# Transparent SNARKs from DARK Compilers

Ben Fisch
Stanford University, Findora

Joint work with Alan Szepieniec and Benedikt Bünz

# New Result

- **SNARK with no trusted-setup**
  - 10KB for $1M$ gates (128-bit security)
  - Verification under 100 ms
  - $O_\lambda(\log n)$ *size and verification for circuit size n*

Previously 200-600 KB

Previously $O_\lambda(\log^2 n)$

- **New Tool: Polynomial Commitment using Group of Unknown Order**
  - *Trustless setup* using Class Groups

# Recent Comparison

- Implementation comparison in [XZZPS'19]

| | libSNARK [14] | Ligero [6] | Bulletproofs [20] | Hyrax [50] | libSTARK [9] | Aurora [12] | Libra |
|---|---|---|---|---|---|---|---|
| $\mathcal{G}$ | $O(C)$ per-statement trusted setup | no trusted setup | | | | | $O(n)$ one-time trusted setup |
| $\mathcal{P}$ | $O(C \log C)$ | $O(C \log C)$ | $O(C)$ | $O(C \log C)$ | $O(C \log^2 C)$ | $O(C \log C)$ | $O(C)$ |
| $\mathcal{V}$ | $O(1)$ | $O(C)$ | $O(C)$ | $O(\sqrt{n} + d \log C)$ | $O(\log^2 C)$ | $O(C)$ | $O(d \log C)$ |
| $\lvert\pi\rvert$ | $O(1)$ | $O(\sqrt{C})$ | $O(\log C)$ | $O(\sqrt{n} + d \log C)$ | $O(\log^2 C)$ | $O(\log^2 C)$ | $O(d \log C)$ |
| $\mathcal{G}$ | 1027s | NA | | | | | 210s |
| $\mathcal{P}$ | 360s | 400s | 13,000s | 1,041s | 2,022s | 3199s | 201s |
| $\mathcal{V}$ | 0.002s | 4s | 900s | 9.9s | 0.044s | 15.2s | 0.71s |
| $\lvert\pi\rvert$ | 0.13KB | 1,500KB | 5.5KB | 185KB | 395KB | 174.3KB | 51KB |

# Recent Comparison

- Impler

Supersonic is a **transparent** SNARK with log(C) proof size and log(C) verification

(On this table:
<10KB size and <0.1s verification)

| | libSNAR [14] | | | | | | Libra |
|---|---|---|---|---|---|---|---|
| $\mathcal{G}$ | $O(C)$ per-statem trusted se | | | | | | $O(n)$ one-time trusted setup |
| $\mathcal{P}$ | $O(C \log$ | | | | | $C)$ | $O(C)$ |
| $\mathcal{V}$ | $O(1)$ | $O(C)$ | $O(C)$ | $O(\sqrt{n} + d \log C)$ | $O(\log^2 C)$ | $O(C)$ | $O(d \log C)$ |
| $\lvert\pi\rvert$ | $O(1)$ | $O(\sqrt{C})$ | $O(\log C)$ | $O(\sqrt{n} + d \log C)$ | $O(\log^2 C)$ | $O(\log^2 C)$ | $O(d \log C)$ |
| $\mathcal{G}$ | 1027s | NA | | | | | 210s |
| $\mathcal{P}$ | 360s | 400s | 13,000s | 1,041s | 2,022s | 3199s | 201s |
| $\mathcal{V}$ | 0.002s | 4s | 900s | 9.9s | 0.044s | 15.2s | 0.71s |
| $\lvert\pi\rvert$ | 0.13KB | 1,500KB | 5.5KB | 185KB | 395KB | 174.3KB | 51KB |

# The Problems with Trusted Setup

- If subverted, prover can create fake proof
- Can be alleviated through distributed setup
  - Expensive and difficult, but done for Zcash (was broken)
- Low flexibility: New functionality → New setup
  - HAWK: Every smart contract has a new setup

# DARK Proofs and Supersonic

# Polynomial Commitment

$f(X) \in \mathbb{F}_p[X]$ degree at most $\boldsymbol{d}$

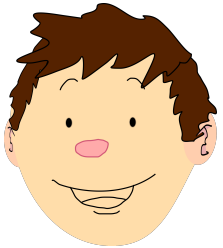
C

small value

z

$\pi, y$

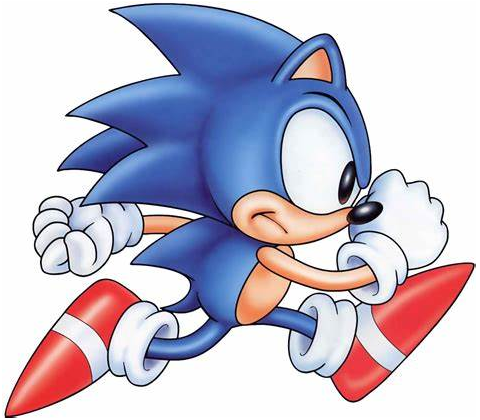
Small Proof that f(z) = y



# Sonic: A new kind of SNARK [MBMK18]

SNARK system using polynomial commitments

One time trusted setup for polynomial commit [KZG10]



