Fast finality notes

Barnabé Monnot, Protocol::Architecture

protocol.ethereum.foundation

L2 NS event, 04.02.2026

(many slides taken from Fradamt <3)

Why fast finality

Consistent demand from L2s! Upshot:

- Rollups with shorter runaheads
- Decreasing cost of capital
- Native interop speed accelerates
- Also, fix issues with current consensus

Fast Conf Rule can help some,
but fast BFT-style finality still critical
(highest security, provable onchain)

Also: Fast L1 slots

More of a demand for the L1. Upshot:

- Better UX!
- Increases censorship-resistance (assuming same validator set)
- Better onchain markets, DEX prices refresh, less losses to arbs
- Some interop improvements for L1 <> L2

IT'S ONE FINALITY ROUND MICHAEL, HOW LONG CAN IT BE ?

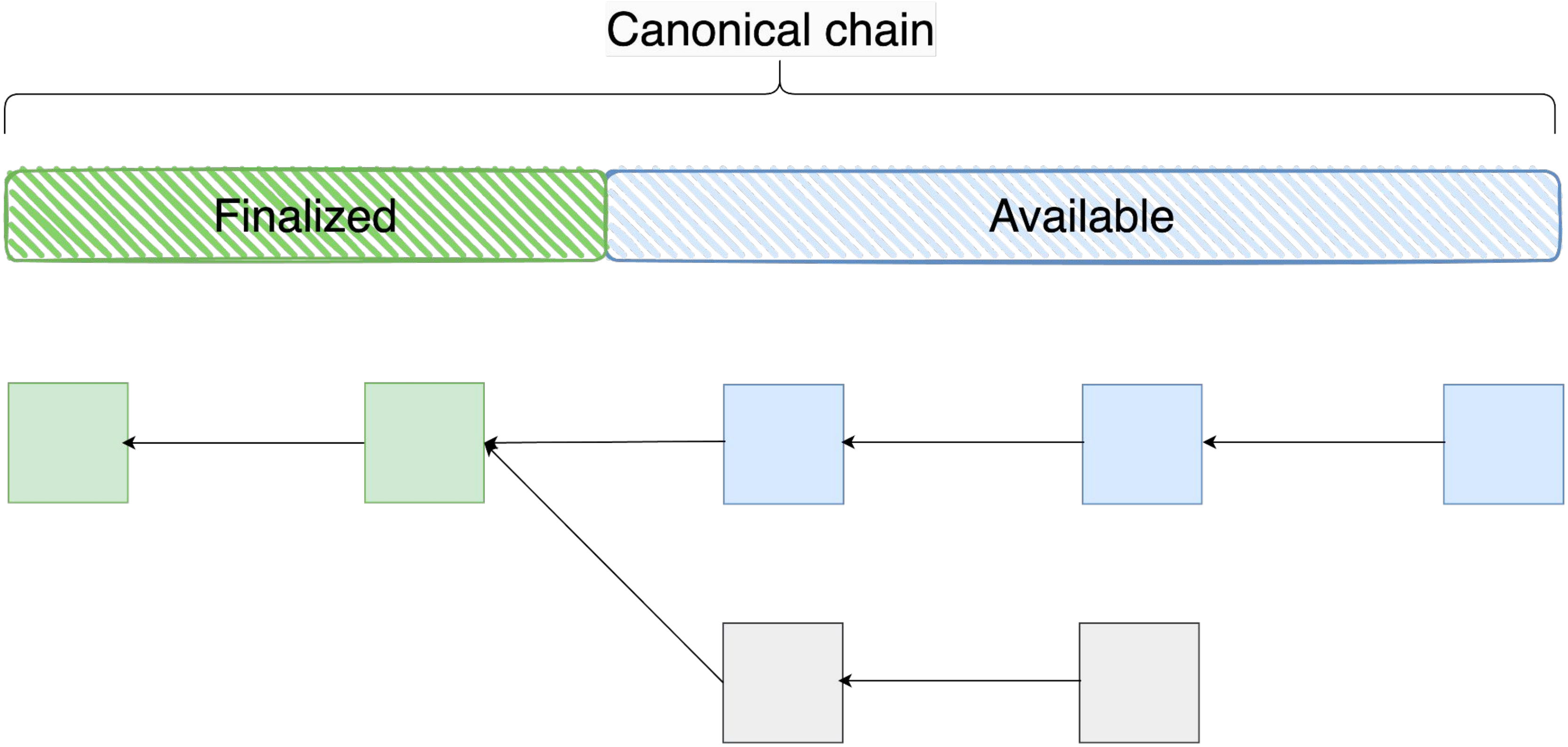
Today

- Finality time: ~13-19 minutes :(:(
- Slot time: 12 seconds :(:(

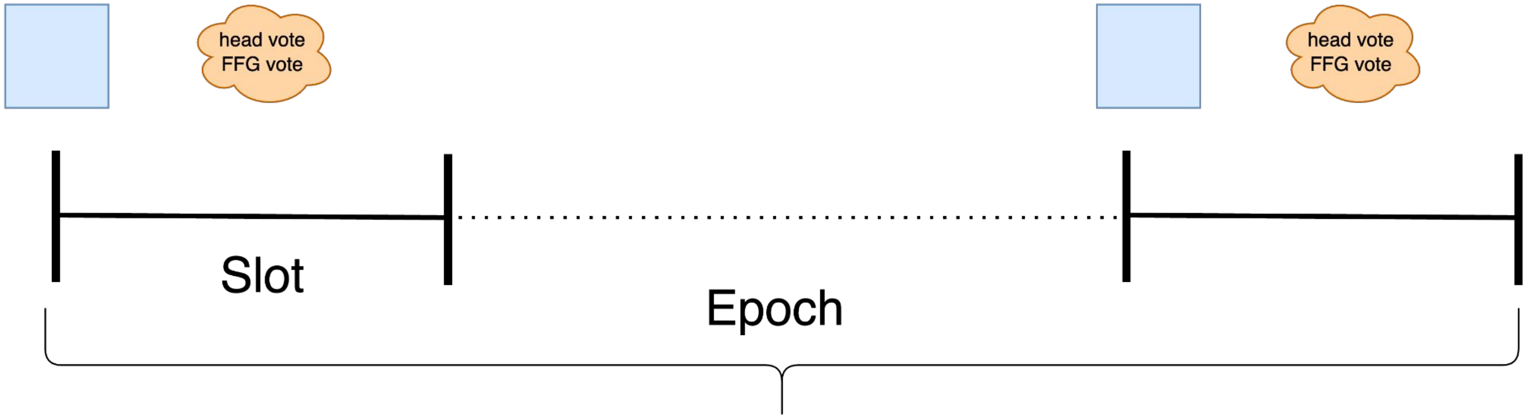
How do we go lower?

