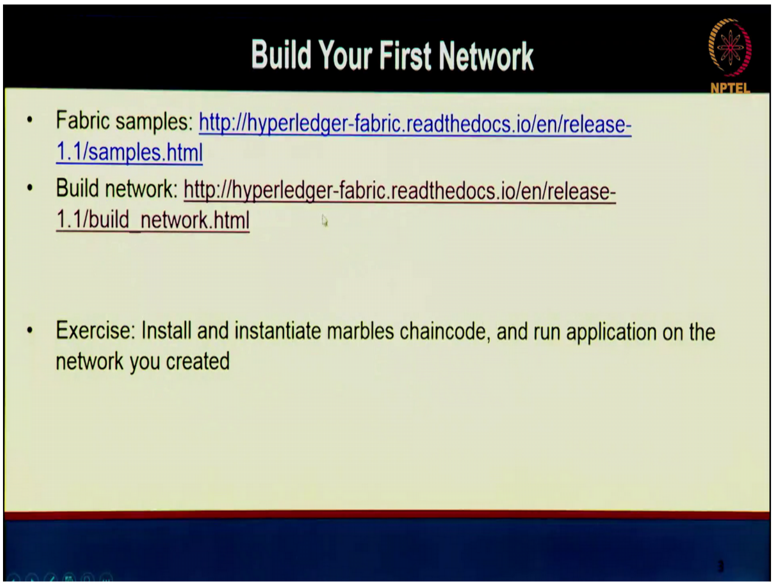
Blockchains Architecture, Design and Use Cases
Prof. Sandip Chakraborty
Prof. Praveen Jayachandran
Department of Computer Science Engineering
Indian Institute of Technology, Kharagpur

Lecture – 27
Fabric Demo, deploy from scratch – III

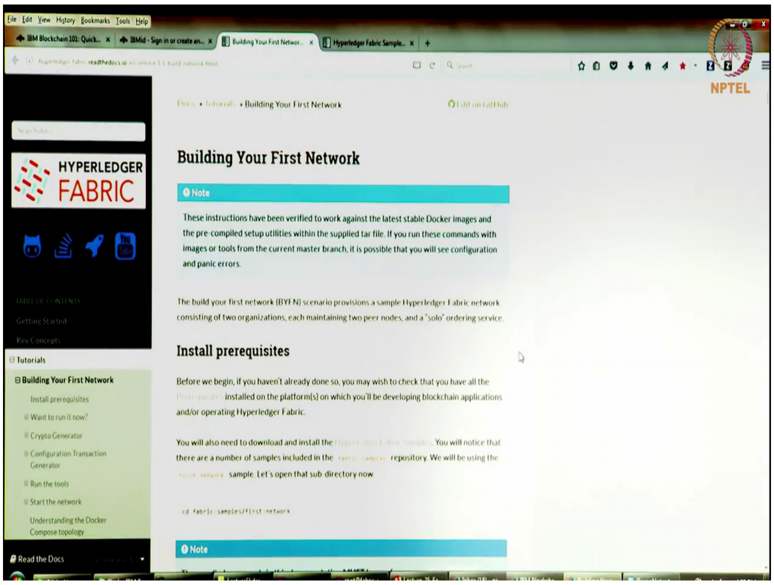
Hello everyone, good to have you back we are going to be we did a couple of demos on hyperledger fabric using IBM blockchain cloud, but now we are going to look at how you could do it yourself in your own virtual machine for instance or your own laptop, for instance you can you do it wherever you feel like. But I am going be using Ubuntu vm and ask for certain sort of prerequisites to be installed in the previous lecture. So, let us get started so, a couple of links for how you can build your own network.

(Refer Slide Time: 00:43)



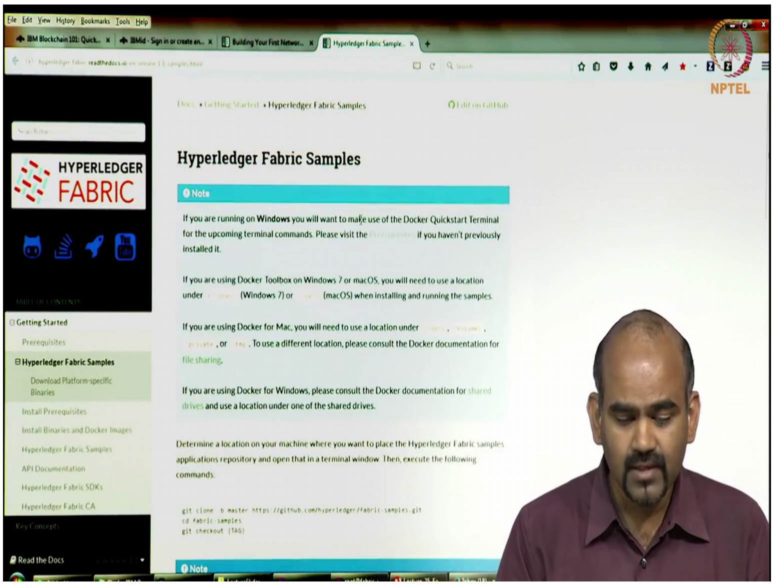
So, we are going to start from that I am going to start from this link on building a network. It once we are done here I would encourage you to try and install an application on top of the network that we are going to build. So, the instance I will show you where the marbles chain code is it and you can go deploy that same chain code on the custom network that you have developed on your or you have built on your virtual machine.

(Refer Slide Time: 01:10)



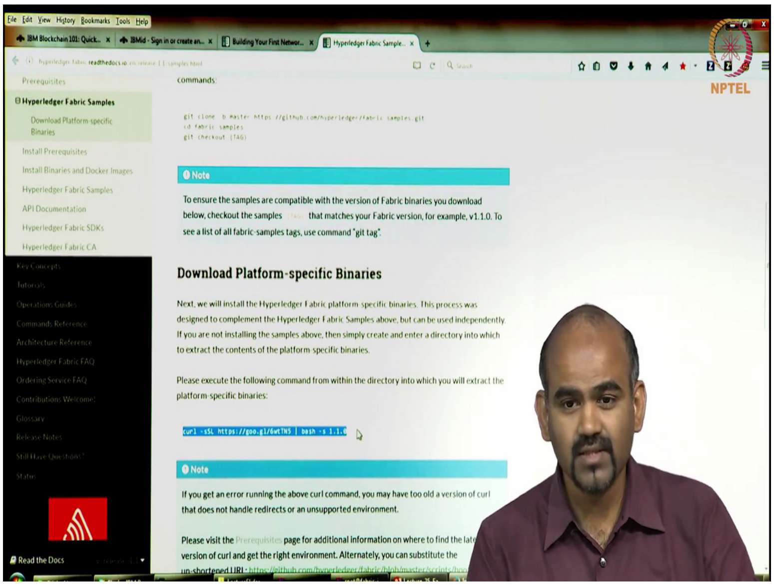
So, let us get there so, let us start with the instructions on how to build your network. So, this is the page that the pointer to. So, first you have to start with the installing some prerequisites, in the prerequisites as a link to the prerequisites you can go that install those that are needed for this for this exercise.

(Refer Slide Time: 01:29)



And there are what I have done as apart from setting of the prerequisites just to save time in the video.

(Refer Slide Time: 01:38)

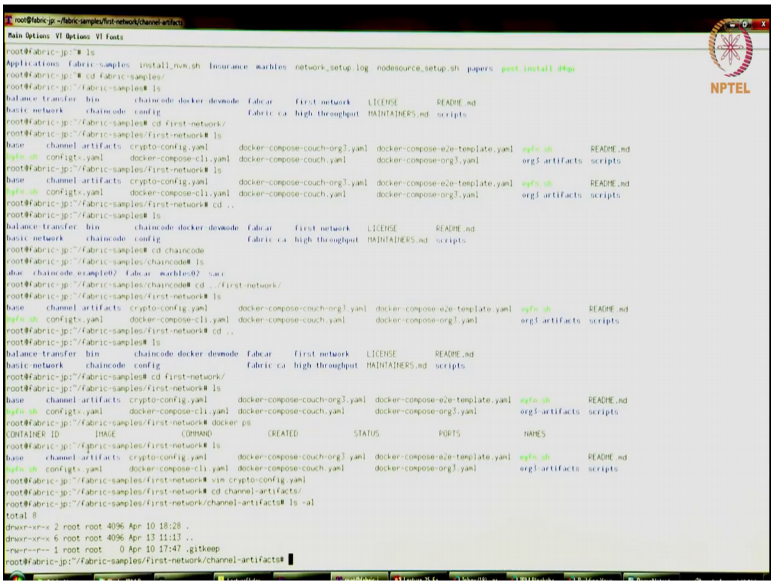


What have done is, I have also gone ahead and cloned this part this step. So, first you need to go clone piece of code these are the fabric samples. So, we are going to use that as the starting point so, go please clone this GitHub repository and let us get to the fabric samples folder.

So, just follow these steps here, this will what we will do is we will go and pick up a lot of docker containers in the background. So, you will get the peer container, reorder container, the ca container. So, you will get all of those a docker images and it will have that setup in your vm. So, for me took about 10 minutes for this to run, but for you for you it depends on your network right. So, it might be plus or minus of few minutes.

So, apart from once you done those steps where those fabric samples get the binaries as well, actually the docker containers are going to be fetched in the in the binaries. So, this step here you have to go fetch the binaries so, this is the peer order binary. So, this is where actually I will take the 10 minutes sorry about that. So, this getting the clone github itself is just a code so, that will be fast. So, these 2 steps are something you have to run it probably will take you 10 - 12 minutes to do that, but I have already done that for my vm so, we are going to start with that ok.

