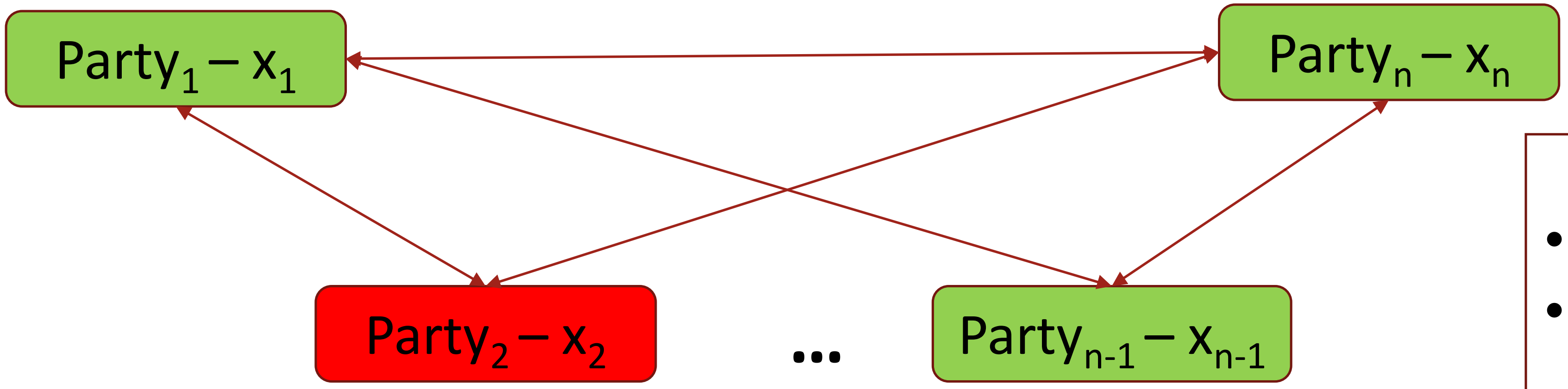
Fully Linear PCPs and their Cryptographic Applications

Niv Gilboa – Ben-Gurion University

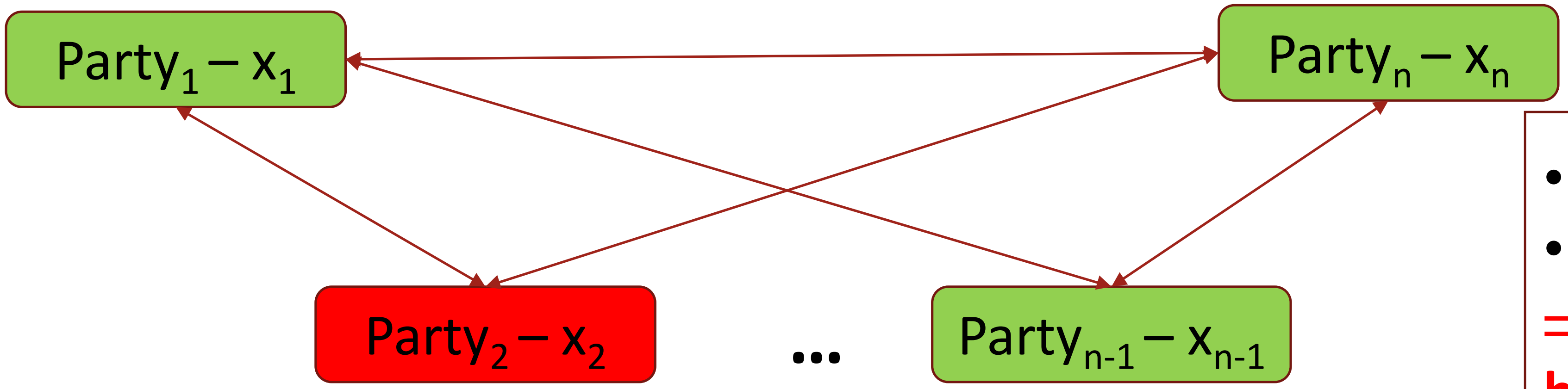
Based On

- Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai, ``Fully Linear PCPs and their Cryptographic Applications'', Crypto 2019.
- Elette Boyle, Niv Gilboa, Yuval Ishai, and Ariel Nof, `` Practical Fully Secure Three-Party Computation via Sublinear Distributed Zero-Knowledge Proofs'', ACM-CCS 2019.

Goal



- Compute $f(x_1,\ldots,x_n)$
- Semi-honest adversary



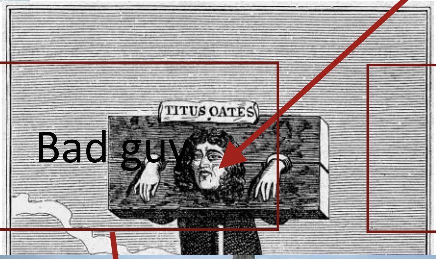
- Prove correctness
- Com. =o(circuit size)

$\Rightarrow$ Malicious com. $\approx$ semi-honest com.

Flavors of Malicious Security

- Security with abort
  - Incorrect execution
  - Who’s the bad guy? ➔
  - Abort

Bad guy



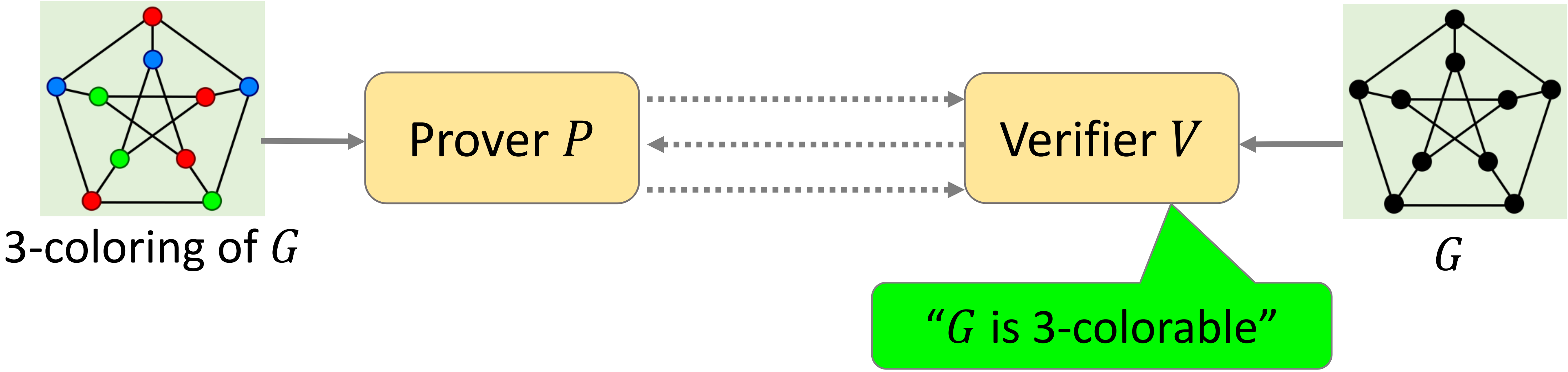
Bad guy

Good guy

- Full security – guaranteed output delivery
  - Incorrect execution
  - What we want
  - What we actually get

Zero-knowledge proofs

[GMR89]



Complete. Honest P convinces honest V.

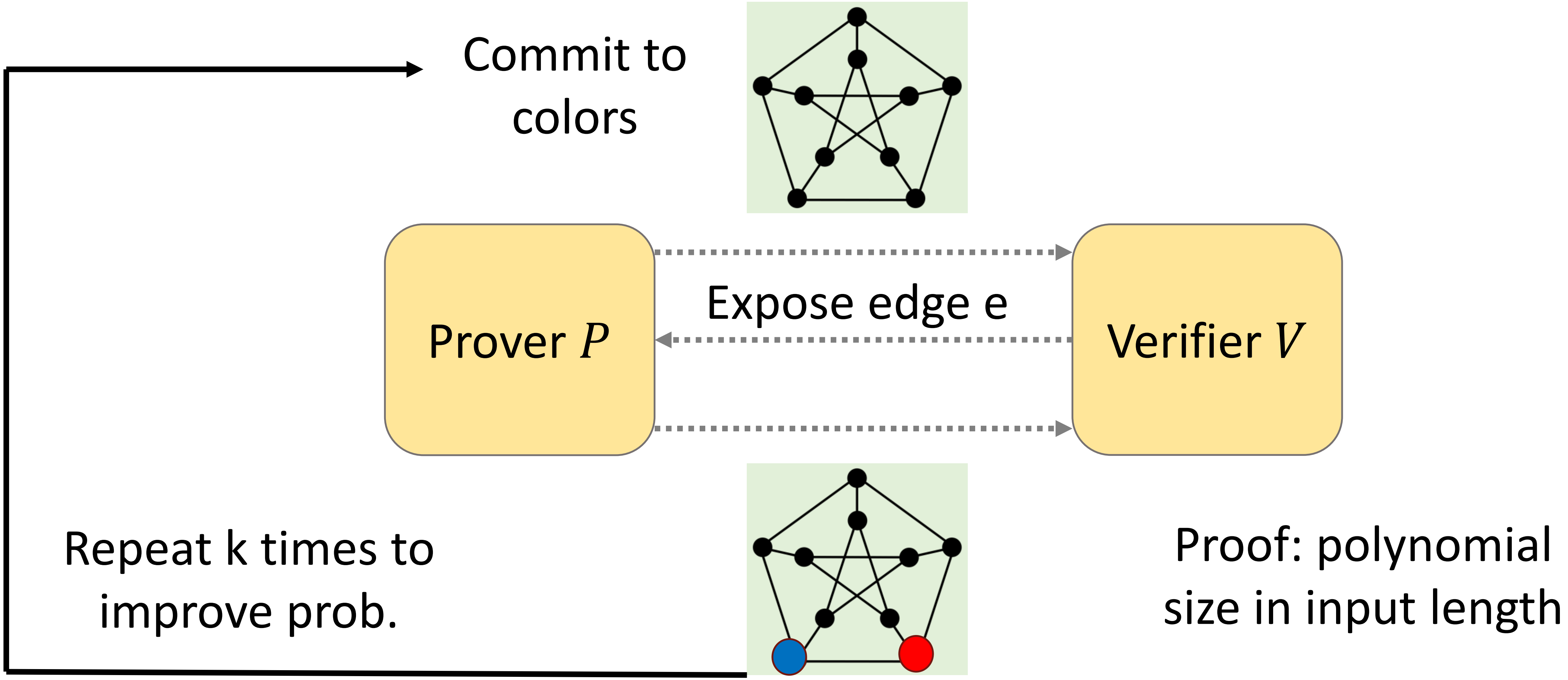
Sound. Dishonest P^* rarely fools honest V.

