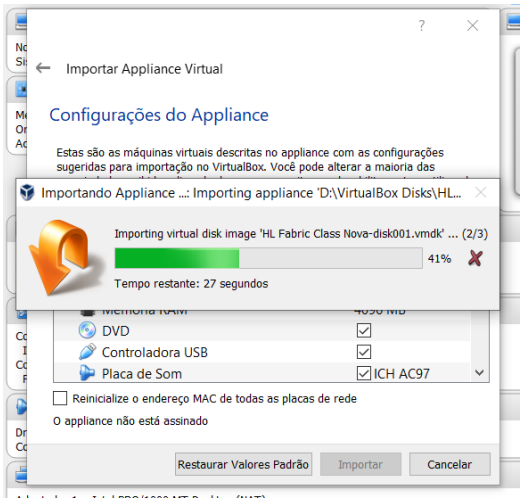
Aula – Criando a primeira rede Hyperledger Fabric

Configurando o Virtual Box

Baixe o arquivo “HL Fabric Class Nova.ova”

Abra o Virtual Box.

Selecione “Arquivo/Importar Appliance” para Instalar nova Máquina.



Selecione a nova máquina e clique em “Iniciar”.

Credenciais de acesso

As credenciais de acesso da VM são:

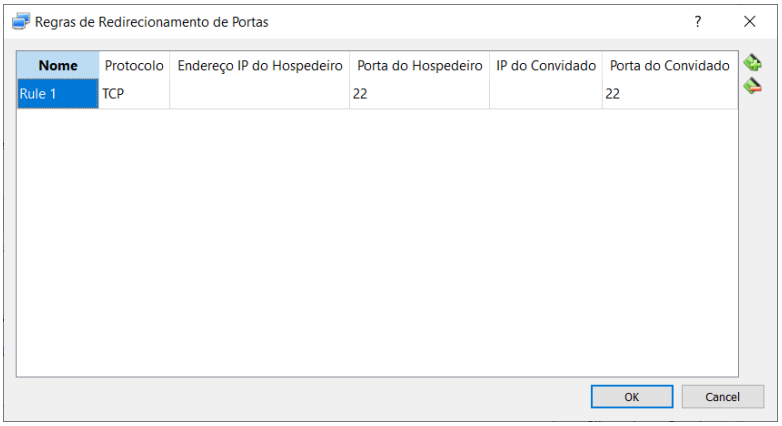
- **Username**: student
- **Password**: 123456

Utilizando o Putty

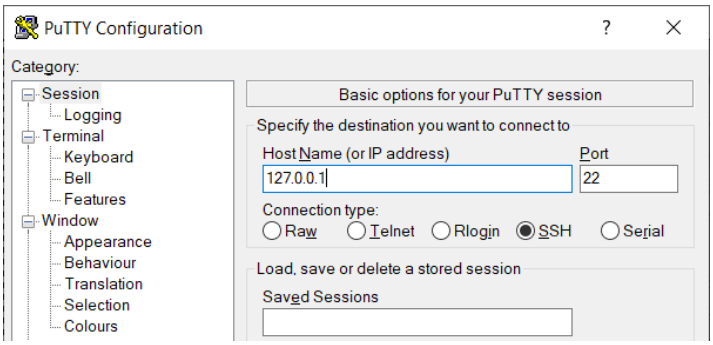
É sugerido utilizar o cliente “Putty” para acessar a VM ao invés de realizar o acesso pelo VBox.

Para tal, deve-se configurar o redirecionamento de portas no VBox: **Configurações -> Rede (Deve estar em NAT) -> Avançado -> Redirecionamento de Portas**

Acrescentar uma regra de redirecionamento para a porta 22 (SSH).



Depois, fazer o acesso via **Putty.**



Configuração da VM

A VM já está com as seguintes ferramentas configuradas (necessárias para desenvolvimento Hyperledger Fabric):

- Curl
- Git
- Docker
- Docker-compose
- GoLang

Tarefa 1

Ir para o diretório */home/student*

Baixar as ferramentas Hyperledger Fabric versão 1.4

```
curl -sSL http://bit.ly/2ysbOFE | bash -s -- 1.4.4 1.4.4 0.4.18
```

```
export PATH=$PATH:$HOME/fabric-samples/bin
```

Clonar o repositório:

```
git clone https://github.com/goledgerdev/goledger-challenge
```

Dentro do diretório *goledger-challenge/goledger-network* gerar os certificados de uma rede com uma organização através do comando:

```
./bgldgr.sh generate
```

Utilizar o *openssl* para mostrar os detalhes de um certificado de *peer*, *orderer*, admin e um certificado *tls* qualquer.

openssl x509 -text -noout -in [arquivo certificado]

Gravar os resultados do comando openssl.

Tarefa 2

Subir os containers.

Dentro da pasta *goledger-network* executar o seguinte comando.

```
docker-compose -f docker-compose-cli.yaml up -d
```

Mostrar as primeiras linhas do log do *peer* e do *orderer* depois os containers subirem.

Tarefa 3

Criar um channel *mychannel* e adcionar a *org1* no channel

Acessar o container *cli*:

```
docker exec -it cli bash
```

Dentro do container *cli* criar o channel *mychannel*:

```
peer channel create \
        -o orderer.example.com:7050 \
        -c mychannel \
        -f ./channel-artifacts/channel.tx \
        --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/ordere
rs/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```

Incluir a *org1* no channel *mychannel*:

```
peer channel join -b mychannel.block
```

Mostrar o resultado das operações de criar o *channel* e *join* org.

Tarefa 4

Instalar e instanciar um chaincode *fabcar* no channel *mychannel*.

Dentro do container cli, para instalar o chaincode no peer0 da org deve chamar o seguinte comando.

```
peer chaincode install -n fabcar -v 1.0 -p github.com/chaincode/fabcar/go/
```

Para instanciar o chaincode *fabcar*, versão 1.0 no channel *mychannel*:

```
peer chaincode instantiate \
        -o orderer.example.com:7050 \
        --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/ordere
rs/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem \
        -C mychannel \
        -n fabcar \
        -v 1.0 \
        -c '{"Args":["init"]}'
```

Mostrar o resultado das operações de instalar e instanciar o chaincode no channel, assim como listar o chaincode instalado e instanciado através dos seguintes comandos.

```
peer chaincode list --installed
```

```
peer chaincode list --instantiated -C mychannel
```

Tarefa 5

Chamar a função *initLedger* do chaincode *fabcar*

```
peer chaincode invoke \
        -o orderer.example.com:7050 \
        -C mychannel \
        -n fabcar \
        -c '{"function":"initLedger","Args":[]}' \
        --waitForEvent \
        --tls \
        --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/ordere
rs/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem \
        --peerAddresses peer0.org1.example.com:7051 \
        --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peer
s/peer0.org1.example.com/tls/ca.crt
```

Mostrar o resultado da operação e o log do container de chaincode (*dev-peer0*...)

Tarefa 6

Chamar a função *queryAllCars* chaincode *fabcar*

```
peer chaincode query \
        -C mychannel \
        -n fabcar \
        -c '{"Args":["queryAllCars"]}'
```

Mostrar o resultado da operação e o log do container de chaincode (*dev-peer0*...)