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 succinct if
- The proof \( \Pi \) is of length \( |\Pi| = \text{poly}(\lambda, \log |C|) \)
- The verifier runs in time \( \text{poly}(\lambda, |x|, \log |C|) \)

- **SNARK**: Succinct Non-Interactive Argument of Knowledge

- **zk-SNARG/zk-SNARK**: SNARG/SNARK + Zero-Knowledge
Probabilistically Checkable Proof (PCP)

**Prover**

\((x, w)\)

**Verifier**

\((x)\)

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

\[\begin{array}{ccccccc}
0 & 1 & 1 & 0 & 1 & \cdots & 1 \\
\end{array}\]

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

\((x, w)\)

\[
\text{Com}\left(\begin{array}{cccc}
0 & 1 & 1 & 0 \\
1 & \cdots & \cdots & 1
\end{array}\right)
\]

Verifier

\((x)\)

\[i, j, k\]

Open Com(\(\pi_i\)), Com(\(\pi_j\)), Com(\(\pi_k\))
Merkle Tree

Why (computationally) binding?

Collision Resistance of Hash

Can we make it hiding?

Commitment to b

Why (computationally) binding?

 Collision Resistance of Hash

Can we make it hiding?

Commitment to b

\[ c = H(\text{all } b) \]

\[ c = H(b \| r) \]
Is it ZK?

Prover

\[(x, w)\]

\[MT(\text{Com}(01101\ldots1))\]

Verifier

\[(x)\]

\[ij, k\]

Open \(\text{Com}(Ti), \text{Com}(Tj), \text{Com}(Tk)\)

\[\rightarrow\]

ZKP
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**

\[(PK, sk) \leftarrow \text{Gen}(1^\lambda)\]

\[q_1 \in \mathbb{F}_m, \quad C_1 \leftarrow \text{Enc}_{pk}(q_1)\]

\[\vdots\]

\[q_t \in \mathbb{F}_m, \quad C_t \leftarrow \text{Enc}_{pk}(q_t)\]

\[PK, C_1, \ldots, C_t \quad \text{(publish)}\]

**Verifier**

Additively Homomorphic

\[\text{Final Check} (\chi, \text{Lpcp}_p, r_1, \ldots, r_t)\]

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

Common Reference String

$C_1 := g^{q_1}$
$\vdots$
$C_t := g^{q_t}$
$C_T := g^{T_{lpep}}$

Prover

$(x, w)$

$\pi \in \mathbb{F}_m$

$g^{<\pi, q_1>} \rightarrow g^{r_1}$
$\vdots$
$g^{<\pi, q_t>} \rightarrow g^{r_t}$

$g^x g^{T_{lpep}} g^{r_1} \cdots g^{r_t}$

Final Check $(x, T_{lpep}, r_1, \ldots, r_t)$

quadratic function
Bilinear Pairings

Asymmetric pairing

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

\[ e : G_2 \times G_2 \rightarrow G_T \]

\[ e(g^a, g^b) = g_T^{ab} \]

Symmetric pairing:

\[ e : G \times G \rightarrow G_T \]

\[ e(g^a, g^b) = g_T^{ab} \]
Secure Multi-Party Computation

Alice

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

Bob

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 $\Rightarrow$ Information-Theoretic (IT) Security

- PPT bounded $\Rightarrow$ Computational Security