J CERCHIARO
PRACTICA HF CERT

1. Despliega la red. Parece que existen algunos contenedores que no se levantan correctamente. Identifícalos y resuelve los problemas asociados.

Se corrigen las líneas 42-91 palabras server y hyperledge

```
42      - ORDERER_ADMIN_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
```

```
91      - ../organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com:/etc/hyperledger/fabric
```

2. Despliega el chaincode “merca-chaincode”. Aunque el proceso de ciclo de vida del chaincode es satisfactorio, parece que este no se despliega correctamente. Identifica el/los problemas y resuélvelos. Documenta este proceso de investigación y resolución.

Para despegar los chain codes

./network.sh up createChannel -ca ./network.sh deployCC -ccn merca-chaincode -ccl javascript -ccv 1.0.0 -ccp ../merca-chaincode

Se selecciona el archivo que solo está en RW y se cambia a XRW ejecutable xon el fin de desplegar el chaincode.

```
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-chain$ ./network.sh deployCC -ccn merca-chaincode -ccl javascript -ccv 1.0.0 -ccp ../merca-chaincode
Using docker and docker compose
deploying chaincode on channel 'mychannel'
executing with the following
- CHANNEL_NAME: mychannel
- CC_NAME: merca-chaincode
- CC_SRC_PATH: ../merca-chaincode
- CC_SRC_LANGUAGE: javascript
- CC_VERSION: 1.0.0
- CC_SEQUENCE: auto
- CC_END_POLICY: NA
- CC_COLL_CONFIG: NA
- CC_INIT_FCN: NA
- DELAY: 3
- MAX_RETRY: 5
- VERBOSE: false
executing with the following
- CC_NAME: merca-chaincode
- CC_SRC_PATH: ../merca-chaincode
- CC_SRC_LANGUAGE: javascript
- CC_VERSION: 1.0.0
+ '[' false = true ']'
+ peer lifecycle chaincode package merca-chaincode.tar.gz --path ../merca-chaincode --lang node --label merca-chaincode_1.0.0
+ res=0
Chaincode is packaged
```

3. Desde Merca-link nos comentan que los logs de los peers no proporcionan la información deseada. Modifica el logging de la red para que se ajuste a los siguientes requerimientos: - Los logs de los orderers deben estar en formato JSON. - Los logs de

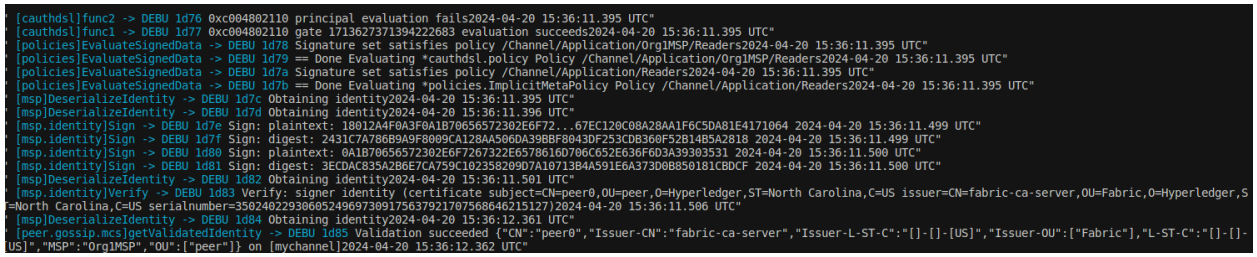
los peers deben de mostrar la fecha y la hora al final. - Dado que estamos en fase de pruebas, los logs de los peers deben de tener un nivel de logging de DEBUG por defecto. Además, los logs asociados al logger “gossip” tendrán un nivel por defecto de WARNING y los del logger de “chaincode” de INFO para no sobrecargar demasiado los logs.

Para realizar esto se toman los cod siguientes

Muestra solo logs con severidad error, panic y fatal. FABRIC_LOGGING_SPEC=error # Nivel por defecto es warning; logs de msp y gossip son info; logs de chaincode son error. FABRIC_LOGGING_SPEC=warning:msp,gossip=info:chaincode=error # Nivel por defecto es debug; logs de policies y orderer son info. FABRIC_LOGGING_SPEC=debug:policies,orderer=info

Muestra los logs en formato json FABRIC_LOGGING_FORMAT=json # Customización del formato de logging FABRIC_LOGGING_FORMAT="%{color}%{time:2006-01-02 15:04:05.000 MST} [%{module}] %{shortfunc} -> %{level:.4s} %{id:03x}%{color:reset} %{message}"

Al correr de nuevo se puede ver el nuevo formato incluyendo lo solicitado DEB, la fecha la final, etc.



