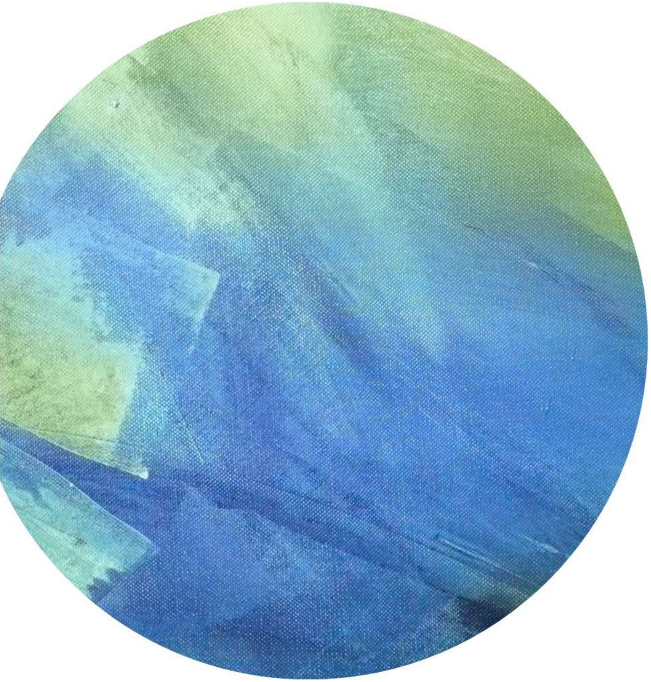
Lecture 14



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

$$\boxed{PP : (p, g, h = g^a)}$$

$$com(x; r) = g^x \cdot h^r$$

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 $r' = (x-x')a^{-1} + r \bmod q$.

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

commitment string

$$\underline{\underline{g^x \cdot h^r}} = g^x \cdot (g^a)^r = g^{\boxed{x+ar}}$$

For an arbitrary $\underline{x'}$, $\exists\ \underline{r'} = \frac{(x-x')}{a} + r$ s.t. $g^{x'+ar'} = g^{x+ar}$

Pedersen Commitments

Suppose committer outputs c, and 2 openings (x, r), (x', r')

$$c = g^{x+ar} = g^{x'+ar'}$$

- 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.

then $x + ar = x' + ar'$

$\Rightarrow x - x' = a(r' - r)$

$\Rightarrow a = \frac{x - x'}{(r' - r)}$

$$g^x = \frac{c}{h^r}$$

Application: Coin Tossing [Blum '86]

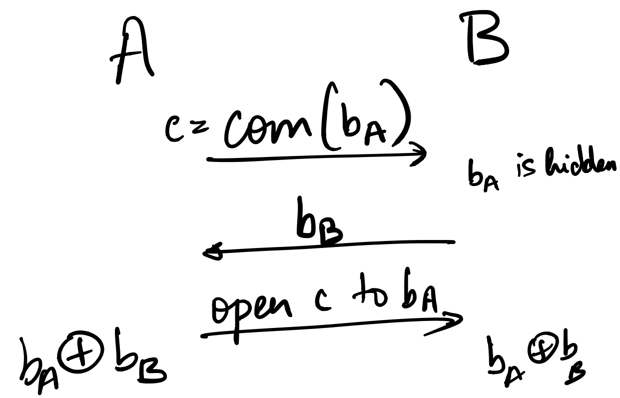
- Alice and Bob want to decide on something by tossing a coin over a phone. How to do this securely?

