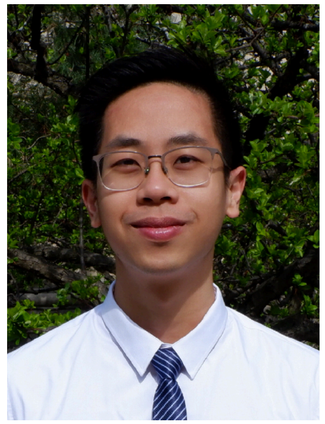
Spartan & Bulletproofs are Simulation-Extractable (for Free!)

Quang Dao
CMU



Paul Grubbs
Michigan

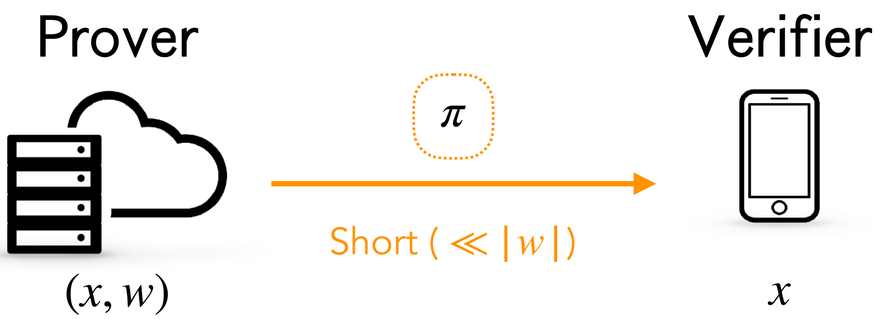
Eurocrypt 2023

zkSNARKs: Security & Use Cases

zkSNARKs: Security & Use Cases

(Zero-knowledge Succinct Non-interactive ARguments of Knowledge)

short, non-interactive proofs

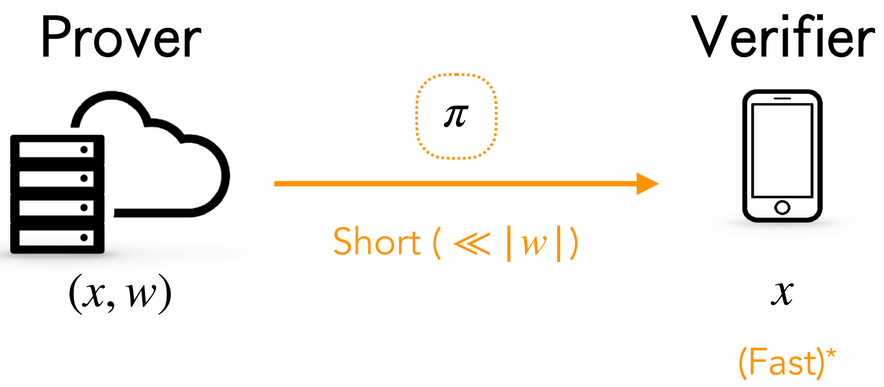


*For this talk, zkSNARKs may be without fast verification.

zkSNARKs: Security & Use Cases

(Zero-knowledge Succinct Non-interactive ARguments of Knowledge)

short, *non-interactive* *proofs*



Knowledge Soundness: If V accepts, then P must "know" w.

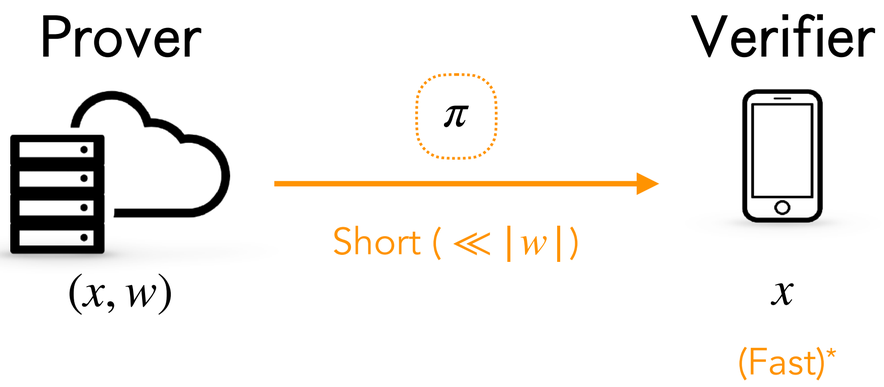
Zero-Knowledge: π hides w.

*For this talk, zkSNARKs may be without fast verification.

zkSNARKs: Security & Use Cases

(Zero-knowledge Succinct Non-interactive ARguments of Knowledge)

short, *non-interactive* *proofs*



Knowledge Soundness: If V accepts, then P must "know" w.

Zero-Knowledge: π hides w.

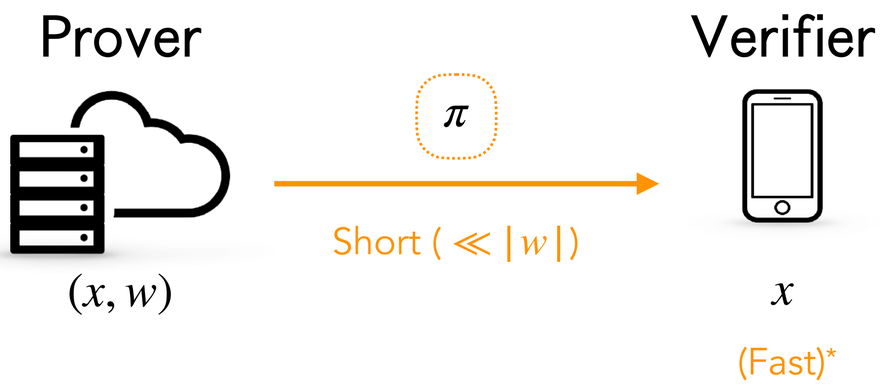
Applications in blockchains:

*For this talk, zkSNARKs may be without fast verification.

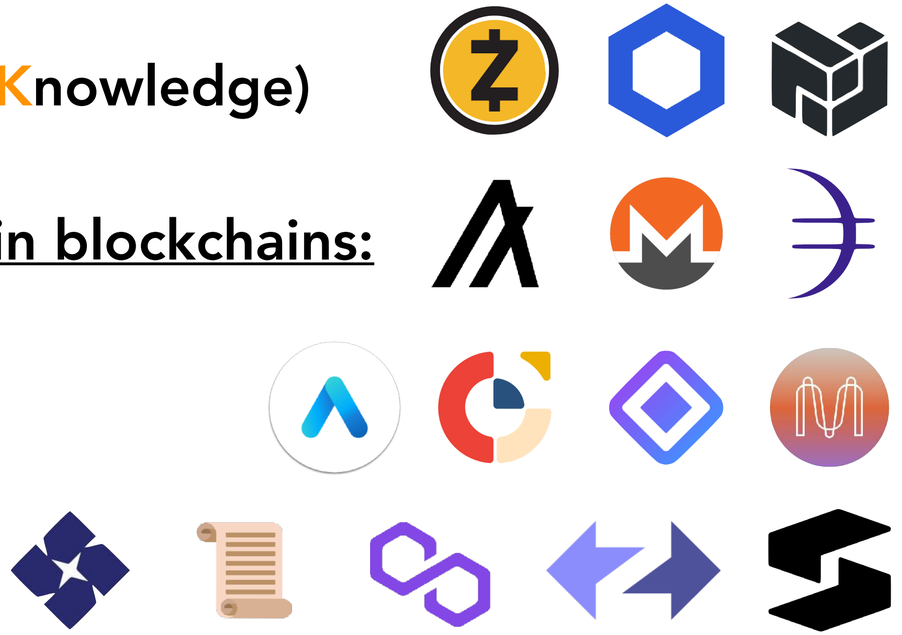
zkSNARKs: Security & Use Cases

(Zero-knowledge Succinct Non-interactive ARguments of Knowledge)

short, *non-interactive* *proofs*



Applications in blockchains:



Knowledge Soundness: If V accepts, then P must "know" w.

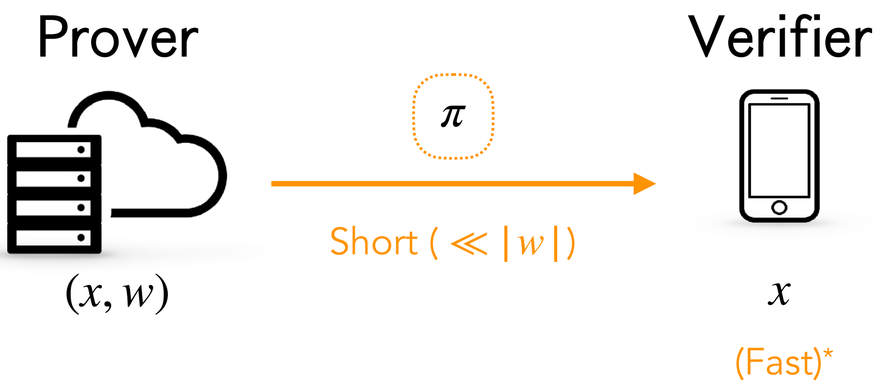
Zero-Knowledge: π hides w.

*For this talk, zkSNARKs may be without fast verification.

zkSNARKs: Security & Use Cases

(Zero-knowledge Succinct Non-interactive ARguments of Knowledge)

short, *non-interactive* *proofs*



Knowledge Soundness: If V accepts, then P must "know" w.

Zero-Knowledge: π hides w.

Applications in blockchains:

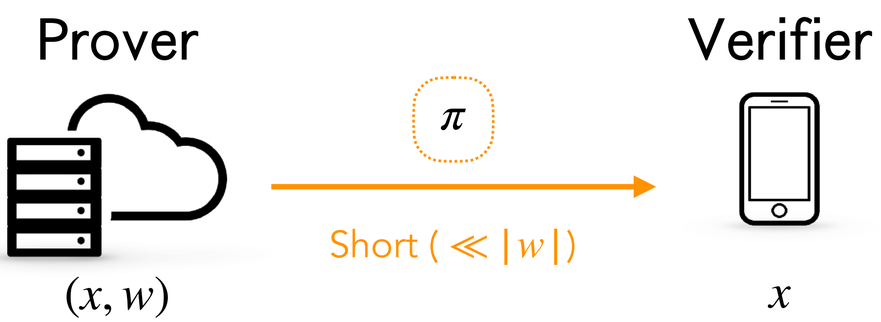
- Private smart contracts
- Private transactions
- ZK-Rollups

*For this talk, zkSNARKs may be without fast verification.

zkSNARKs: Security & Use Cases

(**Z**ero-**k**nowledge **S**uccinct **N**on-interactive **AR**guments of **K**nowledge)

short, *non-interactive* *proofs*



Knowledge Soundness: If V accepts, then P must "know" w.

Zero-Knowledge: π hides w.

Applications in blockchains:

- Private smart contracts
- Private transactions
- ZK-Rollups

Other applications:

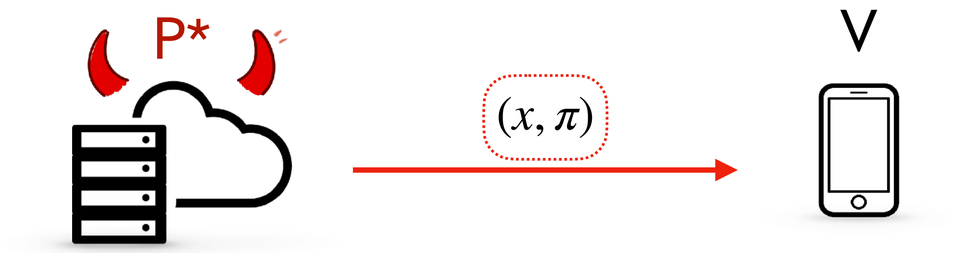
- Proof of solvency [DBBCB15]
- Image provenance [NT16], [BD22], [KHSS22]
- Content moderation [RMM22], [GAZBW22]
- And many more!

*For this talk, zkSNARKs may be without fast verification.

Standard ZKP security is not enough

Standard ZKP security is not enough

Adaptive attack: choose the statement adaptively based on the proof

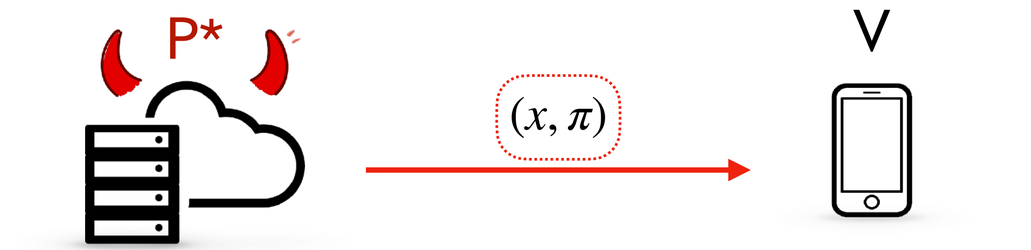


Compute π and x simultaneously

Accept π on x

Standard ZKP security is not enough

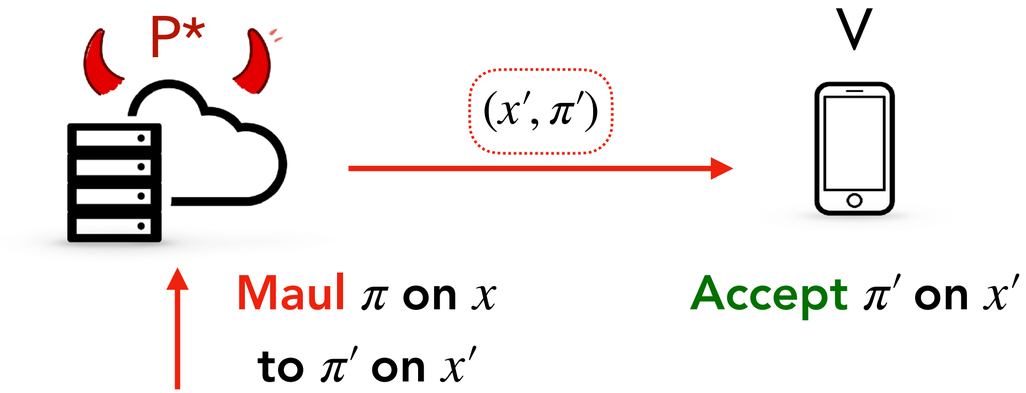
Adaptive attack: choose the statement adaptively based on the proof



Compute π and x simultaneously

Accept π on x

Malleability attack: modify an existing proof into a new proof without knowing the witness



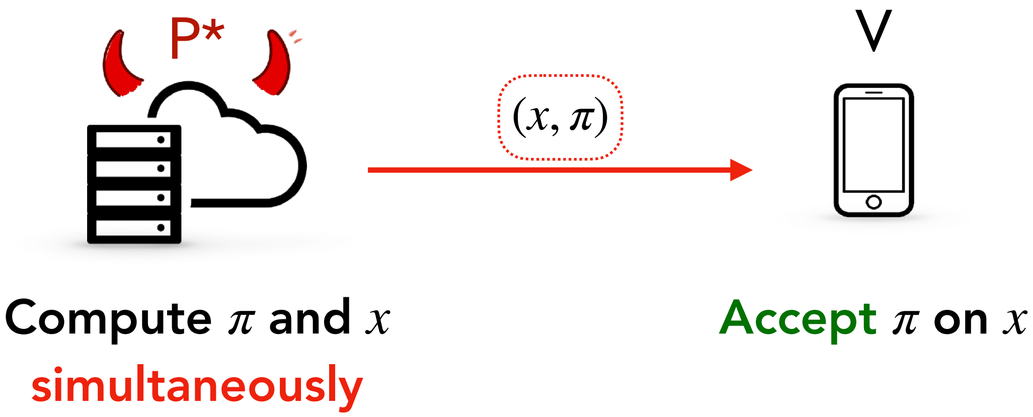
Maul π on x to π' on x'

Accept π' on x'

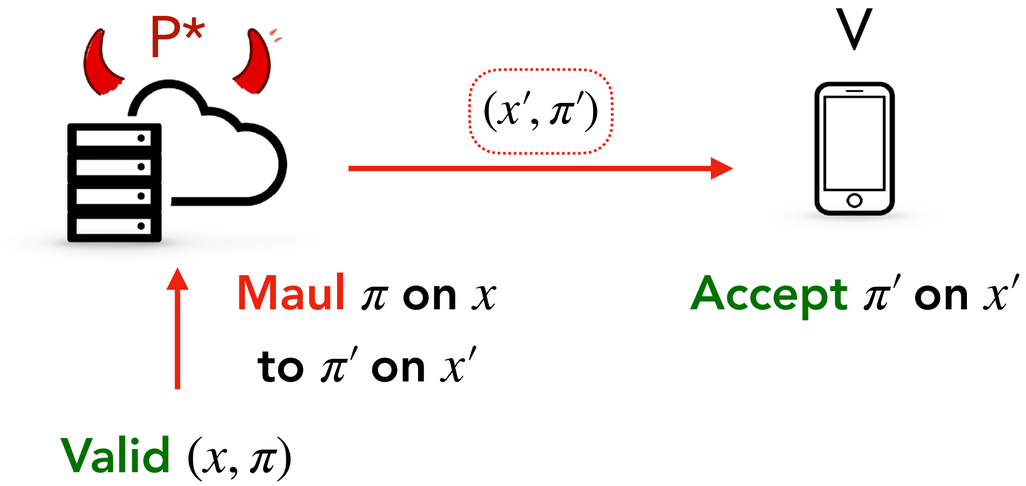
Valid (x, π)

Standard ZKP security is not enough

Adaptive attack: choose the statement adaptively based on the proof



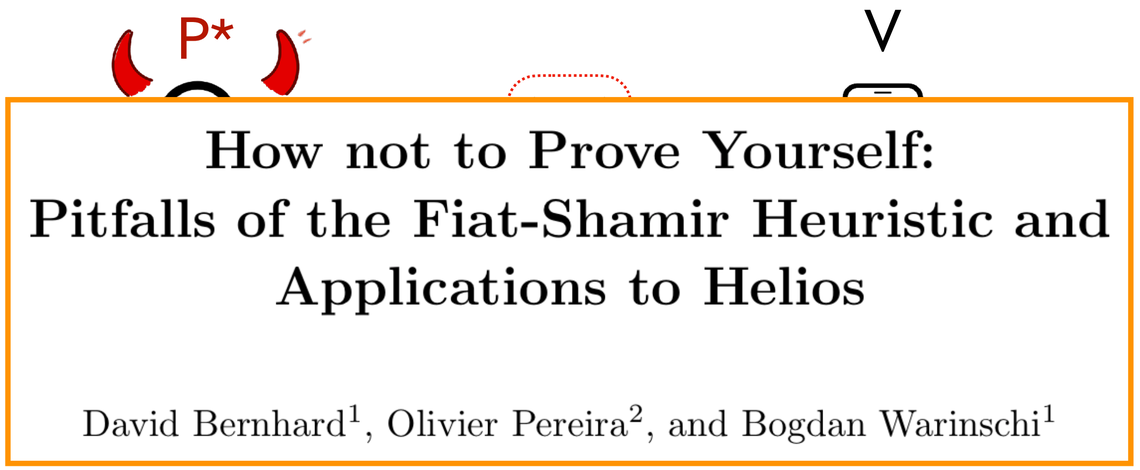
Malleability attack: modify an existing proof into a new proof without knowing the witness



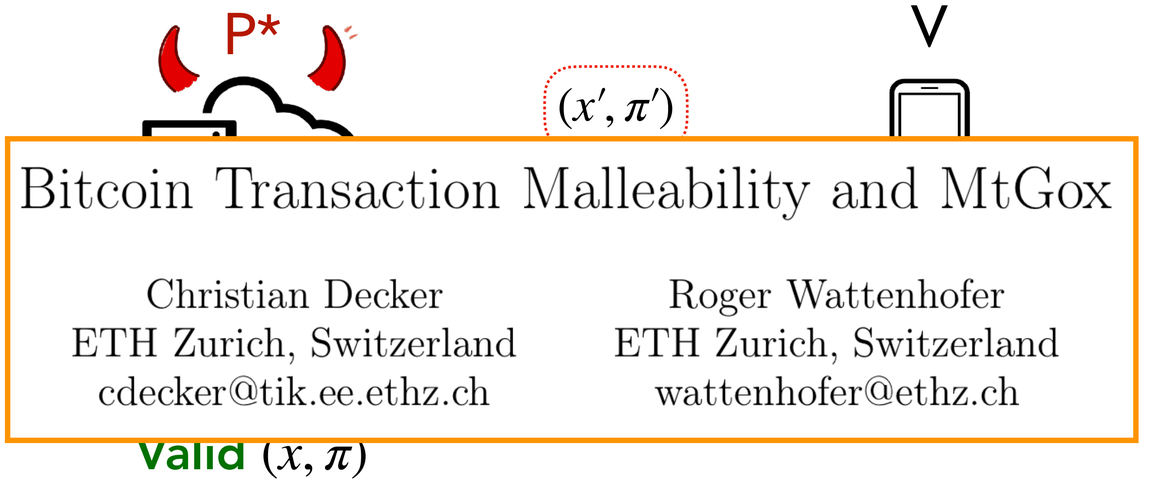
Not ruled out by (non-adaptive) knowledge soundness & zero-knowledge!

Standard ZKP security is not enough

Adaptive attack: choose the statement adaptively based on the proof



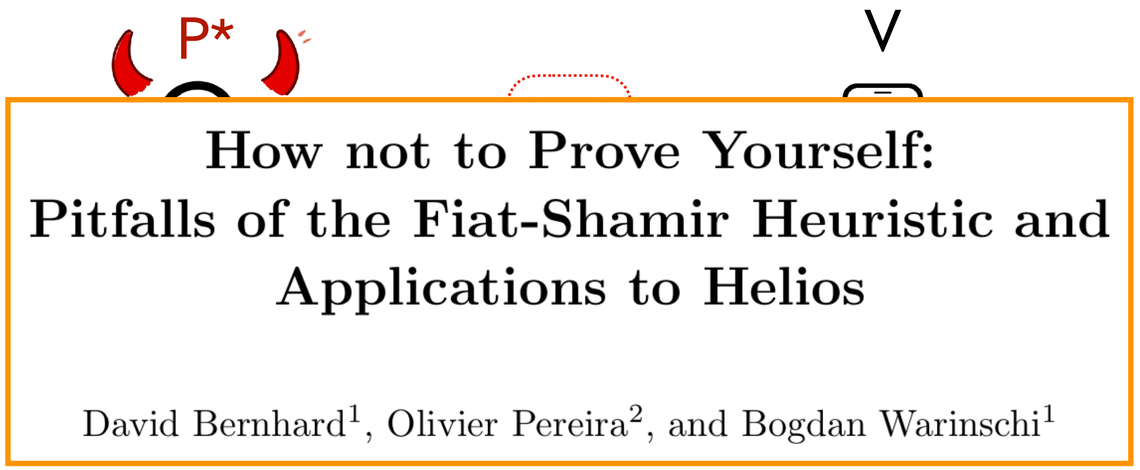
Malleability attack: modify an existing proof into a new proof without knowing the witness



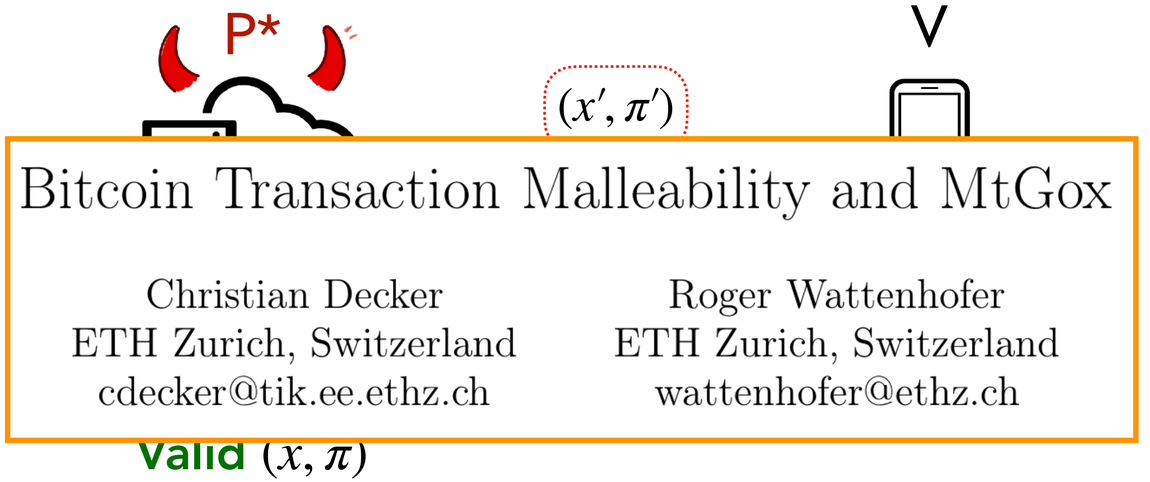
Not ruled out by (non-adaptive) knowledge soundness & zero-knowledge!

Standard ZKP security is not enough

Adaptive attack: choose the statement adaptively based on the proof



Malleability attack: modify an existing proof into a new proof without knowing the witness



Not ruled out by (non-adaptive) knowledge soundness & zero-knowledge!

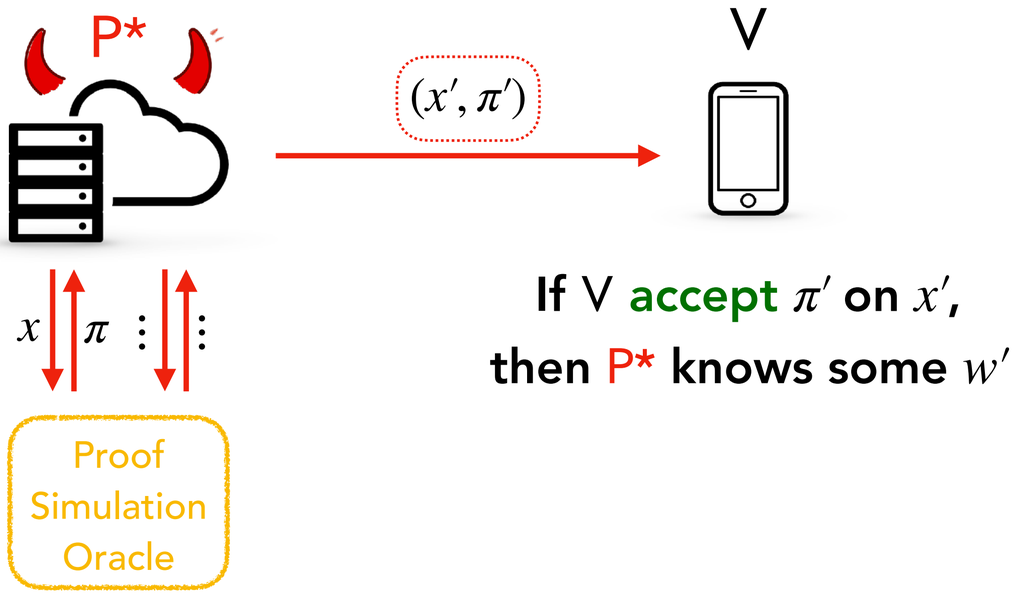
$\Longrightarrow$ We need *stronger* security properties for deployment

Simulation Extractability

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



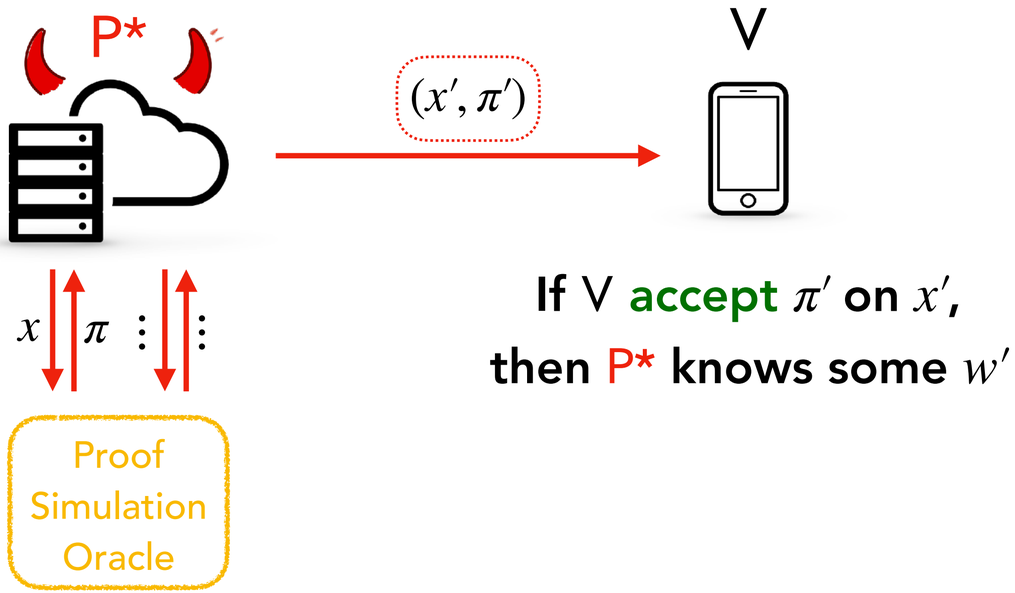
If V accept π' on x',

then P* knows some w'

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



If V accept π' on x', then P* knows some w'

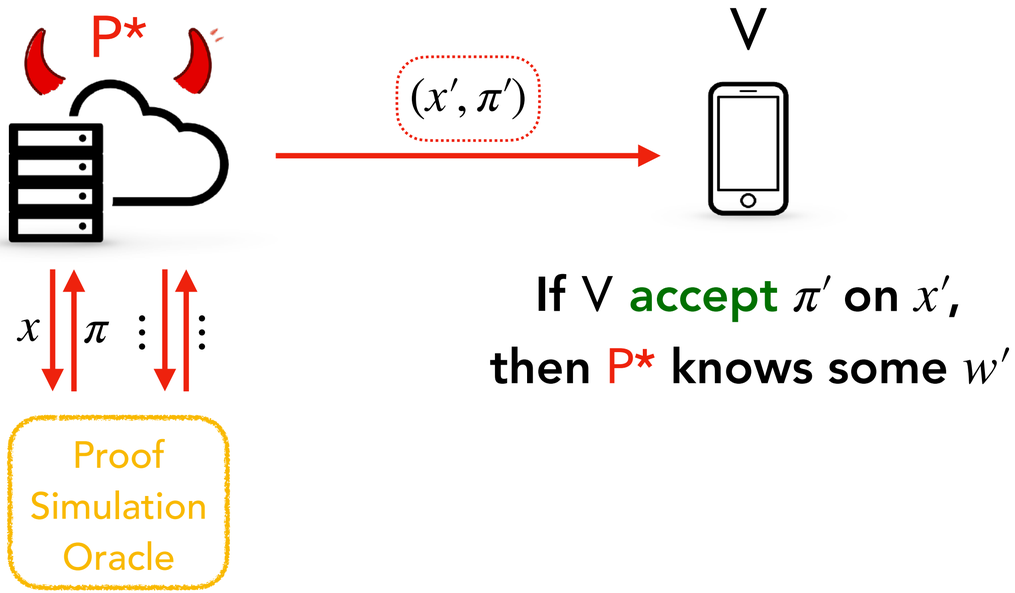
Rules out adaptive & malleability attacks.

