Today

- Small ZK Recap
- Proofs of Knowledge
- Schnorr Protocol
- Sigma Protocols
- Variants (AND/OR)
Recap - Zero Knowledge Proofs

Let $L \leq \exists \mathcal{O}, 1^*$ be an NP-language.

A ZK Proof System is a tuple of efficient interactive algs $\langle P, V \rangle$ s.t. they satisfy

**Properties**

1. **Completeness:** $\forall x \in L$, $Pr[\langle P(x, w), V(x) \rangle = 1] = 1$

2. **Soundness:** $\forall x \not\in L$, $\forall P^*$, $Pr[\langle P^*, V(x) \rangle = 1] \leq \text{neg}(\lambda)\downarrow$

3. **ZK:** $\exists \text{ PPT Sim}, \forall \text{ PPT } V^*$,
   
   $\exists \text{ View}_{V^*}(\langle P(x, w), V(x) \rangle) \approx \exists \text{ Sim}_{V^*}(x)$

   "malicious verifier ZK"

**Honest Verifier ZK (HVZK):** $\exists \text{ PPT Sim},$

$\exists \text{ View}_{V}(\langle P(x, w), V(x) \rangle) \approx \exists \text{ Sim}(x)$

honest verifier

no oracle access

**Proofs of Knowledge:**

Soundness (informally): the verifier is convinced that $x$ (a graph $G$) is in a language $L$ (HAM CYCLE Graphs).

However, in many cases, we want to verify that a prover actually "knows" a witness (a HAM CYCLE)

We would like a Proof of Knowledge (PoK).
i.e. if \( V \) accepts w.h.p., then \( P \) must know a witness \( w \).

Does the soundness property imply a proof of knowledge?

\[ \text{No! Consider the following NP-Relation,} \]

\[ R_{\text{composite}} = \{ (N, p) \mid p | N \land p \notin \{1, N \} \} \]

- Verifying a number is not prime is not the same as factoring

**Intuition:** How can we guarantee that a malicious prover \( p^* \) knows a \( w \) s.t. \((x, w) \in R\)?

**A trivialPok (attempt 1):**

\[
\begin{align*}
& \quad P(x, w) \\
\xrightarrow{W} & \quad V(x) \\
\end{align*}
\]

Check \((x, w) \in R\)?

\(\text{Eff since } R \text{ is an NP relation}\)

**Issue:** This cannot be \( ZK \), as the simulator would have to efficiently output a valid witness.

**Attempt 2**

\[
\begin{align*}
& \quad P(x, w) \\
\quad \xrightarrow{m_i} & \quad V(x) \\
\quad \leftarrow r_i & \quad C
\end{align*}
\]

Given messages \( \{m_i\} \) from \( p^* \), we can compute a satisfying witness ... SAME ISSUE!
What if instead we can interrogate the prover multiple times?

Let $E$ be a ppt alg called an extractor

$E$ has black box access to $P^*$, can interact, “rewind” it to previous rounds.

**What does rewinding mean?**

An interactive ppt alg $P^*$ can be described as a series of next message functions: [BG92]

Let $p \in \{0,1\}^*$ represent the prover’s private randomness

$$
\begin{align*}
&\begin{array}{c}
P^* \\
\hline
V(x) \\
\hline
r_1 \xrightarrow{m_1} \quad \leftrightarrow \quad p^*_1(p) \rightarrow r_1 \\
\vdots \\
m_n \xrightarrow{r_{n+1}} \\
\end{array} \\
&\begin{array}{c}
P^*_2(p, m_1) \rightarrow r_2 \\
\vdots \\
p^*_n(p, m_1, \ldots, m_n) \rightarrow r_{n+1}
\end{array}
\end{align*}
$$

We define $E^P$ as the extractor that has oracle access to the functions $E_{P_i}^*(p, \cdot)$; Furthermore, we allow the extractor to force a resampling of the initial prover randomness $p$. 
(P, V) is a PoK for NP relation R with knowledge error Η if
\[ \exists \text{ PPT } \quad \forall x, \forall p^*, \]
\[ \Pr \left[ (x, w) \in R : w \leftarrow E \left( p^* \left( x \right) \right) \right] \geq \Pr \left[ \left< p^*, v \left( x \right) \right> = 1 \right] - \delta \]

**Schnorr's Protocol**

Let G be a group of prime order q with a generator g.
Define \( R_{\text{Dlog}} = \{ (h, x) \mid h = g^x \} \). Note \( L(R) = G \) is a trivial language. Thus, soundness is a trivial property to satisfy, but is PoK?
- Prover wants to convince V(h) that it knows the discrete log of h.

\[
\begin{align*}
\text{P} \left( x \in \mathbb{Z}_q, h = g^x \in G \right) & \quad u = g^r \\
\text{r} & \quad \text{c} \quad \text{z} = r + cx \\
\text{V} \left( h \right) & \quad \text{Output} \quad g^z = u \cdot h^c
\end{align*}
\]

Claim: Schnorr's Protocol is an honest-verifier ZK-PoK of DLog.

