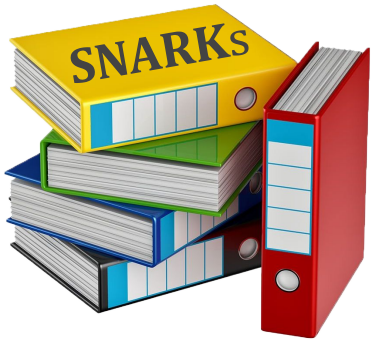
SNARKPack

Practical Groth16 Aggregation

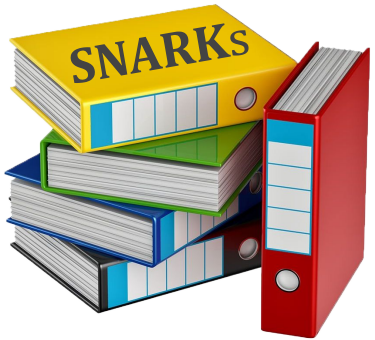
Joint work with
Mary Maller

Anca Nitulescu
Nicolas Gailly
Protocol Labs

In Brief

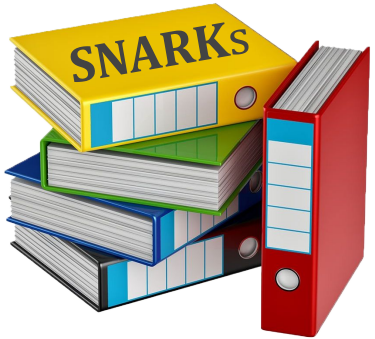


In Brief



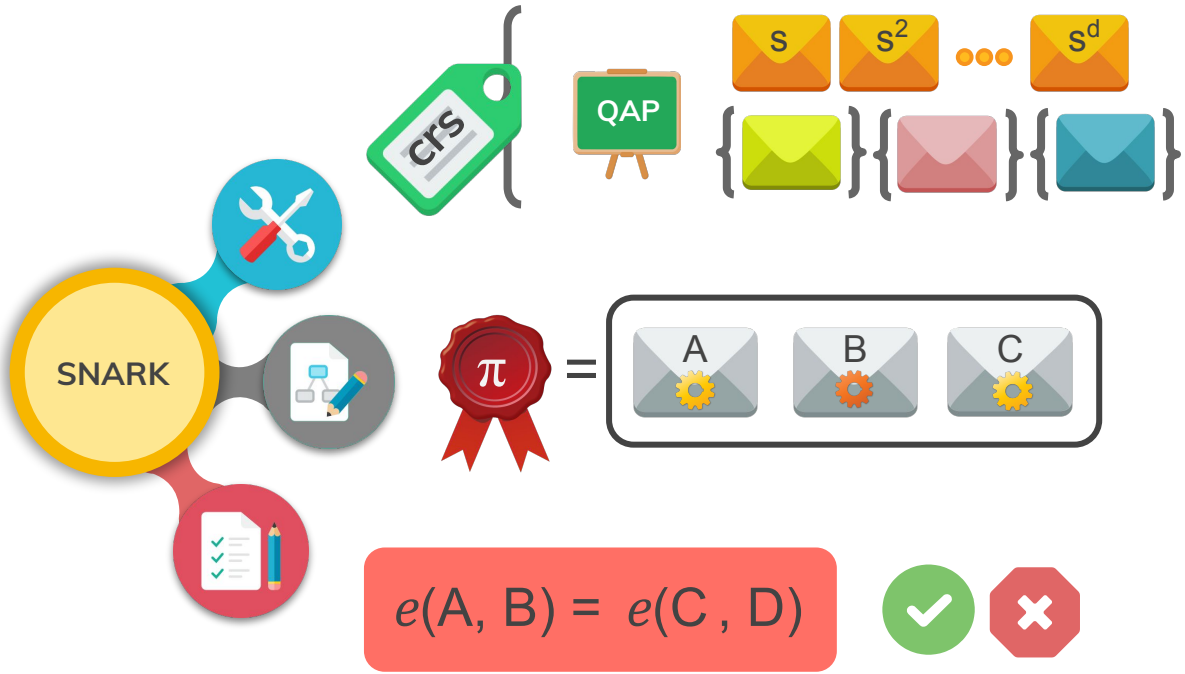
SNARKPack

In Brief



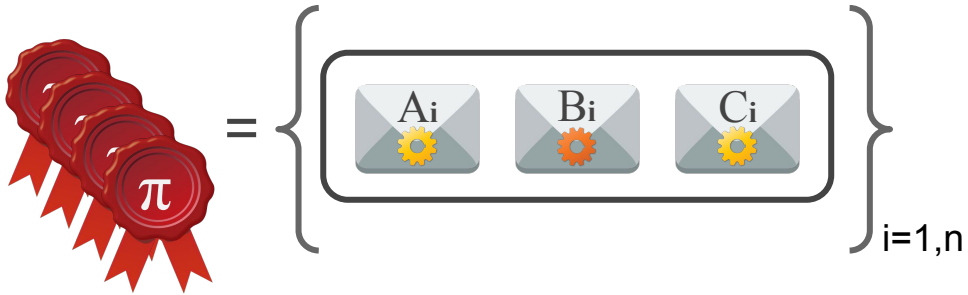
SNARKPack

Groth16



Many SNARKs

Proofs

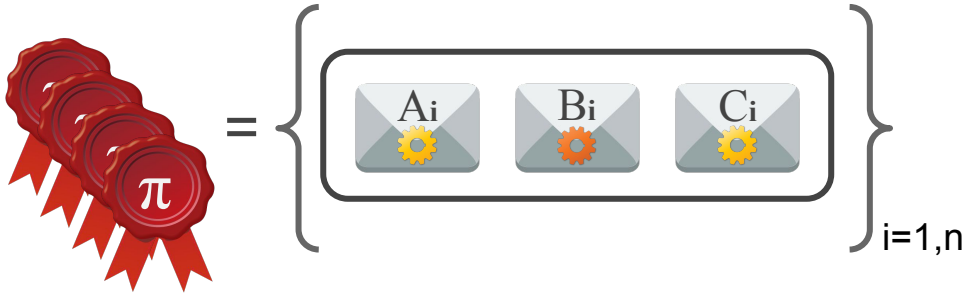


Verification ($D = g^d$)

$e(A_i, B_i) = e(C_i, D)$

Many SNARKs

Proofs



Verification ($D = g^d$)

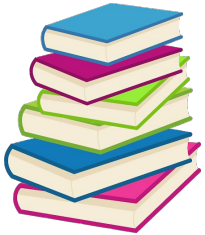
$e(A_1, B_1) = e(C_1, D)$

$e(A_2, B_2) = e(C_2, D)$

...

