# Proof of Stake (III)

## Finality Gadgets and PoS Challenges

Zeta Avarikioti
georgia.avarikioti@tuwien.ac.at

December 10th, 2025

TU WIEN

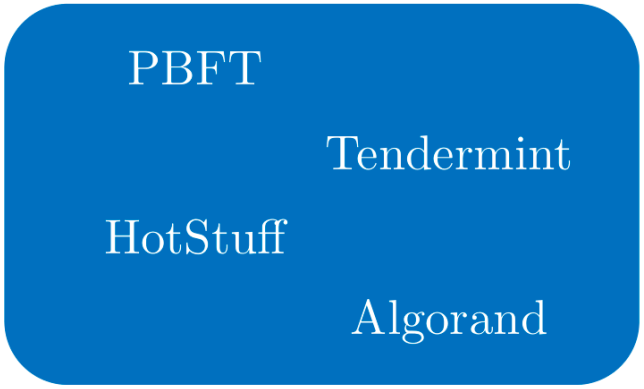
S&P Security & Privacy

# Overview

- Proof of Stake design space
- Ethereum 2.0
- Availability-Finality dilemma & finality gadgets
  - Origin of the dilemma
  - Ebb-and-flow property
  - Ethereum 2.0
  - Snap-and-chat protocols
- Costless simulation & long-range attacks
  - Challenges
  - Countermeasures

# The Availability-Finality Dilemma

# The tale of two protocols




Partially synchronous BFT protocols


Nakamoto-style longest-chain based protocols

# The tale of two protocols

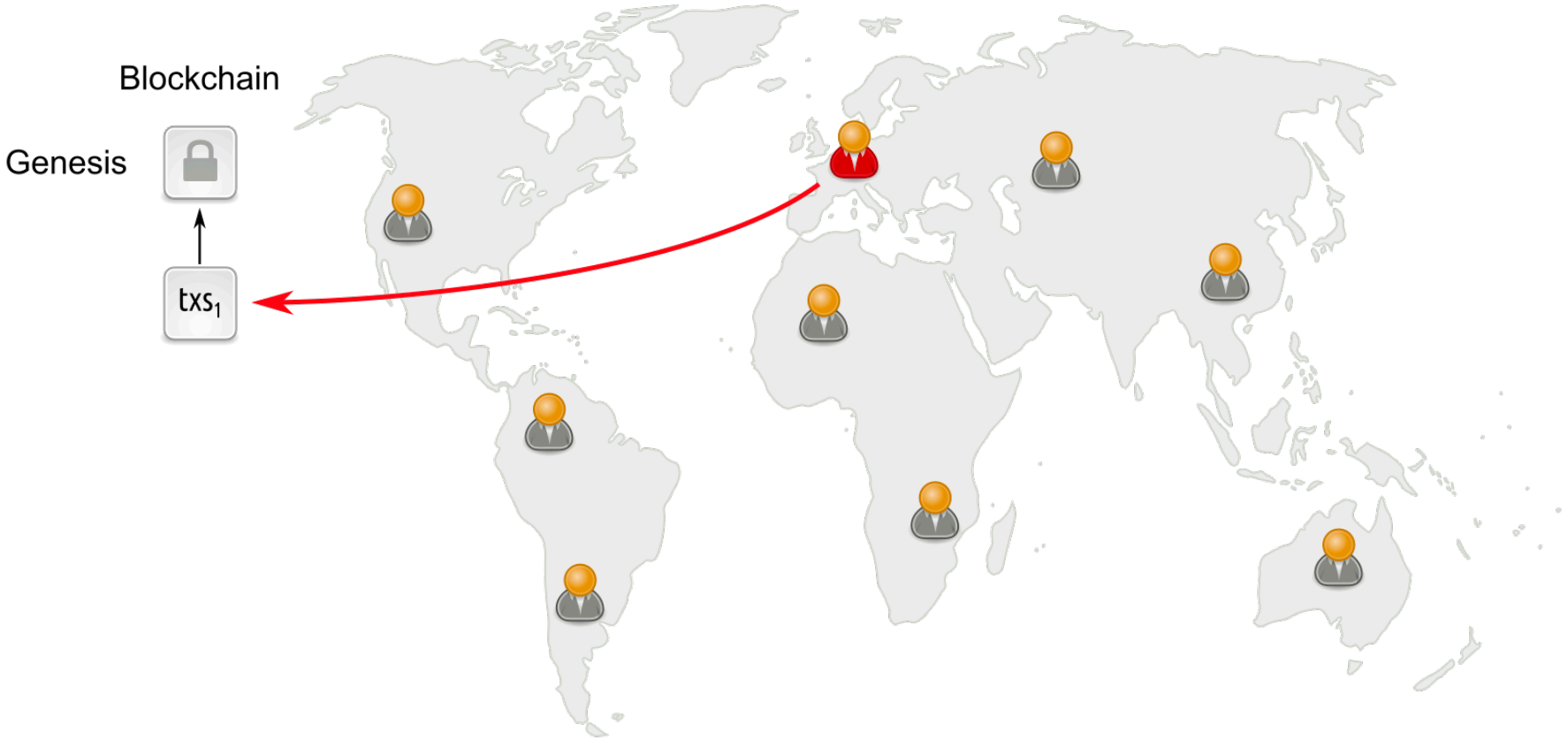
Bitcoin

Ethereum 1.0

Ouroboros

Nakamoto-style longest-chain based protocols

## Normal operation



# The tale of two protocols

Bitcoin

Ethereum 1.0

Ouroboros

Nakamoto-style longest-chain based protocols

Low participation

# The tale of two protocols

Bitcoin

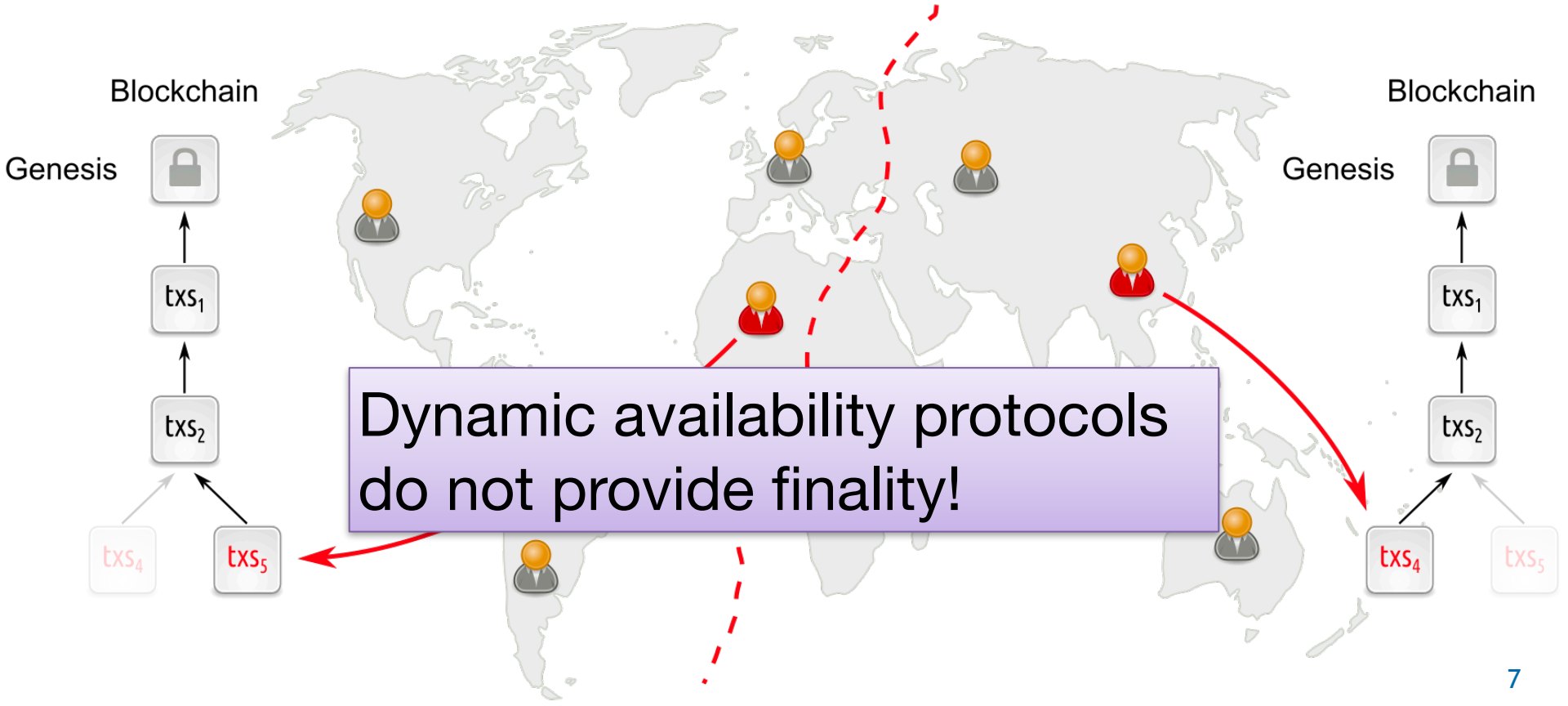
Ethereum 1.0

Ouroboros

Nakamoto-style longest-chain based protocols

Network partition

Indistinguishable from low participation!



