ARCADIAN-IoT

ARCADIAN-IOT TRAINING

BLOCKCHAIN

Sergio Sanz (EVIDEN - ATOS)

26/09/2023

AGENDA

1. Introduction to Hyperledger Fabric network
2. Smart Contract employed in ARCADIAN-IoT
3. Services deployed and test demo
4. Hyperledger Fabric network Deployment

1. INTRODUCTION TO HYPERLEDGER FABRIC NETWORK

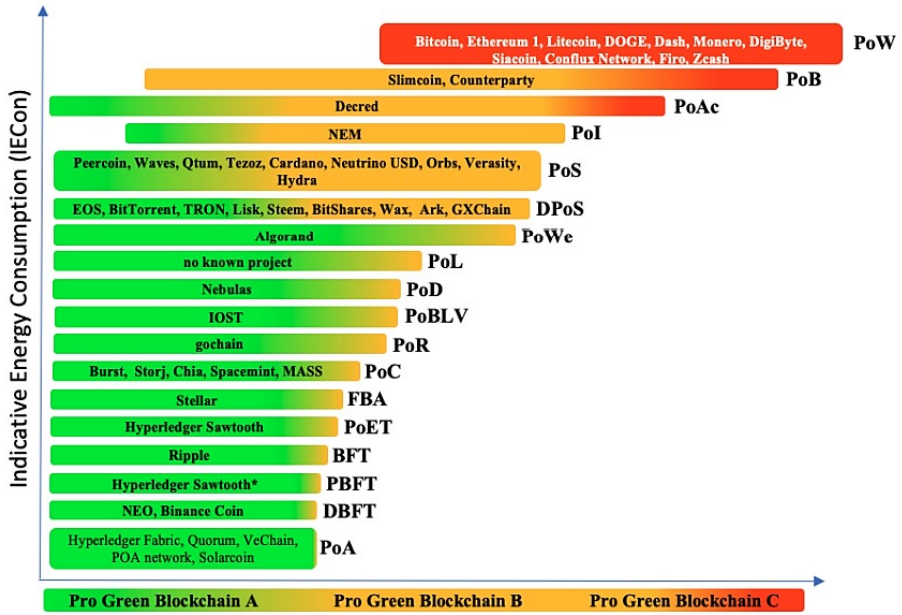
HYPERLEDGER FABRIC INTRODUCTION

ARCADIAN-IoT

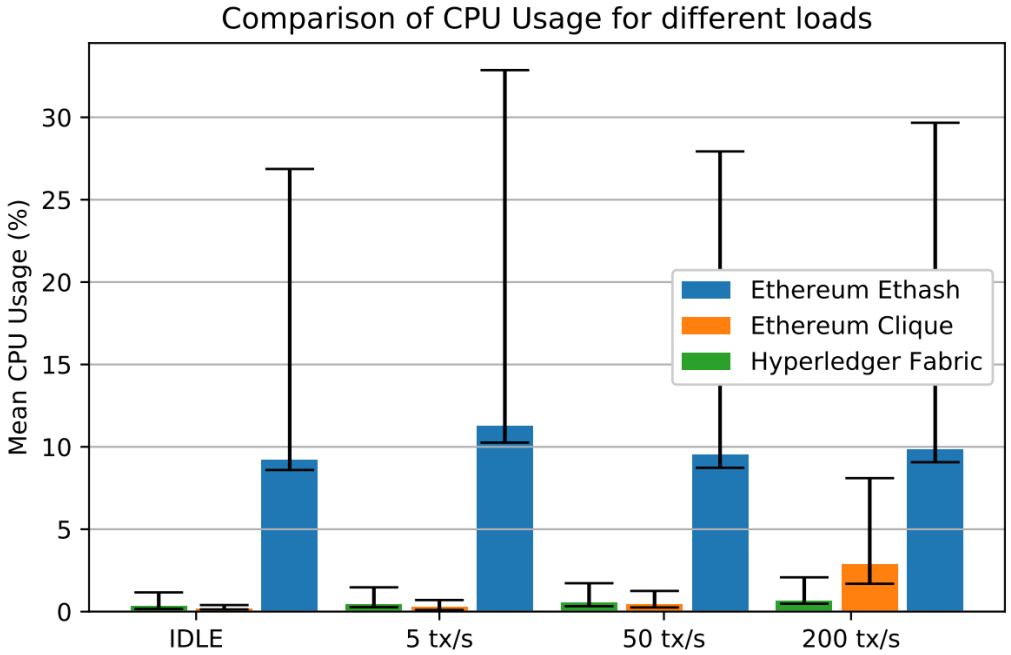
- What is Hyperledger Fabric?
  - Open Source blockchain framework designed for building enterprise grade permissioned blockchain networks
- Why use Hyperledger Fabric?
  - High performance with great efficiency
- Key Features:
  - Permissioned Network (suitable for enterprises with privacy and security requirements)
  - Modular Architecture (allows organizations to customize and select the components they need)
  - Smart Contracts (Chaincode) define the business logic of the blockchain application
  - Scalability and Performance (suitable for applications with a high volume of transactions)
  - Privacy and Confidentiality (confidential transactions between specific participants)

