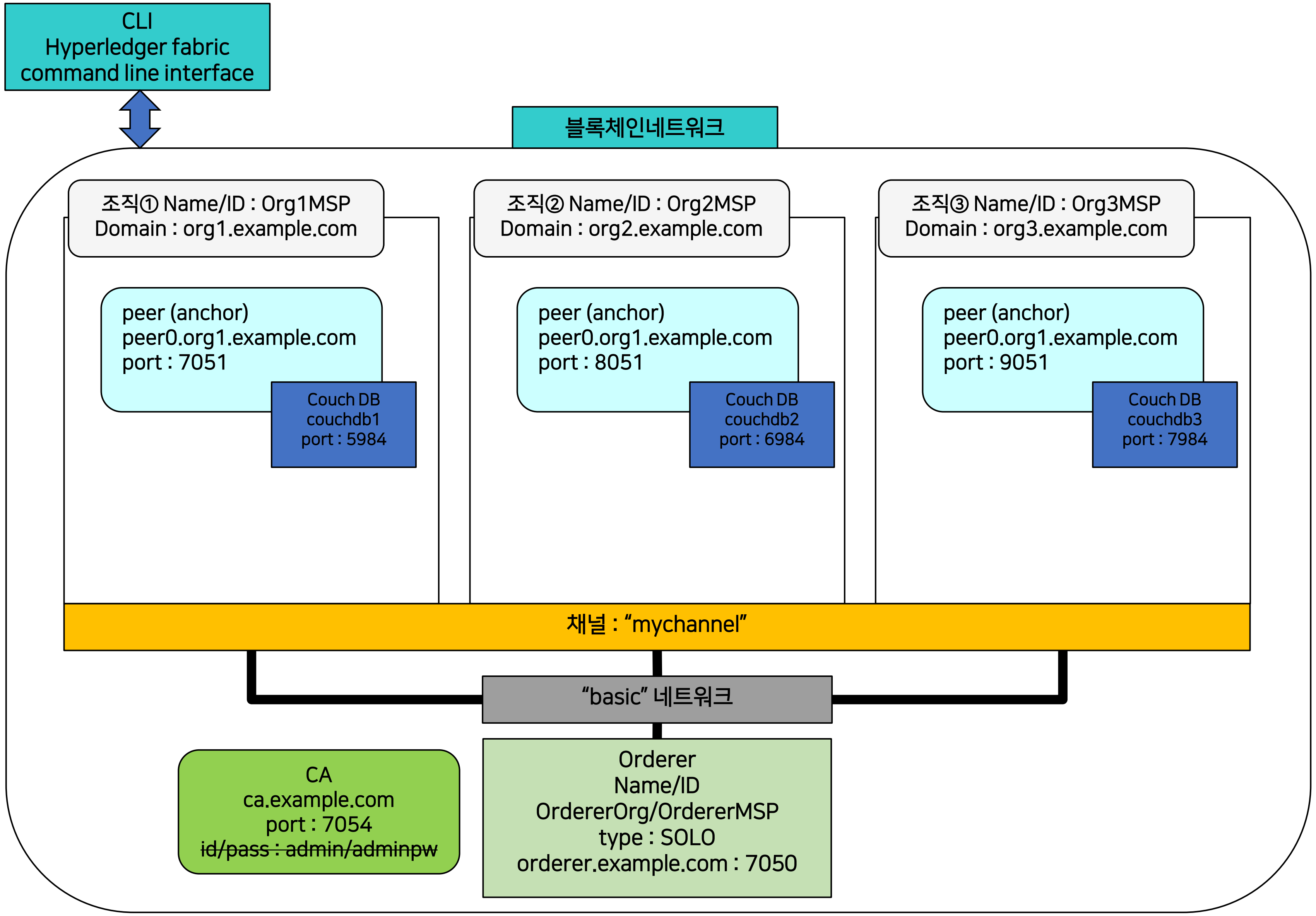
네트워크구성 확장

- Basic-network를 수정하여 다른 형태의 네트워크 구성
  - 3개의 Organization
  - CA 구성
  - 각 Organization에 ancker peer 구성
  - TLS 암호화 통신 구현 (필요 시)
  - 3개의 CouchDB

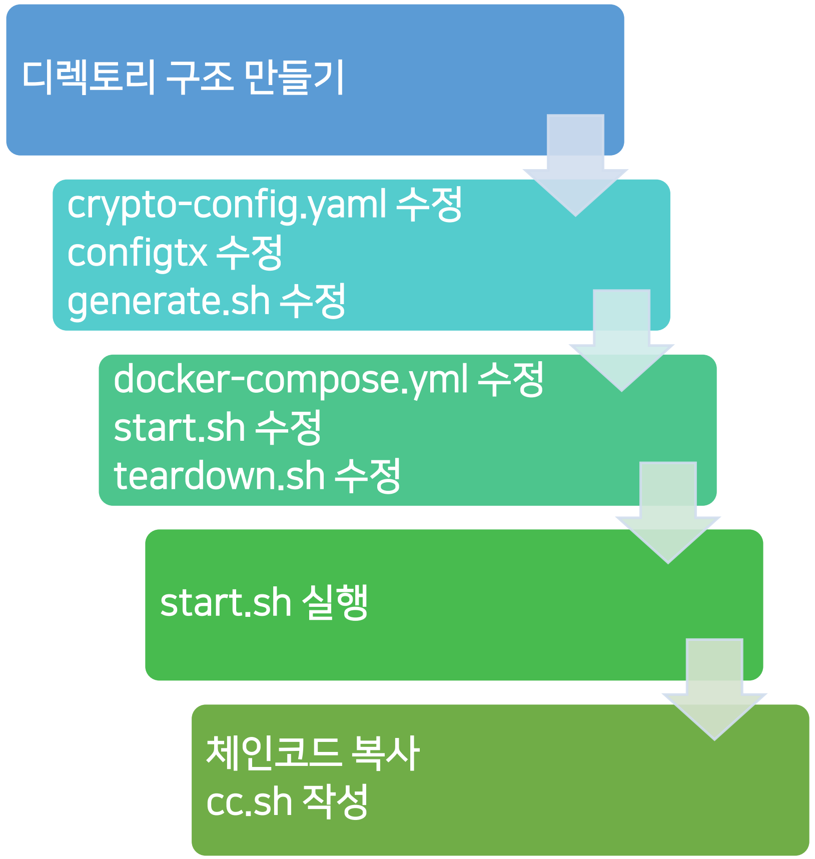
네트워크구성 확장



CLI
Hyperledger fabric
command line interface
블록체인네트워크
조직① Name/ID : Org1MSP
Domain : org1.example.com
peer (anchor)
peer0.org1.example.com
port : 7051
Couch DB
couchdb1
port : 5984
조직② Name/ID : Org2MSP
Domain : org2.example.com
peer (anchor)
peer0.org1.example.com
port : 8051
Couch DB
couchdb2
port : 6984
조직③ Name/ID : Org3MSP
Domain : org3.example.com
peer (anchor)
peer0.org1.example.com
port : 9051
Couch DB
couchdb3
port : 7984
채널 : "mychannel"
"basic" 네트워크
CA
ca.example.com
port : 7054
~~id/pass : admin/adminpw~~
Orderer
Name/ID
OrdererOrg/OrdererMSP
type : SOLO
orderer.example.com : 7050

네트워크구성 확장

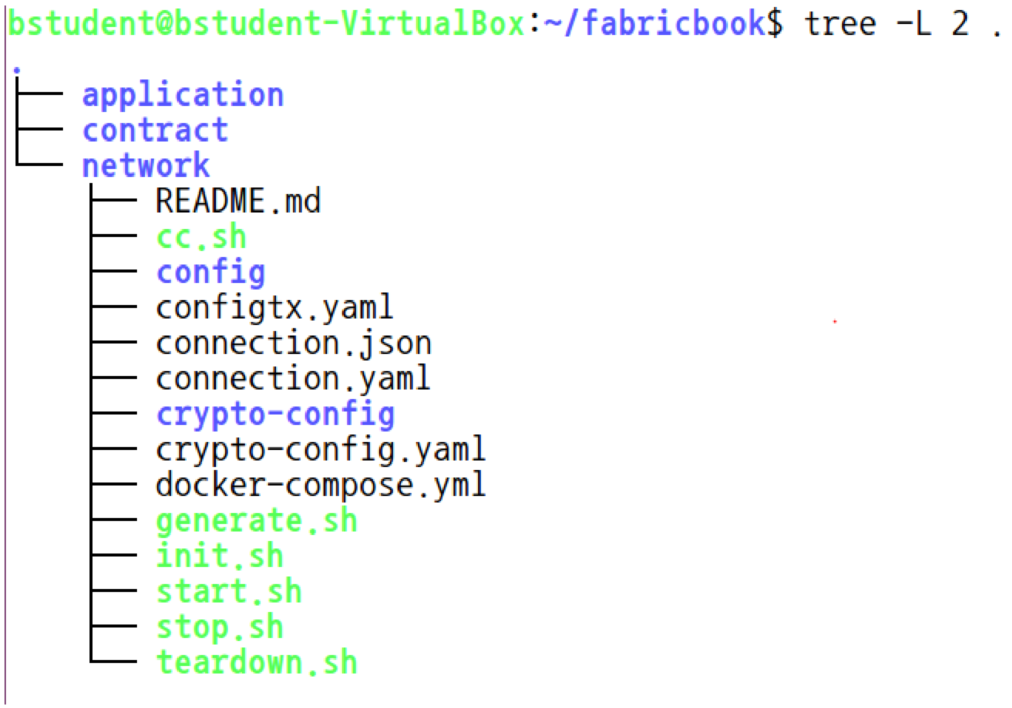
- 네트워크 확장시 작성 필요한 파일
  - .env (숨김파일)
  - configtx.yaml
  - crypto-config.yaml
  - generate.sh
  - docker-compose.yml
  - start.sh
  - teardown.sh



네트워크구성 확장

- 양식 복사 및 수정
  - 새로운 네트워크 생성의 양식인 'basic-network'복사
  - 홈 디렉토리에 'fabricbook'디렉토리를 생성하여 복사

```
~$cp -r fabric-samples/basic-network/ fabricbook/network
~$cd fabricbook/
~$mkdir application
~$mkdir contract
```



```
bstudent@bstudent-VirtualBox:~/fabricbook$ tree -L 2 .
.
├── application
├── contract
└── network
    ├── README.md
    ├── cc.sh
    ├── config
    ├── configtx.yaml
    ├── connection.json
    ├── connection.yaml
    ├── crypto-config
    ├── crypto-config.yaml
    ├── docker-compose.yml
    ├── generate.sh
    ├── init.sh
    ├── start.sh
    ├── stop.sh
    └── teardown.sh
```

네트워크구성 확장

- crypto-config.yaml
  - cryptogen 툴이 crypto-config.yaml 파일 사용
  - crypto-config.yaml 파일을 이용해서 organization과 그 구성원들에게 인증서 발급
  - peer와 user수 설정

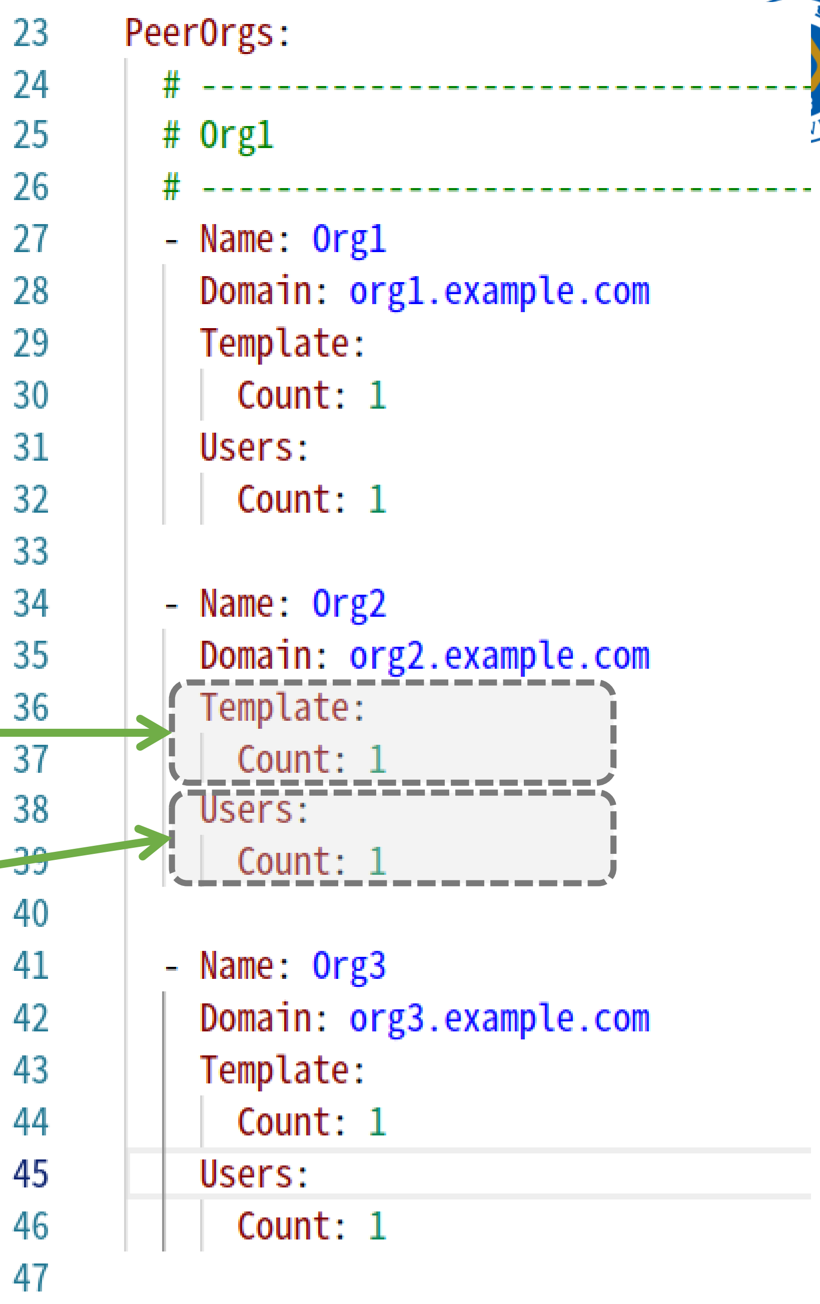
```
- Name: Org2
  Domain: org2.example.com
  Template:
    Count: 1
  Users:
    Count: 1
```

네트워크 준비

- crypto-config.yaml 수정

생성될 peer의 수

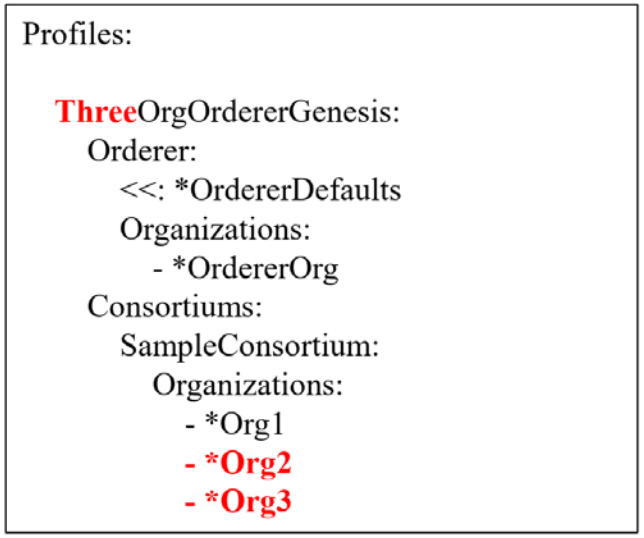
생성될 user(1~n)@org1.example의 수



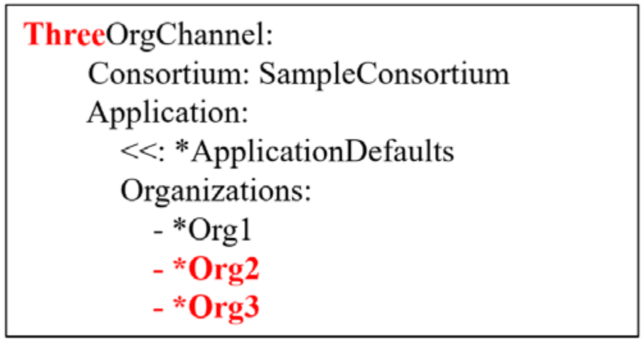
```yaml
PeerOrgs:
  # ---------------------------------------------------------------------------
  # Org1
  # ---------------------------------------------------------------------------
  - Name: Org1
    Domain: org1.example.com
    Template:
      Count: 1
    Users:
      Count: 1

  - Name: Org2
    Domain: org2.example.com
    Template:
      Count: 1
    Users:
      Count: 1

  - Name: Org3
    Domain: org3.example.com
    Template:
      Count: 1
    Users:
      Count: 1

```

네트워크 준비

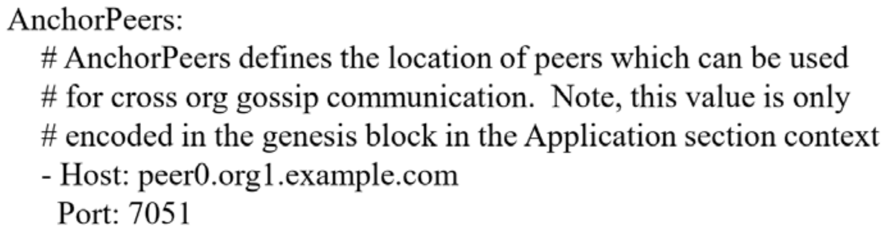
- configtx.yaml
  - configtxgen 툴이 configtx.yaml파일 사용
  - 네트워크의 channel과 genesis block 생성을 위한 설정
  - anchor peer 설정
  - orderer 설정
  - 네트워크 전체의 구조 및 설정 내용 포함
  - Organization 생성 (3개의 Organization 생성)



```
Profiles:

  ThreeOrgOrdererGenesis:
    Orderer:
      <<: *OrdererDefaults
      Organizations:
        - *OrdererOrg
    Consortiums:
      SampleConsortium:
        Organizations:
          - *Org1
          - *Org2
          - *Org3
```



