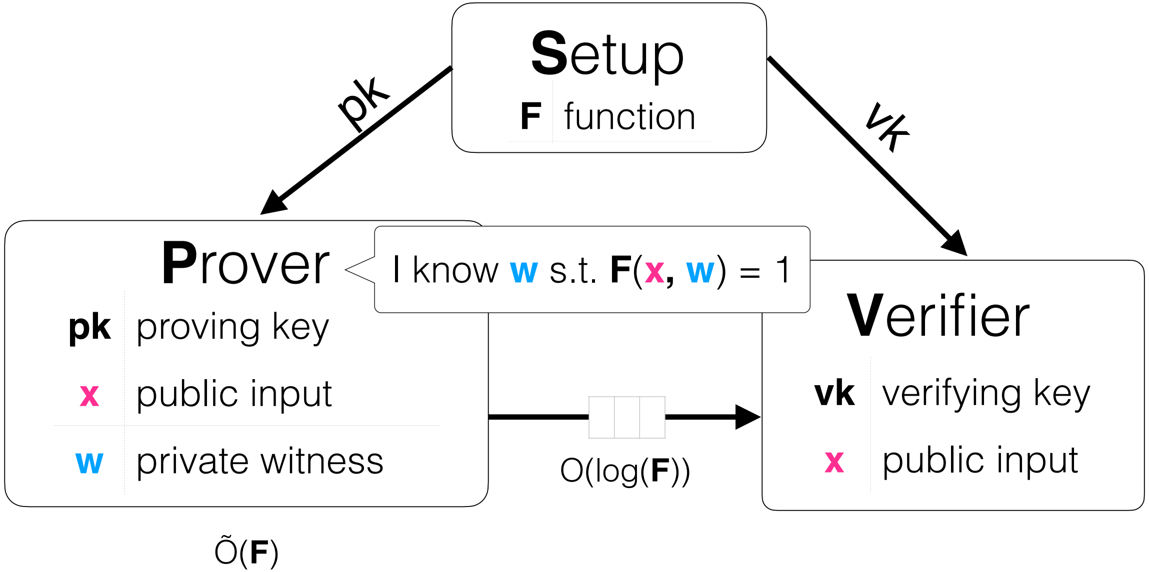
Theory and Practice of Succinct Zero Knowledge Proofs

Lecture 03:
SNARKs from Polynomial Interactive Oracle Proofs

Pratyush Mishra
UPenn
Fall 2023

Succinct Non-Interactive Arguments (SNARGs)

[Mic94, Groth10, GGPR13, Groth16…
…, GWC19, CHM**M**VW20, …]



SNARKs

- **Completeness**: $\forall\ (F, x, w) \in \mathscr{R}$, $\Pr\left[\mathbf{V}(\mathsf{vk}, x, \pi) = 1 \ : \ \begin{array}{c}(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Setup}(F)\\ \pi \leftarrow \mathbf{P}(\mathsf{pk}, x, w)\end{array}\right] = 1.$

- **Knowledge Soundness**: $\forall$ efficient $\tilde{\mathbf{P}}$, $\exists$ extractor $\mathbf{E}$ s.t.

$$\Pr\left[\begin{array}{c} V(\mathsf{vk}, x, \pi) = 1 \\ \wedge \\ (F, x, w) \notin \mathscr{R} \end{array} \ : \ \begin{array}{c}(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Setup}(F)\\ \pi \leftarrow \tilde{\mathbf{P}}(\mathsf{pk}, x) \\ w \leftarrow \mathbf{E}_{\tilde{\mathbf{P}}}(\mathsf{pk}, x)\end{array}\right] \approx 0$$

