Consensus Mechanisms

Lesson 4: How Blockchains Agree on Truth

Prof. Joerg Osterrieder

Spring 2026

Learning Objectives

After this lesson, you will be able to:

- Explain the Byzantine Generals Problem
- Describe how Proof of Work achieves consensus
- Compare Proof of Work and Proof of Stake
- Evaluate trade-offs between different consensus mechanisms

Prerequisites: Lessons 1-3 (Intro, Blockchain, Cryptography)

Conse

is how decentralized networks agree without central authority

Lesson Outline

1. The Consensus Problem
2. Proof of Work
3. Proof of Stake
4. Other Mechanisms
5. Choosing a Consensus Mechanism

The Consensus Problem

Why Consensus Matters

The Problem:

- Distributed network with no central authority
- Nodes may be offline, slow, or malicious
- Need to agree on single version of truth

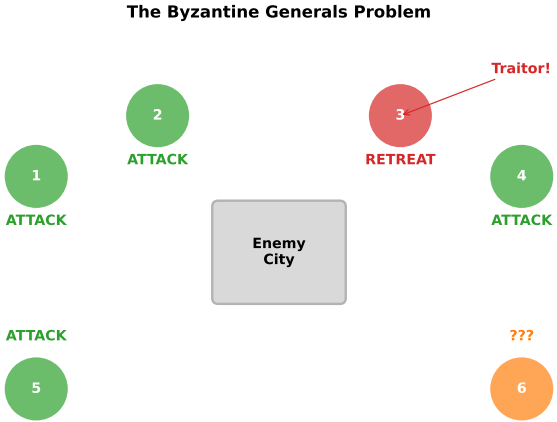
Requirements for Consensus:

- **Agreement**: All honest nodes decide same value
- **Validity**: Decided value was proposed by some node
- **Termination**: All honest nodes eventually decide
- **Fault tolerance**: Works even if some nodes fail

With

consensus, blockchains cannot function

The Byzantine Generals Problem



Problem: How can loyal generals agree when some may be traitors?

How can generals coordinate when some may be traitors sending conflicting messages? Named after the Byzantine Empire, known for court intrigue and untrustworthy generals.

Byzantine Fault Tolerance

The Problem (1982):

- Generals must agree to attack or retreat
- Messages can be lost or corrupted
- Some generals may be traitors

The Solution:

- Need at least $3f + 1$ generals to tolerate f traitors
- *Why 3f+1?* With f traitors, you need $2f + 1$ honest nodes for majority. Since traitors may send conflicting messages, you need extra f to ensure honest majority prevails.
- Multiple rounds of message passing
- Majority voting with verification

Blockchain Innovation: Bitcoin solved this for open networks using economics—making identity creation costly through Proof of Work. Creating fake identities (Sybil attack) requires real computational resources, replacing the need for a known participant list.

Class

BFT requires knowing all participants; Bitcoin makes creating fake identities prohibitively expensive

Proof of Work

What is Proof of Work?

Definition: Consensus mechanism where participants prove computational work to add blocks.

The Mining Puzzle:

- Find nonce such that: $H(\text{block header}) < \text{target}$
- *(A hash is a large number in hexadecimal; "less than target" means the hash's numeric value is below the threshold.)*
- Hash must start with certain number of zeros
- Only way to find: brute-force trial and error

Why It Works:

- Hard to find valid nonce (requires work)
- Easy to verify (just compute one hash)
- Cannot cheat without doing the work

converts electricity into security

Mining Process

Step-by-Step:

1. Collect pending transactions from mempool
2. Create block header with Merkle root
3. Try nonce $= 0$, compute hash
4. If hash $>$ target, increment nonce and retry
5. If hash $<$ target, broadcast winning block
6. Other miners verify and accept
7. Winner receives block reward + fees

Bitcoin Stats:

- Target: Adjust every 2016 blocks (2 weeks)
- Goal: One block every 10 minutes

miners try billions of nonces per second

Difficulty Adjustment

Problem: Network hash rate changes over time.

