Zero Knowledge Proofs (ZKPs) as a Systems Challenge

Paarijaat Aditya

Workshop: Security, Privacy & Verifiable Computing for contemporary distributed systems

June 15th, 2023

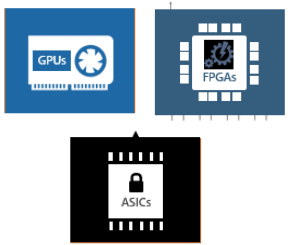
 © 2023 Nokia

NOKIA BELL LABS

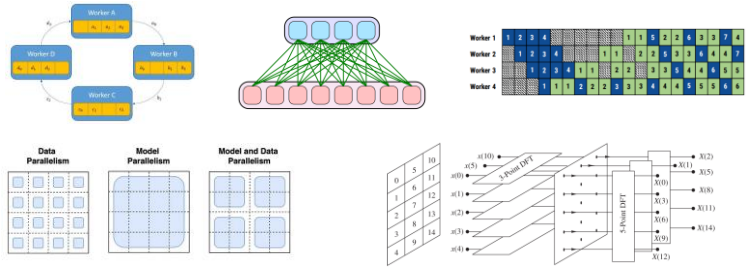
Systems Challenge?

- Creating new or making an existing end-to-end systems faster or better
- New insight + many existing components → makes the overall system better
  - Often the new insight comes when you are trying to build something end-to-end
- Innovative ways of using existing math/crypto + optimizing implementation

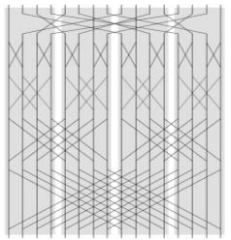
Like machine learning became a systems challenge → Zero Knowledge Proofs will also become a systems challenge



Hardware Acceleration



Distributing and Pipelining Computations



Optimizing data transfers

© 2023 Nokia

ZKPs as a Systems Challenge?

Program to be proven

Big Integer arithmetic:

- 100s of large vectors with 100s of millions values
- Large values (256-bit)
- Modulo large primes

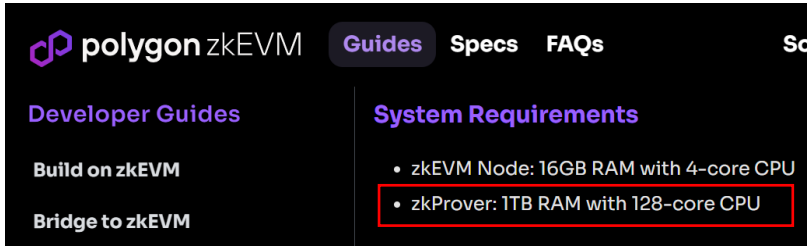
ZKP Machinery

→

Huge resource req.

- Compute
- Memory
- Network

→



r6a.32xlarge	$7.2/hour	128 vCPU	1024 GB RAM
6x c7g.8xlarge	6x$1.1562 = $6.9/hour	6x32 = 192 vCPU	6x64 GB = 384 GB RAM

- Efficient resource usage → less memory req. per node
- Support bigger applications → vectors with billions of values
- Faster proof generation → more compute cores

Need Distributed proof construction + Delegated proof construction
(besides hardware acceleration, better math)

Work towards challenging applications,
e.g., ZK-ML, Media Provenance

© 2023 Nokia

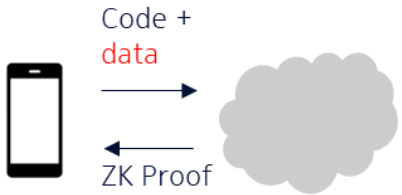
NOKIA BELL LABS

Agenda

Quick Introduction to ZKP
Proof Production Pipeline







Distributed Proof Construction

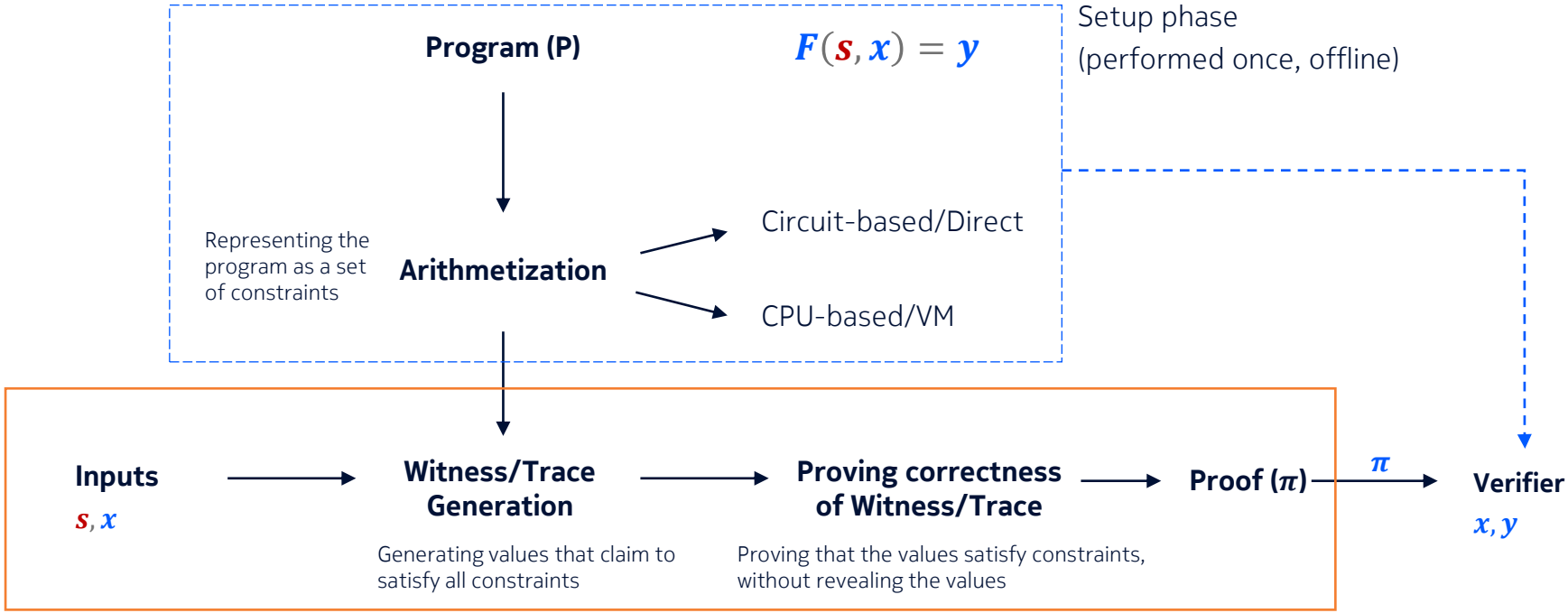
Distribute proof construction
across multiple machines

