The Chinese University of Hong Kong

Department of Financial Technology

FTEC5520 –Applied Blockchain & Cryptocurrency

Lab2-part1-report

Hyperledger Fabric Setup & Practice on AWS(Task1,2,3)

# 1 Questions Answering:

There will be 3 questions only for **task1,2 and 3**.

**Q1:** Basic concepts and knowledge about Hyperledger Fabric. (2 mark)

1. What’s the difference between member and peer node in Hyperledger fabric?

*A peer is a member of the blockchain and is running Hyperledger Fabric. A legally separate entity that owns a unique root certificate for the network. Network components such as peer nodes and application clients will be linked to a member.*

1. What’s the channel in Hyperledger Fabric? Why do we need it?

*Channels allow a specific set of peers and applications to communicate with each other within a blockchain network.In Hyperledger Fabric, each channel has a completely separate ledger. This means a completely separate blockchain, and completely separate world states, including namespaces. It is possible for applications and smart contracts to communicate between channels so that ledger information can be accessed between them.*

1. What’s chaincode in Hyperledger Fabric? What did you do in task 1~3 about chaincode?

*Fabric implements smart contracts with a technology concept it calls chaincode — simply a piece of code that accesses the ledger, written in one of the supported programming languages. In task 1~3，I create a fabric channel and deploy the chaincode, install chaincode on peer node, instantiate the chaincode on the channel, and also query the chaincode on peer.*

1. What’s Membership Service Provider (MSP)? Why do we need it? What did you do about this component in task1~3?

*The implementation of the MSP requirement is a set of folders that are added to the configuration of the network and is used to define an organization both inwardly and outwardly. MSP contains a list of permissioned identities.The MSP identifies which Root CAs and Intermediate CAs are accepted to define the members of a trust domain by listing the identities of their members, or by identifying which CAs are authorized to issue valid identities for their members. We need MSP because Fabric is a permissioned network, blockchain participants need a way to prove their identity to the rest of the network in order to transact on the network. In task1~3 when a peer connects to a channel, its digital certificate identifies its owning organization via a channel MSP.*

**Hint:** You may refer to this [chapter of official docs of Hyperledger Fabric](https://hyperledger-fabric.readthedocs.io/en/release-1.4/key_concepts.html).

**Q2:** Please read step 1 ~ 7 of the [AmazonManagedBlockchain official documents](https://docs.aws.amazon.com/managed-blockchain/latest/managementguide/managed-blockchain-get-started-tutorial.html) and answer the following questions. (2 mark)

**Note:** Please describe the procedure in your own words and list the commands.

1)How to create blockchain network, member and node using AWS CLI instead of GUI?

*Use the create-network command and set the user name and password.The command returns the Network ID and the Member ID. Use the list-networks command to confirm the network status.The command returns information about the network, including an available status. To create a node, Use the create-node command and set the value of --network-id, --member-id appropriately.*

2)What’s the CAEndpoint of here? How can we get the value in AmazonManagedBlockchain service dashboard? How to enroll an Admin User with admin permissions to your member’s CA? **Hint:** fabric-ca-client enroll

3)How to create a channel and join your peer node to the channel your created?

*1.Create configtx. Use a text editor to create a file with the following contents and save it as configtx.yaml. Replace memberID, set MSPDir to the same directory location.*

*2.Set environment variables. Use export ORDERER command and add the export statement to ~/.bash\_profile. Apply the changes by using source ~/.bash\_profile command.*

*3.Create the channel. Run the command:*

*docker exec cli peer channel create -c mychannel -f /opt/home/mychannel.pb -o $ORDERER --cafile /opt/home/managedblockchain-tls-chain.pem --tls*

*4.Join peer node to the channel.Run the command :*

*docker exec cli peer channel join -b mychannel.block -o $ORDERER --cafile /opt/home/managedblockchain-tls-chain.pem --tls*

4)How to install, instantiate, query and invoke the Chaincode? Please use the example we used in task3.

