

---

# Adding a new peer to an existing organization

---

In this lab manual, we will see the procedure that needs to be followed to add a new peer to Org1.

**Note:** Make sure you have a working network (Fabric-network) with all the certificates generated, and the containers are up and running. Also, ensure that the chaincode is deployed successfully, and three command terminals(host, peer0\_Org1 and peer0\_Org2) are opened with the necessary environment variables set. Otherwise, follow the steps mentioned in the [Lab Manual - Setting up the Hyperledger Fabric network.](#)

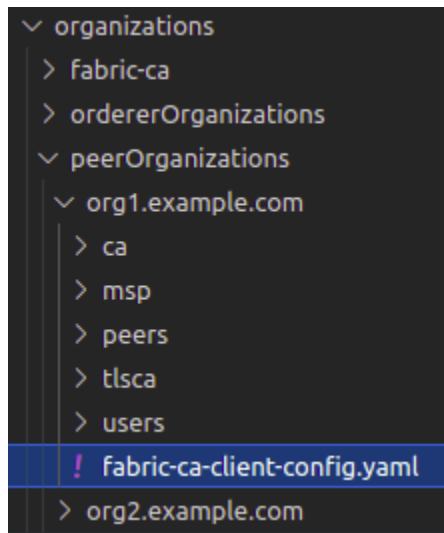
Before we start, let's brief the steps involved while adding a new peer to an existing organization.

- Generating crypto materials for the new peer that is to be added.
- Creating docker-compose files for the peer to launch the new peer container.
- Join the peer to the required channel.
- Installing chaincode onto the new peer.

## **Step1.** Generating Crypto materials for the new peer

We will use the fabric-ca-client to generate the crypto materials to register the new peer to the organization. Before we use the fabric-ca-client, we need to set the proper environments for the fabric-ca-client to interact with the fabric-ca-server.

First, we need to set the **FABRIC\_CA\_CLIENT\_HOME**, which refers to the path where the fabric-ca-client-config.yaml file is located.



Note: Execute the following commands from within the Fabric-network folder.

#####Execute the command in **host** terminal#####

**export**

**FABRIC\_CA\_CLIENT\_HOME=\${PWD}/organizations/peerOrganizations/org1.example.com/**

Once you set the **FABRIC\_CA\_CLIENT\_HOME** environment variable, let's try registering the new peer using the **fabric-ca-client register** command. Execute the following command.

#####Execute the command in **host** terminal#####

**fabric-ca-client register --caname ca-org1 --id.name peer1 --id.secret peer1pw --id.type peer --tls.certfiles \${PWD}/organizations/fabric-ca/org1/tls-cert.pem**

```
kba@Lab:~/CHF/Fabric-network$ fabric-ca-client register --caname ca-org1 --id.name peer1 --id.secret peer1pw --id.type peer --tls.certfiles ${PWD}/organizations/fabric-ca/org1/tls-cert.pem
2023/08/04 11:58:43 [INFO] Configuration file location: /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/fabric-ca-client-config.yaml
2023/08/04 11:58:43 [INFO] TLS Enabled
2023/08/04 11:58:43 [INFO] TLS Enabled
Password: peer1pw
```

