This Lecture:

- SNARGs from PCP (continued)
- SNARGs from Linear PCP
- Introduction to MPC
**Succinct Non-Interactive Argument (SNARG)**

\[ \forall p^* \forall \text{PPT } p^* \text{ (in soundness)} \]

**Def** A non-interactive proof/argument system is **succint** if

- The proof \( \pi \) is of length \( |\pi| = \text{poly}(\lambda, \log |C_1|) \)
- The verifier runs in time \( \text{poly}(\lambda, |x|, \log |C_1|) \)

- **SNARK**: Succinct Non-Interactive Argument of Knowledge
- **zk-SNARG/zk-SNARK**: SNARG/SNARK + Zero-Knowledge
Probabilistically Checkable Proof (PCP)

**Prover**

\((x, w)\)

**Verifier**

\((x)\)

\[ \pi \in \{0, 1\}^m \]

\[ \begin{array}{cccccc}
0 & 1 & 1 & 0 & 1 & \ldots & 1 \\
\end{array} \]

**PCP Theorem (Informal):**

Every NP language has a PCP where the Verifier reads only a constant number of bits of the proof.
First Attempt

Prover

\((x, w)\)

Com(01101 \ldots 1)

Verifier

\((x)\)

i, j, k

Open Com(\(T_i\)), Com(\(T_j\)), Com(\(T_k\))
Merkle Tree

Why (computationally) binding?

Can we make it hiding?
Is it ZK?

Prover

\( (x, w) \)

\[ MT(\text{Com}([0 1 1 0 1 \ldots 1])) \]

Verifier

\( (x) \)

\[ i, j, k \]

Open \( \text{Com}(\pi_i), \text{Com}(\pi_j), \text{Com}(\pi_k) \)
Linear PCP

**Prover**

$((x, w)) \rightarrow \pi \in F^m$

independent of $x$

$\left\{ \begin{array}{l} q_2 \in F^m \Rightarrow \langle \pi, q_2 \rangle \in F \\ \vdots \\ q_t \in F^m \Rightarrow \langle \pi, q_t \rangle \in F \end{array} \right.$

**Verifier**

$(x)$

Constructing LPCP for Circuit Satisfiability:

- From Walsh-Hadard code, $m = O(1c^2)$
- From quadratic span programs, $m = O(1c)$
**Preprocessing Model (Designated Verifier)**

**Prover**

\[(pk, sk) \leftarrow \text{Gen}(1^Y)\]

\[q_i \in \mathbb{F}_p^m \quad C_i \leftarrow \text{Enc}_{pk}(q_i)\]

\[\vdots\]

\[q_t \in \mathbb{F}_p^m \quad C_t \leftarrow \text{Enc}_{pk}(q_t)\]

\[\text{PK, } C_1, \ldots, C_t, \ldots\]

\[\text{PCP}\]  \[
\text{(publish)} \]

\[\text{Verifier}\]

\[\text{TLpcp}\]

\[\text{(publish)}\]

\[(x, w) \quad \pi \in \mathbb{F}_p^m\]

\[\text{Enc}_{pk}(\langle \pi, q_i \rangle) \rightarrow R_i\]

\[\vdots\]

\[\text{Additively Homomorphic}\]

\[\text{Final Check } (x, \text{TLpcp}, R_2, \ldots, R_t)\]

\[\text{quadratic function}\]
Publicly Verifiable

\[ \text{Prover} \quad (x, w) \]
\[ \pi \in \mathbb{F}^m \]
\[ g^{\langle \pi, q_t \rangle} \rightarrow g^{r_t} \]
Final Check \((x, T_{\text{pcp}}, r_1, \ldots, r_t)\)

\[ \text{Verifier} \quad (x) \]
\[ q_t \in \mathbb{F}^m \]
\[ g^{\langle \pi, q_t \rangle} \rightarrow g^{r_t} \]

\[ C_1 := g^{q_1} \]
\[ \vdots \]
\[ C_t := g^{q_t} \]
\[ C_T := g^{T_{\text{pcp}}} \]
Gen \((1^\lambda)\)
Bilinear Pairings

Cyclic groups $G_1$, $G_2$, $G_T$ with generators $g_1, g_2, g_T$, respectively.

$$e : G_1 \times G_2 \rightarrow G_T$$

$$e(g_1^a, g_2^b) = g_T^{ab}$$
Secure Multi-Party Computation

Alice  

Second date?  
\[ f(x, y) = x \land y \]

Who is richer?  
\[ f(x, y) = \begin{cases} 0 & \text{if } x > y \\ 1 & \text{otherwise} \end{cases} \]

Common friends?  
\[ f(x, y) = x \land y \]
Secure Two-Party Computation (2PC)

Applications:
- Password Breach Alert (Chrome/Firefox/Azure/iOS Keychain)
- Privacy-Preserving Contact Tracing for COVID-19 (Apple & Google)
- Ads Conversion Measurements / Personalized Advertising (Google/Meta)
Secure Multi-Party Computation (MPC)

\[ z = f(x_1, \ldots, x_n) \]
Secure Multi-Party Computation (MPC)

Applications:
- Privacy-Preserving Inventory Matching (J.P. Morgan)
- Setup Ceremony to securely generate CRS (Zcash)
- Distributed Key Management (Unbound / Coinbase)
- Federated Learning (Google Keyboard Search Suggestion)
- Auctions (Danish sugar beet auction)
- Boston gender wage gap (Boston Women's Workforce Council)
- Study / Analysis on Medical Data
- Fraud Detection (banks)
Setting

- n parties $P_1, P_2, \ldots, P_n$
  - with private inputs $X_1, X_2, \ldots, X_n$

- Jointly compute $f(X_1, X_2, \ldots, X_n)$

Communication:
- Authenticated secure point-to-point channels between each pair $(P_i, P_j)$
  (sometimes also assume broadcast channel)

- The adversary can "corrupt" a subset of the parties
  (e.g. at most $t$ parties)

What properties do we want?
General Security Properties

- **Correctness**: The function is computed correctly.
- **Privacy**: Only the output is revealed.
- **Independence of Inputs**: Parties cannot choose inputs depending on others' inputs.
- **Security with Abort**: Adversary may “abort” the protocol. (preventing honest parties from receiving the output)
- **Fairness**: If one party receives output, then all receive output.
- **Guaranteed Output Delivery (GOD)**: Honest parties always receive output.
Adversary’s Power

Allowed adversarial behavior:

· Semi-honest/passive/honest-but-curious:
  
  Follow the protocol description honestly, but try to extract more information by inspecting transcript.

· Malicious/active:
  
  Can deviate arbitrarily from the protocol description.

Adversary’s Computing Power:

· Unbounded computing power \Rightarrow \text{Information-Theoretic (IT) Security}

· PPT bounded \Rightarrow \text{Computational Security}
Feasibility Results

Computational Security:

Semi-honest Oblivious Transfer (OT)

\[ \downarrow \]

Semi-honest MPC for any function with \( t < n \)

\[ \downarrow \]

Malicious MPC for any function with \( t < n \)

Information-Theoretic (IT) Security:

Semi-honest/malicious MPC for any function with \( t < n/2 \)

(honest majority)

\[ \uparrow \]

necessary
Oblivious Transfer (OT)

Sender

\[ m_0, m_2 \in \{0, 1\}^l \]

Input: \( m_0, m_2 \in \{0, 1\}^l \)

\[ b \in \{0, 1\} \]

Input: \( b \in \{0, 1\} \)

\[ L \]

Output: \( L \)

\[ m_b \]

Output: \( m_b \)