# The tale of two protocols

PBFT

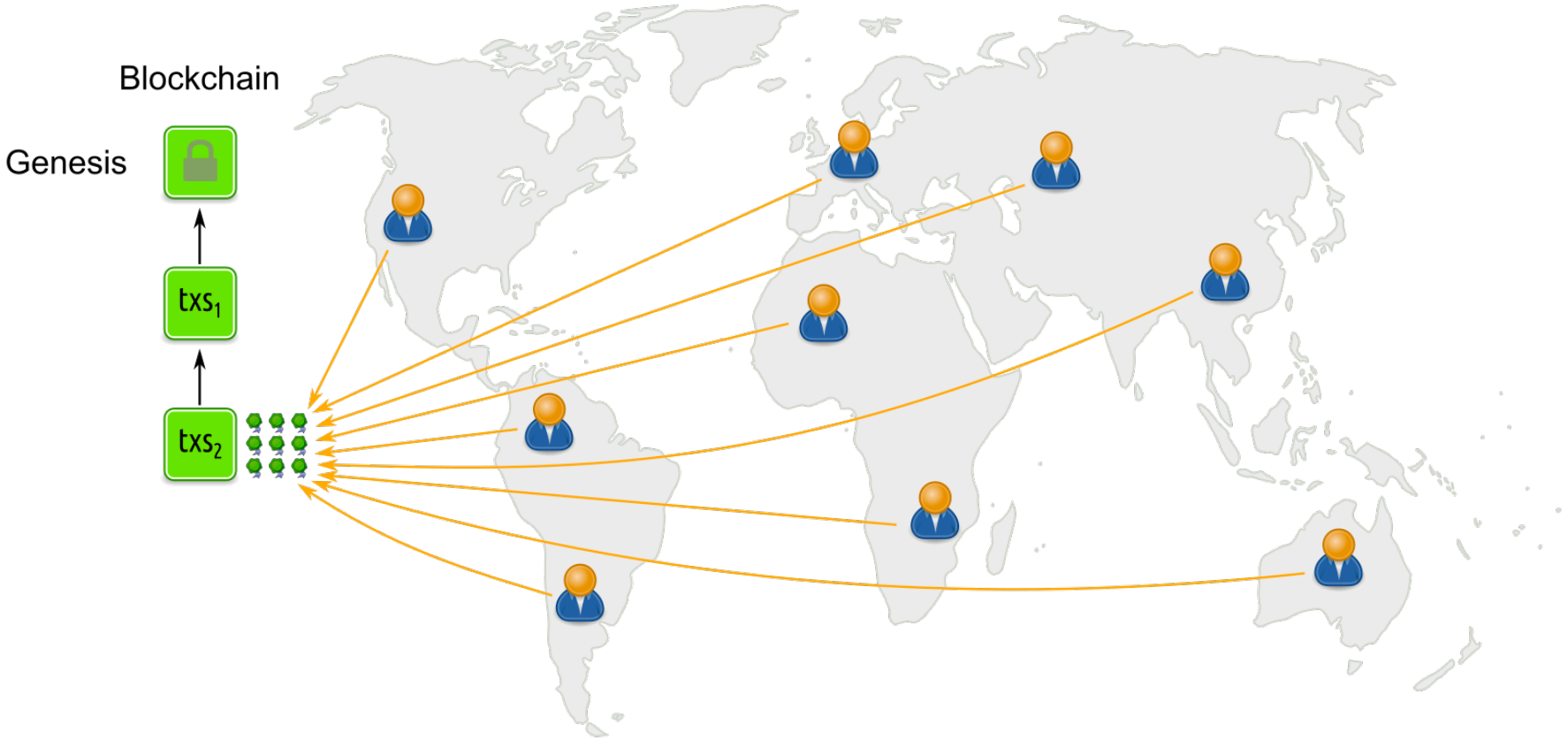
Tendermint

HotStuff

Algorand

Partially synchronous
BFT protocols

## Normal operation



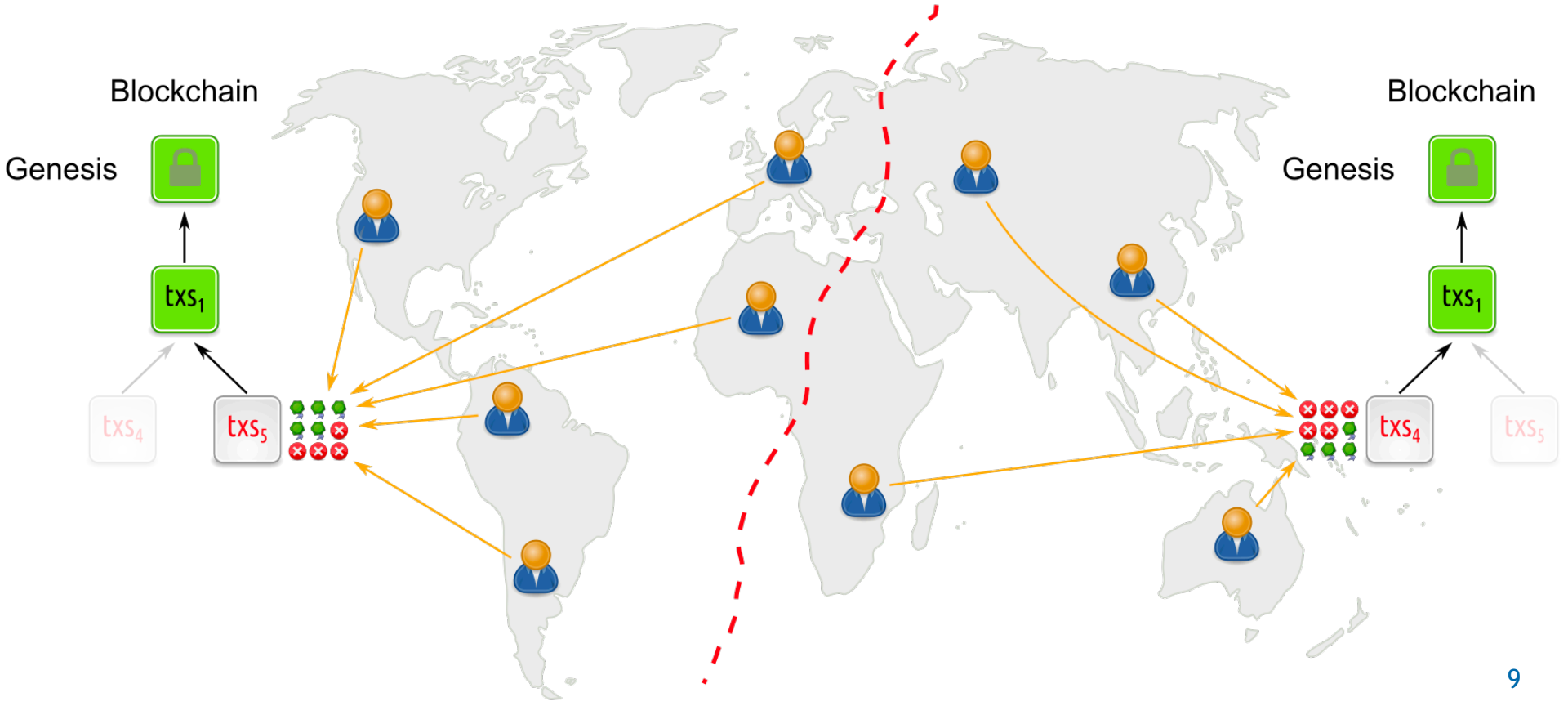
# The tale of two protocols

PBFT
Tendermint
HotStuff
Algorand

Partially synchronous
BFT protocols

Network partition

Loses liveness but maintain finality!

