Verifiable Secret Sharing from Symmetric Key Cryptography with Improved Optimistic Complexity

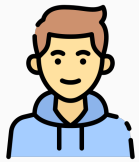
Ignacio Cascudo[1] Daniele Cozzo[1] Emanuele Giunta[1,2]

IMDEA Software Institute, Madrid, Spain.
name.surname@imdea.org

Universidad Politecnica de Madrid, Madrid, Spain.

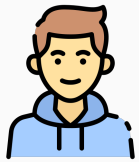
Secret Sharing

dealer

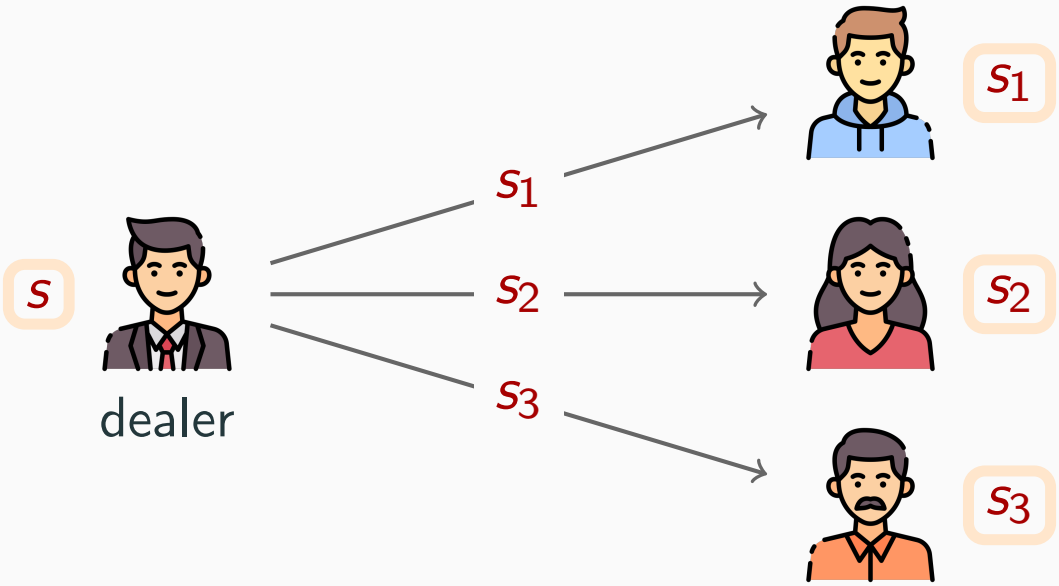


Secret Sharing

dealer



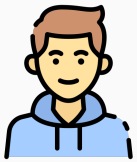
Secret Sharing



Secret Sharing

s

dealer



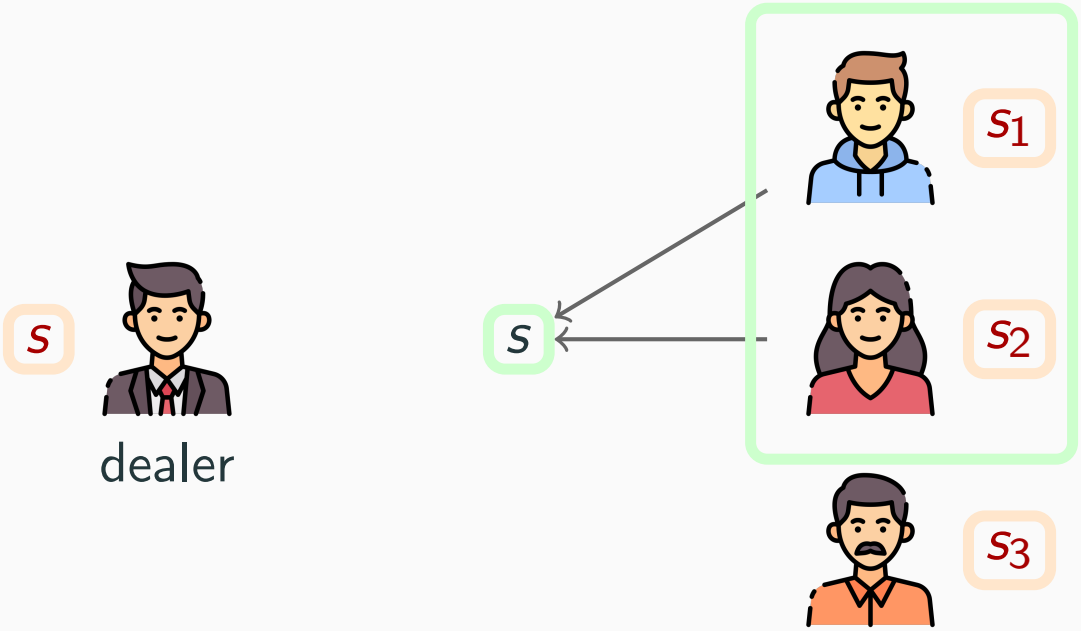
s_1

s_2

s_3

t privacy

Secret Sharing



t privacy

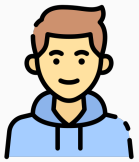
$t + 1$ reconstruction

Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.

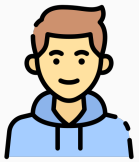
$s \in \mathbb{F}$

dealer



Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.



$s \in \mathbb{F}$

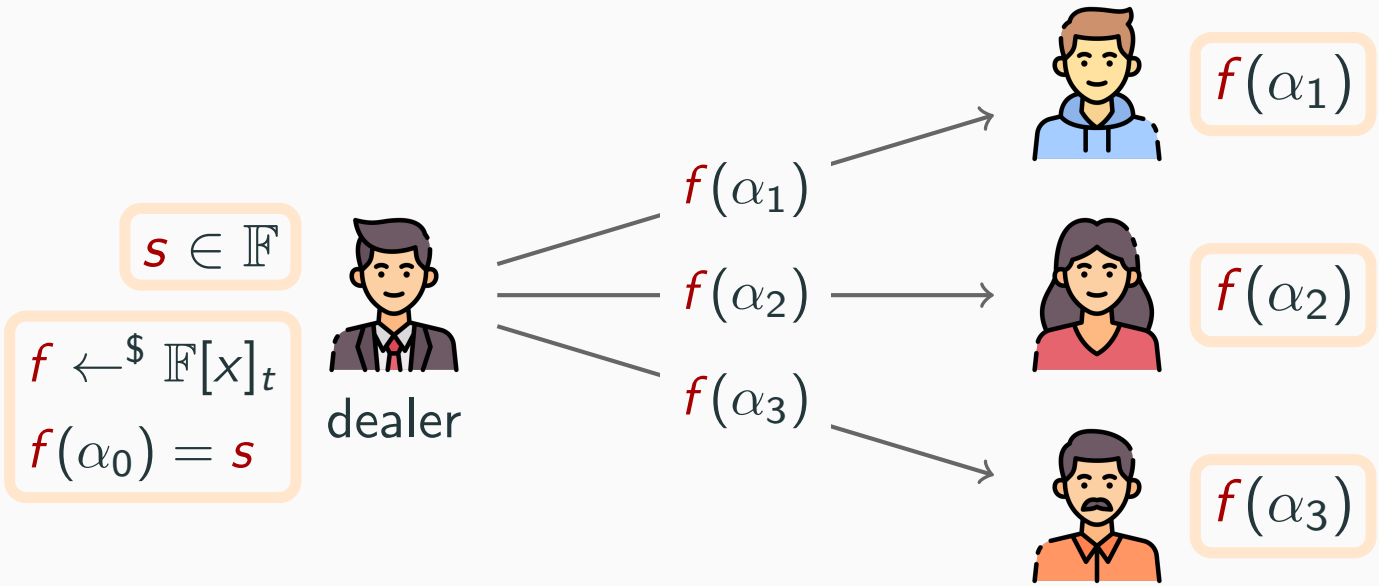
$f \leftarrow^{\$} \mathbb{F}[x]_t$

$f(\alpha_0) = s$

dealer

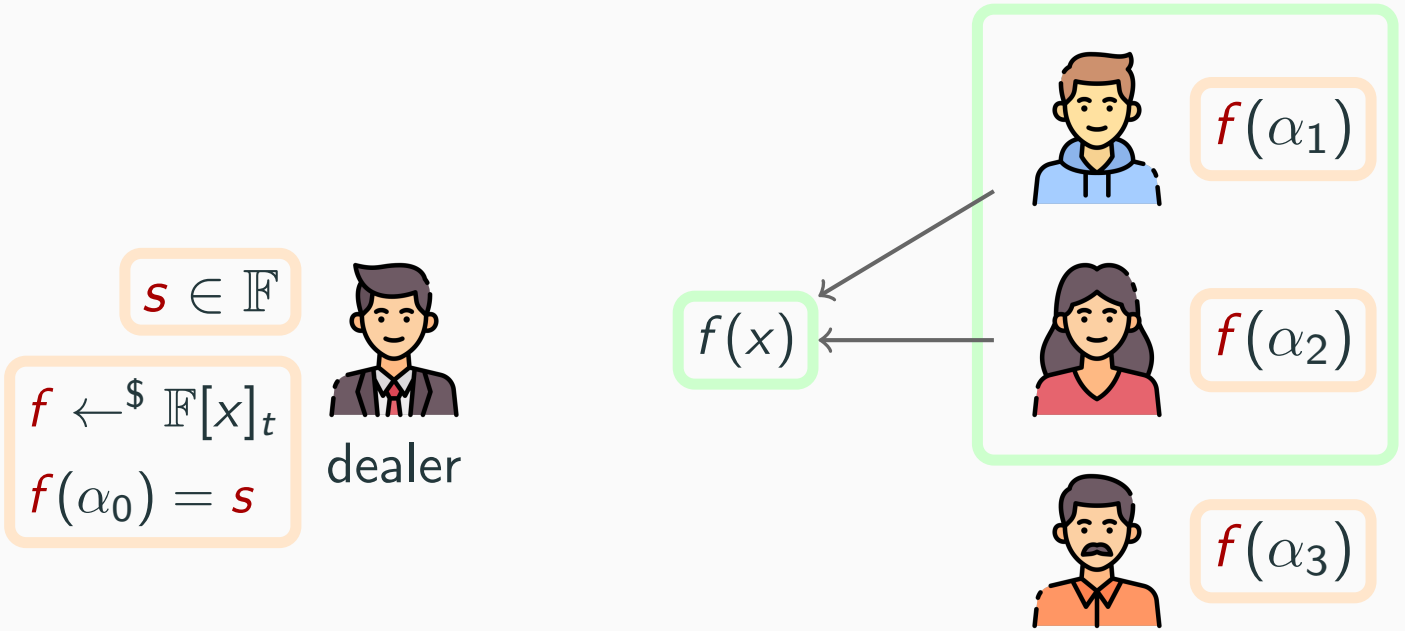
Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.



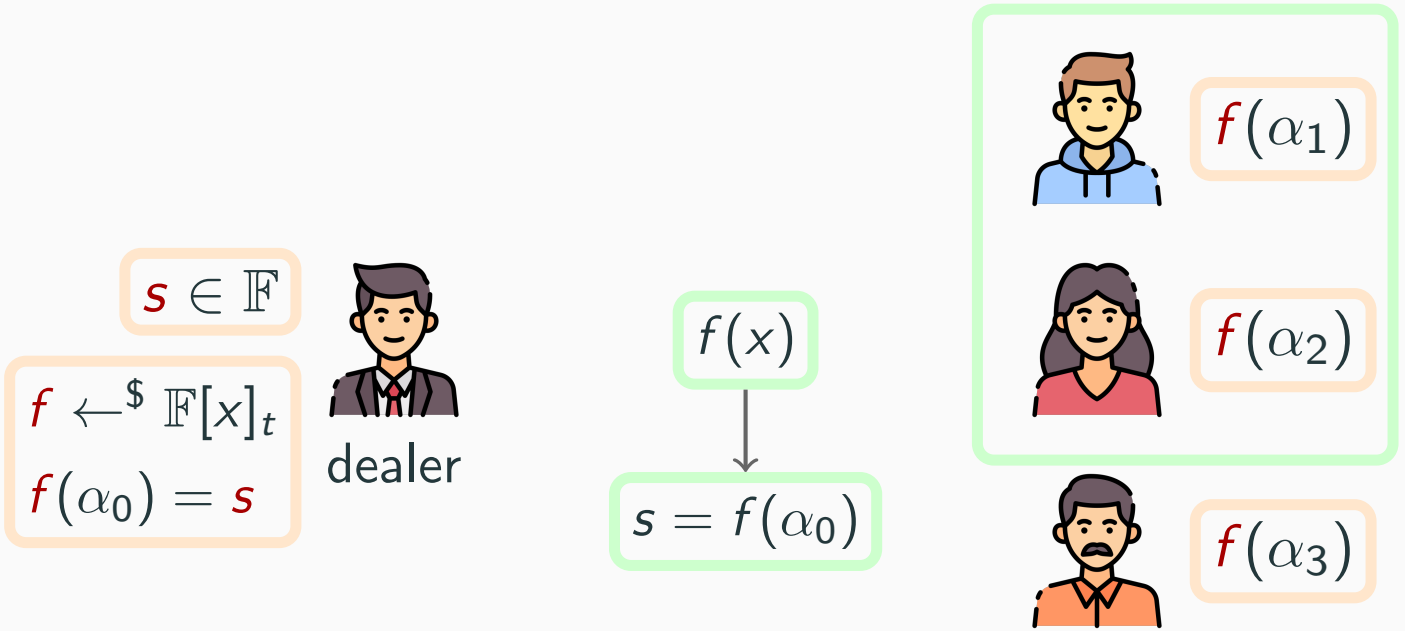
Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.