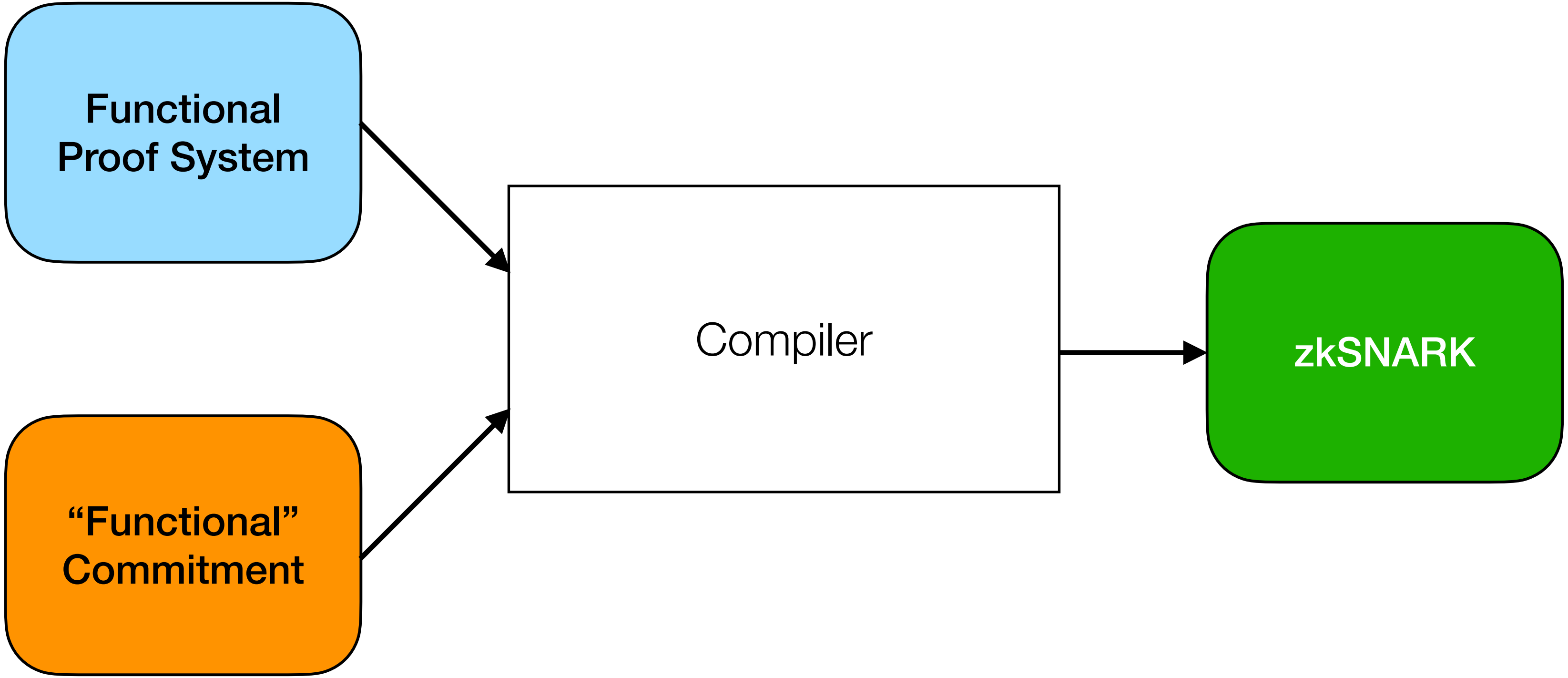
- **Zero Knowledge**: $\exists$ **simulator** Sim s.t. $\forall\ (F, x, w) \in \mathscr{R}$, and all $\tilde{\mathbf{V}}$,

$$\Pr\left[\mathbf{V}(\mathsf{vk}, x, \pi) \ : \ \begin{array}{c}(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Setup}(F)\\ \pi \leftarrow \mathsf{Sim}(\mathsf{pk}, x)\end{array}\right] = \Pr\left[\mathbf{V}(\mathsf{vk}, x, \pi) \ : \ \begin{array}{c}(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{Setup}(F)\\ \pi \leftarrow \mathbf{P}(\mathsf{pk}, x, w)\end{array}\right]$$

- **Succinctness**: $|\pi| = O(\log |F|)$ and $\mathsf{Time}(\mathbf{V}) = O(\log |F|, |x|)$

Constructing zkSNARKs

Blueprint


Functional
Proof System
"Functional"
Commitment
Compiler
zkSNARK

Which function to pick?

Polynomial commitments:

- $F_z(m)$**:** Interpret m as <u>univariate poly</u> $f(X)$ in $\mathbb{F}[X]$ and evaluate at z

Multilinear commitments: e.g., $f\left(x_1, \ldots, x_k\right) = x_1 x_3 + x_1 x_4 x_5 + x_7$

- $F_{\vec{z}}(m)$**:** Interpret m as <u>multilinear poly</u> $f(X)$ in $\mathbb{F}[\overrightarrow{X}]$ and evaluate at $\vec{z}$

Vector commitments:

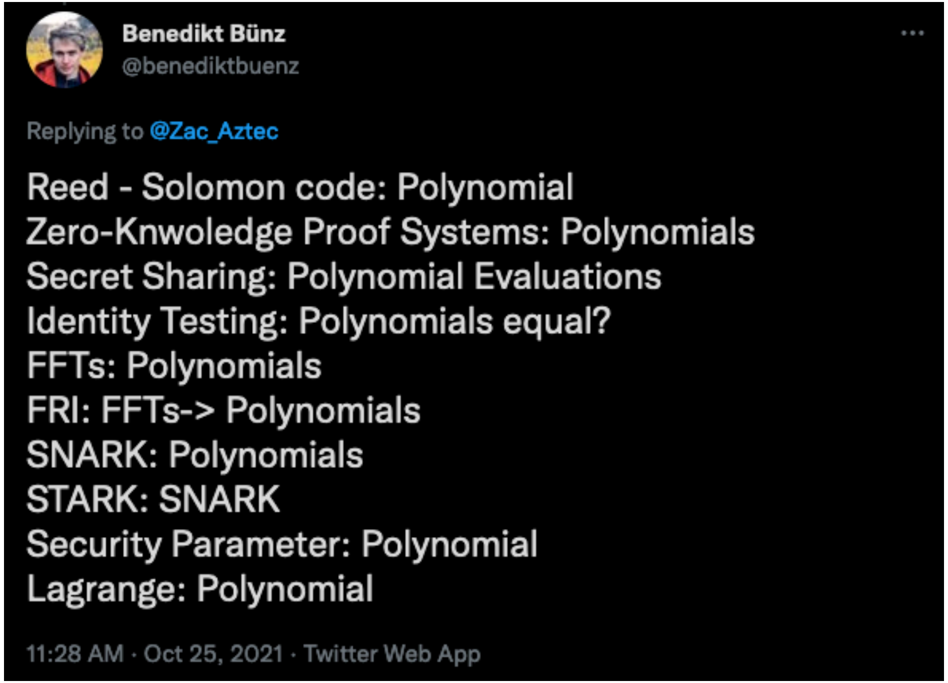
- $F_i(m)$**:** Interpret m as <u>vector</u> v in $\mathbb{F}^n$ and return v_i

