Public Randomness Extraction with Ephemeral Roles and Worst-Case Corruptions

Jesper Buus Nielsen
Aarhus

João Ribeiro
CMU

Maciej Obremski
CQT & NUS
Cryptography needs randomness!

How do we generate it?
Cryptography needs randomness!

How do we generate it?
Cryptography needs randomness!

How do we generate it?

\[ b \leftarrow \{0,1\} \]
Cryptography needs randomness!

How do we generate it?

Randomness generator

$b \leftarrow \{0,1\}$
Cryptography needs randomness!

How do we generate it?

Randomness generator

\[ b \leftarrow \{0,1\} \]
Cryptography needs randomness!

How do we generate it?

but maintaining stateful environments is hard, especially under targeted denial-of-service attacks!
Cryptography needs randomness!

How do we generate it?

but maintaining stateful environments is hard, especially under targeted denial-of-service attacks!
Cryptography needs randomness!

How do we generate it?

but maintaining stateful environments is hard, especially under targeted denial-of-service attacks!
Ground set of $N$ parties

$n$ ephemeral roles, $n \ll N$

$R_1$ $R_2$ $R_3$ ...
Ground set of $N$ parties

$n$ ephemeral roles, $n \ll N$

role selection mechanism

$R_1 \quad R_2 \quad R_3 \quad \ldots$
Ground set of $N$ parties

$n$ ephemeral roles, $n \ll N$

Role selection mechanism

Broadcast
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry\textsuperscript{1}, Shai Halevi\textsuperscript{1}, Hugo Krawczyk\textsuperscript{1}, Bernardo Magri\textsuperscript{2}, Jesper Buus Nielsen\textsuperscript{*2}, Tal Rabin\textsuperscript{1}, and Sophia Yakoubov\textsuperscript{13}

$n$ ephemeral roles, $n \ll N$

broadcast

$n$ parties

$X_1$

$R_2$

$R_3$

$\ldots$

role selection mechanism

$n$ ephemeral roles, $n \ll N$
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry¹, Shai Halevi¹, Hugo Krawczyk¹, Bernardo Magri², Jesper Buus Nielsen⁺², Tal Rabin¹, and Sophia Yakoubov³

$n$ ephemeral roles, $n \ll N$

$M_{1 \rightarrow 2}$

$M_{1 \rightarrow 3}$

role selection mechanism

Ground set of $N$ parties

$R_2$

$R_3$

$X_1$
**YOSO: You Only Speak Once**
Secure MPC with Stateless Ephemeral Roles

Craig Gentry\(^1\), Shai Halevi\(^1\), Hugo Krawczyk\(^1\), Bernardo Magri\(^2\), Jesper Buus Nielsen\(^*2\), Tal Rabin\(^1\), and Sophia Yakoubov\(^i3\)

---

$n$ ephemeral roles, $n \ll N$

Ground set of $N$ parties

\(M_{1\rightarrow 3}\)

\(M_{1\rightarrow 2}\)

\(M_{2\rightarrow 3}\)

\(X_1\)

\(X_2\)

\(R_2\)

\(R_3\)

role selection mechanism
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry\textsuperscript{1}, Shai Halevi\textsuperscript{1}, Hugo Krawczyk\textsuperscript{1}, Bernardo Magri\textsuperscript{2}, Jesper Buus Nielsen\textsuperscript{*1,2}, Tal Rabin\textsuperscript{1}, and Sophia Yakoubov\textsuperscript{1,3}

$n$ ephemeral roles, $n \ll N$

Role selection mechanism

Ground set of $N$ parties

$X_1 \xrightarrow{M_{1 \rightarrow 2}} X_2 \xrightarrow{M_{2 \rightarrow 3}} R_3$
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry¹, Shai Halevi¹, Hugo Krawczyk¹, Bernardo Magri², Jesper Buus Nielsen*², Tal Rabin¹, and Sophia Yakoubov³