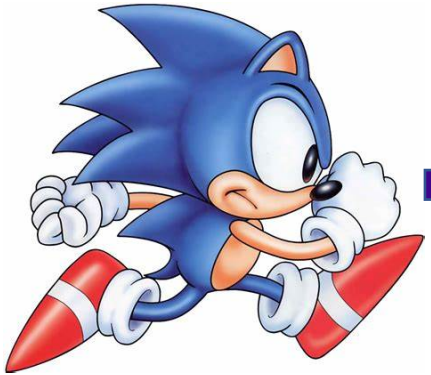
Already improved by
Plonk and Marlin

# DARK Proofs: Polynomial commitment

- Uses Diophantine Arguments of Knowledge
- Class groups for homomorphic integer commitments
- log(d) communication
- log(d) verification

NO TRUSTED SETUP

# Sonic + DARK Proofs = Supersonic



SNARK with short proofs and no trusted setup

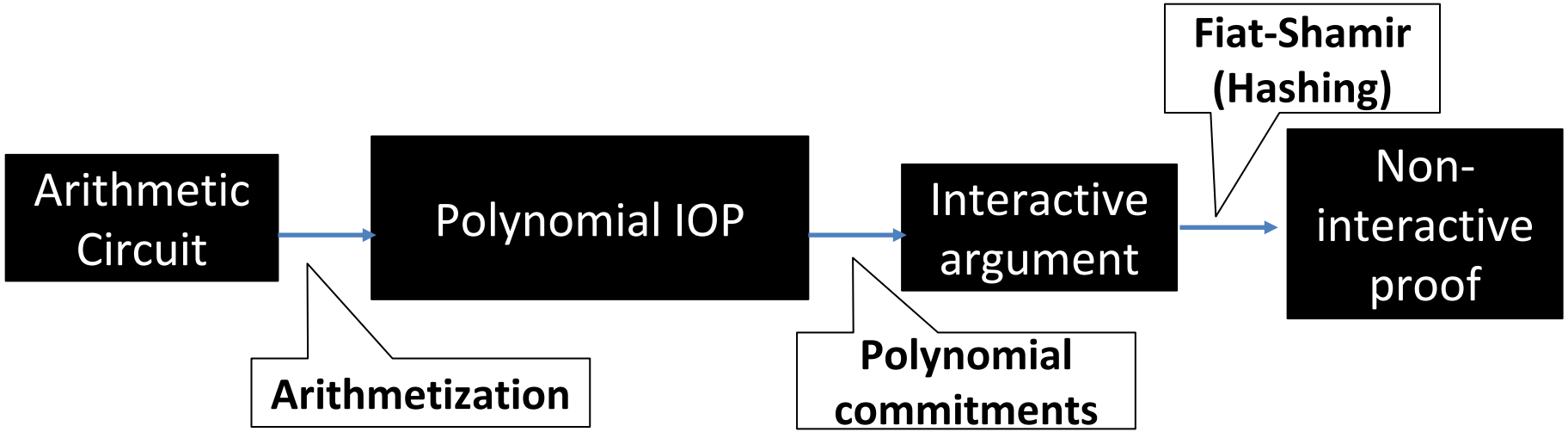
Using PLONK (improvement of Sonic)

Proof size: ～7.8KB for 1 million gates (100-bit security)