$e(A_n, B_n) = e(C_n, D)$

SNARK Batching



Verification

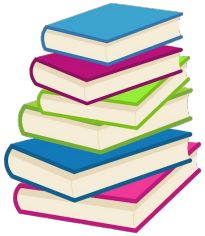
$e(A_1, B_1) = e(C_1, D)$

$e(A_2, B_2) = e(C_2, D)$

...

$e(A_n, B_n) = e(C_n, D)$

SNARK Batching



Verification

$e(A_1, B_1) = e(C_1, D)$

$e(A_2, B_2) = e(C_2, D)$

...

$e(A_n, B_n) = e(C_n, D)$

$r \leftarrow \$$

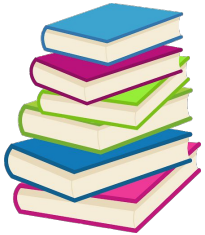
$r \square e(A_1, B_1) = e(C_1, D)$

$r^2 \square e(A_2, B_2) = e(C_2, D)$

...

$r^n \square e(A_n, B_n) = e(C_n, D)$

SNARK Batching



Verification

$e(A_1, B_1) = e(C_1, D)$

$e(A_2, B_2) = e(C_2, D)$

...

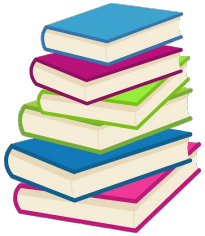
$e(A_n, B_n) = e(C_n, D)$

$r \leftarrow \$$

Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

SNARK Batching



Verification

$e(A_1, B_1) = e(C_1, D)$

$e(A_2, B_2) = e(C_2, D)$

...

$e(A_n, B_n) = e(C_n, D)$

$r \leftarrow \$$

Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

SNARK Aggregation

Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

$$\Leftrightarrow$$

$$\prod e(A_i, B_i^{r^i}) = e(\prod C_i^{r^i}, D)$$

$$\langle g \rangle = \mathbb{G}, \langle \tilde{g} \rangle = \tilde{\mathbb{G}}$$
$$e : \mathbb{G} \times \mathbb{G} \to \tilde{\mathbb{G}}$$
$$e(g^a, g^b) = \tilde{g}^{ab}$$

SNARK Aggregation

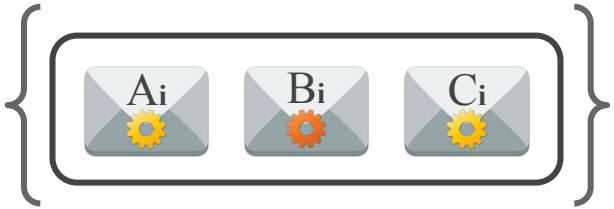
Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

$$\Leftrightarrow$$

$$\prod e(A_i, B_i^{r^i}) = e(\prod C_i^{r^i}, D)$$

Aggregation



SNARK Aggregation

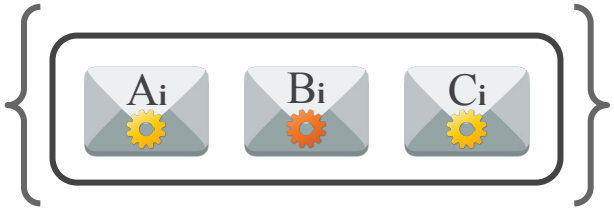
Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

$$\Leftrightarrow$$

$$\prod e(A_i, B_i^{r^i}) = e(\prod C_i^{r^i}, D)$$

Aggregation



$$Z_{AB} = \prod e(A_i, B_i^{r^i})$$

$$Z_C = \prod C_i^{r^i}$$

SNARK Aggregation

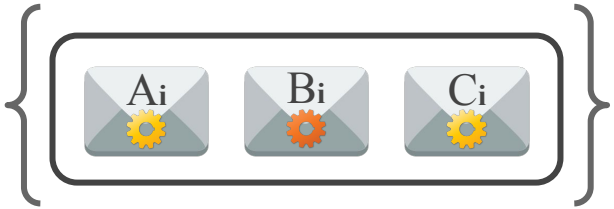
Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

$$\Leftrightarrow$$

$$Z_{AB} = e(Z_C, D)$$

Aggregation



$$Z_{AB} = \prod e(A_i, B_i^{r^i})$$

$$Z_C = \prod C_i^{r^i}$$

SNARK Aggregation

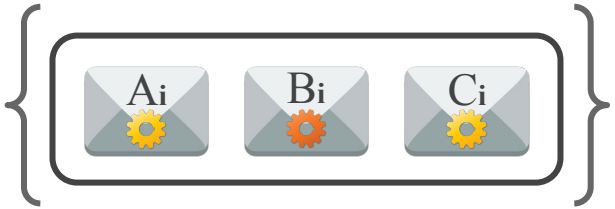
Batch Verification

$$\prod e(A_i, B_i)^{r^i} = \prod e(C_i, D)^{r^i}$$

$$\Leftrightarrow$$

$$Z_{AB} = e(Z_C, D)$$

Aggregation



$$Z_{AB} = \prod e(A_i, B_i^{r^i})$$

$$Z_C = \prod C_i^{r^i}$$

Construction

Tools: MIPP & TIPP

Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

$$\langle \mathbf{A}, \mathbf{b} \rangle = \prod A_i^{b_i}$$

$$\langle \mathbf{A}, \mathbf{B} \rangle = \prod e(A_i, B_i)$$

$$A_i, B_i \in \mathbb{G}, \quad b_i \in \mathbb{Z}_q$$

Tools: MIPP & TIPP

Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

$$\langle \mathbf{A}, \mathbf{b} \rangle = \prod A_i^{b_i}$$

$$\langle \mathbf{A}, \mathbf{B} \rangle = \prod e(A_i, B_i)$$

$$Z_c = \prod C_i^{r^i}$$

$$Z_{AB} = \prod e(A_i, B_i^{r^i})$$

Aggregation

Tools: MIPP & TIPP

Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

$$\langle \mathbf{A}, \mathbf{b} \rangle = \prod A_i^{b_i}$$

$$\langle \mathbf{A}, \mathbf{B} \rangle = \prod e(A_i, B_i)$$

$$Z_C = \langle \mathbf{C}, r \rangle$$

$$Z_{AB} = \langle \mathbf{A}, \mathbf{B}^r \rangle$$

Aggregation

Tools: MIPP & TIPP

Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

$$\langle \mathbf{C}, \mathbf{r} \rangle = \prod C_i^{r_i}$$

$$\langle \mathbf{A}, \mathbf{B}^{\mathbf{r}} \rangle = \prod e(A_i, B_i^{r_i})$$