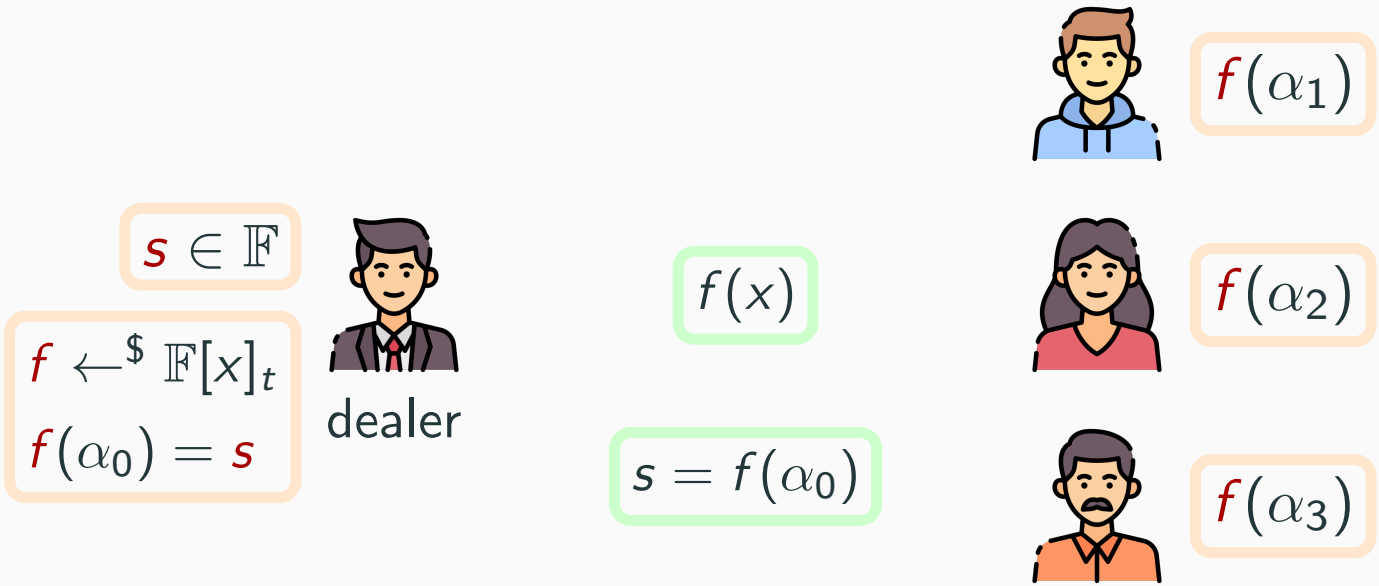
Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.



Shamir Secret Sharing

$s \in \mathbb{F}$ and let $\alpha_0, \alpha_1, \ldots, \alpha_n \in \mathbb{F}$ be distinct points.



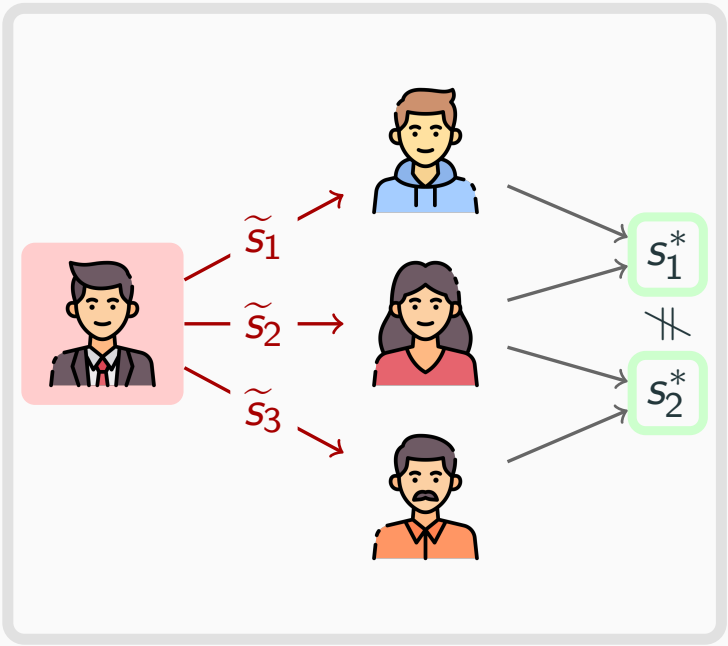
Works over any ring R if $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$ is an **exceptional set**, i.e. $\alpha_i - \alpha_j$ is invertible.

Limitations of Secret Sharing

Secret Sharing in general is **not binding** toward the secret:

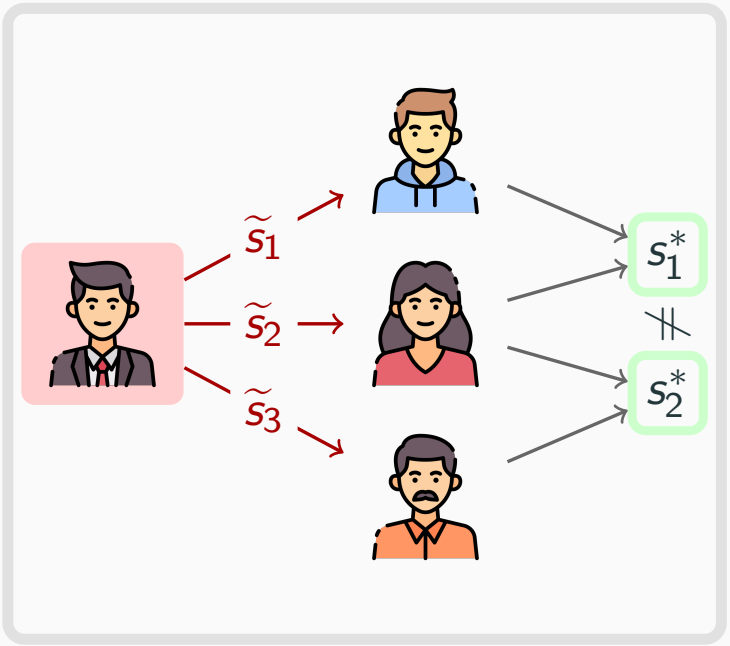
Limitations of Secret Sharing

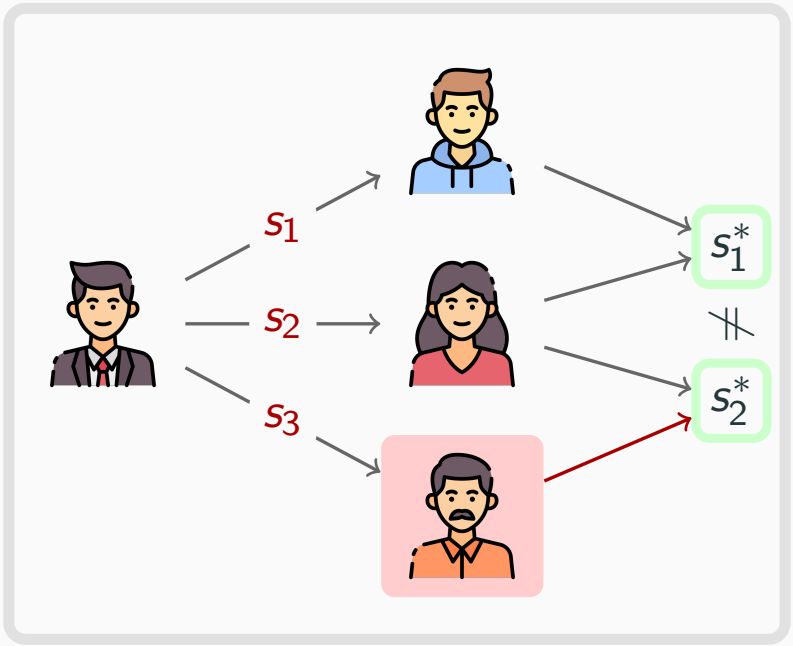
Secret Sharing in general is **not binding** toward the secret:



Limitations of Secret Sharing

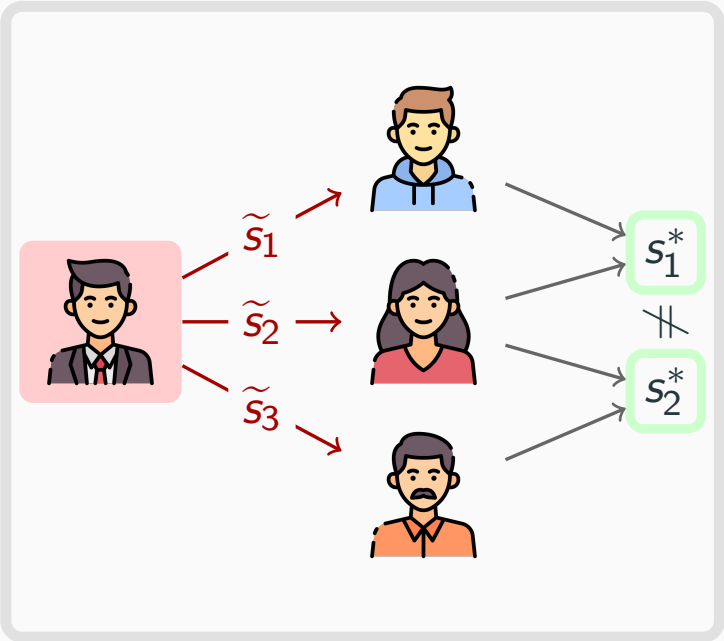
Secret Sharing in general is **not binding** toward the secret:

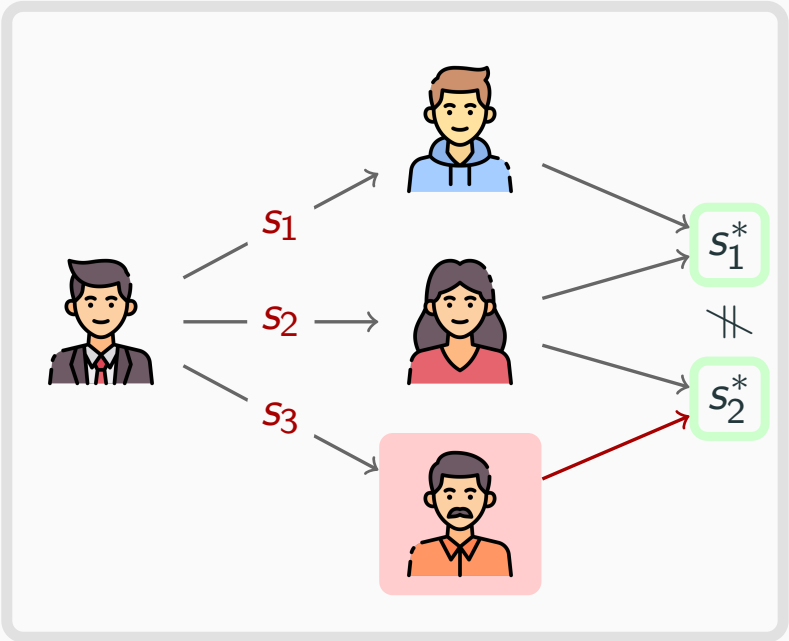




Limitations of Secret Sharing

Secret Sharing in general is **not binding** toward the secret:





Issue for applications such as **Distributed Key Generation**.

