

A Plug-and-Play Long-Range Defense System for Proof-of-Stake Blockchains

Lucien K. L. Ng, Panagiotis Chatzigiannis, Duc V. Le,
Mohsen Minaei, Ranjit Kumaresan, Mahdi Zamani

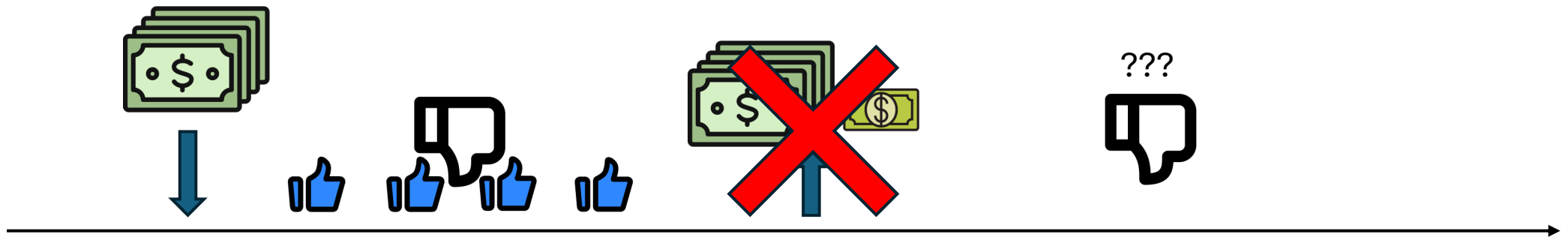


From PoW to PoS

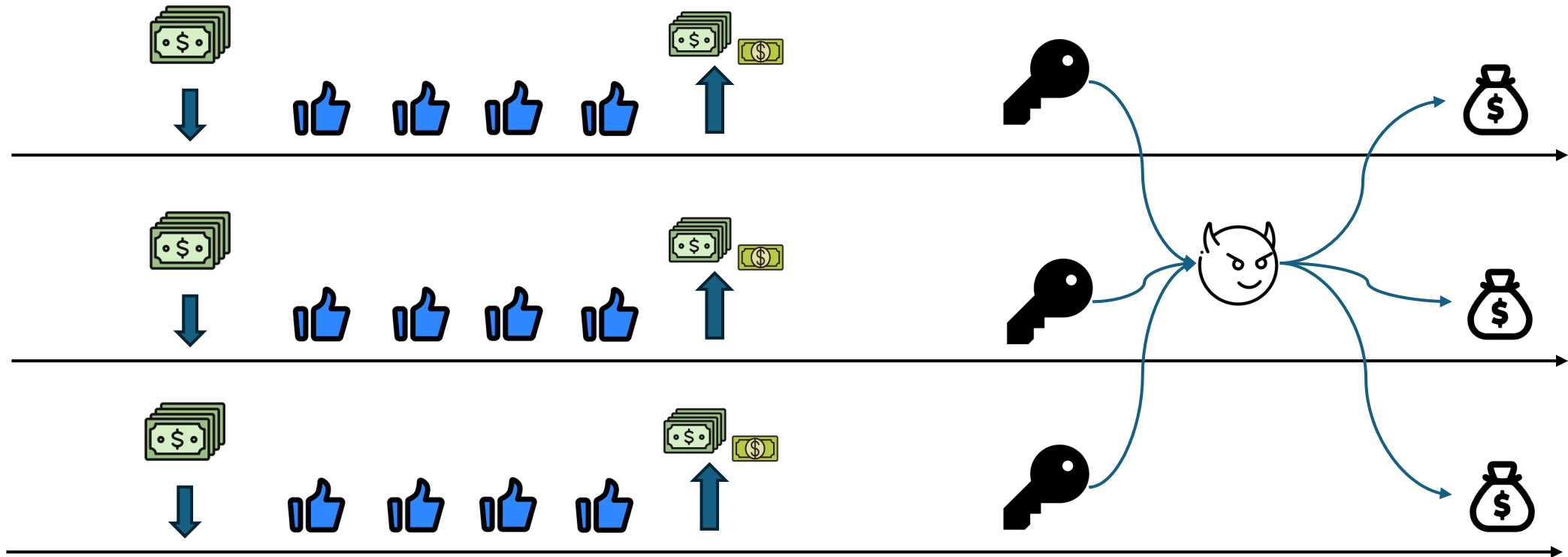
- Proof-of-Work (PoW) consensus is highly energy-inefficient
 - Validators (Miners) use ton of electricity just for reaching consensus
- Proof-of-Stake (PoS) is more energy-efficient
 - Leaders are selected based on their staked wealth on-chain

How (Penalty-Based) PoS works?