Delegated Proof Construction

Delegating proof construction
in a privacy-preserving manner

© 2023 Nokia

ZK Proofs Production Pipeline



© 2023 Nokia

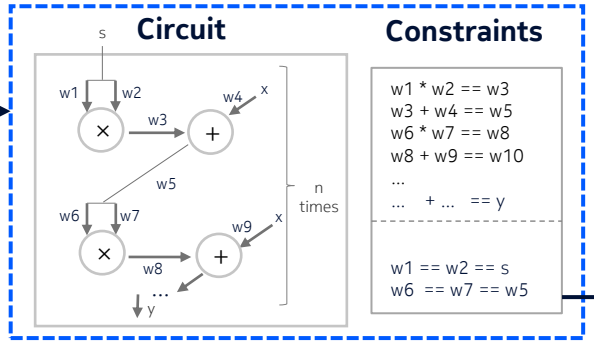
Program (P)

```
function foo(s,x) {
    let y = s;
    for i in 0..2 {
        y = y^2 + x;
    }
    return y;
}
```

Arithmetization

Offline preprocessing

Represent program as a circuit and a set of constraints



Number of wires/constraints = N

ECDSA sig. verif.	~ 90K
SHA256 (1KB)	~ 0.5M
Rollup 50 Txs	~ 34M
zkEVM	> 200M

Overhead of Proof generation:

$O(N \log^k(N))$ or $O(N)$

Big Integer Arithmetic:

- On large vectors (100s of million)
- With large values (256 bit)
- With modulo large primes

Inputs

s=1, x=10

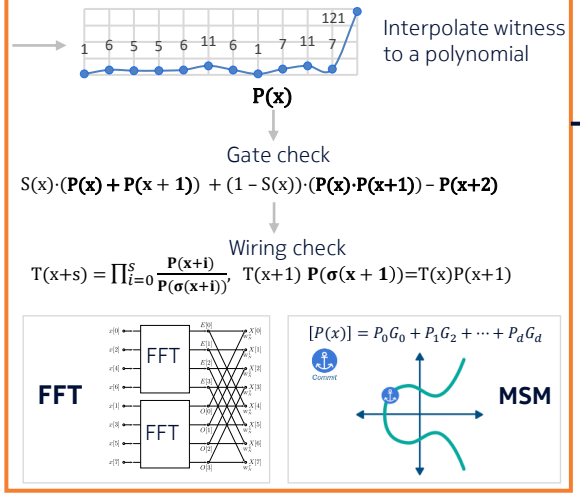
Witness generation

wire	value
w1, w2	1
w3	1
w4	10
w5, w6, w7	11
w8	121
...	...

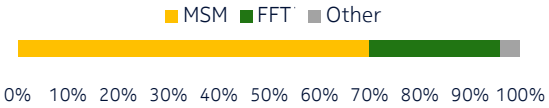
Witness/Trace

Generating values that claim to satisfy constraints

Proof generation



ZKP Machinery



Proof → **Verifier**

I know a value s, such that prog. (P) produces output y=131 for x=10

FFT – Fast Fourier Transform $O(N \log(N))$
(Actually NTT – Number Theoretic Transform)

MSM – Multi-Scaler Multiplications $O(N)$
(Elliptic curve operations)

Setup phase - performed once offline (known both to prover and verifier)

Proving phase - performed each time proof is constructed

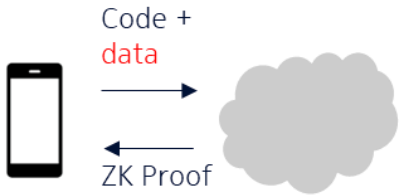
© 2023 Nokia

Agenda

Quick Introduction to ZKP
Proof Production Pipeline







Distributed Proof Construction

Distribute proof construction
across multiple machines

Delegated Proof Construction

Delegating proof construction
in a privacy-preserving manner

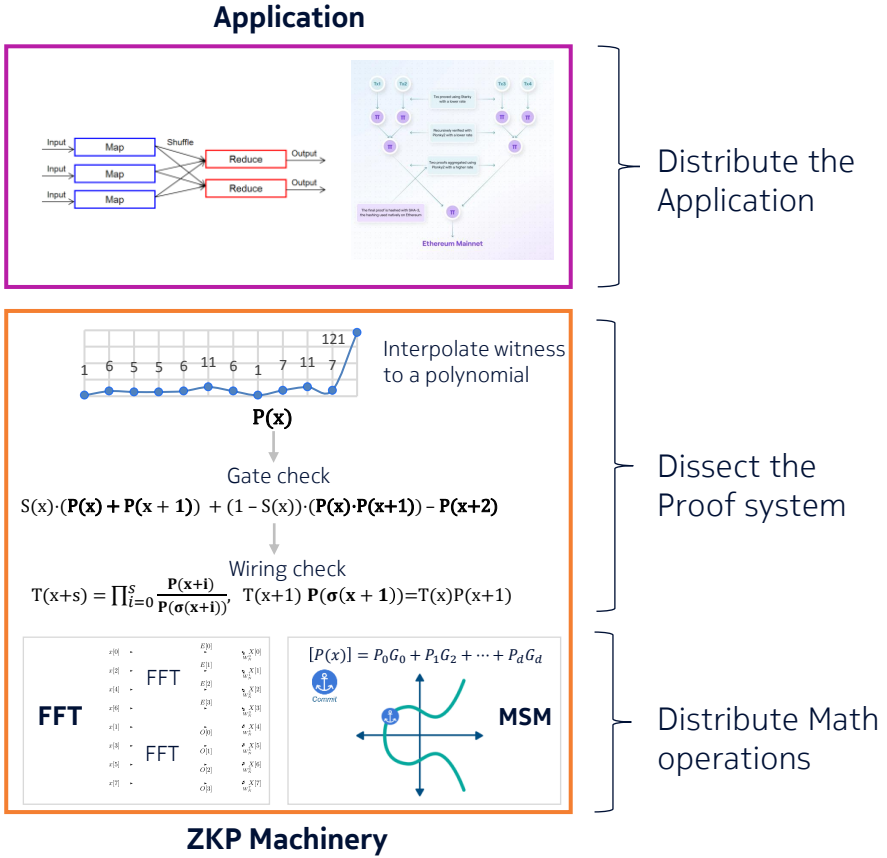
© 2023 Nokia

Distributing Zero Knowledge Proof Construction

Distribute across multiple nodes



- Support larger circuits (billions of constraints)
- Faster proof generation (more cores available)
- Efficient resource usage (less resources per node)