Required for many applications. [KMSWP16], [BCG+20]

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



If V accept π' on x',
then P* knows some w'

Prior works:

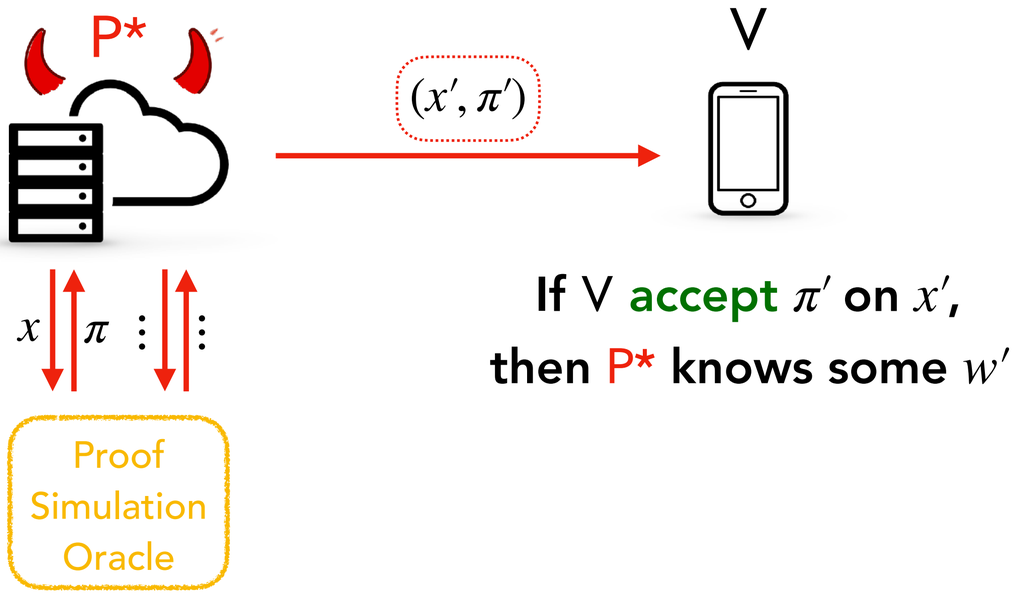
Rules out adaptive & malleability attacks.

Required for many applications. [KMSWP16], [BCG+20]

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



If V accept π' on x',
then P* knows some w'

Rules out adaptive & malleability attacks.

Required for many applications. [KMSWP16], [BCG+20]

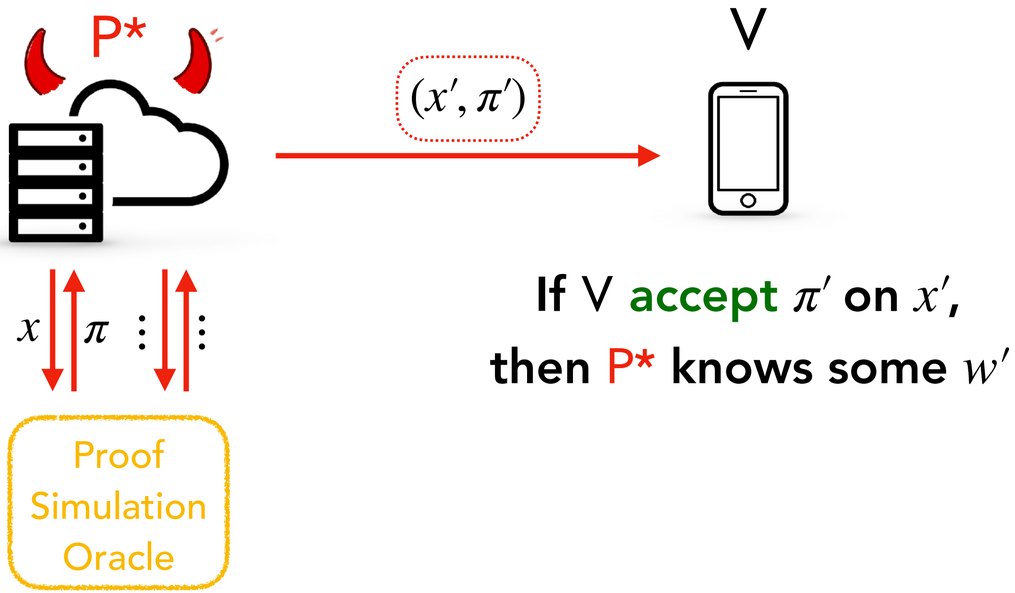
Prior works:

- Constructing SIM-EXT zkSNARKs directly.
  [GM17], [Lipmaa20]
- Achieving SIM-EXT via generic transformations.
  [KZMQCP15], [ARS20], [BS21], [BKSV21]

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



If V accept π' on x', then P* knows some w'

Rules out adaptive & malleability attacks.

Required for many applications. [KMSWP16], [BCG+20]

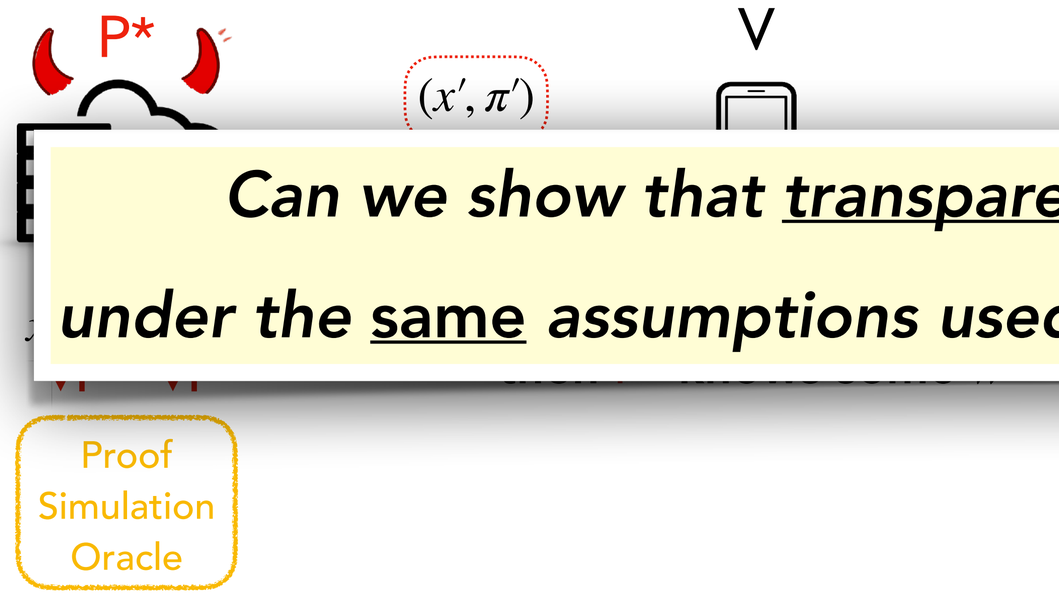
Prior works:

- Constructing SIM-EXT zkSNARKs directly. [GM17], [Lipmaa20]
- Achieving SIM-EXT via generic transformations. [KZMQCP15], [ARS20], [BS21], [BKSV21]
- Proving certain zkSNARKs are SIM-EXT out-of-the-box.
  - Sonic, Plonk, Marlin [GKKNZ22] $\Longleftarrow$ *not transparent*
  - Bulletproofs [GOPTT22] $\Longleftarrow$ *require stronger-than-necessary assumption (AGM)*

Simulation Extractability

SIM-EXT (informal): [Sahai99], [DDOPS01]

Knowledge soundness holds *even when* P* gets *extra power*.



Rules out adaptive & malleability attacks.

Required for many applications. [KMSWP16], [BCG+20]

Can we show that transparent zkSNARKs satisfy SIM-EXT under the same assumptions used to prove (knowledge) soundness?

Prior works:

- Constructing SIM-EXT zkSNARKs directly. [GM17], [Lipmaa20]
- Achieving SIM-EXT via generic transformations.

ut-of-

- Sonic, Plonk, Marlin [GKKNZ22] ⟸ *not transparent*
- Bulletproofs [GOPTT22] ⟸ *require stronger-than-necessary assumption (AGM)*

Our Results

Our Results

We show that **Spartan** and **Bulletproofs**, two *transparent* zkSNARKs, satisfy SIM-EXT in the random oracle model (ROM) assuming the discrete log assumption (DLOG) holds.

- Bulletproofs [BBBPWM18] has seen deployment in Monero, MimbleWimble, etc.
- Spartan [Setty20] is a state-of-the-art zkSNARK for prover time.

Our Results

We show that **Spartan** and **Bulletproofs**, two *transparent* zkSNARKs, satisfy SIM-EXT in the random oracle model (ROM) assuming the discrete log assumption (DLOG) holds.

- Bulletproofs [BBBPWM18] has seen deployment in Monero, MimbleWimble, etc.
- Spartan [Setty20] is a state-of-the-art zkSNARK for prover time.

These assumptions (DLOG + ROM) are the ***minimal*** ones used to prove their soundness.

Our Results

We show that **Spartan** and **Bulletproofs**, two *transparent* zkSNARKs, satisfy SIM-EXT in the random oracle model (ROM) assuming the discrete log assumption (DLOG) holds.

- Bulletproofs [BBBPWM18] has seen deployment in Monero, MimbleWimble, etc.
- Spartan [Setty20] is a state-of-the-art zkSNARK for prover time.

These assumptions (DLOG + ROM) are the ***minimal*** ones used to prove their soundness.

To prove our results, we develop a few tools that might be of independent interest:

Our Results

We show that **Spartan** and **Bulletproofs**, two *transparent* zkSNARKs, satisfy SIM-EXT in the random oracle model (ROM) assuming the discrete log assumption (DLOG) holds.

- Bulletproofs [BBBPWM18] has seen deployment in Monero, MimbleWimble, etc.
- Spartan [Setty20] is a state-of-the-art zkSNARK for prover time.

These assumptions (DLOG + ROM) are the ***minimal*** ones used to prove their soundness.

To prove our results, we develop a few tools that might be of independent interest:

- A **template** for proving SIM-EXT from smaller properties
  (building on the work of Ganesh, Khoshakhlagh, Kohlweiss, Nitulescu & Zajac [GKKNZ22])

Our Results

We show that **Spartan** and **Bulletproofs**, two *transparent* zkSNARKs, satisfy SIM-EXT in the random oracle model (ROM) assuming the discrete log assumption (DLOG) holds.

- Bulletproofs [BBBPWM18] has seen deployment in Monero, MimbleWimble, etc.
- Spartan [Setty20] is a state-of-the-art zkSNARK for prover time.

These assumptions (DLOG + ROM) are the ***minimal*** ones used to prove their soundness.

To prove our results, we develop a few tools that might be of independent interest:

- A **template** for proving SIM-EXT from smaller properties
  (building on the work of Ganesh, Khoshakhlagh, Kohlweiss, Nitulescu & Zajac [GKKNZ22])
- A more general **tree extraction** lemma for proving knowledge soundness
  (building on the work of Attema, Fehr & Klooß [AFK22])

Agenda

1. Breaking SIM-EXT into smaller properties
2. Instantiating SIM-EXT template for Bulletproofs
3. Knowledge Soundness via Generalized Tree Builder

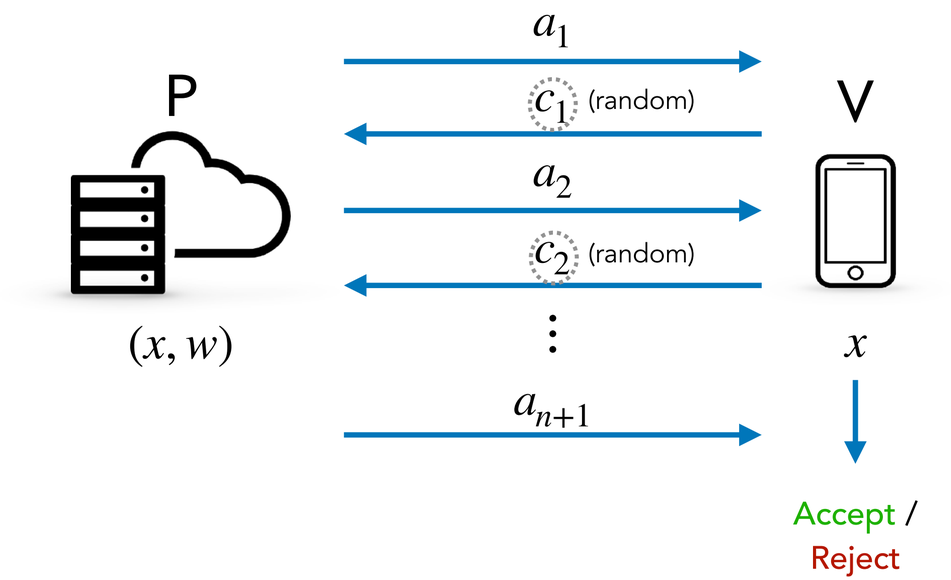
Agenda

1. Breaking SIM-EXT into smaller properties
2. Instantiating SIM-EXT template for Bulletproofs
3. Knowledge Soundness via Generalized Tree Builder

The Fiat-Shamir Transform & SIM-EXT Insight

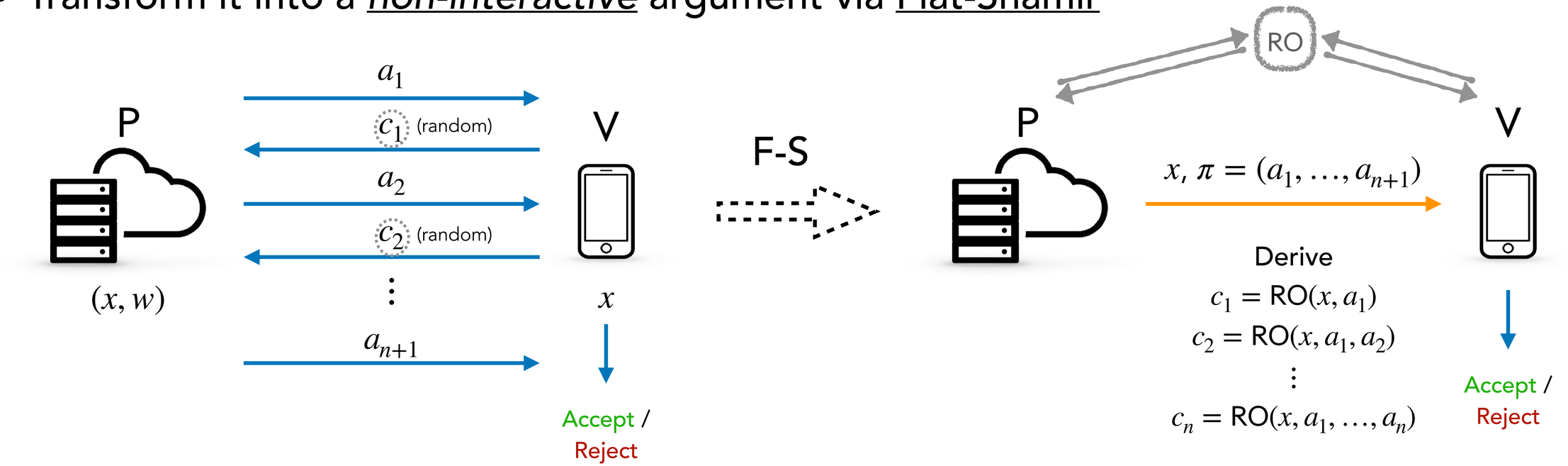
The Fiat-Shamir Transform & SIM-EXT Insight

- Construct an *interactive*, *public-coin* argument



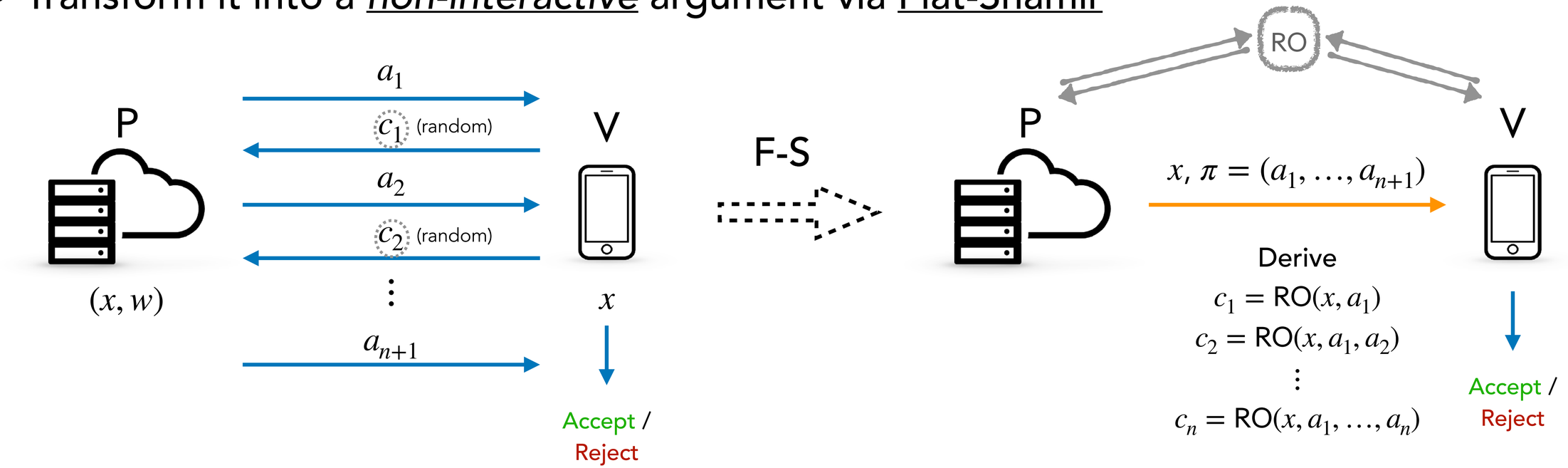
The Fiat-Shamir Transform & SIM-EXT Insight

- Construct an *interactive*, *public-coin* argument
- Transform it into a *non-interactive* argument via Fiat-Shamir



The Fiat-Shamir Transform & SIM-EXT Insight

- Construct an *interactive*, *public-coin* argument
- Transform it into a *non-interactive* argument via Fiat-Shamir



Insight: [GKKNZ22] ***Assuming 2 smaller properties, SIM-EXT of F-S argument may be reduced to its knowledge soundness (KS).***

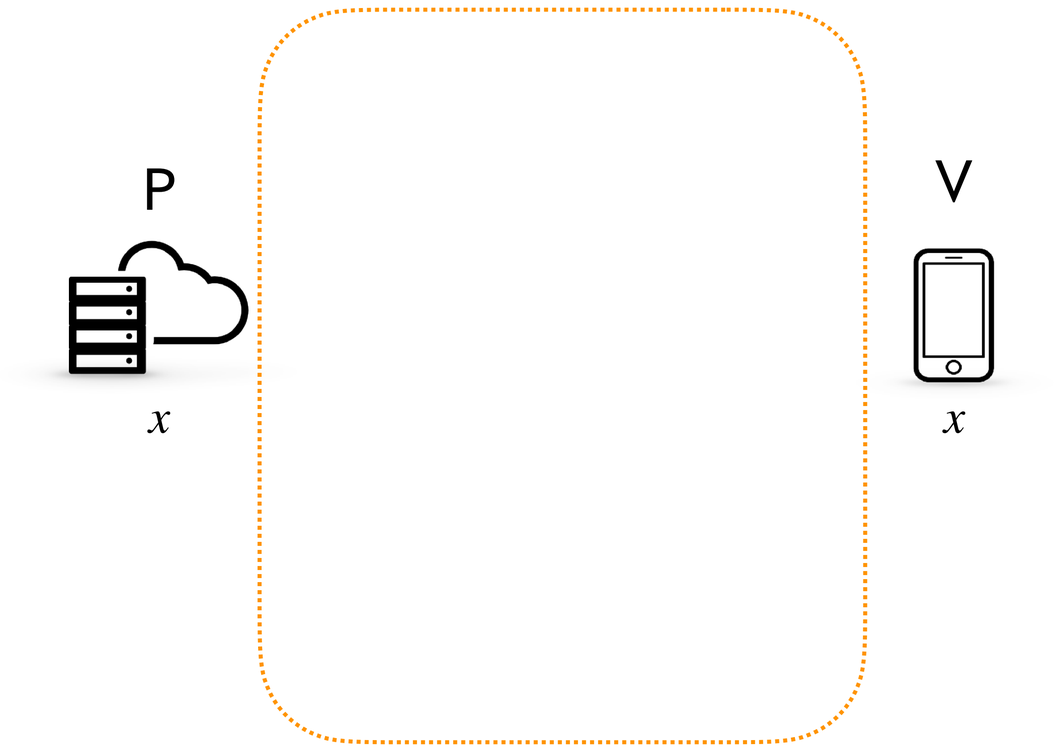
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge and k-Unique Response

Zero-Knowledge (ZK): The simulator Sim may choose all challenges *before* computing P's messages.

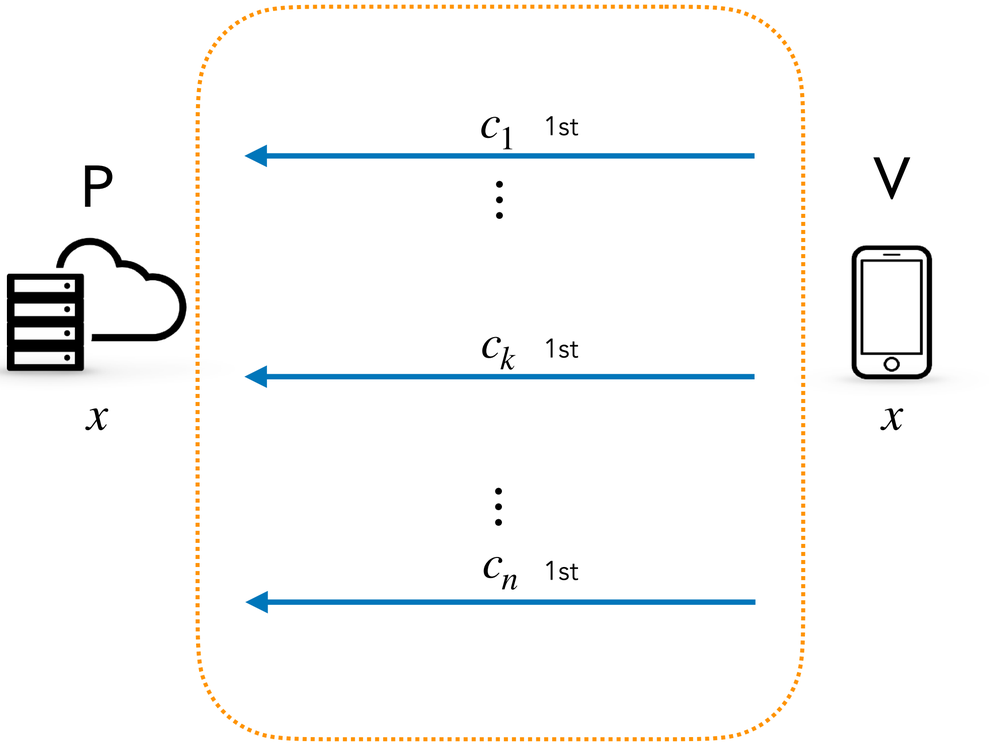
k-Zero-Knowledge and k-Unique Response

Zero-Knowledge (ZK): The simulator Sim may choose all challenges *before* computing P's messages.



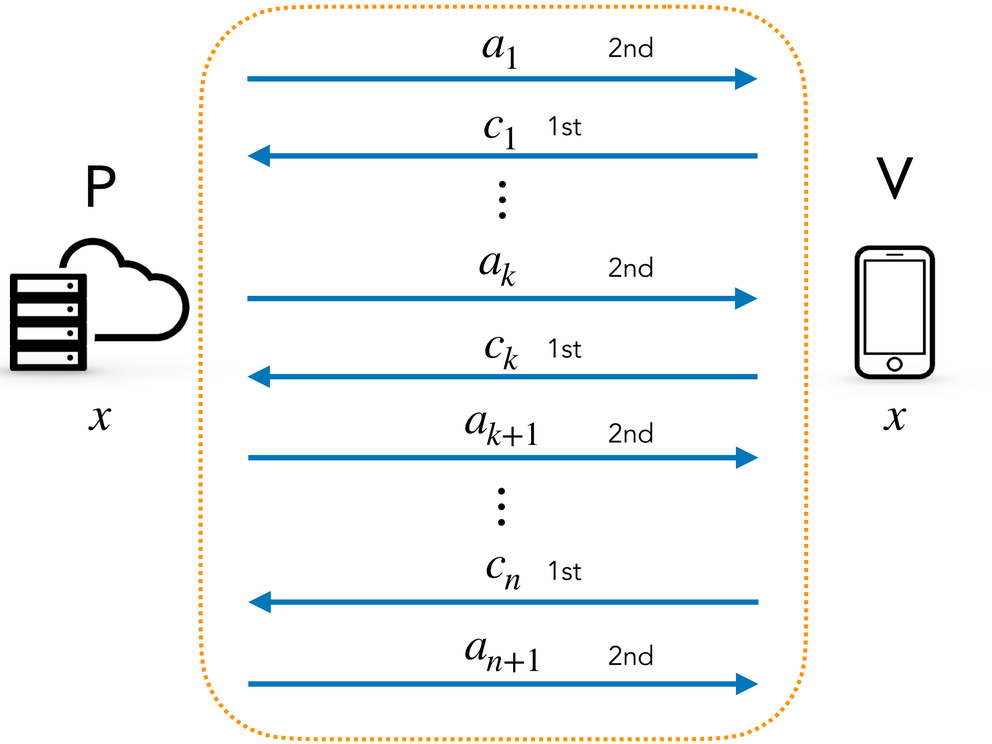
k-Zero-Knowledge and k-Unique Response

Zero-Knowledge (ZK): The simulator Sim may choose all challenges *before* computing P's messages.

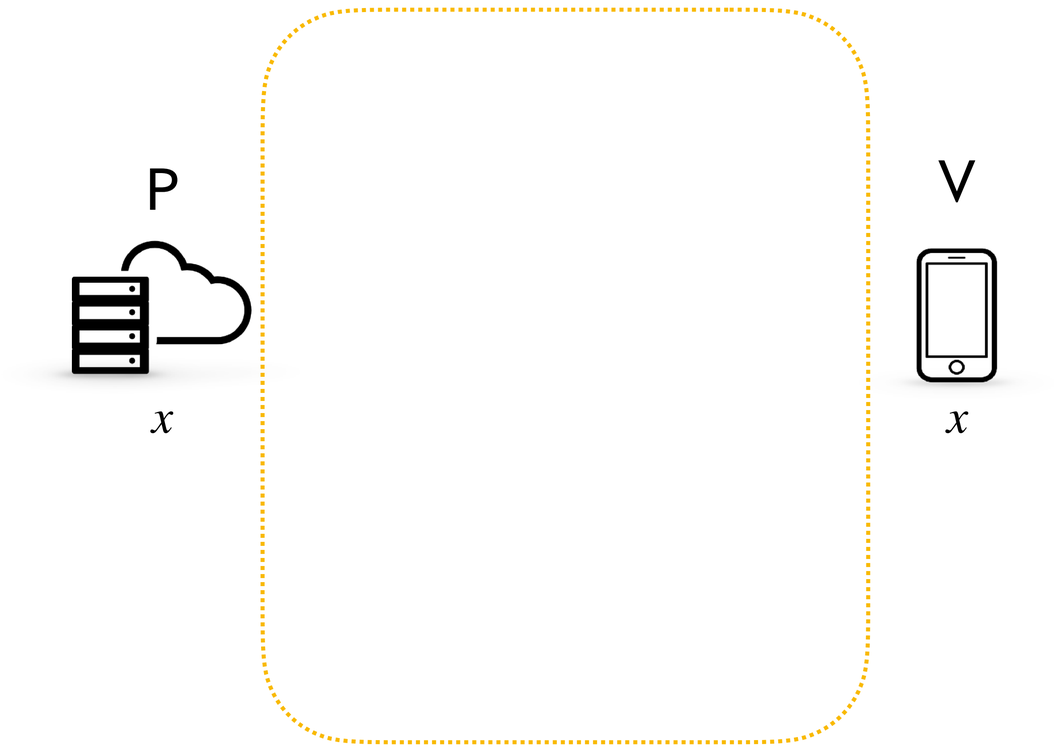


k-Zero-Knowledge and k-Unique Response

Zero-Knowledge (ZK): The simulator Sim may choose all challenges *before* computing P's messages.



k-Zero-Knowledge and k-Unique Response

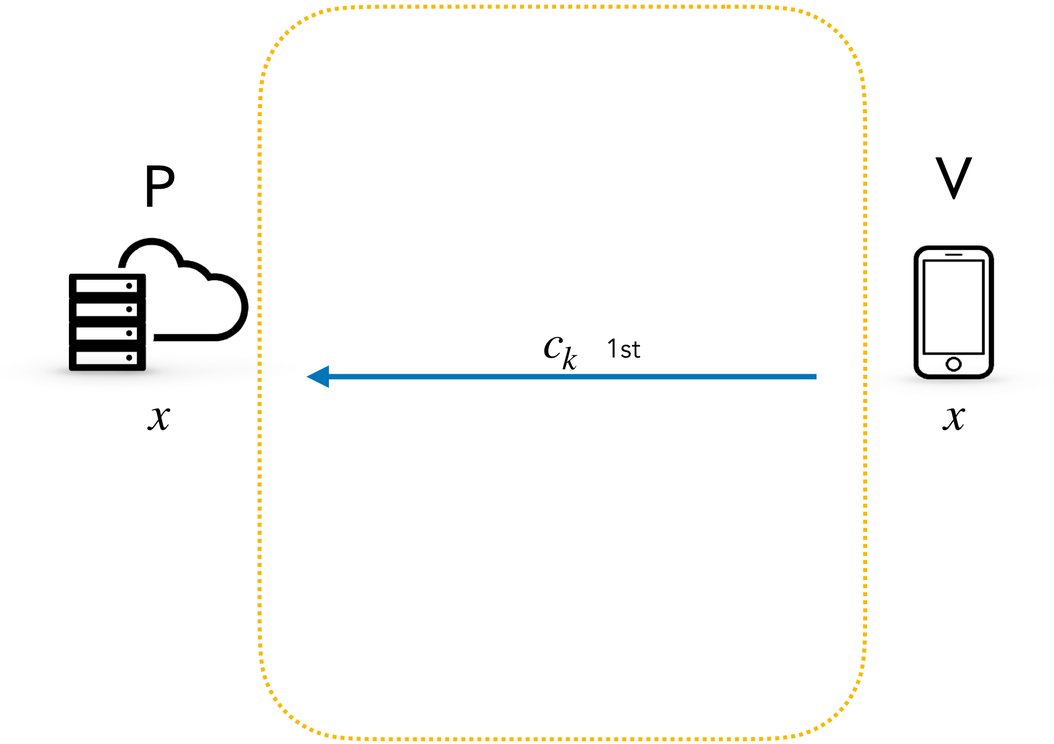


k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.

