L04: Consensus Mechanisms

Extended Slides – BSc Blockchain Course

Digital Finance

2026

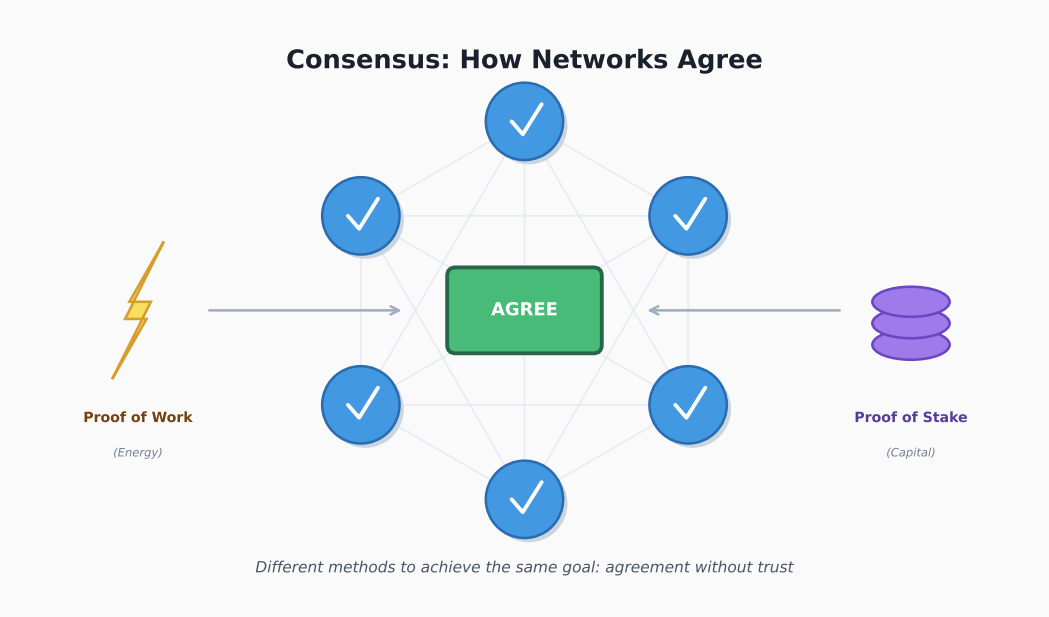
Learning Objectives

By the end of this lesson, you will be able to:

1. Compare Proof of Work and Proof of Stake mechanisms
2. Explain Byzantine Fault Tolerance requirements
3. Analyze trade-offs between finality, security, and efficiency
4. Understand slashing conditions and validator economics
5. Evaluate consensus mechanisms for various use cases

Prerequisites: L03 Bitcoin Deep Dive.

Agreement Without Authority



Purpose: Consensus mechanisms solve the fundamental problem: how do strangers agree on truth without a central authority? This is the heart of decentralization.

The difference between PoW and PoS affects security, energy use, and economics.

The Consensus Problem

Distributed Agreement Challenge:

- Multiple nodes must agree on single truth
- Some nodes may be offline or malicious
- No central authority to arbitrate
- Network messages can be delayed/lost

FLP Impossibility (1985):

- Impossible to guarantee consensus in async network
- With even one faulty process
- Must relax safety, liveness, or synchrony

Blockchain consensus makes practical trade-offs around FLP.

Byzantine Generals Problem

Original Problem (1982):

