CONSENSYS

Linea

gnark & 2-chain aggregation for Linea

Proof Day: NYC Edition

Youssef El Housni

Team

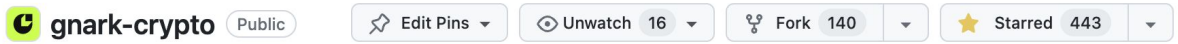
Who?

- Arya Pourtabatabaie
- Ivo Kubjas
- *Youssef El Housni*
- Thomas Piellard
- Gautam Botrel

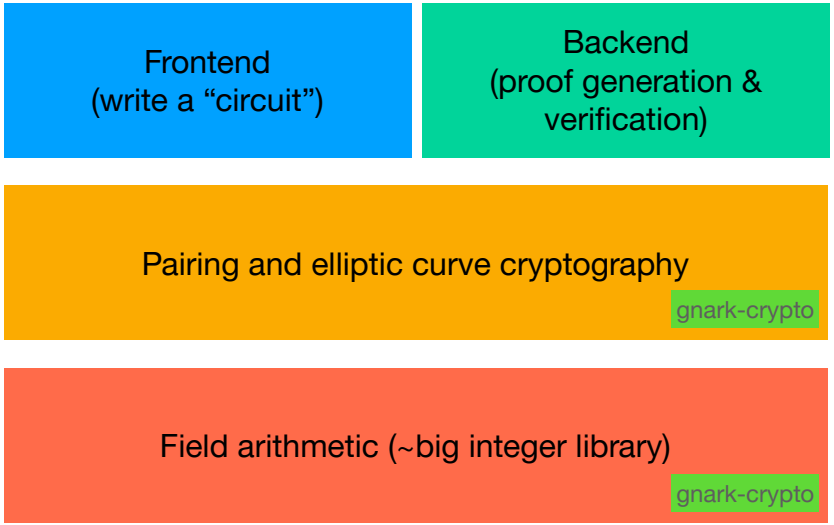
What?

We're building gnark, a fast and easy to use open source zkSNARK library, in Go.

and gnark-crypto, a fast cryptographic library, in Go.



gnark under the hood



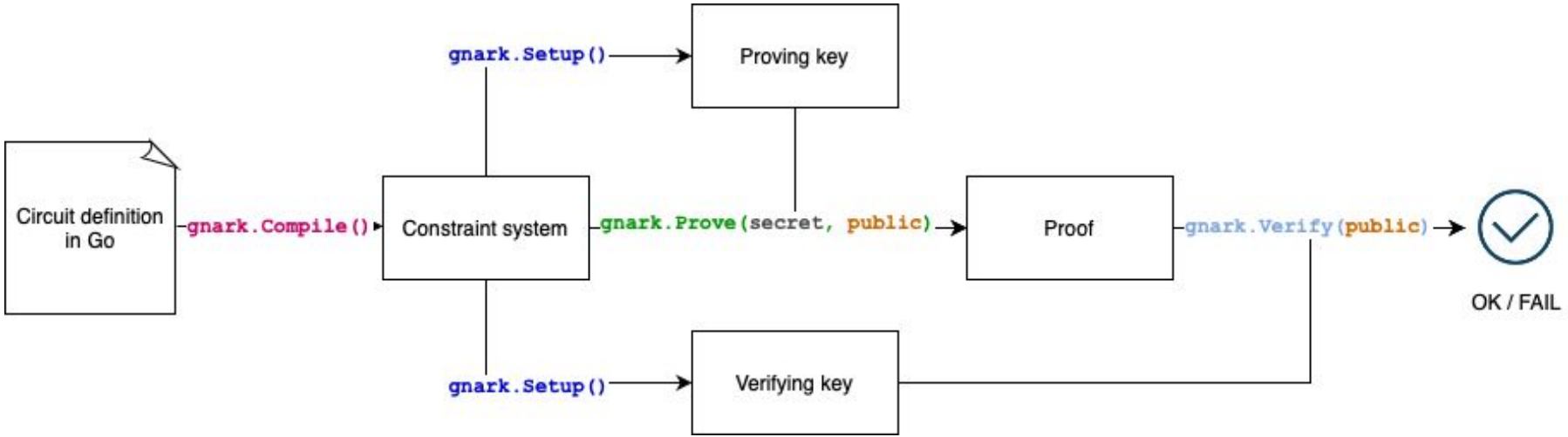
- Groth16, PLONK w/ KZG (or FRI)
- std: hashes, signatures, pairings, commitments…
- Native and non-native field arithmetic

- BN254, BLS12-381, BLS12-377/BW6-761, BLS24…
- Fast cryptographic primitives (MSM, pairings,...)
- KZG, FRI, Plookup…
- Sumcheck (GKR)

- 768-bit, 384-bit, 256-bit, goldilocks… on multi-targets
- SotA mul, Pornin’s inverse, FFT…

gnark workflow

```
pk, vk, err := groth16.Setup(ccs)
proof, err := groth16.Prove(ccs, pk, witness)
err := groth16.Verify(proof, vk, publicWitness)
```



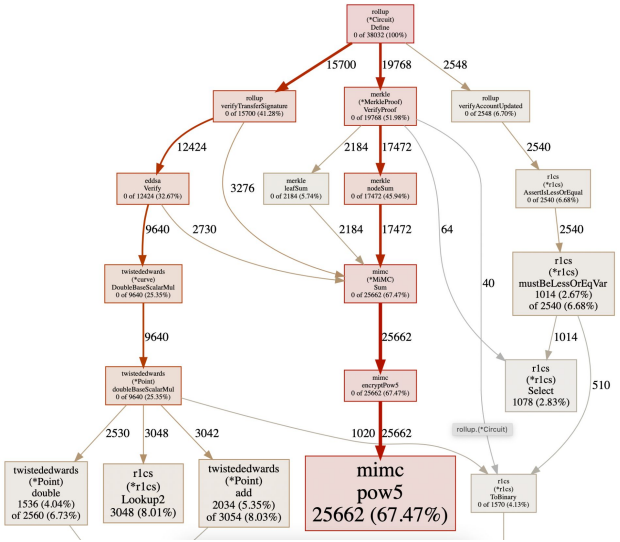

```
ccs, err = frontend.Compile(ecc.BN254.ScalarField(), r1cs.NewBuilder, &c)
ccs, err = frontend.Compile(ecc.BLS12_381.ScalarField(), scs.NewBuilder, &c)
```

