

이더리움 네트워크를 구축 및 모니터링

이더리움이란

- 스마트 계약을 실행할 수 있는 플랫폼
- 블록 체인 기술을 기반으로 스마트 계약 기능을 구현하기 위한 분산 컴퓨팅 플랫폼이자 운영체제

- 블록 체인 :

관리 대상 데이터를 '블록'이라고 하는 소규모 데이터들이 P2P 방식을 기반으로 생성된 체인 형태의 연결고리 기반 분산 데이터 저장 환경에 저장하여 누구라도 임의로 수정할 수 없고 누구나 변경의 결과를 열람할 수 있는 분산 컴퓨팅 기술 기반의 원장 관리 기술

- 이더리움 클라이언트 종류

클라이언트	언어	개발자
go-ethereum	Go	Ethereum Foundation
Parity	Rust	Ethcore
cpp-ethereum	C++	Ethereum Foundation
pyethapp	Python	Ethereum Foundation
ethereumjs-lib	JavaScript	Ethereum Foundation
Ethereum(J)	Java	<ether.camp>
ruby-ethereum	Ruby	Jan xie
ethereumH	Haskell	BlockApps

이더리움 네트워크 구축(/w docker)

- 테스트 할 클라이언트
 - geth :
 - Go 언어를 기반 제작
 - 이더리움 풀노드를 구현/작동 위한 명령어 라인 인터페이스를 제공
 - 클라이언트 소프트웨어중 가장 많이 사용
- Genesis.json 생성
 - Geth를 이용해 Ethereum private Network를 구성하기 위해서는 처음 제네시스 블록을 생성해 주어야 하는데 Genesis.json에 제네시스 블록에 대한 설정
- Genesis.json config 정보
 - chainId : 현재 Chain을 식별하는 값

- homesteadBlock : 블록체인의 Release버전 (기본값은 0)
- difficulty : 이 블록의 nonce값을 발견하는 난이도 레벨을 설정. difficulty가 높을수록 블록 생성 속도가 느려진다.
- gasLimit : 체인 전체에 대한 블록 당 가스지출의 제한량을 설정
- alloc : Genesis 블록 생성 시 지정한 지갑에 할당된 양을 미리 채운다.
- nonce : PoW 알고리즘에 사용되는 nonce값
- mixhash : nonce값과 결합하여 이 블록에 충분한 양의 계산이 수행되었음을 증명하는 256bit의 해시값(mixHash는 해당 이더리움 체인 내에서의 난이도를 결정한다)
- parentHash : 이전 block header의 Keccak 256bit 해시값
- timestamp : block을 생성할 때 Unix time함수의 결과값
- 예제 :

```
{
  "config": {
    "chainId": 4444,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "difficulty": "0x20000",
  "gasLimit": "0x2febd8",
  "alloc": {},
  "extraData": "",
  "nonce": "0x0000000000000000",
  "mixhash":
    "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash":
    "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x00"
}
```

- Gitlab 을 이용한 Image Build

- Dockerfile

```
FROM ubuntu

ARG DEBIAN_FRONTEND=noninteractive
ENV TZ=Asia/Seoul

RUN apt-get update
RUN apt-get install -y build-essential libgmp3-dev golang git
tzdata
```

```

RUN git clone https://github.com/ethereum/go-ethereum.git
#RUN mkdir /home/go-ethereum
#COPY go-ethereum /home/go-ethereum
WORKDIR go-ethereum

RUN git checkout v1.9.21
RUN make geth
RUN cp build/bin/geth /usr/local/bin/

WORKDIR /DATA_STORAGE
COPY ./genesis.json /DATA_STORAGE

RUN geth --datadir "/DATA_STORAGE" init /DATA_STORAGE/genesis.json

```

■ Genesis.json

```

{
  "config": {
    "chainId": 10,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip150Hash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "constantinopleBlock": 0,
    "petersburgBlock": 0,
    "istanbulBlock": 0,
    "ethash": {}
  },
  "nonce": "0x0",
  "timestamp": "0x5e4a53b2",
  "extraData":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "gasLimit": "0x47b760",
  "difficulty": "0x80000",
  "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "alloc": {
    "0000000000000000000000000000000000000000000000000000000000000088": {
      "balance":
"0x2000000000000000000000000000000000000000000000000000000000000000"
    }
  },
  "number": "0x0",