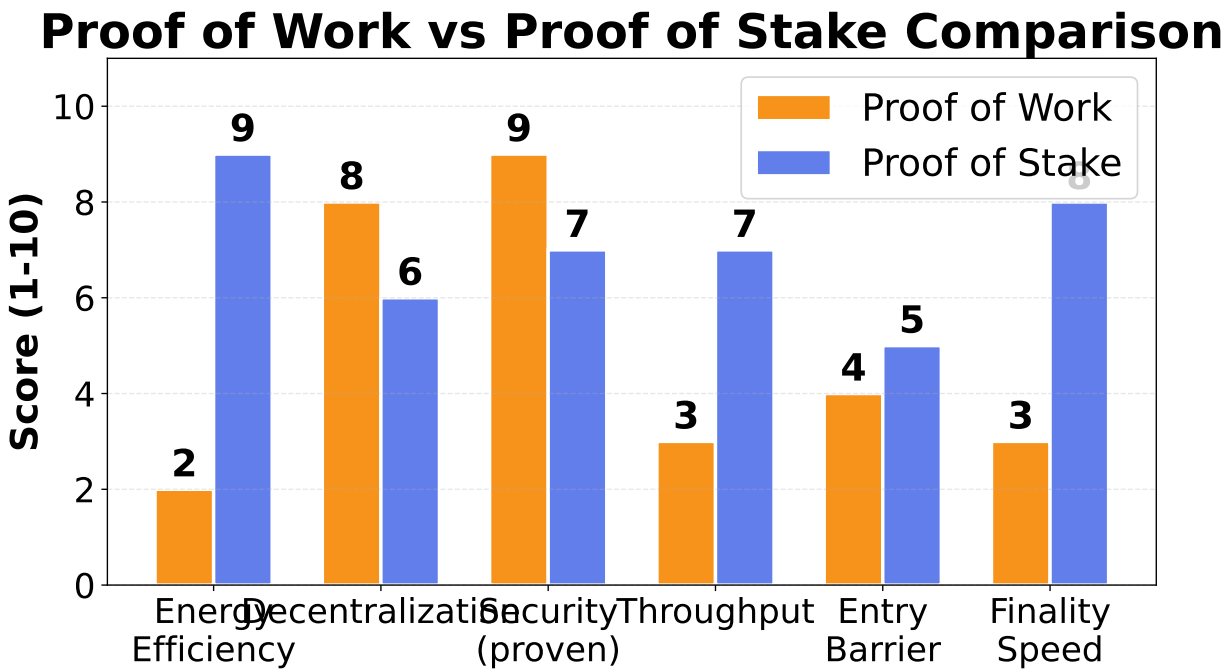
- Generals must coordinate attack
- Some generals may be traitors
- Only message passing (no shared memory)

Solution Requirements:

- $n \geq 3f + 1$ total nodes
- Where f = Byzantine (malicious) nodes
- Need 2/3 honest majority (67%)

Classic BFT requires knowing all participants – not permissionless.

PoW vs PoS Comparison



Different mechanisms optimize for different properties.

Proof of Work Explained

How PoW Works:

1. Miners compete to find valid block hash
2. Hash must be below difficulty target
3. Winner broadcasts block, gets reward
4. Other miners verify and build on it

Security Model:

- 51% attack: majority hashrate enables double-spends on recent txs
- Cannot steal funds or rewrite old history (only reorganize recent blocks)
- Sybil resistant: can't fake work

PoW converts energy expenditure into security.

Proof of Stake Explained

How PoS Works:

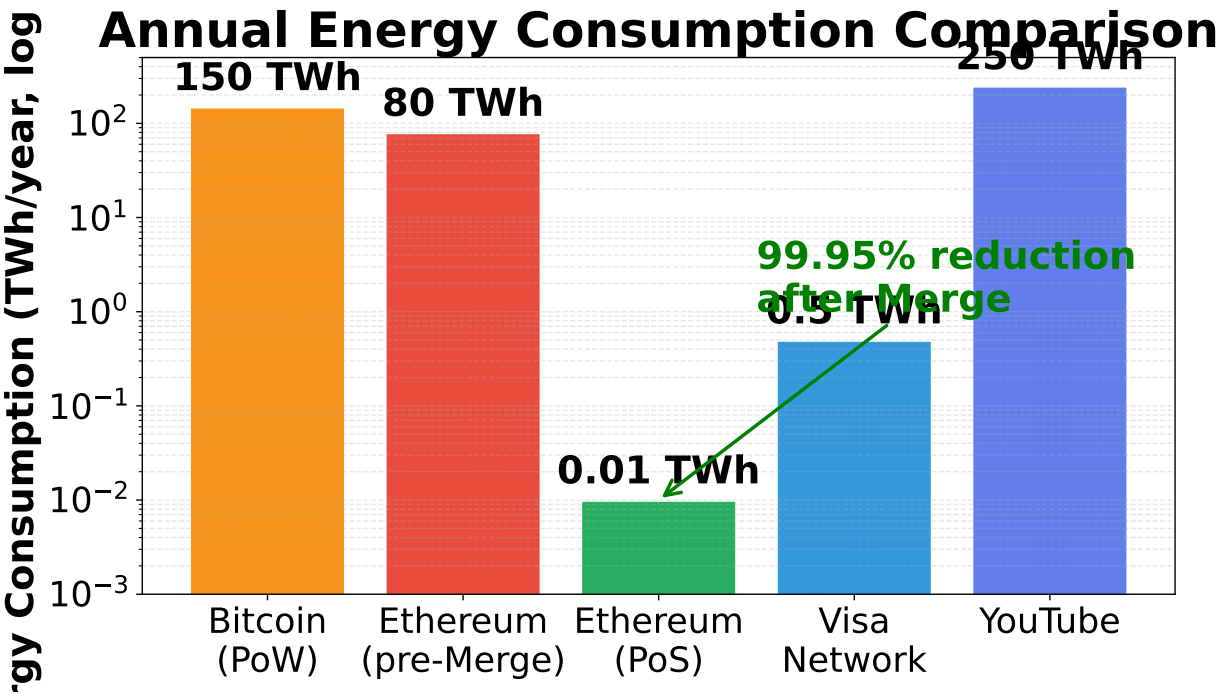
1. Validators lock tokens as stake
2. Protocol selects proposer (various methods)
3. Proposer creates block
4. Other validators attest (vote)
5. Rewards distributed, misbehavior slashed

