Publicly Verifiable Zero-Knowledge from *(Collapsing)* Blockchains

Alessandra Scafuro
NC State

Luisa Siniscalchi
Aarhus University

Ivan Visconti
University of Salerno

Publicly Verifiable Zero-Knowledge Proofs

+ **One** proof convinces ***all*** verifiers π

Publicly Verifiable Zero-Knowledge Proofs

+ **One** proof convinces ***all*** verifiers π

Soundness: only proof of true statements are accepted by a verifier

Zero-Knowledge: nothing besides truthfulness of the theorem is revealed

Publicly Verifiable Zero-Knowledge Proofs

+ **One** proof convinces ***all*** verifiers π

Soundness: only proof of true statements are accepted by a verifier

Zero-Knowledge: nothing besides truthfulness of the theorem is revealed

+ Public verifiability especially desired in blockchain applications

Publicly Verifiable Zero-Knowledge Proofs

requires setup assumptions

PVZK ≈ Non-interactive ZK (NIZK)

- [**trusted*** party] CRS
- [heuristic] Random Oracle

Publicly Verifiable Zero-Knowledge Proofs

blockchain
requires ~~setup~~ assumptions

PVZK ≈ Non-interactive ZK (NIZK)

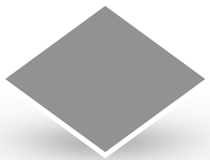
- [**trusted*** party] CRS
- [heuristic] Random Oracle

Publicly Verifiable Zero-Knowledge Proofs

blockchain
requires ~~setup~~ assumptions

PVZK ≈ Non-interactive ZK (NIZK)

Blockchain protocols
with
their underlying assumption



Blockchain Assumption to Replace Trusted Setups

Public verifiable ZK from Blockchain Assumption?

Blockchain Assumption to Replace Trusted Setups

Public verifiable ZK from Blockchain Assumption?

[Goyal Goyal TCC 2017]

GG-NIZK: Non-interactive ZK from a Proof-of-Stake Blockchain**

Blockchain Assumption to Replace Trusted Setups

Public verifiable ZK from Blockchain Assumption?

[Goyal Goyal TCC 2017]

GG-NIZK: Non-interactive ZK from a Proof-of-Stake Blockchain**

** Additional Limitations on Adversary

** Additional Assumptions on the stakeholder behaviour

Our contribution

1

2

Our contribution

1

Subtleties in using a blockchain as an assumption

a single static adv playing as blockchain user ***invalidates*** *ZK of GG-NIZK*

2

Our contribution

1. Subtleties in using a blockchain as an assumption

   a single static adv playing as blockchain user ***invalidates*** *ZK of GG-NIZK*

2. PVZK from a “generic blockchain assumption”

   it remains zk even if ***all blockchain players*** *are eventually* ***corrupt***
   (collapsing blockchain)

1 Subtleties in using a blockchain as an assumption

a single adv playing as blockchain user ***invalidates*** *ZK of* ***GG-NIZK***

GG-NIZK from Proof-of-Stake Blockchain

GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake blockchain assumption

at any point in time, if you look at the total current stake, the majority belongs to honest people

GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake blockchain assumption

at any point in time, if you look at the total current stake, the majority belongs to honest people

=> adv cannot create forks

GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake blockchain assumption

at any point in time, if you look at the total current stake, the majority belongs to honest people

=> adv cannot create forks

NIZK from Proof-of-Stake [GG17]

GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake blockchain assumption

at any point in time, if you look at the total current stake, the majority belongs to honest people

=> adv cannot create forks

NIZK from Proof-of-Stake [GG17]

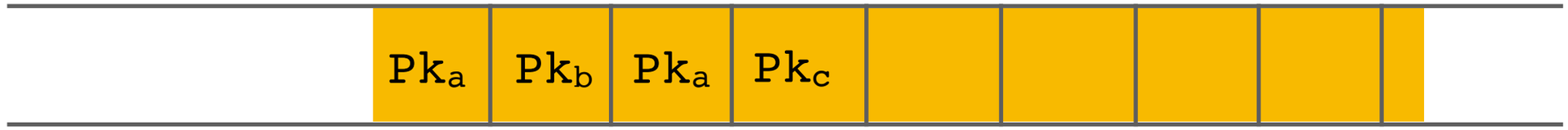
NI Witness Indistinguishability —> NI Zero-Knowledge

trapdoor theorem

“x in L” OR "I computed a fork”

1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)



1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)

"x in L" OR "I know a fork of length K"

1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)

"x in L" OR "I know a fork of length K"

1. w -> w_1 w_2 ..... w_n

1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)

"x in L" OR "I know a fork of length K"

1. w -> w_1 w_2 ..... w_n

2. Encrypt with stakeholders keys published on the BC

c_1 c_2 ..... c_n

1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)

"x in L" OR "I know a fork of length K"

1. w -> w_1 w_2 ..... w_n

2. Encrypt with stakeholders keys published on the BC

c_1 c_2 ..... c_n

3. NIWI proof NIWI

"These are encryptions under stakeholders keys of shares of the witness

1 GG-NIZK from Proof-of-Stake Blockchain

Proof-of-Stake Blockchain (param K)

"x in L" OR "I know a fork of length K"