- Validators stake their coins on the blockchain
- If they comply with the protocol, they will earn reward
 - And they can withdraw their stake and reward after a lock-up period
- If they misbehave, their stake will be forfeited
- What if they misbehave after they have withdrawn their coins?

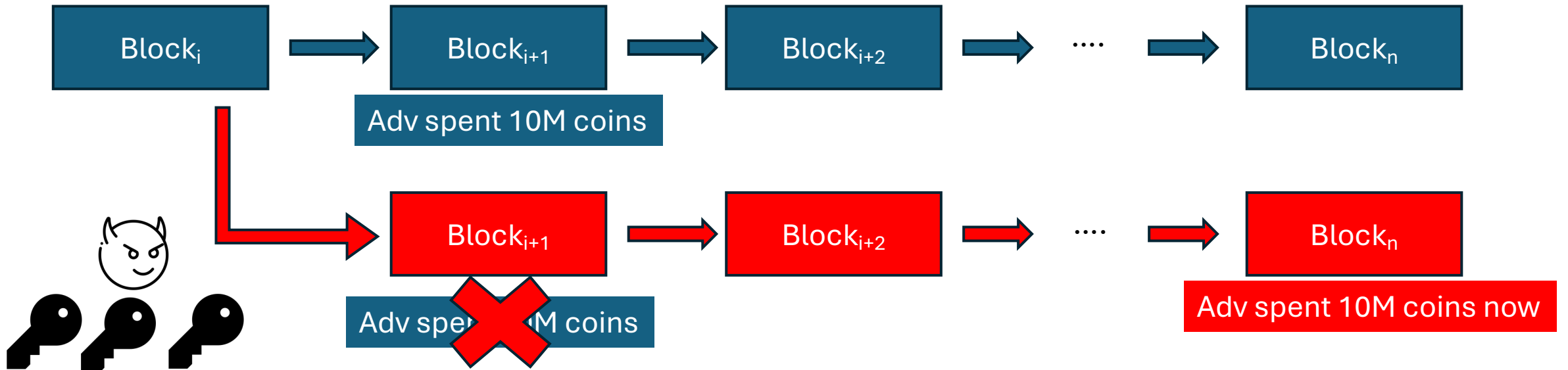


Posterior Key Corruption



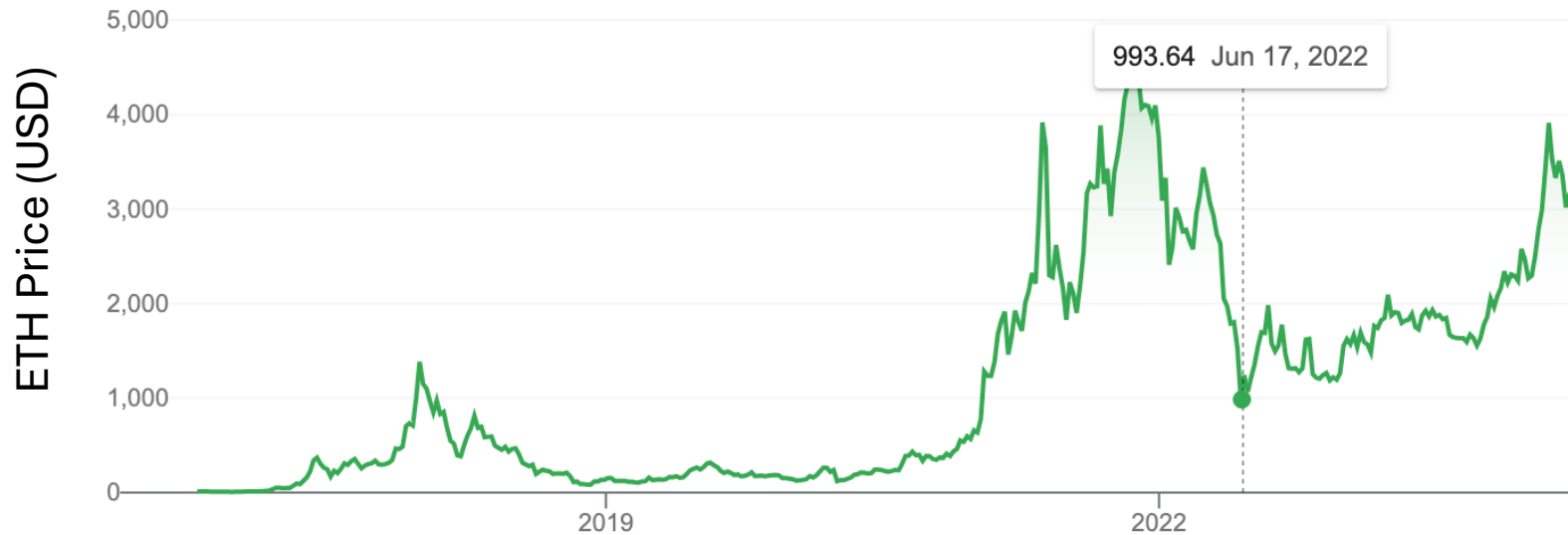
Long-Range Attack

- Once an adversary has gathered enough old validation keys
 - it can fork another valid chain and double-spend!



Possibility of Long-Range Attack

- Coin values can fluctuate a lot
- Selling old keys become more profitable (than protecting the assets)
- Attackers launch long-range attacks when coin value bounces back



Unsatisfying Solution: Checkpointing

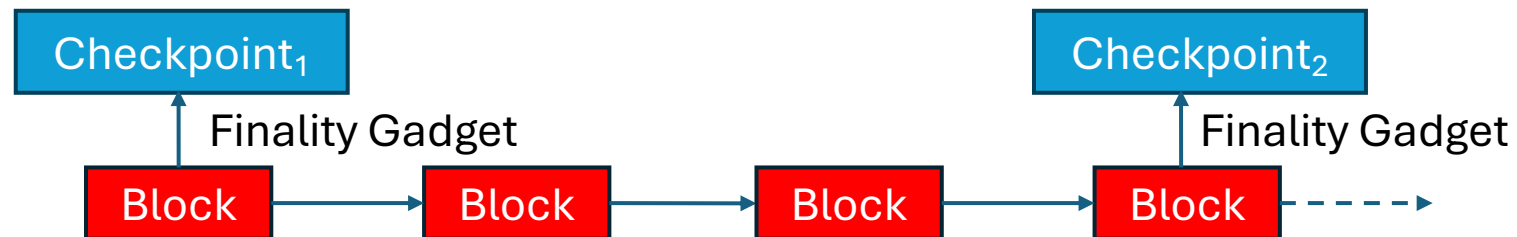
- Digests of old blocks are hardcoded in the node's software
 - Centralization Issue: the software developers can launch attacks
- A few centralized servers broadcast the digests
- When there are conflicts, who should the client trust?

The screenshot shows the Lodestar interface for an Ethereum beacon chain checkpoint sync provider. At the top, it displays 'Lodestar' and 'Network: Mainnet, Operation Mode: Full'. Below this is the title 'An Ethereum beacon chain checkpoint sync provider'. The interface features two main sections: 'Latest Finalized' (Epoch: 278705, Block Root: 0xdeb58...af0df9) and 'Latest Justified' (Epoch: 278706, Block Root: 0x0f1fa3...abee5c). A 'Get started!' button is positioned between these two sections. At the bottom, there is a 'Checkpoints' tab and a table of historical finalized epoch boundaries. The table includes columns for Epoch, Slot, Time, State Root, and Block Root. The current checkpoint being served is marked with a flag icon.