Security Model:

- Attack requires acquiring 33-51% of stake
- Attack cost = token purchase + slashing risk
- "Nothing at stake" mitigated by slashing

PoS uses economic incentives instead of energy.

Energy Consumption



Energy debate: PoW proponents argue it backs security with real resources.

Finality Types

Probabilistic Finality (PoW):

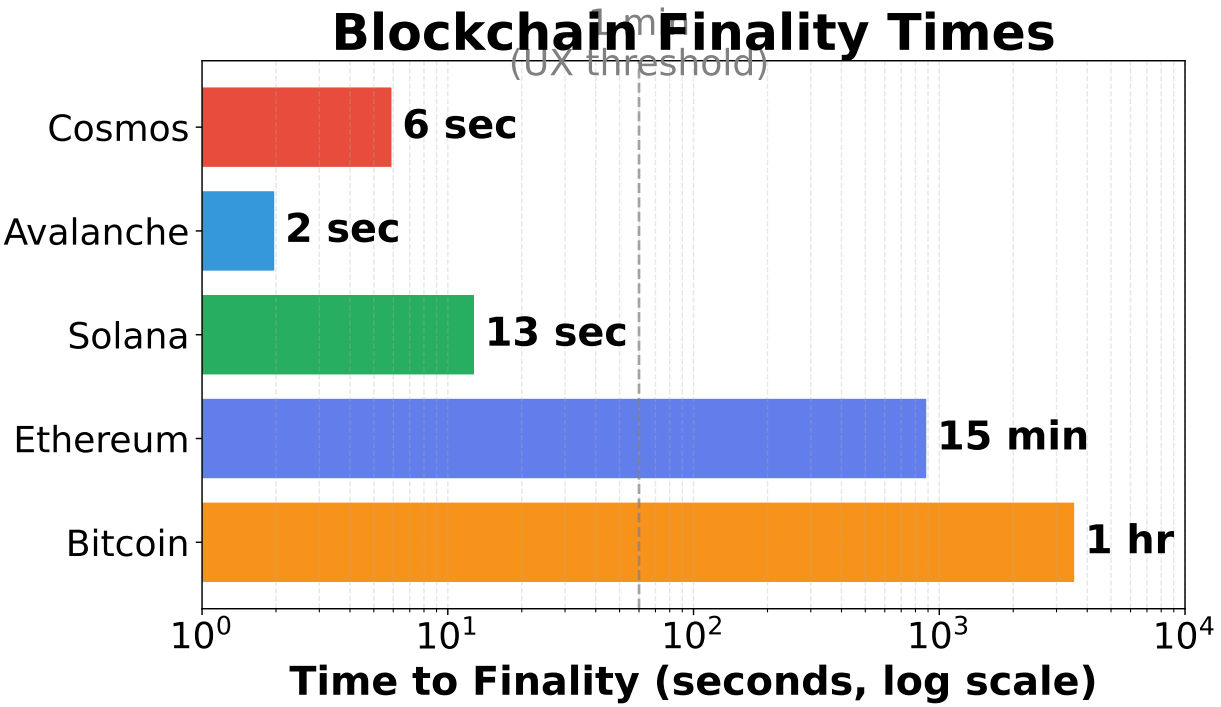
- Never 100% final, just increasingly unlikely to reverse
- More confirmations = more security
- Bitcoin: 6 confirmations standard

Economic Finality (PoS):

