Zero Knowledge - Session 1

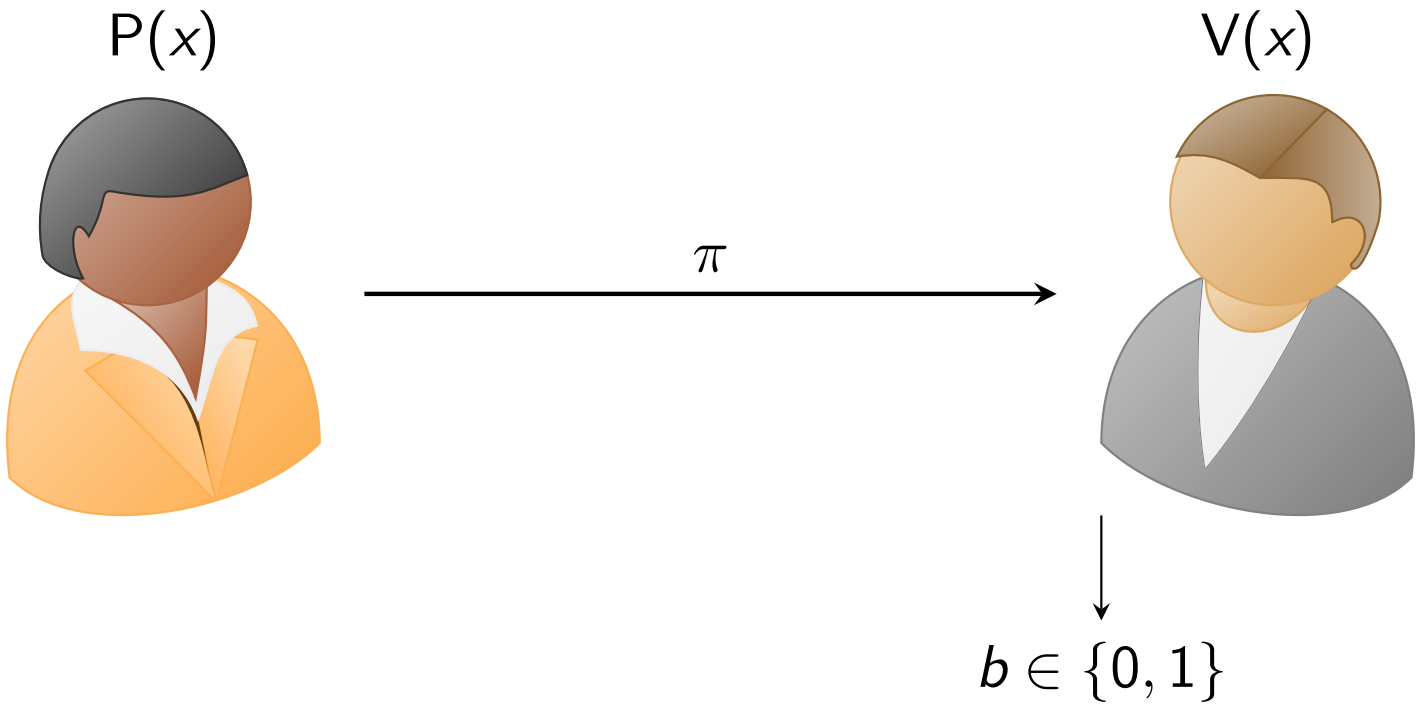
Elahe Sadeghi

October 2020

What is a zero-knowledge proof?

concept of zero-knowledge proof systems

A zero-knowledge (ZK) proof system is a protocol by which a prover can convince a verifier that a particular statement is true, while revealing nothing more than that fact.