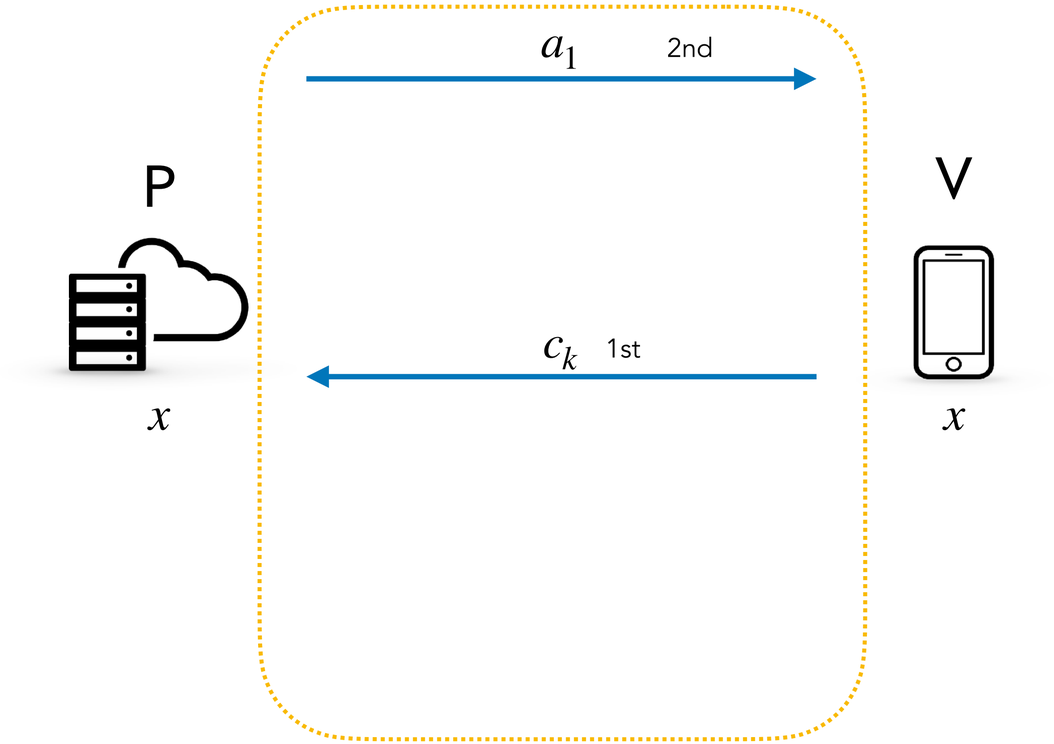
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



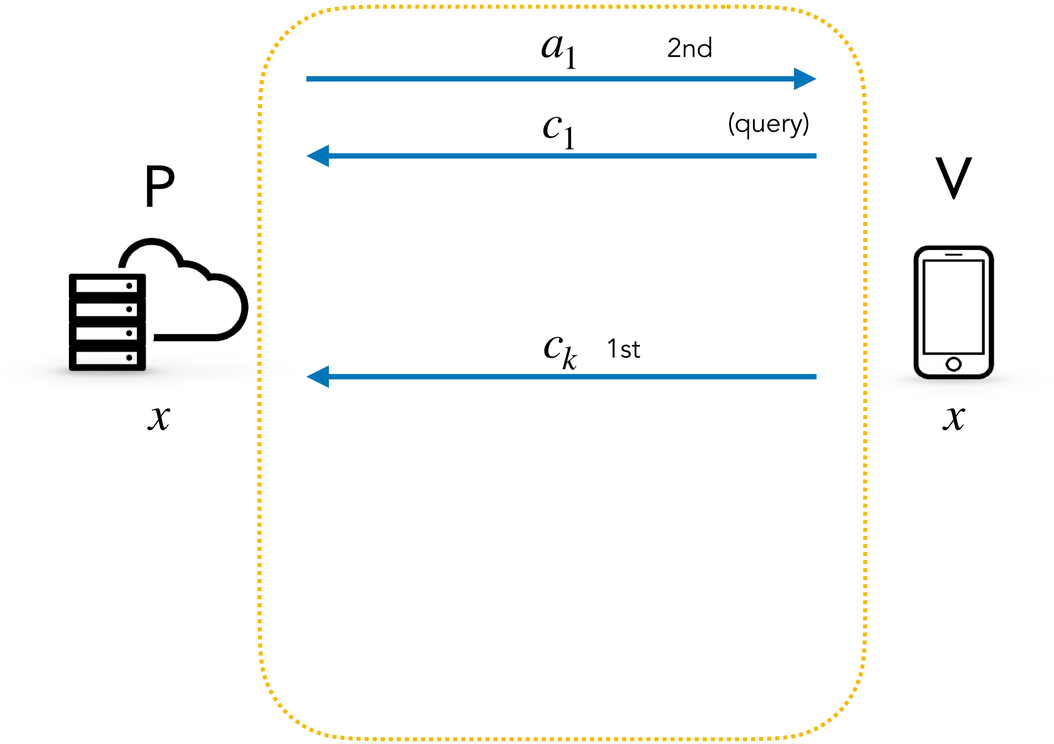
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



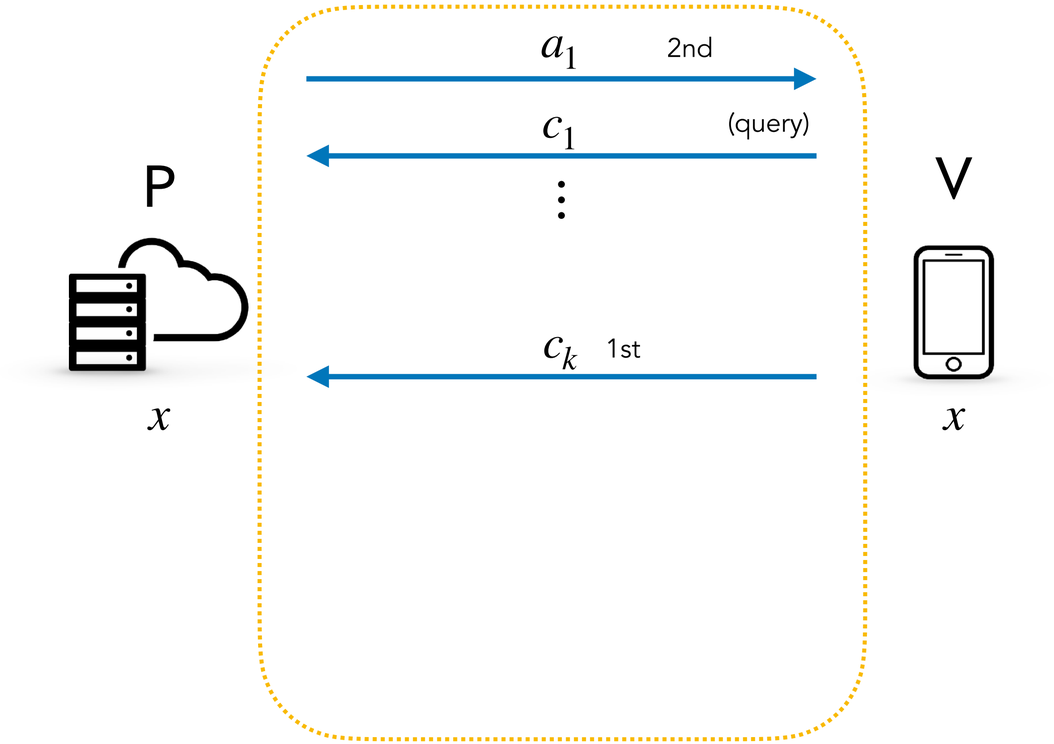
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



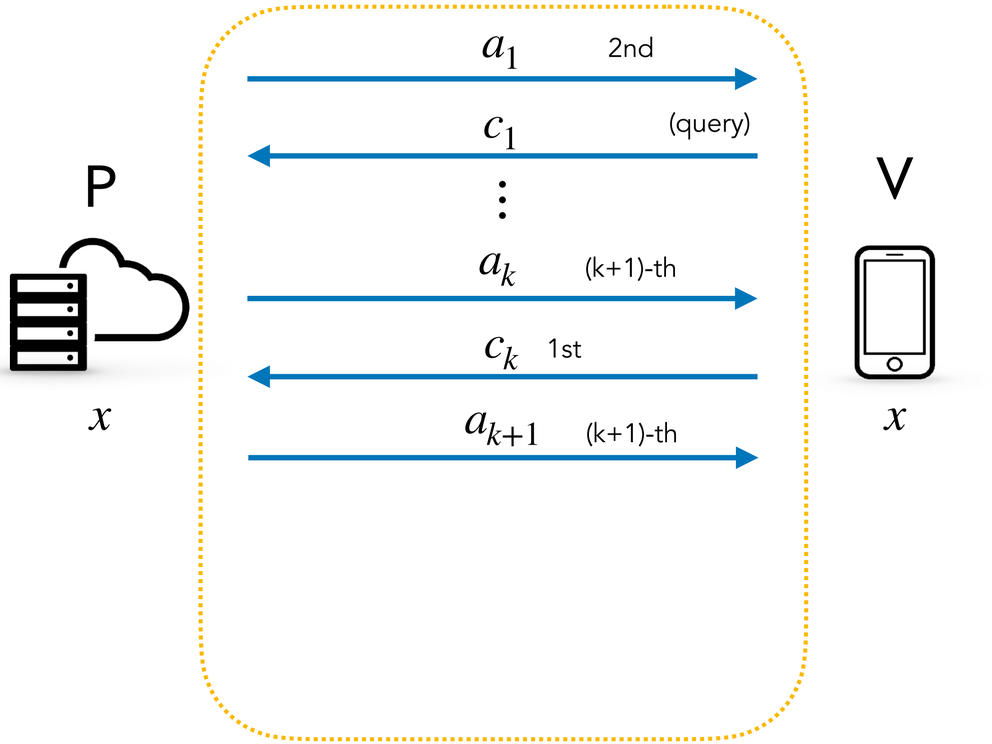
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



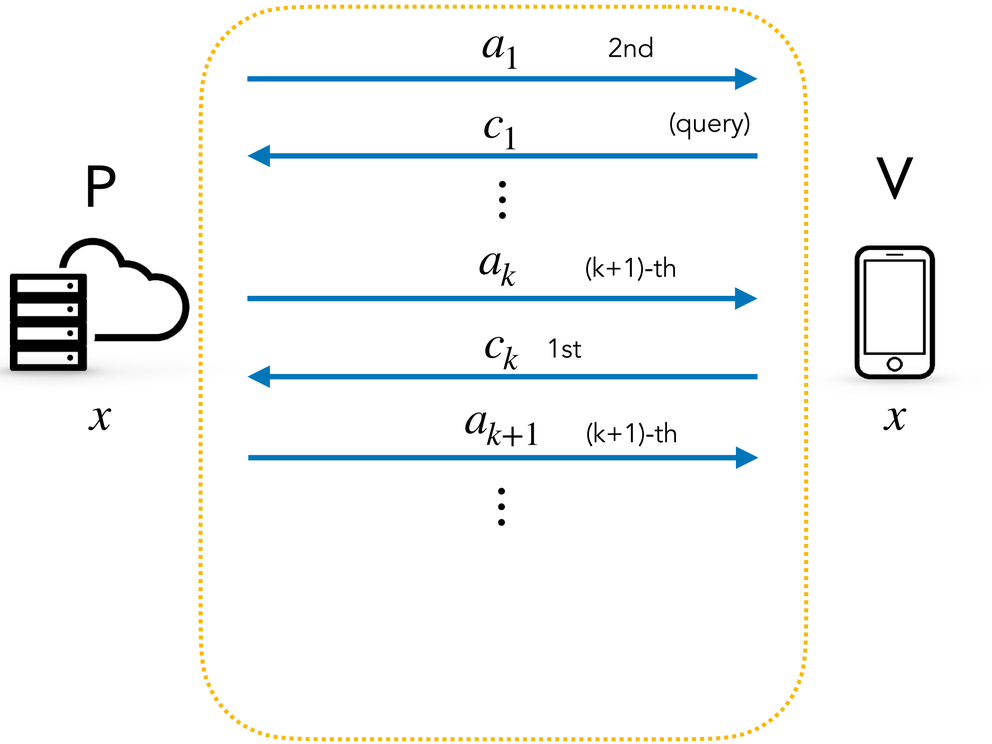
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



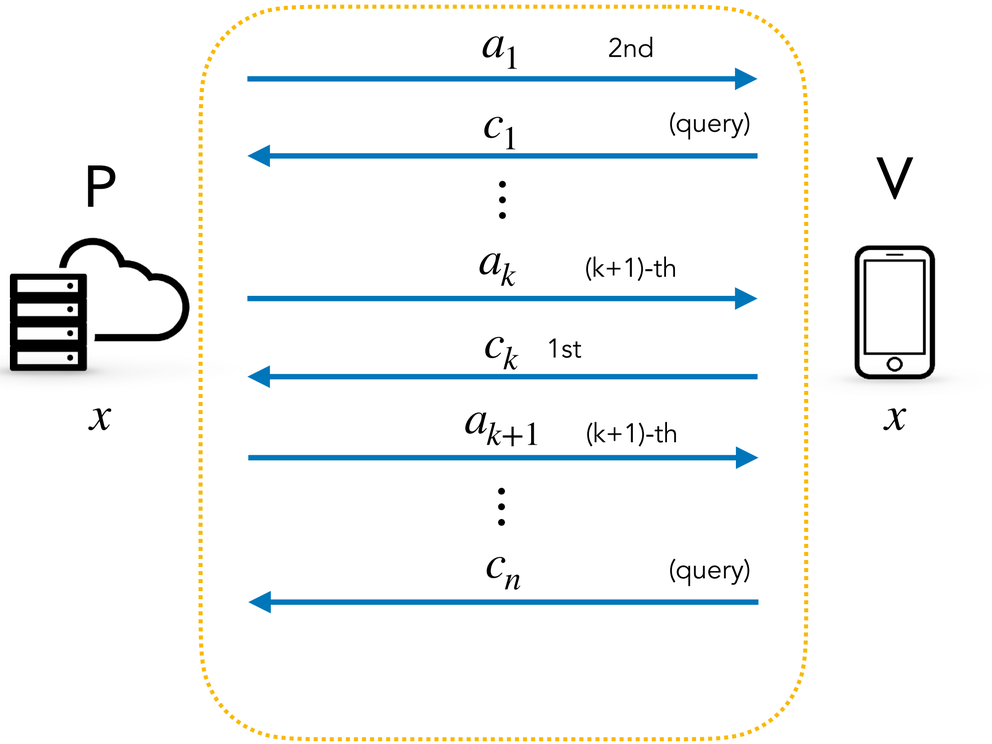
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



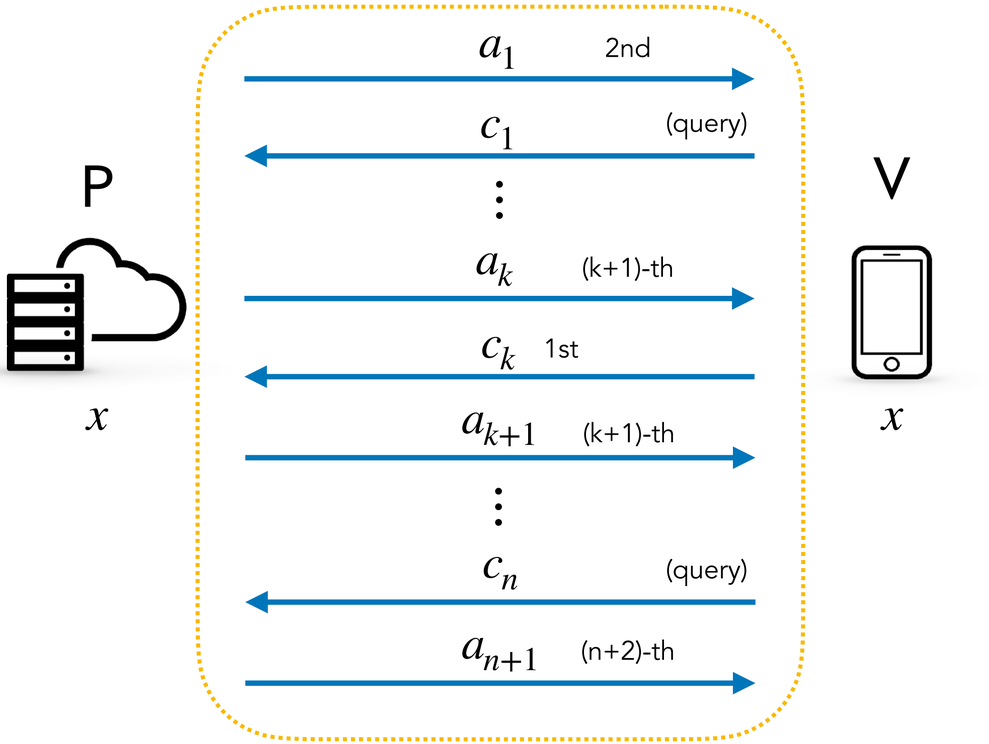
k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



k-Zero-Knowledge and k-Unique Response

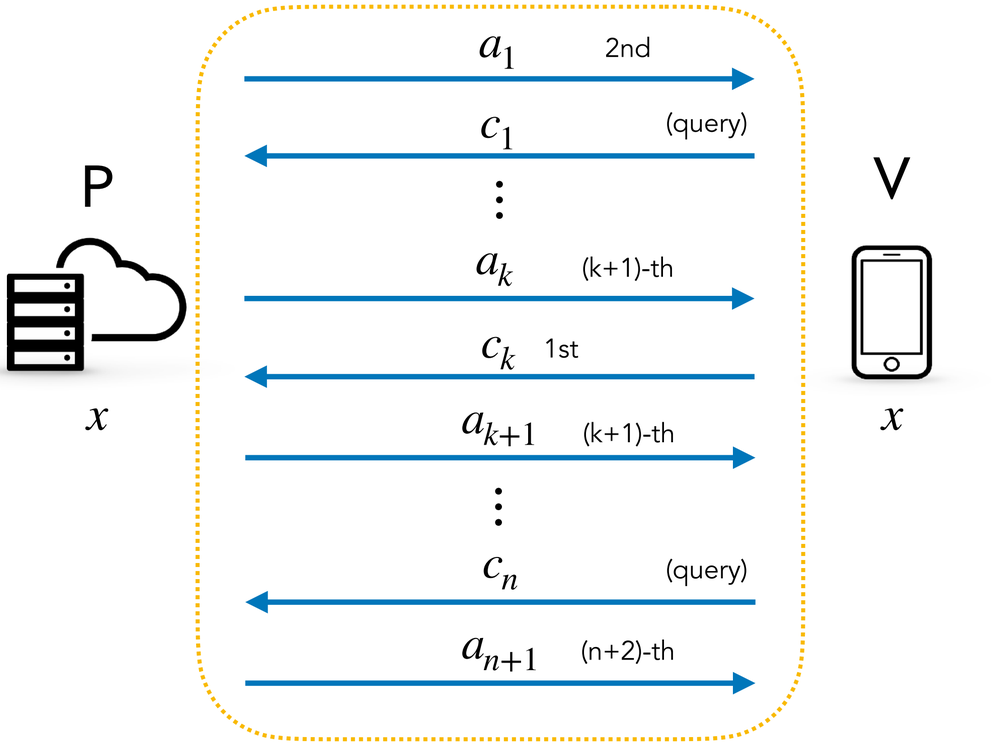
k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



k-Zero-Knowledge and k-Unique Response

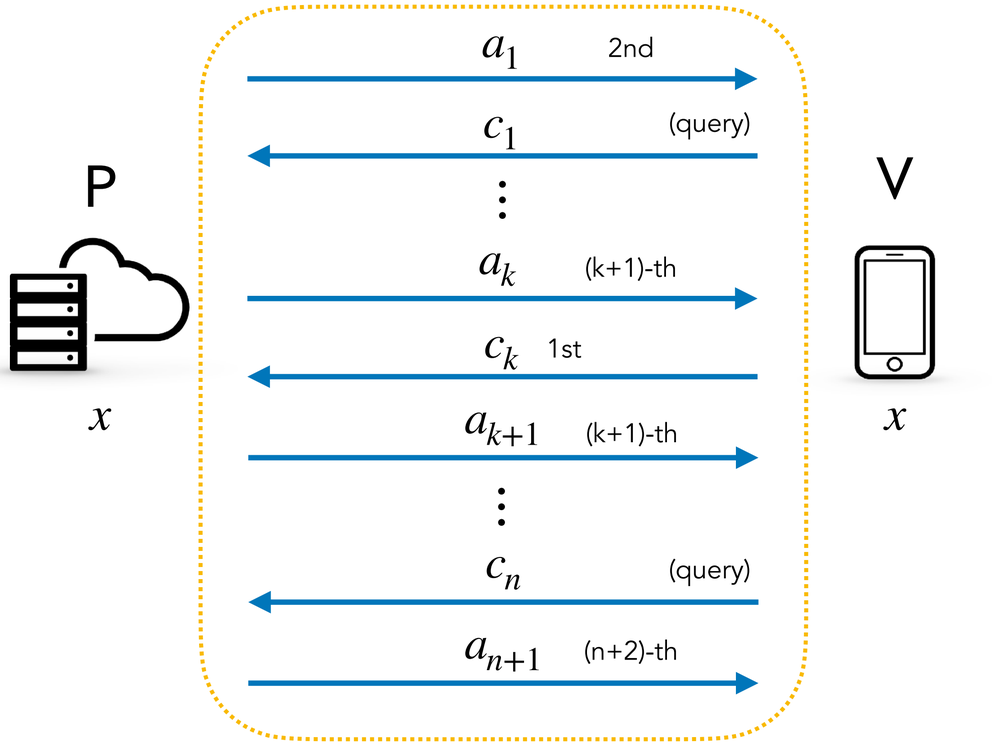
k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.

k-Unique Response (k-UR): P* cannot output accepting proofs $\pi \neq \pi'$ that agree up to round k, *even* given power to choose statement x and k^{th} challenge c_k.

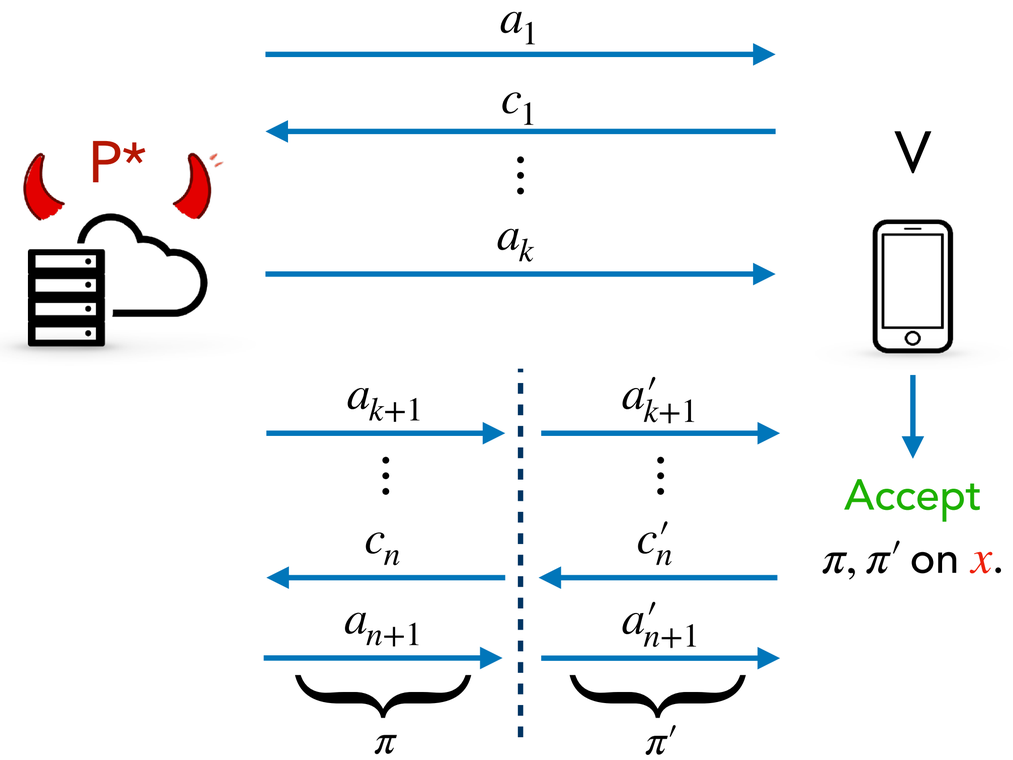


k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.

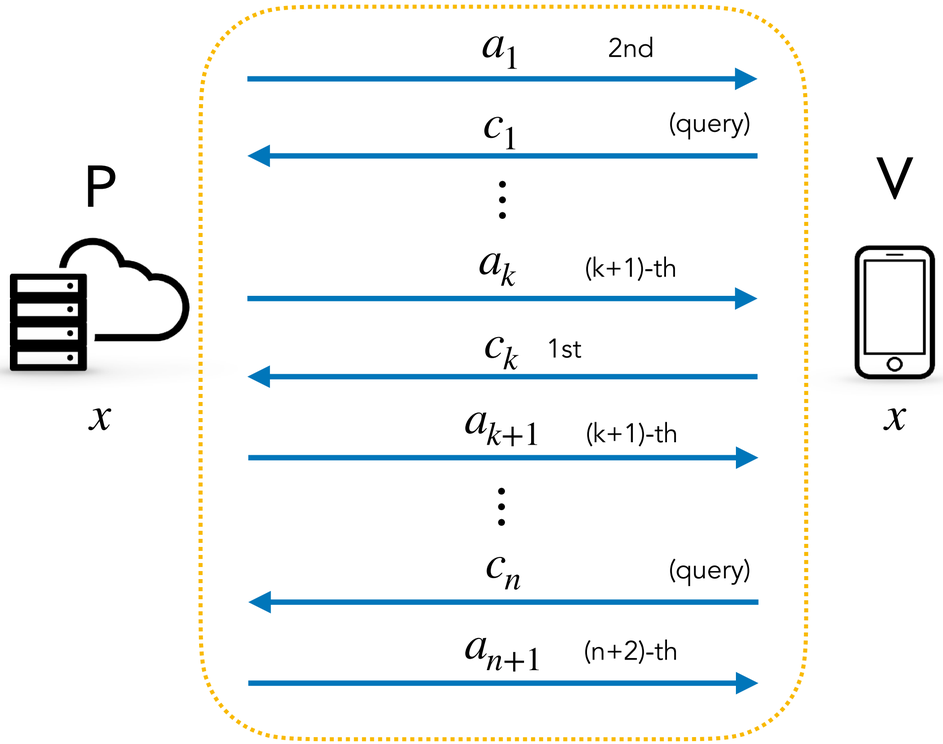


k-Unique Response (k-UR): P* cannot output accepting proofs $\pi \neq \pi'$ that agree up to round k, *even* given power to choose statement x and k^{th} challenge c_k.

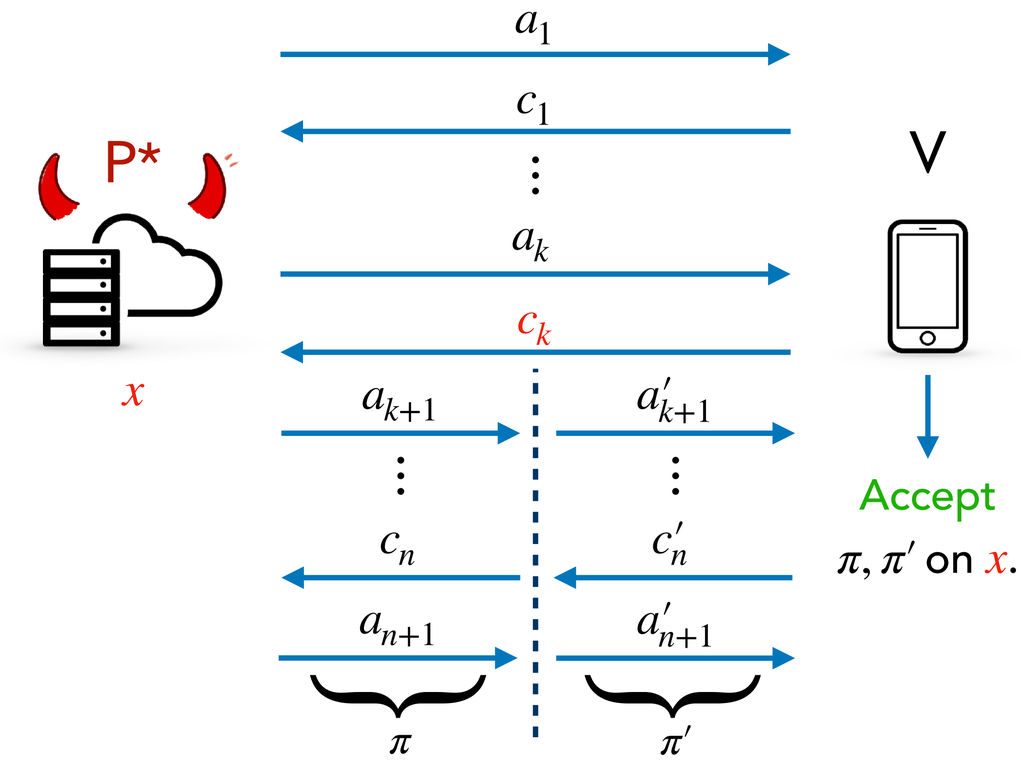


k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.



k-Unique Response (k-UR): P* cannot output accepting proofs $\pi \neq \pi'$ that agree up to round k, *even* given power to choose statement x and k^{th} challenge c_k.

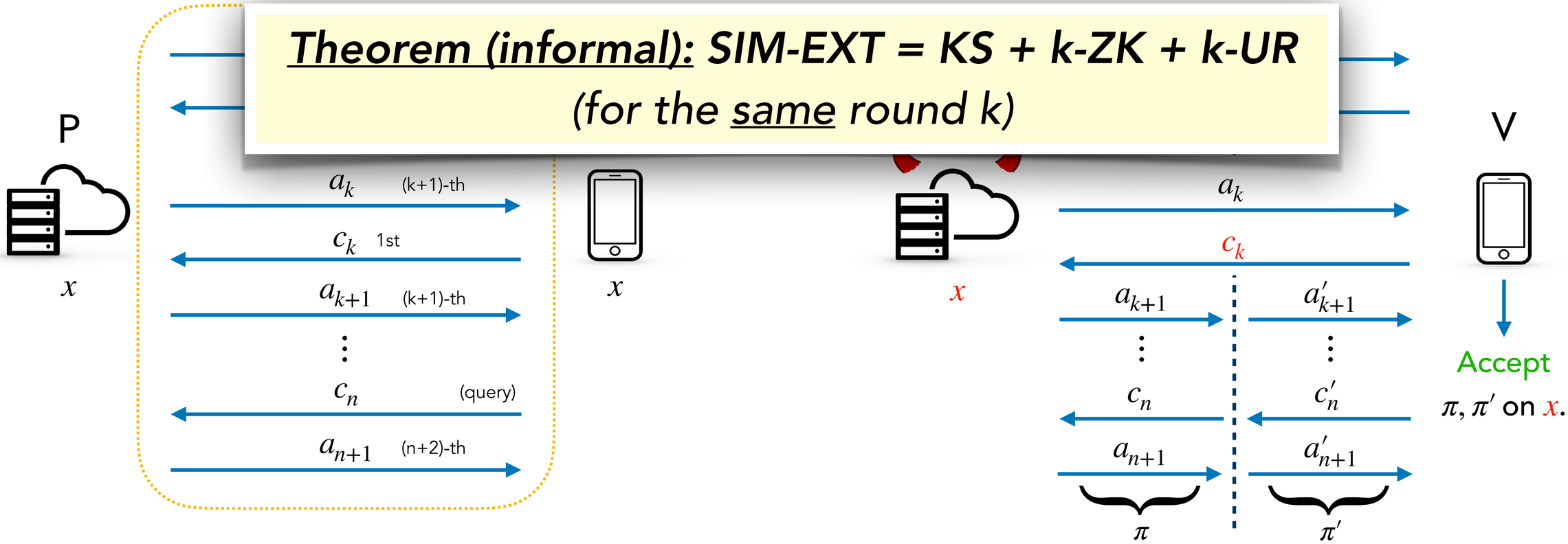


k-Zero-Knowledge and k-Unique Response

k-Zero-Knowledge (k-ZK): The simulator Sim_k may *only* choose k^{th} challenge, and compute other messages in order.

k-Unique Response (k-UR): P* cannot output accepting proofs $\pi \neq \pi'$ that agree up to round k, *even* given power to choose statement x and k^{th} challenge c_k.

