Lecture II
- \( \Sigma \)-protocols for boolean gate constraints

- Non-interactive ZK?
  \( \Rightarrow \) Fiat-Shamir Heuristic
  - Schnorr Signatures
  - HVZK \( \Sigma \)-protocol \( \Rightarrow \) NIZK (ROM)
Towards a Σ-protocol for Circuit-SAT

Recall: Pedersen Commitments

\[ g, h \leftarrow G \]
\[ \text{Commit}(m \in \mathbb{Z}_p, r \in \mathbb{Z}_p) = g^m h^r \]

Say you have 3 commitments \( c_1, c_2, c_3 \). A prover wants to convince verifier that it knows \( m_1, m_2, m_3 \in \{0,1\}^3 \) and \( r_1, r_2, r_3 \in \mathbb{Z}_p \) s.t. \( \forall i \in \{1,2,3\} \ \ c_i = g^{m_i} h^{r_i} \) and \( m_1 \land m_2 = m_3 \)

This corresponds to land you are given in HW3!

Idea: since \( m_1, m_2, m_3 \) are bits, there are only 8 possible combinations of values, and only 4 of these combos are in the language \( \text{Land} \)

So it suffices to prove:

\((m_1 = 0 \ \text{AND} \ m_2 = 0 \ \text{AND} \ m_3 = 0) \)
\[ \text{OR} \]
\((m_1 = 0 \ \text{AND} \ m_2 = 1 \ \text{AND} \ m_3 = 0) \)
\[ \text{OR} \]
\((m_1 = 1 \ \text{AND} \ m_2 = 0 \ \text{AND} \ m_3 = 0) \)
\[ \text{OR} \]
\((m_1 = 1 \ \text{AND} \ m_2 = 1 \ \text{AND} \ m_3 = 1) \)
we know how to do AND/OR of \( \mathbb{Z}_p \) protocols from last time, so we just need to see how to prove that \( a_i \) commits to 0 or 1.

Q: How to show \( m_i = 0 \) or \( m_i = 1 \)?

Recall Schnorr’s Protocol

A Pok for \( \{ (x = (g, h) \in G^2, \text{ we } \mathbb{Z}_p) : g^w = h^3 \} \)

\[
\begin{align*}
    P &: (g, h, w) \\
    r &\leftarrow \mathbb{Z}_p \\
    g^r &\leftarrow c \\
    z &\leftarrow wc + r \\
    g^z &= h^c g^r
\end{align*}
\]

A: To show \( m_i = 0 \), show 
\[
    c = g^m h^r 
\]

→ use Schnorr to show 
\[
    c = h^r \quad \text{“} w \text{”}
\]

\[
    c = g^h \quad \text{“} g \text{”}
\]

To show \( m_i = 1 \), show 
\[
    c = g^m h^r 
\]

→ use Schnorr to show 
\[
    c = h^r \quad \text{“} w \text{”}
\]

\[
    c = h^g \quad \text{“} g \text{”}
\]

\( \Rightarrow \) Works for any truth table!

\( \Rightarrow \) HW: Circuit - SAT
NIZKs

Q: 2 protocols give us 3-message ZK protocols. Can we do better? Can we get 1-message ZK protocols? If so, for which languages?

"Non-Interactive Zero Knowledge (Proofs)"

\[ \pi \]

Suppose we have a complete, sound, ZK, non-interactive proof.

\[ \exists \text{Sim}(x) \rightarrow \pi ', \text{that verifier (efficient distinguisher) can't distinguish from real proof} \]

\[ \forall x \in L \leftrightarrow \exists \pi \ \text{Verify}(x, \pi ) = 1 \leftrightarrow \text{Verify}(x, \text{Sim}(x)) \]

\[ \text{sound/comllete} \]

\[ \text{ZK} \]

\[ \text{a PPT alg for } x \in L(BPP) \]

*Intuition: when proof is 1 message, Sim alg should be able to output the message \( \pi \)

But NIZKs are possible if we change the model

\[ \text{RO Model} \]

\[ \text{CRS} \]

\[ \text{P} \rightarrow \pi \rightarrow \text{V} \]
$N_1$, $ZK$, $Pok$ in ROM with Fiat-Shamir Heuristic

Fiat-Shamir allows us to convert $\sum$-protocol for NP relation $R$ into a NIZKPok in RO model

$\sum$-Protocols

$P((x, w) \in \Pi) \xrightarrow{\text{commitment } t} V(x) \xrightarrow{\text{challenge } c} \xrightarrow{\text{response } z} \{0, 1\}^*$