Epoch	Slot	Time	State Root	Block Root
278705	8918560	15 minutes ago	0x63a42ca719c1140075ea662ebce35014780b5ba953ff4afdc22bc30b923b8589	0xdeb589dfaf6775a62bb025a146492a155ebc3095898d12e351af42d81af0df9
278704	8918528	22 minutes ago	0xe9cb5a7d4f67f14ffa6ee580f85043ef4eed4c43ffae518bd1e6babeaf335bd	0xf1d2cd78e333ebb35854e9a6898471bb5f29de23c0e9e01676d4d2adf3f43e0df
278703	8918496	28 minutes ago	0xb93823c81f94b99dafc1b47dc4cf2a54c4a1b964f46026d46791331fc9724f8e	0x8f4fe1a18583aa82501f1cc02ab7ea05cbd898efc772ae87d260901fab2fd0ec
278702	8918464	35 minutes ago	0xd61b6cac618434eca5c1ba8d28755c54887d2c01e7d4d935513f569e54ac7f24	0x4f27f7199fabbc29955a979e1cae611e310e882e1b866102559cf1ab18e7c1ae
278701	8918432	41 minutes ago	0x183c95c2446c328bb17b4df3d5be637ac3eb577b553531a05261672437952f2e	0x2b088db01096ffb6ada7f6dc9d9b5b17ccfbb53317295f9b28d9100175112e0

Unsatisfying Solution: Finality Gadgets

- Ethereum requires 2/3 of the validators to sign on the checkpoints
 - (Otherwise, the transactions in the checkpoint are not finalized)
 - It assumes $<1/3$ validators are malicious
- It is just asking the adversary to acquire more old keys



Unsatisfying Solution: Always-Online Nodes

- Servers monitoring the chain know which fork is authentic
 - The checkpoints produced first are the genuine ones
 - (More about it later)
- But how about clients?