Inner-product commitments:

- $F_{\vec{q}}(m)$**:** Interpret m as <u>vector</u> $\vec{v}$ in $\mathbb{F}^n$ and return $\langle \vec{v}, \vec{q} \rangle$

Which to pick?

A: Polynomials!



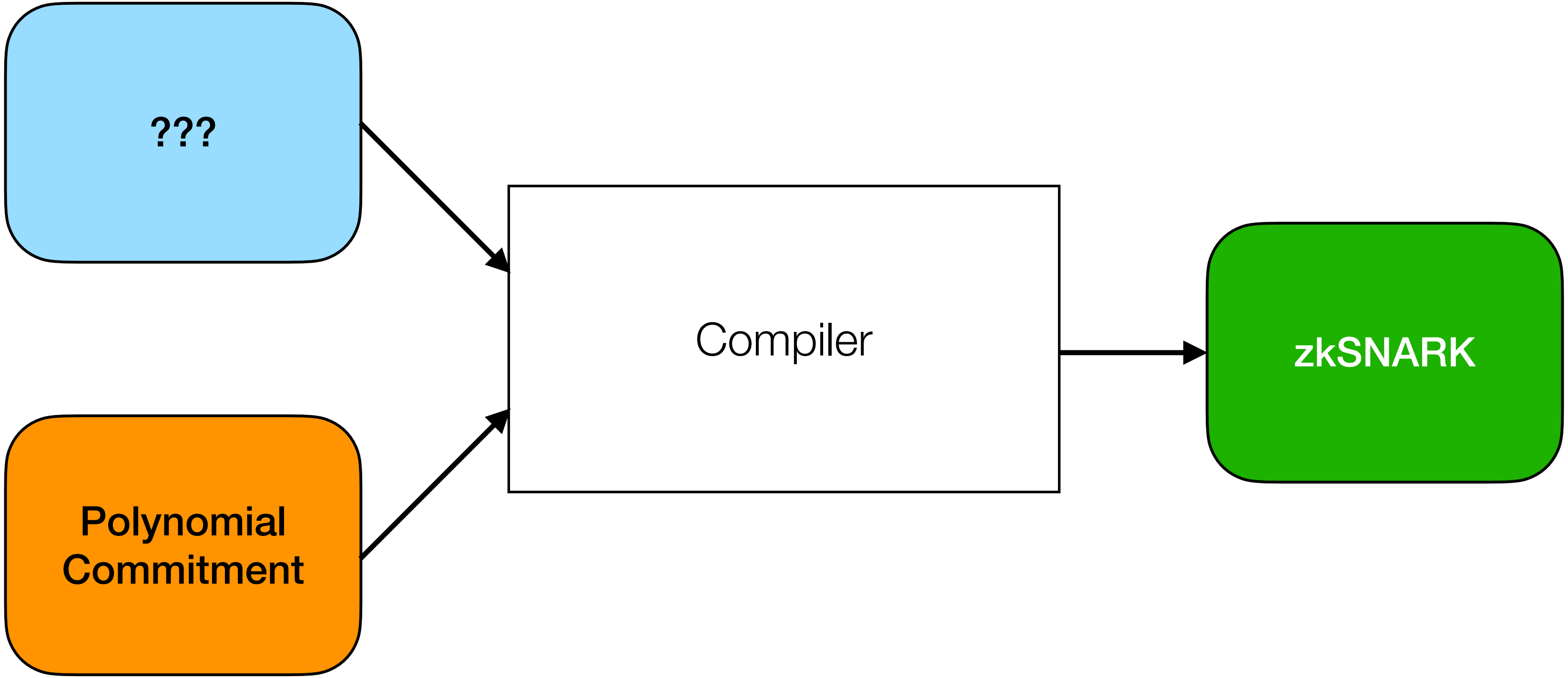
Benedikt Bünz
@benediktbuenz

Replying to @Zac_Aztec

Reed - Solomon code: Polynomial
Zero-Knwoledge Proof Systems: Polynomials
Secret Sharing: Polynomial Evaluations
Identity Testing: Polynomials equal?
FFTs: Polynomials
FRI: FFTs-> Polynomials
SNARK: Polynomials
STARK: SNARK
Security Parameter: Polynomial
Lagrange: Polynomial

