FABRIC
NETWORK

TEST-NETWORK ARCHITECTURE

- **No of Orgs** : 2
  - Org1 - 1 peer (peer0.org1.example.com)
  - Org2 - 1 peer (peer0.org2.example.com)
- **Ordering Service** : Raft (Single Node - orderer.example.com)
- **Database Type** - CouchDB (couchdb0, couchdb1)
- **Certificate Authority** : Separate CA for org1, org2 and orderer
  - ca_org1
  - ca_org2
  - ca_orderer

STEPS TO BUILD THE NETWORK

- **Build the network**
  - Generate the crypto material **(fabric-ca or cryptogen)**
  - Bring up the components (2 orgs with 1 peer each and 1 orderer)
  - Generate the genesis block file
  - Create channel and join orderer
  - Joining the peers to the created channel
  - Anchor peer update
- **Deploy the chaincode** (Follow the chaincode lifecycle)
- **Invoke/Query the chaincode**

GENERATING CERTIFICATES

KBA

Generating Cryptographic Materials

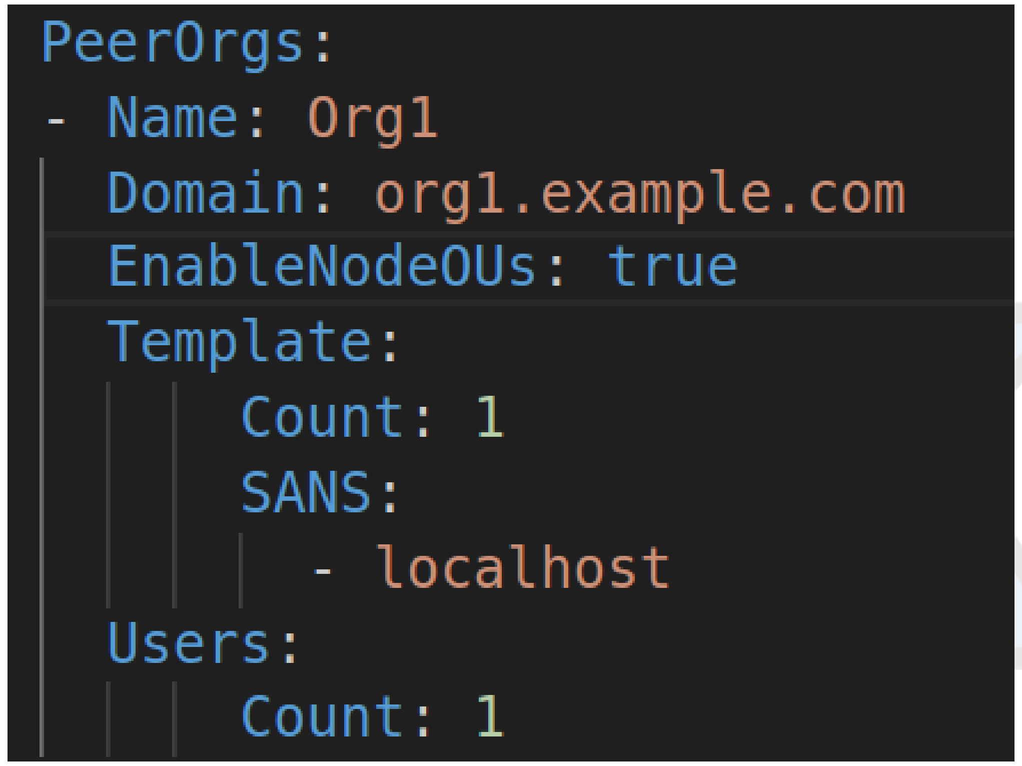
- Cryptogen
- Fabric-CA

CRYPTOGEN

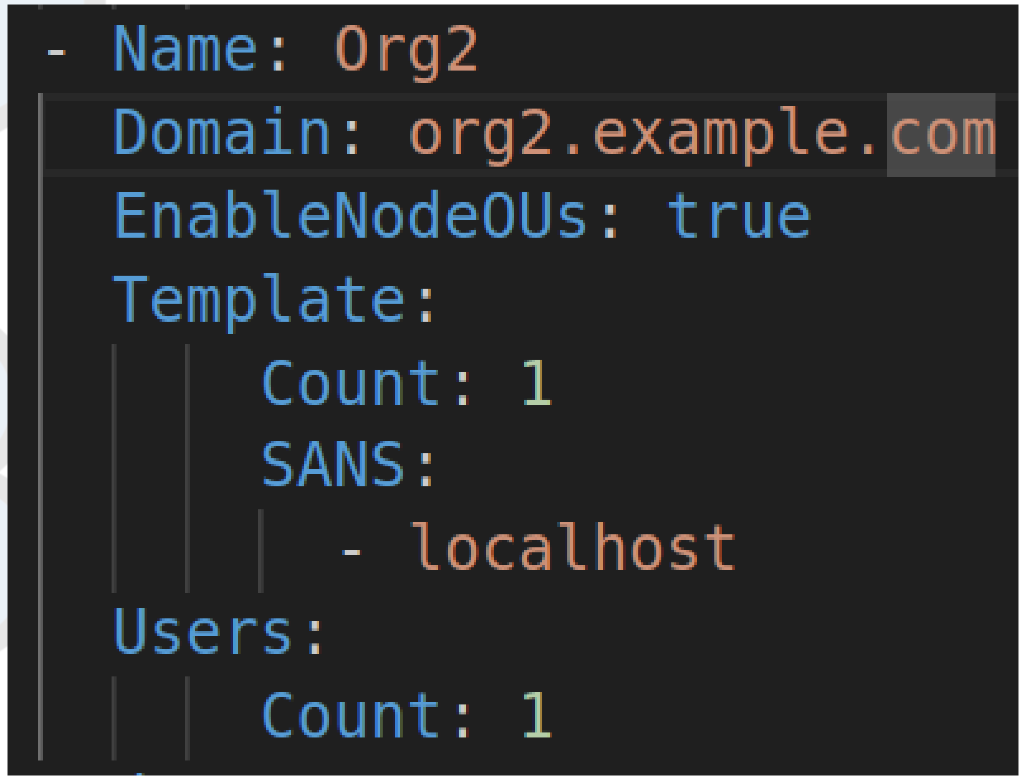
- Configuration File

- Generation

- **PeerOrgs**: Specifies the peer organizations that are part of the network
- **Name** – Name of organization
- **Domain** – Domain associated with organization
- **EnableNodeOUs** – Organizational Units, feature which allows the MSP to recognize identities based on their OUs like peer, orderer, admin, client.
- **Template** – Peer node specification
  - **Count** – Number of peers
  - **SANS** – Subject Alternative Names for the certificates like localhost
- **Users** – Number of users