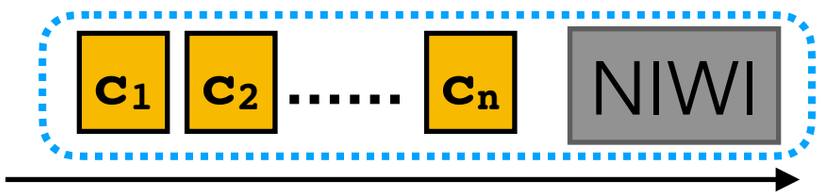
1. w -> w_1 w_2 ..... w_n

2. Encrypt with stakeholders keys published on the BC

c_1 c_2 ..... c_n

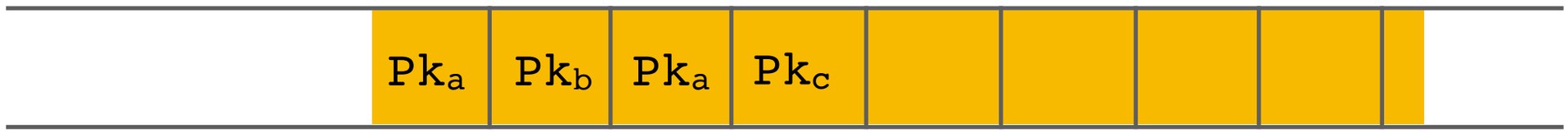
3. NIWI proof NIWI

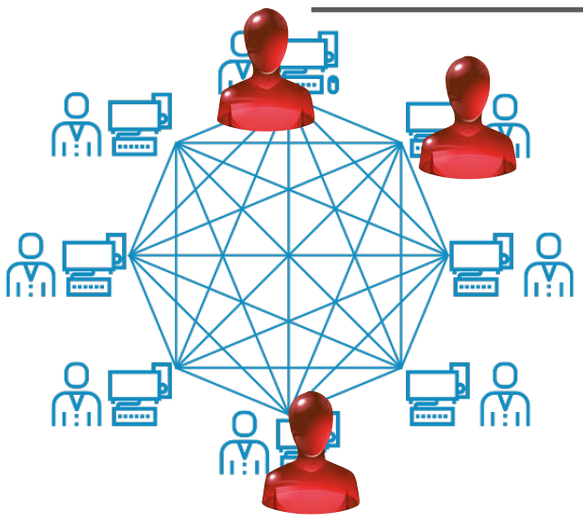
Public Proof

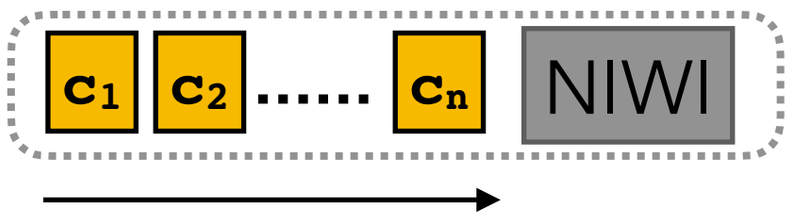


"These are encryptions under stakeholders keys of shares of the witness

1 Soundness of GG-NIZK

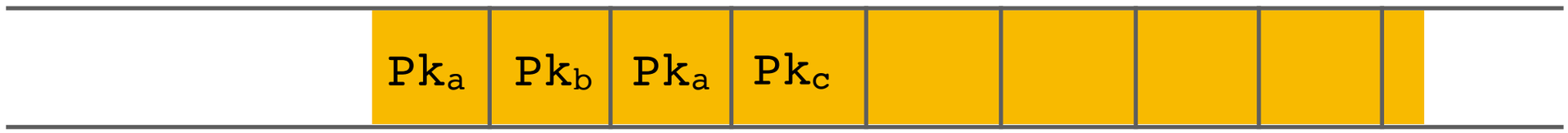


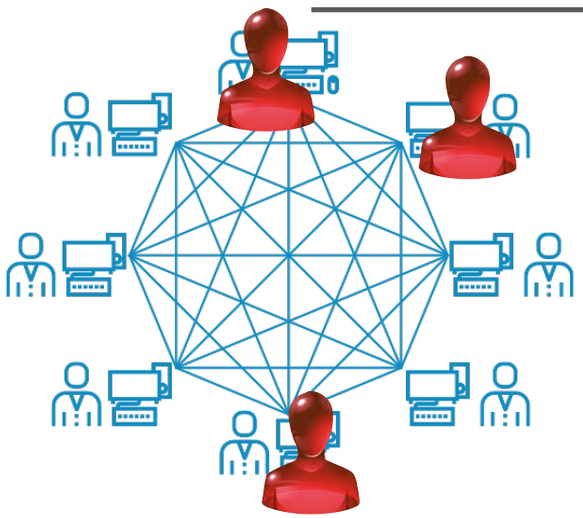


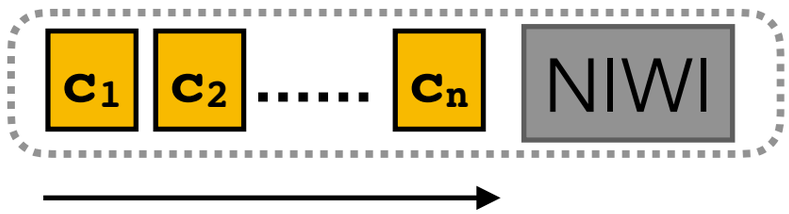


Soundness “x ∉ L” OR “I computed a fork”

1 Soundness of GG-NIZK







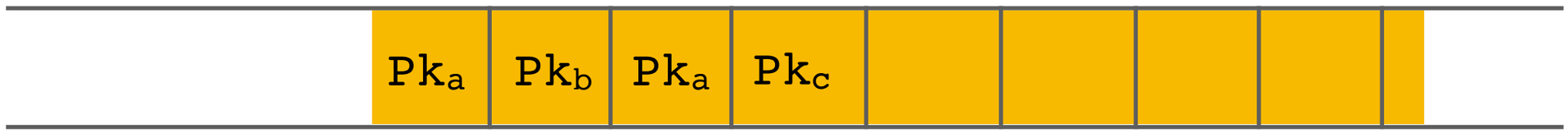
Soundness “x il L” OR “I computed a fork”

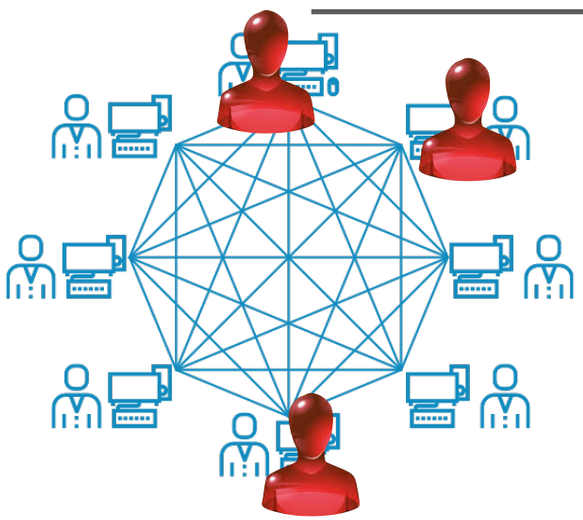
Assumptions:

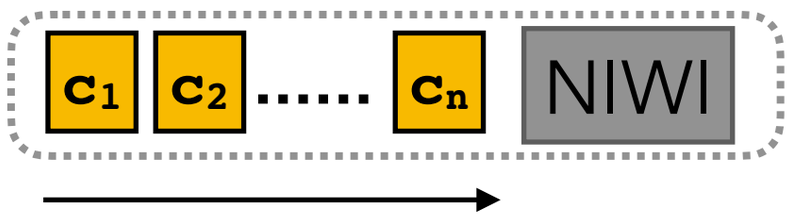
Soundness NIWI

+ honest majority of stake

1 Soundness of GG-NIZK







Soundness

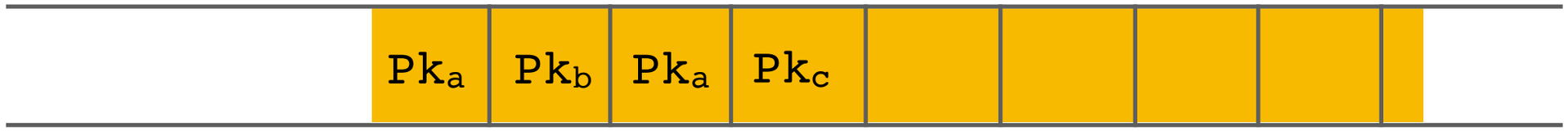
"x ∉ L" OR ~~"I computed a fork"~~

Assumptions:

Soundness NIWI

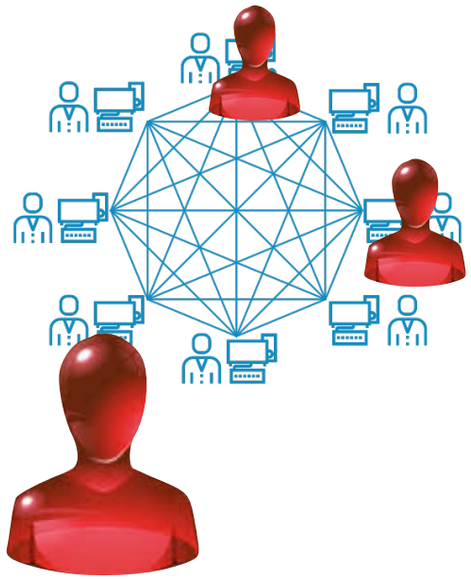
+ honest majority of stake

1 Zero-Knowledge of GG-NIZK

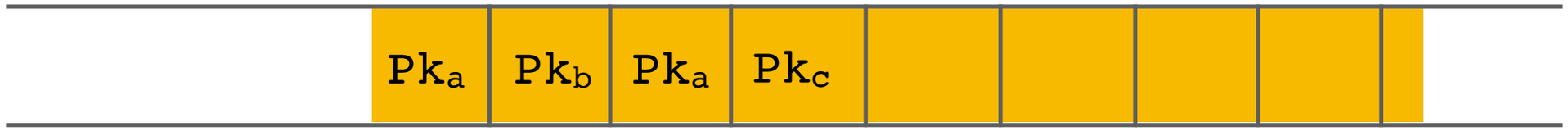


Simulator

"x in L" OR "I know a fork of length K"

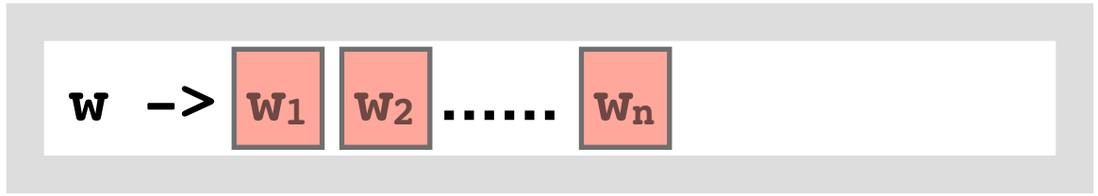


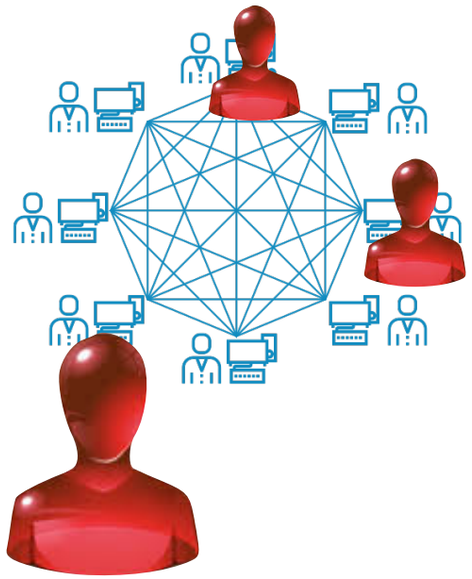
1 Zero-Knowledge of GG-NIZK



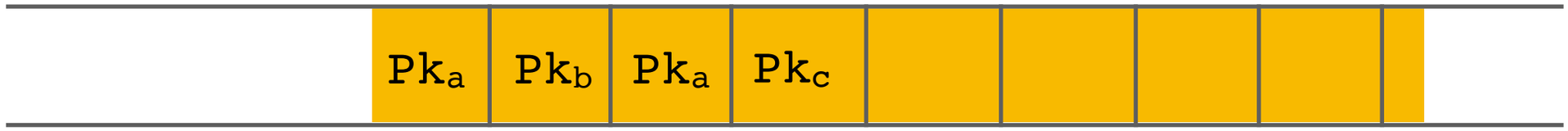
Simulator

"x in L" OR "I know a fork of length K"



