HYPERLEDGER FABRIC

Hyperledger Fabric (Androulaki *et al.* 2018) is an enterprise-grade permissioned blockchain framework with an extensible and modular architecture. It is an open-source platform hosted by the Linux foundation. It is the first to support smart contracts to be written in a general-purpose programming language such as Node.js, Go, and Java contrasted to other platforms which only have custom or domain-specific language. The permissioned nature is realized through a membership service provider which offers cryptographic identities to the network participants for authentication and authorization. Being a consortium blockchain platform the participants are known in advance but still untrusted (say, contenders in an industry). Moreover, there is no built-in cryptocurrency making it suitable for a wide range of business applications. With no cryptocurrency, the attack surface is reduced and no additional cost is incurred for mining operations. It offers a pluggable consensus mechanism (such as a byzantine fault-tolerant or crash fault-tolerant mechanism) depending on the application and trust model. Apart from the industry-level identity management system provided, it offers various privacy and confidentiality features through advanced cryptographic primitives. The Fabric has a maximum throughput of 20,000 transactions per second with a latency of 12.36 ms scaling over 100 peers (Gorenflo *et al*. 2020).

The modular architecture of the Fabric network makes it an extensible framework as it adapts to the changing needs and enhancements. This stands in contrast to an integrated architecture where a small change affects the whole system owing to the absence of modules. As Hyperledger Fabric is a modular architecture, it is necessary to understand the several components of the network and the communication between the nodes in the network. The fundamental conception of the Fabric architecture makes it easier to approach the existing shortcomings of the Fabric design and propose viable solutions. The network architecture, unique transaction flow, and privacy features offered in Hyperledger Fabric are discussed in the subsequent section.

1.2.1 Hyperledger Fabric Network Architecture

The core components of a Hyperledger Fabric network are a ledger, smart contracts, peer nodes, orderer, channel, and certificate authority.

A *ledger* stores data as key-value pairs which are updated or read by a transaction. The ledger consists of two unique yet associated elements: Blockchain and world state. The world state contains only the latest updated key-value pair while the blockchain holds the entire history of values up to the current state.

A *smart contract* or a chaincode is the application logic to be executed in order to update a key-value pair.

Peers in Hyperledger Fabric can be endorser peers or committing peers. Endorser peers are the only authorized entity in the network to execute the chaincode. Committing peers on the other hand only verify the transactions within a block and are responsible for committing them onto the blockchain. This contrasts with other blockchain platforms where all the nodes execute the smart contract. The advantages of executing the transactions only by a subset of nodes are twofold: 1. Privacy of execution is guaranteed by allowing only certain peers as dictated by the endorsement policy to have access to the chaincode and transaction data, 2. Improved performance by not wanting all the peers to execute every chaincode.

Orderer as the name suggests orders the incoming transactions based on the timestamp and packs them into a block. They run a pluggable consensus algorithm depending on the use case.

A salient privacy feature in Fabric is the option of having *channels*. It allows a subset of participants to transact with each other. Each channel has a ledger that is accessible and visible only to its participants.

A Fabric *Certificate Authority* (CA) provides identities to every registered member. Figure 1.3 shows the complete Hyperledger Fabric network components.

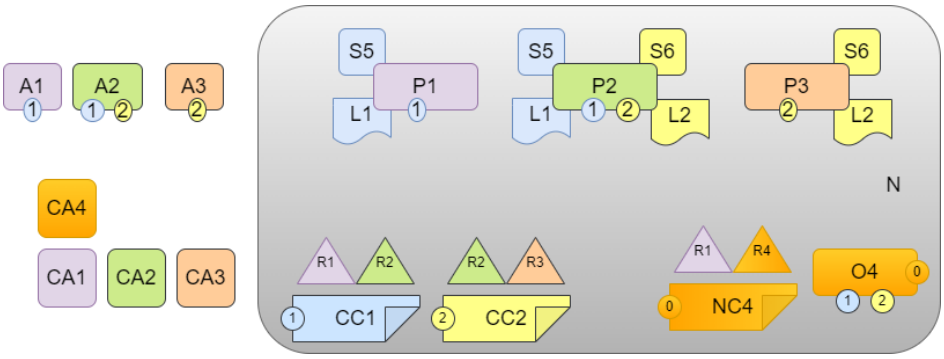

Figure 1.3 Hyperledger Fabric network

The Fabric is leveraged among four organizations: *R1, R2, R3, and R4*. Peers *P1, P2, and P3* are hosted by organizations *R1, R2, and R3* respectively. Peer *P1* holds a copy of ledger *L1* and is part of channel *C1* and runs the chaincode *S5* specific to *C1*. Peer *P3* is a part of channel *C2* and has a copy of ledger *L2* running the chaincode *S6*. *P2* is associated both with channel *C1* and *C2* running chaincodes *S5* and *S6* and holds the copy of both the ledger *L1* and *L2*. Organizations *R1* and *R4* host the ordering service. There are separate certificate authorities for each organization. Organizations *R1, R2, and R3* have client services *A1, A2, and A3* for initiating business transactions. *CC1 and CC2* are the channel configuration files for channels *C1 and C2* respectively.