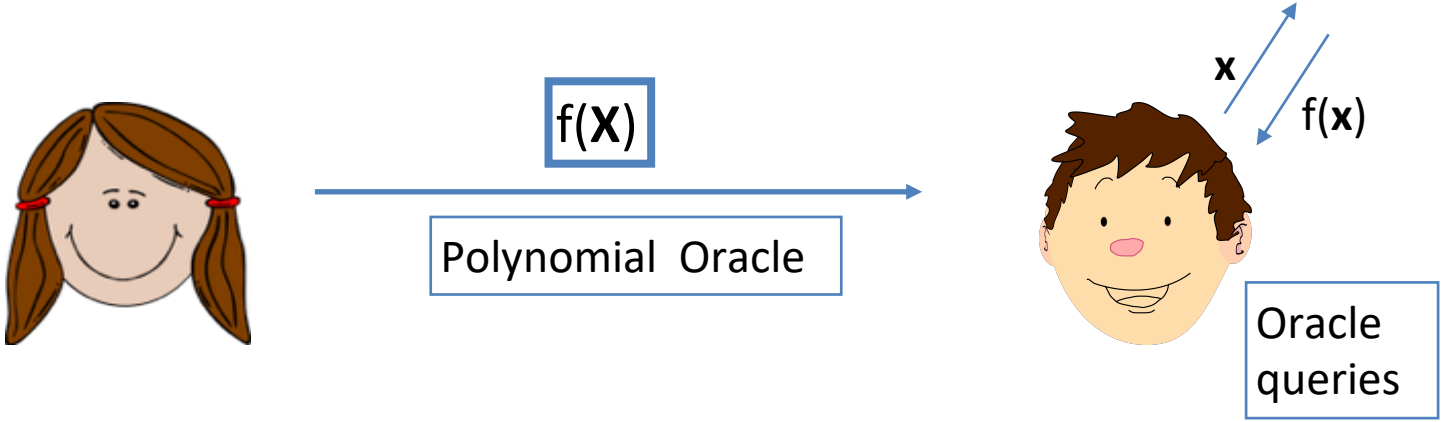
Verification: <100ms

# SNARK Constructions via Polynomial IOPs

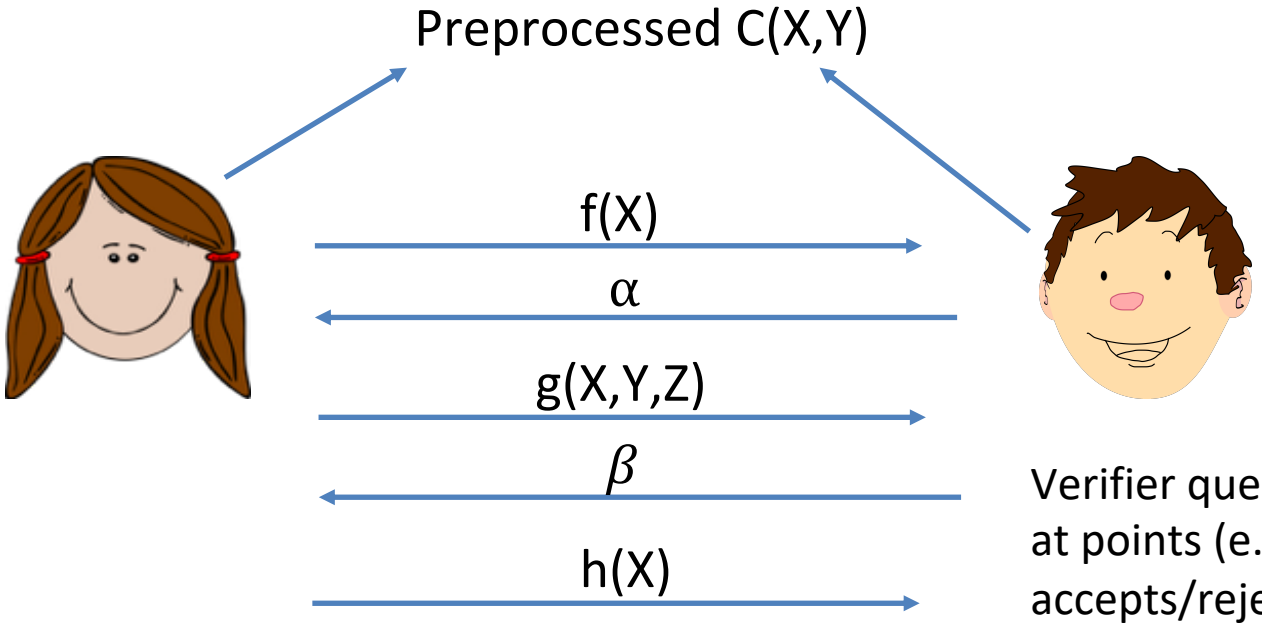
- **Sonic** (MBKM, CCS'19), **Plonk** (GWC, ePrint'19), **Marlin** (CHM+, ePrint'19), **Fractal** (COS, ePrint'19), **Libra** (XZZ+, ePrint'19), **Spartan** (S, ePrint'19) also implicit in **STARKs** (BBHR, C'19) and **Aurora** (BCR+, EC'19).



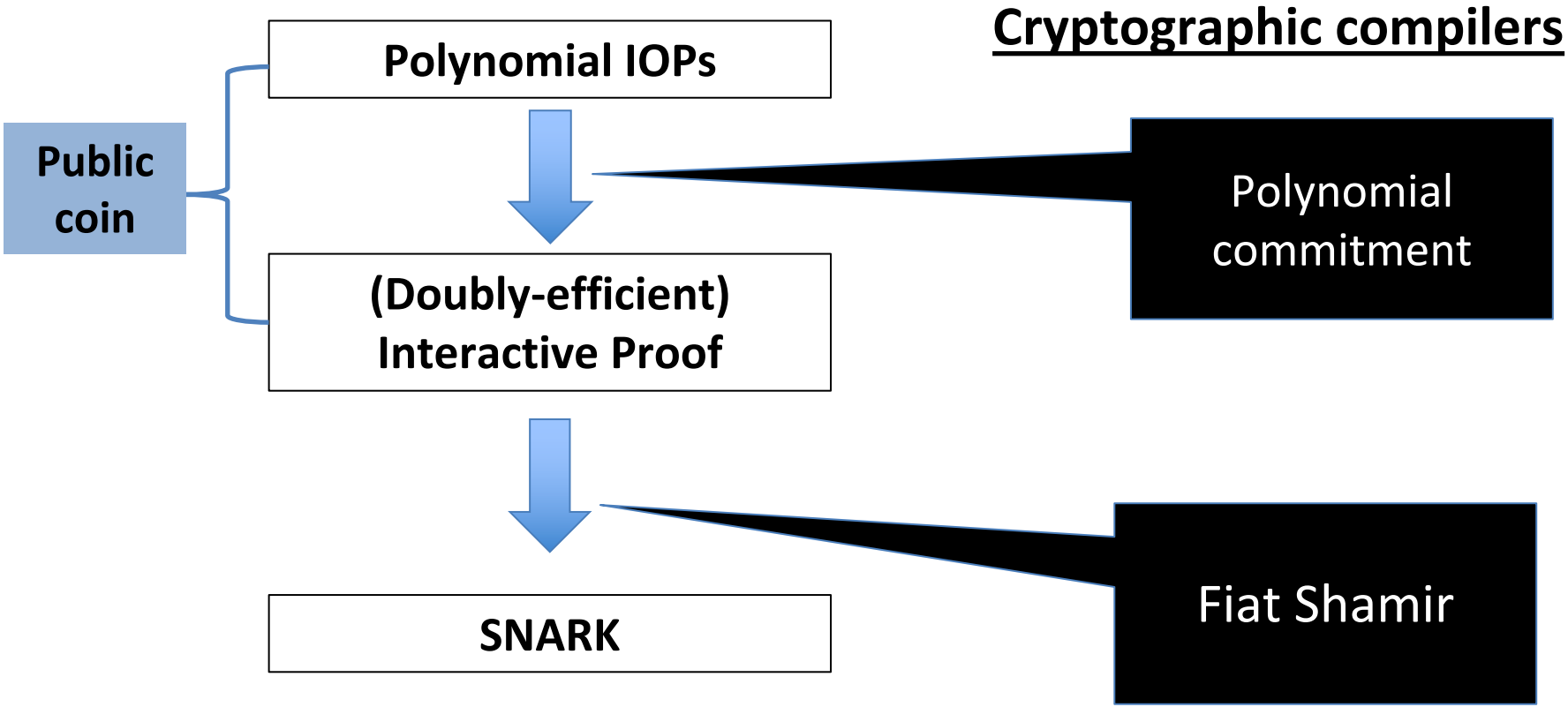
# Polynomial IOPs (A framework for proofs)