ZK. Dishonest V^* learns only that $G \in 3\mathrm{COL}$.

→ V^* **learns nothing else about** G

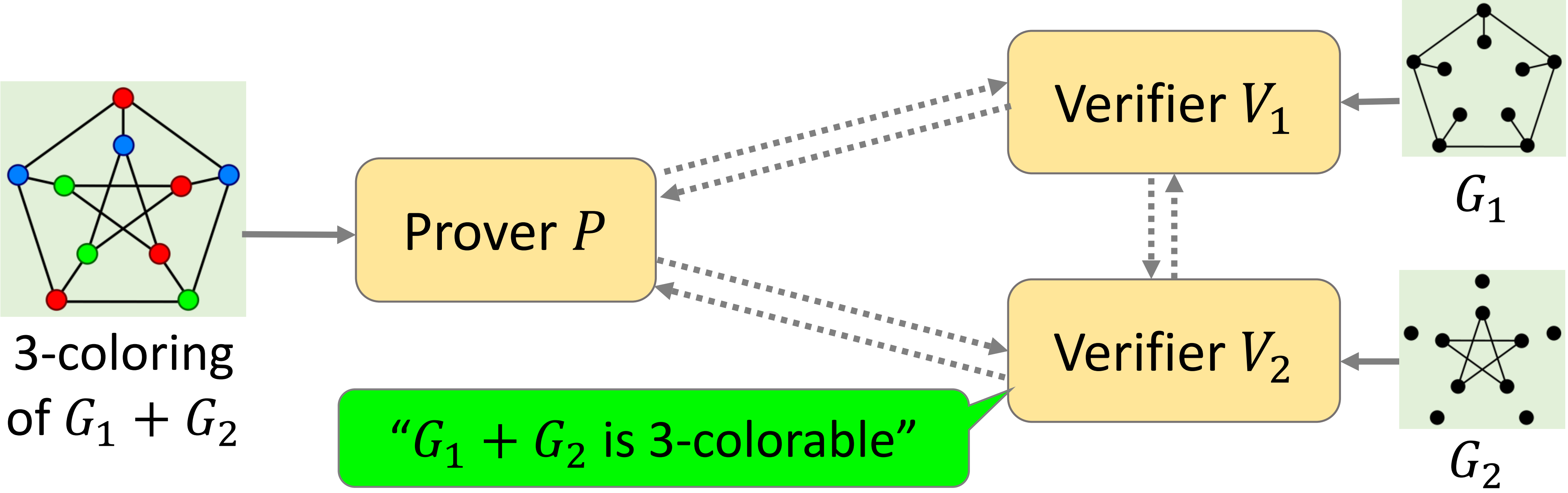
ZK for NP Statements (3-Colorability)

[GMW91]



This talk

Zero-knowledge proofs on **distributed** data



Complete. Honest P convinces honest (V_1, V_2).

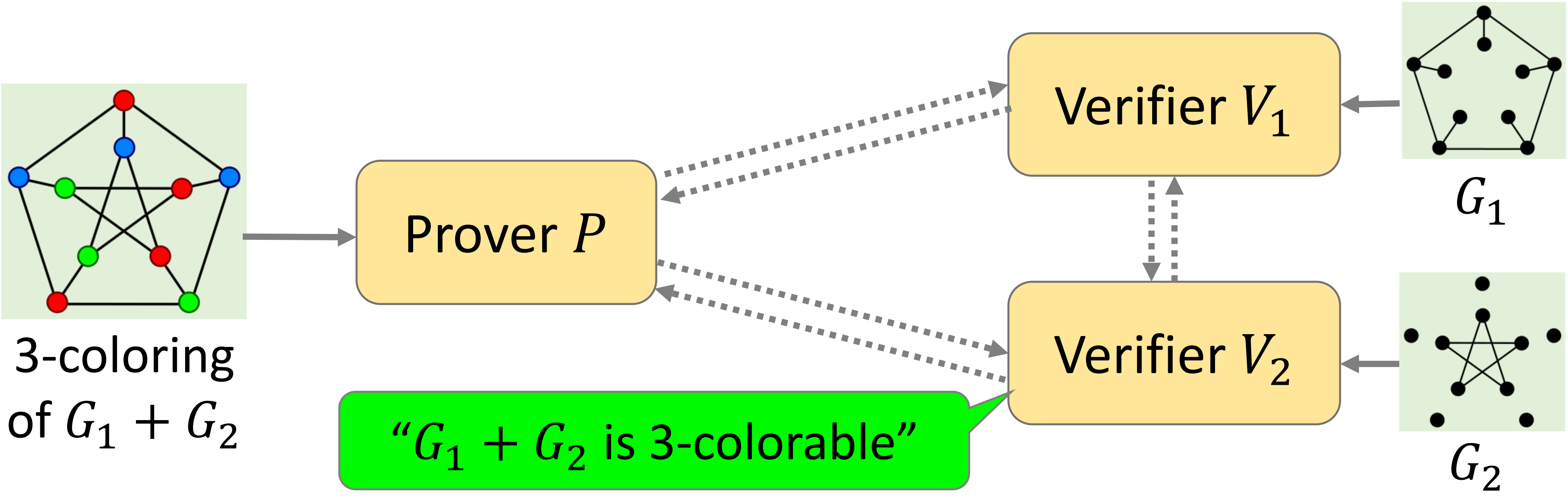
Sound. Dishonest P^* rarely fools honest (V_1, V_2).

Strong ZK. Dishonest V_1^* (or V_2^*) learns only that $G_1 + G_2 \in 3\mathrm{COL}$.

→ V_1 **learns nothing else about** G_2

This talk

Zero-knowledge proofs on **distributed** data



***k*-round protocol** = As in other multiparty protocols

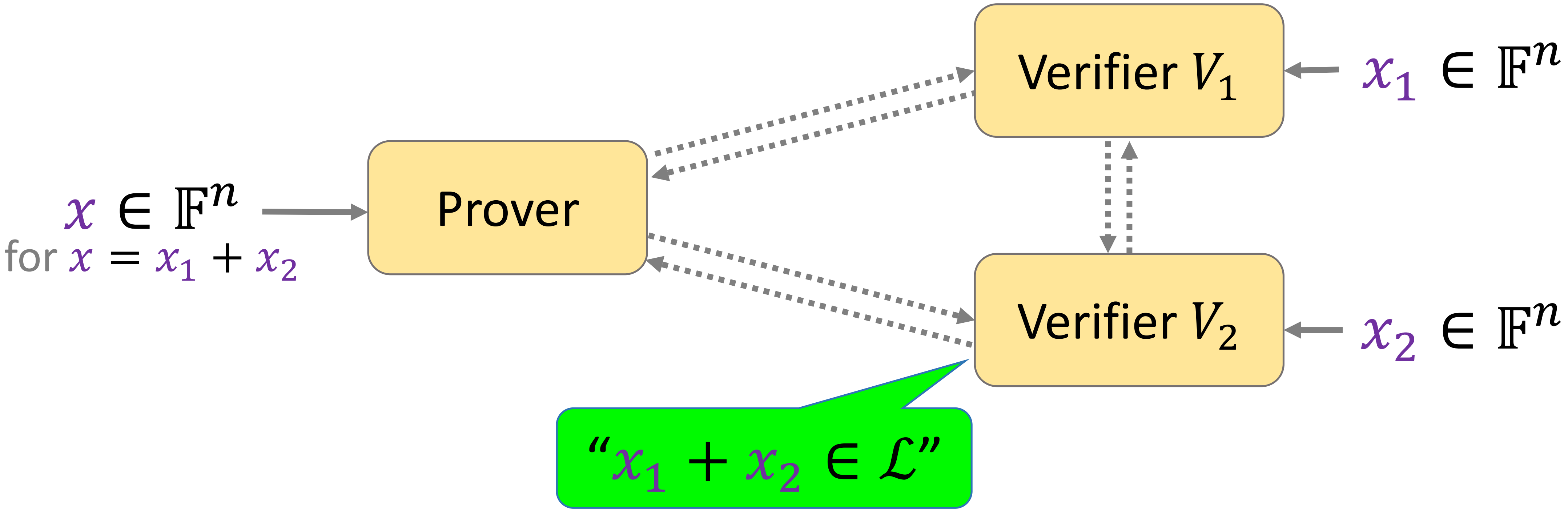
Public coin = Verifiers' messages to prover are random strings

More than two verifiers

Special case

Zero-knowledge proofs on **secret-shared** data

Language $\mathcal{L} \subseteq \mathbb{F}^n$, for finite field $\mathbb{F}$.