$n$ ephemeral roles, $n \ll N$

Role selection mechanism

Ground set of $N$ parties
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry\textsuperscript{1}, Shai Halevi\textsuperscript{1}, Hugo Krawczyk\textsuperscript{1}, Bernardo Magri\textsuperscript{2}, Jesper Buus Nielsen\textsuperscript{*2}, Tal Rabin\textsuperscript{1}, and Sophia Yakoubov\textsuperscript{13}

\[ X_2 \xrightarrow{M_2 \rightarrow 3} X_3 \]
\[ X_1 \xrightarrow{M_1 \rightarrow 2} X_2 \]
\[ X_1 \xrightarrow{M_1 \rightarrow 3} X_3 \]

\( n \) ephemeral roles, \( n \ll N \)

role selection mechanism

Ground set of \( N \) parties

\( R_3 \)
**YOSO: You Only Speak Once**

Secure MPC with Stateless Ephemeral Roles

Craig Gentry\(^1\), Shai Halevi\(^1\), Hugo Krawczyk\(^1\), Bernardo Magri\(^2\), Jesper Buus Nielsen\(^*2\),
Tal Rabin\(^1\), and Sophia Yakoubov\(^{i3}\)

---

**Graphical Representation**

- **Ground set of \( N \) parties**
- **Role selection mechanism**
- **\( M_{1 \rightarrow 3} \)**
- **\( M_{1 \rightarrow 2} \)**
- **\( M_{2 \rightarrow 3} \)**

**Textual Description**

\( n \) ephemeral roles, \( n \ll N \)

\( X_1 \)

\( X_2 \)

\( X_3 \)

---

\( n \) ephemeral roles, \( n \ll N \)
**YOSO: You Only Speak Once**
Secure MPC with Stateless Ephemeral Roles

Craig Gentry$^1$, Shai Halevi$^1$, Hugo Krawczyk$^1$, Bernardo Magri$^2$, Jesper Buus Nielsen$^2$,
Tal Rabin$^1$, and Sophia Yakoubov$^1$

$n$ ephemeral roles, $n \ll N$
YOSO: You Only Speak Once
Secure MPC with Stateless Ephemeral Roles

Craig Gentry¹, Shai Halevi¹, Hugo Krawczyk¹, Bernardo Magri², Jesper Buus Nielsen*², Tal Rabin¹, and Sophia Yakoubov³

\[
\begin{align*}
    t & = \left( \frac{1}{2} - \delta \right) N \\
    \text{role selection mechanism assumed uniformly random} \\
    \implies \text{MPC with } t = (1/2 - \delta)N \text{ corruptions!}
\end{align*}
\]

\( \begin{align*}
    n & \text{ ephemeral roles, } \\
    n & \ll N
\end{align*} \)

\( \begin{align*}
    X_1 & \xrightarrow{M_{1 \rightarrow 3}} \\
    X_2 & \xrightarrow{M_{1 \rightarrow 2}} \\
    X_3 & \xrightarrow{M_{2 \rightarrow 3}}
\end{align*} \)

Abstracted away

Ground set of \( N \) parties
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

$R_1 \quad R_2 \quad R_3 \quad R_4$
YOSO public randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

\[ R_1 \quad \text{😈} \quad R_3 \quad \text{😈} \]
YOSO public randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by static chosen corruptions

private messages

$R_1$ → $R_3$ → broadcast

$X_1$
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**
YOSO public randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by static chosen corruptions
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**
YOSO **public** randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

\[ B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif} \]
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

\[
B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif}
\]

*only public values*
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

\[
\begin{align*}
B &= \text{Ext}(X_1, X_2, X_3, X_4) \\
&\approx \text{Unif}
\end{align*}
\]

only public values
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

Why study worst-case corruptions?
- Role selection mechanism may be biased!

\[ B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif} \]

*only public values*
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

Why study worst-case corruptions?
- Role selection mechanism may be biased!
- Go beyond round-based MPC techniques

\[
B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif}
\]

only public values
YOSO public randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

Why study worst-case corruptions?
- Role selection mechanism may be biased!
- Go beyond round-based MPC techniques
- Clean model

\[ B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif} \]

only public values
YOSO *public* randomness extraction with worst-case corruptions

Replace i.i.d. random corruptions by **static chosen corruptions**

Why study worst-case corruptions?
- Role selection mechanism may be biased!
- Go beyond round-based MPC techniques
- Clean model
- Relationship to other randomness extraction settings

\[
B = \text{Ext}(X_1, X_2, X_3, X_4) \approx \text{Unif}
\]

only public values
How are “messages to the future” implemented?