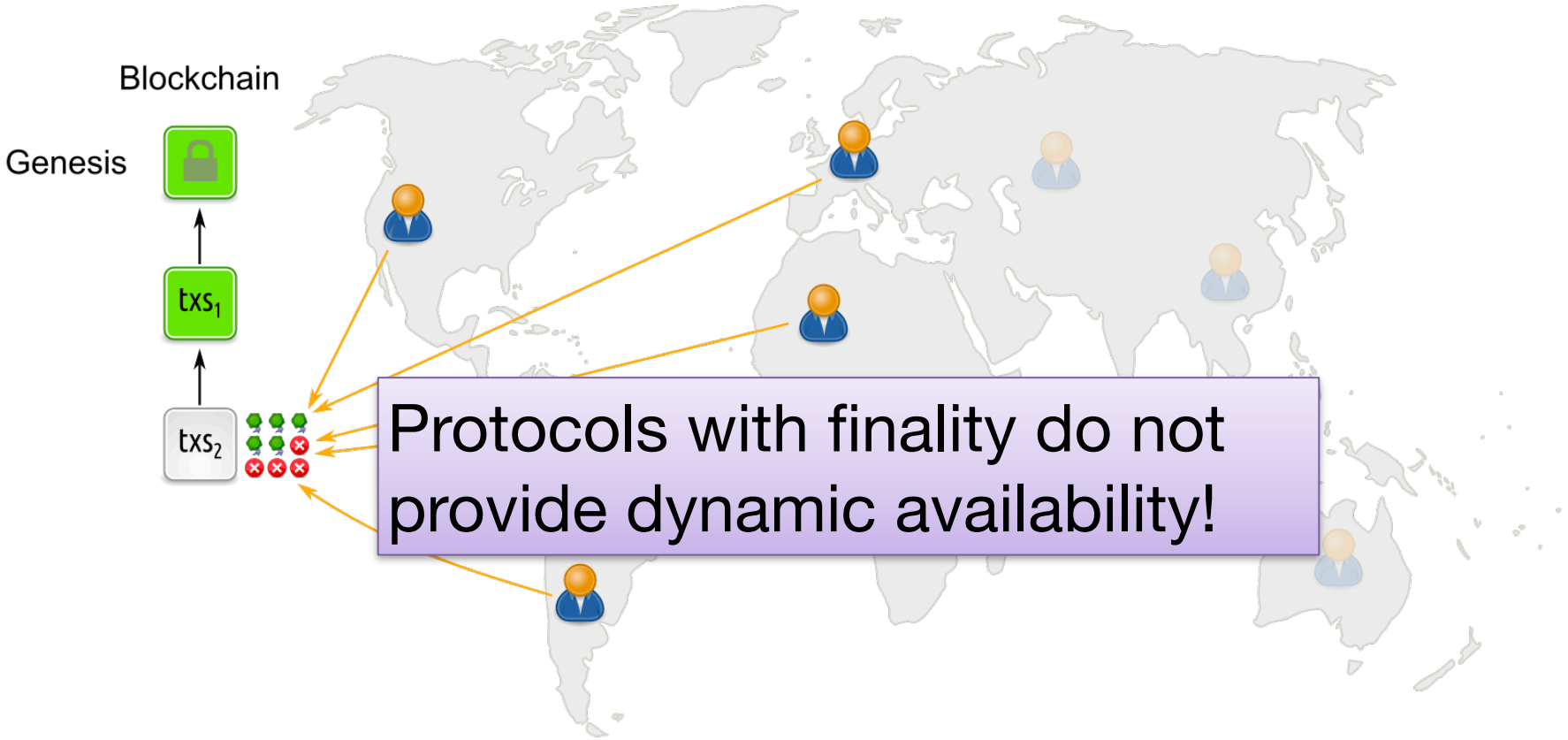


# The tale of two protocols

PBFT
Tendermint
HotStuff
Algorand

Partially synchronous BFT protocols

Low participation
Still no liveness :(

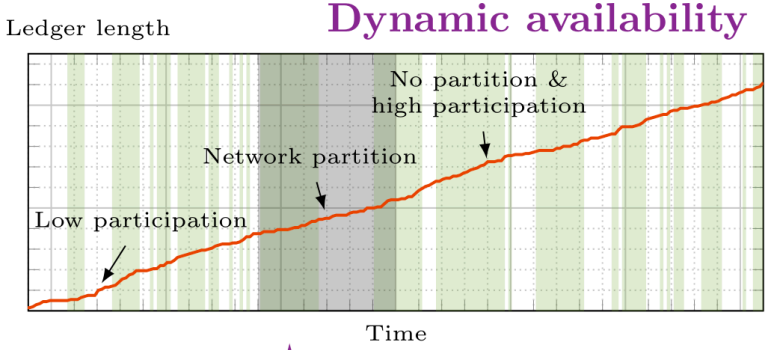


# From Protocols to Assumptions

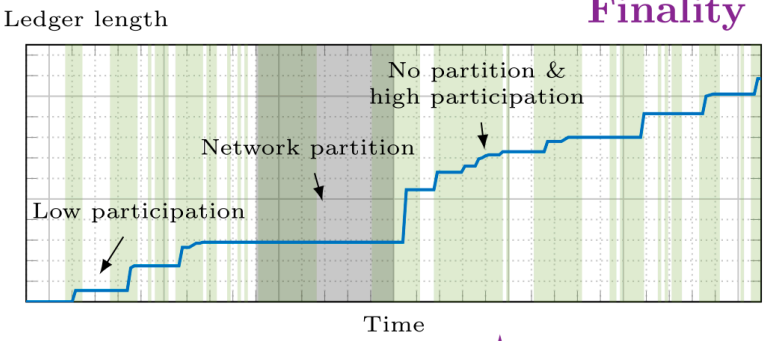
They
are just asleep :D

They
are plotting against
me…

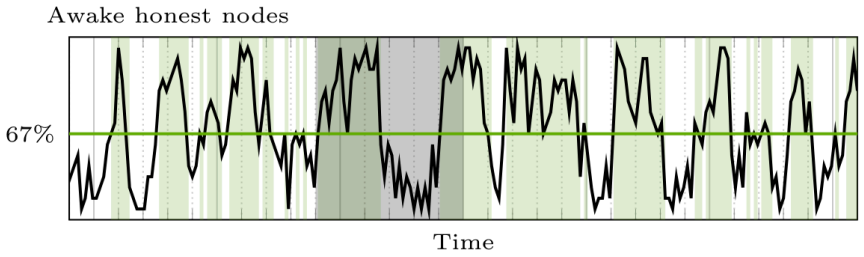
**Dynamic availability**




**Finality**

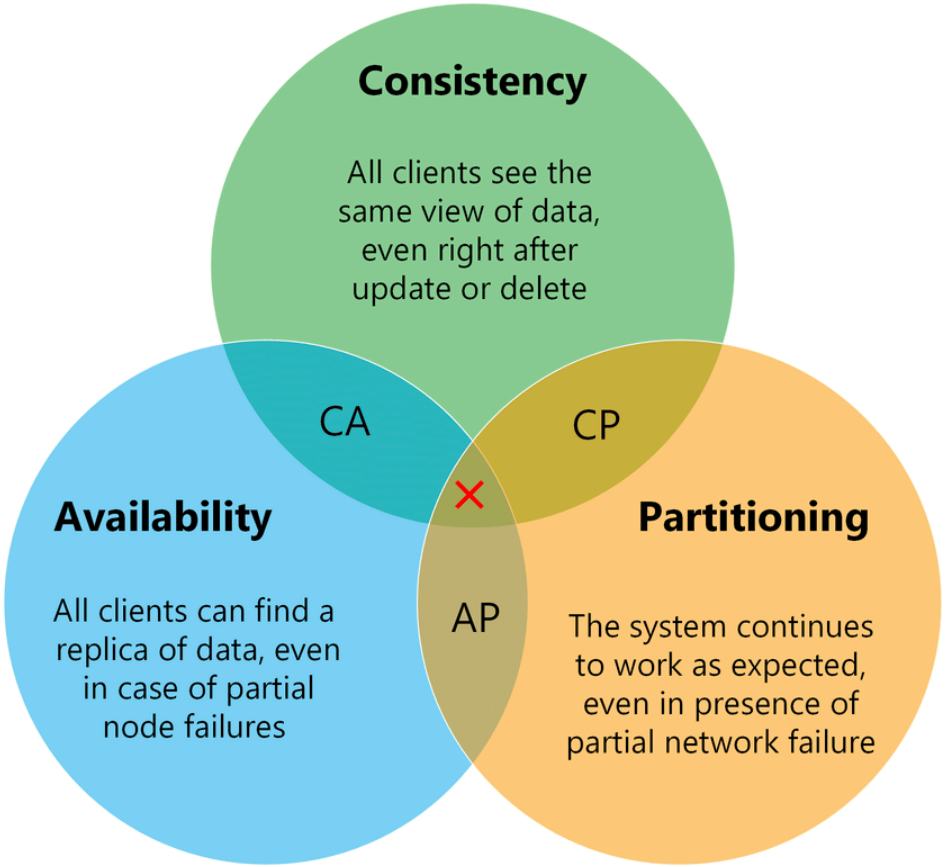



**Longest chain protocol**




**BFT protocol**

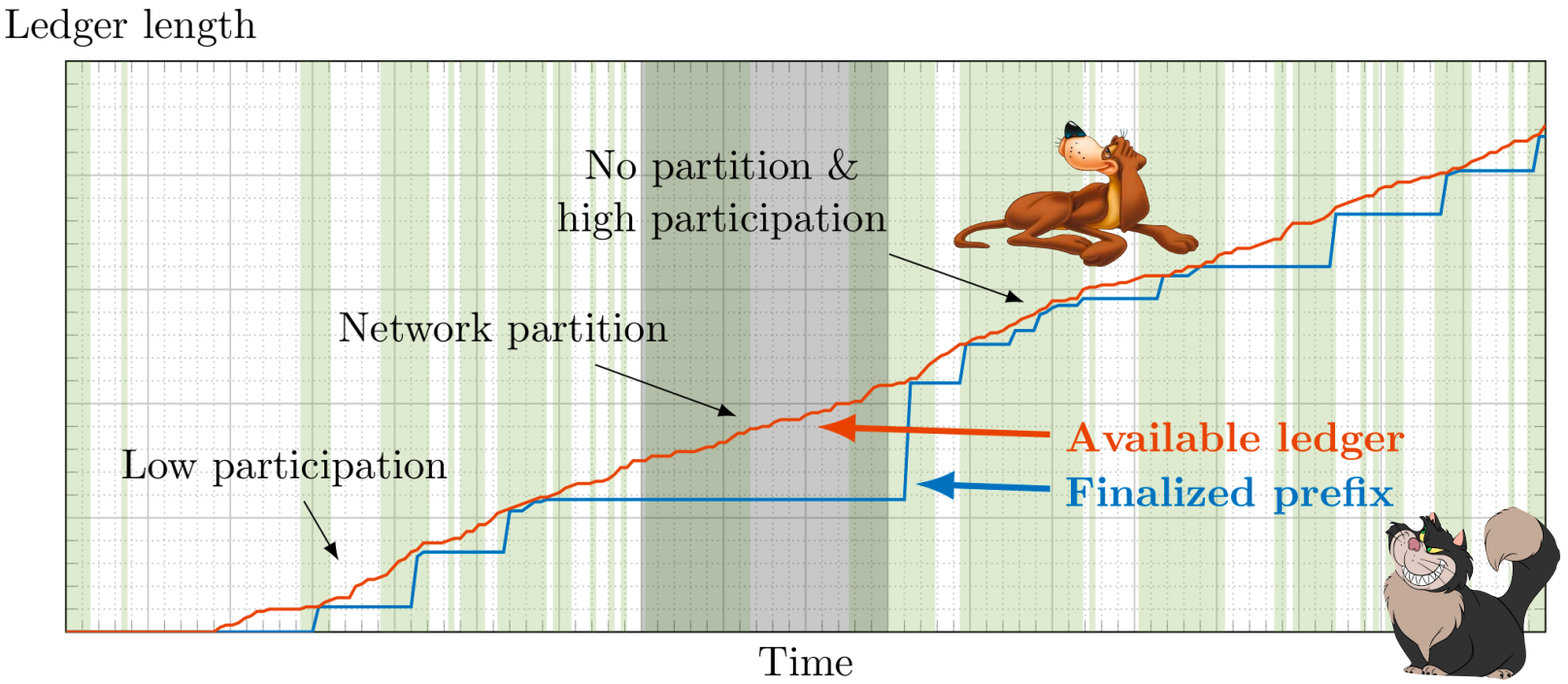
# Can we have both?




There cannot be a SMR protocol - one that outputs a *single* ledger - that provides both properties

# Ebb-and-Flow Property

- Design a protocol with *two confirmation rules* that outputs *two ledgers*
  - One ledger available under dynamic participation
  - Another ledger with finality under network partition
  - Both ledgers eventually agree on a common history!



# Ebb-and-Flow Property

- Setup
  - Asynchronous until GST after which it becomes synchronous.
  - Honest nodes sleep and wake up until *Global Awake Time* (GAT). Adversarial nodes are always awake.

- A protocol $\Pi$ is a $(\beta_1, \beta_2)$**-secure ebb-and-flow** protocol if it outputs an **available** ledger $LOG_{da}$ and a **finalized** ledger $LOG_{fin}$ satisfying:

1) **Finality:** $LOG_{fin}$ is safe at all times, and live after $max\{GST, GAT\}$, provided that less than $\beta_1$ portion of all nodes are adversarial.

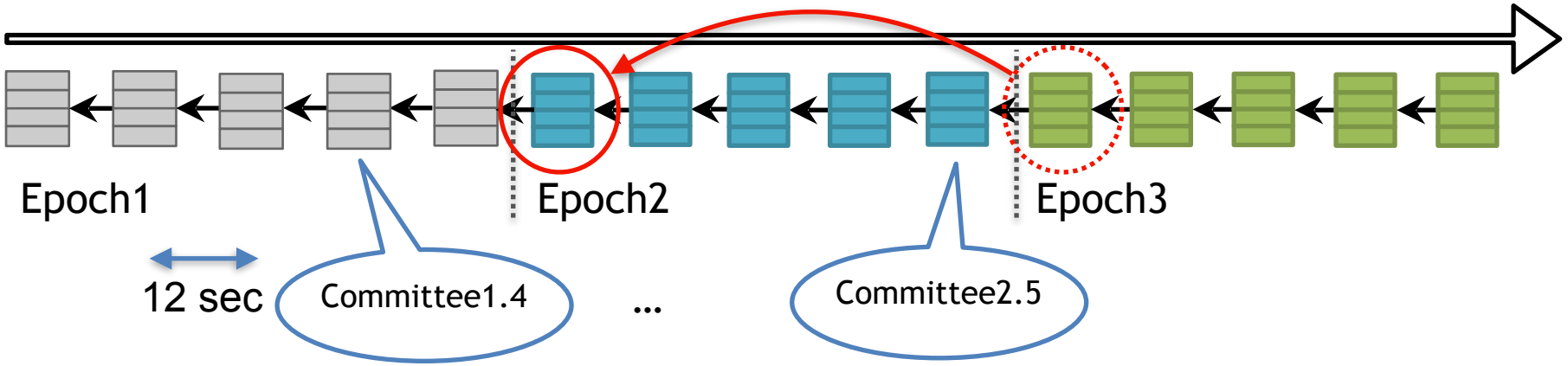
2) **Dynamic Availability:** If $GST = 0$, then $LOG_{da}$ is safe and live at all times, provided that at all times fewer than $\beta_2$ portion of the awake nodes are adversarial.

3) **Prefix:** $LOG_{fin}$ is a prefix of $LOG_{da}$ at all times.

Do Ebb-and-Flow protocols exist?

# Ethereum 2.0 (revisited)

# Ethereum recap




- Epoch = every 32 consecutive time slots of 12sec (6min 24sec)
- Every epoch, all staked validators are randomly partitioned to committees
- Each committee has one block proposer while the rest vote on the block
- If within 4sec, the proposer is inactive, committee votes for the previous block
- The fork choice rule is LMD GHOST and accounts for these votes
- A block is finalized when 2/3 of the validators in two consecutive epochs vote on it

# Ethereum 2.0: Gasper Protocol

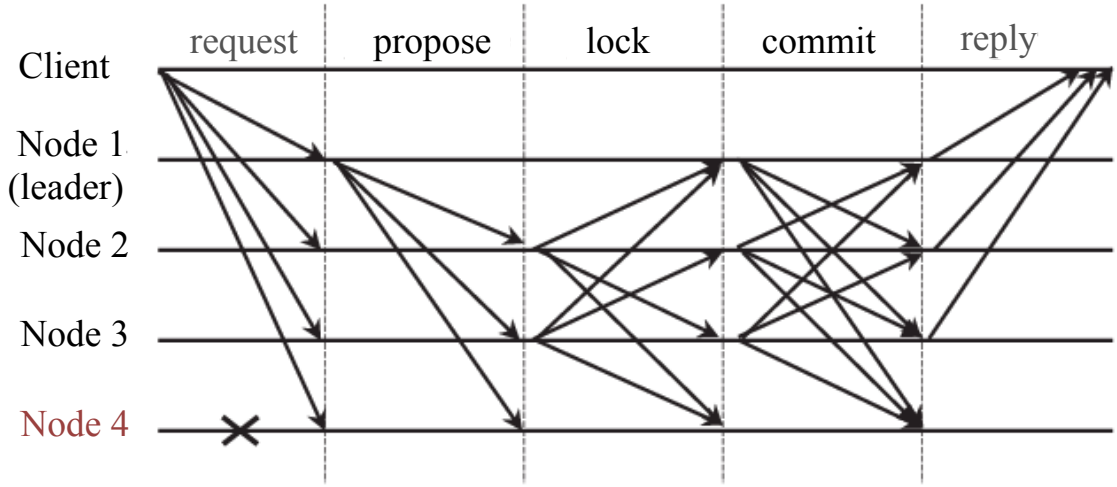
- Dynamically Available Protocol = LMD GHOST

- Finality Gadget (BFT) = Casper

# Ethereum Casper

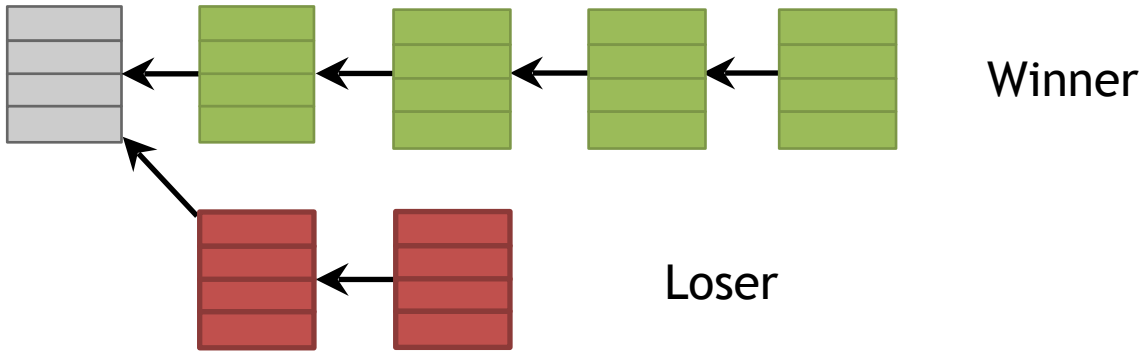
Casper the Friendly Finality Gadget (Chained Tendermint)

- Recall: leader waits Δ to receive messages from all (honest) nodes (after GST)
- **2 rounds of voting!**

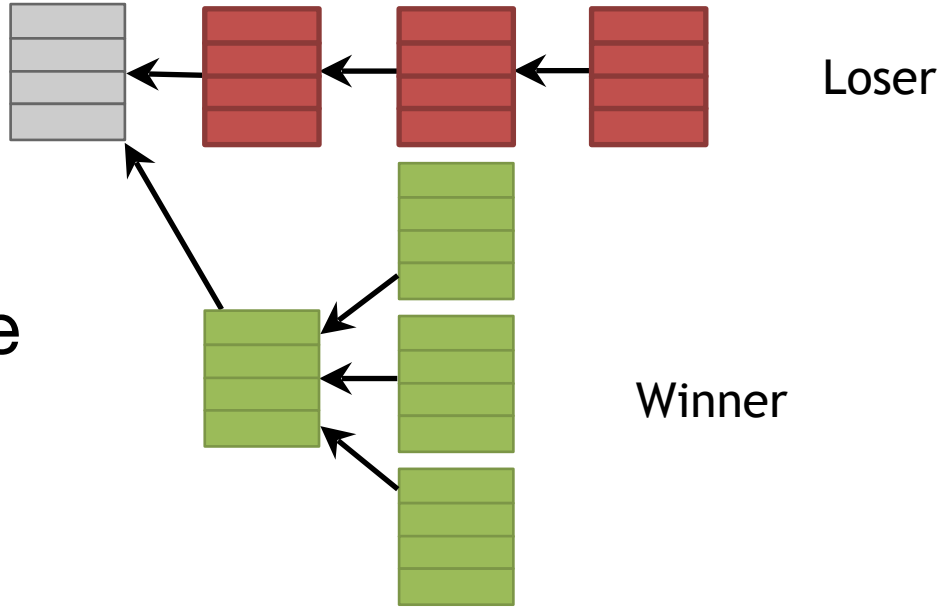


# GHOST

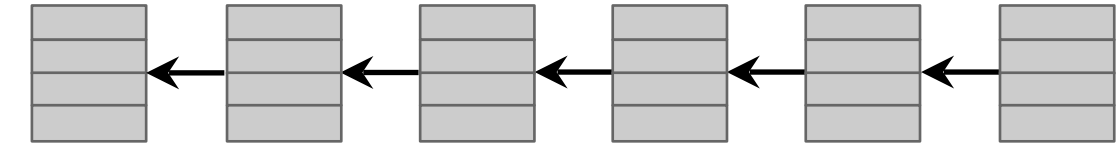
**Bitcoin:** Longest chain selection rule




