**Solana:**

Overview:

Solana (SOL) aims to be a competitor to Ethereum, boasting improved scalability (number of transactions per second) and significantly lower transaction fees (regardless of network size).

SOL relies on a **proof of stake** (PoS) architecture (as opposed to Ethereum which is Proof of Work), in addition to a new concept of **proof of history** (PoH) explained below.

**Proof of Stake (PoS) vs Proof of Work (PoW):**

Traditionally, cryptocurrencies like BTC and ETH utilise a PoW model to verify transactions – this involves miners **competing** to find the correct nonce value that gives the block hash the required number of leading 0s. This consumes large amounts of computational time/power and uses a significant amount of energy. This ‘work’ is rewarded with mining fees for verifying blocks.

Proof of stake does not involve a competition process to verify transactions, instead, **stakeholders are able to verify transactions immediately**. To be a stakeholder you must hold a significant amount of the token, and the proportion of transactions you are allowed to verify depends on your staked amount. Verifying transactions earns the stakeholder a ‘commission’ reward, equivalent to the mining fee in the PoW model.

* Consumes significantly less energy – stakeholders only need to compute a single hash in order to verify a transaction, compared to the huge number required in a PoW model.
* A stakeholder would need to hold over 51% of all tokens in order to mount an attack and verify fraudulent transactions.

**Synchronicity and Proof of History:**

PoW and Block Time:

PoW models have a concept of ‘block time’ – the average time it takes for new blocks to be verified and added to the blockchain (mining difficulty is adjusted based on the network hash power to keep this block time roughly constant). The block time must be large enough to minimise the odds of two miners verifying and attempting to add the same block to the chain. **This significantly limits the throughput of PoW systems**.

For example, BTC has a block time of ~10 minutes, and a transaction needs on average at least two ‘confirmations’ (transaction present in two blocks added to the chain) which leads to an average transaction speed of ~20 minutes. ETH by contrast has a block time of 14-15s, significantly less than BTC, but still yielding transaction times in the order of minutes.

PoS and block timestamps:

PoS models do not have this constraint, since the verification process for new blocks happens (near) instantaneously and only ever by a single verifier (no chance of two miners verifying the same block). However, this relies on a degree of synchronicity between blocks, since the verifier needs to know the order in which to process incoming blocks.

This is solved by including a timestamp within each block in order to keep track of the block order. However, due to several factors such as network latency and clock drift this timestamp is never 100% reliable (accuracy ± 1-2 hours), and therefore the block time must be lengthened accordingly to ensure that the median value of block timestamps is increasing (standard is that the block timestamp must exceed the median time of the previous 11 blocks to be verified). Thus, PoS does not yield any significant throughput advantages over PoW.

Proof of History (PoH):

PoH is **not** a consensus method (like PoS and PoW) but rather a method to significantly decrease the time required to achieve PoS consensus. This is achieved by using a Verifiable Delay Function (VDF) in order to timestamp and synchronise the order of processed blocks, which removes the clock time limitations discussed previously.

Verifiable Delay Functions (VDF): [601.pdf (iacr.org)](https://eprint.iacr.org/2018/601.pdf) A VDF is a cryptographic technique used to prove sequential work, the simplest example being a hash chain. Each hash in the chain requires a certain amount of computational time, and the hash value depends on the previous hash value. Hence, successive hashes in the chain must have been computed in sequential order with an (approximate) amount of time between each hash. A VDF may therefore be understood as ‘proof of sequential work’.

**Solana:**

Solana utilises this concept to verify the order of processed blocks, removing the need for block timestamps that rely on clock time. Each block contains a hash chain of ‘timestamps’, that are created whenever a node verifies a block. These ‘timestamps’ can be used to verify that all of the data within the block must have been generated before this block was timestamped, and after its previous timestamp. Therefore, blocks can arrive to validating nodes in any order without compromising the verification, which eliminates the need to ensure blocks are received in the correct time order (by increasing the block time).

Rather than dealing with entire blocks, transactions are bundled into smaller packages called **entries**. Each entry is timestamped and sent to validator nodes for validation. These nodes are able to determine the order at which a large number of entries should be processed, in near real time.