WHY USE HYPERLEDGER FABRIC

- **Efficiency of Hyperledger Fabric**
  - **PoA is energy efficient ranked the highest on the IECon chart: 10x better than Ethereum (3)**
  - **Minimal CPU usage incentivizes energy efficiency**



IECon chart (1)



Comparison of CPU Usage for different loads (2)

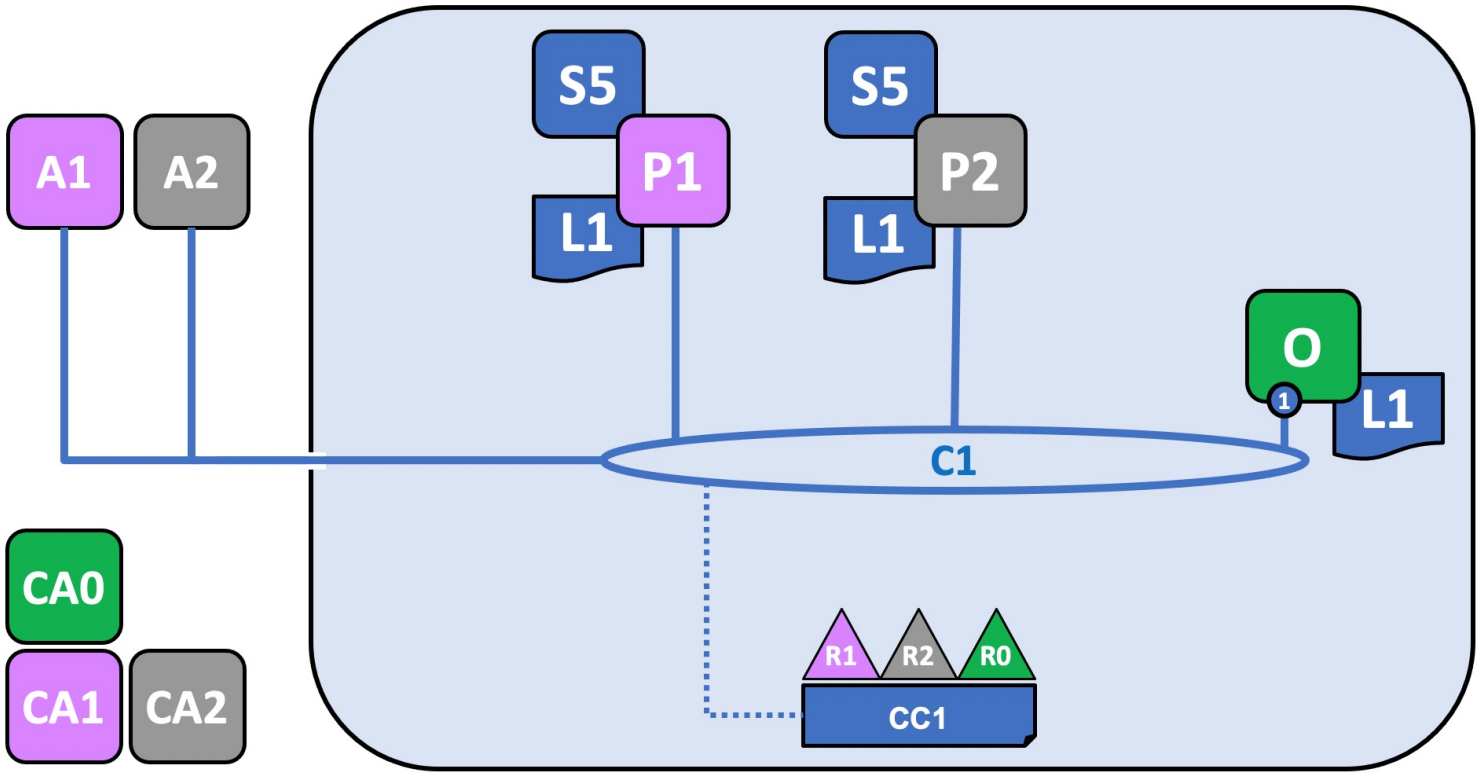
(1) A. O. Bada, A. Damianou, C. M. Angelopoulos and V. Katos, "Towards a Green Blockchain: A Review of Consensus Mechanisms and their Energy Consumption," 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2021, pp. 503-511, doi: 10.1109/DCOSS52077.2021.00083.

