Zero Knowledge - Session 5

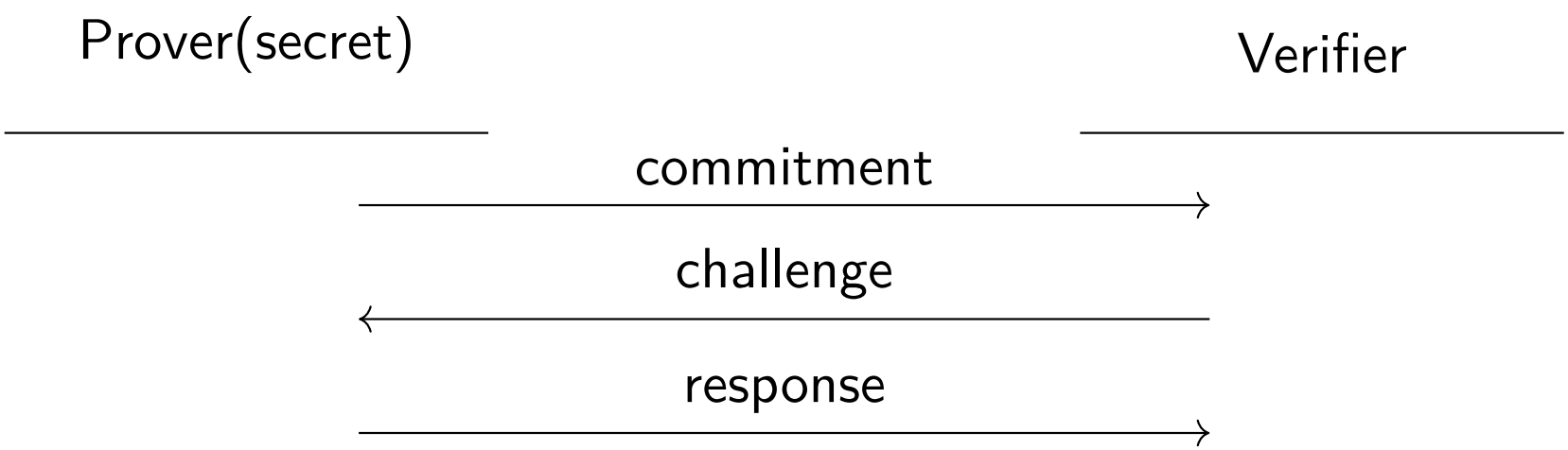
Elahe Sadeghi

October 2020

Σ-protocols

So far, we saw every language in $\mathcal{NP}$ has a zero knowledge proof.
A general view of this is Σ-protocols. Σ-protocols have a structure of "commitment-challenge-response".

- Such construction is as follows:

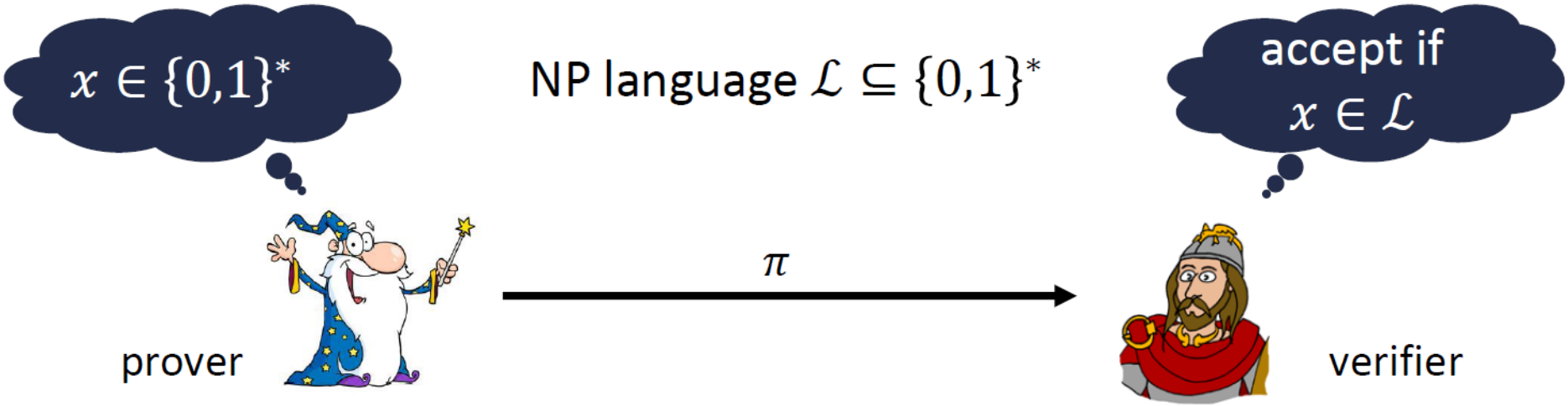


Σ-protocols

Problem: Prover and verifier must be online at the same time. Because the protocol requires interaction!

$\Rightarrow$ Solution: Make the protocols **non-interactive**.

Non-Interactive Zero-Knowledge



Completeness: $\forall x \in \mathcal{L} : \Pr[\langle P, V \rangle(x) = \text{accept}] = 1$

"Honest prover convinces honest verifier of true statements"

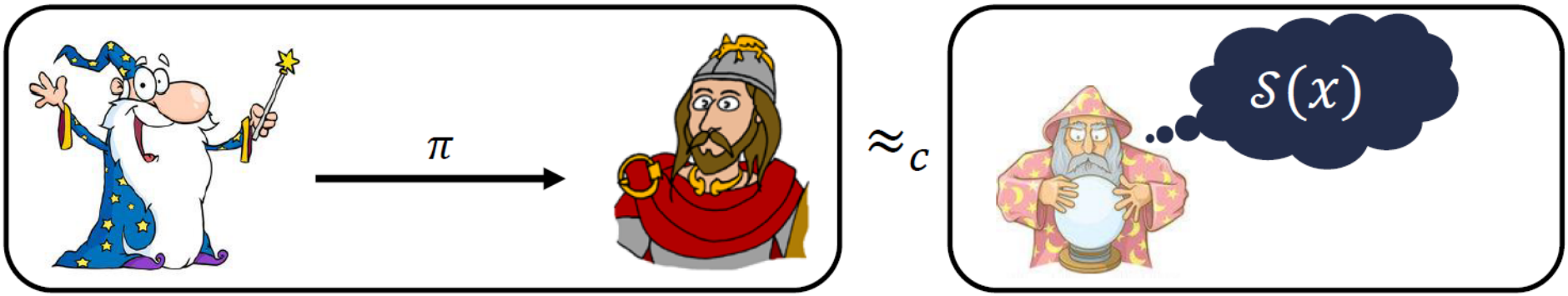
Soundness: $\forall x \notin \mathcal{L},\ \forall P^* : \Pr[\langle P^*, V \rangle(x) = \text{accept}] \leq \varepsilon$

"No prover can convince honest verifier of false statement"

can consider both computational and statistical variants

Non-Interactive Zero-Knowledge

NP language $\mathcal{L}$



real distribution ideal distribution

Zero-Knowledge: for all efficient verifiers V^*, there exists an efficient simulator $\mathcal{S}$ where