Clients

- New clients have no knowledge about blockchain's history
- Existing Clients might be offline for a long period
- They may see two equally valid checkpoints when logging on

- The client are also “light”
 - with limited computation and communication capability

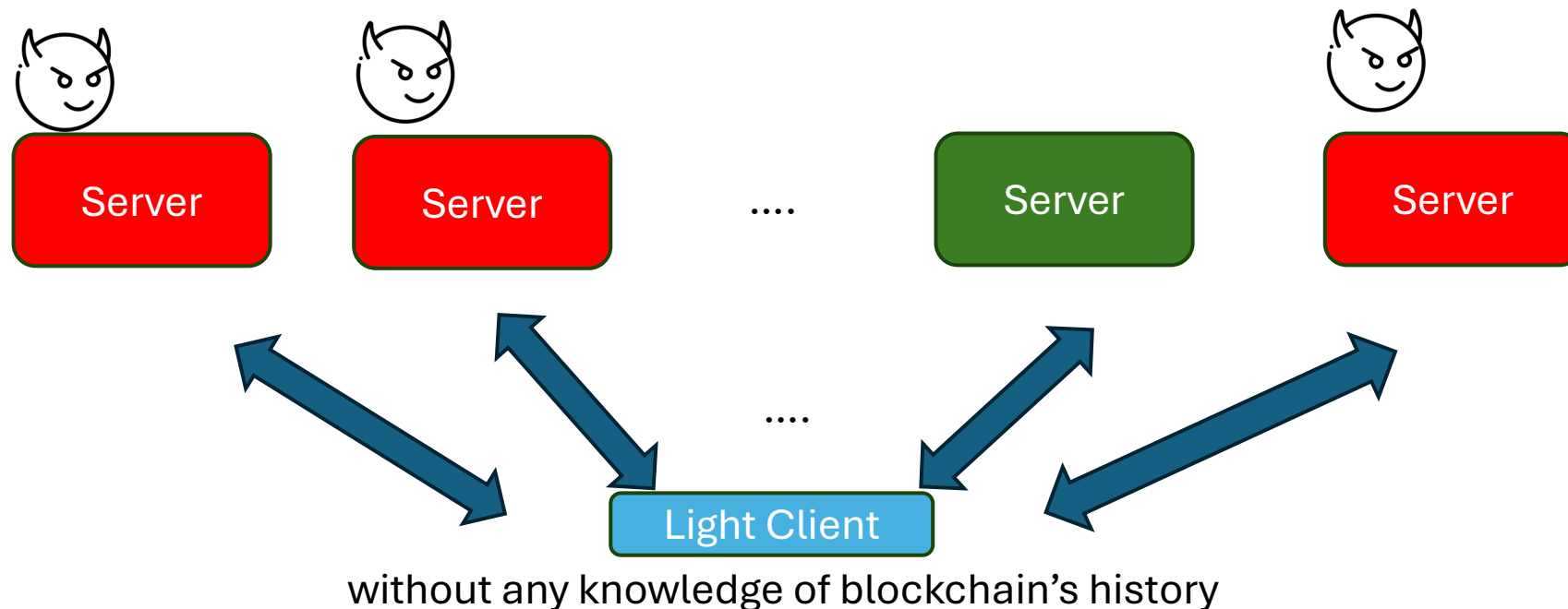
- Can the servers help them?
 - Wait... the servers can be malicious

Our Solution

- A defense system against long-range attacks
 - It helps light clients to distinguish which fork is authentic
- Advantages:
 - Plug-and-play: No soft nor hard-forks needed
 - Reasonable Assumptions: our defense works as long as one server is honest
 - Light-client friendly: Clients only need to verify succinct proofs