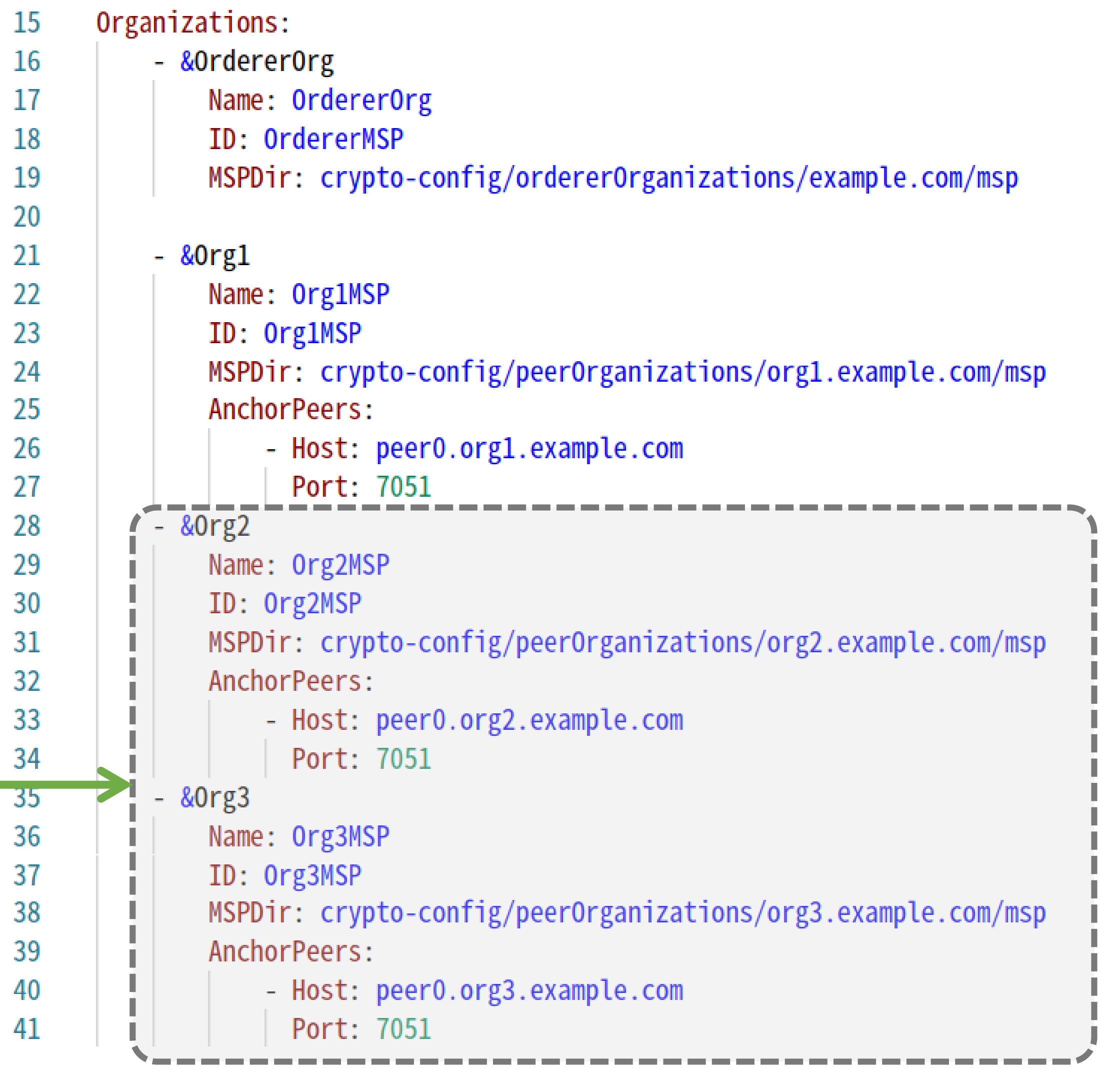
```
ThreeOrgChannel:
    Consortium: SampleConsortium
    Application:
      <<: *ApplicationDefaults
      Organizations:
        - *Org1
        - *Org2
        - *Org3
```



```
AnchorPeers:
  # AnchorPeers defines the location of peers which can be used
  # for cross org gossip communication.  Note, this value is only
  # encoded in the genesis block in the Application section context
  - Host: peer0.org1.example.com
    Port: 7051
```

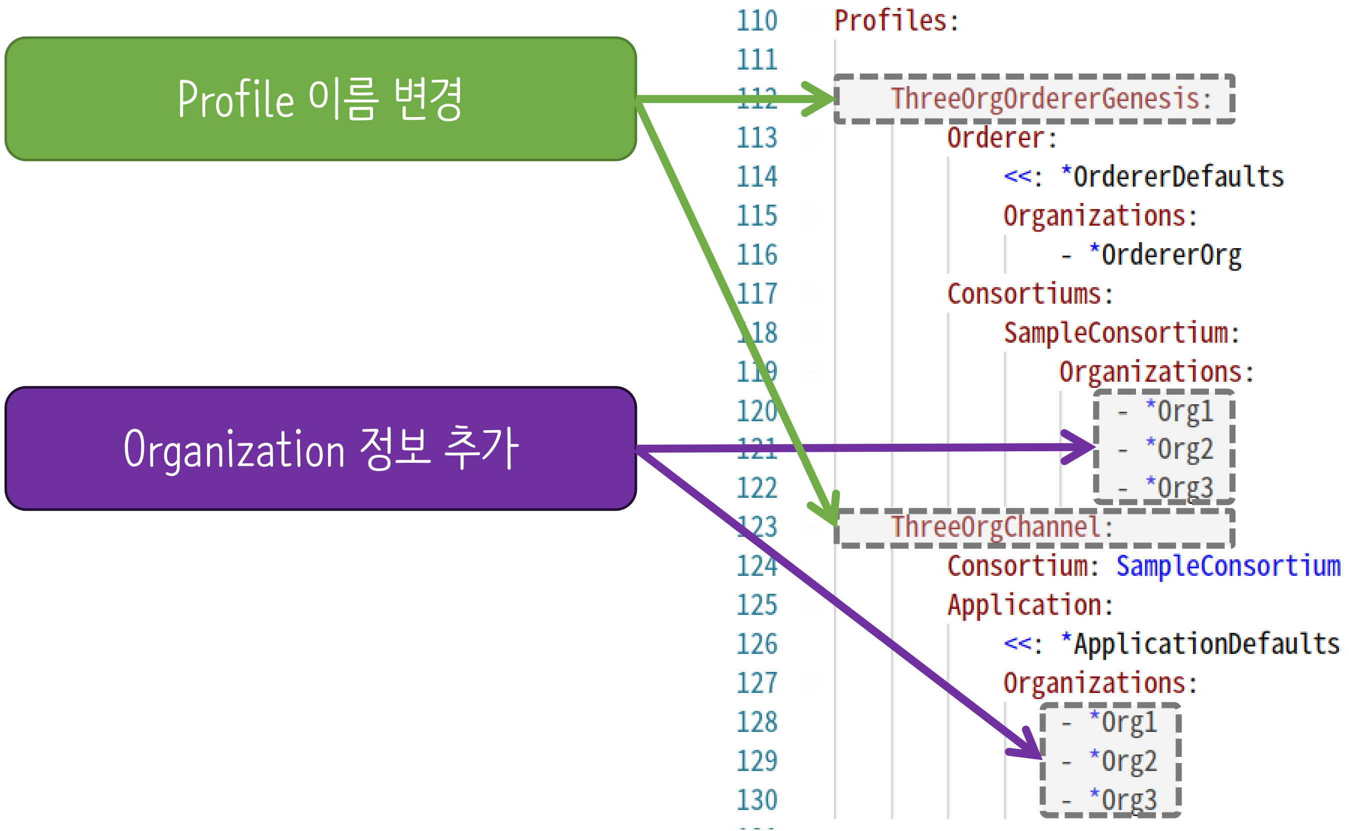
네트워크 준비

- configtx.yaml 수정



```yaml
Organizations:
    - &OrdererOrg
        Name: OrdererOrg
        ID: OrdererMSP
        MSPDir: crypto-config/ordererOrganizations/example.com/msp

    - &Org1
        Name: Org1MSP
        ID: Org1MSP
        MSPDir: crypto-config/peerOrganizations/org1.example.com/msp
        AnchorPeers:
            - Host: peer0.org1.example.com
              Port: 7051
    - &Org2
        Name: Org2MSP
        ID: Org2MSP
        MSPDir: crypto-config/peerOrganizations/org2.example.com/msp
        AnchorPeers:
            - Host: peer0.org2.example.com
              Port: 7051
    - &Org3
        Name: Org3MSP
        ID: Org3MSP
        MSPDir: crypto-config/peerOrganizations/org3.example.com/msp
        AnchorPeers:
            - Host: peer0.org3.example.com
              Port: 7051
```

Organization 정보 추가



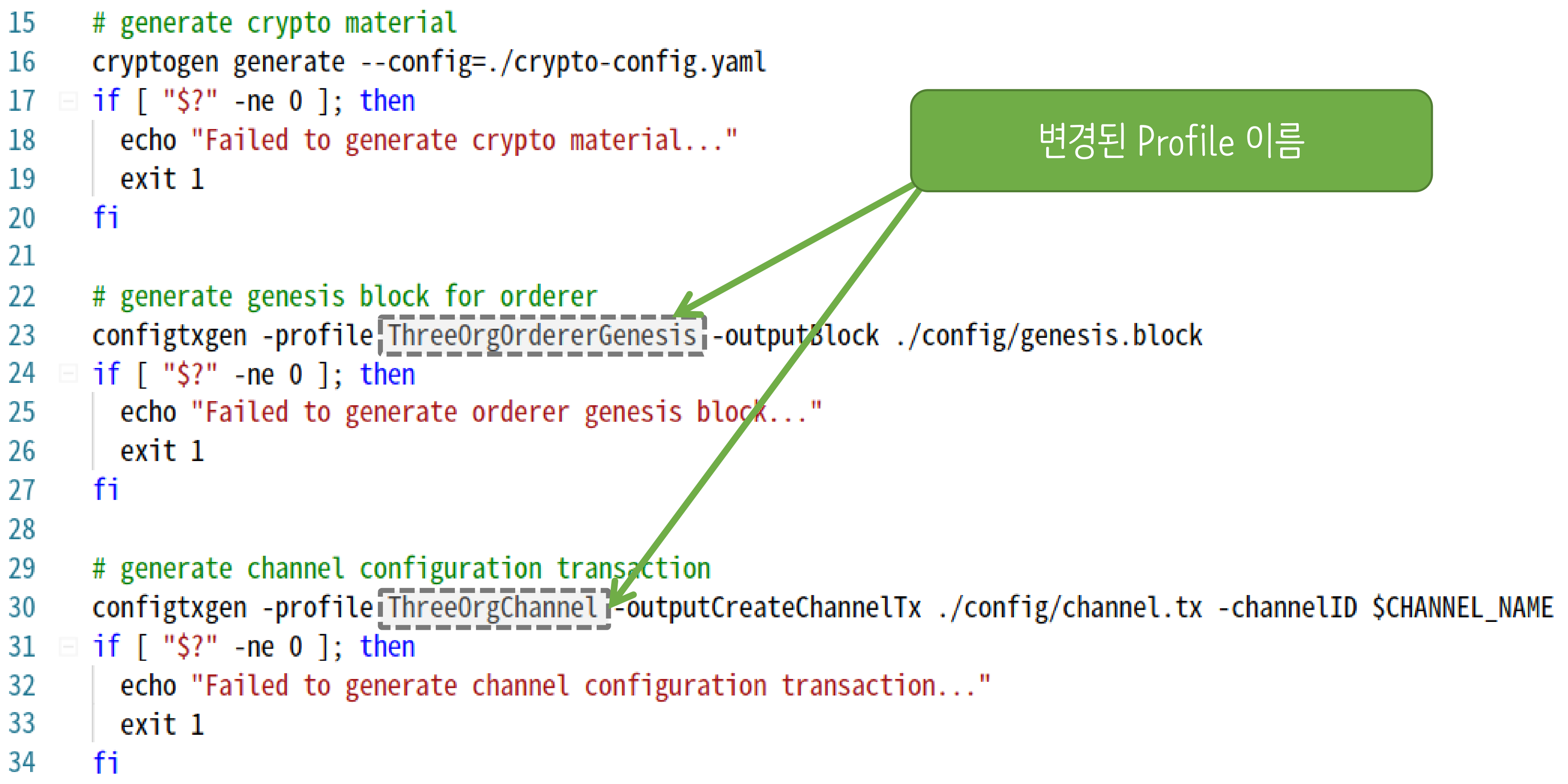
```
Profiles:

    ThreeOrgOrdererGenesis:
        Orderer:
            <<: *OrdererDefaults
            Organizations:
                - *OrdererOrg
        Consortiums:
            SampleConsortium:
                Organizations:
                    - *Org1
                    - *Org2
                    - *Org3
    ThreeOrgChannel:
        Consortium: SampleConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Org1
                - *Org2
                - *Org3
```

네트워크 준비

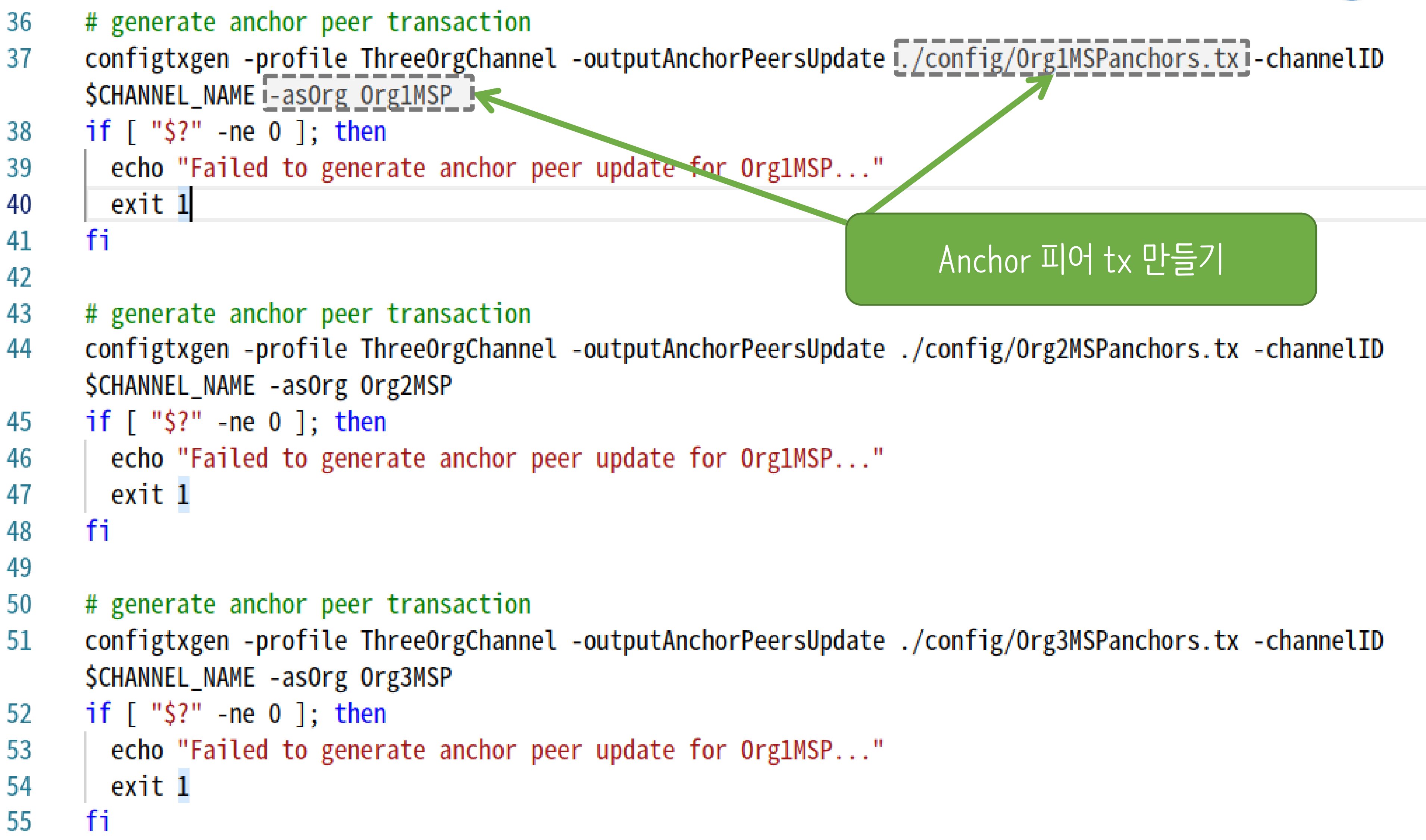
- generate.sh
  - cryptogen, configtxgen 툴을 사용하여 블록체인 네트워크를 위한 요소들 자동 생성 스크립트
    - 이전 crypto material and config transactions 삭제
    - crypto material 생성
    - genesis block for orderer 생성
    - channel configuration transaction 생성
    - anchor peer transaction 생성

네트워크 준비

- generate.sh



```
# generate crypto material
cryptogen generate --config=./crypto-config.yaml
if [ "$?" -ne 0 ]; then
  echo "Failed to generate crypto material..."
  exit 1
fi

# generate genesis block for orderer
configtxgen -profile ThreeOrgOrdererGenesis -outputBlock ./config/genesis.block
if [ "$?" -ne 0 ]; then
  echo "Failed to generate orderer genesis block..."
  exit 1
fi

# generate channel configuration transaction
configtxgen -profile ThreeOrgChannel -outputCreateChannelTx ./config/channel.tx -channelID $CHANNEL_NAME
if [ "$?" -ne 0 ]; then
  echo "Failed to generate channel configuration transaction..."
  exit 1
fi
```