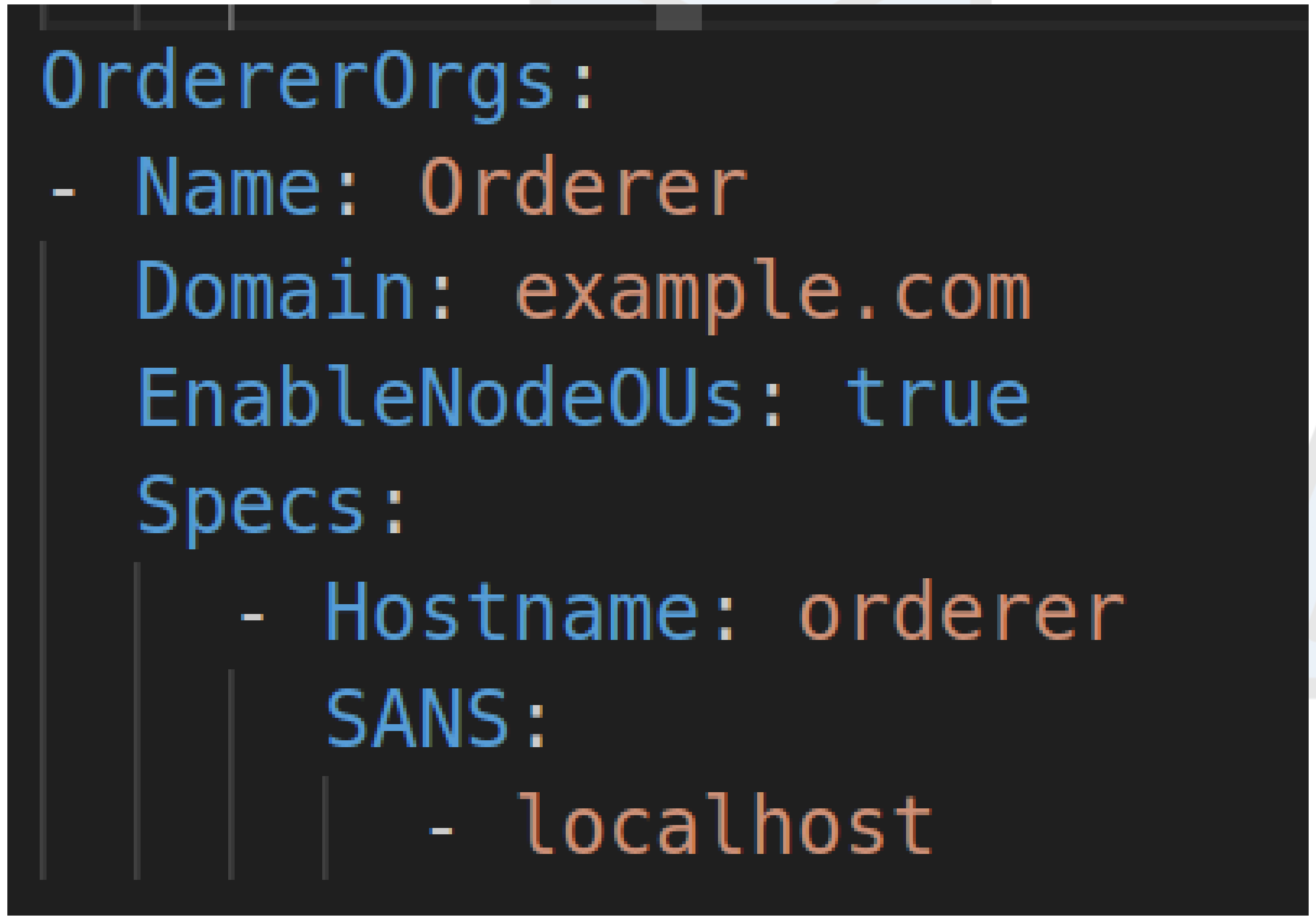


```
PeerOrgs:
- Name: Org1
  Domain: org1.example.com
  EnableNodeOUs: true
  Template:
    Count: 1
    SANS:
      - localhost
  Users:
    Count: 1
```



```
- Name: Org2
  Domain: org2.example.com
  EnableNodeOUs: true
  Template:
    Count: 1
    SANS:
      - localhost
  Users:
    Count: 1
```

- **OrdererOrgs**: Specifies the orderer organizations that are part of the network
- **Name** – Name of organization
- **Domain** – Domain associated with organization.
- **EnableNodeOUs** – Organizational Units feature which allows the MSP to recognize identities based on their OUs like orderer and admin.
- **Specs** – Configuration for generating cryptographic materials for orderer nodes.
    - **SANS** – Subject Alternative Names for the certificates like localhost



```yaml
OrdererOrgs:
- Name: Orderer
  Domain: example.com
  EnableNodeOUs: true
  Specs:
    - Hostname: orderer
      SANS:
        - localhost
```