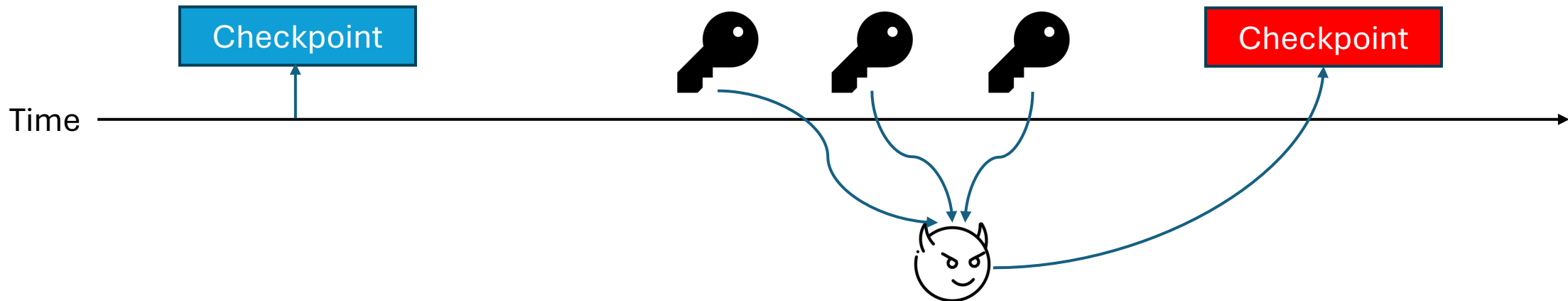
System Setting and Threat Model

- For simplicity, we assume there are only two servers
 - One is honest, and the other is malicious
- The attacker can only corrupt keys of past (but not current) validators



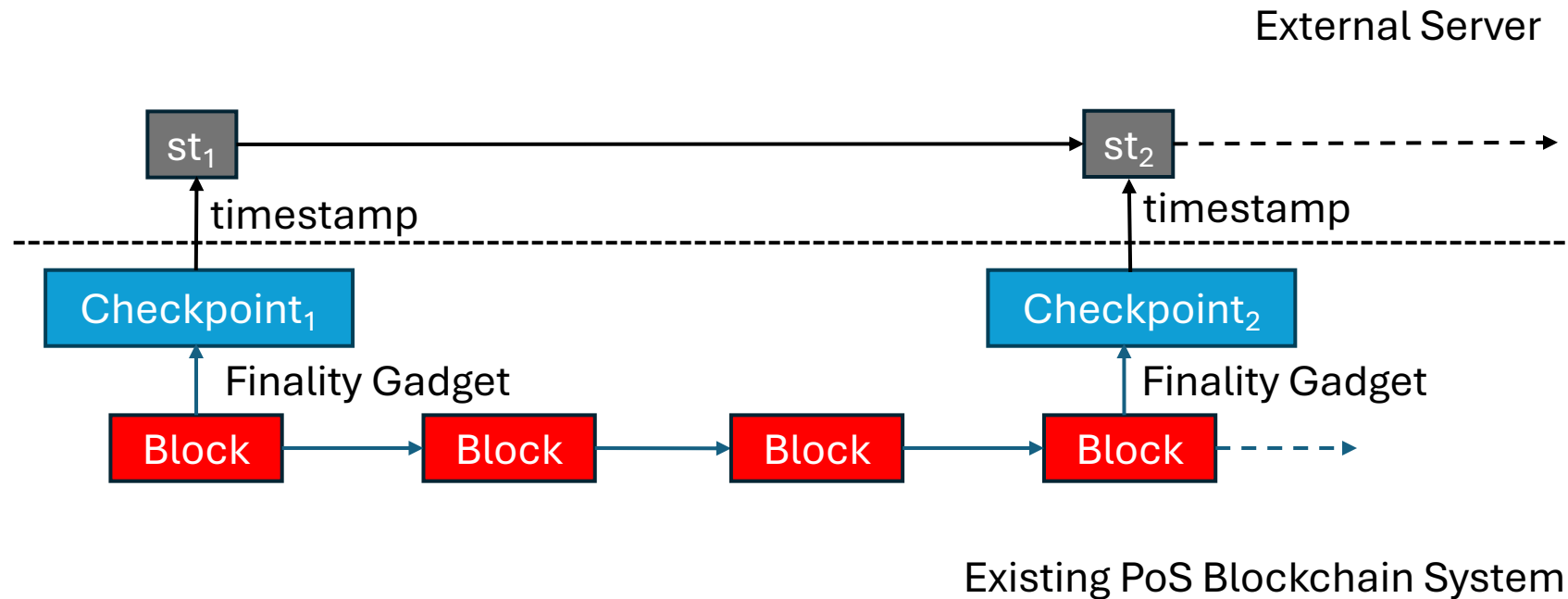
Timestamp

- The checkpoints produced first are the genuine ones
- The servers timestamp the checkpoints and their finality proofs
- How to timestamp? Verifiable Delay Functions (VDF)

