Lesson 04 — Consensus Mechanisms

Study Notes

Cryptoeconomics — BSc Level

Joerg Osterrieder

2025

Contents

Learning Objectives

By the end of this lesson, students should be able to:

1. **Explain** the Byzantine Generals Problem and state Lamport's $n \geq 3f + 1$ theorem precisely.
2. **Describe** how Proof of Work achieves Sybil-resistant consensus through the mining algorithm and difficulty adjustment.
3. **Compare** Proof of Work and Proof of Stake across dimensions of energy, security, decentralisation, and finality.
4. **Analyse** advanced attack strategies including selfish mining and MEV extraction.
5. **Evaluate** BFT-based consensus variants (PBFT, Tendermint, HotStuff) and delegated mechanisms (DPoS).

Byzantine Generals Problem

Formal Setup

Byzantine Generals Problem (Lamport, Shostak, Pease 1982)

n generals, connected only by messengers, must unanimously decide to *attack* or *retreat*. Up to f of them are *traitors* who send contradictory messages to different peers. The honest generals must reach the same decision despite the traitors.

Lamport's Theorem: Consensus is achievable if and only if $n \geq 3f + 1$, i.e., fewer than one-third of participants are Byzantine (arbitrarily faulty).

The intuition for the $3f + 1$ bound: with f traitors, the honest majority must be at least $2f + 1$ (to outvote traitors). But traitors may also impersonate absent honest generals, so an additional f nodes are required to detect the deception — giving $f + (2f + 1) = 3f + 1$ total.

Classical BFT Assumptions

Classical BFT algorithms (e.g., PBFT) operate under the following assumptions:

- The identity and total count n of all participants is *known in advance* (closed membership).
- Message complexity is $O(n^2)$ per consensus round.
- Network is partially synchronous: messages arrive within a known bounded delay.

These assumptions make classical BFT impractical for open, permissionless networks where participants can join and leave freely.

Relevance to Blockchain

Bitcoin does not implement classical BFT. Instead it achieves consensus via a *Sybil-resistance mechanism*: creating a new identity costs real computational work, so an attacker cannot cheaply outnumber honest nodes.

Sybil Attack

An adversary creates many fake identities to gain disproportionate influence over a vote or network. Named after the 1973 book "Sybil" about a patient with dissociative identity disorder. PoW defeats Sybil attacks by making identity creation proportional to hashrate expenditure.

The security guarantee shifts from "less than $\frac{1}{3}$ are traitors" to "honest participants control more than 50% of hashrate (PoW) or stake (PoS)."

Proof of Work

Mining Algorithm

PoW Mining Puzzle

Find a *nonce* $\eta \in \mathbb{Z}_{\geq 0}$ such that:

$$H(\text{block header} \| \eta) < T$$

where H is SHA-256, $\|$ denotes concatenation, and T is the current *target* value. The hash output is treated as an unsigned 256-bit integer; a smaller output corresponds to a hash with more leading zeros.

Step-by-step mining process:

1. Collect pending transactions from the *mempool*.
2. Assemble a block header: previous block hash, Merkle root of transactions, timestamp, difficulty bits, nonce (initially 0).
3. Compute $H(\text{header})$. If result $< T$, broadcast the block.
4. Otherwise increment η and repeat (brute-force search).
5. Upon winning: collect block reward (newly issued BTC) plus all transaction fees.

Asymmetry: finding a valid nonce requires $\sim T^{-1}$ hash evaluations on average, but *verification* requires exactly one hash computation. This asymmetry is the foundation of PoW security.

Difficulty Adjustment

Bitcoin's target adjusts every **2016 blocks** (approximately two weeks) to maintain an average inter-block time of **10 minutes**:

$$T_{\text{new}} = T_{\text{old}} \times \frac{t_{\text{actual}}}{t_{\text{expected}}}$$

where $t_{\text{expected}} = 2016 \times 10$ min $= 20{,}160$ minutes. If the last 2016 blocks were mined in less than two weeks the target *decreases* (harder); if they took longer it *increases* (easier).