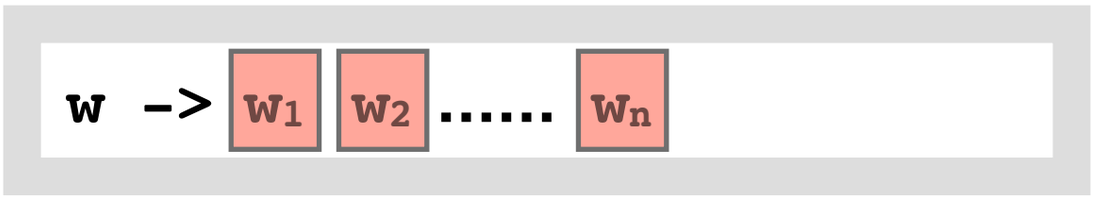


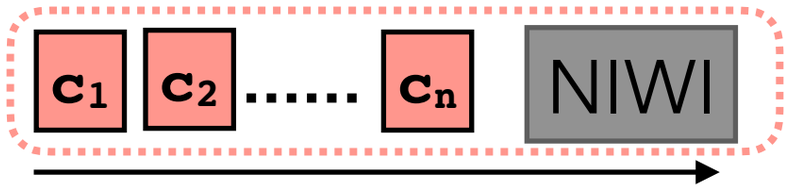
1 Zero-Knowledge of GG-NIZK

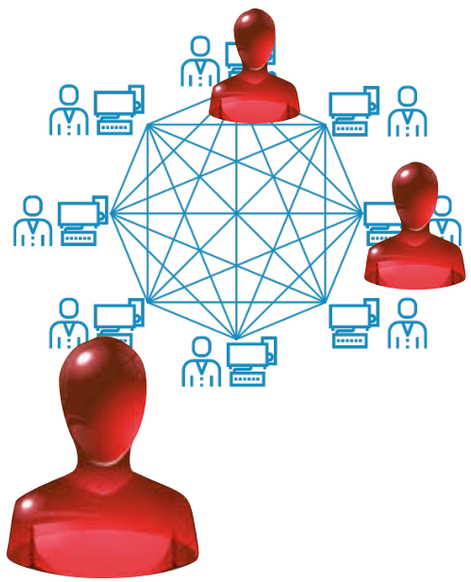


Simulator

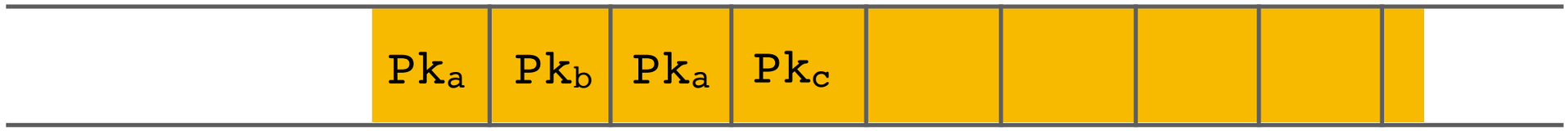
"x in L" OR "I know a fork of length K"





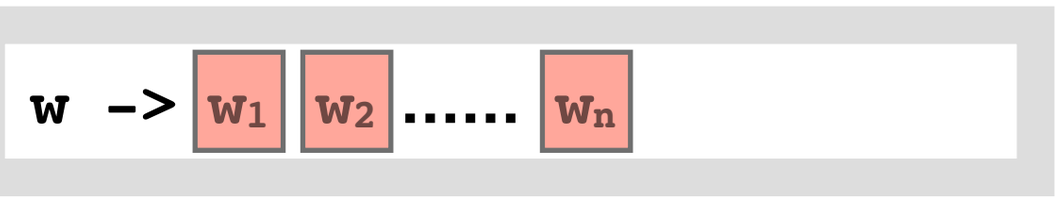


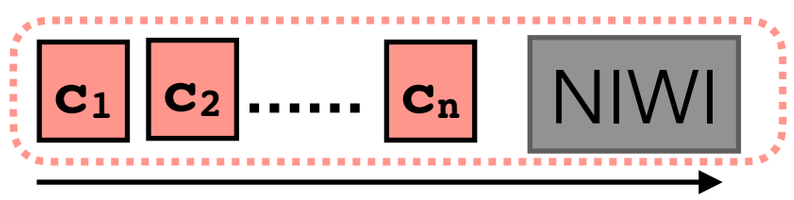
1 Zero-Knowledge of GG-NIZK

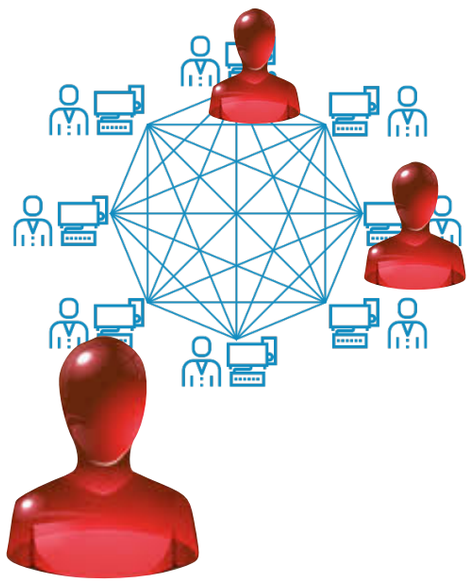


Simulator

"x in L" OR "I know a fork of length K"



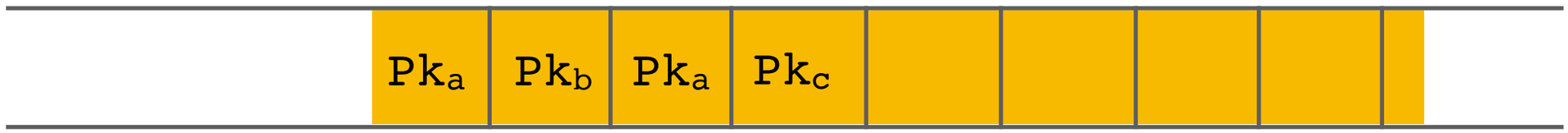




Witness Indistinguishability, CPA security

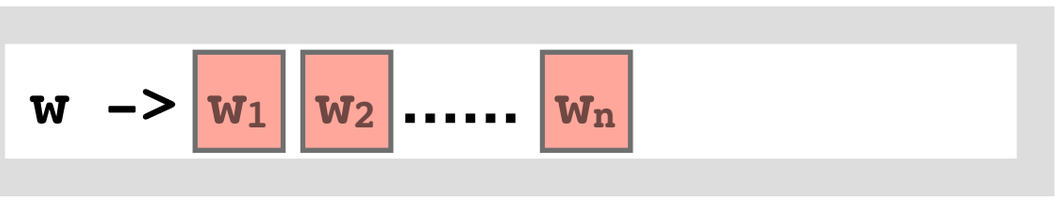
+ honest majority of stake

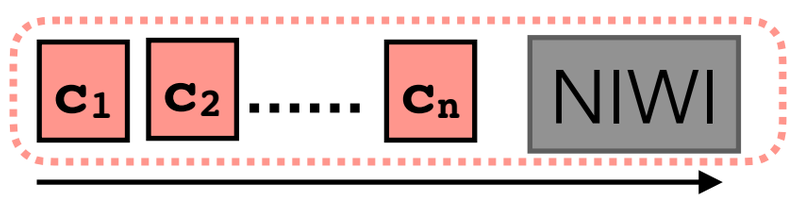
1 Zero-Knowledge of GG-NIZK

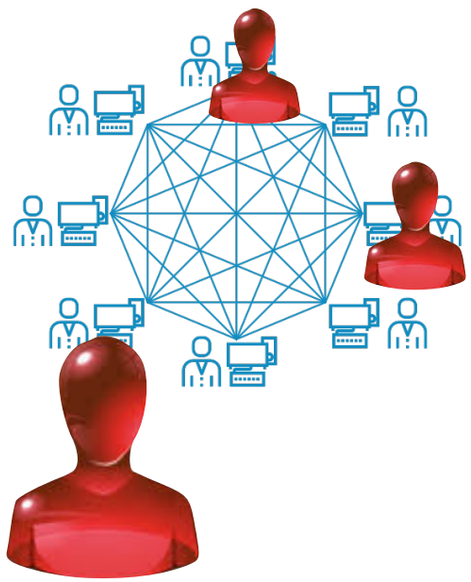


Simulator

"x in L" OR "I know a fork of length K"







Witness Indistinguishability, CPA security

+ honest majority of stake

Additional Assumptions:

❌ No adaptive corruption of blockchain players

❌ Honest stakeholders never reveal their stake key

Our contribution

1 Subtleties in using a blockchain as an assumption

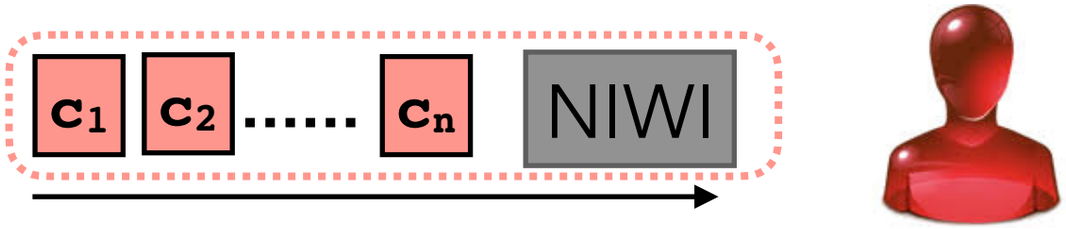
*a single adv playing as blockchain user **invalidates** ZK of GG-NIZK*

1 Invalidating GG-NIZK zero-knowledge property

Assume all the restrictions are satisfied.