- Mathematically final after enough attestations
- Reversal requires sacrificing stake (slashing)
- Ethereum: 2 epochs (12.8 min) for finality

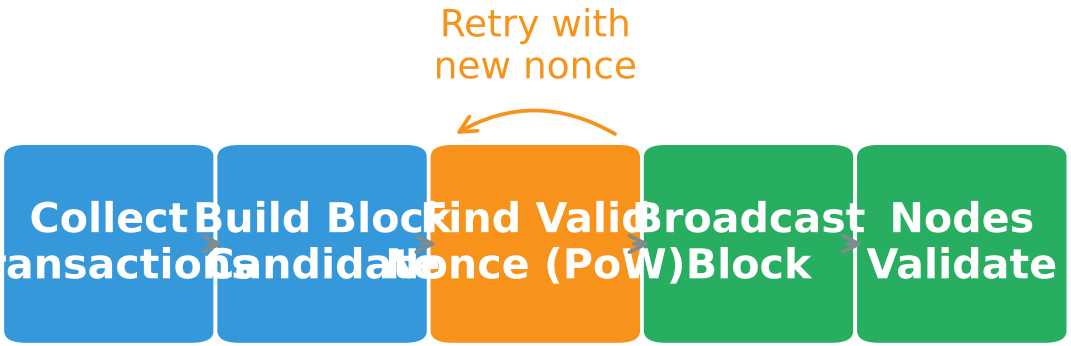
Finality affects user experience and exchange policies.

Finality Times



Faster finality enables better UX but may sacrifice security.

Nakamoto Consensus (Proof of Work)

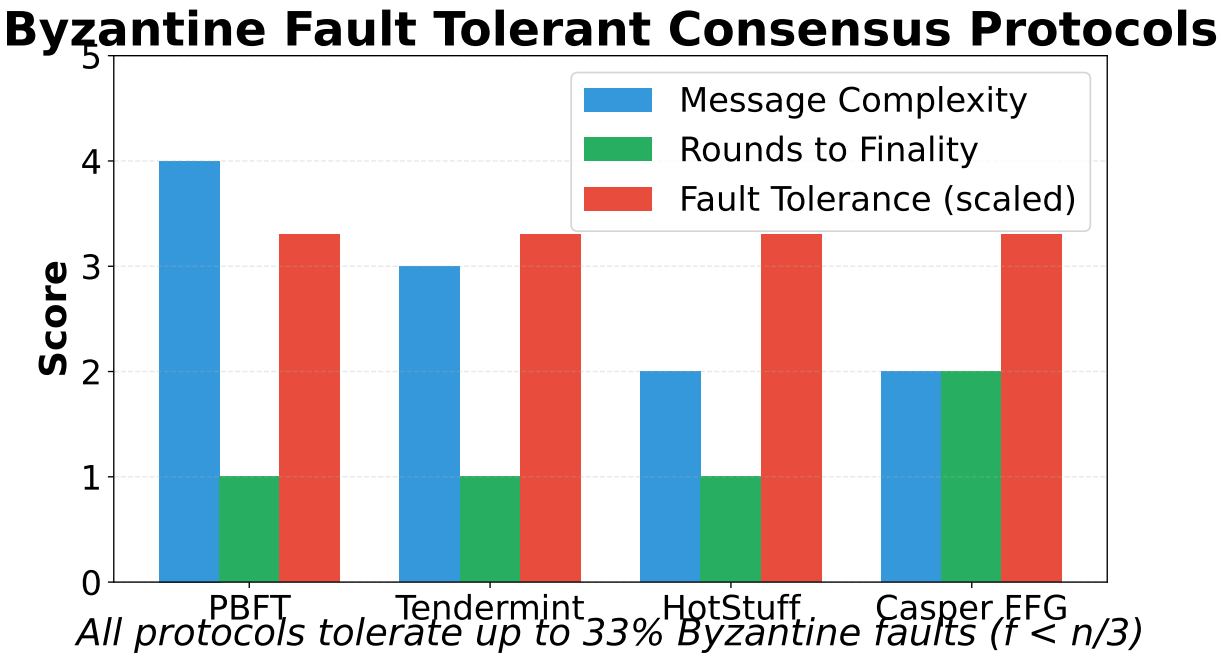


Key Properties:

- Probabilistic finality
- Energy-backed security
- Longest chain wins
- Permissionless

Satoshi's key insight: replace identity with computational work.