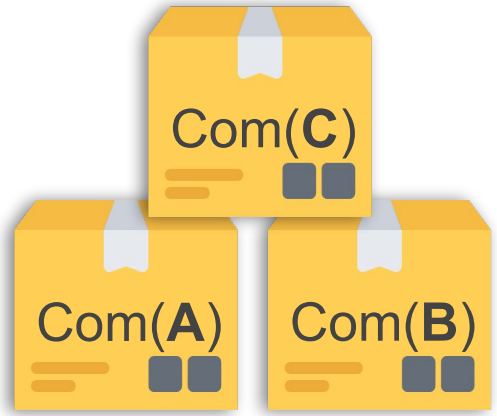
$$Z_C = \langle \mathbf{C}, \mathbf{r} \rangle$$

$$Z_{AB} = \langle \mathbf{A}, \mathbf{B}^{\mathbf{r}} \rangle$$

Aggregation

Tools: MIPP & TIPP

Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

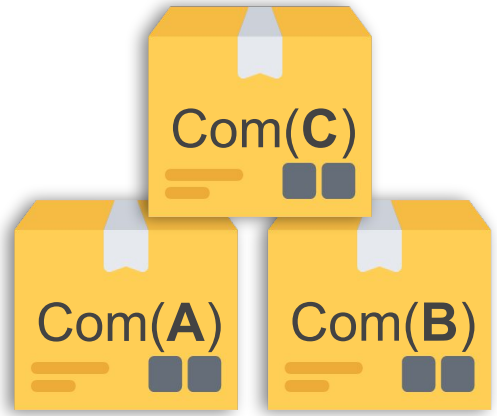


$$\langle \mathbf{C}, \mathbf{r} \rangle = \prod C_i^{r_i}$$

$$\langle \mathbf{A}, \mathbf{B}^{\mathbf{r}} \rangle = \prod e(A_i, B_i^{r_i})$$

Tools: MIPP & TIPP

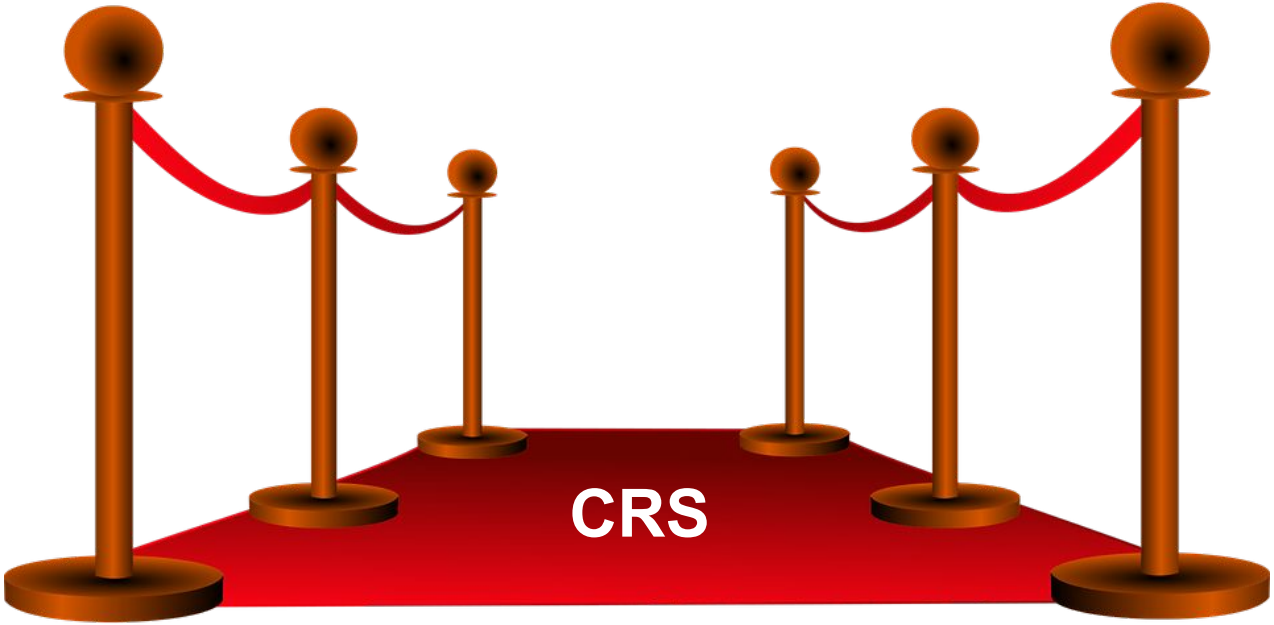
Proofs for Inner Pairing Products and Applications - Bünz, Maller, Mishra, Tyagi, Vesely

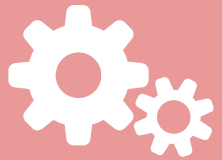


$$\langle \mathbf{C}, \mathbf{r} \rangle = \prod C_i^{r_i}$$

$$\langle \mathbf{A}, \mathbf{B}^{\mathbf{r}} \rangle = \prod e(A_i, B_i^{r_i})$$