(Refer Slide Time: 03:09)

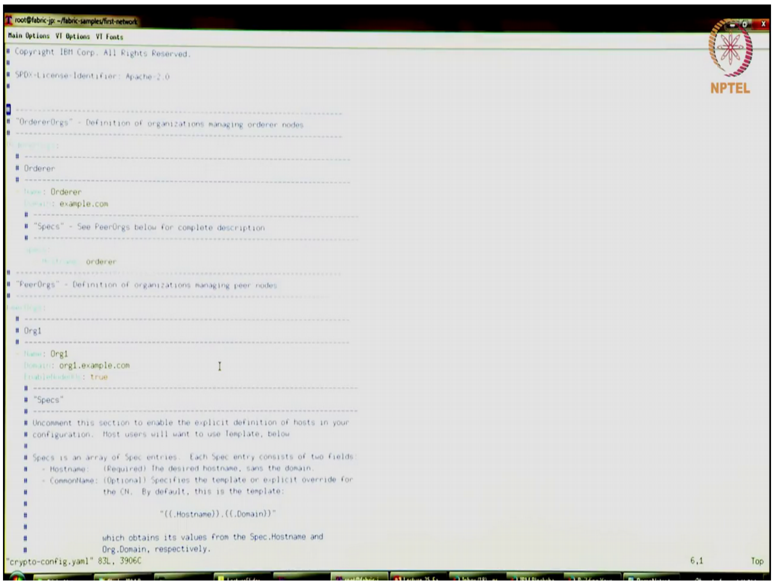


So, let us go back to the network now. So, what are we going to do, what are we let us get to the, this is the command line interface for my virtual machine. So, I am in this fabric samples slash first network. So, I have got let me get back to the fabric samples folder. So, in this there are many folders here, but I am going to use the first network, but there are other examples as well. So, there is a basic network I will give you a different topology will give you certain other aspects of setting up a network.

Let us start with this first network ok. So, first of all before I start let me show that I have a clean state now. So, I am looking at all the docker containers I have on this system there are no docker containers. So, this is the code that I am going to use to setup my network I have already cloned that code and I told you how to do that.

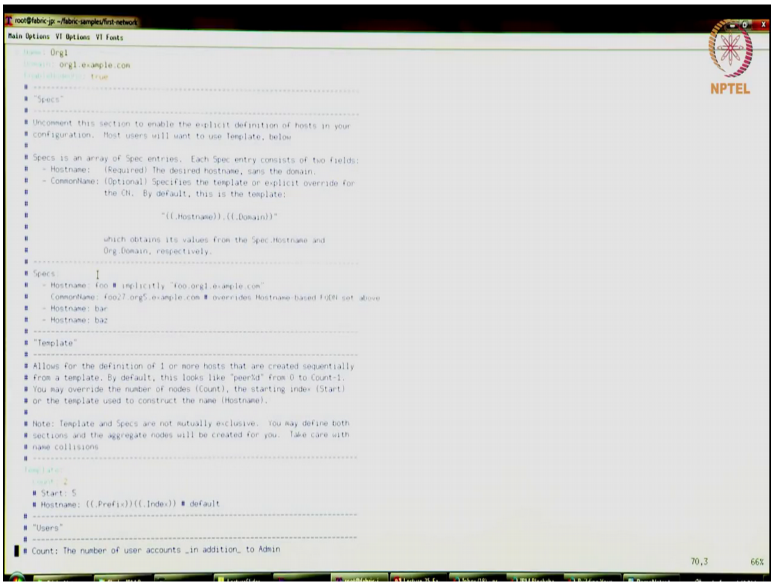
Now, is a next step I will first show you what is the network that we are going to build right. So, this network we are going to configure this in a file called crypto config dot yaml. So, this is a file in your folder I am going to open that file.

(Refer Slide Time: 04:15)



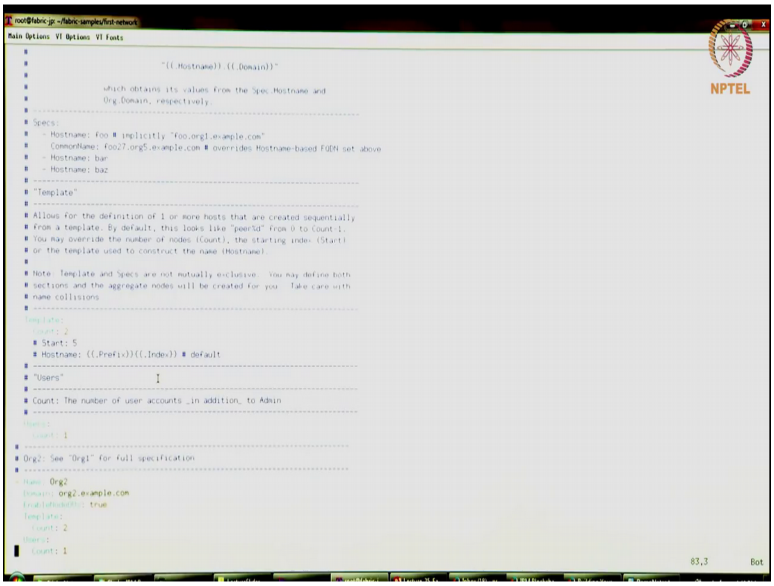
So, let us look at what this network this is basically a specification of the network itself as a YAML file and we are going be using this specification and some of the scripts that we have to setup a network. So, what is this have? So, this has who are orderer organizations. So, there is a name of the organization called orderer and it has the domain it is example dot com and it has some specs right and then there are peer organizations we are going to have one entry for each peer. So, there is a name called org one dot example dot com.

(Refer Slide Time: 04:52)



And then there it says template count colon 2, what this tells is there are going to be 2 peers belonging to organization. So, there are 2 peers so, count colon 2 belonging to organization 1.

(Refer Slide Time: 05:05)



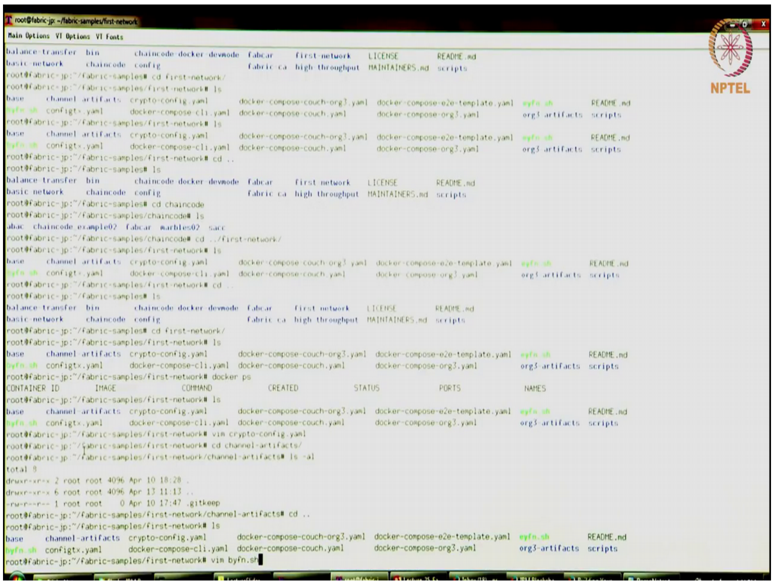
The second one this is going to be an organization 2 and here again we are going to have 2 peers connected to organization 2. So, we are going to have 2 organizations each running 2 peers so, that is going to be your network here.

Now, it also has a user count. So, how many users are you are going be created for each organizations so, that is also part of this specification right. So, we will run through this example I will encourage you to also follow along in your virtual machine or your development environment whatever you are using. For go along these step let us set this up, but separately please do feel free to play around with this configurations. So, for instance you can change the number of peers, the number of users you can go play around with some of these things then you will see how the networks get setup.

So, let us get back so, this is the configuration file for the network. So, this needs to be filled out there is a template here that you can use directly that is what we will use right now. So, that is the configuration file and before we start I will also show you what crypto configuration exists. So, right now there is nothing in the network right, no containers, no organizations created. So, there is this folder called channel artifacts right, if you look at that right now it is empty it just has a dot gitkeep file. So, there is nothing right now in the channel artifacts, there is no channel that is been created, nothing in the configuration for the channel.

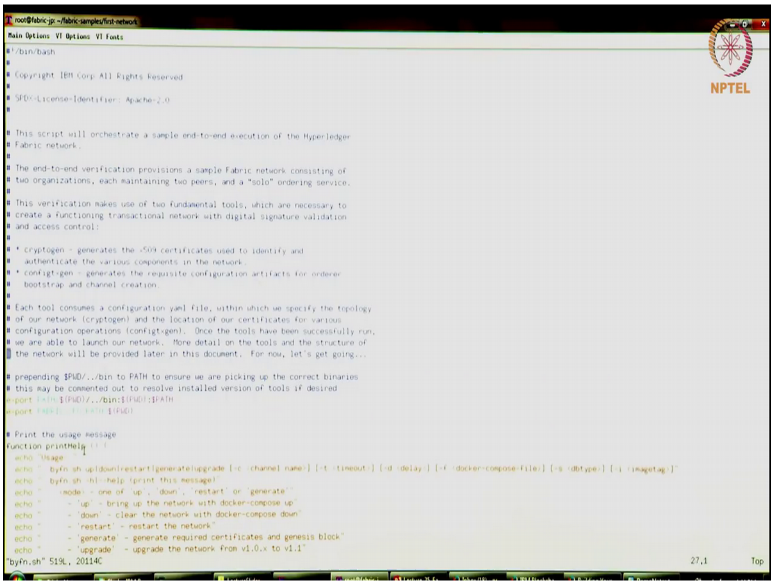
There is also another directory that will get created here called crypto config, which will contain all the crypto materials for your network. Basically the orderers certificates, the MSPs, the peers, the CAs, all of those the crypto material for all of those will be in that folder will get created as we proceed. So, I will show you how that gets created and when it gets created.