BFT Variants



All protocols tolerate up to 33% Byzantine faults ($f < n/3$)

Modern BFT protocols improve efficiency via threshold signatures and aggregation.

Tendermint/Cosmos Consensus

Key Features:

- BFT-based with instant finality
- Round-robin proposer selection
- 2/3 validator signatures required
- Used by Cosmos ecosystem

Trade-offs:

- Fast: 1-7 second finality
- Requires known validator set
- Stops if 1/3+ validators offline

Tendermint prioritizes finality over liveness.

Ethereum's Casper FFG

Hybrid Consensus:

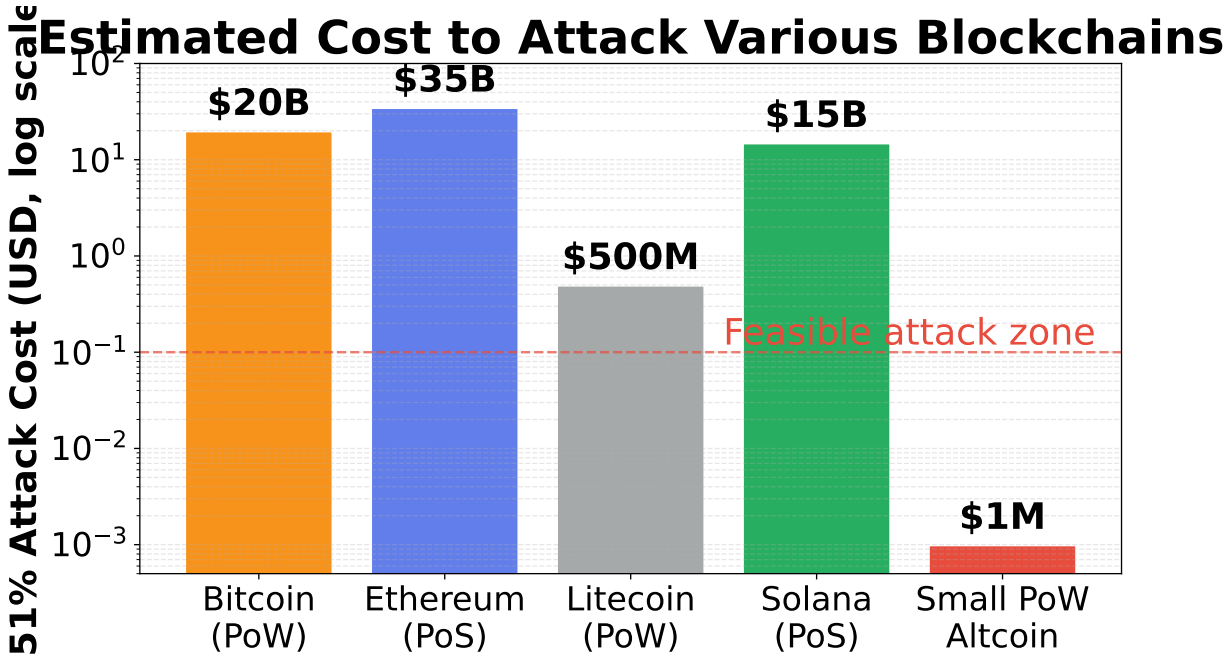
- LMD-GHOST for fork choice (liveness)
- Casper FFG for finality (safety)
- Combines best of both worlds

Finality Process:

- Validators attest to blocks each epoch
- Checkpoint finalized with 2/3 attestations
- Two epochs (64 slots) for full finality