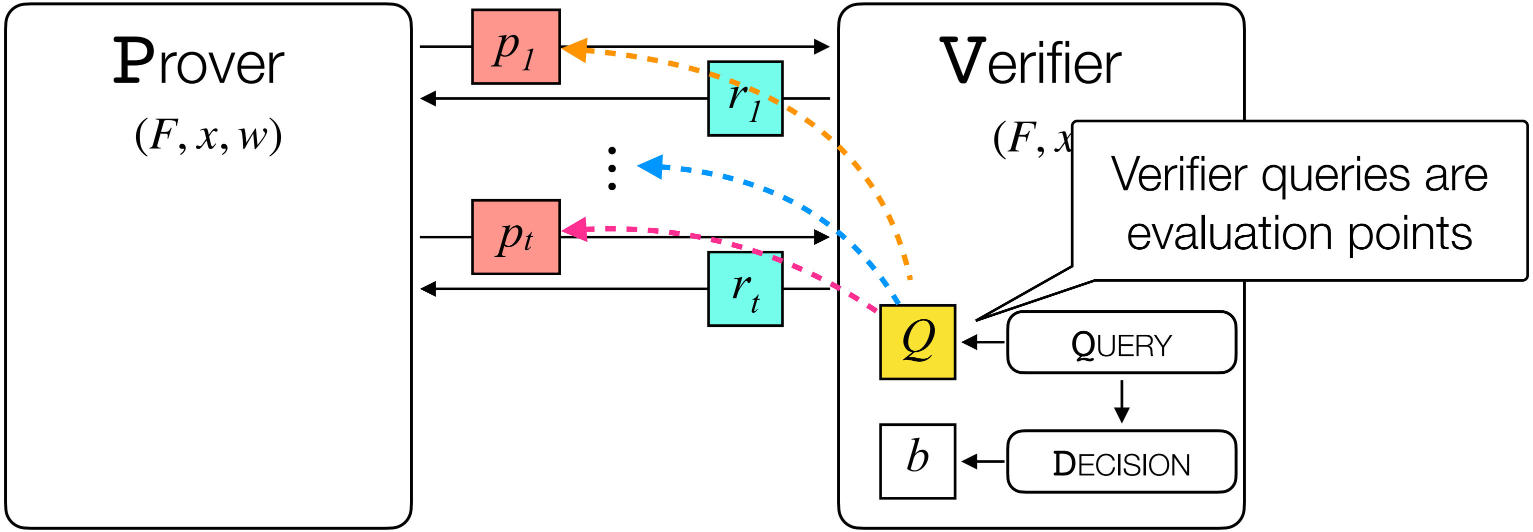
11:28 AM · Oct 25, 2021 · Twitter Web App

Let's pick polynomials



Polynomial Interactive Oracle Proofs

Polynomial IOPs [GWC19, CHMMVW20, BFS20]



- **Completeness**: Whenever $(F, x, w) \in \mathscr{R}$, there is a strategy for $\mathbb{P}$ that outputs **only polynomials**, and which causes $\mathbb{V}$ to accept.
- **Knowledge Soundness**: Whenever $\mathbb{V}$ accepts against a $\mathbb{P}$ that outputs **only polynomials**, then $\mathbb{P}$ "knows" w such that $(F, x, w) \in \mathscr{R}$.
- **Bounded-query ZK**: Whenever $(F, x, w) \in \mathscr{R}$, a $\mathbb{V}$ that makes up to b queries to polys learns nothing about w.

Majority of innovation is in PIOPs

Sonic: Zero-Knowledge SNARKs from Linear-Size Universal and Updatable Structured Reference Strings

Mary Maller
mary.maller.15@ucl.ac.uk
University College London

Sean Bowe
sean@z.cash
Electric Coin Company

Markulf Kohlweiss
mkohlwei@ed.ac.uk
University of Edinburgh
IOHK

Sarah Meiklejohn
s.meiklejohn@ucl.ac.uk
University College London

PlonK: Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge

Ariel Gabizon* — Aztec
Zachary J. Williamson — Aztec
Oana Ciobotaru

Flookup: Fractional decomposition-based lookups in quasi-linear time independent of table size

Ariel Gabizon — Zeta Function Technologies
Dmitry Khovratovich — Ethereum Foundation

Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS

Alessandro Chiesa
alexch@berkeley.edu
UC Berkeley

Yuncong Hu
yuncong_hu@berkeley.edu
UC Berkeley

Mary Maller
mary.maller.15@ucl.ac.uk
UCL

Pratyush Mishra
pratyush@berkeley.edu
UC Berkeley

Psi Vesely
psi@ucsd.edu
UCL

Nicholas Ward
npward@berkeley.edu
UC Berkeley

Spartan: Efficient and general-purpose zkSNARKs without trusted setup

Srinath Setty
Microsoft Research

HyperPlonk: Plonk with Linear-Time Prover and High-Degree Custom Gates

Binyi Chen — Espresso Systems
Benedikt Bünz — Stanford University, Espresso Systems
Dan Boneh — Stanford University
Zhenfei Zhang — Espresso Systems