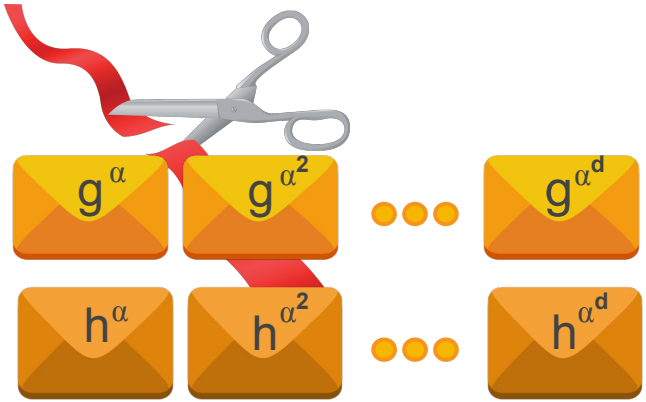
Trusted Setup



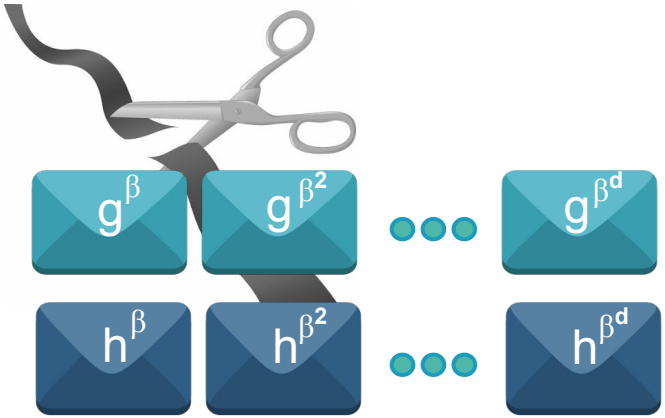


Aggregation from existing CRS

SNARK Aggregation



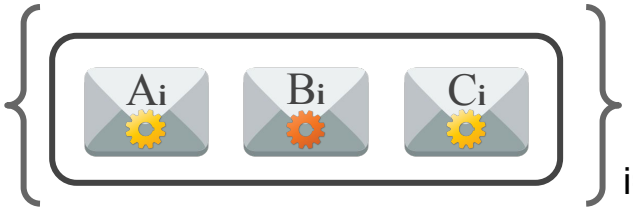
Filecoin: Powers of Tau



Zcash: Powers of Tau

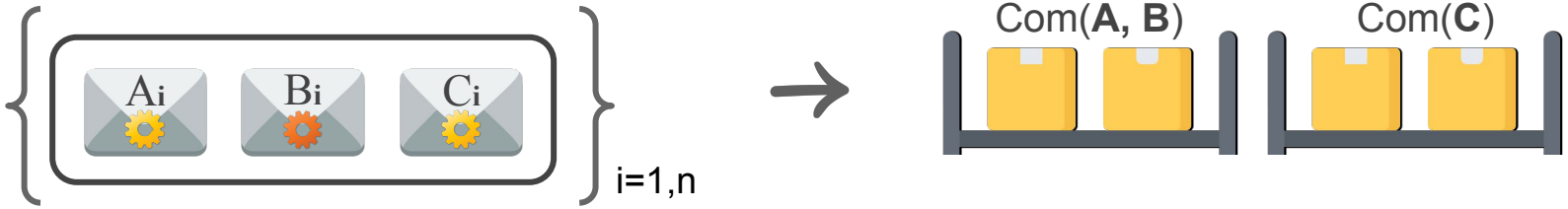
SNARK Aggregation

Aggregation



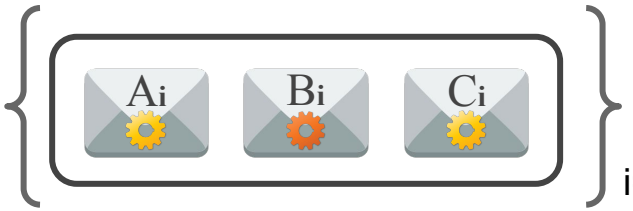
SNARK Aggregation

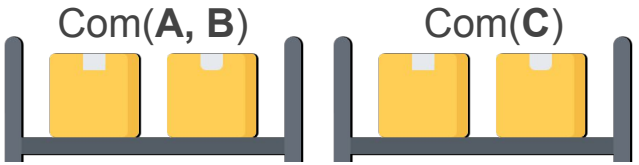
Aggregation



SNARK Aggregation

Aggregation





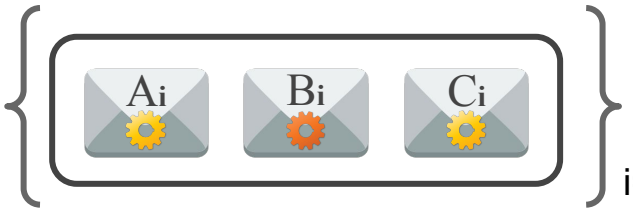
MIPP: $\langle \mathbf{C}, \mathbf{r} \rangle = \prod C_i^{r_i}$

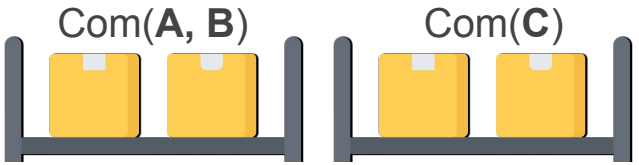
TIPP: $\langle \mathbf{A}, \mathbf{B}^{\mathbf{r}} \rangle = \prod e(A_i, B_i^{r_i})$

log n

SNARK Aggregation

Aggregation





$\text{MIPP:}\langle \mathbf{C},\mathbf{r}\rangle = \prod C_i^{r_i}$

$\text{TIPP: } \langle \mathbf{A}, \mathbf{B}^{\mathbf{r}}\rangle = \prod e(A_i, B_i^{r_i})$