© 2023 Nokia

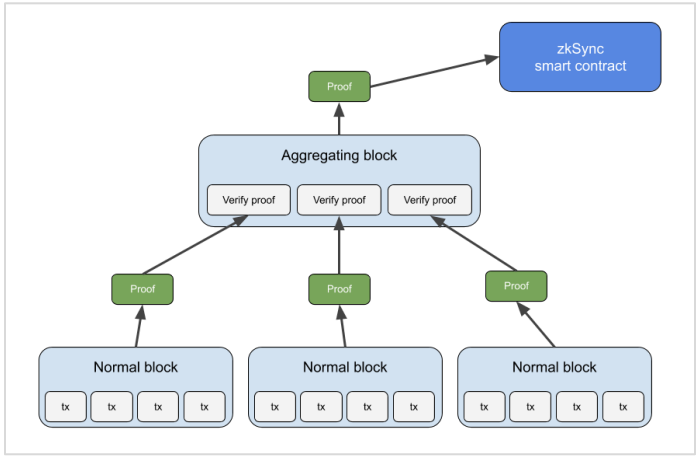
Distribute the Application

Prover overhead (compute and memory) ∝ Circuit/Trace size (N)

$$O\left(N\ log^{k}(N)\right) \quad \text{or} \quad O(N)$$

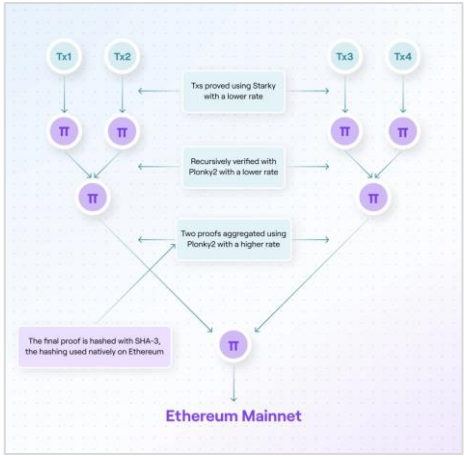
Rollup block with 50 Txs.
→ 34 Million constraints (R1CS)
→ 38 mins to prove (Ark-Spartan) [1] (38 seconds now)
(64 cores, 2.65GHz, 128 GB of RAM)

zkSync 1.1, 300 TPS → 3000 TPS



Source: (Matter Labs) - zkSync v1.1 "Reddit Edition": Recursion (up to 3,000 TPS!)

Inherent parallelism in the application
+
Proof Aggregation
=
Prover of a ZK-Rollup Server



Source: (Polygon Labs) - Plonky2: A Deep Dive

[1] Table 4 in HyperPlonk paper: https://eprint.iacr.org/2022/1355.pdf

© 2023 Nokia

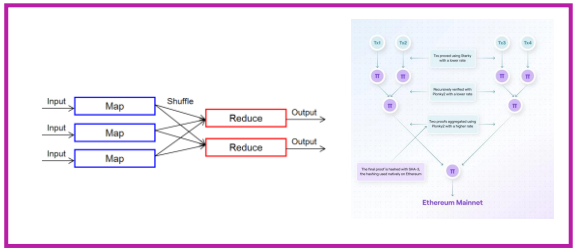
Distributing Zero Knowledge Proof Construction

Distribute across multiple nodes

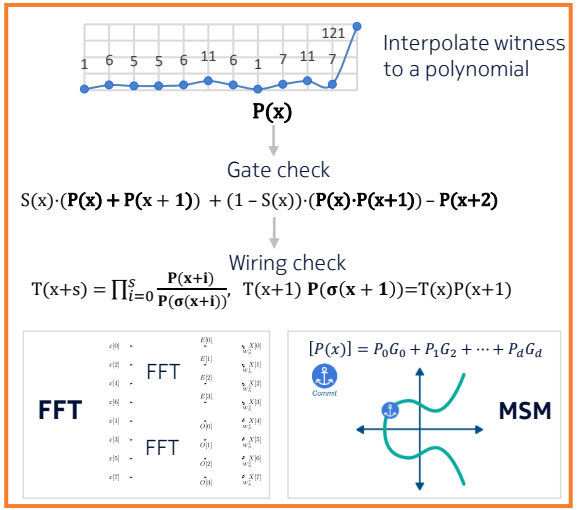


- Support larger circuits (billions of constraints)
- Efficient resource usage (less resources per node)
- Faster proof generation (more cores available)

Application



Distribute the Application



Dissect the Proof system

Distribute Math operations

ZKP Machinery

© 2023 Nokia

Distribute the Math Operations

Wire values

Constraints

ZKP Machinery

Polynomial Computations
(Multiplications, Interpolation, Evaluations)

FFT*

Polynomial Commitments

Multi-scaler Multiplications (MSM)

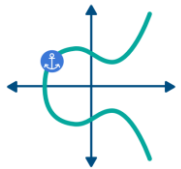
$$[a(x)] = a_0 G_0 + a_1 G_1 + \ldots + a_d G_d$$

Elliptic curves points

Polynomial coefficients

Commit

MSM is used to produce a single elliptic curve point that serves as the polynomial commitment



