Multi-Party Computation

- **Setup**: n parties \( \{P_1, P_2, \ldots, P_n\} \); t corrupt

- **Input**: \( P_i \) has input \( x_i \)

- **Goal**: Compute \( f(x_1, x_2, x_3) \)
Secure Multi-Party Computation

- **Setup**: \( n \) parties \( \{P_1, P_2, \ldots, P_n\} \); \( t \) corrupt

- **Input**: \( P_i \) has input \( x_i \)

- **Goal**: Compute \( f(x_1, x_2, x_3) \)

- **Properties**:
  - **Correctness**: Protocol output = \( f(x_1, x_2, x_3) \)
  - **Privacy**: Nothing beyond function output revealed
Motivation
Motivation

Fairness and full-security (G.O.D)
Motivation

Fairness and full-security (G.O.D)

If adversary gets output, everyone does
Motivation

Fairness and full-security (G.O.D)

- If adversary gets output, everyone does
- Adversary cannot prevent honest parties from getting output
Motivation

Fairness and full-security (G.O.D) : impossible in dishonest majority [Cleve86]

- If adversary gets output, everyone does
- Adversary cannot prevent honest parties from getting output
Motivation

Fairness and full-security (G.O.D) : impossible in dishonest majority [Cleve86]

If adversary gets output, everyone does

Adversary cannot prevent honest parties from getting output

How to bypass this impossibility ?
Motivation

Fairness and full-security (G.O.D) : impossible in dishonest majority [Cleve86]

If adversary gets output, everyone does
Adversary cannot prevent honest parties from getting output

How to bypass this impossibility ?

Honest Majority
Motivation

Fairness and full-security (G.O.D): impossible in dishonest majority [Cleve86]

If adversary gets output, everyone does

Adversary cannot prevent honest parties from getting output

How to bypass this impossibility?

Honest Majority

Use external help
Motivation

Fairness and full-security (G.O.D) : impossible in dishonest majority [Cleve86]

- If adversary gets output, everyone does
- Adversary cannot prevent honest parties from getting output

How to bypass this impossibility?

- Honest Majority
- Use external help
Motivation

Fairness and full-security (G.O.D): impossible in dishonest majority [Cleve86]

- If adversary gets output, everyone does
- Adversary cannot prevent honest parties from getting output

How to bypass this impossibility?

- Honest Majority
- Use external help
Modelling the Trusted Party (TP)
Modelling the Trusted Party (TP)

What does the TP do?
Modelling the Trusted Party (TP)

What does the TP do?

Rule number two, I can’t make anybody fall in love with anybody else.
What does the TP do?

How many times do you use the TP?

Rule number two, I can’t make anybody fall in love with anybody else.
Modelling the Trusted Party (TP)

What does the TP do?

Rule number two, I can’t make anybody fall in love with anybody else.

How many times do you use the TP?
Modelling the Trusted Party (TP)

What does the TP do?

How many times do you use the TP?

Trivial Solution
Modelling the Trusted Party (TP)

What does the TP do?

Compute the function directly

Trivial Solution

How many times do you use the TP?
Modelling the Trusted Party (TP)

What does the TP do?

How many times do you use the TP?

Trivial Solution

Compute the function directly

One
Modelling the Trusted Party (TP)

What does the TP do? 
Compute the function directly

How many times do you use the TP? 
One

Trivial Solution
Modelling the Trusted Party (TP)

What does the TP do?

Trivial Solution

Compute the function directly

How many times do you use the TP?

One
What does the TP do?

Trivial Solution

[IOS12]

How many times do you use the TP?

One

Compute the function directly

“Small” (Independent of function)
Modelling the Trusted Party (TP)

What does the TP do?

- [ ] Compute the function directly
- [✔️] “Small” (Independent of function)

How many times do you use the TP?

- [✔️] One

Trivial Solution

[IOS12]
Modelling the Trusted Party (TP)

What does the TP do?

- **Trivial Solution**
  - Compute the function directly: ✗
  - “Small” (Independent of function): ✓

How many times do you use the TP?

- **[IOS12]**
  - One: ✓
  - n (number of parties): ✗

Rule number two, I can't make anybody fall in love with anybody else.
Modelling the Trusted Party (TP)

What does the TP do?

- What does the TP do? ✗ Compute the function directly
- [IOS12] ✓ “Small” (Independent of function)

How many times do you use the TP?

- How many times do you use the TP? ✓ One
- n (number of parties) ✗
- [IOS12] ✓ One
What does the TP do? | How many times do you use the TP?
--- | ---
Compute the function directly | One
“Small” (Independent of function) | n (number of parties)
Exponential in n | One
Modelling the Trusted Party (TP)

What does the TP do?

- Trivial Solution
  - Compute the function directly: ✗
  - "Small" (Independent of function): ✓
  - Exponential in n: ✗

- [IOS12]
  - "Small" (Independent of function): ✓
  - Exponential in n: ✗

How many times do you use the TP?

- One
- n (number of parties)

GOAL

- Small: poly(n, λ)
- One: ✓
Modelling the Trusted Party (TP)

What does the TP do?

- **Trivial Solution**
  - Compute the function directly [X]

- **[IOS12]**
  - “Small” (Independent of function) [✓]

- **[IOS12]**
  - Exponential in $n$ [X]

How many times do you use the TP?