(Refer Slide Time: 07:11)



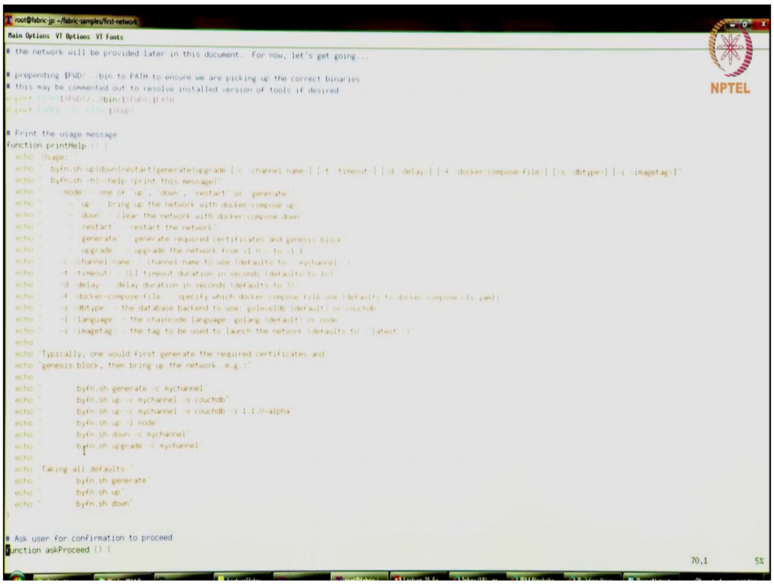
So that is just to show that clean slate from which we are starting. So, the there is a do it all script right. So, this is script called build your first network byfn dot sh. So, let us look at what that script really does right. So, this is the script that we are going to open.

(Refer Slide Time: 07:20)



So, let us look at some of what some of the functions that it does right.

(Refer Slide Time: 07:27)

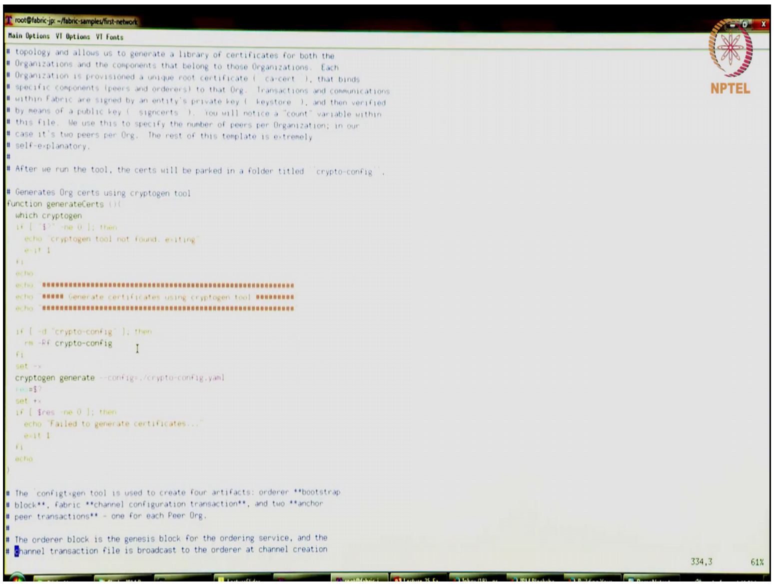


So, this first print helps this gives you how to use this, it has many functions this going to this generate function is going to generate some of the crypto material needed to setup the network. The byfn dot sh up command is going to bring up the network it is going to create all the docker containers, for the peers, reorderers, the CA is going to bring that up.

And is going to create a channel called mychannel right, we can change that to it is with the minus c parameter, you can bring down the network also byfn dot sh down and it is possible to upgrade the network from one fabric version to another fabric version. So, these are the commands that are that are available and certain set of optional parameters like the channel name, timeout delay, what is the docker compose file and so on right. So, if nothing a specified just default values will be used otherwise you can specify what parameters to use here.

So, let us look at a few of these interesting functions that show what happens right. So, let us go to one of them right. Let us first go to thing called generate let me just jump to that thing you can run through this file later, but I am going to use.

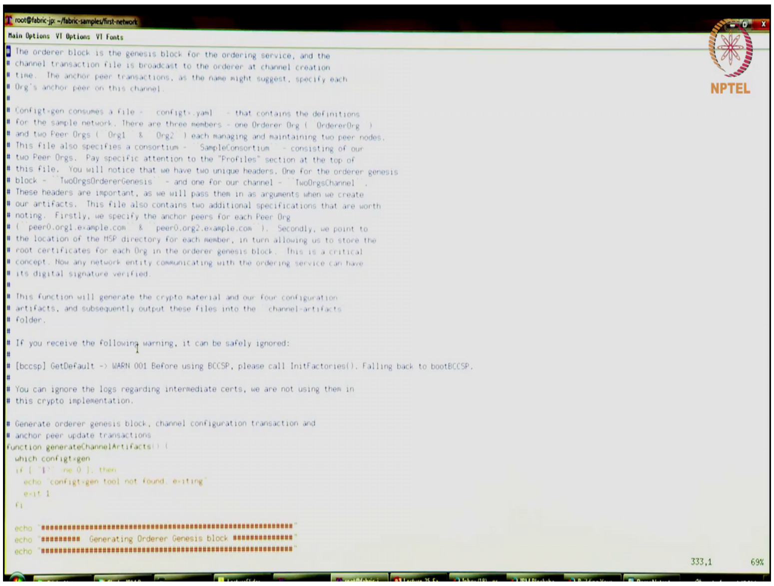
(Refer Slide Time: 08:45)



So, this is file called generatecerts. So, this sorry this function called generatecerts what it is going to do is, it is going to generate a certificate for a particular component or a user right. So, that is peer certificates, clone certificates so, user certificates or orderers certificates, generatecerts is going to do that.

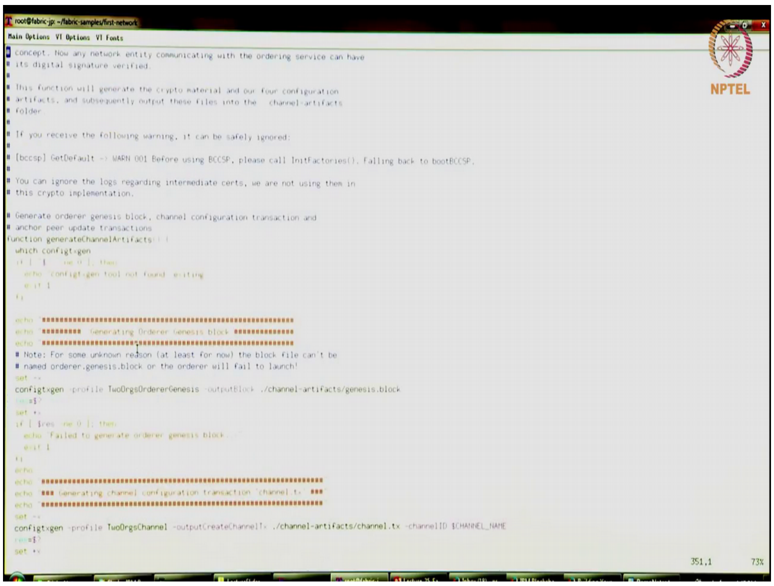
And there is a cryptogen tool that I have talked about in of the lectures. So, this is a tool that is provided with fabric it is a convenience tool for you to generate some of these certificates these x 5 o 9 certificates for you right. You can also do this without cryptogen using of fabric CA using a fabric CA to generate these certificates, but in this example we are going to use this cryptogen tool.

(Refer Slide Time: 09:32)



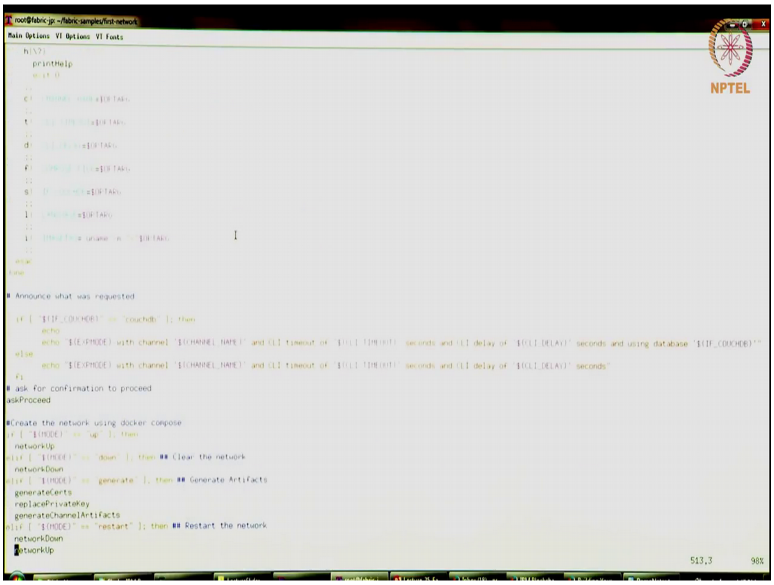
Generates channel artifacts by the way there are lot of extensive comments here that you see it all explains what happening in each function, how it happens step by step. The generate channel artifacts is going to generate basically the genesis block and the channel configuration right.

(Refer Slide Time: 09:50)



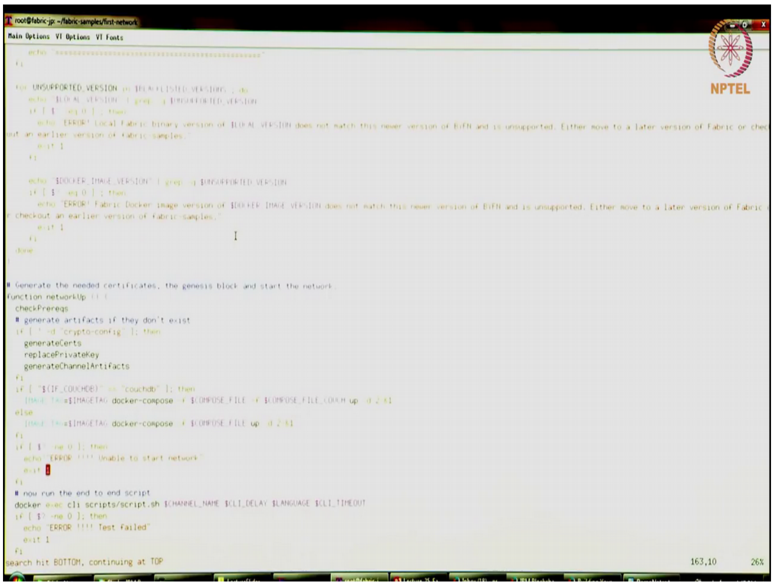
So, maybe we will come back to the explaining that step.

(Refer Slide Time: 10:00)



And then there is a function called networkup.