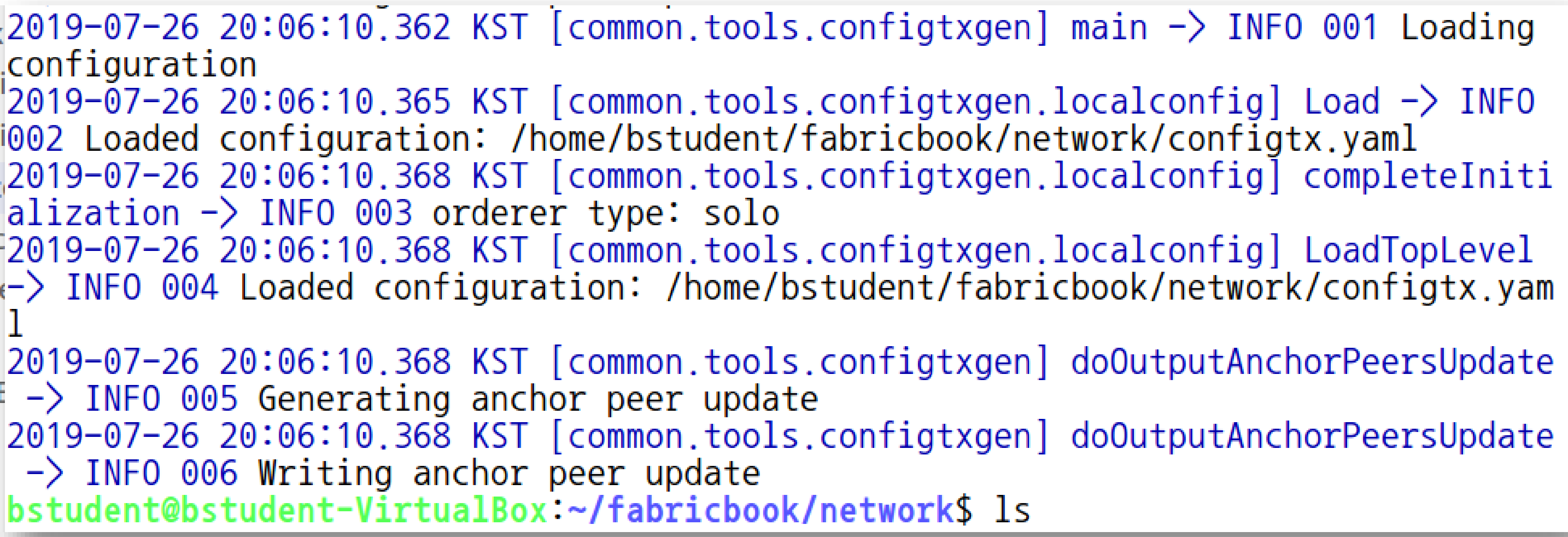


```
# generate anchor peer transaction
configtxgen -profile ThreeOrgChannel -outputAnchorPeersUpdate ./config/Org1MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org1MSP
if [ "$?" -ne 0 ]; then
  echo "Failed to generate anchor peer update for Org1MSP..."
  exit 1
fi

# generate anchor peer transaction
configtxgen -profile ThreeOrgChannel -outputAnchorPeersUpdate ./config/Org2MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org2MSP
if [ "$?" -ne 0 ]; then
  echo "Failed to generate anchor peer update for Org1MSP..."
  exit 1
fi

# generate anchor peer transaction
configtxgen -profile ThreeOrgChannel -outputAnchorPeersUpdate ./config/Org3MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org3MSP
if [ "$?" -ne 0 ]; then
  echo "Failed to generate anchor peer update for Org1MSP..."
  exit 1
fi
```

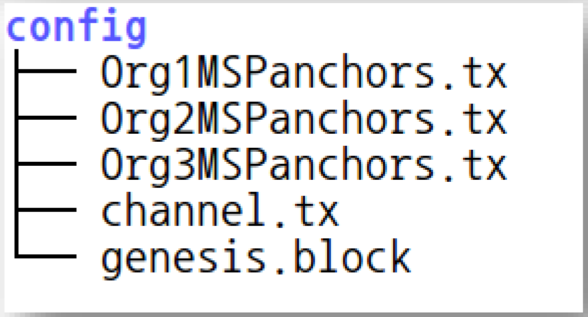
Anchor 피어 tx 만들기

네트워크 준비

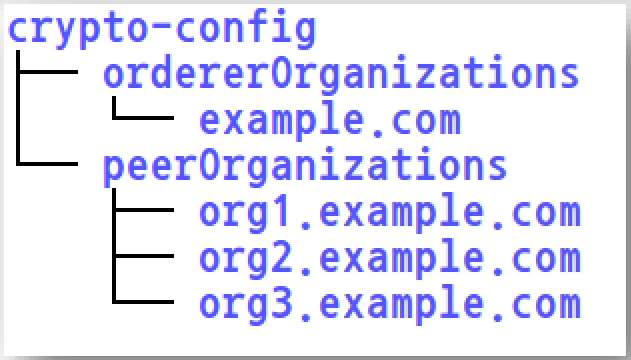
- generate.sh 수행



```
2019-07-26 20:06:10.362 KST [common.tools.configtxgen] main -> INFO 001 Loading
configuration
2019-07-26 20:06:10.365 KST [common.tools.configtxgen.localconfig] Load -> INFO
002 Loaded configuration: /home/bstudent/fabricbook/network/configtx.yaml
2019-07-26 20:06:10.368 KST [common.tools.configtxgen.localconfig] completeIniti
alization -> INFO 003 orderer type: solo
2019-07-26 20:06:10.368 KST [common.tools.configtxgen.localconfig] LoadTopLevel
-> INFO 004 Loaded configuration: /home/bstudent/fabricbook/network/configtx.yam
l
2019-07-26 20:06:10.368 KST [common.tools.configtxgen] doOutputAnchorPeersUpdate
 -> INFO 005 Generating anchor peer update
2019-07-26 20:06:10.368 KST [common.tools.configtxgen] doOutputAnchorPeersUpdate
 -> INFO 006 Writing anchor peer update
bstudent@bstudent-VirtualBox:~/fabricbook/network$ ls
```



```
config
├── Org1MSPanchors.tx
├── Org2MSPanchors.tx
├── Org3MSPanchors.tx
├── channel.tx
└── genesis.block
```

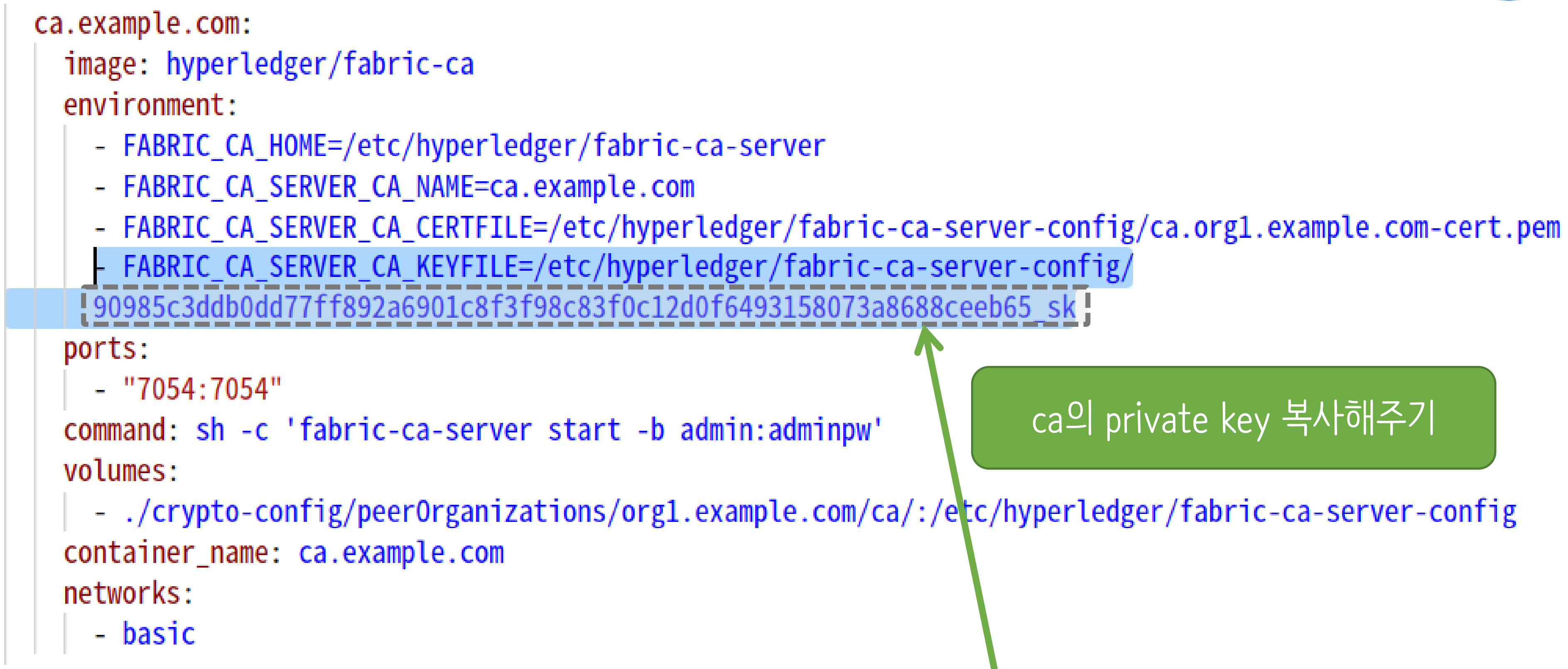


```
crypto-config
├── ordererOrganizations
│   └── example.com
└── peerOrganizations
    ├── org1.example.com
    ├── org2.example.com
    └── org3.example.com
```

네트워크구성 확장

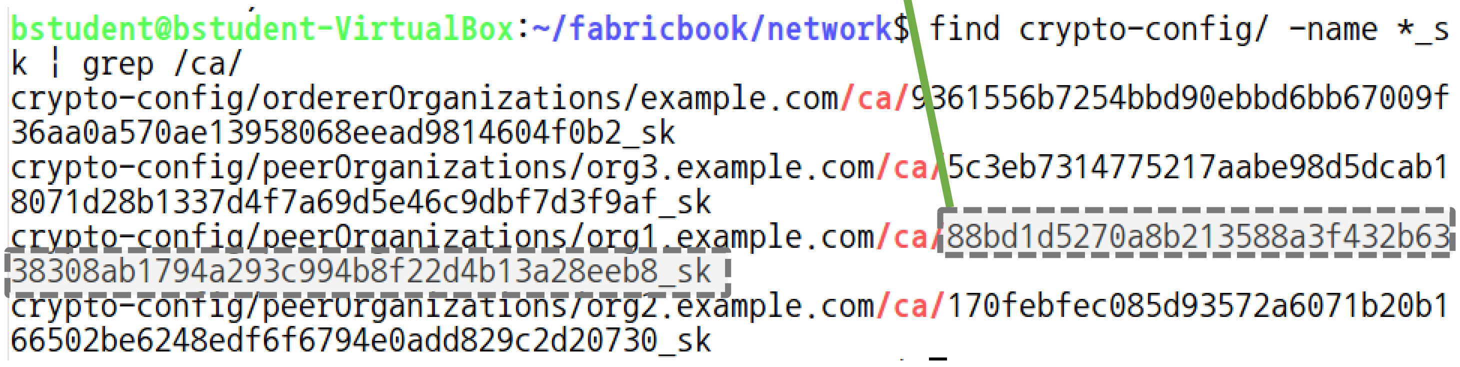
- docker-compose.yml
  - 여러 컨테이너를 일괄 관리할 수 있는 "docker compose"의 구성 관리 파일
  - docker-compose 파일 수정 요소
    - CA 컨테이너와 peer 컨테이너의 정의
      - Org2, Org3 추가
    - CA 관리자(Admin) 패스워드 변경 (필요 시)
    - TLS 암호통신 유효화 (필요 시)
    - CLI 컨테이너 구성
    - CouchDB 컨테이너 추가 및 수정
      - couchdb1, couchdb2, couchdb3

docker-compose파일 수정



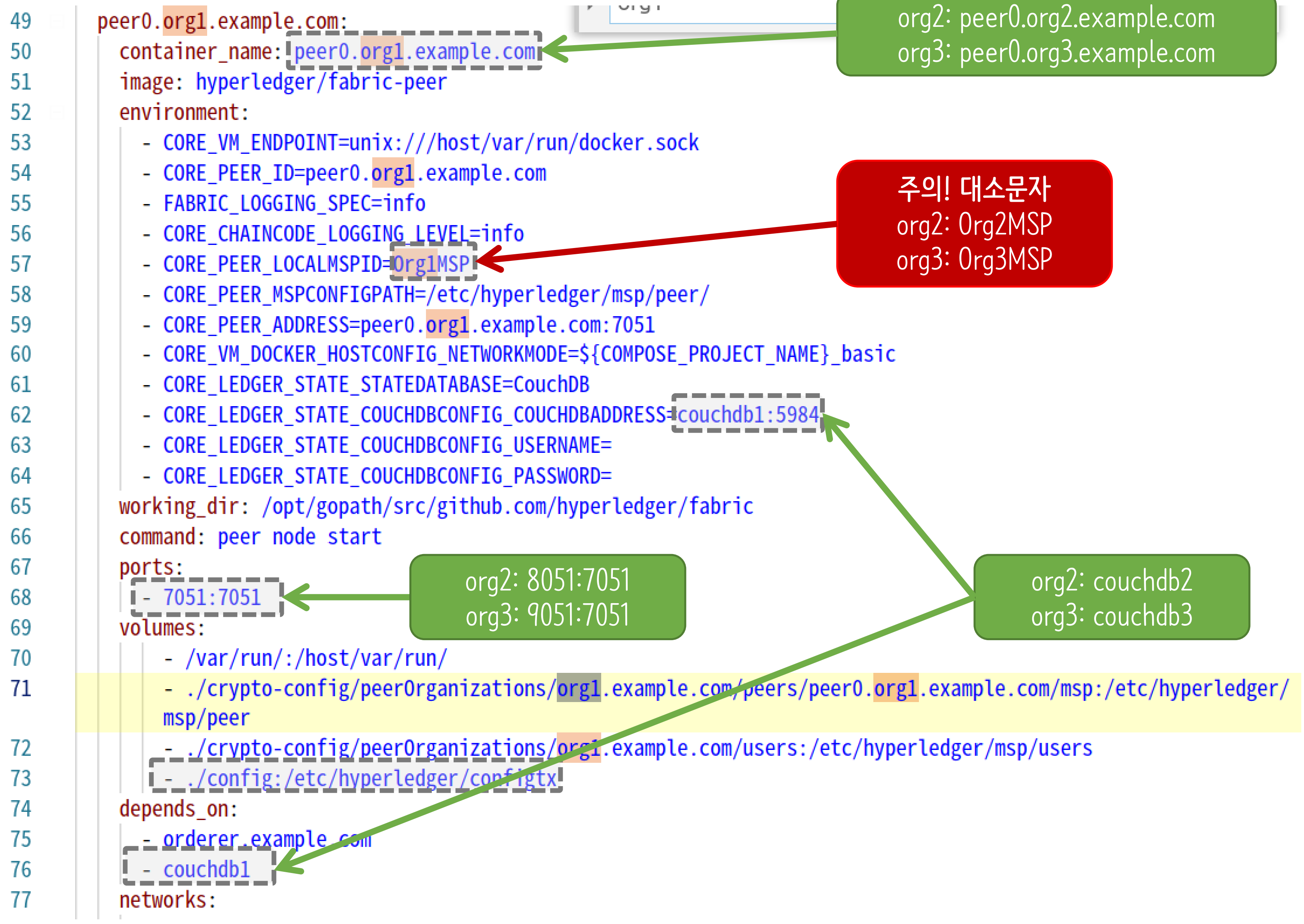
```yaml
ca.example.com:
  image: hyperledger/fabric-ca
  environment:
    - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
    - FABRIC_CA_SERVER_CA_NAME=ca.example.com
    - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-server-config/ca.org1.example.com-cert.pem
    - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-server-config/
    90985c3ddb0dd77ff892a6901c8f3f98c83f0c12d0f6493158073a8688ceeb65_sk
  ports:
    - "7054:7054"
  command: sh -c 'fabric-ca-server start -b admin:adminpw'
  volumes:
    - ./crypto-config/peerOrganizations/org1.example.com/ca/:/etc/hyperledger/fabric-ca-server-config
  container_name: ca.example.com
  networks:
    - basic
```

ca의 private key 복사해주기



```
bstudent@bstudent-VirtualBox:~/fabricbook/network$ find crypto-config/ -name *_s
k | grep /ca/
crypto-config/ordererOrganizations/example.com/ca/9361556b7254bbd90ebbd6bb67009f
36aa0a570ae13958068eead9814604f0b2_sk
crypto-config/peerOrganizations/org3.example.com/ca/5c3eb7314775217aabe98d5dcab1
8071d28b1337d4f7a69d5e46c9dbf7d3f9af_sk
crypto-config/peerOrganizations/org1.example.com/ca/88bd1d5270a8b213588a3f432b63
38308ab1794a293c994b8f22d4b13a28eeb8_sk
crypto-config/peerOrganizations/org2.example.com/ca/170febfec085d93572a6071b20b1
66502be6248edf6f6794e0add829c2d20730_sk
```

docker-compose파일 수정



```
peer0.org1.example.com:
  container_name: peer0.org1.example.com
  image: hyperledger/fabric-peer
  environment:
    - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
    - CORE_PEER_ID=peer0.org1.example.com
    - FABRIC_LOGGING_SPEC=info
    - CORE_CHAINCODE_LOGGING_LEVEL=info
    - CORE_PEER_LOCALMSPID=Org1MSP
    - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/peer/
    - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
    - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}_basic
    - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
    - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb1:5984
    - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
    - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
  working_dir: /opt/gopath/src/github.com/hyperledger/fabric
  command: peer node start
  ports:
    - 7051:7051
  volumes:
      - /var/run/:/host/var/run/
      - ./crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperledger/
      msp/peer
      - ./crypto-config/peerOrganizations/org1.example.com/users:/etc/hyperledger/msp/users
      - ./config:/etc/hyperledger/configtx
  depends_on:
    - orderer.example.com
    - couchdb1
  networks:
```