Theorem (informal): SIM-EXT = KS + k-ZK + k-UR
(for the same round k)



Agenda

1. SIM-EXT = KS + k-ZK + k-UR (for same k)
2. **Instantiating SIM-EXT template for Bulletproofs**
3. Knowledge Soundness via Generalized Tree Builder

Bulletproofs Range Proof

Bulletproofs Range Proof

Public Private

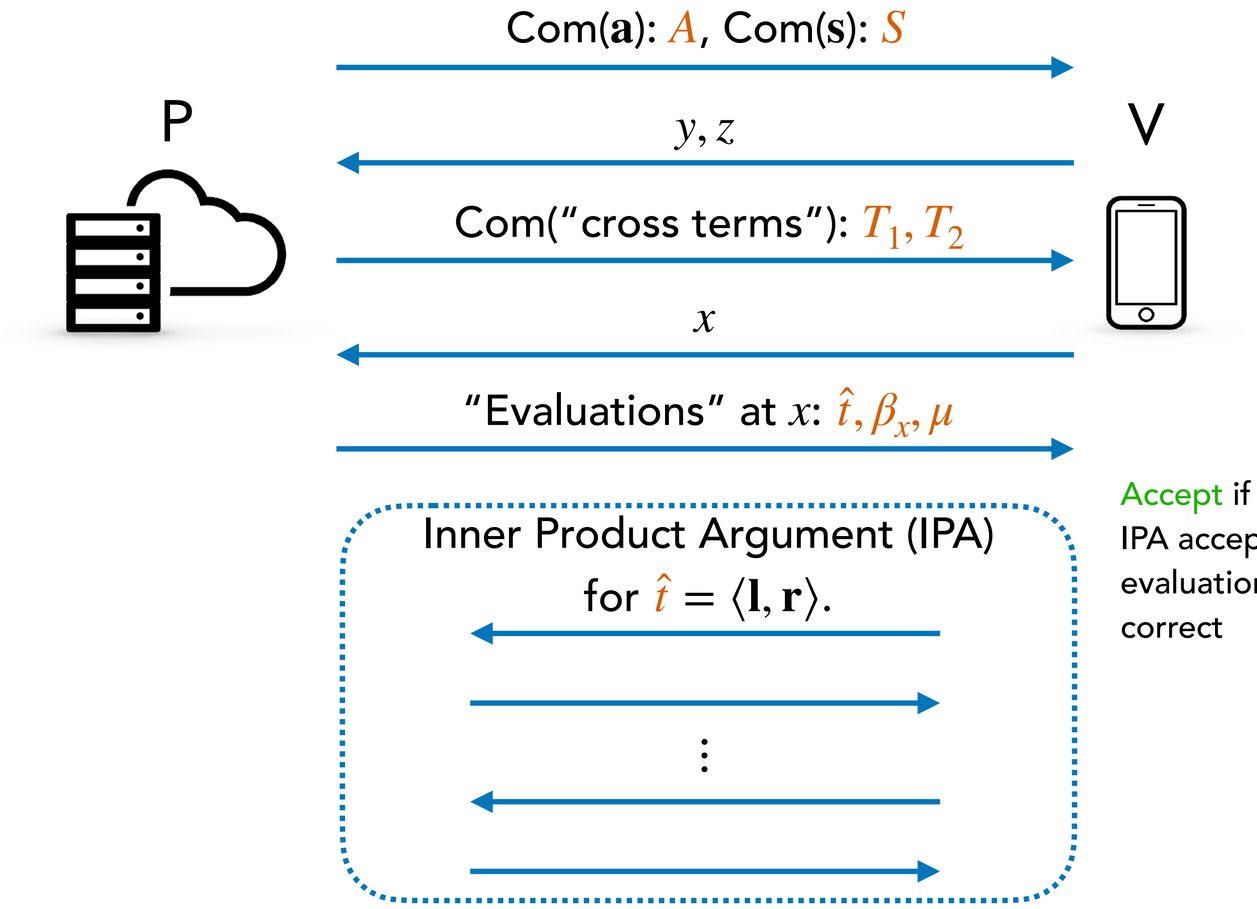
Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$



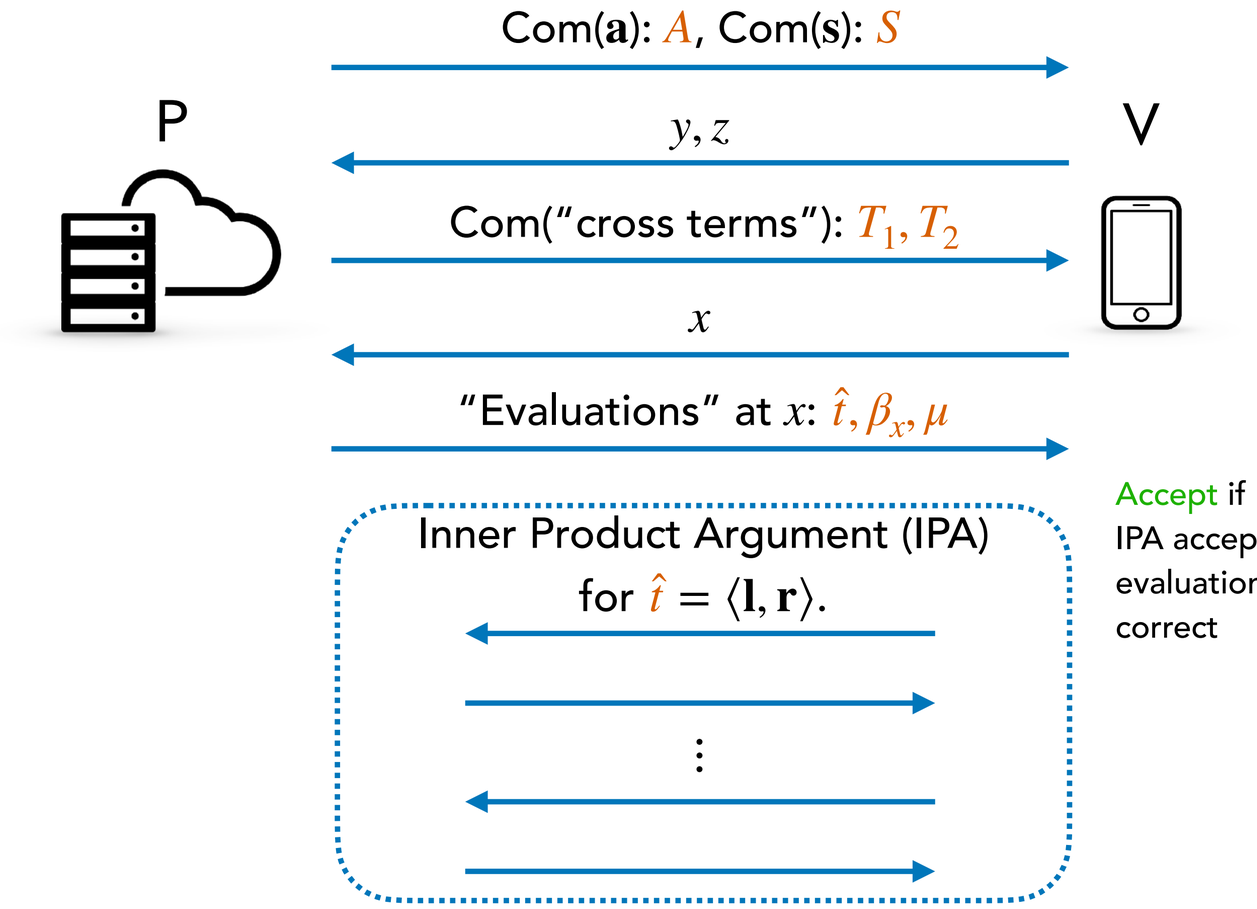
Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

Recall: We need to show Bulletproofs satisfy KS, k-ZK, and k-UR for the same round k.



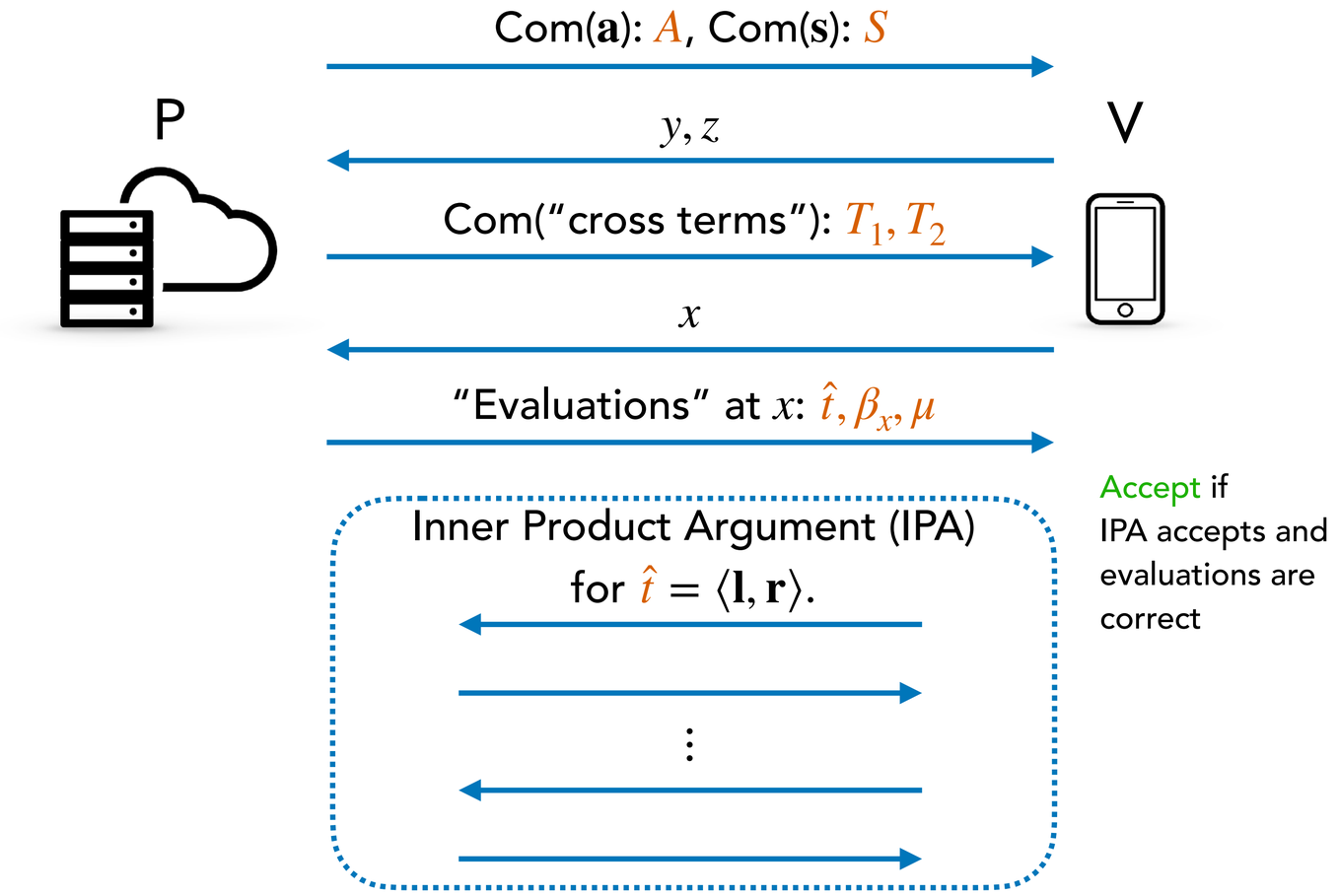
Bulletproofs Range Proof

Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

Recall: We need to show Bulletproofs satisfy KS, k-***ZK***, and k-***UR*** for the same round k.

Let $\mathbf{a} = \text{BinDecomp}(v)$.



Bulletproofs Range Proof

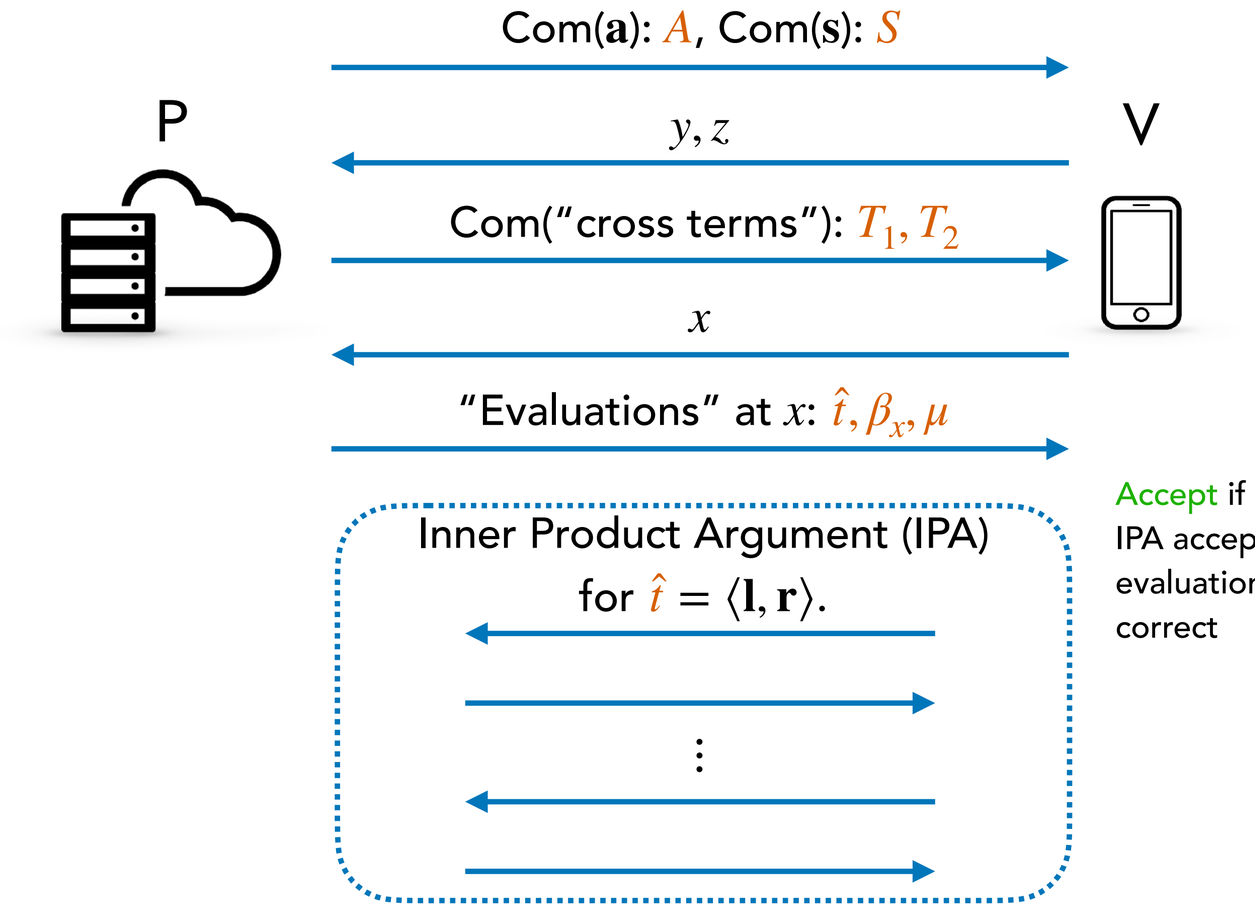
Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

Recall: We need to show Bulletproofs satisfy KS, k-***ZK***, and k-***UR*** for the same round k.

Q: Which round k to prove k-ZK and k-UR?



Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

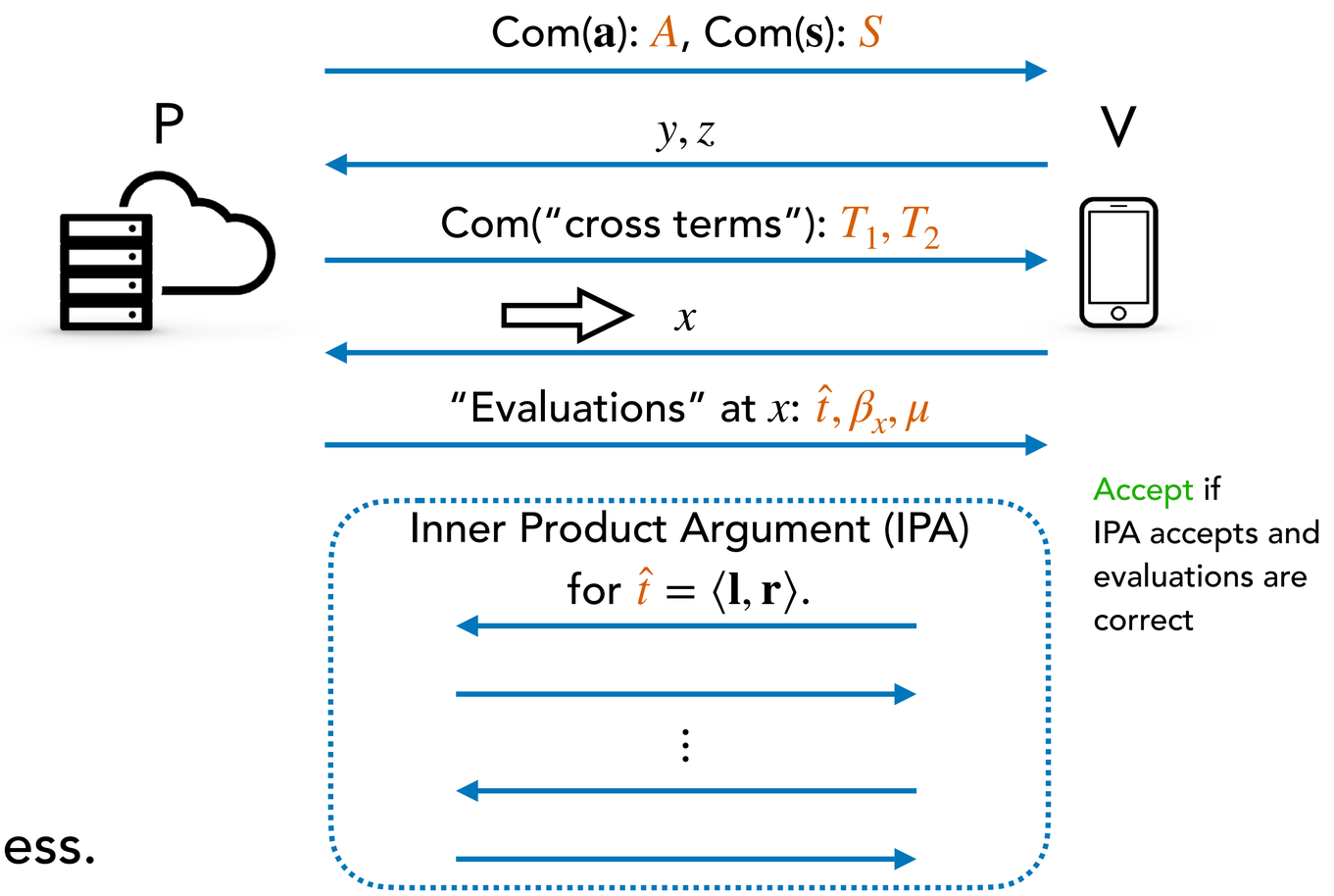
Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

Recall: We need to show Bulletproofs satisfy KS, k-***ZK***, and k-***UR*** for the same round k.

Q: Which round k to prove k-ZK and k-UR?

A: Choose the *last* round with P's randomness.
($k = 2$ in this case)



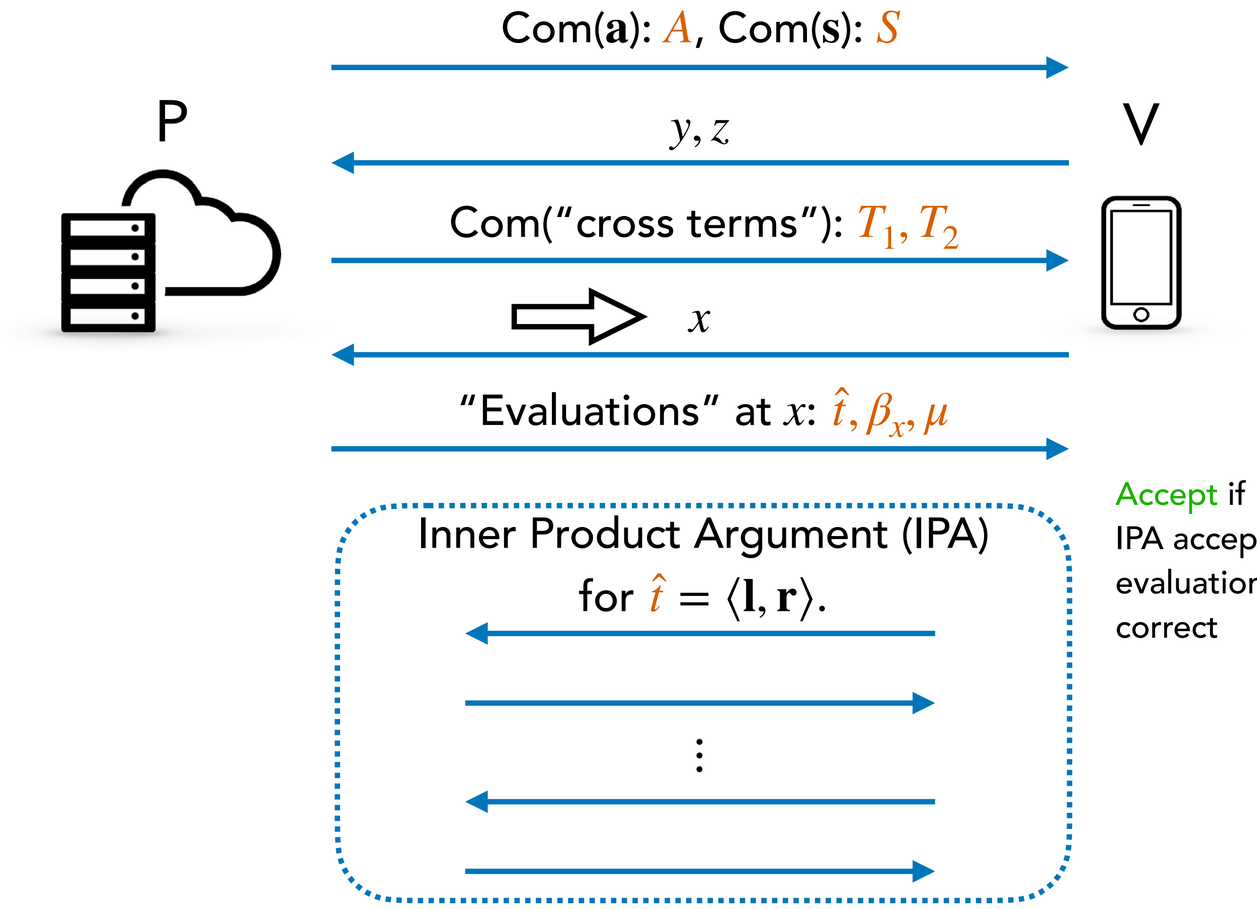
Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.



Bulletproofs Range Proof

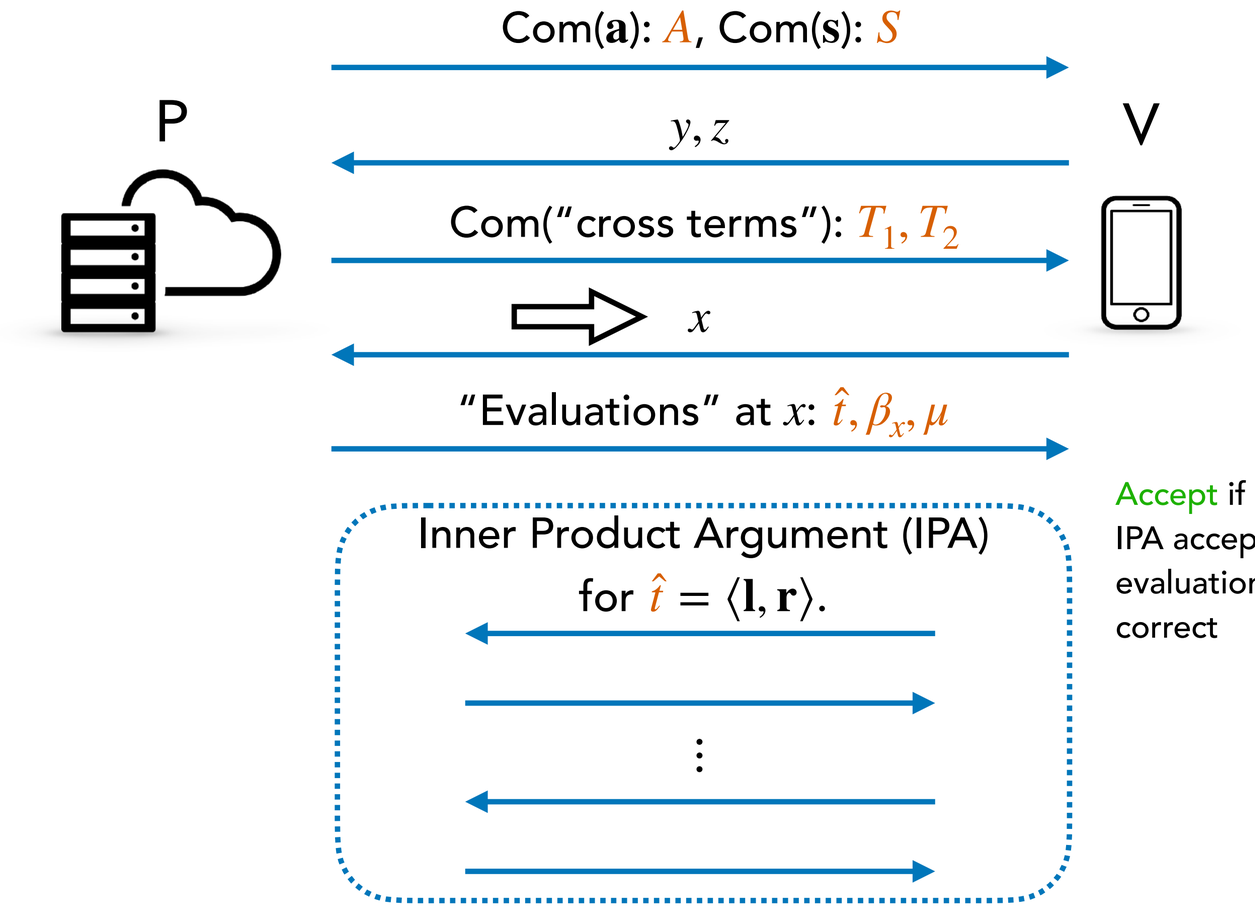
Public Private

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

Problem: How to simulate IPA?

Let $\mathbf{a} = \text{BinDecomp}(v)$.



Bulletproofs Range Proof

Public Private

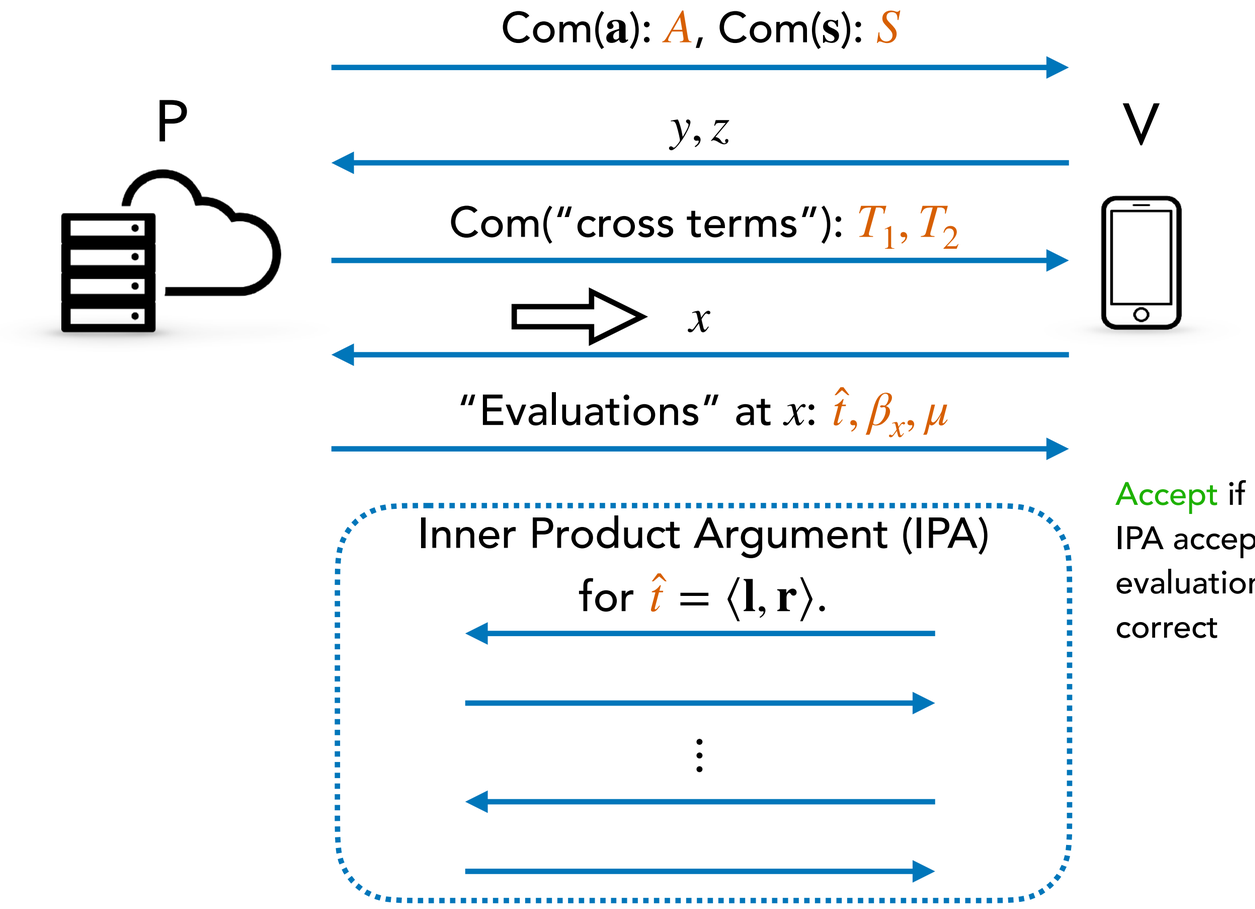
Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

Problem: How to simulate IPA?

Idea:

Let $\mathbf{a} = \text{BinDecomp}(v)$.



Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public Private

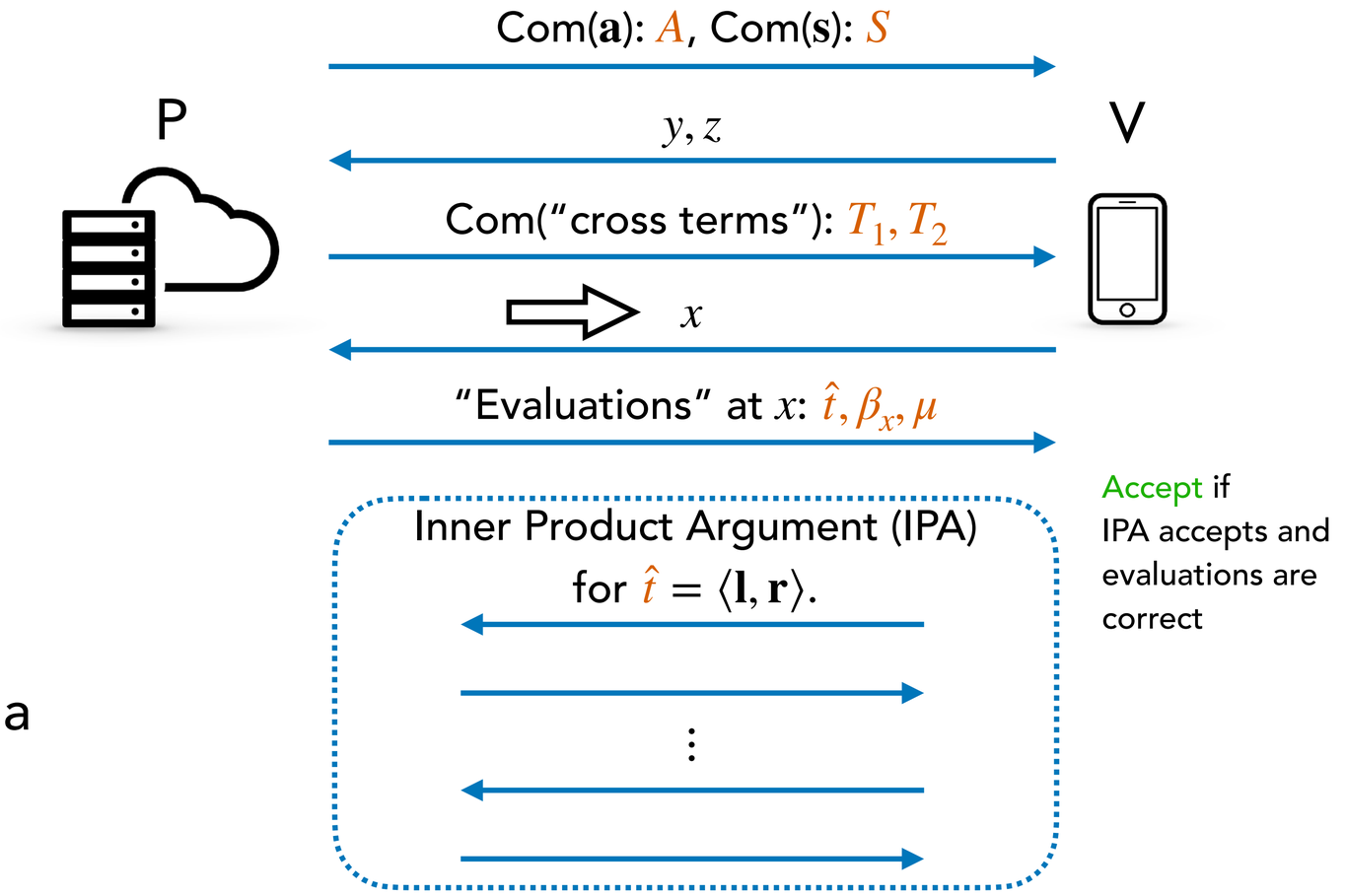
Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

Problem: How to simulate IPA?

Idea:

1. Run the honest prover's algorithm with a *"fake"* witness.



Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public Private

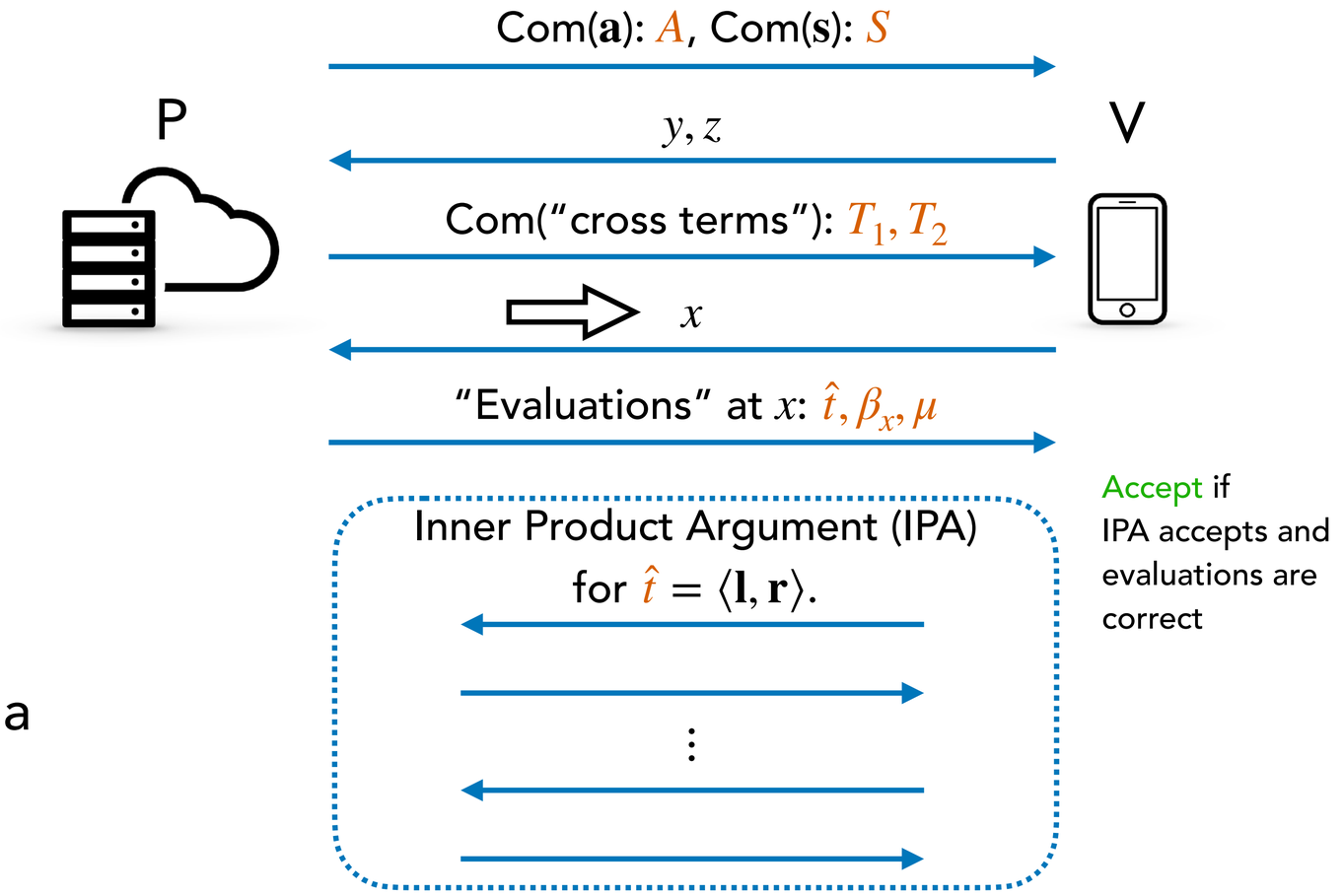
Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

Problem: How to simulate IPA?

Idea:

1. Run the honest prover's algorithm with a *"fake"* witness.
2. Resolve contradiction via choosing k^{th} and $(k+1)^{th}$ message *at the same time*.



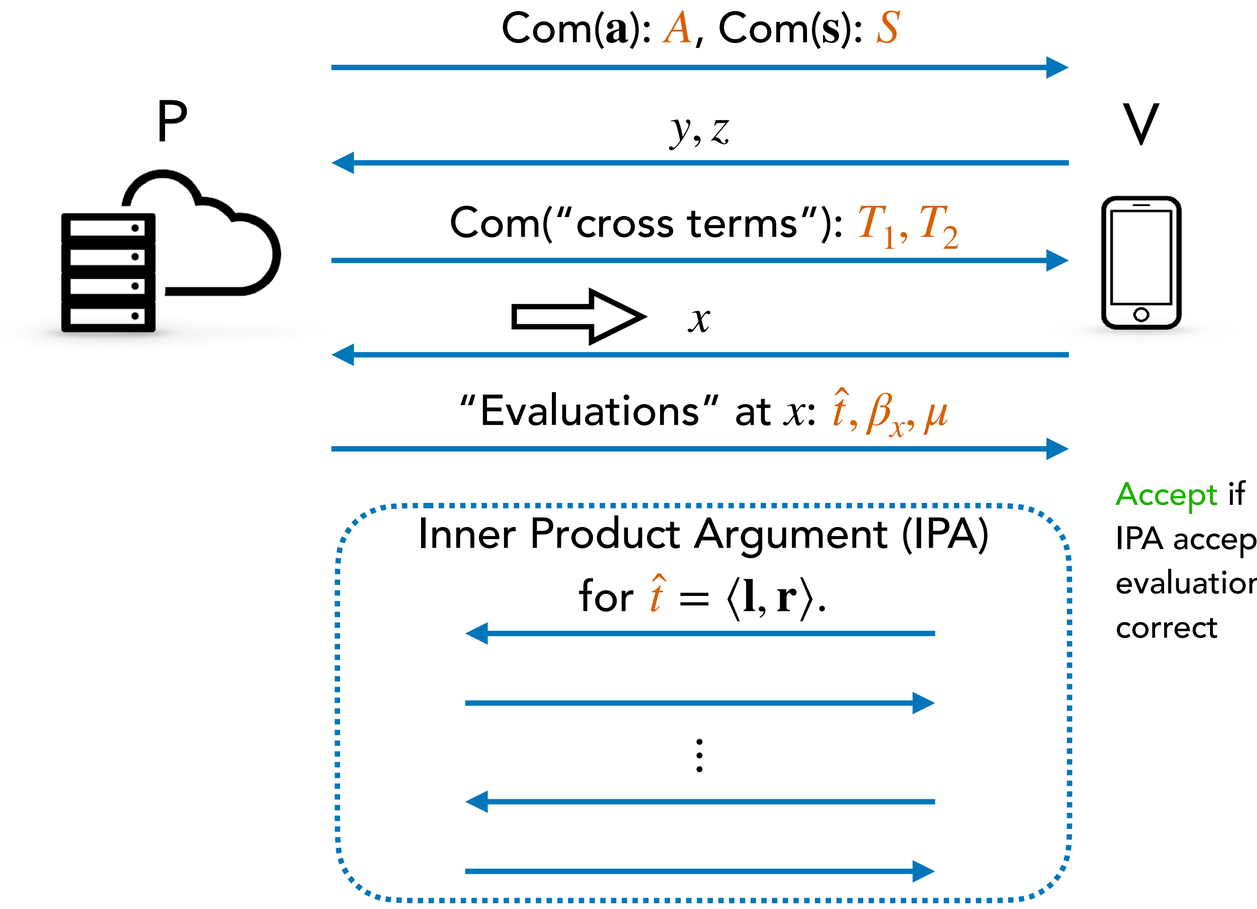
Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.



Bulletproofs Range Proof

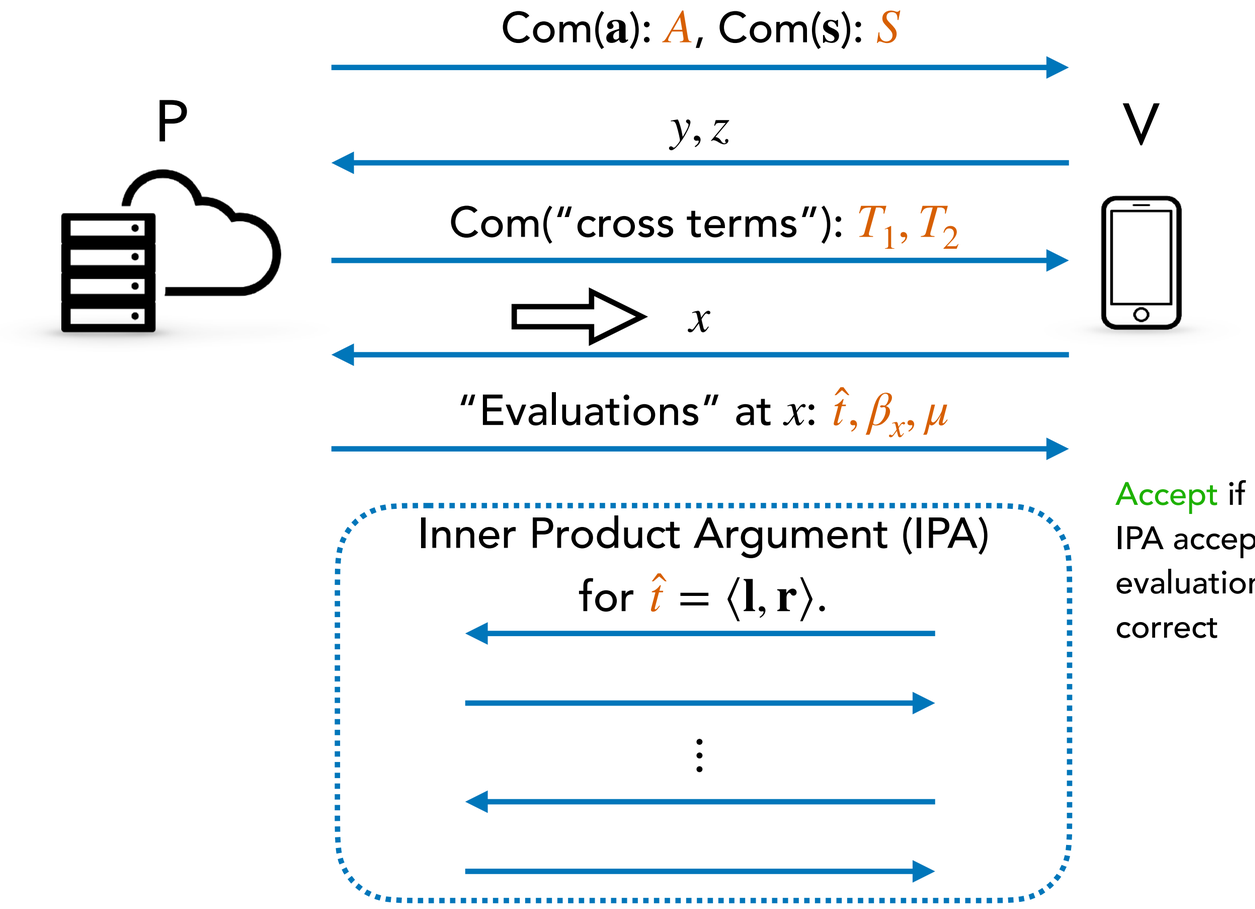
Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

1. Pick random 2^{nd} challenge x.



Bulletproofs Range Proof

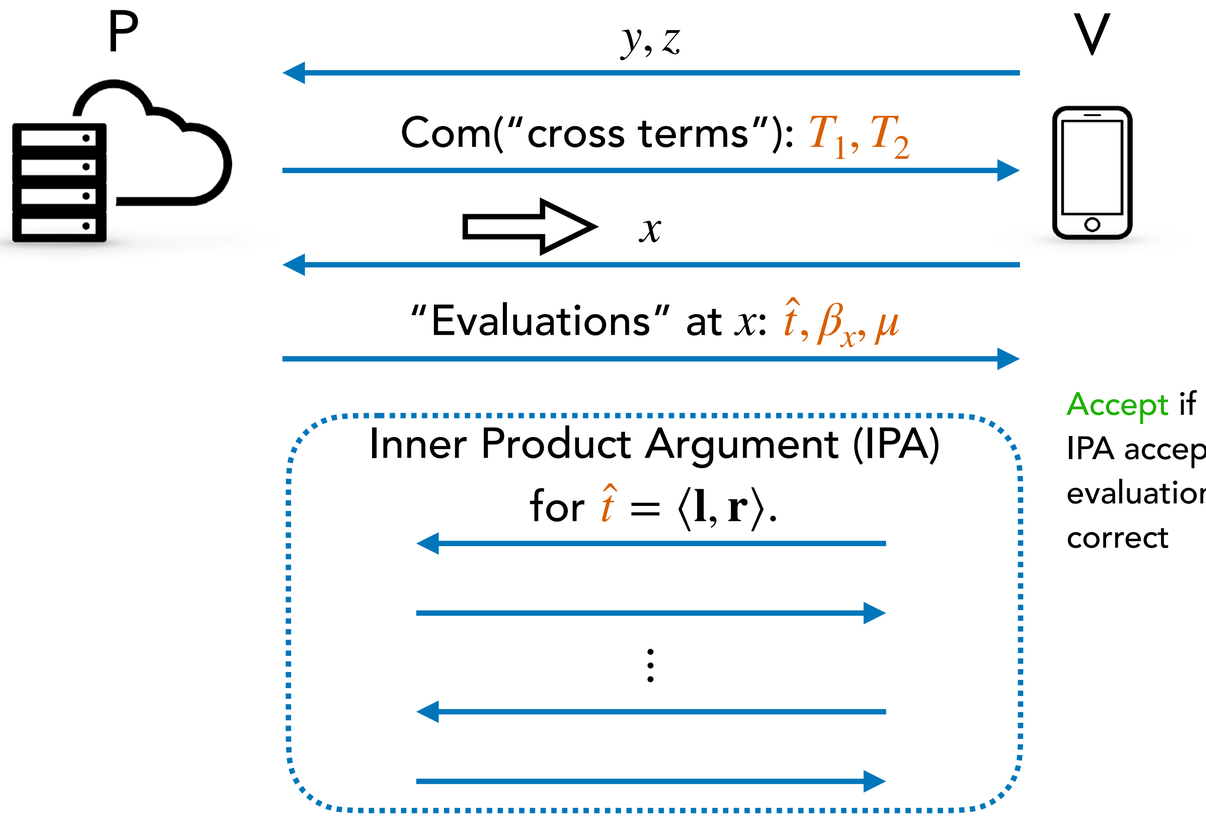
Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

1. Pick random 2^{nd} challenge x.
2. Pick *arbitrary* witness $\mathbf{a}$, *random* blind $\mathbf{s}$. Compute A, S.



Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

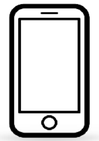
Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.

1. Pick random 2^{nd} challenge x.
2. Pick *arbitrary* witness $\mathbf{a}$, *random* blind $\mathbf{s}$.
   Compute A, S.
3. Pick *random* evaluations $\hat{t}, \beta_x, \mu$.
   Choose T_1, T_2 *consistent* with evaluations.

P

V



Com($\mathbf{a}$): A, Com($\mathbf{s}$): S

y, z

Com("cross terms"): T_1, T_2

x

"Evaluations" at x: $\hat{t}, \beta_x, \mu$

Inner Product Argument (IPA)
for $\hat{t} = \langle \mathbf{l}, \mathbf{r} \rangle$.

Accept if
IPA accepts and
evaluations are
correct

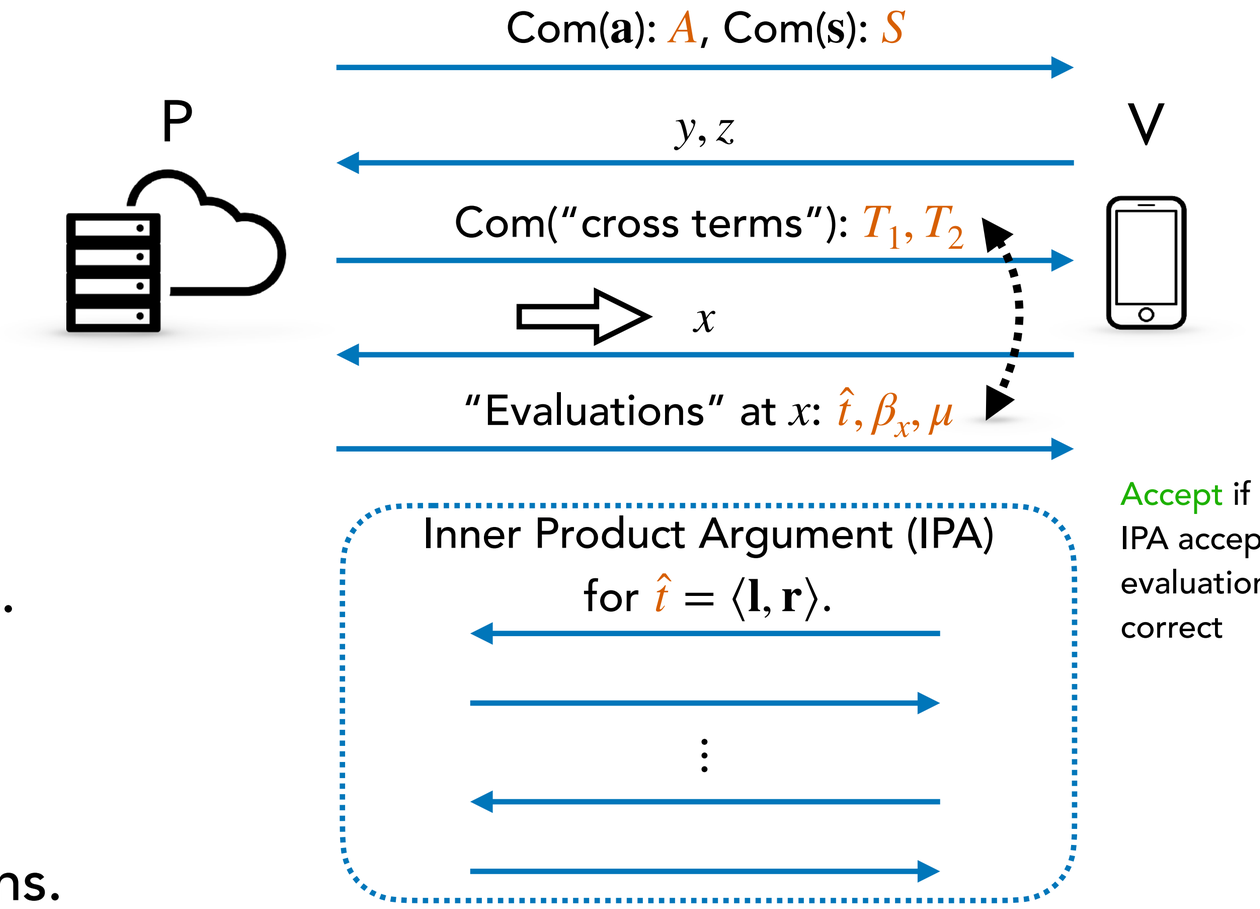
Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

2-ZK: Simulator can only choose x first.



Accept if IPA accepts and evaluations are correct

1. Pick random 2^{nd} challenge x.
2. Pick *arbitrary* witness $\mathbf{a}$, *random* blind $\mathbf{s}$.
   Compute A, S.
3. Pick *random* evaluations $\hat{t}, \beta_x, \mu$.
   Choose T_1, T_2 *consistent* with evaluations.

$$g^{\hat{t}} \cdot h^{\beta_x} = V^{z^2} \cdot g^{\delta(y,z)} \cdot T_1^{\,x} \cdot T_2^{\,x^2}$$

(eval check)

Bulletproofs Range Proof

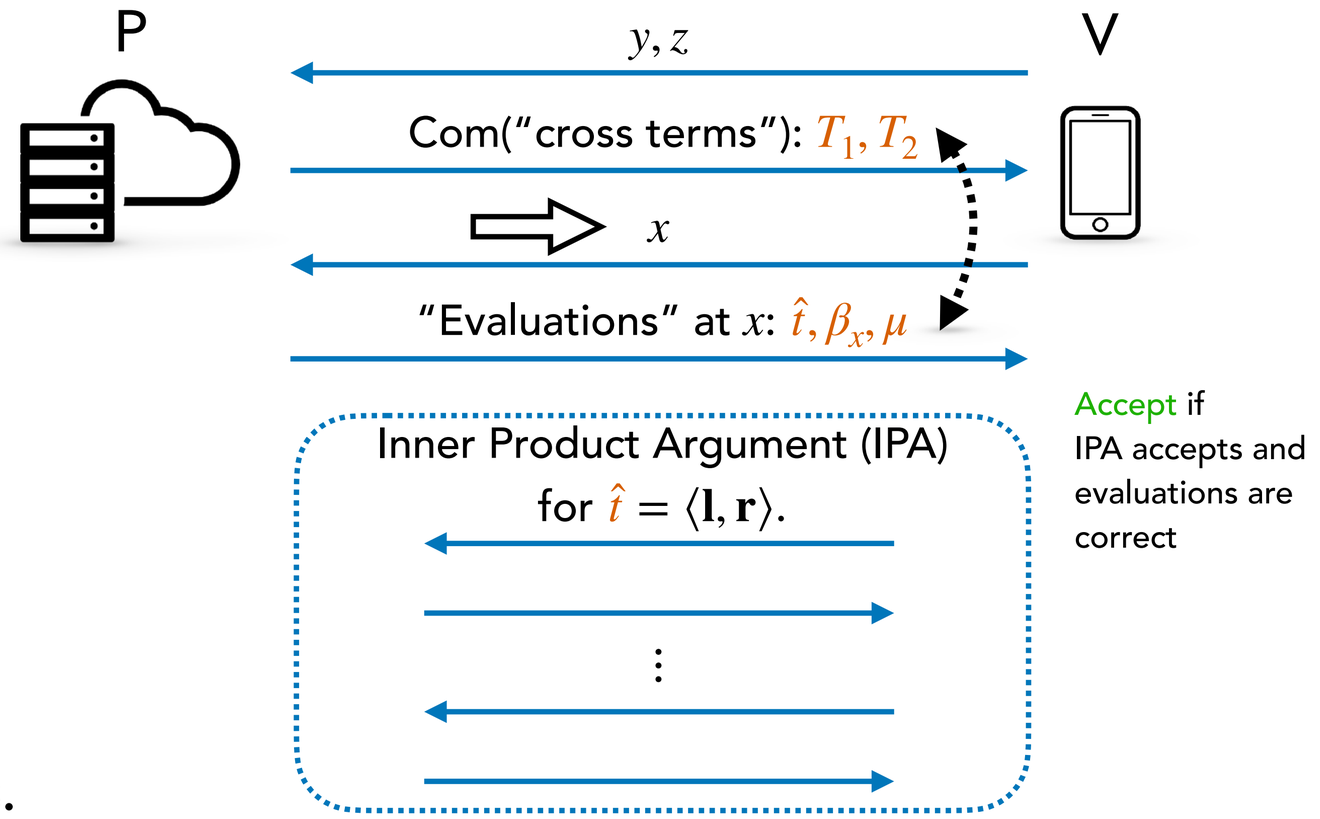
Let $\mathbf{a} = \text{BinDecomp}(v)$.

Public: V Private: v, r

Relation: $V = g^v h^r$ and $0 \leq v \leq 2^n - 1$

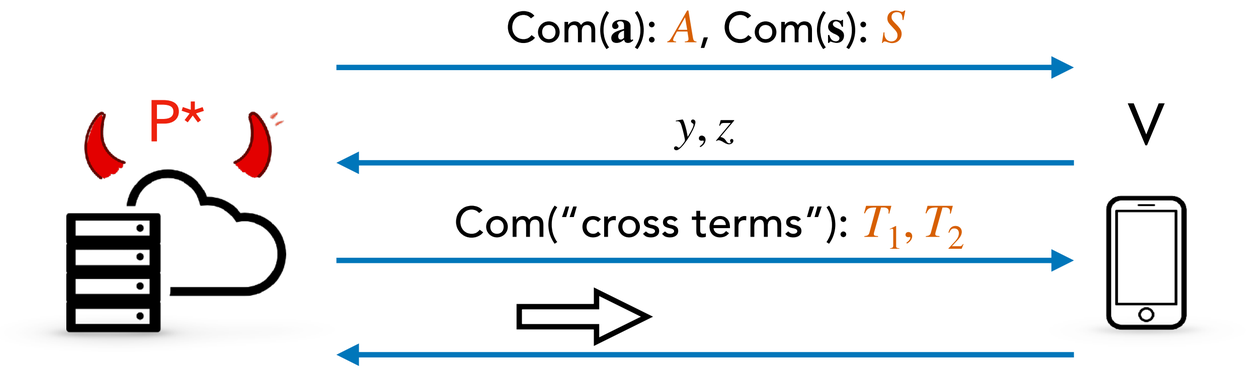
2-ZK: Simulator can only choose x first.

1. Pick random 2^{nd} challenge x.
2. Pick *arbitrary* witness $\mathbf{a}$, *random* blind $\mathbf{s}$.
   Compute A, S.
3. Pick *random* evaluations $\hat{t}, \beta_x, \mu$.
   Choose T_1, T_2 *consistent* with evaluations.
4. Execute IPA with satisfying witness $\mathbf{l}, \mathbf{r}$
   (derived from $\mathbf{a}, \mathbf{s}$).



Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

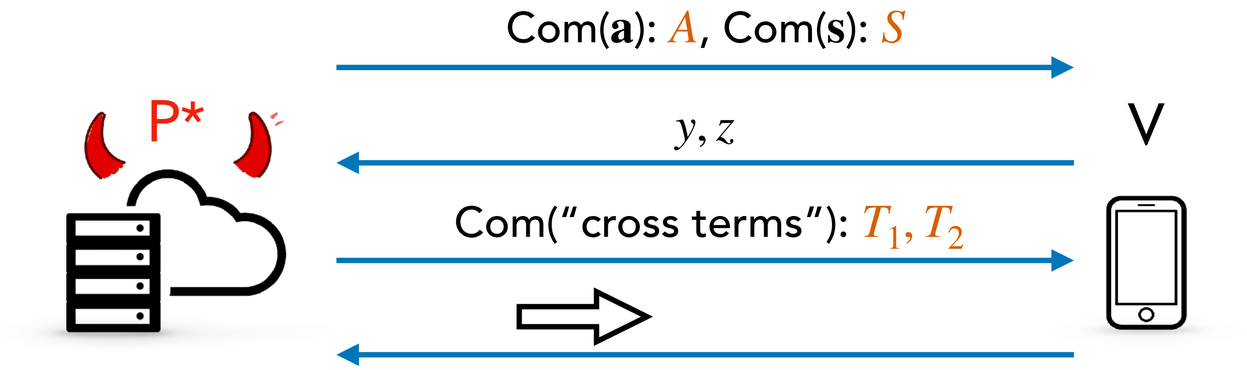


Accept if
IPA accepts and
evaluations are
correct

Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

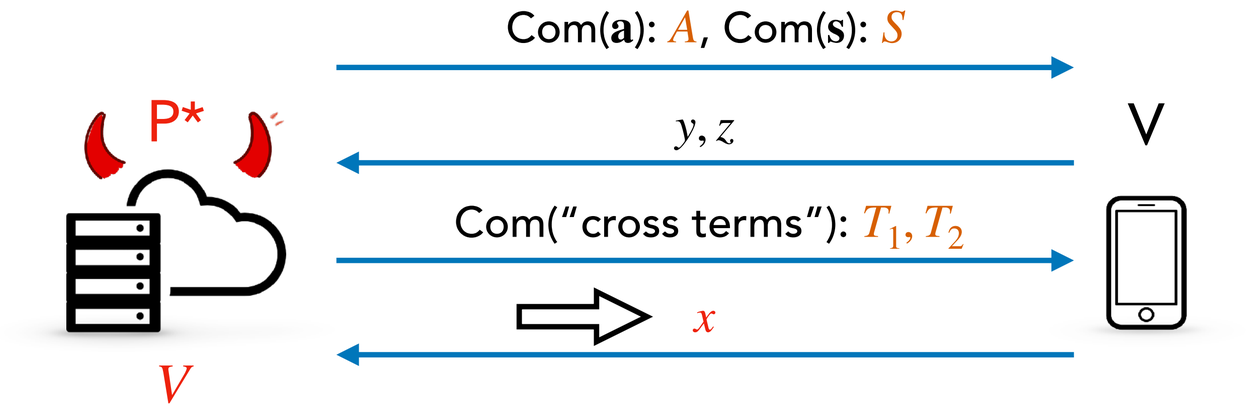


Accept if IPA accepts and evaluations are correct

Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

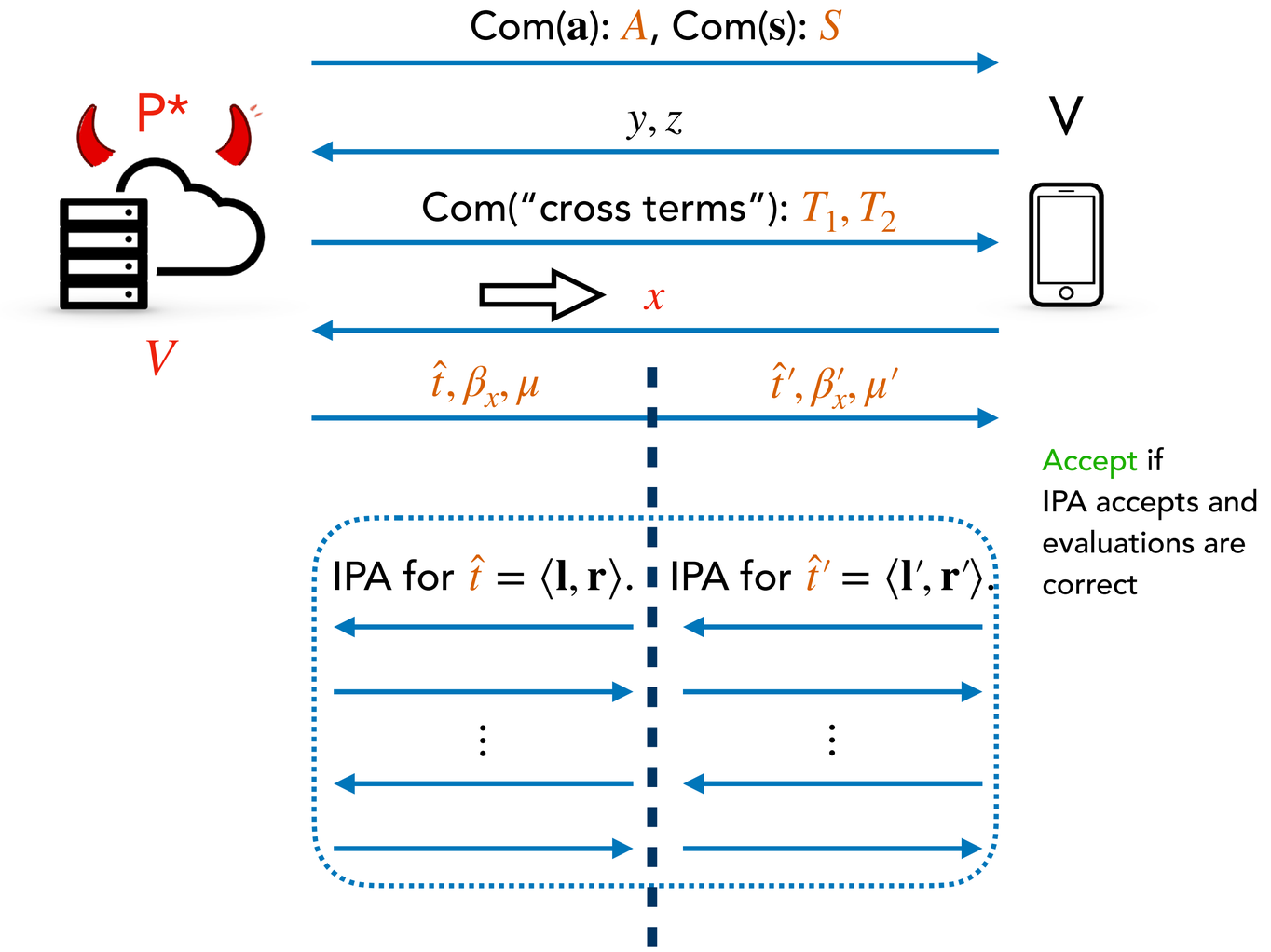


Accept if IPA accepts and evaluations are correct

Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

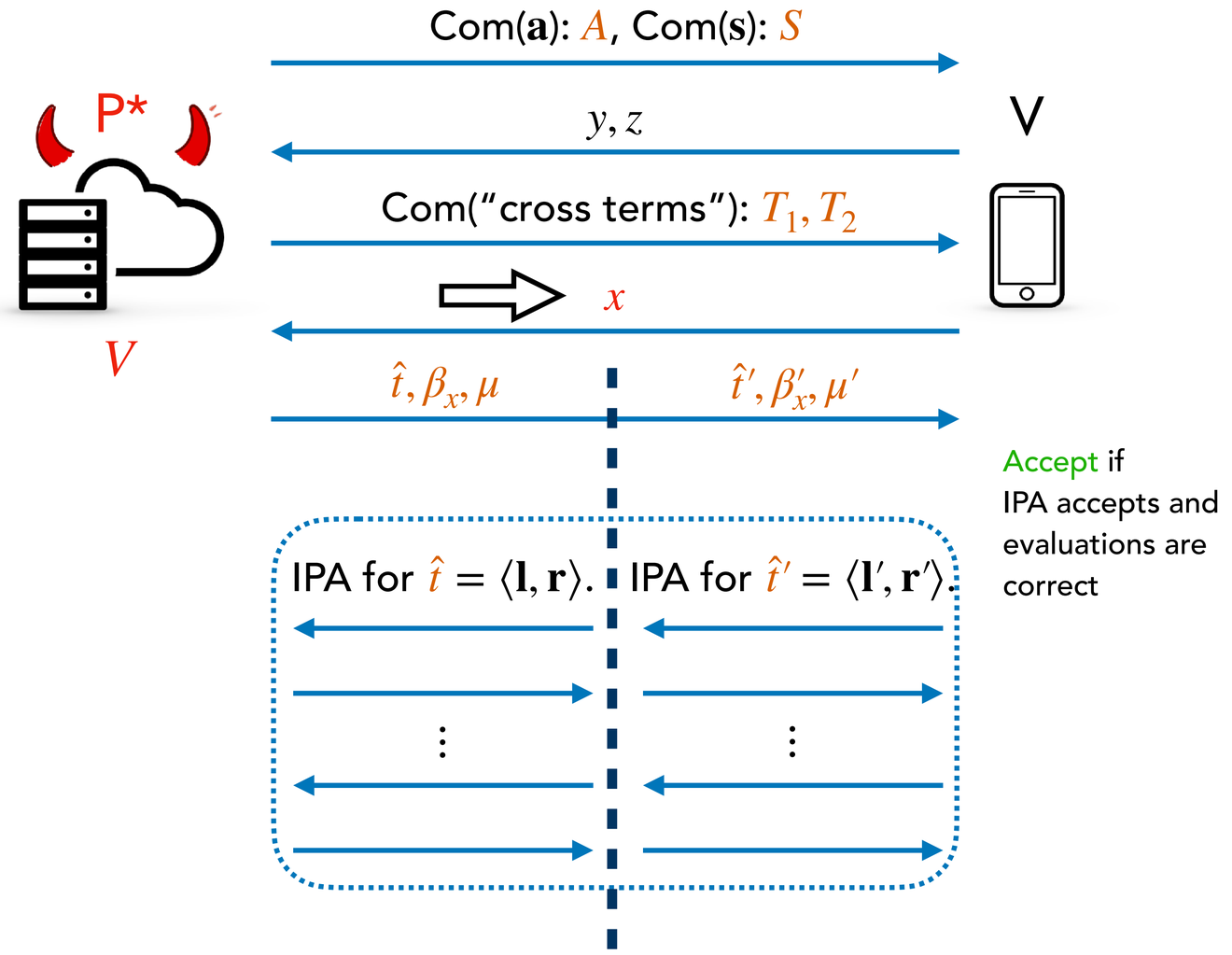


Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.

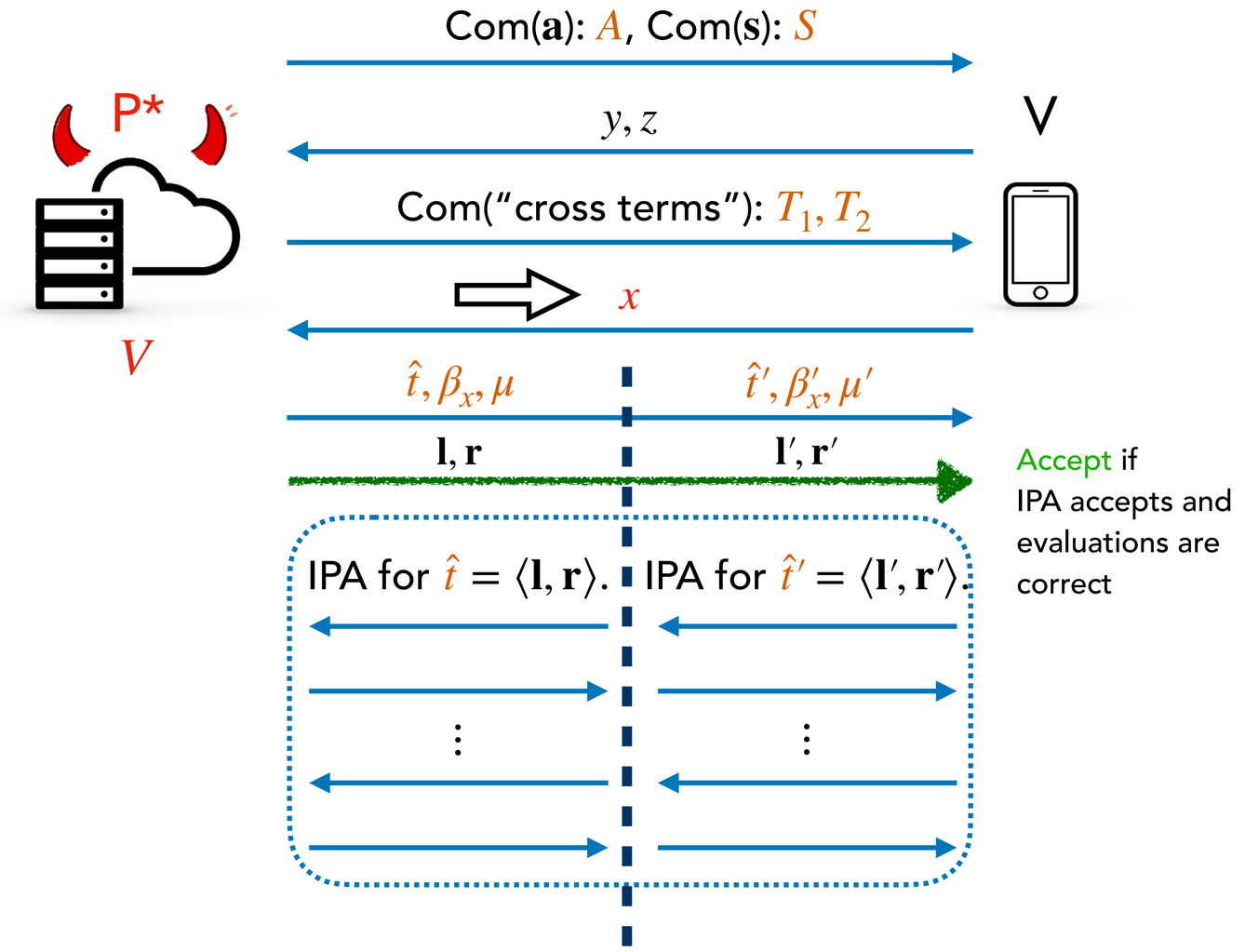


Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.

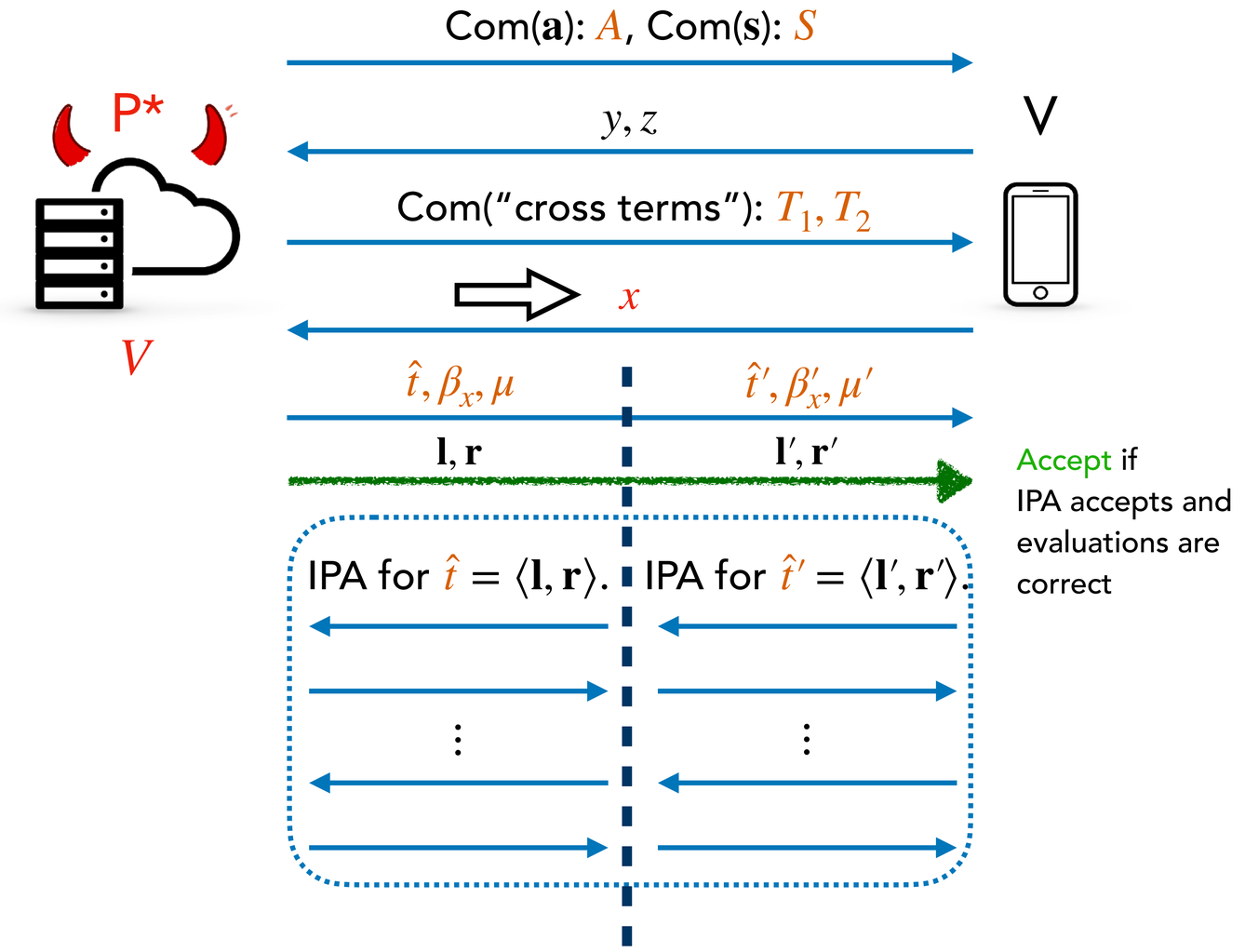


Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.
2. If $(\hat{t}, \beta_x) \neq (\hat{t}', \beta'_x)$, we have a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.



Bulletproofs Range Proof

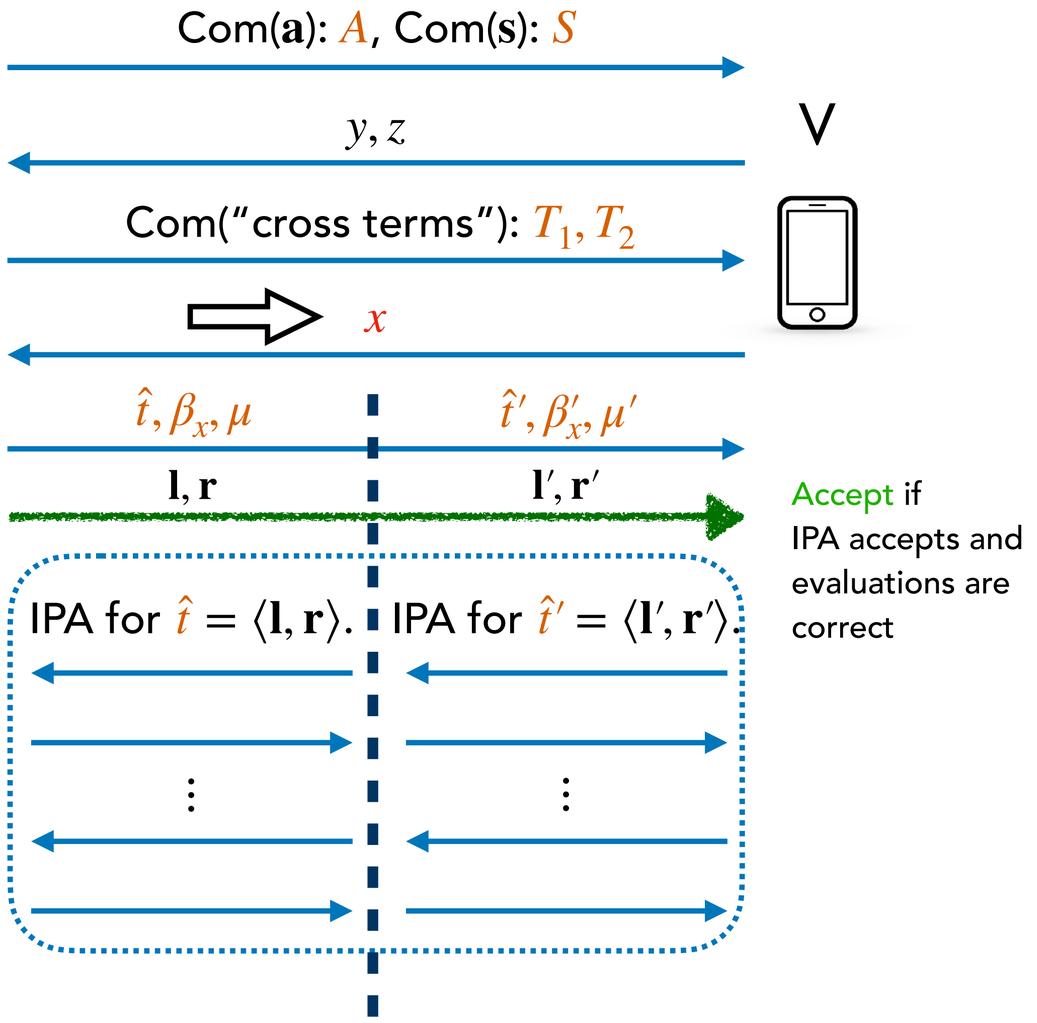
Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.
2. If $(\hat{t}, \beta_x) \neq (\hat{t}', \beta'_x)$, we have a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.

$$g^{\hat{t}} \cdot h^{\beta_x} = V^{z^2} \cdot g^{\delta(y,z)} \cdot T_1^{x} \cdot T_2^{x^2} = g^{\hat{t}'} \cdot h^{\beta'_x}$$

(eval check)

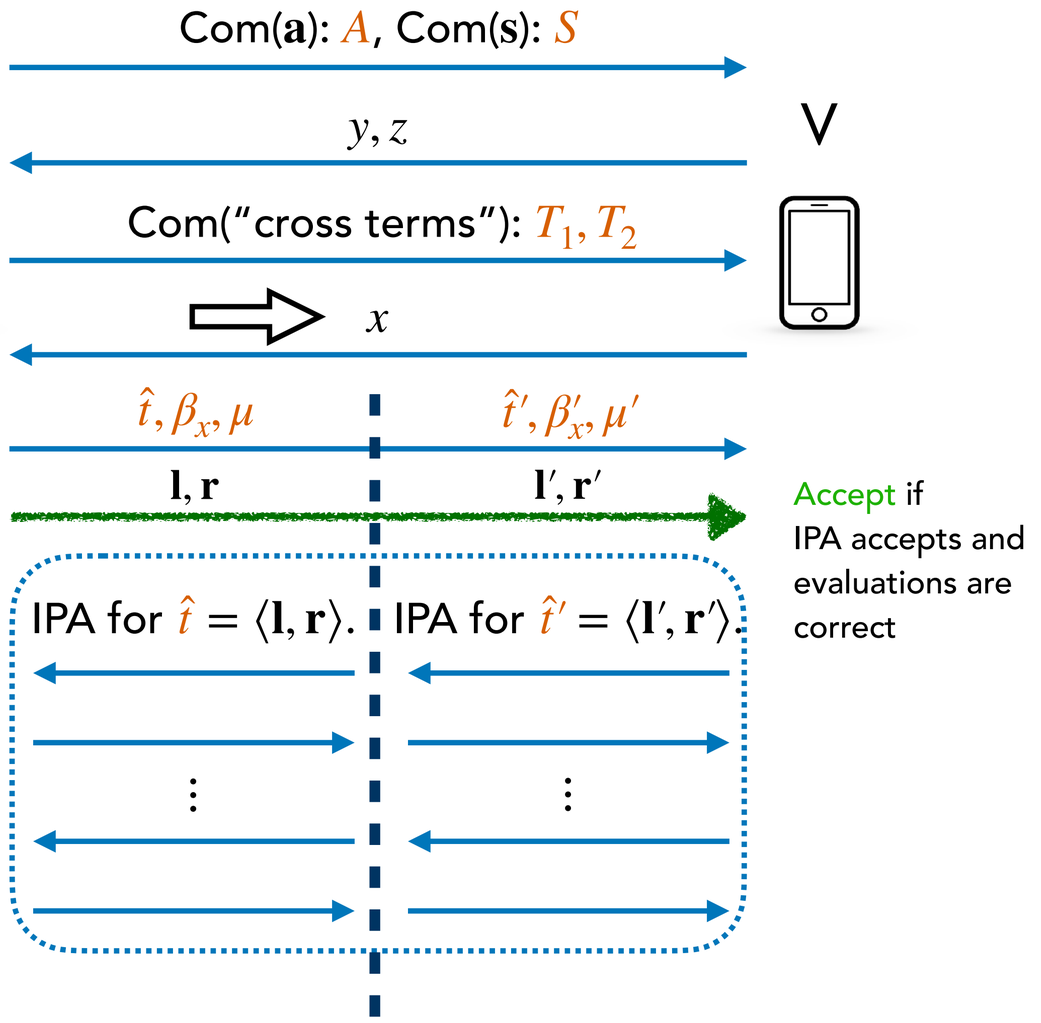


Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.
2. If $(\hat{t}, \beta_x) \neq (\hat{t}', \beta'_x)$, we have a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.
3. Else if $(\mathbf{l}, \mathbf{r}, \mu) \neq (\mathbf{l}', \mathbf{r}', \mu')$, we also get a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.

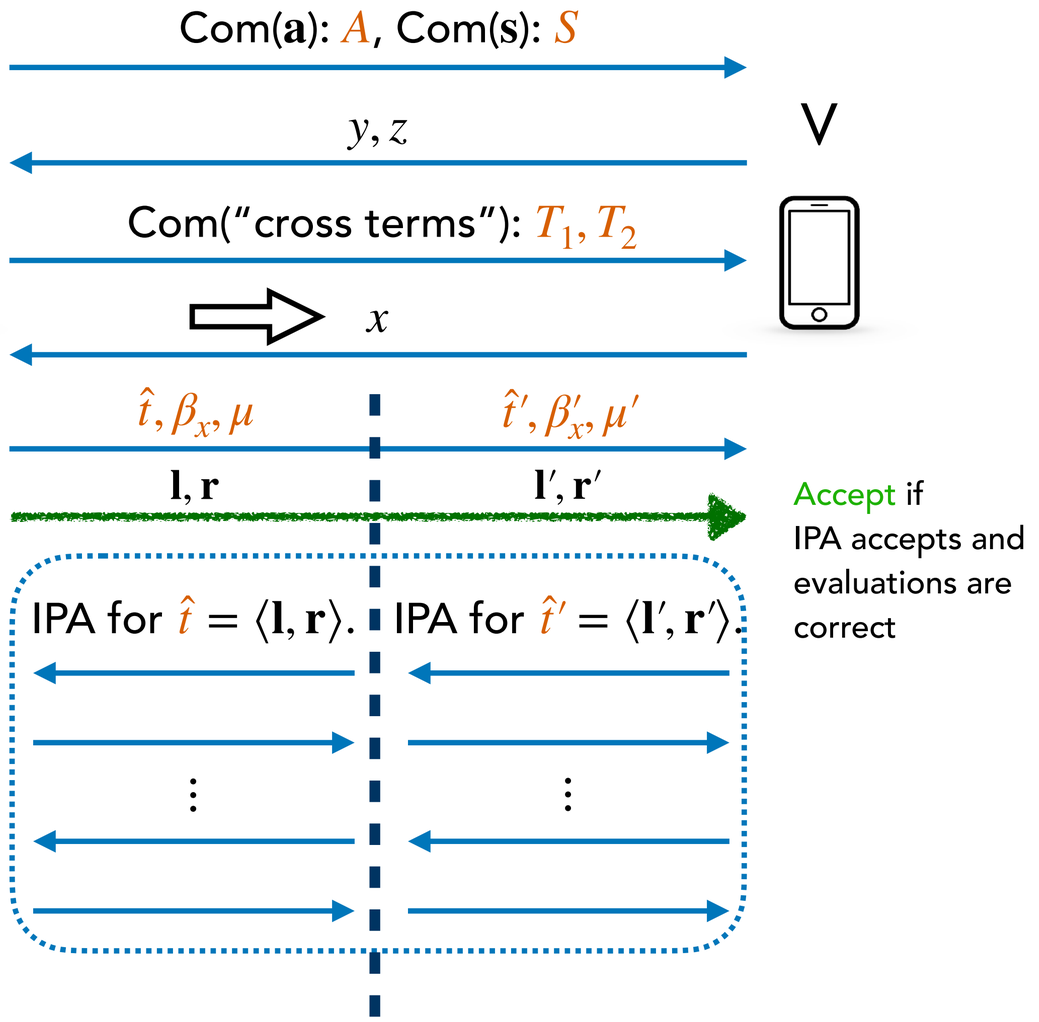


Bulletproofs Range Proof

Let $\mathbf{a} = \text{BinDecomp}(v)$.

2-UR: P* cannot produce two accepting proofs $\pi \neq \pi'$ that agree on A, S, T_1, T_2 (even if it can choose V and x).

1. Use KS extractor for IPA to extract $(\mathbf{l}, \mathbf{r})$ from π_{IPA}, $(\mathbf{l}', \mathbf{r}')$ from π'_{IPA}.
2. If $(\hat{t}, \beta_x) \neq (\hat{t}', \beta'_x)$, we have a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.
3. Else if $(\mathbf{l}, \mathbf{r}, \mu) \neq (\mathbf{l}', \mathbf{r}', \mu')$, we also get a non-trivial DLOG relation $\Longrightarrow$ P* breaks DLOG.
4. Else $(\mathbf{l}, \mathbf{r}) = (\mathbf{l}', \mathbf{r}')$ but $\pi_{\text{IPA}} \neq \pi'_{\text{IPA}} \Longrightarrow$ P* breaks DLOG.