gnark features

gnark

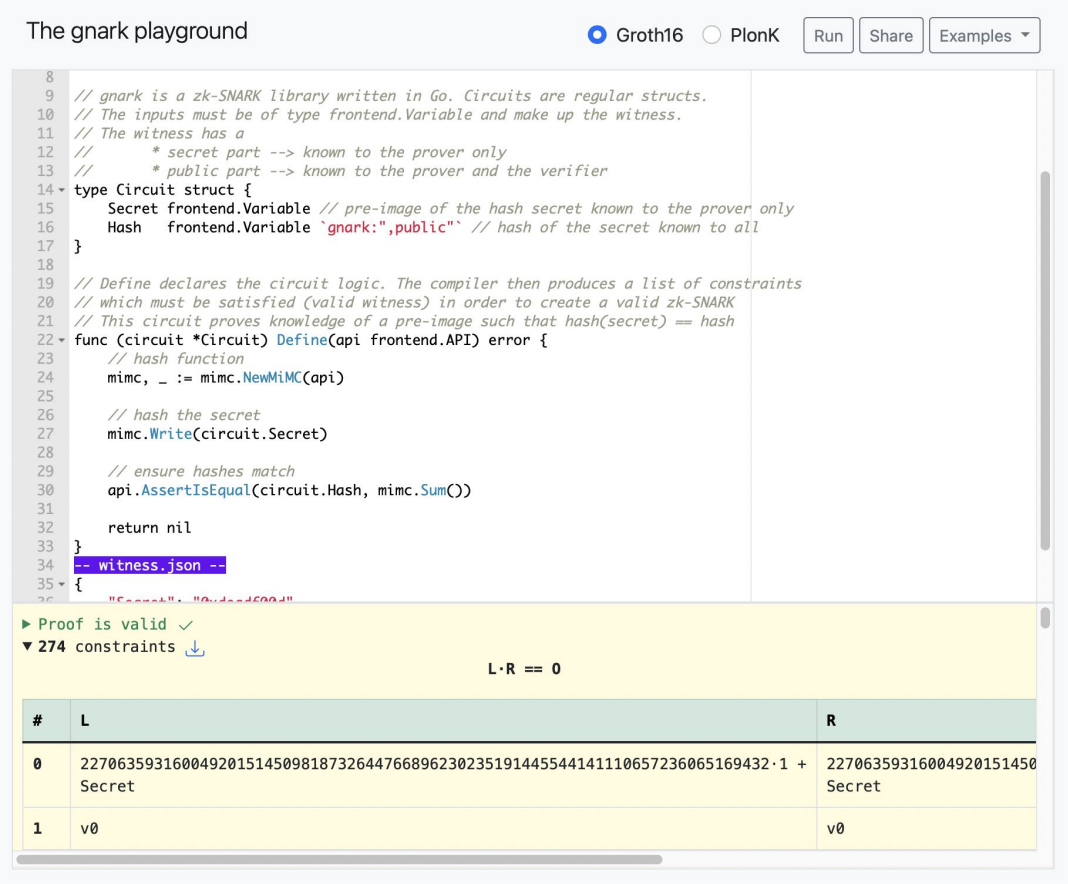
- No DSL, plain Go, **no dependencies**
- Compiles large circuit (seconds)
- Playground, constraints profiler, …
- multiple curves and backends
- MPC trusted setup
- Web2 and Solidity verification
- Several packages audited (by Algorand, EF, Worldcoin and Linea)
- One code base which performs well on:
  - Server (CPU)
  - Mobile (70% first place zprize)

```
func (circuit *Circuit) Define(api frontend.API) error {
    // compute x**3 and store it in the local variable x3.
    x3 := api.Mul(circuit.X, circuit.X, circuit.X)

    // compute x**3 + x + 5 and store it in the local variable res
    res := api.Add(x3, circuit.X, 5)

    // assert that the statement x**3 + x + 5 == y is true.
    api.AssertIsEqual(circuit.Y, res)
```