- **One** [✓]

GOAL

- **Small** $\text{poly}(n, \lambda)$ [✓]
- **One** [✓]
GOAL

Small poly(n, λ )

One
GOAL

Small \( \text{poly}(n, \lambda) \)

One

No
Exponential-size TP is inherent if decoder is universal.
<table>
<thead>
<tr>
<th>GOAL</th>
<th>Small</th>
<th>poly(n, λ)</th>
<th>One</th>
</tr>
</thead>
</table>

No

Exponential-size TP is inherent if decoder is universal

Universal Decoder: Function-Independent Computation done to derive output from the TP response
Exponential-size TP is inherent if decoder is universal

Universal Decoder: Function-Independent Computation done to derive output from the TP response

[IOS12] is tight
Universal Decoder: Function-Independent Computation
done to derive output from the TP response
Universal Decoder: Function-Independent Computation done to derive output from the TP response.
Universal Decoder: Function-Independent Computation done to derive output from the TP response.
Universal Decoder: Function-Independent Computation done to derive output from the TP response

Still impossible with information theoretic security in plain model

Allow function-dependent decoding

Exponential-size TP is inherent if decoder is universal

Irrespective of computational assumptions or setup

[IOS12] is tight

How about computational?

How about i.t with setup?

GOAL
Small poly(n, λ )
One
GOAL Small poly(n, λ) One

Irrespective of computational assumptions or setup

Exponential-size TP is inherent if decoder is universal

How about computational?

Possibility! (Based on functional encryption)

How about i.t with setup?

Still impossible with information theoretic security in plain model

Universal Decoder: Function-Independent Computation done to derive output from the TP response

[IOS12] is tight

Allow function-dependent decoding

No
Universal Decoder: Function-Independent Computation done to derive output from the TP response

Irrespective of computational assumptions or setup

How about computational?  How about i.t with setup?

Open

Exponential-size TP is inherent if decoder is universal

[IOS12] is tight

Possible! (Based on functional encryption)

Still impossible with information theoretic security in plain model

GOAL

- Small \( \text{poly}(n, \lambda) \)
- One

No
Exponential-size TP is inherent if decoder is universal.

Irrespective of computational assumptions or setup.

Possible! (Based on functional encryption)

How about computational?

How about i.t with setup?

Small \( \text{poly}(n, \lambda) \) 

One

[IOS12] is tight

Allow function-dependent decoding

Still impossible with information theoretic security in plain model

Open
Functional Encryption (FE)
Functional Encryption (FE)

Setup

\[ mpk \]

\[ msk \]
Functional Encryption (FE)

Setup → $msk$ → Keygen

$mpk$ → $sk_f$
Functional Encryption (FE)

Setup \( \rightarrow \) Keygen

- Setup \( \rightarrow \) Keygen: \( msk \rightarrow f \)
- Keygen \( \rightarrow \) Enc: \( sk_f \)

- Input \( x \)
- Ciphertext \( c \)
Functional Encryption (FE)

Setup

<table>
<thead>
<tr>
<th>mpk</th>
</tr>
</thead>
</table>

Keygen

| msk | f |

Enc

| Input x |

Dec

| Ciphertext c |

| Output f(x) |

sk_f
Fully-Secure MPC with 1 call to small TP
Fully-Secure MPC with 1 call to small TP
MPC with identifiable abort

Either output or at least one cheater identified

Fully-Secure MPC with 1 call to small TP
Fully-Secure MPC with 1 call to small TP

Either output or at least one cheater identified

MPC with identifiable abort

$mpk, sk_f$
Fully-Secure MPC with 1 call to small TP

Either output or at least one cheater identified

MPC with identifiable abort

$mpk, sk_f$
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

$mpk, sk_f$
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

$mpk, sk_f$
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

\[ mpk, sk_f \]

\[ x_1, x_2, x_3, x'_4 \]
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

$mpk, sk_f$

$x_1, x_2, x_3, x_4$

$x_1', x_2', x_3', x_4'$
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

\[ mk, sk_f \]

\[ x_1, x_2, x_3, x_4', x_1', x_2', x_3', x_4 \]
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

\[ mpk, sk_f \]
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

$mpk, sk_f$

$P_1, P_2, P_3, P_4$

$x_1, x_2, x_3, x_4$

$x_1', x_2', x_3', x_4'$
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

\[ mpk, sk_f \]

Output: \( f(x_1, x_2, x_3, x_4') \)
Fully-Secure MPC with 1 call to small TP

MPC with identifiable abort

\[ mpk, sk_f \]

Output: \[ f(x_1, x_2, x_3, x_4') \]

Size: \[ \text{poly}(n, \lambda, d) \] (sub-exp LWE)

Size: \[ \text{poly}(n, \lambda) \] (iO)
Semi-Honest TP ?
Semi-Honest TP?

Dishonest Majority of active corruptions

AND

Semi-honest TP
Semi-Honest TP?

Colluding Model

Dishonest Majority of active corruptions
AND
Semi-honest TP
Semi-Honest TP?

Colluding Model

Dishonest Majority of active corruptions
AND
Semi-honest TP

Fairness impossible!
Semi-Honest TP?

Colluding Model:
- Dishonest Majority of active corruptions
- AND
- Semi-honest TP

Non-Colluding Model:
- Dishonest Majority of active corruptions
- OR
- Semi-honest TP

Fairness impossible!
Dishonest Majority of active corruptions
AND
Semi-honest TP

Fairness impossible!

Dishonest Majority of active corruptions
OR
Semi-honest TP

FE-Based construction works! (With tweaks)
**GOAL**

Possible! (Based on functional encryption)

Small $\text{poly}(n, \lambda)$

One

Exponential-size TP is inherent if decoder is universal

Allow function-dependent decoding

Still impossible with information theoretic security in plain model

How about computational?

How about i.t with setup?

Open

Thank you :)
GOAL

Small \( \text{poly}(n, \lambda) \)

One

- Exponential-size TP is inherent if decoder is universal
- Allow function-dependent decoding
- Still impossible with information theoretic security in plain model

What happens with 2 calls?

- Possible! (Based on functional encryption)
- How about computational?
- How about i.t with setup?

Open

Thank you :)