Fully Linear PCP / IOP

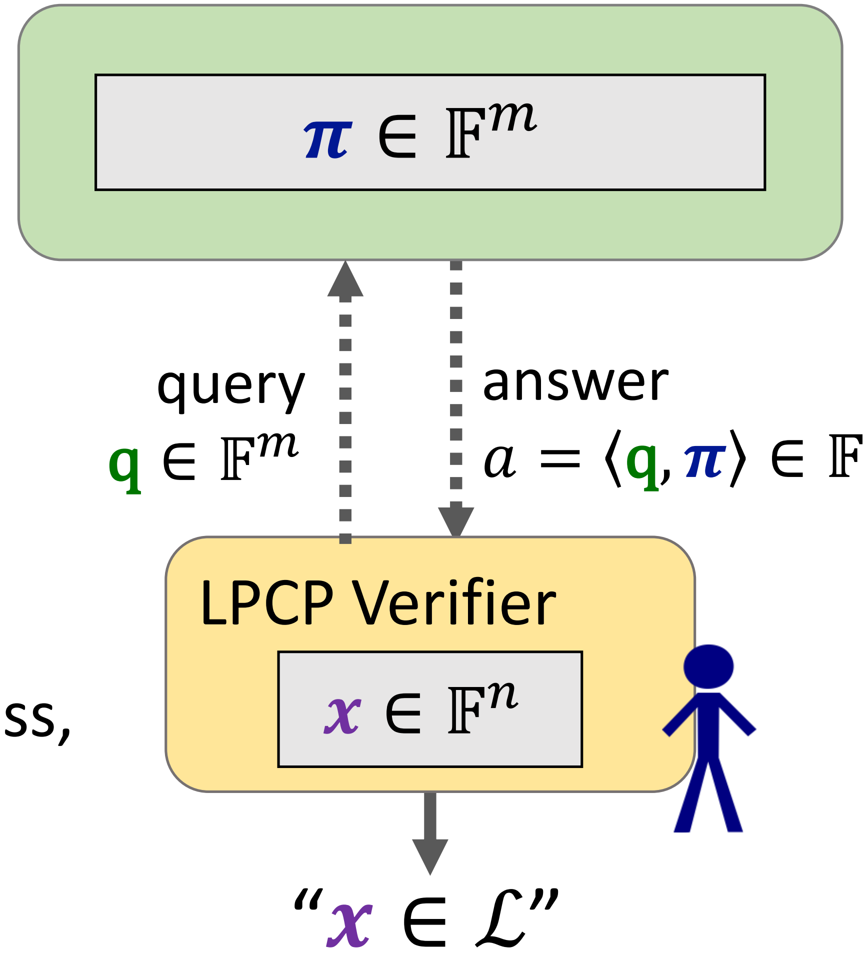
Linear Probabilistically Checkable Proofs (PCPs) [IKO07]

Finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

Linear PCP proof is a vector $\boldsymbol{\pi}$.

Linear PCP verifier
– takes $\boldsymbol{x}$ as input,
– makes $O(1)$ linear queries to $\boldsymbol{\pi}$.

Must satisfy notions of completeness, soundness, and zero knowledge.



<u>Fully</u> linear probabilistically checkable proofs (PCPs)

[This line of work]

Finite field $\mathbb{F}$, language $\mathcal{L} \subseteq \mathbb{F}^n$

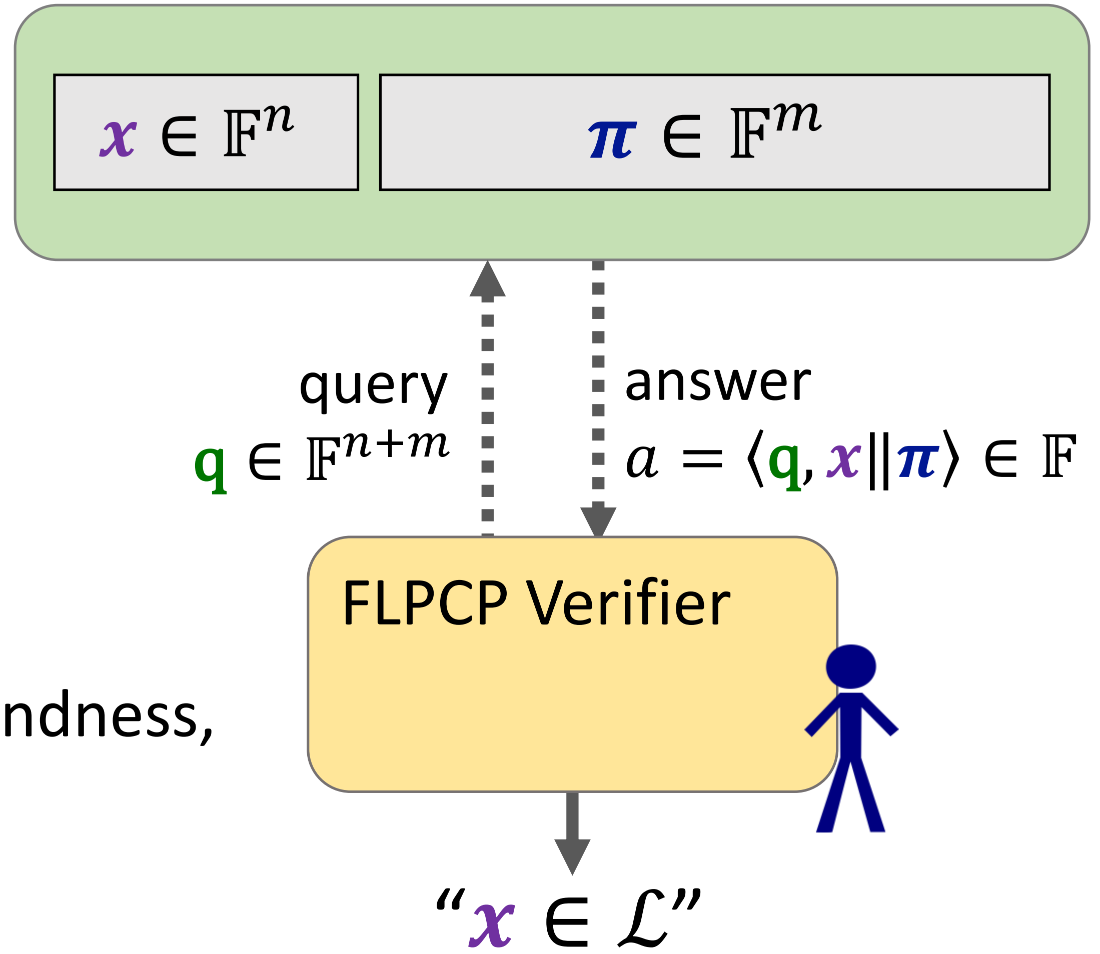
<u>Fully</u> linear PCP proof is a vector $\boldsymbol{\pi}$.

<u>Fully</u> linear PCP verifier

~~– takes $\boldsymbol{x}$ as input,~~

– makes $O(1)$ linear queries to $(\boldsymbol{x}\|\boldsymbol{\pi})$.