```

```

    "gasUsed": "0x0",
    "parentHash":
    "0x0000000000000000000000000000000000000000000000000000000000000000
    0"
  }

```

- Gitlab CI

- .gitlab-ci.yml

```

stages:
- build

build_stage:
image: docker
stage: build

services:
- docker:dind

variables:
  REGISTRY: localhost:5000

script:
- >
  docker build --no-cache
  . -t $REGISTRY/test/ehereumtet:v20210917
  -t $REGISTRY/test/ehereumtest:latest
- docker push $REGISTRY/test/ehereumtest --all-tags
tags:
- test

```

- Pipeline 정상 확인

 ethereum1

- 서버스 Start(/w docker-compose)

- docker-compose.yml

```

version: '3.7'

services:
ethertest.node1 :
  container_name: ethertest.node1
  hostname: ether-node1
  command: "geth --http --port 30305 --http.api eth,net --http.addr
0.0.0.0 --http.port 8545 --http.vhosts=* --syncmode fast --datadir
/DATA_STORAGE --verbosity 3 --nodiscover --networkid 4444"
  working_dir: /DATA_STORAGE

```

```

image: test:5000/infra/ehereumtest:latest
tty : true
ports:
- 8545:8545
- 30305:30305
environment:
ENV: ETHERNODE1
RPCPORT: 8545
PORT: 30305

```

- docker-compose up

```

# docker-compose up -d
Creating network "ether_default" with the default driver
Pulling ethertest.node1 (test:5000/infra/ehereumtest:latest)...
latest: Pulling from infra/ehereumtest
35807b77a593: Already exists
ef643d44c2ee: Pull complete
7139af26c119: Downloading [=====>
] 91.91MB/211.7MB
7137cd8e245b: Downloading [=====>
] 87.05MB/191.5MB
c226085421cb: Download complete
f12e551582a5: Downloading [=====>
] 71.37MB/100.5MB
b3ba493cdcd3: Waiting
c5bea69d9725: Waiting
e760116f864e: Waiting
4c75ff8e161e: Waiting

```

- docker log 확인

```

# docker-compose logs
ethertest.node1 | INFO [09-17|17:11:50.977] New local node record
seq=1 id=f6af76c5a9466c74 ip=127.0.0.1 udp=0 tcp=30305
ethertest.node1 | INFO [09-17|17:11:50.977] Started P2P networking
self="enode://22e38f891e66fc616106479620cb979e3a1cd2a228f5d63ccaf538e8db991c
78feb75a3b73e4f5cb87784f7ee06471c00c5a46076d4fb5092f07016aa8ff004b@127.0.0.1
:30305?discport=0"
ethertest.node1 | INFO [09-17|17:11:50.978] IPC endpoint opened
url=/DATA_STORAGE/geth.ipc
ethertest.node1 | INFO [09-17|17:11:50.979] HTTP server started
endpoint=[:]:8545 cors= vhosts=*

```

Docker Monitoring

- geth 기본 옵션인 Influx/Grafana 를 활용한 모니터링

- InfluxDB 외 옵션

```
Prometheus ((pull model)
InfluxDB (push model)
telegraf
grafana
datadog
chronograf
```

- InfluxDB/Grafana Start(/w docker-compose)

```
version: '2'
services:
  influxdb:
    image: "influxdb:1.7"
    volumes:
      - ./influxdb:/var/lib/influxdb
    ports:
      - "8086:8086"
      - "8083:8083"

  grafana:
    image: "grafana/grafana:latest"
    volumes:
      - ./grafana:/var/lib/grafana
    ports:
      - "3000:3000"
    links:
      - influxdb:database
```

- Influxdb database Create


```
# docker exec -it grafana_influx_influxdb_1 influx
Connected to http://localhost:8086 version 1.7.11
InfluxDB shell version: 1.7.11


> CREATE DATABASE geth
```


- Command Line 에 InfluxDB 관련 추가

```
# --metrics --metrics.influxdb --metrics.influxdb.endpoint
http://localhost:8086 --metrics.influxdb.tags host=ether-node1
```

- Grafana 대시보드 구성

ethereum2

ethereum3

ethereum4