EE 374: Fundamentals of Blockchain Infrastructure **Stanford, Spring 2025**

Lecture 11: Tendermint Recap and Economic Security for PoS Consensus

May 5, 2025

Lecturer: Prof. David Tse Scribe: Ivan-Aleksandar Mavrov

1 Agenda

In this lecture, we begin by recapping the Tendermint protocol, by including its fully specified "locking" and "temporary lock dropping" rules. We discuss trust assumptions versus economic security, contrasting Proof-of-Work's reliance on hash power with the economic guarantees of Proof-of-Stake. We then examine Ethereum's validator lifecycle of Staking and unbonding to show how these mechanisms enforce accountable and slashable safety. Finally, we discuss the limitations of slashable safety in PoS systems and the necessity of trust assumptions to prevent long-range attacks and unpenalized violations.

2 Tendermint Protocol Recap

2.1 Model

- **Validators:** Set of $n = 3f + 1$ validators, among which at most f are controlled by a Byzantine adversary.
- **Network model:** We work under the assumption of a **partially synchronous** network model.

2.2 Protocol State

Each honest validator follows the protocol by maintaining the following:

1. **Height** H: The height of the current block in the chain.
2. **Round** r: For each height level the protocol executes sequential rounds $r = 0, 1, 2, \cdots$
3. **Step:** Each round is built of the following three steps: **Proposal, Pre-vote, Pre-commit**.
4. **Locked round & locked block** : Upon sending a pre-commit for a block B at round r_{lock}, a validator becomes locked on block B.
5. **Valid round & valid block** : The highest round r such that the validator has seen $2f + 1$ round-r pre-votes for a block B_r.

2.3 Protocol Steps

At each height level H, the protocol proceeds in rounds $r = 0, 1, 2, \ldots$ until the protocol successfully commits a block.

1. **Proposal Step**: At time $t = 3\Delta r$, the proposal step begins with a round-r leader, which we denote by L_r. The leader sends and signs a proposal:

   $$\langle \text{Proposal}, r, vr, B \rangle_{L_r},$$

   where $vr = -1$ or vr denotes some round number smaller than r.

2. **Pre-vote Step**: At time $t = 3\Delta r + \Delta$, the pre-vote step begins.

   For a validator V who is not locked on any block, let $\langle \text{Proposal}, r, vr, B \rangle$ be the first round-r proposal observed by V. Then at $t = 3\Delta r + \Delta$, V sends and signs a pre-vote for this proposal:

   $$\langle \text{Pre-vote}, r, vr, h(B) \rangle_V$$

   under the following rules:

   - If $vr = -1$, the validator sends a pre-vote.
   - If $vr \geq 0$, pre-vote only if V has observed $2f + 1$ round-vr pre-votes for B of the form $\langle \text{Pre-vote}, vr, vr', h(B) \rangle$, for any value of $vr' < vr$.
   - Otherwise, send a nil pre-vote: $\langle \text{Pre-vote}, r, vr = -1, \perp \rangle_V$ where $\perp$ denotes the *nil* block.

3. **Pre-commit Step**: At time $t = 3\Delta r + 2\Delta$ the pre-commit step begins. For a validator V let B be the first round-r block such that V has observed $2f + 1$ round-r pre-votes $\langle \text{Pre-vote}, r, vr, h(B) \rangle$. Then V sends and signs a pre-commit message:

   $$\langle \text{Pre-commit}, r, h(B) \rangle_V$$

   If no such block B exists, then V sends pre-commit for the *nil* block $\langle \text{Pre-commit}, r, \perp \rangle_V$.

Proposal Rule Let r' be the largest round such that the leader L_r has observed $2f + 1$ round-r' pre-votes of the form $\langle \text{Pre-vote}, r', vr', h(B') \rangle$ for a block B'. Then, at round r, L_r proposes:

$$\langle \text{Proposal}, r, vr = r', B' \rangle$$

This ensures that B' is both a valid round-r' block and supported by a supermajority of validators.

