Consensus Mechanisms

A Standalone Mini-Course

BSc Blockchain Course

Why Can't Distributed Computers Just Agree?

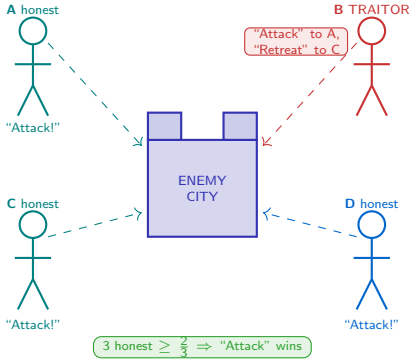
Imagine four army generals surrounding an enemy city. To win, they must all attack at the same moment – a split force fails. They communicate only by messenger. One general is a traitor who sends different orders to different allies.

The formal question: Can loyal generals agree on a common plan despite an unknown number of traitors?

Lamport, Shostak & Pease (1982) proved:

- Agreement holds if at most $\frac{1}{3}$ of generals are traitors
- $n \geq 3f + 1$ total nodes needed to tolerate f faults
- Signed messages allow $n \geq 2f + 1$

Why blockchains care: every node is a general. The "city" is the next valid block. Traitors are malicious participants. The protocol must produce the same decision at every honest node – without any trusted coordinator.



The Byzantine Generals Problem is the exact problem every blockchain node faces when deciding which block to add next. The $\frac{2}{3}$ honest-majority threshold reappears unchanged in Casper FFG, Tendermint, HotStuff, and PBFT.

Source: Lamport, Shostak & Pease (1982). "The Byzantine Generals Problem." ACM TOPLAS 4(3), 382–401. Fischer, Lynch & Paterson (1985). ACM JACM 32(2).

When Consensus Fails: The Stakes Are Real

Consensus failures are not theoretical. Whenever a blockchain's hashrate or stake is sufficiently concentrated, an attacker can rewrite recent history – double-spending coins that were already accepted as final.

In January 2020, Ethereum Classic suffered three 51% attacks in four days. Blocks were reorganised. Transactions were reversed. An exchange absorbed millions in losses – not because of a bug, but because the chain was small enough that renting majority hashrate cost less than the profit from double-spending.

The same pattern repeated against Bitcoin Gold and Verge in 2018. The attacker's tool in every case: a rental market for hashing power.

Reflect before we go further

Open your phone. Find a crypto wallet or exchange app.

- How many confirmations does it require before marking a transaction "final"?
- What would happen to your balance if those blocks were reorganised?
- Which chains you hold are large enough that a 51% attack would cost more than it profits?

Bring your answers to class. We will use them as a running example throughout.

Source: Saad et al. (2021). "Exploring the Attack Surface of Blockchain." IEEE Communications Surveys. crypto51.app historical attack cost data.

Proof of Work: Converting Energy to Security?

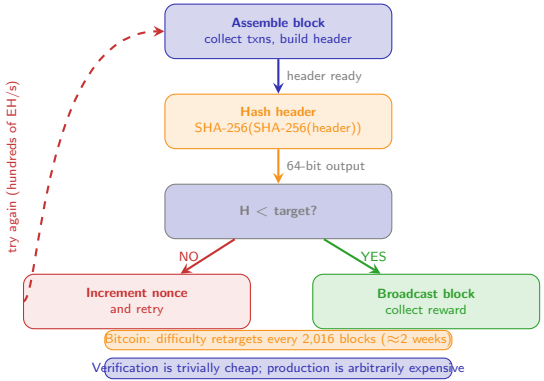
The PoW loop runs on every miner: Step 1 – Assemble candidate block.
Collect pending transactions, build a block header containing the previous block hash, Merkle root, timestamp, and a nonce at zero.

Step 2 – Hash the header.
Compute $H = \text{SHA-256(SHA-256(header))}$. This takes microseconds.

Step 3 – Check against target.
If $H <$ target, the block is valid. Otherwise, increment nonce and repeat. Bitcoin miners collectively perform hundreds of exahashes per second.

Step 4 – Broadcast and earn.
The winning miner broadcasts the block. Peers verify in milliseconds. Miner earns the block reward plus transaction fees.