FFT is used for fast Polynomial Multiplications, evaluations, and interpolations

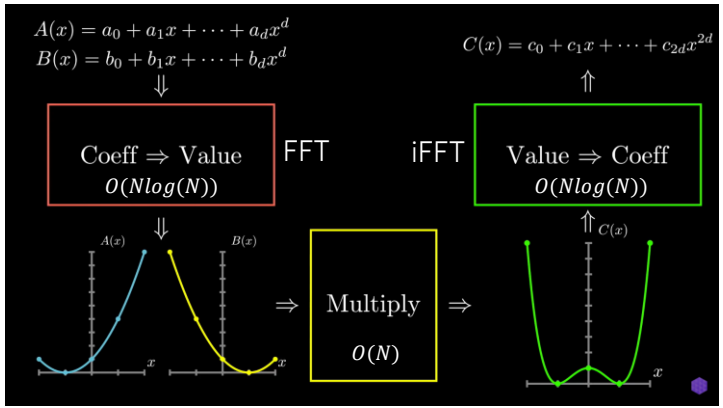


Image source: Reducible (YouTube)

Element-wise product

$$A(x)\,B(x) = iFFT\left(FFT(A(x)) \circ FFT(B(x))\right)$$

Groth16 quotient polynomial

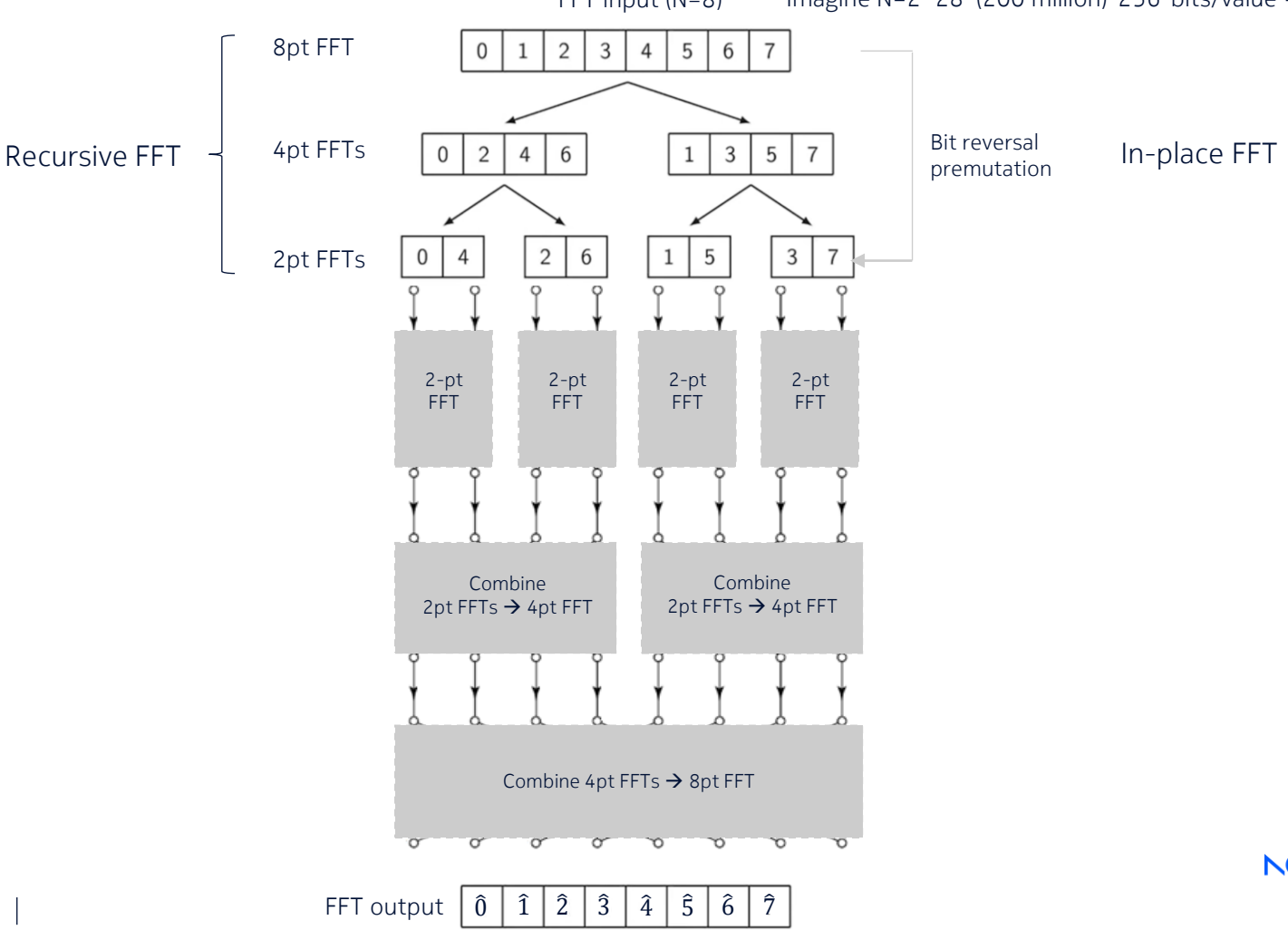
$$\frac{A(x)\,B(x) - C(x)}{x^N - 1}$$

PlonK quotient polynomial

$$\begin{aligned} \mathsf{t}(X) = {} & (\mathsf{a}(X)\mathsf{b}(X)\mathsf{q}_\mathsf{M}(X) + \mathsf{a}(X)\mathsf{q}_\mathsf{L}(X) + \mathsf{b}(X)\mathsf{q}_\mathsf{R}(X) + \mathsf{c}(X)\mathsf{q}_\mathsf{O}(X) + \mathsf{PI}(X) + \mathsf{q}_\mathsf{C}(X))\tfrac{1}{\mathsf{Z}_\mathsf{H}(X)} \\ & + ((\mathsf{a}(X) + \beta X + \gamma)(\mathsf{b}(X) + \beta k_1 X + \gamma)(\mathsf{c}(X) + \beta k_2 X + \gamma)\mathsf{z}(X))\tfrac{\alpha}{\mathsf{Z}_\mathsf{H}(X)} \\ & - ((\mathsf{a}(X) + \beta \mathsf{S}_{\sigma 1}(X) + \gamma)(\mathsf{b}(X) + \beta \mathsf{S}_{\sigma 2}(X) + \gamma)(\mathsf{c}(X) + \beta \mathsf{S}_{\sigma 3}(X) + \gamma)\mathsf{z}(X\omega))\tfrac{\alpha}{\mathsf{Z}_\mathsf{H}(X)} \\ & + (\mathsf{z}(X) - 1)\,\mathsf{L}_1(X)\tfrac{\alpha^2}{\mathsf{Z}_\mathsf{H}(X)} \end{aligned}$$

