

[Hyperledger Besu](#) is an Ethereum client designed to be enterprise-friendly for both public and private permissioned network use cases, with an extractable EVM implementation. It can also be run on test networks such as Sepolia and Görli. Hyperledger Besu includes several consensus algorithms including Proof of Stake, Proof of Work, and Proof of Authority (IBFT 2.0, QBFT, and Clique). Its comprehensive permissioning schemes are designed specifically for use in a consortium environment.

Key Characteristics

What is Hyperledger Besu?

Hyperledger Besu is an open source Ethereum client developed under the Apache 2.0 license and written in Java. It can be run on the Ethereum public network or on private permissioned networks, as well as test networks such as Rinkeby, Ropsten, and Görli. Hyperledger Besu includes several consensus algorithms including PoS, PoW, PoA, and IBFT, and has comprehensive permissioning schemes designed specifically for uses in a consortium environment.

What are Hyperledger Besu's Features?

Hyperledger Besu's features include:

-
- **The Ethereum Virtual Machine (EVM)** : The EVM is the Turing complete virtual machine that allows the deployment and execution of smart contracts via transactions within an Ethereum blockchain. It is used in conjunction with the rest of the Besu client or as a standalone library.
- **Consensus Algorithms:** Hyperledger Besu implements various consensus algorithms which are involved in transaction validation, block validation, and block production (i.e., mining in Proof of Work). They include:
 - **Proof of Stake:** Alongside a consensus client, Besu can be used to connect to and participate in Ethereum Mainnet proof-of-stake.
 - **Proof of Authority:** Hyperledger Besu implements several Proof of Authority protocols. Proof of Authority consensus protocols are used when participants are known to each other and there is a level of trust between them—in a permissioned consortium network, for example.
 - **QBFT:** QBFT is the recommended enterprise-grade consensus protocol for private networks. In QBFT networks, approved accounts, known as validators, validate transactions and blocks. Validators take turns to create the next block. Before inserting the block onto the chain, a super-majority (greater than or equal to $\frac{2}{3}$) of validators must first sign the block.
 - **IBFT 2.0:** In IBFT 2.0 networks, transactions and blocks are validated by approved accounts, known as validators. Validators take turns creating the next block. Existing validators propose and vote to add or remove validators. IBFT 2.0 has immediate finality. When using IBFT

2.0, there are no forks and all valid blocks are included in the main chain.

- *Clique*: Clique is more fault-tolerant than IBFT 2.0. Clique tolerates up to half of the validators failing. IBFT 2.0 networks require greater than or equal to $\frac{2}{3}$ of validators to be operating to create blocks. Clique does not have immediate finality. Implementations using Clique must be aware of forks and chain reorganizations occurring.
 - **Proof of Work (Ethash)**: Proof of Work is used for mining activities on Ethereum Classic.
- **Storage**: Hyperledger Besu uses a RocksDB key-value database to persist chain data locally. This data is divided into a few sub-categories:
 - **Blockchain**: Blockchain data is composed of block headers that form the “chain” of data that is used to cryptographically verify blockchain state; block bodies that contain the list of ordered transactions included in each block; and transaction receipts that contain metadata related to transaction execution including transaction logs.
 - **World State**: Every block header references a world state via a stateRoot hash. The world state is a mapping from addresses to accounts. Externally owned accounts contain an ether balance, while smart contract accounts additionally contain executable code and storage.
 - **Bonsai**: Bonsai storage is a novel paradigm for storing Ethereum state, built specifically for keeping Ethereum Mainnet storage requirements low. Bonsai organizes states into a new structure, to provide implicit pruning and low requirements. More details [here](#).
 - **Forest**: A traditional trie structure, suitable for private networks and archival nodes.
- **P2P networking**: Hyperledger Besu implements Ethereum’s devp2p network protocols for inter-client communication and an additional sub-protocol for IBFT2:
 - **Discovery**: A UDP-based protocol for finding peers on the network
 - **RLPx**: A TCP-based protocol for communication between peers via various “sub-protocols”:
 - ETH Sub-protocol (Ethereum Wire Protocol): Used to synchronize blockchain state across the network and propagate new transactions.
 - IBF Sub-protocol: Used by IBFT2 consensus protocol to facilitate consensus decisions.
- **User-facing APIs**: Hyperledger Besu provides mainnet Ethereum and EEA JSON-RPC APIs over HTTP and WebSocket protocols as well as a GraphQL API.
 - **JSON-RPC**
 - HTTP JSON-RPC Service
 - WebSocket JSON-RPC Service
 - **GraphQL**
- **Monitoring**: Hyperledger Besu allows you to monitor node and network performance.
 - Node performance is monitored using Prometheus or the debug_metrics JSON-RPC API method. It can be visually monitored through [Grafana](#).

- Network Performance is monitored with Grafana or the [Quorum Explorer](#) (part of the [Developer Quickstart](#)).
- **Privacy:** Privacy in Hyperledger Besu refers to the ability to keep transactions private between the involved parties. Other parties cannot access the transaction content, sending party, or list of participating parties. Besu uses a Private Transaction Manager to implement privacy.
- **Permissioning:** A permissioned network allows only specified nodes and accounts to participate by enabling node permissioning and/or account permissioning on the network.

Hyperledger Besu implements the [Enterprise Ethereum Alliance](#) (EEA) specification. The EEA specification was established to create common interfaces amongst the various open and closed source projects within Ethereum, to ensure users do not have vendor lock-in, and to create standard interfaces for teams building applications. Besu implements enterprise features in alignment with the EEA client specification