$$\forall x \in \mathcal{L} : \langle P, V^* \rangle(x) \approx \mathcal{S}(x)$$

can consider both computational and statistical variants

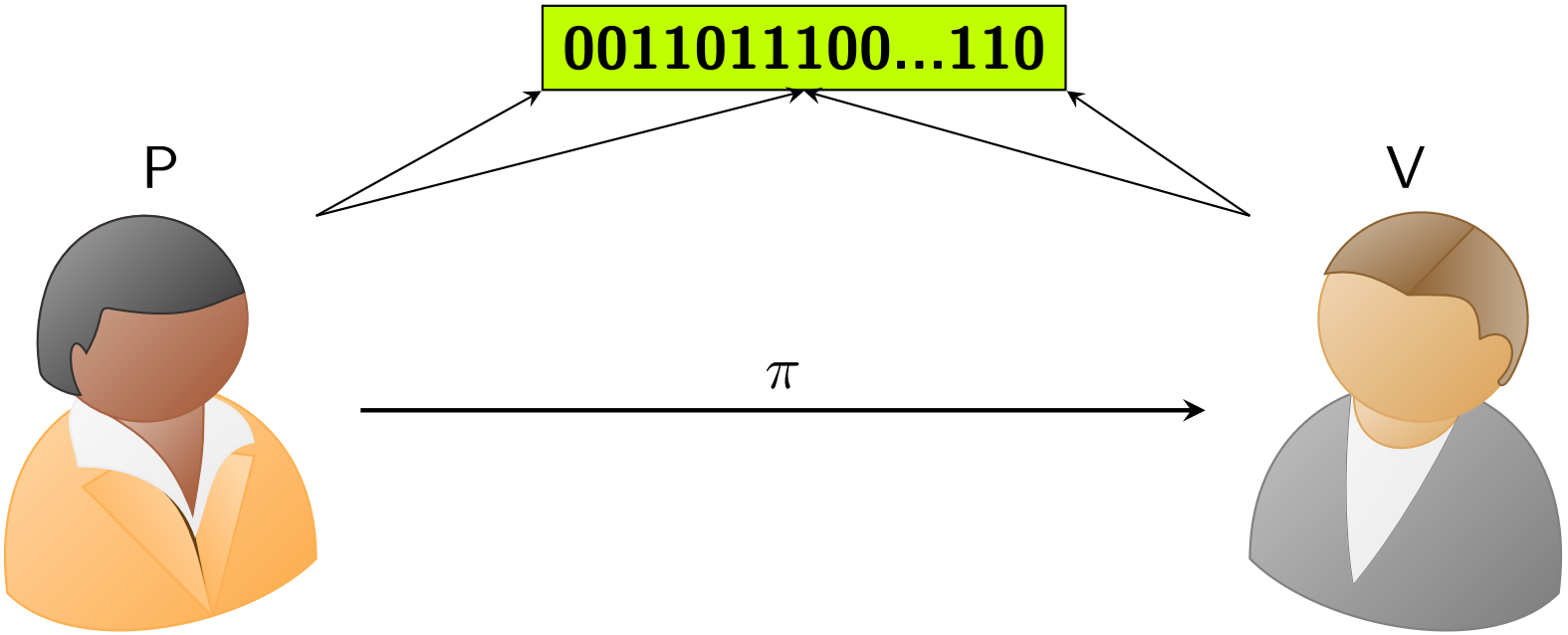
Non-Interactive Zero-Knowledge

Non-Interactive Zero-Knowledge

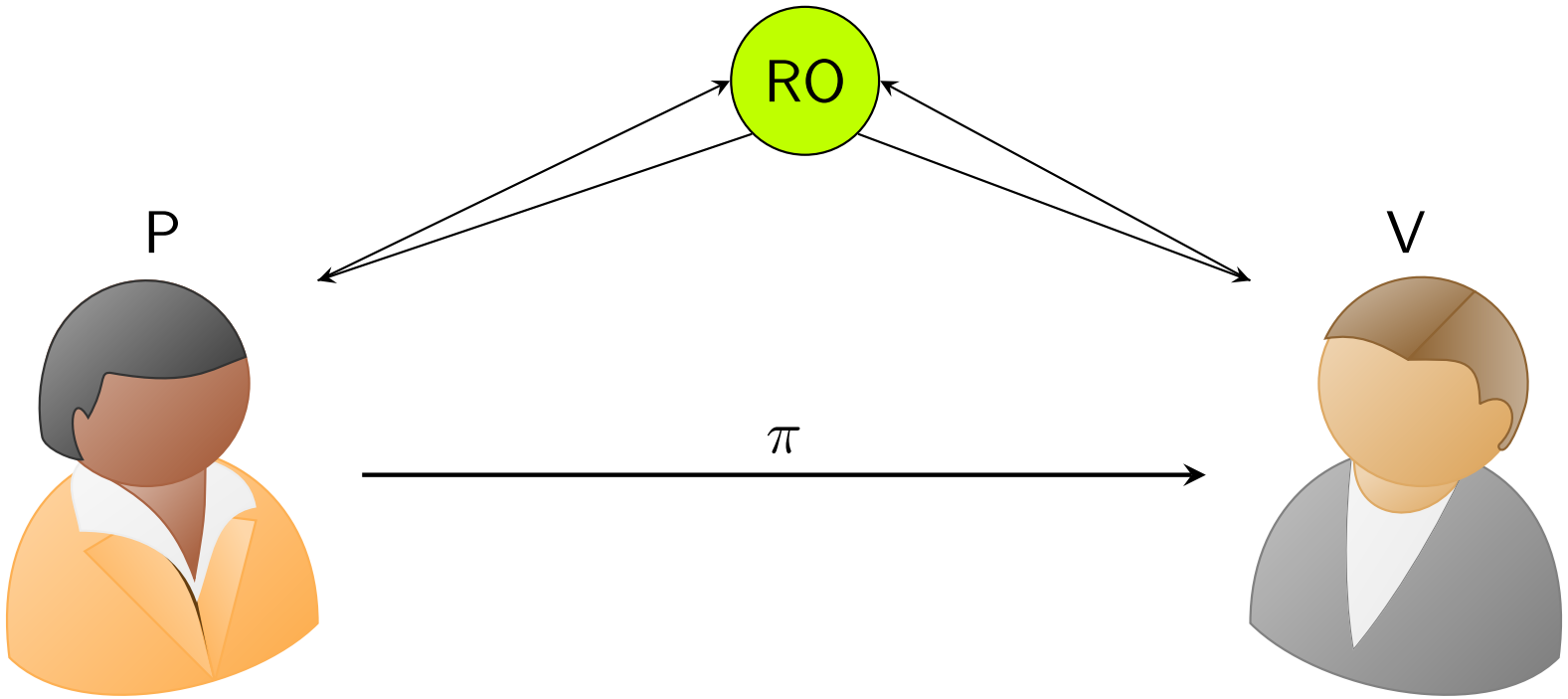
A zero-knowledge proof system where the proof is a single message from the prover to the verifier.
It is important because interaction is **expensive** in practice.

- Unfortunately, NIZK proofs are only possible for sufficiently easy languages. ($\mathcal{BPP}$ languages)
  -> The reason is that we can use the simulator to decide the language:
  1. if $x \in L$, $\pi = S(x)$ should be accepted by the verifier. (by ZK property)
  2. if $x \notin L$, $\pi = S(x)$ should not be accepted by verifier. (by soundness)

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

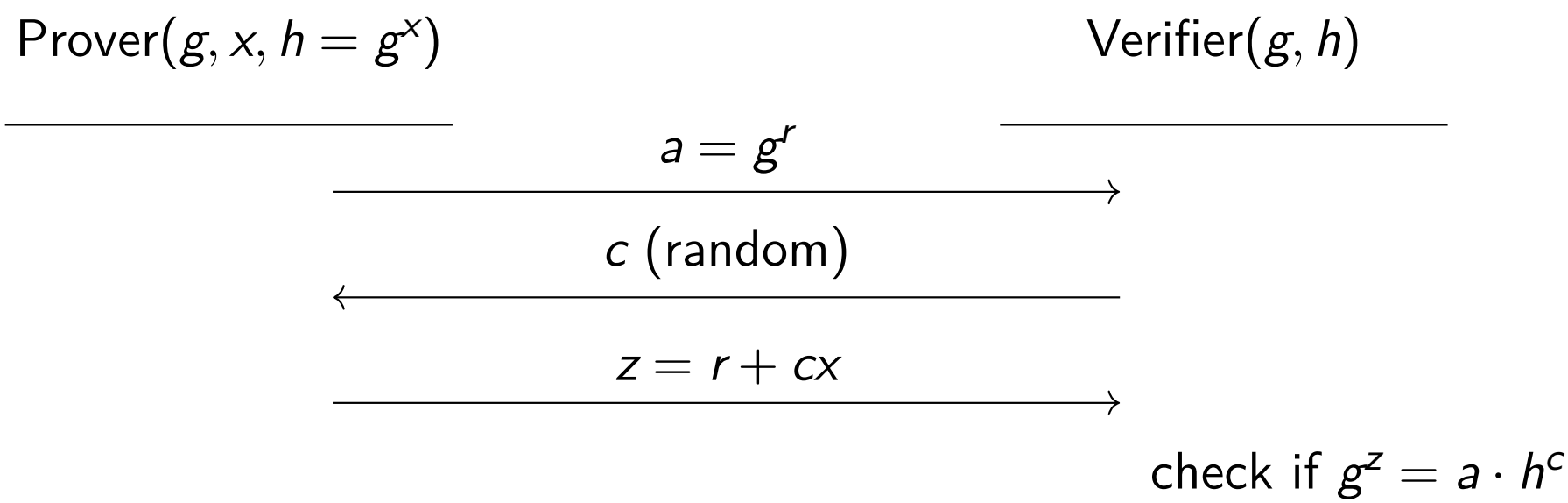


- **Random Oracle Model**:



Fiat-Shamir Transform

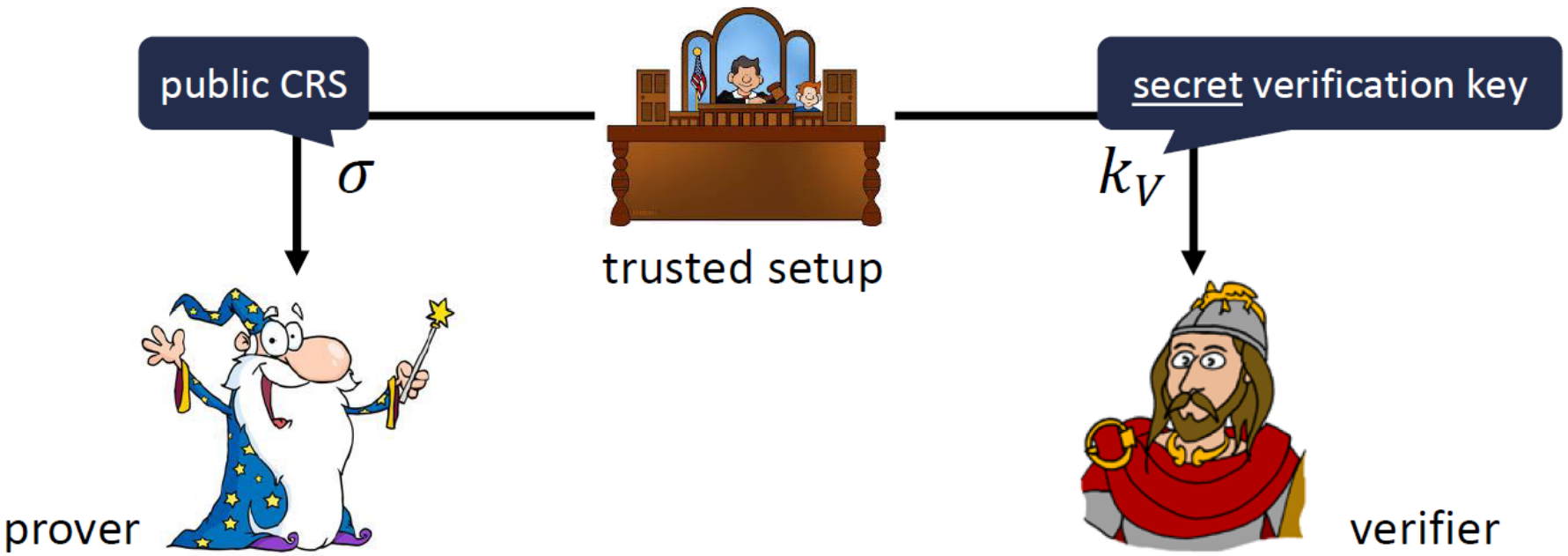
- **Fiat-Shamir transform**: From Σ-protocols to NIZK in RO Model
  Recall Schnorr's protocol (from Σ-protocols):



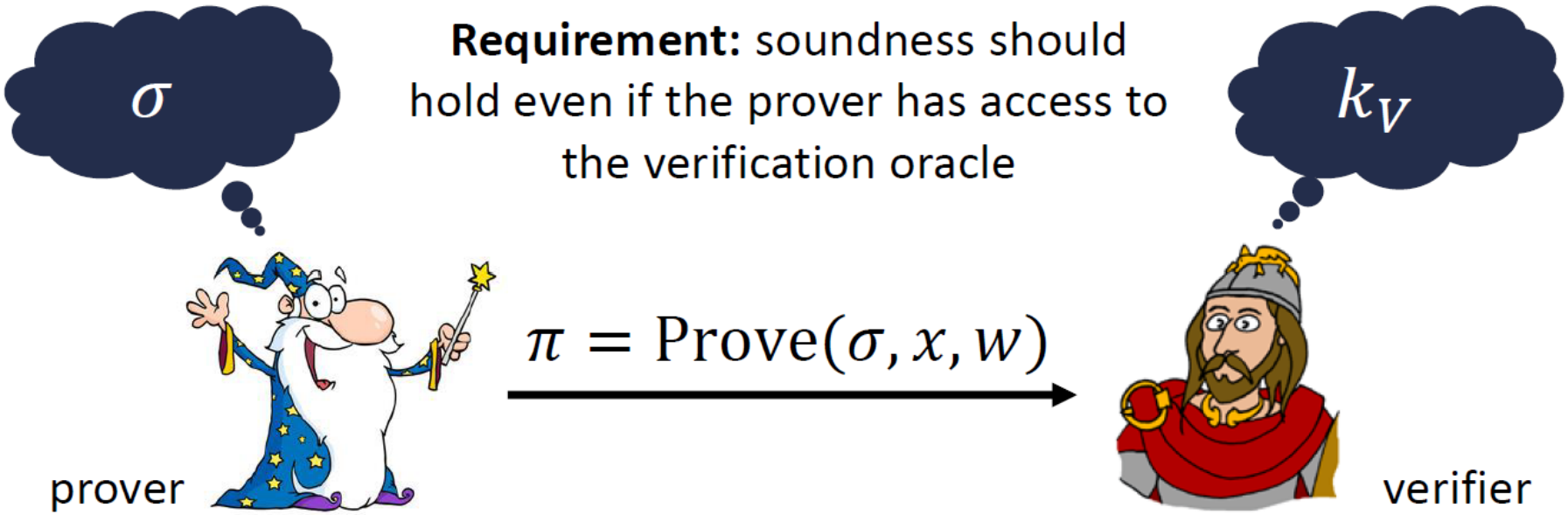
-> The idea is using a hash function (random oracle) H instead of verifier's challenge. So prover can computes it **on its own**.

$$\Rightarrow c = H(g, h, a)$$

Designated-Verifier NIZK



Designated-Verifier NIZK



NIZK in relaxed models

- **Designated-Verifier NIZK (DV-NIZK):** As discussed earlier
- **Designated-Prover NIZK (DP-NIZK):** As DV-NIZK, except that instead of the verifier, the prover has a secret proving key k_P that is used to construct proofs.
- **NIZK in the Preprocessing model:** As before, there is an initial (trusted) setup phase that generates both a proving key k_P and a verification key k_V.

NIZK in Preprocessing Model

(trusted) setup algorithm generates both proving key k_P and a verification key k_V (statement-independent)

k_P

k_V

prover

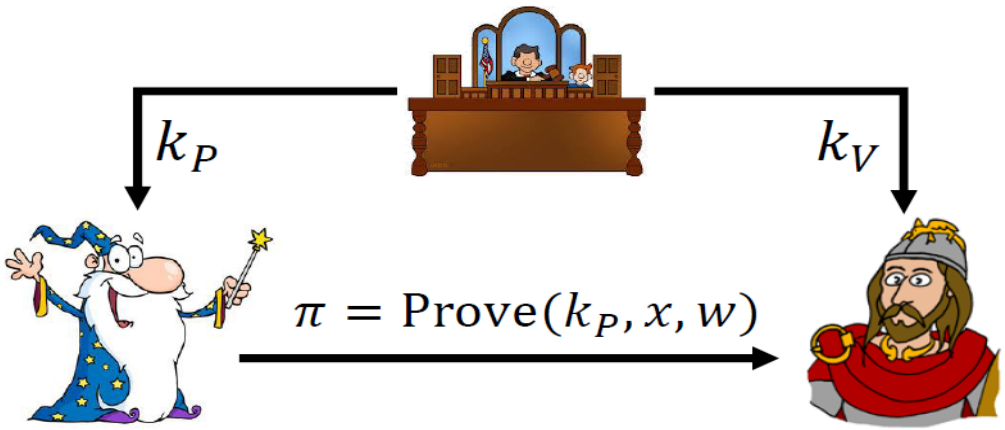
$\pi = \text{Prove}(k_P, x, w)$

prover algorithm takes proving key k_P, NP statement x, and NP witness w

verifier

$\text{Verify}(k_V, x, \pi)$

NIZK in Preprocessing Model



main requirement:
reusability

suffices for many applications of NIZKs

simpler than CRS model:

- soundness holds assuming k_V is hidden
- zero-knowledge holds assuming k_P is hidden

CRS model: k_P and k_V are both public

NIZK in relaxed models

* An important question is that whether we can construct a NIZK in any of this models for all NP with standard assumption.