In this talk: Adversary learns incoming messages to corrupted role only when role is executed
[Campanelli-David-Khoshakhlagh-Kristensen-Nielsen '21]

Simple models inspired by concrete implementations of such a mechanism.
How are “messages to the future” implemented?

In this talk: Adversary learns incoming messages to corrupted role only when role is executed
[Campanelli-David-Khoshakhlagh-Kristensen-Nielsen '21]

Simple models inspired by concrete implementations of such a mechanism.
How are “messages to the future” implemented?

In this talk: Adversary learns incoming messages to corrupted role only when role is executed

[Campanelli-David-Khoshakhlagh-Kristensen-Nielsen '21]
How are “messages to the future” implemented?

In this talk: Adversary learns incoming messages to corrupted role only when role is executed
[Campanelli-David-Khoshakhlagh-Kristensen-Nielsen ’21]
Our central question

What is the maximum corruption rate that allows for low-bias randomness extraction?
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

$$P_1^{(1)} \ldots P_n^{(1)} \quad P_1^{(2)} \ldots P_n^{(2)} \quad \ldots \quad P_1^{(r)} \ldots P_n^{(r)}$$
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

\[
\begin{array}{cccc}
P^{(1)}_1 & \ldots & P^{(1)}_n \\
\ldots & \ldots & \ldots & \ldots \\
\text{emulate round 1} & \ldots & \ldots & \ldots \\
\end{array}
\]
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

$p^{(j)}_i$ sends secret state to $p^{(j+1)}_i$

$P^{(1)}_1 \ldots P^{(1)}_n \quad P^{(2)}_1 \ldots P^{(2)}_n \quad \ldots \quad P^{(r)}_1 \ldots P^{(r)}_n$

emulate round 1
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

$$P_i^{(j)} \text{ sends secret state to } P_i^{(j+1)}$$

<table>
<thead>
<tr>
<th>$P_1^{(1)}$</th>
<th>$P_2^{(1)}$</th>
<th>...</th>
<th>$P_n^{(1)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>emulate round 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P_1^{(2)}$</th>
<th>$P_2^{(2)}$</th>
<th>...</th>
<th>$P_n^{(2)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>emulate round 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P_1^{(r)}$</th>
<th>$P_2^{(r)}$</th>
<th>...</th>
<th>$P_n^{(r)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>emulate round $r$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

$$P_1^{(1)} \cdots P_n^{(1)} \quad P_1^{(2)} \cdots P_n^{(2)} \quad \cdots \quad P_1^{(r)} \cdots P_n^{(r)}$$

- $P_i^{(j)}$ sends secret state to $P_i^{(j+1)}$

Tolerated corruption rate decreased from $\delta$ to $\delta/r$
A naive approach to YOSO protocols

Starting point: Round-based $n$-party $r$-round protocol secure against $t = \delta n$ corruptions.

Emulate rounds in YOSO:

\[
P_i^{(j)} \text{ sends secret state to } P_i^{(j+1)}
\]

\[
P_1^{(1)} \cdots P_n^{(1)} \quad P_1^{(2)} \cdots P_n^{(2)} \quad \cdots \quad P_1^{(r)} \cdots P_n^{(r)}
\]

emulate round 1 \quad emulate round 2 \quad \cdots \quad emulate round $r$

Tolerated corruption rate decreased from $\delta$ to $\delta/r$

3 rounds, $\delta \approx 1/3$ corruption rate $\implies$ YOSO protocol secure against $\approx 1/9$ corruption rate
Our results

Feasibility

Impossibility
Our results

**Feasibility**

Zero-error randomness extraction against $t$ corruptions with $n = 5t$ roles

(or $n = 6t + 1$ roles against stronger adversary)

**Impossibility**
Our results

Feasibility

Zero-error randomness extraction against $t$ corruptions with $n = 5t$ roles
(or $n = 6t + 1$ roles against stronger adversary)

Impossibility

Randomness extraction with bias $< 0.01$ against $t$ corruptions requires $n \geq 4t + 1$ roles
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.
Protocol