---

In this command, we are passing the following arguments,

**caname** - The ca-name for the organization, for test network we have given ca name as ca-org1.

**id.name** - name of the peer to be registered

**id.secret** - Password of the peer to be registered

**id.type** - Type of identity (Client | Peer | Admin)

**tls.certfiles** - tls cert files of the ca, available inside organizations/fabric-ca/org1

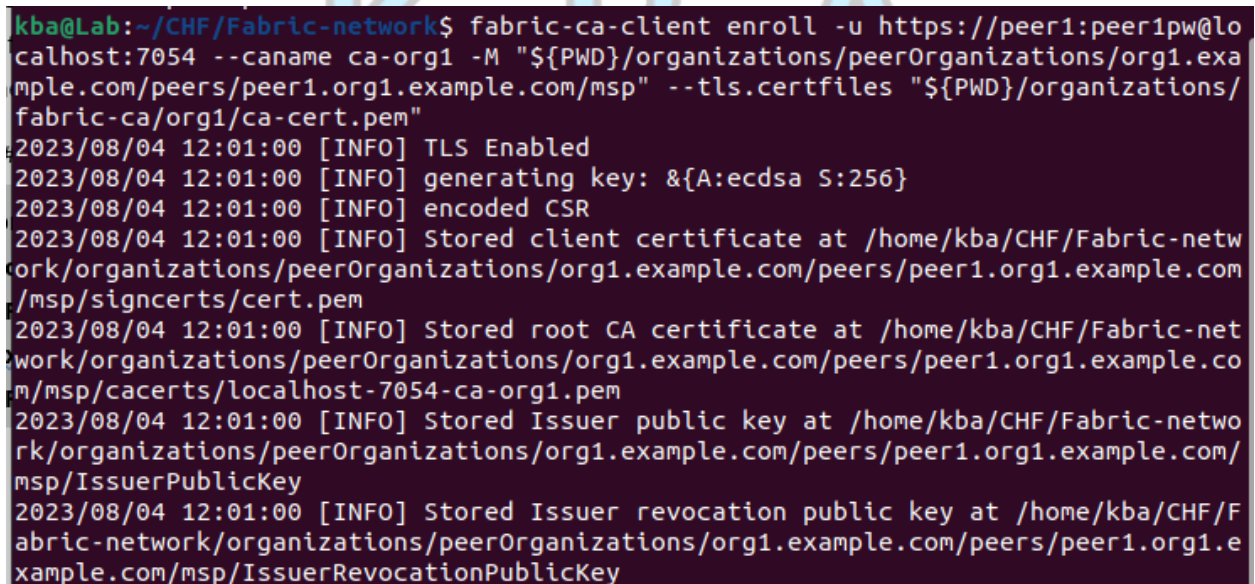
We now need to enroll the user. For enrolling the user, we need to use the id.user and id.secret of the peer that we registered. We also need to pass the ca name and MSP. Execute the following command.

#####Execute the command in **host** terminal#####

**fabric-ca-client enroll -u https://peer1:peer1pw@localhost:7054 --caname ca-org1 -M**

**"\${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp" --tls.certfiles**

**"\${PWD}/organizations/fabric-ca/org1/ca-cert.pem"**



```
kba@Lab:~/CHF/Fabric-network$ fabric-ca-client enroll -u https://peer1:peer1pw@localhost:7054 --caname ca-org1 -M "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp" --tls.certfiles "${PWD}/organizations/fabric-ca/org1/ca-cert.pem"
2023/08/04 12:01:00 [INFO] TLS Enabled
2023/08/04 12:01:00 [INFO] generating key: &{A:ecdsa S:256}
2023/08/04 12:01:00 [INFO] encoded CSR
2023/08/04 12:01:00 [INFO] Stored client certificate at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp/signcerts/cert.pem
2023/08/04 12:01:00 [INFO] Stored root CA certificate at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp/cacerts/localhost-7054-ca-org1.pem
2023/08/04 12:01:00 [INFO] Stored Issuer public key at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp/IssuerPublicKey
2023/08/04 12:01:00 [INFO] Stored Issuer revocation public key at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp/IssuerRevocationPublicKey
```