Lunar: a Toolbox for More Efficient Universal and Updatable zkSNARKs and Commit-and-Prove Extensions

Matteo Campanelli[1], Antonio Faonio[2], Dario Fiore[3], Anaïs Querol[3,4], and Hadrián Rodríguez[3]

Caulk: Lookup Arguments in Sublinear Time

Arantxa Zapico*[1], Vitalik Buterin[2], Dmitry Khovratovich[2], Mary Maller[2], Anca Nitulescu[3], and Mark Simkin[2]

[1] Universitat Pompeu Fabra†
[2] Ethereum Foundation‡
[3] Protocol Labs§

plookup: A simplified polynomial protocol for lookup tables

Ariel Gabizon — Aztec
Zachary J. Williamson — Aztec

cq:* Cached quotients for fast lookups

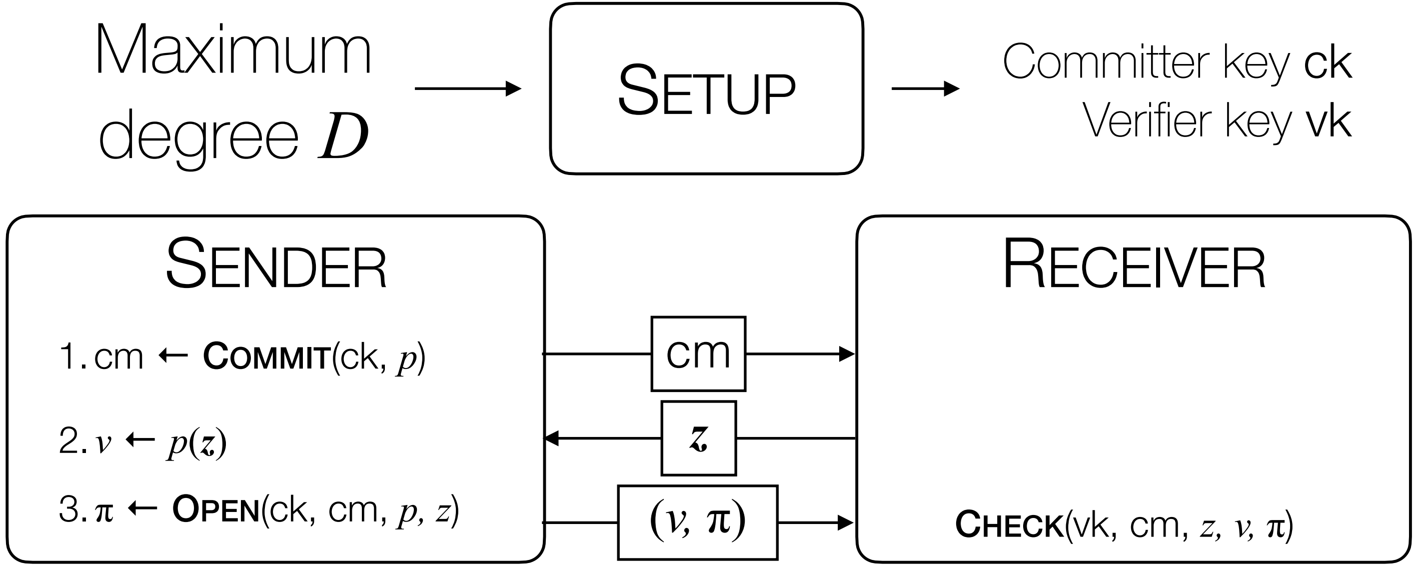
Liam Eagen — Blockstream
Dario Fiore — IMDEA software institute
Ariel Gabizon — Zeta Function Technologies

Baloo: Nearly Optimal Lookup Arguments

Arantxa Zapico*, Ariel Gabizon[3], Dmitry Khovratovich[1], Mary Maller[1], and Carla Ràfols[2]