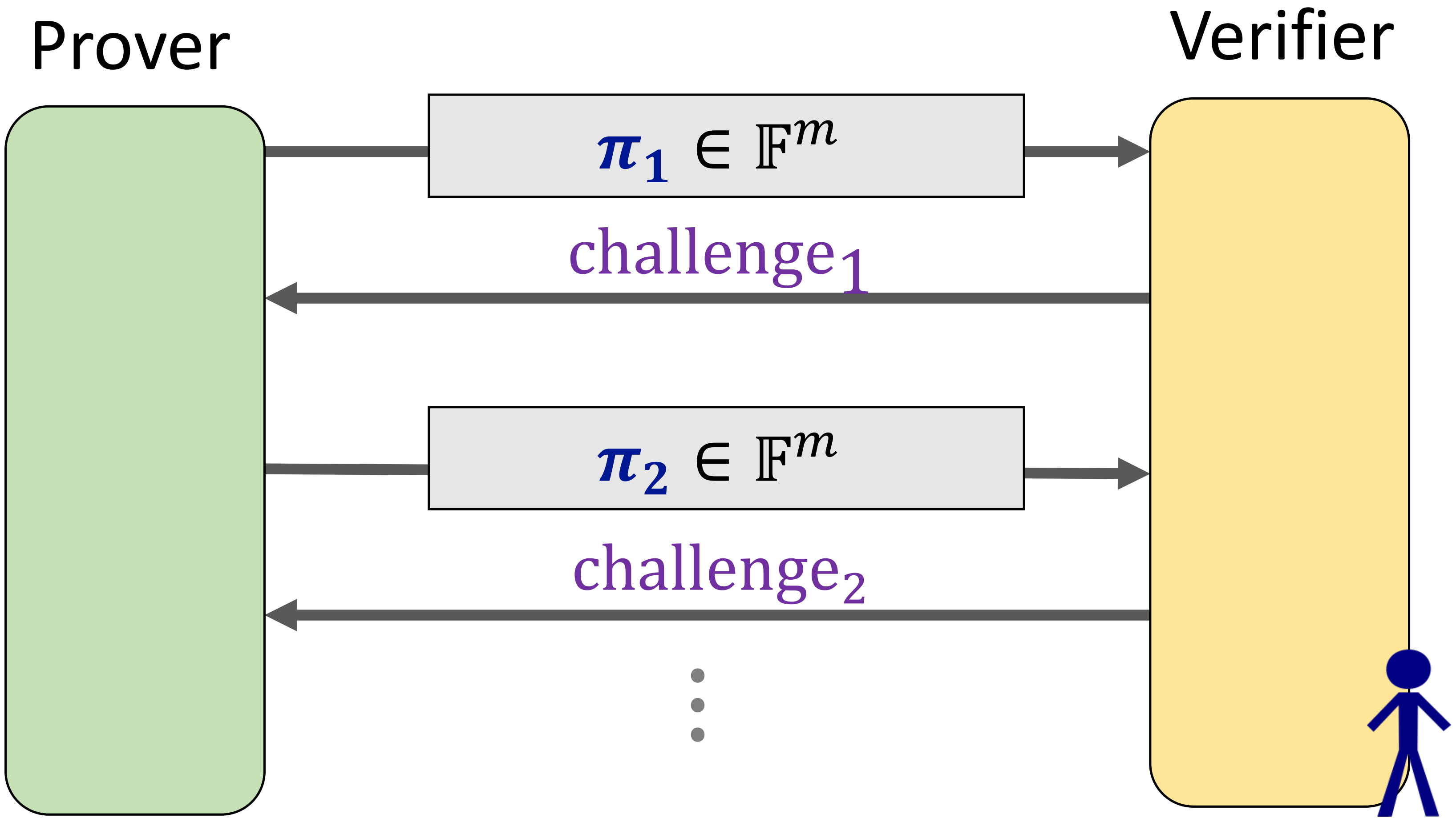
Must satisfy notions of completeness, soundness, and zero knowledge.



Fully linear IOPs

An interactive analogue of fully linear PCPs

Linear analogue + ZK of: [BCS16], [RRR16]



At the end of the interaction, verifier makes linear queries to

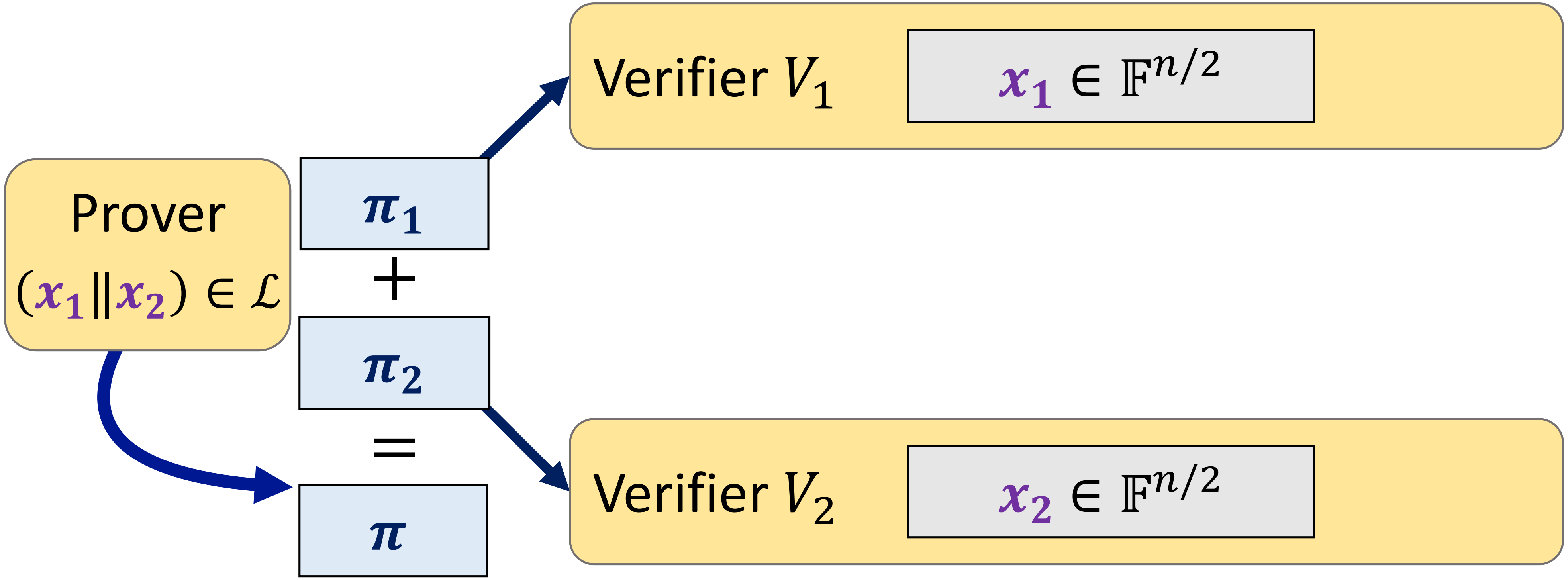
$$(x|\pi_1|\pi_2|\ldots|\pi_t)$$

and accepts or rejects.

Naturally captures many existing proof protocols (GKR, …)

If language $\mathcal{L}$ has an efficient fully linear PCP/IOP, it has an efficient ZK proof on distributed data.

1. Generate FLPCP proof and split it using secret sharing.



If language $\mathcal{L}$ has an efficient fully linear PCP/IOP, it has an efficient ZK proof on distributed data.

2. Sample query vectors using common randomness.

Verifier V_1 | $\boldsymbol{x_1} \in \mathbb{F}^{n/2}$ | $\boldsymbol{\pi_1}$

Query $\mathbf{q} = $ 5 | 1 | 2 | 7 | 4 | 9

Verifier V_2 | $\boldsymbol{x_2} \in \mathbb{F}^{n/2}$ | $\boldsymbol{\pi_2}$

If language $\mathcal{L}$ has an efficient fully linear PCP/IOP, it has an efficient ZK proof on distributed data.

3. Publish shares of query answers and reconstruct.

Verifier V_1 | $\boldsymbol{x}_1 \in \mathbb{F}^{n/2}$ | $\boldsymbol{\pi}_1$

$\langle \mathbf{q}, \boldsymbol{x}_1 \| \boldsymbol{\pi}_1 \rangle \in \mathbb{F}$

$+$

$\langle \mathbf{q}, \boldsymbol{x}_2 \| \boldsymbol{\pi}_2 \rangle \in \mathbb{F}$