Overview

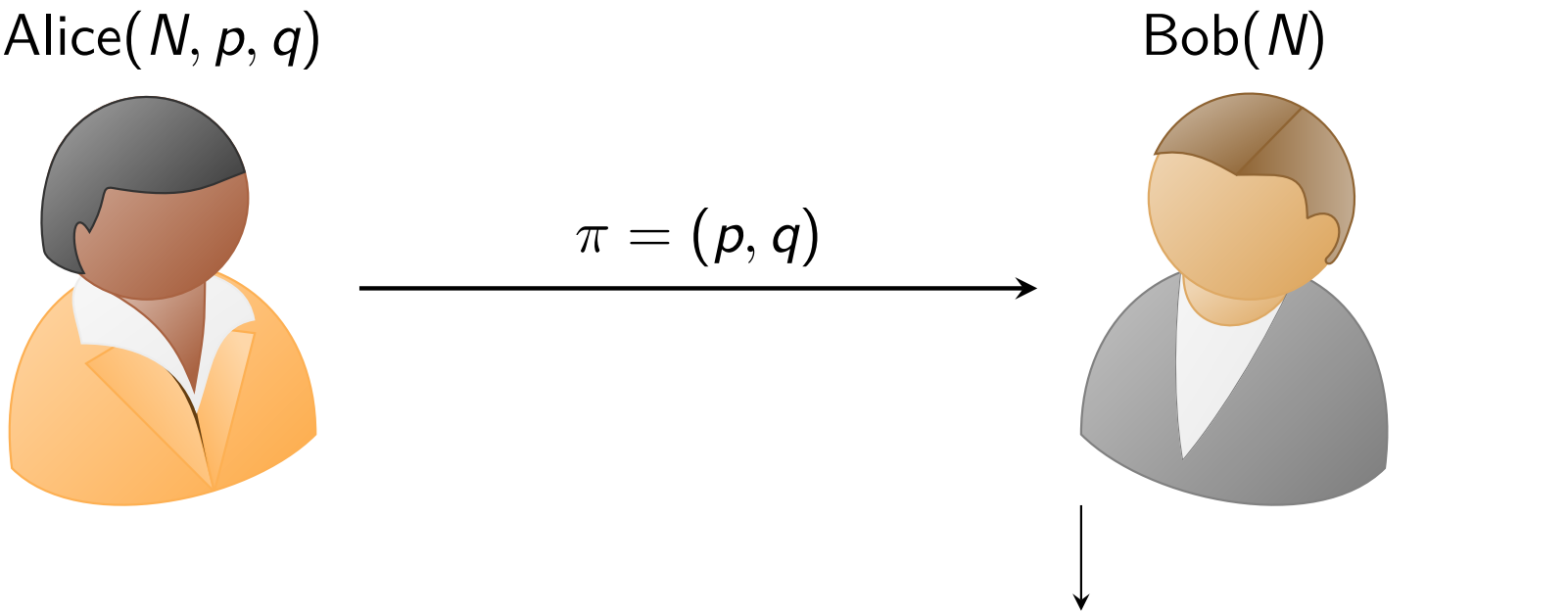
What is a zero-knowledge proof?

concept of zero-knowledge proof systems

A zero-knowledge (ZK) proof system is a protocol by which a prover can convince a verifier that a particular statement is true, while revealing nothing more than that fact.

Proof Systems

- Consider following example:
  Suppose Alice wants to convince Bob that $N = pq$, where p,q are prime and secret.



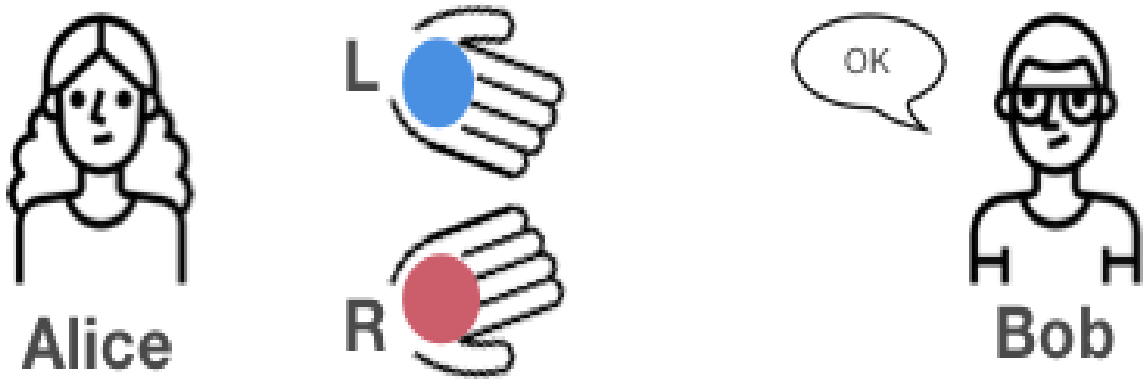
$=>$ Note: After this protocol, Bob learns the factorization of N.

Zero-Knowledge Proofs

Zero-Knowledge (intuition)

An interactive proof system is zero-knowledge if there exists an efficient simulator that can simulate the transcript of the protocol.

