2. 이더리움 및 DB 구축

노트북: 블록체인
만든 날짜: 2019-08-26 오전 10:11 **수정한 날짜:** 2019-10-09 오후 4:26
작성자: 감자
태그: Docker
URL: http://whyitworld.blogspot.com/2016/03/docker-2.html

AWS VM 활용 이더리움 네트워크 구축

-3개의 AWS를 이용하여 프로젝트를 운영할것임
-1번 AWS : Ethereum
-2번 AWS : Hyperledger fabric
-3번 AWS : Front-End, Back-End, DB

Frontend

```
-지갑생성, 지갑열람, 코인충전
```

Backend

```
-MySQL 도커 이미지 다운로드 및 테이블 구조 확인

-이더리움 네트워크 연동, 지갑 정보 조회, 코인충전구현
```

---

(1) 1번 AWS로 이더리움 네트워크를 구축하는 방법

STEP1. AWS VM 활용 이더리움 네트워크 구축

-AWS 계정만든 후 EC2 인스턴스 생성 (주소 : 13.125.178.26)

※ EC2 : 가상의 컴퓨터(대부분의 목적이 서버) 를 만들고 관리하는 서비스. EC2에서 만드는 가상의 컴퓨터를 하나의 인스턴스라고 부른다.

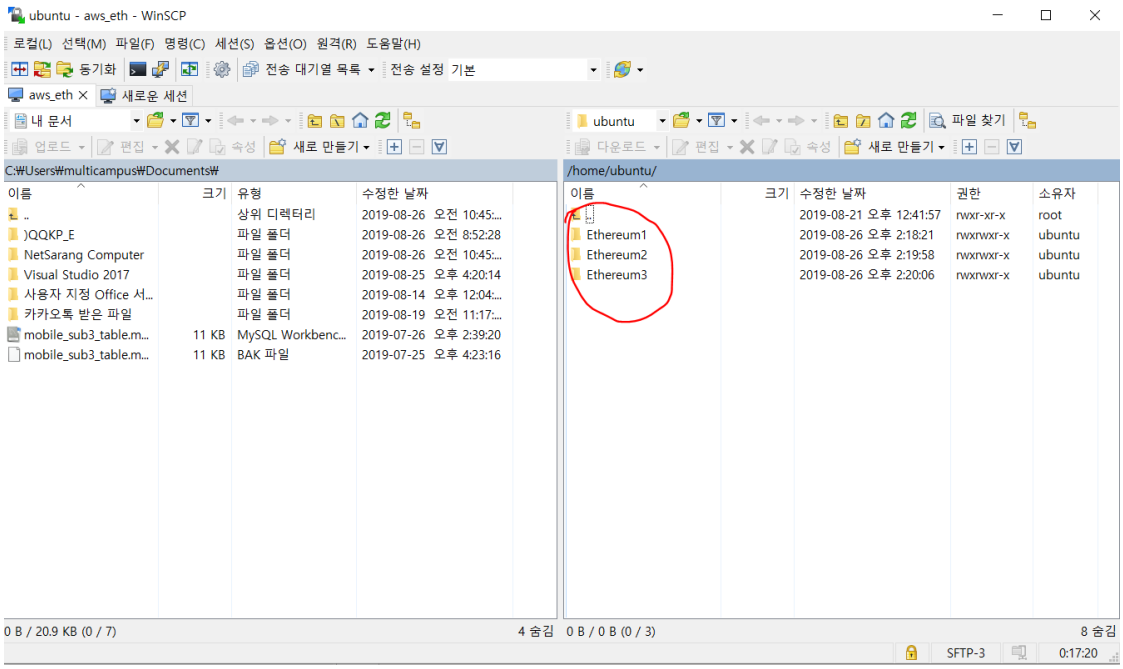
인증키 변환 및 PuTTY 접속 방법 참고:
https://docs.aws.amazon.com/ko_kr/AWSEC2/latest/UserGuide/putty.html -> 이대로 사용하면 참 쉽다 ㅎㅎ

T01C05.pem --> PuTTYgen으로 .pem 파일을 .ppk로 변경 (이건 ssafy에서 제공해준 aws서버키임)