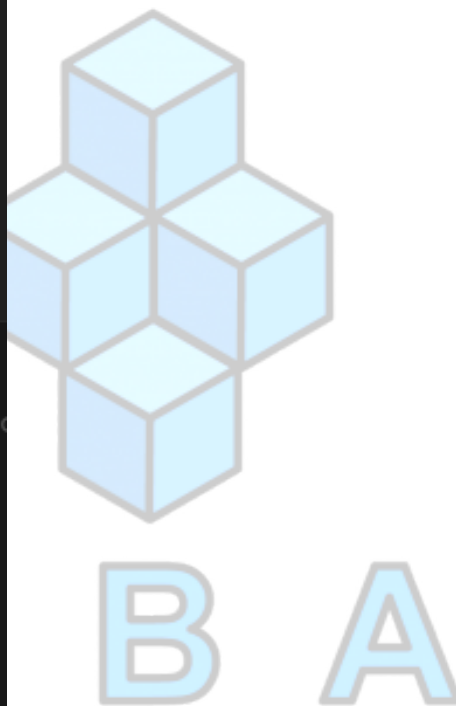
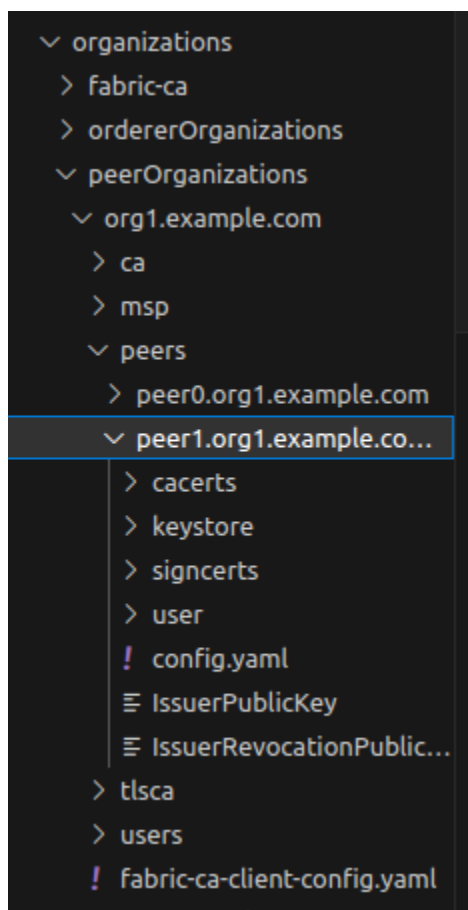
---

Once the peer is enrolled and the MSP is generated, arrange these certificates to correct folders using this command,

#####Execute the command in **host** terminal#####

```
cp "${PWD}/organizations/peerOrganizations/org1.example.com/msp/config.yaml"
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp/config.yaml"
```

Now the crypto materials will be stored inside the msp folder of peer1.



Now, we need to enroll a TLS Server Certificate. Execute the following command.

#####Execute the command in **host** terminal#####

```
fabric-ca-client enroll -u https://peer1:peer1pw@localhost:7054 --caname
ca-org1 -M
```

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls" --enrollment.profile tls --csr.hosts peer1.org1.example.com --csr.hosts localhost --tls.certfiles "${PWD}/organizations/fabric-ca/org1/ca-cert.pem"
```

```
kba@Lab:~/CHF/Fabric-network$ fabric-ca-client enroll -u https://peer1:peer1pw@localhost:7054 --caname ca-org1 -M "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls" --enrollment.profile tls --csr.hosts peer1.org1.example.com --csr.hosts localhost --tls.certfiles "${PWD}/organizations/fabric-ca/org1/ca-cert.pem"
2023/08/04 12:03:12 [INFO] TLS Enabled
2023/08/04 12:03:12 [INFO] generating key: &{A:ecdsa S:256}
2023/08/04 12:03:12 [INFO] encoded CSR
2023/08/04 12:03:12 [INFO] Stored client certificate at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/signcerts/cert.pem
2023/08/04 12:03:12 [INFO] Stored TLS root CA certificate at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/tlscacerts/tls-localhost-7054-ca-org1.pem
2023/08/04 12:03:12 [INFO] Stored Issuer public key at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/IssuerPublicKey
2023/08/04 12:03:12 [INFO] Stored Issuer revocation public key at /home/kba/CHF/Fabric-network/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/IssuerRevocationPublicKey
```

Now organize the folders using the **cp** command. Execute the following commands.

#####Execute the command in **host** terminal#####

**cp**

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/tlscacerts/"*
```

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/ca.crt"
```

**cp**

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/signcerts/"*
```

