< 하이퍼레져 페브릭 1.4 환경설정 >

2019년 1월 9일 정식 release된 hyperledger fabric 1.4.0에 대한 사전준비 방법

1. ssh : putty나 X-shell을 사용하기 위한 설치(네트워크는 '어텝터에 브릿지')

$ sudo apt-get install openssh-server → 우분투는 기본적으로 ssh-client가 설치되어 있고 server는 미설치
server를 설치해줘야 본 서버에서 client가 설치된 서버로 원격접속 가능!!
http://programmingskills.net/archives/315 참고

$ sudo service —status-all

2. rmate : 텍스트 편집을 좀 더 쉽게 하기 위해서 설치

$ wget https://raw.githubusercontent.com/sclukey/rmate-python/master/bin/rmate
$ chmod +x ./rmate → 파일 권한 설정(rmate 파일의 모든 사용자에게 실행(x)기능 추가)
$ sudo mv ./rmate /usr/local/bin/rmate → local로 이동

3. user 환경변수 설정 : .bashrc와 .profile의 차이점(https://uroa.tistory.com/114 참고)

$ vi .bashrc (or .profile) → 들어가서 파일 가장 하단(ctrl + G)에 아래 변수 6개 작성

```
export GOROOT=/usr/local/go
export GOPATH=$HOME/gopath
export PATH=$PATH:$GOROOT/bin
export FABRIC_HOME=$GOPATH/src/github.com/hyperledger/fabric
export PATH=$PATH:$GOPATH/src/github.com/hyperledger/fabric/.build/bin
export PATH=$PATH:$GOPATH/src/github.com/hyperledger/fabric-ca/bin
```

$ source .bashrc → 환경변수 6개 설정 후 활성화

4. curl : command url의 약자로 cli 창에서 url에 접속할 수 있는 방법

$ sudo apt-get install –y curl → curl과 wget의 차이점(https://brocess.tistory.com/114 참고)

5. git

$ sudo apt-get install –y git

6. tree

$ sudo apt-get install –y tree

7. libtool : fabric-ca 설치 시 필요한 라이브러리로 dynamic link를 처리하기 위해 필요함

$ sudo apt-get install –y libltdl-dev
→ https://www.lesstif.com/pages/viewpage.action?pageId=12943542 참고
http://bbs.nicklib.com/application/3308 참고(동적 라이브러리 생성에 필요)

8. net-tool

$ sudo apt-get install –y net-tools

9. go lang : 1.11.x 이상이어야 가능

$ wget https://dl.google.com/go/go1.11.linux-amd64.tar.gz
$ sudo tar –xvf go1.11.linux-amd64.tar.gz → x(extract, 발췌), v(verbose, 장황한), f(file) 압축푸는 옵션
$ sudo mv go /usr/local
재접속

$ which go → 앞서 환경변수를 설정해줬기 때문에 확인해보면 /usr/local/go/bin/go에 들어있음

$ go version → 1.11.x 확인

10. nodejs : node-sdk를 쓰기 위해서는 8.9.x 이상이어야 가능

$ curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash - → -E(preserve-env, bash 환경보호)

$ sudo apt-get install –y nodejs

$ node --version → 8.15.1 나옴

버전문제의 경우 재설치 해야 함

$ sudo apt-get remove nodejs

$ curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -

$ sudo apt-get update → 조금 오래걸림

$ sudo apt-get upgrade

$ sudo apt-get install -y nodejs

$ sudo npm install –g npm@5.6.0 → npm 버전 업그레이드(ERR! 이란 글자가 보이면 안됨)

```
project-b@projectb-VirtualBox:~$ sudo npm install -g npm@5.6.0
/usr/bin/npm -> /usr/lib/node_modules/npm/bin/npm-cli.js
/usr/bin/npx -> /usr/lib/node_modules/npm/bin/npx-cli.js
+ npm@5.6.0
added 363 packages from 147 contributors, removed 274 packages and updated 43 packages in 50.177s
```

$ sudo npm rebuild

---

11. python : 우분투 16.04에서는 2.7과 3.5 기본 제공(2.7을 사용해야 fabric 가능)

$ sudo apt-get install –y python

$ sudo apt-get install –y python-pip → 시간 좀 걸림

12. docker : 17.06.2 이상이어야 함

$ curl -fsSL https://get.docker.com/ | sudo sh → 시간 오래 걸림(fail, silient, show-error, location)