Coupled protocols

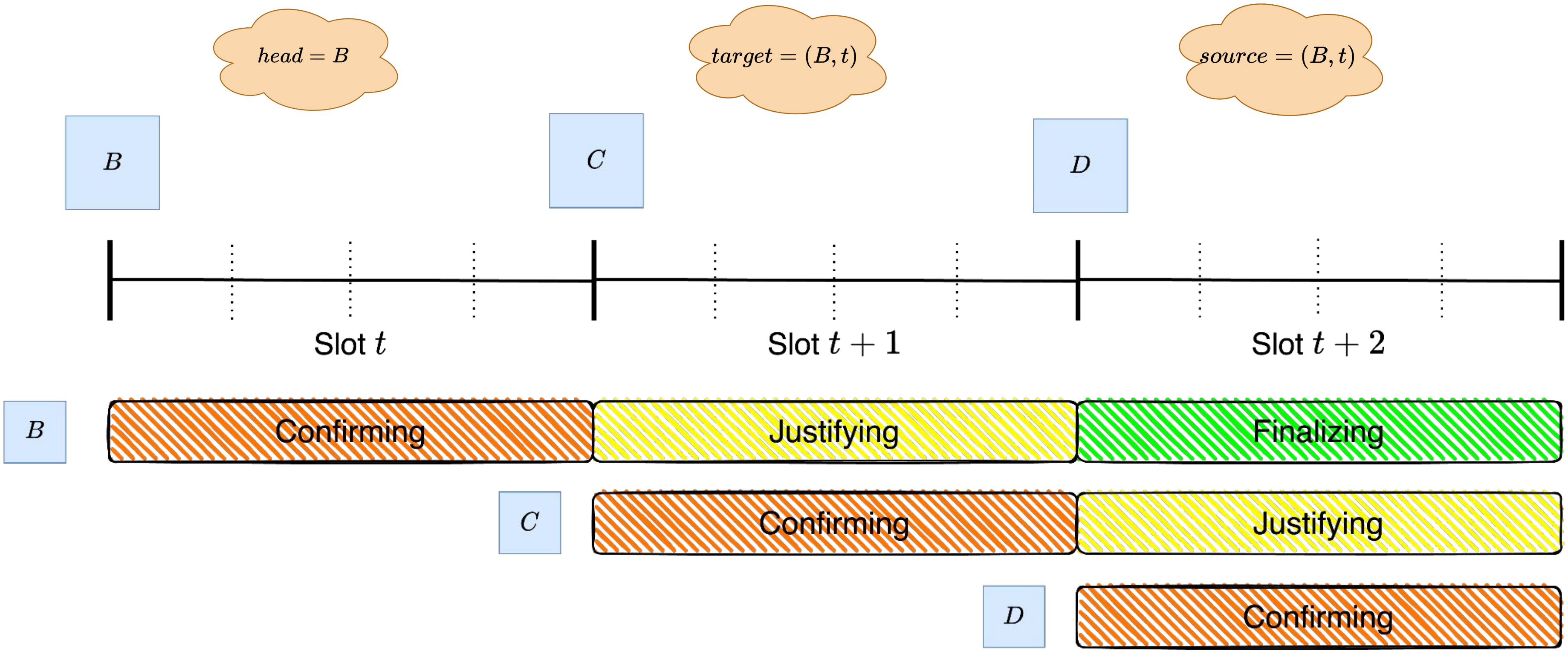
Available chain + Finalised chain



Gasper

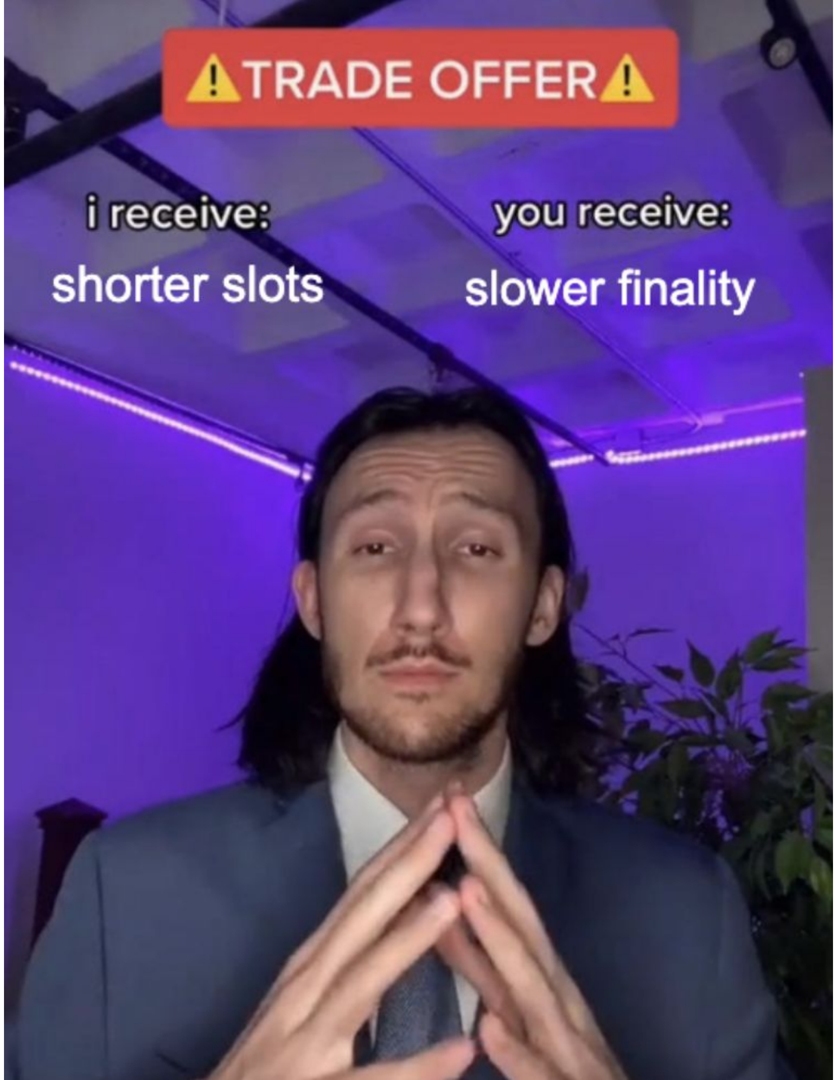


3-Slot-Finality



Coupled protocols

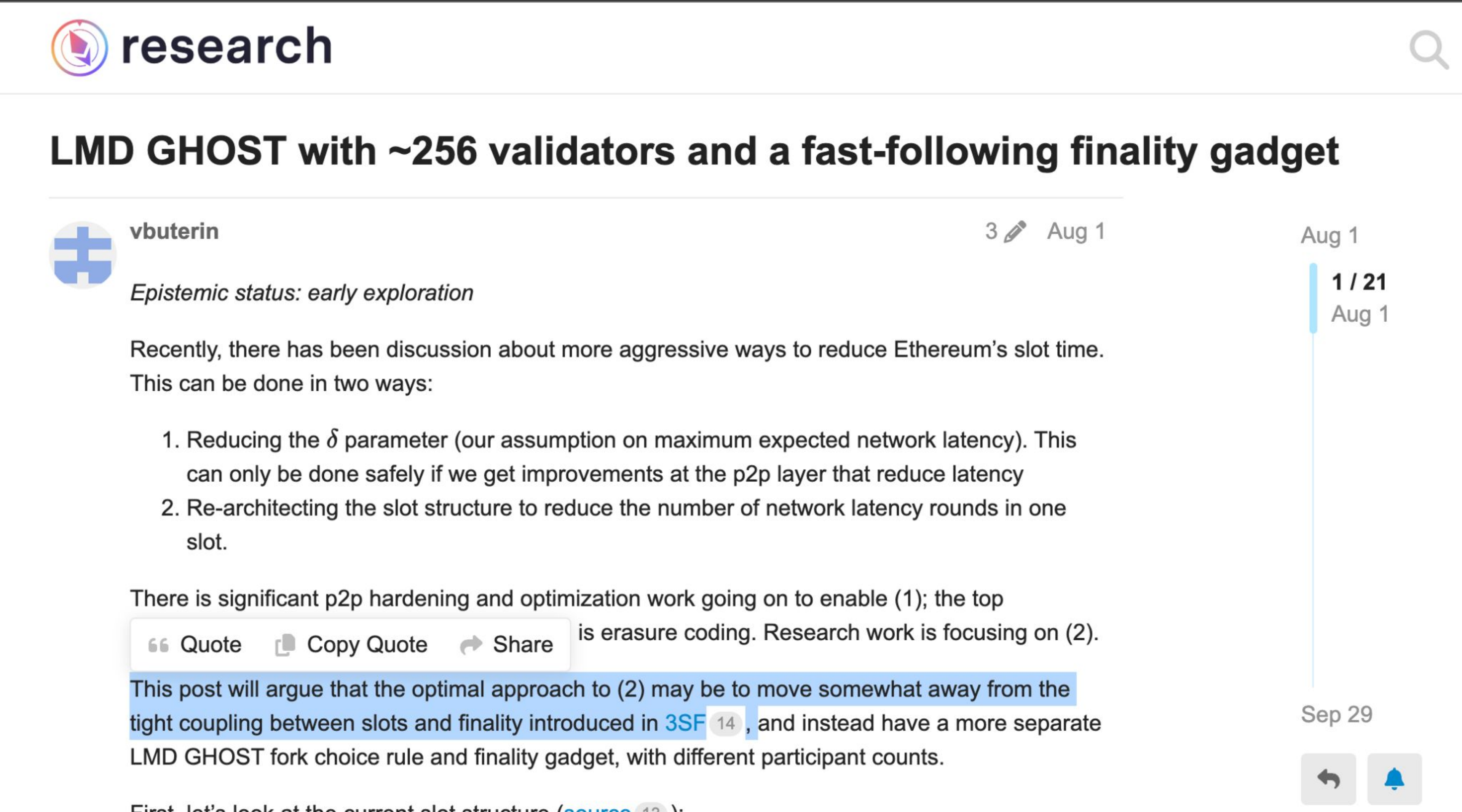
- Slot time includes block production + consensus logic
    - Increase block frequency => Reduce consensus throughput :(
    - Increase consensus throughput => Reduce block frequency :(