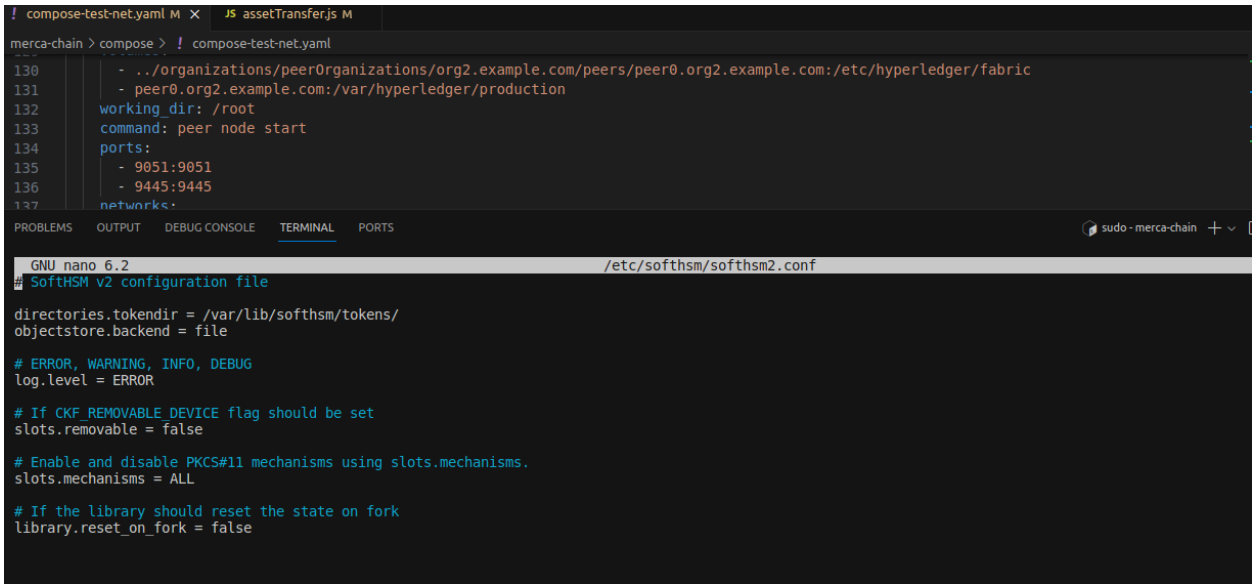
4. Los técnicos de Merca-link nos muestran su preocupación acerca de las claves criptográficas, que se almacenan en texto plano dentro de los directorios de la red. Como una primera medida de securización de las claves y de las operaciones criptográficas sugieren el acoplamiento de un HSM a las CAs, utilizándose softhsm2 para esta fase de pruebas y configurando un token diferente para cada CA.