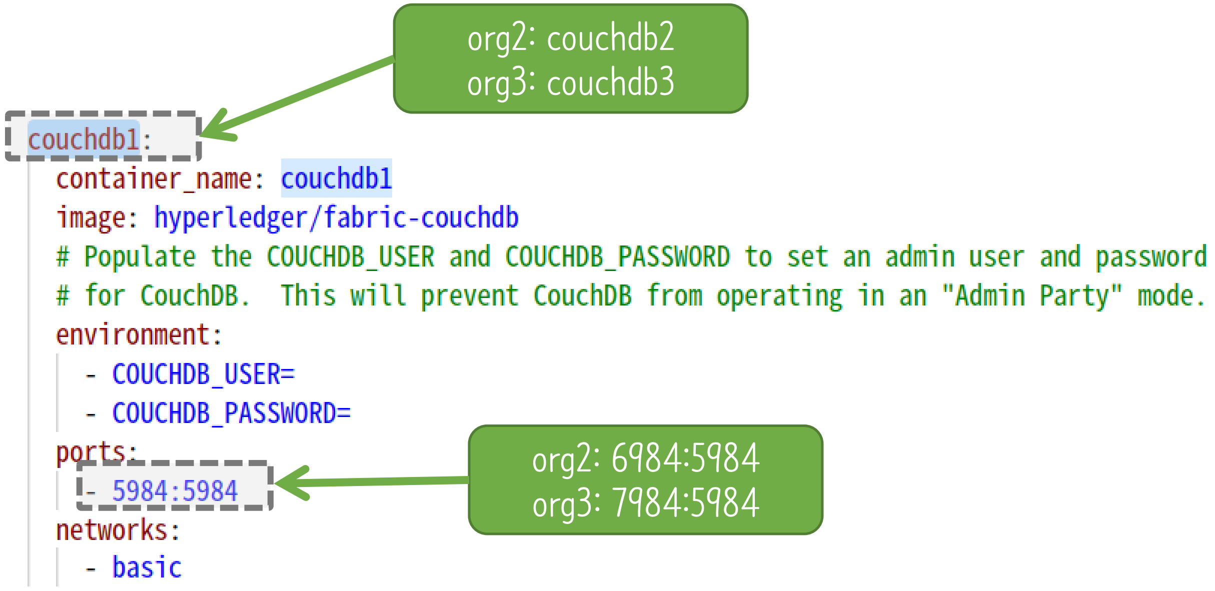
org2: peer0.org2.example.com
org3: peer0.org3.example.com

주의! 대소문자
org2: Org2MSP
org3: Org3MSP

org2: 8051:7051
org3: 9051:7051

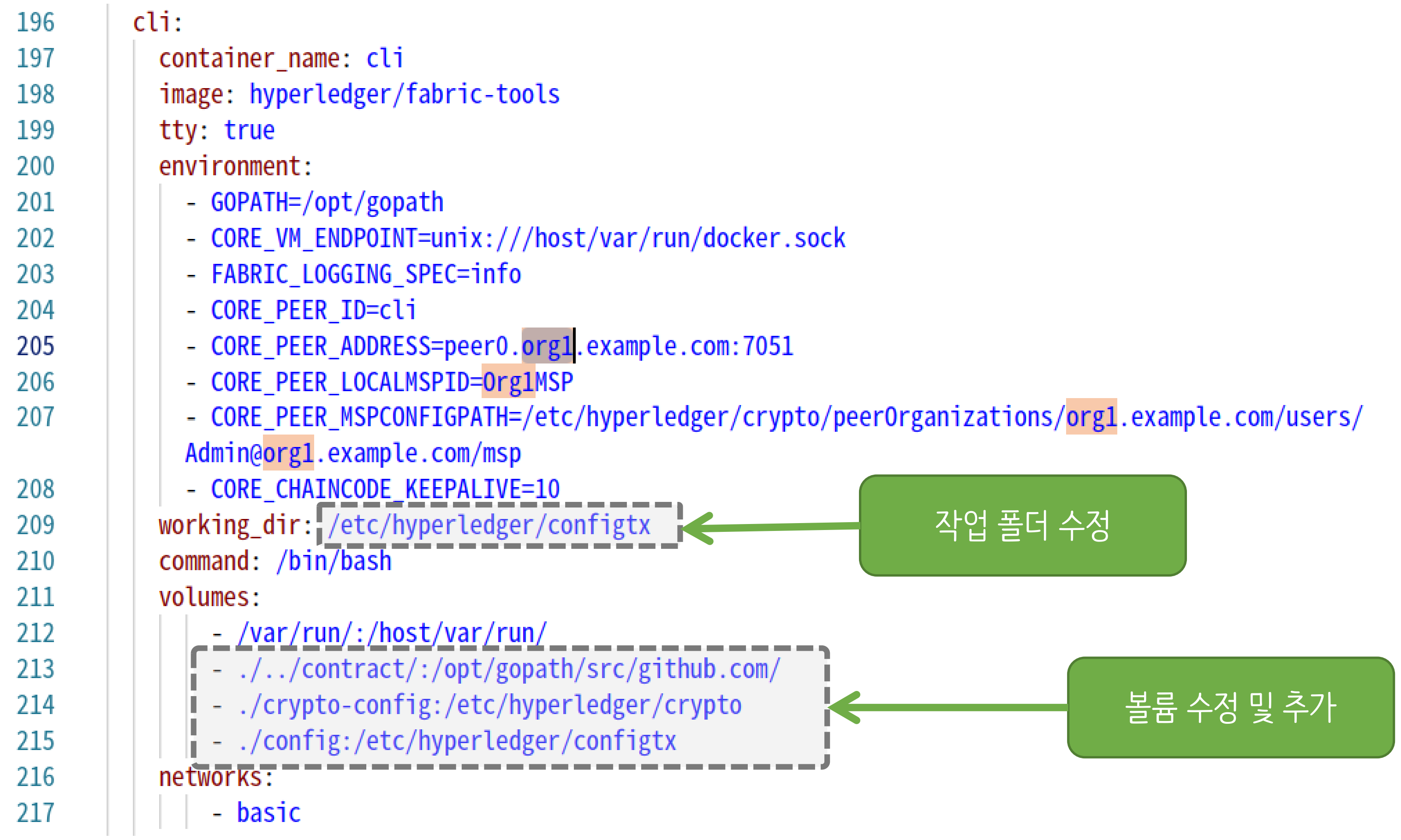
org2: couchdb2
org3: couchdb3

docker-compose파일 수정



```
couchdb1:
  container_name: couchdb1
  image: hyperledger/fabric-couchdb
  # Populate the COUCHDB_USER and COUCHDB_PASSWORD to set an admin user and password
  # for CouchDB.  This will prevent CouchDB from operating in an "Admin Party" mode.
  environment:
    - COUCHDB_USER=
    - COUCHDB_PASSWORD=
  ports:
    - 5984:5984
  networks:
    - basic
```

docker-compose파일 수정



```
cli:
  container_name: cli
  image: hyperledger/fabric-tools
  tty: true
  environment:
    - GOPATH=/opt/gopath
    - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
    - FABRIC_LOGGING_SPEC=info
    - CORE_PEER_ID=cli
    - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
    - CORE_PEER_LOCALMSPID=Org1MSP
    - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/crypto/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
    - CORE_CHAINCODE_KEEPALIVE=10
  working_dir: /etc/hyperledger/configtx
  command: /bin/bash
  volumes:
      - /var/run/:/host/var/run/
      - ./../contract/:/opt/gopath/src/github.com/
      - ./crypto-config:/etc/hyperledger/crypto
      - ./config:/etc/hyperledger/configtx
  networks:
      - basic
```

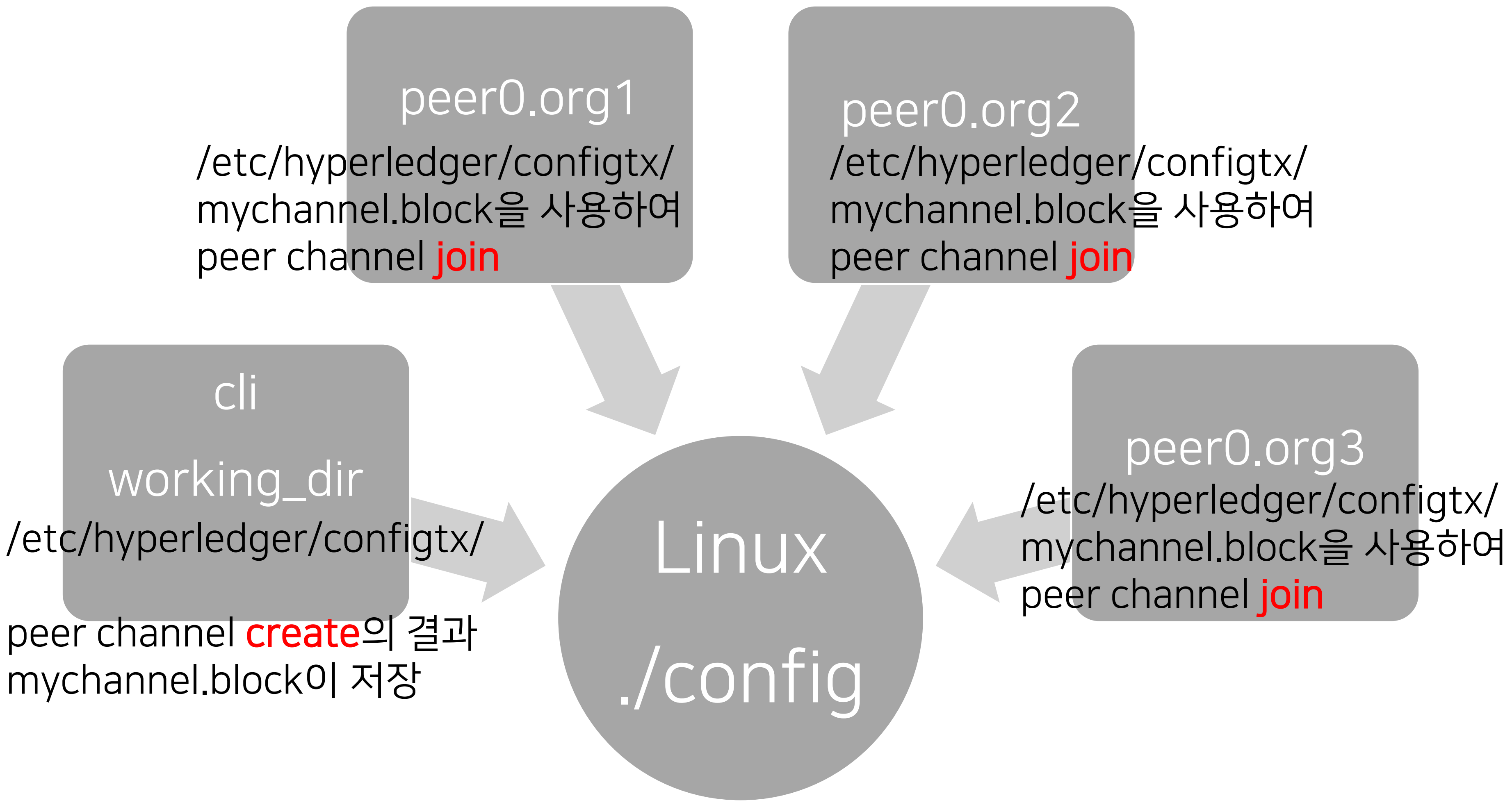
start.sh 수정

```
docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com couchdb1 couchdb2 couchdb3 peer0.org1.example.com  peer0.org2.example.com peer0.org3.example.com cli
docker ps -a
```

실행될 컨테이너 목록 수정

```
# Create the channel
docker exec cli peer channel create -o orderer.example.com:7050 -c mychannel -f /etc/hyperledger/configtx/channel.tx
# Join peer0.org1.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example.com/msp" peer0.org1.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
sleep 5
# Join peer0.org2.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org2MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org2.example.com/msp" peer0.org2.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
sleep 5
# Join peer0.org2.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org3MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org3.example.com/msp" peer0.org3.example.com peer channel join -b /etc/hyperledger/configtx/mychannel.block
sleep 5
```

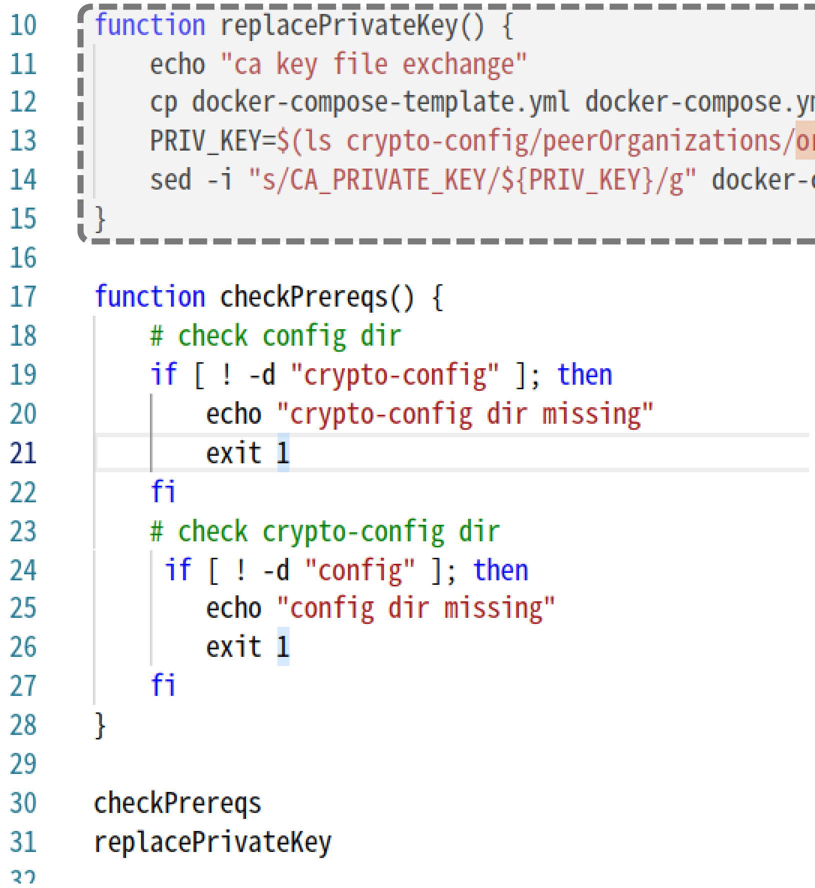
컨테이너 볼륨구조


peer0.org1
/etc/hyperledger/configtx/
mychannel.block을 사용하여
peer channel join
peer0.org2
/etc/hyperledger/configtx/
mychannel.block을 사용하여
peer channel join
cli
working_dir
/etc/hyperledger/configtx/
peer channel create의 결과
mychannel.block이 저장
Linux
./config
peer0.org3
/etc/hyperledger/configtx/
mychannel.block을 사용하여
peer channel join

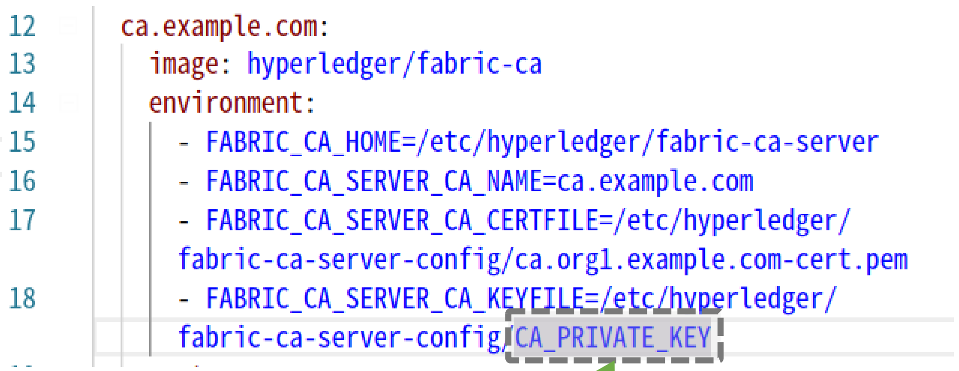
house keeping

- docker-compose 수행 전 확인사항

crypto-config디렉토리 내 org1의 ca key파일을 자동으로 docker-compose.yml로 복사 해주는 쉘 스크립트 코드



```
function replacePrivateKey() {
    echo "ca key file exchange"
    cp docker-compose-template.yml docker-compose.yml
    PRIV_KEY=$(ls crypto-config/peerOrganizations/org1.example.com/ca/ | grep _sk)
    sed -i "s/CA_PRIVATE_KEY/${PRIV_KEY}/g" docker-compose.yml
}

function checkPrereqs() {
    # check config dir
    if [ ! -d "crypto-config" ]; then
        echo "crypto-config dir missing"
        exit 1
    fi
    # check crypto-config dir
     if [ ! -d "config" ]; then
        echo "config dir missing"
        exit 1
    fi
}

checkPrereqs
replacePrivateKey
```