NIZK in relaxed models

- Several works study NIZKs with preprocessing but all either (1) only consider specific algebraic languages rather than general NP, (2) are not reusable or (3) require assumptions such as factoring from which we already have standard NIZKs.

NIZK in relaxed models

- It was not until recently that Kim and Wu [eprint] in a breakthrough result gave a novel construction of reusable DP-NIZKs for general NP languages under the LWE assumption.

Multi-Theorem Preprocessing NIZKs from Lattices

Sam Kim
Stanford University
skim13@cs.stanford.edu

David J. Wu
Stanford University
dwu4@cs.stanford.edu

Abstract

Non-interactive zero-knowledge (NIZK) proofs are fundamental to modern cryptography. Numerous NIZK constructions are known in both the random oracle and the common reference string (CRS) models. In the CRS model, there exist constructions from several classes of cryptographic assumptions such as trapdoor permutations, pairings, and indistinguishability obfuscation. Notably absent from this list, however, are constructions from standard *lattice* assumptions. While there has been partial progress in realizing NIZKs from lattices for specific languages, constructing NIZK proofs (and arguments) for all of NP from standard lattice assumptions remains open.

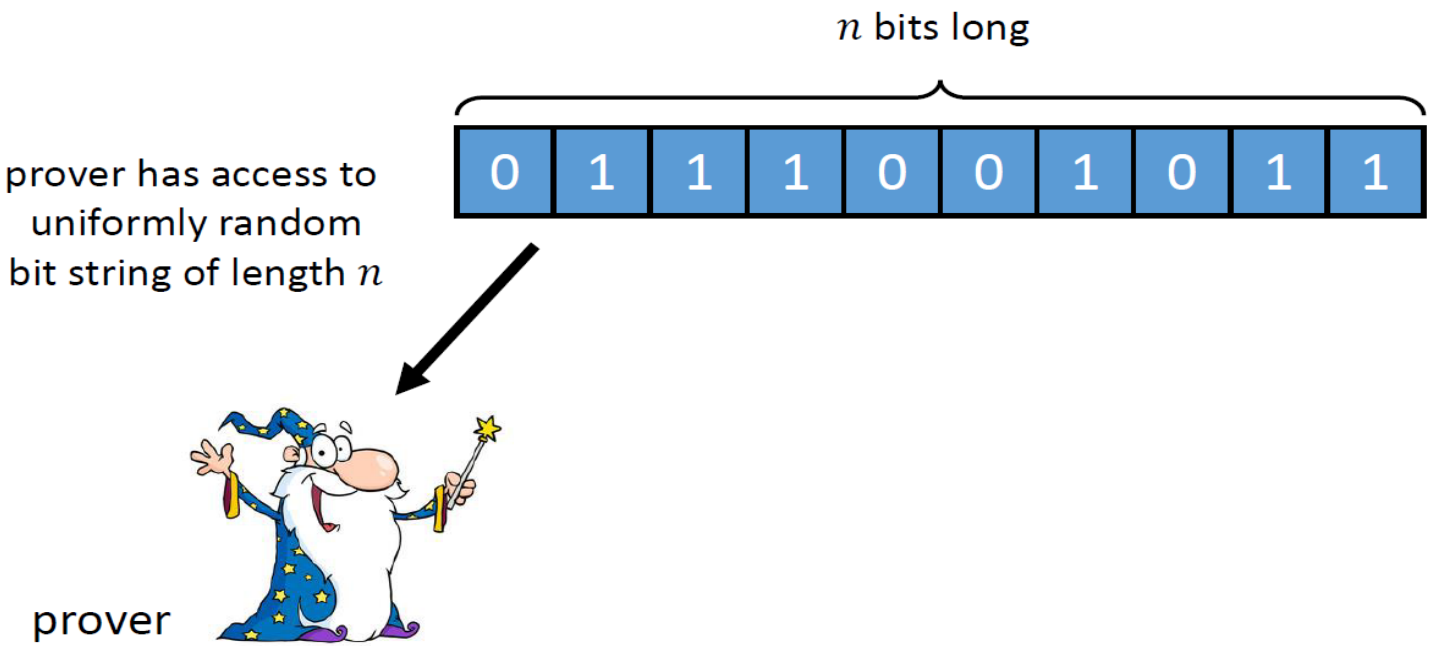
In this work, we make progress on this problem by giving the first construction of a *multi-theorem* NIZK for NP from standard lattice assumptions in the *preprocessing* model. In the preprocessing model, a (trusted) setup algorithm generates proving and verification keys. The proving key is needed to construct proofs and the verification key is needed to check proofs. In the multi-theorem setting, the proving and verification keys should be reusable for an unbounded number of theorems without compromising soundness or zero-knowledge. Existing constructions of NIZKs in the preprocessing model (or even the designated-verifier model) that rely on weaker assumptions like one-way functions or oblivious transfer are only secure in a single-theorem setting. Thus, constructing multi-theorem NIZKs in the preprocessing model does not seem to be inherently easier than constructing them in the CRS model.

We begin by constructing a multi-theorem preprocessing NIZK directly from context-hiding homomorphic signatures. Then, we show how to efficiently implement the preprocessing step using a new cryptographic primitive called *blind homomorphic signatures*. This primitive may be of independent interest. Finally, we show how to leverage our new lattice-based preprocessing

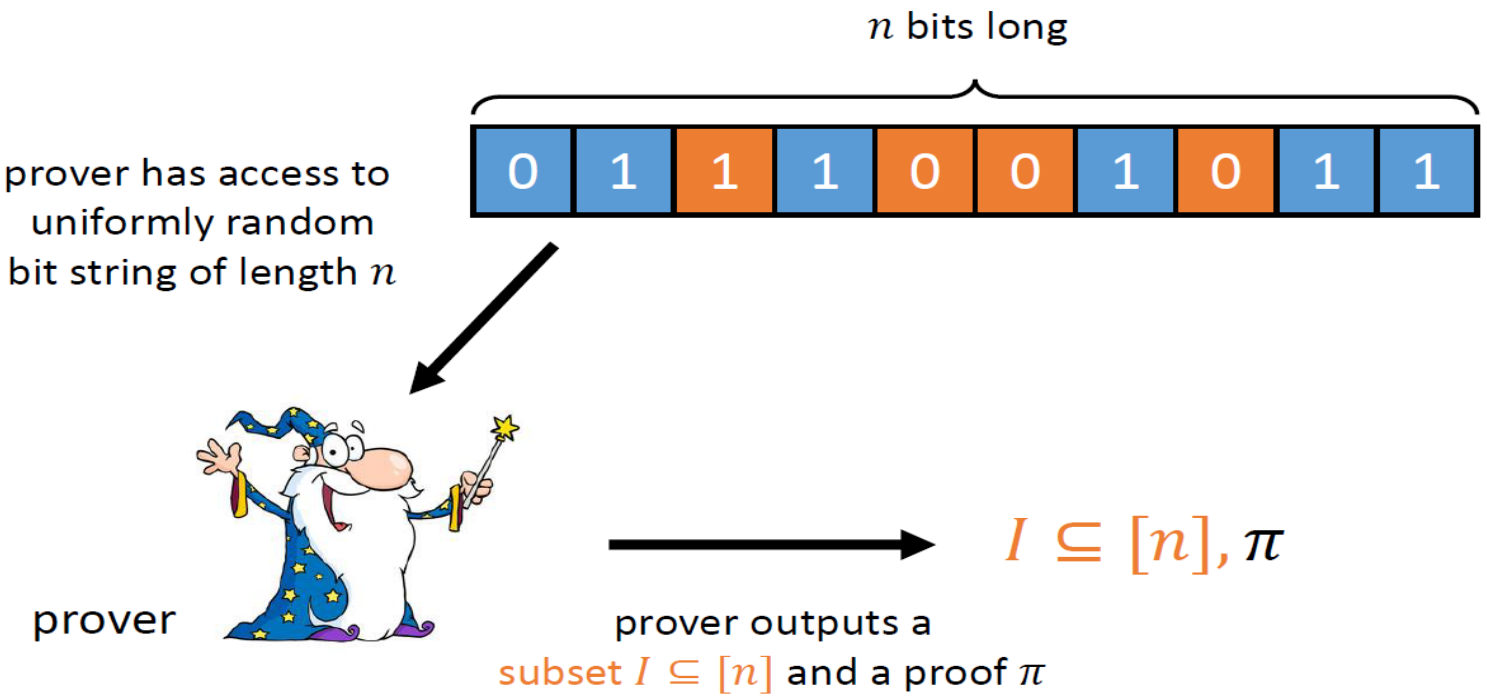
NIZKs in the Hidden Bits Model

- We don't have any actual construction for statistical NIZKs for NP in CRS model.