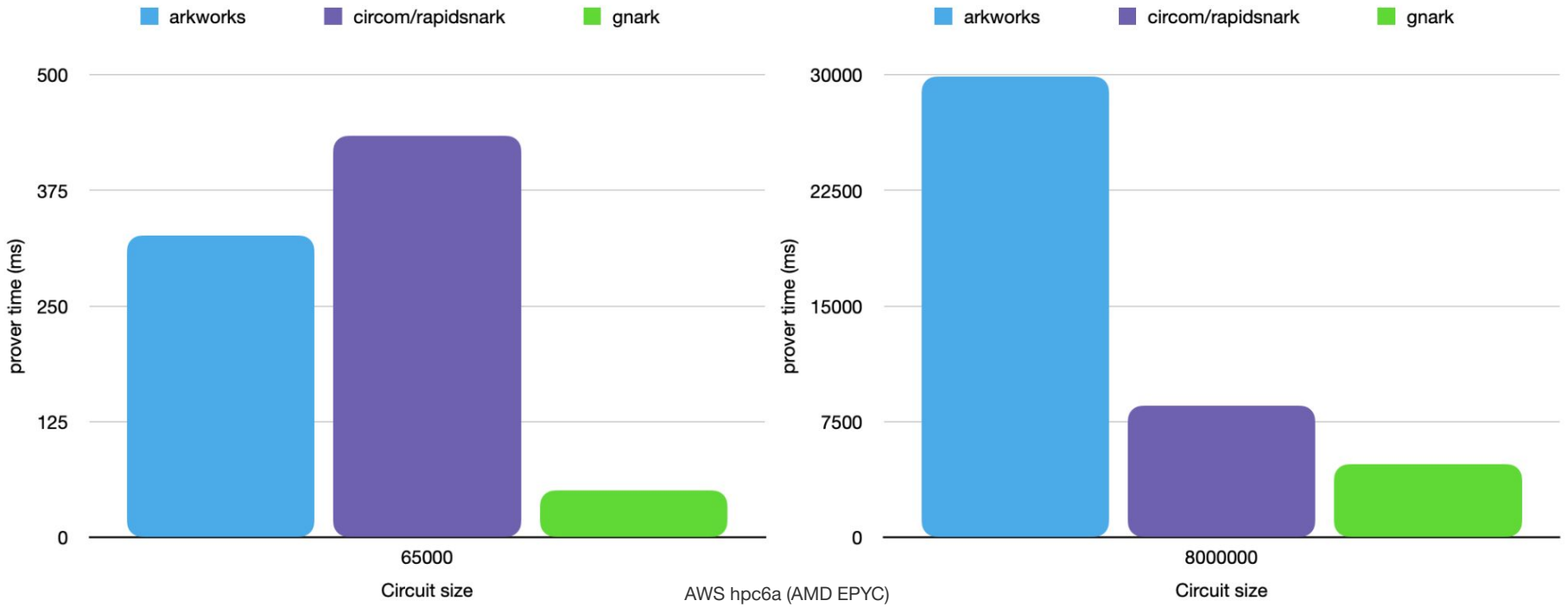
Constraints profiler

gnark playground: *play.gnark.io*



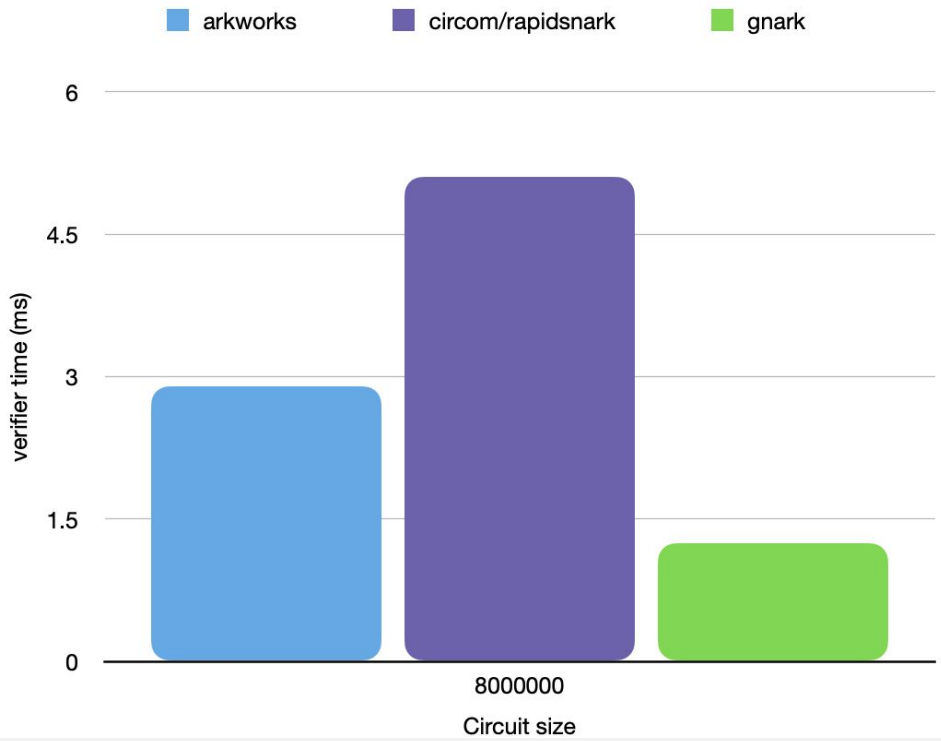
gnark is very fast

Groth16 SNARK prover on BN254: MSM, FFT, parallelism



gnark is very fast

Groth16 SNARK verifier: Pairing on BN254

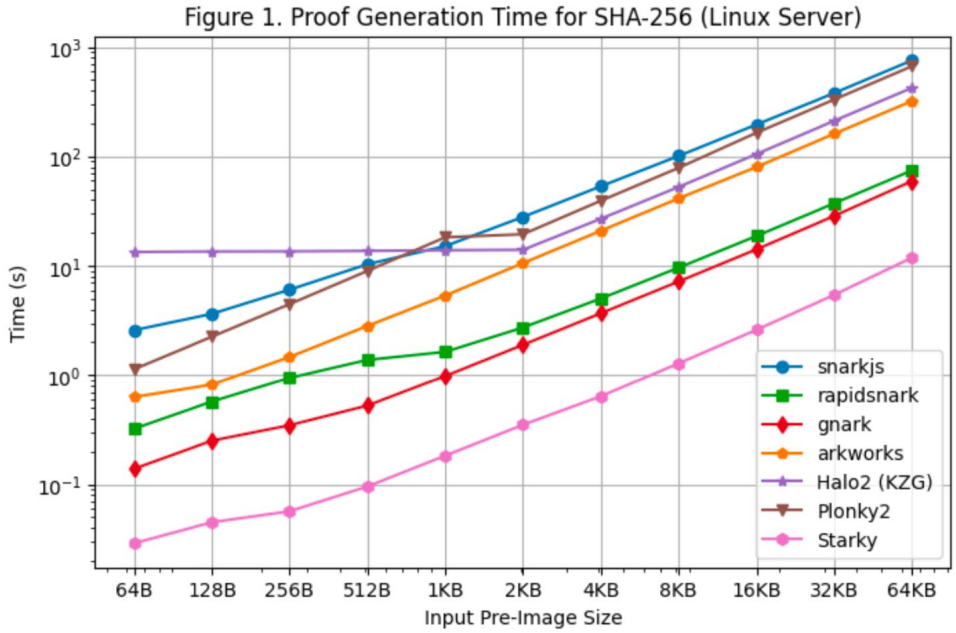


AWS hpc6a (AMD EPYC)