(Refer Slide Time: 10:09)

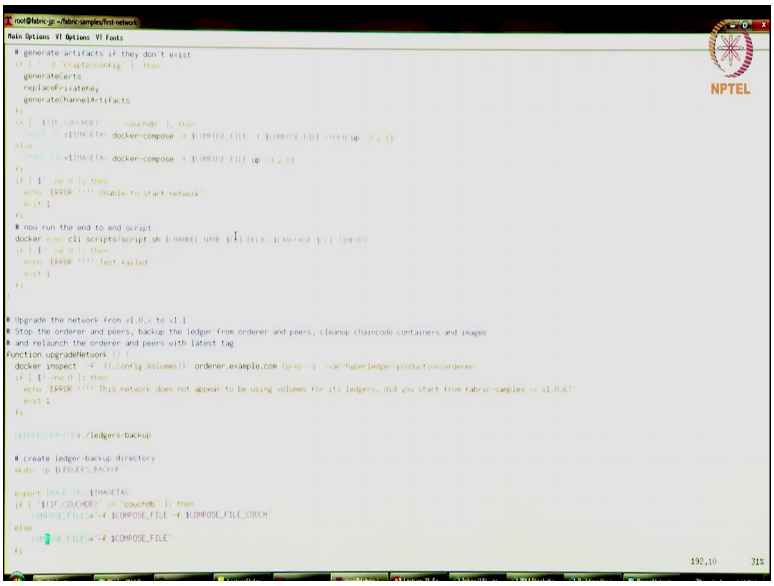


So, this is that function so, this is the function this main function this is going to bring up the network, once all the crypto material been generated all the identities been created for the peer orderer client and so on.

The network up function is going to bring up all the docker containers running all of these entities. So, by the way we are going to have just 1 vm right now running all of these components, but eventually think about the way blockchain would actually be set up is maybe each organization would probably run their own server running those components may even be a distributed systems like the peer may be on a different server, from the certificate authority and so on.

And so, eventually this will be a completely distributed system, but just for demonstration purposes making it easy we just going to do it as docker containers within a single VM. There are also instructions on how you can set up a multi VM environment for like think of it as multiple VMs for each organization that is also possible the instructions for that and I would encourage you to look at that later, there are links from some of the ones pointers that I have given myself.

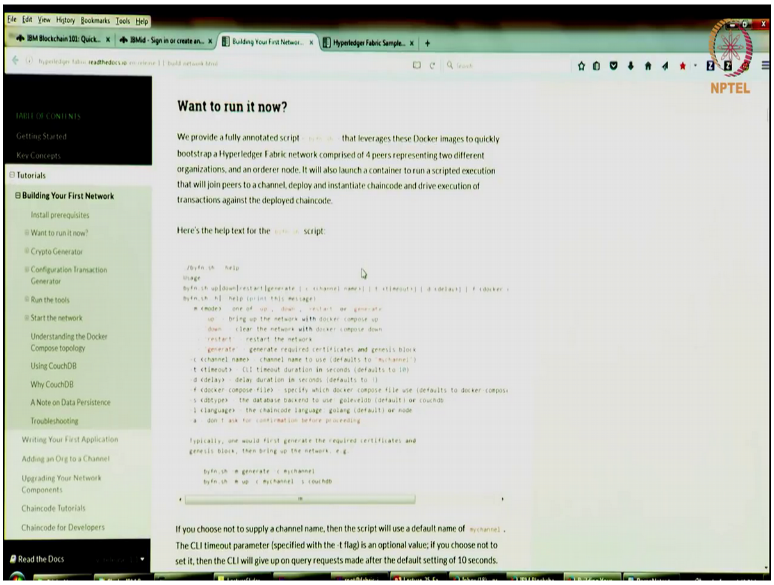
(Refer Slide Time: 11:19)



So, this network up function is the one that is going to bring up the, it is going to call docker compose. So, there is docker compose file has all the components and the it has pointers to the docker images to be brought up. So, based on what you have configured you will be able to you will be bringing up that many peers, orderers and so on so, that is what is happening in the networkup function.

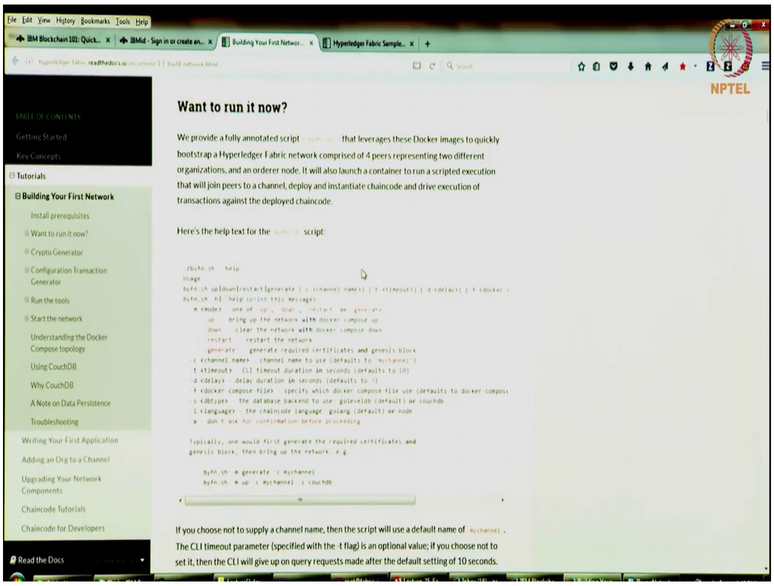
So, let us head back out of this thing, there are many other functions that you can look at, but let us head back, I will go back to the to the html file gives you the overview of how to build your network.

(Refer Slide Time: 12:01)



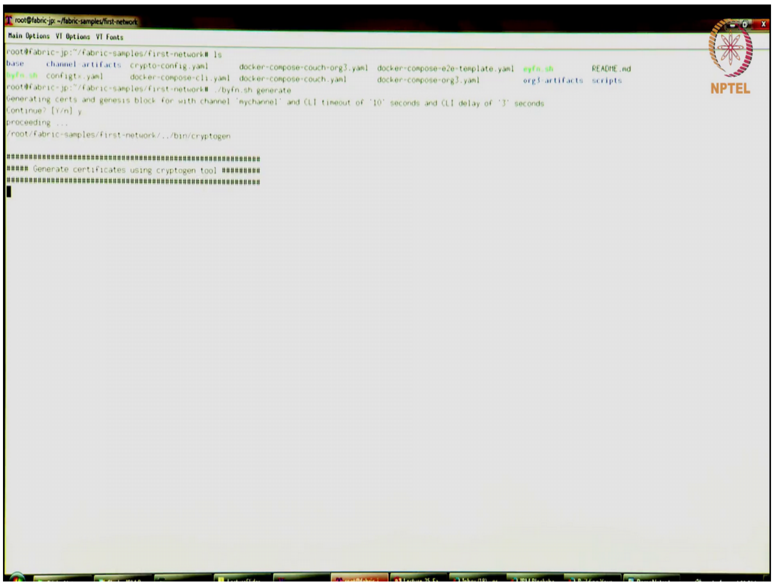
So, going back to some of these things right so, what are some of the things we are doing we are going to be generating certificates first.

(Refer Slide Time: 12:06)



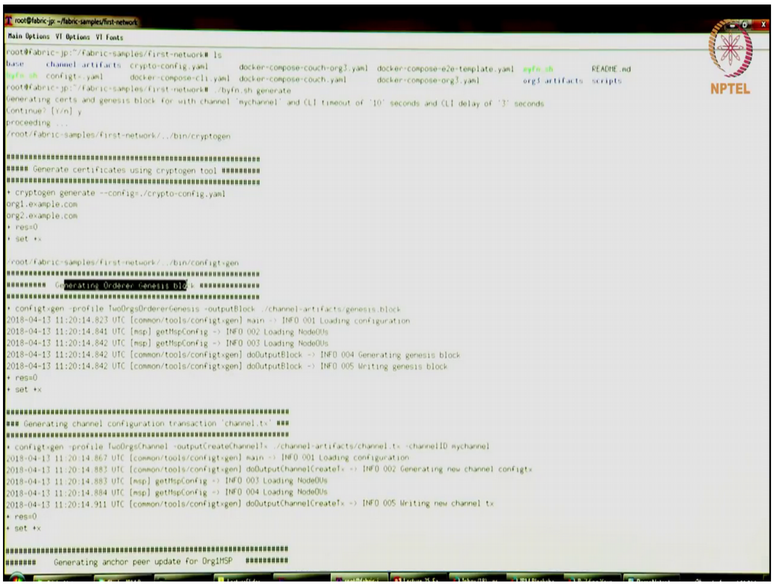
Then we are going to generate the network artifacts the channel aspects where we are going to create the orderer genesis block and then the channel configuration and then we are going to add peers to that channel. So, this is how this is what we are going to be doing. So, let us do that.

(Refer Slide Time: 12:23)



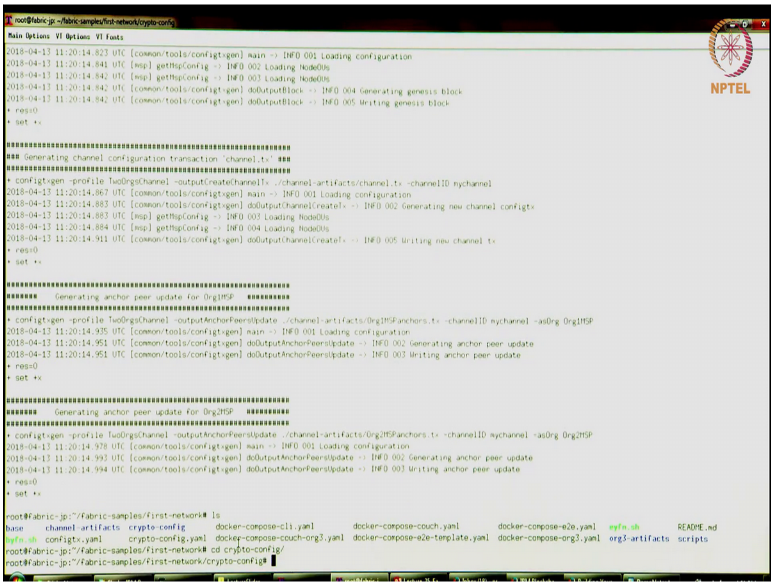
So, let me clear, now the first thing I am going to do is generate the identities for the different components. So, this is using the generate function generate command. So, I am going to run that it will ask you a prompt yes or no, you can hit yes.

