< Raft 알고리즘 fabric 실습 >

참고 사이트 - ① : https://medium.com/coinmonks/hyperledger-fabric-the-taste-of-raft-4f9f0df20b5e
- ② : https://suckzoo.github.io/tech/2018/01/03/raft-1.html
- ③ : http://thesecretlivesofdata.com/raft/

1. 개요 : 'Raft'는 fabric 1.4.1부터 새로 도입된 합의 알고리즘 중 하나이다.
이 알고리즘을 통해 하나의 orderer가 꺼지더라도, 자동리더선출을 통해 블록체인 네트워크가 문제없이 돌아가는 것을 볼 수 있다.
- 참고 사이트 : https://raft.github.io/ 및 http://thesecretlivesofdata.com/raft/

2. 실습 : first-network를 통해 진행
: 2 Org / 5 orderers / 2 peer in each Org with CouchDB / cli / channel / chaincode

```
alpha@fabric2:~/fabric-samples/first-network$ ls
base                crypto-config.yaml              docker-compose-e2e-template.yaml  eyfn.sh
byfn.sh             docker-compose-cli.yaml         docker-compose-etcdraft2.yaml     org3-artifacts
channel-artifacts   docker-compose-couch-org3.yaml  docker-compose-kafka.yaml         README.md
configtx.yaml       docker-compose-couch.yaml       docker-compose-org3.yaml          scripts
```

① first-network에서 MSP 등 기본 org 생성	$ cd fabric-samples/first-network
	$ ../bin/cryptogen generate --config=./crypto-config.yaml
	`alpha@fabric2:~/fabric-samples/first-network$ ../bin/cryptogen generate --config=./crypto-config.yaml`
	`org1.example.com`
	`org2.example.com`
	* 참고 : fabric-samples/bin 아래 구조(8개의 실행파일 존재 / fabric2.0에서는 token 추가)
	`alpha@fabric2:~/fabric-samples/first-network$ tree ../bin`
	`../bin`
	`├── configtxgen`
	`├── configtxlator`
	`├── cryptogen`
	`├── discover`
	`├── fabric-ca-client`
	`├── idemixgen`
	`├── orderer`
	`├── peer`
	`└── token`

② EtcdRaft 알고리즘을 통한 genesis block 생성 및 채널 생성	$ export FABRIC_CFG_PATH=$PWD
	$ ../bin/configtxgen -profile SampleMultiNodeEtcdRaft -channelID byfn-sys-channel -outputBlock ./channel-artifacts/genesis.block
	`alpha@fabric2:~/fabric-samples/first-network$ ../bin/configtxgen -profile SampleMultiNodeEtcdRaft -channelID byfn-sys-channel -outputBlock ./channel-artifacts/genesis.block`
	`2019-05-30 16:36:08.676 KST [common.tools.configtxgen] main -> INFO 001 Loading configuration`
	`2019-05-30 16:36:08.910 KST [common.tools.configtxgen.localconfig] completeInitialization -> INFO 002 orderer type: etcdraft`
	`2019-05-30 16:36:08.910 KST [common.tools.configtxgen.localconfig] completeInitialization -> INFO 003 Orderer.EtcdRaft.Options unset, setting to tick_interval:"500ms" election_tick:10 heartbeat_tick:1 max_inflight_blocks:5 snapshot_interval_size:20971520`
	`2019-05-30 16:36:08.910 KST [common.tools.configtxgen.localconfig] Load -> INFO 004 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml`
	`2019-05-30 16:36:09.090 KST [common.tools.configtxgen.localconfig] completeInitialization -> INFO 005 orderer type: solo`
	`2019-05-30 16:36:09.091 KST [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 006 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml`
	`2019-05-30 16:36:09.095 KST [common.tools.configtxgen] doOutputBlock -> INFO 007 Generating genesis block`
	`2019-05-30 16:36:09.096 KST [common.tools.configtxgen] doOutputBlock -> INFO 008 Writing genesis block`
	$ export CHANNEL_NAME=mychannel → 환경변수 설정
	$ ../bin/configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/channel.tx -channelID $CHANNEL_NAME
	`alpha@fabric2:~/fabric-samples/first-network$ ../bin/configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-artifacts/channel.tx -channelID $CHANNEL_NAME`
	`2019-05-30 16:38:18.635 KST [common.tools.configtxgen] main -> INFO 001 Loading configuration`
	`2019-05-30 16:38:18.811 KST [common.tools.configtxgen.localconfig] Load -> INFO 002 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml`
	`2019-05-30 16:38:19.030 KST [common.tools.configtxgen.localconfig] completeInitialization -> INFO 003 orderer type: solo`
	`2019-05-30 16:38:19.030 KST [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 004 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml`
	`2019-05-30 16:38:19.031 KST [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 005 Generating new channel configtx`
	`2019-05-30 16:38:19.036 KST [common.tools.configtxgen] doOutputChannelCreateTx -> INFO 006 Writing new channel tx`