**Q3:** What’s the difference between Hyperledger Fabric and Ethereum after you learnt about both the two frameworks including the hands-on lab sessions? (1 mark)

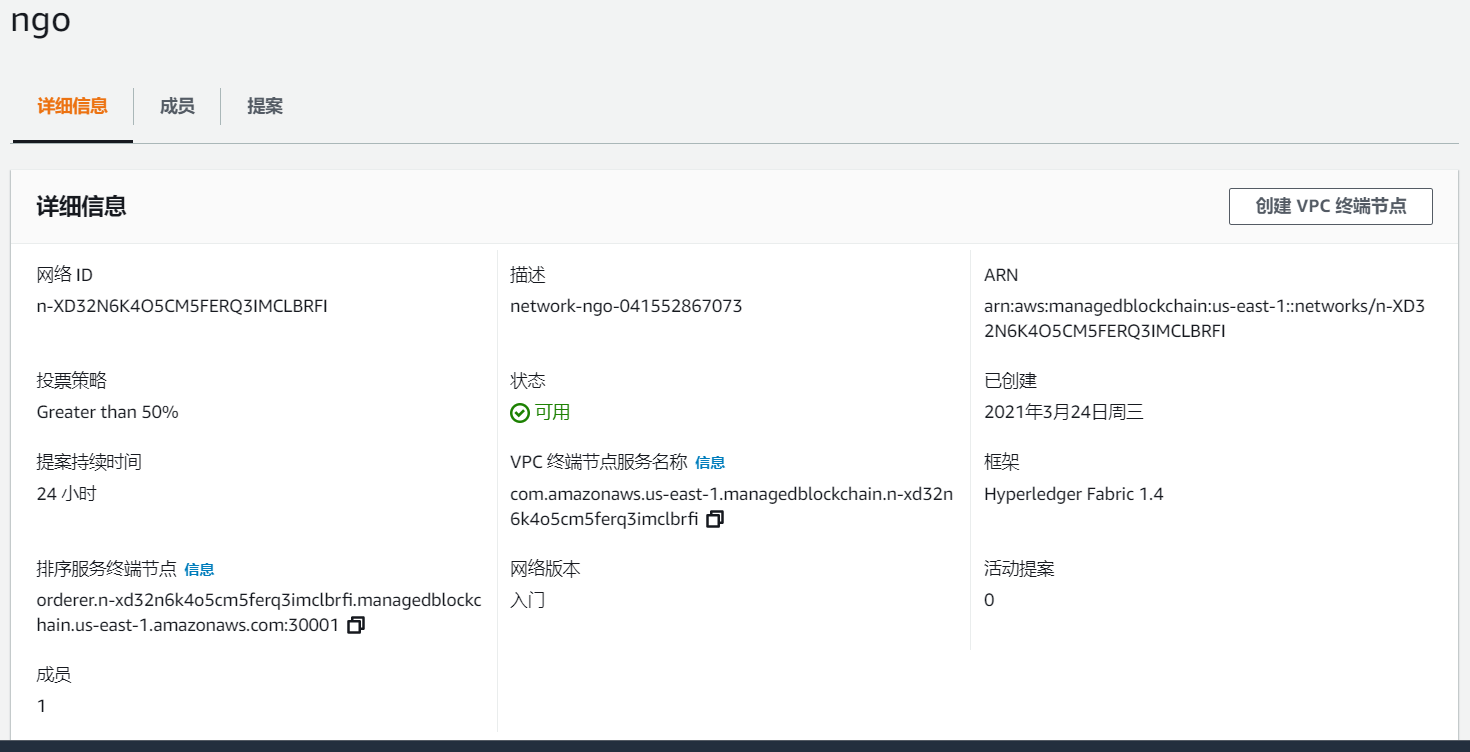
**Note:** Detailed answer expected but no more than 1 page. Please refer to the lecture slides the [article-1](https://medium.com/@philippsandner/comparison-of-ethereum-hyperledger-fabric-and-corda-21c1bb9442f6), [article-2](https://medium.com/quillhash/ethereum-or-hyperledger-fabric-259f3c9b8da6) and try to merge some contents of previous questions if necessary.