If no such round r' exists, the leader can propose any block B with:

$$\langle \text{Proposal}, r, vr = -1, B \rangle.$$

Locking Rule A validator **locks** on a block B at round r upon sending a round-r pre-commit for B. Once locked on a block B from round r, the validator does **not** send pre-votes for any block $B' \neq B$ in any future round $r'' > r$, **unless** B' is proposed in a message of the form: $\langle \text{Proposal}, r'', vr = r', B' \rangle$, and the validator has observed $2f + 1$ round-r' pre-votes for B', where $r'' > r' > r$.

3 Example Execution of the Protocol

In this section we present an example, which illustrates the protocol execution under an adversary attack. The attack takes place across 3 consecutive rounds $r = 0, 1, 2$ which take a total of 9Δ time.

Round $r = 0$:

- $(t = 0)$: Adversary leader L_0 proposes block B_0. The adversary shows the proposal to only $f + 1$ honest validators by time Δ.
- $(t = \Delta)$: $f + 1$ honest validators send round-0 pre-votes for B_0, and f honest validators send round-0 pre-votes for $\perp$ (nil block).
- $(t = 2\Delta - \epsilon)$: Adversary validators send f round-0 pre-votes for B_0 only to honest validator v_1^h.
- $(t = 2\Delta)$: Only v_1 sends a round-0 pre-commit for B_0 and is locked on B_0.

Round $r = 1$:

- $(t = 3\Delta)$: An honest leader $L_1 \neq v_1^h$ proposes a block $B_1 \neq B_0$.
- $(t = 4\Delta)$: All $2f$ honest validators except v_1^h send round-1 pre-votes for B_1.
- $(t = 4\Delta)$: v_1^h sends round-1 pre-vote for $\perp$ (nil block).
- $(t = 5\Delta - \epsilon)$: Adversary shows 1 more round-0 pre-vote for B_1 to the $2f$ honest validators except v_1^h. These honest validators will broadcast this vote to everyone including v_1^h.
- $(t = 5\Delta)$: $2f$ honest validators send round-1 pre-commits for B_1, and they lock on block B_1
- $(t = 6\Delta - \epsilon)$: v_1^h observes $2f + 1$ round-1 pre-votes for B_1.

Round $r = 2$:

- $(t = 6\Delta)$: An honest leader $L_2 \neq v_1^h$ proposes block B_1 with $\langle \text{Proposal}, r = 2, vr = 1, B_1 \rangle_{L_2}$.
- $(t = 7\Delta)$: All $2f$ honest validators other than v_1 send round-2 pre-votes for B_1.
- $(t = 7\Delta)$: v_1^h temporarily drops its lock on B_0 and also sends a round-2 pre-vote for B_1, because v_1^h has already observed $2f + 1$ round-1 pre-votes for B_1.
- $(t = 8\Delta)$: B_1 has received $2f + 1$ round-2 pre-commits and gets confirmed here.

4 Trust Assumptions vs. Economic Security

As we discussed in previous lectures, trust and safety in blockchain systems exists on a spectrum from centralized trust to trustless mathematical guarantees. In Proof of Work systems like Bitcoin security relies on **trust assumptions** such as "secure if $> 50\%$ of hash power is honest". Such assumption nevertheless belongs to the "Trust Us, Bros" category of **Decentralized Trust**.

On the other hand, Proof of Stake systems such as Tendermint and Gasper provide an opportunity to build security that is based on Economic Security guarantees. For instance, economic security can be based on the claim that if security is violated, the adversary validators must lose at least \$10 million USD worth of stake. The size of the penalty should be designed so as to guarantee that the adversary cannot possibly profit by deviating from the protocol in the first place.

The ability of a PoS protocol to penalize the adversary relies on the **Accountable Safety** property. Accountability states that when the protocol detects a violation, it can also provide an irrefutable evidence that proves a given subset of validators must have violated the protocol and must thus be penalized. However, an implicit assumption that is necessary for translating Accountable Safety to Economic Security is that the coins staked by the adversary validators must still be staked at the time when the protocol detects a violation, in order for the protocol to be able to penalize the adversaries. In the following section we will discuss some of the technical steps taken by Ethereum to justify the validity of this assumption.