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA3zHvYutzbHOyAFqRdiNtybOWpnCLidoJuI7a/ryX6mwkmJbiQreeEiWis490
zH1E+n/TyWxkmqjc02WyQr3lf8mw+g2fqqWe0p5F6Ys/uC7WeuOk9eQIee/HVS43SL6gZhpWx7dA
2kLTbY3SZmlT+6bThr5irvpYMUXxeiQoVJajIgMFFKZrsmuRcChuqnocs8GH9da1VefIreYQdJ1E
nmpyqciQTurK14YX7GpOmKHNS68UuOQgkAThGYQvGd2AI/8suQ3aGw6dXdfJrqLuunH0kj+YBE1c
SxepNmJUAE9y8exLWK5xy/HSgKToJ8FvfX28gxVbc/vTAUoI5fAizQIDAQABAoIBAHZNMHZ9GCvm
0vVvC+TTeJDNmM9VnhCOOlLRnXAiJF+DvI4Ig5sbWjSCME/+fMkaqOIhuO7tfhGf6ZQ/NhI8NxXZ
oO2ZJQU9TgaTD7lvLnIAmXx/huMyX+/J0RzsJy9uovQbh93J2x0VCJ3r6xaPQ42CamHkLchTN0W+
rfSwj5PtWpKOlHymGpqQvJhTjF9qb2P7jXqnuIQzrimxv9jJ3aUXNdsKo6vj+wh5ee40U/pn1Hgk
MgMVMw8YrlnAAf/sWr/cPcNmhCyfzKcS1QXlUfQhgJqqRqyoyEKLv8A+6bUo7Gp7GtASaZNtlqsv
```

```
+Bt2U6xpaQraITekehks2pVpIr0CgYEA+orOKjpyDXupMh3ThM8nMrRsMLxWpLRUGj74YxKW/Arm
xFjY7o1Wc4ND0PsTiZlJ6sxGkHJI+JXHb94C425/W24E2tN4tJNyYPivOJEqpi9ca1YynnmTv43I
iUigTchskVz5nPUy1O3t/g4U8DkxZA6Yx5u/E8Q8KMVKH4OSb7cCgYEA5A6flQBB1J4K50TyhBbd
ZDZ7IQMHk7sNZ0bFky/ptypsjikTyDyU6qFwCEnHxkvktmuBWfFBpoq7Rql+oS4cPwxm5H8t13+q
foJj7cS+c3fgG6OaSv3snGXnoV6ay8EVJxyTTufr4oMBem3ArgaVpufQr83DVfaWkPdjcNRPeZsC
gYBVAZuJwGLKKFbz5iLKQO7vyT+cGYrcP0L51Lu6Aaiww5IgtHSGG/ZlIvhzg3zRhu0JeV3HzL7t
eQQ99HnOaMNVZjLHfFDME+sO89Gfw8FQuWkGEi0CkAWNTLHpwnaBbatIdt9KXfdtP30sYeLAzubv
J/S2ZSWjs0UmICpSHvYkaQKBgQC1AaZHbNzTFF0zEdYqYLt3PoabvPvWlTAeOF24ZN7rMhDXBGNf
GjZXw6pX6bdV/Aabgtg3uvSQaqwZeY24ABELOZyw3wBTelCn+HW30eAXZZ3KUh+0RbX2rCscOSpD
bgyL7nI8vnbKYZBay6GSfkdpIbTQ1E+lewImLTQP6XLq4QKBgD+2xP1RDI5MSFOGtmxSL1cUJtWi
rhIgiL71niM3CofXFHOVs7XmVOvx2a693yByo9WFvCcih6F/WKXqgT92p7PKYiEwlmA1aOcas01L
rsg9x5gVW722hqQKj03xDrB3GY8fvYxO+/gzsE+pR4rGLLKrjsWSVN/Hr2TZ/d1yixXW
-----END RSA PRIVATE KEY-----
```

--> WinSCP 으로 AWS 서버와 쉽게 연결




```
#폴더 생성
mkdir ./dev/eth_prac001

#가상머신에 geth 설치
sudo apt-get update
sudo apt-get install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get install ethereum

#잘 설치되었나 확인
geth version

#디렉토리 구조 만들기
mkdir -p dev/eth_localdata
cd dev/eth_localdata/

#Geth를 이용해 Ethereum Private Network를 구성하기 위해서는 처음 제네시스 블록을 생성해 주어야 한다.
#CustomGenesis.json에 제네시스 블록에 대한 설정을 해주게 된다.
vi CustomGenesis.json