$ sudo usermod –aG docker $(whoami) → aG(append Group), $(whoami)에 ID입력

$ docker --version → 17.06.2 이상 확인

13. docker-compose : 1.14.0 이상이어야 함

$ sudo apt-get update → 시간 오래 걸림(apt=Advanced Packaging Tool의 약자로 패키지 관리 명령어 도구)

$ upgrade → 시간 굉장히 오래 걸림

$ sudo apt-get install –y docker-compose

$ docker images → 설치 후 바로하면 permission denide 뜸 / logout 후 재접속해서 빈 파일 확인

$ docker-compose --version → 1.14.0 이상 확인

또는

$ sudo curl -L https://github.com/docker/compose/releases/download/1.18.0/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose

$ sudo chmod +x /usr/local/bin/docker-compose

14. hyperledge fabric-samples

$ curl –sSL http://bit.ly/2ysbOFE | bash –s -- 1.4.0

→ 15분 가량 소요

위의 명령은 네트워크를 설정하고 배치해야하는 모든 플랫폼 관련 바이너리를 다운로드하고 추출하는 bash 스크립트를 다운로드하고 실행(peer, orderer, fabric-ca, kafca, couch-db 등 9개)

사실상 fabric 설치 코드는 위에 한줄!!(나머지는 docker를 사용하기 위한 환경설정 수준)

$ docker images → 9개의 이미지 별 인스턴스가 2개씩 생성됨(1.4.0과 latest / 이 각각이 fabric 동작의 핵심)

```
===> List out hyperledger docker images
hyperledger/fabric-javaenv      1.4.0     3d91b3bf7118    7 weeks ago     1.75GB
hyperledger/fabric-javaenv      latest    3d91b3bf7118    7 weeks ago     1.75GB
hyperledger/fabric-tools        1.4.0     0a44f4261a55    8 weeks ago     1.56GB
hyperledger/fabric-tools        latest    0a44f4261a55    8 weeks ago     1.56GB
hyperledger/fabric-ccenv        1.4.0     5b31d55f5f3a    8 weeks ago     1.43GB
hyperledger/fabric-ccenv        latest    5b31d55f5f3a    8 weeks ago     1.43GB
hyperledger/fabric-orderer      1.4.0     54f372205580    8 weeks ago     150MB
hyperledger/fabric-orderer      latest    54f372205580    8 weeks ago     150MB
hyperledger/fabric-peer         1.4.0     304fac59b501    8 weeks ago     157MB
hyperledger/fabric-peer         latest    304fac59b501    8 weeks ago     157MB
hyperledger/fabric-ca           1.4.0     1a804ab74f58    8 weeks ago     244MB
hyperledger/fabric-ca           latest    1a804ab74f58    8 weeks ago     244MB
hyperledger/fabric-zookeeper    0.4.14    d36da0db87a4    4 months ago    1.43GB
hyperledger/fabric-zookeeper    latest    d36da0db87a4    4 months ago    1.43GB
hyperledger/fabric-kafka        0.4.14    a3b095201c66    4 months ago    1.44GB
hyperledger/fabric-kafka        latest    a3b095201c66    4 months ago    1.44GB
hyperledger/fabric-couchdb      0.4.14    f14f97292b4c    4 months ago    1.5GB
hyperledger/fabric-couchdb      latest    f14f97292b4c    4 months ago    1.5GB
```

15. MySQL

$ docker run -d -p 3306:3306 -e MYSQL_ALLOW_EMPTY_PASSWARD=true --name mysql mysql:5.7

→ 도커를 실행하는데 데몬버전(백에서) 실행하고 포트는 3306(HOST PC와 DOCKER를 연결해주는)을 사용, 환경변수, 비밀번호는 없이 사용하고 이름은 mysql이라 부여하겠다. 버전은 mysql:5.7 사용!

$ docker exec -it mysql bash → mysql이란 이름을 bash셸 모드로 실행

mysql –h127.0.0.1 -uroot → 로그인(위에 비번 설정 안했으니까 그냥 들어가짐) 이후 SELECT문 등 사용 가능!!