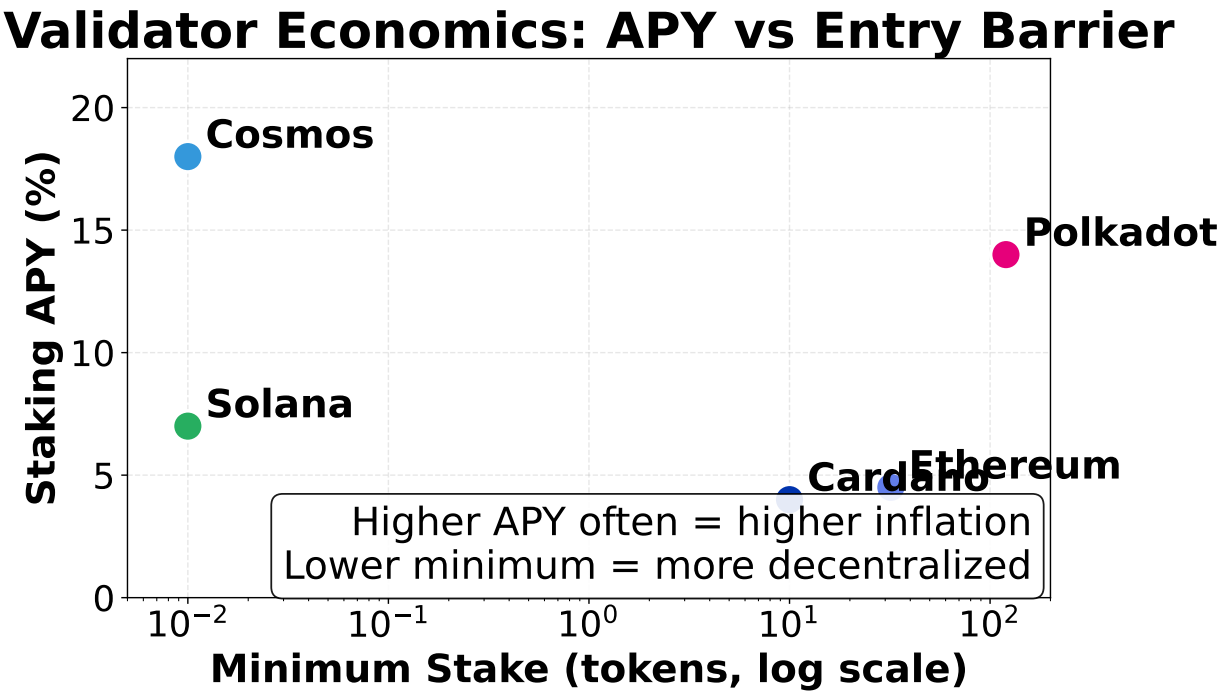
Ethereum's design allows chain to continue even without finality.

Attack Costs



Attack cost = economic security of the chain.

Validator Economics



Consider: nominal APY vs real yield (after inflation).

Slashing Conditions

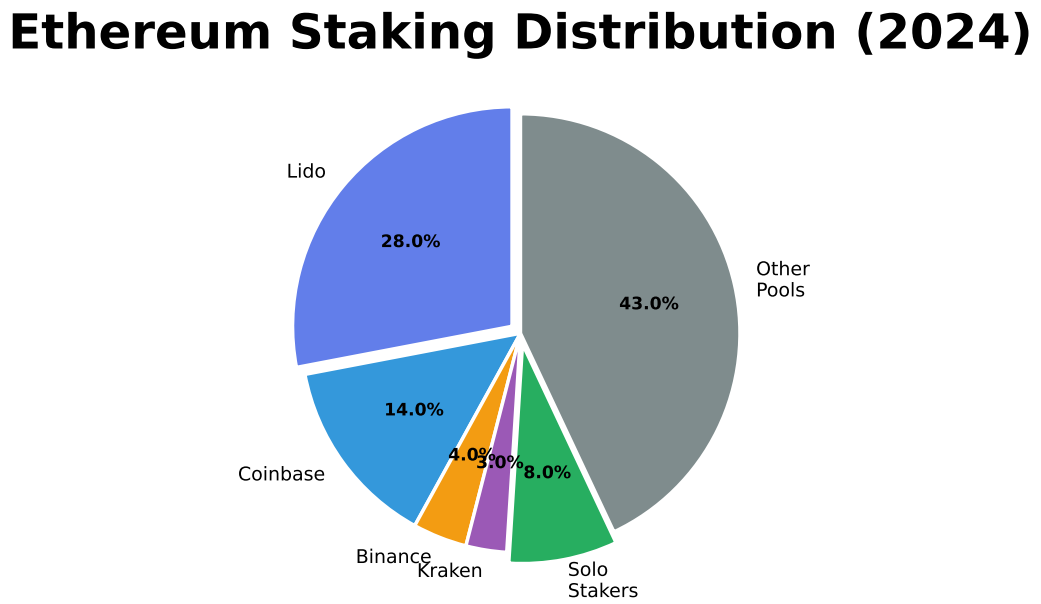
PoS Slashing Conditions

Violation	Penalty	Description
Downtime (Offline)	0.5-1%	Inactivity leak for offline validators
Double Vote / Equivocation	3-5%	Voting for two blocks at same height
Surround Vote	100%	Voting for block that contradicts earlier vote

Severity: Low (inactivity) -> Medium (equivocation) -> High (surround)

Slashing makes attacks expensive – "skin in the game".