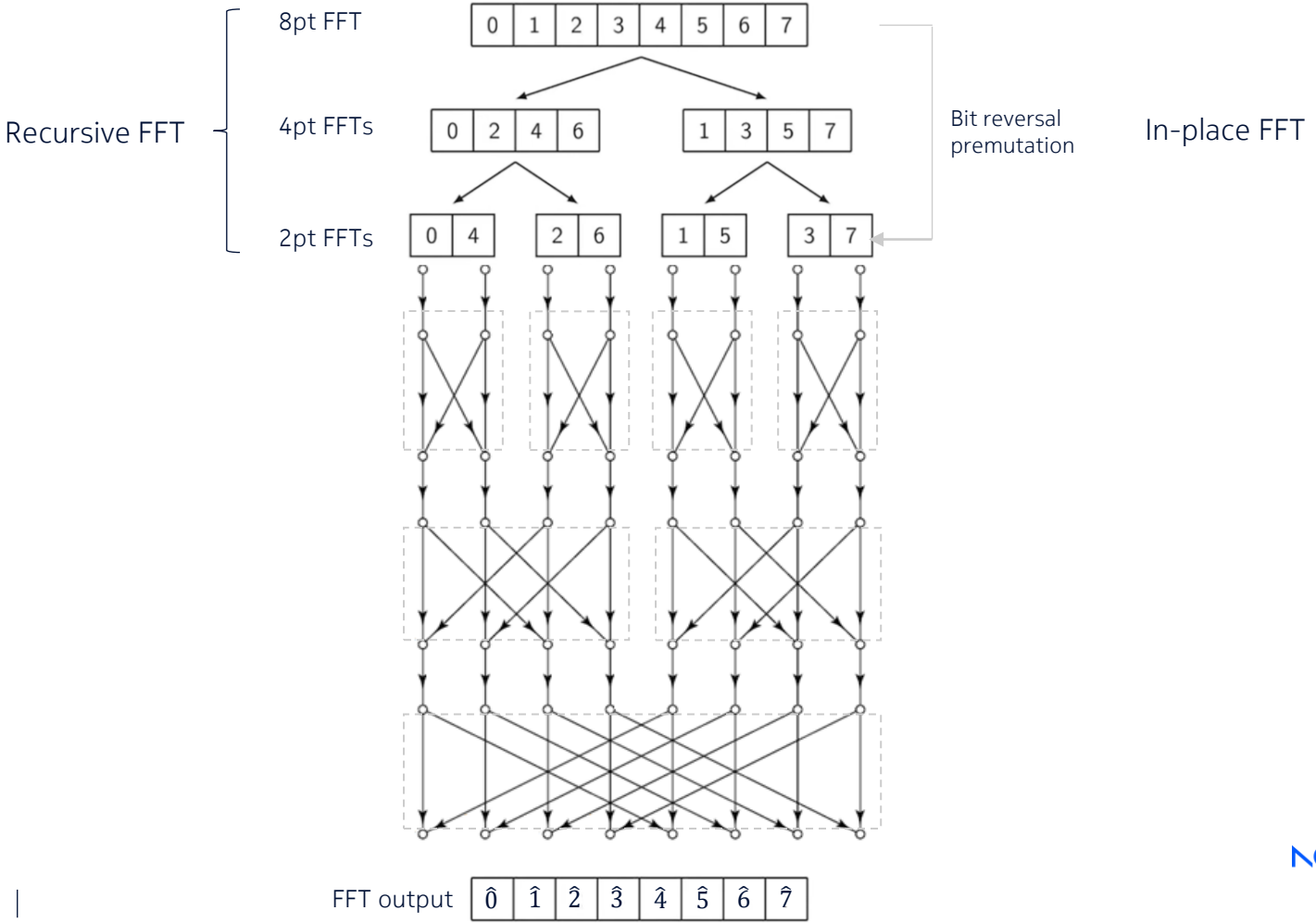
* FFT refers to Number Theoretic Transform (NTT) in this ppt