1.2.2 Transaction Flow in Hyperledger Fabric

Fabric follows a distinct transaction flow of execute-order-validate (Androulaki *et al.* 2018). This flow is segregated into three divisions such that all three are executed independently. Endorser peers execute the chaincode, the orderer orders the transactions,

and the committer nodes verify and commit the transactions in the blockchain. Assuming the Fabric architecture presented in Figure 1.3 is up and running. Organizations *R1* and *R2* involve in an asset exchange where *R1* is a buyer of a food product sold by *R2*. The transaction flow is described below and depicted in Figure 1.4:

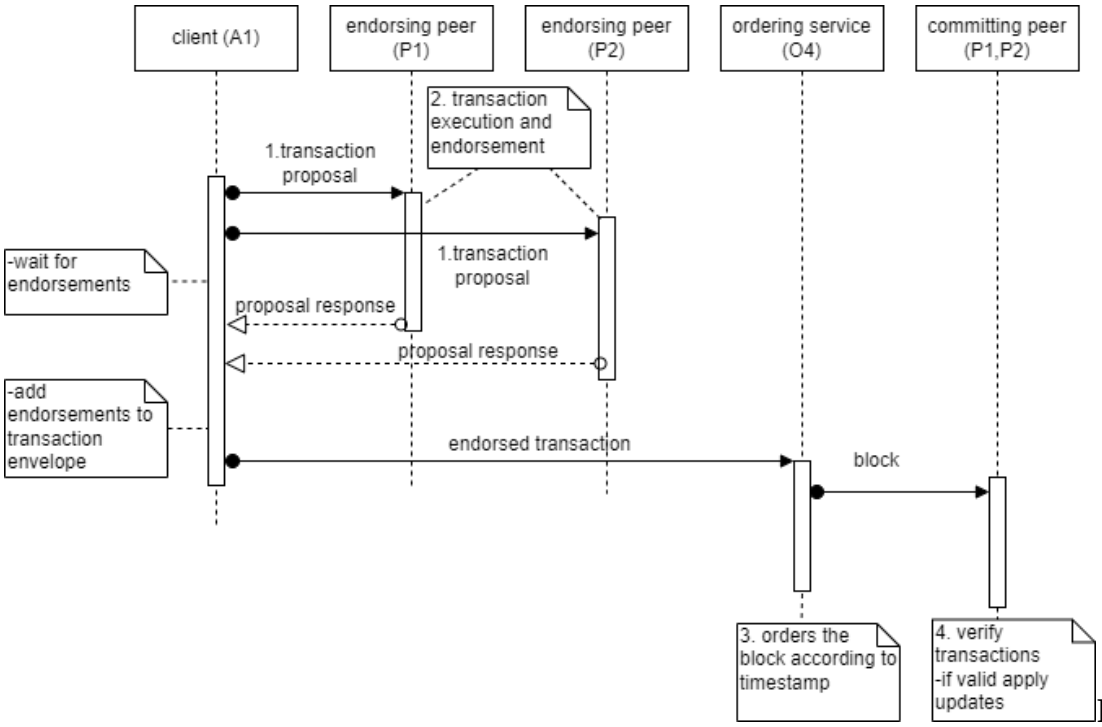

Figure 1.4

Transaction flow in Hyperledger Fabric

Transaction initiation: Client application *A1* sends a transaction proposal to the Fabric network for a purchase order. The endorsement policy is of the form $(R1.P1\ AND\ R2.P2)$. This implies that peer *P1* from an organization *R1* and peer *P2* from an organization *R2* must execute and endorse the transaction proposed by *A1*. Based on the endorsement policy, the transaction proposal is sent to the endorsing peers *P1* and *P2*.

Transaction execution: Upon receiving the transaction proposal from *A1* the peers check the signature on the proposal. It also verifies if it is an authorized client to submit transactions. After confirming that the transaction request is valid, the appropriate chaincode for asset exchange is invoked. The input parameters specified in the transaction proposal are passed to the chaincode arguments and executed by the endorsing peers. The

results of the chaincode are packed as a transaction proposal and signed or endorsed by the endorsers which are then sent back to the client *A1*. The client forwards the transaction response to the orderer after receiving a sufficient number of endorsements.

Transaction ordering: The orderer *O4* packs the transactions based on the timestamps and broadcasts the block of transactions in the channel *C1*.

Transaction commit: Every peer associated with the channel *C1* checks all the transactions and commits the validated ones in the blockchain. This updates the world state of the ledger and eventually, the block is appended to the blockchain. The client application *A1* is then notified about the success of the transaction commit.

1.2.2.1 Endorsement policy

During the endorsement mechanism, endorsing peers execute the chaincode pertaining to the requested transaction and sign the results of the chaincode execution indicating that it was executed by the said peers. The set of endorsing peers is decided based on the endorsement policy.

Definition 1.1. [Endorsement policy] An endorsement policy can be defined as a minimal set of organizations needed to endorse a transaction to make it a valid one. It is a logical expression with policy principals acting as operands and *(AND; OR)* serving as logical operators. A policy principal comprises the organization's identity and the role of the entity within the organization. A role can take a value among four possibilities: i) admin, ii) client, iii) member, and iv) peer.

A typical endorsement policy looks like the one given in Equation (1.2).

$$Org1.admin\ AND\ Org2.admin \tag{1.2}$$

The endorsement policy above enforces that the admin of the organization with identity $'Org1'$ with role $'admin'\ AND\ 'admin'$ of $'Org2'$ should endorse the transaction. During validation, the committing nodes check for endorsements (or signatures) from both the organization admins. And the transaction is committed on the blockchain only if the endorsements satisfy the endorsement policy. There are three basic types of endorsement policies:

- AND variant: $Org1.admin\ AND\ Org2.admin$
- OR variant: $Org1.admin\ OR\ Org2.admin$
- Threshold variant: $Outof(2, Org1.admin, Org2.admin, Org3.admin)$

Apart from these basic variants, there is a compound variant that contains both the logical operators AND and OR.

Endorsement policy can be set up at three different levels: i) chaincode-level - policy is specific to a particular chaincode, ii) collection-level - collections are private databases that store sensitive information pertaining to a single organization. The ledger holds the hashes of the information stored in collections. Collection-level policies are set for specified collections. iii) state-level - is a fine-grained level that applies to a specific key-value pair.