In this lecture we will cover the following topics:

- Witness Hiding Σ-protocols
- Alternative Construction of ZK Argument
- Concurrent ZK
- NIZK in RO-model
- Σ-protocol for NP

1 Witness Hiding Σ-protocols

Definition 1 A protocol for a relation $R(x, y)$, with instance generation algorithm $Gen(1^λ) \rightarrow (x, y)$, is called Witness Hiding (WH), if

$$\forall V^* \ Pr[V^* \text{ using } View_{V^*}(V^*(y) \leftrightarrow P(x)) \text{ computes } x' \text{ s.t. } R(x', y) = 1] = negl(λ)$$

Lemma 2 If $R$ is hard w.r.t. $Gen$, then any ZK protocol for $R$ is WH.

Proof: [Sketch] Let $Π$ be a ZK protocol for $R$. Then, by ZK property, there exists a simulator $Sim^{V^*}$, s.t.

$$View_{V^*}(V^*(y) \leftrightarrow P(x)) \approx View_{V^*}(V^*(y) \leftrightarrow Sim^{V^*}).$$

This implies that if $V^*$ using $View_{V^*}(V^*(y) \leftrightarrow P(x))$ computes an $x'$ s.t. $R(x', y) = 1$, then also using $View_{V^*}(V^*(y) \leftrightarrow Sim^{V^*})$ $V^*$ must compute an $x''$ s.t. $R(x'', y) = 1$. However, the later contradicts the hardness of $R$.

Now, let us see some constructions of WH Σ-protocols:

Construction 1 Use of Σ-protocols with small challenge space $S$ to derive a ZK protocol and thus a WH protocol.
Construction 2  Use of Σ-protocols for SPR functions.

Lemma 3  If Π is a Σ-protocol for an SPR function f, then Π is also WH.

Proof: [Sketch] Assume that V* using ViewV*(V*(y) ↔ P(x)) computes x', s.t. f(x') = y(= f(x)).

Since Π is WI, V* learns nothing about x. Therefore, with good probability x' ≠ x. However, the latter contradicts the fact that f is SPR.

Example: Okamoto’s Scheme with f(x_1, x_2) = g_1^{x_1} g_2^{x_2}.

Construction 3  OR of two Σ-protocols of the same relation

Lemma 4  If Π is a Σ-protocol for a hard relation R, then Π_{OR} = Π_1 ∨ Π_2 (OR proof for two independent instances of R) is WH.

Proof: The proof is the same as above using the fact that Π is WI.

2 Alternative Construction of ZK Argument

Theorem 5  Let Σ be a Σ-protocol for L and Π_{TC} a WH Σ-protocol for Trapdoor Commitment. Then the following protocol is a ZK argument for L.

\[
\begin{align*}
&P(x) & V(y) \\
&\leftarrow CK & (CK, TK) \leftarrow Gen \\
&\Pi_{TC} \\
&\leftarrow & \rightarrow \\
&(\gamma, d) \leftarrow Com_{CK}(a) & V \text{ proves the knowledge of } TK \\
&\gamma \rightarrow c \\
&\leftarrow (d, z) \\
\end{align*}
\]

Proof: [Sketch] To prove ZK, we have to define the simulator Sim_{V*} for every V*:
To prove soundness we will use the WH property of $\Pi_{TC}$. If $P^*$ cheats with non negligible probability, then $P^*$ brakes the binding property. However, this would imply that $P^*$ would get a witness $TK$ for the trapdoor commitment. However, this is a contradiction since $\Pi_{TC}$ is WH.

3 Concurrent ZK

Here, we want to capture the case, when a single prover runs the same protocol, possibly on different instances, with many verifiers simultaneously, but not necessarily in parallel. The verifiers are considered to be coordinated by a single malicious verifier, who is now able, for instance, to make some verifiers delay their responses to the prover.

A major problem with concurrent ZK is that rewinding does not work. Namely, to prove that a ZK protocol, which is secure with a single verifier, is also secure in the concurrent case, we might have problems if the simulator needs rewinding.