Block Arrival as a Poisson Process

Because each hash attempt independently succeeds with probability $p = T/2^{256}$ and the memoryless property holds, block inter-arrival times follow an **exponential distribution**:

$$f(t) = \lambda e^{-\lambda t}, \qquad \lambda = p \times \text{hashrate}$$

The target is set so $1/\lambda \approx 600$ seconds. Consequences:

- Variance in inter-block times is high (CV = 1 for exponential).
- Two miners may find blocks nearly simultaneously, causing a temporary fork — resolved by the longest-chain rule.
- Shorter target block times increase fork frequency, weakening security.

Security: The 51% Threshold

An attacker with fraction $\alpha > 0.5$ of total hashrate can:

- Rewrite recent blocks (double-spend their own transactions).
- Exclude transactions from specific addresses (censorship).

What an attacker *cannot* do even with 100% hashrate:

- Steal coins from addresses they do not control.
- Create coins beyond the protocol's issuance schedule.
- Reverse transactions buried under many blocks.

Example: Cost of a 51% Attack on Bitcoin

As of early 2026, Bitcoin's hashrate exceeds 700 EH/s. Acquiring 51% requires $\approx$ 350 EH/s of hardware (tens of billions USD in ASICs) plus ongoing electricity costs. Revenue from double-spending a handful of transactions is orders of magnitude smaller, making the attack economically irrational. The attacker also devalues the network they control.

Proof of Stake

Validator Selection

Proof of Stake

Validators *lock* ("stake") tokens as collateral. The protocol pseudo-randomly selects block proposers, weighted by staked amount. Honest participation earns staking rewards; dishonest participation results in *slashing* (partial or total loss of staked tokens).

Ethereum's PoS (post-Merge) specifics:

- **Minimum stake**: 32 ETH per validator.

- **Slot time**: 12 seconds; one proposer per slot.
- **Committees**: Validators are grouped into committees that *attest* (vote) to the proposer's block.
- **Randomness**: RANDAO beacon provides unbiased randomness for validator selection, refreshed every epoch (32 slots, ≈ 6.4 min).

Staking Economics

Validators earn rewards from two sources:

1. **Block proposals**: Base reward for each proposed block.
2. **Attestations**: Reward for correctly attesting to the canonical chain head; penalised for missing or wrong attestations.

Real yield for stakers:

$$r_{\text{real}} = r_{\text{nominal}} - \pi$$

where π is the protocol's net issuance rate (inflation minus burn). If EIP-1559 burns exceed new issuance, $r_{\text{real}} > r_{\text{nominal}}$.

Slashing Conditions

Slashing

Slashable offences in Ethereum PoS:

- **Double proposal**: Signing two different blocks for the same slot.
- **Surround voting**: Signing conflicting attestations that "surround" each other in epoch height.

Penalty: initial slash of $\frac{1}{32}$ of stake, escalating to 100% if many validators are slashed simultaneously (the *correlation penalty* deters coordinated attacks). Slashed validators are also ejected from the validator set.

Non-slashable but penalised: being offline. The *inactivity leak* gradually drains stake of validators who fail to participate, ensuring the chain can finalise even if a large fraction goes offline.

Nothing-at-Stake Problem

In naive PoS, a validator could cheaply vote on multiple competing forks (no work is spent), preventing consensus. Slashing solves this: voting on conflicting chains is a slashable offence with large financial cost.

PoW vs PoS Comparison

Dimension	Proof of Work	Proof of Stake
Energy	Very high; Bitcoin ≈ 150 TWh/year. Converts electricity to security.	Low; Ethereum ≈ 0.01 TWh/year post-Merge (−99.95%).
Decentralisation	Open to anyone with hardware. ASIC manufacturing concentration is a practical centralisation risk.	Open to anyone with capital. Liquid staking protocols (e.g., Lido with > 30% of staked ETH) create concentration risk.
Security model	Attack cost = hardware + electricity. Sybil-resistant by construction. Long-range rewriting requires re-doing all work.	Attack cost = acquiring ≥ 33% of stake. Long-range attacks possible (mitigated by weak subjectivity checkpoints).
Finality	*Probabilistic*: probability of reversal decays exponentially with confirmations. 6 blocks (≈ 60 min) is conventional for Bitcoin.	*Economic/deterministic*: Ethereum finalises after 2 epochs (≈ 12.8 min); reversal requires burning ≥ $\frac{1}{3}$ of all staked ETH.
Throughput	Bitcoin: 3–7 TPS. Limited by block size and 10-min target.	Ethereum: 15–30 TPS on base layer; much higher with L2 rollups.
Sybil resistance	Hashrate is costly to acquire.	Stake is costly to acquire; also slashed for misbehaviour.