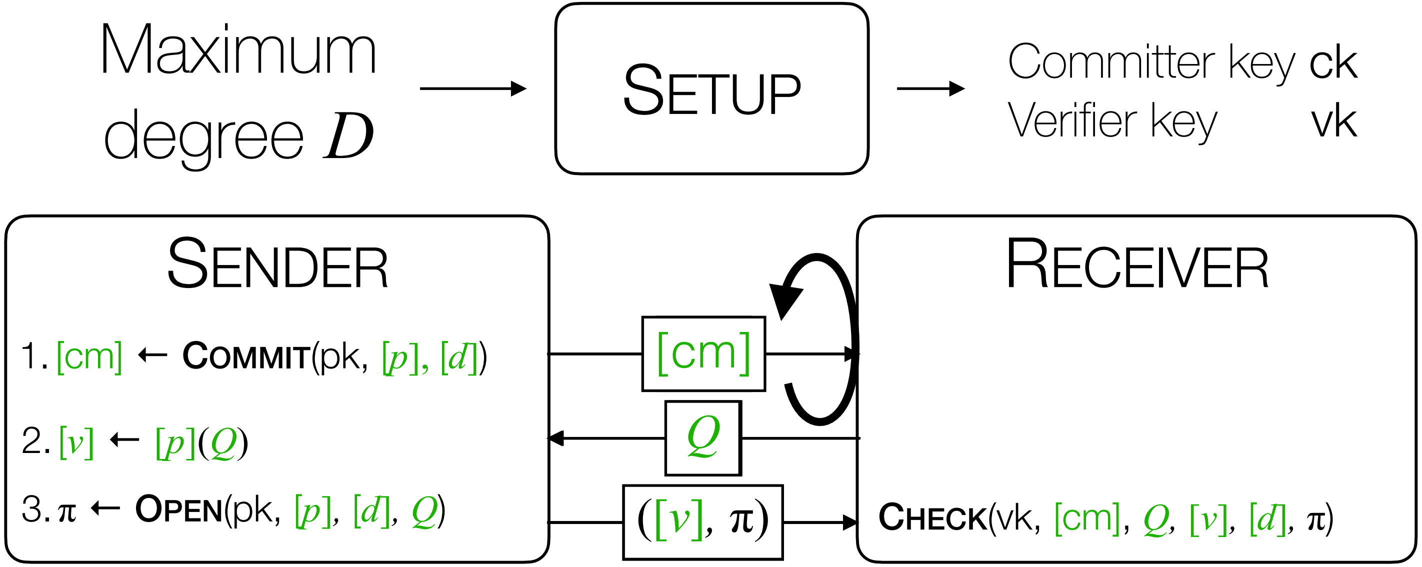
Polynomial Commitments

Polynomial Commitments



- **Completeness**: Whenever $p(z) = v$, **R** accepts.
- **Extractability**: Whenever **R** accepts, **S**’s commitment **cm** “contains” a polynomial p of degree at most D.
- **Hiding**: **cm** and **π** reveal *no* information about p other than v

Polynomial Commitments



For efficiency improvements, you need

- Batch commitment
- Batch opening

A selection of constructions

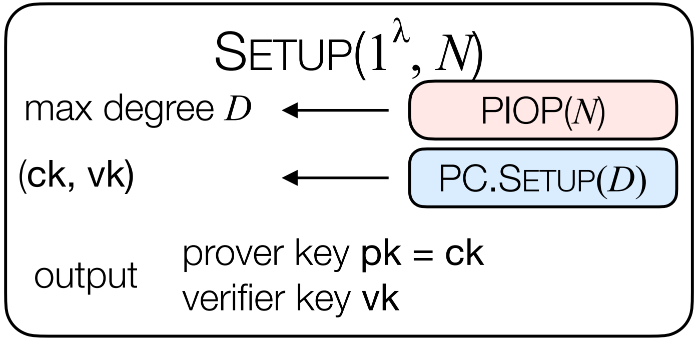
In the last 10 years, several constructions with different

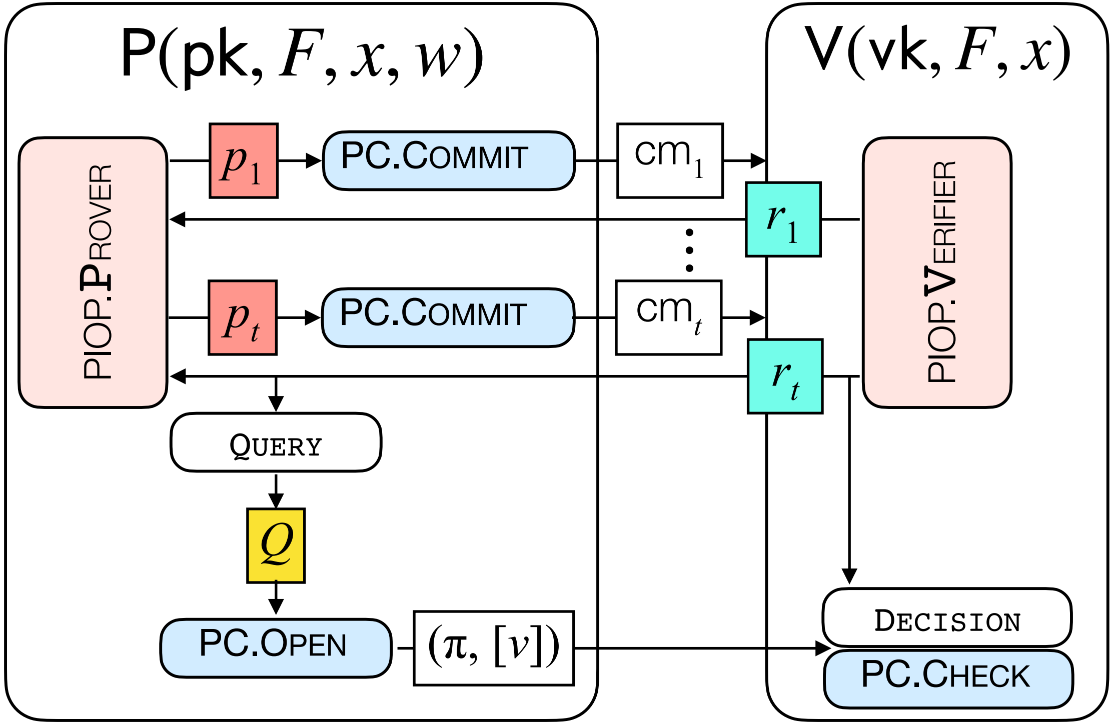
- Cryptographic assumptions
- Prover and verifier efficiency and proof sizes
- Homomorphism and batching properties