(Refer Slide Time: 12:41)



And it has now generated all the identities. So, let us go over what it is done. So, we have used the cryptogen tool right, it is created an orderer genesis block. It is created a channel configuration transaction called channel dot tx and this is now initialize the channel is now created.

(Refer Slide Time: 13:05)



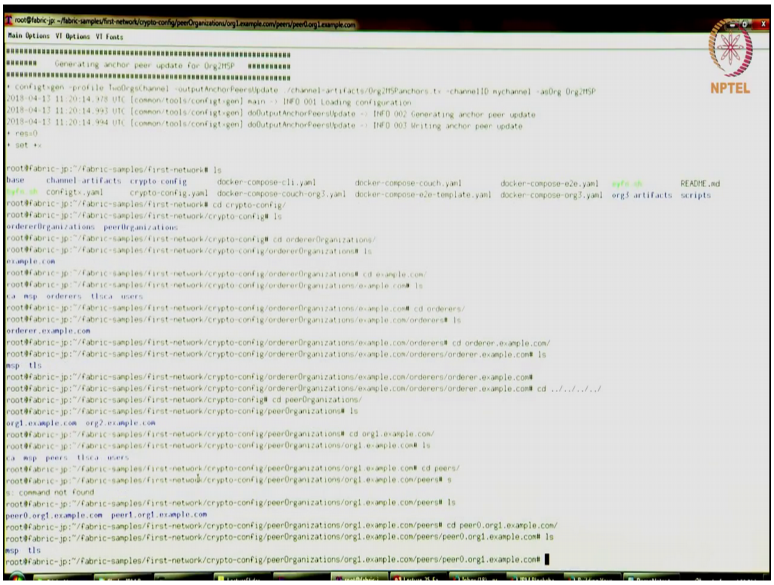
And then there is there are 2 anchor peers one for each organizations Org1MSP and Org2MSP all both have one anchor peer which has been added to the channel and the way it is structured is only the anchor peer is going to be talking to the ordering service. The second peer in each organization will just talk to the first peer and obtain that information and that is because it is both of those peers are within the same (Refer Time:13:32) boundary of the organization.

So, the second peer can trust the first peer, but the each peer in the 2 organizations will not trust each other they will both independently get the blocks from the orderer and will verify the orderer signature right. So, all the identities are set up the channel is set up the peers have been set up. So, that is what is happen so far.

So, while we added so, the other thing to notice here is now the crypto config folder has been created. So, this is where all the identities are currently stored right now and again a word of caution right now you stored all these identities on file system. It may not be the best idea to do this it is probably a good idea to put this inside let us say hardware security module. If you are drawing this in production should really be storing these identities inside a crypto module in hardware so, where it gets does not get stolen right.

You could also (Refer Time: 14:32) software for encrypted in a database. So, all those are options, but for now for the simple example just to make things easier of you to get started as a developer we just put it on the file system.

(Refer Slide Time: 14:52)



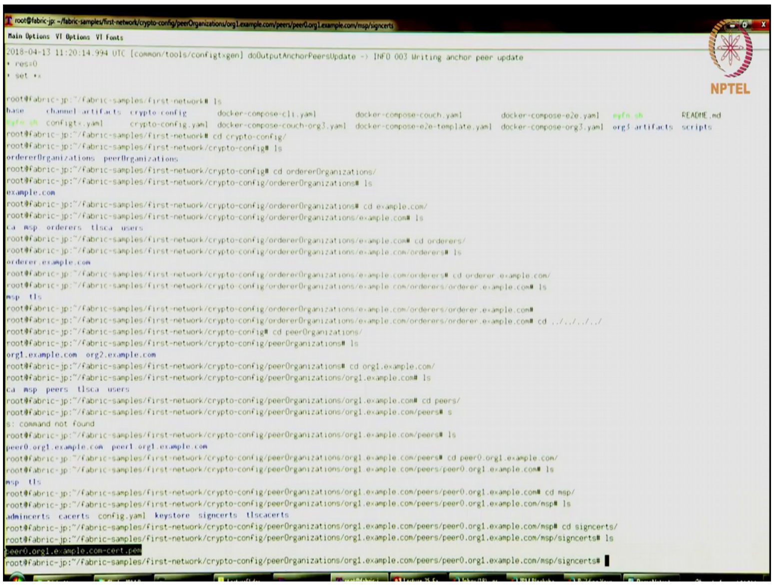
Let us look at the crypto config folder. So, it has orderer organizations and it has peerorganization. So, just for you to just to give you feel for what this looks like you just go look at the folder structure, within the orderer organization example dot com is the orderer that you created and within that it has the ca msp orderers, the tls certificates and users.

So, it has folders for each of these we could go inside let us say let us go into orderers, it has orderer dor example dot com and it will it will show you that msp and the tls certificates for the orderer right. So, likewise let us go back a few steps, you can go to peerorganizations and you will see there is org 1 dot example dot com and org 2 dot example dot com let us go into one of them.

So, org 1 dot example dot com again has the same folders right same folder structure this ca msp peers inside of the orderers you know have the peers folder and then again the tls certificates for network communication and the users for that organization. So, if you go into peers sorry into going to peers for this peer we are going to for this organization we are creating 2 peers, peer 0 dot org 1 dot example dot com and peer 1 dot org 1 dot example dot com.

So, again if you go into one of the peers, let us go into the msp of that peer this has for instance now the sign certificates. So, this will be where the private key for the peers. So, signcerts is there so, this if you see is the private key that the peer will use to sign this or sign whatever messages it is sending out in the network. So, maybe we can also open and view what this looks like.

(Refer Slide Time: 17:12)

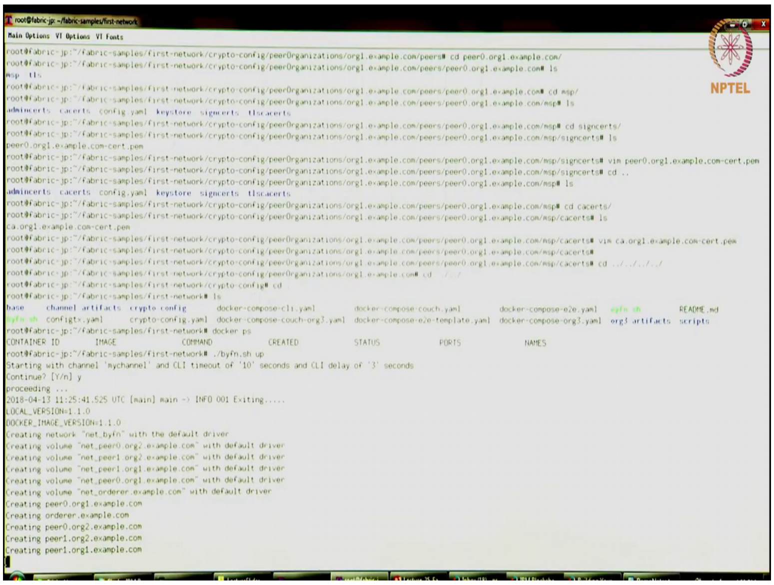


So, this is the certificate that it is using to sign right, you can also see some of the other things for instance you can go tls ca certs and cacerts.

So, if you go to cacerts this has the org one ca certificate right. So, all of these certificates have been generated and have been stored inside the crypto config folder. So, let us head back right. So, I have shown you where the certificates reside now I will do a docker ps again now there are no containers after this point, we only created the identities we have created the genesis block, but there is nothing else. So, genesis block is just the data structure right nothing else.

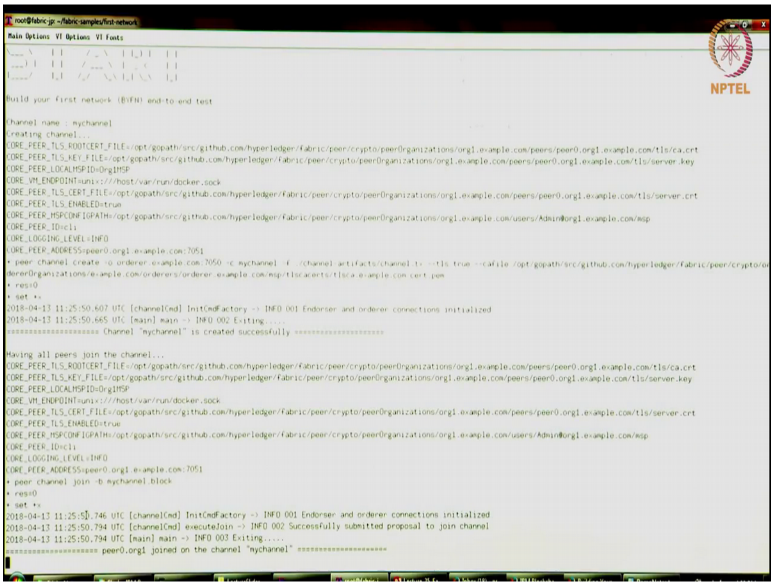
So, next up let us bring up the network. So, we have we are; what we are going to bring up is 2 peers, 1 orderer, 2 cas for each of the organizations and the that is what you are going to bring up now. So, let us do that so, we are going to do byfn dot sh up. So, this command is going to bring up the network is going to do a bunch of things again a prompt press yes.