There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”

$R_1 \quad R_2 \quad \ldots \quad R_{3t+1} \quad R'_1 \quad R'_2 \quad \ldots \quad R'_{2t+1}$

samplers \quad publishers
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”

\[
R_1 \quad R_2 \quad \ldots \quad R_{3t+1} \quad R'_1 \quad R'_2 \quad \ldots \quad R'_{2t+1}
\]

samplers \hspace{2cm} publishers

High-level idea:
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”

\[
R_1 \quad R_2 \quad \ldots \quad R_{3t+1} \quad R'_1 \quad R'_2 \quad \ldots \quad R'_{2t+1}
\]

samplers \quad publishers

**High-level idea:**
1. Several subsets of samplers commit to values and send them to publishers;
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”

\[ R_1 \quad R_2 \quad \ldots \quad R_{3t+1} \quad R'_1 \quad R'_2 \quad \ldots \quad R'_{2t+1} \]

samplers                  publishers

**High-level idea:**
1. Several subsets of samplers commit to values and send them to publishers;
2. Publishers broadcast whatever they receive;
There is a zero-error randomness extraction protocol secure against $t$ chosen corruptions in the with $n = 5t + 2$ roles.

YOSOfied version of Maurer’s “Secure MPC made simple”

$$R_1 \quad R_2 \quad \ldots \quad R_{3t+1} \quad R'_1 \quad R'_2 \quad \ldots \quad R'_{2t+1}$$

samplers publishers

High-level idea:
1. Several subsets of samplers commit to values and send them to publishers;
2. Publishers broadcast whatever they receive;
3. Extract random bit by taking majorities and XOR.
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

\[ R_{i_1} \quad R_{i_2} \quad \ldots \quad R_{i_j} \quad \ldots \quad R_{i_{2t+1}} \quad R'_{i_1} \quad \ldots \quad R'_{i_{2t+1}} \]

samplers

publishers
$S = \{ i_1 < i_2 < \cdots < i_{2t+1} \} \subseteq [3t+1]$
Protocol

\[ S = \{ i_1 < i_2 < \cdots < i_{2t+1} \} \subseteq [3t + 1] \]

Diagram:

- \( R_{i_1} \)
- \( R_{i_2} \)
- \( \ldots \)
- \( R_{i_j} \)
- \( \ldots \)
- \( R_{i_{2t+1}} \)

Samples \( x_1 \) from \( \{0,1\} \)

Publishers:

- \( R'_1 \)
- \( \ldots \)
- \( R'_{2t+1} \)
Protocol

\[ S = \{ i_1 < i_2 < \cdots < i_{2t+1} \} \subseteq [3t + 1] \]
$S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1]$

**Protocol**

If not all equal, broadcast (Complain, $S$), procedure aborts

$x_1 \leftarrow \{0,1\}$

$R_{i_1} \rightarrow R_{i_2} \rightarrow \cdots \rightarrow R_{i_j} \rightarrow \cdots \rightarrow R_{i_{2t+1}}$

$x_1 \rightarrow R_{i_1} \rightarrow \cdots \rightarrow R_{i_{2t+1}}$

samplers

$S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1]$
Protocol

\[ S = \{ i_1 < i_2 < \cdots < i_{2t+1} \} \subseteq [3t + 1] \]

- \( x_1 \leftarrow \{0, 1\} \)
- \( x_1 \rightarrow 2 \)
- \( x_1 \rightarrow 2 \rightarrow j \)
- \( \vdots \)
- \( x_j \rightarrow 1 \rightarrow j \)

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts.

If all equal some \(z\), send \(z\) to all publishers.

\( R_i \) publishers

\( R'_i \)

consistency check

samplers
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

If all equal some \( z \), send \( z \) to all publishers.