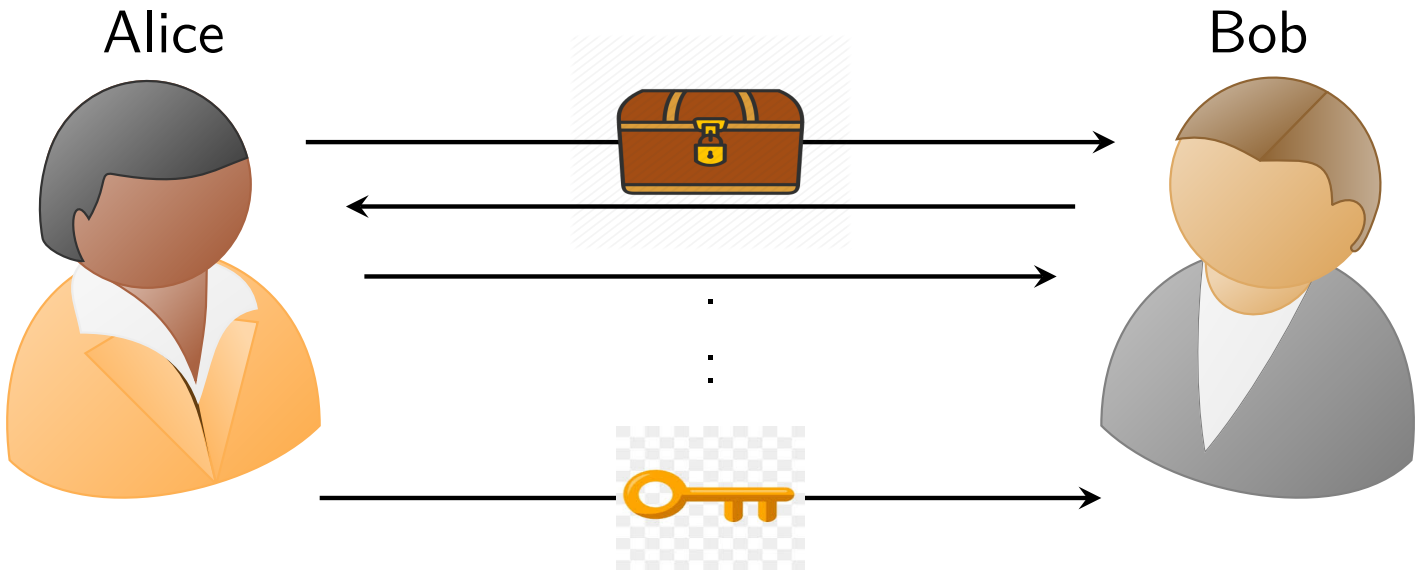
Zero-Knowledge Proofs



Zero-Knowledge Proofs

- **Theorem:** Anything that can be proved, can be proved in zero-knowledge!
  proof. By using **commitment schemes**.

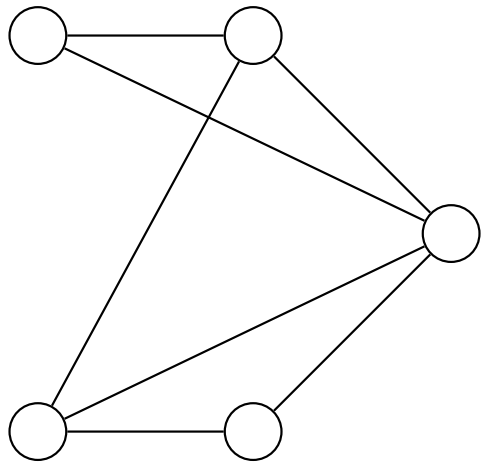
- A simple example of a commitment scheme:



Theorem: We can prove anything that can be proved, can be proved in zero-knowledge!

We show this theorem for $\mathcal{NP}$ languages. So, it suffices to construct a zero-knowledge proof system for a "$\mathcal{NP}$-complete" language.

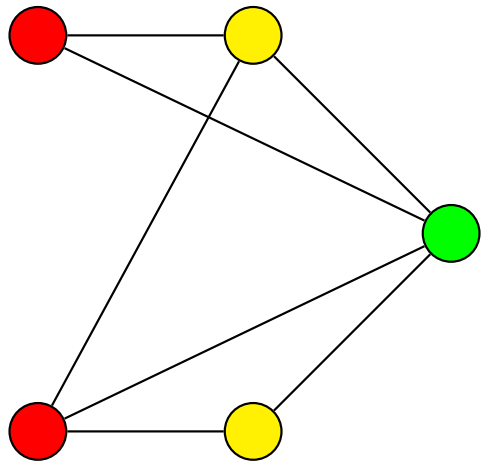
Consider the language of **graph 3-colorability**: Given a graph G, can you color the vertices such that no adjacent nodes have the same color?



Zero-Knowledge Proofs

Everything provable is provable in zero-knowledge!