# Polynomial IOPs (A framework for proofs)



# Polynomial IOP Compilation



# Polynomial IOP Compilation

**Theorem** [in this work]:
Any HVZK Polynomial IOP can be compiled into an HVZK interactive argument of knowledge using Polynomial Commitments

Similar theorem shown in **Marlin** (CHM+, ePrint'19)

# Sonic: Polynomial IOP for NP

**Theorem [MBKM19]:**

- There exists a **2-round** polynomial IOP for any NP relation R (with multiplicative complexity $n$) that makes **1 *bivariate*** and **6 *univariate*** *queries* to degree $2n$ polynomial oracles. *Also preprocesses a single bivariate polynomial oracle.*

- Transforms to **5-round** polynomial IOP with **39 *univariate*** oracle queries overall at **12 distinct points** to 27 univariate degree 2n polynomial oracles. The *preprocessing checks 12 of the polynomials*.

# PLONK: Polynomial IOP for NP

**Theorem [GWC19]:**

- There is a **3-round** polynomial IOP for any NP relation R (with arithmetic complexity $n$) that makes **12** *queries* overall to 12 univariate degree n polynomial oracles. The total number of **distinct query points is 2**. The *preprocessing checks 7 of the polynomials*.

# Marlin: Polynomial IOP for NP

**Theorem [MBKM19]:**

- There is a **4-round** polynomial IOP for any NP relation R (with R1CS complexity $n$) that makes **20 queries** *at* ***3 distinct*** **query points** to 19 univariate polynomials (max degree 6n). The preprocessing checks 8 univariate degree n polynomials.

# Supersonic with Different IOPs

128-bit security

| Polynomial IOP | Polynomials | Eval points | \|SNARK\| | concrete size |
|---|---|---|---|---|
| Sonic [MBKM19] | 12 in pp + 15 | 12 | $(15 + 2\log_2(n))\mathbb{G}$ $+(12 + 13\log_2(n))\mathbb{Z}_p$ | 15.3 KB |
| PLONK [GWC19] | 7 in pp + 7 | 2 | $7 + 2\log_2(n)\mathbb{G}$ $+ (2 + 3\log_2(n))\mathbb{Z}_p$ | 10.1 KB |
| Marlin [CHM+19] | 9 in pp + 10 | 3 | $10 + 2\log_2(6n)\mathbb{G}$ $+ (3 + 4\log_2(6n))\mathbb{Z}_p$ | 12.3 KB |

# Comparison of Polynomial Commitments

| Scheme | Transp. | $\lvert pp \rvert$ | Prover | Verifier | $\lvert \pi \rvert$ |
|---|---|---|---|---|---|
| DARK *(this work)* | yes | $O(1)$ | $O(d^\mu \mu \log(d))$ EXP | $2\mu \log(d)$ EXP | $2\mu \log(d)$ $\mathbb{G}_U$ |
| Based on Pairings | no | $d^\mu$ $\mathbb{G}_B$ | $O(d^\mu)$ EXP | $\mu$ Pairing | $\mu$ $\mathbb{G}_B$ |
| [BCC+16b, $\sqrt{\cdot}$] | yes | $\sqrt{d^\mu}\mathbb{G}_P$ | $O(d^\mu)$ EXP | $O(\sqrt{d^\mu})$EXP | $O(\sqrt{d^\mu})$ $\mathbb{G}_P$ |
| Bulletproofs | yes | $2d^\mu\mathbb{G}_P$ | $O(d^\mu)$ EXP | $O(d^\mu)$EXP | $2\mu \log(d)$ $\mathbb{G}_P$ |
| FRI ($\mu = 1$) | yes | $O(1)$ | $O(\lambda d)$ H | $O(\lambda \log^2(d))$ H | $O(\lambda \log^2(d))$ H |