Solution: Automatic difficulty adjustment

- If blocks too fast: increase difficulty
- If blocks too slow: decrease difficulty
- Bitcoin: Adjusts every 2016 blocks

Formula:

$$\text{New Difficulty} = \text{Old Difficulty} \times \frac{\text{Actual Time}}{\text{Expected Time}}$$

Diffic

has increased by 10^{14} since Bitcoin's launch

Security Analysis

51% Attack:

- Attacker needs majority of hash power
- Can rewrite recent blocks
- Cannot steal funds or create coins
- Can double-spend their own transactions

Economic Defense:

- Cost: Billions in hardware and electricity
- Reward: Potentially double-spend some transactions
- Risk: Devalue the network you're attacking

Attac

Bitcoin is economically irrational

PoW: Strengths and Weaknesses

Sybil Attack: When an adversary creates many fake identities to gain disproportionate influence. Named after a case study of dissociative identity disorder.

Strengths:

- Proven security (15+ years for Bitcoin)
- Permissionless participation
- Objective chain selection (most work)
- Sybil resistant

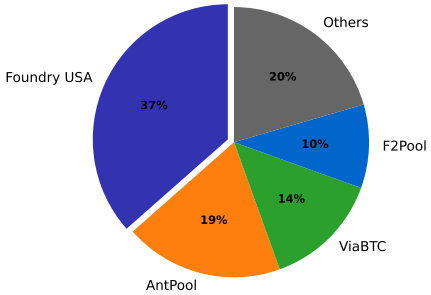
Weaknesses:

- High energy consumption
- Hardware centralization (ASICs—Application-Specific Integrated Circuits)
- Slow finality
- Limited throughput

prioritizes security and decentralization over efficiency

Mining Pool Centralization

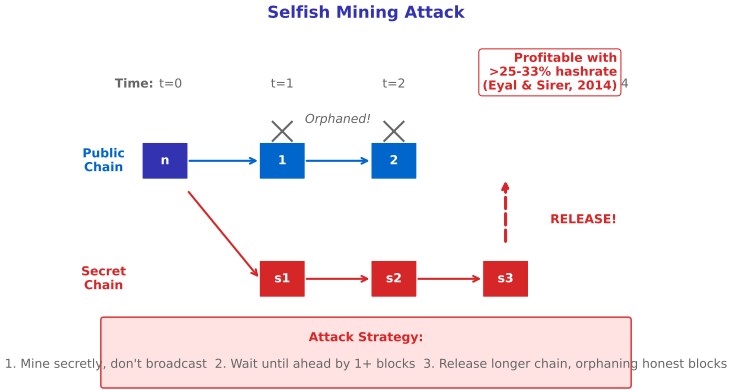
Bitcoin Mining Pool Distribution (as of January 2026)



Top 4 pools control ~80% of hashrate

Pool concentration creates censorship and 51% attack risks

Advanced Attack: Selfish Mining



Selfis

mining can be profitable with 25-33% of hash power—less than 51%

Understanding Selfish Mining

Strategy:

- Mine blocks but don't broadcast immediately
- Build a secret longer chain
- Release when ahead to orphan honest blocks

Why It Works:

- Honest miners waste work on orphaned blocks
- Attacker gets more than proportional rewards
- Effective with γ (network connectivity advantage; γ represents the attacker's network connectivity advantage)

Defense Mechanisms:

- Uniform tie-breaking (Bitcoin doesn't use this)
- Shorter block times make attack harder
- Monitoring for suspicious mining patterns

First described by Eyal & Sirer (2013)—remains a theoretical concern

Proof of Stake

What is Proof of Stake?

Definition: Consensus mechanism where validators stake tokens as collateral.

Key Idea:

- Replace computational work with economic stake
- Validators selected based on staked amount
- Malicious behavior = lose staked tokens (slashing)

Why It Works:

- Validators have "skin in the game"
- Attacking hurts your own investment
- Economic incentives align with network security

converts capital at risk into security

Validator Selection

Methods:

- **Random selection**: Weighted by stake
- **Coin age**: Stake $\times$ time held
- **Delegation**: Users delegate to validators

Ethereum (Post-Merge):

- Minimum stake: 32 ETH
- Random selection per slot (12 seconds)
- Committees of validators vote
- Rewards for correct attestations

Rand

prevents prediction and manipulation

Slashing: The Punishment Mechanism

What Gets Slashed:

- Double signing (proposing two blocks)
- Surround voting (conflicting attestations)
- Being offline (minor penalty)