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/server.crt"
```

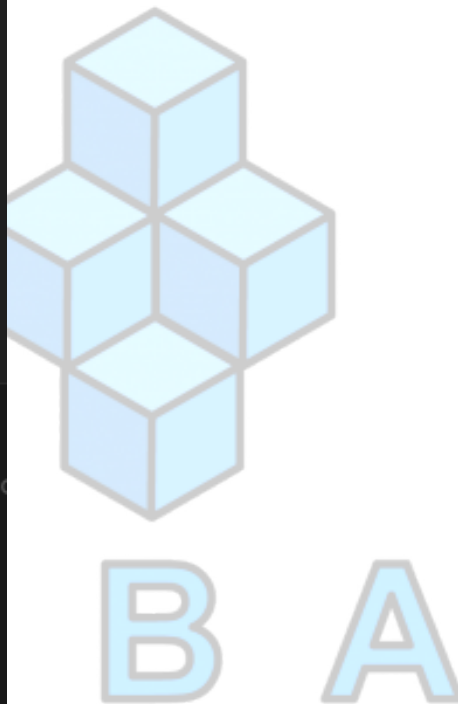
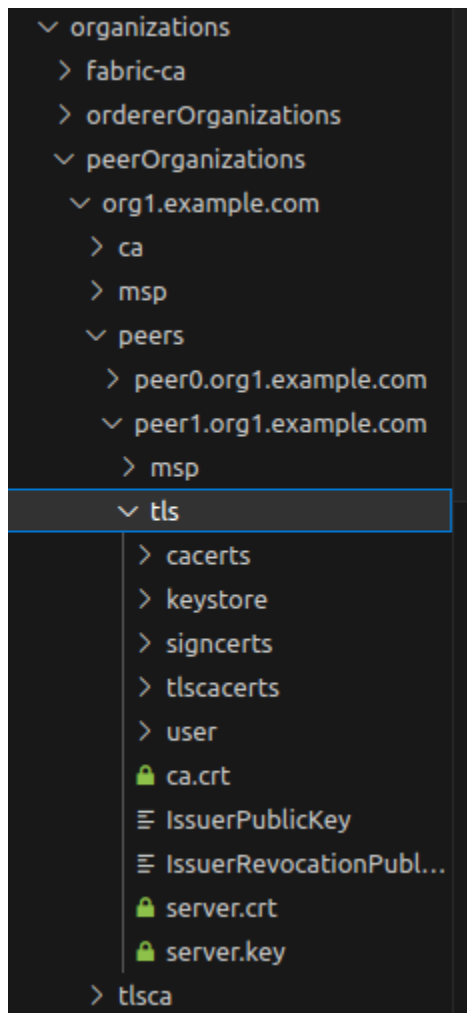
---

cp

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/keystore/"*
```

```
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/server.key"
```

Now we have the tls server certificate in the tls folder.



Once all the certificates and keys are copied, we can start creating the docker-compose file.

**Step2.** Creating docker-compose files for launching the new peer

Create a file named, **docker-compose-peer1org1.yaml** inside the docker folder. Add the following to the file.

```
version: '3.7'

volumes:
  peer1.org1.example.com:

networks:
  test:
    name: fabric_test

services:
  couchdb2:
    container_name: couchdb2
    image: couchdb:3.3.2
    labels:
      service: hyperledger-fabric
    environment:
      - COUCHDB_USER=admin
      - COUCHDB_PASSWORD=adminpw
    ports:
      - "6984:5984"
    networks:
      - test

  Peer1.org1.example.com:
    container_name: peer1.org1.example.com
    image: hyperledger/fabric-peer:latest
    labels:
      service: hyperledger-fabric
    environment:
      - FABRIC_LOGGING_SPEC=INFO
      #- FABRIC_LOGGING_SPEC=DEBUG
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=fabric_test
      - CORE_PEER_TLS_ENABLED=true
      - CORE_PEER_PROFILE_ENABLED=false
      - CORE_PEER_TLS_CERT_FILE=/etc/hyperledger/fabric/tls/server.crt
      - CORE_PEER_TLS_KEY_FILE=/etc/hyperledger/fabric/tls/server.key
      - CORE_PEER_TLS_ROOTCERT_FILE=/etc/hyperledger/fabric/tls/ca.crt
      # Peer specific variables
      - CORE_PEER_ID=peer1.org1.example.com
```

```

- CORE_PEER_ADDRESS=peer1.org1.example.com:8051
- CORE_PEER_LISTENADDRESS=0.0.0.0:8051
- CORE_PEER_CHAINCODEADDRESS=peer1.org1.example.com:8052
- CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:8052
- CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org1.example.com:8051
- CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.org1.example.com:8051
- CORE_PEER_LOCALMSPID=Org1MSP
- CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/fabric/msp
- CORE_OPERATIONS_LISTENADDRESS=peer1.org1.example.com:9447
- CORE_METRICS_PROVIDER=prometheus
- CHAINCODE_AS_A_SERVICE_BUILDER_CONFIG={"peername":"peer1org1"}
- CORE_CHAINCODE_EXECUTETIMEOUT=300s
- CORE_LEDGER_STATE_STATEDATABASE=CouchDB
- CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb2:5984
- CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=admin
- CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=adminpw
volumes:
- /var/run/docker.sock:/host/var/run/docker.sock
-
../organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com:/etc/hyperledger/fabric
- peer1.org1.example.com:/var/hyperledger/production
working_dir: /root
command: peer node start
ports:
- 8051:8051
- 9447:9447
depends_on:
- couchdb2
networks:
- test