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts.

\[ x_1 \leftarrow \{0, 1\} \]

\[ x_1 \rightarrow (0, 1) \]

\[ x_2 \rightarrow (0, 1) \]

\[ \vdots \]

\[ x_{j-1} \rightarrow (0, 1) \]

\[ x_{j} \rightarrow (0, 1) \]

\[ \vdots \]

\[ z_{2t+1} \rightarrow (0, 1) \]

Samplers consistency check

Publishers
$S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1]$

**Protocol**

If all equal some $z$, send $z$ to all publishers

If not all equal, broadcast $(\text{Complain}, S)$, procedure aborts

$w_1 = \text{maj}\{z_{j\rightarrow 1}\}$
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

If all equal some \( z \), send \( z \) to all publishers

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts

\[ w_1 = \text{maj}\{z_{j-1}\} \]

\[ w_{2t+1} = \text{maj}\{z_{j \rightarrow 2t+1}\} \]

\[ R_i \]

\[ R_i \rightarrow \]

\[ x_1 \leftarrow \{0, 1\} \]

\[ x_1 \rightarrow 2 \]

\[ x_j \rightarrow j \]

\[ x_{j-1} \rightarrow j \]

publishers

\[ R' \]

\[ R' \rightarrow \]

\[ R'_{i \rightarrow 2t+1} \]

\[ \{s_i \mid i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

\[ w_1 = \text{maj}\{z_{j \rightarrow 1}\} \]

\[ w_{2t+1} = \text{maj}\{z_{j \rightarrow 2t+1}\} \]

\[ w_S = \begin{cases} 
0, & \text{if } S \text{ received a complaint,} \\
\text{maj}(w_j), & \text{else.}
\end{cases} \]
Protocol

\[ S = \{ i_1 < i_2 < \cdots < i_{2t+1} \} \subseteq [3t+1] \]

If all equal some \( z \), send \( z \) to all publishers

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts

\[ w_1 = \text{maj}\{z_{j \rightarrow 1}\} \]

\[ w_{2t+1} = \text{maj}\{z_{j \rightarrow 2t+1}\} \]

\[ w_S = \begin{cases} 
0, & \text{if } S \text{ received a complaint,} \\
\text{maj}(w_j), & \text{else.}
\end{cases} \]

\[ b = \bigoplus_S w_S \]
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t+1] \]

\[ x_1 \leftarrow \{0,1\} \]

\[ x_1 \rightarrow 1 \]

\[ x_1 \rightarrow 2 \]

\[ x_j \rightarrow j \]

\[ z \]

If all equal some \( z \), send \( z \) to all publishers

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts

Key properties:

\[ w_1 = \text{maj}\{z_{j\rightarrow 1}\} \]

\[ w_{2t+1} = \text{maj}\{z_{j\rightarrow 2t+1}\} \]

\[ w_S = \begin{cases} 0, & \text{if } S \text{ received a complaint,} \\ \text{maj}(w_j), & \text{else.} \end{cases} \]

\[ b = \bigoplus_{S} w_S \]
Protocol

$S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1]$

Key properties:
- Every set $S$ has a strict honest majority;

If all equal some $z$, send $z$ to all publishers
If not all equal, broadcast (Complain, $S$), procedure aborts

$w_S = \begin{cases} 0, & \text{if } S \text{ received a complaint,} \\ \text{maj}(w_j), & \text{else.} \end{cases}$

$b = \bigoplus_{S} w_S$
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

**Key properties:**

- Every set \( S \) has a strict honest majority;
- If \( S \) did not receive a complaint, then all honest roles in \( S \) agree on the same value;
Protocol

\[ S = \{ i_1 < i_2 < \cdots < i_{2r+1} \} \subseteq [3t + 1] \]

**Key properties:**
- Every set \( S \) has a strict honest majority;
- If \( S \) did not receive a complaint, then all honest roles in \( S \) agree on the same value;
- There is a set \( S^* \) such that all roles \((P_i)_{i \in S^*}\) are honest.
Protocol

\[ S = \{i_1 < i_2 < \cdots < i_{2t+1}\} \subseteq [3t + 1] \]

Key properties:
- Every set \( S \) has a strict honest majority;
- If \( S \) did not receive a complaint, then all honest roles in \( S \) agree on the same value;
- There is a set \( S^* \) such that all roles \( (P_i)_{i \in S^*} \) are honest.

\[ w_{S^*} \text{ is uniform and independent of } (w_S)_{S \neq S^*}. \]