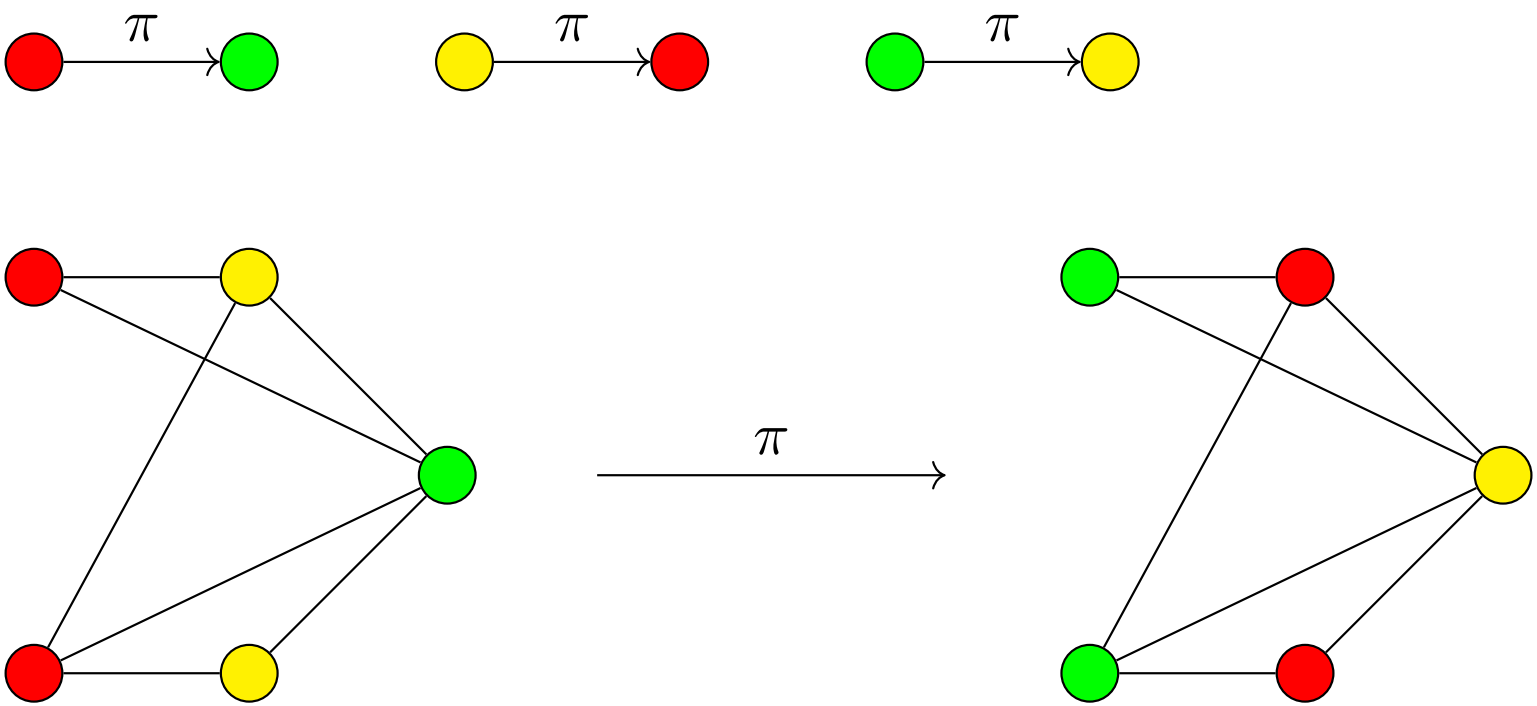
Alice wants to prove that a given graph G is 3-colorable.



Zero-Knowledge Proofs

Everything provable is provable in zero-knowledge!

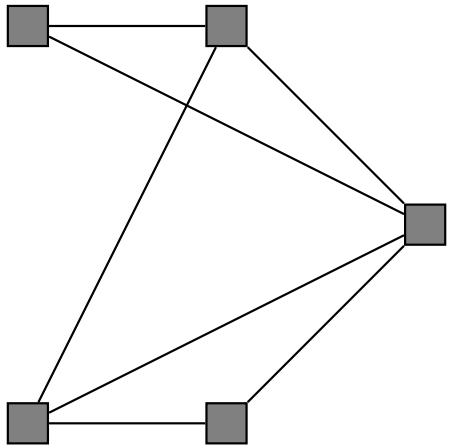
- Alice chooses a random permutation and applies it to the color of each node.



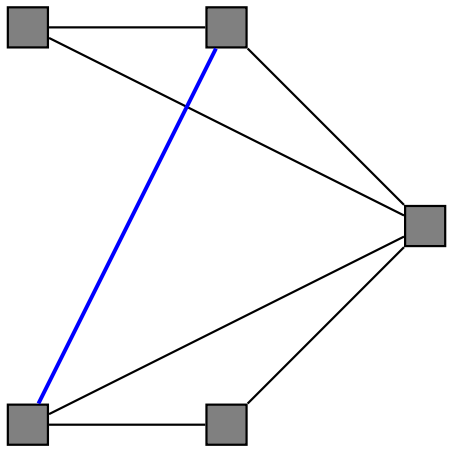
Zero-Knowledge Proofs

Everything provable is provable in zero-knowledge!

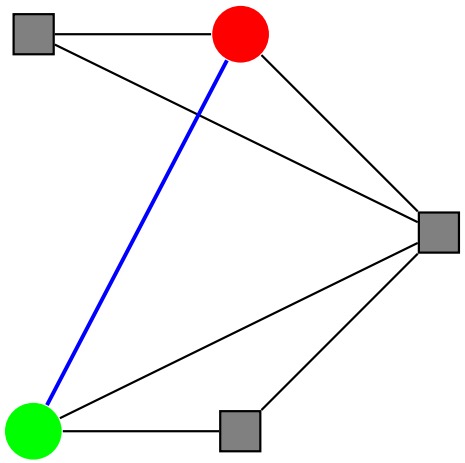
Alice Commits to color of each node and sends all the commitments to Bob.



Bob challenges Alice to reveal coloring of a single edge.