**GHOST:**
Heaviest subtree



# Why not longest chain?



12 sec

Each validator slot worths 32ETH

~29.5% of ETH staked

Latency & economic safety

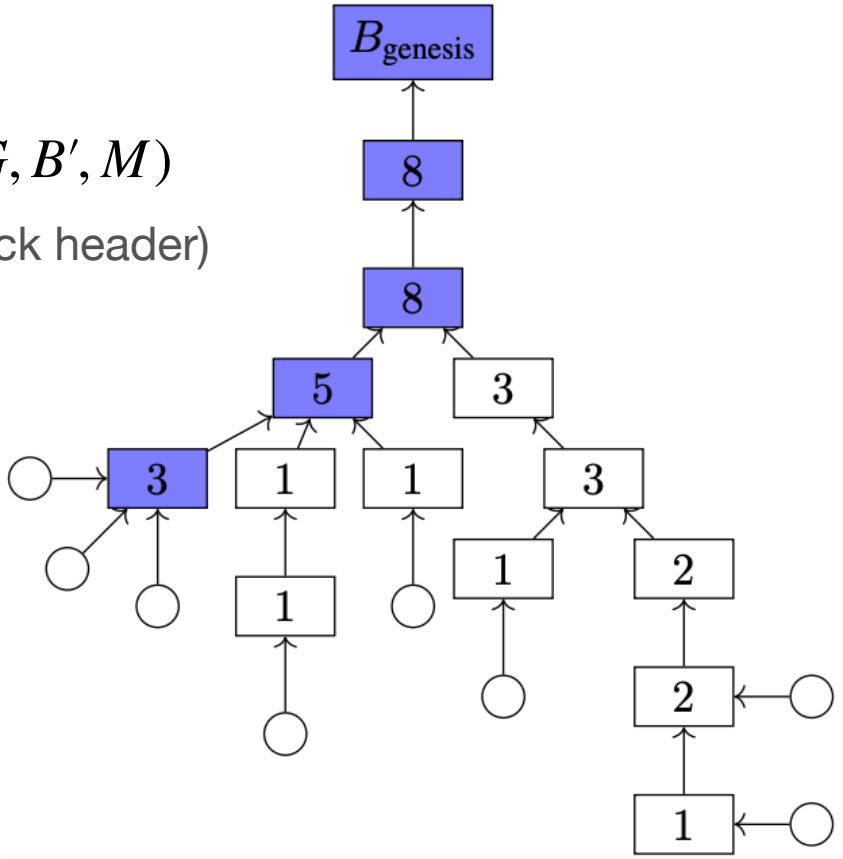
All validators will have voted when 1.000.000 blocks - 20 weeks!
Even 5% of validators would take ~1 week!

# LMD GHOST

## LMD GHOST fork choice rule

- $B \leftarrow B_{genesis}$
- $M \leftarrow$ the most recent attestation of the validators
- **while** B is not a leaf block in G **do**
  - $B \leftarrow argmax_{B' \text{ child of } B}\ w(G, B', M)$

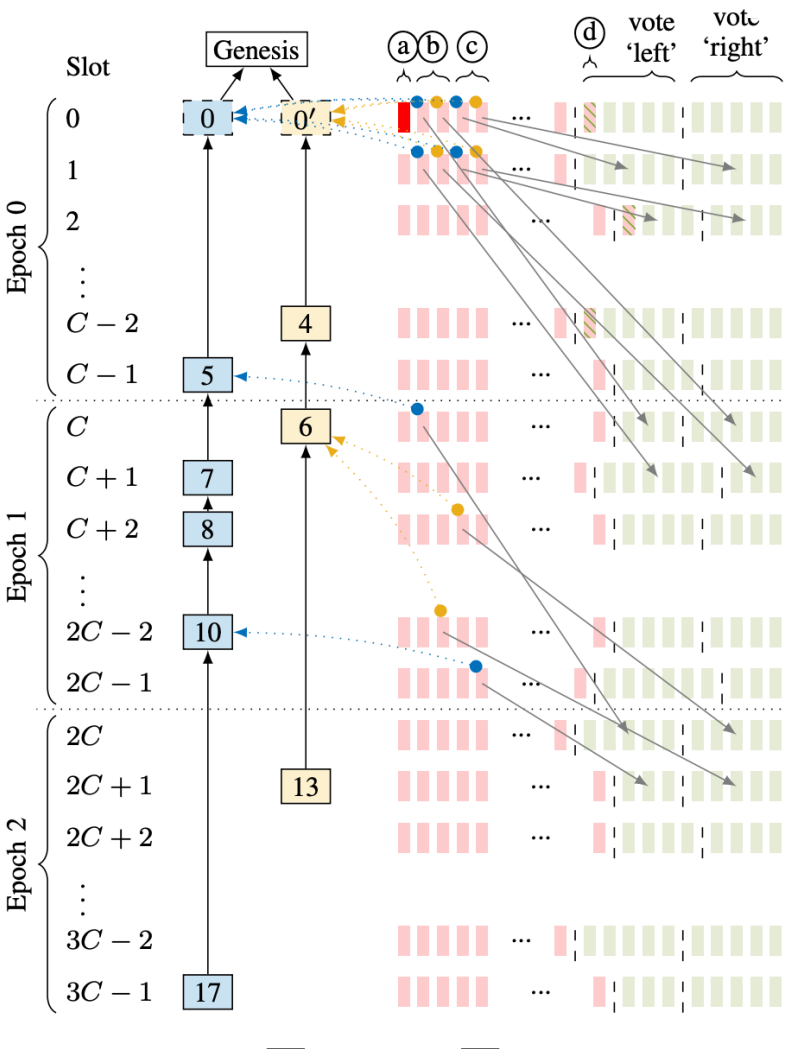
  (Ties are broken by hash of the block header)
- **return** B



# Balance Attacks

Secure in general?

- Balance attacks via manipulating the network
- sqrt(N) adversarial validators per slot suffice!




**Three Attacks on Proof-of-Stake Ethereum**


Caspar Schwarz-Schilling[1], Joachim Neu[2], Barnabé Monnot[1], Aditya Asgaonkar[1], Ertem Nusret Tas[2], and David Tse[2]

[1] Ethereum Foundation
{caspar.schwarz-schilling,barnabe.monnot,aditya.asgaonkar}@ethereum.org
[2] Stanford University
{jneu,nusret,dntse}@stanford.edu


**Two Attacks On Proof-of-Stake GHOST/Ethereum**


Joachim Neu jneu@stanford.edu
Ertem Nusret Tas nusret@stanford.edu
David Tse dntse@stanford.edu

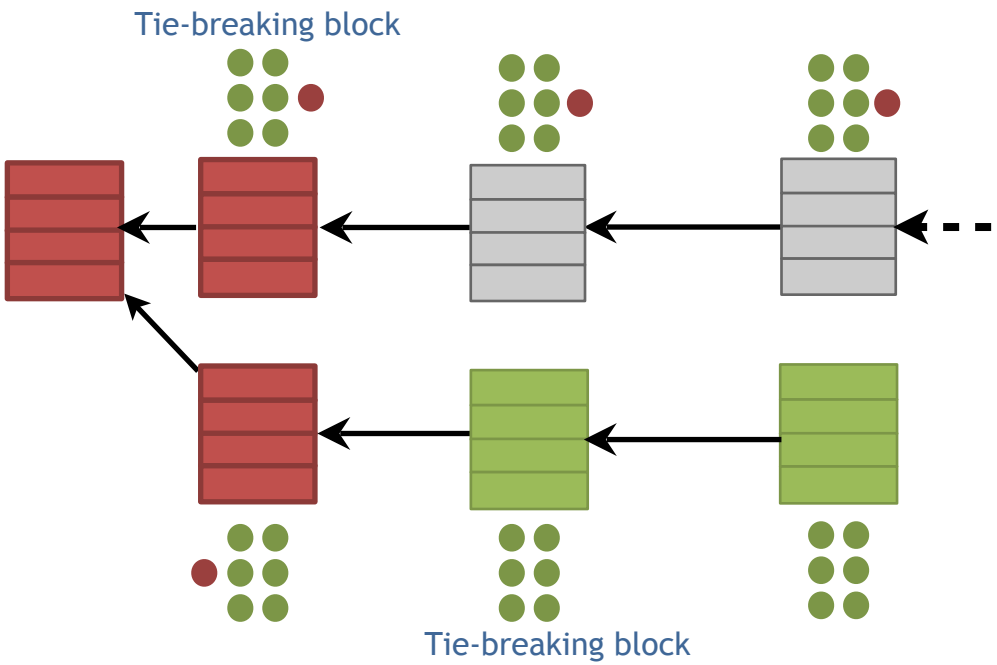

March 2, 2022


**Abstract**

We present two attacks targeting the Proof-of-Stake (PoS) Ethereum consensus protocol. The first attack suggests a fundamental conceptual incompatibility between PoS and the Greedy Heaviest-Observed Sub-Tree (GHOST) fork choice paradigm employed by PoS Ethereum. In a nutshell, PoS allows an adversary with a vanishing amount of stake to produce an unlimited number of equivocating blocks. While most equivocating blocks will be orphaned, such orphaned 'uncle blocks' still influence fork choice under the GHOST paradigm, bestowing upon the adversary devastating control over the canonical chain. While the Latest Message Driven (LMD) aspect of current PoS Ethereum prevents a straightforward application of this attack, our second attack shows how LMD specifically can be exploited to obtain a new variant of the balancing attack that overcomes a recent protocol addition that was intended to mitigate balancing-type attacks. Thus, in its current form, PoS Ethereum without and with LMD is vulnerable to our first and second attack, respectively.


l against Proof-of-Stake
nizations of the under-
idual validators' profits
lversarial network delay
:ely. We provide refined
:he requirements on ad-
dering the attacks more
d attacks, we obtain a

# Balancing Attacks




Sway vote

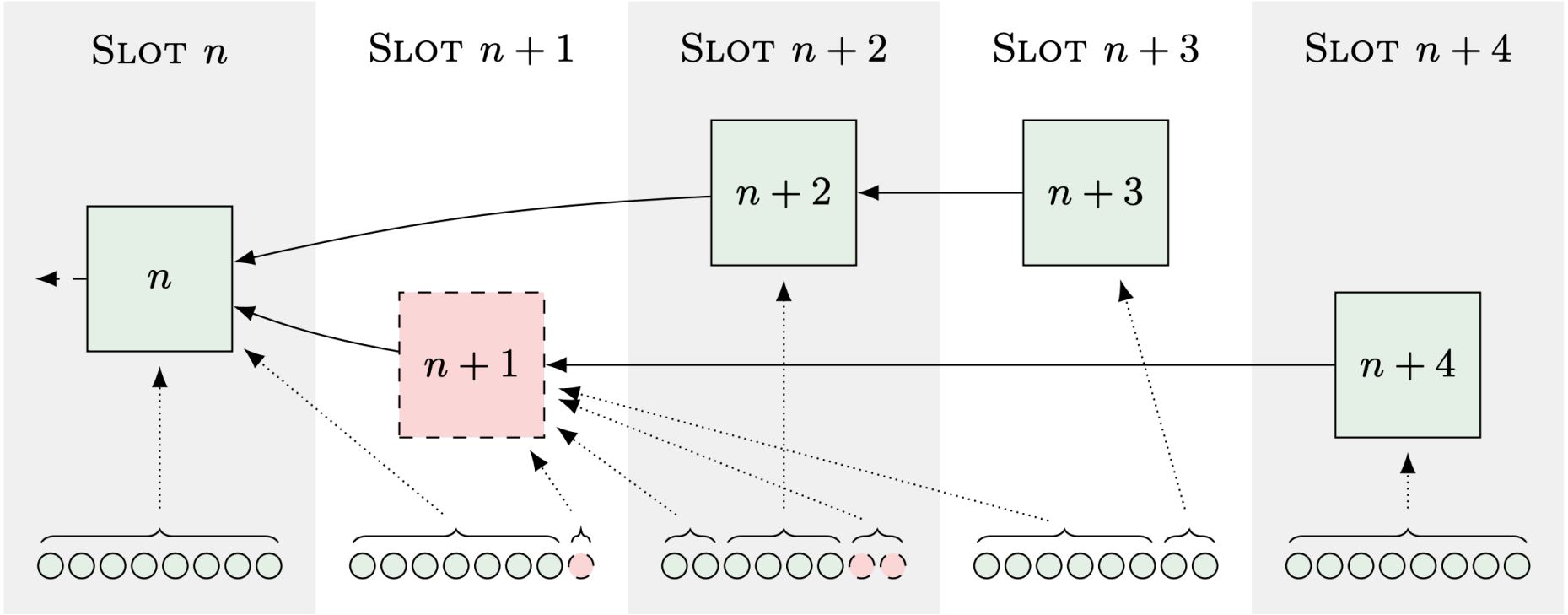
Practically: release the sway votes at around 100ms