(2) Dimitri Saingre. Understanding the energy consumption of blockchain technologies : a focus on smart contracts. Distributed, Parallel, and Cluster Computing [cs.DC]. Ecole nationale supérieure MinesTélécom Atlantique, 2021. English. ffNNT : 2021IMTA0280ff. fftel-03546651

HYPERLEGER FABRIC ARCHITECTURE

ARCADIAN-IoT

- Ordering Service
- Peer Nodes
- Membership Service Provider (MSP)
- Channels



2. SMART CONTRACT EMPLOYED IN ARCADIAN-IOT

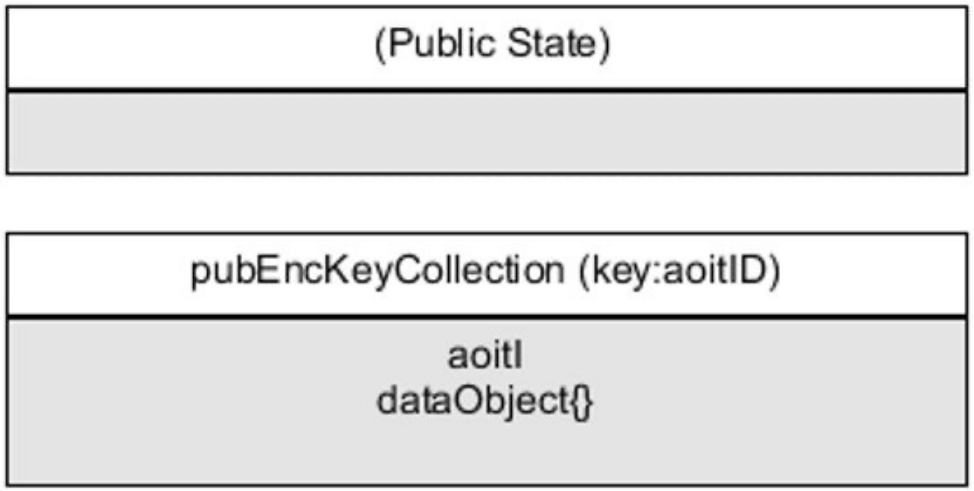
PUBLISHER SMART CONTRACT

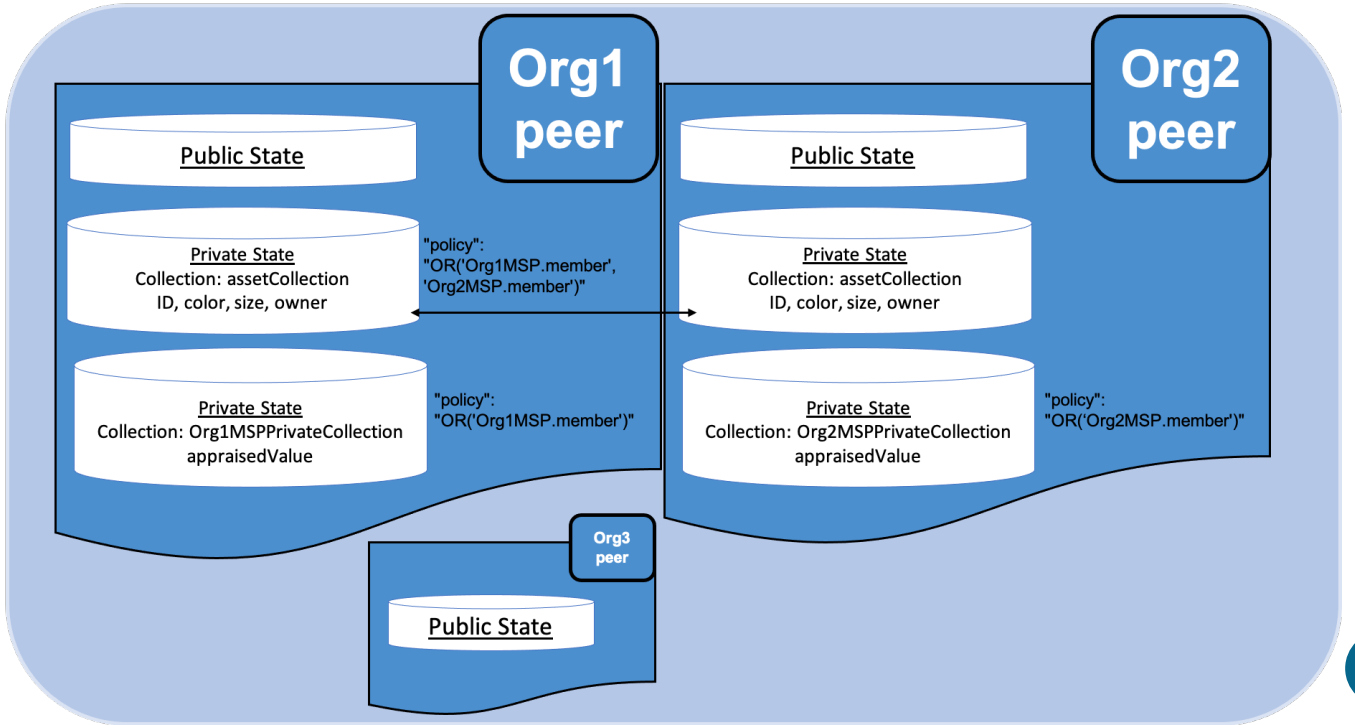
ARCADIAN-IoT

- Written in JavaScript
- Transactions
  - createReputation: Creates a new private asset with the specified *`reputation ID`*, *`collection`* and data object.
  - readReputation: Reads the private asset with the specified *`reputation ID`* from the *`collection`*.
  - updateReputation: Updates the private asset with the specified *`reputation ID`* and data object.
  - deleteReputation: Deletes the private asset with the specified `reputation ID` from the *`collection`*. It will also purge private data so it didn't appear in blocks or logs

COLLECTION FILE

- This file is a JSON file that defines collections for Hyperledger Fabric. Collections are used to specify the endorsement policy and access control for private data in a Hyperledger Fabric network.
- reputationCollection member-only read and write access for reputation objects
- abeKeyCollection member-only read and write access for abeKey objects





COLLECTION.JSON

ARCADIAN-IoT

```
  {
    "name": "reputationCollection",
    "policy": "OR('Org1MSP.member',
'Org2MSP.member')",
    "requiredPeerCount": 0,
    "maxPeerCount": 1,
    "blockToLive":0,
    "memberOnlyRead": true,
    "memberOnlyWrite": true,
    "endorsementPolicy": {
      "signaturePolicy":
"OR('Org1MSP.member', 'Org2MSP.member')"
    }
  },
```

```
  {
    "name": "abeKeyCollection",
    "policy": "OR('Org1MSP.member',
'Org2MSP.member')",
    "requiredPeerCount": 0,
    "maxPeerCount": 1,
    "blockToLive":0,
    "memberOnlyRead": true,
    "memberOnlyWrite": true,
    "endorsementPolicy": {
      "signaturePolicy":
"OR('Org1MSP.member', 'Org2MSP.member')"
    }
  }
```