Consequences:

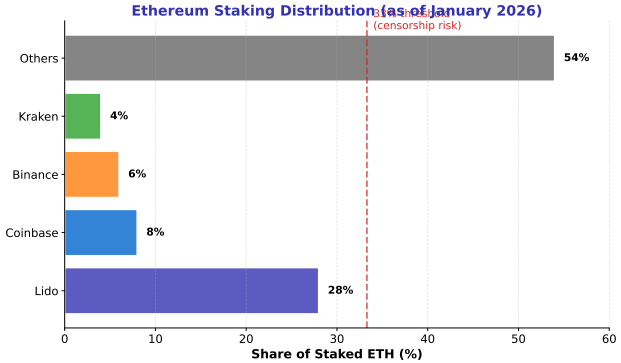
- Lose portion of staked tokens
- Forced exit from validator set
- Cannot rejoin immediately

Purpose: Make attacks economically destructive.

Slash

makes the cost of attack explicit and immediate

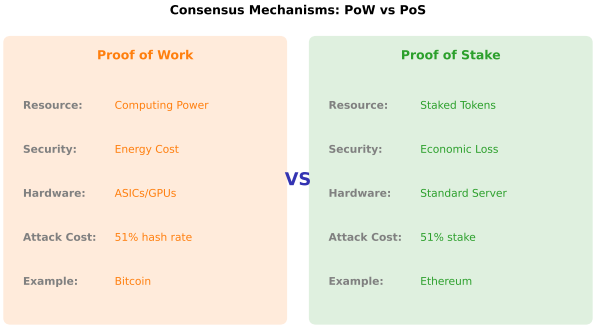
Staking Centralization Risks



Liqui

staking protocols like Lido concentrate stake—33% threshold enables censorship

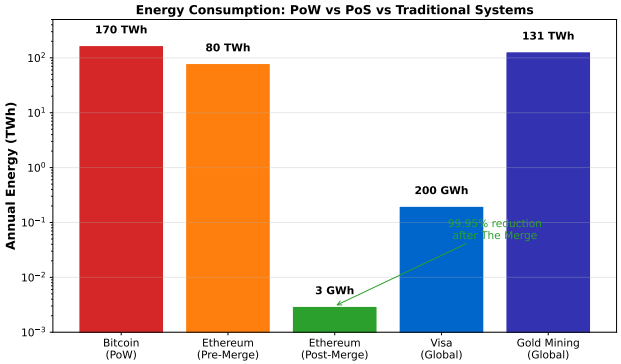
PoW vs PoS Comparison



PoW

spend energy to earn, PoS: risk capital to earn

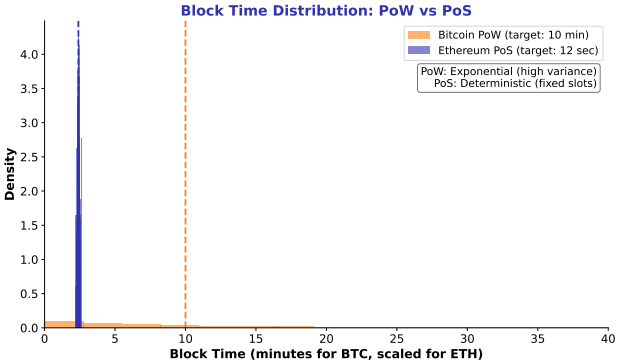
Energy Consumption



Ether

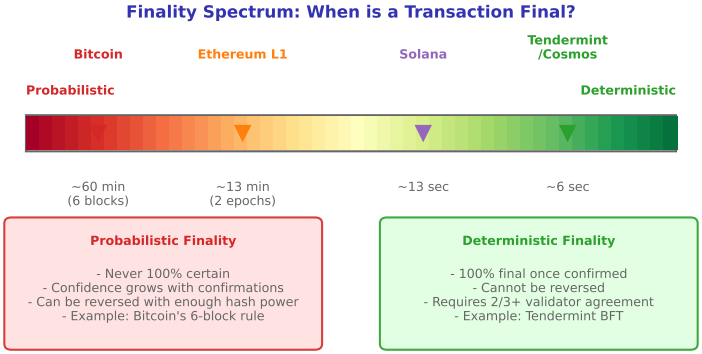
reduced energy use by 99.95% switching from PoW to PoS

Block Time Consistency: PoW vs PoS



PoW follows exponential distribution (high variance), PoS uses fixed time slots (predictable)

Finality: When Is a Transaction Final?