Energy expenditure is what makes history immutable: rewriting a block requires redoing all the PoW since that block.



The elegance of PoW: verification costs one hash, but block production costs as much as the honest network spends. This asymmetry is the foundation of Nakamoto consensus – honest miners always outrun an attacker who must redo all the work.

Source: Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System." bitcoin.org/bitcoin.pdf. Antonopoulos, A.M. (2017). "Mastering Bitcoin," 2nd ed. O'Reilly.

Proof of Stake: Converting Capital to Security?

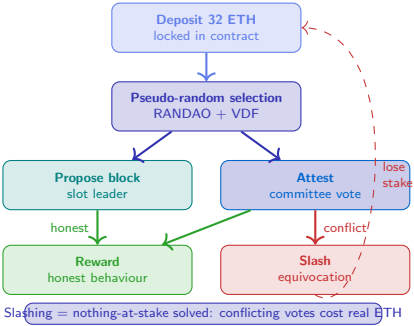
PoS replaces energy expenditure with locked capital: Validator entry: Deposit 32 ETH into the staking contract. This ETH is locked and subject to slashing for misbehaviour. Creating a second identity costs another 32 ETH – this is the Sybil-resistance mechanism.

Block proposal: Each 12-second slot, one validator is pseudo-randomly selected proportional to effective balance. That validator proposes a block.

Attestation: A committee of validators attests to each proposal with a signed vote. Aggregated signatures reduce bandwidth.

Slashing (the key innovation): If a validator signs two conflicting blocks for the same slot (*equivocation*), the protocol slashes part of their stake and ejects them from the validator set.

Slashing solves nothing-at-stake: there is now something at stake.



Economic security in PoS is measurable: the cost to corrupt consensus equals the cost to acquire a third of all staked ETH and accept the slash. Unlike mining equipment, slashed ETH cannot be resold – the attack is unrecoverable.

Source: Buterin et al. (2020). "Combining GHOST and Casper." arXiv:2003.03052. Ethereum Foundation (2023). ethereum.org/en/developers/docs/consensus-mechanisms/pos

BFT and Finality: When Is a Transaction Really Final?

Three types of finality – fundamentally different guarantees:

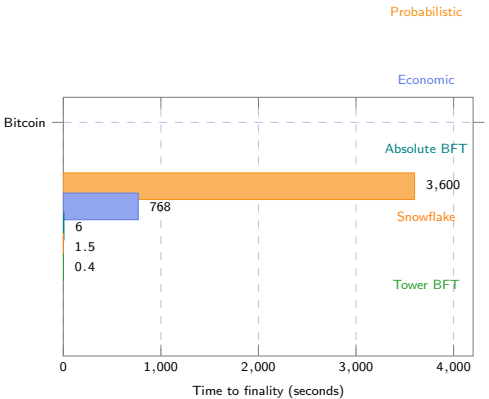
Probabilistic finality (PoW):
A block at depth k is reverted with probability that shrinks with each additional confirmation. Six confirmations ($\approx$60 min) are conventional for large Bitcoin transfers – but certainty is never mathematical.

Economic finality (Casper FFG):
Once a checkpoint is finalised, reverting it requires slashing a third of all staked ETH. The finality is not mathematical but economically irrational to violate. Reached roughly every 13 minutes on Ethereum.

Absolute / BFT finality (Tendermint):
Once two-thirds of validators have signed a commit, the block is final immediately and irreversibly – no fork is possible unless more than a third of nodes collude. Cosmos achieves this in seconds.

The tradeoff: BFT finality requires synchrony and known validator sets. PoW finality works without either.

Time-to-finality across networks:



Approximate values; Bitcoin = 6-confirmation convention.

For payments and DeFi, finality time directly translates to user waiting time and settlement risk. A 60-minute PoW finality is acceptable for large store-of-value transfers but impractical for decentralised exchange trading. Mechanism choice is application-driven.

Source: Buterin & Griffith (2019). arXiv:1710.09437; Kwon, J. (2014). "Tendermint: Consensus without Mining." tendermint.com; Rocket et al. (2020).