Common Pitfall

Misconception: "PoS is strictly better than PoW."

Correction: PoS introduces different trust assumptions. PoW's security is grounded in *physics* (energy expenditure); PoS security rests on *economic rationality* and the assumption that attackers would not destroy the value of their own stake. PoS also introduces long-range attack vectors absent in PoW.

DPoS and Variations

Delegated Proof of Stake

DPoS

Token holders vote to elect a small fixed set of *block producers* (delegates). Only delegates produce and validate blocks. Delegates can be voted out at any time if they behave maliciously or go offline.

Block production: Delegates take turns in a round-robin schedule. Fast finality is achieved because the delegate set is small and known.

Trade-offs:

+ Fast block times and high throughput (EOS: 0.5s blocks, 1000+ TPS).
+ Deterministic finality once a supermajority of delegates sign.
- Only 21 delegates in EOS — smaller attack surface than 500,000+ Ethereum validators.
- Delegate collusion is feasible; vote-buying has been documented on-chain.

Examples: EOS (21 block producers), Tron (27 super representatives), Lisk.

Proof of Authority

Validators are pre-approved entities whose real-world *identity* (rather than tokens or hashrate) is at stake. Suitable for permissioned or consortium chains where participants are known and legally accountable. Examples: VeChain, Ethereum testnets (Goerli used PoA before deprecation).

BFT Consensus Variants

PBFT Three-Phase Protocol

Practical Byzantine Fault Tolerance (Castro & Liskov, 1999) operates in three phases among a known validator set of size $n \geq 3f + 1$:

1. **Pre-prepare**: The *primary* (leader) broadcasts a proposal with sequence number s.
2. **Prepare**: Each replica broadcasts a "prepare" message for s; waits for $2f$ matching prepares to enter the *prepared* state.
3. **Commit**: Each prepared replica broadcasts "commit"; waits for $2f + 1$ matching commits before *executing* the request and replying to the client.

Message complexity: $O(n^2)$ per round. This limits PBFT to tens to low hundreds of validators. View-change protocol handles primary failures. Examples: Hyperledger Fabric (early versions), Tendermint.

Tendermint

Tendermint adapts PBFT for blockchain:

- **Propose** → **Prevote** → **Precommit** phases per block height and round.
- Locks: once a validator prevotes a block it cannot prevote a different block at the same height, preventing equivocation.
- Finality is *deterministic*: once $\geq \frac{2}{3}$ of voting power precommits, the block is irreversibly final.
- Safety holds even in asynchronous networks; liveness requires eventual synchrony.

Used by Cosmos Hub, Binance Chain, and many Cosmos SDK chains.

HotStuff

HotStuff (Yin et al., 2019) achieves $O(n)$ message complexity (linear) by using a *chain of three-round quorum certificates* (QC). Key innovation: a leader collects votes via a *threshold signature*, producing a single compact QC rather than n^2 messages. Used by Facebook's Diem (Libra), Aptos, and Sui (variants of HotStuff/BFT).

Selfish Mining Attack

Strategy

Described by Eyal & Sirer (2013), selfish mining is a strategy where a miner with hashrate fraction α withholds newly found blocks instead of broadcasting immediately:

1. Upon finding block B_1, keep it secret; continue mining B_2.
2. When the honest network finds a competing block B_1' (at same height), immediately release B_1.
3. If selfish miner is ahead by 2 blocks, release both; honest miners' block is orphaned.

Profitability Analysis

Let γ be the fraction of honest miners who adopt the selfish miner's block when there is a tie (representing network connectivity advantage). The selfish miner's revenue share exceeds α when:

$$\alpha > \frac{1-\gamma}{3-2\gamma}$$

For $\gamma = 0$ (worst case for attacker): profitable above $\alpha > \frac{1}{3} \approx 33\%$. For $\gamma = 1$: profitable above $\alpha > 0$. This is concerning because it shows the 51% threshold is not the only critical point.

Mitigation

- **Uniform tie-breaking**: Nodes randomly choose between competing same-height blocks (reduces γ).
- **Shorter block times**: Reduces the window during which a secret chain can be built (but increases natural fork rate).

