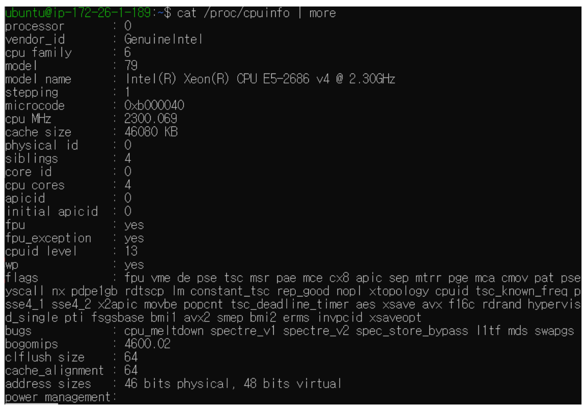
⚙️

포팅메뉴얼(ELK 서버)

🛠️개발환경

서버 인스턴스 사양

- CPU 정보: Intel Xeon(R) Core 4개

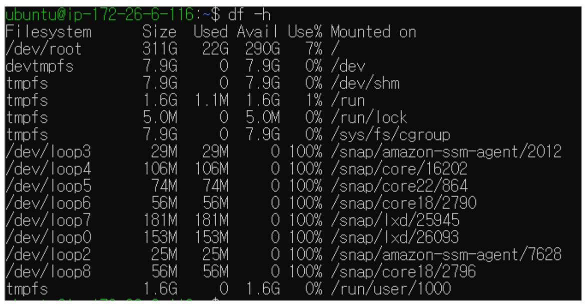


```
ubuntu@ip-172-26-1-189:~$ cat /proc/cpuinfo | more
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 79
model name      : Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
stepping        : 1
microcode       : 0xb000040
cpu MHz         : 2300.069
cache size      : 46080 KB
physical id     : 0
siblings        : 4
core id         : 0
cpu cores       : 4
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse
yscall nx pdpe1gb rdtscp lm constant_tsc rep_good nopl xtopology cpuid tsc_known_freq p
sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervis
d_single pti fsgsbase bmi1 avx2 smep bmi2 erms invpcid xsaveopt
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs
bogomips        : 4600.02
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:
```

- RAM : 16GB

```
ubuntu@ip-172-26-1-189:~$ cat /proc/meminfo
MemTotal:       16370596 kB
```

- Disk : 300GB



```
ubuntu@ip-172-26-6-116:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       311G   22G  290G   7% /
devtmpfs        7.9G     0  7.9G   0% /dev
tmpfs           7.9G     0  7.9G   0% /dev/shm
tmpfs           1.6G  1.1M  1.6G   1% /run
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           7.9G     0  7.9G   0% /sys/fs/cgroup
/dev/loop3       29M   29M     0 100% /snap/amazon-ssm-agent/2012
/dev/loop4      106M  106M     0 100% /snap/core/16202
/dev/loop5       74M   74M     0 100% /snap/core22/864
/dev/loop6       56M   56M     0 100% /snap/core18/2790
/dev/loop7      181M  181M     0 100% /snap/lxd/25945
/dev/loop0      153M  153M     0 100% /snap/lxd/26093
/dev/loop2       25M   25M     0 100% /snap/amazon-ssm-agent/7628
/dev/loop8       56M   56M     0 100% /snap/core18/2796
tmpfs           1.6G     0  1.6G   0% /run/user/1000
```

- 현재 메모리 사용량

```
ubuntu@ip-172-26-6-116:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:           15Gi        10Gi       3.6Gi       0.0Ki       1.2Gi       4.5Gi
Swap:         7.0Gi       4.0Mi       7.0Gi
```

ELK

- elastic search: 7.16.2
- logstash : 7.16.2
- kibana : 7.16.2

Infra

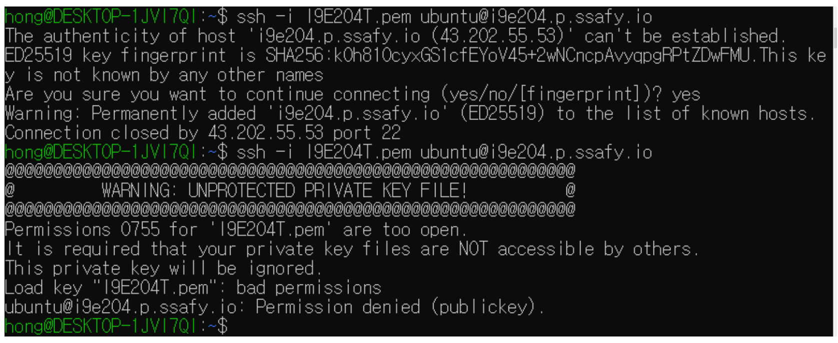
- Docker : 24.0.5

🔨배포 과정

📂빌드 환경설정

1. SSAFY EC2 접속(WSL 이용)

- 바탕화면에 있던 펨키를 우분투 폴더로 복사 붙여넣기
  - cp /mnt/c/Users/SSAFY/Desktop/K9E104T.pem ~/
- ssh 접속
  - cd ~
  - 키 권한 변경



```
hong@DESKTOP-1JVI7QI:~$ ssh -i I9E204T.pem ubuntu@i9e204.p.ssafy.io
The authenticity of host 'i9e204.p.ssafy.io (43.202.55.53)' can't be established.
ED25519 key fingerprint is SHA256:k0h810cyxGS1cfEYoV45+2wNCncpAvyqpgRPtZDwFMU.This ke
y is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'i9e204.p.ssafy.io' (ED25519) to the list of known hosts.
Connection closed by 43.202.55.53 port 22
hong@DESKTOP-1JVI7QI:~$ ssh -i I9E204T.pem ubuntu@i9e204.p.ssafy.io
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0755 for 'I9E204T.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "I9E204T.pem": bad permissions
ubuntu@i9e204.p.ssafy.io: Permission denied (publickey).
hong@DESKTOP-1JVI7QI:~$
```