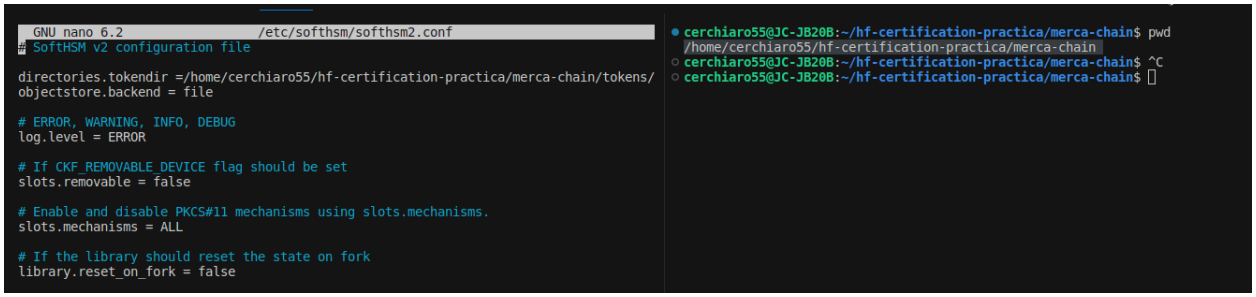
Descargas
sudo apt install softhsm2

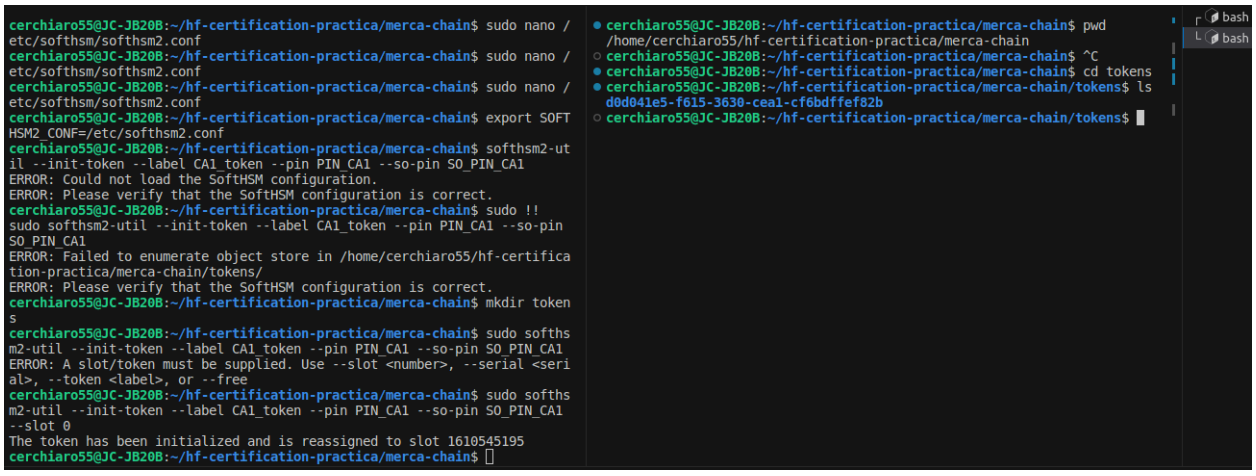
Tokens

softhsm2-util --init-token --slot <slot_number> --label <token_label> --pin <pin> --so-pin <so_pin>

softmh2







```
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-chain$ sudo !!
sudo softhsm2-util --show-slots
Available slots:
Slot 1610545195
    Slot info:
        Description:      SoftHSM slot ID 0x5ffef82b

        Manufacturer ID:  SoftHSM project
        Hardware version: 2.6
        Firmware version: 2.6
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.6
        Firmware version: 2.6
        Serial number:    cea1cf6bdffef82b
        Initialized:      yes
        User PIN init.:   yes
        Label:            CA1_token
Slot 1711334517
    Slot info:
        Description:      SoftHSM slot ID 0x6600e475

        Manufacturer ID:  SoftHSM project
        Hardware version: 2.6
        Firmware version: 2.6
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.6
        Firmware version: 2.6
        Serial number:    230b9044e600e475
        Initialized:      yes
        User PIN init.:   yes
        Label:            CA2_token
```

```
        Manufacturer ID:  SoftHSM project
        Hardware version: 2.6
        Firmware version: 2.6
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.6
        Firmware version: 2.6
        Serial number:    230b9044e600e475
        Initialized:      yes
        User PIN init.:   yes
        Label:            CA2_token
Slot 1903152988
    Slot info:
        Description:      SoftHSM slot ID 0x716fcf5c

        Manufacturer ID:  SoftHSM project
        Hardware version: 2.6
        Firmware version: 2.6
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.6
        Firmware version: 2.6
        Serial number:    85a7e767f16fcf5c
        Initialized:      yes
        User PIN init.:   yes
        Label:            CA_OR_token
Slot 3
    Slot info:
        Description:      SoftHSM slot ID 0x3

        Manufacturer ID:  SoftHSM project
        Hardware version: 2.6
        Firmware version: 2.6
        Token present:    yes
    Token info:
        Manufacturer ID:  SoftHSM project
        Model:            SoftHSM v2
        Hardware version: 2.6
        Firmware version: 2.6
        Serial number:
        Initialized:      no
        User PIN init.:   no
        Label:
```

```
# It can be any of: {"amcl.Fp256bn", "gurvy.Bn254", "amcl.Fp256Miraclbn"}.
# If unspecified, it defaults to 'amcl.Fp256bn'.
curve: amcl.Fp256bn

############################################################################
# BCCSP (BlockChain Crypto Service Provider) section is used to select which
# crypto library implementation to use
############################################################################
bccsp:
  default: PKCS11
  PKCS11:
    Library: /etc/hyperledger/fabric-ca-server/softhsm/libsofthsm2.so
    Pin: PIN_CA_OR
    Label: CA_OR_token
    hash: SHA2
    security: 256
    Immutable: false

############################################################################
# Multi CA section
#
# Each Fabric CA server contains one CA by default.  This section is used
# to configure multiple CAs in a single server.
#
```