5 Ethereum Staking and Unbonding

In our discussion of the Tendermint Protocol we concluded that Accountable Safety is sufficient for Economic Security in the case when the set of validators does not change, so that the amount staked by the adversaries remains available to the protocol at all times. This assumption is clearly undesirable in any blockchain that aims at allowing its coins of being a medium of exchange. The Ethereum protocol has taken the approach of adding frictions to the process of changing the set of validators in order to provide sufficient amount of time for the protocol to detect potential violations (that might lead to double-spends).

To become a validator in Ethereum, one must **Stake** 32 ETH and submit validator credentials to the beacon chain. However, the validator does not join immediately. They must wait through a delay (at least four epochs long), designed to prevent last-minute manipulation of the RANDAO encryption. After this, the entry is governed by a churn limit, which limits how many new validators can join over a given time.

On the exit side, Ethereum includes an **Unbonding** period after a validator requests withdrawal. This delay is essential to economic security: if validators could instantly withdraw their stake, adversary validators could withdraw their funds before being penalized for a violation that is detected at a later height level. The delay is ensured by the following stages: Exit queue, unbonding delay, Withdrawal queue. Together this steps ensure that the ETH protocol has at least 7 days to detect a violation by the unbonding validators.

6 From Accountable to Slashable Safety

PoS systems strive to be accountably safe, which ensures that if a safety violation occurs (e.g., two conflicting blocks are finalized), then it must be possible to identify with an irrefutable evidence

at least one-third of the validators who caused it. However, accountability alone is not enough for economic security — those validators must also be slashable, which means that the protocol should be able to penalize them by confiscating their stake. Thus for Slashable Safety, the evidence of protocol violation must be present before the stake is withdrawn.

Ethereum addresses this by delaying withdrawals through the unbonding period. However, a negative theoretical result states that achieving the Slashable Safety guarantee without additional trust assumptions **is impossible** due to long range attacks as shown on Figure 1 This result highlights a critical limitation of Proof of Stake systems: their security often hinges not only on economic incentives but also on network and validator behavior assumptions.

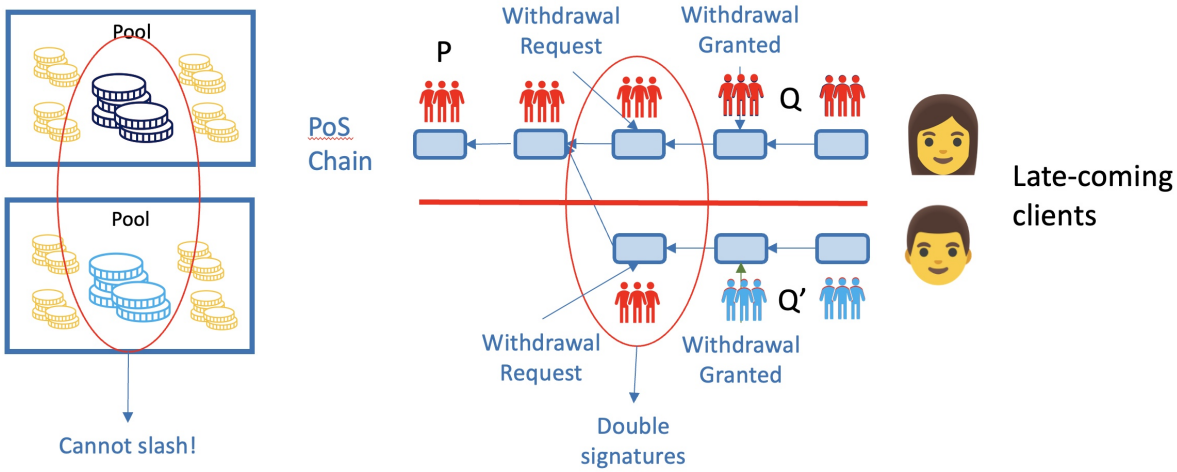

Figure 1: PoS Long Range Attack.

This figure illustrates a Longe Range Attack scenario in which clients joining later may only see the two differing branches after the malicious validators have already unbounded. The protocol no longer has access to their stakes and it becomes impossible for the protocol to penalize the malicious set of validators.

References