3. SERVICES DEPLOYED AND TEST DEMO

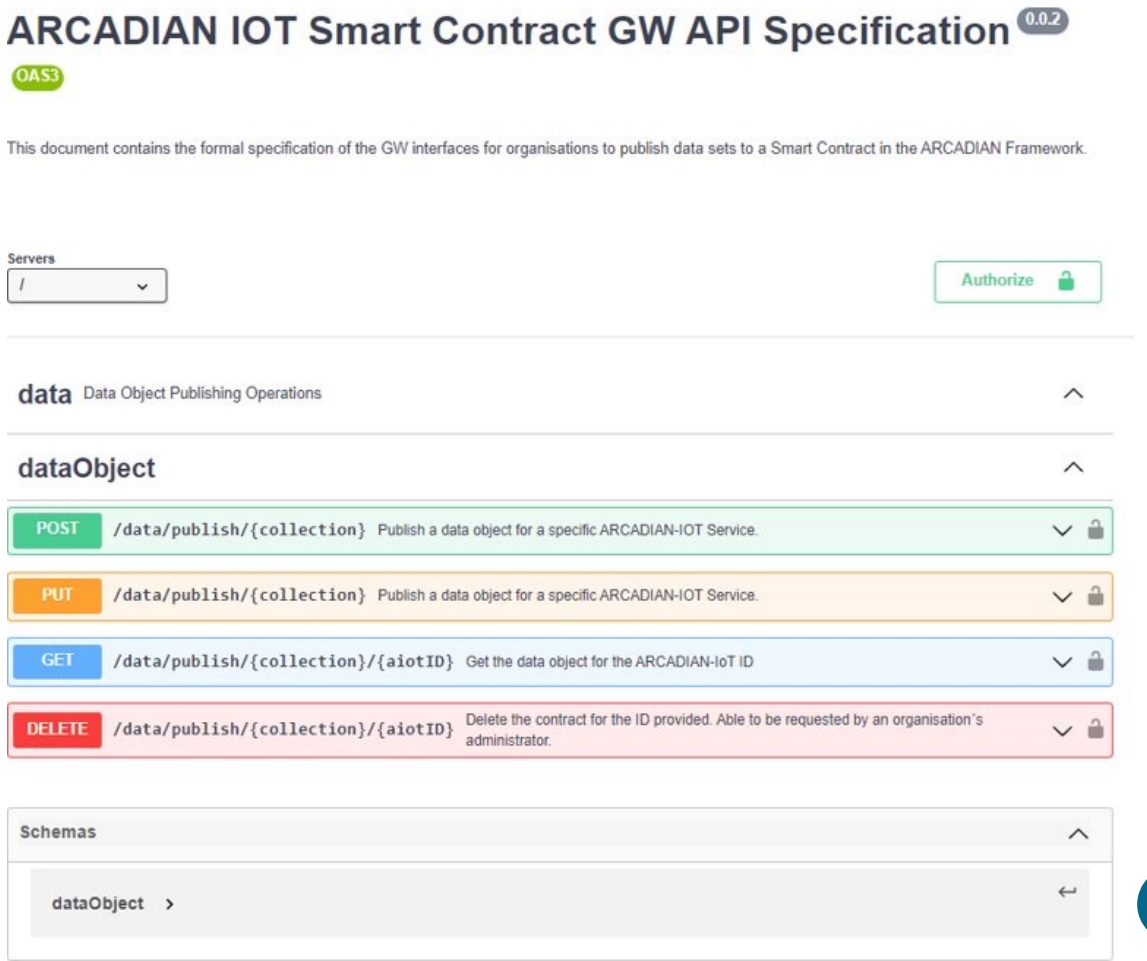
PUBLISHER GATEWAY DAPP

ARCADIAN-IoT

- JavaScript Rest API that interacts with Publisher Smart contract
- Routes for application:
  - GET `/data/:collection/:objectKey`: Retrieves a data object for a given object key and private data collection.
  - DELETE `/data/publish/:collection/:objectKey`: Deletes a data object for a given object key and private data collection.
  - POST `/data/publish/:collection`: Creates a new data object for a given private data collection.
  - PUT `/data/publish/:collection/:objectKey`: Updates an existing data object for a given object key and private data collection.

TESTING ON TEST NETWORK

- For simplification purposes we will perform the following actions on the test network with smart contract already deployed and a running gw application.
- Login into the swagger url
  - Create, read, read with wrong collection.
  - Update and verify the changes are done
  - Delete and verify it has been deleted