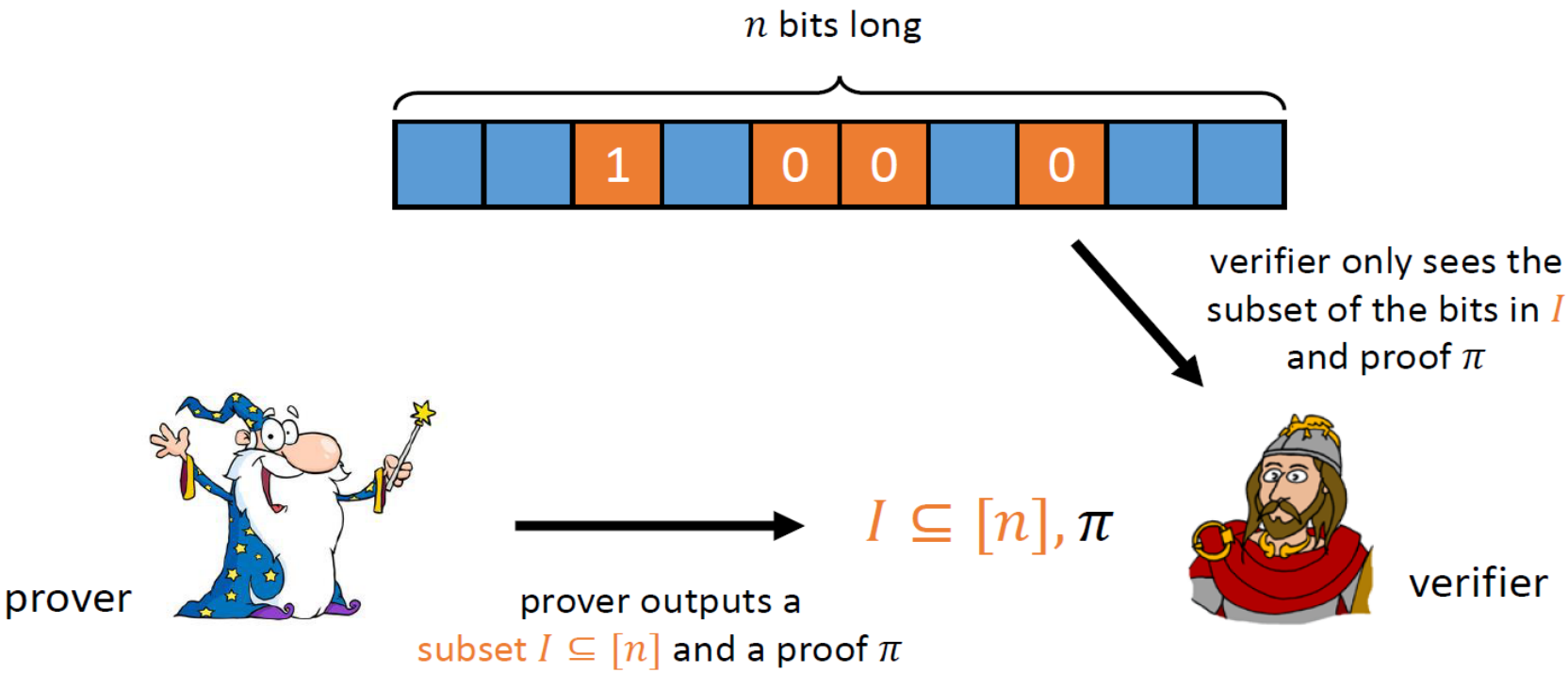
$\Rightarrow$ One try: The Hidden-Bits Model



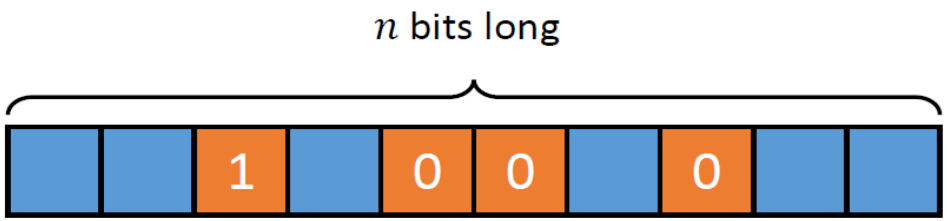
NIZKs in the Hidden Bits Model



NIZKs in the Hidden Bits Model



NIZKs in the Hidden Bits Model

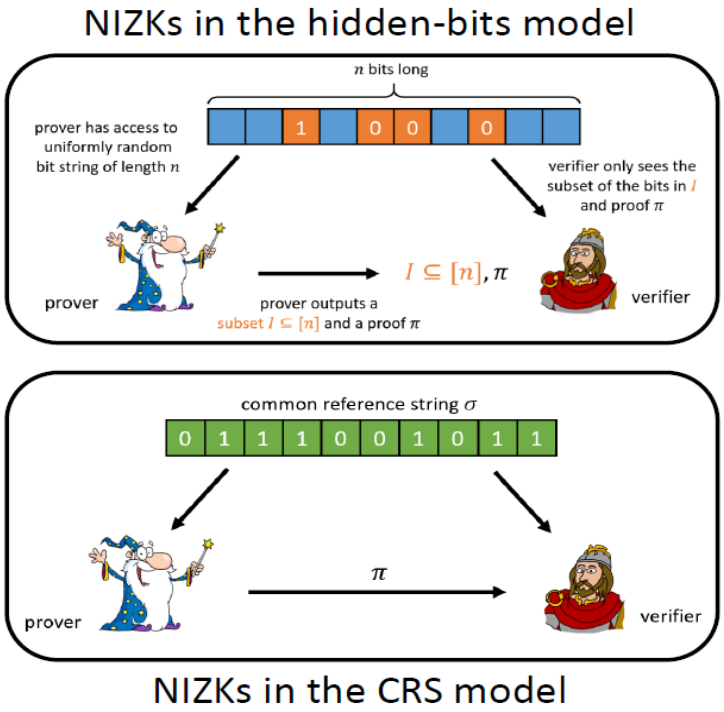


verifier only sees the subset of the bits in I and proof π

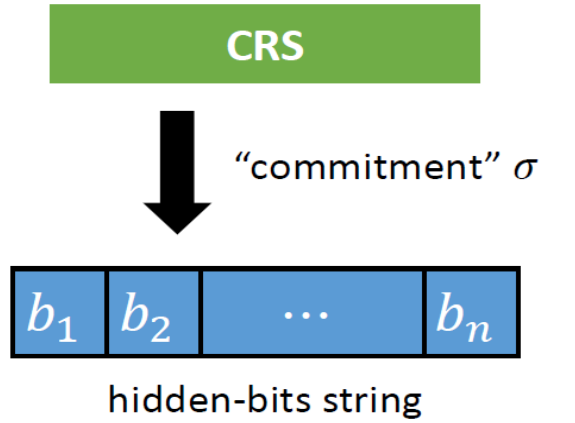
verifier

[FLS90]: There exists a perfect NIZK proof for any NP language in the hidden-bits model