Verifiable Secret Sharing

Defined by two protocols (`Share`, `Reconstruct`) satisfying:

Verifiable Secret Sharing

Defined by two protocols (`Share`, `Reconstruct`) satisfying:

- **Privacy:** After `Share`, any set of t parties has no information on the shared secret s.

Verifiable Secret Sharing

Defined by two protocols (`Share`, `Reconstruct`) satisfying:

- **Privacy:** After `Share`, any set of t parties has no information on the shared secret s.
- **Commitment:** After `Share` there exists a unique s such that honest users in `Reconstruct` obtain s against t corruptions.

Verifiable Secret Sharing

Defined by two protocols (`Share`, `Reconstruct`) satisfying:

- **Privacy:** After `Share`, any set of t parties has no information on the shared secret s.
- **Commitment:** After `Share` there exists a unique s such that honest users in `Reconstruct` obtain s against t corruptions.
- **Strong Commitment:** As before, but after `Share` honest users also get private shares consistent with s.

Previous Work

With **honest majority** ($n \geq 2t + 1$) and **Synchronous Communication**:

Previous Work

With **honest majority** ($n \geq 2t + 1$) and **Synchronous Communication**:

Statistical Security
[BCW88, RB89, . . .]
- Arithmetic Operations Only
- High Communication: $\Omega(n^2)$
- Information-Theoretic Security

Previous Work

With **honest majority** ($n \geq 2t + 1$) and **Synchronous Communication**:

Statistical Security
[BCW88, RB89, . . .]
- Arithmetic Operations Only
- High Communication: $\Omega(n^2)$
- Information-Theoretic Security

PKE-Based
[Fel87, Ped92, Sch99, . . .]
- Expensive PKE operations
- Low Communication: $O(n)$
- Not Post-Quantum

Previous Work

With **honest majority** ($n \geq 2t + 1$) and **Synchronous Communication**:

Statistical Security
[BCW88, RB89, . . .]
- Arithmetic Operations Only
- High Communication: $\Omega(n^2)$
- Information-Theoretic Security

PKE-Based
[Fel87, Ped92, Sch99, . . .]
- Expensive PKE operations
- Low Communication: $O(n)$
- Not Post-Quantum

SKE/ROM-Based
[GRR99, BKP11, ABCP23]
- Cheaper SKE operations
- Plausibly Post-Quantum

Previous Work

With **honest majority** ($n \geq 2t + 1$) and **Synchronous Communication**:

Statistical Security
[BCW88, RB89, . . .]
- Arithmetic Operations Only
- High Communication: $\Omega(n^2)$
- Information-Theoretic Security

PKE-Based
[Fel87, Ped92, Sch99, . . .]
- Expensive PKE operations
- Low Communication: $O(n)$
- Not Post-Quantum

SKE/ROM-Based
[GRR99, BKP11, ABCP23]
- Cheaper SKE operations
- Plausibly Post-Quantum

Our Results

	[ABCP23]
Dealer Computation	$O(n \log n)$
Dealer Upload	$O(n)$
Worst Case:	
Verifier Computation	$O(n)$
Verifier Download	$O(n)$
ϑ **Active Corruptions:**	
Verifier Computation	$O(n)$
Verifier Download	$O(n)$

Our Results

	[ABCP23]
Dealer Computation	$O(n \log n)$
Dealer Upload	$O(n)$
Worst Case:	
Verifier Computation	$O(n)$
Verifier Download	$O(n)$
ϑ **Active Corruptions:**	
Verifier Computation	$O(n)$
Verifier Download	$O(n)$

Our Results

	[ABCP23]	Our Work
Dealer Computation	$O(n \log n)$	$O(n \log n)$
Dealer Upload	$O(n)$	$O(n(\log n)^2)$
Worst Case:		
Verifier Computation	$O(n)$	$O(n)$
Verifier Download	$O(n)$	$O(n)$
ϑ **Active Corruptions:**		
Verifier Computation	$O(n)$	$O(\vartheta \log(n)^2)$
Verifier Download	$O(n)$	$O(\vartheta \log(n)^2)$

Our Results

	[ABCP23]	Our Work
Dealer Computation	$O(n \log n)$	$O(n \log n)$
Dealer Upload	$O(n)$	$O(n(\log n)^2)$
Worst Case:		
Verifier Computation	$O(n)$	$O(n)$
Verifier Download	$O(n)$	$O(n)$
ϑ **Active Corruptions:**		
Verifier Computation	$O(n)$	$O(\vartheta \log(n)^2)$
Verifier Download	$O(n)$	$O(\vartheta \log(n)^2)$

Our Results

	[ABCP23]	Our Work
Dealer Computation	$O(n \log n)$	$O(n \log n)$
Dealer Upload	$O(n)$	$O(n(\log n)^2)$
Worst Case:		
Verifier Computation	$O(n)$	$O(n)$
Verifier Download	$O(n)$	$O(n)$
ϑ **Active Corruptions:**		
Verifier Computation	$O(n)$	$O(\vartheta \log(n)^2)$
Verifier Download	$O(n)$	$O(\vartheta \log(n)^2)$