If not all equal, broadcast \((\text{Complain}, S)\), procedure aborts

If all equal some \( z \), send \( z \) to all publishers
A simple improved protocol for one corruption

\[ R_1 \quad R_2 \quad R'_1 \quad R'_2 \quad R'_3 \]

samplers

publishers
A simple improved protocol for one corruption

\[ x_1 \leftarrow \{0, 1\} \]

\[
\begin{array}{ccc}
R_1 & R_2 & R'_1 & R'_2 & R'_3 \\
\end{array}
\]
A simple improved protocol for one corruption

$R_1 \xleftarrow{} \{0,1\} x_1 \rightarrow R_2 \rightarrow R'_1 \rightarrow R'_2 \rightarrow R'_3$

samplers

$R_1$, $R_2$, $R'_1$, $R'_2$, $R'_3$

publishers
A simple improved protocol for one corruption
A simple improved protocol for one corruption

\[ x_1 \leftarrow \{0,1\} \]

\[ x_2 \leftarrow \{0,1\} \]

samplers

\[ R_1 \]

\[ R_2 \]

publishers

\[ R'_1 \]

\[ R'_2 \]

\[ R'_3 \]
A simple improved protocol for one corruption

\[ x_1 \leftarrow \{0,1\} \]

\[ x_2 \leftarrow \{0,1\} \]

\[
\begin{align*}
R_1 & \quad R_2 \\
R'_1 & \quad R'_2 & \quad R'_3
\end{align*}
\]
A simple improved protocol for one corruption

\[ x_1 \xleftarrow{} \{0,1\} \]
\[ x_2 \xleftarrow{} \{0,1\} \]

Publishers:
- \( R_1 \)
- \( R_2 \)
- \( R_1' \)
- \( R_2' \)
- \( R_3' \)

Samplers:
- \( R_1 \)
- \( R_2 \)
A simple improved protocol for one corruption

$\begin{align*}
R_1 &\xleftarrow{} \{0,1\} \\
R_2 &\xleftarrow{} \{0,1\} \\
R'_1 &\xrightarrow{} 1 \\
R'_2 &\xrightarrow{} 1 \\
R'_3 &\xrightarrow{} 1 \\
x_1 &\xleftarrow{} \{0,1\} \\
x_2 &\xleftarrow{} \{0,1\}
\end{align*}$
A simple improved protocol for one corruption
A simple improved protocol for one corruption
A simple improved protocol for one corruption

$$y_1 = \text{maj}(x_{1\rightarrow 1}, x_{1\rightarrow 2}, x_{1\rightarrow 3})$$
A simple improved protocol for one corruption

$$y_1 = \text{maj}(x_{1\rightarrow 1}, x_{1\rightarrow 2}, x_{1\rightarrow 3})$$

$$y_2 = \text{maj}(x_{2\rightarrow 1}, x_{2\rightarrow 2}, x_{2\rightarrow 3})$$
A simple improved protocol for one corruption

\[ y_1 = \text{maj}(x_{1\to1}, x_{1\to2}, x_{1\to3}) \]
\[ y_2 = \text{maj}(x_{2\to1}, x_{2\to2}, x_{2\to3}) \]
\[ b = y_1 \oplus y_2 \]

\text{samplers}

\[ x_1 \leftarrow \{0,1\} \]
\[ x_2 \leftarrow \{0,1\} \]

\text{publishers}

\[ R_1 \quad R_2 \]
\[ R'_1 \quad R'_2 \quad R'_3 \]
A simple improved protocol for one corruption

$y_1 = \text{maj}(x_{1\rightarrow1}, x_{1\rightarrow2}, x_{1\rightarrow3})$

$y_2 = \text{maj}(x_{2\rightarrow1}, x_{2\rightarrow2}, x_{2\rightarrow3})$

$b = y_1 \oplus y_2$

Can be generalized using $n = 5t$ roles.
Impossibility result

Every randomness extraction protocol with bias $< 0.01$ against $t$ corruptions requires $n \geq 4t + 1$ roles.
Every randomness extraction protocol with bias < 0.01 against $t$ corruptions requires $n \geq 4t + 1$ roles.

$$n^* = n^*(t) = \text{Smallest number of roles for which we can handle } t \text{ corruptions}$$