The FLS Compiler



cryptographic compiler



Prover can selectively open σ to (i, b_i) for indices i of its choosing

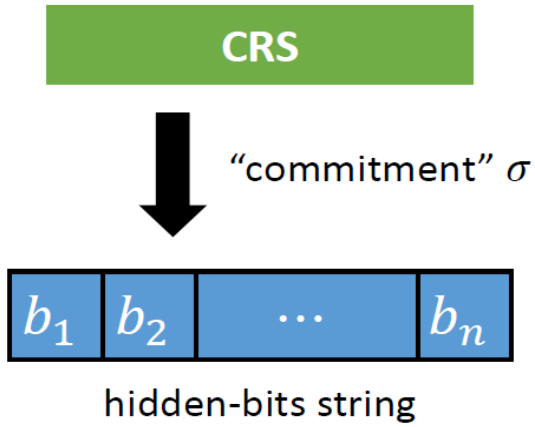
The FLS Compiler

Main properties:

- **Binding:** Can only open σ to a single bit for each position
- **Hiding:** Unopened bits should be hidden
- **Succinctness:** $|\sigma| \ll n$

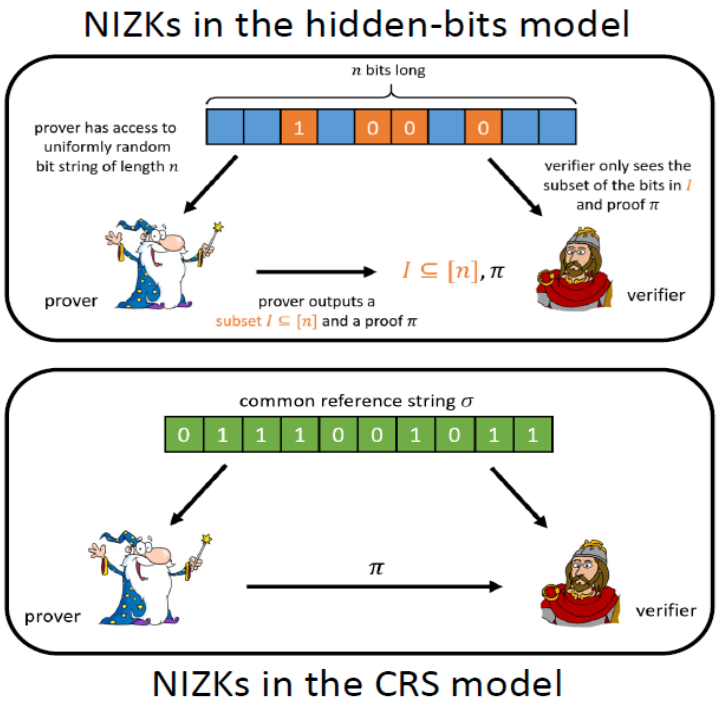
Soundness: If $|\sigma| \ll n$ and there are not too many "bad" hidden-bits strings $\Rightarrow$ prover cannot find a "bad" σ that fools verifier

Zero-Knowledge: Unopened bits hidden to verifier

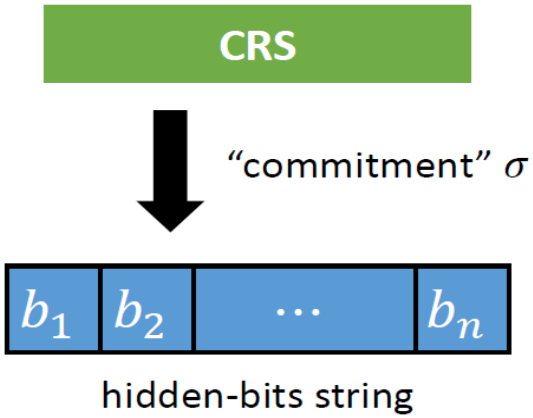


Prover can selectively open σ to (i, b_i) for indices i of its choosing

The FLS Compiler



cryptographic compiler



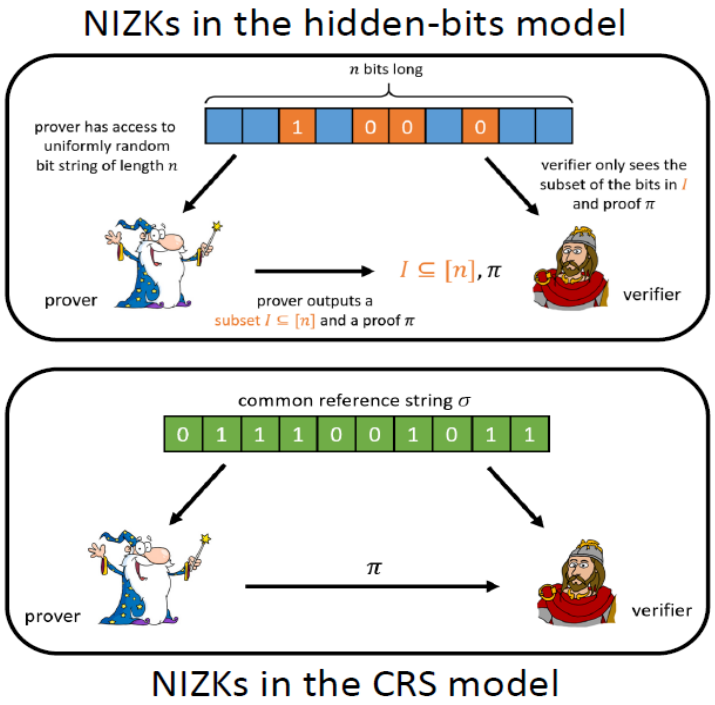
Instantiations:

[FLS90]: trapdoor permutations (computational NIZK proofs)

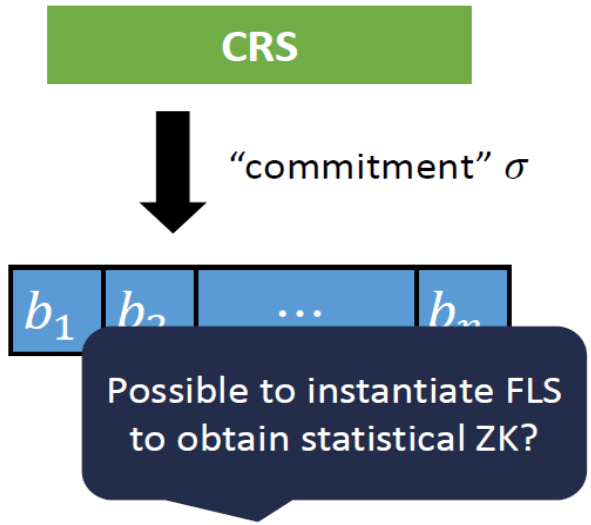
[CHK03]:CDH over a pairing group (computational NIZK proofs)

[QRW19, CH19, KNYY19]:hidden-bits generators from CDH (computational DV-NIZK proofs)

The FLS Compiler



cryptographic compiler



Instantiations:

[FLS90]: trapdoor permutations (computational NIZK proofs)

[CHK03]:CDH over a pairing group (computational NIZK proofs)

[QRW19, CH19, KNYY19]:hidden-bits generators from CDH (computational DV-NIZK proofs)

Dual-Mode NIZKs

- How to construct statistical NIZKs in CRS model? [eprint]