⚠️TRADE OFFER⚠️
i receive:
shorter slots
you receive:
slower finality

Coupled protocols

- Slot time includes block production + consensus logic
  - Increase block frequency => Reduce consensus throughput :(
  - Increase consensus throughput => Reduce block frequency :(
- Can we break the dilemma?

Decoupled protocols



LMD GHOST with ~256 validators and a fast-following finality gadget

Epistemic status: early exploration

Recently, there has been discussion about more aggressive ways to reduce Ethereum's slot time. This can be done in two ways:

1. Reducing the δ parameter (our assumption on maximum expected network latency). This can only be done safely if we get improvements at the p2p layer that reduce latency
2. Re-architecting the slot structure to reduce the number of network latency rounds in one slot.

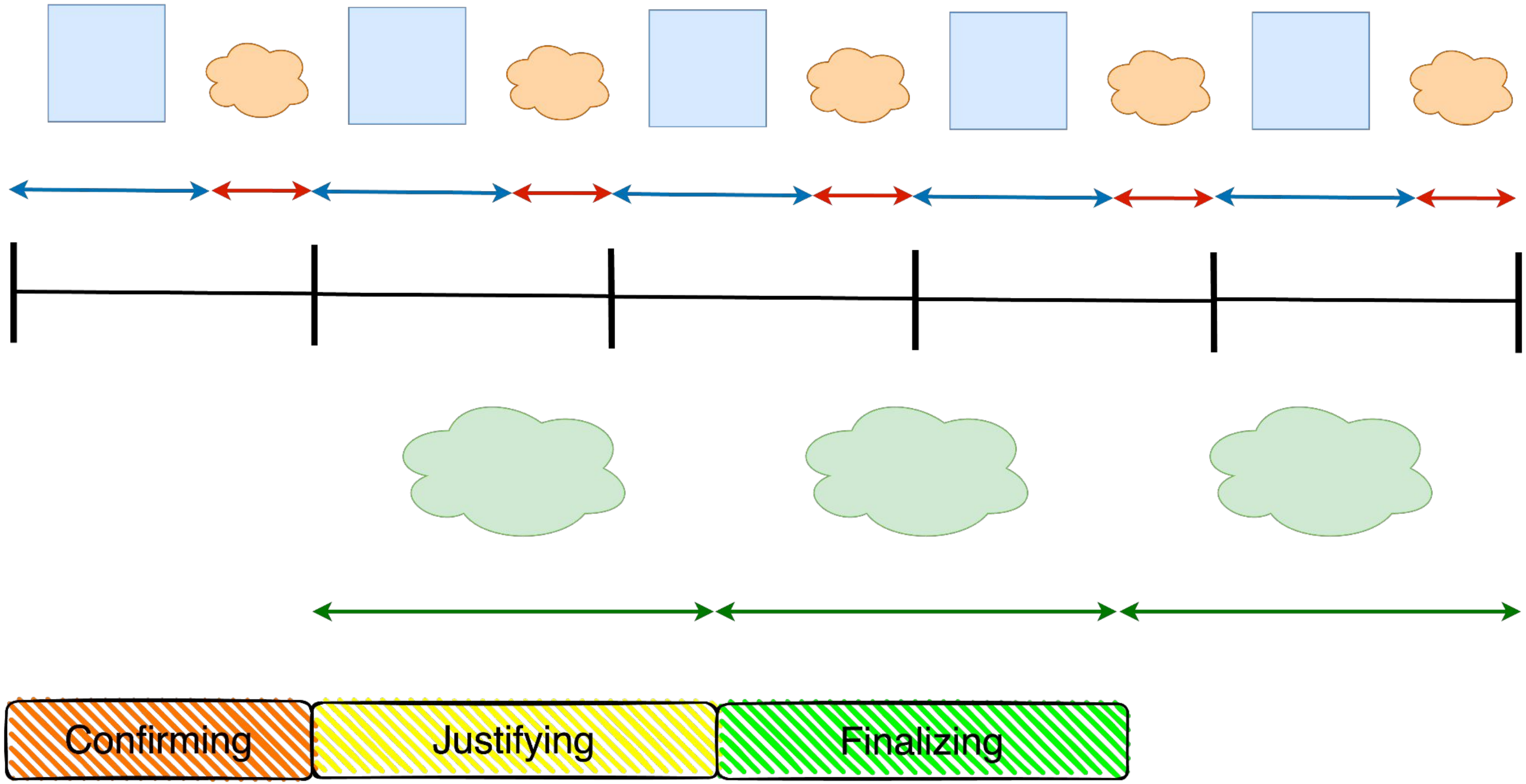
There is significant p2p hardening and optimization work going on to enable (1); the top

is erasure coding. Research work is focusing on (2).

This post will argue that the optimal approach to (2) may be to move somewhat away from the tight coupling between slots and finality introduced in 3SF 14, and instead have a more separate LMD GHOST fork choice rule and finality gadget, with different participant counts.

First, let's look at the current slot structure (source 13):

Independent timelines



Independent timelines: Upshot

- Optimise slot time and finality time independently
- Not blocking on sufficient validator consolidations