③ Org별 앵커피어 생성	$ ../bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/Org1MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org1MSP $ ../bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/Org2MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org2MSP alpha@fabric2:~/fabric-samples/first-network$ ../bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-artifacts/Org2MSPanchors.tx -channelID $CHANNEL_NAME -asOrg Org2MSP 2019-05-30 16:41:53.606 KST [common.tools.configtxgen] main -> INFO 001 Loading configuration 2019-05-30 16:41:53.797 KST [common.tools.configtxgen.localconfig] Load -> INFO 002 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml 2019-05-30 16:41:53.972 KST [common.tools.configtxgen.localconfig] completeInitialization -> INFO 003 orderer type: solo 2019-05-30 16:41:53.973 KST [common.tools.configtxgen.localconfig] LoadTopLevel -> INFO 004 Loaded configuration: /home/alpha/fabric-samples/first-network/configtx.yaml 2019-05-30 16:41:53.973 KST [common.tools.configtxgen] doOutputAnchorPeersUpdate -> INFO 005 Generating anchor peer update 2019-05-30 16:41:53.977 KST [common.tools.configtxgen] doOutputAnchorPeersUpdate -> INFO 006 Writing anchor peer update
④ cli, couchdb, etcdraft docker 이미지 생성	$ docker-compose -f docker-compose-cli.yaml -f docker-compose-couch.yaml -f docker-compose-etcdraft2.yaml up -d alpha@fabric2:~/fabric-samples/first-network$ docker-compose -f docker-compose-cli.yaml -f docker-compose-couch.yaml -f docker-compose-etcdraft2.yaml up -d Creating network "net_byfn" with the default driver Creating volume "net_orderer.example.com" with default driver Creating volume "net_orderer2.example.com" with default driver Creating volume "net_peer0.org2.example.com" with default driver Creating volume "net_peer0.org1.example.com" with default driver Creating volume "net_peer1.org1.example.com" with default driver Creating volume "net_peer1.org2.example.com" with default driver Creating volume "net_orderer5.example.com" with default driver Creating volume "net_orderer4.example.com" with default driver Creating volume "net_orderer3.example.com" with default driver Creating orderer.example.com Creating orderer3.example.com Creating orderer2.example.com Creating couchdb2 Creating couchdb0 Creating couchdb3 Creating orderer5.example.com Creating couchdb1 Creating orderer4.example.com Creating peer1.org2.example.com Creating peer0.org2.example.com Creating peer1.org1.example.com Creating peer0.org1.example.com Creating cli

⑤ cli에 들어가서 chaincode 관련 작업 할 사전준비	$ docker exec -it cli bash → cli 안으로 들어가서 명령 실행
	# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
	CORE_PEER_ADDRESS=peer0.org1.example.com:7051
	CORE_PEER_LOCALMSPID="Org1MSP"
	CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt

⑥ 2번에서 실행한걸 바탕으로 channel 생성 (채널명 = mychannel)	# export CHANNEL_NAME=mychannel # peer channel create -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/channel.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem bash-4.4# peer channel create -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/channel.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem 2019-05-30 07:45:53.725 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized 2019-05-30 07:45:53.768 UTC [cli.common] readBlock -> INFO 002 Got status: &{NOT_FOUND} 2019-05-30 07:45:53.797 UTC [channelCmd] InitCmdFactory -> INFO 003 Endorser and orderer connections initialized 2019-05-30 07:45:54.000 UTC [cli.common] readBlock -> INFO 004 Got status: &{SERVICE_UNAVAILABLE} 2019-05-30 07:45:54.008 UTC [channelCmd] InitCmdFactory -> INFO 005 Endorser and orderer connections initialized 2019-05-30 07:45:54.211 UTC [cli.common] readBlock -> INFO 006 Got status: &{SERVICE_UNAVAILABLE} 2019-05-30 07:45:54.215 UTC [channelCmd] InitCmdFactory -> INFO 007 Endorser and orderer connections initialized 2019-05-30 07:45:54.420 UTC [cli.common] readBlock -> INFO 008 Got status: &{SERVICE_UNAVAILABLE} 2019-05-30 07:45:54.425 UTC [channelCmd] InitCmdFactory -> INFO 009 Endorser and orderer connections initialized 2019-05-30 07:45:54.631 UTC [cli.common] readBlock -> INFO 00a Received block: 0

⑦ peer0.org1을 channel에 join 시키기	# peer channel join -b mychannel.block bash-4.4# peer channel join -b mychannel.block 2019-05-30 07:46:21.925 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized 2019-05-30 07:46:22.239 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel

⑧ peer0.org2를 channel에 join 시키기	# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
	CORE_PEER_ADDRESS=peer0.org2.example.com:9051
	CORE_PEER_LOCALMSPID="Org2MSP"
	CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
	# peer channel join -b mychannel.block

⑨ 각각의 org별 anchor peer 등록하기	# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
	CORE_PEER_ADDRESS=peer0.org1.example.com:7051
	CORE_PEER_LOCALMSPID="Org1MSP"
	CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt → 환경변수 설정
	# peer channel update -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/Org1MSPanchors.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem → Org1 anchor peer 등록
	---------- 	
	# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
	CORE_PEER_ADDRESS=peer0.org2.example.com:9051
	CORE_PEER_LOCALMSPID="Org2MSP"
	CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt → 환경변수 설정
	# peer channel update -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/Org2MSPanchors.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem → Org2 anchor peer 등록
	bash-4.4# peer channel update -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/Org2MSPanchors.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
	2019-05-30 07:50:25.448 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
	2019-05-30 07:50:25.468 UTC [channelCmd] update -> INFO 002 Successfully submitted channel update

⑩ 모든 설정이 완료되었으니 chaincode 설치 후 instantiate화 하기 (현재 peer0.org2)	# peer chaincode install -n mycc -v 1.0 -l java -p /opt/gopath/src/github.com/chaincode/chaincode_example02/java/ root@bd1fe8a41266:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode install -n mycc -v 1.0 -l java -p /opt/gopath/src/github.com/chaincode/chaincode_example02/java/ 2019-05-30 08:49:19.206 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc 2019-05-30 08:49:19.207 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc 2019-05-30 08:49:19.258 UTC [chaincodeCmd] install -> INFO 003 Installed remotely response:<status:200 payload:"OK" > # peer chaincode instantiate -o orderer.example.com:7050 --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc -l java -v 1.0 -c '{"Args":["init","a", "100", "b","200"]}' -P "AND ('Org1MSP.peer','Org2MSP.peer')" root@bd1fe8a41266:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode instantiate -o orderer.example.com:7050 --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc -l java -v 1.0 -c '{"Args":["init","a", "100", "b","200"]}' -P "AND ('Org1MSP.peer','Org2MSP.peer')" 2019-05-30 08:51:03.616 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 001 Using default escc 2019-05-30 08:51:03.616 UTC [chaincodeCmd] checkChaincodeCmdParams -> INFO 002 Using default vscc

