Set up Hyperledger Test Network

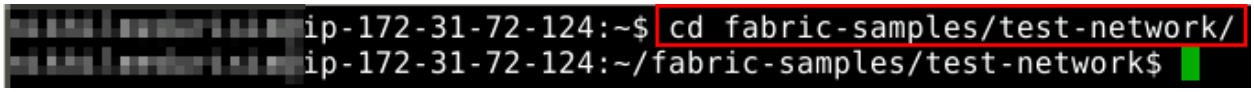
Steps to be followed:

1. Setting up the standard Hyperledger test network
2. Setting up a test network with CA containers
3. Setting up a test network with CouchDB containers
4. Setting up a test network with all the above parameters

Step 1: Setting up the standard Hyperledger test network

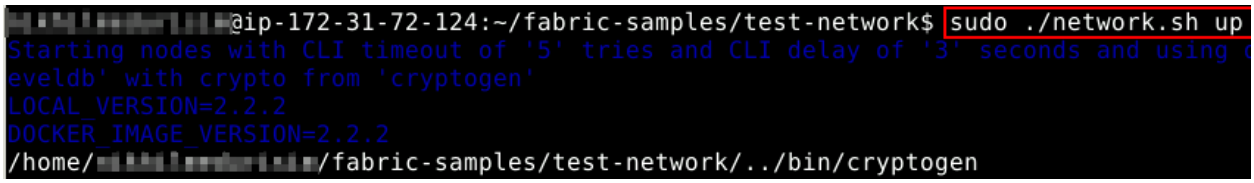
1.1 Navigate to the **test-network** folder by using the following command:

cd fabric-samples/test-network



1.2 We may start the test network by executing the following command:

sudo ./network.sh up



```
Generating certificates using cryptogen tool
Creating Org1 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org1.yaml --output=organi
zations
org1.example.com
+ res=0
Creating Org2 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org2.yaml --output=organi
zations
org2.example.com
+ res=0
Creating Orderer Org Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-orderer.yaml --output=org
anizations
+ res=0
Generating CCP files for Org1 and Org2
/home/nikhilmedurisim/fabric-samples/test-network/../bin/configtxgen
Generating Orderer Genesis block
+ configtxgen -profile TwoOrgsOrdererGenesis -channelID system-channel -outputBlock ./system-ge
nesis-block/genesis.block
2021-05-09 12:23:14.489 UTC [common.tools.configtxgen] main -> INFO 001 Loading configuration
2021-05-09 12:23:14.513 UTC [common.tools.configtxgen.localconfig] completeInitialization -> IN
FO 002 orderer type: etcdraft
2021-05-09 12:23:14.513 UTC [common.tools.configtxgen.localconfig] completeInitialization -> IN
```

```
2021-05-09 12:23:14.515 UTC [common.tools.configtxgen] doOutputBlock -> INFO 006 Writing genesi
s block
+ res=0
Recreating peer0.org1.example.com ... done
Recreating orderer.example.com    ... done
Recreating peer0.org2.example.com ... done
Recreating cli                    ... done
CONTAINER ID   IMAGE                               COMMAND             CREATED
 STATUS                  PORTS                                                 NAMES
efaec78ac0e2   hyperledger/fabric-tools:latest     "/bin/bash"         Less than a second ago
 Up Less than a second                                                         cli
14cb4c1dd37b   hyperledger/fabric-orderer:latest   "orderer"           2 seconds ago
 Up 1 second             0.0.0.0:7050->7050/tcp, :::7050->7050/tcp             orderer.example.
com
42d575381bf3   hyperledger/fabric-peer:latest      "peer node start"   2 seconds ago
 Up Less than a second   7051/tcp, 0.0.0.0:9051->9051/tcp, :::9051->9051/tcp   peer0.org2.examp
le.com
f3e6b3768e4f   hyperledger/fabric-peer:latest      "peer node start"   2 seconds ago
 Up 1 second             0.0.0.0:7051->7051/tcp, :::7051->7051/tcp             peer0.org1.examp
le.com
2013de4b4d52   hello-world                         "/hello"            2 days ago
 Exited (0) 2 days ago                                                         magical_mestorf
```