❌ No adaptive corruption of blockchain players

❌ Honest stakeholders never reveal their stake key

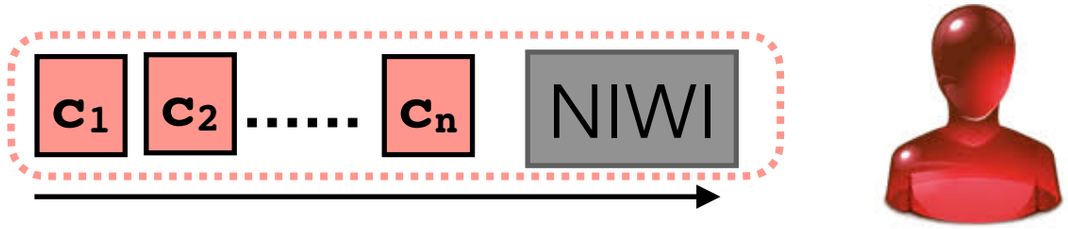


1 Invalidating GG-NIZK zero-knowledge property

Assume all the restrictions are satisfied.

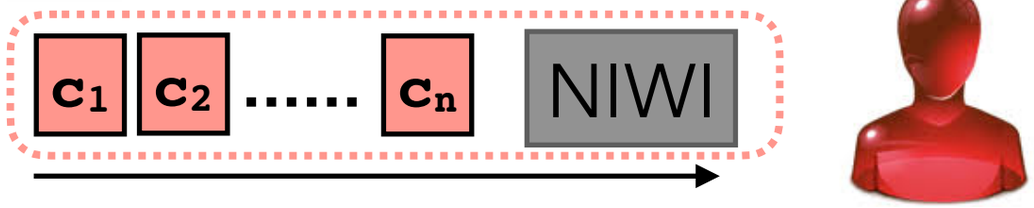
❌ No adaptive corruption of blockchain players

❌ Honest stakeholders never reveal their stake key

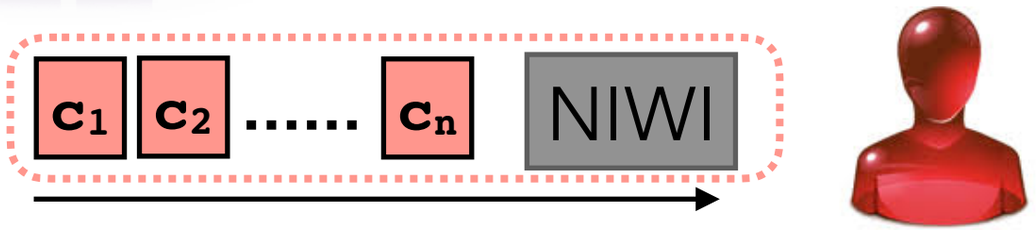


ZK can be violate by playing as a ***user*** of the Blockchain with a legitimate *smart contract*

1 Invalidating GG-NIZK zero-knowledge property



1 Invalidating GG-NIZK zero-knowledge property

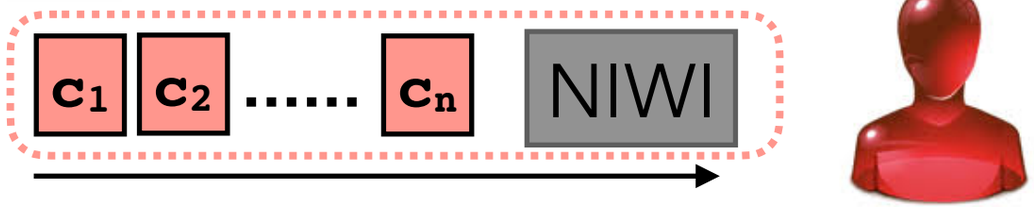


DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert $\mathsf{Ledger}[\mathcal{C}] > \$\mathsf{reward}$.
   - $\mathsf{Ledger}[\mathcal{C}] := \mathsf{Ledger}[\mathcal{C}] - \$\mathsf{reward}$.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If $\mathsf{ctx} = \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards $\mathsf{Ledger}[\mathsf{Pt}] := \mathsf{Ledger}[\mathsf{Pt}] + \$\mathsf{reward}$.
   - Set state := CLAIMED.

1 Invalidating GG-NIZK zero-knowledge property



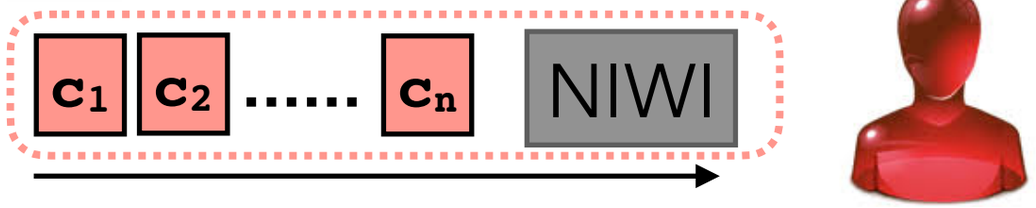
DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert $\mathsf{Ledger}[\mathcal{C}] > \$\mathsf{reward}$.
   - $\mathsf{Ledger}[\mathcal{C}] := \mathsf{Ledger}[\mathcal{C}] - \$\mathsf{reward}$.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If $\mathsf{ctx} = \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards $\mathsf{Ledger}[\mathsf{Pt}] := \mathsf{Ledger}[\mathsf{Pt}] + \$\mathsf{reward}$.
   - Set state := CLAIMED.

Observations **Honest Stakeholder** **Adversary**

1 Invalidating GG-NIZK zero-knowledge property



DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert Ledger$[\mathcal{C}] >$ $reward.
   - Ledger$[\mathcal{C}]$:= Ledger$[\mathcal{C}]$ − $reward.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If ctx $= \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards Ledger[Pt] := Ledger[Pt] + $reward.
   - Set state := CLAIMED.

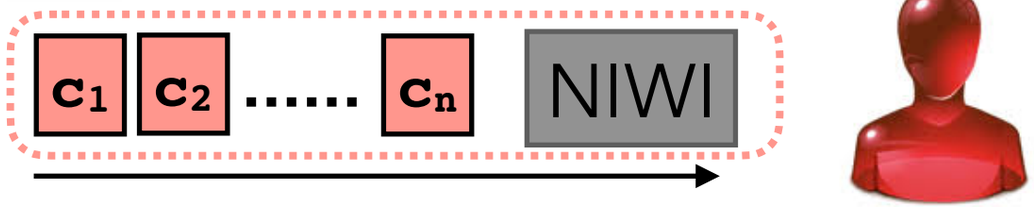
Observations

Honest Stakeholder

- not reveling key

Adversary

1 Invalidating GG-NIZK zero-knowledge property



DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert $\mathsf{Ledger}[\mathcal{C}] > \$\mathsf{reward}$.
   - $\mathsf{Ledger}[\mathcal{C}] := \mathsf{Ledger}[\mathcal{C}] - \$\mathsf{reward}$.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If $\mathsf{ctx} = \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards $\mathsf{Ledger}[\mathsf{Pt}] := \mathsf{Ledger}[\mathsf{Pt}] + \$\mathsf{reward}$.
   - Set state := CLAIMED.