- Adversary controls two consecutive blocks
- We asume a synchronous network model
- Adversary controls:
  - either the network (theoretical model) and 6 nodes in total, or
  - Assume a realistic network model and sqrt(n) nodes per slot

- LMD GHOST breaks ties based on the hash (randomly)

# Balancing attack effects

- Balancing attacks affect liveness and enable ex-ante reorg attacks
- Ex-post reorg attacks are very hard due to parallel attestation (unlike PoW)



# Balancing attack effects

- Balancing attacks affect liveness and enable ex-ante reorg attacks
- Ex-post reorg attacks are very hard due to parallel attestation (unlike PoW)

- Reorg attacks affect chain quality
  - Max latency: uncertain confirmation
  - Min capacity: reduced tx

- (Ex-ante) Reorg attacks become more important with the existence of MEV
  - They allow you to “steal” other people’s MEV
  - When there is a lot of MEV in a block, rational validators are incentivised to reorg it and deviate from the honest protocol execution

# Ethereum revisited

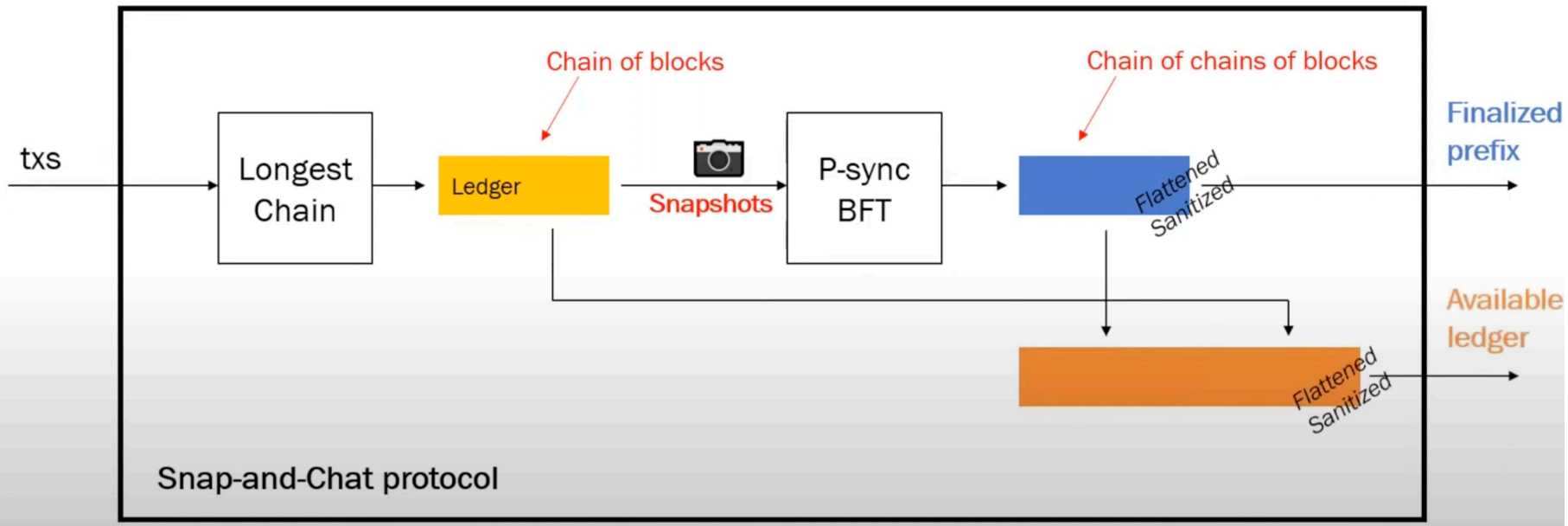
- LMD GHOST is not secure, so Ethereum is not truly an ebb-and-flow protocol!
- So how can we design an ebb-and-flow protocol after all?

# Snap and Chat Protocols

# Snap-and-Chat Protocols

Two protocols are executed in parallel

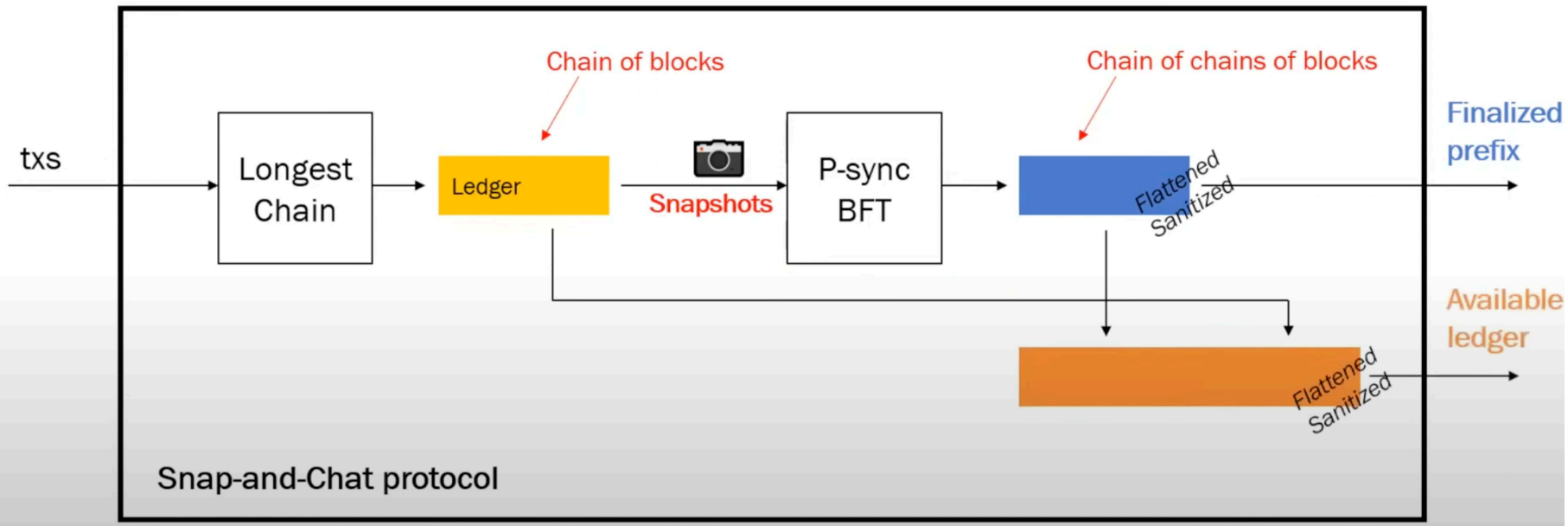
- $\Pi_{lc}$ takes as input transactions and outputs an ever-increasing **available** ledger $LOG_{lc}$ of ordered transactions (e.g., k-deep confirmation rule)
  - Nodes occasionally take *snapshots* (whole prefixes) of their local $LOG_{lc}$
- $\Pi_{bft}$ takes as input the snapshots of $LOG_{lc}$ and outputs an ordered list of these snapshots, the **final** ledger $LOG_{bft}$



# Snap-and-Chat Protocols

How are the two protocols combined?

- The finalized ledger $LOG_{fin}$ is produced by $LOG_{bft}$ after it is a) flattened - all snapshots are concatenated, and b) sanitized - keep only the first appearance of each transaction
- The available ledger $LOG_{da}$ is produced by appending $LOG_{lc}$ to $LOG_{fin}$ and sanitizing
  - During the BFT voting process, nodes do not vote for a snapshot that conflict the longest chain prefix (modified BFT)
  - The append-and-sanitize operation ensures that the transactions in $LOG_{fin}$ take precedence over the transactions in $LOG_{lc}$ in case of conflict

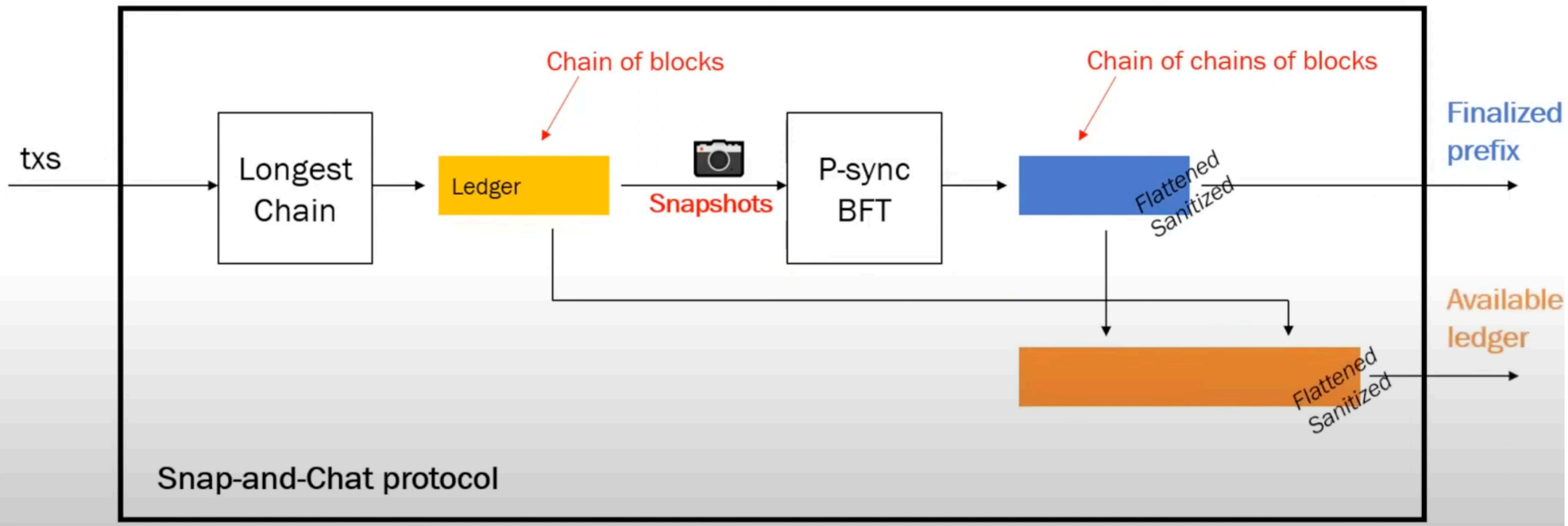


# Snap-and-Chat Protocols

- Snap-and-Chat protocols are $(1/3, 1/2)$-secure ebb-and-flow

  Optimal resilience - best possible guarantees for both ledgers:

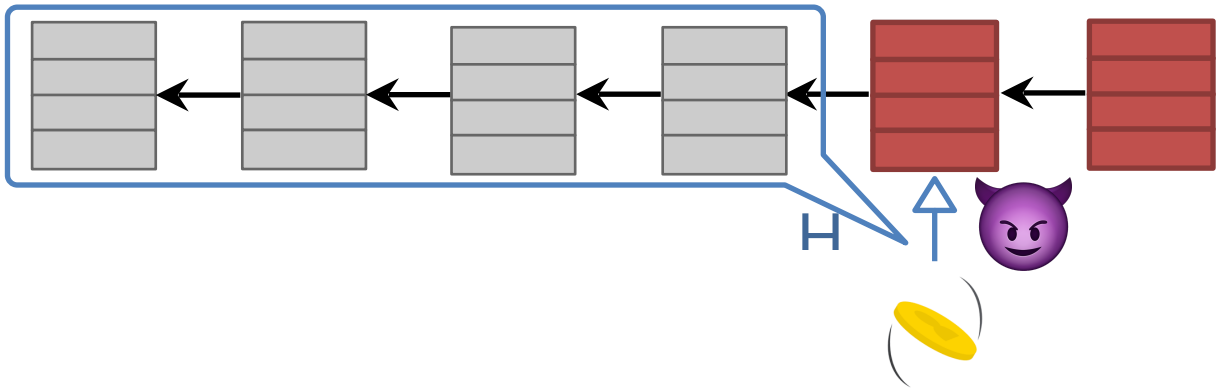
  - If $GAT = 0$ (partially synchrony), then $LOG_{fin}$ has the optimal achievable resilience (1/3)
  - If $GST = 0$ and $GAT = \infty$ (synchrony with dynamic participation), then $LOG_{da}$ has the optimal achievable resilience (1/2)