Stronger adversary: $4t + 1 \leq n^* \leq 6t + 1$

Weaker adversary (this talk): $4t + 1 \leq n^* \leq 5t$
Impossibility for 3 roles, 1 corruption

\[ R_1 \quad R_2 \quad R_3 \]
Impossibility for 3 roles, 1 corruption

$R_1 \xrightarrow{X_1} R_2 \xrightarrow{} R_3$
Impossibility for 3 roles, 1 corruption
Impossibility for 3 roles, 1 corruption
Impossibility for 3 roles, 1 corruption

\[ B = \text{Ext}(X_1, X_2, X_3) \]
Impossibility for 3 roles, 1 corruption

\[ B = \text{Ext}(X_1, X_2, X_3) \]
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that
\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]

\[B = \text{Ext}(X_1, X_2, X_3)\]
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that
\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]

Bias!
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that

\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]

Bias!

\( X_1, X_2 \) must fully determine output w/ high prob
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that
\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]

Bias!

\( X_1, X_2 \) must fully determine output w/ high prob
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that
\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]
and both are constant

Bias!

\( X_1, X_2 \) must fully
determine output
w/ high prob
Impossibility for 3 roles, 1 corruption

If there are such that \( x(0) \equiv 3 \), \( x(1) \equiv 3 \),

\[
\mathcal{B} = \text{Ext}(X_1, X_2, x(b)) = b
\]

Bias!

Sample \( X_2^{(0)}, X_2^{(1)} \) \( i.i.d. \) \( (X_2 \mid X_1, M_{1\rightarrow 2}) \)

If \( \text{Ext}(X_1, X_2^{(0)}, \cdot) \neq \text{Ext}(X_1, X_2^{(1)}, \cdot) \)
and both are constant

Bias!

\( X_1, X_2 \) must fully determine output w/ high prob
Impossibility for 3 roles, 1 corruption

If there are \( x_3^{(0)}, x_3^{(1)} \) such that
\[ \text{Ext}(X_1, X_2, x_3^{(b)}) = b \]

Bias!

Bias!

\( B = \text{Ext}(X_1, X_2, X_3) \)

Sample \( X_2^{(0)}, X_2^{(1)} \) i.i.d. \( (X_2 | X_1, M_{1 \rightarrow 2}) \)

If \( \text{Ext}(X_1, X_2^{(0)}, \cdot ) \neq \text{Ext}(X_1, X_2^{(1)}, \cdot ) \)

and both are constant

\( X_1, X_2 \text{ must fully determine output w/ high prob} \)

W/ high prob over sampling
\( X_2^{(0)}, X_2^{(1)} \) i.i.d. \( (X_2 | X_1, M_{1 \rightarrow 2}) \)

must have
\[ \text{Ext}(X_1, X_2^{(0)}, \cdot ) \equiv \text{Ext}(X_1, X_2^{(1)}, \cdot ) \]
Impossibility for 3 roles, 1 corruption

If there are $x_3^{(0)}, x_3^{(1)}$ such that

$$\text{Ext}(X_1, X_2, x_3^{(b)}) = b$$

$B = \text{Ext}(X_1, X_2, X_3)$

Sample $X_2^{(0)}, X_2^{(1)} \overset{i.i.d.}{\sim} (X_2 | X_1, M_{1\rightarrow 2})$

If Ext($X_1, X_2^{(0)}, \cdot$) $\neq$ Ext($X_1, X_2^{(1)}, \cdot$) and both are constant

W/ high prob over sampling

$X_2^{(0)}, X_2^{(1)} \overset{i.i.d.}{\sim} (X_2 | X_1, M_{1\rightarrow 2})$

must have

Ext($X_1, X_2^{(0)}, \cdot$) $\equiv$ Ext($X_1, X_2^{(1)}, \cdot$)

$X_1, X_2$ must fully determine output w/ high prob
Impossibility for 3 roles, 1 corruption

If there are \( x_1(0), x_1(1) \) such that
\[
3
x(0) \quad 3
1 \quad 3
x(1) \quad 3
\]

Then

\[
\text{Bias!}
\]

Sample \( X_1^{(0)}, X_1^{(1)} \) i.i.d. \( X_1 \)
With prob \( \approx 1/2 \) lead to different
coin values and \( R_1 \) can predict
values w/ high prob