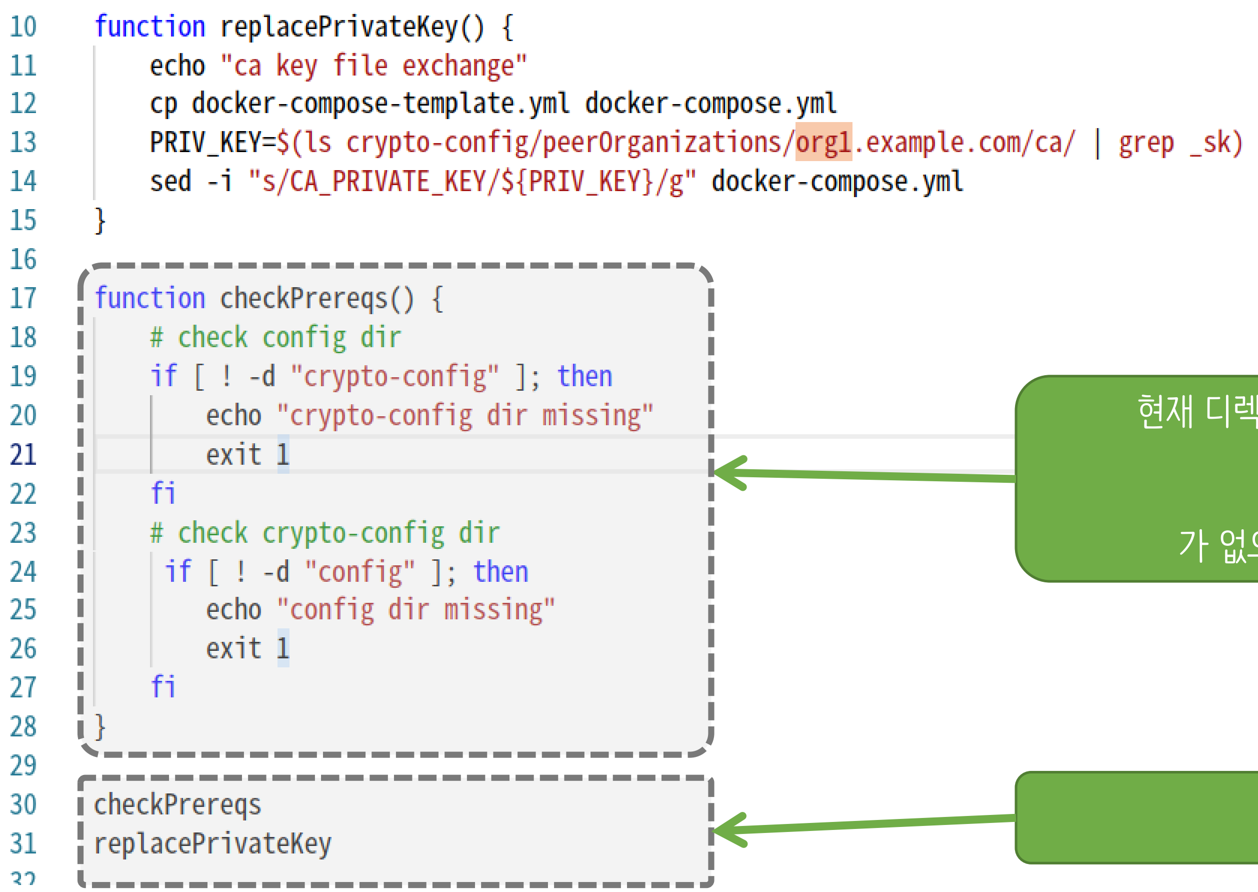


```
ca.example.com:
  image: hyperledger/fabric-ca
  environment:
    - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
    - FABRIC_CA_SERVER_CA_NAME=ca.example.com
    - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/
fabric-ca-server-config/ca.org1.example.com-cert.pem
    - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/
fabric-ca-server-config/CA_PRIVATE_KEY
```

docker-compose.yml파일을 복사하여 docker-compose-template파일로 저장하고 CA key자리에 CA_PRIVATE_KEY로 수정

house keeping

- docker-compose 수행 전 확인사항



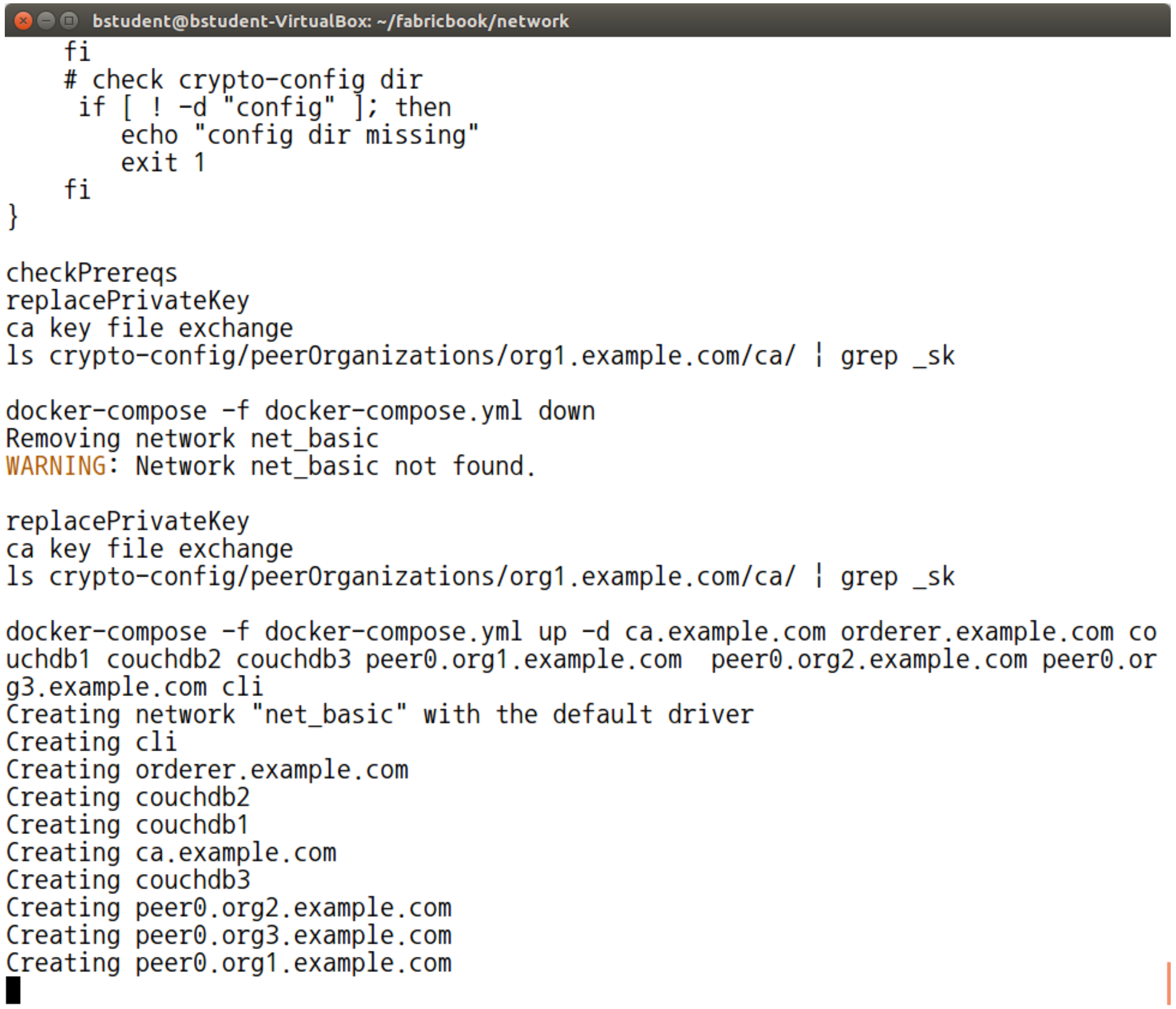
```
function replacePrivateKey() {
    echo "ca key file exchange"
    cp docker-compose-template.yml docker-compose.yml
    PRIV_KEY=$(ls crypto-config/peerOrganizations/org1.example.com/ca/ | grep _sk)
    sed -i "s/CA_PRIVATE_KEY/${PRIV_KEY}/g" docker-compose.yml
}

function checkPrereqs() {
    # check config dir
    if [ ! -d "crypto-config" ]; then
        echo "crypto-config dir missing"
        exit 1
    fi
    # check crypto-config dir
     if [ ! -d "config" ]; then
        echo "config dir missing"
        exit 1
    fi
}

checkPrereqs
replacePrivateKey
```

현재 디렉토리에 준비된 디렉토리
config
crypto-config
가 없으면 쉘프로그램 종료

함수 호출

네트워크 실행

- start.sh를 수행하여 수행된 네트워크 확인

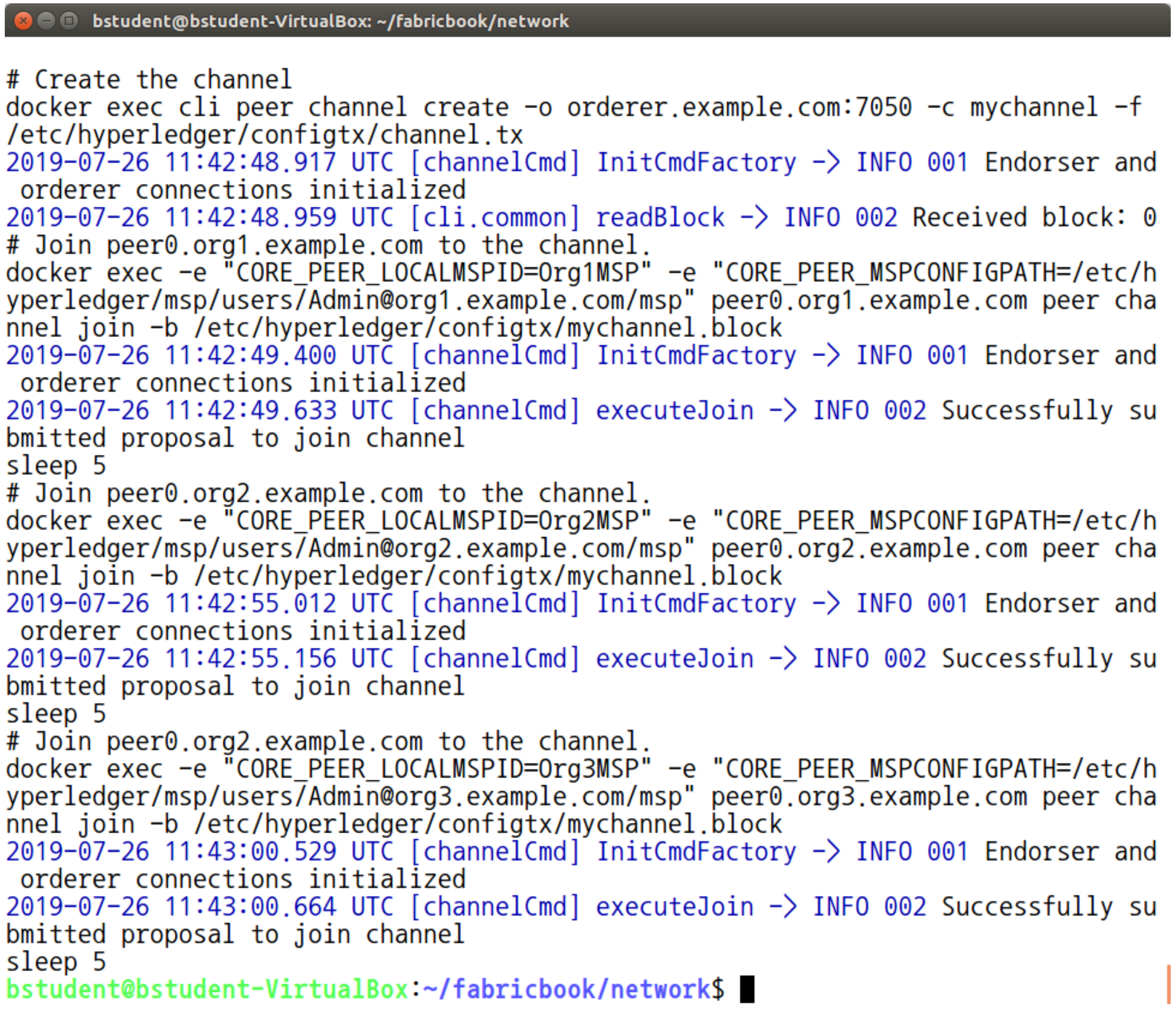


bstudent@bstudent-VirtualBox: ~/fabricbook/network

```
    fi
    # check crypto-config dir
     if [ ! -d "config" ]; then
        echo "config dir missing"
        exit 1
    fi
}

checkPrereqs
replacePrivateKey
ca key file exchange
ls crypto-config/peerOrganizations/org1.example.com/ca/ ¦ grep _sk

docker-compose -f docker-compose.yml down
Removing network net_basic
WARNING: Network net_basic not found.

replacePrivateKey
ca key file exchange
ls crypto-config/peerOrganizations/org1.example.com/ca/ ¦ grep _sk

docker-compose -f docker-compose.yml up -d ca.example.com orderer.example.com co
uchdb1 couchdb2 couchdb3 peer0.org1.example.com  peer0.org2.example.com peer0.or
g3.example.com cli
Creating network "net_basic" with the default driver
Creating cli
Creating orderer.example.com
Creating couchdb2
Creating couchdb1
Creating ca.example.com
Creating couchdb3
Creating peer0.org2.example.com
Creating peer0.org3.example.com
Creating peer0.org1.example.com
```

네트워크 실행

- start.sh를 수행하여 수행된 네트워크 확인

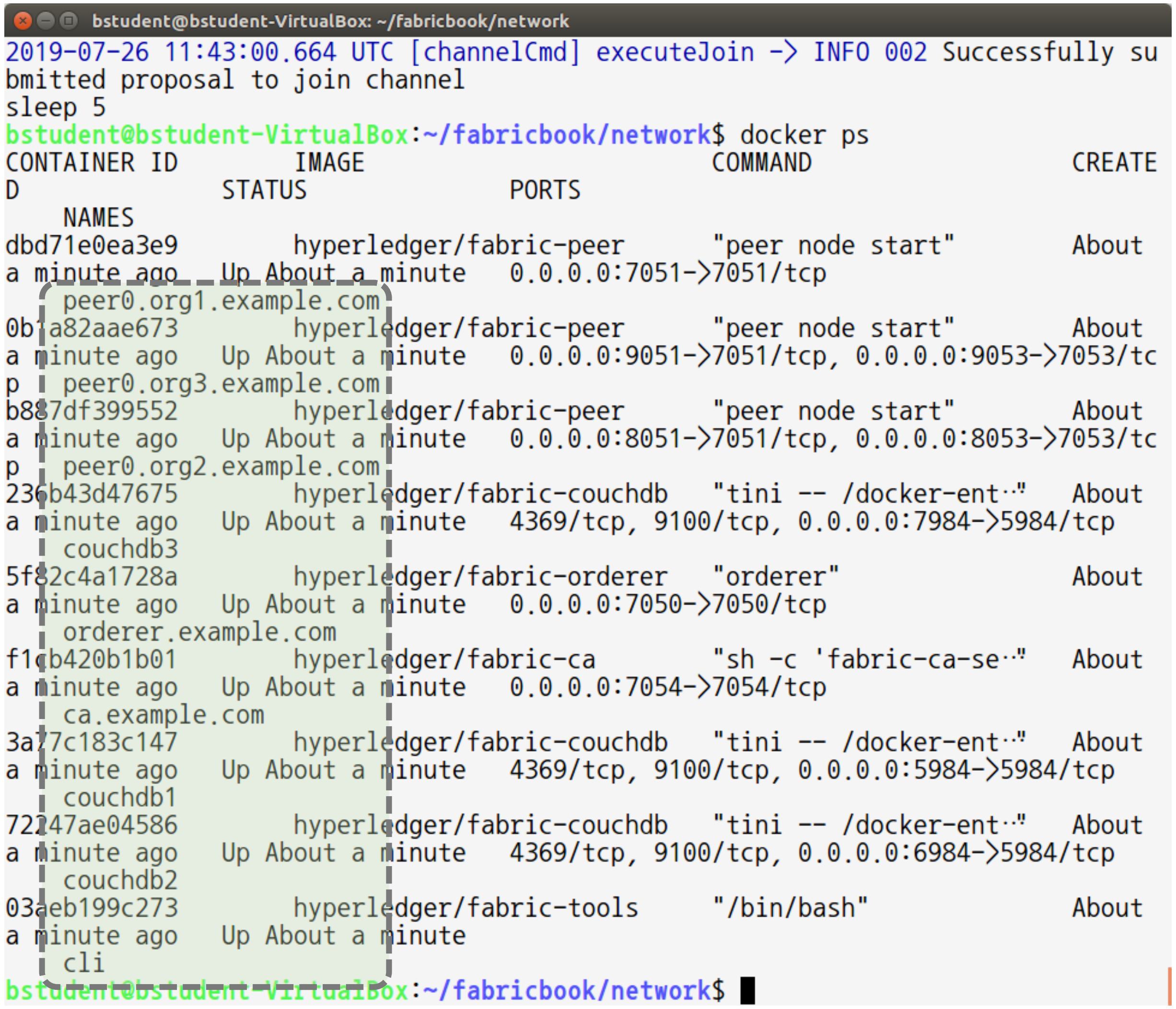


bstudent@bstudent-VirtualBox: ~/fabricbook/network

```
# Create the channel
docker exec cli peer channel create -o orderer.example.com:7050 -c mychannel -f
/etc/hyperledger/configtx/channel.tx
2019-07-26 11:42:48.917 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and
 orderer connections initialized
2019-07-26 11:42:48.959 UTC [cli.common] readBlock -> INFO 002 Received block: 0
# Join peer0.org1.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org1MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/h
yperledger/msp/users/Admin@org1.example.com/msp" peer0.org1.example.com peer cha
nnel join -b /etc/hyperledger/configtx/mychannel.block
2019-07-26 11:42:49.400 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and
 orderer connections initialized
2019-07-26 11:42:49.633 UTC [channelCmd] executeJoin -> INFO 002 Successfully su
bmitted proposal to join channel
sleep 5
# Join peer0.org2.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org2MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/h
yperledger/msp/users/Admin@org2.example.com/msp" peer0.org2.example.com peer cha
nnel join -b /etc/hyperledger/configtx/mychannel.block
2019-07-26 11:42:55.012 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and
 orderer connections initialized
2019-07-26 11:42:55.156 UTC [channelCmd] executeJoin -> INFO 002 Successfully su
bmitted proposal to join channel
sleep 5
# Join peer0.org2.example.com to the channel.
docker exec -e "CORE_PEER_LOCALMSPID=Org3MSP" -e "CORE_PEER_MSPCONFIGPATH=/etc/h
yperledger/msp/users/Admin@org3.example.com/msp" peer0.org3.example.com peer cha
nnel join -b /etc/hyperledger/configtx/mychannel.block
2019-07-26 11:43:00.529 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and
 orderer connections initialized
2019-07-26 11:43:00.664 UTC [channelCmd] executeJoin -> INFO 002 Successfully su
bmitted proposal to join channel
sleep 5
bstudent@bstudent-VirtualBox:~/fabricbook/network$ 
```