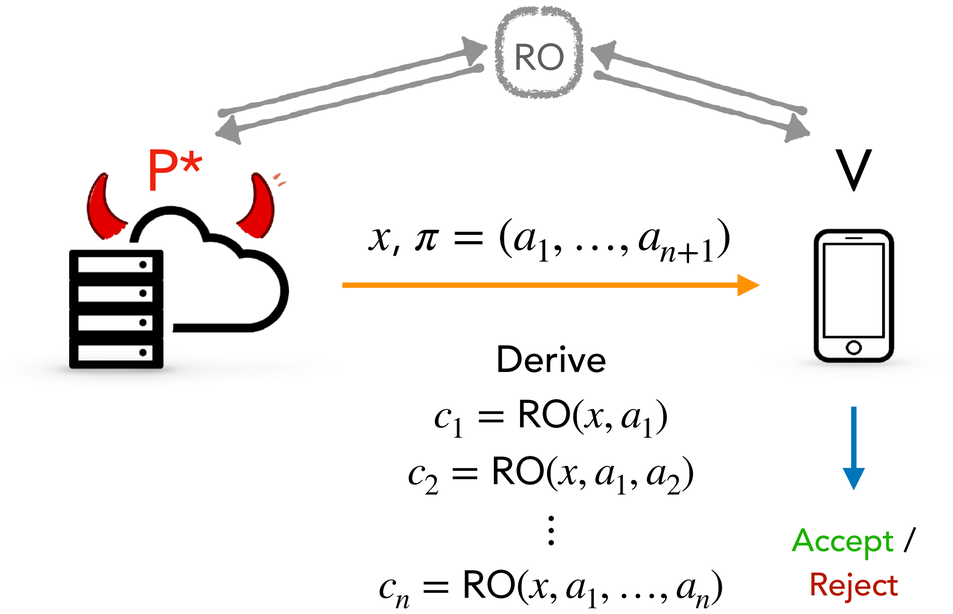
Agenda

1. SIM-EXT = KS + k-ZK + k-UR (for same k)
2. k-ZK and k-UR for Bulletproofs
3. **Knowledge Soundness via Generalized Tree Builder**

Knowledge Soundness from Special Soundness

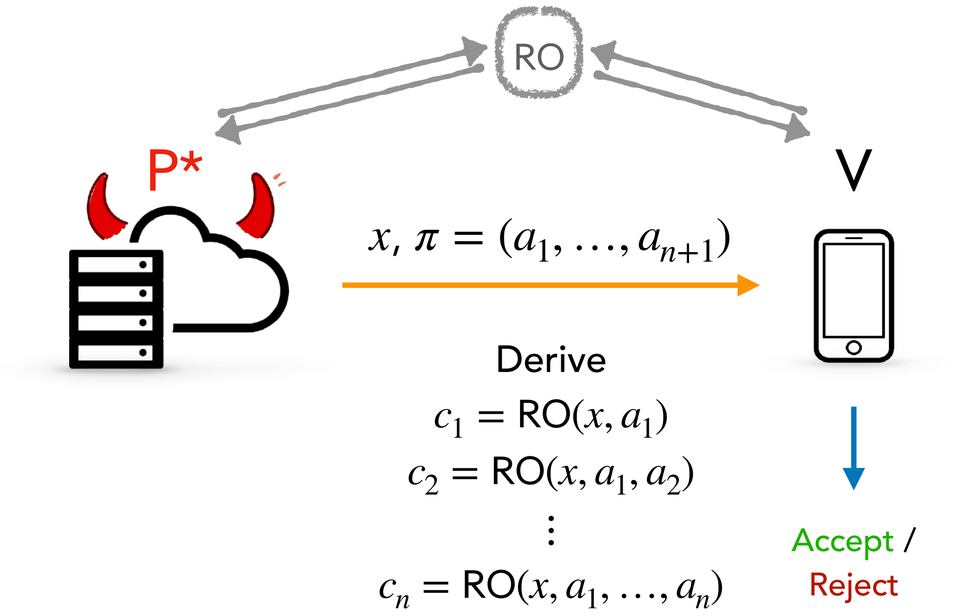
Knowledge Soundness from Special Soundness

F-S Argument:



Knowledge Soundness from Special Soundness

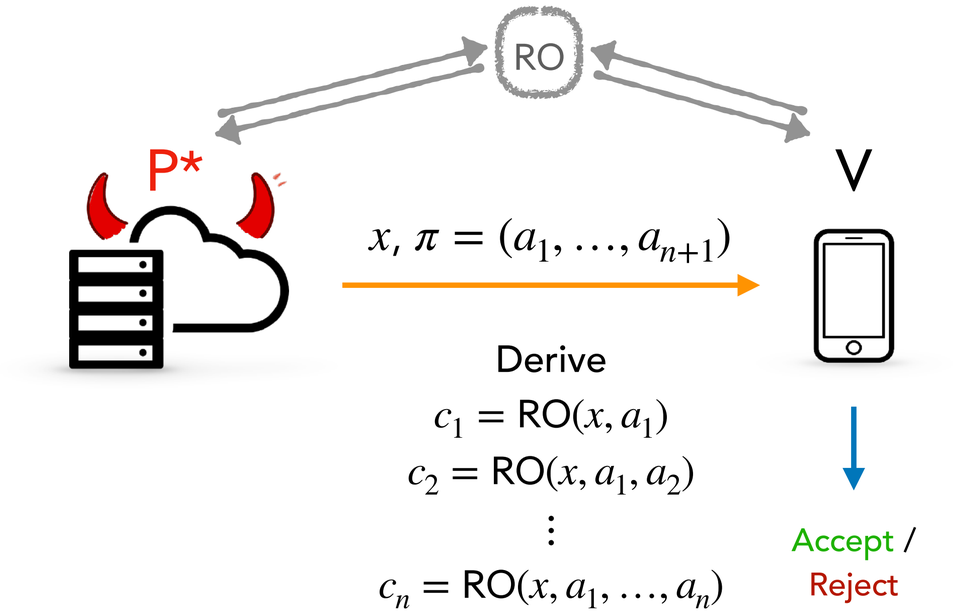
F-S Argument:



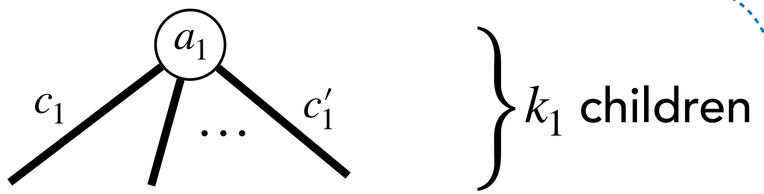
$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

Knowledge Soundness from Special Soundness

F-S Argument:

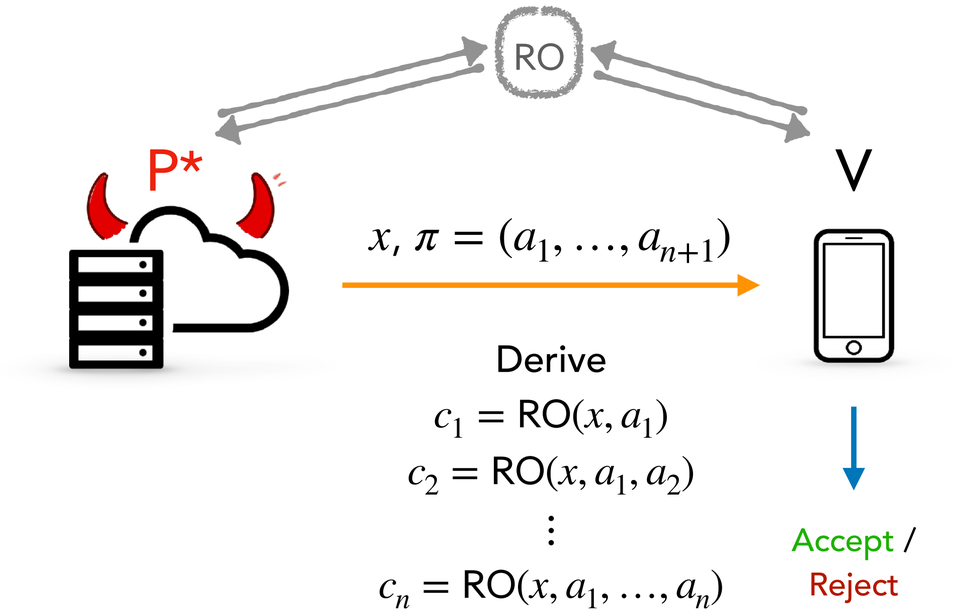


$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

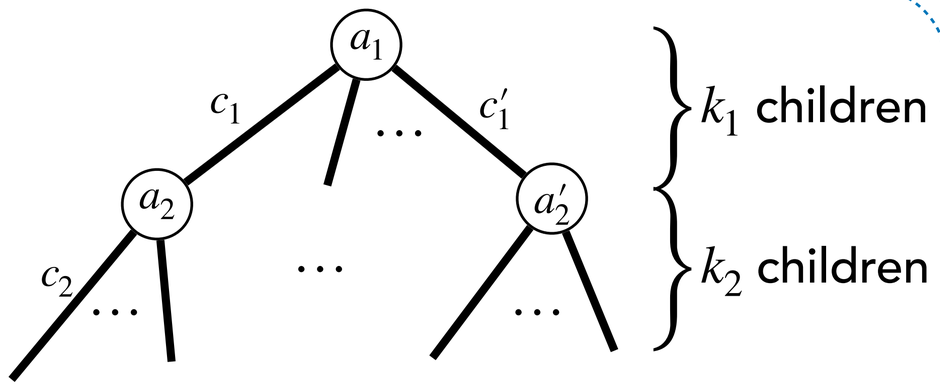


Knowledge Soundness from Special Soundness

F-S Argument:

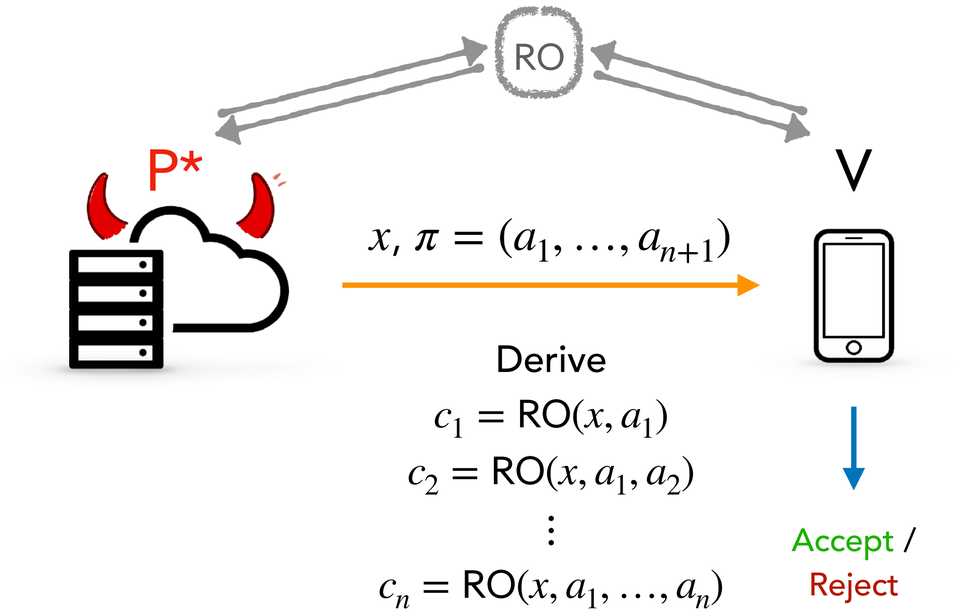


$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

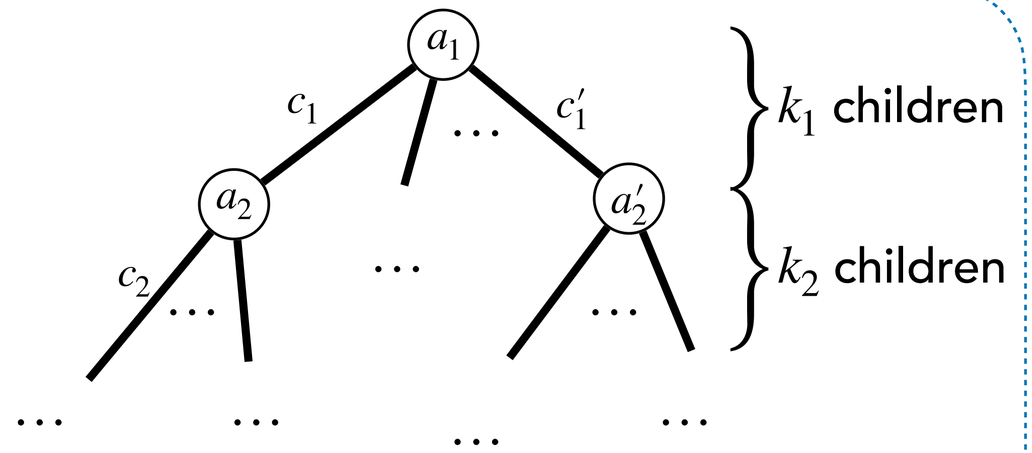


Knowledge Soundness from Special Soundness

F-S Argument:

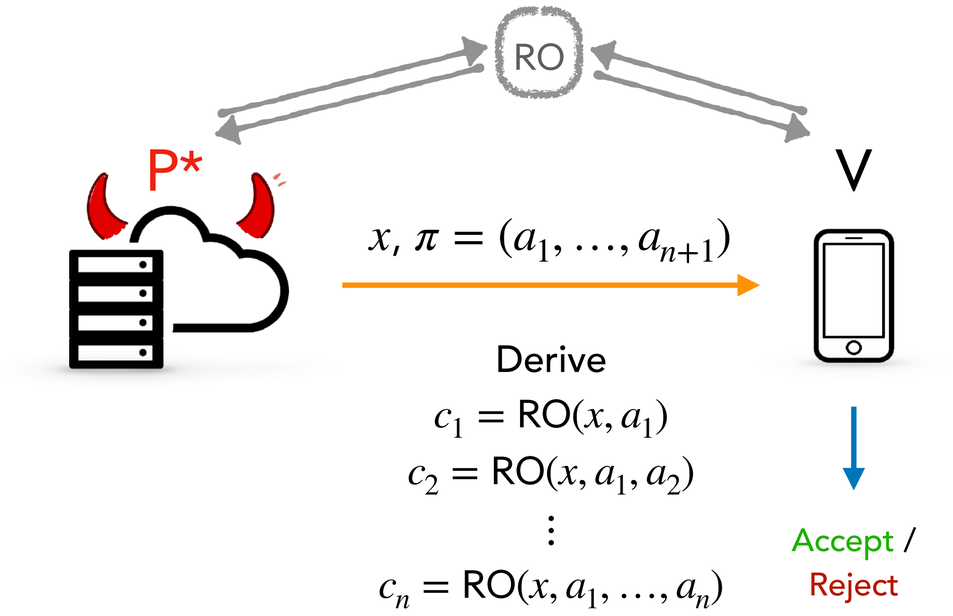


$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

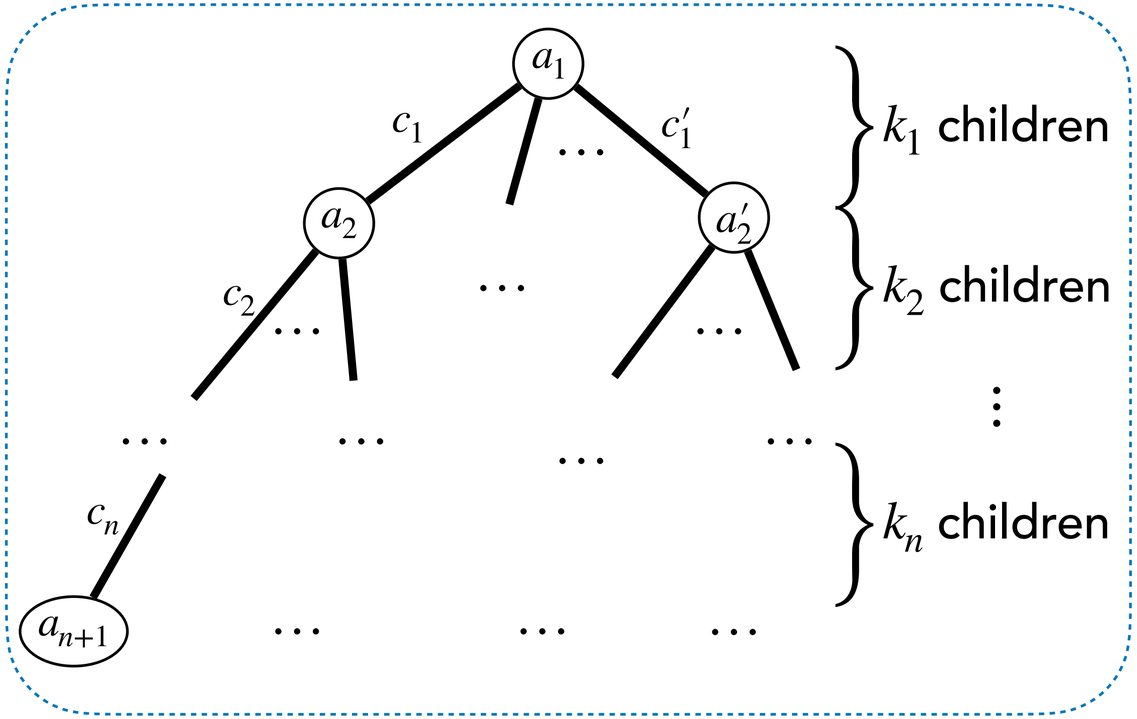


Knowledge Soundness from Special Soundness

F-S Argument:

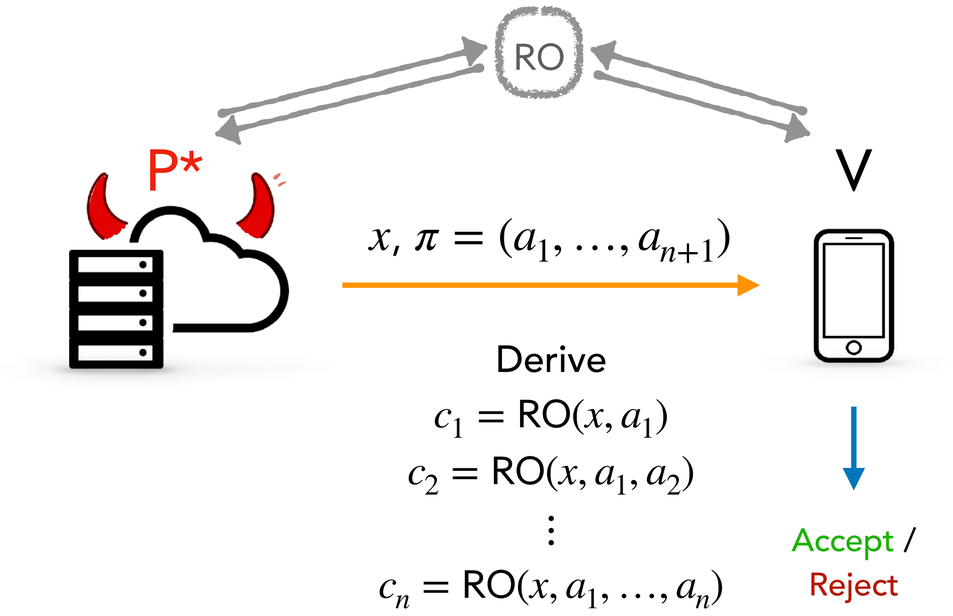


$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

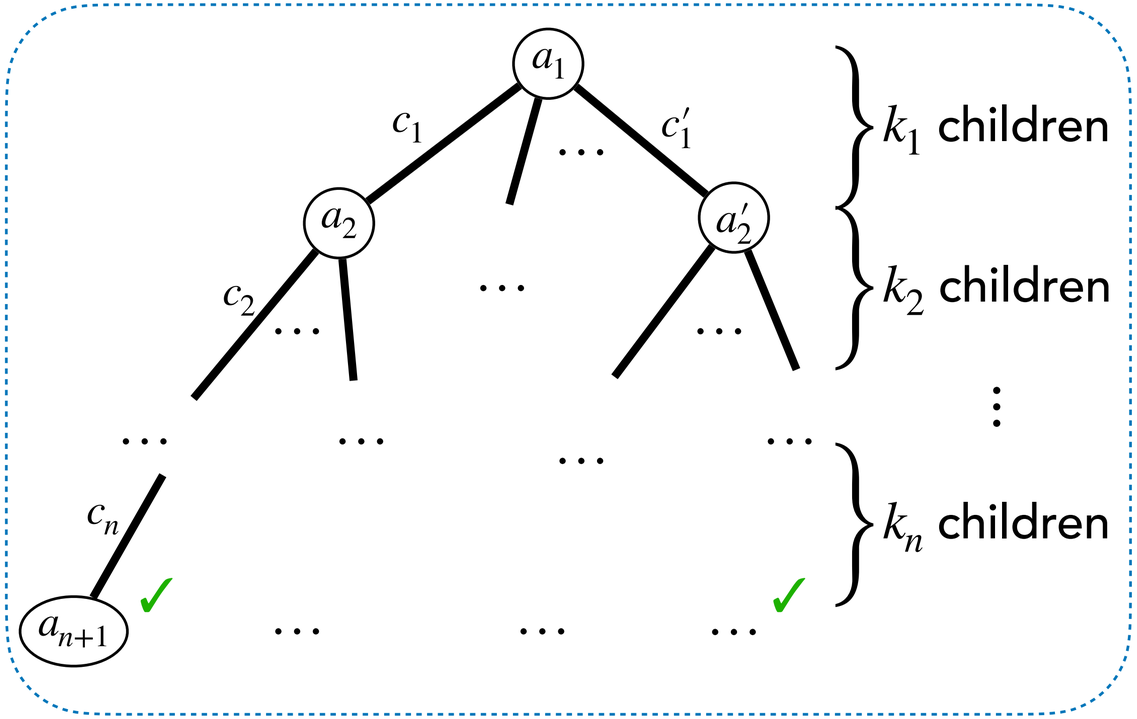


Knowledge Soundness from Special Soundness

F-S Argument:

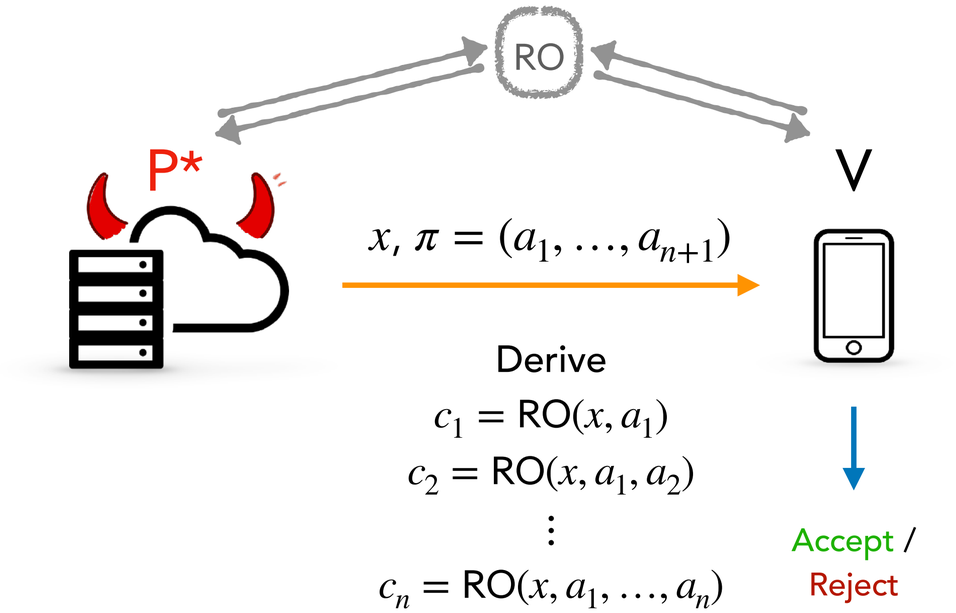


$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts

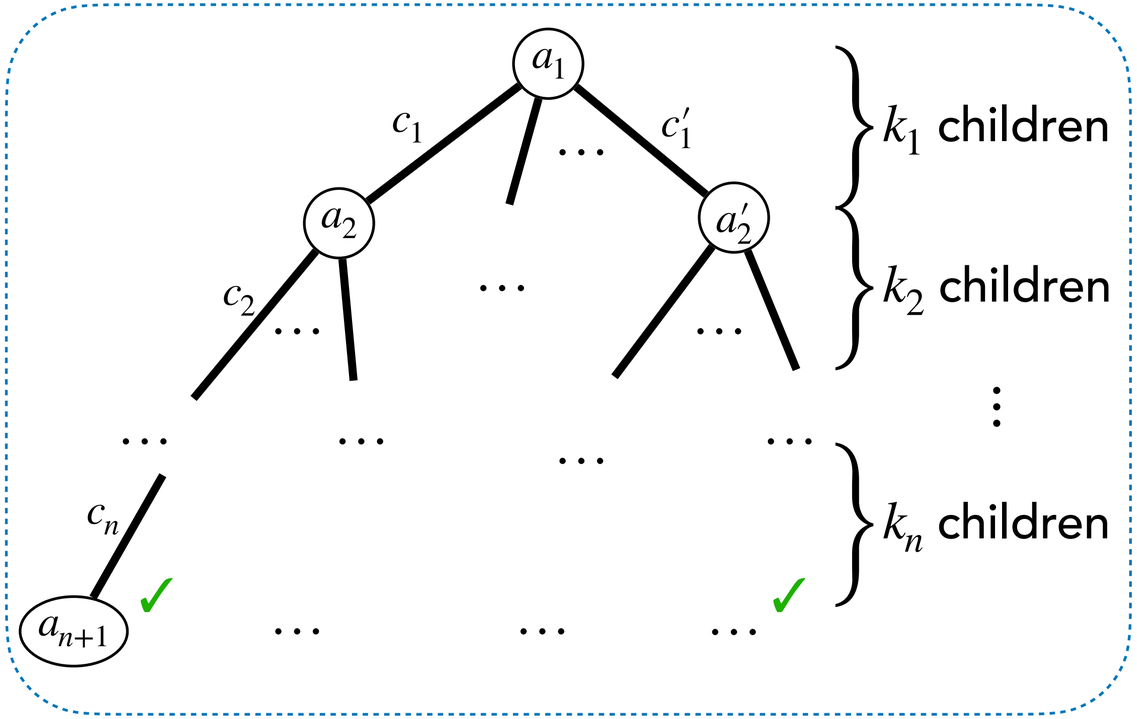


Knowledge Soundness from Special Soundness

F-S Argument:



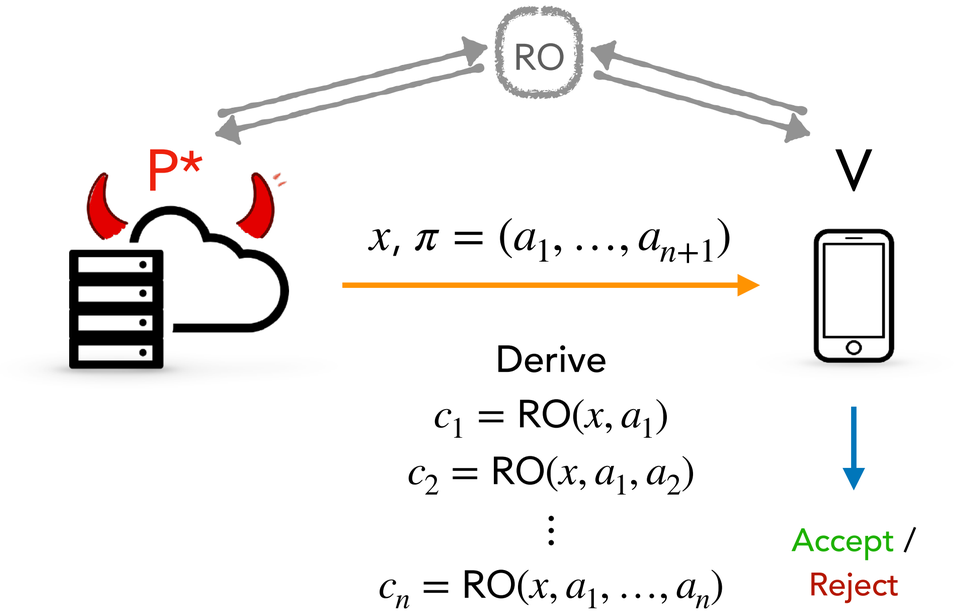
$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts



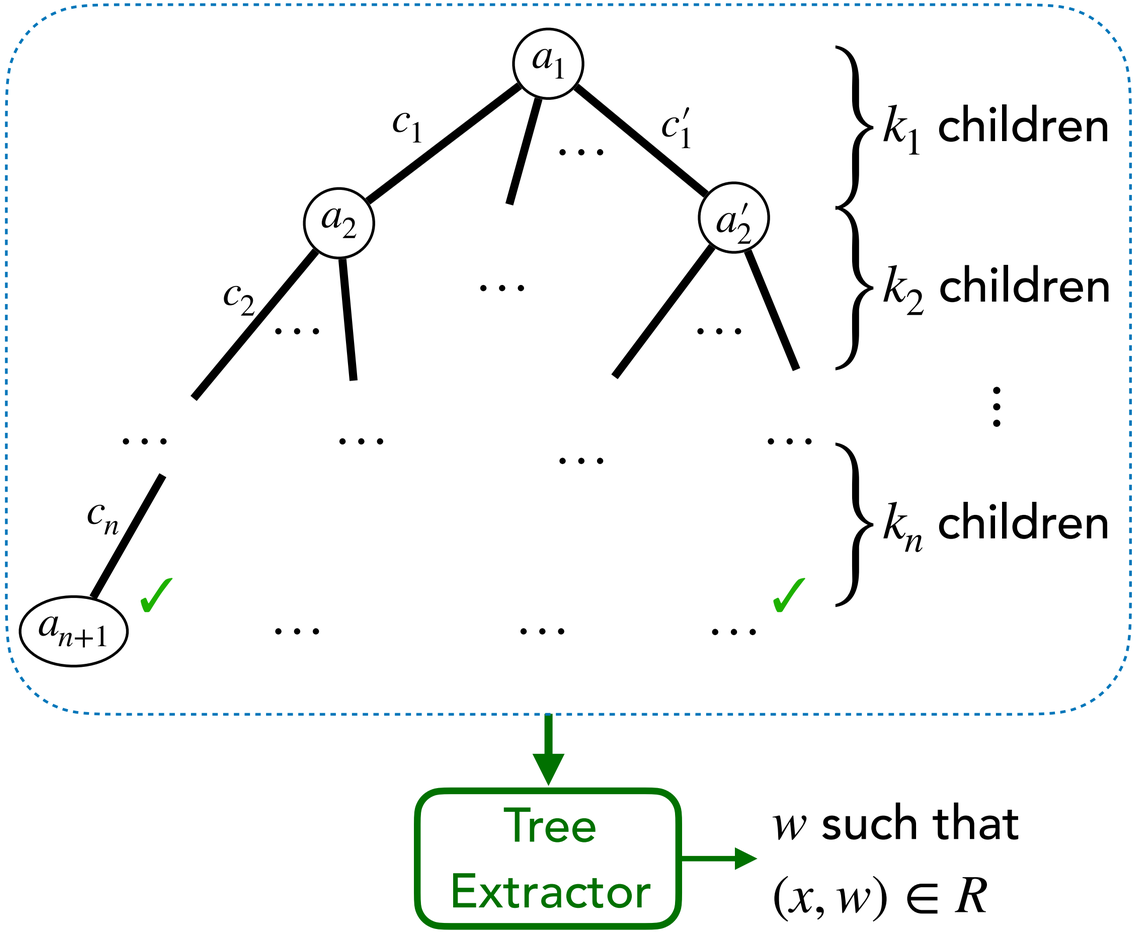
Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Knowledge Soundness from Special Soundness

F-S Argument:



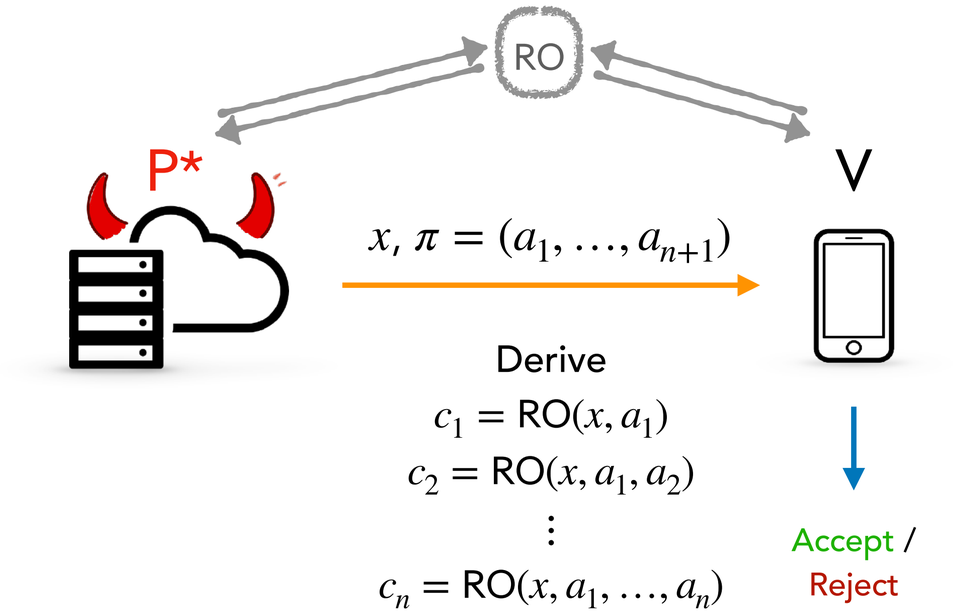
$(k_1, \ldots, k_n)$-Tree of Accepting Transcripts



Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Knowledge Soundness from Special Soundness

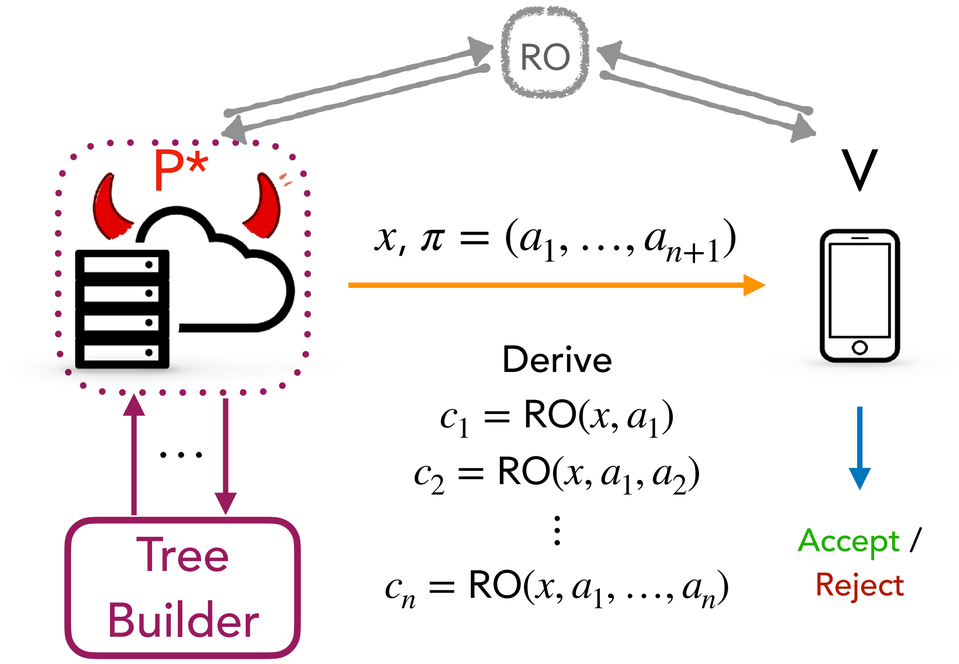
F-S Argument:



Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Knowledge Soundness from Special Soundness

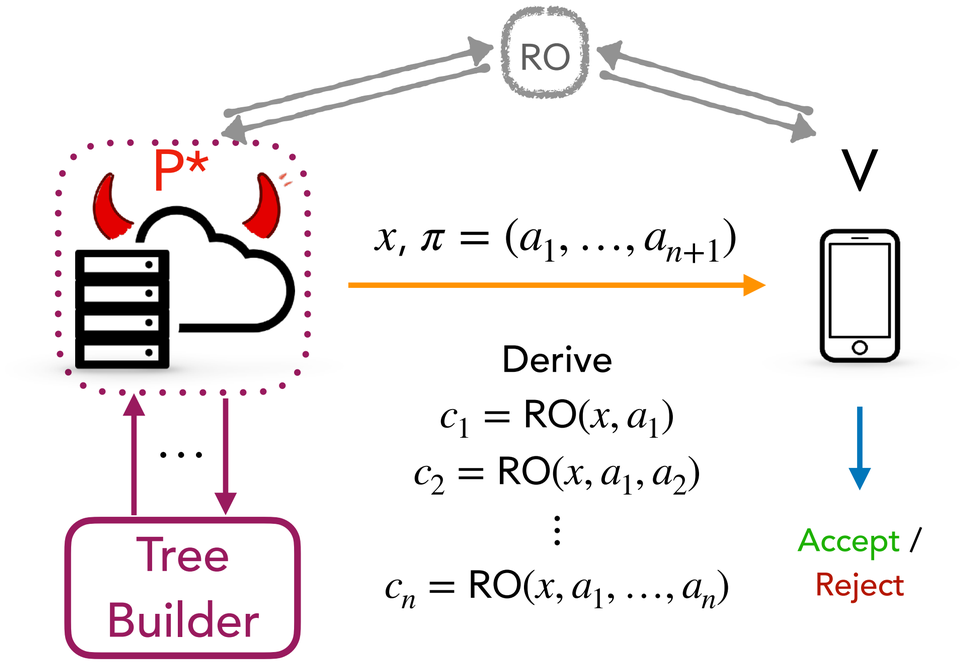
F-S Argument:



Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Knowledge Soundness from Special Soundness

F-S Argument:

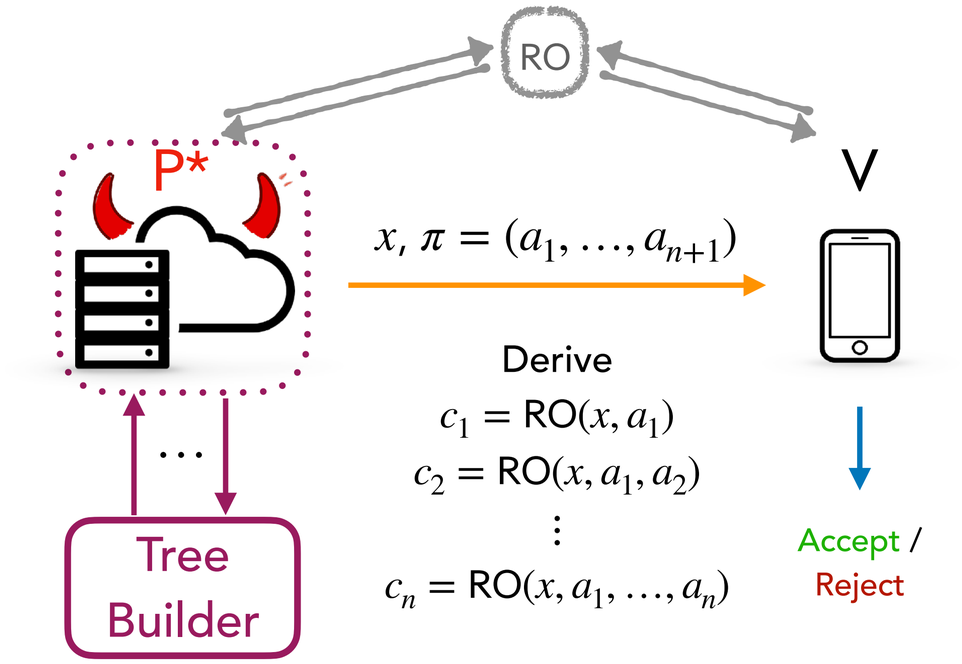


Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Attema et al. (TCC '22): There exists a *tree-builder* TB that builds a $(k_1, \ldots, k_n)$-tree of accepting transcripts in expected poly-time.