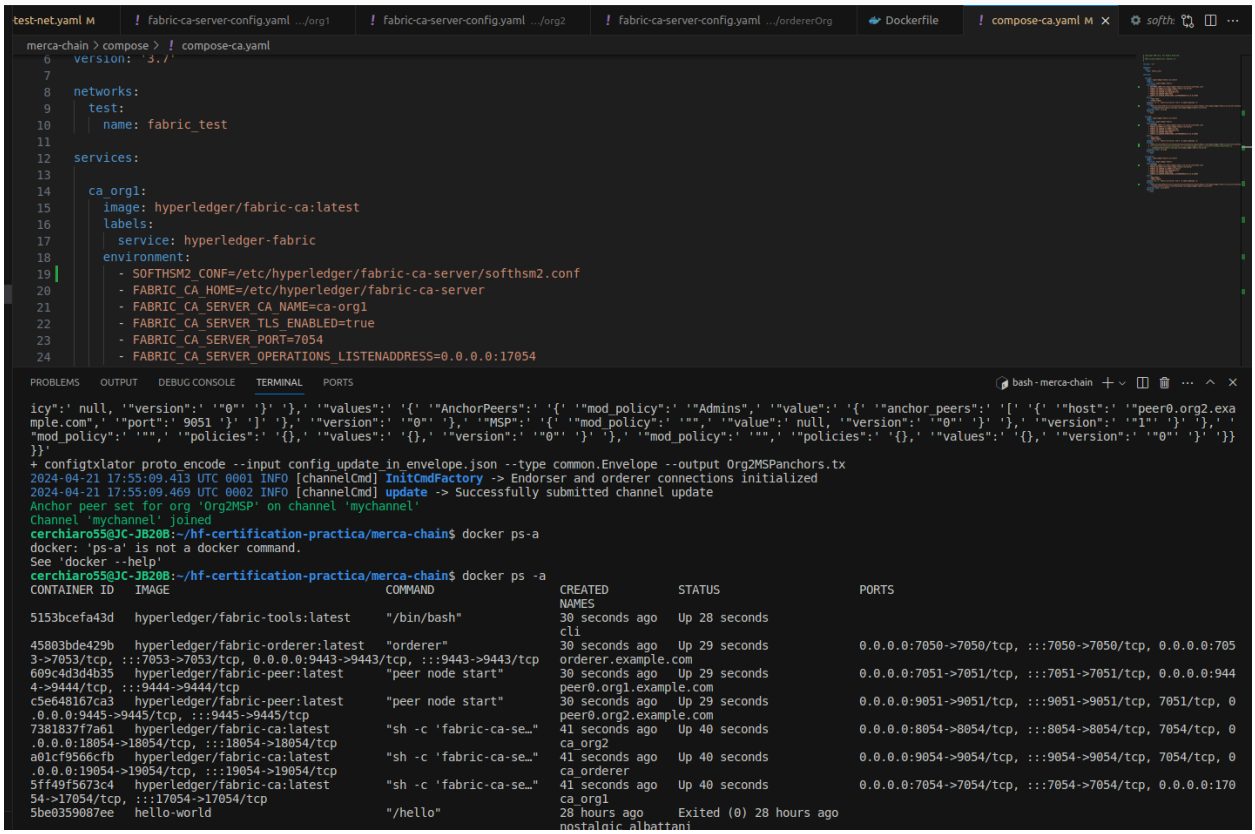
```
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - /usr/local/lib/softhsm/libsofthsm2.so:/etc/hyperledger/fabric-ca-server/softhsm/libsofthsm2.so
      - ../organizations/fabric-ca/org1:/etc/hyperledger/fabric-ca-server
    container_name: ca_org1
    networks:
      - test

  ca_org2:
    image: hyperledger/fabric-ca:latest
    labels:
      service: hyperledger-fabric
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_CA_NAME=ca-org2
      - FABRIC_CA_SERVER_TLS_ENABLED=true
      - FABRIC_CA_SERVER_PORT=8054
      - FABRIC_CA_SERVER_OPERATIONS_LISTENADDRESS=0.0.0.0:18054
    ports:
      - "8054:8054"
      - "18054:18054"
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - /usr/local/lib/softhsm/libsofthsm2.so:/etc/hyperledger/fabric-ca-server/softhsm/libsofthsm2.so
```

```
      - FABRIC_CA_SERVER_CA_NAME=ca-orderer
      - FABRIC_CA_SERVER_TLS_ENABLED=true
      - FABRIC_CA_SERVER_PORT=9054
      - FABRIC_CA_SERVER_OPERATIONS_LISTENADDRESS=0.0.0.0:19054
    ports:
      - "9054:9054"
      - "19054:19054"
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - /home/cerchiaro55/hf-certification-practica/merca-chain/tokens:/etc/hyperledger/fabric-ca-server/softhsm/tokens
      - /usr/local/lib/softhsm/libsofthsm2.so:/etc/hyperledger/fabric-ca-server/softhsm/libsofthsm2.so
      - ../organizations/fabric-ca/ordererOrg:/etc/hyperledger/fabric-ca-server
    container_name: ca_orderer
    networks:
      - test
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Model:            SoftHSM v2
Hardware version: 2.6
Firmware version: 2.6
Serial number:    85a7e767f16fcf5c
Initialized:      yes
User PIN init.:   yes
```

