Outline

Commitments

Zero Knowledge
Commitments
Pedersen Commitments

- Public parameters: \((p, g, h)\)
  - \(p\): large prime (1024 bit)
  - \(g\): generator
  - \(h\): \(g^a\) for hidden \(a\)

- Protocol
  - To commit to \(x\), \(C\) chooses random \(r\) and sends \((g^x h^r)\) to \(R\).
  - To open, \(C\) sends \(x\) and \(r\) to \(R\).

- Benefits:
  - One can prove many things about the committed value without opening it
Pedersen Commitments

• Unconditionally hiding
  • Given a commitment $c$, every value $x$ is equally likely to be the value committed in $c$.
  • For example, given $x, r$, and any $x'$, there exists $r'$ such that $g^x h^r = g^{x'} h^{r'}$, in fact $r' = (x-x') a^{-1} + r \mod q$.  

Given $c$, not even an adv. with unbounded run-time can find $x$.  

\[ g^x h^r = g^x (g^a)^r = g^{x+ar} \]

For an arbitrary $x'$, \exists $r' = (x-x') + r \mod q$ s.t. $g^{x'+ar} \cdot g$
Pedersen Commitments

• Computationally binding
  
  • Suppose committer sent $g^x h^r \mod p$ for some $(x, r)$
  
  • Now it finds $x' \neq x$ and $r'$ such that $c = g^{x'} h^{r'}$.
  
  • This means that the sender `knows’’ $\log_g(h) = (x' - x) \cdot (r - r')^{-1}$.
  
  • This means: assuming DL is hard, the sender cannot open the commitment to a different value.
Application: Coin Tossing

- Alice and Bob want to decide on something by tossing a coin over a phone. How to do this securely?

- Solution: Alice commits to a random bit \( b_A \leftarrow \{0, 1\} \), and sends \( \text{Com}(b_A; r) \) to Bob.

- Bob selects a random bit \( b_B \leftarrow \{0, 1\} \) and sends it to Alice.

- Alice decommits \( b_A \).

- Alice and Bob output \( b_A \oplus b_B \).
Zero-Knowledge
Real World

NP Statement \( x \)
Witness that \( x \) is true

\[ \text{com}(t; r): \text{Prove } t > 0 \text{ w/o revealing } t. \]
\( x \in \text{NP language } L \)
if \( \exists w \) s.t.
\( R_L(x, w) = 1 \)
where \( R_L \) is an efficiently computable relation.

\[ L = \{ c : f(t, r) \text{ s.t. } t > 0 \text{ and } c = \text{com}(t; r) \} \]
Real World

Prover

NP Statement $x$
Witness that $x$ is true

Verifier

Didn't learn witness

Outputs
view

: cannot output secrets about statement (e.g., "t" in our prev. example)
Real World

Prover

NP Statement \( x \)

Verifier

Didn't learn witness

Ideal World (Proof)

Simulator

NP Statement \( x \)

No witness

Witness that \( x \) is true

Outputs view

Any information in this view
Real World

Prover

NP Statement \( x \)

Witness that \( x \) is true

Verifier

Outputs view

Ideal World (Proof)

Simulator

NP Statement \( x \)

No witness

Verifier

Didn't learn witness

Output view
Real World

Prover

Verifier

Witness that \( x \) is true

Outputs view

Ideal World (Proof)

Simulator

Verifier

NP Statement \( x \)

Didn't learn witness

NP Statement \( x \)

Didn't learn witness

Outputs similar view

Trying to assert:

\[
\begin{align*}
\lambda \in (t, r) & \quad \exists c : c = com(t, r) \text{ s.t. } t > D \\
\end{align*}
\]
Real proof hides all predicates of witness that are hard to compute given just $x$.

Real World

Verifier

Simulator

Verifier

Ideal World (Proof)

Outputs similar view

Can't distinguish the two views of $V$. $orall$ p.p.t. $D, \Pr[D(\text{real world view of } V) = 1] - \Pr[D(\text{ideal world view of } V) = 1] = \text{negl.}$
Graph Isomorphism
Graph Isomorphism

Prover

$X = (A, B)$

Verifier

$\eta : 1 \rightarrow a$

$5 \rightarrow g$

$\text{Knows } \eta \text{ s.t. } A = \eta (B)$
Graph Isomorphism

Prover

\[ X = (A, B) \]

Verifier

\[ G = \varphi(A) \]

"Soundness"

Has soundness error \[ \frac{1}{2} \].
Graph Isomorphism

**Prover**

\[ X = (A, B) \]

\( A = \eta(B) \)

\( G = \varphi(A) \)

\( G \subseteq \mathbb{C} \)

\( \Pi_1(A) = G \), \( \Pi_2(B) = G \)

\( A = \Pi_1^{-1}(G) = \Pi_1^{-1}(\Pi_2(B)) \)

\( \Pi_1^{-1}(\Pi_2(.)) \) is \( \eta \).

**Verifier**

\( G \)

\( c = A \) or \( B \)

\( \Pi \cdot \Pi(c) = G \)

\( (G, c, \Pi \cdot \Pi(c)) = G \)
Graph Isomorphism

Simulator

\[ X = (A, B) \]

Verifier

Knows \( c \) in advance

When Sim guesses \( A \) then \( c \neq B \)

(end vice versa)

This cannot happen because \( G \) hides \( (A/B) \).

If \( A \) and \( B \) are isomorphic,

then \( \text{dist}(\varphi(A)) = \varphi(B) \)

for random \( \varphi \)

Does not know \( \pi \)!

\[ \pi \text{ s.t. } \pi(c) \neq G \]

if \( c \neq \text{guess} \) we are done! \( \pi \neq \varphi \).
Graph Isomorphism

Simulator
\[ \mathcal{X} = (A, B) \]

Verifier

Doesn't know \( \eta \).

Runtime of Simulator
\[ O(n \cdot k) \]

K knows c in advance

\[ k = 256 \]
Graph Isomorphism

**Simulator**

\[ X = (A, B) \]

Knows \( c \) in advance

\[ G = \varphi(c) \]

**Verifier**

\[ c = A \text{ or } B \]
3-coloring

Color all vertices with only three colors (R, G, B) such that no edge should connect two vertices of the same color.
3-coloring

Prover

Verifier
3-coloring

Prover

Verifier
3-coloring

Prover

Verifier
3-coloring

Prover

Verifier
3-coloring

Simulator

Verifier
3-coloring

Simulator

Verifier
3-coloring

Simulator

Verifier