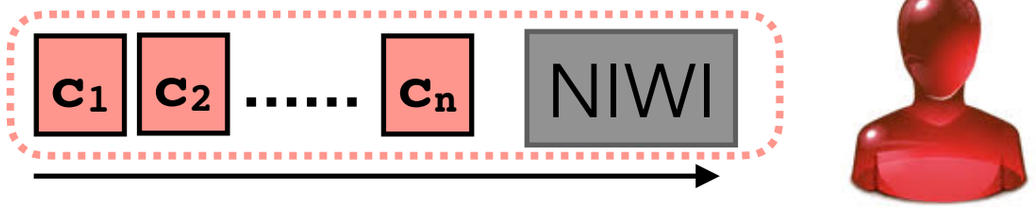
Observations

Honest Stakeholder

- not reveling key
- unaware that this ctx is part of zk

Adversary

1 Invalidating GG-NIZK zero-knowledge property



DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert $\mathsf{Ledger}[\mathcal{C}] > \$\mathsf{reward}$.
   - $\mathsf{Ledger}[\mathcal{C}] := \mathsf{Ledger}[\mathcal{C}] - \$\mathsf{reward}$.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If $\mathsf{ctx} = \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards $\mathsf{Ledger}[\mathsf{Pt}] := \mathsf{Ledger}[\mathsf{Pt}] + \$\mathsf{reward}$.
   - Set state := CLAIMED.

Observations

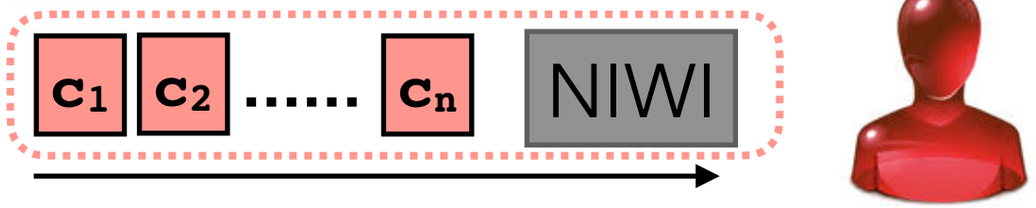
Honest Stakeholder

- not reveling key
- unaware that this ctx is part of zk

Adversary

- no corruption

1 Invalidating GG-NIZK zero-knowledge property



DecryptionForBarbados

1. Init: Upon receiving (init, $reward, ctx, PK_i) from a contractor $\mathcal{C}$:
   - Assert Ledger$[\mathcal{C}] >$ $reward.
   - Ledger$[\mathcal{C}]$:= Ledger$[\mathcal{C}]$ − $reward.
   - Set state := INIT.

2. Claim: On input (claim, v) from a player Pt_i:
   - Parse $v = (m, r)$.
   - If ctx $= \mathsf{Enc}_{\mathsf{PK}_i}(m, r)$ then set rewards Ledger[Pt] := Ledger[Pt] + $reward.
   - Set state := CLAIMED.

Observations

Honest Stakeholder

- not reveling key
- unaware that this ctx is part of zk

Adversary

- no corruption
- no bribing

1 Observations from our attack

A

B

1 Observations from our attack

A The long-term security of the **external protocol, should not depend on permanent secrets of blockchain players.**

B

1 Observations from our attack

A The long-term security of the **external protocol, should not depend on permanent secrets of blockchain players.**

B **No PVZK** is known from a blockchain assumption

Our contribution

1 Subtleties in using blockchain assumption without marrying the threat model

a single adv playing as blockchain user **invalidates** ZK of GG-NIZK

2 PVZK from a “generic blockchain assumption”

*it remains zk even if **all blockchain players** are eventually **corrupt** **(collapsing blockchain)***

2 Property of a Generic Blockchain

Chain-Quality

Any sequence of **N** consecutive blocks contains **K** blocks generated by honest players (where **N**, **K** are parameters that depend on adversarial resources).

2 Our Blockchain Assumption

Chain-Quality

Any sequence of **N** consecutive blocks contains **K** blocks generated by honest players (where **N**, **K** are parameters that depend on adversarial resources).

Our **Chain-Quality assumption**

2 Our Blockchain Assumption

Chain-Quality

Any sequence of **N** consecutive blocks contains **K** blocks generated by honest players (where **N**, **K** are parameters that depend on adversarial resources).

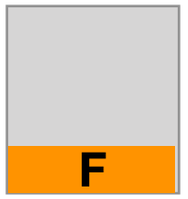
Our **Chain-Quality assumption**

Any sequence of **N** consecutive blocks contains at least **K** blocks generated by honest players and contain an *high-min entropy* string

2 A bit more concretely

Block

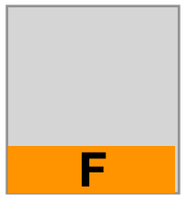
A block contains a distinguished field **F** (eg. `coinbase` transaction)



2 A bit more concretely

Block

A block contains a distinguished field **F** (eg. `coinbase` transaction)



Our **Assumption:**

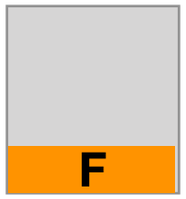
Any sequence of **N** consecutive blocks contains **K** blocks with *fresh* F and > 1/2 of them are generated by honest players and set to an high-min entropy string.

=> adv cannot predict too many **F** fields at *posting* time.

2 A bit more concretely

Block

A block contains a distinguished field **F** (eg. `coinbase` transaction)



Our **Assumption:**

Any sequence of **N** consecutive blocks contains **K** blocks with *fresh* F and > 1/2 of them are generated by honest players and set to an high-min entropy string.

 => adv cannot predict too many **F** fields at *posting* time.

Example Bitcoin:

Any sequence of **100** blocks contains **50 blocks with** fresh coinbase addresses. And at least 26 of them were created by honest miners.

2 Our PVZK protocol

Ingredients

2 Our PVZK protocol

Ingredients

- Any Blockchain satisfying our assumption

2 Our PVZK protocol

Ingredients

- Any Blockchain satisfying our assumption
- Statistically Binding Commitment (OWP)

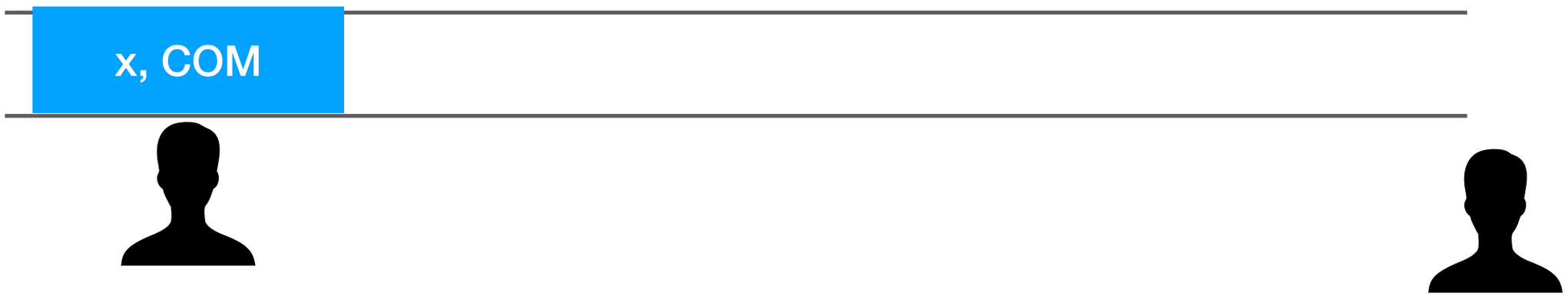
2 Our PVZK protocol

Ingredients

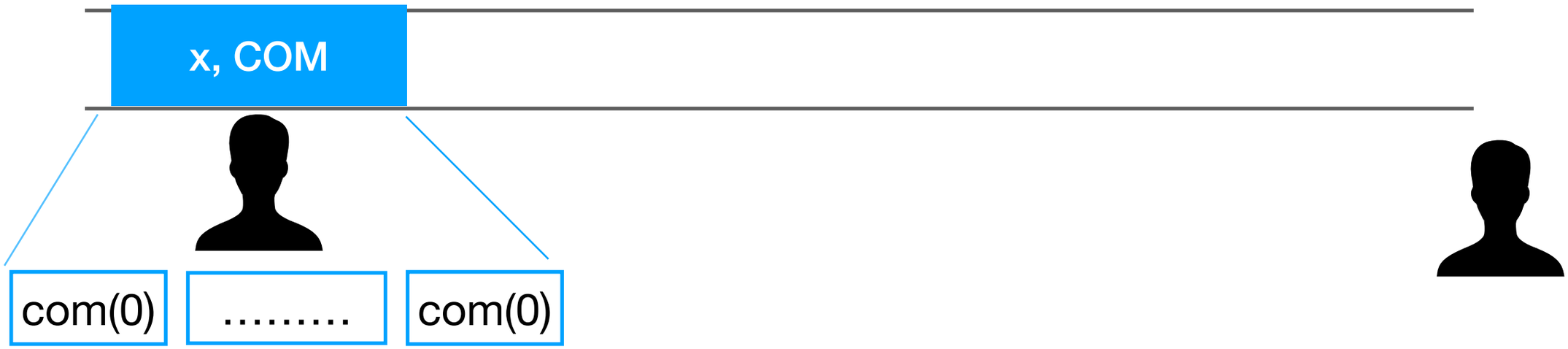
- Any Blockchain satisfying our assumption
- Statistically Binding Commitment (OWP)
- Publicly Verifiable Witness Indistinguishable [SSV19] (OWP)
  - ☑ Witness Indistinguishable even if the **Blockchain Collapse**
  - ☑ It can be based on any blockchain that satisfies our assumption