FFT



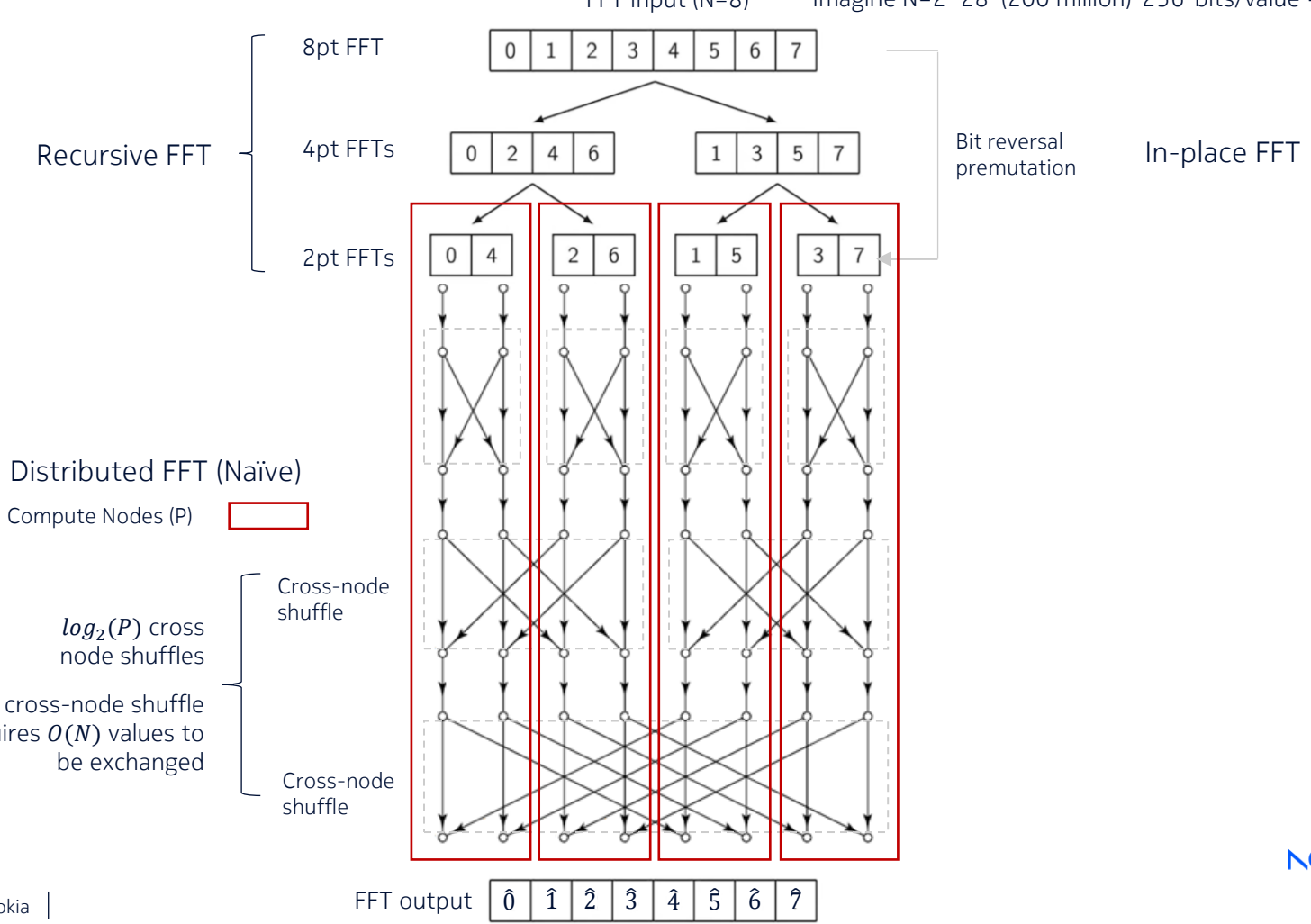
© 2023 Nokia

FFT



© 2023 Nokia

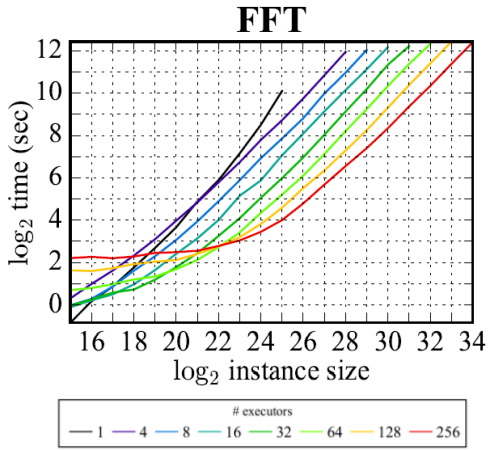
FFT



© 2023 Nokia

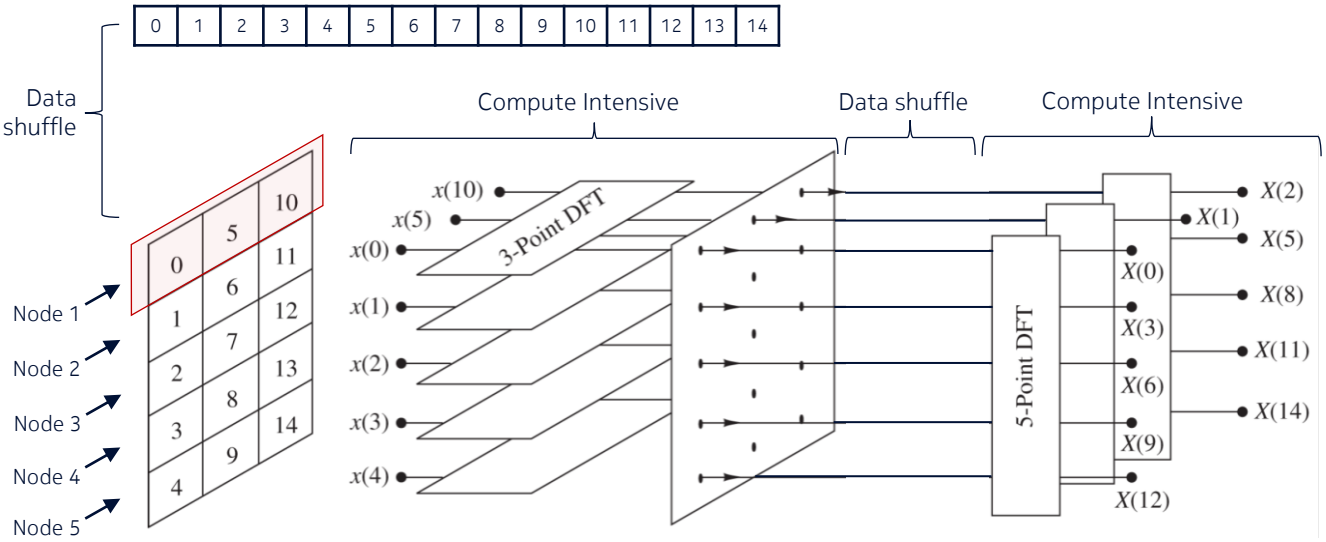
Map-Reduce FFT

- Generalized Cooley–Tukey algorithm implemented as Map-Reduce
  - Two data shuffles (regardless of the number of nodes)
- Used by DIZK: A Distributed Zero Knowledge Proof System [2018]



20 node EC2 cluster
(each with 32 vCPUs, 244 GB)
used for 256 Spark executors.