4. HYPERLEDGER FABRIC DEPLOYMENT

DEPLOYING NETWORK

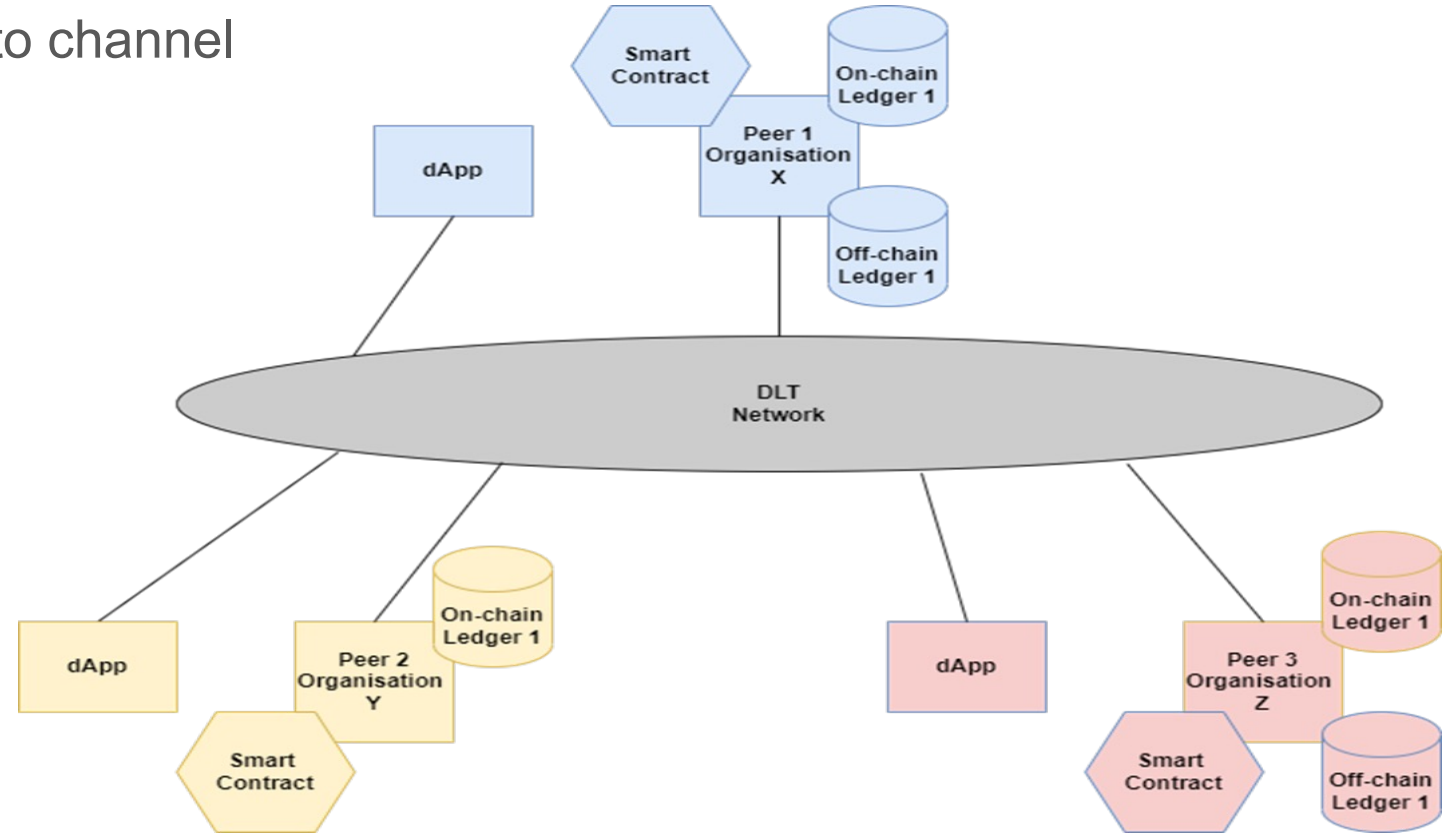
ARCADIAN-IoT

- System Requirements
- Choosing the Right Version: 2.5
- Network Topology: Multi-Organization
- Prerequisites: Docker, Node.js, Fabric Samples
- Cryptographic Materials (Certificates, Keys)
- Configuring Network Parameters

SETTING UP THE NETWORK INFRASTRUCTURE

ARCADIAN-IoT

- Network topology
  - 3 orgs with 1 peer
  - 1 channel with one Smart Contract deployed
  - dApp connected to channel



USING DOCKER SWARM

ARCADIAN-IoT

- In this example we will use Docker Swarm considering we have a host for each organization.

On host 1: *docker swarm init --advertise-addr <host-1 ip address>*
docker swarm join-token manager

- Save *docker swarm join-token manager*

On host 2 and 3: *<output from join-token manager>* `--advertise-addr <host n ip>`

- Create an overlay

On host 1: *docker network create --attachable --driver overlay* ***first-network***

- Check with `docker network ls`

SETTING UP CONFIGURATION

ARCADIAN-IoT

- In configtx.yaml setting up the 3 organizations
- In crypto-config.yaml setting up the organizations and their peers to generate a directory with all the cryptography neccesary
- In hostn.yaml with the following services: peer, orderer, couchdb

GENERATE CRYPTO MATERIAL

ARCADIAN-IoT

```
cryptogen generate --config=./crypto-config.yaml

configtxgen -profile SampleMultiNodeEtcdRaft -channelID networkchannel -
outputBlock ./channel-artifacts/genesis.block

configtxgen -profile ThreeOrgsChannel -channelID mychannel -