New Constructions of Statistical NIZKs:
Dual-Mode DV-NIZKs and More

Benoît Libert* Alain Passelègue† Hoeteck Wee‡ David J. Wu§

Abstract

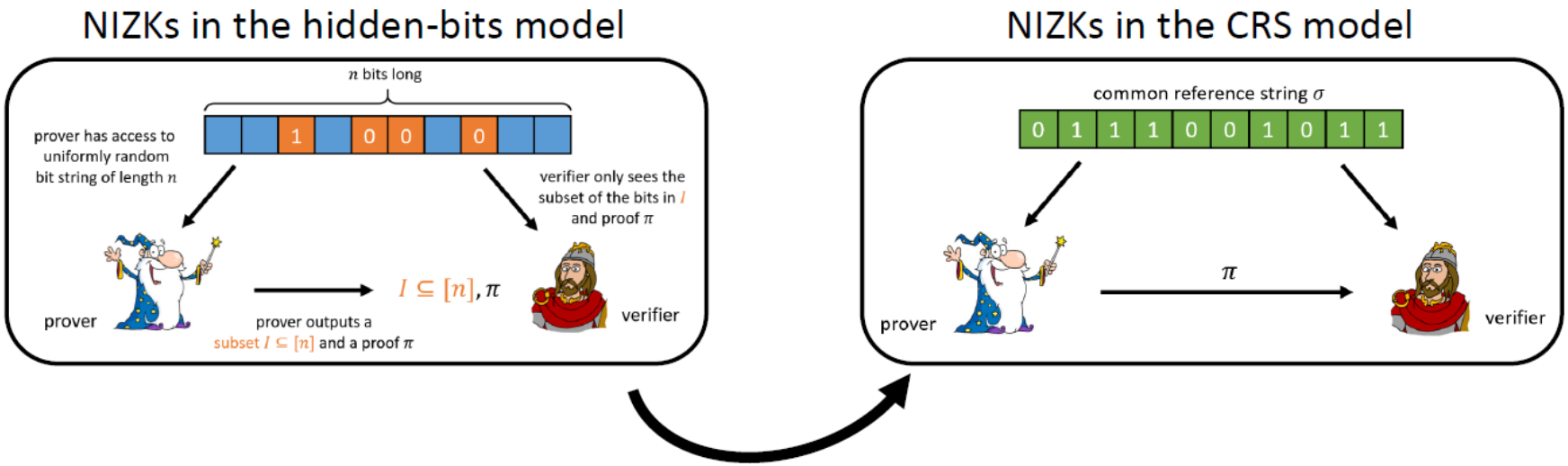
Non-interactive zero-knowledge proofs (NIZKs) are important primitives in cryptography. A major challenge since the early works on NIZKs has been to construct NIZKs with a *statistical* zero-knowledge guarantee against unbounded verifiers. In the common reference string (CRS) model, such "statistical NIZK arguments" are currently known from k-Lin in a pairing-group and from LWE. In the (reusable) designated-verifier model (DV-NIZK), where a trusted setup algorithm generates a reusable verification key for checking proofs, we also have a construction from DCR. If we relax our requirements to *computational* zero-knowledge, we additionally have NIZKs from factoring and CDH in a pairing group in the CRS model, and from nearly *all* assumptions that imply public-key encryption (e.g., CDH, LPN, LWE) in the designated-verifier model. Thus, there still remains a gap in our understanding of statistical NIZKs in both the CRS and the designated-verifier models.

In this work, we develop new techniques for constructing statistical NIZK arguments. First, we construct statistical DV-NIZK arguments from the k-Lin assumption in *pairing-free* groups, the QR assumption, and the DCR assumption. These are the first constructions in pairing-free groups and from QR that satisfy statistical zero-knowledge. All of our constructions are secure even if the verification key is chosen maliciously (i.e., they are "malicious-designated-verifier" NIZKs), and moreover, they satisfy a "dual-mode" property where the CRS can be sampled from two computationally indistinguishable distributions: one distribution yields *statistical DV-NIZK arguments* while the other yields *computational DV-NIZK proofs*. We then show how to adapt our k-Lin construction in a pairing group to obtain new *publicly-verifiable* statistical NIZK arguments from pairings with a *qualitatively weaker* assumption than existing constructions of pairing-based statistical NIZKs.

Our constructions follow the classic paradigm of Feige, Lapidot, and Shamir (FLS). While the FLS framework has traditionally been used to construct computational (DV)-NIZK proofs, we newly show that the same framework can be leveraged to construct dual-mode (DV)-NIZKs.

Dual-Mode NIZKs

- The idea of their work:



This work: dual-mode hidden bits generator

- "Binding mode:" computational DV-NIZK proofs
- "Hiding mode:" statistical DV-NIZK arguments

Witness Indistinguishability

- We don't have any construction for statistical NIZKs for NP in the standard model.

$\rightarrow$ So we use a relaxed property: Witness Indistinguishability (WI)

- WI is an extremely useful relaxation of ZK: The interaction does not reveal which of the NP-witnesses for $x \in L$ was used in the proof

– **witness-indistinguishable**: $\forall w_1, w_2$:

$$(P(w_1), V^*) \stackrel{c}{\approx} (P(w_2), V^*)$$

– **witness independent**: $\forall w_1, w_2$:

$$(P(w_1), V^*) \stackrel{s}{\approx} (P(w_2), V^*)$$

Witness Indistinguishability

- Recall NP-Relations:
  $L \in NP$ if $\exists$poly-time recognizable relation R_L so that

$$x \in L \Leftrightarrow \exists w : (x, w) \in R_L$$

- Set of NP-witnesses for $x \in L$:

$$R_L(x) = \{w \mid (x, w) \in R_L\} = \{w \mid V(x, w) = 1\}$$

* $L \in NP$ can have many different NP-relations R_L.

Witness Indistinguishability

Witness Indistinguishable

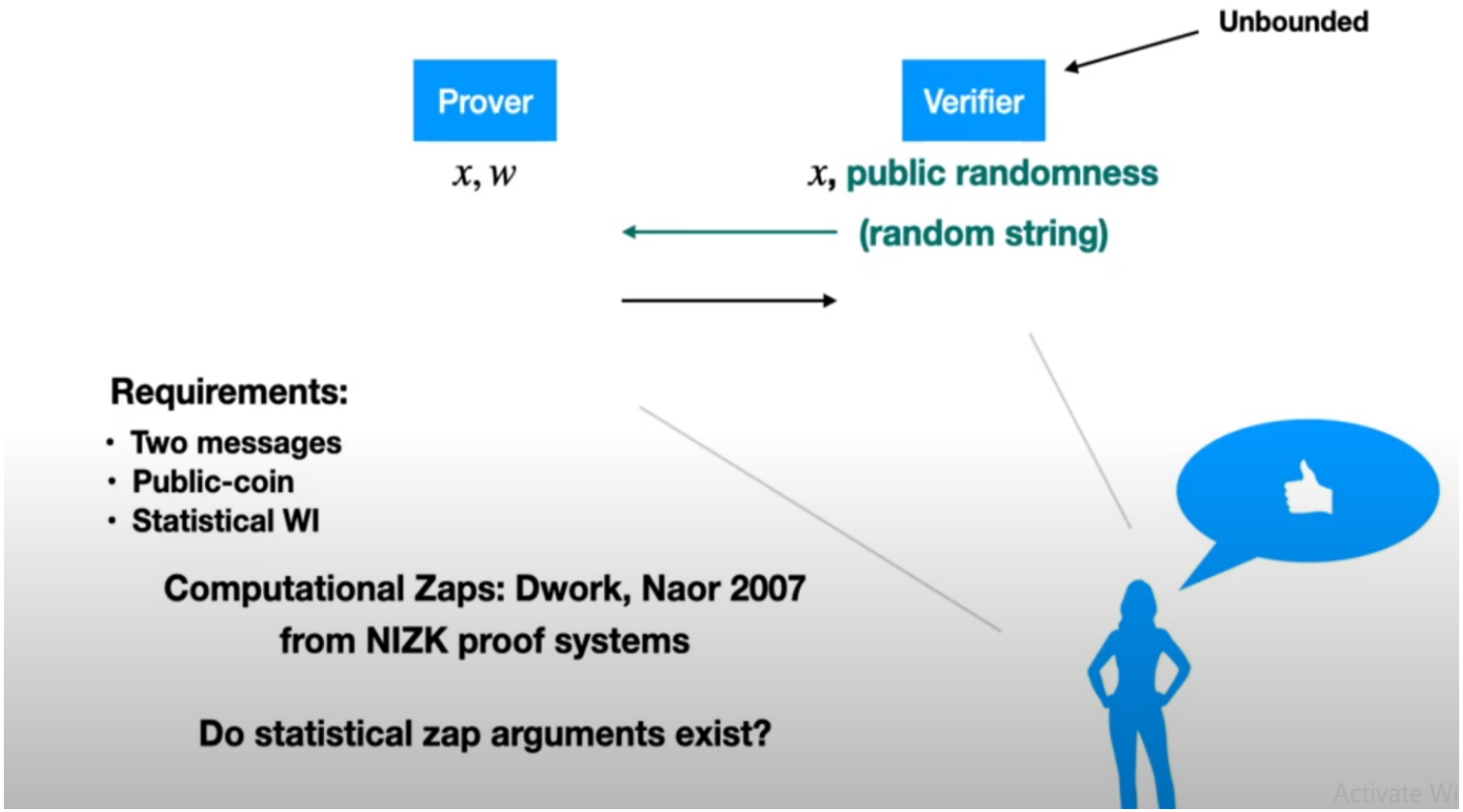
(P, V) is witness indistinguishable with respect to NP-relation R_L if $\forall \text{PPT} V^*, \forall x \in L, \forall w_1, w_2 \in R_L(x)$:

$$(P(w_1), V^*)(x) \overset{c}{\approx} (P(w_2), V^*)(x)$$

* Every ZKP is also WI.