2 Our PVZK protocol

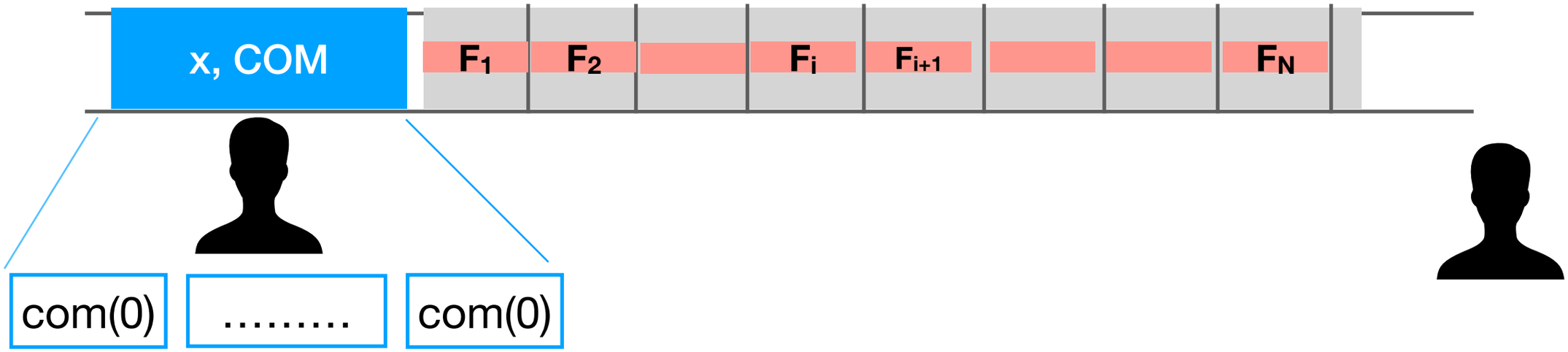
2 Our PVZK protocol



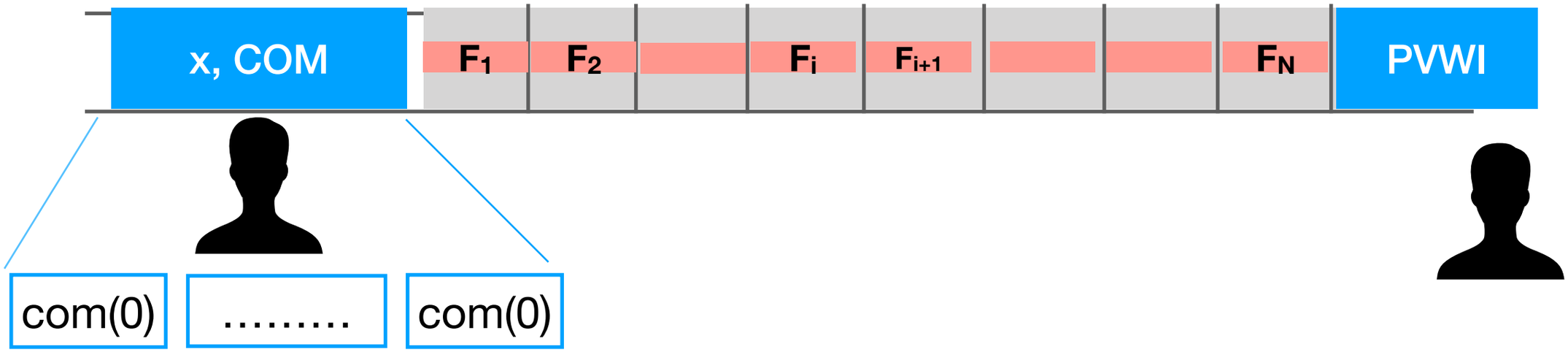
2 Our PVZK protocol



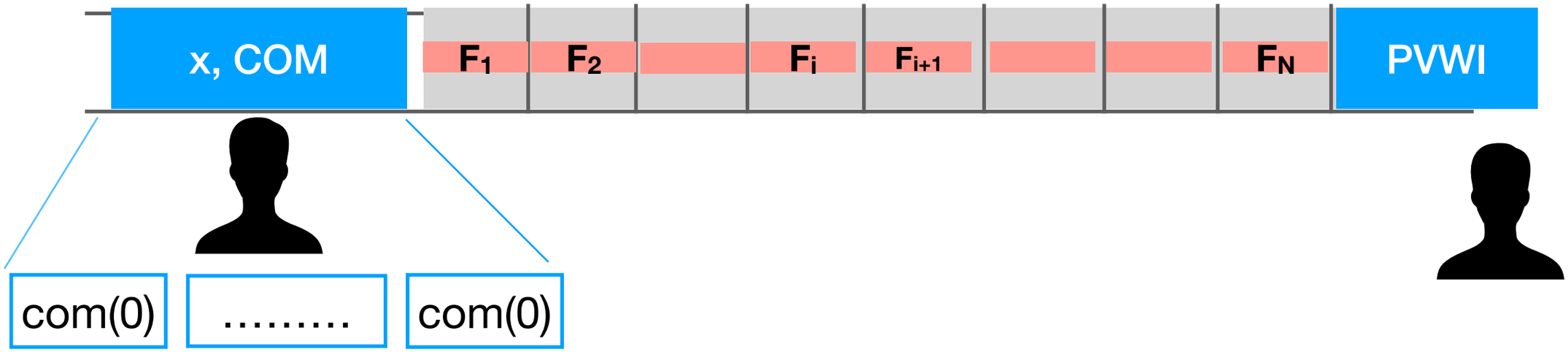
2 Our PVZK protocol



2 Our PVZK protocol



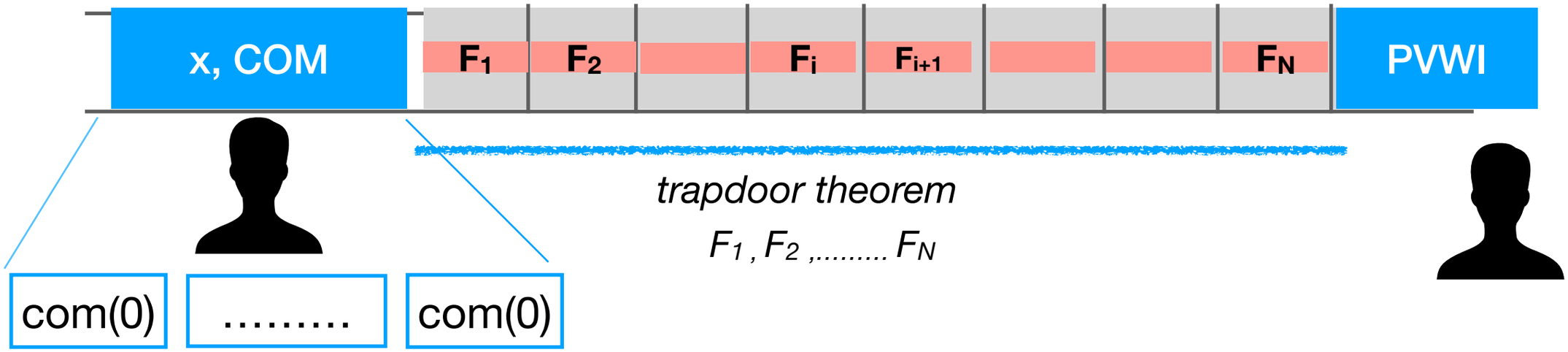
2 Our PVZK protocol



trapdoor theorem

TH: “x in L” OR “I will predict the next K/2+1 field in the BC”

2 Our PVZK protocol



trapdoor theorem

TH: “x in L” OR “I will predict the next K/2+1 field in the BC”

2

Our PVZK protocol

trapdoor theorem

TH: “x in L” OR “I will predict the next K/2+1 field in the BC”