# Comparison of SNARKs

| Scheme | Transp. | \|pp\| | Prover | Verifier | $\|\pi\|$ | $n = 2^{20}$ |
|---|---|---|---|---|---|---|
| Supersonic | yes | $O(1)$ | $O(n \log(n))$ EXP | $O(\log(n))$ EXP | $O(\log(n)\ \mathbb{G}_U)$ | 10.1KB |
| PLONK [GWC19] | no | $2n\ \mathbb{G}_B$ | $O(n)$ EXP | 1 Pairing | $O(1)\ \mathbb{G}_B$ | 720b |
| Groth16 [Gro16] | no | $2n\ \mathbb{G}_B$ | $O(n)$ EXP | 1 Pairing | $O(1)\ \mathbb{G}_B$ | 192b |
| BP [BBB+18] | yes | $2n\ \mathbb{G}_P$ | $O(n)$ EXP | $O(n)$ EXP | $2\log(n)\ \mathbb{G}_P$ | 1.7KB |
| STARK | yes | $O(1)$ | $O(\lambda T)$ H | $O(\lambda \log^2(T))$ H | $O(\lambda \log^2(T))$ H | 600 KB |

# Main Construction

# New Polynomial Commitment

**Step 1**

**Integer encoding $f \in \mathbb{F}_p[X]$ degree at most d:**

- Represent $\hat{f}(X) = a_0 + a_1 X + \cdots + a_d X^d$ , $a_{\mathrm{i}} \in [0, p)$
- Choose $q \geq p$
- Output $\hat{f}(q) \in \mathbb{Z}$

Still same size as $f$

- Example: $\hat{f}(X) = 4X^3 + 2X^2 + X + 3,\ p = 5$
- $q = 10, \hat{f}(q) = 4213$

# New Polynomial Commitment

**Step 1**

## Integer encoding $f \in \mathbb{F}_p[X]$ degree at most d:

**Fact 1:** Every integer in $[0, q^{d+1}]$ is uniquely decodable to integer polynomial with **positive** coefficients < q

- For q > p: integer $\hat{f}(q)$ uniquely encodes $f \in \mathbb{F}_p[X]$

**Fact 2:** Every integer in $[-\frac{q^{d+1}}{2}, \frac{q^{d+1}}{2}]$ is uniquely decodable to $h \in \mathbb{Z}[X]$ with coefficients of **absolute value** < q/2

# Integer Encoding

**Step 2**

## Homomorphic properties of encoding

**Additive homomorphism:**

- $f(q) + g(q) = (f + g)(q)$ if q is "large enough"
- $f(10) = 4231, \mathrm{g}(10) = 1443\ f(10) + g(10) = 5674$
- $h(X) = 5X^3 + 6X^2 + 7X + 4, h(10) = 5674$

**Monomial homomorphism:**

- $X^k \cdot f(X) \rightarrow q^k f(q)$
- $\mathrm{X}^2 \cdot h(X) = 5X^5 + 6X^4 + 7X^3 + 4\mathrm{X}^2$
- $100 \cdot h(10) = 567400$

# Groups of Unknown Order

**Step 3**

**Commit $f \in \mathbb{F}_p[X]$ degree at most d**

**Setup**$(1^\lambda) \rightarrow$ pp: $= [\mathbb{G}, g \in \mathbb{G}]$

$\mathbb{G}$ has unknown order

$\mathbb{Z} \rightarrow \mathbb{G}$: $\boldsymbol{x \rightarrow C = g^x}$, **homomorphic map** ($\boldsymbol{g^x g^y = g^{x+y}}$)

**Commit**(pp, $f$):

- Lift $f$ to $\hat{f} \in \mathbb{Z}[X]$ \\ coeff. in [0, p)
- $C \leftarrow g^{\hat{f}(q)}$ \\ q/2 ≥ p

# Class Groups [BW88,L12]

CL(Δ) – Class group of quadratic number field $\mathbb{Q}(\sqrt{\Delta})$

$\Delta = -p$ (a large random prime)