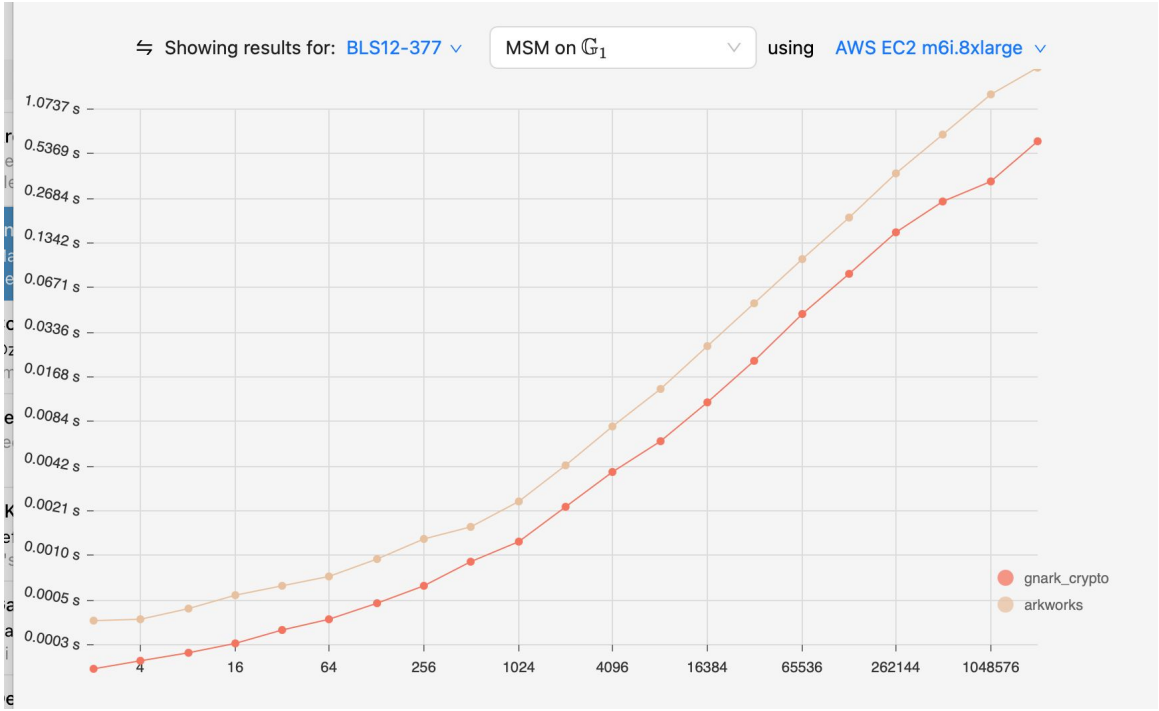
gnark is very fast (celer benchmark)

SHA2 preimage knowledge

https://github.com/celer-network/zk-benchmark

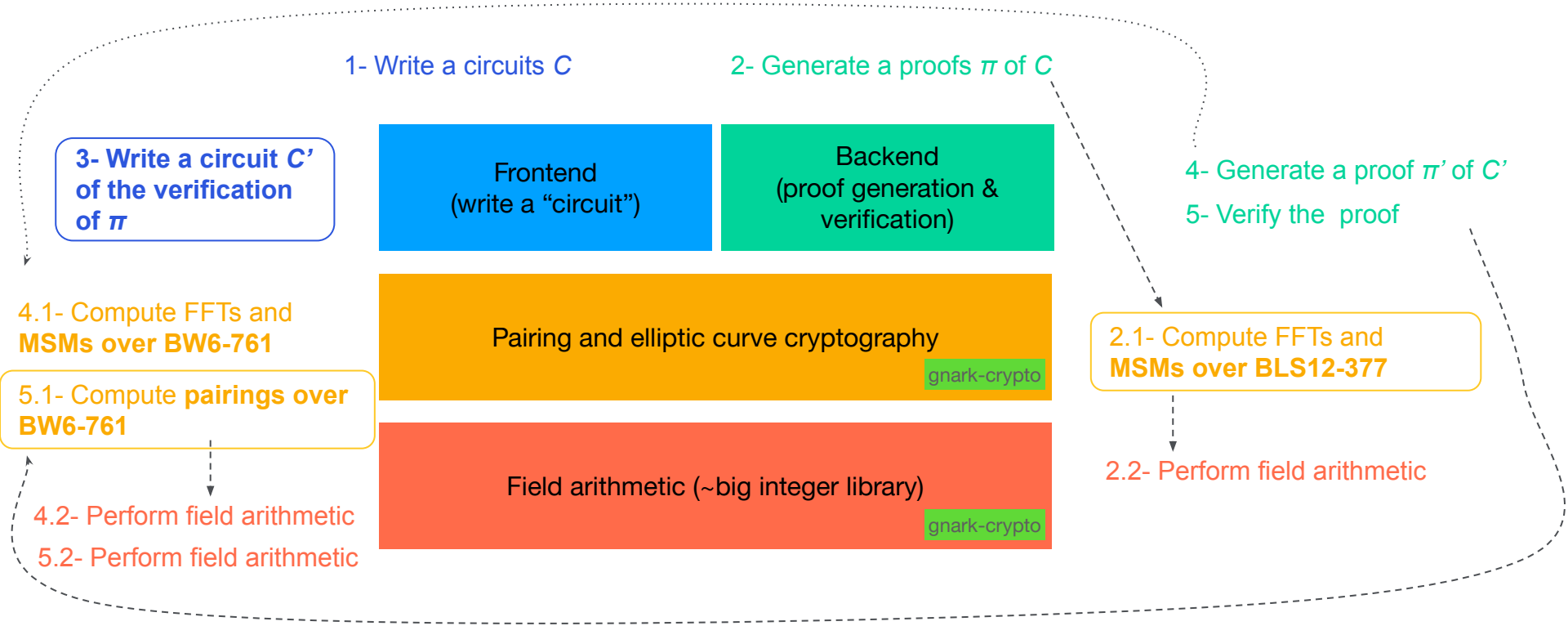


gnark is very fast (zka.lc benchmark)



Why is gnark that fast?