4 billion sized FFT (256 bit)
in ~1000 sec



Source: Digital Signal Processing: Principles, Algorithms, and Applications (J. Proakis. and D. Manolakis)

© 2023 Nokia

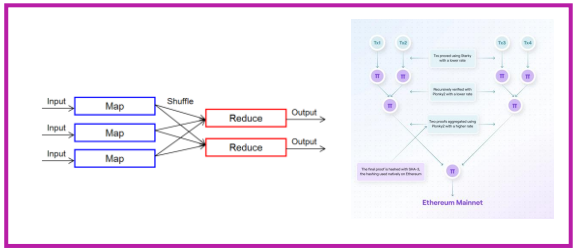
Distributing Zero Knowledge Proof Construction

Distribute across multiple nodes

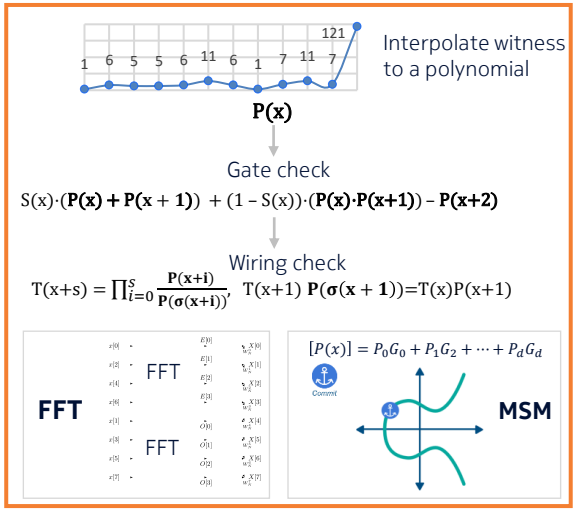


- Support larger circuits (billions of constraints)
- Efficient resource usage (less resources per node)
- Faster proof generation (more cores available)

Application



Distribute the Application



Dissect the Proof system

Distribute Math operations

ZKP Machinery

© 2023 Nokia

Dissect the Proving System

Even 2 shuffles per FFT might be painful depending on the proving system

- DIZK implemented Groth16

$$\frac{A(x)\,B(x) - C(x)}{x^N - 1}$$

3 wire polynomials → quotient polynomial involves computing the product of 2 large degree (N) polynomials

- ZPRIZE TurboPlonk+DIZK challenge

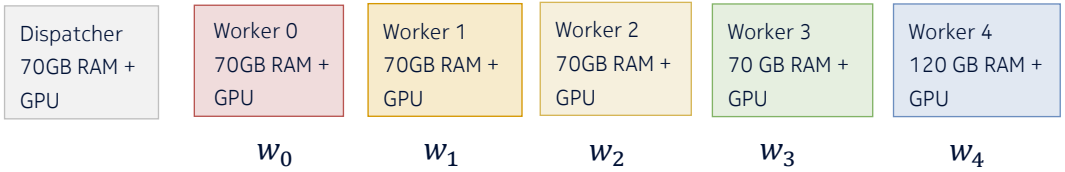
$$q_c + \mathrm{p} + q_{m_0} w_0 w_1 + q_{m_1} w_2 w_3 + q_{a_0} w_0 + q_{a_1} w_1 + q_{a_2} w_2 + q_{a_3} w_3 + q_{h_0} w_0^5 + q_{h_1} w_1^5 + q_{h_2} w_2^5 + q_{h_3} w_3^5 + q_e w_0 w_1 w_2 w_3 w_4 - q_o w_4 + (z-1)L_1 \;+\; \frac{(w_0 + \beta x + \gamma)(w_1 + \beta\kappa_1 x + \gamma)\,(w_2 + \beta\kappa_2 x + \gamma)(w_3 + \beta\kappa_3 x + \gamma)(w_4 + \beta\kappa_4 x + \gamma)\,z -}{(w_0 + \beta\sigma_0 + \gamma)(w_1 + \beta\sigma_1 + \gamma)\,(w_2 + \beta\sigma_2 + \gamma)(w_3 + \beta\sigma_3 + \gamma)(w_4 + \beta\sigma_4 + \gamma)\,z(\omega x)}$$

18 wire polynomials (5 witness + 13 selector) → quotient polynomial involves computing product of 6 N-degree polynomials (many times)
(For FFTs, the nearest power of 2 will be **8N** degree polynomials)

- For **N=2^{28} (200 million)**, a N-degree polynomial occupies 8GB RAM → **64GB RAM** (8N degree polynomial for FFTs)
- Just storing 18 polynomials of 64GB each = **1.1TB RAM**
- With 2 shuffles per FFT (DIZK), Running FFTs on 18 polynomials of 64GB each will incur **2.2TB network transfers**
  - 2.2TB network transfers @ 1 Gbytes/sec → **40mins**

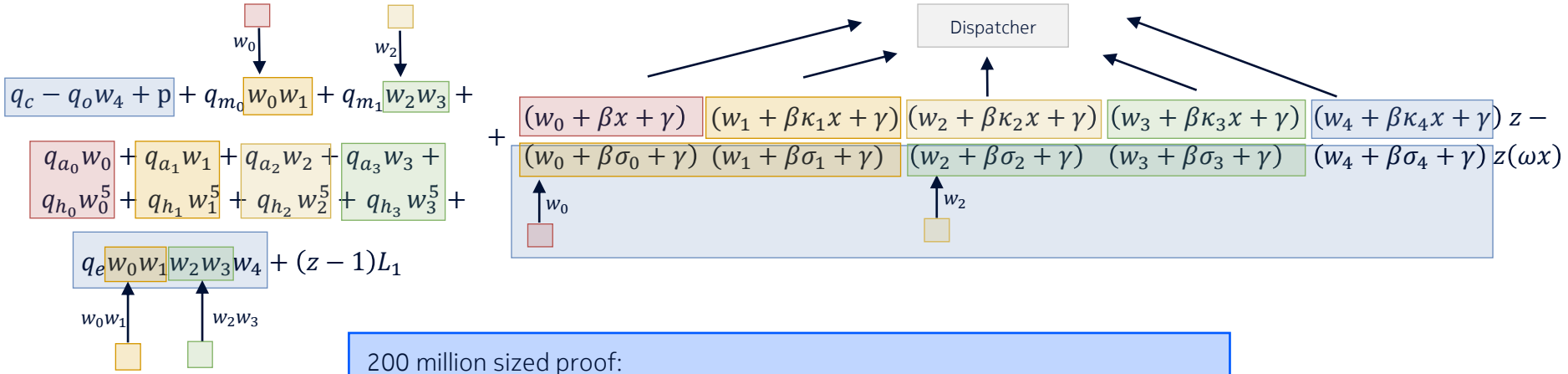
© 2023 Nokia
NOKIA BELL LABS

Zprize Plonk+DIZK challenge winner



Round 1: Each worker interpolates (FFT) 1 witness vector to 1 wire polynomial, and computes its commitment (MSM)
FFT and MSM are not distributed, but run completely on each worker