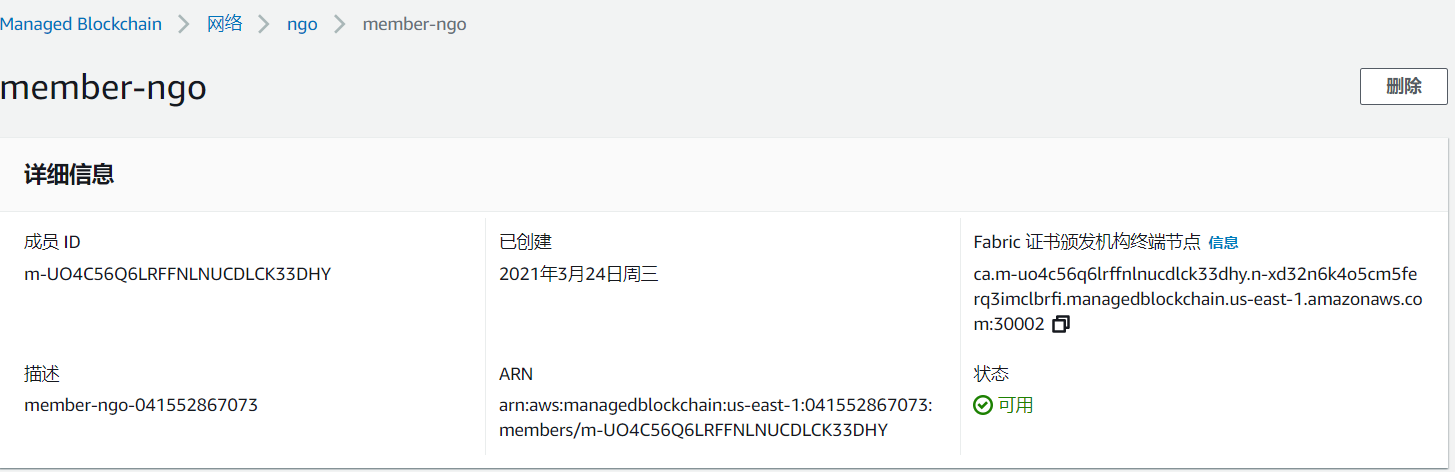
1. *Confidentiality aspect. As Ethereum is a public network it won’t require permissions before use. It is highly transparent and all the transactions are available on the Blockchain network which can be accessed by all the people on the network. Hyperledger does not allow access to everyone. It is secure as it allows permission to only authorised people to view transactions made on the network.*
2. *Accessibility. You don’t require prior permission to access transaction details on Ethereum as it is a public blockchain platform. Anyone can mine Ether after downloading Ethereum. However, Hyperledger has strict control in place when it comes to accessibility. Only permitted members are allowed to view the transactions made on the platform. The permission can be obtained before joining the Hyperledger network.*
3. *Purpose. Ethereum can function on Smart Contracts on the EVM for apps since they are decentralised and accessible to the public. Hyperledger leverages blockchain technology and can offer support for pluggable implementations of various components. This helps in delivering greater levels of confidentiality, scalability, and flexibility.*
4. *Consensus mechanism. Hyperledger fabric offers ample opportunities to users to leverage the consensus mechanisms. The platform uses* *Kafka consensus algorithm natively. It also comes equipped with Solo and Raft protocols that work perfectly for developers. On the other hand, the Ethereum consensus mechanism leverages the latest versions of the Proof of Work algorithm. It is considered to be powerful and efficient, however, it can slow down the process.*
5. *Cryptocurrency aspect.Ether is the built-in crypto-currency of Ethereum and every participant can access it. Hyperledger does not have an in-built cryptocurrency. This helps in dealing with scalability issues of the network and allows it to manage high transaction rates thereby automating the process throughout the network.*