[ABCP23]: Construction

s

dealer

verifier i

[ABCP23]: Construction

s

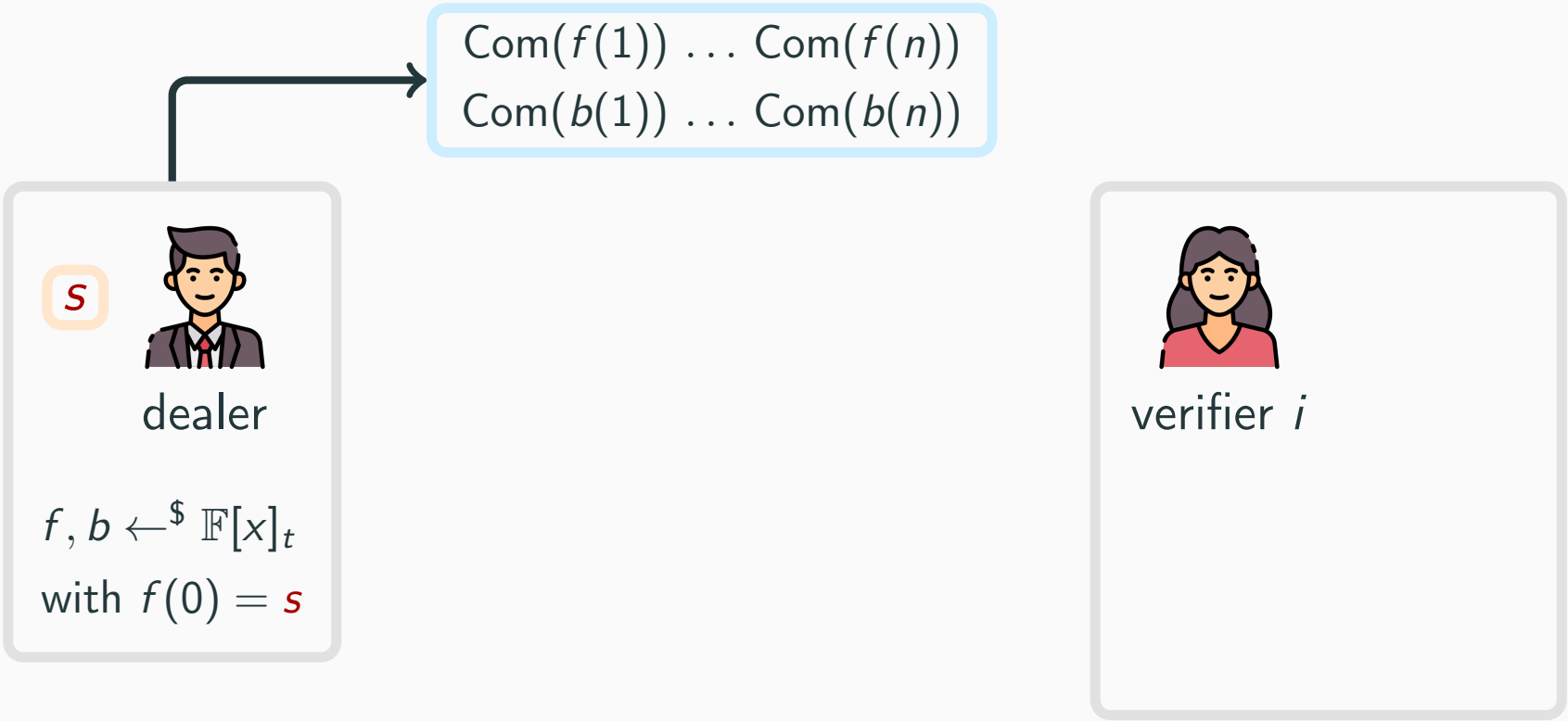
dealer

$f, b \leftarrow^{\$} \mathbb{F}[x]_t$

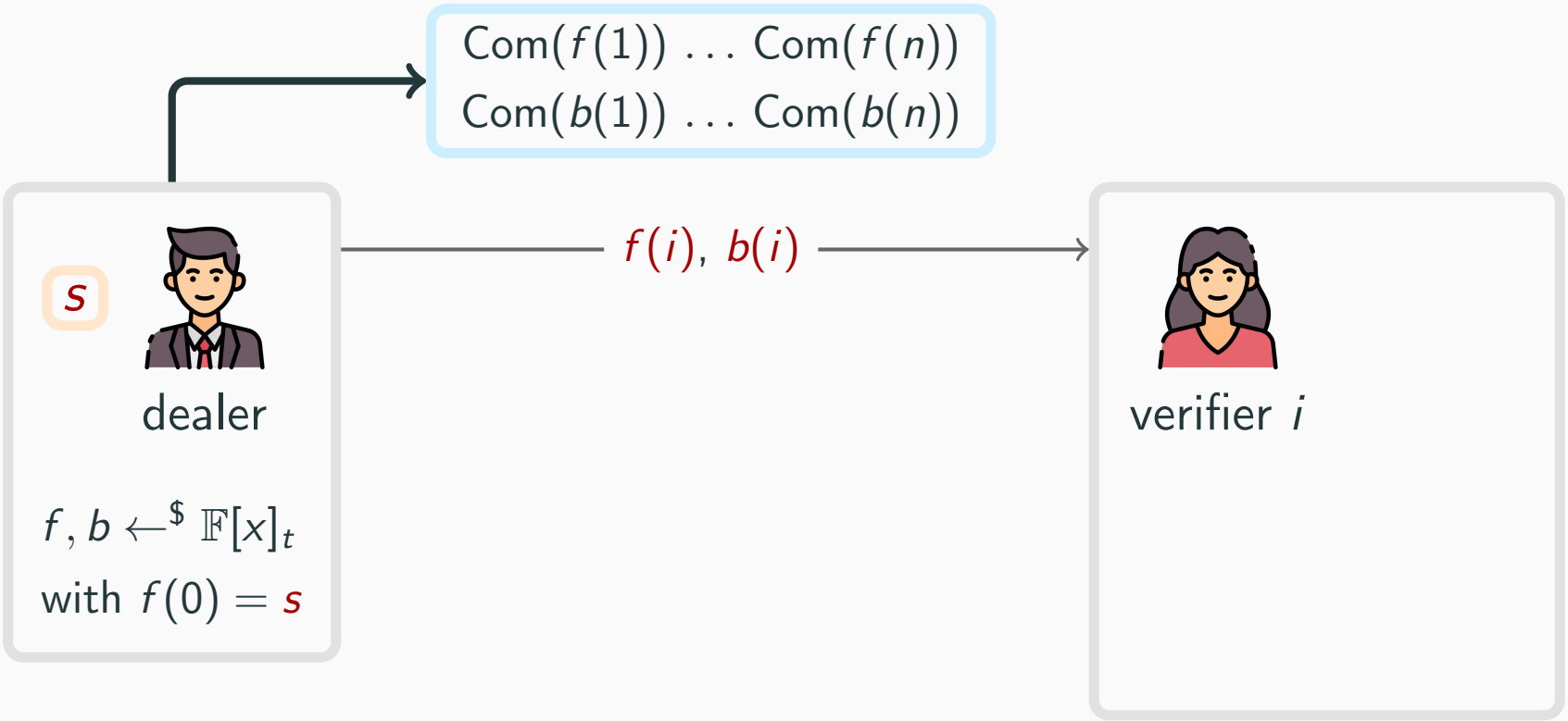
with $f(0) = s$

verifier i

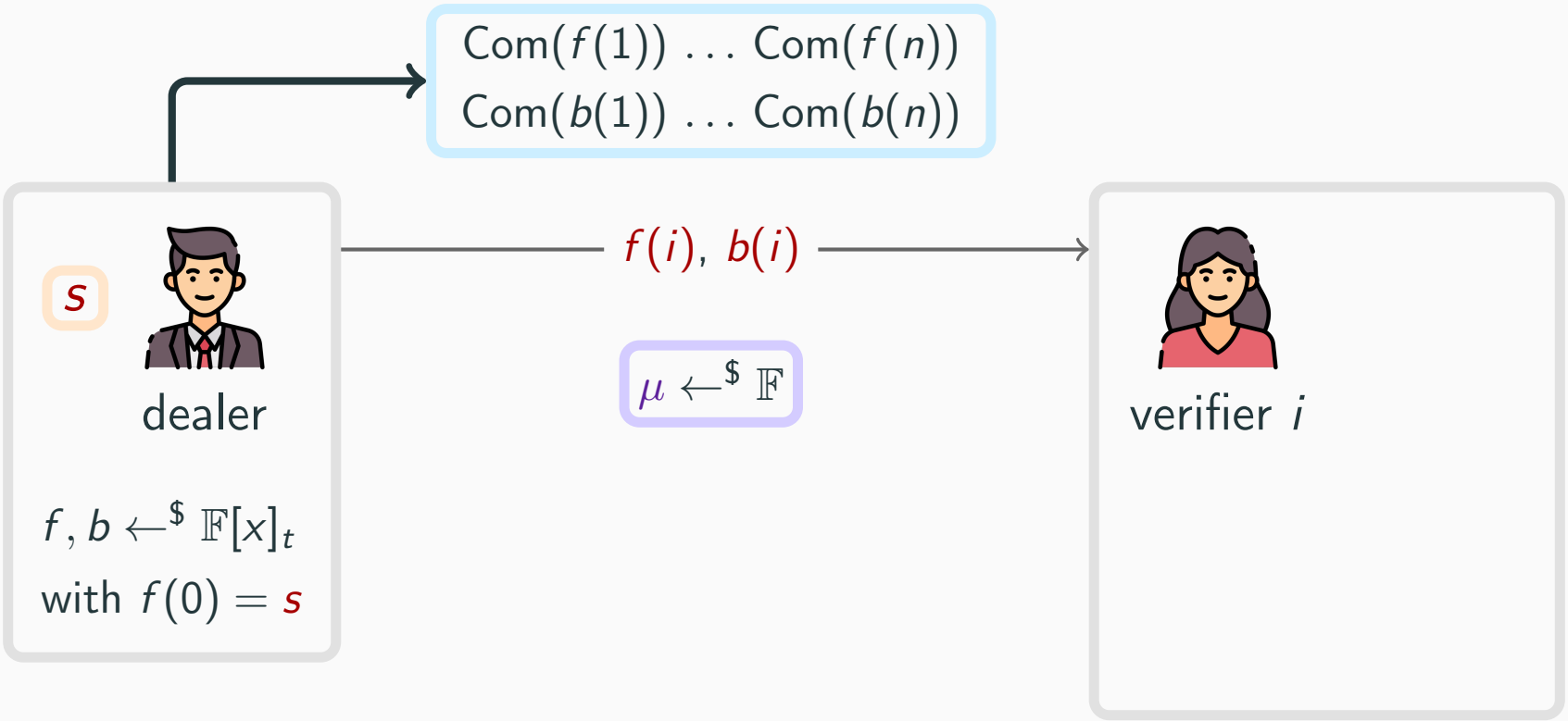
[ABCP23]: Construction



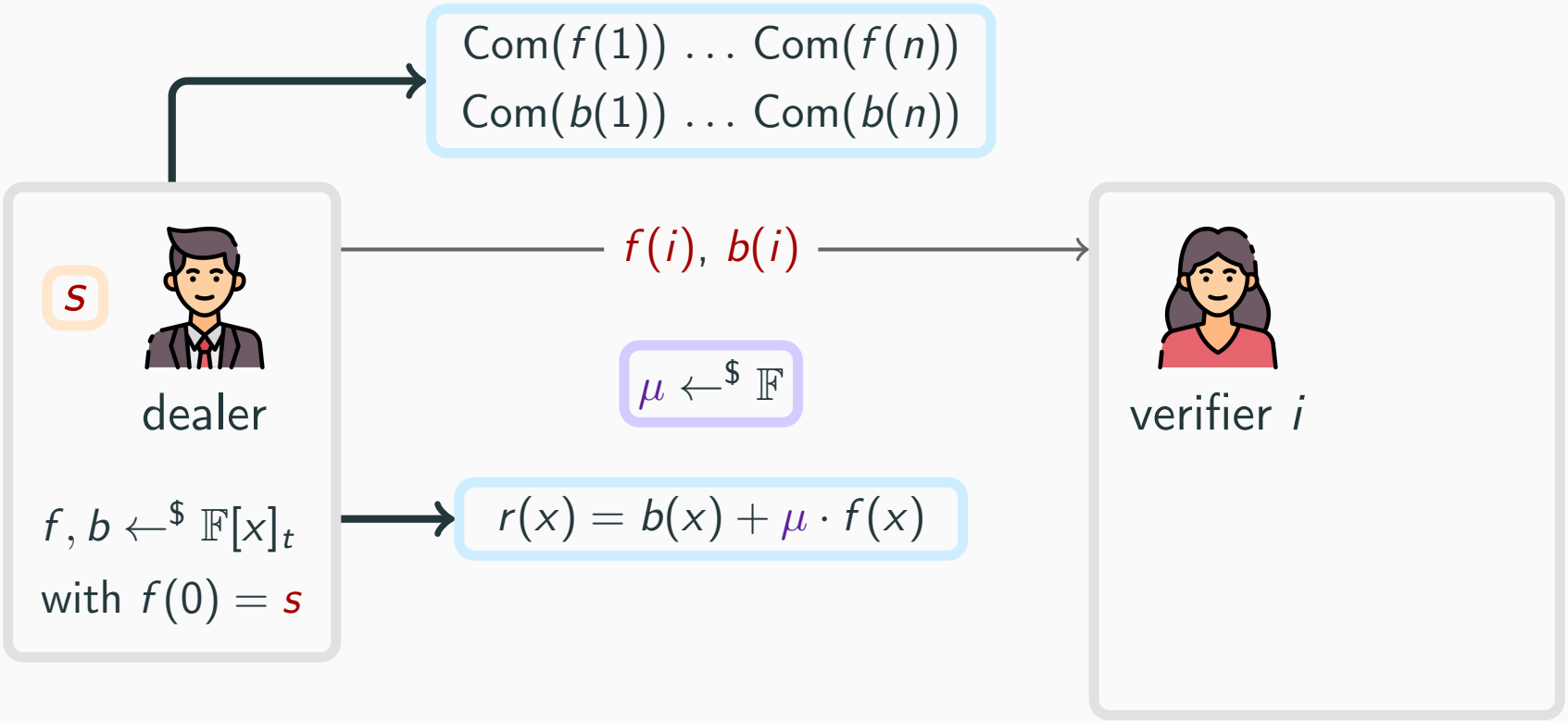
[ABCP23]: Construction



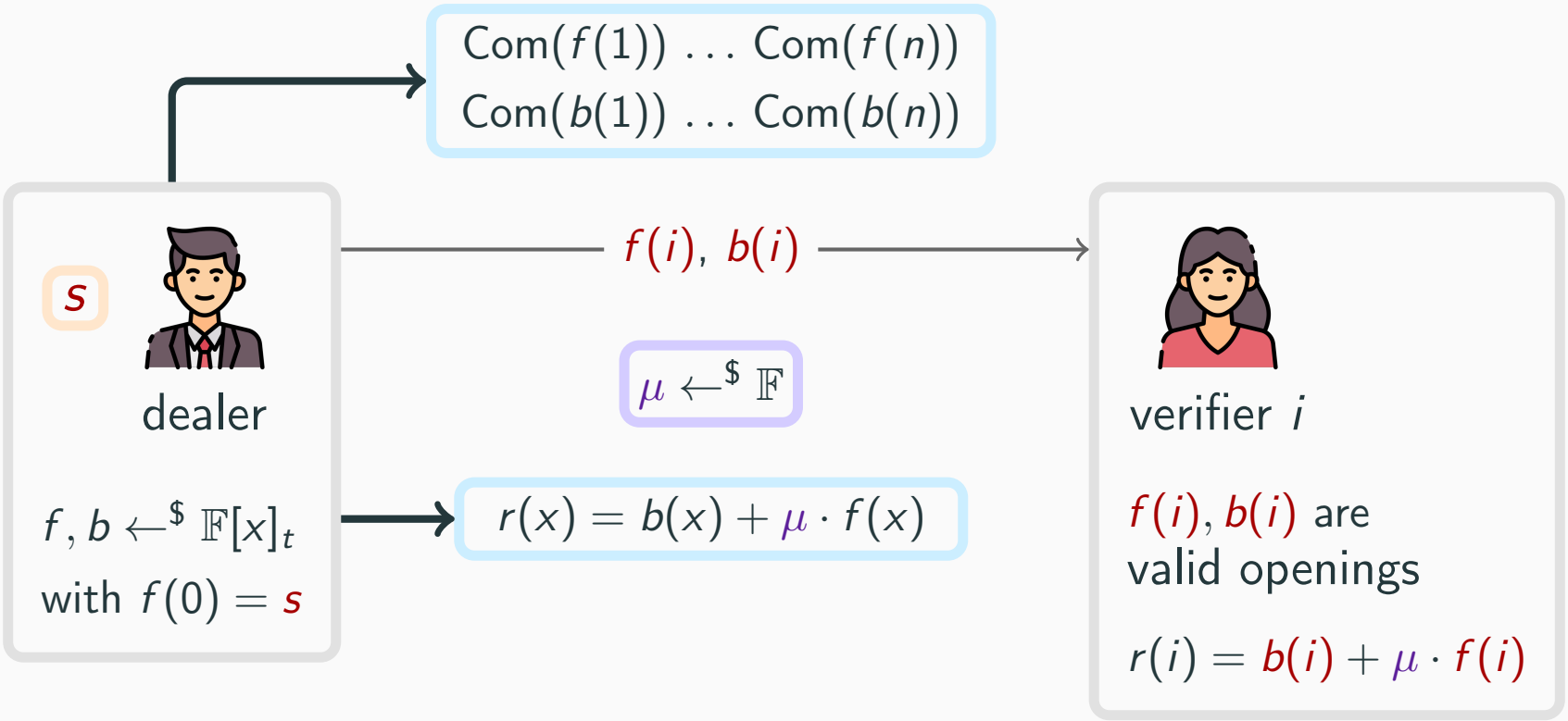
[ABCP23]: Construction



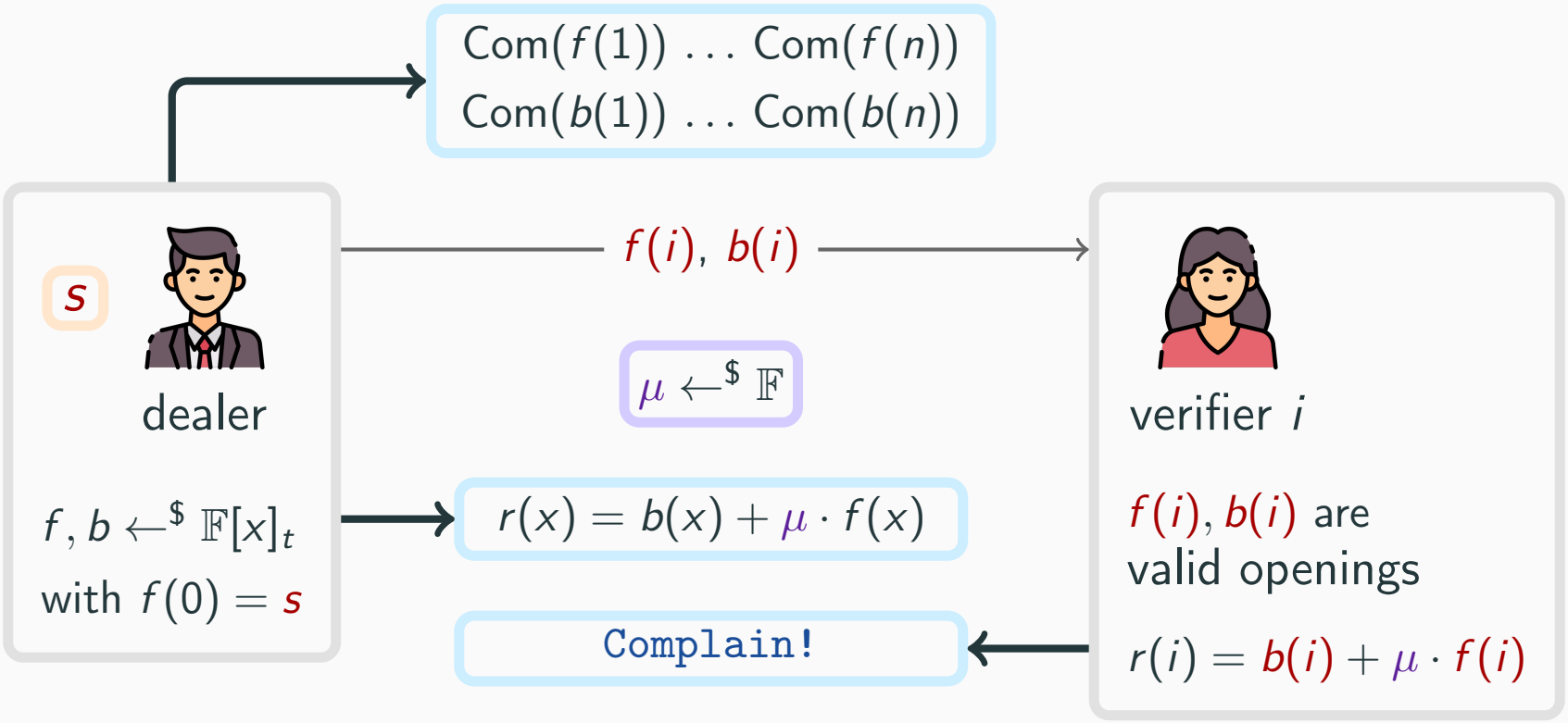
[ABCP23]: Construction



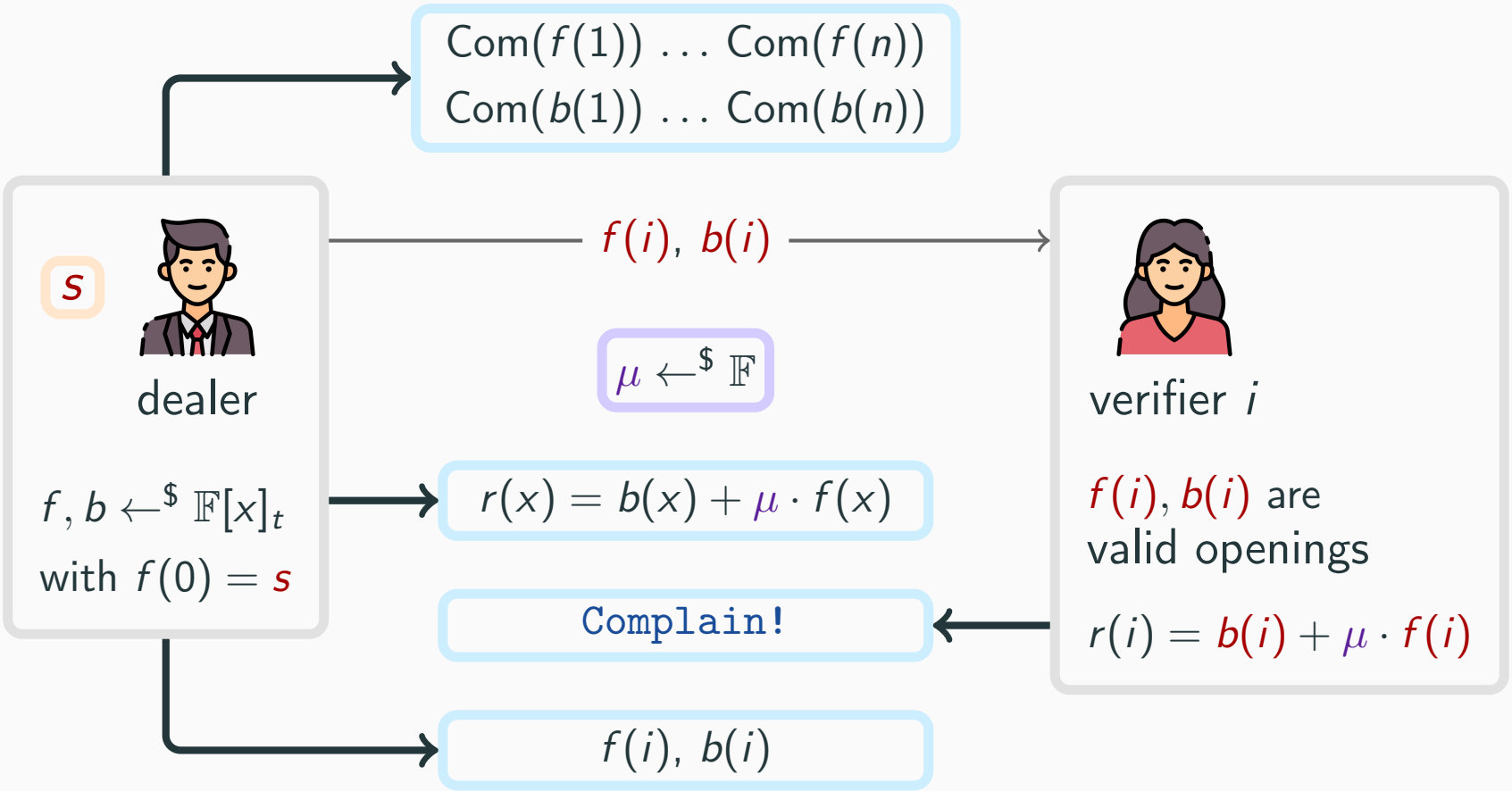
[ABCP23]: Construction



[ABCP23]: Construction



[ABCP23]: Construction



[ABCP23]: Zooming Out

Three Main Steps:

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b
2. **Low Degree Test** for $r = f + \mu b$

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b
2. **Low Degree Test** for $r = f + \mu b$
3. **Complain Phase**

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b $\Rightarrow$ Use a **Merkle Tree**
