Types of Ethereum Clients

Ethereum uses a modular architecture where a full node requires two separate, but interconnected, types of clients: an execution client and a consensus client. They communicate using the Engine API to jointly run the network.

Execution Clients

The execution client (EL) is responsible for the interaction and execution of transactions and smart contracts on the Ethereum Virtual Machine (EVM).

- Key Roles:
    - Process Transactions: Validates incoming transactions, checks signatures and account balances, and processes them.
    - Run the EVM: Interprets and executes smart contract code.
    - Manage State: Stores and updates the current state of the blockchain, including account balances and contract storage.
    - Provide API: Exposes a JSON-RPC API, which is used by wallets, dApps, and block explorers to interact with the network.
- Popular Examples: Geth (Go Ethereum), Nethermind, Besu, Erigon, and Reth.

Consensus Clients

The consensus client (CL), also known as the beacon node client, is responsible for managing the Proof-of-Stake (PoS) consensus protocol and coordinating the network.

- Key Roles:
    - Manage Consensus: Ensures all nodes agree on the correct order and validity of blocks using the PoS algorithm.
    - Handle Validator Duties: Manages validator activities such as proposing new blocks, attesting (voting) on blocks, and applying penalties (slashing) for misbehavior.

    - Maintain the Beacon Chain: Is the primary link to the Beacon Chain, which coordinates the network and manages staked ETH.
    - Ensure Finality: Helps finalize blocks, making them a permanent part of the blockchain.
- Popular Examples: Lighthouse, Prysm, Teku, Nimbus, and Lodestar.

How They Work Together

The consensus client acts like the "brain" of the operation, while the execution client is the "engine".

1. The execution client listens for and executes user transactions.
2. The consensus client manages the process of creating and validating blocks, coordinating with the execution client via the Engine API to include the executed transaction data in a new block.
3. Once the consensus client finalizes a block, it informs the execution client, which then updates its copy of the blockchain's state.