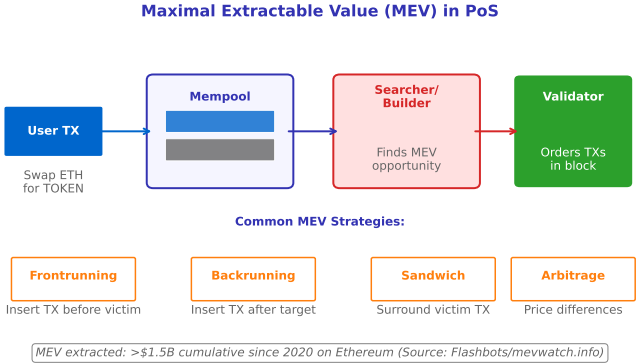


Trade-off: Faster finality often requires more trust assumptions

Finali

determines when you can safely consider a transaction irreversible

MEV: Maximal Extractable Value



MEV is "invisible tax" on users—validators profit from transaction ordering

Understanding MEV in PoS

What is MEV?

- Value extractable by manipulating transaction order
- Validators/builders can reorder, insert, or censor TXs
- Previously called "Miner Extractable Value"

Impact on Users:

- Sandwich attacks: Users get worse prices
- Frontrunning: Arbitrageurs take profit first
- Failed transactions: Competing bots cause reverts

Solutions (Ethereum):

- **Flashbots**: Private mempool, fair ordering
- **MEV-Boost**: Separates builder and proposer roles
- **Intent-based systems**: Hide transaction details

MEV

is a fundamental challenge—cannot be eliminated, only redistributed

Other Mechanisms

Delegated Proof of Stake (DPoS)

How It Works:

- Token holders vote for delegates
- Fixed number of delegates produce blocks
- Delegates can be voted out

Examples: EOS, Tron, Lisk **Trade-offs:**

+ Fast finality, high throughput
- More centralized (few delegates)
- Potential for collusion

sacrifices decentralization for performance

Practical Byzantine Fault Tolerance (PBFT)

How It Works:

- Pre-defined set of validators
- Three-phase voting protocol
- Tolerates up to 1/3 malicious nodes

Examples: Hyperledger Fabric, Tendermint **Trade-offs:**

+ Fast, deterministic finality
+ No energy waste
- Requires known validator set
- Does not scale well

PBFT

is ideal for permissioned blockchains

Proof of Authority (PoA)

How It Works:

- Validators are pre-approved entities
- Identity is at stake, not tokens or work
- Used in private/consortium chains

Examples: VeChain, Ethereum testnets **Trade-offs:**

+ Very fast and efficient
+ Low hardware requirements
- Centralized trust
- Not suitable for public chains

prioritizes efficiency over decentralization

Choosing a Consensus Mechanism

The Blockchain Trilemma

Cannot optimize all three simultaneously:

1. **Security**: Resistance to attacks
2. **Decentralization**: No single point of control
3. **Scalability**: High transaction throughput

Trade-offs by Mechanism:

- PoW: Security + Decentralization (sacrifices scalability)
- DPoS: Security + Scalability (sacrifices decentralization)
- PBFT: Security + Scalability (sacrifices decentralization)

Every consensus mechanism makes trade-offs

Key Takeaways

What You Learned Today:

1. Byzantine Generals Problem: Agreement with potential traitors
2. Proof of Work: Spend energy for security
3. Proof of Stake: Risk capital for security
4. Trade-offs: Security vs. decentralization vs. scalability

Core Insight: Consensus mechanisms convert some scarce resource (energy, capital, reputation) into network security.