#위 json파일에 넣기
{
"config": {
"chainId": 15150,
"homesteadBlock": 0,
"eip155Block": 0,
"eip158Block": 0
```

```
},
"difficulty": "0x10",
"coinbase": "0x0000000000000000000000000000000000000000",
"gasLimit": "9999999",
"alloc": {},
"extraData": "",
"nonce": "0xdeadbeefdeadbeef",
"mixhash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
"parentHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
"timestamp": "0x00"
}

#geth 구동
geth --datadir ~/Ethereum1/dev/eth_localdata/ init
~/Ethereum1/dev/eth_localdata/CustomGenesis.json
geth --datadir ~/Ethereum2/dev/eth_localdata/ init
~/Ethereum2/dev/eth_localdata/CustomGenesis.json
geth --datadir ~/Ethereum3/dev/eth_localdata/ init
~/Ethereum3/dev/eth_localdata/CustomGenesis.json


sudo apt install tree
tree 쳐서 확인해보기

#static-nodes.json 만들기 -> 1번 노드를 메인드로 사용할것이므로 1번에만 쓰면 됨 나머지만 연결
{
 "geth --networkid 15150 --maxpeers 5 --datadir
~/Ethereum2/dev/eth_localdata/ --port 3334 console 2>>
~/Ethereum2/dev/eth_localdata/geth.log",
 "geth --networkid 15150 --maxpeers 5 --datadir
~/Ethereum3/dev/eth_localdata/ --port 3335 console 2>>
~/Ethereum3/dev/eth_localdata/geth.log"
}


#구동시키기
#1번 노드
geth --networkid 15150 --datadir ~/Ethereum1/dev/eth_localdata --port 3333 --
rpc --rpcport 3336 --rpcaddr 0.0.0.0 --rpccorsdomain "*" --rpcapi
"admin,net,miner,eth,rpc,web3,txpool,debug,db,personal" --allow-insecure-
unlock console 2>> ~/Ethereum1/dev/eth_localdata/geth.log
=> unlock 명령어 먹히려면 --allow~~해야함

#2번 노드
nohup geth --networkid 15150 --maxpeers 3  --port 3334 --datadir
~/Ethereum2/dev/eth_localdata/ --mine -minerthreads 1 --etherbase
"0x34731f88a25c5f21922233d86372f91ae3539ca2" & console 2>>
~/Ethereum2/dev/eth_localdata/geth.log

#3번 노드
nohup geth --networkid 15150 --maxpeers 3  --port 3335 --datadir
~/Ethereum3/dev/eth_localdata/ --mine -minerthreads 1 --etherbase
"0x34731f88a25c5f21922233d86372f91ae3539ca2" & console 2>>
~/Ethereum3/dev/eth_localdata/geth.log

*참고-------------------------------------------------------------------------
------------------------
-nohup이랑 console 앞에 &을 붙여서 계속 돌아가고 있게 할수 있다.
nohup geth --networkid 15150 --datadir ~/Ethereum1/dev/eth_localdata --port
3333 --rpc --rpcport 8545 --rpcaddr 0.0.0.0 --rpccorsdomain "*" --rpcapi
"admin,net,miner,eth,rpc,web3,txpool,debug,db,personal" & console 2>>
~/Ethereum1/dev/eth_localdata/geth.log
=> 빨간색 부분 넣으면 rpc 포트 열어서 접속: 해킹당할 수 있음.

nohup geth --networkid 15150 --datadir ~/Ethereum1/dev/eth_localdata --port
3333 --mine -minerthreads 1 --etherbase
"0x34731f88a25c5f21922233d86372f91ae3539ca2" & console 2>>
~/Ethereum1/dev/eth_localdata/geth.log
```

```
=> rpc 포트 열지 않아서 해킹안당하지만 우리도 못들어감 ㅋㅋ
  geth 작동시킬때 빨간 부분을 넣으면 백그라운드로 돌면서 채굴도 동시에 함

※ nohup 으로 키고 geth를 끄지 않았는데 ps로 안뜨면 (ps -ef | grep geth)로 확인 => 끄는 방법 : kill [PID]
-------------------------------------------------------------------------------------------------------

#1번노드: 2, 2번노드: 1, 3번노드: 1이 잘 나오나 확인
net.peerCount

#주소가 나옴 복사
admin.nodeInfo.enode

personal.newAccount()
:비밀번호 입력
```