2. **Low Degree Test** for $r = f + \mu b$
3. **Complain Phase**

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b $\Rightarrow$ Use a **Merkle Tree**
2. **Low Degree Test** for $r = f + \mu b$ $\Rightarrow$ New **Distributed Proof**
3. **Complain Phase**

[ABCP23]: Zooming Out

Three Main Steps:

1. **Commitment** to f and b $\Rightarrow$ Use a **Merkle Tree**
2. **Low Degree Test** for $r = f + \mu b$ $\Rightarrow$ New **Distributed Proof**
3. **Complain Phase** $\Rightarrow$ Use MT **Subvector Opening**

[ABCP23]: Zooming Out

Three Main Steps:

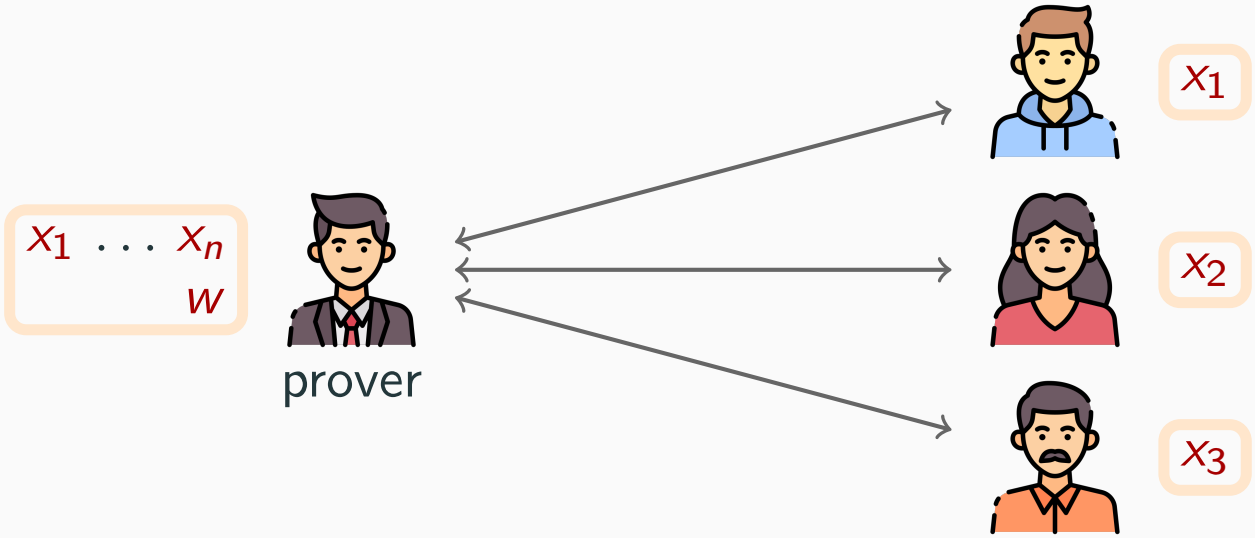
1. **Commitment** to f and b $\Rightarrow$ Use a **Merkle Tree**
2. **Low Degree Test** for $r = f + \mu b$ $\Rightarrow$ New **Distributed Proof**
3. **Complain Phase** $\Rightarrow$ Use MT **Subvector Opening**

Distributed Proofs [BBC+19]

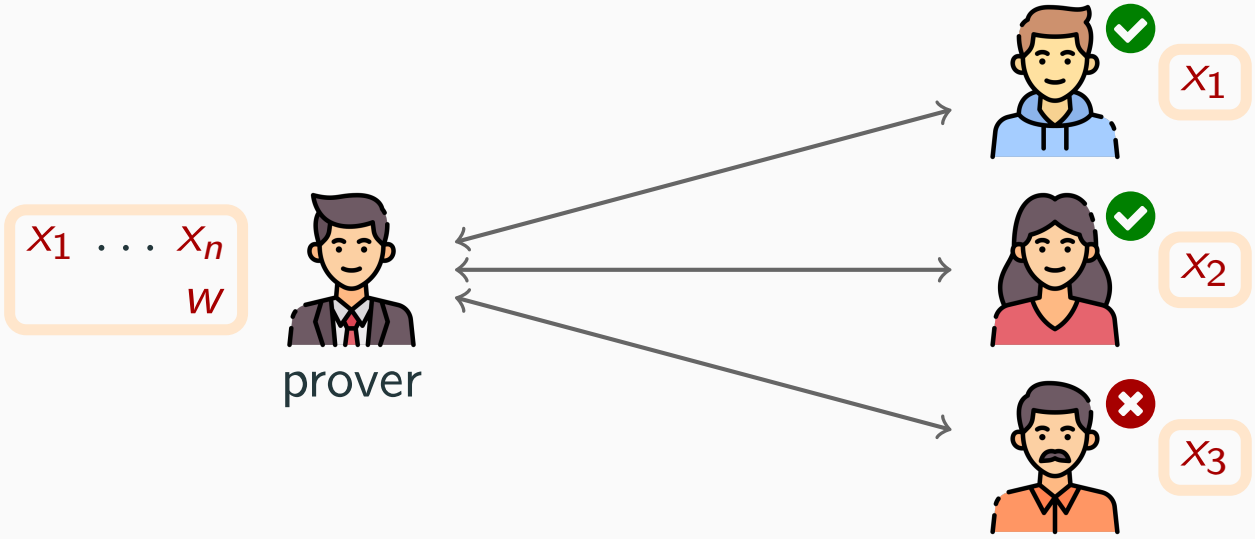
prover

Distributed Proofs [BBC+19]