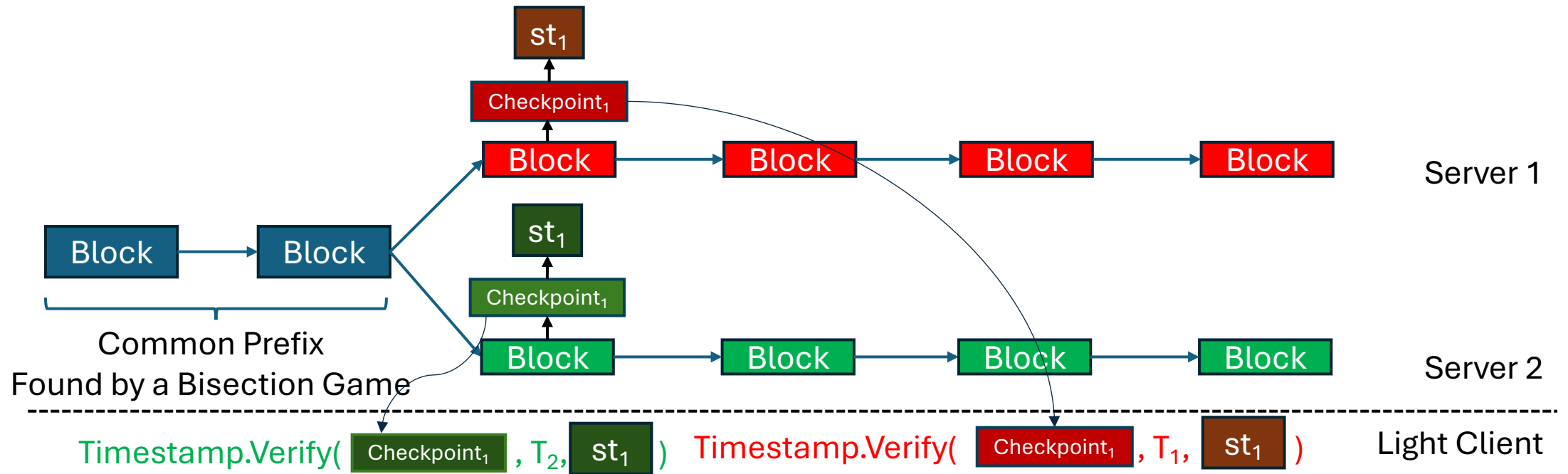


Server's Workflow

- No change to the blockchain's consensus protocol



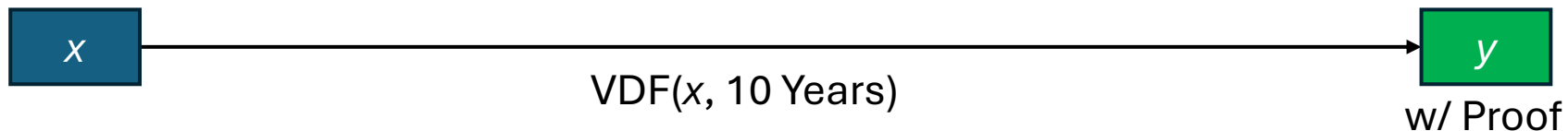
Client's Workflow



- Accept the block with an earlier timestamp

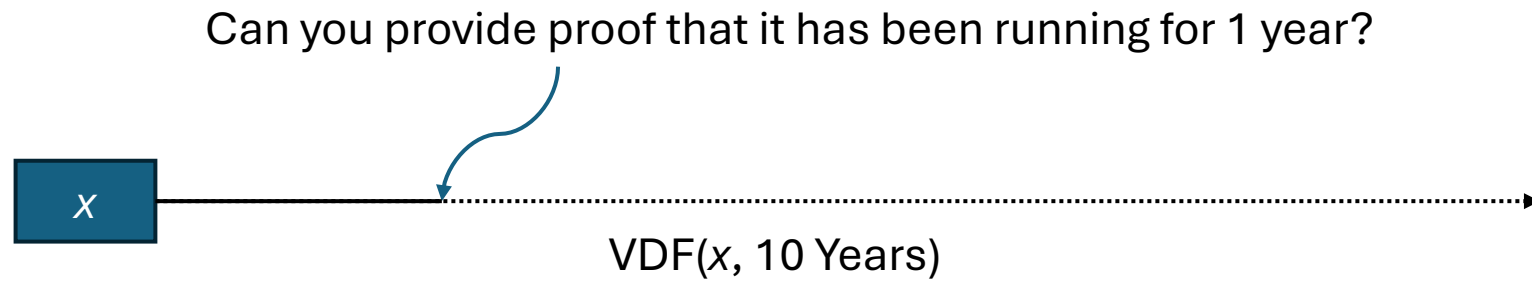
Background: Verifiable Delay Function (VDF)