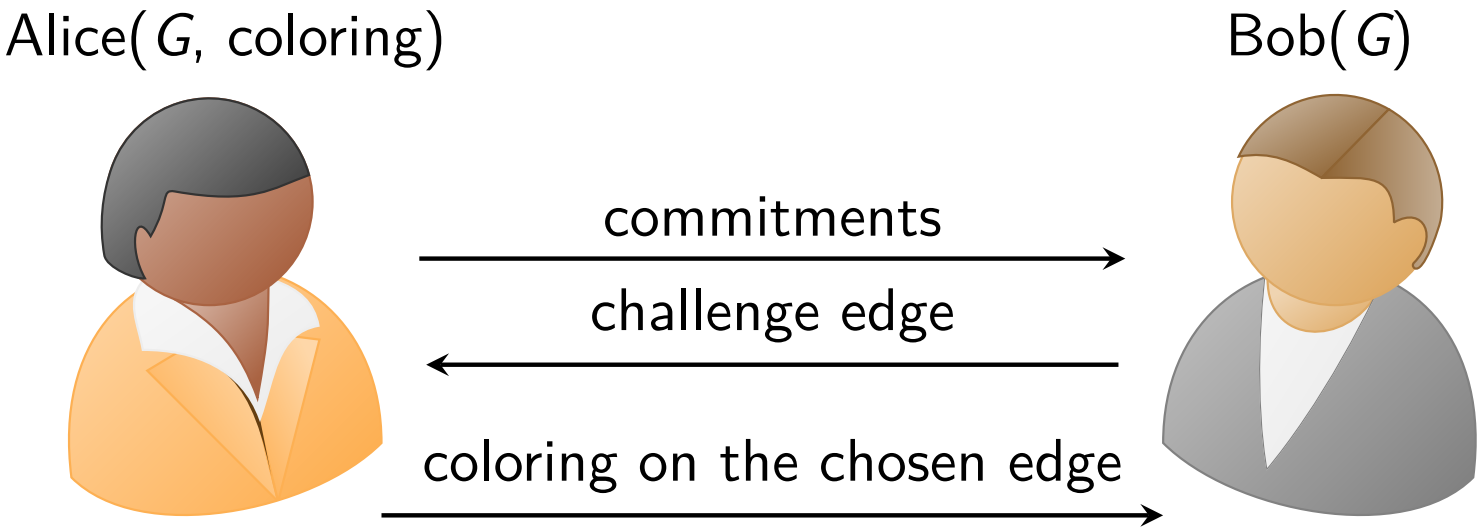


Alice reveals the coloring on the chosen edge and opens the entries in the commitment.



So why is it a zero-knowledge proof?!

So our ZK proof for graph 3-coloring will be as following:



- Bob accepts if colors are valid and different from each other, and rejects otherwise.

Zero-Knowledge Proofs

Everything provable is provable in zero-knowledge!

- **Completeness**: if coloring is valid, Alice can always answer the challenge properly.
- **Soundness**: If G is not 3-colorable, there is two adjacent nodes that have the same color. Since Alice already committed to these colors, Bob will reject with probability of at least $\frac{1}{|E|}$, which is the probability to choose that edge.
  With repeating this protocol, we can reduce this to $e^{-|E|}$.

- **Zero-Knowledge**: Consider the following algorithm:
  1. Choose an arbitrary coloring for G with just three colors.
  2. Assume the verifier will pick one random edge.
  3. If the two colors were different ,then outputs them. Otherwise, restart and try again.

  This Algorithm succeeds with probability $\frac{2}{3}$. So it produces a valid transcript with probability $1 - \frac{1}{3^\lambda} = 1 - negl(\lambda)$ after λ attempts. So we can conclude that the verifier with running this algorithm, and without interacting with the prover, can have a transcript with the same distribution.

Zero-Knowledge Proofs

Everything provable is provable in zero-knowledge!

So we have a zero-knowledge proof for all $\mathcal{NP}$ languages!

- **Identification from ZK protocols**:
  1. Client publishes $y = f(x)$.
  2. Client proves to server in ZK that she knows $f^{-1}(y)$.

Why is it important to be ZK?

Because of **impersonation attacks**!

Applications

- **Protocol Desgin**:
  1. Design against parties that follow instructions.
  2. Use ZK proof to force honest behavior.

  An example can be electronic voting systems. In e-voting systems:
  1. Eevery vote should be **valid**.
  2. System should preserve **privacy**.

  Because of these two properties, we need zero-knowledge proofs for these systems.