(Refer Slide Time: 18:03)



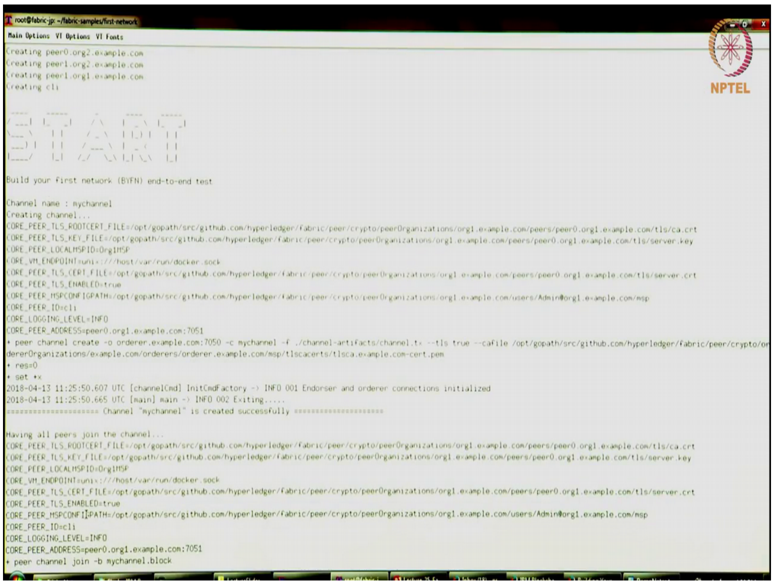
So, now, this is going to create all of these entities. So, the peer 0 dot org 1 peer 0 dot org 2.

(Refer Slide Time: 18:09)



Let us let it complete and then we will go over the output. So, it is going to run through a bit few things and I will walk you through what is actually doing in the backend and actually you can go back to the code and also figure out each of these steps and what it is doing in the backend. So, while it is running may be it can just it is take a couple of minutes we will start going through what it does right.

(Refer Slide Time: 18:33)

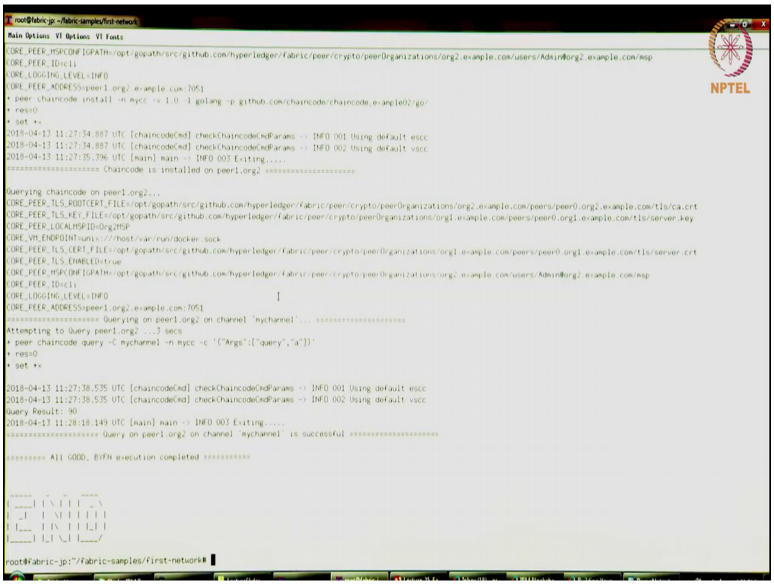


So, first it created the channel. So, using the channel configuration and the genesis block it went ahead and created this channel called "mychannel" it created that successfully. So, it is all clear then we are going to add the peers to the channel. So, peer 0 dot org 1 has joined the channel mychannel, peer 1 dot org 1 join the joined the channel, then peer 0 of org 2 I think it is skipped let us go back a bit peer 0 of org 2 and peer 1 of org 2 both joined the channel.

Then anchor peers for org 1 msp org 2 msp are updated so, we say that org 1. So, the peer 0 is going to be the anchor peer. So, we set the anchor peers for both the organizations org 2 and org 1. So, anchor peers are set these are the peers that are going to communicating with the orderer then we went ahead and installed a chaincode. So, this part I did not tell you about. So, I kept you (Refer Time: 19:43) about what it is going to do.

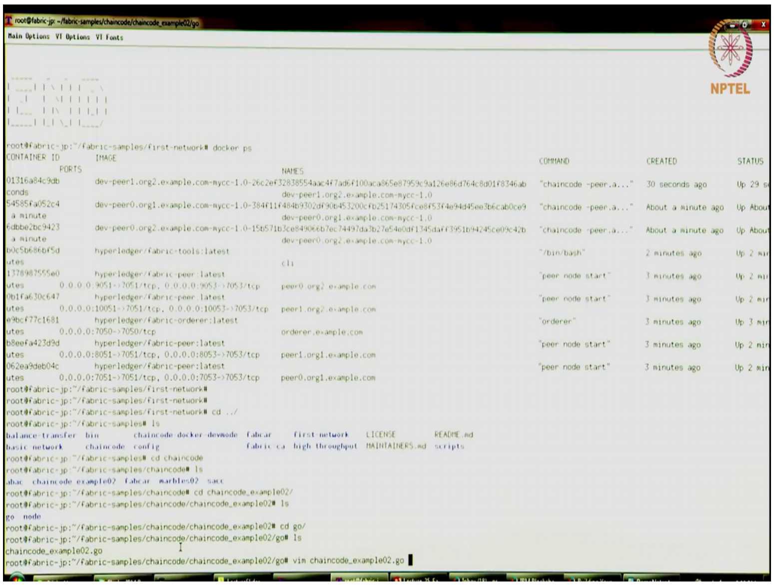
There is a chaincode that we have picked and installed. So, next chap we will go look at what chaincode we used right. So, there is a chaincode we setup on this channel, we installed that chaincode on peer 1 peer 1 dot org 2 and it is instantiated on the channel mychannel right. So, let us go back in the output. So, chaincode is installed on peer 0 org 1 chaincode is installed on peer 0 org 2. So, these are 2 anchor peers we installed that on them.

(Refer Slide Time: 20:27)



Then after that it is rolled over a bit script is finished now. So, that makes it easier for us to go over things so, the chaincode has been added. So, then the chaincode was instantiated on the channel “mychannel” right using peer 0, this is already joined the channel. So, once the chaincode is all set up then you can query and invoke the chaincode you can perform the different transactions. So, we will do that. But before that let us first look at what containers have started.

(Refer Slide Time: 20:58)



So, I am going to do docker ps. So, these are the containers that have started. So, we have peer 0 dot org 1 we have peer 0 dot org 2 peer 1 dot org 1 and peer 1 dot org 2. So, these are the 4 peers 2 peers for each organization we have one orderer that is come up and note that we have not brought up ca in this example we use the cryptogen tool to generate the certificates we did not use a fabric ca. So, that is also the power of an msp, we do not really the fabric itself does not care at where the certificates come from as long as the msp knows the certificates right, it knows the public keys for verification.

So, you can pre generate the certificates bring them on board into fabric and plug them and that is what we are doing now the other option of course, is to have a fabric ca generate the certificates and give it to the network. So, apart from that there is chaincodes that have been deployed. So, chaincodes have been deployed on org 1 org 2 and actually 2 peers on org 2 right. So, that is on both peer 0 and peer 1 chaincodes have been deployed and they have brought they those containers are up.

And another cool feature of fabric is that it automatically handles the default (Refer Time: 22:21) with chaincodes, let us say these are all docker containers right the chaincode might crash what if this docker container this process crashes. The next time you perform a query or a transaction on that node, the chaincode will automatically come up and then the transaction will run. So, fabric will takes care of bring up the net[work] the chaincode whenever necessary right.

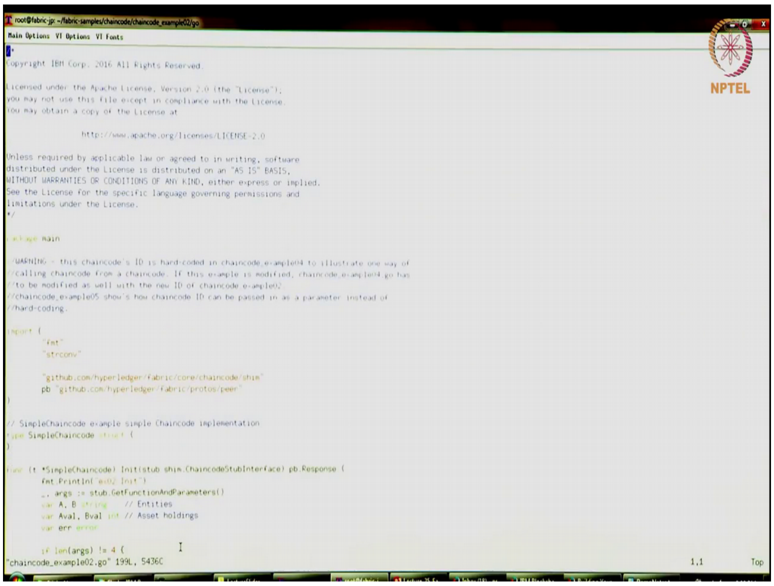
And all you have to do is when you instantiate you are bringing up the chaincode on that node only if you do a transaction or a query on another peer we will that think about, the reason we got these 3 up was because we performed transactions as part of this script you will see that there were queries and there were queries performed on those peers and when you perform those queries those chaincodes came up automatically.

So, when talking about chaincodes let us go see what that chaincode is we actually deploy a very simple chaincode. We will go up one directory we will go to the parent directory fabric samples there is a folder called chaincode right. So, this actually has multiple chaincode examples, but the one we have used now is chaincode example 02 right. I will walk through the chaincode example 02 for you and note that there is also marbles chaincode. So, this is precisely the chaincode that we used then and deployed in the IBM cloud right in the marbles example.

So, this chaincode is also available here, you can use this and deploy the marbles chaincode also on top of this network. So, I would not be showing that to you, but I encourage you to do that as an exercise take this marbles chaincode install it on multiple peers instantiate it on the mychannel channel that you have created right. So, try it as tried that try that as an exercise.