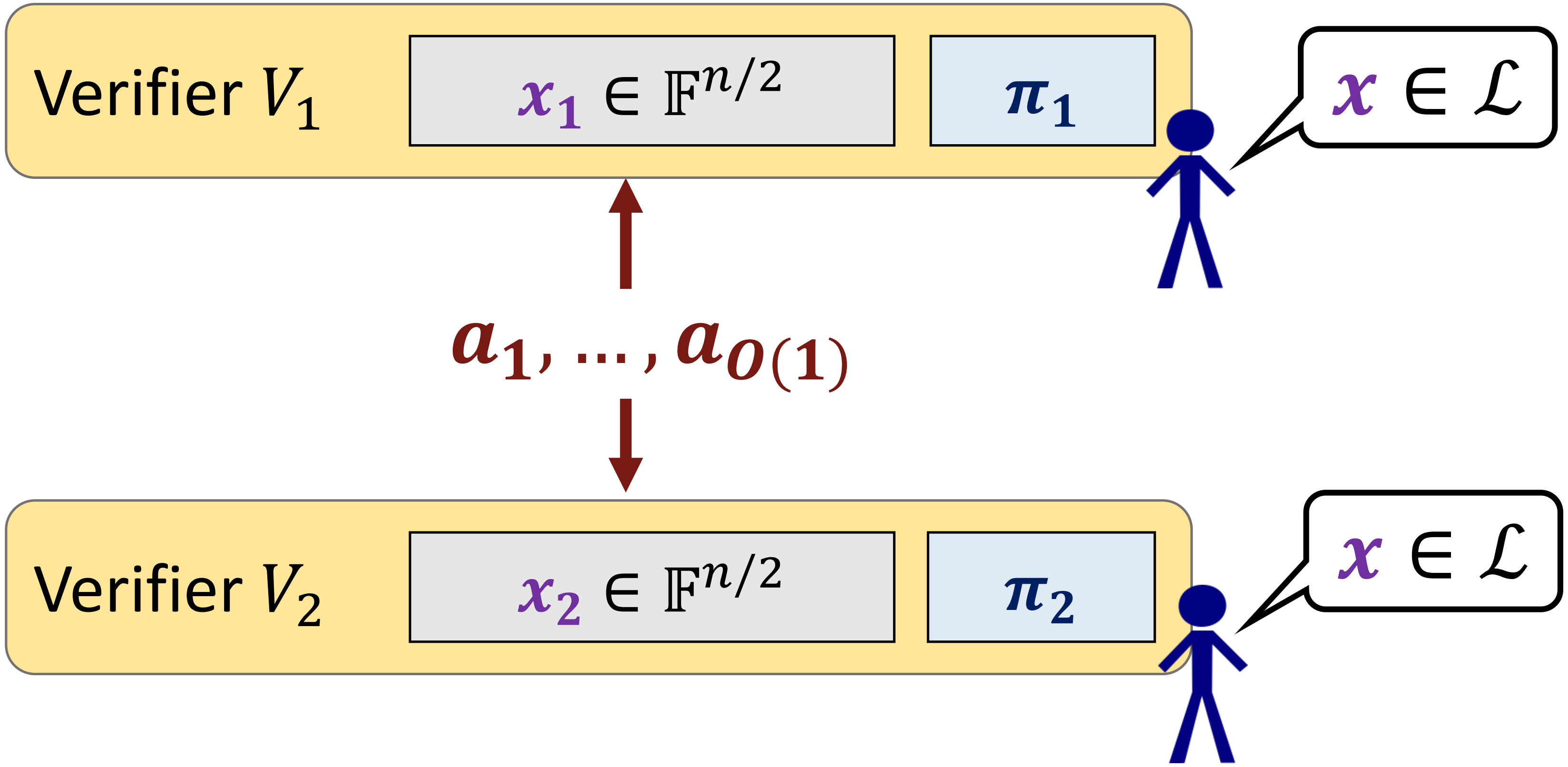
$$= \langle \mathbf{q}, \boldsymbol{x} \| (\boldsymbol{\pi}_1 + \boldsymbol{\pi}_2) \rangle$$
$$= \langle \mathbf{q}, \boldsymbol{x} \| \boldsymbol{\pi} \rangle = \text{answer}$$

Verifier V_2 | $\boldsymbol{x}_2 \in \mathbb{F}^{n/2}$ | $\boldsymbol{\pi}_2$

If language $\mathcal{L}$ has an efficient fully linear PCP/IOP, it has an efficient ZK proof on distributed data.

4. Recover $O(1)$ query answers, run FLPCP verifier.

Communication:
$|\text{proof}| + O(1)$



Selected results: New ZK proofs I

$\mathbb{F}$ - finite field, $\mathcal{L} \subseteq \mathbb{F}^n$- language ($n \ll |\mathbb{F}|$), $G: \mathbb{F}^{\boldsymbol{L}} \to \mathbb{F}$– algebraic gate

Theorem. If $\mathcal{L}$ is recognized by a circuit $\boldsymbol{C}$ that has **M** G-gates, and some addition gates, there is a public-coin ZK proof on distributed data for $\mathcal{L}$ with:

- $\boldsymbol{O}(\mathbf{1})$ rounds and
- communication cost $\boldsymbol{O}(\boldsymbol{L} + \boldsymbol{M}(\boldsymbol{deg}.\boldsymbol{G}))$. (elements of $\mathbb{F}$)

Selected results: New ZK proofs II

Theorem. If $\mathcal{L}$ has a **degree-two** arithmetic circuit, there is a public-coin ZK proof on distributed data for $\mathcal{L}$ with:

- $\boldsymbol{k}$ rounds and
- communication cost $\boldsymbol{O}($ $\boldsymbol{n^{O(1/k)}}$ mproves: $\Omega(n)$ [BC17])

19

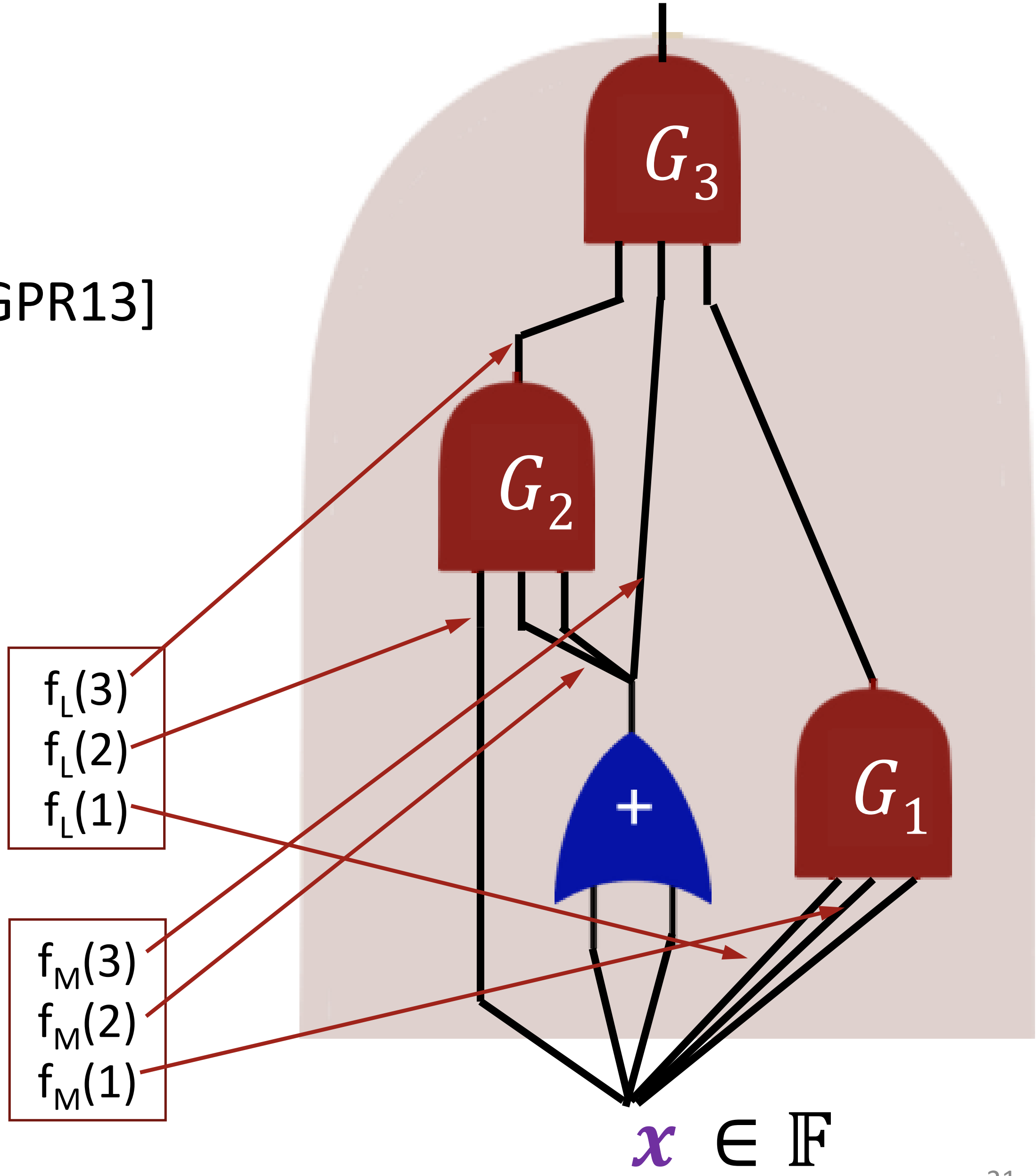
Extensions to:

