



IBM Software Services



Workshop Blockchain: Hyperledger Fabric 1.4

Laboratório 03

Versão: 1.0

Author: Richard S Marques – rseberin@br.ibm.com

Summary

Summary	2
1. Introdução	3
2. Proposta do laboratório	3
3. Execução.....	3
3.1. Criar os recursos	3

Data	Versão	Autor	Descrição
10/10/2019	1.0	Richard Marques	Criação do documento

1. Introdução

Este documento tem o objetivo de guiar o participante do workshop nas tarefas propostas no laboratório de número 03 do workshop de blockchain: Hyperledger Fabric 1.4.

Este e os demais laboratórios foram testados em um servidor Ubuntu 18.04 c/ 4 Cores de CPU e 8 GB de memória RAM.

Este laboratório tem como pré-requisito que o laboratório anterior (número 02) tenha sido concluído com sucesso.

2. Proposta do laboratório

A proposta deste laboratório será:

- Criar as configurações da Rede
- Gerar o genesis block
- Subir 3 instâncias de Zookeeper
- Subir 3 instâncias de Kafka
- Subir 3 instâncias de Orderer

3. Execução

Siga atentamente todos os passos para concluir o laboratório:

3.1. Criar os recursos

Para esta atividade navegue até a pasta “lab03” e execute o script “./criarRecursos.sh”

```
blockchain@ubuntu:~/workshop$ cd lab03
blockchain@ubuntu:~/workshop/lab03$ ./criarRecursos.sh
2019-10-15 07:38:27.969 PDT [common.tools.configtxgen] main -> INFO 001 Loading configuration
2019-10-15 07:38:27.995 PDT [common.tools.configtxgen.localconfig] completeInitialization -> INFO 002 orderer type: kafka
2019-10-15 07:38:27.996 PDT [common.tools.configtxgen.localconfig] Load -> INFO 003 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.006 PDT [common.tools.configtxgen.localconfig] completeInitialization -> INFO 004 orderer type: kafka
2019-10-15 07:38:28.006 PDT [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 005 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.013 PDT [common.tools.configtxgen] doOutputBlock -> INFO 006 Generating genesis block
2019-10-15 07:38:28.013 PDT [common.tools.configtxgen] doOutputBlock -> INFO 007 Writing genesis block
2019-10-15 07:38:28.039 PDT [common.tools.configtxgen] main -> INFO 001 Loading configuration
2019-10-15 07:38:28.048 PDT [common.tools.configtxgen.localconfig] Load -> INFO 002 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.058 PDT [common.tools.configtxgen.localconfig] completeInitialization -> INFO 003 orderer type: kafka
2019-10-15 07:38:28.059 PDT [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 004 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.059 PDT [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 005 Generating new channel configtx
2019-10-15 07:38:28.061 PDT [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 006 Writing new channel tx
2019-10-15 07:38:28.093 PDT [common.tools.configtxgen] main -> INFO 001 Loading configuration
2019-10-15 07:38:28.103 PDT [common.tools.configtxgen.localconfig] Load -> INFO 002 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.115 PDT [common.tools.configtxgen.localconfig] completeInitialization -> INFO 003 orderer type: kafka
2019-10-15 07:38:28.115 PDT [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 004 Loaded configuration: /tmp/crypto/crypto-config/configtx.yaml
2019-10-15 07:38:28.115 PDT [common.tools.configtxgen] doOutputAnchorPeersUpdate -> INFO 005 Generating anchor peer update
2019-10-15 07:38:28.115 PDT [common.tools.configtxgen] doOutputAnchorPeersUpdate -> INFO 006 Writing anchor peer update
[sudo] password for blockchain:
service/zookeeper1 created
service/zookeeper2 created
service/zookeeper3 created
statefulset.apps/zookeeper-pod1 created
statefulset.apps/zookeeper-pod2 created
statefulset.apps/zookeeper-pod3 created
service/kafka1 created
service/kafka2 created
service/kafka3 created
statefulset.apps/kafka-pod1 created
statefulset.apps/kafka-pod2 created
statefulset.apps/kafka-pod3 created
service/orderer-svc created
service/orderer1 created
service/orderer2 created
service/orderer3 created
statefulset.apps/orderer-pod1 created
statefulset.apps/orderer-pod2 created
statefulset.apps/orderer-pod3 created
```

Ao final do processo, execute o comando “kubectl get pods” e verifique se existem os 9 pods a mais rodando, 3 de Kafka, 3 de Zookeeper e 3 de Orderer, como na imagem a seguir:

```
[blockchain@ubuntu:~/workshop/lab03$ kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
ca-0                 1/1     Running   0           11h
kafka-pod1-0         1/1     Running   0           9m55s
kafka-pod2-0         1/1     Running   0           9m54s
kafka-pod3-0         1/1     Running   0           9m54s
orderer-pod1-0       1/1     Running   0           9m48s
orderer-pod2-0       1/1     Running   0           9m48s
orderer-pod3-0       1/1     Running   0           9m48s
postgres-0           1/1     Running   0           11h
zookeeper-pod1-0     1/1     Running   0           10m
zookeeper-pod2-0     1/1     Running   0           10m
zookeeper-pod3-0     1/1     Running   0           10m
```