네트워크 실행

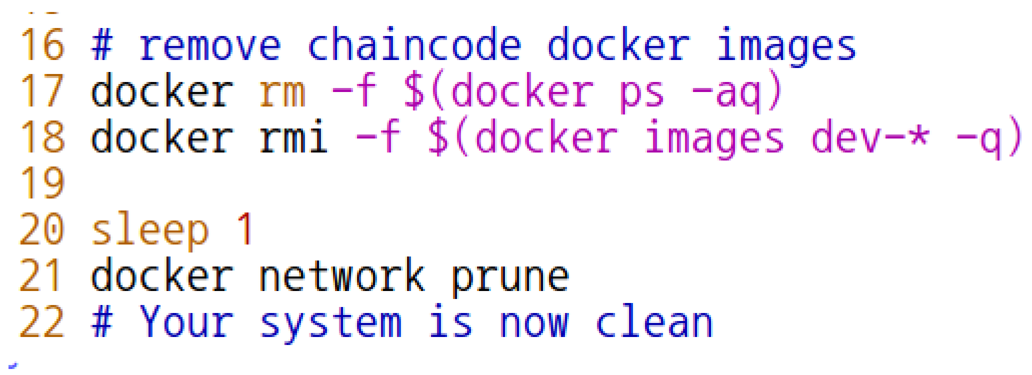
- start.sh를 수행하여 수행된 네트워크 확인



```
bstudent@bstudent-VirtualBox: ~/fabricbook/network
2019-07-26 11:43:00.664 UTC [channelCmd] executeJoin -> INFO 002 Successfully su
bmitted proposal to join channel
sleep 5
bstudent@bstudent-VirtualBox:~/fabricbook/network$ docker ps
CONTAINER ID        IMAGE                            COMMAND                  CREATE
D                   STATUS                   PORTS
    NAMES
dbd71e0ea3e9        hyperledger/fabric-peer          "peer node start"        About
a minute ago        Up About a minute        0.0.0.0:7051->7051/tcp
    peer0.org1.example.com
0b7a82aae673        hyperledger/fabric-peer          "peer node start"        About
a minute ago        Up About a minute        0.0.0.0:9051->7051/tcp, 0.0.0.0:9053->7053/tc
p   peer0.org3.example.com
b887df399552        hyperledger/fabric-peer          "peer node start"        About
a minute ago        Up About a minute        0.0.0.0:8051->7051/tcp, 0.0.0.0:8053->7053/tc
p   peer0.org2.example.com
236b43d47675        hyperledger/fabric-couchdb       "tini -- /docker-ent…"   About
a minute ago        Up About a minute        4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp
    couchdb3
5f82c4a1728a        hyperledger/fabric-orderer       "orderer"                About
a minute ago        Up About a minute        0.0.0.0:7050->7050/tcp
    orderer.example.com
f1cb420b1b01        hyperledger/fabric-ca            "sh -c 'fabric-ca-se…"   About
a minute ago        Up About a minute        0.0.0.0:7054->7054/tcp
    ca.example.com
3a77c183c147        hyperledger/fabric-couchdb       "tini -- /docker-ent…"   About
a minute ago        Up About a minute        4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp
    couchdb1
72247ae04586        hyperledger/fabric-couchdb       "tini -- /docker-ent…"   About
a minute ago        Up About a minute        4369/tcp, 9100/tcp, 0.0.0.0:6984->5984/tcp
    couchdb2
03aeb199c273        hyperledger/fabric-tools         "/bin/bash"              About
a minute ago        Up About a minute
    cli
bstudent@bstudent-VirtualBox:~/fabricbook/network$
```

네트워크구성 확장

- teardown.sh
  - 실행 시 Docker 컨테이너 삭제



```
# remove chaincode docker images
docker rm -f $(docker ps -aq)
docker rmi -f $(docker images dev-* -q)

sleep 1
docker network prune
# Your system is now clean
```

체인코드 구현 설치 배포

- 체인코드 복사

```
$cp ~/fabric-samples/chaincode/sacc/ ~/fabricbook/contract
```

- 체인코드 설치

```
$docker exec -it cli bash
#peer chaincode install -n sacc -v 1.0 -p github.com/sacc
```

- 체인코드 배포

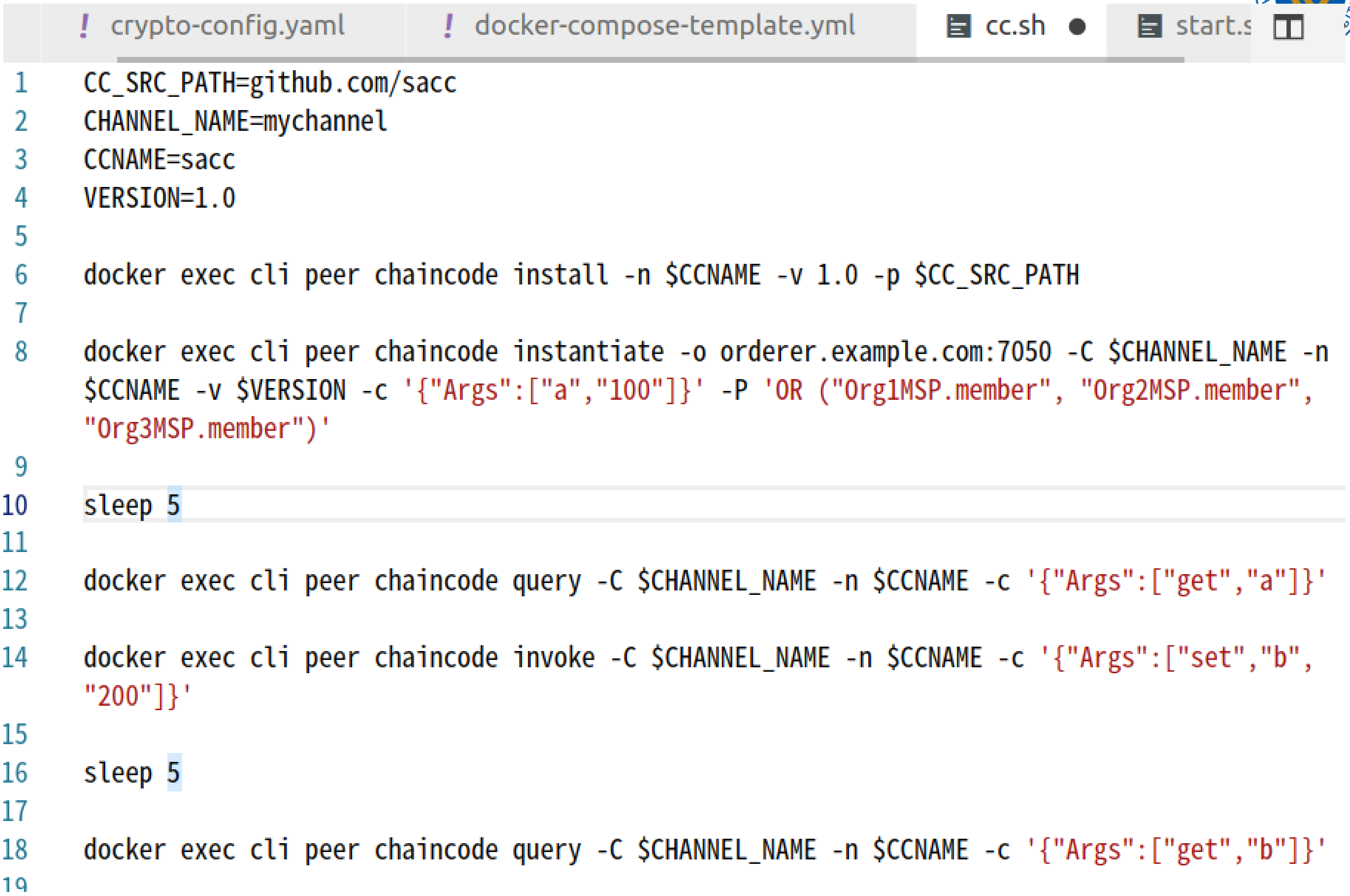
```
#peer chaincode instantiate -n sacc -v 1.0 -C mychannel -c
'{"Args":["a","100"]}' -P 'OR ("Org1MSP.member",
"Org2MSP.member","Org3MSP.member")'
```

- 체인코드 작동확인

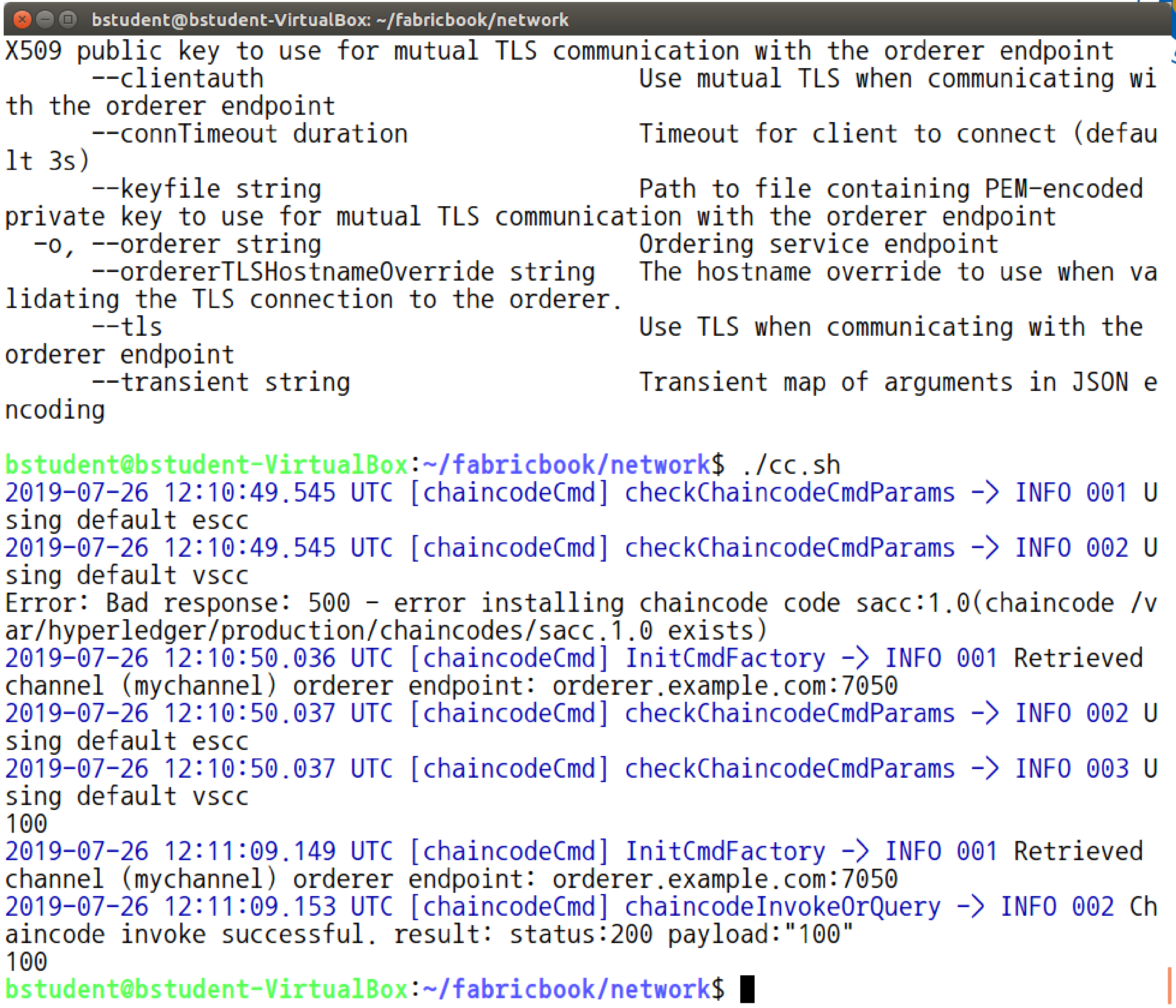
```
#peer chaincode query -n sacc -C mychannel -c '{"Args":["get","a"]}'
#peer chaincode invoke -n sacc -C mychannel -c
'{"Args":["set","b","100"]}'
#peer chaincode query -n sacc -C mychannel -c '{"Args":["get","b"]}'
```

체인코드 쉘스크립트 작성

- cc.sh 작성
  - 체인코드 설치
  - 체인코드 배포
  - 체인코드 작동확인 코드가 포함



```
CC_SRC_PATH=github.com/sacc
CHANNEL_NAME=mychannel
CCNAME=sacc
VERSION=1.0

docker exec cli peer chaincode install -n $CCNAME -v 1.0 -p $CC_SRC_PATH

docker exec cli peer chaincode instantiate -o orderer.example.com:7050 -C $CHANNEL_NAME -n $CCNAME -v $VERSION -c '{"Args":["a","100"]}' -P 'OR ("Org1MSP.member", "Org2MSP.member", "Org3MSP.member")'

sleep 5

docker exec cli peer chaincode query -C $CHANNEL_NAME -n $CCNAME -c '{"Args":["get","a"]}'

docker exec cli peer chaincode invoke -C $CHANNEL_NAME -n $CCNAME -c '{"Args":["set","b", "200"]}'

sleep 5

docker exec cli peer chaincode query -C $CHANNEL_NAME -n $CCNAME -c '{"Args":["get","b"]}'
```

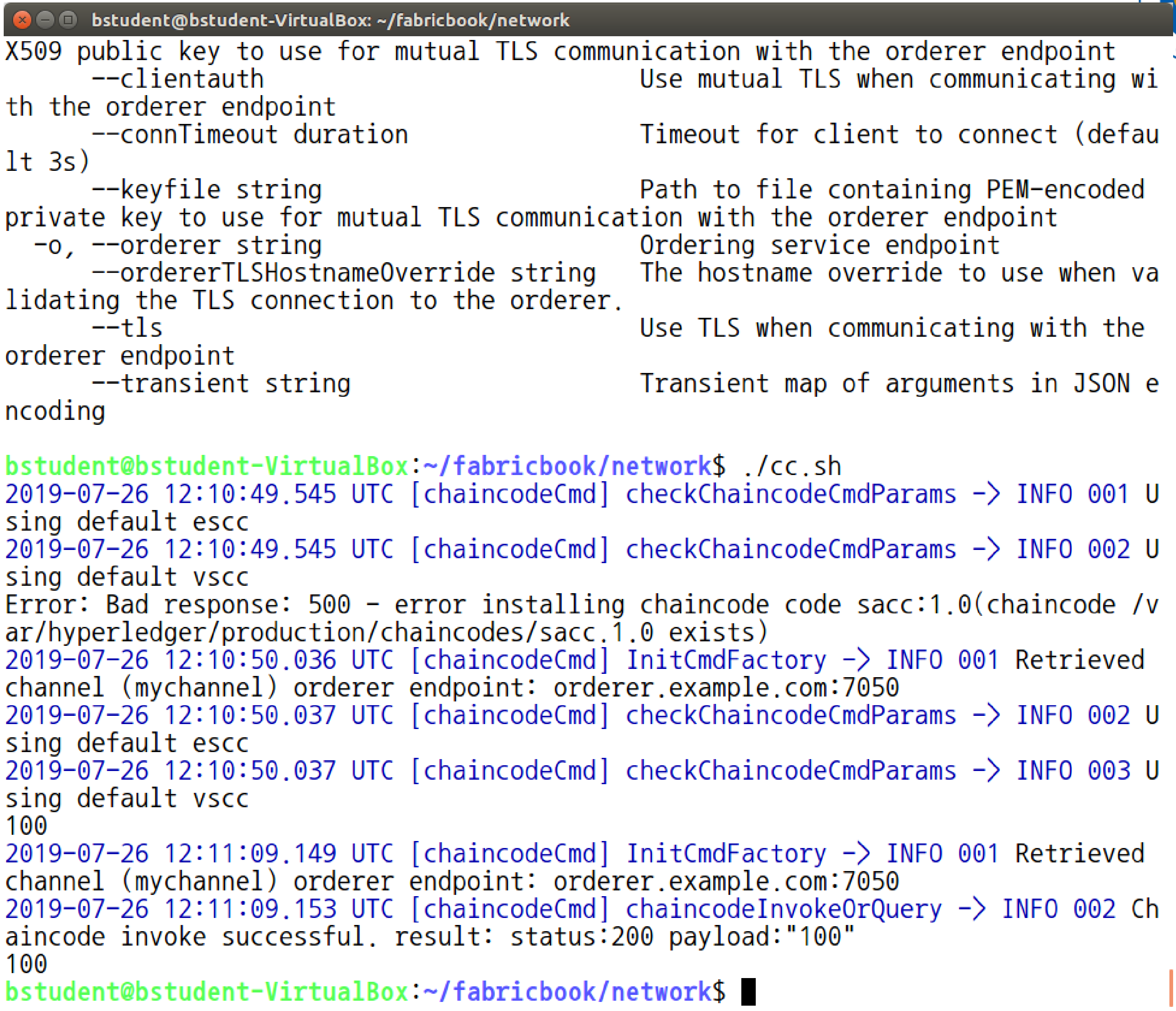


```
bstudent@bstudent-VirtualBox: ~/fabricbook/network
X509 public key to use for mutual TLS communication with the orderer endpoint
      --clientauth                              Use mutual TLS when communicating wi
th the orderer endpoint
      --connTimeout duration                    Timeout for client to connect (defau
lt 3s)
      --keyfile string                          Path to file containing PEM-encoded
private key to use for mutual TLS communication with the orderer endpoint
  -o, --orderer string                          Ordering service endpoint
      --ordererTLSHostnameOverride string   The hostname override to use when va
lidating the TLS connection to the orderer.
      --tls                                     Use TLS when communicating with the
orderer endpoint
      --transient string                        Transient map of arguments in JSON e
ncoding

bstudent@bstudent-VirtualBox:~/fabricbook/network$ ./cc.sh
2019-07-26 12:10:49.545 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-07-26 12:10:49.545 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
Error: Bad response: 500 - error installing chaincode code sacc:1.0(chaincode /v
ar/hyperledger/production/chaincodes/sacc.1.0 exists)
2019-07-26 12:10:50.036 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved
channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-07-26 12:10:50.037 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default escc
2019-07-26 12:10:50.037 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 U
sing default vscc
100
2019-07-26 12:11:09.149 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved
channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-07-26 12:11:09.153 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 002 Ch
aincode invoke successful. result: status:200 payload:"100"
100
bstudent@bstudent-VirtualBox:~/fabricbook/network$
```