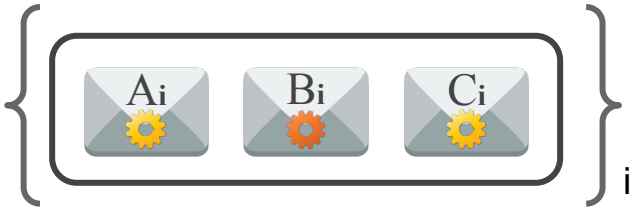
log n

$Z_C = \langle \mathbf{C}, \mathbf{r}\rangle$

$Z_{AB} = \langle \mathbf{A}, \mathbf{B}^{\mathbf{r}}\rangle$

SNARK Aggregation

Aggregation



i=1,n

$Z_C = \langle C, r \rangle$

$Z_{AB} = \langle A, B^r \rangle$

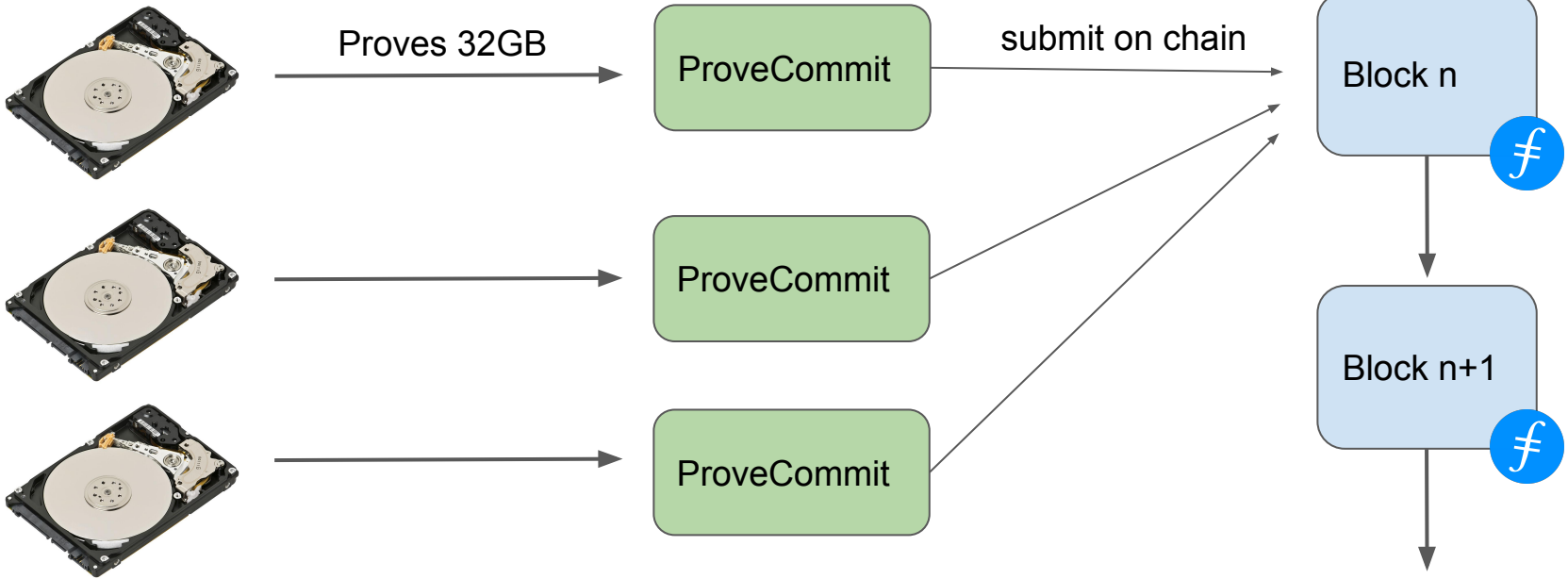