This high degree of synchronicity allows transactions to be confirmed at a significantly greater rate than PoS or PoW alone, allowing Solana to confirm transactions in **800ms** (current implementation with 150 nodes – see below).

**More detail:**

The Solana network is split into independent **clusters,** that each contain validation nodes. Each cluster works **independently** from the others and are used for various purposes (each cluster is therefore effectively a separate blockchain). In what follows we will consider a single cluster of nodes as each cluster is independent.

Within a cluster, nodes are separated into Leader (one – rotating) and Verifier roles. The leader is responsible for processing incoming transactions and bundling them into timestamped ‘entries’, where the timestamp determines that this entry was **produced after the previous entry**.

These entries are then distributed to the validator nodes that are responsible for

1. Verifying (hashing) the entry contents – this is then compared with the network to achieve consensus (referred to as ‘voting’ – where 2/3 majority is required).
2. Verifying the order of entries, such that verified entries are added to the blockchain in sequence.

The key point about this architecture is that it allows each node to verify (hash) many entries before needing to vote on the authenticity of the next sequential entry. This removes the need for the network to process every entry in the order it was created, since the order of entry creation can be accurately determined from the VDF timestamps.

This has several advantages. It allows the entries to be sent to validator nodes as quickly as they can be processed by the leader node, regardless of whether or not the current entry is the next sequential entry to be processed.

**Entry validation process summary:**

Leader:

* Leader bundles transactions into an entry, with a timestamp determining that the current entry was produced **after** the previous entry.
* These entries are distributed to nodes, who may receive entries (due to latency) in a different order than they were sent by the leader.

Validators:

* The node hashes each entry it receives (**regardless of the order**).
* The node keeps track of the last entry that was **verified to be in sequence**.
* When the node receives the **next** **sequential** **entry** (determined by timestamp) it ‘votes’ on the authenticity of that entry (compares its hash value with the network to attempt consensus).
* Once the network achieves consensus, the entry is added to the blockchain and the network waits until consensus is achieved on the **next sequential entry**.

This implementations allows nodes to verify entries before needing to vote on the authenticity of the entry. Authenticity voting is therefore able to proceed the instant that a node receives and hashes the next sequential block (which may have already been hashed if the entry was received before a previous entry, due to network latency).

**PoW comparison:**

It is helpful to compare the Solana approach with the PoW model. In this model, new transactions are added to larger ‘blocks’. When a new block is created, all nodes in the network compete to ‘mine’ that single block. Once a single miner has found a nonce value that verifies that block, all other miners in the network proceed to verify that block with the found nonce value.

Effectively, the computation time and power expended by the rest of the network is ‘lost’, as they wait for a single miner to verify the next block. Only once a single miner has verified the block can the rest catch up and verify that same block. Furthermore, the time it takes for a single miner to mine a block must be large enough such that the probability of two miners simultaneously verifying the same block (with different nonce values) must be significantly low. This long ‘block time’ combined with the ‘wasted’ computational power expended by the network makes PoS with PoH synchronisation significantly more favourable.

In the PoS/PoH model, significantly less computation power is wasted since all nodes only hash each entry once, and only as many hashes that are needed to achieve consensus (2/3 majority) are ever computed (the remaining 1/3 of nodes that hashed that particular entry would thus be the only ‘wastage’). Furthermore, since PoH allows the network to accurately determine the order in which entries should be processed, there is no need for any additional ‘block time’ to ensure the sequence of blocks is preserved. Entries can simply be processed as quickly as they can be sent between nodes – leading to significant decreases in the time required to confirm entries (sub second).

**Sources:**

Original Solana paper: [Solana: A new architecture for a high performance blockchain](https://solana.com/solana-whitepaper.pdf)

Solana documentation (Architecture : Cluster section): [A Solana Cluster | Solana Docs](https://docs.solana.com/cluster/overview)

Verifiable delay functions : [601.pdf (iacr.org)](https://eprint.iacr.org/2018/601.pdf)