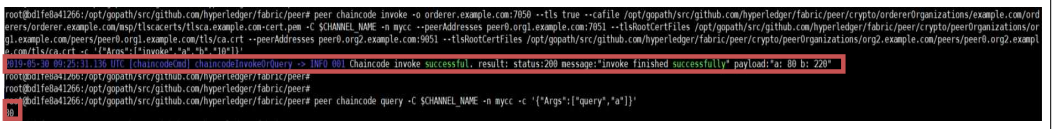
⑪ peer0.org1에 체인코드 설치 후 쿼리 날려보기	# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
	CORE_PEER_ADDRESS=peer0.org1.example.com:7051
	CORE_PEER_LOCALMSPID="Org1MSP"
	CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt → 환경변수 설정
	# peer chaincode install -n mycc -v 1.0 -l java -p /opt/gopath/src/github.com/chaincode/chaincode_example02/java/ → 체인코드 설치
	# peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'
	`root@bd1fe8a41266:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'`
	`100`

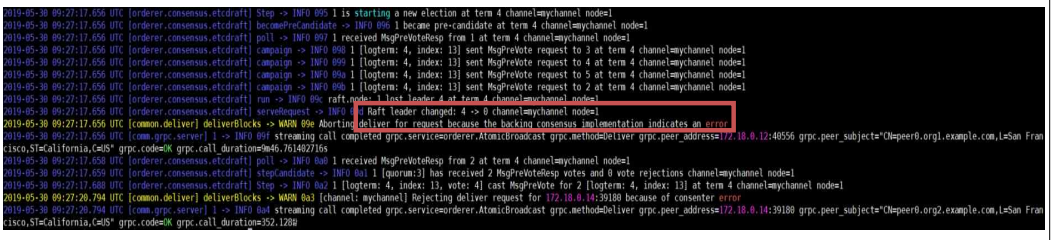
⑫ invoke 후 query 날리기 (현재 peer0.org1)	# peer chaincode invoke -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"Args":["invoke","a","b","10"]}' `root@bd1fe8a41266:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode invoke -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"Args":["invoke","a","b","10"]}'` `2019-05-30 09:02:54.105 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200 message:"invoke finished successfully" payload:"a: 90 b: 210"` # peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}' `root@bd1fe8a41266:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}'` `90`
⑬ 이런식으로 모든 peer에 chaincode 설치 후 확인해보면 이상 무!(docker ps 해보면 다 켜져있음)	