Completeness
\[ g^z = g^{r+cx} = g^r (g^x)^c = u \cdot h^c \]
HVZK:

Simulator runs the protocol in "reverse":

\[ \text{Sim}(\ h) \]

1) sample \( z \leftarrow \mathbb{Z}_q \)
2) sample \( c \leftarrow \mathbb{Z}_q \)
3) set \( u := g^z \cdot h^{-c} \)
4) Output \((u, c, z)\)

Can we get malicious ZK?

Issue: malicious verifiers challenge \( c \) may not be uniform random so strategy above of sampling \( z \) first no longer works.

Folklore Result: to get full ZK, have the verifier commit to their challenge before seeing \( u \). However, this introduces an additional round of communication.  

[Lindell: errata-ZK-sigma]
Proof of Knowledge

Suppose $P^*$ convinces an honest verifier $V(h)$ with probability $\epsilon = 1$.

Intuition: Let us rewind the prover to operate on different challenges

\[ \begin{array}{c}
(1) \\
P^* & \rightarrow & u \\
\leftarrow & c_i \leftarrow Z_q \\
\rightarrow & z_i \\
g^{z_i} = u \cdot h^{c_i} \\
\end{array} \quad \begin{array}{c}
(2) \\
P^* & \rightarrow & u \\
\leftarrow & c_x \leftarrow Z_q \\
\rightarrow & z_x \\
g^{z_x} = u \cdot h^{c_x} \\
\end{array} \]

Since we assumed $\epsilon = 1$, then $(u, c_i, z_i)$ and $(u, c_x, z_x)$ are two accepting transcripts. Thus, $g^{z_i} = u \cdot h^{c_i}$ and $g^{z_x} = u \cdot h^{c_x}$

\[ g^{z_i - z_x} = h^{c_i - c_x} \]

W.H.P. $c_i \neq c_x$:

\[ g^{\frac{z_i - z_x}{c_i - c_x}} = h \quad \Rightarrow \quad x = \frac{z_i - z_x}{c_i - c_x} \] is the $	ext{Dislog}$ of $h$.

More formally:

\[ \mathcal{E}_{P^*} \]

1) Run $P^*$ to get $u$.
2) Send $c_i \leftarrow Z_q$, and receive $z_i$.
3) Rewind $P^*$, send $c_x \leftarrow Z_q$, and receive $z_x$.
4) If $c_i = c_x$, output fail. O/W output $x = \frac{z_i - z_x}{c_i - c_x}$.
Analysis

\[ \Pr[(h, x) \in R_{\text{play}} : x \leftarrow E^*(h)] = 1 - \frac{1}{\ell} \geq \Pr[\langle p^*, v \rangle(h) = 1] - \frac{1}{\ell} \]

Thus, \( K = \frac{1}{\ell} \).

We assumed \( \Pr[\langle p^*, v \rangle(h) = 1] = 1 \), but more generally what about \( \Pr = \varepsilon \)?

Rewinding Lemma (BS 19.2)

If \( p^* \) succeeds with probability \( \varepsilon \), then using the "rewinding lemma", we can argue the extractor obtains two accepting transcripts (with \( c_1 \neq c_2 \)) with prob at least \( \varepsilon^2 - \varepsilon/\ell \).

Sigma Protocols (Z-Protocols)

More broadly, the Schnorr Protocol belongs to a family of three message protocols called Sigma Protocols.

\[ P(x, w) \]

\[ \begin{array}{c}
\text{t "commitment"} \\
\longrightarrow \\
\text{c "challenge"} \\
\leftarrow \\
\text{z "response"} \\
\longrightarrow \\
\text{v(x)}
\end{array} \]

\[ \begin{array}{c}
c \leftarrow C \\
\text{Output 0/1 deterministically from } (x, t, c, z)
\end{array} \]
Properties:

1) Perfect Completeness
2) Special Soundness: \( \exists \) extractor \( E \) that given two accepting transcripts \( (t, c, z), (t, c', z') \) with \( c \neq c' \) outputs \( w \) s.t. \((x, w) \in R \).