Stake Distribution



Note: Top 3 entities control 46% of stake

Centralization risk is a major PoS critique.

Other Consensus Mechanisms

Delegated PoS (DPoS):

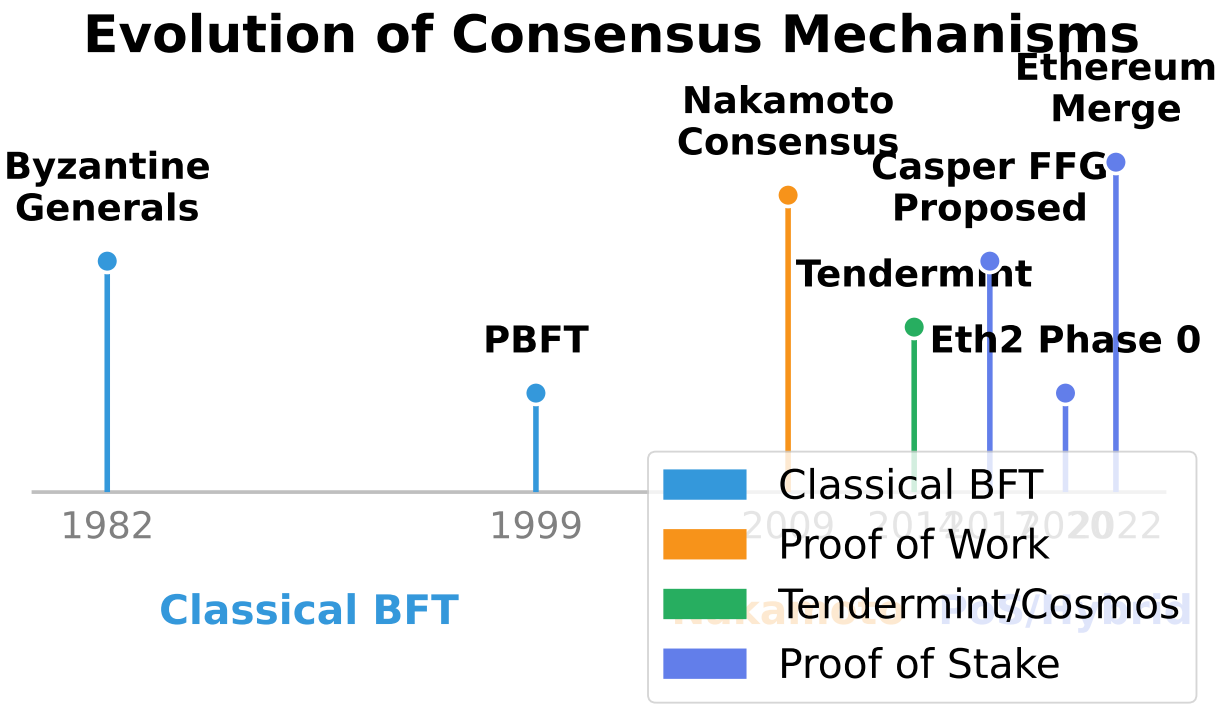
- Token holders vote for delegates
- Small validator set (21-101)
- High throughput, lower decentralization
- Example: EOS, TRON

Proof of Authority (PoA):

- Known, trusted validators
- Used in private/consortium chains
- Fast but not permissionless

Many variations exist – each with unique trade-offs.

Consensus Timeline



40 years from theory to global financial infrastructure.

Choosing a Consensus Mechanism

Consider These Factors:

1. **Security requirements:** How much value at risk?
2. **Decentralization needs:** Permissionless or consortium?
3. **Throughput requirements:** How many TPS needed?
4. **Finality needs:** Probabilistic OK or need instant?
5. **Energy constraints:** Environmental concerns?

There is no universally "best" consensus mechanism.

Key Takeaways

Remember These Points

1. PoW: energy-backed security, proven, permissionless
2. PoS: capital-backed security, efficient, faster finality
3. BFT requires 2/3 honest (tolerates 1/3 Byzantine)
4. Slashing penalizes misbehavior in PoS
5. Trade-offs: decentralization vs throughput vs finality
6. No perfect solution – choose based on requirements

Next Lesson: Ethereum and Smart Contracts.