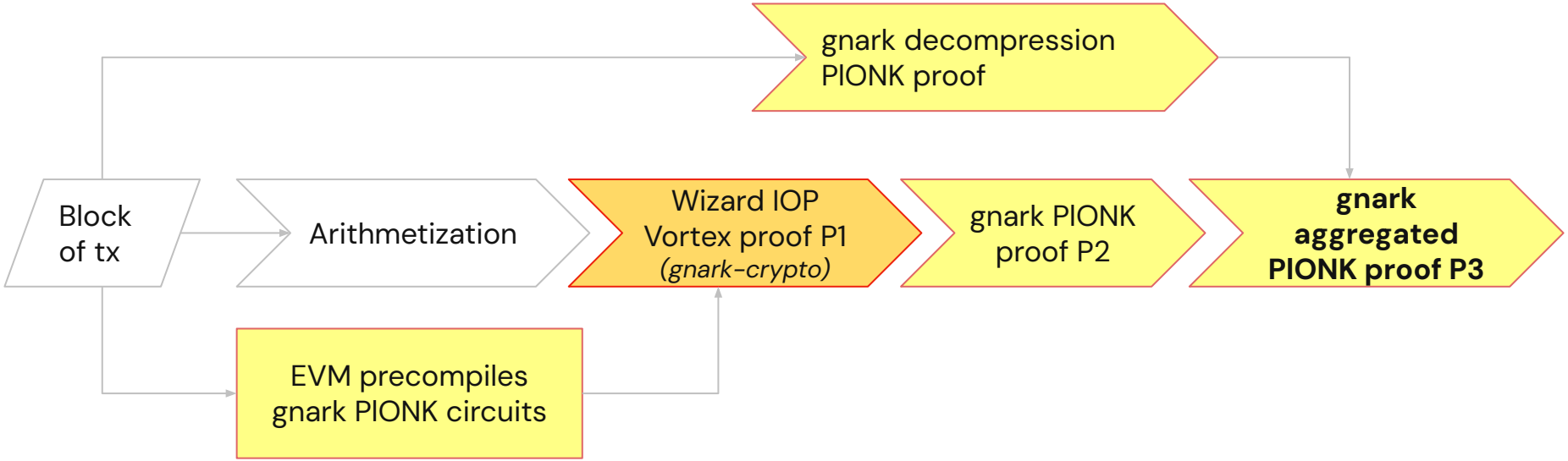
Example: 1-layer recursive PLONK proof



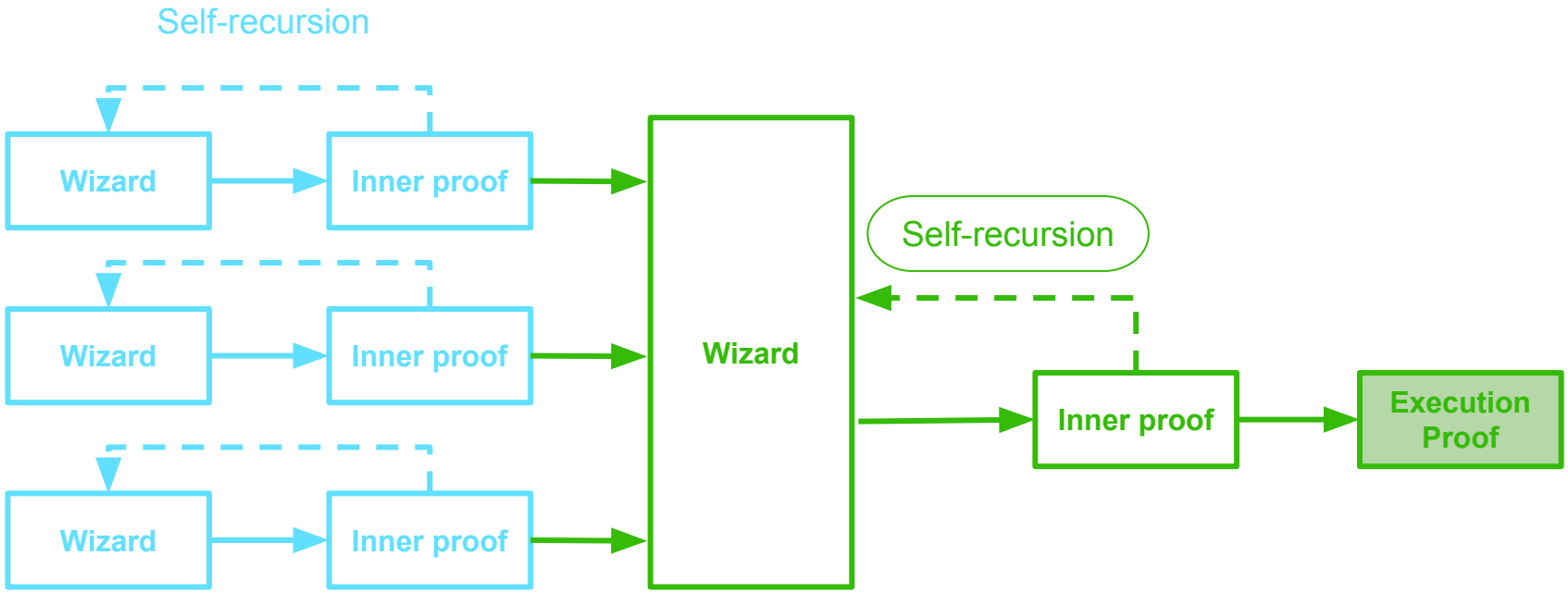
gnark applications

- zkEVMs (**Linea**)
- Rollups (zkBNB)
- Binance proof of solvency
- Worlcoin Groth16 prover
- Celer zkBridge and Brevis zkCoprocessor
- gnark-crypto: Algorand, EIP-4844 (go-kzg) and geth (EIP-2537)...
- Vocdoni - blockchain voting
- Noir or Sindri with a gnark backend
- Ingonyama (hardware accelerator): GPU support for Groth16 and PlonK.
- Some formal verification (Lean) on gnark circuits.
- Succinct labs (i.e. verify plonky 2 proofs in a gnark circuit)
- Base: keystores, FFLONK
- …