Example: Suppose that we have a 4-round ZK proof. Let $V_1, \ldots, V_n$ be the verifiers in the concurrent case. Then $V^*$ is allowed to schedule the runs in the following way:
To prove that this protocol is ZK we have to construct a simulator. Suppose, that in the case of one verifier the simulator does not need to rewind during the first two rounds, but it needs rewinding after the third round. The simulation in the concurrent case of the first two rounds with $V_1, \ldots, V_{n-1}$ and all rounds with $V_n$ are straightforward. However, when simulating rounds 3 and 4 with $V_{n-1}$ the simulator might need to rewind with $V_{n-1}$ from round 1, but when going to round 3 again, the simulator must simulate the interaction with $V_n$ again. In the same way, when simulating rounds 3 and 4 with $V_i$, the simulator might need to simulate again the whole interaction with $V_i, \ldots, V_n$. In the end, if $R(n)$ is the number of rewinding in the case of $n$ verifiers, we have that $R(n) = R(n-1) + 1 + R(n-1) = 2R(n-1) + 1$, which implies that $R(n) = \Omega(2^n)$. Therefore, the total simulation might need exponential time. Of course, there might be a smarter way to simulate, but we just want to show intuitively why the simulator might be hard to construct.

**Definition 6** \( \Pi \) is BB concurrent ZK if for every $V^* = (V_1^*, \ldots, V_{\text{poly}(\lambda)}^*)$ there exists a simulator \( \text{Sim} \) running in time \( \text{poly}(\lambda) \) such that

\[
\text{View}_{V^*}(\text{Sim}^{V^*} \leftrightarrow V^*) \approx \text{View}_{V^*}(V_1^*, \ldots, V_{\text{poly}(\lambda)}^* \leftrightarrow P)
\]

**Theorem 7** BB Concurrent ZK requires $\Omega(\log \lambda)$ rounds.

This bound can be matched, but the construction is tricky, thus will not be covered.

We will construct concurrent ZK in the Common Reference String (CRS) model. (c.f. public parameter model)
**View 1** We assume that there exists an independent algorithm $Setup \rightarrow PK$. Both the prover and the verifier get $PK$ and the requirement is

$$(PK, View^*_V(V^* \leftrightarrow P)) \approx Sim^V$$

Namely, the simulator can sample a new $PK'$ and use this instead of $PK$.

**View 2** We assume that there exists an independent algorithm $Gen \rightarrow (PK, TK)$. Both the prover and the verifier get only $PK$, but the simulator also gets $TK$. The requirement is

$$(PK, View^*_V(V^* \leftrightarrow P)) \approx (PK, Sim^V(TK))$$

Actually, we have the following implications:

\[
\text{Public parameter model} \Rightarrow \text{View 2} \Rightarrow \text{View 1},
\]

which means that if a protocol is proven secure in one model, then it is also secure in the model to its right.

In order to construct a Concurrent ZK protocol we will use the protocol of theorem 5 and the following intuitive claim:

**Claim 8** If a ZK protocol $\Pi$ is straightline simulatable, namely the simulator which proves the ZK property does not use rewinding, then $\Pi$ is also Concurrent ZK.

**Theorem 9** The following protocol is straightline simulatable and thus concurrent ZK.

\[
Gen \rightarrow (CK, TK) \\
CRS = CK \\
P(x) \\
V(y) \\
(\gamma, d) \leftarrow Com_{CRS}(a) \\
\gamma \rightarrow c \\
\gamma \leftarrow (d, z) \\
\rightarrow
\]

Another construction for concurrent ZK in the CRS model is to use NIZK, which are trivially straightline simulatable as there is no malicious $V^*$. However, note that this way we lose deniability and non-transferability.

**4 NIZK in RO model**

In the random oracle model all parties have access to the same oracle $H$, which on input $x$ returns a random value $y$, but every time it is queried on $x$ it returns $y$, namely it is a random function. The advantage in the RO model is that the $Sim$ has complete control over the RO and can actually program it. This means that whenever $V^*$ asks for the $H(z)$ for some $z$, the $Sim$ can return to $V^*$ a different value (which follows an appropriate distribution) from what the random oracle would return.
Definition 10 A protocol is ZK in the random oracle model, if:

1. Completeness is satisfied for all $H$
2. Soundness is satisfied with high probability over random $H$
3. (ZK) There exists a $\text{Sim}^{V^*}$ s.t. $(H, \text{View}_{V^*H}(V^*H \rightarrow P)) \approx \text{Sim}^{V^*}$

The simulator has the ability to extract $V^*$’s queries to the random oracle and then, as we said, to program $H$.