2

Our PVZK protocol

trapdoor theorem

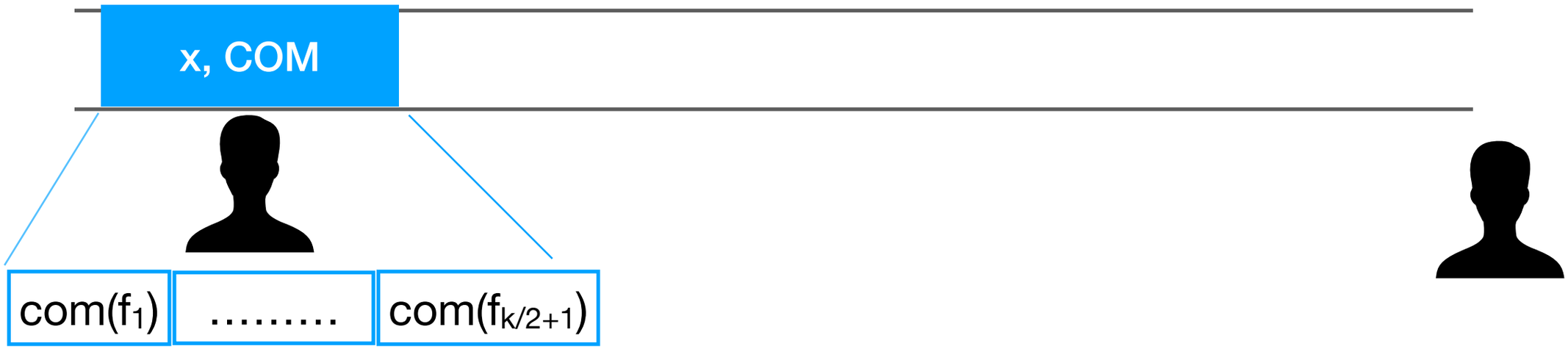
TH: “x in L” OR “I will predict the next K/2+1 field in the BC”

x, COM

2 Our PVZK protocol

trapdoor theorem

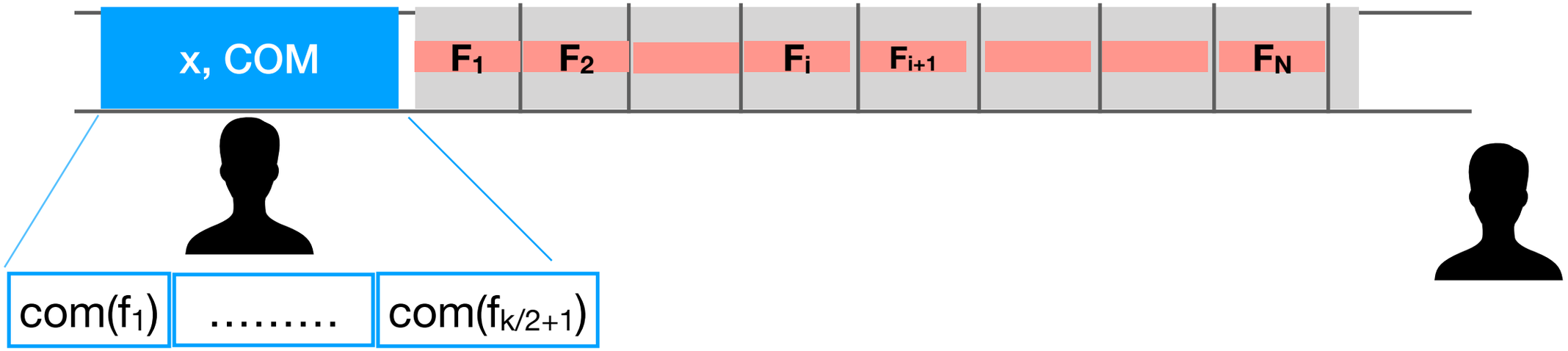
TH: “x in L” OR “I will predict the next K/2+1 field in the BC”



2 Our PVZK protocol

trapdoor theorem

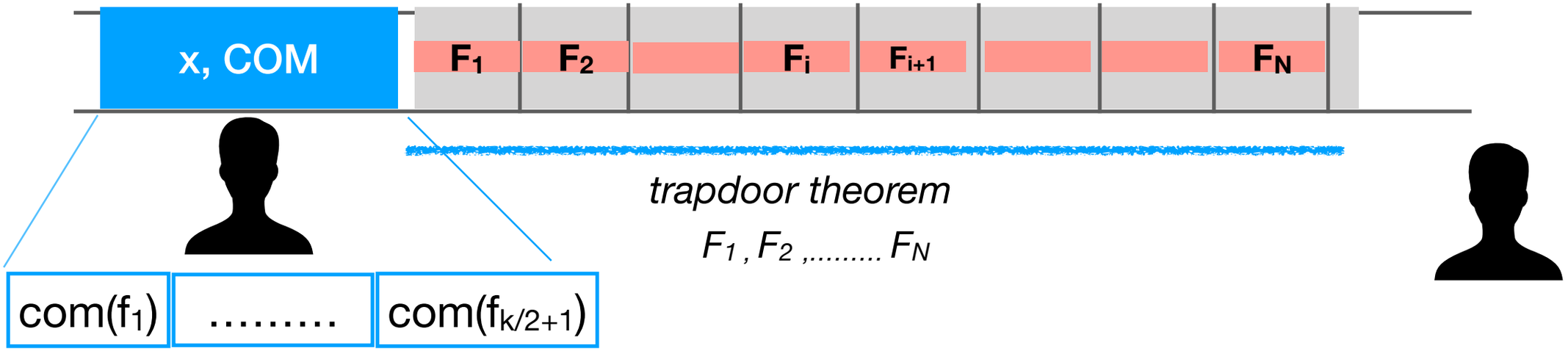
TH: "x in L" OR "I will predict the next K/2+1 field in the BC"



2 Our PVZK protocol

trapdoor theorem

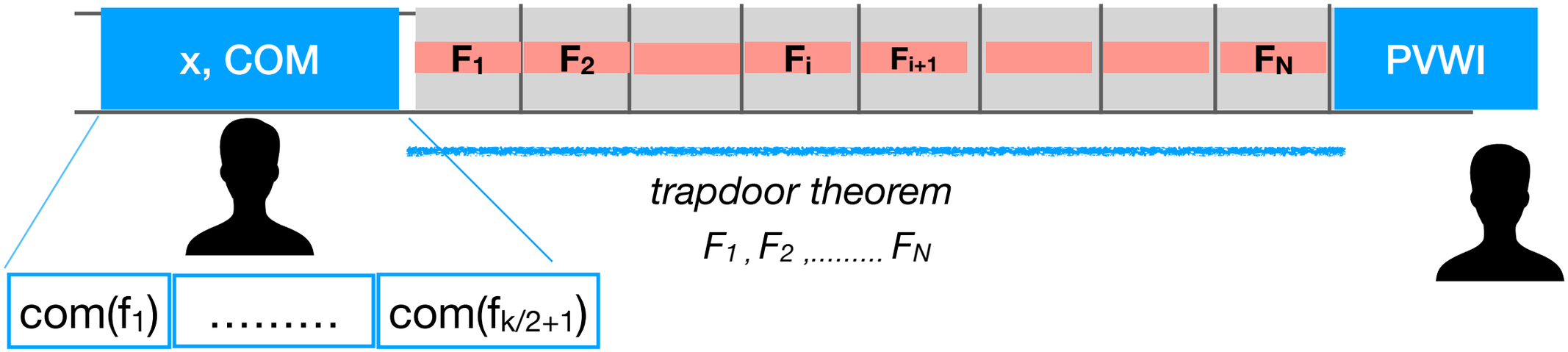
TH: "x in L" OR "I will predict the next K/2+1 field in the BC"



2 Our PVZK protocol

trapdoor theorem

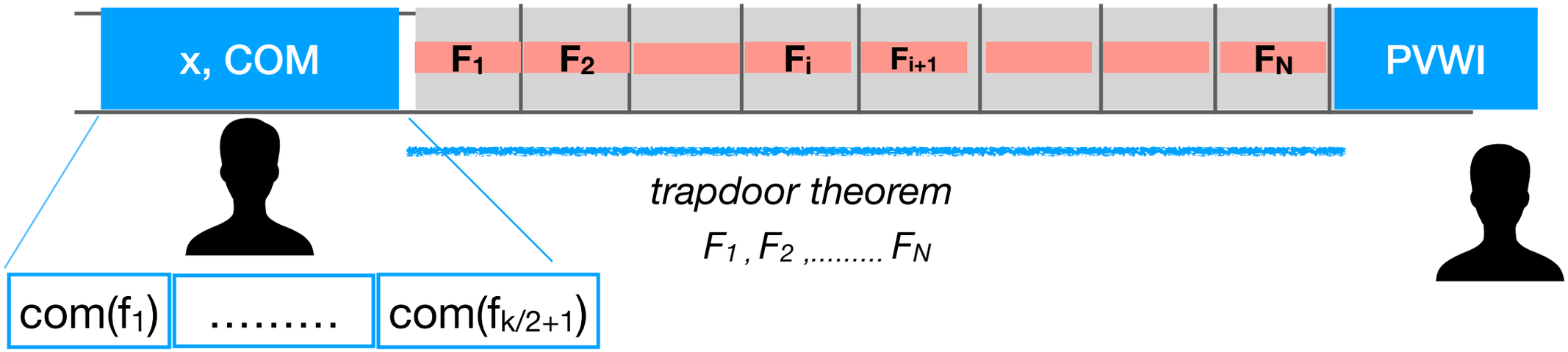
TH: "x in L" OR "I will predict the next K/2+1 field in the BC"