ZAP Arguments

Statistical Zap Arguments



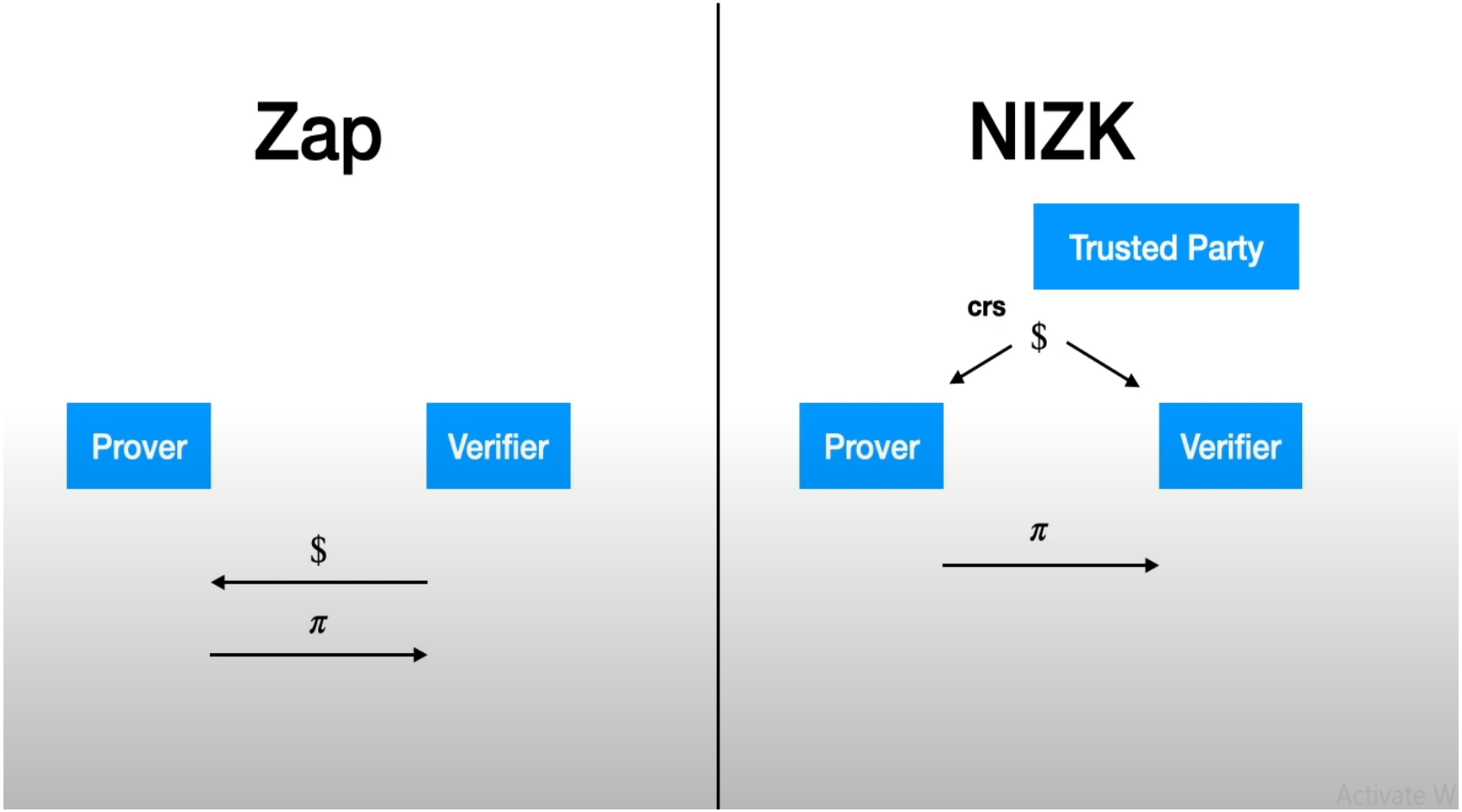
Requirements:

- Two messages
- Public-coin
- Statistical WI

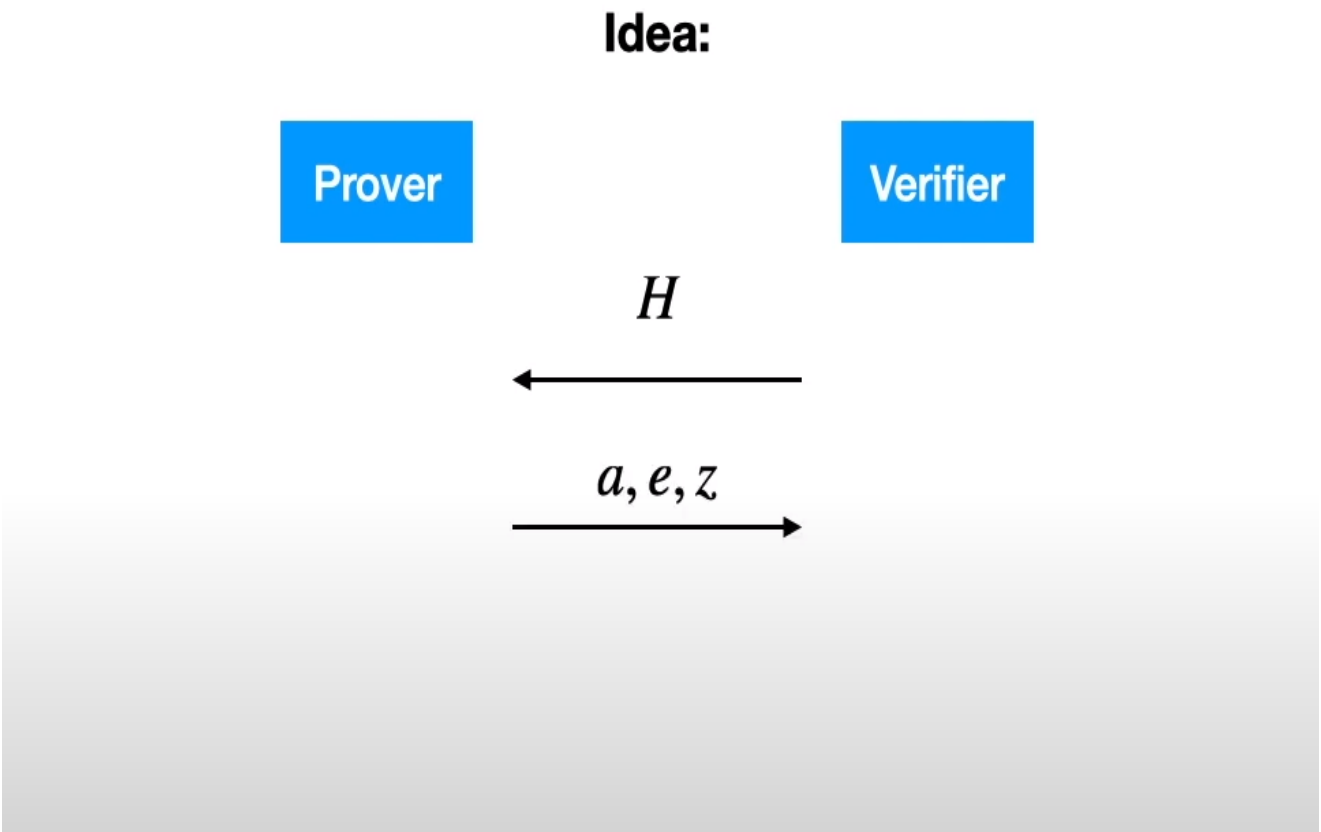
Computational Zaps: Dwork, Naor 2007
from NIZK proof systems

Do statistical zap arguments exist?

ZAP Arguments



ZAP Arguments



Thank you!