outputCreateChannelTx ./channel-artifacts/channel.tx

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org1MSPanchors.tx -channelID mychannel -asOrg Org1MSP

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org2MSPanchors.tx -channelID mychannel -asOrg Org2MSP

configtxgen -profile ThreeOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org3MSPanchors.tx -channelID mychannel -asOrg Org3MSP
```

LAUNCHING HOST AND JOINING CHANNELS

ARCADIAN-IoT

- To launch a host `docker-compose -f host1.yaml up -d`
- To create channel

```
docker exec cli peer channel create -o orderer.example.com:7050 -c mychannel
-f ./channel-artifacts/channel.tx --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganization
s/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-
cert.pem
```

- Join channel on every host

```
docker exec peerx.orgy.example.com peer channel join -c mychannel
```

- Check on every host

```
docker exec peerx.orgy.example.com peer channel getinfo -c mychannel
```

```
Blockchain info: {"height":3,"currentBlockHash":"QkC8ZScdXsUqstfBRpWIKC6waeAc5wuGeG9
OhG78IS4=","previousBlockHash":"qlmIwjT2v4R5ZV+ovrnEURI9u/X8X7KMoWxjGVF8LJc="}
```

PACKAGING AND INSTALLING CHAINCODE

ARCADIAN-IoT

- For every host:
  - First package Chaincode(Smart Contract)

```
docker exec cli peer lifecycle chaincode package
${SC_NAME}.tar.gz --path ${SC_PATH} --lang node --label
${SC_NAME}_${SC_VERSION}
```

  - Install Chaincode

```
docker exec cli peer lifecycle chaincode install
${SC_NAME}.tar.gz
```