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

- Bob selects a random bit $\boxed{b_B \leftarrow \{0, 1\}}$ and sends it to Alice $\boxed{g, h}$

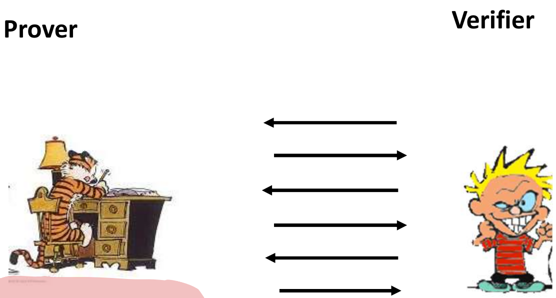
- Alice decommits b_A

- Alice and Bob output b_A xor b_B



Zero-Knowledge

Real World



NP Statement **x**

Witness that **x** is true

$\boxed{\text{com}(t;r)\text{: Prove } t > 0 \text{ w/o revealing } t.}$

$x \in$ NP language L

if $\exists\, w$ s.t.

$$R_L(x, w) = 1$$

where R_L is an efficiently computable relation.

$$L = \{ c : \exists (t, r) \text{ s.t } t > 0 \text{ and } c = \text{com}(t;r) \}$$

Real World

Prover

NP Statement **x**

Witness that **x** is true

challenges $\longleftarrow$

answers $\longrightarrow$

Verifier

Didn't learn witness

Outputs view

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

Real World

Ideal World (Proof)

Prover

NP Statement **x**

Witness that **x** is true

Verifier

Didn't learn witness

Outputs view

Any information in this view

Simulator

NP Statement **x**

No witness

Real World

Prover

NP Statement **x**

Witness that **x** is true

Verifier

Didn't learn witness

Outputs view

Ideal World (Proof)

Simulator

NP Statement **x**

No witness

Verifier

Output view

Real World

Prover

NP Statement **x**

Witness that **x** is true

Verifier

Didn't learn witness

Outputs view

$x = c$

$w = (t, r)$

Trying to assert:

$c = com(t; r)$ s.t. $t > 0$

Ideal World (Proof)

Simulator

NP Statement **x**

No witness

Verifier

Didn't learn witness

Outputs similar view

$x = c$

$\cancel{w = (t, r)}$

Real proof hides all predicates of witness that are hard to compute given just x.

Real World

Honest **Prover**

$\mathcal{C}$

NP Statement **x**

Witness that **x** is true

G_0

c_0

π_0

G_1

c_1

Verifier

Didn't learn witness

Outputs view

Ideal World (Proof)

Simulator

NP Statement **x**

No witness

G_0

c_0

π_0

Verifier

Didn't learn witness

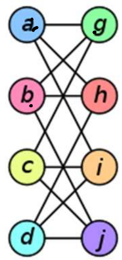
Outputs similar view

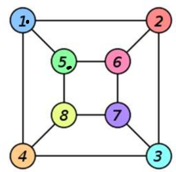
Cannot distinguish the two views of V.

$\forall$ P.P.T. D,

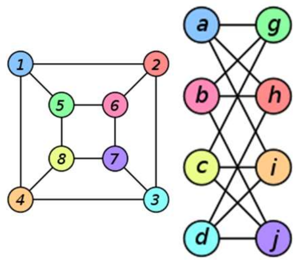
$$\left| \Pr\left[D(\text{Real world view of } V) = 1\right] - \Pr\left[D(\text{Ideal world view of } V) = 1\right] \right| \leq \text{negl}.$$

Graph Isomorphism





Graph Isomorphism



Prover

X = (A, B)

Knows η s.t.
A = η (B)

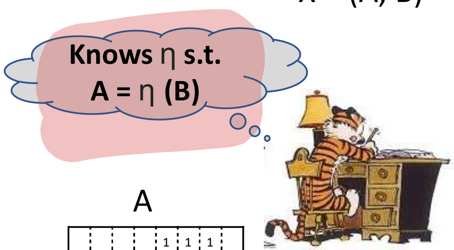
$\eta: 1 \leftrightarrow a$
$5 \leftrightarrow g$
$\vdots$