- Rings Z_{2^k}
- Degree O(1) circuits

Constructions

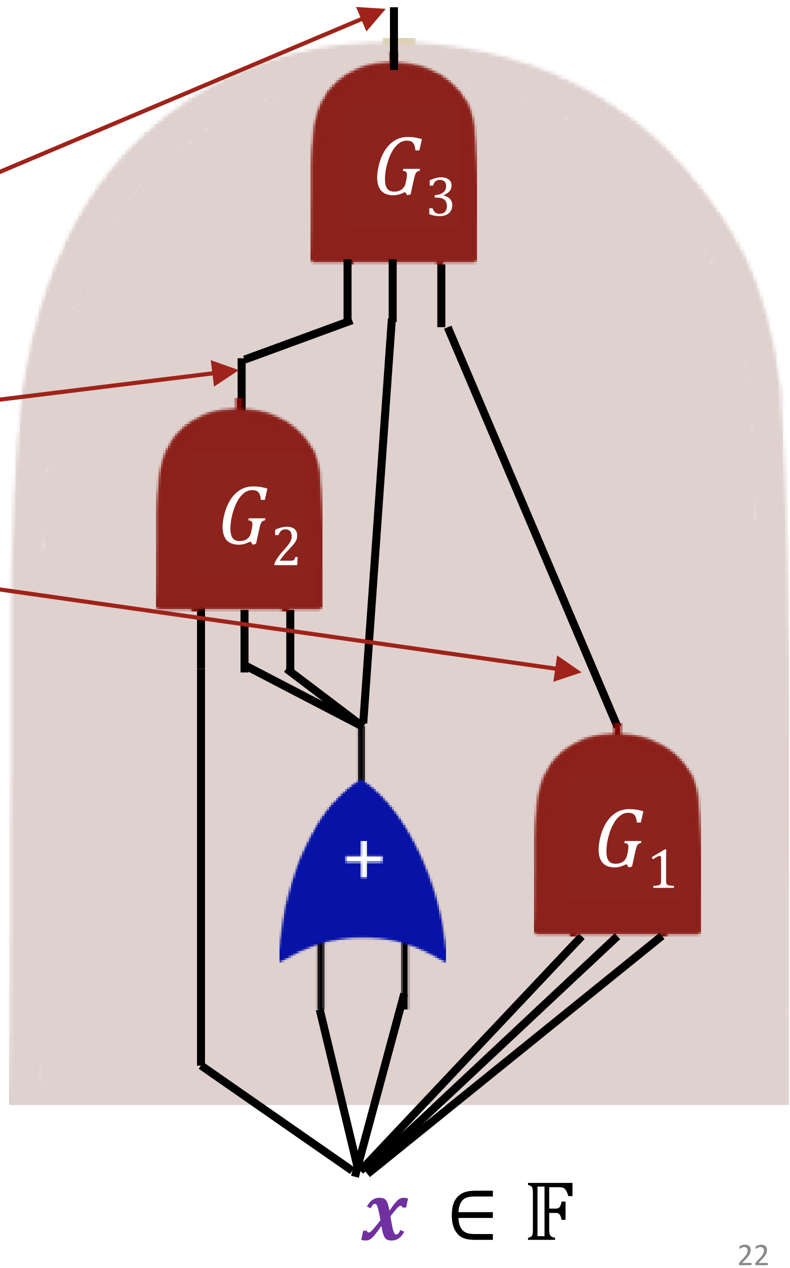
Short proofs for structured circuits I

- Ideas similar to [LFKN92, AW09, GGPR13]
- Circuit over field $\mathbb{F}$:
  - Linear gates
  - “Large” algebraic G-gates
- Order gates
- Define Polynomials
  - f_L – left inputs
  - f_M – middle inputs
  - f_R – right inputs



Short proofs for structured circuits II

- $p = G(f_L, f_M, f_R)$ defines outputs
- p(1)
- p(2)
- p(3)
- p(#G gates)=C(x)

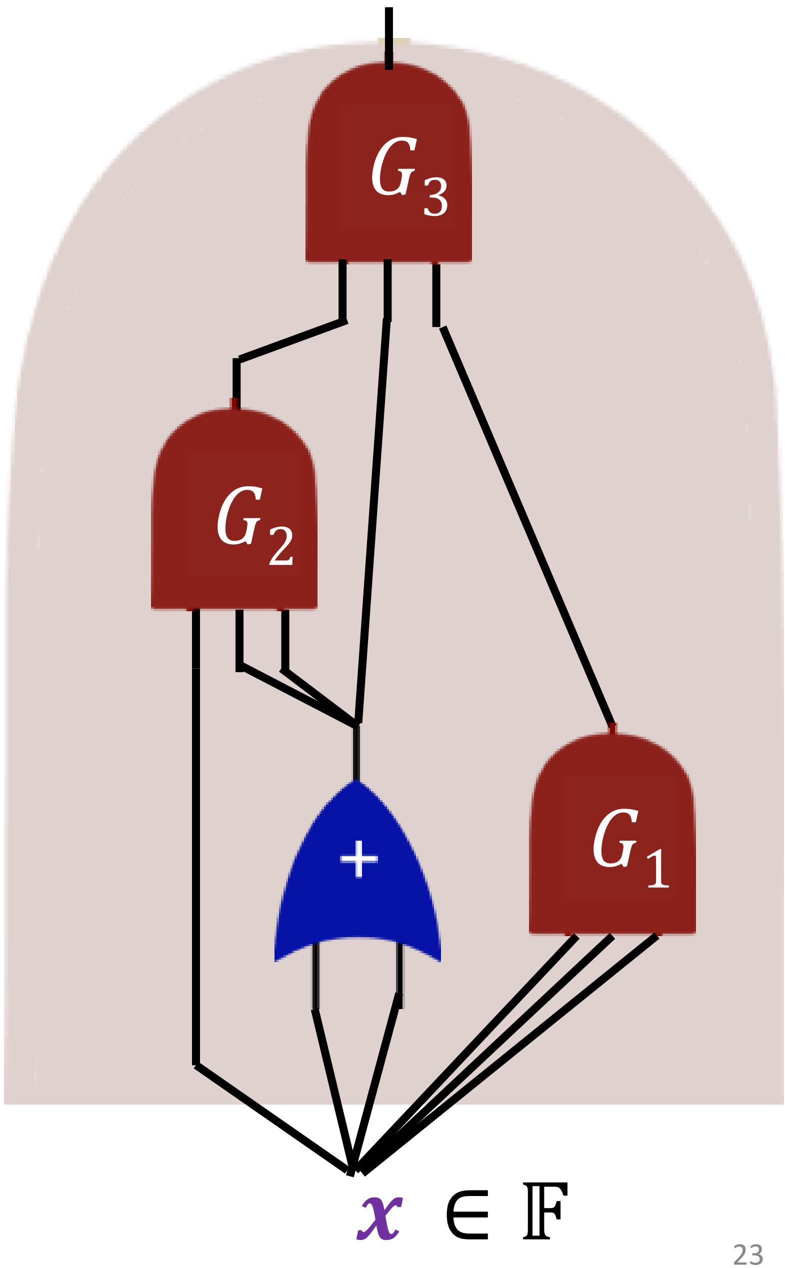


Short proofs for structured circuits II

- Prover sends p
- Length: (#G gates)(degree G)
- Verifier checks
  - p(#G gates)=0 ($x \in \mathcal{L}$)
  - p(r)=G(f_L,f_M,f_R)(r) for random r
- Verifier work requires
  - Interpolation
  - Evaluation

Linear!

- ZK by extra randomization of f_L/f_M/f_R



Corollary

- $O(\sqrt{n})$ FL-PCP for **any** degree 2 circuit (Improves: Prio $\Omega(n)$ [BC17])
- C(x) degree 2 ➔ $C(x) = x^{-1}Ax$ for some matrix A
- $C(x) = \langle \mathbf{x}, Ax \rangle$
- C made up of
  - G gate – inner product on length $n^{1/2}$ inputs
  - Linear gates
- Proof size: $2n^{1/2}$