⑭ Raft 리더 찾기
(인증서 다 나오고 log 중반부에 위치)
*** 현재 리더 : orderer5**
	$ docker logs orderer3.example.com
	`2019-05-30 08:48:07.416 UTC [orderer.common.server] Start -> INFO 01c Starting orderer:`
	`Version: 1.4.1`
	`Commit SHA: 87074a7`
	`Go version: go1.11.5`
	`OS/Arch: linux/amd64`
	`2019-05-30 08:48:07.416 UTC [orderer.common.server] Start -> INFO 01d Beginning to serve requests`
	`2019-05-30 08:48:07.615 UTC [orderer.consensus.etcdraft] apply -> INFO 01e Applied config change to add node 1, current nodes in channel: [1 2 3 4 5] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.615 UTC [orderer.consensus.etcdraft] apply -> INFO 01f Applied config change to add node 2, current nodes in channel: [1 2 3 4 5] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.615 UTC [orderer.consensus.etcdraft] apply -> INFO 020 Applied config change to add node 3, current nodes in channel: [1 2 3 4 5] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.615 UTC [orderer.consensus.etcdraft] apply -> INFO 021 Applied config change to add node 4, current nodes in channel: [1 2 3 4 5] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.615 UTC [orderer.consensus.etcdraft] apply -> INFO 022 Applied config change to add node 5, current nodes in channel: [1 2 3 4 5] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.903 UTC [orderer.consensus.etcdraft] Step -> INFO 023 3 [logterm: 1, index: 5, vote: 0] cast MsgPreVote for 5 [logterm: 1, index: 5] at term 1 channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.913 UTC [orderer.consensus.etcdraft] Step -> INFO 024 3 [term: 1] received a MsgVote message with higher term from 5 [term: 2] channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.913 UTC [orderer.consensus.etcdraft] becomeFollower -> INFO 025 3 became follower at term 2 channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.913 UTC [orderer.consensus.etcdraft] Step -> INFO 026 3 [logterm: 1, index: 5, vote: 0] cast MsgVote for 5 [logterm: 1, index: 5] at term 2 channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.945 UTC [orderer.consensus.etcdraft] run -> INFO 027 raft.node: 3 elected leader 5 at term 2 channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:07.948 UTC [orderer.consensus.etcdraft] serveRequest -> INFO 028 Raft leader changed: 0 -> 5 channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:43.047 UTC [orderer.consensus.etcdraft] writeBlock -> INFO 029 Writing block [1] (Raft index: 7) to ledger channel=byfn-sys-channel node=3`
	`2019-05-30 08:48:43.064 UTC [fsblkstorage] newBlockfileMgr -> INFO 02a Getting block information from block storage`
	`2019-05-30 08:48:43.105 UTC [orderer.consensus.etcdraft] HandleChain -> INFO 02b EvictionSuspicion not set, defaulting to 10m0s`
	`2019-05-30 08:48:43.105 UTC [orderer.consensus.etcdraft] createOrReadWAL -> INFO 02c No WAL data found, creating new WAL at path '/var/hyperledger/production/orderer/etcdraft/wal/mychannel' channel=mychannel node=3`
	`2019-05-30 08:48:43.138 UTC [orderer.common.multichannel] newChain -> INFO 02d Created and starting new chain mychannel`
	`2019-05-30 08:48:43.138 UTC [orderer.consensus.etcdraft] Start -> INFO 02e Starting Raft node channel=mychannel node=3`
	`2019-05-30 08:48:43.141 UTC [orderer.common.cluster] Configure -> INFO 02f Entering, channel: mychannel, nodes: [ID: 5,`