Properties

1. Completeness: $\forall x \in L, \Pr[\langle P(x, w), V(x) \rangle = 1] = 1$
2. Special Soundness: $\exists$ deterministic efficient $E$ s.t. $\forall$ pairs of accepting $(t, c, z)$ $(t', c', z')$ w/ $c \neq c'$ $(x, E(t, c, z, t', c', z')) \in R$
   * special case of knowledge soundness
3. Special Honest Verifier Zero Knowledge: $\exists$ deterministic efficient $\text{Sim}(x, c) \rightarrow (t, z)$ s.t.
   - $\forall (x, w) \in \Pi \{t, c, z\}: c \notin C; t, z \leftarrow \text{Sim}(x, c) = \{P(x, w), V(x)\}$
   - $\forall x, t, z \leftarrow \text{Sim}(x, c) \rightarrow (t, c, z)$ is an accepting transcript

Notice that $V$ 1) sends only random values to $P$
2) has no secret state

We call this "public coin"
**Fiat-Shamir Idea:** replace verifier's message with the random oracle \( c \leftarrow H(x,t) \in \mathbb{Z}_q \)

**Schnorr:** (Prove knowledge of \( w \) st. \( h = g^w \))

**Schnorr-FS:** (Prove knowledge of \( w \) st. \( h = g^w \) non-interactively)

\[
\begin{align*}
\text{P}(g,h,w) & \quad r \leftarrow \mathbb{Z}_p \quad g^r \\
\text{V}(g,h) & \quad c \leftarrow H(g,h,g^r) \quad g^z \equiv h^c g^r
\end{align*}
\]

**Analysis**
1. **Completeness** is direct
2. **ZK** follows from HVZK of underlying ZK-protocol \( \rightarrow \) **RO** behaves like honest verifier!

**Q:** What does **ZK** mean in **ROM**?

**A:** Simulate \( P \leftrightarrow V \) transcript \& **RO** queries called "programming" the **RO**

**Sim:**
- map \( M : \mathbb{G}^3 \rightarrow \mathbb{Z}_p \)
- \( c \leftarrow \mathbb{Z}_p \)
- \( t, z \leftarrow \text{Sim}_{\text{Schnorr}}((g,h), c) \)
- set \( M[(g,h,t)] \leftarrow c \)
- output \( (t,c,z) \)

on **RO** query \( x \): if \( x \notin M \), set \( M[x] \leftarrow \$ \mathbb{Z}_p \), output \( M[x] \)
3. \( \text{Pok} \)

Q: How do we prove Pok in \( \text{ROM} \)?

**Standard**

\[
P \xrightarrow{\text{mi}} V \quad \text{E can rewind + choose},
\]

\[
P \xleftarrow{\text{ri}} \quad \text{E can rewind + choose}
\]

A: Ext behaves just like protocol extractor, except instead of rewinding & choosing \( V \) messages, extractor rewinds, chooses \( V \) messages, and reprograms \( RO \) for new challenge.

**Schnorr - FS soundness**

\[
E:
\begin{align*}
- & c \neq c' \leftarrow C \\
- & \text{run } P^* \text{ when it queries } \text{RO} \text{ for challenge, give } c \\
- & \text{run } P^* \text{ when it queries } \text{RO} \text{ for challenge, give } c' \\
- & \text{use 2 transcripts + } E_{\text{Schnorr}} \text{ to get witness}
\end{align*}
\]

**Bonus: Signatures**

- simply add \( m \) to the hash and let \( \text{pk} = g^{sk} \)
- \( H: \mathbb{G}_1 \times \mathcal{M} \rightarrow \mathbb{Z}_p \)
- \( \text{Sign}(\text{pk}, \text{sk}, m, g) : \)
  \[
  \begin{align*}
  & r \leftarrow \mathbb{Z}_p^* \\
  & c \leftarrow H(\text{pk}, g, g^r, m) \\
  & z \leftarrow \text{sk} \cdot c + r \\
  & \sigma \leftarrow (z, g^r)
  \end{align*}
  \]
- \( \text{Verify}(\text{pk}, m, \sigma, g) : \)
  \[
  \begin{align*}
  & c' \leftarrow H(\text{pk}, g, g^r, m) \\
  & g^z = \text{pk}^c \cdot g
  \end{align*}
  \]
Notes
- in this specific case, don’t need pk in hash
- could send c, not \( g^r \), and compute \( g^c = g^{2/pk} \)
  and check \( c = H(pk, g, g^r, m) \)
- soundness error is \( \frac{1}{\ell_c} \) so c can be 128 bits
- \( z \) is in \( \mathbb{Z}_q \), which would be 256 bits for EC group
→ total size of signature = 128 + 256 = 384 bits

Compare:
  RSA-FDH = 3072 bits
  BLS : 384 bits (pairing group size)

In practice, ECDSA signatures are widely used;
same idea as Schnorr but worse; why is it used? Patents!

A general perspective:
Fiat-Shamir lifts a \( \Sigma \)-protocol w/ completeness + SHVZK + SKS to a non-interactive ZK-PoK (in the ROM)

It’s also useful for other constant-round public-
coin protocols (and some \( \ell(l) \)-round protocols too!)