Reducing communication for simple languages

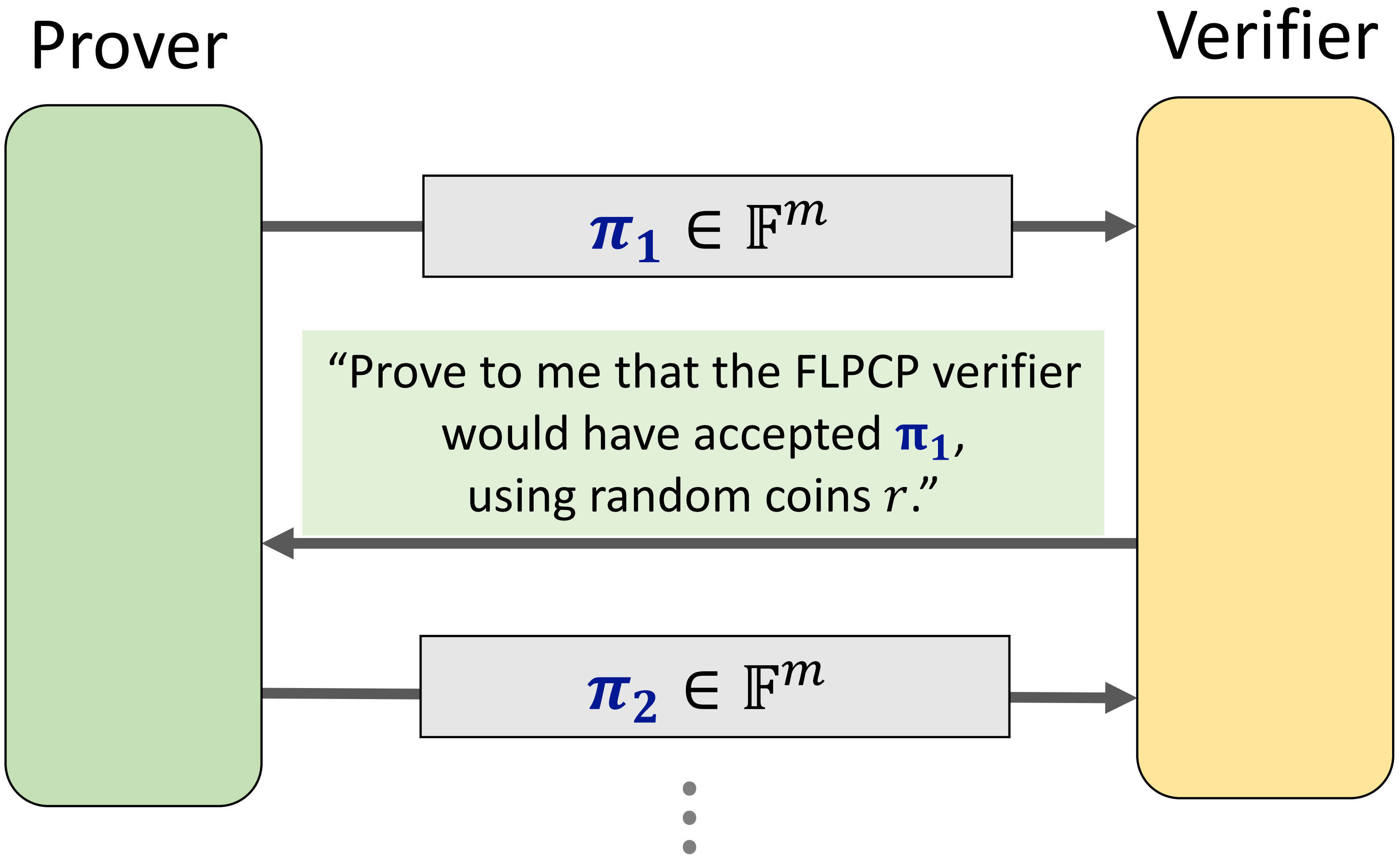
Let $\mathbb{F}$ be a finite field. Let $\mathcal{L} \subseteq \mathbb{F}^n$ be a language. $(n \ll \mathbb{F})$

Theorem. If $\mathcal{L}$ has a **degree-two** arithmetic circuit, there is a public-coin ZK proof on distributed data for $\mathcal{L}$ with:

- $O(\log n)$ rounds and
- communication cost $\boldsymbol{O(\log n)}$.

- Uses our new FLPCP
- Idea: Recursively outsource the verifier's work to the prover.

FL-Interactive Oracle Proof (FLIOP)



For circuits with "SIMD" structure, proof size shrinks: $O(|\mathcal{C}|) \rightarrow O(\log|\mathcal{C}|)$

Low-degree circuits have the necessary structure

Semi-Honest to Malicious MPC Compiler

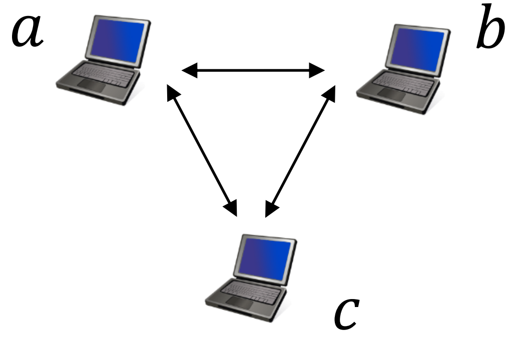
Secure Multi-Party Computation (MPC)

What is the **communication** complexity of **securely** evaluating f?

- **HE:** $\tilde{O}(|a,b,c| + |f(a,b,c)|)$ [G09,BGI16a]
  - Based on heavy cryptographic tools

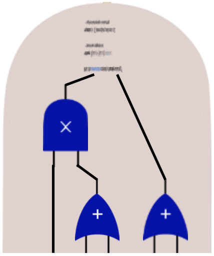
Allowed: Black-box use of PRG

- … elements/party, small const $\alpha \geq 2$
  - Long line of work improving α in various settings



$\boldsymbol{\alpha \geq 2}$ Step 2: (standard) "active" security

Lightweight $\boldsymbol{\alpha = 1}$ Step 1: (weak) "passive" security



C = (Boolean/arithmetic) circuit representation of f

Results

- Generic MPC: compiler from semi-honest to malicious
  - "Natural" protocols
  - Semi-honest majority
  - **Any** number of parties – secure with abort
  - **Constant** Number of parties – full security
  - In this talk – focus on 3PC
  - Sub-linear communication (in circuit size)
  - Soundness – $1/|F|$, but reduce by extension field
- Specific MPC functionalities
  - Even better communication!

Comparison of 3PC Protocols

The protocol	# of elements sent per party per multiplication gate				Full security?
	Boolean Circuits	Circuits over $\mathbb{F}_{2^8}$	Circuits over the ring $\mathbb{Z}_{2^{64}}$	Circuits over large finite fields ($\|\mathbb{F}\| \geq 2^{40}$)	
Araki et al. [ABF+17]	7	7	7	7	No
Chaudhari et al. [CCPS19]	7(offline)+4/3(online)	-	7(offline)+4/3(online)	-	No
Chida et al. [CGH+18]	41	6	41	2	No
Eerikson et al. [EOP+19]	123	-	5	-	No
This work	1	1	1	1	Yes

3PC: Main Theorem

Given any **passive-secure 3PC** protocol with **“natural”** structure, then can achieve **active** security with **+o(|C|)/party extra comm**

“Natural” 3PC protocol:

Input Shares of Adv: Commit to his input

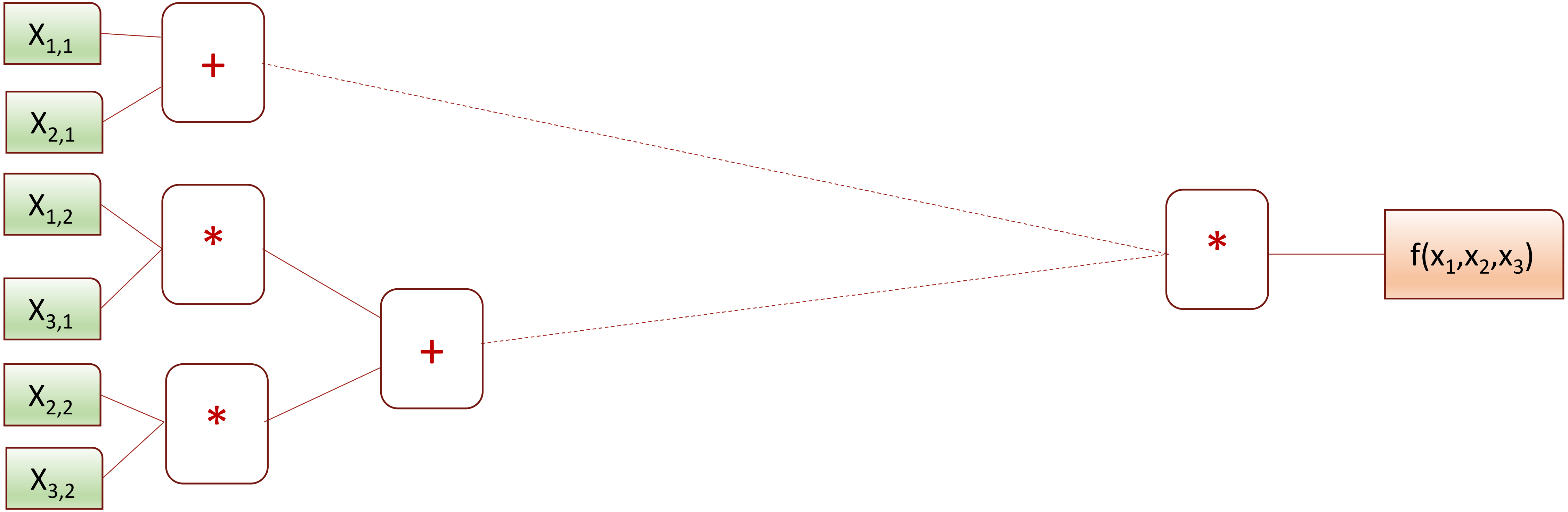
Before final message: Total random garbage

If… Some degree-2 relation holds on msgs
… then [robustly shared y] $= \boldsymbol{C}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$

Final round: Robust shares of output

Natural Protocol – Example [AFLNO16]

Step 0: Represent f as circuit



Natural Protocol - Example

Step 1: Secret Share inputs

Step 2: Secret Share zeros

Party 1	Party 2				Party 1	Party 2	Party 3
X		→	a+b+c=x	→	a, b	b, c	a, c
	y	→	d+e+f=x	→	d, e	e, f	d, f
Seed k	Seed k	→			Long shared mask_{12}		

Natural Protocol - Example

Step 3: distributed evaluation of every gate

x → a+b+c=x

y → d+e+f=y

	Party 1	Party 2	Party 3
x⊕y	a+d, b+e	b+e, c+f	a+d, c+f
x·y	a(d+e)+ b(d+e)	bf+ce	af+cd