어플리케이션 작동확인

- fabcar/javascript 디렉토리에서
  - enrollAdmin.js, registerUser.js, invoke.js, query.js package.json를 fabricbook/application으로 복사
- js파일 내 connection.json 연결부분 수정
- 인증서 생성
- query, invoke 체인코드 명 및 함수, 인자들 수정

- 기능 수행



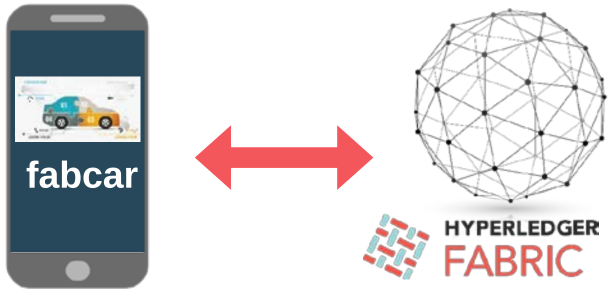
```
X509 public key to use for mutual TLS communication with the orderer endpoint
      --clientauth                              Use mutual TLS when communicating wi
th the orderer endpoint
      --connTimeout duration                    Timeout for client to connect (defau
lt 3s)
      --keyfile string                          Path to file containing PEM-encoded
private key to use for mutual TLS communication with the orderer endpoint
  -o, --orderer string                          Ordering service endpoint
      --ordererTLSHostnameOverride string       The hostname override to use when va
lidating the TLS connection to the orderer.
      --tls                                     Use TLS when communicating with the
orderer endpoint
      --transient string                        Transient map of arguments in JSON e
ncoding

bstudent@bstudent-VirtualBox:~/fabricbook/network$ ./cc.sh
2019-07-26 12:10:49.545 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 U
sing default escc
2019-07-26 12:10:49.545 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default vscc
Error: Bad response: 500 - error installing chaincode code sacc:1.0(chaincode /v
ar/hyperledger/production/chaincodes/sacc.1.0 exists)
2019-07-26 12:10:50.036 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved
channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-07-26 12:10:50.037 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 U
sing default escc
2019-07-26 12:10:50.037 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 003 U
sing default vscc
100
2019-07-26 12:11:09.149 UTC [chaincodeCmd] InitCmdFactory -> INFO 001 Retrieved
channel (mychannel) orderer endpoint: orderer.example.com:7050
2019-07-26 12:11:09.153 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 002 Ch
aincode invoke successful. result: status:200 payload:"100"
100
bstudent@bstudent-VirtualBox:~/fabricbook/network$
```

etc.

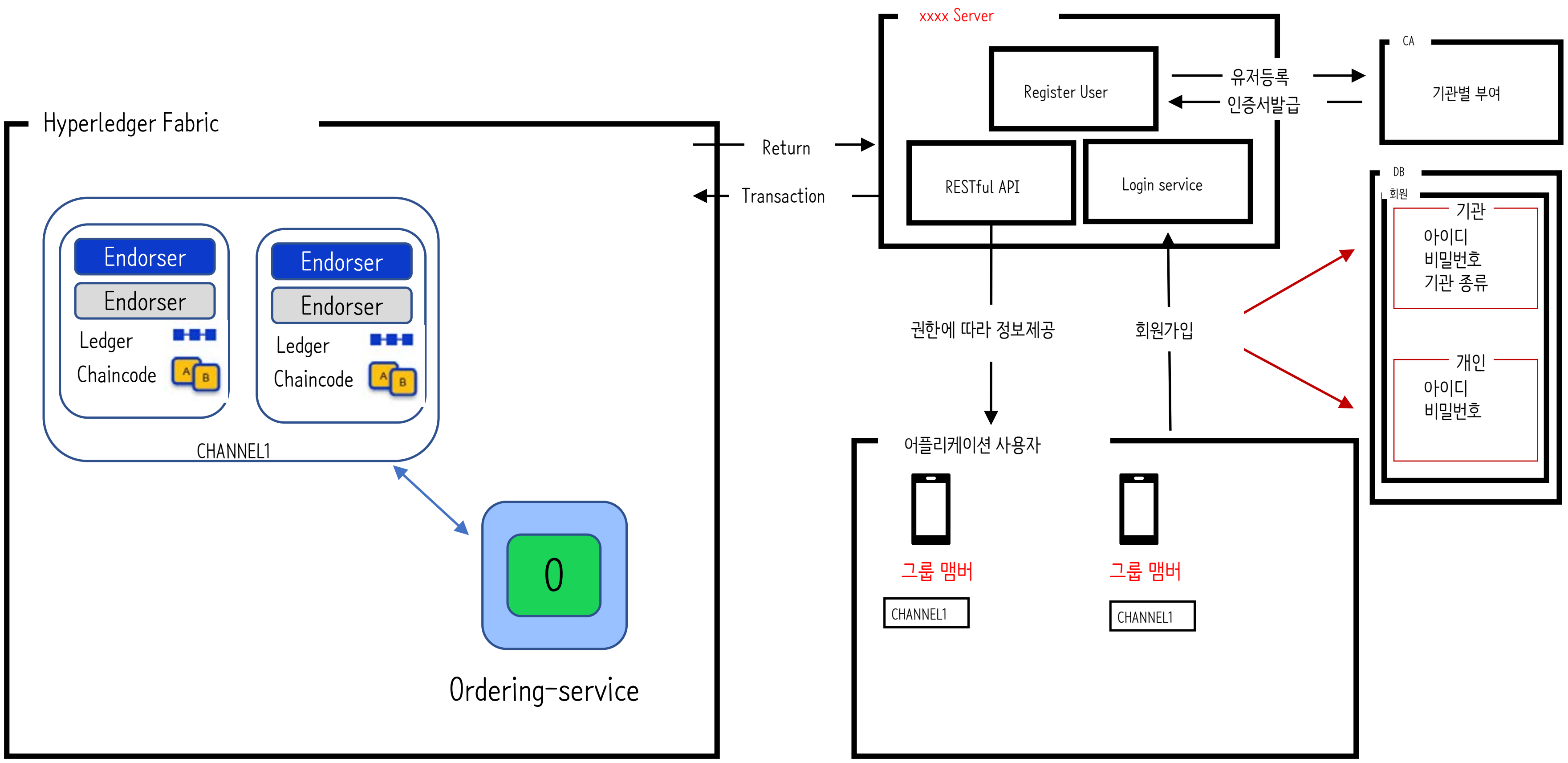
- Private Data 사용하기
  - PD Collection 정의 및 체인코드 SDK 를 사용하여 체인코드 구현
  - PD Collection 정보를 포함한 체인코드 배포
  - TRANSIENT DB를 활용한 ARGUMENT전달
  - ORG별 Private 데이터 접근권한 확인
  - Private 데이터 자동삭제 확인
- TLS옵션 사용하기
  - docker-compose에서 TLS 옵션을 포함하여 정의하기
  - 오더러를 사용하는 모든 명령에 TLS옵션 추가하기
    - 채널생성및 조인
    - 체인코드 배포 및 invoke
  - Application에서 TLS옵션으로 포함하여 게이트웨이에 접근하기

자동차정보앱 블록체인 (예시)

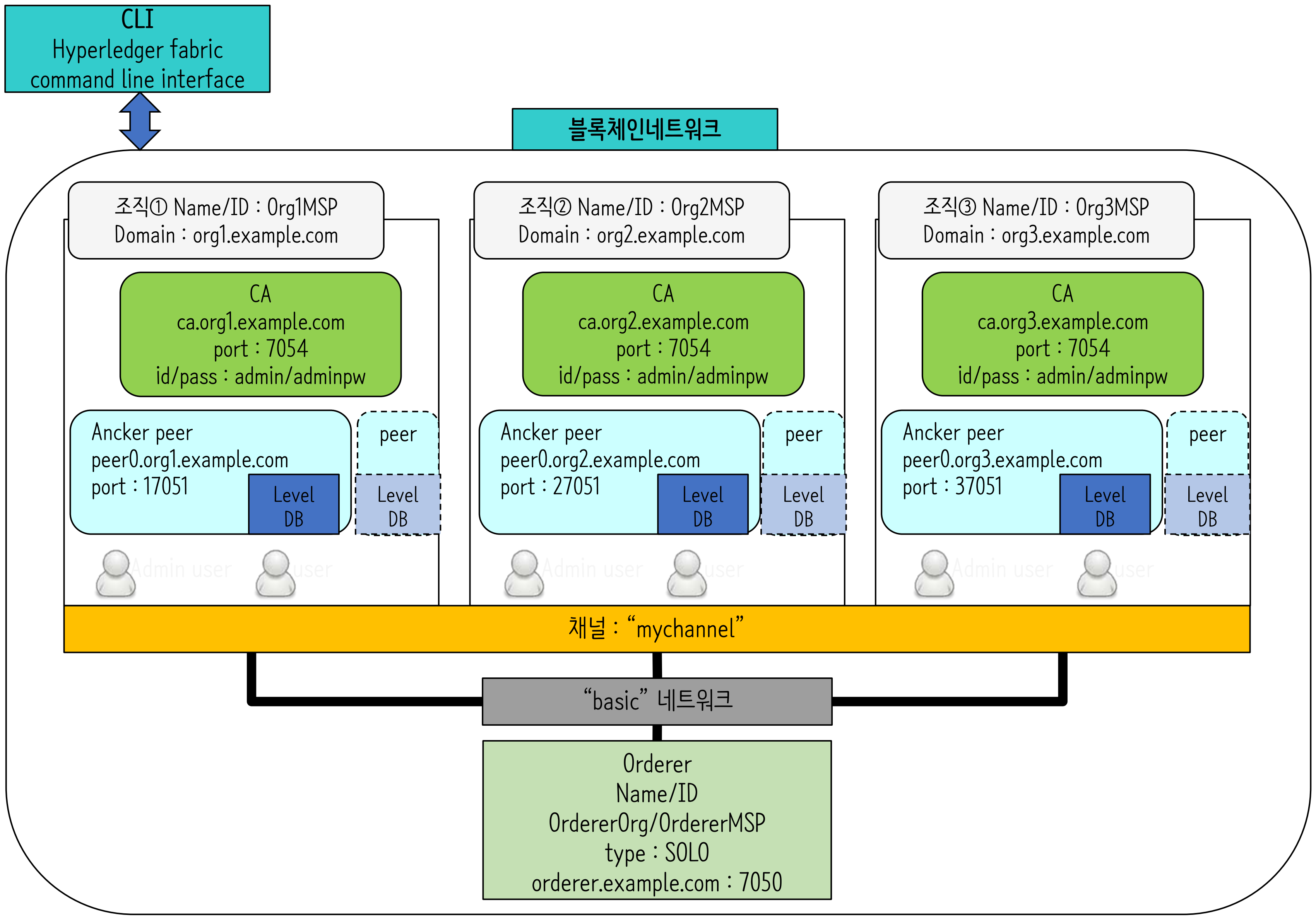
- 자동차정보앱 개요
  - 자동차모델에 대한 정보를 저장하는 블록체인을 구축하고
  - 생산자와 소비자사이를 이어줄 수 있는 편리한 Dapp을 구성
- 자동차정보앱 블록체인의 장점
  - 플랫폼과 데이터의 품질 향상 (투명성, 신뢰성, 가용성, 안정성)
  - 다른 기술과 융합(확장) 수월 (IoT, 인공지능, 빅데이터 등)
- 자동차정보앱 블록체인의 목적
  - 분산화 (Decentralization)
  - 보안성 강화 (Security)
  - 성능 향상과 투명성 (Performance and Transparency)
- 자동차정보앱 블록체인 개발을 통한 의미
  1. web3 기술 획득 데모 (Demonstration of procurement of web3 technology)
  2. 광범위한 블록체인과 데이터베이스로 확장 가능



전체 구조도작성 및 설계

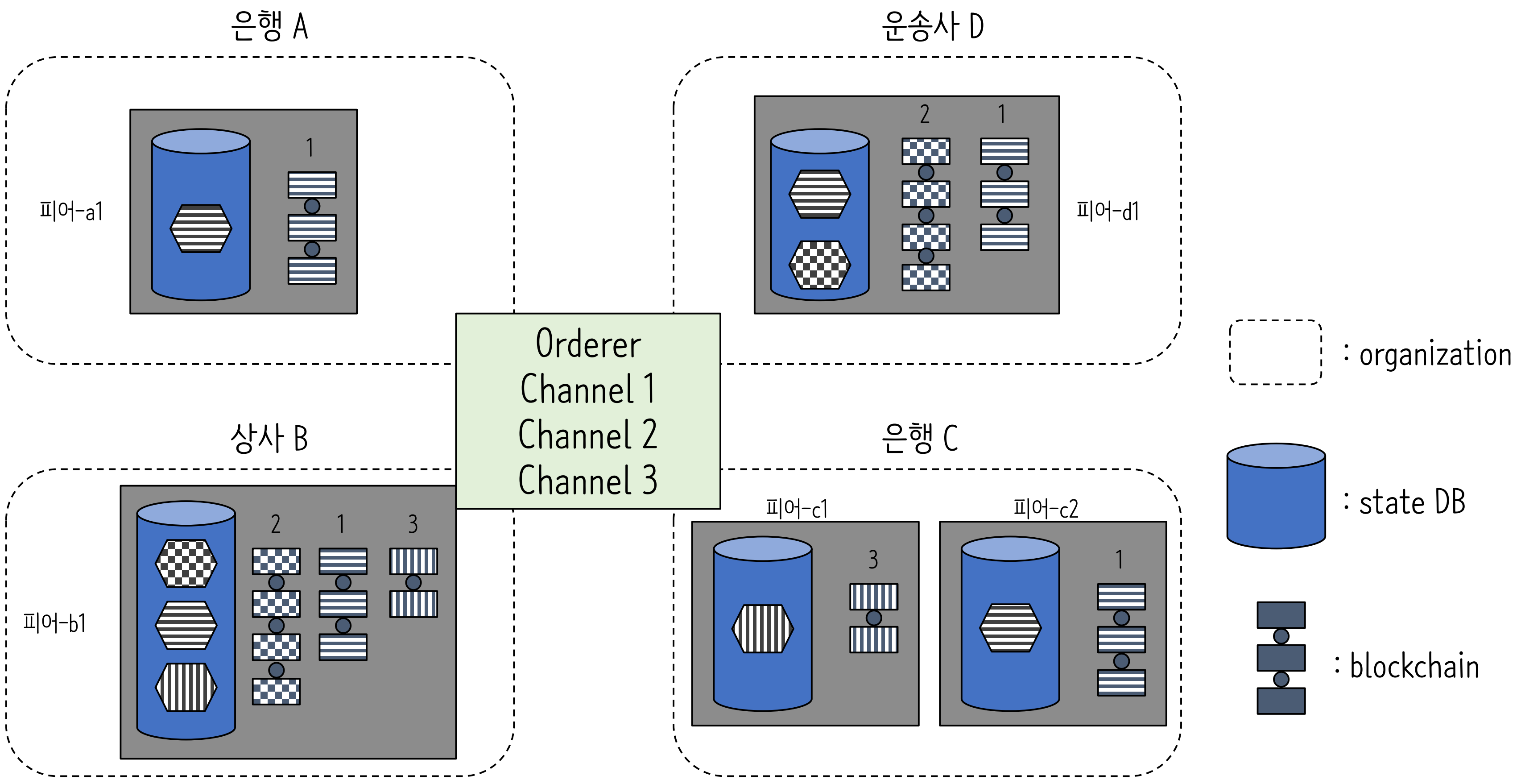

Hyperledger Fabric
Endorser
Endorser
Ledger
Chaincode
Endorser
Endorser
Ledger
Chaincode
CHANNEL1
0
Ordering-service
Return
Transaction
xxxx Server
Register User
RESTful API
Login service
유저등록
인증서발급
CA
기관별 부여
DB
회원
기관
아이디
비밀번호
기관 종류
개인
아이디
비밀번호
권한에 따라 정보제공
회원가입
어플리케이션 사용자
그룹 맴버
CHANNEL1
그룹 맴버
CHANNEL1