1.3 We can also check the Docker images running the peers in our test network by executing the following command:

sudo docker ps

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo docker ps
CONTAINER ID   IMAGE                               COMMAND             CREATED         STATUS
     PORTS                                                 NAMES
efaec78ac0e2   hyperledger/fabric-tools:latest     "/bin/bash"         8 minutes ago   Up 8 minut
es                                                         cli
42d575381bf3   hyperledger/fabric-peer:latest      "peer node start"   8 minutes ago   Up 8 minut
es   7051/tcp, 0.0.0.0:9051->9051/tcp, :::9051->9051/tcp   peer0.org2.example.com
f3e6b3768e4f   hyperledger/fabric-peer:latest      "peer node start"   8 minutes ago   Up 8 minut
es   0.0.0.0:7051->7051/tcp, :::7051->7051/tcp             peer0.org1.example.com
```

1.4 We can check the communication channel for this test network by executing the

following command:

docker exec peer0.org1.example.com peer channel list

```
[redacted]ip-172-31-72-124:~/fabric-samples/test-network$ docker exec peer0.org1.example.
com peer channel list
2021-05-09 12:34:49.591 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connec
tions initialized
Channels peers has joined:
[redacted]p-172-31-72-124:~/fabric-samples/test-network$
```

> **Note:** Since there are no active channels, an empty list is displayed.

1.5 We will create a communication channel for the peers in the test network using the following command:

sudo ./network.sh createChannel -c testchannel

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh createChannel
 -c testchannel
Creating channel 'testchannel'.
If network is not up, starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds
 and using database 'leveldb
Generating channel create transaction 'testchannel.tx'
+ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/testchannel.tx
 -channelID testchannel
```

```
Channel 'testchannel' created
Joining org1 peer to the channel...
Using organization 1
+ peer channel join -b ./channel-artifacts/testchannel.block
+ res=0
2021-05-09 12:41:47.351 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connec
tions initialized
2021-05-09 12:41:47.389 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposa
l to join channel
Joining org2 peer to the channel...
Using organization 2
+ peer channel join -b ./channel-artifacts/testchannel.block
+ res=0
2021-05-09 12:41:50.432 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connec
tions initialized
2021-05-09 12:41:50.465 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposa
l to join channel
Setting anchor peer for org1...
Using organization 1
Fetching channel config for channel testchannel
Using organization 1
Fetching the most recent configuration block for the channel
+ peer channel fetch config config_block.pb -o orderer.example.com:7050 --ordererTLSHostnameOve
rride orderer.example.com -c testchannel --tls --cafile /opt/gopath/src/github.com/hyperledger/
```

> **Note:** The **-c** parameter allows us to name the channel. In this case it is **testchannel**.

1.6 Now if we check for the communication channel again, we will see the channel created.

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ docker exec peer0.org1.example.com peer channel list
2021-05-09 12:47:49.242 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
Channels peers has joined:
testchannel
```

1.7 After we are done experimenting with the test network, we must turn it off using the following command:

sudo ./network.sh down

```
[redacted]ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh down
Stopping network
Stopping cli                    ... done
Stopping peer0.org2.example.com ... done
Stopping orderer.example.com    ... done
Stopping peer0.org1.example.com ... done
Removing cli                    ... done
Removing peer0.org2.example.com ... done
Removing orderer.example.com    ... done
Removing peer0.org1.example.com ... done
Removing network net_test
Removing volume net_orderer.example.com
Removing volume net_peer0.org1.example.com
Removing volume net_peer0.org2.example.com
Removing network net_test
WARNING: Network net_test not found.
Removing volume net_peer0.org3.example.com
WARNING: Volume net_peer0.org3.example.com not found.
No containers available for deletion
No images available for deletion
```

Step 2: Setting up a test network with CA containers

2.1 We may start a test network with CA containers using the following command:

sudo ./network.sh up -ca

```
[redacted]ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh up -ca
Starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using database 'leveldb' with crypto from 'Certificate Authorities'
LOCAL_VERSION=2.2.2
DOCKER_IMAGE_VERSION=2.2.2
CA_LOCAL_VERSION=1.4.9
CA_DOCKER_IMAGE_VERSION=1.4.9
Generating certificates using Fabric CA
Creating network "net_test" with the default driver
Creating ca_org1    ... done
Creating ca_orderer ... done
Creating ca_org2    ... done
Creating Org1 Identities
Enrolling the CA admin
```