- Check with `docker ps` the new chaincode associated container

APPROVING THE CHAINCODE

ARCADIAN-IoT

- Get the package ID for the installed chaincode:

```
docker exec cli peer lifecycle chaincode queryinstalled
```

- Approving the Chaincode for Your Organization:
- ```
  docker exec cli peer lifecycle chaincode approveformyorg --tls --cafile
  /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizati
  ons/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.c
  om-cert.pem --channelID mychannel --name ${SC_NAME} --
  version  ${SC_VERSION}  --package-id $CC_PACKAGE_ID --sequence 1 --
  collections-config ${SC_PATH}/collections.json --signature-policy
  "OR('Org1MSP.peer','Org2MSP.peer')"
  ```
- Check approval status

```
docker exec cli peer lifecycle chaincode checkcommitreadiness --channelID
mychannel -name ${SC_NAME} --version ${SC_VERSION} --sequence 1  --output
json
```
```

COMMIT CHAINCODE

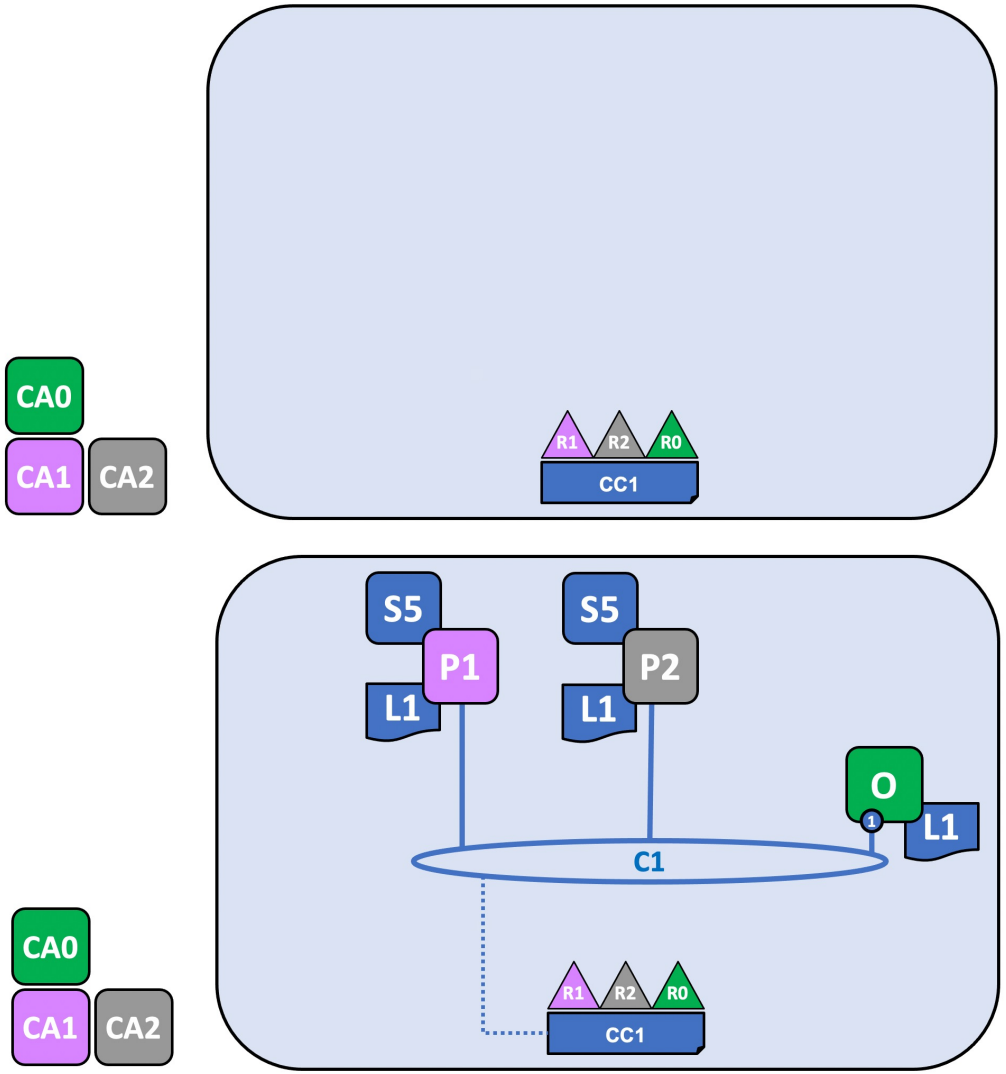
ARCADIAN-IoT

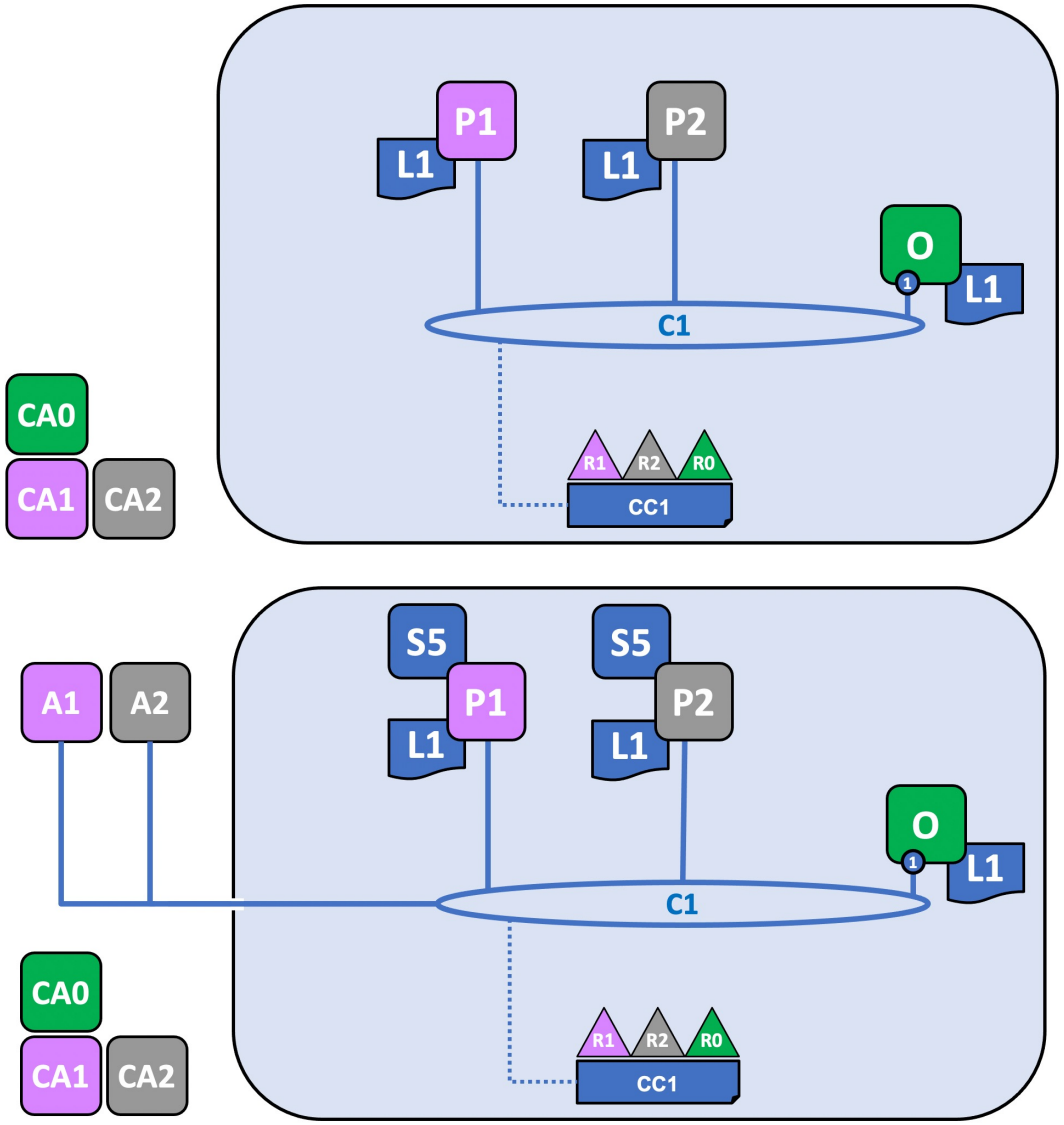
- Commit chaincode

```
docker exec cli peer lifecycle chaincode commit -o orderer.example.com:7050 --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt --channelID mychannel --name ${SC_NAME} --version ${SC_VERSION} --sequence 1 --signature-policy 'OR('\''Org1MSP.peer'\'','\''Org2MSP.peer'\'')' --collections-config /opt/gopath/src/github.com/chaincode/ssi-ledgeruself-fabric-chaincode/collections.json
```

- Check commit status `docker exec cli peer lifecycle chaincode querycommitted --channelID mychannel --name ${SC_NAME}`

GRAPHICAL VIEW





Source:Hyperledger Fabric doc

SOLVING ISSUES

ARCADIAN-IoT

- How to log Logging Control — hyperledger-fabricdocs main documentation
- Troubleshooting in Chain Code with Logspout Deploying a smart contract to a channel — hyperledger-fabricdocs main documentation

ARCADIAN-IoT

THANK YOU FOR YOUR ATTENTION

arcadian-iot.eu

ARCADIAN-IoT project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 101020259

THIS PROJECT IS PART OF THE NEXT GENERATION IoT

www.ngiot.eu