16. fabric 1.4

$ mkdir –p $GOPATH/src/github.com/hyperledger → home에서 실행(path 옵션을 통해 4개 dir 한 번에 생성)

$ cd $GOPATH/src/github.com/hyperledger → $GOPATH는 위에 환경변수 설정해둠($HOME/gopath)

$ git clone –b release-1.4 https://github.com/hyperledger/fabric → 오래 걸림

$ curl -sSL http://bit.ly/2ysbOFE | bash –s -- 1.4.0 → 위에꺼나 이거나 둘 중 하나(이건 안 해봄)

$ cd $GOPATH/src/github.com/hyperledger/fabric

$ git branch → 버전확인 'release-1.4' 라고 뜨는 것 확인

$ make → unit-test 이후 반복적으로 코드들 나오면 ctrl+c를 통해 종료(약 1시간(?) 소요 so 오래 걸림)

→ make는 프로그램 그룹을 유지하는데 필요한 유틸리티로 자동 compile 해주는 역할

소스파일로 이루어진 것들을 컴파일 해 실행파일로 만들어 줌(https://maeuminpaper.tistory.com/83 참고)

https://wiki.kldp.org/KoreanDoc/html/gcc_and_make/gcc_and_make-3.html 참고

gcc(GNU Compiler Collection) 명령어를 통해 make가 만들어짐

17. fabric-ca 1.4

$ cd $GOPATH/src/github.com/hyperledger → $GOPATH는 위에 환경변수 설정해둠($HOME/gopath)

$ git clone –b release-1.4 https://github.com/hyperledger/fabric-ca → 오래 걸림

$ cd $GOPATH/src/github.com/hyperledger/fabric-ca

$ make fabric-ca-server → 아래 결과화면 참고

$ make fabric-ca-client → 아래 결과화면 참고

```
project-b@projectb-VirtualBox:~/gopath/src/github.com/hyperledger/fabric-ca$ make fabric-ca-server
Building fabric-ca-server in bin directory ...
Built bin/fabric-ca-server
project-b@projectb-VirtualBox:~/gopath/src/github.com/hyperledger/fabric-ca$ make fabric-ca-client
Building fabric-ca-client in bin directory ...
Built bin/fabric-ca-client
```

18. window 엑스트라

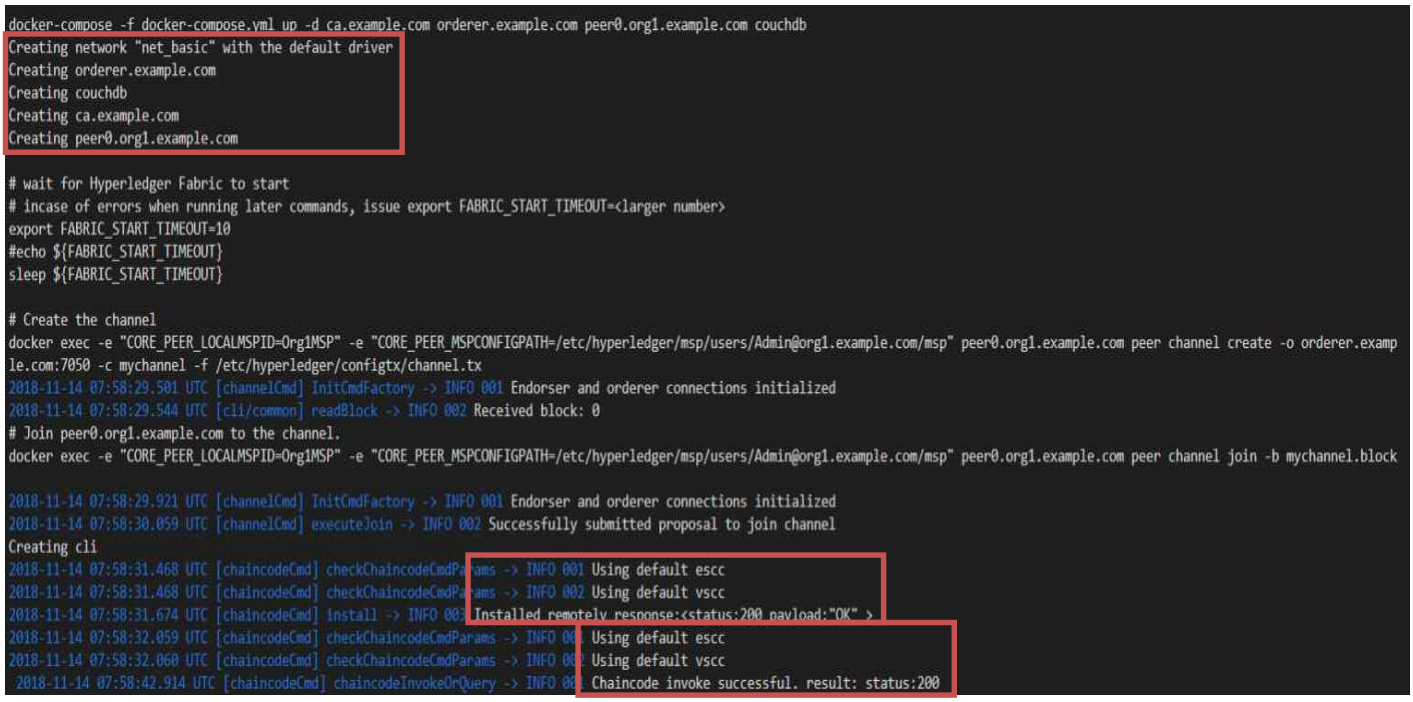
$ git config --global core.autocrlf false
$ git config --global core.longpaths true
위 두 명령어 실행 후
$ git config --get core.autocrlf → false 확인
$ git config --get core.longpaths → true 확인
$ sudo npm install --global windows-build-tools
→ node.js의 자유로운 활용을 위한 Visual Studio C++ 빌드도구 설치(linux라 오류 발생)
$ sudo npm install --global grpc --unsafe-perm → --unsafe-perm 옵션을 통해 npm이 root 계정으로 실행하도록 지정