⑮ 리더 orderer 죽이기	$ docker stop orderer5.example.com

⑯ 다른 orderer를 통한 로그 확인 : 마지막 부분에 리더 바뀐 걸 알 수 있음!
*** 현재 리더 : orderer4**
	$ docker logs orderer2.example.com
	`2019-05-30 09:17:24.415 UTC [orderer.consensus.etcdraft] logSendFailure -> ERRO 073 Failed to send StepRequest to 5, because: aborted channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.421 UTC [orderer.consensus.etcdraft] poll -> INFO 074 2 received MsgVoteResp from 1 at term 3 channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.421 UTC [orderer.consensus.etcdraft] stepCandidate -> INFO 075 2 [quorum:3] has received 2 MsgVoteResp votes and 0 vote rejections channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.421 UTC [orderer.consensus.etcdraft] poll -> INFO 076 2 received MsgVoteResp from 4 at term 3 channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.422 UTC [orderer.consensus.etcdraft] stepCandidate -> INFO 077 2 [quorum:3] has received 3 MsgVoteResp votes and 0 vote rejections channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.422 UTC [orderer.consensus.etcdraft] becomeLeader -> INFO 078 2 became leader at term 3 channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.422 UTC [orderer.consensus.etcdraft] run -> INFO 079 raft.node: 2 elected leader 2 at term 3 channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.429 UTC [orderer.consensus.etcdraft] serveRequest -> INFO 07a Raft leader changed: 0 -> 2 channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.435 UTC [orderer.consensus.etcdraft] serveRequest -> INFO 07b Start accepting requests as Raft leader at block [1] channel=byfn-sys-channel node=2`
	`2019-05-30 09:17:24.687 UTC [orderer.consensus.etcdraft] Step -> INFO 07c 2 [logterm: 3, index: 11, vote: 5] cast MsgPreVote for 4 [logterm: 3, index: 11] at term 3 channel=mychannel node=2`
	`2019-05-30 09:17:24.691 UTC [orderer.consensus.etcdraft] Step -> INFO 07d 2 [term: 3] received a MsgVote message with higher term from 4 [term: 4] channel=mychannel node=2`
	`2019-05-30 09:17:24.691 UTC [orderer.consensus.etcdraft] becomeFollower -> INFO 07e 2 became follower at term 4 channel=mychannel node=2`
	`2019-05-30 09:17:24.691 UTC [orderer.consensus.etcdraft] Step -> INFO 07f 2 [logterm: 3, index: 11, vote: 0] cast MsgVote for 4 [logterm: 3, index: 11] at term 4 channel=mychannel node=2`
	`2019-05-30 09:17:24.691 UTC [orderer.consensus.etcdraft] run -> INFO 080 raft.node: 2 lost leader 5 at term 4 channel=mychannel node=2`
	`2019-05-30 09:17:24.699 UTC [orderer.consensus.etcdraft] serveRequest -> INFO 081 Raft leader changed: 5 -> 0 channel=mychannel node=2`
	`2019-05-30 09:17:24.702 UTC [orderer.consensus.etcdraft] run -> INFO 082 raft.node: 2 elected leader 4 at term 4 channel=mychannel node=2`
	`2019-05-30 09:17:24.710 UTC [orderer.consensus.etcdraft] serveRequest -> INFO 083 Raft leader changed: 0 -> 4 channel=mychannel node=2`

⑰ 두 개의 orderer가 죽어도 network가 잘 돌아가는지 확인하기 위해 cli창에 들어가서 테스트 해보기 (invoke 및 query)	$ docker exec -it cli bash # peer chaincode invoke -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C $CHANNEL_NAME -n mycc --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses peer0.org2.example.com:9051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"Args":["invoke","a","b","10"]}' # peer chaincode query -C $CHANNEL_NAME -n mycc -c '{"Args":["query","a"]}' 
⑱ 한 번 더 리더 orderer 죽이기 *** 현재 리더 : orderer4**	$ docker stop orderer4.example.com → 새로운 리더 선출됨(3번)

⑲ 한 번 더 리더 orderer 죽이기
*** 현재 리더 : orderer3**
	$ docker stop orderer3.example.com
	→ 이제 새로운 리더 선출 안 됨(WHY? 우리가 5개의 orderer를 만들었는데 3개를 죽이면서 과반수 이상 동의해야한다는 룰에 어긋나게 되므로 선출되지 않음)
	 
	--- 	
	$ docker exec -it cli bash → cli로 들어가 chaincode test 해보면 오류 발생!!
	 
	→ query를 날리면 orderer 없이 값을 가져오는 것이므로 작동하지만, invoke는 peer들 간의 합의가 필요하기 때문에 Raft leader가 없어서 작동하지 않음.

⑳ 죽인 orderer를 다시 살려서 로그를 체크 *** 현재 리더 : orderer1**	$ docker start orderer3.example.com → 0에서 1로 바뀜 
결론 : 우리는 Raft 알고리즘을 이용해 orderer의 leader가 충돌 등의 사유로 죽으면, 자동선출을 통해 네트워크를 지속적으로 유지할 수 있다!	

* 실습 내용은 수동으로 설정한 것이고 한방에 키는 법도 있음(실제 개발 간 이 방법을 사용)
$ ./byfn.sh up -o etcdraft -l java -s couchdb → 약 3분정도 소요(script 파일이라 자동실행)

3. 참고자료

① etcdraft에서 etcd란? https://www.joinc.co.kr/w/man/12/etcd : 분산 key-value store