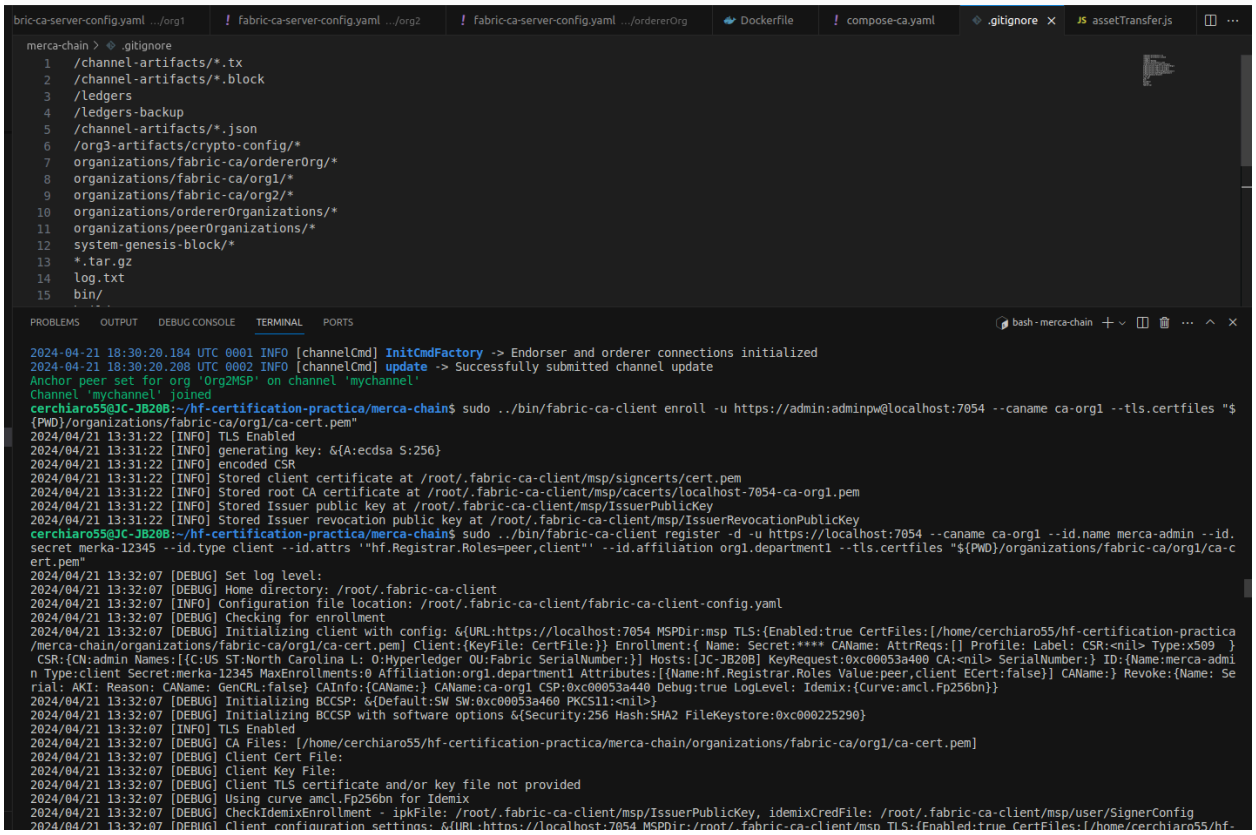
```
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-ch
/home/cerchiaro55/hf-certification-practica/merca-chain
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-ch
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-ch
cerchiaro55@JC-JB20B:~/hf-certification-practica/merca-ch
d0d041e5-f615-3630-cea1-cf6bdffef82b
```



5. Así mismo, explícales cómo configurar y utilizar el fabric-ca-client con el softHSM configurado. Registra, a modo de prueba, un usuario de nombre “merca-admin” y contraseña “merka-12345”, que tenga capacidad para registrar usuarios clientes y nuevos peers.

