BCDV 1025
Enterprise Blockchain

2023 February

week 01 - class 03

Consensus Mechanism

In Hyperledger Fabric, the consensus is achieved once the organization Endorsed the transaction based on the Endorsement Policy.

Endorsement Policy

- Every chaincode has an endorsement policy which specifies the set of peers on a channel that must execute chaincode and endorse the execution results in order for the transaction to be considered valid.
- By default, endorsement policies are specified in the chaincode definition, which is agreed to by channel members and then committed to a channel
- If you are using Private Data Collections, you will have to specify it in the chaincode definition.
- The endorsement Policy is defined in the configtx.yaml
- Each channel has its own Endorsement Policy.
- Each participating organization must approve the Endorsing Policy and the Chaincode definition.
- Once the chaincode definition is approved by all the orgs it is committed to the channel
- The endorsement Policy is based on the MSP Role of the organization.
- MSP ID and ROLE represent one of the four accepted roles:

1. `member`
2. `admin`
3. `client`
4. `peer`

Here are a few examples of valid principles:

- `'Org0MSP.admin'` : any administrator of the `Org0MSP` MSP
- `'Org1MSP.member'` : any member of the `Org1MSP` MSP
- `'Org1MSP.client'` : any client of the `Org1MSP` MSP
- `'Org1MSP.peer'` : any peer of the `Org1MSP` MS

The following are the valid Expressions for Endorsement Policy:

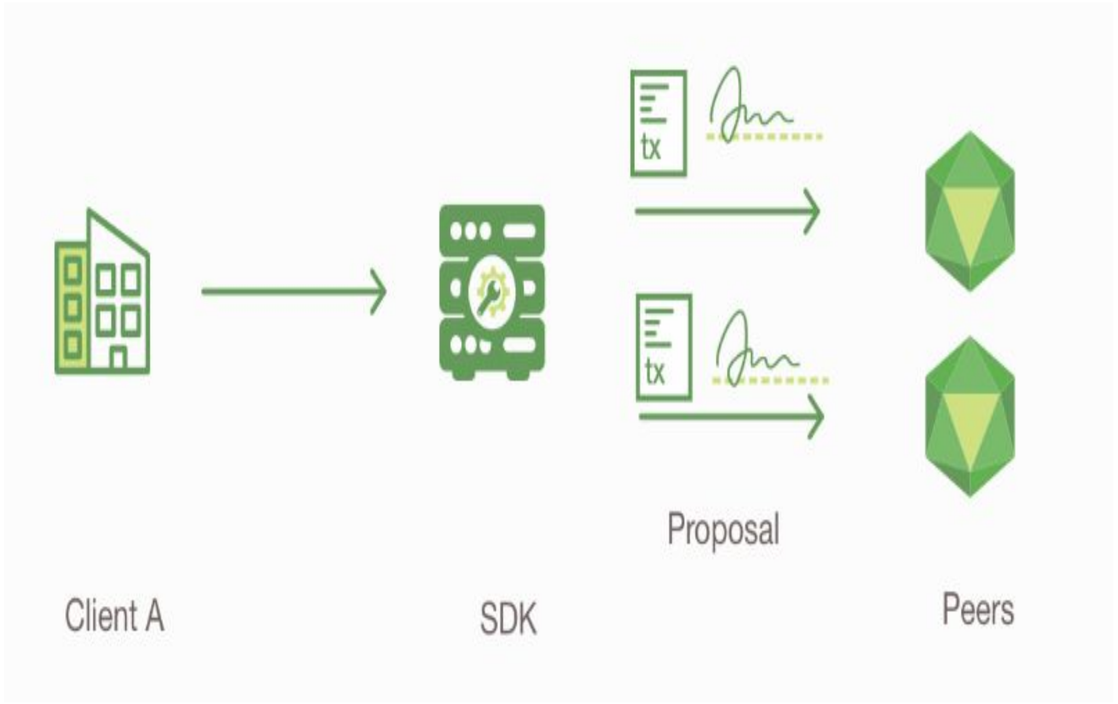
1. `AND`
2. `OR`
3. `OutOf`

Once the transaction is Endorsed by each organization based on their individual Endorsement policies the following three types of consensus can be set up to approve the transaction