Verification

MIPP + TIPP

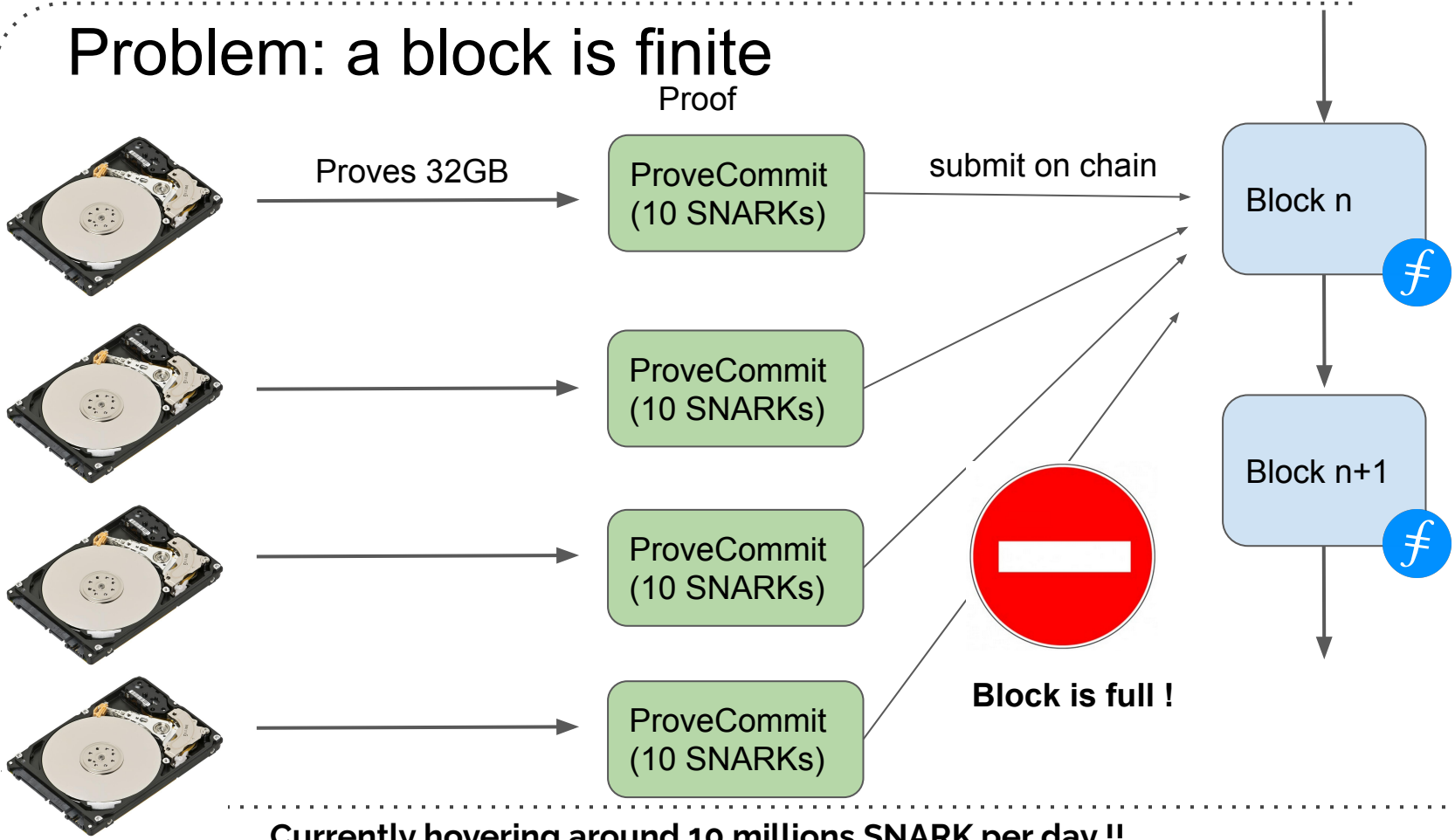
$Z_{AB} = e(Z_C, D)$

Application: Filecoin

Proof of storage

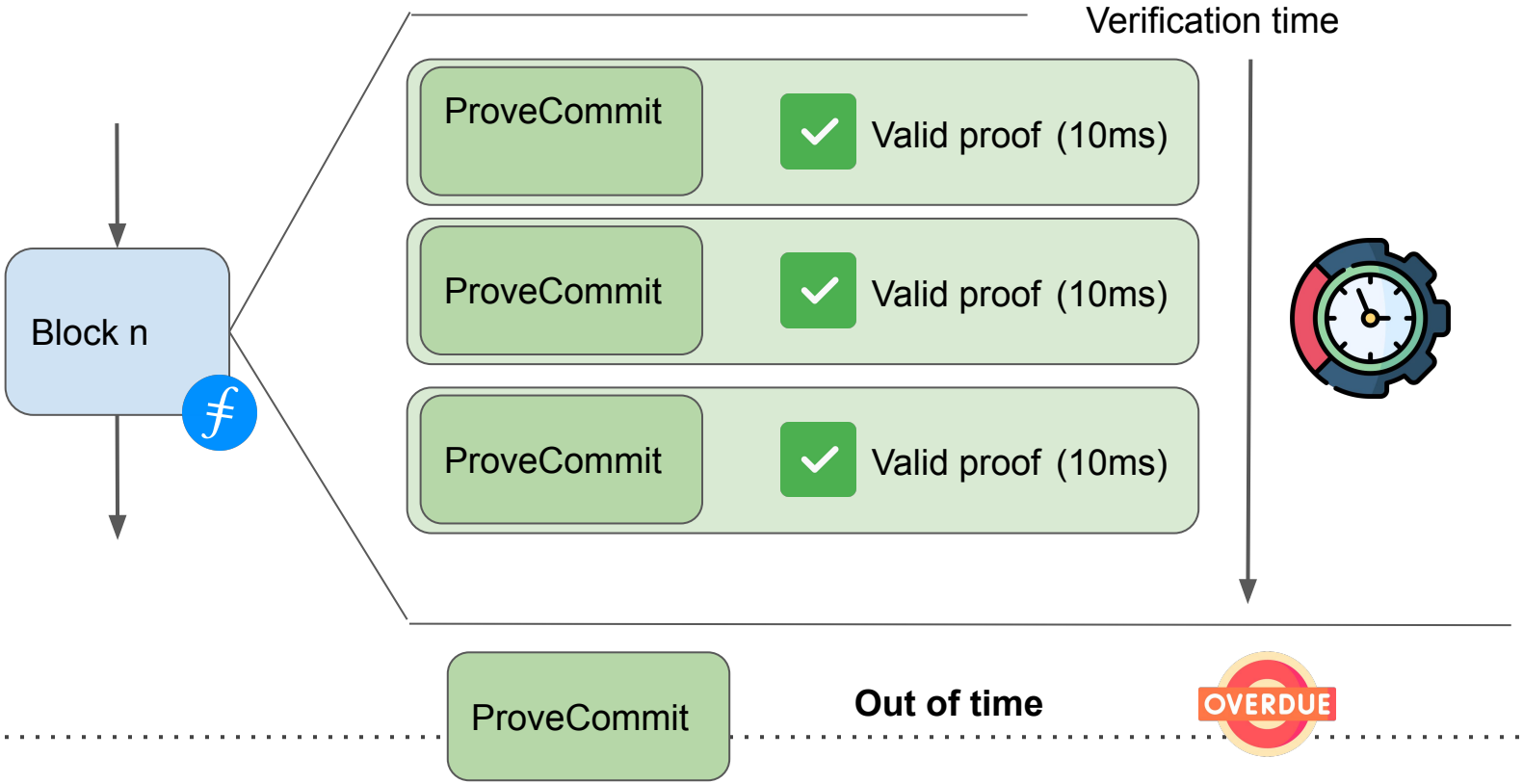


Problem: a block is finite



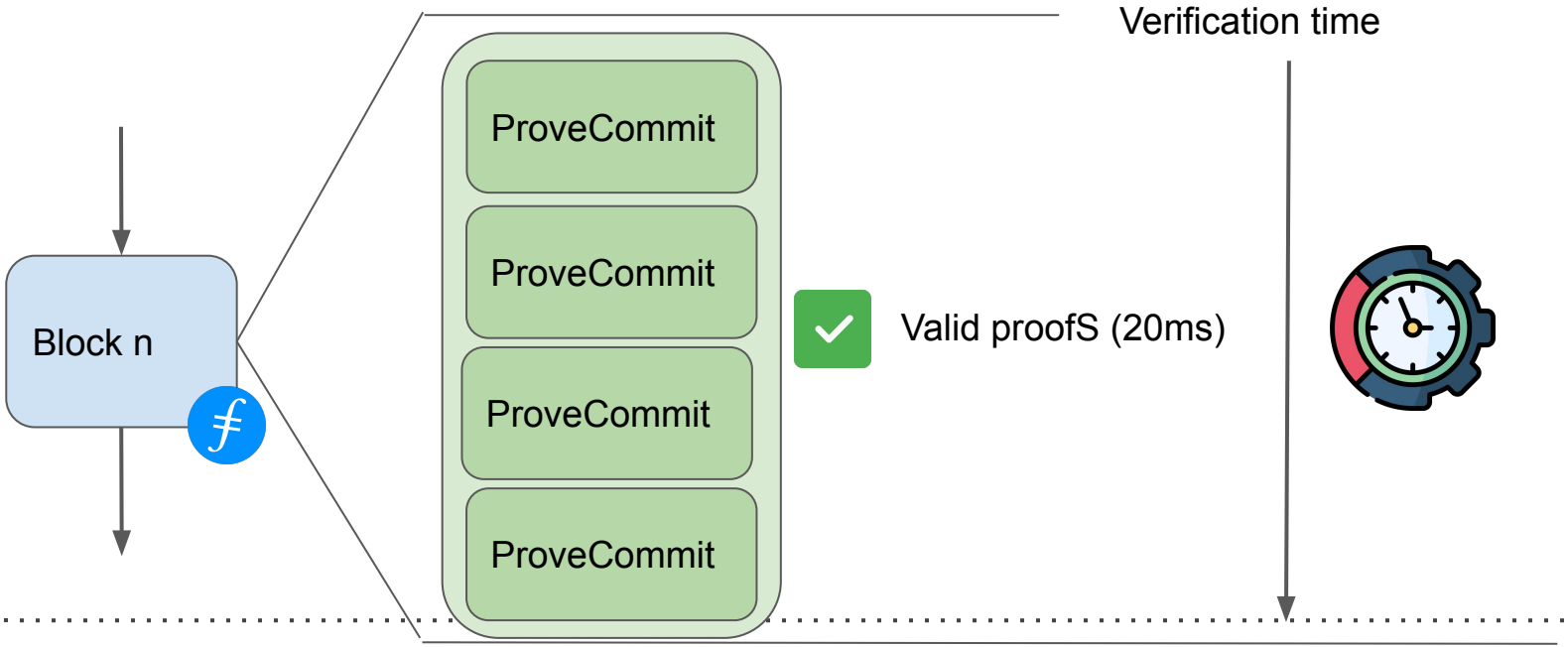
Currently hovering around 10 millions SNARK per day !!