```

Now we can try launching the docker-compose file using the following command.

#####Execute the command in **host** terminal#####

**docker-compose -f docker/docker-compose-peer1org1.yaml up -d**



```
kba@Lab:~/CHF/Fabric-network$ docker-compose -f docker/docker-compose-peer1org1.yaml up -d
Creating volume "docker_peer1.org1.example.com" with default driver
WARNING: Found orphan containers (ca_orderer, ca_org2, peer0.org2.example.com, orderer.example.com, peer0.org1.example.com, cli, couchdb1, couchdb0, ca_org1) for this project. If you removed or renamed this service in your compose file, you can run this command with the --remove-orphans flag to clean it up.
Creating couchdb2 ... done

Creating peer1.org1.example.com ... done
```

### Step3. Joining the peer to existing channel

Now that we have launched the new peer, we need to join the peer to the channel. So open another command terminal and set the environment variables accordingly. Let's mention this terminal as **peer1\_Org1 terminal**, because we are going to set the environment variables that refer to peer1.org1.example.com .

**Note:** Hereafter, if any commands need to be executed in this terminal, we will mention them as **peer1\_Org1 terminal**.

Here let's set the environment variables corresponding to peer1.org1.example.com. Execute the following commands.

#####Execute the command in **peer1\_Org1** terminal#####

**export FABRIC\_CFG\_PATH=./peercfg**

**export CHANNEL\_NAME=mychannel**

**export CORE\_PEER\_LOCALMSPID=Org1MSP**

**export CORE\_PEER\_TLS\_ENABLED=true**

**export**

**CORE\_PEER\_TLS\_ROOTCERT\_FILE=\${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/ca.crt**

---

```
export
```

```
CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
```

```
export CORE_PEER_ADDRESS=localhost:8051
```

```
export
```

```
ORDERER_CA=${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlsacerts/tlsca.example.com-cert.pem
```

Now we can execute the command for joining peer1.org1.example.com to the channel:

```
#####Execute the command in peer1_Org1 terminal#####
```

```
peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
```

```
kba@Lab:~/CHF/Fabric-network$ peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
2023-08-04 12:08:06.769 IST 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
2023-08-04 12:08:09.004 IST 0002 INFO [channelCmd] executeJoin -> Successfully submitted proposal to join channel
```

Now, peer1.org1.example.com joined to mychannel. Confirm it by the **peer channel list** command.

```
#####Execute the command in peer1_Org1 terminal#####
```

```
peer channel list
```

```
kba@Lab:~/CHF/Fabric-network$ peer channel list
2023-08-04 12:09:19.884 IST 0001 INFO [channelCmd] InitCmdFactory -> Endorser and orderer connections initialized
Channels peers has joined:
mychannel
```

**Step4.** Installing chaincode to the new peer

Install Chaincode on peer1.org1.example.com . Execute the following command.