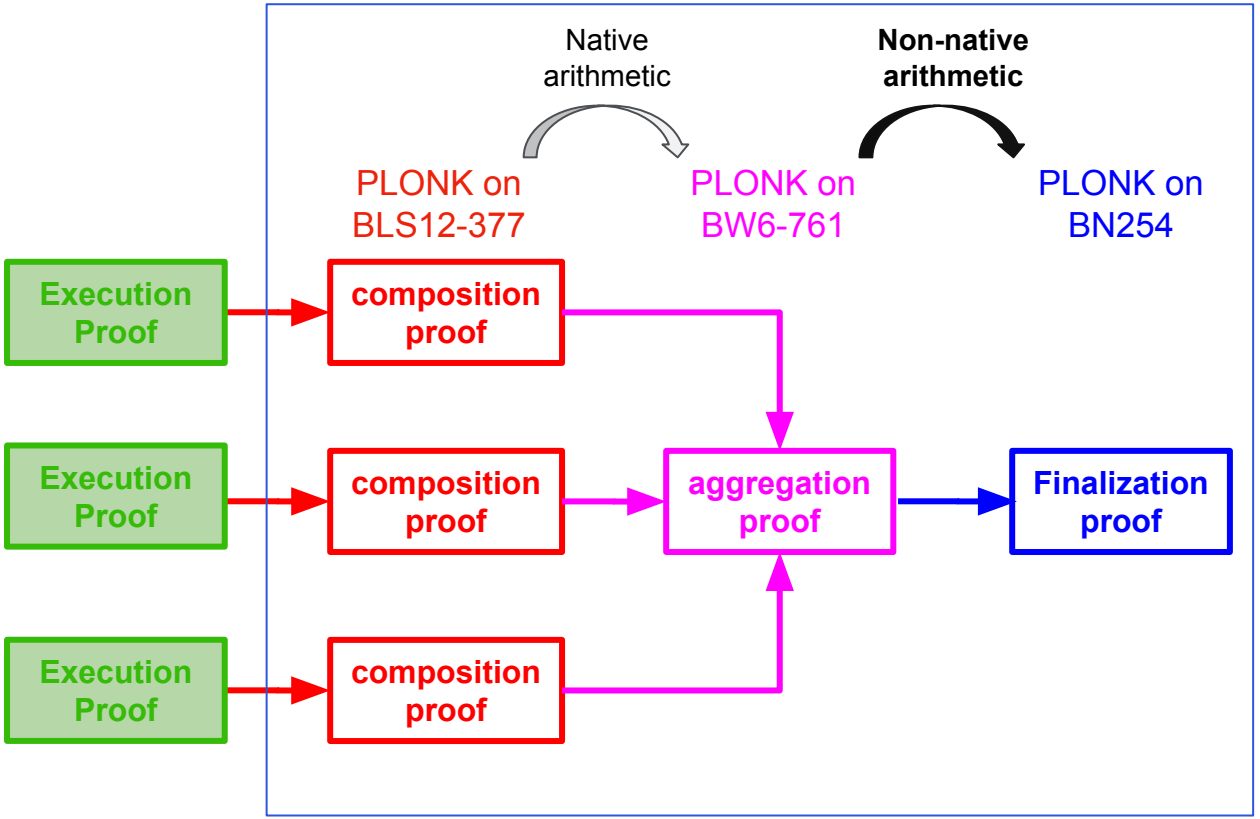
gnark in Linea



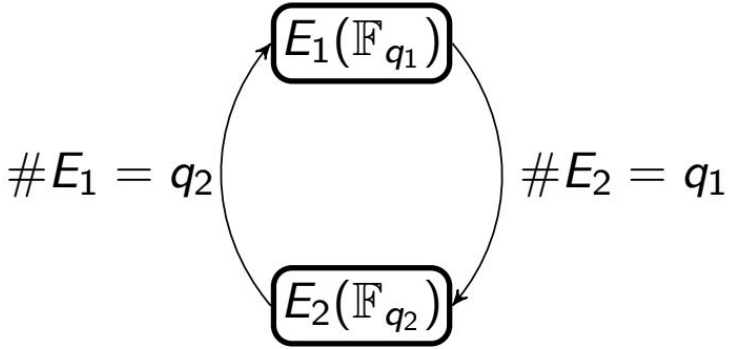
Aggregation for inner proofs



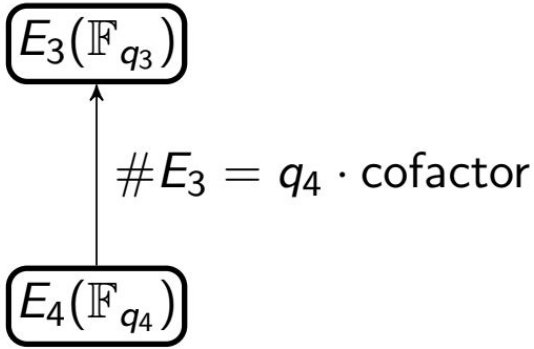
1-layer 2-chain aggregation



2-chain

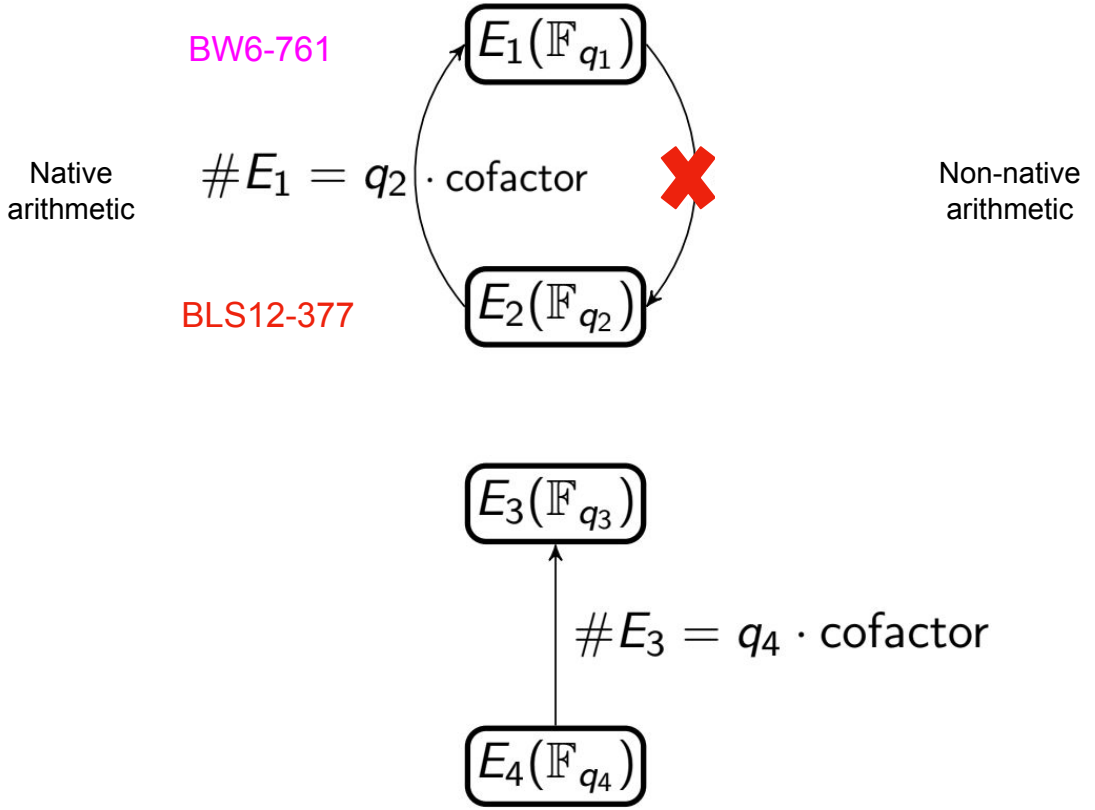