```
Creating volume "net_orderer.example.com" with default driver
Creating volume "net_peer0.org1.example.com" with default driver
Creating volume "net_peer0.org2.example.com" with default driver
WARNING: Found orphan containers (ca_orderer, ca_org2, ca_org1) for this project. If you remove
d or renamed this service in your compose file, you can run this command with the --remove-orph
ans flag to clean it up.
Creating orderer.example.com    ... done
Creating peer0.org2.example.com ... done
Creating peer0.org1.example.com ... done
Creating cli                    ... done
CONTAINER ID   IMAGE                               COMMAND                  CREATED         STA
TUS                 PORTS                                                 NAMES
afe7aa5f8342   hyperledger/fabric-tools:latest     "/bin/bash"              1 second ago    Up
Less than a second                                                        cli
ddcfc56fa59d   hyperledger/fabric-peer:latest      "peer node start"        2 seconds ago   Up
Less than a second   0.0.0.0:7051->7051/tcp, :::7051->7051/tcp            peer0.org1.example.c
om
e3fa13fb2c8b   hyperledger/fabric-peer:latest      "peer node start"        2 seconds ago   Up
Less than a second   7051/tcp, 0.0.0.0:9051->9051/tcp, :::9051->9051/tcp   peer0.org2.example.c
om
```

2.2 We can create a communication channel for this network by executing the following command:

sudo ./network.sh createChannel -c testchannel1

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh createChannel
 -c testchannel1
Creating channel 'testchannel1'.
If network is not up, starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds
 and using database 'leveldb
Generating channel create transaction 'testchannel1.tx'
+ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/testchannel1.t
x -channelID testchannel1
2021-05-09 12:58:49.376 UTC [common.tools.configtxgen] main -> INFO 001 Loading configuration
2021-05-09 12:58:49.393 UTC [common.tools.configtxgen.localconfig] Load -> INFO 002 Loaded conf
iguration: /home/nikhilmedurisim/fabric-samples/test-network/configtx/configtx.yaml
2021-05-09 12:58:49.393 UTC [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 003 Gene
rating new channel configtx
2021-05-09 12:58:49.395 UTC [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 004 Writ
ing new channel tx
+ res=0
Creating channel testchannel1
Using organization 1
```

2.3 We may confirm the creation of a test channel using the following command:

docker exec peer0.org1.example.com peer channel list

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ docker exec peer0.org1.example.
com peer channel list
2021-05-09 13:02:00.500 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connec
tions initialized
Channels peers has joined:
testchannel1
```

Note: The commands to check Docker containers, create a communication channel, and turn off the network remain the same (as shown in **Step 1**) across all the other steps.

Step 3: Setting up a test network with CouchDB containers

3.1 We may start a test network with CouchDB containers using the following command:

sudo ./network.sh up -s couchdb

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh up -s
couchdb
Starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using dat
abase 'couchdb' with crypto from 'cryptogen'
LOCAL_VERSION=2.2.2
DOCKER_IMAGE_VERSION=2.2.2
/home/[redacted]/fabric-samples/test-network/../bin/cryptogen
Generating certificates using cryptogen tool
Creating Org1 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org1.yaml --outpu
t=organizations
org1.example.com
+ res=0
```

```
Digest: sha256:fe85e61b62eec34a0fe7bcc4b85a4c876d733587102edb1a3fa4d77ae74bb436
Status: Downloaded newer image for couchdb:3.1.1
Creating orderer.example.com    ... done
Creating couchdb1               ... done
Creating couchdb0               ... done
Creating peer0.org2.example.com ... done
Creating peer0.org1.example.com ... done
Creating cli                    ... done
CONTAINER ID   IMAGE                               COMMAND                  CREATED
              STATUS                  PORTS
               NAMES
988269e8df84   hyperledger/fabric-tools:latest     "/bin/bash"              Less than a
 second ago   Up Less than a second
               cli
41ae1c6a9fb3   hyperledger/fabric-peer:latest      "peer node start"        1 second ag
o             Up Less than a second   0.0.0.0:7051->7051/tcp, :::7051->7051/tcp
               peer0.org1.example.com
ee3a8a315268   hyperledger/fabric-peer:latest      "peer node start"        2 seconds a
go            Up 1 second             7051/tcp, 0.0.0.0:9051->9051/tcp, :::9051->9051/t
cp             peer0.org2.example.com
ac3f62a7c3c9   couchdb:3.1.1                       "tini -- /docker-ent…"   3 seconds a
go            Up 2 seconds            4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp, :::79
84->5984/tcp   couchdb1
0b9e225958d6   couchdb:3.1.1                       "tini -- /docker-ent…"   3 seconds a
```