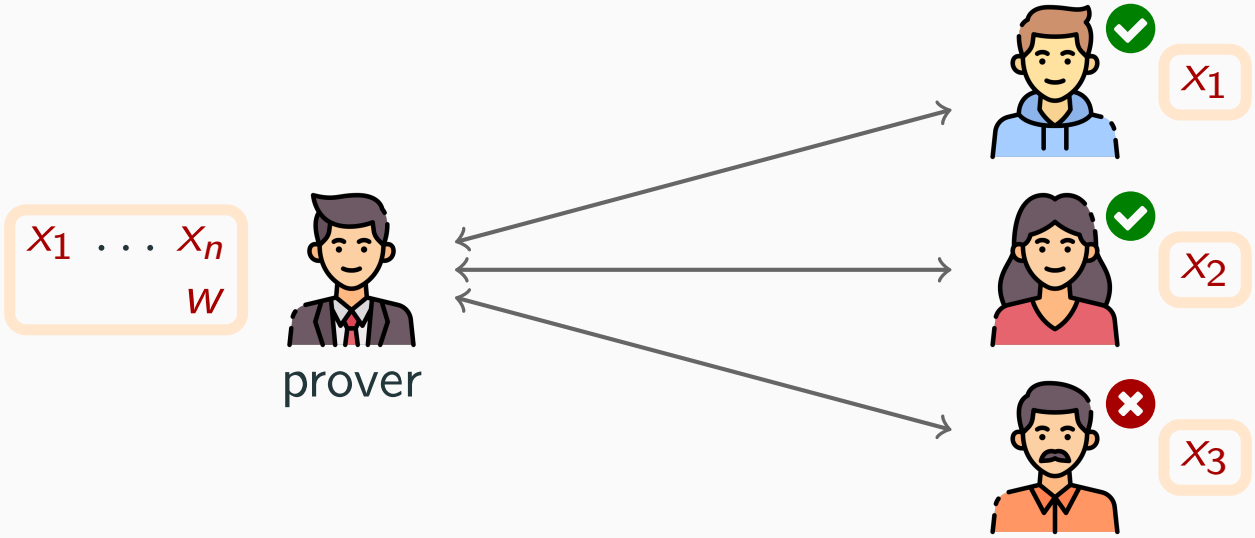
Distributed Proofs [BBC+19]



Distributed Proofs [BBC+19]

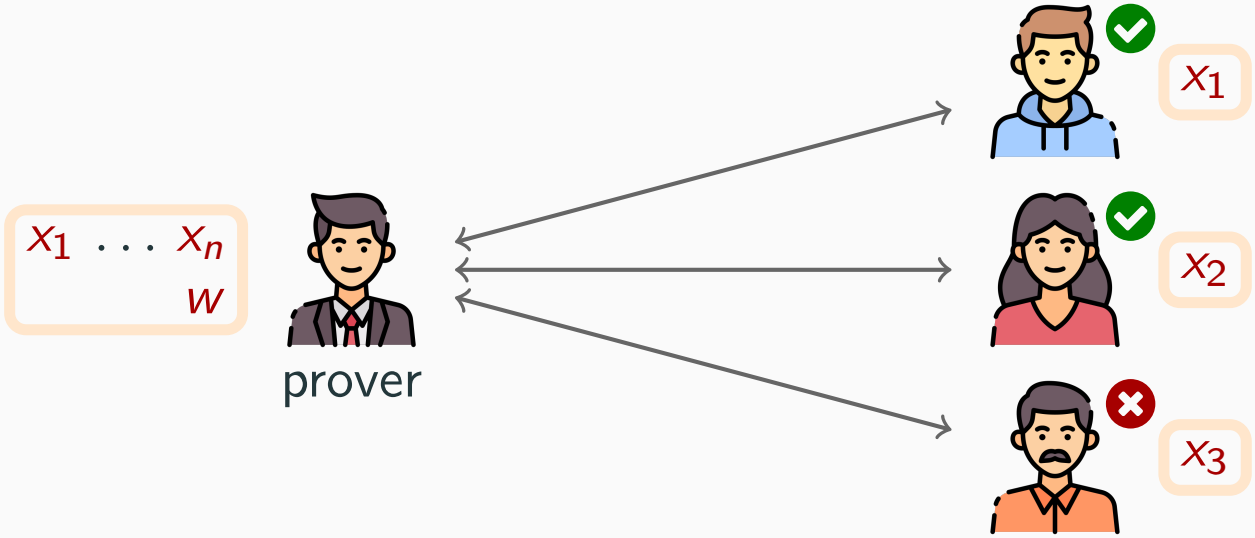


Distributed Proofs [BBC+19]



Correctness: If $(x_1 \ldots x_n, w) \in \mathcal{R}$ all honest verifiers accept

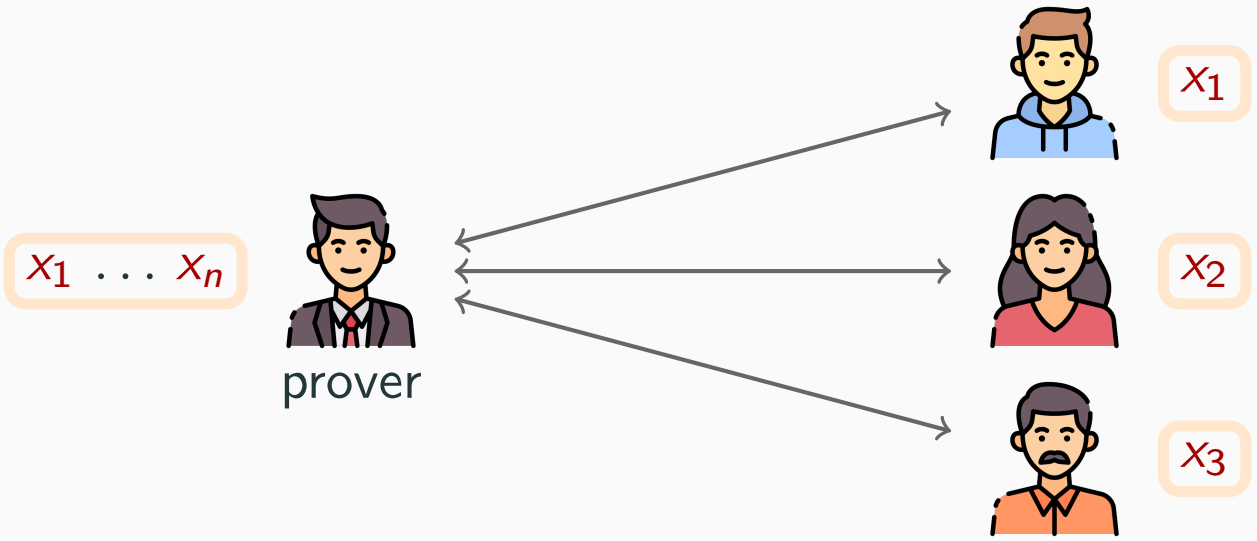
Distributed Proofs [BBC+19]



Correctness: If $(x_1 \ldots x_n, w) \in \mathcal{R}$ all honest verifiers accept

Soundness*: If there exists no z, w with $z_i = x_i$ for all honest V_i and $(z, w) \in \mathcal{R}$, **at least one** honest verifier rejects w.h.p.

Distributed Proofs: Low Degree



Relation: $(x_1 \ldots x_n) \in \mathcal{R}_d$ if there exists $f \in \mathbb{F}[x]$ with $\deg(f) \leq d$ and $f(\alpha_i) = x_i$

Low Degree Proof: Direct Test [ABCP23]

$x_1 \ \ldots \ x_n$

prover

x_i

i-th verifier

Low Degree Proof: Direct Test [ABCP23]

$x_1 \ \ldots \ x_n$

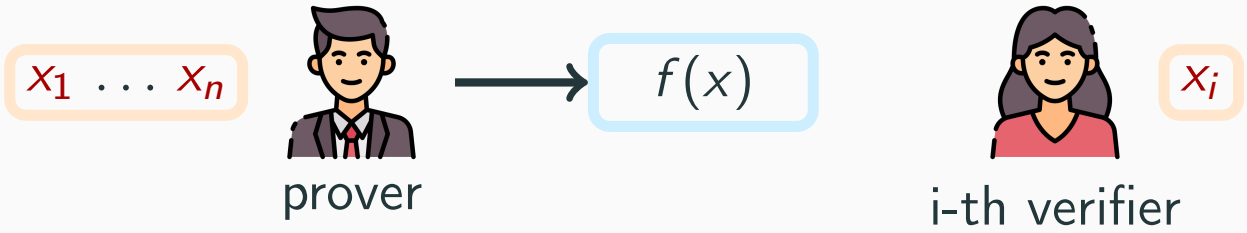
prover

x_i

i-th verifier

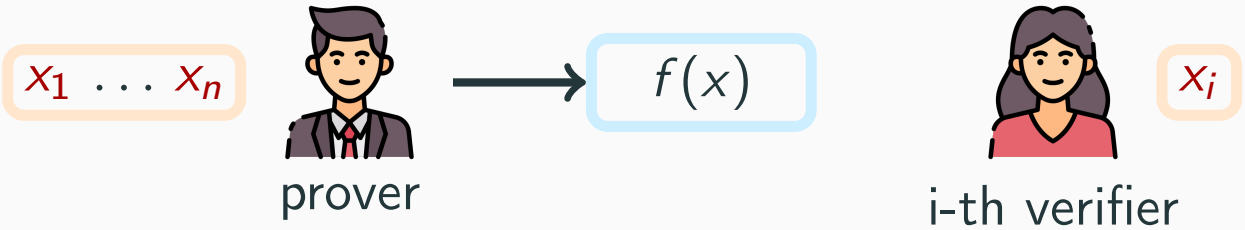
$$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$$

Low Degree Proof: Direct Test [ABCP23]



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

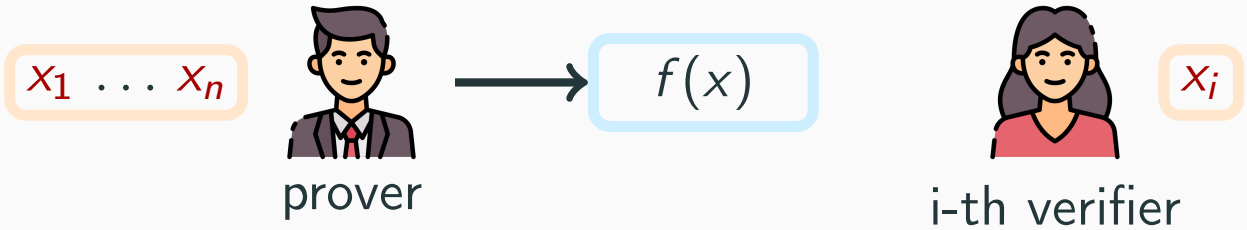
Low Degree Proof: Direct Test [ABCP23]



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(\alpha_i) == x_i$

Low Degree Proof: Direct Test [ABCP23]



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(\alpha_i) == x_i$

⚠ $\Omega(d)$ **communication** and **verification**.