	KZG10	PST13	IPA	Hyrax	Dory	BFS20
crypto	Pairings	Pairings	DLog + RO	DLog + RO	Pairing + RO	GUO + RO
# variables	1	m	1	m	1	1
setup type	Private	Private	Public	Public	Public	Public
commitment size	$O(1)$ G	$O(1)$ G	$O(1)$ G	$O(2^{m/2})$ G	$O(1)$ G	$O(1)$ G
proof size	$O(1)$ G	$O(m)$ G	$O(\log d)$ G	$O(2^{m/2})$ G	$O(\log d)$ G	$O(\log d)$ G
verifier time	$O(1)$ G	$O(m)$ G	$O(d)$ G	$O(2^{m/2})$ G	$O(\log d)$ G	$O(\log d)$ G

PIOP + PC = SNARK

PIOPs + PC Schemes → SNARK





+ Fiat—Shamir to get non-interactivity

Properties

- **Completeness**: Follows from completeness of **PC** and **AHP**.
- **Proof of Knowledge**: Whenever **V** accepts but $C(\mathbb{x}, \mathbb{w}) = 0$, we can construct either an adversarial prover against **PIOP**, or an adversary that breaks extractability of **PC**.
- **Zero Knowledge**: Follows from hiding of **PC** and bounded-query ZK of **AHP**.
- **Verifier efficiency**:

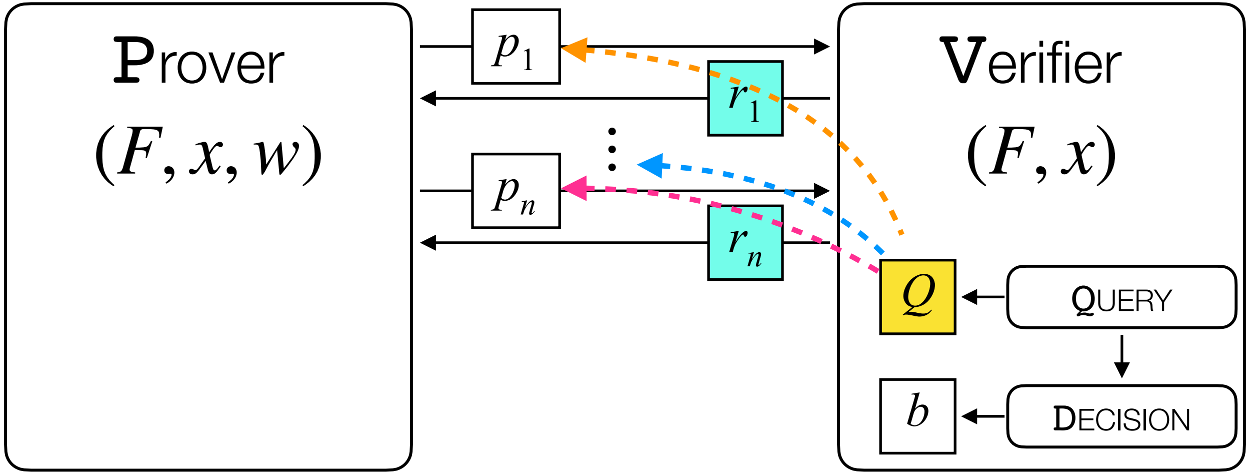
$$T(\textsf{ARG.V}\textsc{erify}) = T(\textsf{PIOP.V}\textsc{erify}) + T(\textsf{PC.C}\textsc{heck})$$

Verifier Complexity of PIOP-based SNARKs

$T(\text{SNARK.V}) = T(\text{CHECK}) + T(\text{PIOP.}\mathbf{V})$

Can make this sublinear (eg: KZG)

What about this?



PIOP Verifier has to **at least** read (F, x)

- When size of $F \ll$ size of computation (eg machine computations), TIME($\mathbf{V}$) **is sublinear.**
- When size of $F =$ size of computation (eg circuit computations), TIME($\mathbf{V}$) **is linear!**

A simple PIOP

Background on polynomials

Polynomial over $\mathbb{F}$:

$p(X) = a_0 + a_1X + \ldots + a_dX^d$ where $a_i \in \mathbb{F}$ and X takes values in $\mathbb{F}$.