# Challenges & Countermeasures

# Challenges

- Grinding: biasing the leader election by manipulating the randomness




- Defence: Good unbiasable leader election
  - Remove nonce, txs, etc.
  - Use crypto to ensure unbiasable epoch randomness, or
  - Use hashing but bound the adversarial advantage

# Challenges


- Prediction: bribe or DoS next leader
  - Recall Bitcoin is unpredictable
  - Public leader schedule known upfront to all parties (Ouroboros, Ethereum 2.0)

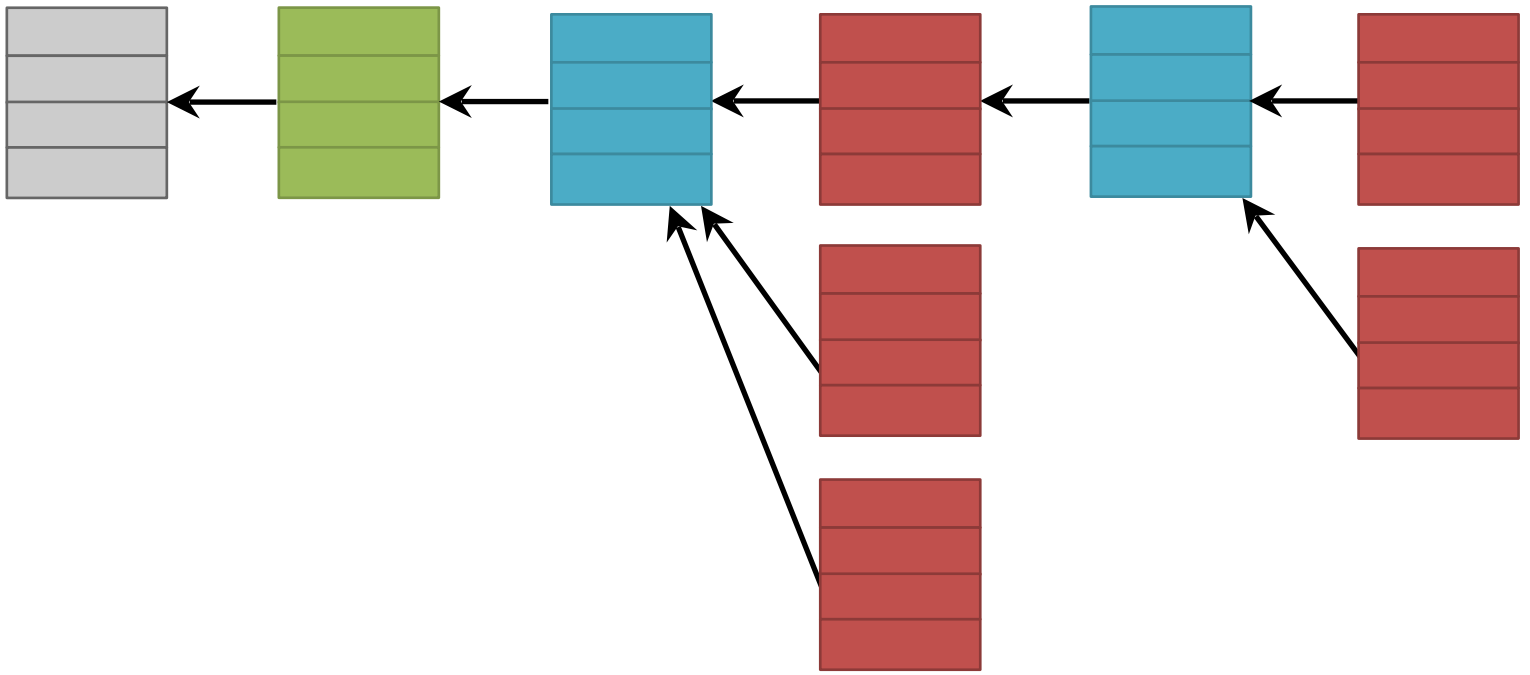
- Defence: Use crypto!
  - VRFs (Algorand, Praos) for local private verifiable election
  - Alternatively, VDFs - delay functions that provide additional benefits

# Challenges


- Costless simulation: no resources are wasted to produce a block!

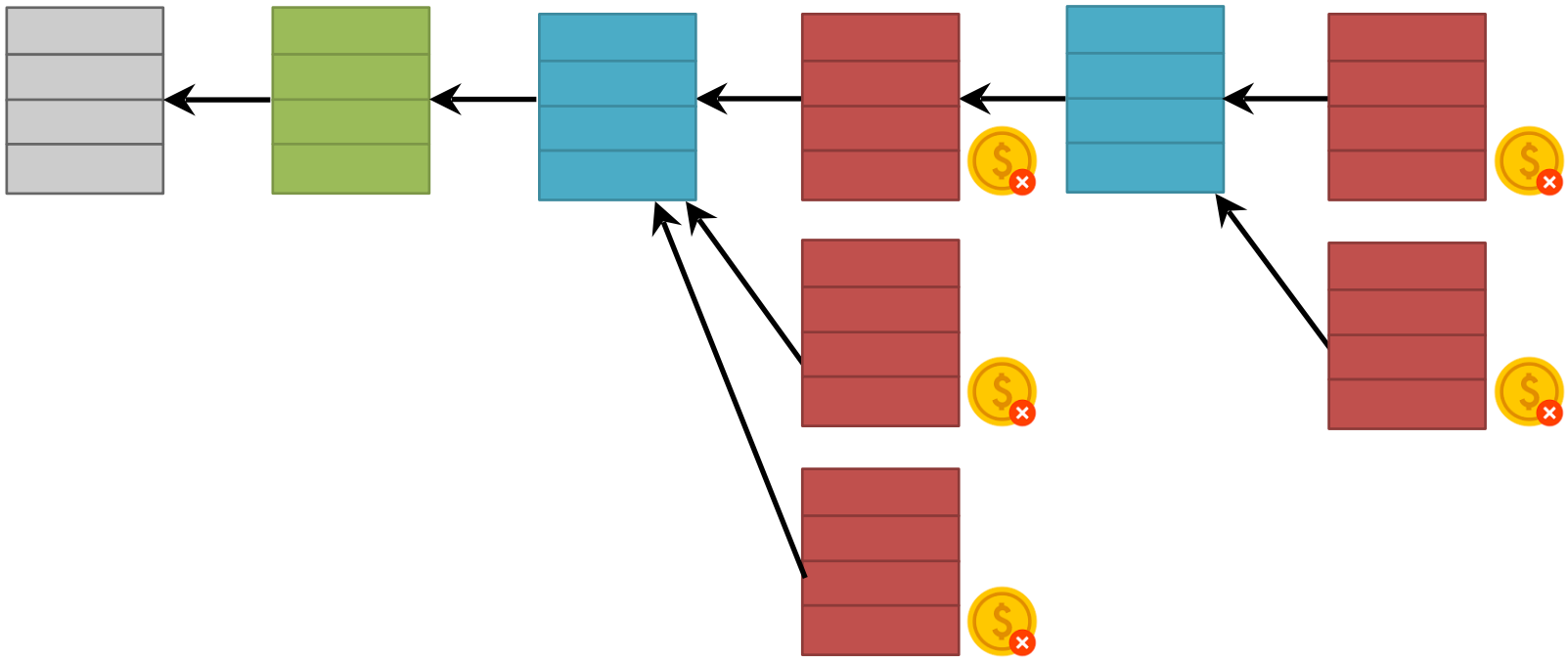
  Several attacks arise…
  - Nothing at stake
  - Posterior corruption (hacking or bribing)
  - Stake bleeding
  - Weak subjectivity

# Nothing at stake



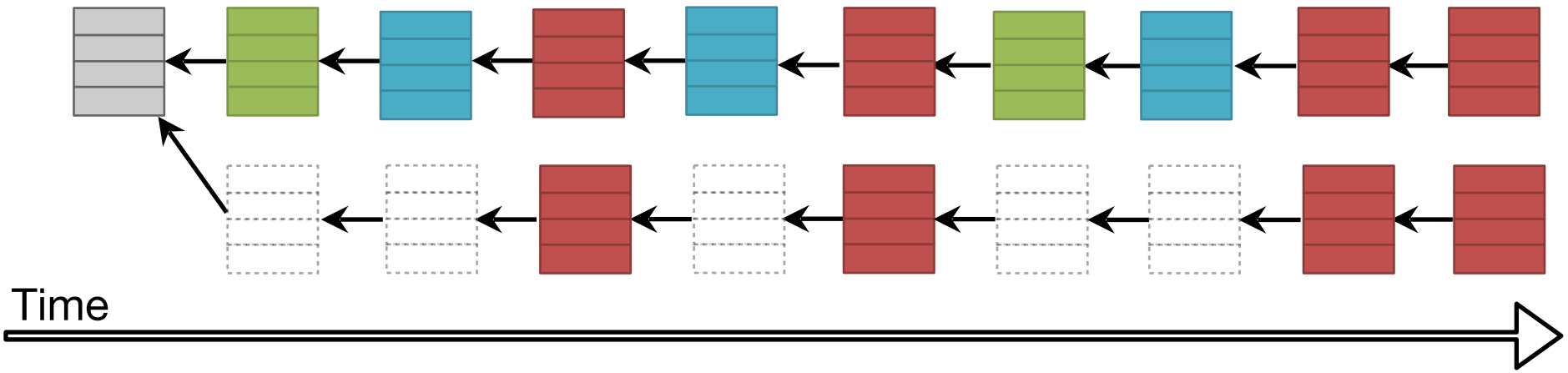
- **Attack:** Sign multiple blocks without any cost
  - Alice will always sign all possible chains
  - Chain does not converge with rational players

# Nothing at stake: slashing



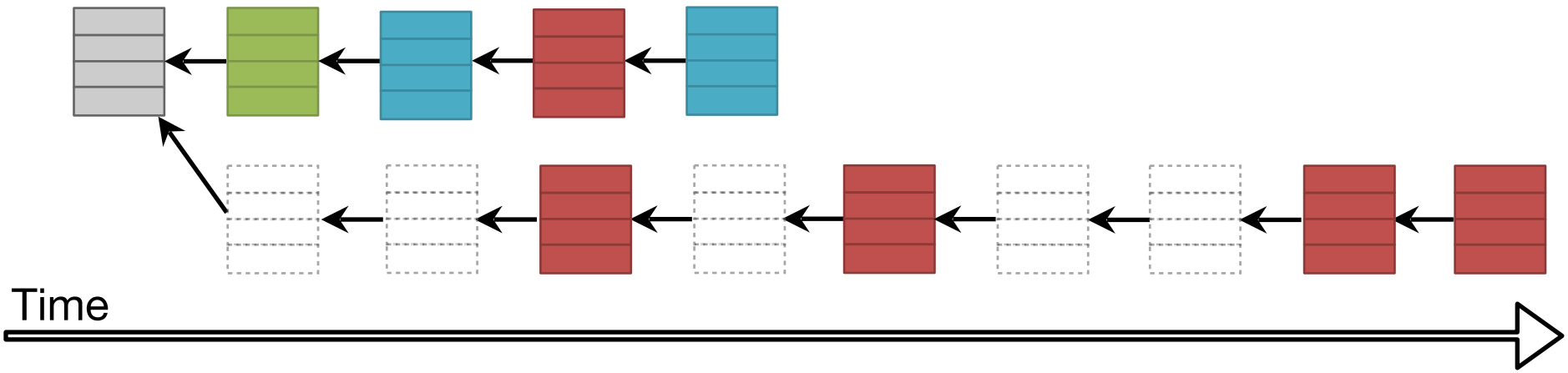
- **Attack:** Sign multiple blocks without any cost
  - Alice will always sign all possible chains
  - Chain does not converge with rational players
- **Defense:** Slash the stake of anyone that double signs!