Knowledge Soundness from Special Soundness

F-S Argument:



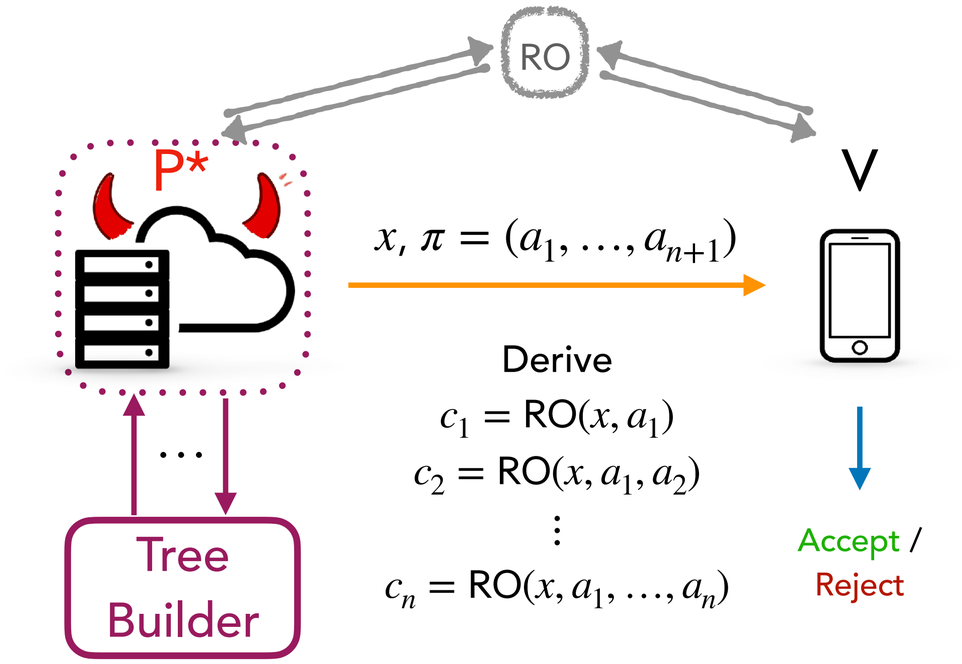
Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Attema et al. (TCC '22): There exists a *tree-builder* TB that builds a $(k_1, \ldots, k_n)$-tree of accepting transcripts in expected poly-time.

Combine TB *with* TE

Knowledge Soundness from Special Soundness

F-S Argument:



Special Soundness: There exists $k_1, \ldots, k_n$ such that a witness w can be extracted from any $(k_1, \ldots, k_n)$-tree of accepting transcripts.

Attema et al. (TCC '22): There exists a *tree-builder* TB that builds a $(k_1, \ldots, k_n)$-tree of accepting transcripts in expected poly-time.

Combine TB *with* TE

Corollary: If a proof system satisfies special soundness, then it satisfies knowledge soundness.

Generalized Special Soundness & Tree Building

Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

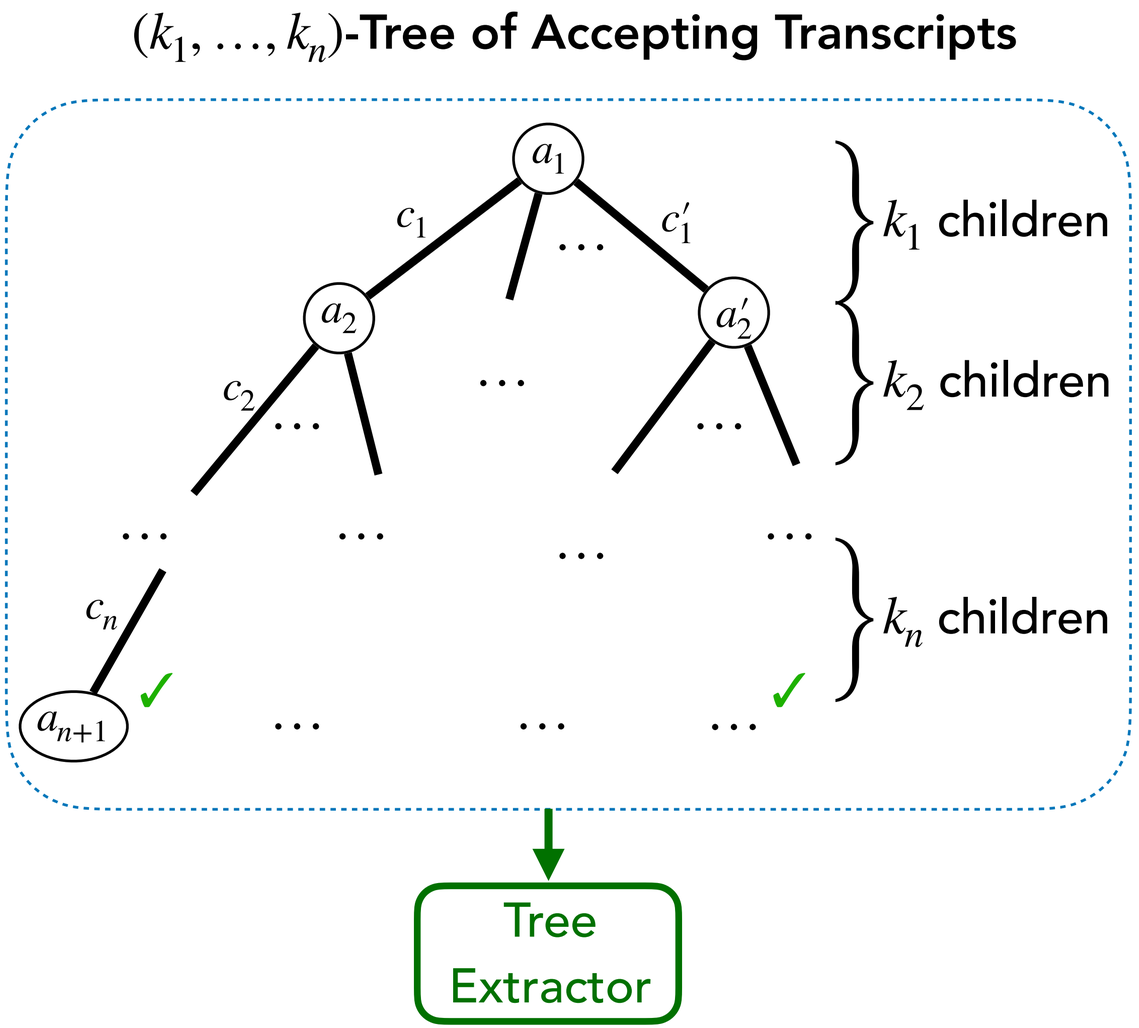
- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).

Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).

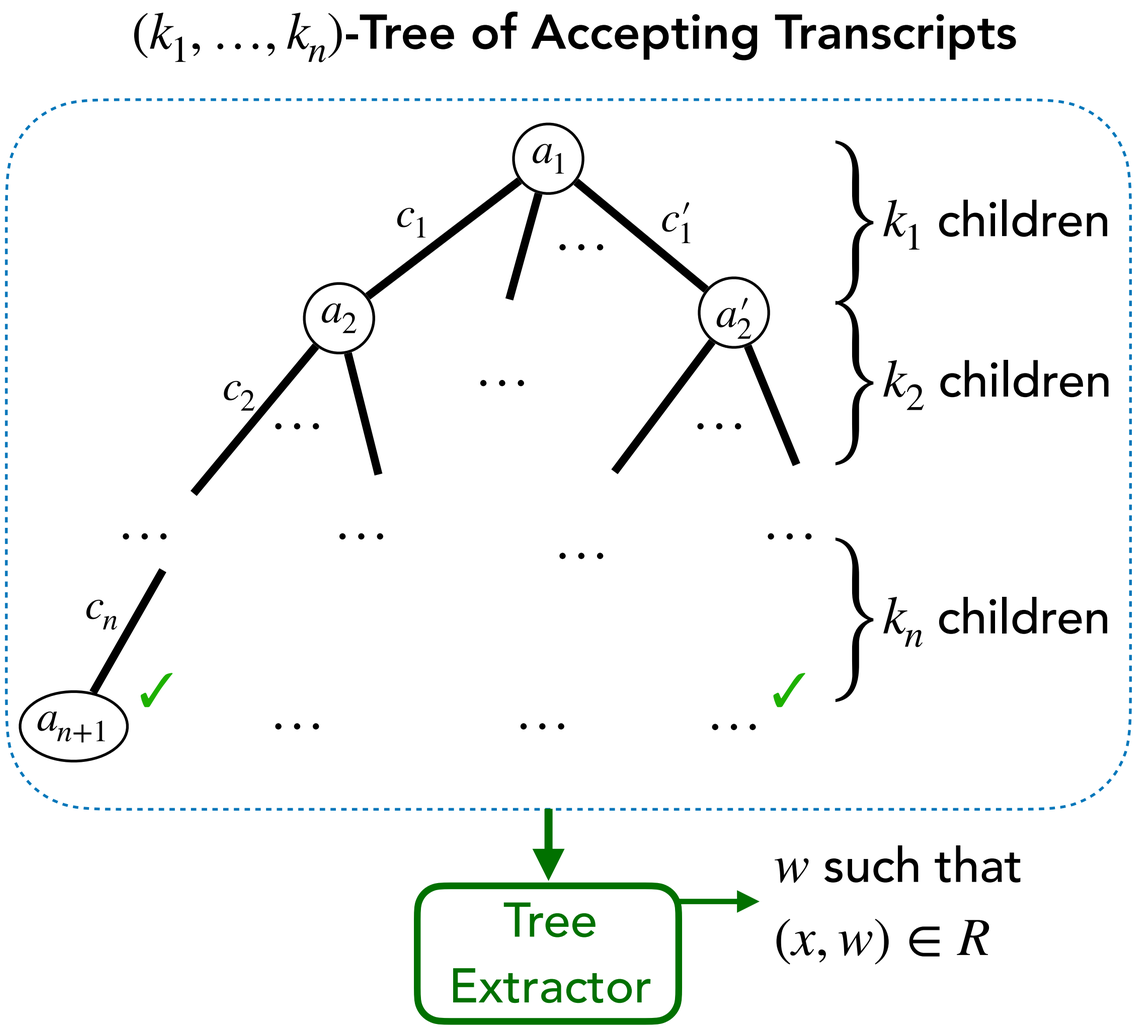


Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).

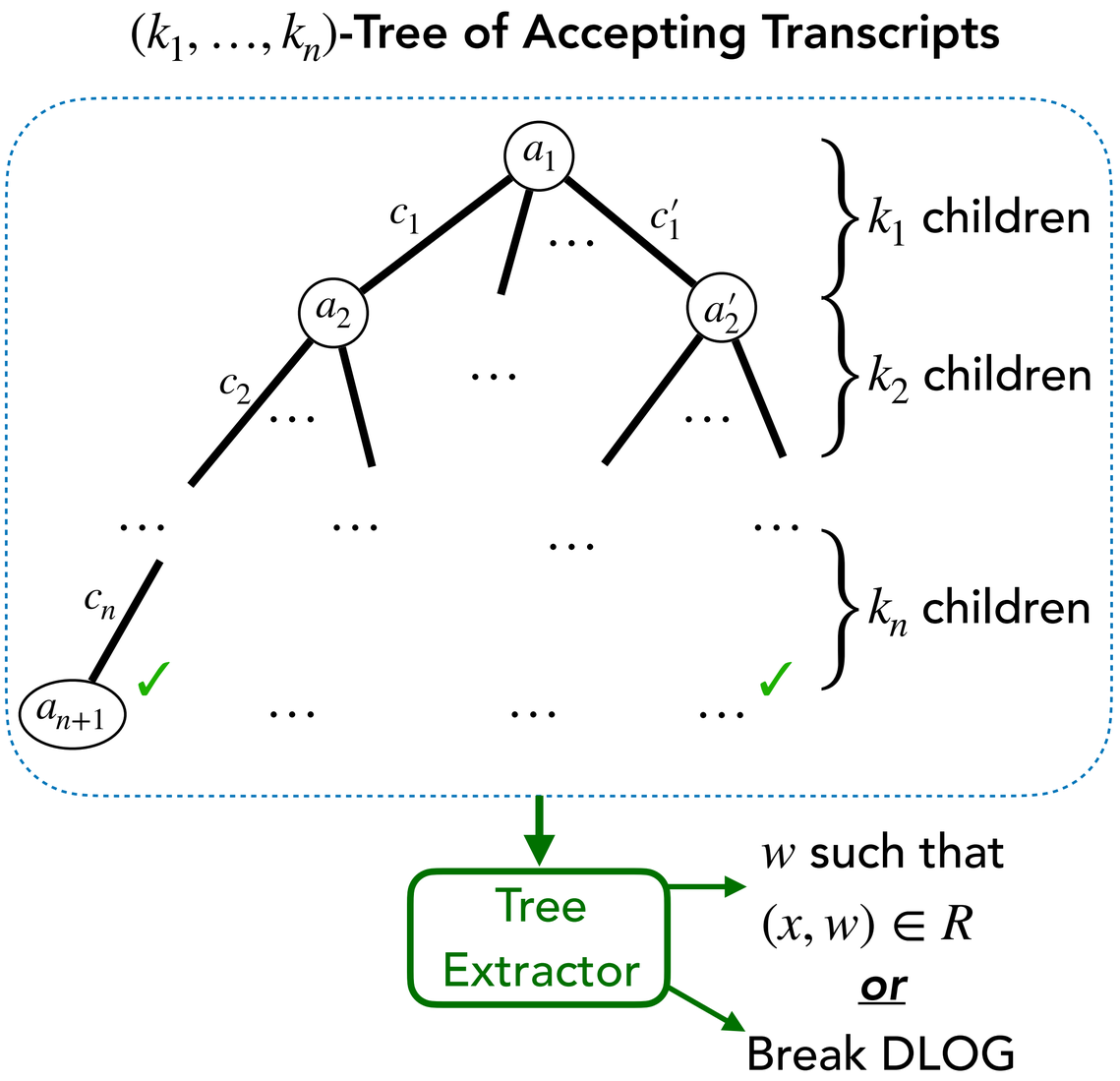


Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).

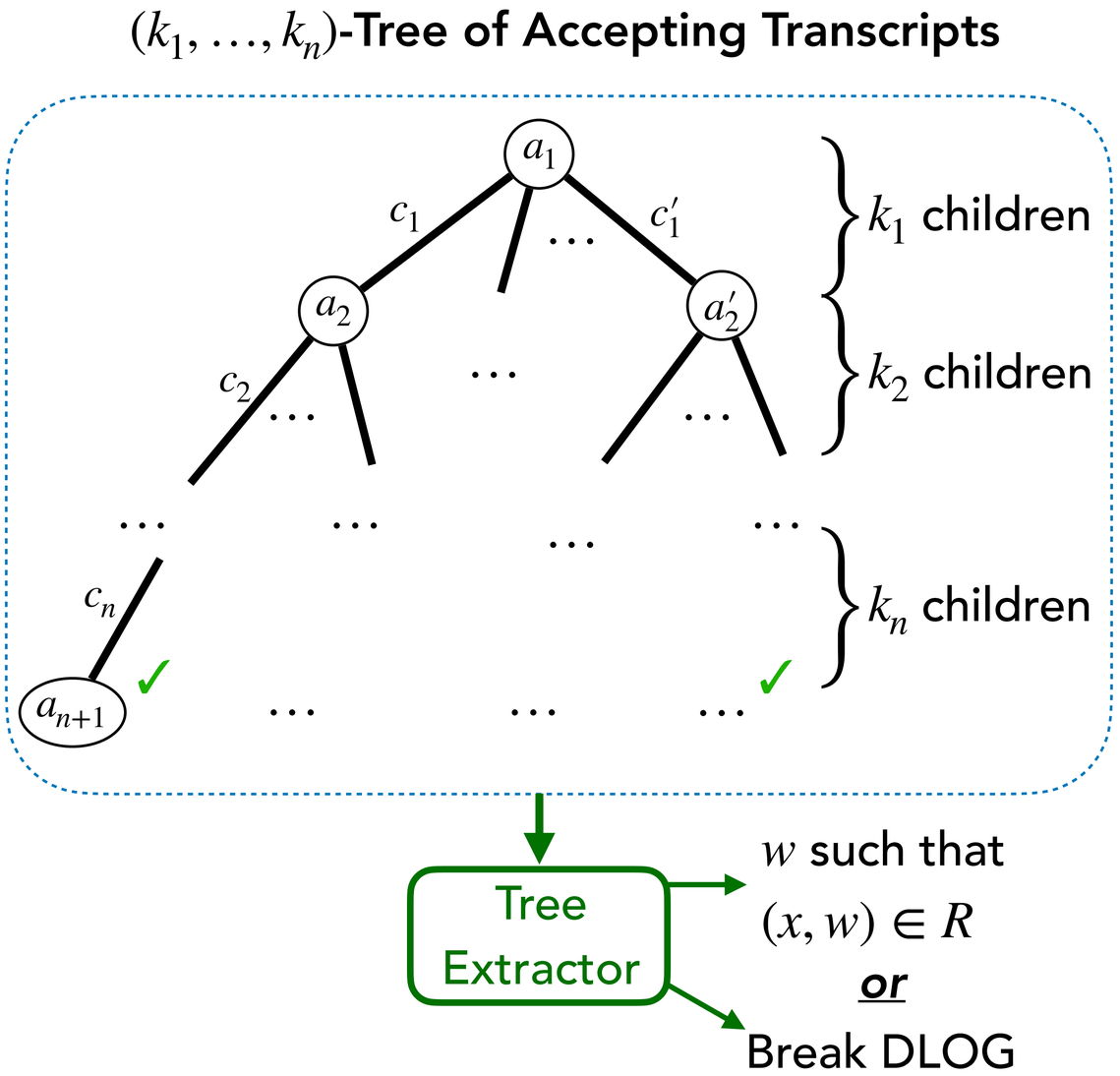


Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).
- The tree of transcripts needs to satisfy *extra* predicates on the challenges at certain levels.

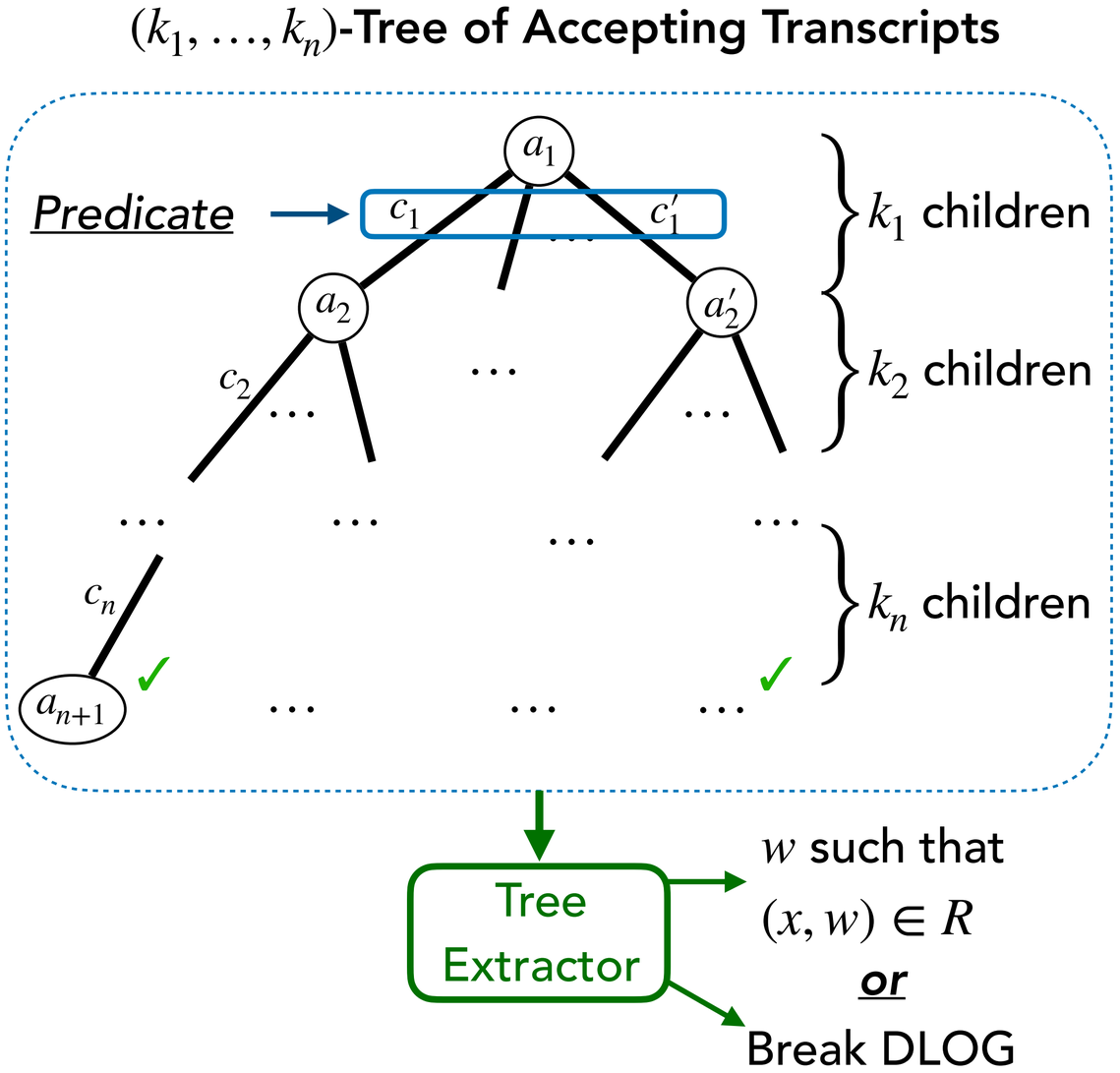


Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).
- The tree of transcripts needs to satisfy *extra* predicates on the challenges at certain levels.



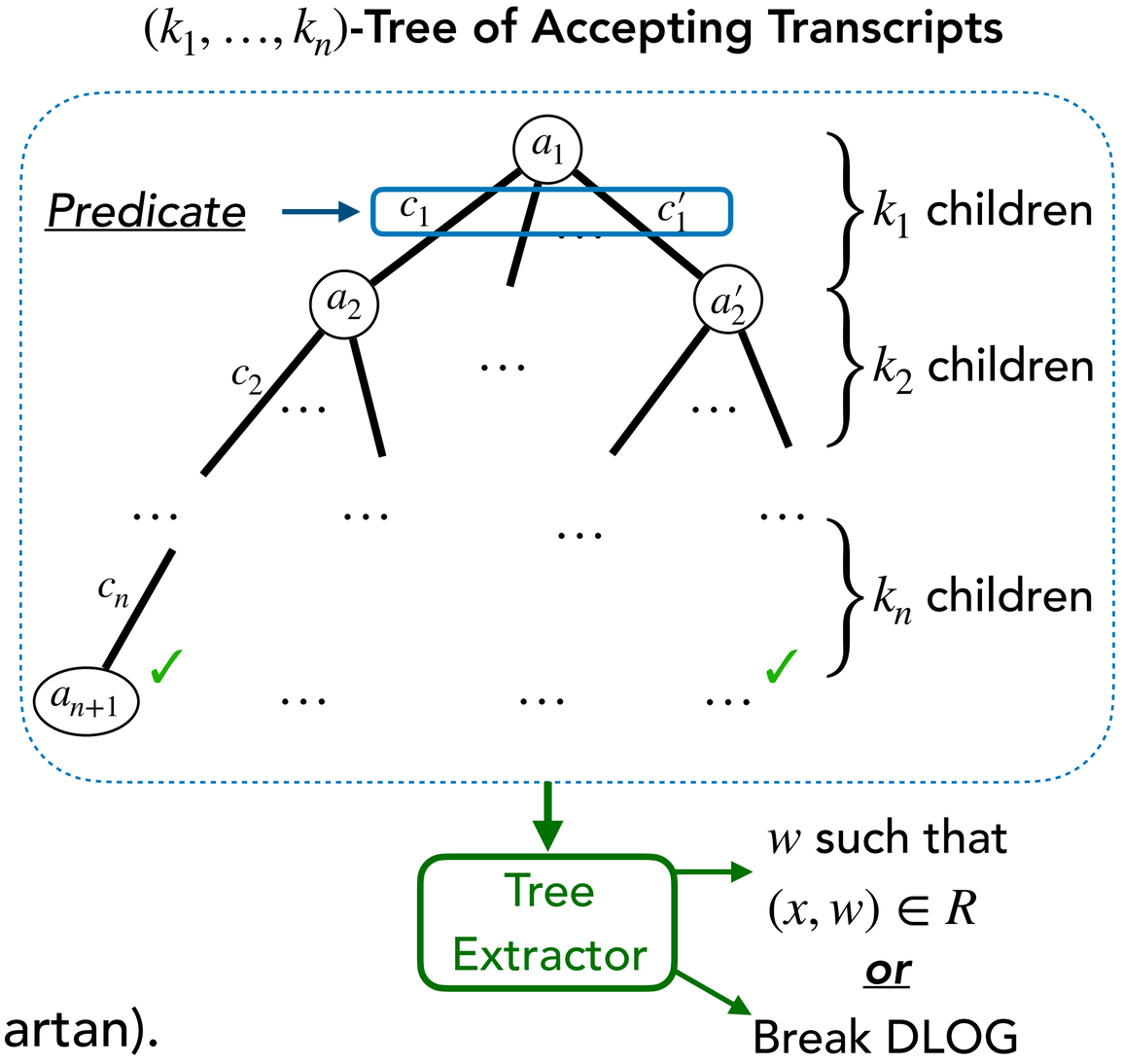
Generalized Special Soundness & Tree Building

Observation: Spartan and Bulletproofs do *not* satisfy special soundness.

However, they satisfy a *generalized* notion:

- Tree extraction can either output a witness or a break of some *computational* assumption (DLOG).
- The tree of transcripts needs to satisfy *extra* predicates on the challenges at certain levels.

We construct a *generalized tree builder* that can handle these predicates (for Bulletproofs and Spartan).



Summary

Summary

We show that Bulletproofs and Spartan satisfies SIM-EXT, a **strong** security notion for zkSNARKs that rules out most attacks in practice.

Summary

We show that Bulletproofs and Spartan satisfies SIM-EXT, a **strong** security notion for zkSNARKs that rules out most attacks in practice.

Limitation: bounds for knowledge soundness are *non-tight* due to rewinding

Summary

We show that Bulletproofs and Spartan satisfies SIM-EXT, a **strong** security notion for zkSNARKs that rules out most attacks in practice.

Limitation: bounds for knowledge soundness are *non-tight* due to rewinding

Open Questions:

- SIM-EXT for general classes of protocols:
  - Polynomial IOPs $\Longrightarrow$ [FFKRZ23]
  - Recursive SNARKs
- Tighter rewinding bounds
- UC security $\Longrightarrow$ [GKOPTT23]

Summary

We show that Bulletproofs and Spartan satisfies SIM-EXT, a **strong** security notion for zkSNARKs that rules out most attacks in practice.

Limitation: bounds for knowledge soundness are *non-tight* due to rewinding

Open Questions:

- SIM-EXT for general classes of protocols:
  - Polynomial IOPs $\Longrightarrow$ [FFKRZ23]
  - Recursive SNARKs
- Tighter rewinding bounds
- UC security $\Longrightarrow$ [GKOPTT23]

Read our paper!
(ePrint 2023/494)

Thank You!