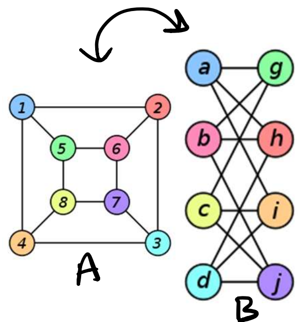
Verifier

Graph Isomorphism

Prover

X = (A, B)





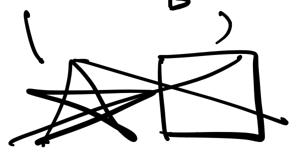
Verifier

A

				1	1	1	
				1	1		1
				1		1	1
					1	1	1
1	1	1					
1	1		1				
1		1	1				
	1	1	1				

$G = \varphi(A)$ →

	1		1		1		
1		1				1	
	1			1			1
		1		1		1	
1	1						1
			1		1		1
		1		1	1		
1			1			1	



$G = \varphi(A)$

$\xrightarrow{\quad G \quad}$

$c \leftarrow \{A, B\}$ each w.p. $1/2$ $\longleftarrow$

sends π s.t. $\pi(c) = G$ $\longrightarrow$

(G, π)

"Soundness".

Has soundness error $\frac{1}{2}$.

Graph Isomorphism

$\pi_1(A) = G \;,\; \pi_2(B) = G$

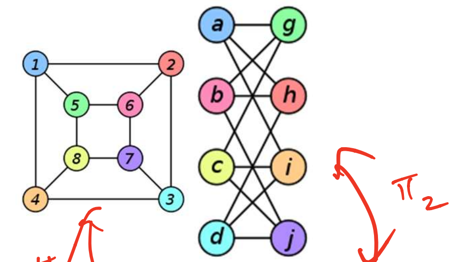
$A = \pi_1^{-1}(G) = \pi_1^{-1}(\pi_2(B))$

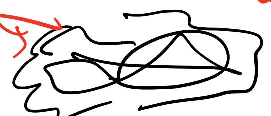
$\pi_1^{-1}(\pi_2(\cdot))$ is η.

Prover

X = (A, B)

Knows η s.t.
A = η (B)





A

				1	1	1	
				1	1		1
				1		1	1
					1	1	1
1	1	1					
1	1		1				
1		1	1				
	1	1	1				

G = φ(A) →

	1		1		1		
1		1				1	
	1			1			1
		1		1		1	
1	1						1
			1		1		1
		1		1	1		
1			1			1	

G →

← c = A or B

π s.t. $\pi(c) = G$ →

Verifier

$(G, c, \pi \text{ s.t. } \pi(c) = G)$

$\forall \varphi \ \exists \varphi'$ s.t.
$\varphi(A) = \varphi'(B)$

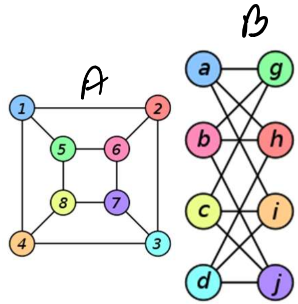
When Sim guess = A
then c = B
(and vice-versa)
This cannot happen
because G hides (A/B).

Graph Isomorphism

If A and B are isomorphic,
then dist. $(\varphi(A)) = (\varphi(B))$ for random φ

Simulator

X = (A, B)



Verifier

Does not know η!

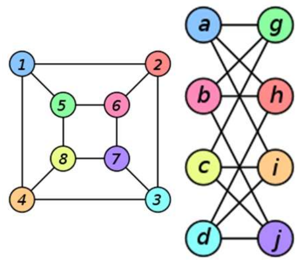
~~Knows c in advance~~

$G = \varphi(\text{guess})$ →
← c
→
π s.t. $\pi(c) = G$
if c = guess, we're done! $\pi = \varphi$.

Graph Isomorphism

Simulator

X = (A, B)



$k = 256$

Verifier

Doesn't know η.