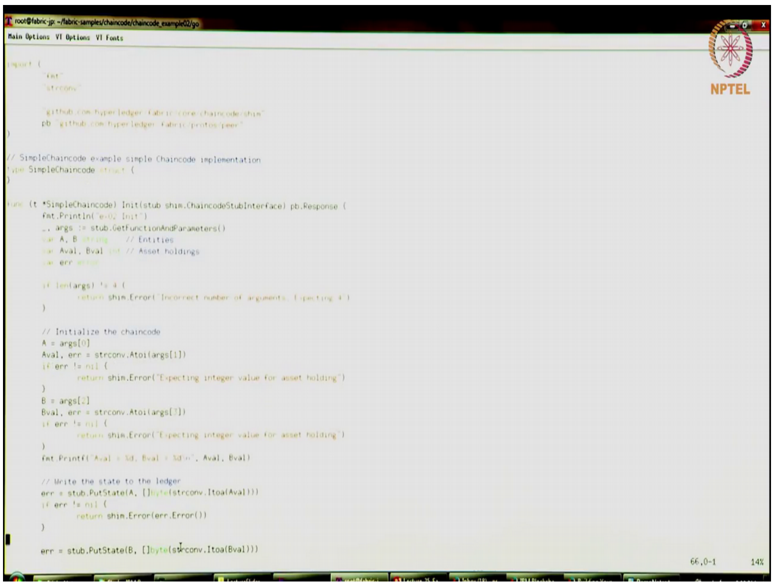
So, now, let us look at chaincode example 02 if you go in there is a go code as well as the node code node js code which is the application code, but let us now only focus on the smart contract or the golang code. So, there is a chaincode example 0 2 dot go.

(Refer Slide Time: 24:30)



So, let us look at this function, as would any smart contract any chaincode look there are 2 main functions, one is the init function and the other is the invoke function right.

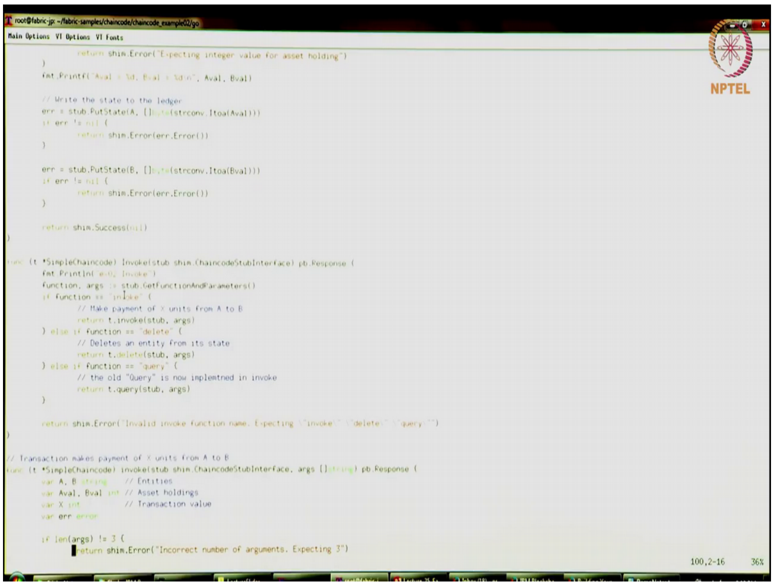
(Refer Slide Time: 24:43)



The init is going to be called as I mentioned when the chaincode is instantiated. So, this is the first time something is going to some transaction is happening on this chaincode, it is been instantiated we will call the init function to initialize any state information. So, what this chaincode does, it is very simple chaincode it has 2 state information it has 2 accounts account A and account B. They both initialized with a certain balance we can think of them as bank accounts and they have a certain balance.

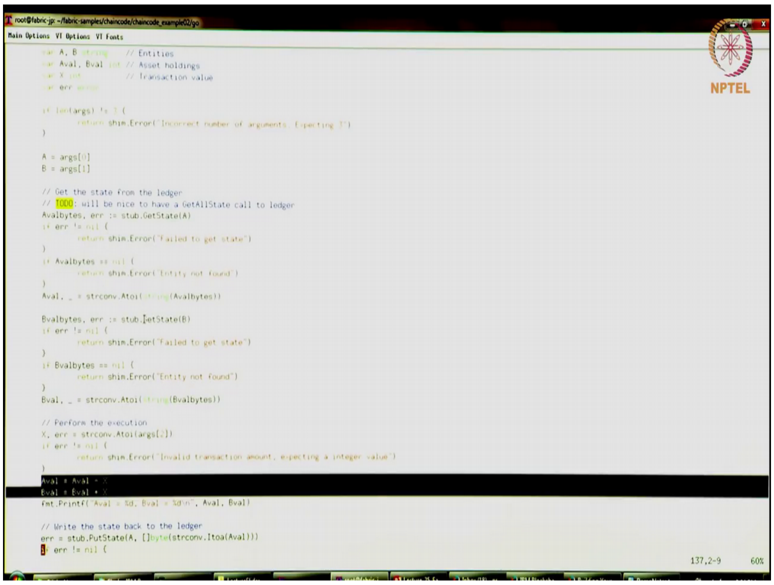
So, in this case there is some error checking that is first done, we are first going to initialize A with a certain balance and B with a certain balance and the balance is really a argument provided to the initialization function to the init function right. So, in this example we are actually going to set A's balance to be 100 and B's balance to be 200. So, this is what is going to get means initialized as this state information for this chaincode.

(Refer Slide Time: 25:44)



And what is the function it supports, it supports invoke delete and query. So, this the invoke function it supports arguments for invoke, delete and query. So, what I can do is that, I can invoke what does invoke do it actually makes a payment of X units from account A to account B. So, it is going to decrement A's balance by X and it is going to increment B's balance by X. So, that invoke function is out here right.

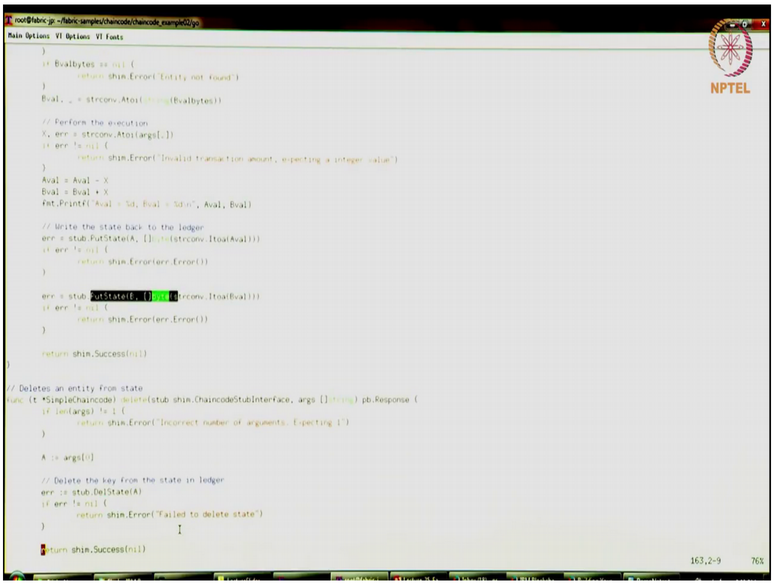
(Refer Slide Time: 26:16)



Here what so, here we are going to take those parameters convert from string to integer, fabric always takes all arguments as strings and you will have to then morph them into any data type of your choice. So, here is the logic where A's balance is decremented by X, B's balance is incremented by X, but first for before doing that you will have to read the current state of the balance of these accounts. So, that is done in this GetState function. So, the getstate will read as balance getstate of A reads A's balance from the ledger, getstate of B reads B's balance from the ledger, they are both updated.

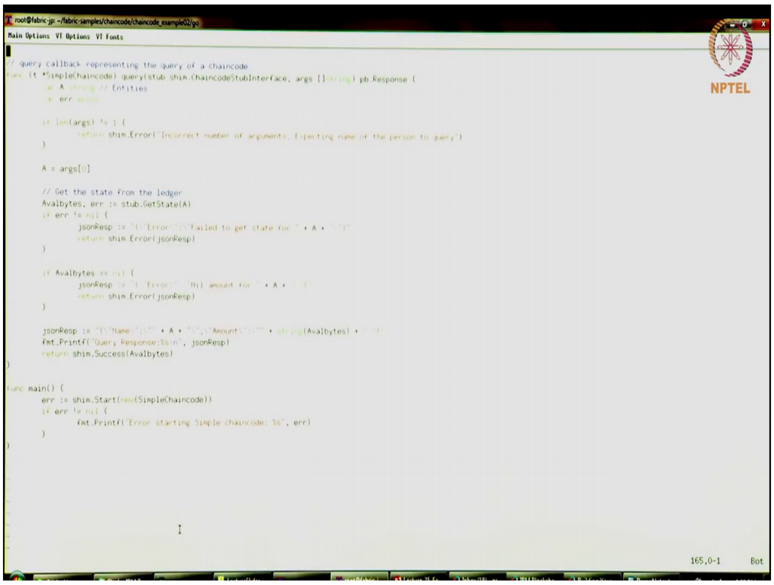
And then we have to write them back into the ledger. So, we are going to do a putstate of a saying this is the new balance for A and likewise for B, you are going to do A putstate for B right. So, that is really what this chaincode is about right.

(Refer Slide Time: 27:12)



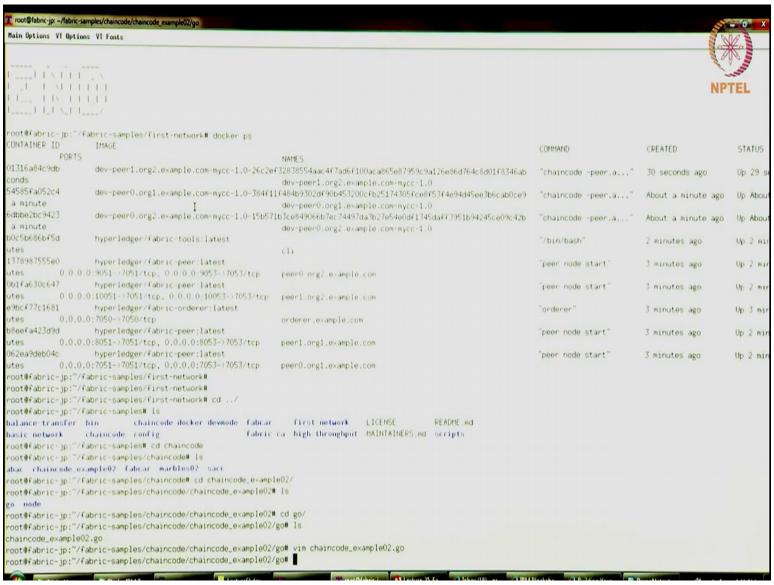
Separately there is also a query function where you can just ask for the account balance, what is the account balance for A or what is the Account balance for B.