3.2 We can create a communication channel for this network by executing the following command:

sudo ./network.sh createChannel -c testchannel2

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh creat
eChannel -c testchannel2
Creating channel 'testchannel2'.
If network is not up, starting nodes with CLI timeout of '5' tries and CLI delay of '3'
 seconds and using database 'leveldb
Generating channel create transaction 'testchannel2.tx'
+ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/testch
annel2.tx -channelID testchannel2
```

3.3 We can turn off the network after our testing using the following command:

sudo ./network.sh down

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-network$ sudo ./network.sh down
Stopping network
Stopping cli                    ... done
Stopping peer0.org1.example.com ... done
Stopping peer0.org2.example.com ... done
Stopping couchdb1               ... done
Stopping couchdb0               ... done
Stopping orderer.example.com    ... done
Removing cli                    ... done
Removing peer0.org1.example.com ... done
Removing peer0.org2.example.com ... done
Removing couchdb1               ... done
Removing couchdb0               ... done
Removing orderer.example.com    ... done
```

Step 4: Setting up a test network with all the above parameters

4.1 We may start a test network with all the above parameters and containers using the following command:

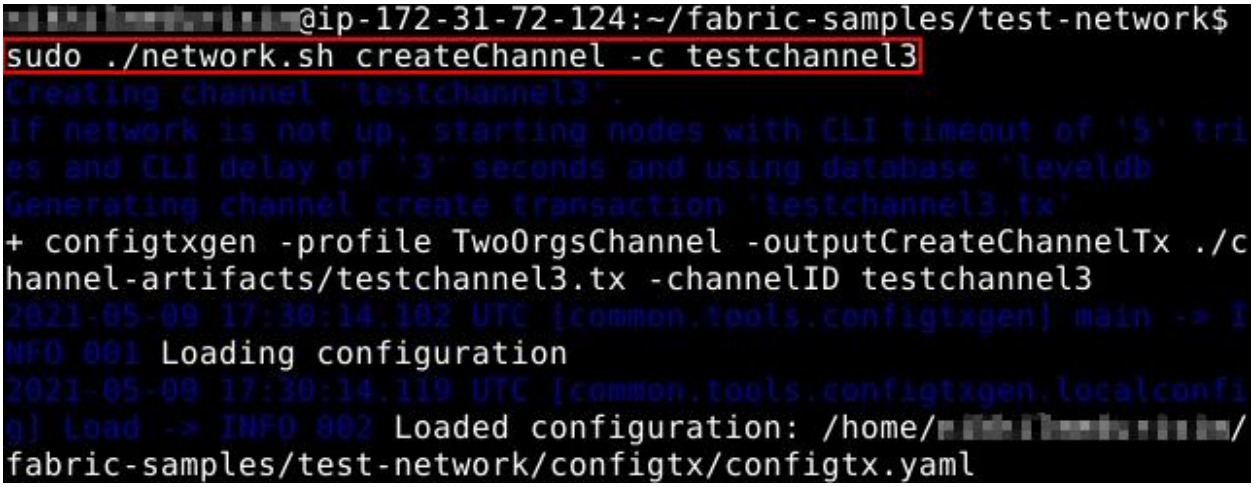
sudo ./network.sh up -ca -s couchdb

```
[redacted]@ip-172-31-72-124:~/fabric-samples/test-networ
k$ sudo ./network.sh up -ca -s couchdb
Starting nodes with CLI timeout of '5' tries and CLI delay of
 '3' seconds and using database 'couchdb' with crypto from 'C
ertificate Authorities'
LOCAL_VERSION=2.2.2
DOCKER_IMAGE_VERSION=2.2.2
CA_LOCAL_VERSION=1.4.9
CA_DOCKER_IMAGE_VERSION=1.4.9
Generating certificates using Fabric CA
Creating network "net_test" with the default driver
Creating ca_org1    ... done
Creating ca_org2    ... done
Creating ca_orderer ... done
Creating Org1 Identities
Enrolling the CA admin
```

4.2 We can create a communication channel for this network by executing the following

command:

sudo ./network.sh createChannel -c testchannel3



4.3 We can turn off the network after our testing using the following command:

sudo ./network.sh down