2 Our PVZK protocol

trapdoor theorem

TH: "x in L" OR "I will predict the next K/2+1 field in the BC"

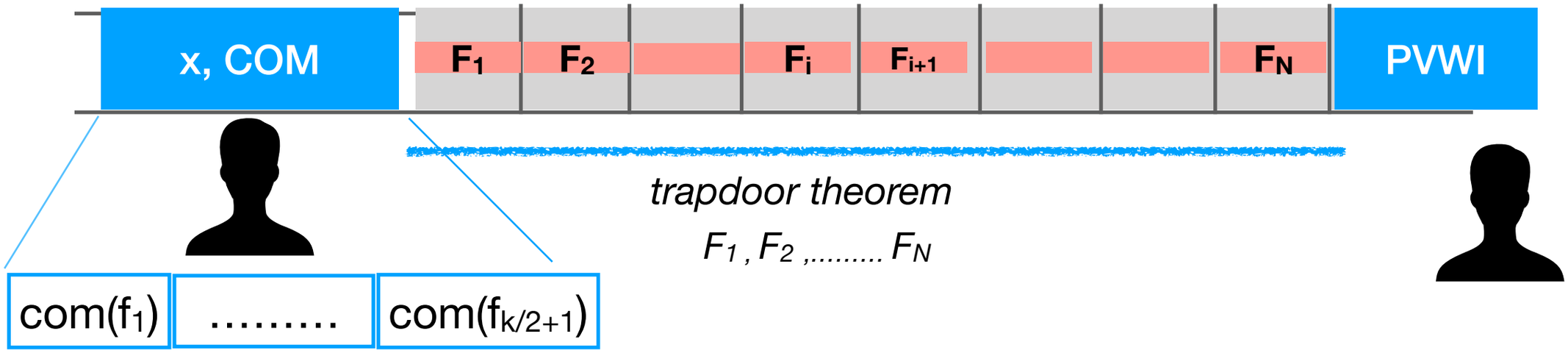


Zero-Knowledge: Simulator

2 Our PVZK protocol

TH: trapdoor theorem

"x in L" OR "I will predict the next K/2+1 field in the BC"



Zero-Knowledge: Simulator

it controls honest parties

=> *Sets the majority of the random fields F*

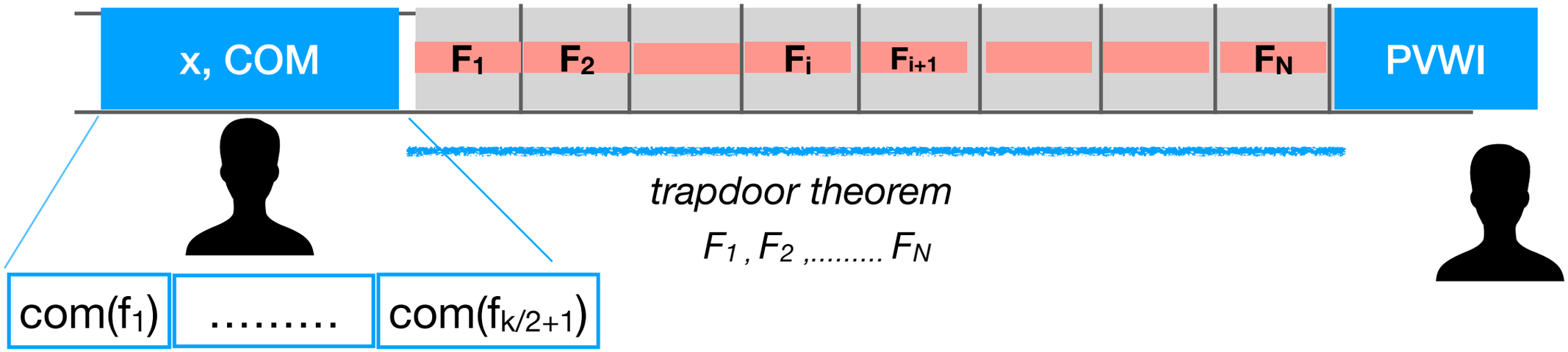
=> *Uses WI with the trapdoor witness*

=> *Indistinguishable due to Com (and WI)*

2 Our PVZK protocol

trapdoor theorem

TH: "x in L" OR "I will predict the next K/2+1 field in the BC"



Zero-Knowledge: Simulator

it controls honest parties

=> *Sets the majority of the random fields F*

=> *Uses WI with the trapdoor witness*

=> *Indistinguishable due to Com (and WI)*

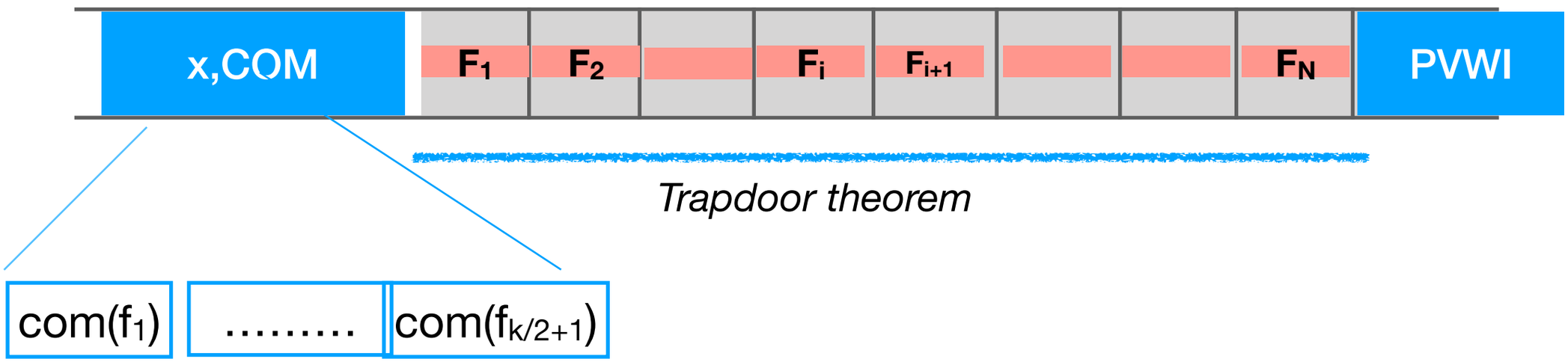
Soundness

follow from our assumption

+

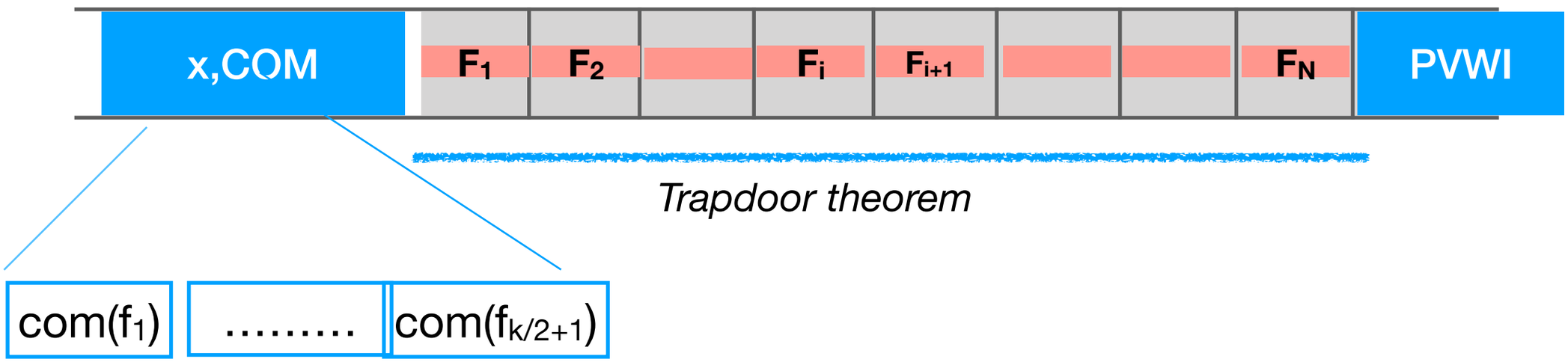
Statistical Soundness of WI

2 ZK even if blockchain Collapses!



Zero-Knowledge in Presence of Blockchain Collapse!

2 ZK even if blockchain Collapses!



Zero-Knowledge in Presence of Blockchain Collapse!

knowing private states and *keys of all the blockchain players*
does not help breaking commitments neither forward WI.

Conclusion: We show

Our PVZK vs GG-NIZK

	Completely non-interactive	**Type of Blockchain**	**Complexity Assumptions**	**Further restrictions on a) consensus protocol, b) stake transfer protocol, c) smart contracts**
[GG17]	Yes	PoS blockchain	NIWI	The honest stakeholders should not: -reveal their secrets even with 0 stake -participate in other applications
PVZK	No - P writes messages in the blockchain	Any blockchain satisfying some assumption	OWPs	None

Thanks