IBC-Go Documentation

Welcome to the documentation for IBC-Go, the Golang implementation of the Inter-Blockchain Communication Protocol! Looking for information on ibc-rs?Click here to go to the ibc-rs github repo.

The Inter-Blockchain Communication Protocol (IBC) is an end-to-end, connection-oriented, stateful protocol for reliable, ordered, and authenticated communication between heterogeneous blockchains arranged in an unknown and dynamic topology.

IBC is a protocol that allows blockchains to talk to each other. Chains that speak IBC can share any type of data as long as it's encoded in bytes, enabling the industry's most feature-rich cross-chain interactions. IBC is secure and permissionless.

The protocol realizes this interoperability by specifying a set of data structures, abstractions, and semantics that can be implemented by any distributed ledger that satisfies a small set of requirements.

IBC can be used to build a wide range of cross-chain applications that include token transfers, atomic swaps, multi-chain smart contracts (with or without mutually comprehensible VMs), cross-chain account control, and data and code sharding of various kinds.

High-level overview of IBC

The following diagram shows how IBC works at a high level:

The transport layer (TAO) provides the necessary infrastructure to establish secure connections and authenticate data packets between chains. The application layer builds on top of the transport layer and defines exactly how data packets should be packaged and interpreted by the sending and receiving chains.

IBC provides a reliable, permissionless, and generic base layer (allowing for the secure relaying of data packets), while allowing for composability and modularity with separation of concerns by moving application designs (interpreting and acting upon the packet data) to a higher-level layer. This separation is reflected in the categories:

- IBC/TAO
- comprises the Transport, Authentication, and Ordering of packets, i.e. the infrastructure layer.
- IBC/APP
- consists of the application handlers for the data packets being passed over the transport layer. These include but are not limited to fungible token transfers (ICS-20), NFT transfers (ICS-721), and interchain accounts (ICS-27).
- Application module:
- groups any application, middleware or smart contract that may wrap downsteam application handlers to provide enhanced functionality.

Note three crucial elements in the diagram:

- The chains depend on relayers to communicate. Relayers are the "physical" connection layer of IBC: off-chain processes responsible for relaying data between two chains running the IBC protocol by scanning the state of each chain, constructing appropriate datagrams, and executing them on the opposite chain as is allowed by the protocol.
- Many relayers can serve one or more channels to send messages between the chains.
- Each side of the connection uses the light client of the other chain to quickly verify incoming messages.

Next Overview