19. 최종 확인

$ cd fabric-samples/fabcar
$ ./startFabric.sh → 명령어 실행 시 docker images가 6개 생성

만약 안 될 경우?

$ docker stop $(docker ps –qa) && docker rm $(docker ps –qa)
$ ./startFabric.sh → 컨테이너 간 꼬여서 error 발생한 것이기 때문에 지우고 다시 실행하면 이상 무!



→ 이렇게 해서 페브릭이 시작되고 $docker ps해보면 6개의 컨테이너가 활성화됨!!
시간이 조금 오래 걸리는 이유는 chaincode를 install하고 invoke하기 때문!
→ 1.4부터는 이 후 진행과정 설명이 들어감 javascript와 typescript 둘 중 하나 선택 이용 가능

```
JavaScript:

  Start by changing into the "javascript" directory:
    cd javascript

  Next, install all required packages:
    npm install

  Then run the following applications to enroll the admin user, and register a new user
  called user1 which will be used by the other applications to interact with the deployed
  FabCar contract:
    node enrollAdmin
    node registerUser

  You can run the invoke application as follows. By default, the invoke application will
  create a new car, but you can update the application to submit other transactions:
    node invoke

  You can run the query application as follows. By default, the query application will
  return all cars, but you can update the application to evaluate other transactions:
    node query
```

위 그림에서 안내된 대로 명령어 순차 실행($ npm install 성공적으로 끝나야 다음 진행 가능함)

$ node enrollAdmin.js → Successfully enrolled admin user "admin" 뜨면 성공(관리자가 생성)

$ node registerUser.js → 유저 만들기(successfully 뜸)

→ 1.4부터는 wallet 기능이 추가되어 키가 ./hfc-key-store에 생성되는게 아니라 지갑에 생성됨

```
project-b@projectb-VirtualBox:~/fabric-samples/fabcar/javascript$ ls
enrollAdmin.js  invoke.js  node_modules  package-lock.json  package.json  query.js  registerUser.js  wallet
project-b@projectb-VirtualBox:~/fabric-samples/fabcar/javascript$ tree wallet/
wallet/
├── admin
│   ├── 39d5c2cc68cb558a95f383e65dc6193fd7da618fd9d6eeeb9dd86512a8253d81-priv
│   ├── 4651516298ad9131f28aea54fd45d087e6cfcd1b94fe25cfd29ce0a4c8d94caa-priv
│   ├── 4651516298ad9131f28aea54fd45d087e6cfcd1b94fe25cfd29ce0a4c8d94caa-pub
│   └── admin
└── user1
    ├── 39d5c2cc68cb558a95f383e65dc6193fd7da618fd9d6eeeb9dd86512a8253d81-priv
    ├── 39d5c2cc68cb558a95f383e65dc6193fd7da618fd9d6eeeb9dd86512a8253d81-pub
    └── user1
```

20. 참고사항

① 만약 디스크 용량 부족이라는 에러가 발생 할 경우?!?!