Generating Cryptographic Materials

- Cryptogen
- Fabric-CA

FABRIC-CA

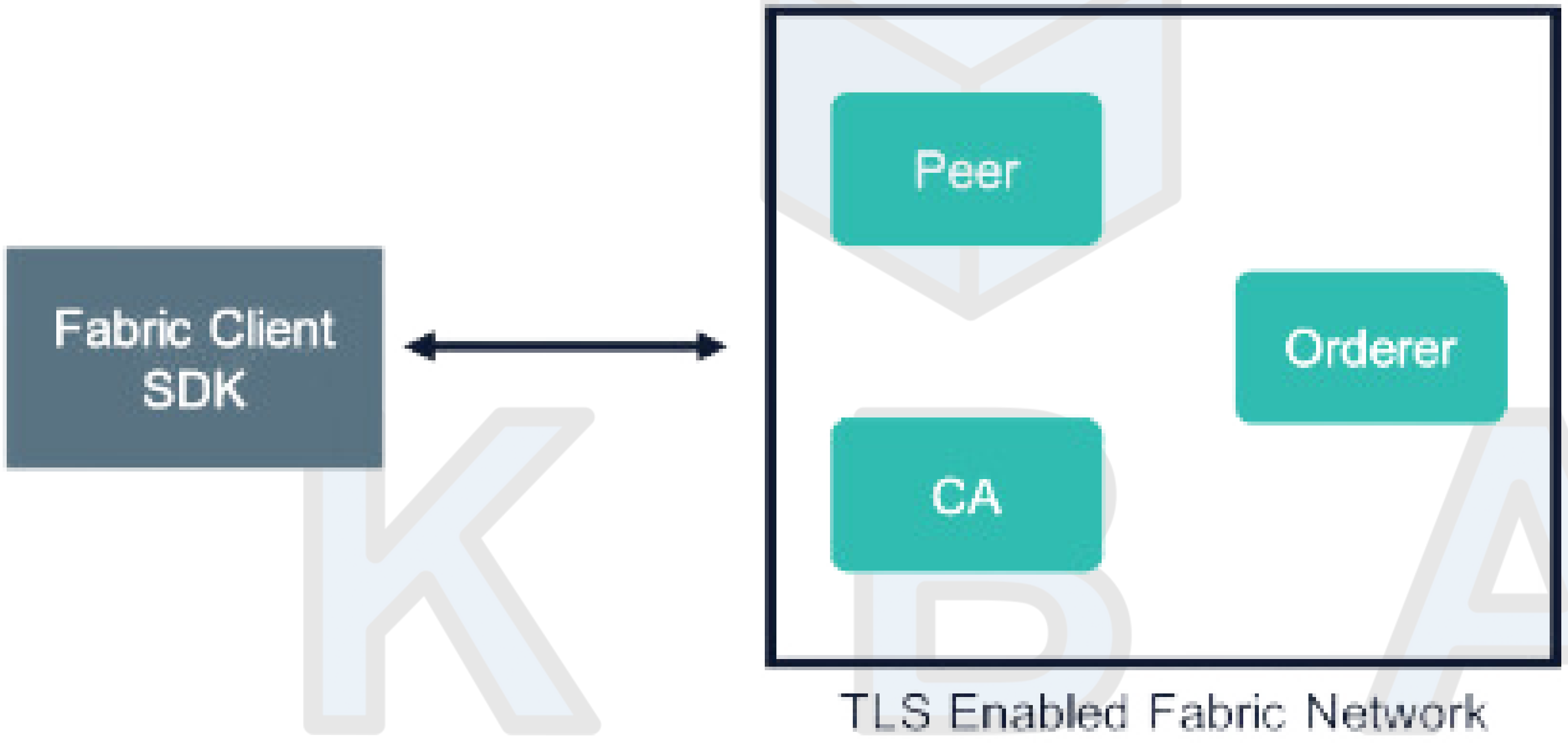
- Fabric-CA is a certificate authority provided by Hyperledger Fabric to provide and manage identities.
- There are 2 binaries provided for fabric-ca
  - **fabric-ca-server** - Initializes and starts Fabric CA, registers an admin
  - **fabric-ca-client** - A client to interact with the ca server to register, enroll and revoke identities.

Steps for running theFABRIC-CA

- Configuration File
- Running CA
- Identity Registration using **Register & Enroll**
- Enable TLS for securing communication between peer, orderer, CA or a Client application.

TLS - Transport Layer Security

- TLS is securing communication between any two parties inside a fabric network.
- These parties can be a nodes such as peer and orderer, etc.


Fabric Client SDK
Peer
Orderer
CA
TLS Enabled Fabric Network