BW6-761

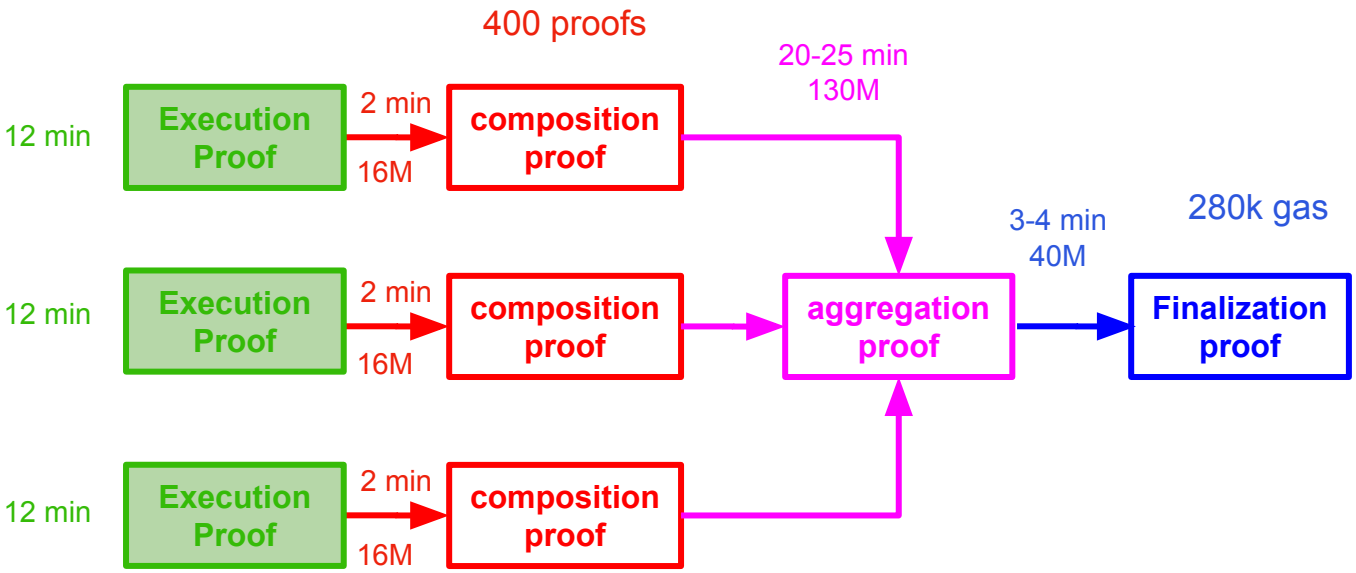


BLS12-377

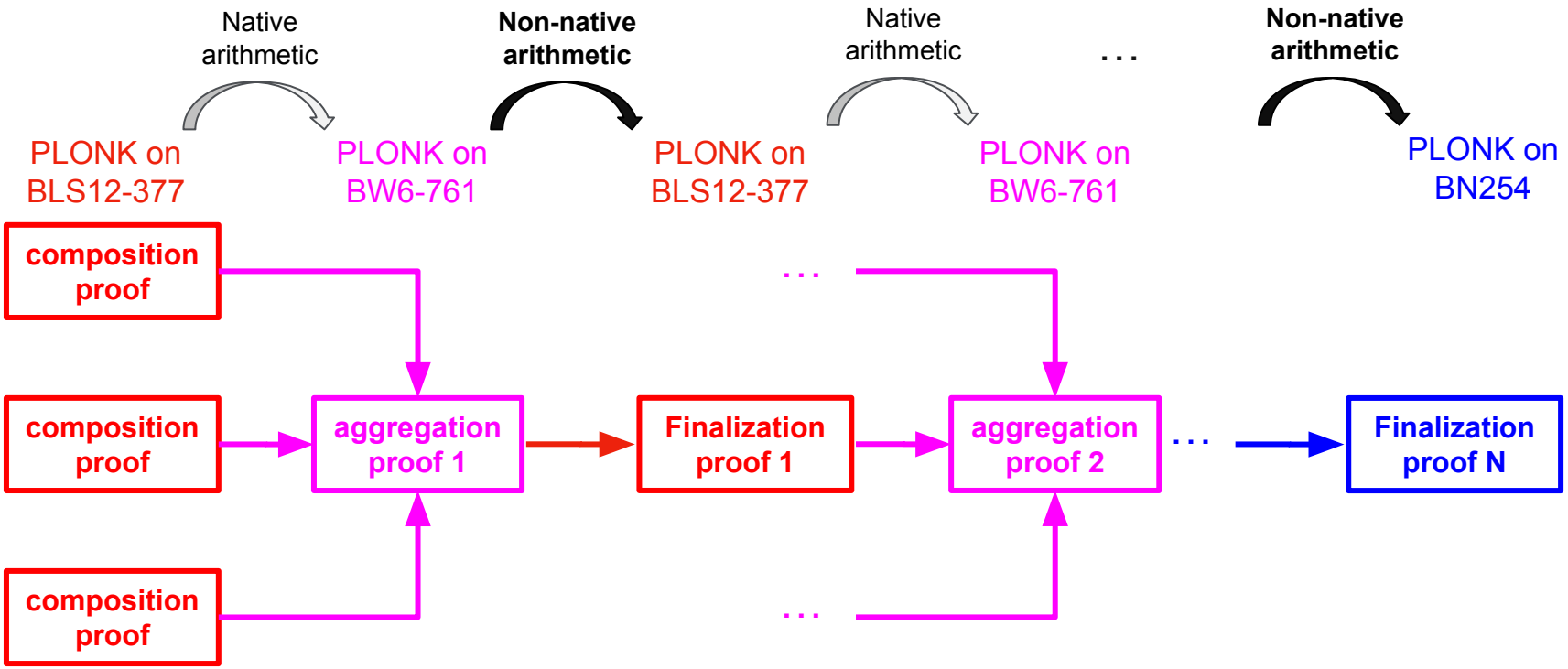
2-chain or *non-native* 2-cycle?



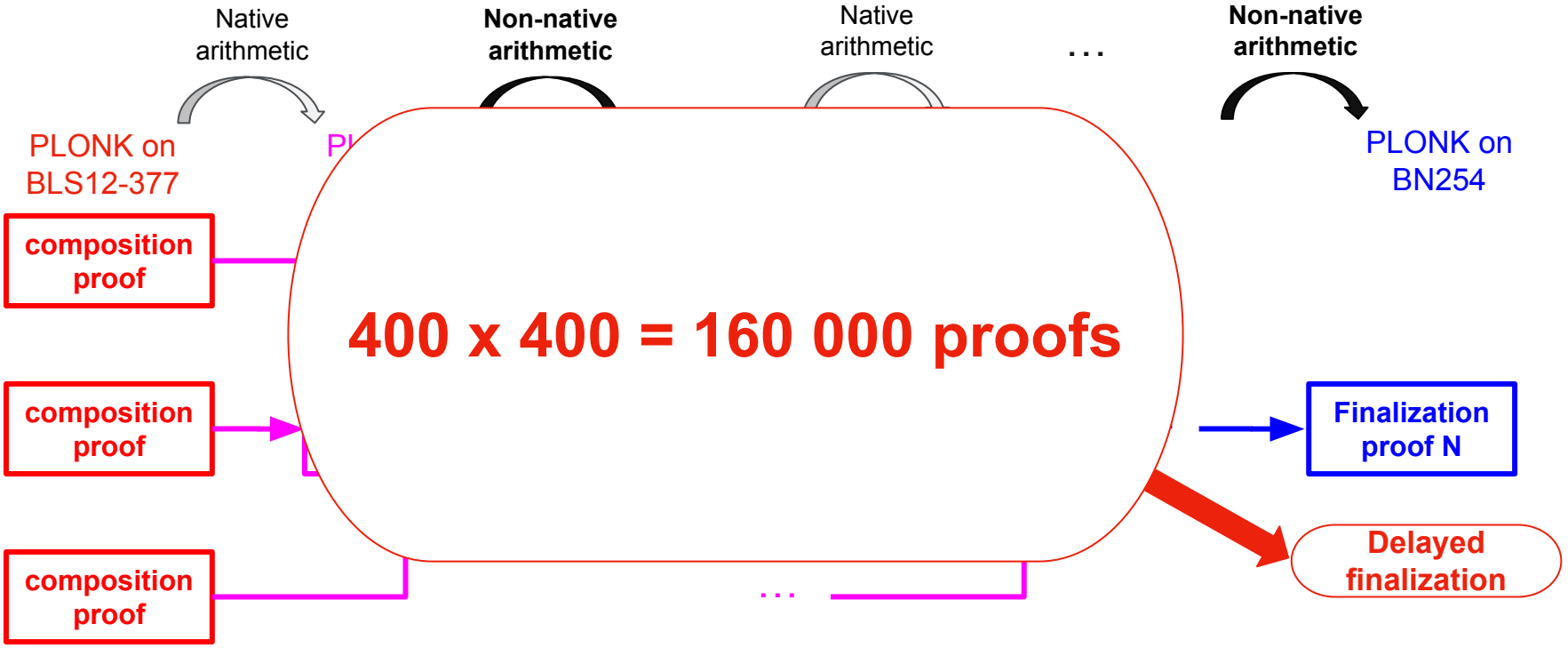
Benchmarks (hp6a)



Multi-layer 2-chain aggregation?



Multi-layer 2-chain aggregation?



Questions?

gnark@consensys.net
youssef.elhousni@consensys.net

X: @gnark_team, @YoussefElHousn3
GH: @yelhousni

linea.build
play.gnark.io
github.com/consensys/gnark
github.com/consensys/gnark-crypto