\( \implies \) PoK (can you see how?)

3) Special Honest Verifier ZK: \( \exists \) efficient \( \text{Sim}(x, c) \rightarrow (t, z) \) s.t. \((t, c, z)\) is an accepting transcript for \( x \).

Additionally, \( \forall (x, w) \in R \),

\[ \{ (t, c, z) : c \leftarrow c', (t, z) \leftarrow \text{Sim}(x, c) \} \approx \{ \text{View}_v(P(x, w), V(x)) \} \]

\( \star \) In literature, you may see \( \mu \)-round Sigma Protocols with \((k_1, k_2, \ldots, k_\mu)\)-special soundness. The Schnorr Protocol is a 1-round Sigma Protocol with 2-special soundness

referring to number of distinct challenges needed at round \( i \leq \mu \).

AND Proofs

Let \((P_0, V_0)\) and \((P_1, V_1)\) be Sigma Protocols for relations \( R_0 \) and \( R_1 \) respectively that use the same challenge space \( C \). Define the following AND-Relation \( R_{\text{and}} \):

\[ R_{\text{and}} := \{((x_0, x_1), (w_0, w_1)) \mid (x_0, w_0) \in R_0 \land (x_1, w_1) \in R_1 \} \]
We can construct a Sigma Protocol \((P,V)\) for \(R_{\text{and as follows:}}\)

\[
\begin{align*}
P((x_0, x_1), (w_0, w_1)) & \quad V((x_0, x_1)) \\
\text{Run } P_0(x_0, w_0) & \rightarrow t_0 \\
P_1(x_1, w_1) & \rightarrow t_1 \\
\text{Feed } c \text{ to } P_0, P_1 \\
\text{to obtain } (z_0, z_1) \\
\end{align*}
\]

\[
\begin{align*}
(t_0, t_1) & \quad (z_0, z_1) \\
c & \quad \text{Accept if both } \\
V_0(x_0, (t_0, c, z_0)) \text{ and } \\
V_1(x_1, (t_1, c, z_1)) \text{ accept}
\end{align*}
\]

**Proof Sketch**

**Special Soundness:**

Given two accepting transcripts \(c \neq c'\):

\[
((t_0, t_1), c, (z_0, z_1)) , ((t_0, t_1), c', (z_0, z_1))
\]

Run extractors \(E_0(x_0, (t_0, c, z_0), (t_0, c', z_0'))\) \(\rightarrow w_0\)

\(E_1(x_1, (t_1, c, z_1), (t_1, c', z_1'))\) \(\rightarrow w_1\)

Output \((w_0, w_1)\)

**HVZK**

\[
\begin{align*}
\text{Sim}((x_0, x_1), c): \\
(t_0, z_0) \leftarrow \text{Sim}_0(x_0, c), \ (t_1, z_1) \leftarrow \text{Sim}_1(x_1, c) \\
\text{Output } ((t_0, t_1), c, (z_0, z_1))
\end{align*}
\]

\(\approx \) Only a sketch: need to argue extractor is correct
and simulator distribution is indistinguishable
**Proof Sketch**

### Special Soundness

Given \( ((t_0, t_1), c, (c_0, z_0, z_i)) \), \( ((t_0, t_1), c', (c_0', z_0', z_i')) \).

Define \( c_1 := c \oplus c_0 \) and \( c_1' := c' \oplus c_0' \). Notice since \( c' \neq c \),
then either \( c_0 \neq c_0' \) or \( c_1 \neq c_1' \).

\[ c_1 \oplus c_1' = (c \oplus c') \oplus (c_0 \oplus c_0') \]

If \( c_0 \neq c_0' \):

Output \( (0, E_0(x_0, (t_0, c_0, z_0), (t_0, c_0', z_0'))) \)

Else:

Output \( (1, E_1(x_1, (t_1, c_1, z_i), (t_1, c_1', z_i'))) \)
$H_U \vdash \exists x_0 \exists x_1, c$

$Sim((x_0, x_1), c)$

- $c_0 \in C$, $c_1 \in C \oplus C_0$
- $(t_0, z_0) \in Sim_0(x_0, c_0)$
- $(t_1, z_1) \in Sim_1(x_1, c_1)$

Output $((t_0, t_1), c, (c_0, z_0, z_1))$

**Summary**

Today, we learned

- What are Proofs of Knowledge
- Example of Pok is Schnorr Protocol (Proof of Dlog)
- Schnorr Protocol belongs to Sigma Protocols
- AND / OR Prots for combining Sigma Prots