**Properties**

- Element representation: integer pairs $(a, b)$

$$|a| \approx |b| \approx \sqrt{-\Delta}$$

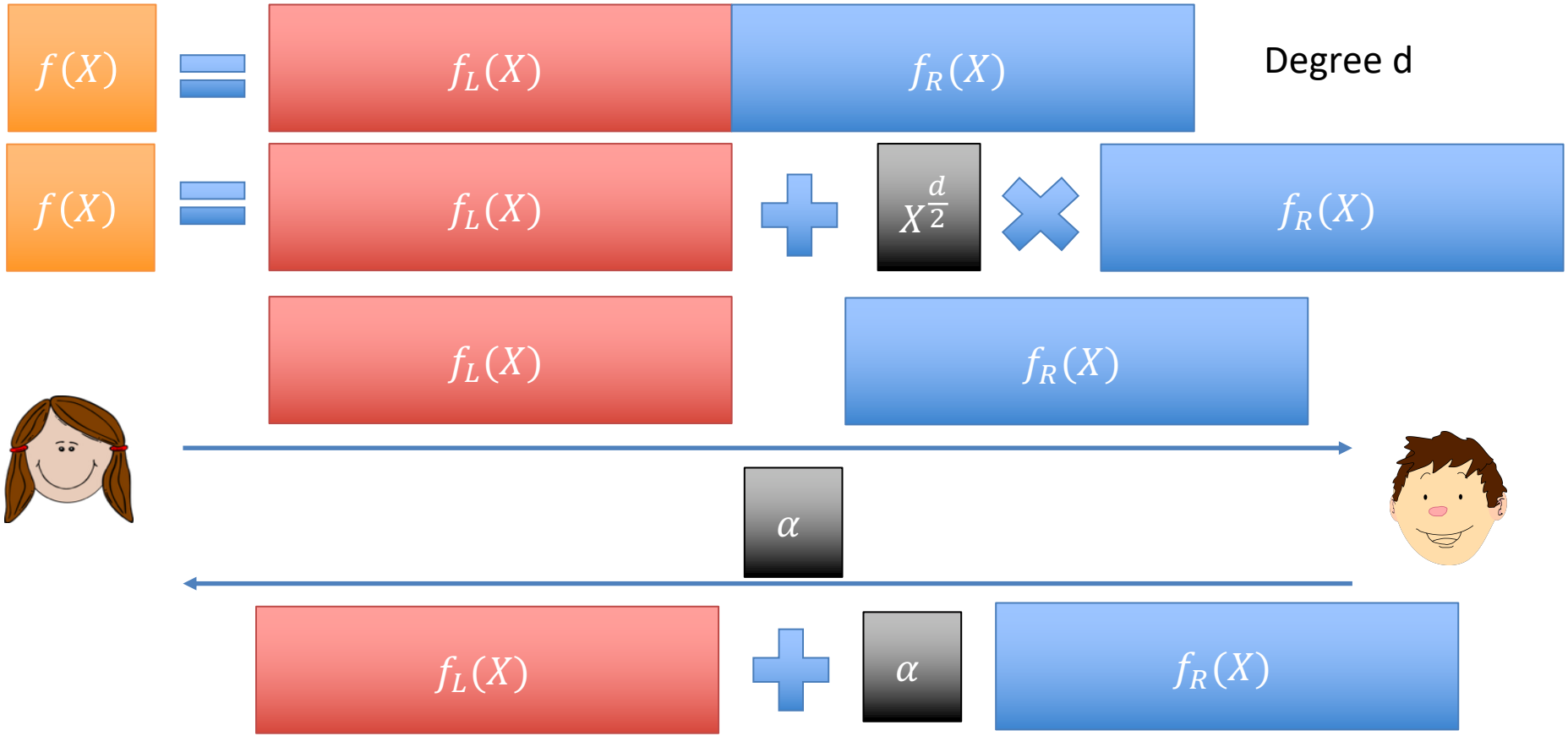
- Tasks believed to be hard to compute:

  **Odd prime roots** **Group order**

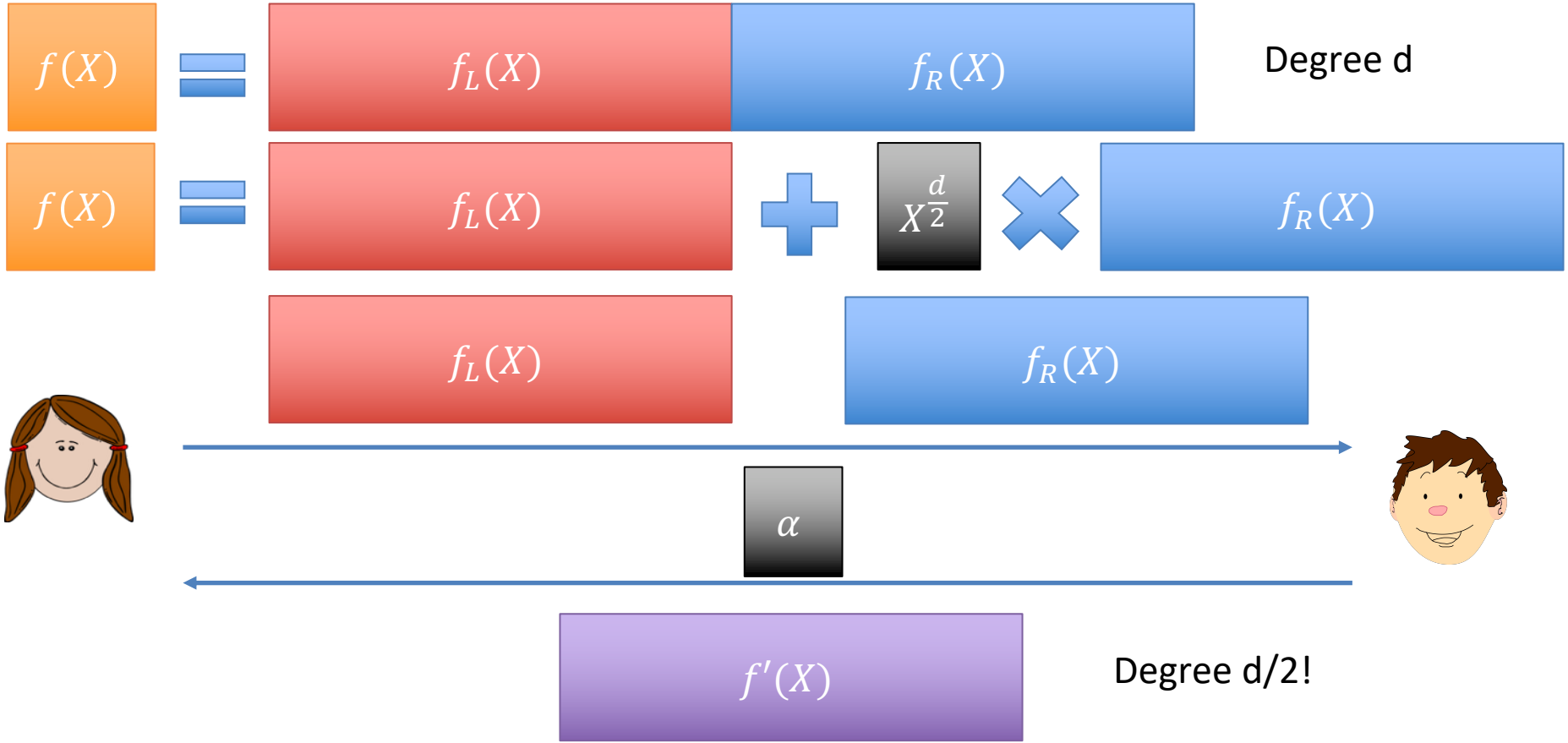
- $\Delta \approx 1200\ bits$ ⇒ 100 bit security