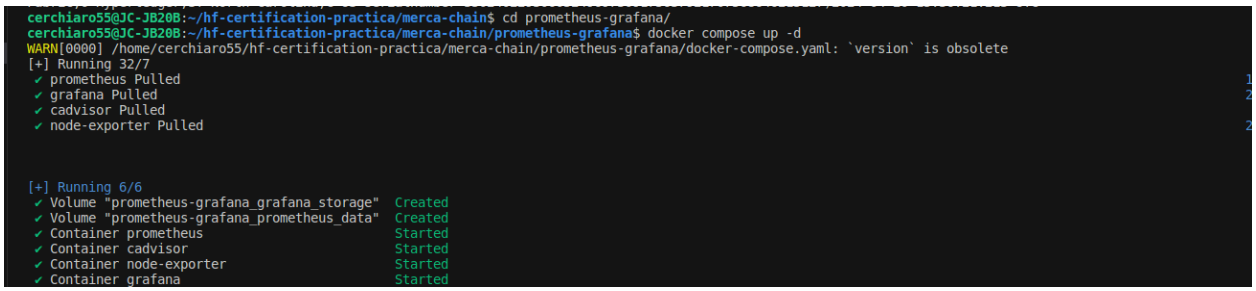
Se utilizan comandos

sudo ../bin/fabric-ca-client register -d -u https://localhost:7054 --caname ca-org1 --id.name merca-admin --id.secret merka-12345 --id.type client --id.attrs '"hf.Registrar.Roles=peer,client"' --id.affiliation org1.department1 --tls.certfiles "${PWD}/organizations/fabric-ca/org1/ca-cert.pem"



6. Los merca-ingenieros de Merca-link consideran que tener herramientas de monitorización a su disposición agilizaría estas tareas de administración en futuras ocasiones. Despliega sendas instancias de Prometheus y Grafana para satisfacer estas necesidades.

A continuación se despliegan grafana y prometheus se entra al folder y se ejecuta el comando



7. Investiga sobre los despliegues distribuidos (esto es, en distintos nodos) de Hyperledger Fabric, y resume en unas pocas líneas tus recomendaciones y los principales puntos a tener en cuenta para que Merka-link despliegue su red de forma distribuida. No olvides referencias los artículos/blogs/páginas web que has consultado.Despliegues distribuidos

El despliegue de nodos distribuidos implica la configuración y conexión de múltiples nodos de manera geográficamente dispersa. Esto se puede lograr mediante tecnologías como contenedores y orquestadores, como Kubernetes, que permiten la gestión centralizada de los nodos y facilitan la escalabilidad y la alta disponibilidad de las aplicaciones distribuidas. Además, se pueden emplear herramientas de automatización para simplificar el proceso de implementación y mantenimiento de la infraestructura distribuida.

```
# docker-compose.yml

version: '3'

services:
  node1:
    image: tu_imagen_de_aplicacion
    ports:
      - "8001:8000"
    networks:
      - app-network

  node2:
    image: tu_imagen_de_aplicacion
    ports:
      - "8002:8000"
    networks:
      - app-network

networks:
  app-network:
    driver: bridge
```

Este archivo de Docker Compose define dos servicios, `node1` y `node2`, cada uno utilizando la misma imagen de aplicación pero exponiendo puertos diferentes. Puedes ejecutar este archivo usando el comando `docker-compose up -d` para lanzar los nodos distribuidos.

Este es un ejemplo básico y puedes personalizarlo según tus necesidades, como agregar más nodos, configurar volúmenes compartidos, o incluir un orquestador como Kubernetes para una gestión más avanzada.

https://hyperledger-fabric.readthedocs.io/es/latest/cc_service.html?highlight=distributed%20deployments#chaincode-as-an-external-service

https://hyperledger-fabric.readthedocs.io/es/latest/cc_service.html?highlight=distributed%20deployments#chaincode-as-an-external-service