Next lesson: Token Economics

Questions for Reflection

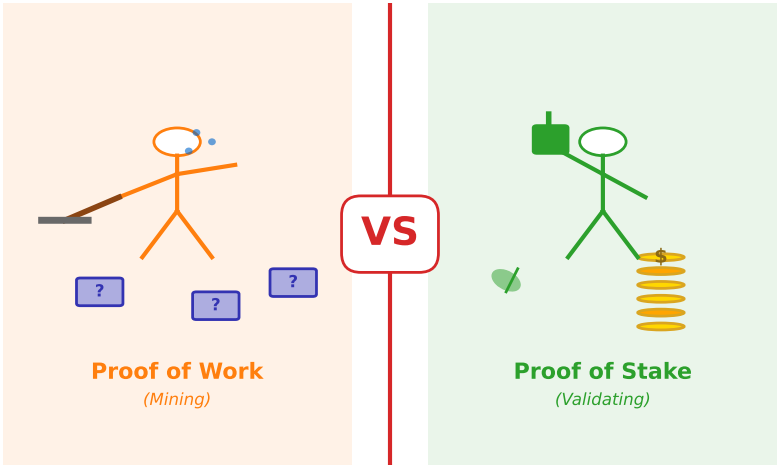
1. Why is PoW called “one CPU one vote”?
2. How does slashing make PoS secure?
3. When would you choose DPoS over PoW?
4. Can the blockchain trilemma be solved?

Discussion: Is PoW’s energy consumption justified for security?

Consi

these questions before our next session

Visual Break



Data Sources

Primary References for Course Data:

1. Cambridge Bitcoin Electricity Consumption Index (CBECI)
2. Ethereum Foundation: PoS Documentation
3. Digiconomist: Bitcoin Energy Consumption Tracker
4. Blockchain.com: Bitcoin Network Hash Rate Charts
5. MEV Watch: Real-time MEV Monitoring

All data current as of January 2026. Energy estimates include both direct mining and cooling infrastructure.

Verify

data sources independently for research purposes

Thank You

Questions?

Course materials: digital-ai-finance.github.io/crypto-economics

Appendix: Technical Deep Dives

Case Study: The Ethereum Merge (September 2022)

What Happened:

- Ethereum switched from PoW to PoS
- Largest blockchain upgrade in history
- Zero downtime during transition

Results:

- Energy reduction: 99.95% (Ethereum Foundation estimate) (from 100 TWh to 0.01 TWh/year)
- New ETH issuance dropped 90%
- Created world's largest staking economy

Technical Achievement:

- Beacon Chain ran parallel since Dec 2020
- Difficulty bomb forced transition
- No "replay" of transactions needed

The Merge proved major PoW chains can transition to PoS safely

Technical: The Nothing-at-Stake Problem

The Problem:
In PoS, validators can vote on multiple chains for free (no work required).

Why It's Dangerous:

- Validators can hedge by supporting all forks
- No cost to supporting conflicting histories
- Could prevent consensus from forming

Solutions:

- **Slashing**: Destroy stake for double-voting (Ethereum)
- **Deposit lockup**: Can't withdraw during dispute window
- **Checkpointing**: Finalize blocks periodically

Ethereum's Approach:
Validators who attest to conflicting blocks lose 1/32 of stake initially, up to 100% if attack is widespread.

Slash

converts "nothing at stake" to "everything at stake"

Technical: Long-Range Attacks

The Attack:

- Attacker acquires old private keys (from early validators)
- Creates alternative history from early block
- New nodes can't distinguish real vs fake chain

Why PoS is Vulnerable:

- No work to rewrite history (unlike PoW)
- Old keys have no current stake (already withdrawn)
- Can create plausible-looking fork

Solutions:

- **Weak subjectivity**: Trust recent checkpoint
- **Key deletion**: Validators must delete old keys
- **Social consensus**: Community rejects obvious attacks

Long-range attacks are theoretical—require social engineering to succeed

Technical: Consensus Mechanisms Comparison

Property	PoW	PoS	DPoS	PBFT
Validators	Anyone	Stakers	Elected	Permissioned
Finality	Probabilistic	Probabilistic*	Fast	Immediate
TPS	3-7	15-30	1000+	1000+
Energy	Very High	Low	Low	Low
51% Attack	Hash power	Stake	Stake+Votes	1/3 nodes
Decentralization	High	Medium-High	Medium	Low
Example	Bitcoin	Ethereum	EOS	Hyperledger

*Ethereum PoS provides economic finality via slashing after 2 epochs (13 min)

No perfect consensus mechanism exists—each makes different trade-offs