Problem (2) : time/gas is finite



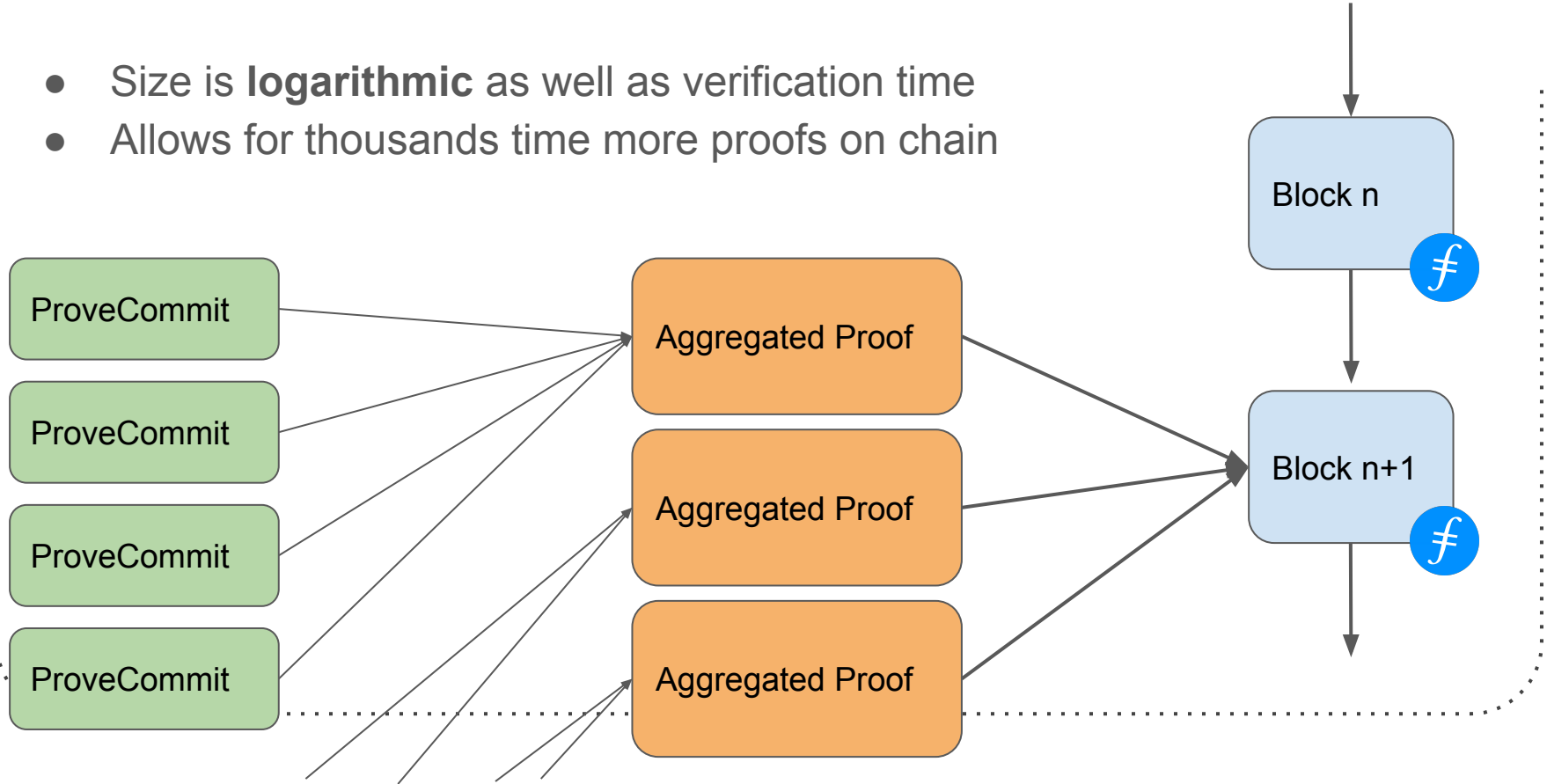
Solution (1) : Batch verification

Operations can efficiently be batched for faster verification

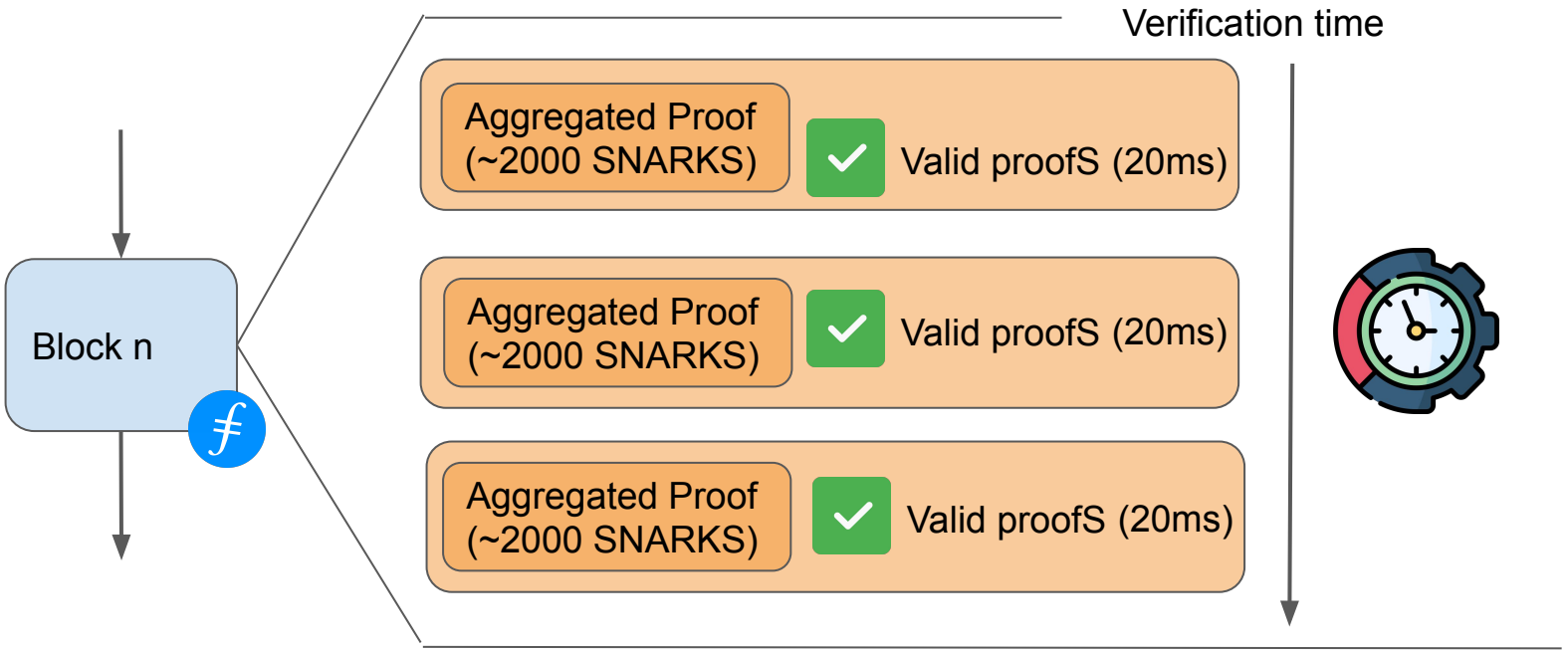


Solution (2) : Aggregation

- Size is **logarithmic** as well as verification time
- Allows for thousands time more proofs on chain



Aggregated proof verification



In practice, we start to go up to x200 more ProveCommits onchain (~x2000 SNARks)