Applications

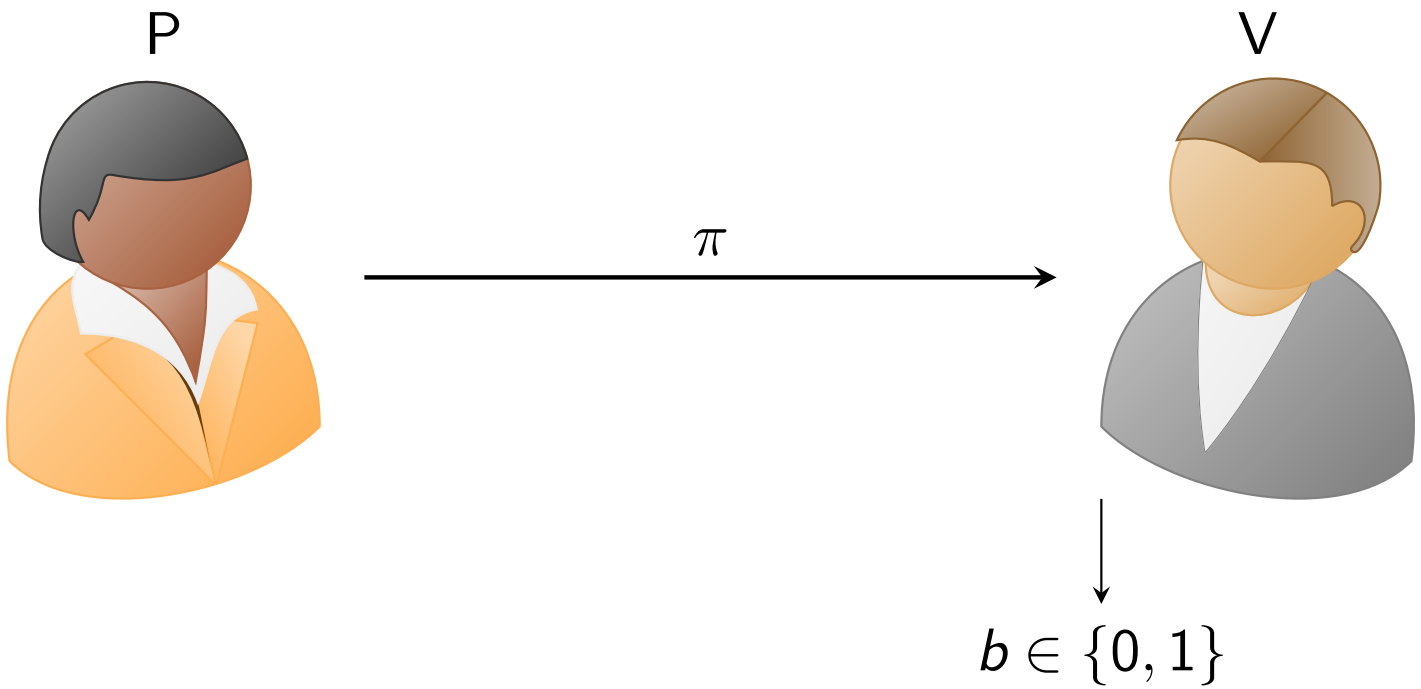
Problem: In both of these applications, prover and verifier must be online at the same time. Because the protocol requires interaction!

$=>$ Solution: Make the protocols **non-interactive**.

Non-Interactive Zero-Knowledge

Non-Interactive Zero-Knowledge

A zero-knowledge proof system where the proof is a single message from the prover to the verifier.
It is important because interaction is **expensive** in practice.