Low Degree Proof: Polynomial Commitment

$x_1 \ \ldots \ x_n$

prover

x_i

i-th verifier

Low Degree Proof: Polynomial Commitment

$x_1 \ \ldots \ x_n$

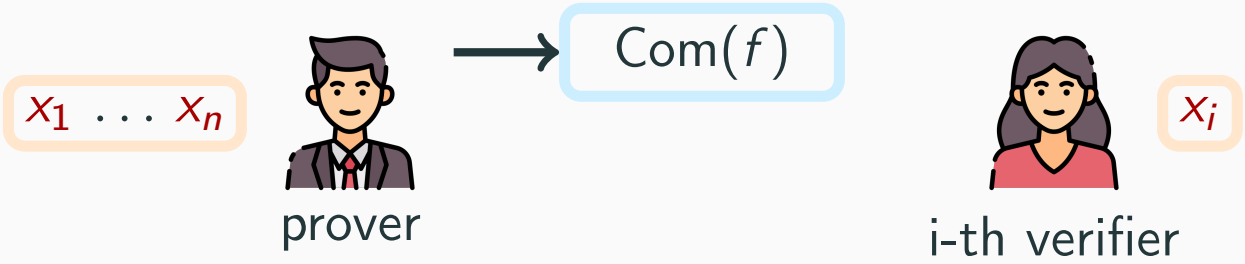
prover

x_i

i-th verifier

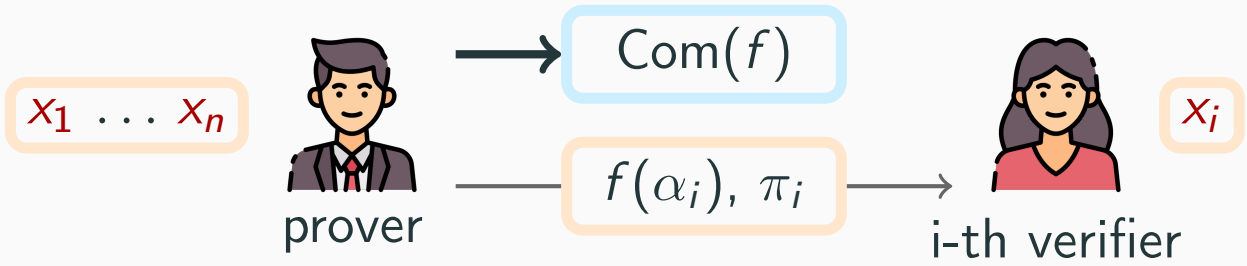
$$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$$

Low Degree Proof: Polynomial Commitment



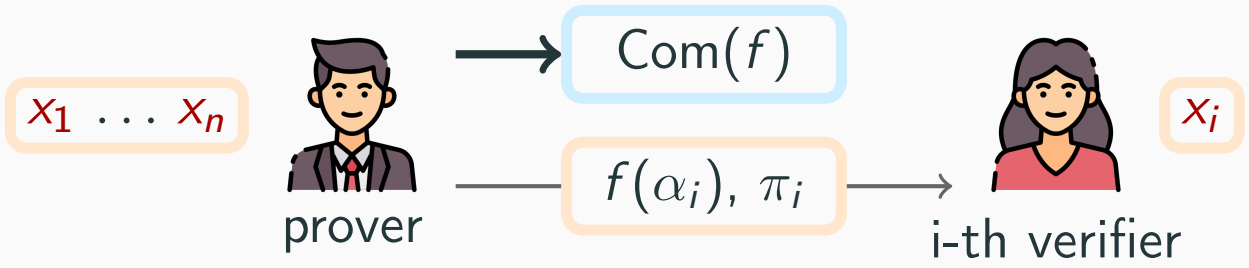
$$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$$

Low Degree Proof: Polynomial Commitment



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

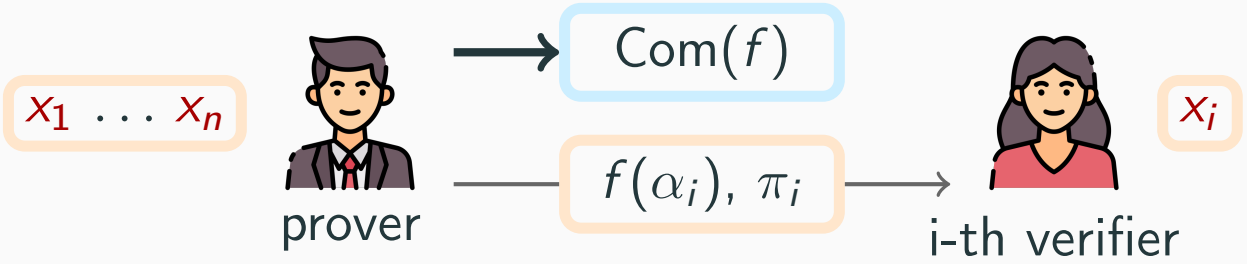
Low Degree Proof: Polynomial Commitment



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

π_i is valid and $f(\alpha_i) = x_i$

Low Degree Proof: Polynomial Commitment



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$ $\quad$ π_i is valid and $f(\alpha_i) = x_i$

⚠ Some PC are **non post-quantum** [KZG10]

⚠ Some PC require $\Omega(n^2)$ **prover time** for multi-point evaluations, such as FRI [BBHR18]