1. `MAJORITY` Orgs
2. `ANY` Orgs
3. `ALL` Orgs

Transaction Flow

1. An application leveraging a supported SDK (Node, Java, Go) utilizes one of the available API's to generate a transaction proposal. The proposal is a request to invoke a chaincode function with certain input parameters, with the intent of reading and/or updating the ledger.
2. The SDK submits the transaction proposal to a target peer, which will manage the transaction submission on behalf of the client.
3. The target peer first forwards the transaction proposal to other peers for execution, as required by the endorsement policy.



1. The endorsing peers verify

a. The **transaction** proposal is well-formed

b. It has not been submitted already in the past (replay-attack protection)

c. The signature is valid (using the MSP)

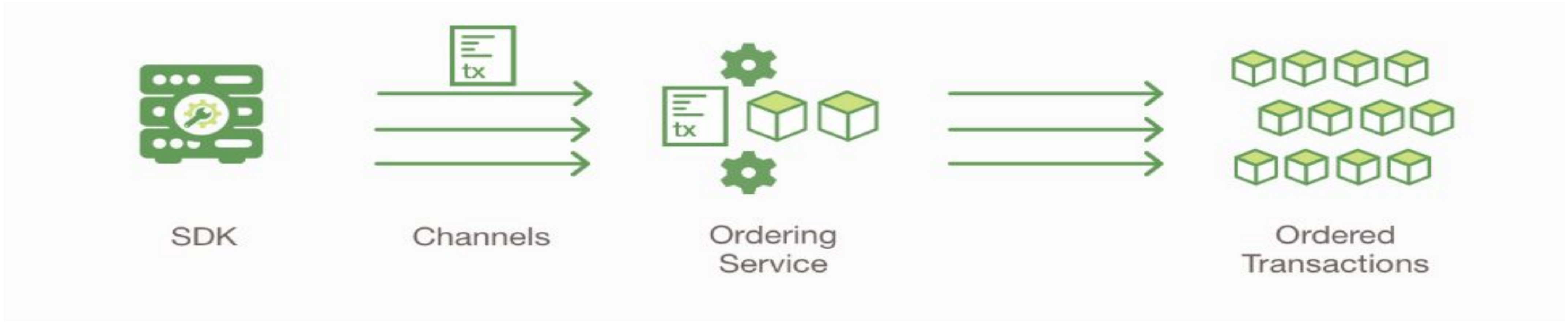
d. The submitter (Client A, in the example) is properly authorized to perform the proposed operation on that channel (namely, each endorsing peer ensures that the submitter satisfies the channel's *Writers* policy).

5. The target peer verifies the proposal responses are the same prior to proceeding with the transaction submission. If a transaction is submitted without this check, the endorsement policy will still be checked and enforced when each peer validates transactions prior to committing them.