~~Knows c in advance~~

G_0 →

← c_0

π_0 s.t. $\pi_0(G_{c_0}) = G_0$

(at most k guesses)

breaking pt

G_1 →

← c_1

π_1 s.t. $\pi_1(G_{c_1}) = G_1$ →

(at most k guesses)

⋮ n times

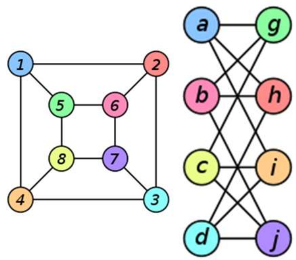
Runtime of Simulator

$O(n.k)$

Graph Isomorphism

Simulator

X = (A, B)



Verifier

Knows c in advance

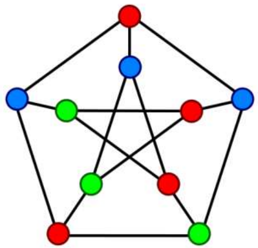
G = φ(c)

G →

c = A or B ←

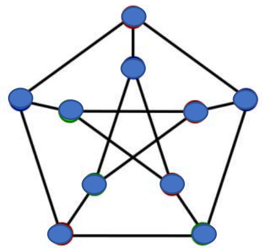
φ →

3-coloring

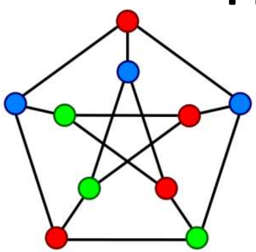


Color all vertices with only three colors (R, G, B) such that
no edge should connects two vertices of the same color.

3-coloring



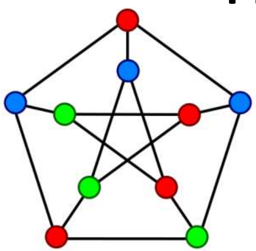
Prover

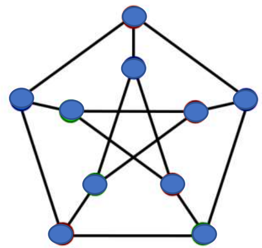


Verifier

3-coloring

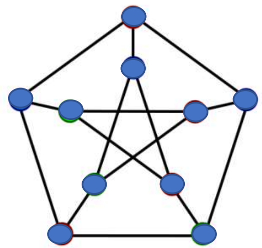
Prover



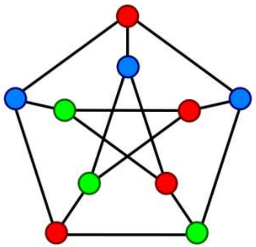


Verifier

3-coloring



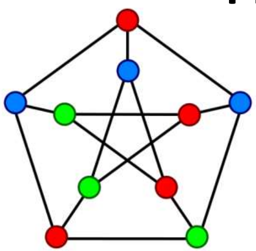
Prover

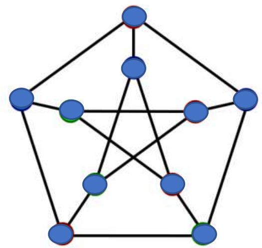


Verifier

3-coloring

Prover

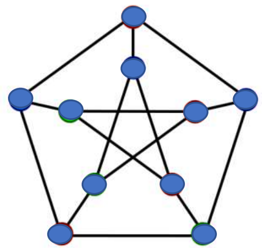




Verifier

3-coloring

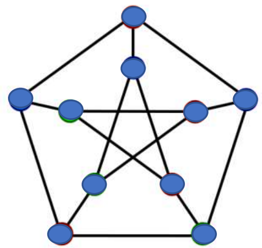
Simulator



Verifier

3-coloring

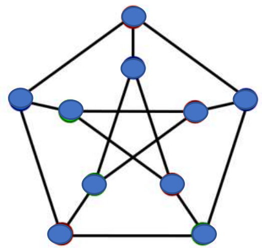
Simulator



Verifier

3-coloring

Simulator



Verifier