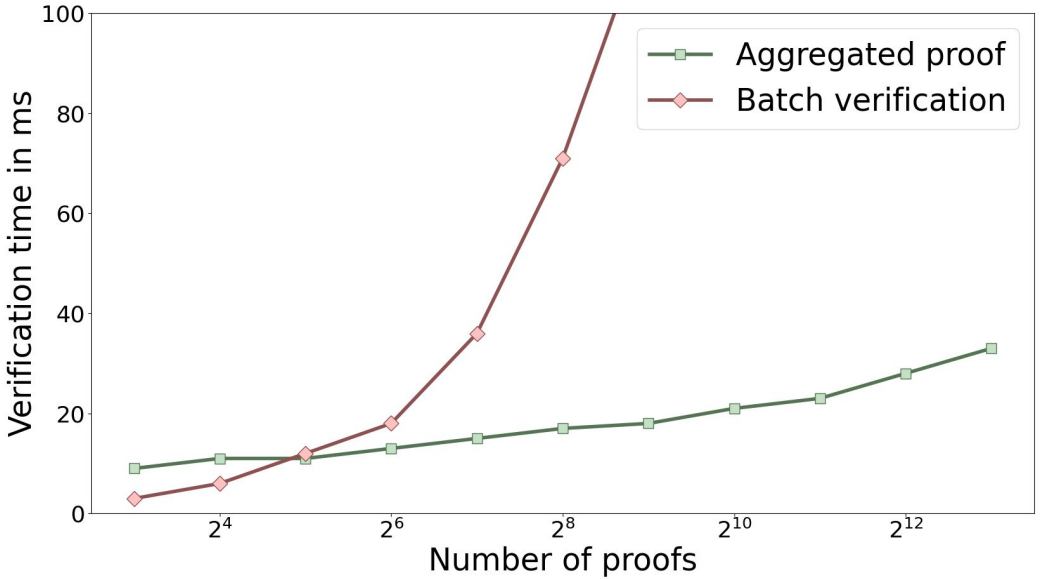
Implementation

Library

- Coded in **Rust,** available at https://github.com/filecoin-project/bellperson branch feat-ipp2
- Initial code from the **arkworks library** https://github.com/arkworks-rs/ripp/
- Ported & optimized in the **bellman** framework (bellperson fork)
- Using **BLS12-381 curves** from the **blst library** https://github.com/supranational/blst
- **SRS** combined from Filecoin & Zcash power of taus
  - Code at https://github.com/nikkolasg/taupipp
  - Up to 2^19

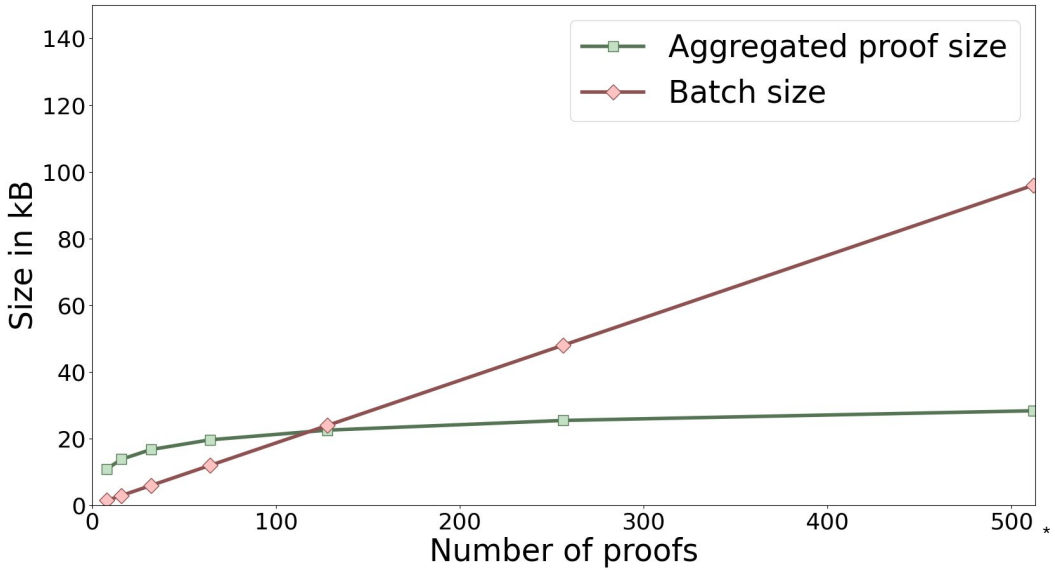
- Benchmark performed on 32c/64t **AMD Raizen Threadripper**
- **Audited** by NCC and second audit in progress by Matteo Campanelli

Verifier Time



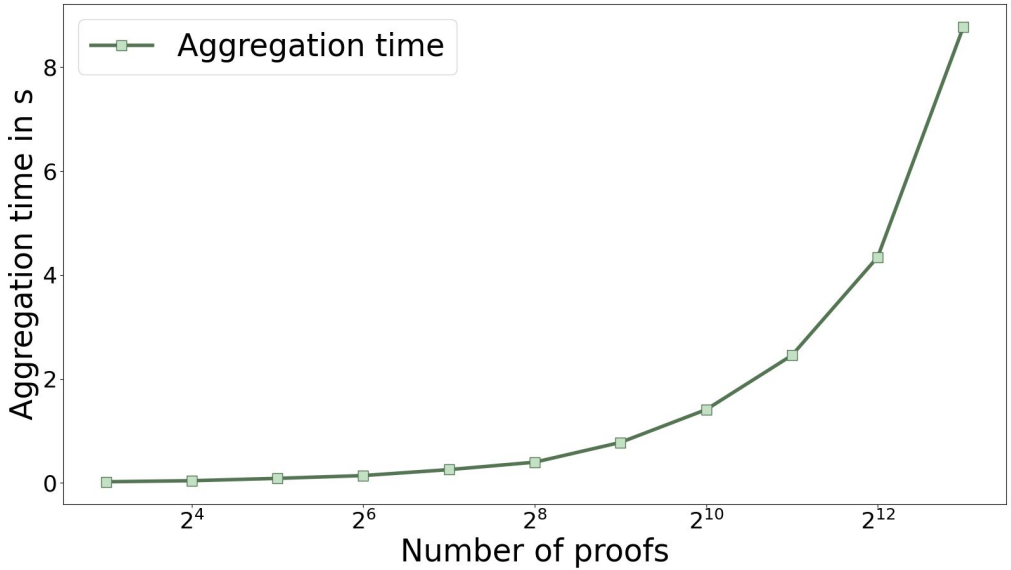
- Verifying aggregate proofs becomes **faster** from **32 proofs**.
- **8192 proofs** in **33ms**
  - “ratio” of 0.004 ms per proof
- **Including unserialization**
- Relies heavily on parallelism

Proof size



- Use **compression** of G_T points
  - Based on Taurus compression
  - From RELIC library implementation
  - 10log(n) + 5 G_T in proofs
- Turnover at 128 proofs
  - 23kB for aggregated
  - 24kB for “all proofs”

Aggregation Time



- 8.7s for 8192 proofs
- Relies heavily on parallelism

- 2^17 proofs in ~2mn

Thanks!

Any questions?

eprint.iacr.org/2021/529

Credits

Special thanks to all those who made and released these resources for free:

- Presentation template by SlidesCarnival
- Illustrations by Iconfinder