    - They would be absolute necessity for 3SF with short block times
    - Even if we don't *require* them, still want strong consolidations for low finality time

Consolidations

- In any world, consolidating vals is key
- Today, ~36,712k ETH staked
  - With 32 ETH per val, ~1.14m vals
  - But only 973k vals active! 🏆
- Continue outreach to staking pools
- Future: Could introduce direct incentives/penalties/mechanisms

How to ship

- Possible timeline
    - First step:
      Gasper + slot time reduction
    - Next steps:
      Decoupled protocol (faster finality)
    - End steps:
      Keep iterating on finality gadget for finality time reductions

“Product” questions

For some ⅙ < x < ⅓, are we ok with finality breaking if x% nodes are malicious/crashed?

Possible statement: “If all except ⅙ nodes are online, then we have finality with 2/9 (~22%) safety in one round.”

- One round = significant latency win
- But stronger finality assumptions
    - Not live if > ⅙ crashed
    - Safe up to ~22%

What finality time unlocks qualitatively new features? Are there phase transitions?

- Is there a difference between 2s and 4s?
- Between 4s and 10s?
- Between 10s and 120s?
- etc.

What is the target slot time?

- Higher than 4s?
- Between 2s and 4s?
- Lower than 2s?

The PQ question

- Eventually, need to switch val keys (BLS) to post-quantum ready schemes
  - Downside: Sig size increases
    => slower finality
  - Upside: Better aggregation properties
- Figure out how to sequence properly
  - Depends ultimately on security risk