# 2 Screen Capture of Main Steps:

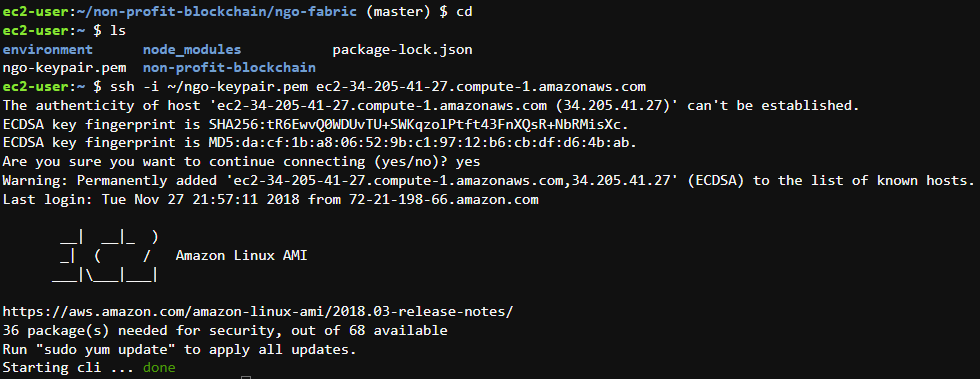
Your screenshots must include these parts at least and detailed description of each screenshots:

1. Your Hyperledger Fabric blockchain network on AmazonManagedBlockchain dashboard:

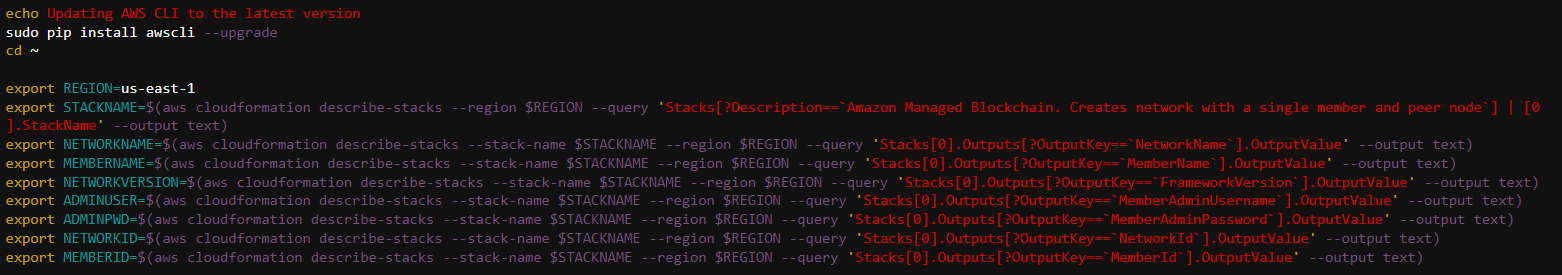




1. SSH from the Cloud9 IDE to peer node. You should know they are two VMs.

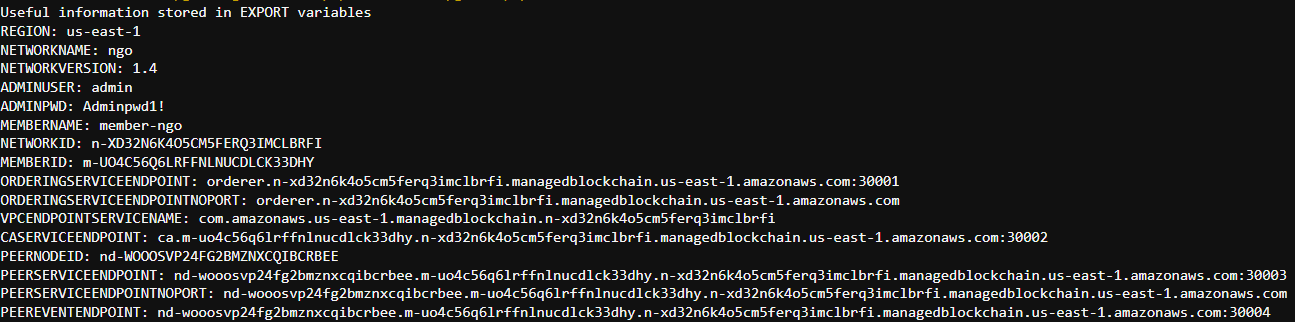


1. The content of **fabric-exports.sh** after insert your blockchain network information



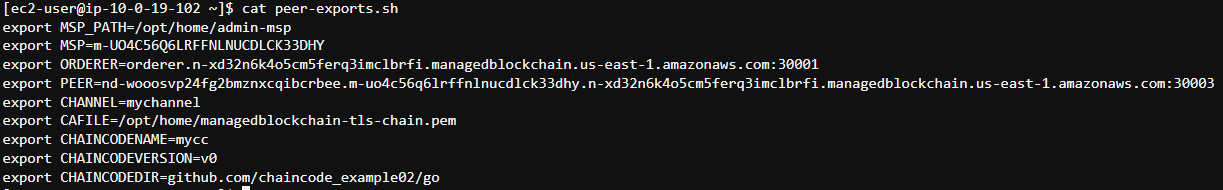
1. The useful information after you ran the command in task3 step 2.16

source fabric-exports.sh



1. Use this command to check the peer export information:

cat ~/peer-exports.sh



1. List the generated configuration using

ls -lt ~/$CHANNEL.pb



1. List the mychannel.block generated fabric channel in task3 step 4.1

ls -lt /home/ec2-user/fabric-samples/chaincode/hyperledger/fabric/peer



1. Query the chaincode on peer in the task3 step 8

docker exec -e "CORE\_PEER\_TLS\_ENABLED=true" -e "CORE\_PEER\_TLS\_ROOTCERT\_FILE=/opt/home/managedblockchain-tls-chain.pem" -e "CORE\_PEER\_ADDRESS=$PEER" -e "CORE\_PEER\_LOCALMSPID=$MSP" -e "CORE\_PEER\_MSPCONFIGPATH=$MSP\_PATH" cli peer chaincode query -C $CHANNEL -n $CHAINCODENAME -c '{"Args":["query","a"]}'



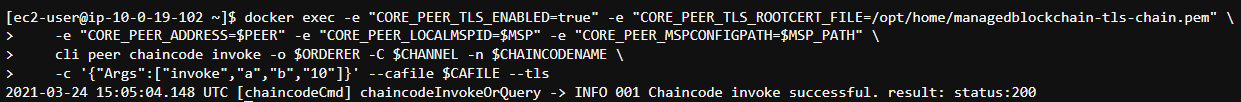
1. Invoke a transaction to transfer 10 dollars from Account a to Account b:

docker exec -e "CORE\_PEER\_TLS\_ENABLED=true" -e "CORE\_PEER\_TLS\_ROOTCERT\_FILE=/opt/home/managedblockchain-tls-chain.pem" **\**

-e "CORE\_PEER\_ADDRESS=$PEER" -e "CORE\_PEER\_LOCALMSPID=$MSP" -e "CORE\_PEER\_MSPCONFIGPATH=$MSP\_PATH" **\**

cli peer chaincode invoke -o $ORDERER -C $CHANNEL -n $CHAINCODENAME **\**

-c '{"Args":["invoke","a","b","10"]}' --cafile $CAFILE --tls



1. Query the balance of Account named ‘a’ again after transaction:

docker exec -e "CORE\_PEER\_TLS\_ENABLED=true" -e "CORE\_PEER\_TLS\_ROOTCERT\_FILE=/opt/home/managedblockchain-tls-chain.pem" **\**

-e "CORE\_PEER\_ADDRESS=$PEER" -e "CORE\_PEER\_LOCALMSPID=$MSP" -e "CORE\_PEER\_MSPCONFIGPATH=$MSP\_PATH" **\**

cli peer chaincode query -C $CHANNEL -n $CHAINCODENAME -c '{"Args":["query","a"]}'