No trusted setup

# Evaluation protocol (DARK) intuition



# Evaluation protocol (DARK) intuition



# Evaluation protocol (DARK)

$C: g^{f(q)}$ = $C_L = g^{f_L(q)}$ $C_R = g^{f_R(q)}$ Degree d

$C: g^{f(q)}$ = $C_L = g^{f_L(q)}$ + $q^{\frac{d}{2}}$ × $C_R = g^{f_R(q)}$

$g^{f_L(q)}$ $g^{f_R(q)}$

$\alpha$

$g^{f'(q)}$ Degree d/2!

# Evaluation protocol (DARK)

$C: g^{f(q)}$ = $C_L = g^{f_L(q)}$ $C_R = g^{f_R(q)}$

$y = f(z) mod\ p$

$C: g^{f(q)}$ = $C_L = g^{f_L(q)}$ + $q^{\frac{d}{2}}$ × $C_R = g^{f_R(q)}$

$g^{f_L(q)}$ $g^{f_R(q)}$

$y_L \leftarrow f_L(z) \in \mathbb{Z}_p$

$y_R \leftarrow f_R(z) \in \mathbb{Z}_p$

$\alpha$

$C =? C_L C_R^{q^{d/2}}$

$y' \leftarrow y_L + \alpha \cdot y_R$

$g^{f'(q)}$

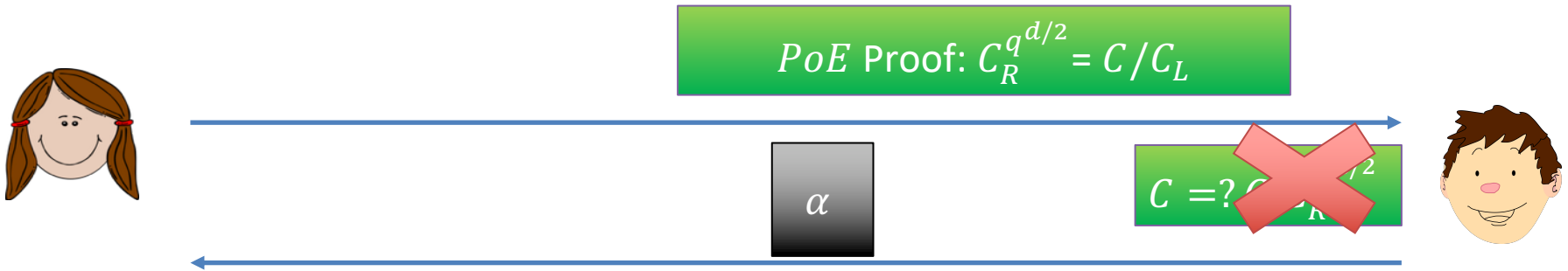
$y =? y_L + z^{q^{d/2}} y_R$

# Evaluation protocol (DARK)

**Proof of Exponentiation (Wesolowski 2018)**

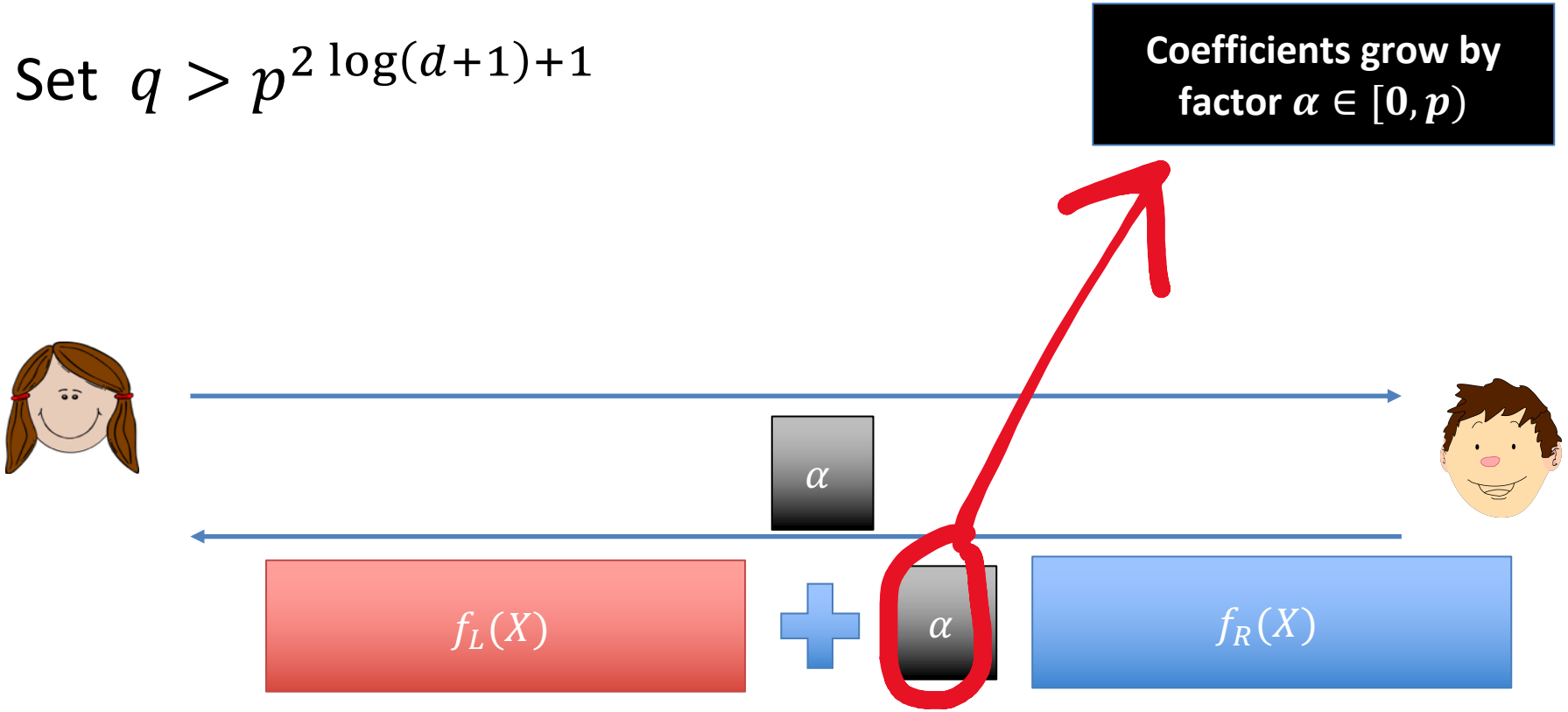
(Same trick used for VDFs)

Verifier does two 128-bit group EXP



# Evaluation protocol (DARK)

Set $q > p^{2\log(d+1)+1}$



# Final Step

$f(X) = f_0$

$C: g^{f(q)}$

$y$

Send $f_0$ (degree 0)

2 log d group elements
2 log d field elements
Lots of batching tricks

Checks that:

1) $\boldsymbol{f_0 < p^{\log(d+1)+1}}$
2) $C = g^{f_0}$
3) $f_0 = y$

# Thanks!

https://eprint.iacr.org/2019/1229

findora

# Optimizations

**Evaluation: open C $= g^{f(q)}$ at $z \in \mathbb{Z}_p$ ; $f(z) = y$**

1. Prover sends $C_L$, $C_R$, $y_0 = f_L(z)$ $y_1 = f_R(z)$