Round 3: Each worker works independently on part of quotient polynomial
Workers transmit their computed polynomials (or chunks of it) to others according to a plan



200 million sized proof:

single node	~ 1TB RAM + Proof time: 80 mins
cluster of 6 workers	~ 70 GB RAM on 4 workers, 120 GB RAM on 1 worker + Proof time: 48 mins

Source: https://github.com/z-prize/2022-entries/tree/main/open-division/prize3-plonk-dizk/winderica

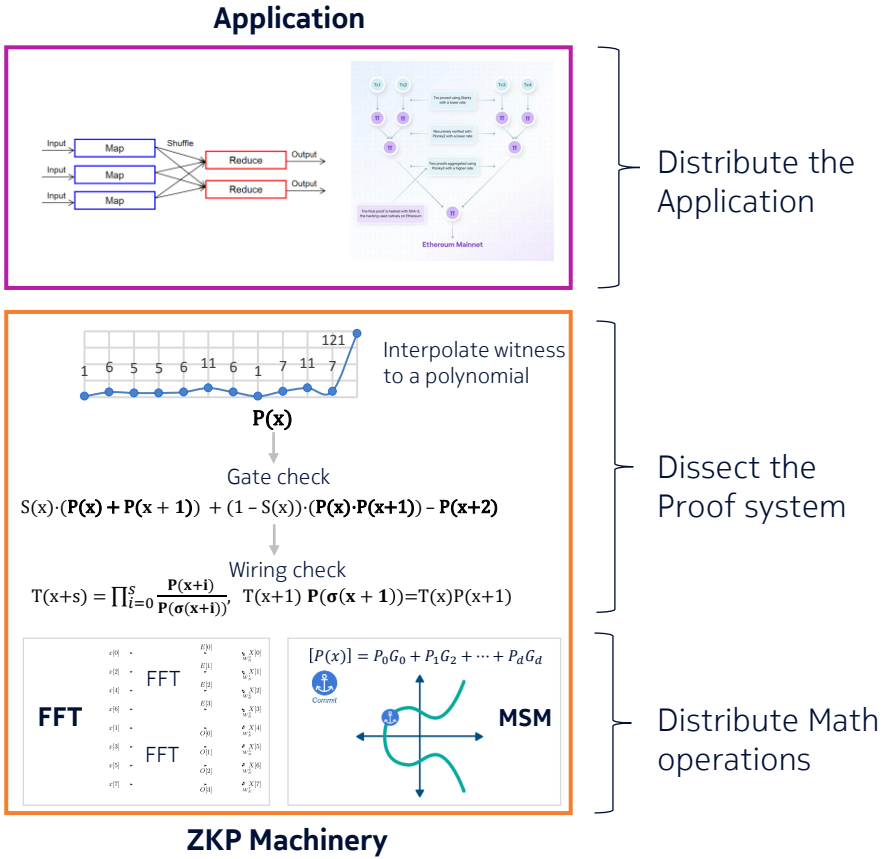
© 2023 Nokia

Distributing Zero Knowledge Proof Construction

Distribute across multiple nodes



- Support larger circuits (billions of constraints)
- Efficient resource usage (less resources per node)
- Faster proof generation (more cores available)



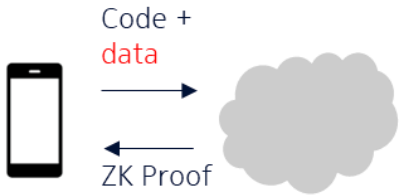
© 2023 Nokia

Agenda

Quick Introduction to ZKP
Proof Production Pipeline







Distributed Proof Construction

Distribute proof construction
across multiple machines

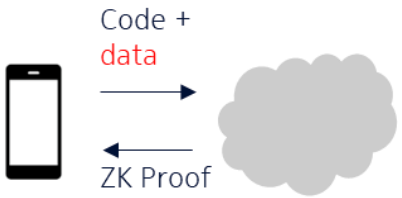
Delegated Proof Construction

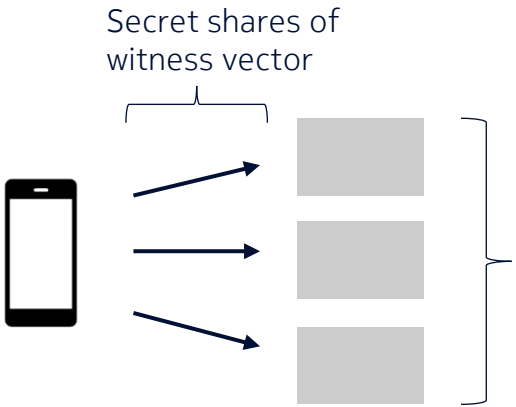
Delegating proof construction
in a privacy-preserving manner

© 2023 Nokia

Privacy-Preserving Delegated Proof Construction

- Edge to Cloud outsourcing of proof construction
- Leaks sensitive private information





Being explored by the community

MPC friendly ZK operations:

- FFTs are linear operations
- MSMs (with fixed EC points) are linear operations

Non-linear ops:

- SHA256 hashes used in ZK-STARKs (Merkle Trees)

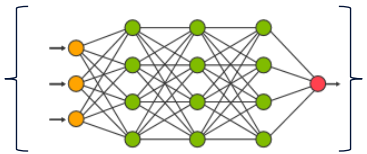
© 2023 Nokia

Conclusion

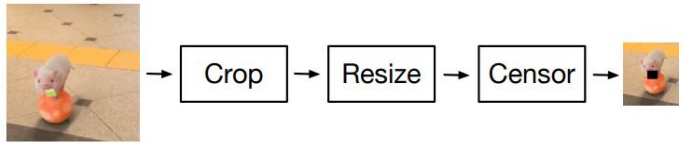
Like machine learning became a systems challenge → Zero Knowledge Proofs will also become a systems challenge

Need Distributed proof construction + Delegated proof construction
(besides hardware acceleration, better math)

- Support bigger applications → vectors with billions of values
- Faster proof generation → more compute cores
- Efficient resource usage → less memory req. per node



ZK-ML
ZK proofs ML inference



Media Provenance Proofs

© 2023 Nokia

paarijaat.aditya@nokia-bell-labs.com

NOKIA
BELL
LABS