---

#####Execute the command in **peer1\_Org1** terminal#####

**peer lifecycle chaincode install basic.tar.gz**

```
kba@Lab:~/CHF/Fabric-network$ peer lifecycle chaincode install basic.tar.gz
2023-08-04 12:10:53.851 IST 0001 INFO [cli.lifecycle.chaincode] submitInstallProp
osal -> Installed remotely: response:<status:200 payload:"\nJbasic_1.0:d3ced865d6
5bb0db0980f7c27e023d7c8c7be01ebca4b37642b1447dd5873818\022\tbasic_1.0" >
2023-08-04 12:10:53.852 IST 0002 INFO [cli.lifecycle.chaincode] submitInstallProp
osal -> Chaincode code package identifier: basic_1.0:d3ced865d65bb0db0980f7c27e02
3d7c8c7be01ebca4b37642b1447dd5873818
```

Check the chaincode installed:

#####Execute the command in **peer1\_Org1** terminal#####

**peer lifecycle chaincode queryinstalled**

```
kba@Lab:~/CHF/Fabric-network$ peer lifecycle chaincode queryinstalled
Installed chaincodes on peer:
Package ID: basic_1.0:d3ced865d65bb0db0980f7c27e023d7c8c7be01ebca4b37642b1447dd58
73818, Label: basic_1.0
```

Now let's try invoking from the newly installed peer. Execute the following command.newly

#####Execute the command in **peer1\_Org1** terminal#####

**peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride  
orderer.example.com --tls --cafile \$ORDERER\_CA -C \$CHANNEL\_NAME -n basic  
--peerAddresses localhost:8051 --tlsRootCertFiles  
"\${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org  
1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles  
"\${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org  
2.example.com/tls/ca.crt" -c  
'{"function": "CreateAsset", "Args": ["asset7", "blue", "15", "Jack", "600"]}'**

```
kba@Lab:~/CHF/Fabric-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile $ORDERER_CA -C $CHANNEL_NAME -n basic --peerAddresses localhost:8051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"createAsset","Args":["asset7","blue","15","Jack","600"]}'
2023-08-04 12:12:35.490 IST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200 payload:{"ID":"asset7","Color":"blue","Size":"15","Owner":"Jack","AppraisedValue":"600"}
```

Now let's try querying from the newly installed peer. Execute the following command.

#####Execute the command in **peer1\_Org1** terminal#####

```
peer chaincode query -C $CHANNEL_NAME -n basic -c '{"function":"ReadAsset","Args":["asset7"]}'
```

```
kba@Lab:~/CHF/Fabric-network$ peer chaincode query -C $CHANNEL_NAME -n basic -c '{"function":"ReadAsset","Args":["asset7"]}'
{"AppraisedValue":"600","Color":"blue","ID":"asset7","Owner":"Jack","Size":"15"}
```

Now we have successfully added a new peer to the existing organization, successfully submitted an asset to the channel and queried it.

## Bring down the network

Execute the following commands to stop the network.

Note: Execute the following commands within the **Fabric-network** folder.

#####Execute the command in **host** terminal#####

```
docker-compose -f docker/docker-compose-2org.yaml down
```

```
docker-compose -f docker/docker-compose-ca.yaml down
```

```
docker-compose -f docker/docker-compose-peer1org1.yaml down
```

```
docker volume rm $(docker volume ls -q)
```

Remove all the crypto material generated for the network.

---

#####Execute the command in **host** terminal#####

**sudo rm -rf channel-artifacts/**

**sudo rm -rf organizations/**

**sudo rm basic.tar.gz**

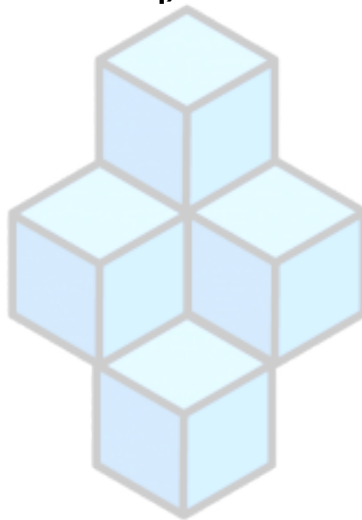
Now, when you try the **docker ps** command, if any containers are listed, then execute the following commands.

**docker rm \$(docker container ls -q) --force**

**docker system prune**

**docker volume prune**

**docker network prune**



**K B A**