**Introduction.**

In this information reading, we will build a simple blockchain network working with hyper-ledger fabric. The fabric allows us to construct/create a private blockchain.No cryptocurrencies will be involved in this demonstration. After we finish this run-through, you should understand the different parts needed to start your private blockchain. The example project will contain our network and will be composed of 3 organizations. Each organization will have two peers in each of them.

**Pre-requisites**

Installing the following **docker** and **Docker-compose** installed in your local machine. To make sure you already have docker currently on your device, you can run **docker --v** in your terminal. This message should return the version. If not **$ sudo apt-get remove docker docker-engine docker.io containerd runc** this command removes the old docker version and then following **$ sudo apt-get update**

**$ sudo apt-get install docker-ce docker-ce-cli containerd.io** for the update to the correct version.

**Insert Docker Version Image here 3**

Fabric provides us with sample networks to play around with and explore. To get these files open up the terminal window and run **$git clone -b master https://github.com/hyperledger/fabric-samples.git.** This will make a **fabric-samples** folder with some fabric samples inside. There is a ton of helpful information in these folders, but we won't explore them here in this specific section.

Moving forward, we will download the platform-specific binaries. This will export some binaries that we will need to check in with later. Move into the **fabric-samples** folder with **cd fabric-samples** and run:

**"curl -sSL https://goo.gl/6wtTN5 | bash -s 1.1.0"**

This will create a bin folder in your **fabric-samples** folder.

That is it! Now we are ready to build the network.

**Cryptogen**

Inside your **fabric-samples** folder, create a new folder called **Smooth-Network** this is where all the critical functions will happen then because we will use the version. **18.03.1-Ce build 9ee9f40** let us do a git checkout **v18.03.01** to ensure no mistakes because of the compiler version.

**Generate our keys & certificates and keys**

Our first step will be to generate all the certificates and keys for the organizations and peers inside them. In our network, everyone will have to prove who they claim to be. The certificates we will generate will prove their identities and allow them to sign and verify transactions. Transactions will be signed by an entity's private key and verified by a public one.

To generate these materials, we will use the **cryptogen** tool. We got this tool by downloading the platform-specific binaries ( you have a file called cryptogen in the bin folder ). The **cryptogen** tool consumes a crypto-config.yaml file. So, inside our **Smooth-network** folder, go ahead and create this file. You can copy and paste the following inside, and I'll explain what is going on:

Insert Image. 4

In Fabric, the Ordering Service is a collection of network entities. The orderers make sure that the transactions are dealt with in the right order ( first-come-first-serve ). The orderer is then responsible for packaging transactions into blocks.

So, what do we have in this file? We first define the Orderer Service. We give it a name, a domain, and a hostname under the specs key. The naming convention goes like this: {Hostname}.{Domain}. So, our orderer will be named *orderer.cool-network.com*.

Next, we define our 3 organizations. We describe their names and domains. The key *EnableNodeOUs* is set to true. This is outside the scope of this article but setting this to true will allow us to define additional policies for the network members. Finally, we have a Count key set to 1 for each organization. This means that there will be two peers per organization. Why? Because each organization will have an Admin by default. So, the number of peers in an organization will be [Count + 1].

**Running the cryptogen tool**

Everything is now in place for us to generate our certificates and keys. Make sure you are inside the Smooth-network folder and run:

../bin/cryptogen generate --config crypto-config.yaml --output=crypto-config

The *cryptogen* tool has a generate method that will do the job for us. We provide a config file and an output folder. You should see the following in your terminal:

Insert crypto image five here

And a crypto-config file just appeared in your Smooth-network folder! This is where all the certificates and keys for the orderers, organizations, and peers are stored.

**Configtxgen**

Next, we will use the configtxgen tool. This tool will do three important things:

* create the orderer genesis block
* create the channel configuration transaction
* create the anchor peer transactions, one for each peer organization

**Channel**: In Fabric, you have the possibility to create channels. These channels allow organizations to create private sub-networks inside the greater network. In our case, we could imagine a channel that only Nvidia and AMD would be a part of, and another that only AMD and B2B would be a part of.

**Anchor peer**: An anchor peer on a channel is a public peer that all others peers can discover and communicate with. Each organization on a channel has an anchor peer.

### Creating the file

The configtxgen tool consumes a *configtx.yaml* file. Create it and copy-paste the following:

Insert Image 6 here

There are a few different sections here. The most important Profiles and Organizations for now. In Profiles, we describe the orderer genesis block, our channel ( with the three organisations ) and we also defined a Consortium. Pay attention to the headers ( OrdererGenesis and channelthreeorgs) as they will be used later. In Organizations, we specify where the certificates are stored for each organization. We just created these certificates with *cryptogen*. We also specify the anchor peers for each organization. ( MSP means Membership Service Provider )

The *order type* key in the *Orderer* section is set to *solo*. Because we are in a development environment, we have only one orderer. If we were in a production environment, we would define several orderers if one or several of them were faulty. For now, this will do just fine.

### Running configtxgen

First, create a folder called channel artifacts to hold our artifacts.

We can now run the *configtxgen* tool. Run the following command:

Notice the *OrdererGenesis* header we specified earlier. This will create a *genesis. Block* file for our orderer genesis block. . You should see the following in your terminal:

Next, we must run a command to create our channel configuration transaction:

Notice the *channelthreeorgs* header we specified earlier. This will create a channel.tx file, and this should appear in your terminal:

Next, we must create our anchor peers' transactions. We will have three commands to run, one for each organization...For AMD

Notice we use the same channel header, and we use the ID we specified earlier (NvidiaMSP). We specify a output file ( here NvidiaAnchor.tx)

This should appear in your terminal:

Great! Now run the following two commands for AMD and B2B, respectively:

## Creating peer-base and docker-compose files

We are almost there. We now need to specify a few docker files to tell our network where to find some images, how to be configured...

First, create a *base* folder. In this folder, create two files, one called *docker-compose-base.yaml* and another called peer-base.yaml.

*peer-base.yaml*

These files set up the base for our network, with some fabric images along with some environments variables.

Finally, the last file we will add is a *docker-compose-cli.yaml* file. This file will be in the root of our Smooth-network project. The file creates a cli container and bootstrap all of our organisations declared in our base folder.

*docker-compose-cli.yaml*

Great, now we can run docker-compose -f docker-compose-cli.yaml up -d. You should see something like this is your terminal:

Now, we can run our cli container with docker start cli and enter inside the container with docker exec -it cli bash:

Our network is now running!! You can explore the different folders. You will find all the certificates, variables, and everything that we defined in our yaml files.

Conclusion.

I hope this tutorial was clear enough for you to understand how to set up a Hyperledger Fabric environment. Of course, for now, this network can't do much without any chain code ( or smart contracts ).