- **GHOST rule**: Weight chain by total work, not just length; orphaned blocks count toward the "heaviest" subtree (Ethereum's approach before PoS).

MEV — Maximal Extractable Value

MEV

Maximal Extractable Value (formerly *Miner* Extractable Value) is the total value a block producer can extract beyond standard block rewards and fees by manipulating the *ordering*, *inclusion*, or *exclusion* of transactions within a block.

Extraction Methods

- **Front-running**: Observe a large pending swap in the mempool; insert an identical trade *before* it to profit from the price impact.
- **Sandwich attack**: Place a buy order *before* and a sell order *after* a victim's swap, profiting from the price move the victim's trade causes.
- **Liquidation sniping**: Monitor lending protocols for undercollateralised positions; submit liquidation transactions at the earliest profitable moment.
- **Arbitrage**: Exploit price differences across DEX pools within a single block (generally considered "good" MEV as it restores price efficiency).

Impact on Users

- Worse execution prices on DEX trades.
- Failed transactions (competing bots outbid and revert the victim's transaction).
- Higher gas prices as bots engage in *priority gas auctions* (PGA).

Flashbots and MEV-Boost

Flashbots introduced a **private transaction relay**: searchers submit transaction bundles directly to block builders, bypassing the public mempool. This reduces on-chain gas wars and toxic MEV while concentrating power in professional builders. **MEV-Boost** (Ethereum PoS) separates the *proposer* (validator who signs the block) from the *builder* (who assembles transactions), via a relay marketplace. Consequence: over 90% of Ethereum blocks are built by a small set of professional builders, raising centralisation concerns.

Finality

Types of Finality

- **Probabilistic finality** (PoW): The probability that a confirmed transaction is reversed decreases exponentially with the number of subsequent blocks. For Bitcoin, after 6 confirmations (≈ 60 min) reversal requires $> 50\%$ of hashrate for an extended period.
- **Deterministic/Absolute finality** (BFT): Once $2f + 1$ validators commit a block, no competing block can ever be committed. Reversal is cryptographically impossible (not merely expensive).
- **Economic finality** (Ethereum PoS): A block is finalised after 2 epochs (≈ 12.8 min). Reverting a finalised block would require burning $\geq \frac{1}{3}$ of all staked ETH, making reversal economically catastrophic but not physically impossible.

System	Finality Type	Time to Finality	Reversal Cost
Bitcoin (PoW)	Probabilistic	~ 60 min (6 blocks)	$> 50\%$ hashrate
Ethereum (PoS)	Economic	≈ 12.8 min	$\geq \frac{1}{3}$ staked ETH
Tendermint	Deterministic	~ 1–3 s	Requires $> \frac{1}{3}$ colluding
PBFT	Deterministic	< 1 s	Requires $> \frac{1}{3}$ colluding

Common Pitfall

Misconception: "More confirmations always mean safer."
Correction: In PoW, the *rate* of new blocks and the *honest hashrate* both matter. A transaction in a chain with low hashrate may need far more than 6 confirmations to reach the same security level as 6 Bitcoin confirmations. Always consider the cost to reorg, not just the block count.

Practice Problems

Example: Problem 1 — Difficulty Adjustment

Question: Bitcoin's last 2016 blocks were mined in 13 days instead of the target 14 days. By what factor does the difficulty change, and in which direction?

Solution:

$$\text{New Difficulty} = \text{Old Difficulty} \times \frac{13 \times 24 \times 60}{14 \times 24 \times 60} = \text{Old Difficulty} \times \frac{13}{14} \approx 0.929 \times \text{Old}$$

The target T *decreases* (difficulty *increases*) by $\approx 7.7\%$ because blocks were arriving faster than desired.

Note: The Bitcoin protocol caps the adjustment at a factor of 4 in either direction per epoch

to prevent runaway difficulty swings.

Example: Problem 2 — Byzantine Generals Bound

Question: A consortium blockchain has 10 validators. What is the maximum number of Byzantine (malicious) validators it can tolerate under classical BFT? If two additional validators are needed for a planned expansion and both may be Byzantine, what is the new minimum total validator count required?

Solution:

- Maximum tolerable faulty nodes with $n = 10$: from $n \geq 3f + 1$ we get $f \leq \frac{n-1}{3} = \frac{9}{3} = 3$.
- After expansion: total faulty nodes could be up to $3 + 2 = 5$. Required minimum total: $n \geq 3 \times 5 + 1 = 16$.