- Informal Definition:
 - $VDF(x, t)$ can only be computed with t unit of time
 - It can be succinctly proven
- It usually is based on repeated squaring assumption
 - $x^{2^t} \bmod N$ is most efficiently computed by sequential squaring
 - The group order is unknown, e.g., N is an RSA modulus
 - Some schemes relies on other assumptions, e.g., lattice-based.



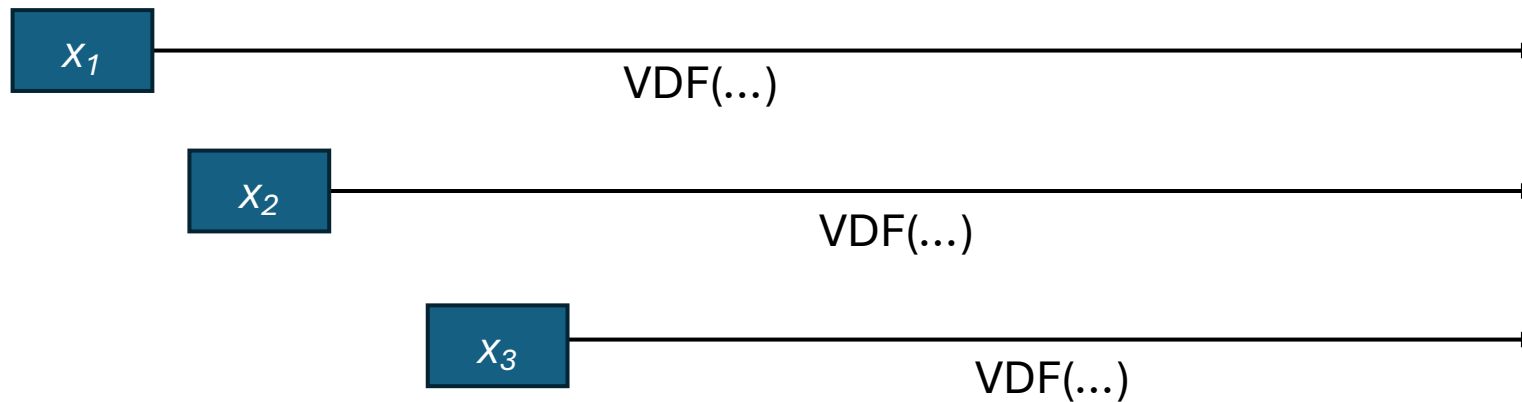
1st Issue about VDF

- VDF is not ever-going



2nd Issue about VDF

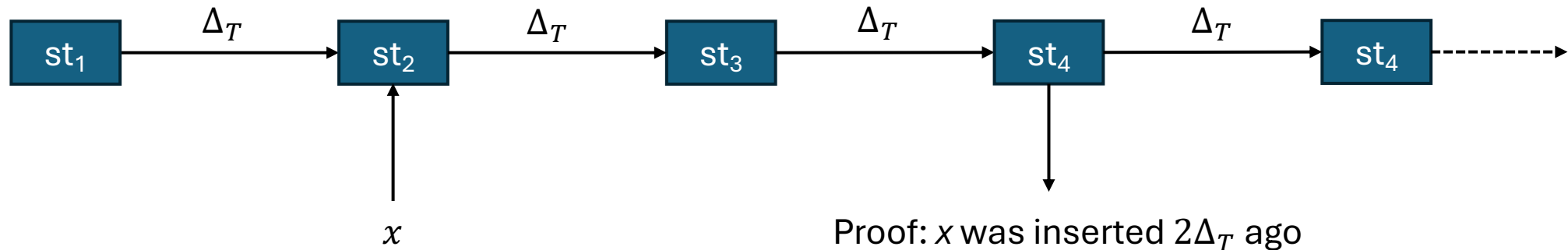
- The input x has to be committed in the beginning



- Ethereum has >250k checkpoints...