What Happens When an Attacker Buys the Majority?

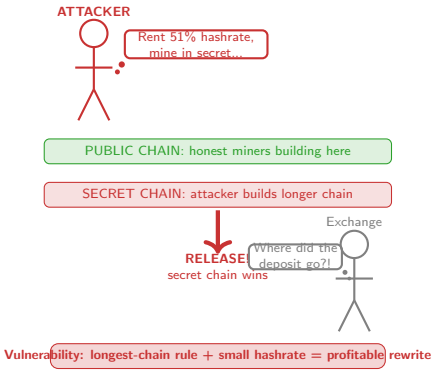
The 51% attack – step by step: **Step 1 – Mine in secret.** Attacker rents majority hashrate from a GPU rental market, then mines a private chain while spending coins on the public chain. The exchange sees a valid deposit.

Step 2 – Collect the goods. Attacker withdraws or sells whatever was purchased with the double-spent coins, before the reorganisation is detected.

Step 3 – Release the secret chain. Once the private chain is longer than the public chain, the attacker broadcasts it. The network's longest-chain rule forces every honest node to adopt it.

Step 4 – History is rewritten. The original deposit transaction is erased. The attacker keeps both the goods and the coins.

Attack cost scales with network size. Small chains can be attacked for a few thousand dollars per hour.



No consensus mechanism is attack-free – each makes a deliberate choice about which attacks it prices out of reach. Understanding the attack surface is prerequisite to evaluating any blockchain system's real security.

Source: crypto51.app; Bonneau, J. (2018). "Why Buy When You Can Rent?" FC 2016. Saad et al. (2021). IEEE Communications Surveys & Tutorials.

The Consensus Landscape: Finality vs. Throughput?

Six major networks plotted by finality speed and transaction throughput:

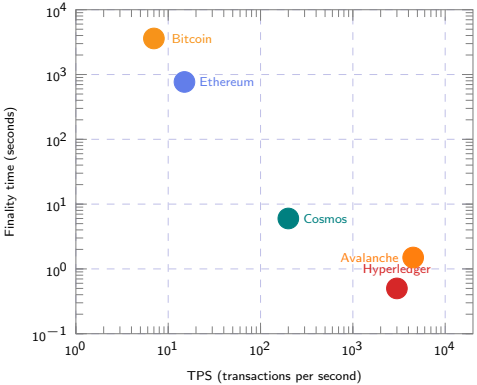
Placing networks on a log-log chart reveals two clusters:

Store-of-value quadrant (top-left):
High finality time, low throughput. Bitcoin and Ethereum (pre-Merge era) optimise for security and decentralisation. Finality time is measured in minutes or hours. Throughput is counted in tens of transactions per second.

High-performance quadrant (bottom-right):
Low finality time, high throughput. Solana and Avalanche achieve sub-second finality and thousands of transactions per second – at the cost of smaller, more specialised validator sets.

BFT middle ground:
Cosmos and Ethereum (post-Merge) sit between the two clusters – combining moderate throughput with economic or absolute finality in seconds rather than hours.

No chain occupies the bottom-left (fast finality, high throughput, maximum decentralisation) – this is the trilemma made visible.



Approximate published values. Axes are log-scale.

Every chain's position on this chart reflects a deliberate engineering tradeoff. Moving toward the high-performance corner requires either trusting fewer validators or accepting weaker decentralisation – there is no free lunch.

Source: Published network documentation and academic surveys. Values are approximate and vary with network conditions. Axes are log-scale.

The Ethereum Merge: Who Won and Who Lost When a Network Changed Its Engine?

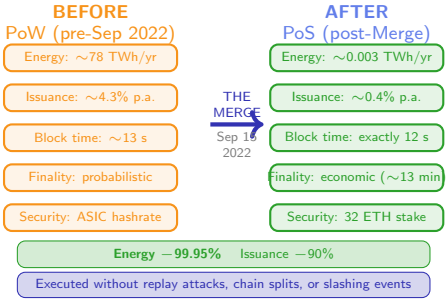
On 15 September 2022, Ethereum switched from Proof of Work to Proof of Stake in a live network upgrade: **The Merge**. The same system looked very different depending on where you stood.