$ df -h → 명령어를 통해 남은 용량 확인(https://technote.kr/156, https://idchowto.com/?p=24058 참고)

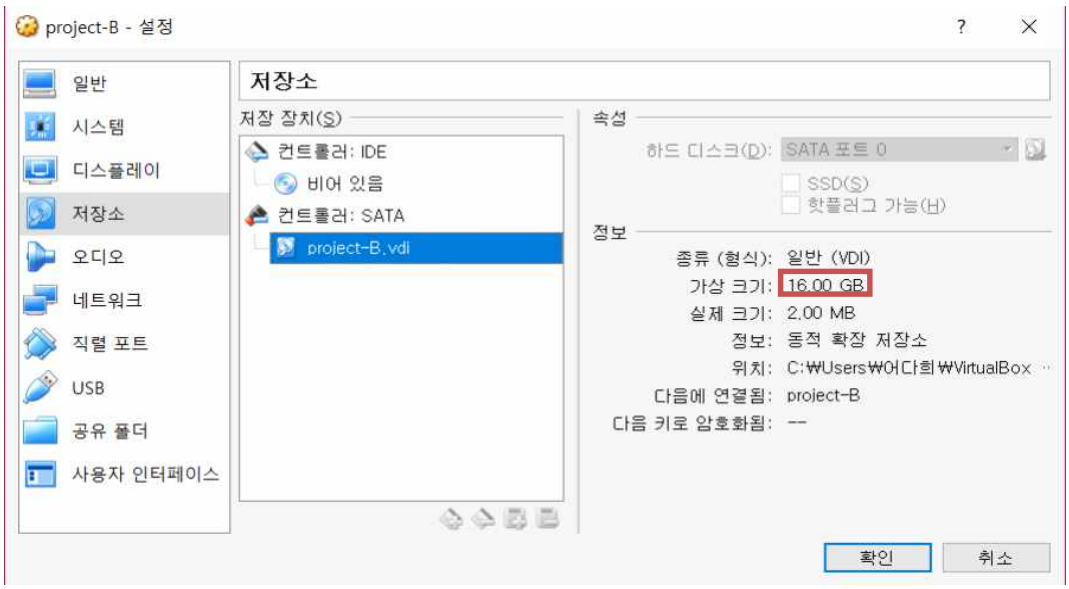
```
project-b@projectb-VirtualBox:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G     0  1.9G   0% /dev
tmpfs           395M  6.1M  389M   2% /run
/dev/sda1        15G  8.4G  5.6G  60% /
tmpfs           2.0G  316K  2.0G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           2.0G     0  2.0G   0% /sys/fs/cgroup
tmpfs           395M  104K  395M   1% /run/user/1000
```

가상머신의 하드디스크 용량을 늘려줘야 함(늘리고, 파티션을 재분배 해줘야 완성)

가상머신이 아닌 VirualBox를 실행시킨 OS에서 실행(즉, window의 cmd창 이용)

$ VBoxManage.exe modifyhd "$(name)" --resize 16384 → VBoxManage.exe는 VB설치 위치에 존재

[C:₩Program Files₩Oracle₩VirtualBox₩VBoxManage.exe]



크기가 16GB로 변경됨

재부팅 후 $df –h 확인해 봐도 그대로임 → 디스크 자체의 파티션을 늘려줘야 가능

$ fdisk /dev/sda

---

p → /dev/sda 내 변경하려고 하는 파티션 정보를 확인
d → 파티션 삭제
2 → /dev/sda2 삭제(부팅디스크 삭제 시 구동이 안되므로 각별 주의!! 1은 안돼!!!)
n → 새로운 파티션 생성
p → Primary 파티션 선택
2 → 2번 파티션 지정(지우고 다시 만드는 것)
엔터 → 실린더 처음값 default(사용자마다 다름)
다시 엔터 → 실린더 맨마지막값 default(사용자마다 다름)
w → 변경사항 저장

$ sudo reboot → 시스템이 리부팅되면, 아래 resize2fs 명령 수행

$ resize2fs /dev/sda2

$ df -h → 최종적으로 사이즈가 늘어난 것을 확인!!

② docker의 기본 명령어들

$ docker ps : process status 현재 구동중인 프로세스 띄워줌

$ docker stop [ID] : 실행중인 프로세스 멈추기

ex) $ docker stop $(docker ps -qa) && docker rm $(docker ps -qa)

$ docker rm [ID] : 프로세스 삭제

$ docker image : 도커의 이미지들 싹 띄워라

$ docker rmi [ID] : 관련 이미지 삭제