Insertable Proof-of-Sequential-Work (InPoSW)

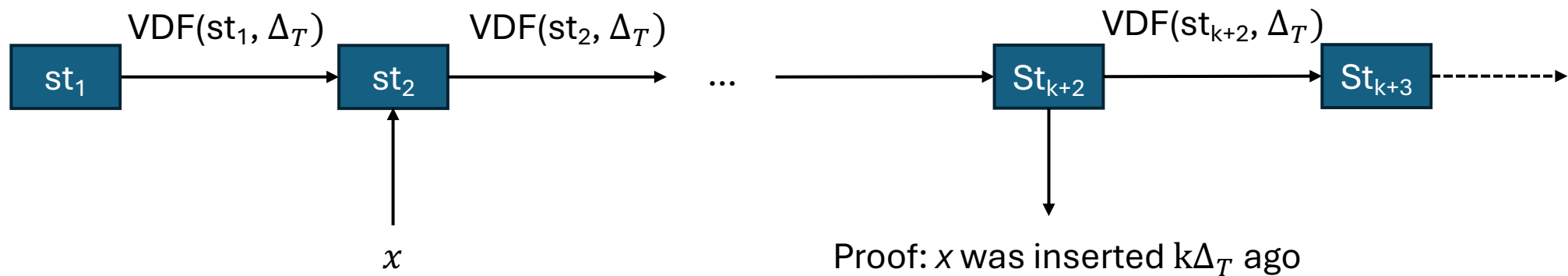
- At any time point (st_i), the prover can
 - Op 1: Insert data for timestamping
 - Op 2: Prove some data was insert $k \cdot \Delta_T$ ago



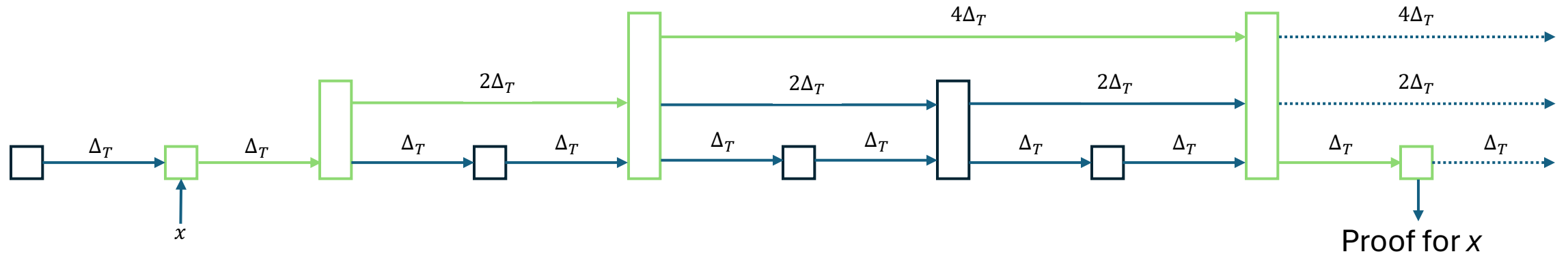
- Remark: Compared to PoW, PoSW cannot be parallelized

Strawman InPoSW Scheme

- 😭 Not succinct: The verifier needs to verify k VDF proofs



Our Skiplist-Style Construction for InPoSW



- Prover Storage: $O(N)$ VDF Proofs
- 😊 Verification Cost: $O(N) \rightarrow O(\log N)$ VDF Verification

Estimation of Concrete Cost

- We use Ethereum as our reference
- After 10 years of running our system
 - The server stores ≈ 546 GB of data
 - $>22x$ less than adopting existing solution
 - The proof size is ≈ 20 KB
 - $>17000x$ less than adopting existing solution
- Prior Solution that can be modified for InPoSW
 - An Incremental PoSW for General Weight Distributions [EC '23]
 - Graph-Labeling PoSW Scheme

- We set $\Delta_T \approx 3.6$ minutes
 - which translates to 2^{33} repeated squaring
- Ethereum emits a checkpoint every 6 minutes

Conclusion

- Long-range attack can bring devastating outcomes to PoS blockchains
 - And existing solutions are unsatisfying
- We propose a solution that
 - has reasonable assumption (at least 1 server being honest)
 - requires no soft/hardfork
 - is light-client friendly
- We propose a construction of InPoSW
 - It allows cost-efficient timestamping on data arriving at different times
 - It could be of independent interest for other timestamp applications

Questions?