Mask
Re-share values

3PC "Passive" secure protocol

Communication, Using PRG tricks

ws protocol instructions

1 /input

1. Secret share inputs
   (note: linear shares)

~~0 amort~~

2. Generate |C| sets of shares of 0

3. Gate-by-gate evaluation
   + : Locally on shares
   x : Cross-terms $a_i b_j$ computable!
   Locally: Compute additive shares
   Compute, mask, & send share

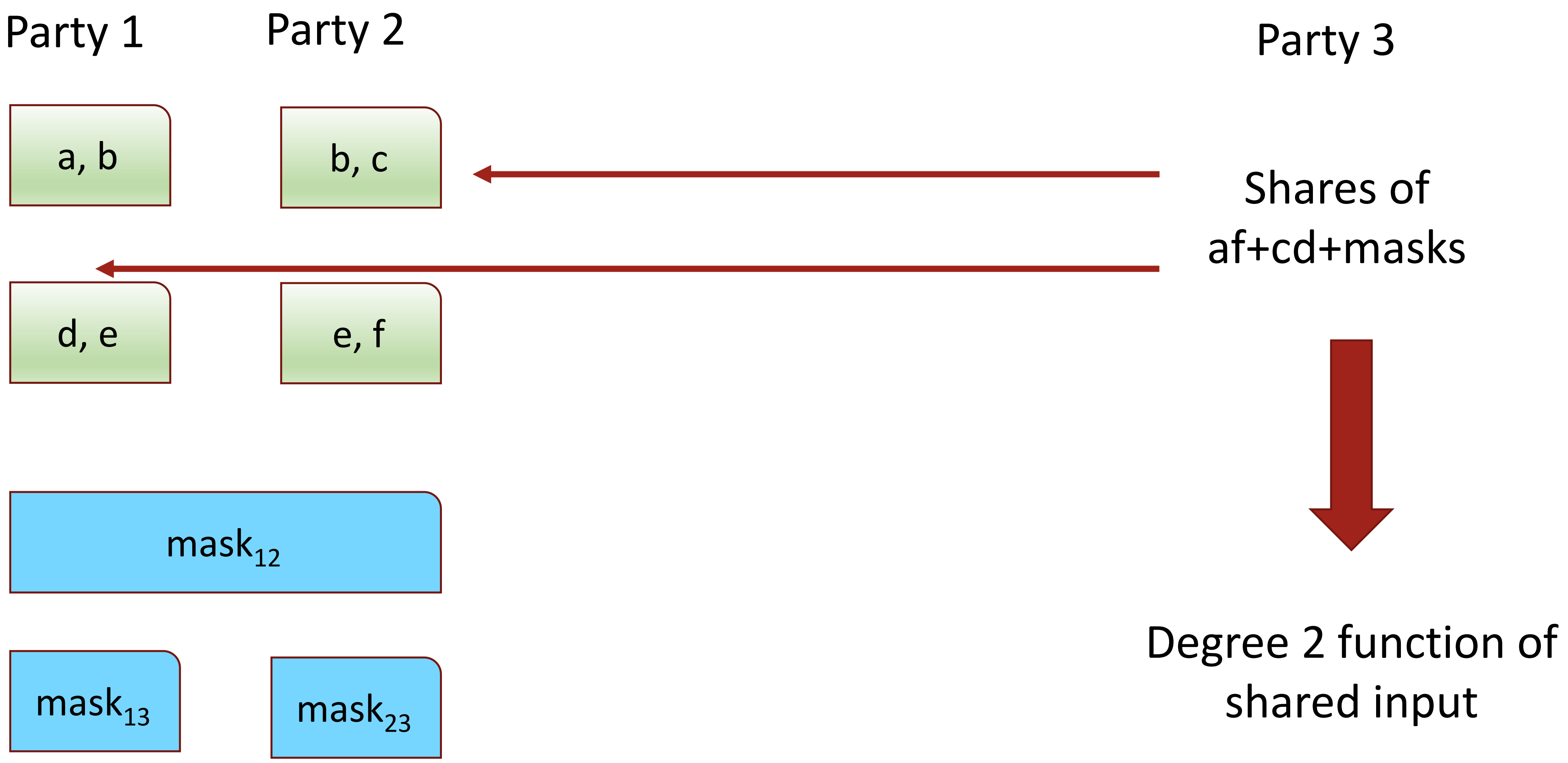
1 /mult

1 /output

4. Output gate: Exchange final shares

Comm Cost: 1 elmt/party/multiplication

Verifying Correct Execution

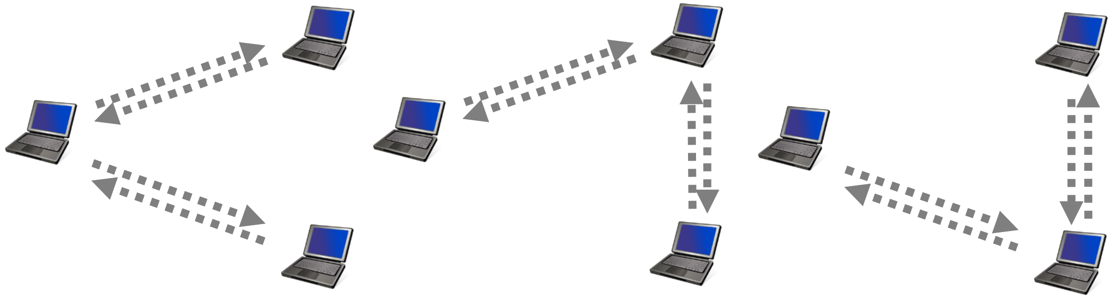


Collective 3PC Protocol

Protocol Π'
(without final message)

Final Message (robust shares, expect same message from two parties

Total extra communication:
$|\text{proof}| + |\text{verifier comm}| = o(|C|)$



Each party proves in ZK their messages in Π' were computed correctly

Fully secure – abort leads to identifying "good" party

3PC Summary

- Fully Linear PCPs: Proving on secret shared / committed / distributed data
- New (passive → active) security compiler for 3PC
- Concrete efficiency:
  - 2^{20} gate circuit
  - 0.5 Kbyte communication
  - 30 field operations per gate GF(2^{47})
  - Soundness 2^{-40}

(Standard) active security

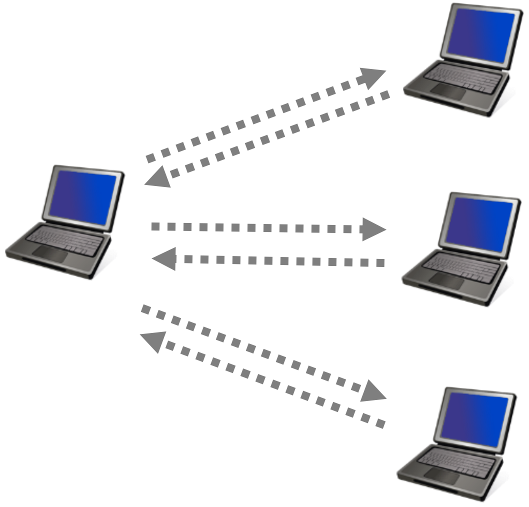
+ o(|C|) communication

(Weak) passive security

Protocols with a particular "natural" structure

Extending to $n > 3$ Parties

- Challenge: **Malicious prover + verifier(s)**
  - Even defining soundness becomes non-trivial
  - Requires "robustness" of pieces of statement x
- Challenge: In MPC protocol with n>3,
  Prover **no longer knows** the full robust statement
  - Involves messages Prover wasn't involved in
- Challenge: Replication based protocol inefficient for $\omega(1)$ servers
- Approach – Parties distribute role of prover. Stay tuned…

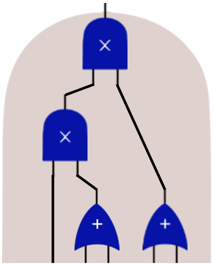


Thank You!

Applying Our Compiler to “Natural” Protocols

- **3 parties, 1 corruption** (“3PC”)
  - Motivated setting: “Minimal” across MPC settings eg: [MRZ15,AFLN+16,ABFL+17,LN17,FLNW17,CGHI+18,GR018,NV18,EnOP+19]
  - Comparison:

Over large field:	$\boldsymbol{\alpha = 2}$	[CGHIKLN18, NV18]
Over Boolean:	$\boldsymbol{\alpha = 7}$	[ABFLLNOWW17]
Any field or $\mathbb{Z}_N$	$\boldsymbol{\alpha = 1}$	[This work] Compiling eg, [AFLNO16,KKW18]



- **Constant $\boldsymbol{n \in O(1)}$ parties, $\boldsymbol{t}$ corruptions,** $n = 2t + 1$

Over large field:	$\boldsymbol{\alpha = 3}$	[CGHIKLN18]
Over Boolean/$\mathbb{Z}_{2^k}$:	$\boldsymbol{\alpha > 40}$	[CGHIKLN18]
Any field or $\mathbb{Z}_N$	$\boldsymbol{\alpha = 3t/(2t+1)} \leq 1.5$	[This line of work] Compiling [DN07]