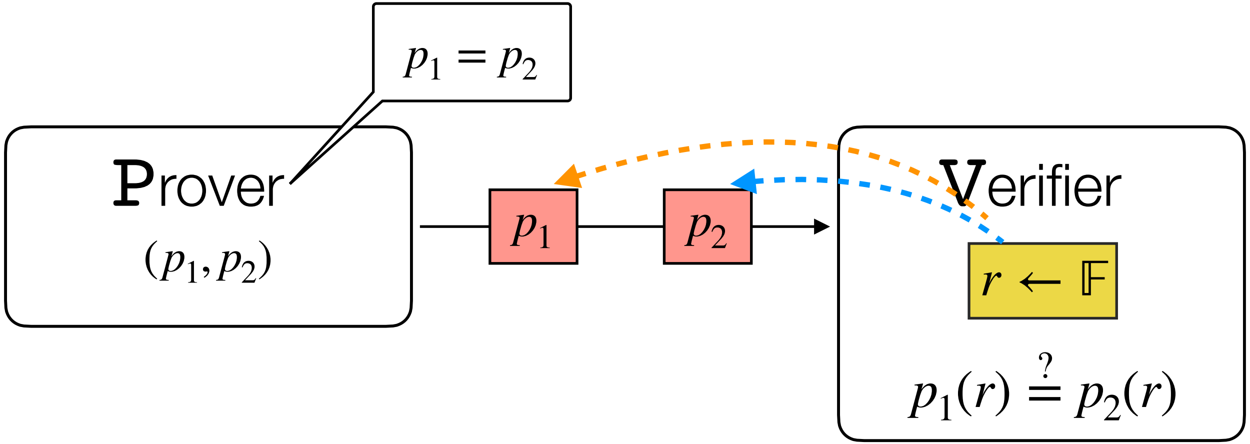
Polynomial Interpolation:

Given a list $A = (a_0, \ldots, a_d)$, and a set $H \subseteq \mathbb{F}$, we can interpolate A over H to obtain $p(X)$ such that $p(h_i) = a_i$ where h_i is the i-th element of H.

Vanishing polynomial:

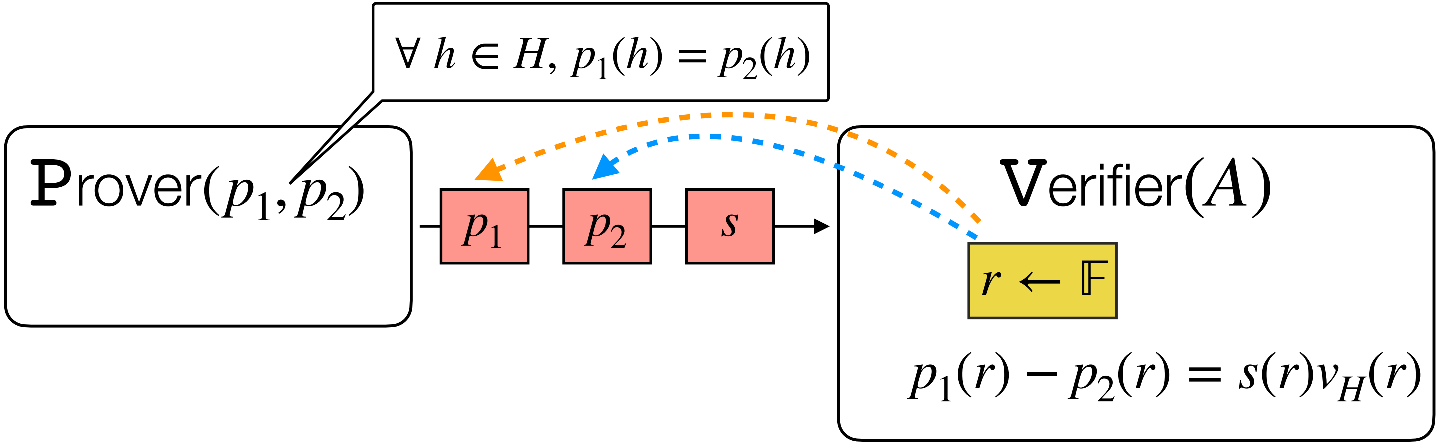
The vanishing polynomial for $H \subseteq \mathbb{F}$ is $v_H(X)$ such that $v_H(h) = 0 \quad \forall\ h \in H$

Warmup: PIOP for Equality



- **Completeness**: If $p_1 = p_2$, then definitely $p_1(r) = p_2(r)$.
- **Soundness**: If $p_1 \neq p_2$, then $p_1(r) = p_2(r) \implies r$ is a root of $q := p_1 - p_2$. But since r is random, this happens with probability $\dfrac{\deg(q)}{|\mathbb{F}|}$

Warmup: PIOP for Equality over Domain



- **Completeness**: If $p_1 = p_2$, then definitely $p_1(r) = p_2(r)$.
- **Soundness**: Define $q := p_1 - p_2$. Then $\forall\, h \in H,\ p_1(h) = p_2(h)$ if and only if $q = s \cdot v_H$. But we can check this via the previous PIOP.