Summary on scaling blockchain

1. Sharding

Sharding is a technique designed to enhance blockchain scalability by dividing the network into smaller partitions called shards. Each shard is responsible for processing a portion of the overall data and transactions, allowing multiple transactions to be executed in parallel. This significantly increases throughput and lowers latency compared to traditional monolithic blockchains.

The process starts with **node selection**, where a subset of nodes is chosen from the entire pool of participants. This step must ensure fairness and robustness. In permissioned blockchains, a trusted authority can handle this selection. However, in permissionless systems, it is crucial that node selection be random, unpredictable, unbiased, and publicly verifiable to prevent any adversary from manipulating shard composition.

Epoch randomness is then generated to serve as a secure and unpredictable input for assigning nodes to shards. Each honest node must independently reach the same randomness value through interaction with others. The randomness must be publicly verifiable, resistant to bias, and available across the network. Interestingly, although randomness is unpredictable before it is generated, once it is revealed, it can be used as a challenge in proof-of-work mechanisms without compromising security.

Using this randomness, **node assignment** divides the selected nodes into `m` groups, each to be allocated to a different shard. This step is performed randomly to avoid centralization and ensure that no single entity can dominate a shard.

Once shards are formed, each shard must internally reach consensus on transactions or proposals through an **intra-shard consensus (ISC)** mechanism. While traditional blockchains use consensus to validate all transactions globally, ISC in sharded systems operates within the shard. It must ensure both consistency (correctness of the shared ledger) and liveness (every valid proposal is processed within a bounded time). ISC can be implemented using strong consistency—where a committee collectively produces blocks—or weak consistency—where a single node acts as the block producer.

Because many transactions span multiple shards, **cross-shard transaction processing (CST)** is a vital component. A CST system takes in a batch of transactions, invokes ISC on the relevant shards, and outputs corresponding logs. The key challenges include enabling secure communication across shards and preventing double spending. Each involved shard must verify that none of the inputs have already been spent before committing the transaction.

As the system evolves, **shard reconfiguration (SR)** updates shard memberships from one epoch to the next. It ensures new nodes are integrated, outdated nodes are replaced, and shard integrity is maintained. A critical requirement is that each shard remains

honest—meaning the proportion of honest nodes must remain above a certain threshold to guarantee security and resilience.

Finally, a **motivation mechanism** is necessary to incentivize nodes to participate honestly in consensus, transaction validation, and reconfiguration. Without well-designed economic rewards and penalties, the network could become vulnerable to selfish behavior or collusion.

2. Sidechains

Sidechains (or Bitcoin-backed altcoins) are separate blockchains designed to extend Bitcoin's features without changing its core protocol. They let users move assets between chains using a **two-way peg**, keeping value parity. New altcoins can be distributed either by **burning bitcoins** or by allocating them based on existing Bitcoin holdings.

Embedding all sidechain rules into Bitcoin would make its scripts too complex and burden Bitcoin nodes. To solve this, sidechains use **SPV (Simplified Payment Verification) proofs**. These allow Bitcoin to verify key sidechain transactions without processing all their data. An SPV proof, included in the Bitcoin transaction, shows that a sidechain transaction is valid and buried under enough proof-of-work.

To reduce costs, sidechains often **sample** only blocks with stricter hash targets. Though faster, this increases fraud risk. A **challenge period** lets anyone contest fake proofs using a stronger chain, balancing efficiency with Bitcoin's base-layer security.

Bitcoin's on-chain throughput is limited by its block size. To match Visa-level volumes (≈47,000 tps) would require blocks nearing 8 GB, generating over 400 TB of data per year—an untenable burden that would centralize the network around only the most powerful nodes.

3. Lightning payment channel

The Lightning solution sidesteps this by opening **off-chain payment channels** between pairs of users. Each channel begins with a funding transaction that locks funds into a 2-of-2 multisignature address on Bitcoin's blockchain. Alice and Bob then exchange signed, unbroadcast transactions reflecting incremental balance updates. These "commitment transactions" are never published unless there's a dispute, so hundreds or thousands of micro-payments occur off-chain without burdening the main network.

To prevent cheating, Lightning uses **revocable commitments**. Every time the balance updates, the old commitment is invalidated by sharing a revocation secret. Should one party maliciously broadcast an outdated state, the counterparty can use that secret to claim all the funds—creating a strong financial deterrent against dishonesty.

Scaling beyond one-to-one channels requires **Hashed Timelock Contracts (HTLCs)**. An HTLC conditions payment on revealing a preimage `R` to a known hash `H` within a deadline. If the receiver produces `R` in time, they claim the funds; if not, the sender reclaims them. HTLCs link channels into a **network**, allowing Alice to pay Carol via Bob without any single intermediary needing to be trusted.

In practice, Lightning forms a graph of channels where payments route automatically along paths that satisfy both hashlock and timelock constraints. Honest participants cooperate to forward payments and earn small fees, while disputes and final settlements fall back to on-chain Bitcoin transactions. This architecture achieves near-infinite off-chain throughput, minimal blockchain load, and preserves Bitcoin's decentralization by keeping only channel opening and closing on-chain.