(Refer Slide Time: 27:12)



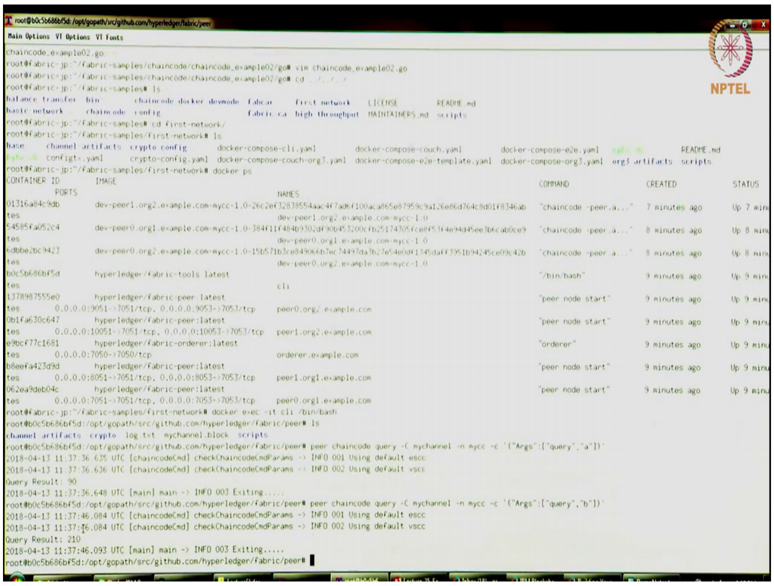
It will return the balance it will query the ledger this is the get state function and it will return that back return that to the user.

(Refer Slide Time: 27:37)



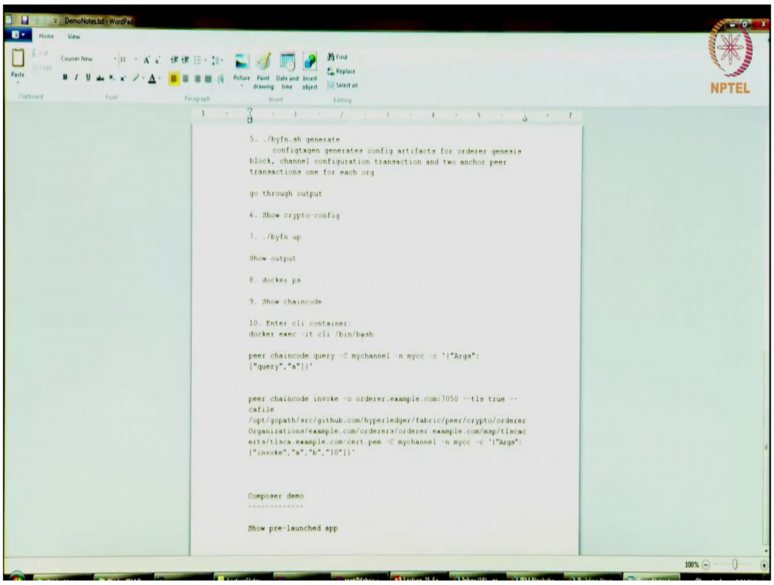
So, that is the, that is all this chaincode is about reads account balances and it allows you to transfer X units from one account to another. So, let us head back and let us go back to the first network that we had.

(Refer Slide Time: 27:50)



I will again show you the containers one other container that we skipped over is the cli container this is the command line interface that you can use to connect to the network right. So, what I am going to do is, I am going to now enter this container this container and run command line invocations and queries on the smart contract that we have deployed.

(Refer Slide Time: 28:14)



So, I am going to use cheat sheet here where I have these readily available. So, what I am going to do is, I have just entered I have used a docker exec command and I have now entered the container the cli container that I have here. So, this is really the cli container and I am in the peer folder inside the container and from here from this cli container this is really think of this as the client right; I am now going to query and invoke the chaincode.

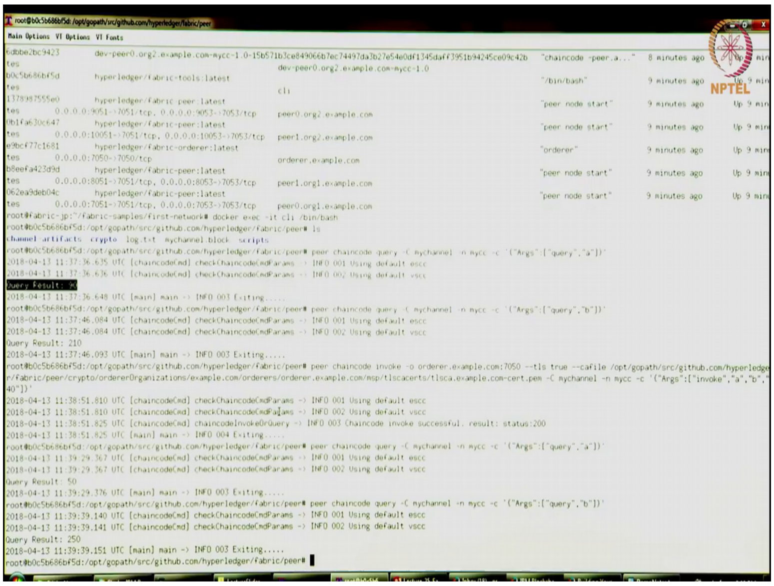
And note that as part of the initial script itself there were a few transactions that happened. So, there was actually one transaction that move 10 units from a to b. So, a balances was initially 100, b's balance was 200 we transferred 10 units from a to b as part of this thing. So, let us first start with querying the network, I have a cli command here. So, I am going to use that so, let us just look at the structure of this cli command.

So, it is just peer chaincode query it as a reference to the channel so, minus c mychannel. So, that is a channel we are going to operate on mycc is the chaincode I have deployed. So, I have installed on the channel. So, it is minus n minus mycc and minus c as the args provided. So, the args are I am going to call the query function and I am going to query for the account balance of a right. So, that is what this cli command is going to do.

So, if you run it you will see the query result is 90 as expected. So, this is the balance for A, likewise let us also query for the balance of B and it says the balance for the B is 210 ok. So, now, let us do our blockchain transaction and again I have a cheat sheet with full commands. So, that I do not have to type that all out right, now let us go over the structure of this invoke command.

So, it is a peer chaincode invoke minus orderer. So, we have we need to know who the orderer is minus minus tls saying I am going to use tls certificates to all my network communications is going to be encrypted, tls is true. What is the ca file? The ca is here is as the then full path to this tlscert, the tlscert for this here, minus c for mychannel minus n mycc this is the chaincode reference and minus c arguments. So, in the arguments we are calling the invoke function we are going to say transfer from a to b so many units.

(Refer Slide Time: 30:53)



So, we going to say in this orderer says 10 units, but let us say we transfer 40 units from a to b ok.

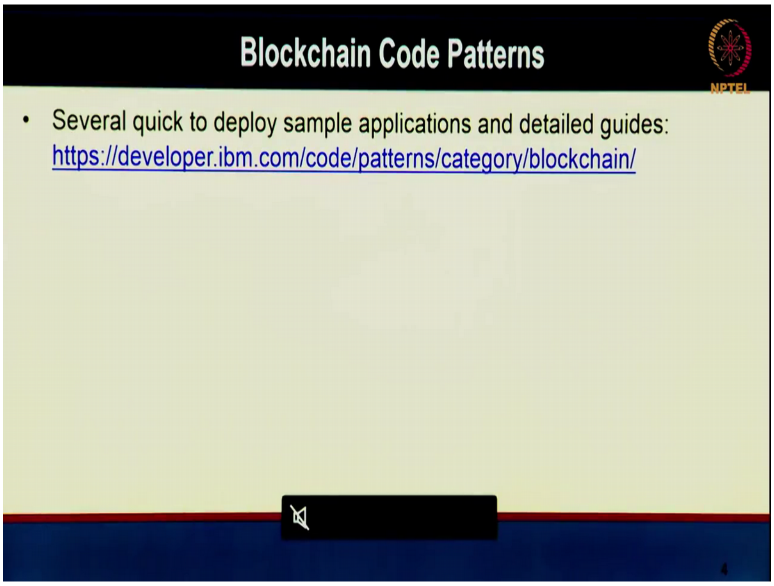
So, what we do is, this is going to be a asynchronous transaction. So, we have submitted this transaction on to the blockchain network and blockchain has told us that yes this is been accepted we get a 200 back, but this is not yet committed on to the ledger right. It might fail for some reason because may be the account balances are not there may be somewhere else tried to also spend at the same time. So, there is a conflict and the state is not state validation fails. So, many reasons why it could fail, but in this case I hope it is it is going to succeed.

So, let us query now for the account balance for a you see that it is now read (Refer Time: 31:38) 40 we transferred forty from a to b. So, from 90 it came down to 50 and we can see that B has gone from 210 to 250. So, that is the atomicity so, either both accounts will be updated or neither will be updated. So, if it fails, then the account balances will remain what was it before it could have remained 90 to 10 if this transaction is failed right.

So, we have gone through how you setup a network by yourself from scratch, you brought up the, you created the identities. You created the certificates you brought up the network as docker containers. So, in this network we have 2 orderers sorry 1 orderer, 2 peers for 2 organizations. So, 4 peers total and we installed instantiated a chaincode we performed certain transactions on that chaincode, we also perform some queries to find out account balances and so, that is that is one simple application deployed on your own blockchain network.

So, as I as an exercise as I mentioned please do try and install the marbles application and try and try and execute some transactions on that marbles application. So, with that let us get back here.

(Refer Slide Time: 33:01)



Apart from this IBM is also put out many code patterns example applications, again the code is there the examples are there. So, you can go quickly try those out as well. So, there are many different code patterns provided they are there in this link here.

You can go look that up in their code patterns. So, with that I hope you have a good feel for how to get started using hyperledger fabric develop your own applications set up your network deploy the applications layer and hopefully you can get started in building your own blockchain applications and do some projects.

Thanks a lot with that we end this demo session we will see you at the next lecture, bye.