네트워크 설계



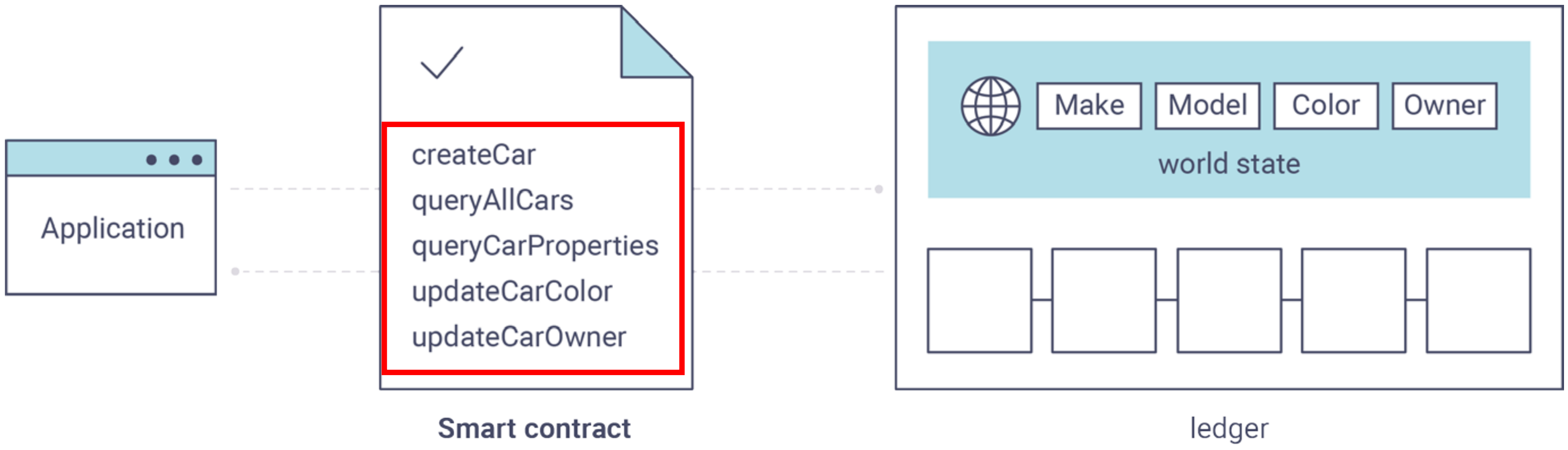
CLI
Hyperledger fabric
command line interface
블록체인네트워크
조직① Name/ID : Org1MSP
Domain : org1.example.com
CA
ca.org1.example.com
port : 7054
id/pass : admin/adminpw
Ancker peer
peer0.org1.example.com
port : 17051
Level DB
peer
Level DB
Admin user
user
조직② Name/ID : Org2MSP
Domain : org2.example.com
CA
ca.org2.example.com
port : 7054
id/pass : admin/adminpw
Ancker peer
peer0.org2.example.com
port : 27051
Level DB
peer
Level DB
Admin user
user
조직③ Name/ID : Org3MSP
Domain : org3.example.com
CA
ca.org3.example.com
port : 7054
id/pass : admin/adminpw
Ancker peer
peer0.org3.example.com
port : 37051
Level DB
peer
Level DB
Admin user
user
채널 : "mychannel"
"basic" 네트워크
Orderer
Name/ID
OrdererOrg/OrdererMSP
type : SOLO
orderer.example.com : 7050

네트워크 구성



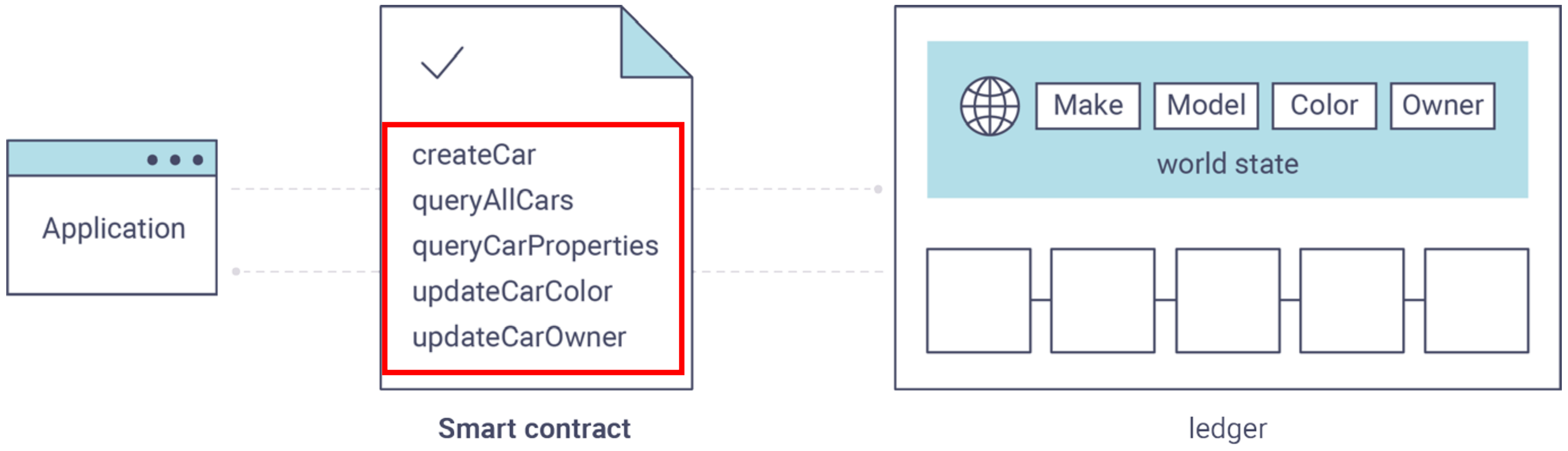
은행 A
피어-a1
1
운송사 D
2
1
피어-d1
Orderer
Channel 1
Channel 2
Channel 3
상사 B
피어-b1
2
1
3
은행 C
피어-c1
3
피어-c2
1
: organization
: state DB
: blockchain

데이터, 기능리스트


Application
createCar
queryAllCars
queryCarProperties
updateCarColor
updateCarOwner
Smart contract
Make
Model
Color
Owner
world state
ledger

dApp 프로토타입 기획

- Dapp의 기능리스트
- Dapp에 저장할 world state
- 체인코드 이름, 함수프로토타입정의



기능에 따른 함수 목록 작성

World State에 저장될 데이터 목록작성

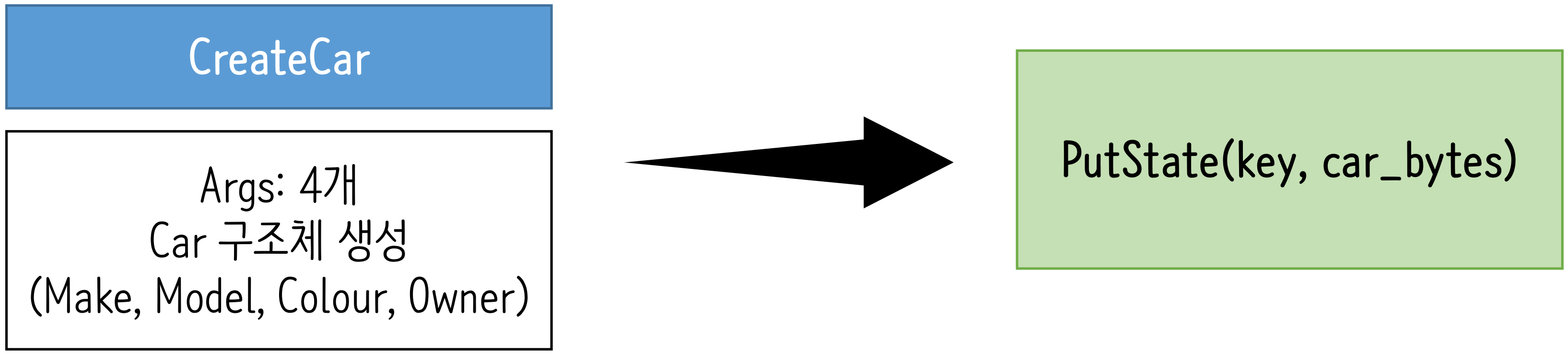
Private data가 필요할 시
Collection목록과 형식 작성

Dapp의 기능리스트

- Fabcar 의 기능
  - 자동차 정보를 추가
  - 자동차 정보를 업데이트
    - 소유주 바꾸기
  - 자동차 정보를 보여주기
    - Key를 이용하여 보여주기
    - 모두 다 보여주기
  - 가격정보를 보여주기
    - 권한이 있는 Peer만 해당 정보를 접근가능 하도록 구성

체인코드 이름, 함수프로토타입정의

- 작성할 체인코드의 이름, 작성할 언어 선택
- 기능리스트에 따르는 함수목록 작성
- Init함수와 Invoke함수의 프로토타입과 기능 정리
- 각 기능 함수의 프로토타입을 정의 하고 함수내에서 해야 할 일을 간략히 정리
- 각 함수와 블록체인 데이터와 연관관계를 정리
- 예)



클라이언트 접속 URI설계

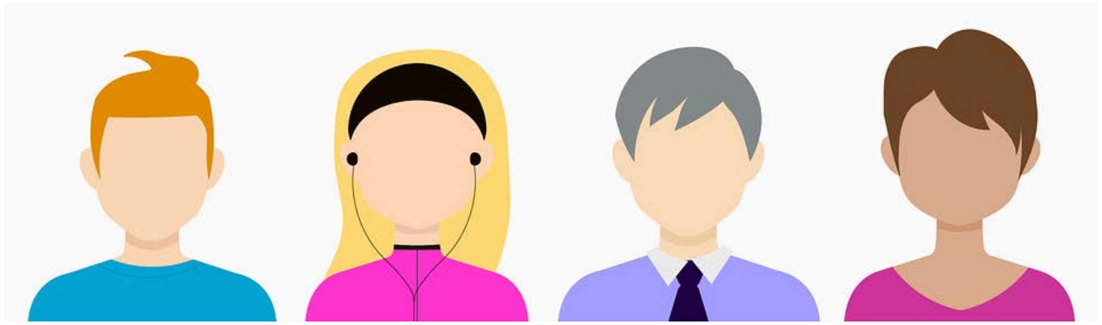
차정보	
make	string
model	string
color	string
owner	string

URI : /cars		
차정보 등록	post	param make model color owner
차정보 조회	get	carno
차정보 삭제	delete	carno
차정보 수정	put	carno owner

클라이언트 접속 URI설계

회원정보	
name	string

URI : /customer		
회원정보 등록	post	name
회원정보 조회	get	cusno
회원정보 삭제	delete	cusno
회원정보 수정	put	cusno name

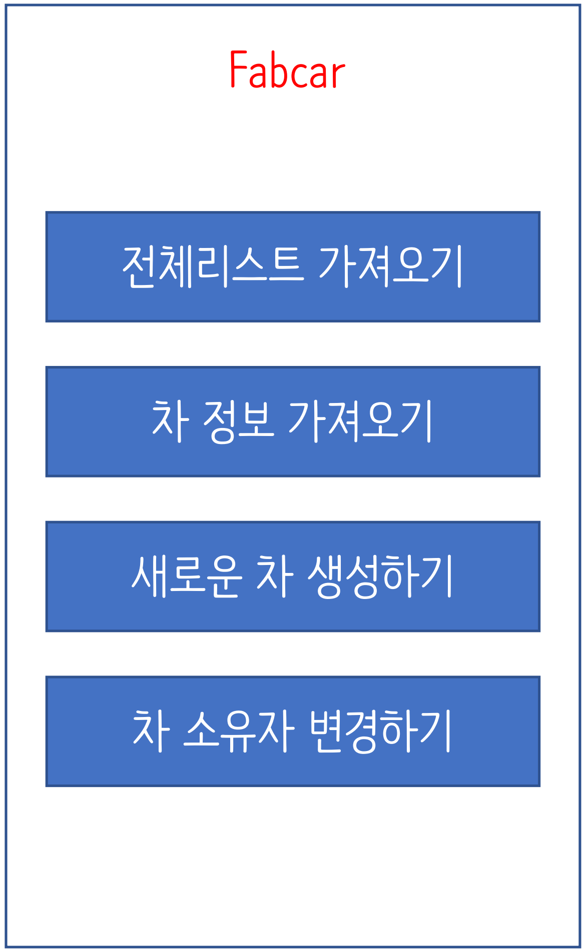


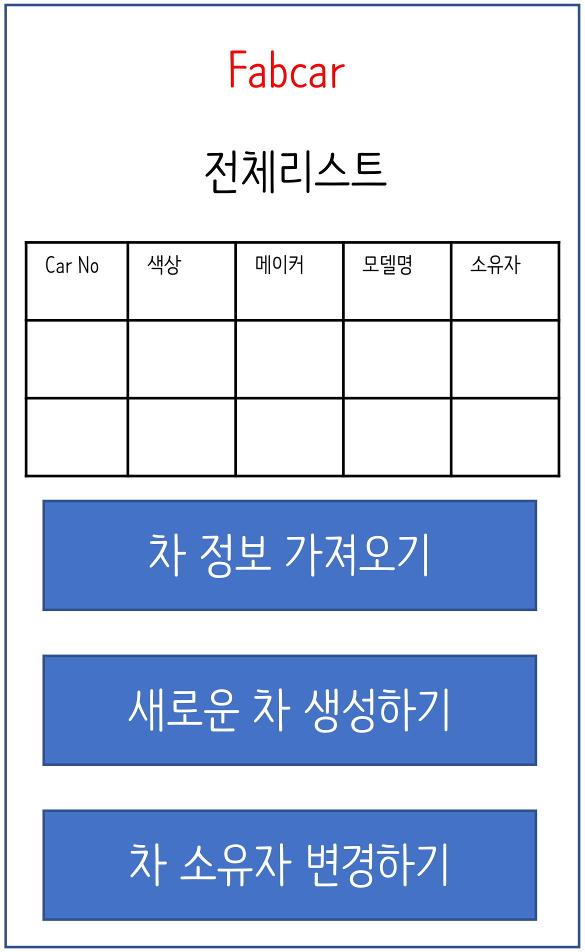
클라이언트 접속 URI설계

거래정보	
차량 key	string
회원 key	string
가격	number

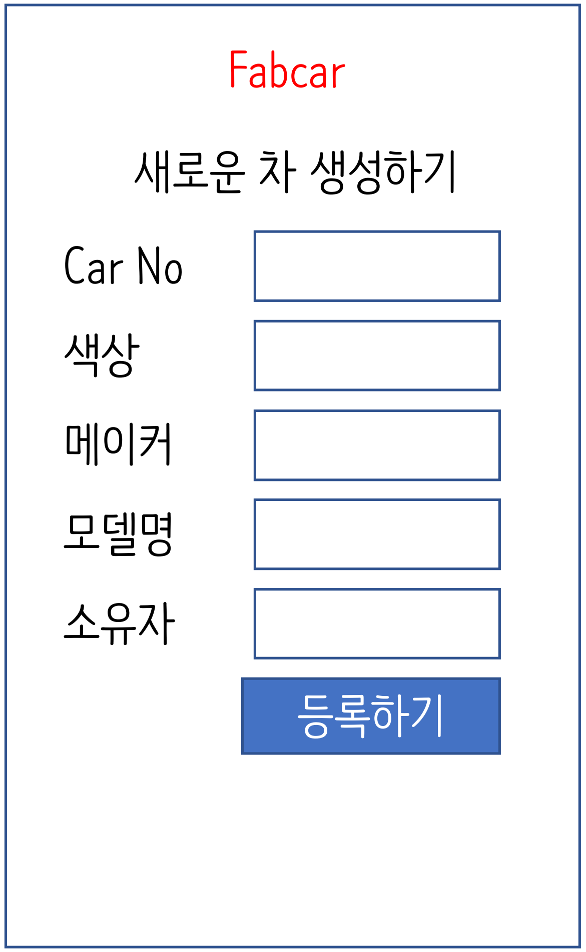
URI : /buy		
구매정보 등록	post	carno cusno price
구매정보 조회	get	buyno
구매정보 삭제	delete	buyno
구매정보 수정	put	buyno price

웹인터페이스





Car No	색상	메이커	모델명	소유자



기능 구현

개발 계획서 작성

필요 디렉토리 구성

네트워크 구현

체인코드 구현

프론트앤드 구현

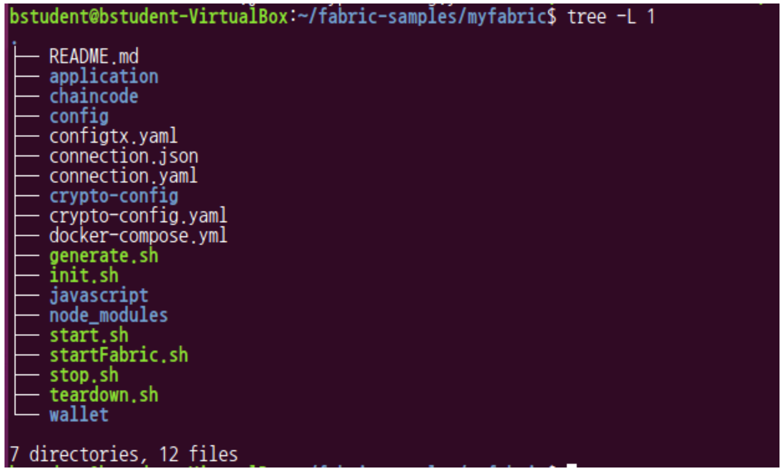
단위별 테스트 및 연동 테스트

구현 개발서 및 주요테스트 결과보고서

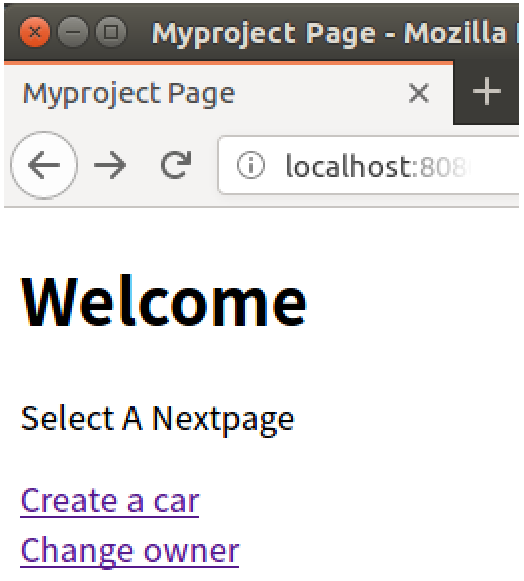
사용메뉴얼

데모준비

- 쉘스크립트를 수행하여 네트워크가동
- 필요시 체인코드 Invoke, Query수행
- 웹서버가동
- 웹서버 접속 후 기능사용



```
bstudent@bstudent-VirtualBox:~/fabric-samples/myfabric$ tree -L 1
.
├── README.md
├── application
├── chaincode
├── config
├── configtx.yaml
├── connection.json
├── connection.yaml
├── crypto-config
├── crypto-config.yaml
├── docker-compose.yml
├── generate.sh
├── init.sh
├── javascript
├── node_modules
├── start.sh
├── startFabric.sh
├── stop.sh
├── teardown.sh
└── wallet

7 directories, 12 files
```



Welcome

Select A Nextpage

Create a car
Change owner