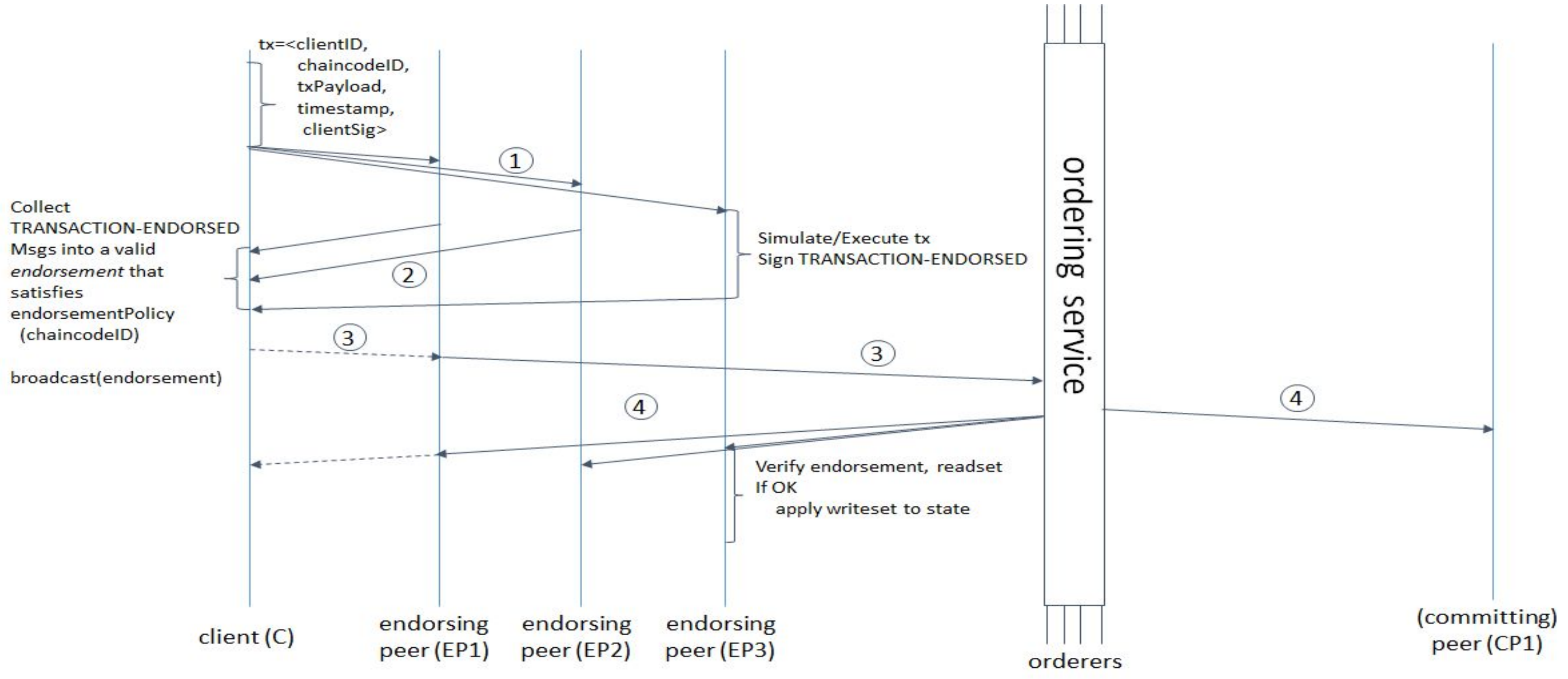
6. The target peer “broadcasts” the **transaction** proposal and response within a “transaction message” to the ordering service.

7. The ordering service receives transactions, orders them, and creates blocks of transactions per channel.

8. The blocks of **transaction**s are “delivered” to all peers on the channel. The transactions within the block are validated to ensure endorsement policy is fulfilled and ensure that there have been no changes to the ledger state for read set variables since the read set was generated by the transaction execution. Transactions in the block are tagged as valid or invalid.



9.Each peer appends the block to the channel's chain, and for each valid **transaction** the write sets are committed to a current state database. An event is emitted by each peer to notify the client application that the transaction (invocation) has been immutably appended to the chain, as well as notification of whether the transaction was validated or invalidated.


tx=<clientID,
chaincodeID,
txPayload,
timestamp,
clientSig>
1
Collect
TRANSACTION-ENDORSED
Msgs into a valid
endorsement that
satisfies
endorsementPolicy
(chaincodeID)
2
Simulate/Execute tx
Sign TRANSACTION-ENDORSED
3
3
broadcast(endorsement)
4
4
Verify endorsement, readset
If OK
apply writeset to state
ordering service
client (C)
endorsing peer (EP1)
endorsing peer (EP2)
endorsing peer (EP3)
orderers
(committing) peer (CP1)