Fiat-Shamir paradigm

Theorem 11 Assume that $L$ has a $\sigma$-protocol with challenge space $\{0, 1\}^\ell$, with $\ell = \omega(\log \lambda)$. Then $(y, (a, z))$, where $z$ is computed with $c = H(a, y)$, is a NIZK Proof of Knowledge in the random oracle model.

Proof:[Sketch, similar to signatures]

To prove ZK we define the simulator $\text{Sim}$

$$\text{Sim}(y) : HV\text{Sim}(y) \rightarrow (a, c, z)$$

program $H(a, y) = c$

Note that $c$ follows the uniform distribution.

To prove PoK we define $\text{Ext}$ similarly. $\text{Ext}$ gets $(y, (a, z))$ from $P^*$, reprograms $H(a, y) = c'$ and hopes to get $(y, (a, z'))$, where now $c' \neq c$. \hfill $\square$

5 $\Sigma$-protocols for $NP$

We are going to to construct a $\Sigma$-protocol for HC, the language of graphs with Hamiltonian Cycle. Below $G$ is a graph with $n$ vertices and $e_{ij} = 1$ if there exists an edge between vertices $i$ and $j$, otherwise $e_{ij} = 0$. Moreover, $S_n$ is the set of all permutations $\pi : \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$.

$P(\text{cycle})$

\[
\begin{align*}
\pi &\xleftarrow{\$} S_n \\
G' &= \pi(G) \\
\delta_{ij} &= \text{Com}(e_{ij}) \\
\end{align*}
\]

$V(G)$

\[
\begin{align*}
\delta_{ij} &\xrightarrow{c} c \leftarrow \{0, 1\} \\
\uparrow \text{openings of all } \delta_{ij}, \pi \\
\text{if } c = 0 \\
\text{if } c = 1 \quad \text{openings of all } \delta_{ij} \text{ corresponding to cycle in } G'
\end{align*}
\]
Theorem 12  The above is a computational $\Sigma$-protocol for HC.

Proof:
To prove special soundness we just observe that if we have openings for both $c = 0$ and $c = 1$, then we can derive a hamiltonian circle in $G$. Moreover, the binding property of the commitment guarantees the consistency of the openings.
To prove HVZK we construct the following simulator

\[
Sim^*(c): \begin{align*}
& \text{if } c = 0 \text{ choose } \pi \leftarrow S_n, \text{ compute } G' \\
& \text{commit to all } \delta_{ij} \\
& \text{if } c = 1 \text{ choose } G' \in HC \text{ and commit to } G'
\end{align*}
\]

\[\square\]

Another protocol for HC  Firstly, we observe that in the above protocol $P$ proves that $C_n \subseteq G$, namely that $C_n$ (a cycle graph with $n$ vertices) is a subgraph of $G$. However, the same protocol can be used for any graph $C$, namely $C \subseteq G$. But proving $C \subseteq G$ is equivalent to proving the dual statement, $\overline{G} \subseteq \overline{C}$ (where $\overline{G}$ is the complement graph of $G$, namely if $\overline{e}_{ij} = 1$ whenever there is an edge between vertices $i$ and $j$ in $\overline{G}$ then it holds that $\overline{e}_{ij} = 1 - e_{ij}$). Now the protocol will be as follows:

\[
P(cycle) \quad V(G)
\]

\[
\begin{array}{c}
\pi \leftarrow S_n \\
C_n' = \pi(C_n) \\
\delta_{ij} = Com(d_{ij}')
\end{array}
\quad \quad
\begin{array}{c}
\delta_{ij} \\
c \leftarrow \{0, 1\}
\end{array}
\]

\[
\begin{array}{c}
\text{if } c = 0 \quad \text{openings of all } \delta_{ij}, \pi \\
\text{if } c = 1 \quad \text{openings of all } \delta_{ij} \text{ corresponding to } G \text{ in } C_n'
\end{array}
\]

where $d_{ij} = 1$ iff there is an edge between vertices $i$ and $j$ in $C_n'$. The proof that this protocol is a $\Sigma$-protocol is analogous to the previous case. A nice property of this protocol, which will be useful later, is that the first two moves are independent of the instance. They only depend on $C_n$, which is fixed, and the randomness used.

Corollary 13  If Commitments exist, then

- There exist ZK proofs for all NP
- There exist Trapdoor Commitments
- We have NIZK in the RO model