Example: Problem 3 — Selfish Mining Threshold

Question: Compute the minimum hashrate fraction α at which selfish mining becomes profitable if $\gamma = 0.5$ (the attacker wins network ties half the time).

Solution:

$$\alpha > \frac{1-\gamma}{3-2\gamma} = \frac{1-0.5}{3-2(0.5)} = \frac{0.5}{2} = 0.25$$

With $\gamma = 0.5$, the threshold is **25%**. This is significantly below the 51% threshold required for a double-spend attack, meaning selfish mining is a more accessible attack vector.

Example: Problem 4 — Sandwich Attack Profit

Question: A DEX uses the constant-product AMM formula $x \times y = k$. A pool holds 100 ETH and 200,000 USDC ($k = 2 \times 10^7$). A victim wants to buy 10 ETH. An MEV bot front-runs by buying 5 ETH first, then back-runs after the victim's trade. Compute the prices the victim receives without and with the sandwich attack (ignore fees).

Solution (simplified):

- **Without attack**: Victim buys 10 ETH from 100/200,000 pool. New ETH reserve: 90. New USDC: $k/90 \approx 222{,}222$. Victim pays $\approx 22{,}222$ USDC for 10 ETH ($\approx 2{,}222$ per ETH).
- **With sandwich**: Bot first buys 5 ETH, pool becomes 95 ETH / 210,526 USDC. Victim now buys 10 ETH from worse pool: pays $\approx 23{,}256$ USDC, i.e. about 1,034 USDC extra. Bot then sells back 5 ETH at the higher price, capturing most of that surplus.

The victim overpays due to the artificial price impact; the bot extracts the difference as MEV.

Key Takeaways

1. **Byzantine Generals Problem**: Consensus with f traitors among n nodes requires $n \geq 3f+1$. Classical BFT needs a known participant list; Bitcoin replaces this with Sybil-resistance via PoW.
2. **Proof of Work**: Converts electricity into security via a brute-force nonce search. Difficulty adjusts every 2016 blocks. Block arrivals follow a Poisson process. The 51% threshold governs double-spend attacks.
3. **Proof of Stake**: Converts locked capital into security. Slashing enforces honesty by making equivocation financially destructive. Economic finality in Ethereum requires ≈ 12.8 min. The nothing-at-stake problem is solved by slashing.
4. **PoW vs. PoS**: PoW is energy-intensive but anchored in physics; PoS is capital-efficient but relies on rational economic incentives. Neither is strictly superior; they embody different security philosophies.
5. **DPoS and BFT**: DPoS trades decentralisation for throughput via elected delegates. PBFT / Tendermint / HotStuff achieve deterministic finality with $O(n^2)$ / $O(n)$ message complexity respectively.
6. **Selfish mining**: Profitable below 51% hashrate (threshold $\approx$ 25–33%). Mitigated by uniform tie-breaking and GHOST.
7. **MEV**: Structural feature of any ordered ledger. Front-running, sandwich attacks, and liquidation sniping extract value from users. Flashbots and MEV-Boost redistribute rather than eliminate MEV, at the cost of builder centralisation.
8. **Finality spectrum**: Probabilistic (PoW) $\rightarrow$ economic (Ethereum PoS) $\rightarrow$ deterministic (PBFT/Tendermint). Faster finality generally requires stronger identity assumptions or smaller validator sets.

Further Reading

- Lamport, L., Shostak, R., & Pease, M. (1982). The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3), 382–401.
- Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. https://bitcoin.org/bitcoin.pdf
- Eyal, I., & Sirer, E. G. (2014). Majority is not Enough: Bitcoin Mining is Vulnerable. *FC 2014*. https://arxiv.org/abs/1311.0243
- Castro, M., & Liskov, B. (1999). Practical Byzantine Fault Tolerance. *OSDI '99*.
- Yin, M., Malkhi, D., Reiter, M. K., Gueta, G. G., & Abraham, I. (2019). HotStuff: BFT Consensus with Linearity and Responsiveness. *PODC '19*.

- Ethereum Foundation. *Proof-of-Stake Documentation*. https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/
- Flashbots Research. *MEV and the Ethereum Ecosystem*. https://writings.flashbots.net/