2. Proof of exponentiation: $C = C_L C_R^{q^{d'}}$
3. Verifier checks $y_0 + z^{d'+1} y_1 = y \; mod \; p$
4. Verifier sends $\alpha$
5. Eval($\boldsymbol{C_L^{\alpha} C_R}$, z, $\boldsymbol{y_0} + \alpha \boldsymbol{y_1}$,d'; $\alpha f_L + f_R$)

2log(d) $\mathbb{G}$
2log(d) $\mathbb{Z}_p$

# Optimizations

**Evaluation: open C** $= \boldsymbol{g^{f(q)}}$ **at** $\vec{\boldsymbol{z}} \in \mathbb{Z}_p^n$ ; $f(\vec{\boldsymbol{z}}) = \vec{y}$

1. Prover sends $C_R, \vec{y} = f_R(\vec{z})$
2. Proof of exponentiation: $C = C_L C_R^{q^{d'}}$
3. Verifier computes $\vec{y}_0 = \vec{y} - \vec{z}^{d'+1}\vec{y}_1 \ mod\ p$
4. Verifier sends $\alpha$
5. Eval($\boldsymbol{C_L^{\alpha} C_R}$, z, $\boldsymbol{y_0} + \alpha \boldsymbol{y_1}$,d'; $\alpha f_L + f_R$)

2log(d) $\mathbb{G}$
(1+n) log(d) $\mathbb{Z}_p$

# Optimizations

**Evaluation: open $\mathbf{C_1} = \boldsymbol{g}^{\boldsymbol{f}(\boldsymbol{q})}$, $\mathbf{C_2} = \boldsymbol{g}^{\boldsymbol{h}(\boldsymbol{q})}$ at $\boldsymbol{z} \in \mathbb{Z}_p$**
such that $f(z) = y_1$ $h(z) = y_2$

1. Prover sends $C_1$ , $C_2$
2. Verifier sends $\alpha$
3. Eval($\boldsymbol{C_1^{\alpha} C_2}$, z, $\boldsymbol{y_0} + \alpha \boldsymbol{y_1}$,d'; $\alpha f(X) + g(X)$)

2log(d) $\mathbb{G}$
2log(d) $\mathbb{Z}_p$

In general 2log(d) $\mathbb{G} + (k+1)\log(d)\,\mathbb{Z}_p$
for n polys at k evaluation points