If \( \text{Ext}(X_1, X_2^{(0)}, \cdot) \neq \text{Ext}(X_1, X_2^{(1)}, \cdot) \)
and both are constant

\[
\text{Bias!}
\]

If there are \( x_3^{(0)}, x_3^{(1)} \)
such that
\[
\text{Ext}(X_1, X_2, x_3^{(b)}) = b
\]

\[
\text{Bias!}
\]

W/ high prob over sampling
\[
X_2^{(0)}, X_2^{(1)} \text{ i.i.d. } (X_2 | X_1, M_{1\rightarrow 2})
\]

Must have
\[
\text{Ext}(X_1, X_2^{(0)}, \cdot) \equiv \text{Ext}(X_1, X_2^{(1)}, \cdot)
\]

\[
X_1, X_2 \text{ must fully determine output w/ high prob}
\]

\[
B = \text{Ext}(X_1, X_2, X_3)
\]
Impossibility for 3 roles, 1 corruption

If there are such that
$x(0) \not\equiv x(1)$,
$\mathcal{D}(X_1, X_2, x(b)) = b$

Bias!
Bias!
Bias!

$B = \operatorname{Ext}(X_1, X_2, X_3)$

Sample $X_1^{(0)}, X_1^{(1)} \sim X_1$
With prob $\approx 1/2$ lead to different coin values and $R_1$ can predict values w/ high prob

$W/ high prob over sampling$
$X_2^{(0)}, X_2^{(1)} \sim (X_2 | X_1, M_{1\rightarrow 2})$

If there are $x_3^{(0)}, x_3^{(1)}$
such that
$\operatorname{Ext}(X_1, X_2, x_3^{(b)}) = b$

$X_1, X_2 must fully determine output w/ high prob$

$X_1, X_2 must have$
$\operatorname{Ext}(X_1, X_2^{(0)}, \cdot) \equiv \operatorname{Ext}(X_1, X_2^{(1)}, \cdot)$
Impossibility for 4 roles, 1 corruption
Impossibility for 4 roles, 1 corruption
Impossibility for 4 roles, 1 corruption
Impossibility for 4 roles, 1 corruption
Impossibility for 4 roles, 1 corruption

- $R_2$ must be able to *influence* final output;
- $R_2$ must be able to *predict* which path leads to each value.
Impossibility for 4 roles, 1 corruption

- $R_2$ must be able to influence final output;
- $R_2$ must be able to predict which path leads to each value.

$R_2$ doesn't know $M_{1 \rightarrow 3}$!
Concluding

• YOSO: Stateless MPC, avoids denial-of-service attacks
• Public randomness extraction in YOSO with worst-case corruptions:
  • Still secure if role selection mechanism is biased
  • Go beyond round-based MPC techniques
  • Related to prior work on multi-source randomness extraction

**Stronger adversary:** $4t + 1 \leq n^* \leq 6t + 1$

**Weaker adversary:** $4t + 1 \leq n^* \leq 5t$
Concluding

• YOSO: Stateless MPC, avoids denial-of-service attacks
• Public randomness extraction in YOSO with worst-case corruptions:
  • Still secure if role selection mechanism is biased
  • Go beyond round-based MPC techniques
  • Related to prior work on multi-source randomness extraction

**Stronger adversary:** $4t + 1 \leq n^* \leq 6t + 1$

**Weaker adversary:** $4t + 1 \leq n^* \leq 5t$

**Open problems:**
• Close gap between our upper and lower bounds
• Protocols with communication complexity $\text{poly}(n)$
• More functionalities!
Concluding

• YOSO: Stateless MPC, avoids denial-of-service attacks
• Public randomness extraction in YOSO with worst-case corruptions:
  • Still secure if role selection mechanism is biased
  • Go beyond round-based MPC techniques
  • Related to prior work on multi-source randomness extraction

**Stronger adversary:** \(4t + 1 \leq n^* \leq 6t + 1\)

**Weaker adversary:** \(4t + 1 \leq n^* \leq 5t\)

Open problems:
• Close gap between our upper and lower bounds
• Protocols with communication complexity \(\text{poly}(n)\)
• More functionalities!

Thanks!