Low Degree Proof: Folding

$x_1 \ldots x_n$

prover

x_i

i-th verifier

Low Degree Proof: Folding

$x_1 \ \ldots \ x_n$

prover

x_i

i-th verifier

$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

Low Degree Proof: Folding

$x_1 \ \ldots \ x_n$

prover

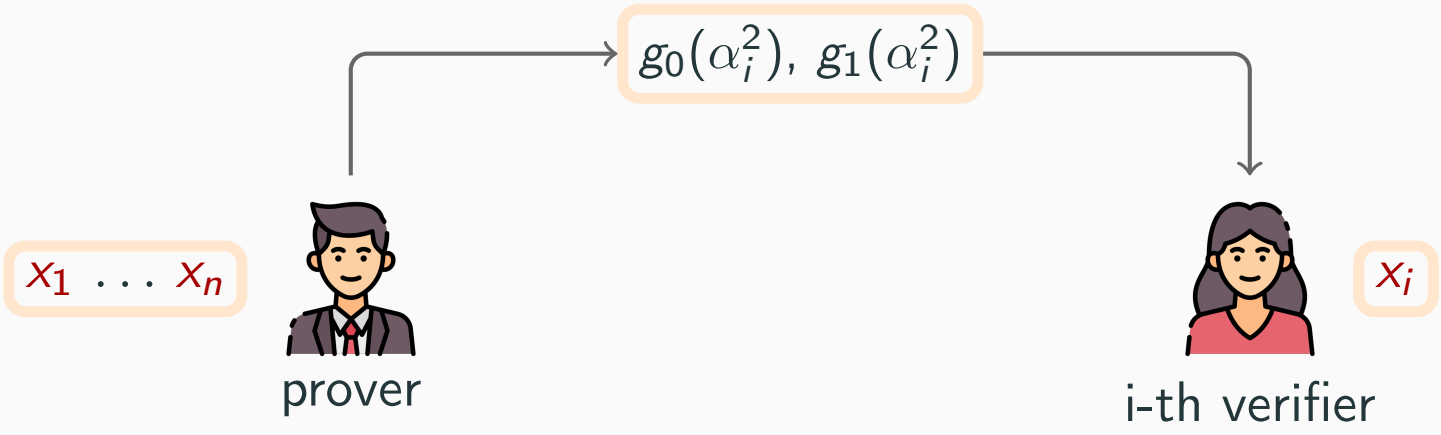
x_i

i-th verifier

$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

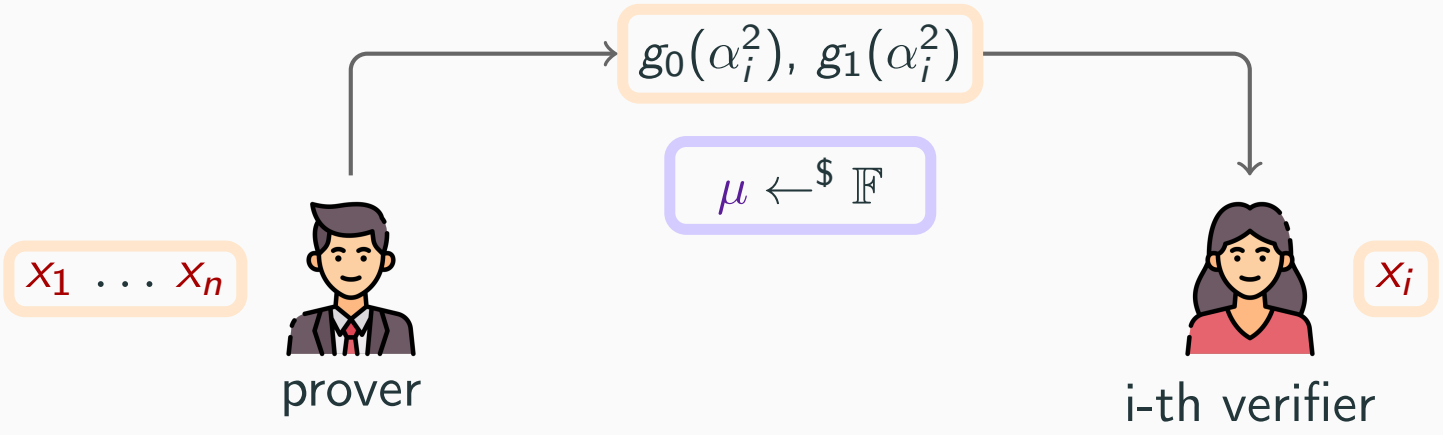
Low Degree Proof: Folding



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

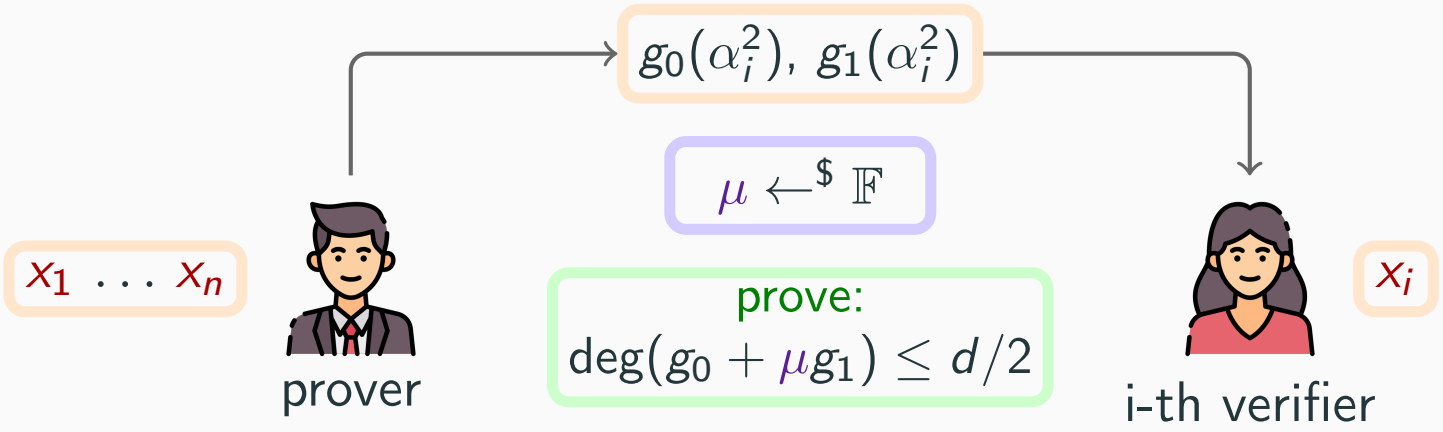
Low Degree Proof: Folding



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

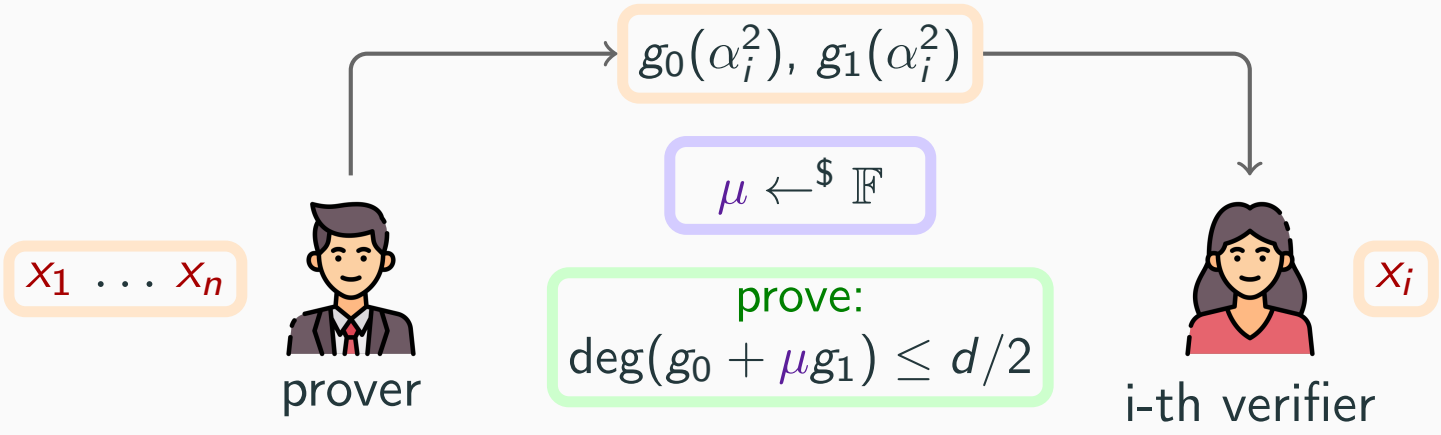
Low Degree Proof: Folding



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

Low Degree Proof: Folding

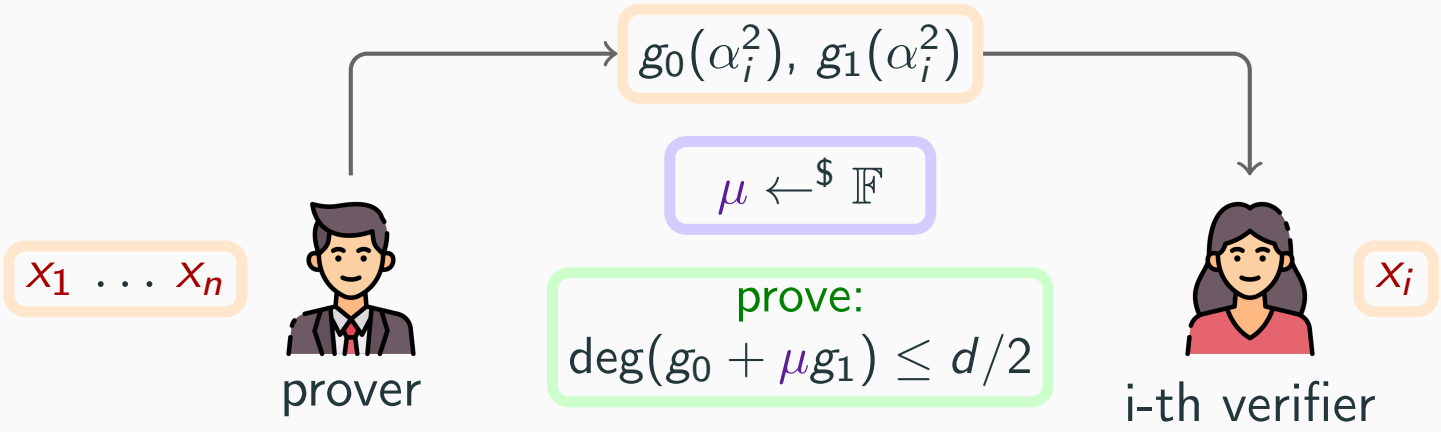


$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

$x_i = g_0(\alpha_i^2) + \alpha_i g_1(\alpha_i^2)$

Low Degree Proof: Folding



$f \in \mathbb{F}[x]_d : f(\alpha_j) = x_j$

$f(x) = g_0(x^2) + x \cdot g_1(x^2)$

$x_i = g_0(\alpha_i^2) + \alpha_i g_1(\alpha_i^2)$

✅ Sound with **at least** $d + 1$ honest verifiers.

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

- Let $m_{j,1}, \ldots, m_{j,n}$ the prover's **private** messages to $V_1 \ldots V_n$.

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

- Let $m_{j,1}, \ldots, m_{j,n}$ the prover's **private** messages to $V_1 \ldots V_n$.
- Let M_j the prover's broadcast message.

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

- Let $m_{j,1}, \ldots, m_{j,n}$ the prover's **private** messages to $V_1 \ldots V_n$.
- Let M_j the prover's broadcast message.
- $R_j = \mathsf{MerkleTree}(m_{j,1}, \ldots, m_{j,n})$.

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

- Let $m_{j,1}, \ldots, m_{j,n}$ the prover's **private** messages to $V_1 \ldots V_n$.
- Let M_j the prover's broadcast message.
- $R_j = \mathsf{MerkleTree}(m_{j,1}, \ldots, m_{j,n})$.
- $\mu_j = \mathsf{H}(M_1, R_1, \ldots, M_j, R_j)$.

Non Interactive Proof in the ROM

We describe a [BCS16]-like compiler to make **public-coin** distributed proofs **non interactive**. At the j-th round:

- Let $m_{j,1}, \ldots, m_{j,n}$ the prover's **private** messages to $V_1 \ldots V_n$.
- Let M_j the prover's broadcast message.
- $R_j = \mathsf{MerkleTree}(m_{j,1}, \ldots, m_{j,n})$.
- $\mu_j = \mathsf{H}(M_1, R_1, \ldots, M_j, R_j)$.
- **Send** $(m_{j,i}, \pi_{j,i})$ to V_i with $\pi_{j,i}$ opening of R_j in i.

Low Degree Proof in the ROM

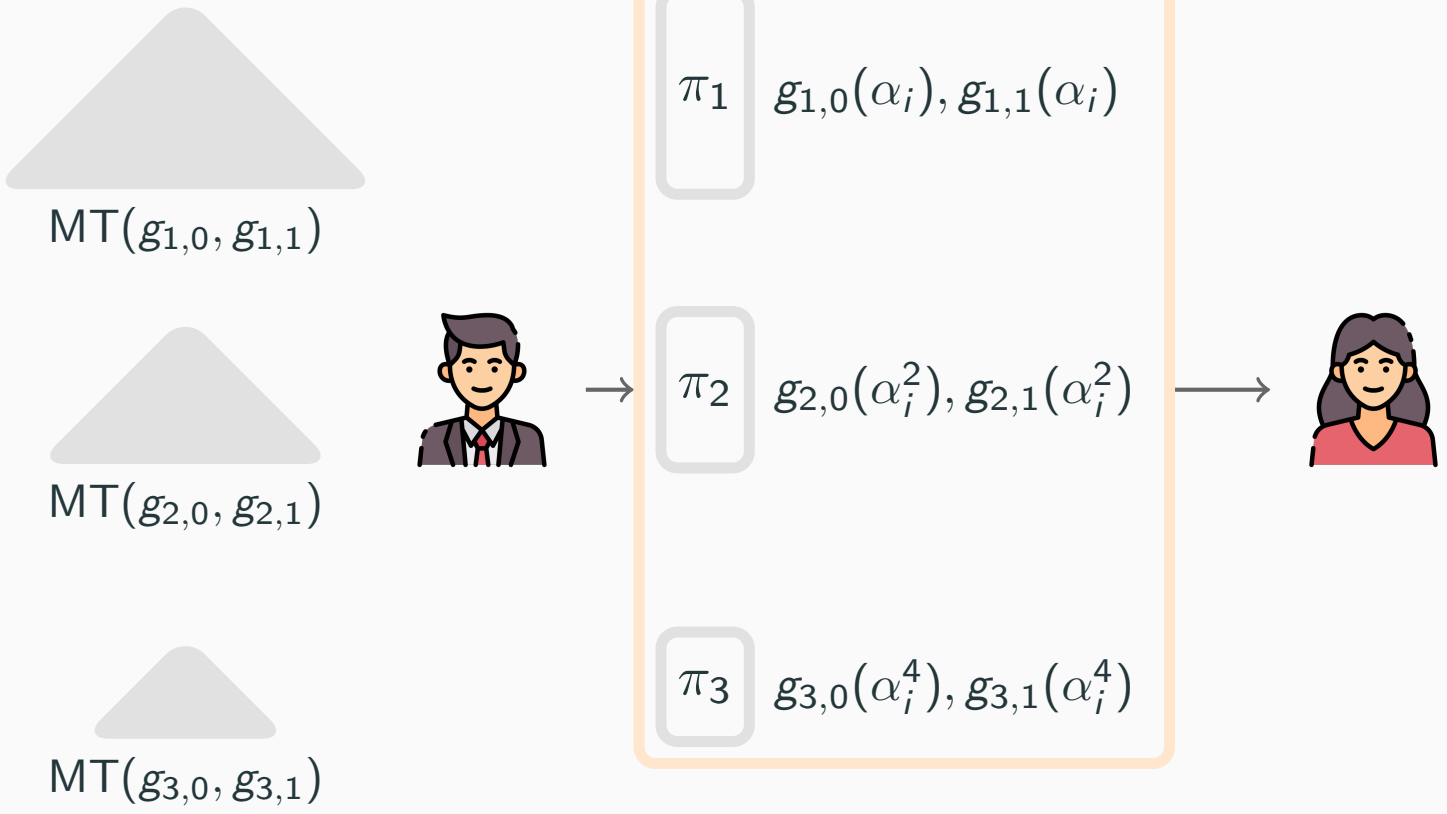
Low Degree Proof in the ROM

$\mathsf{MT}(g_{1,0}, g_{1,1})$

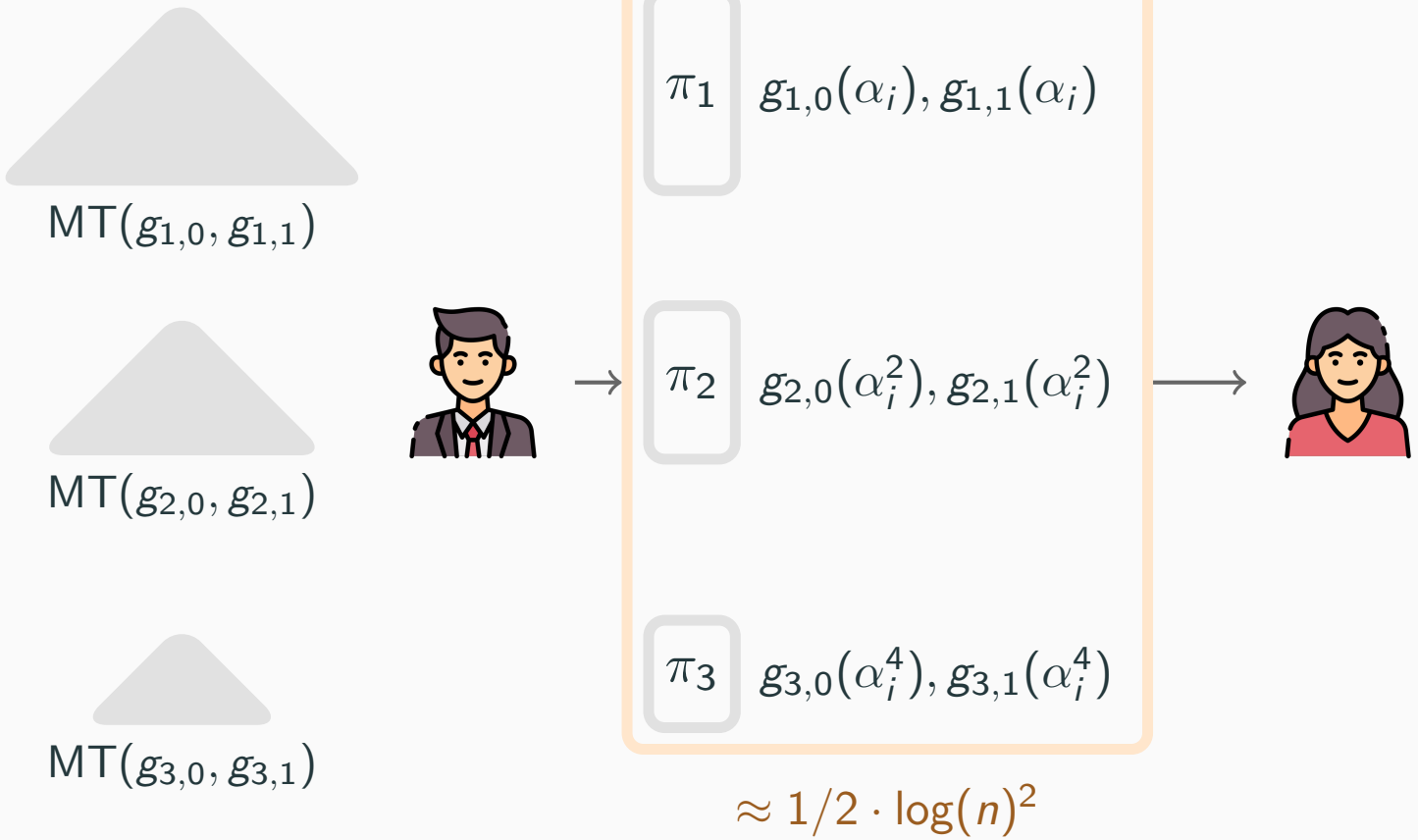
$\mathsf{MT}(g_{2,0}, g_{2,1})$

$\mathsf{MT}(g_{3,0}, g_{3,1})$

Low Degree Proof in the ROM



Low Degree Proof in the ROM



Conclusion

We presented a new (3-round) **VSS** in the **ROM** secure against $t < n/2$ corruptions with:

- **Sublinear** verifier's **download** and **computational** complexity in the best case.
- **Comparable costs** with state of the art VSS [ABCP23] in the worst case.

Thanks for your attention!