Winners:

+ **ETH stakers** – became the new validators overnight; earn fees previously going to miners
+ **ESG-sensitive institutions** – energy use fell by roughly 99.95%; barriers to holding ETH dropped
+ **Long-term ETH holders** – issuance fell sharply; supply became net deflationary under load

Losers:

– **GPU miners** – hardware became worthless for Ethereum overnight; redirected to EthereumPoW fork or other coins
– **Energy-sector suppliers** – data centres and electricity contracts backing Ethereum mining were stranded
– **Small solo stakers** – 32 ETH minimum is a high capital barrier; access shifted toward staking pools



The Merge proved that governance – not cryptography – is the hardest problem in upgrading a live blockchain. The same design choice that reduced energy use by nearly all of it also rendered billions in mining hardware worthless overnight.

Source: Ethereum Foundation (2022). "The Merge." blog.ethereum.org; Digiconomist (2022). Ethereum Energy Consumption Index; ethereum.org/en/roadmap/merge

Five Questions That Determine Your Optimal Consensus Mechanism

Apply these five questions in order:

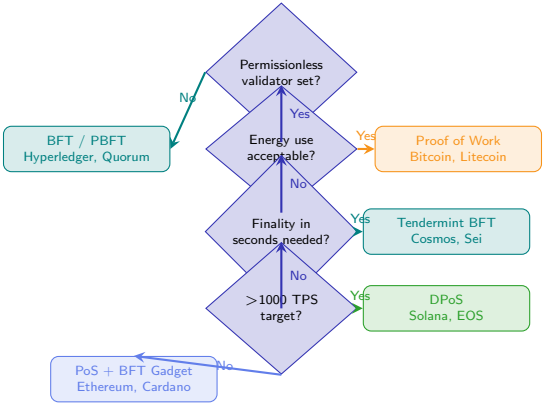
Q1: Permissionless or permissioned?
Open validator set ⇒ PoW or PoS. Known, credentialled validators ⇒ BFT family.

Q2: Energy use acceptable?
If energy expenditure is unacceptable ⇒ PoS or BFT. If physical unforgeability matters ⇒ PoW.

Q3: Finality in seconds required?
Seconds ⇒ BFT (Tendermint, PBFT). Minutes ⇒ PoS with finality gadget. Hours acceptable ⇒ PoW.

Q4: Target throughput above 1,000 TPS?
Yes ⇒ DPoS or BFT with small committee. No ⇒ more decentralisation feasible.

Q5: Stake distribution wide enough?
Concentrated stake ⇒ cartel risk in PoS. Wide distribution ⇒ PoS security holds.



The decision tree collapses a complex design space into a navigable sequence. In practice, hybrid approaches exist: Ethereum combines PoS selection with a BFT finality gadget. The tree is a starting point, not a substitute for reading the mechanism's security proofs.

Source: Garay, Kiayias & Leonardos (2015). "The Bitcoin Backbone Protocol." Eurocrypt; Buterin (2022). "PoS FAQ." ethereum.org; Kwon & Buchman (2019). Cosmos Whitepaper.

Your Challenge

A new fintech company wants to build a real-time payment settlement layer. Their requirements: settle cross-border transfers between regulated banks in under three seconds, process up to 5,000 payments per second at peak, keep validator identities auditable for compliance purposes, and minimise energy use to satisfy their ESG policy.

Apply the Five Questions to This Case

Work through the decision tree from slide 9. For each question, state your answer and the evidence from the scenario that justifies it.

Question	Answer	Justification from scenario
Q1: Permissionless or permissioned?		
Q2: Energy use acceptable (PoW)?		
Q3: Finality in seconds required?		
Q4: Target above 1,000 TPS?		
Q5: Stake distribution wide enough?		
Recommended mechanism:		

Discuss with your neighbour: where do your recommended mechanisms disagree, and which single requirement drives the biggest constraint?

Bridge to L05 – Ethereum Architecture: how does the execution layer (EVM) interact with the consensus layer you just mapped? Why did The Merge leave the EVM entirely unchanged?