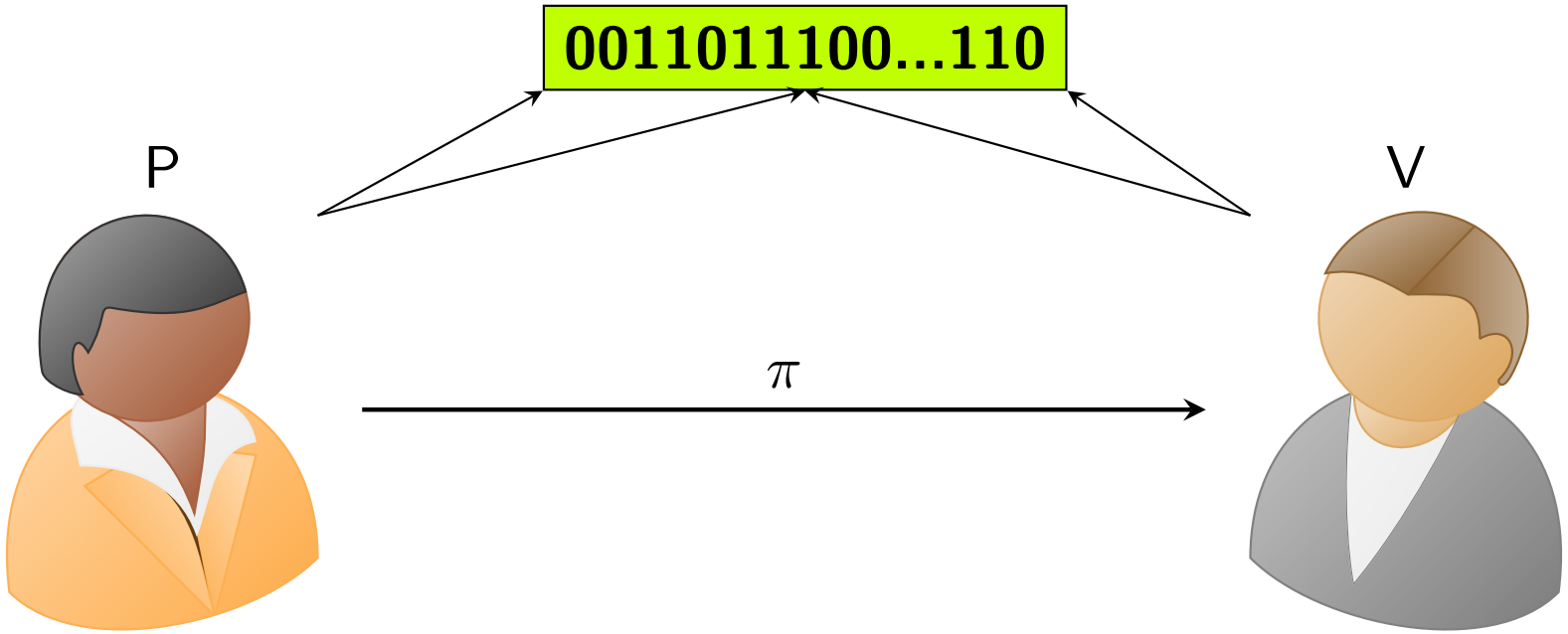


Non-Interactive Zero-Knowledge

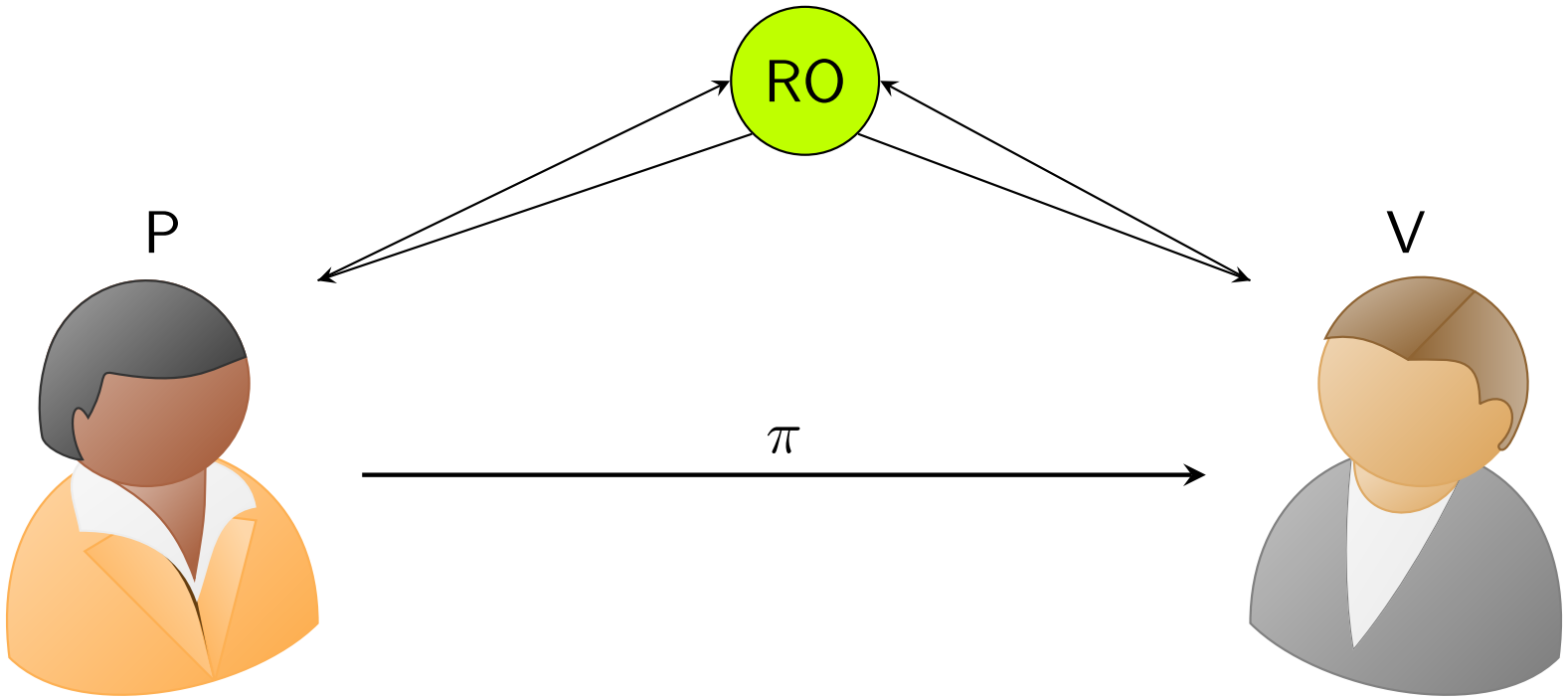
- Unfortunately, NIZK proofs are only possible for sufficiently easy problems. ($\mathcal{BPP}$ languages)

$\rightarrow$ We can have NIZK in other models rather than the standard model.

- **Common Reference/Random String Model**:



- **Random Oracle Model**:

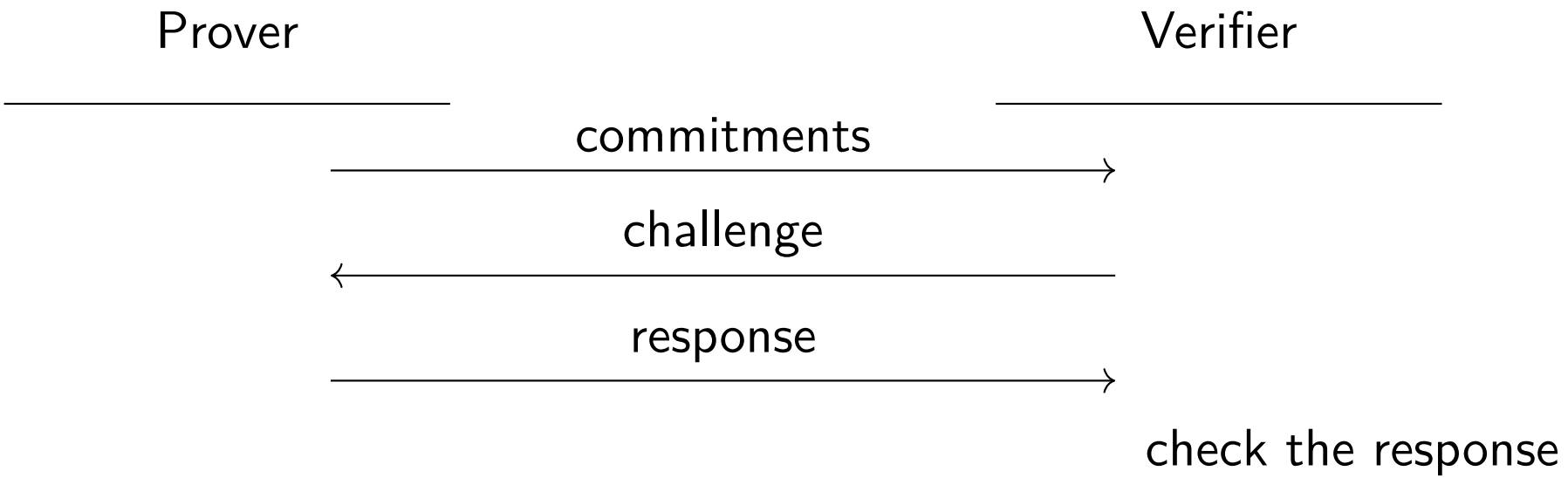


Non-Interactive Zero-Knowledge

Constructing a NIZK proof from a ZK proof!

Non-Interactive Zero-Knowledge

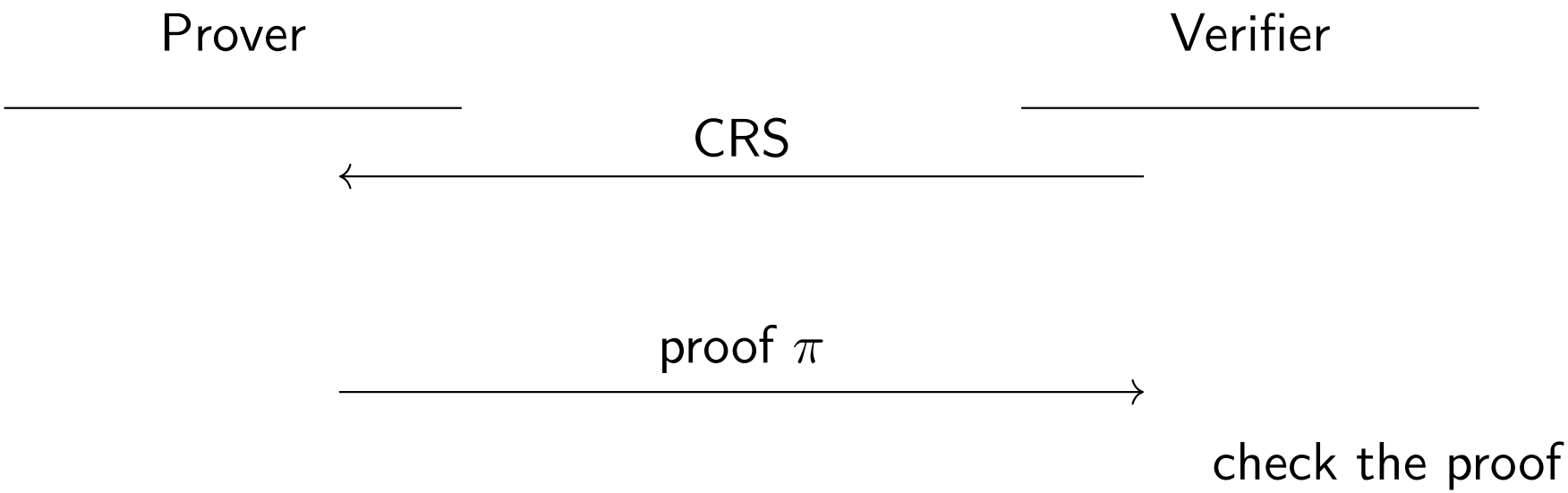
- **Fiat-Shamir transform**: From ZK to NIZK in RO Model



-> The idea is using a random oracle (which you can view it as a random function) instead of verifier's challenge. So prover can computes it **on its own**.

ZAPs

- ZAPs are the relaxation of NIZKs in the CRS model. The only change in these protocols is that the verifier will generate the CRS and send it to the prover.



-> The prover can use this CRS for computing as many proof as its will.

Summary

- ZKPs are protocols for proving the truth of some statement which has 3 properties: Completeness, Soundness, Zero-Knowledge.
- We saw that everything provable is provable in ZK. (by using commitment schemes)
- Because of the cost of interaction, We introduce a new notion: NIZK.
- How to construct NIZK protocols from ZK protocols.
- ZAPs are basically the relaxation of NIZKs in the CRS model.

Thank you!