# Posterior corruption



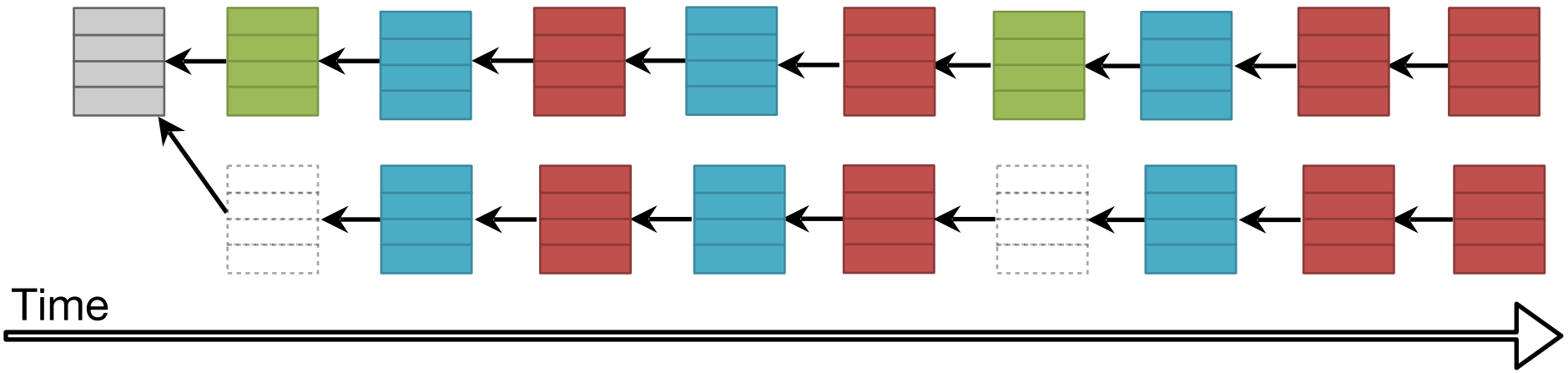

- **Attack:** Alice starts mining blocks in the past in an effort to create an equally long chain
  - Possible because signing blocks in the past is costless
  - Now Alice may control more keys and can use them to sign blocks in the past

# Posterior corruption - no timestamps



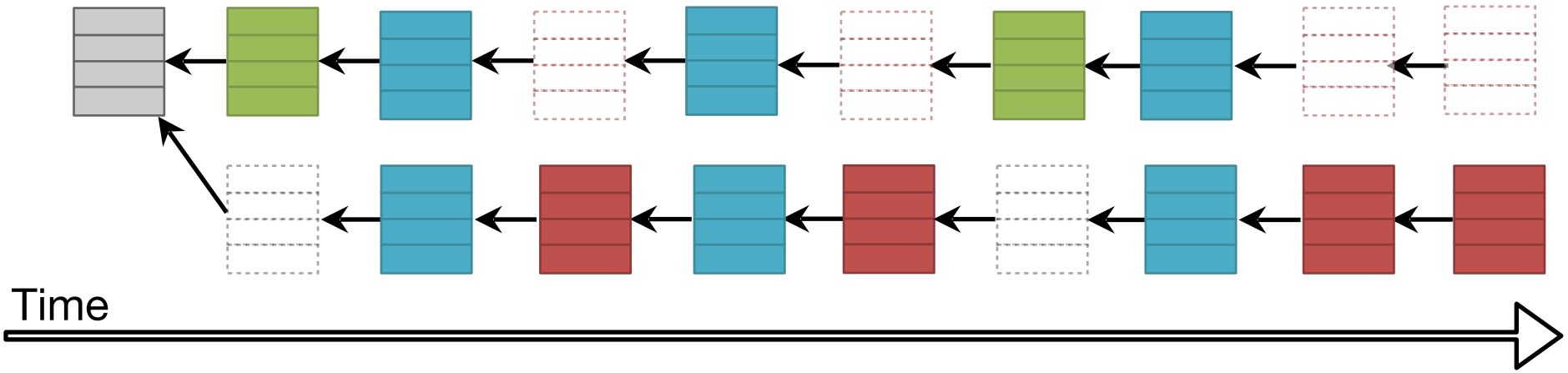

- Alice has the same chance to be elected in both branches
- Alice's branch has only one block generated within the correct time
- If the protocol doesn't check **timestamps**, Alice can create the future blocks as in her branch she controls everything

# Posterior corruption




- Bob sells his stake later in time but his past keys are still useful
- Alice *bribes* or *hacks* Bob's keys and generates blocks in the hidden branch

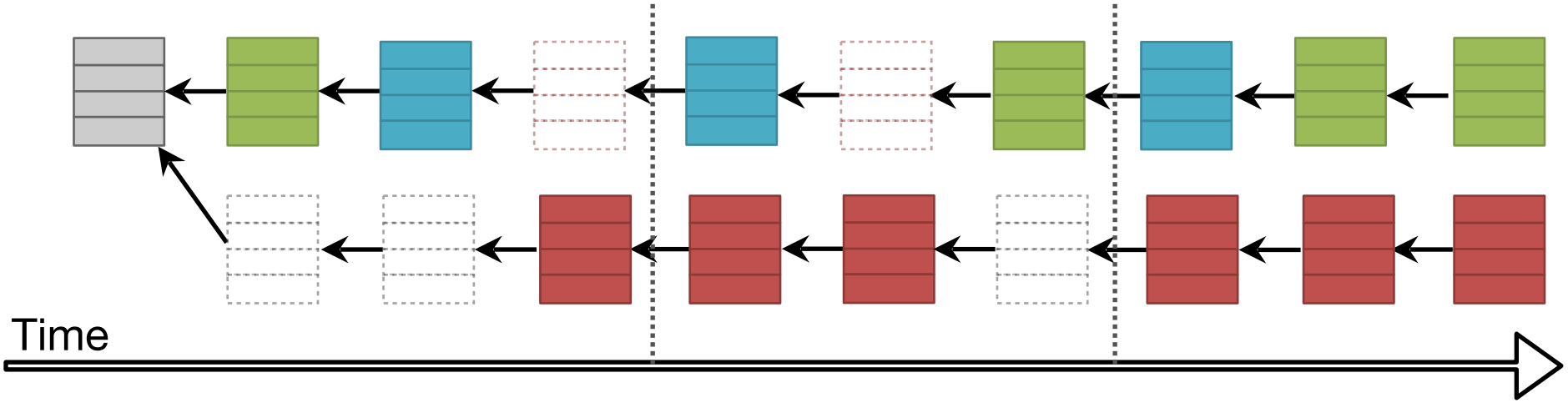
# Posterior corruption




- Bob sells his stake later in time but his past keys are still useful

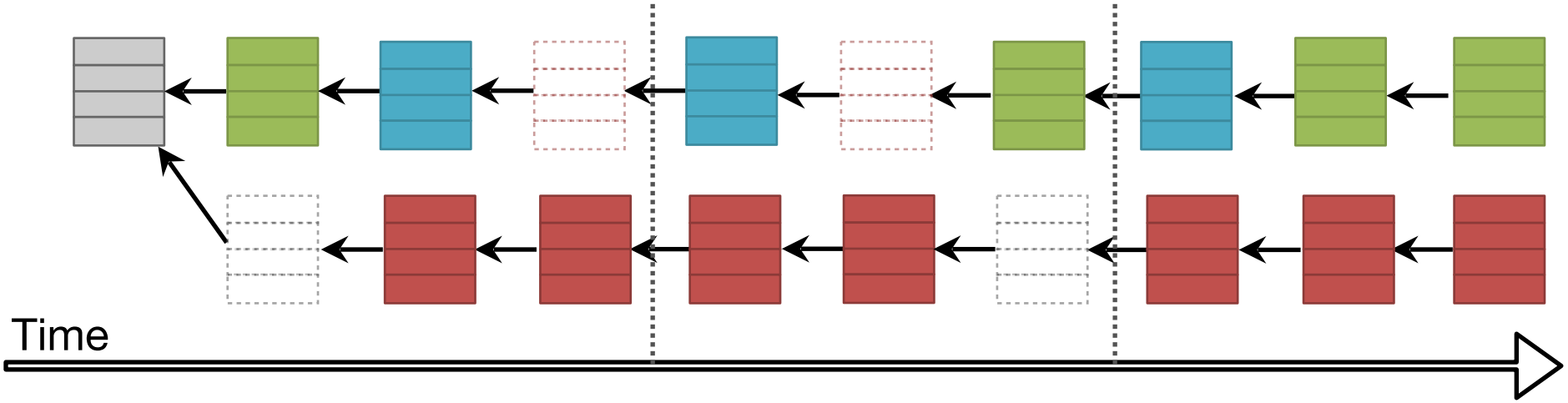
- Alice *bribes* or *hacks* Bob's keys and generates blocks in the hidden branch
- Alice stops producing blocks in the main chain

# Stake bleeding




- Alice does not produce blocks in her slots in the main branch to slow down the chain growth
  - Alice's stake decreases gradually in the main branch
- Alice's controlling stake in her branch *gradually increases* as she is the only one producing blocks
  - Hence, she can generate more and more blocks as time passes by
  - Alice can chose also the transactions with the highest tx fees to increase her stake even faster

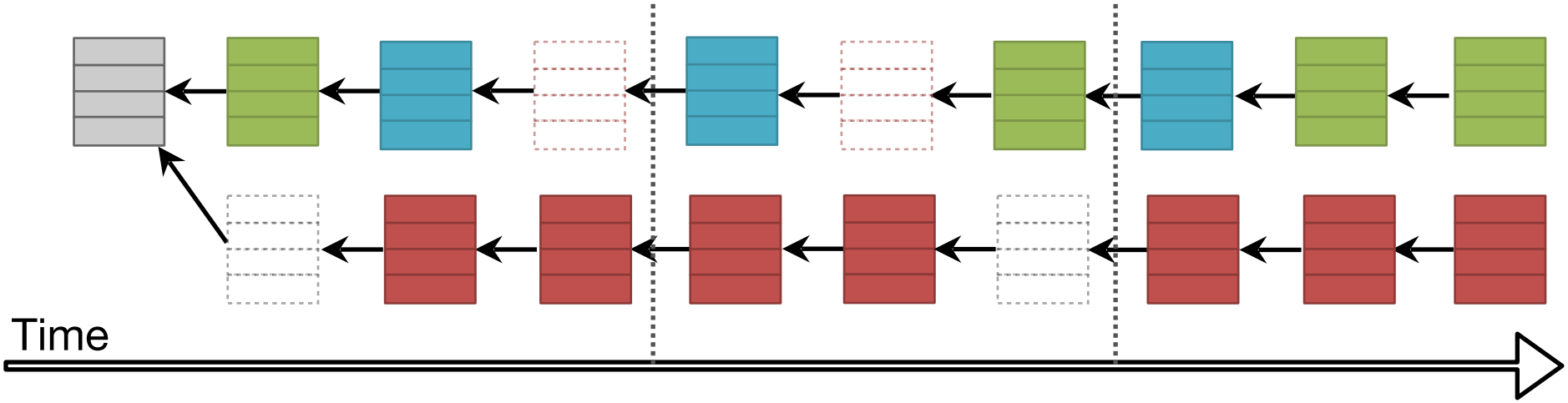
# Weak subjectivity




**Weak subjectivity:**

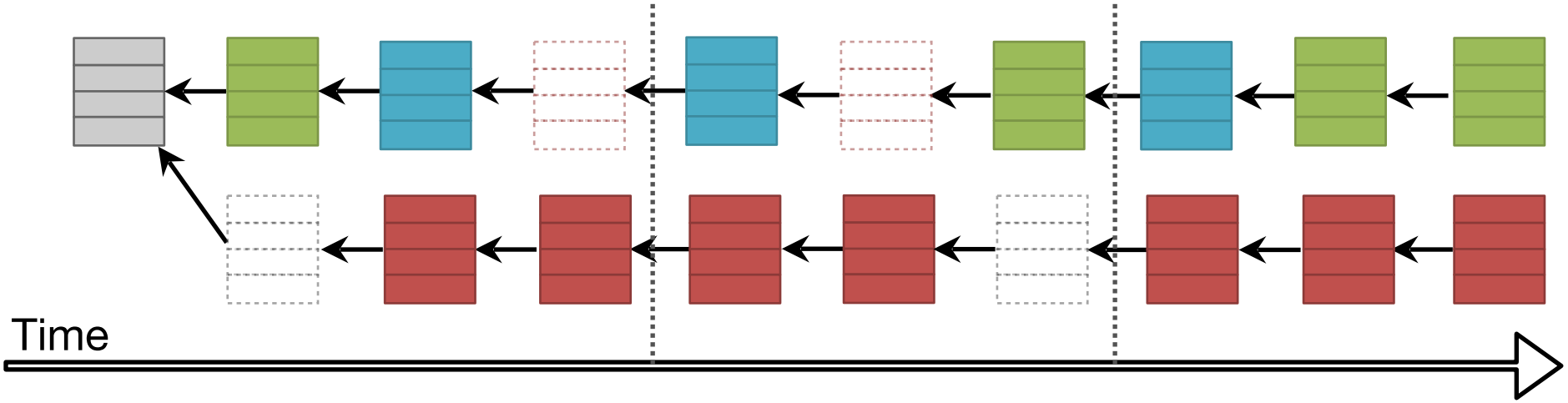
A new bootstrapping node cannot extract the "truth" given **only** a correct genesis block

# Weak subjectivity




- Two settings relevant for long-range attacks in PoS protocols:
  - Dynamic availability (DA) setting: dynamically evolving list of IDs, subset of them is active and honest (longest chain PoS protocols like Ouroboros family)
  - Quasi-permissionless (QP) setting: dynamically evolving list of IDs, all of them active at all times (BFT-based PoS protocols like Algorand)

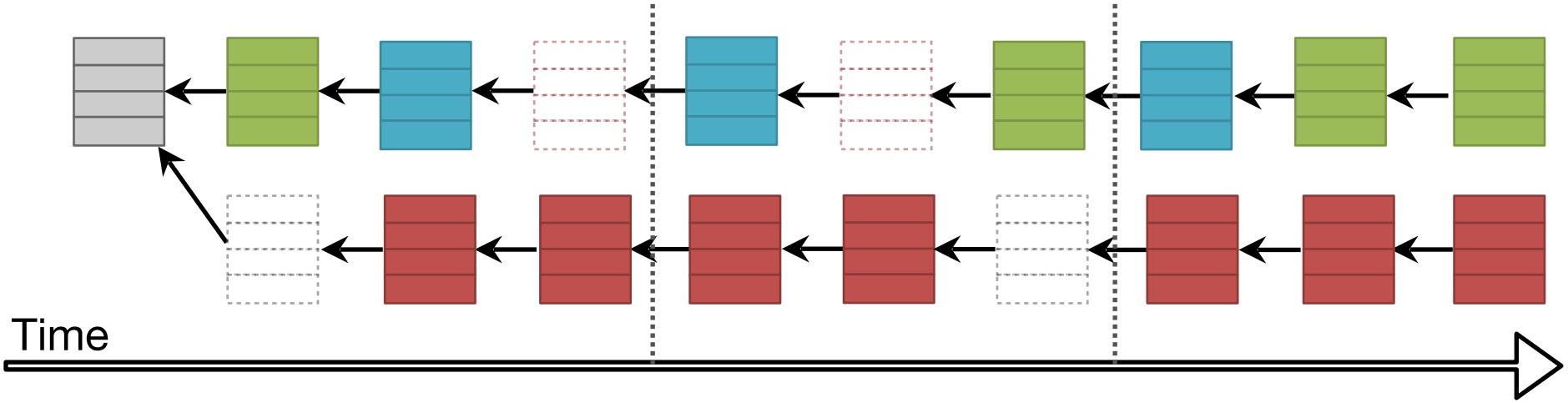
# Weak subjectivity




- Folklore results
  - In the DA setting: the adversary can use her now active stake to create past blocks (when the total active stake was very little, e.g., 10%)
  - In the QP setting: posterior corruption (e.g., bribe old keys) and stake bleeding is still possible. So a minority adversary could still make a chain of length equal to the honest chain (weak subjectivity)
- Formally proven (2023): No PoS protocol can be secure against long-range attacks* in the QP and synchronous setting, when using time-maleable crypto.

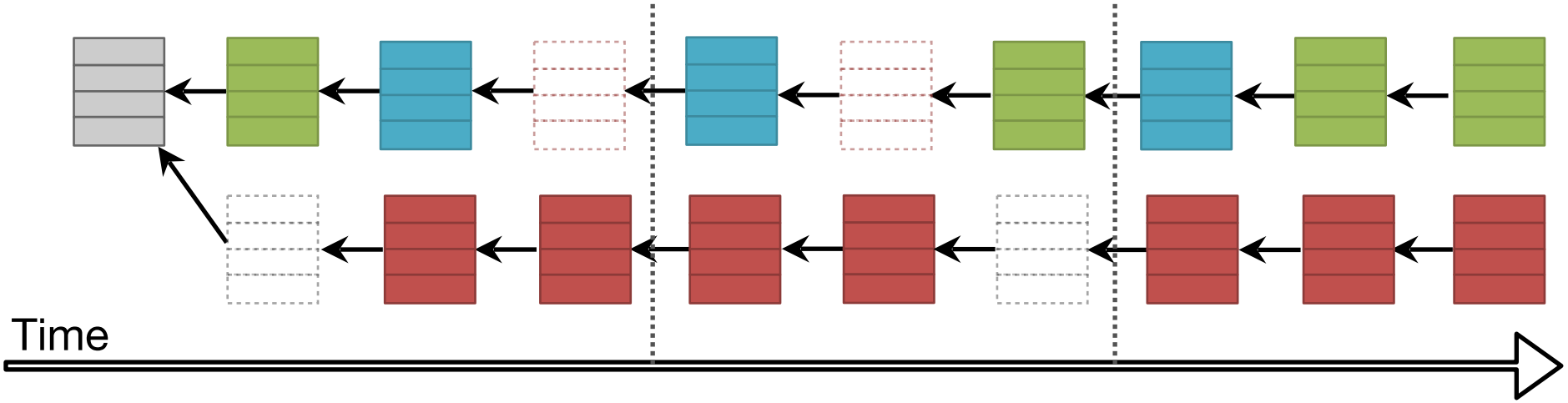
  *long-range attacks: assume the adversary can access old keys by hacking or bribing

# Weak subjectivity countermeasures




- Some suggestions…that really don't work! (& used in Ethereum 2.0)
  - Checkpoints: forbid reverting the chain before specific blocks (something like new "genesis" blocks)
    - On a website → centralization
    - On a PoW chain → emmm…wasn't the goal to get rid of PoW??
  - Finality gadgets: run BFT protocols on top of longest-chain
    - Provide deterministic finality…
    - …which is another problem!

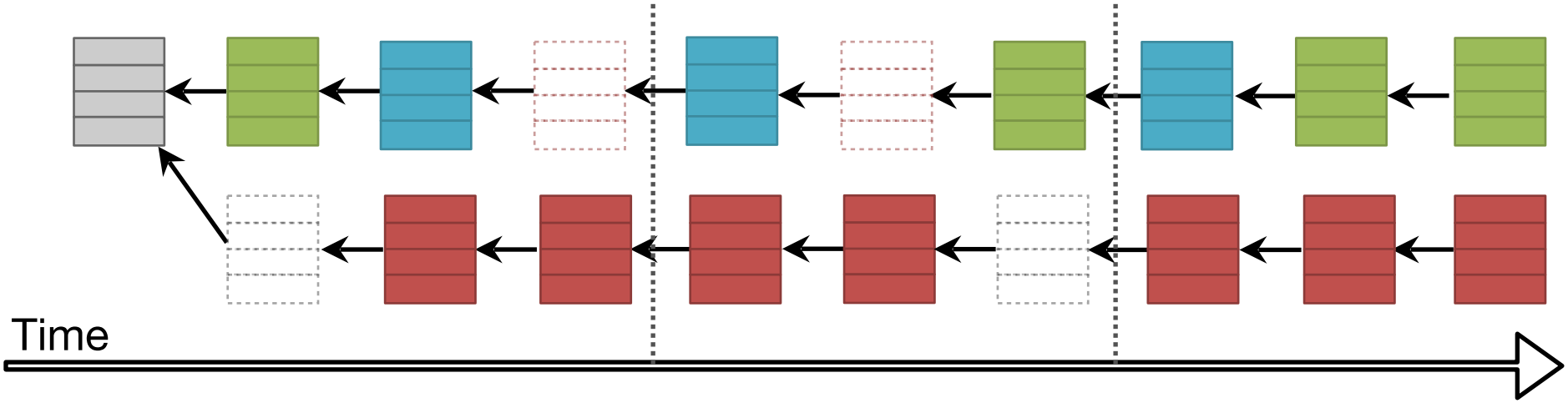
# Weak subjectivity countermeasures




- A suggestion we already saw

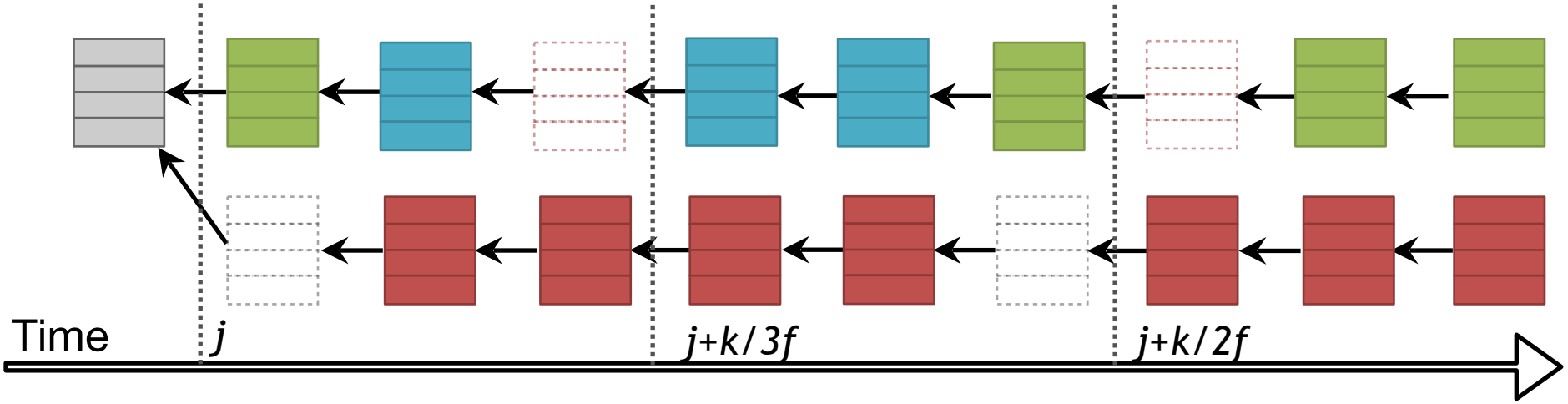
- Key-evolving cryptography
  - Secures the QP setting
  - Assumes honest parties that delete their past keys — its not a provable fact (without trusted hardware)
  - Does not secure DA setting

# Weak subjectivity countermeasures



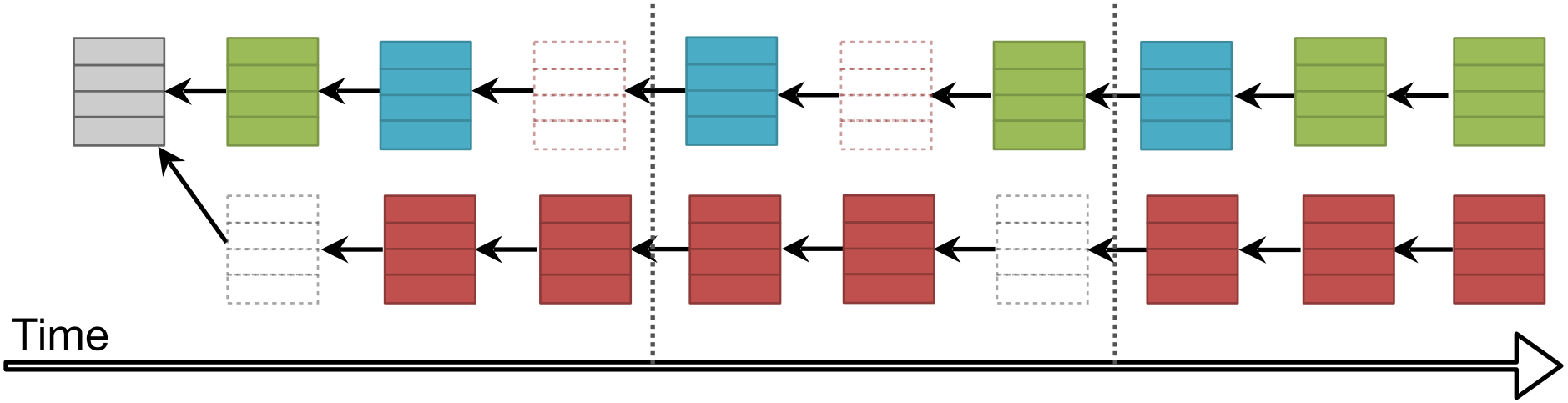

- Some really clever countermeasures
  - Context aware transactions: txs include a parent block
    → alleviates stake bleeding
  - Wrinkle: each coin votes to finalize an epoch/block
    → alleviates but does not eliminate the problem, really impractical

# Weak subjectivity countermeasures




- Some really clever countermeasures
  - Ouroboros Genesis or Plenitude Rule:
    For short forks (less than *k*) choose the longest chain, while for long forks choose the chain that is the densest after the divergence point!
    → only if honest majority of keys not compromised

# Weak subjectivity countermeasures




- How to secure PoS protocol in the DA setting:
  - Use VDFs!!!
    - Insert occasionally in the VDF data from the longest chain PoS leader election to make it resilient against posterior corruption
    - Unrealistic assumption that all parties operate the same hardware — all VDFs run approximately at the same speed

# Overview

- We can have either a dynamic available or a final ledger
- We can, however, build a protocol that returns both of them
- Finality gadgets do *not* provide resilience to long-range attacks
- PoS is *not* as secure as PoW due to weak subjectivity

# Proof of Stake (III)

## Finality Gadgets and PoS Challenges

Questions?

December 10th, 2025

TU WIEN

S&P Security & Privacy