        - 권한 에러
        - 개인키는 권한이 너무 open 되어있어도 경고 문구가 뜨면서 permission deny가 뜨기 때문에 권한을 축소시켜서 (chmod 400) 사용하도록 한다.
        - $ chmod 400 pem키
        - 그래도 안된다면???
            - https://velog.io/@wingnawing/작고-귀여운-AWS-.pem-권한-에러-WARNING-UNPROTECTED-PRIVATE-KEY-FILE
            - 우분투 환경으로 가서 진행해야함!!! cd ~로 가서 pem키 권한을 바꿔주자
    - ssh -i K9E104T.pem ubuntu@k9e104.p.ssafy.io
    - ssh 접속 쉽게 하는법(wsl ubuntu에서 진행)

```
mkdir ~/.ssh
cd ~/.ssh // ssh 폴더 생성 및 이동
cp [로컬 pem 키 위치] ~/.ssh // pem 키 옮기기
vi config  // config 파일 생성
```

    - config

```
Host ssafy
HostName k9e104.p.ssafy.io
User ubuntu
IdentityFile ~/.ssh/K9E104T.pem
```

    - 이후부터 ssh ssafy로 접속

- 방화벽 포트 확인 : sudo ufw status verbose
- 퍼플릭 ip 주소 확인 : curl ifconfig.me

2. EC2 초기 설정

```
$ sudo apt update
$ sudo apt upgrade
$ sudo apt install build-essential

$ sudo ln -sf /usr/share/zoneinfo/Asia/Seoul /etc/localtime # 한국으로 시간 설정
```

(필수)Swap 설정

- https://sundries-in-myidea.tistory.com/102
- https://wooogy-egg.tistory.com/83
- 이거 안해주면 나중에 메모리 초과돼서 서버 안들어가짐 ㅜㅜ

3. 서버에 도커 설치

```
$ sudo apt update
$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
$ sudo wget -qO- https://get.docker.com/ | sh
```

```
$ sudo usermod -aG docker ${USER}
$ sudo systemctl restart docker
```

- sudo를 사용하지 않고 docker를 사용할 수 있다.
- docker 그룹은 root 권한과 동일하므로 꼭 필요한 계정만 포함
- **현재 계정에서 로그아웃한 뒤 다시 로그인!!!**

4. elastic search 설치

- docker-compose 설치

```
// 최신 docker compose를 해당 링크에서 받을 수 있음
sudo curl -L https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
// 권한 부여
sudo chmod +x /usr/local/bin/docker-compose
// 설치 확인
docker-compose version
```

- elastic search와 kibana 설치
    - https://backtony.github.io/spring/elk/2022-03-02-spring-elasticsearch-2/
    - 도커파일 생성(nori 형태소 추가하기 위해)

```
# dockerfile 생성
vi Dockerfile

# 작성
ARG ELK_VERSION
FROM docker.elastic.co/elasticsearch/elasticsearch:${ELK_VERSION}
RUN elasticsearch-plugin install analysis-nori
RUN elasticsearch-plugin install --batch https://github.com/netcrazy/elasticsearch-jaso-analyzer/releases/download/v7.16.2/jaso-analyzer-plugin-7.16.2-plugin.zip
```

    - compose 파일(`es.yml`) → 버전은 springboot 2.7버전에 맞는 7.17.3 사용

```
version: "3.7"
services:
  es:
    build:
      context: /home/ubuntu
      args:
        ELK_VERSION: 7.16.2
    container_name: es
    volumes:
           - elasticsearch-volume:/usr/share/elasticsearch/data
    environment:
      - node.name=single-node
      - cluster.name=backtony
      - discovery.type=single-node
    ports:
      - 9200:9200
      - 9300:9300
    networks:
      - es-bridge

  kibana:
    container_name: kibana
    image: docker.elastic.co/kibana/kibana:7.16.2
    environment:
      SERVER_NAME: kibana
      ELASTICSEARCH_HOSTS: http://es:9200
    ports:
      - 5601:5601
    depends_on:
      - es
    networks:
      - es-bridge

  logstash:
    image: docker.elastic.co/logstash/logstash:7.16.2
    volumes:
      - ./logstash/logstash.conf/:/usr/share/logstash/pipeline/logstash.conf
    ports:
      - 5000:5000
    environment:
      - xpack.monitoring.enabled=true
      - xpack.monitoring.elasticsearch.hosts=http://es:9200
    depends_on:
      - es
    networks:
      - es-bridge

networks:
  es-bridge:
    driver: bridge

volumes:
       elasticsearch-volume:
```

- docker compose 실행

```
# 실행, 데몬으로 띄우려면 맨 뒤에 -d를 붙여준다.
# 기본 실행 도커파일은 docker-compose.yml인데 es.yml로 만들었으므로 지정해주기 위해서 -f 옵션을 사용
docker-compose -f es.yml up -d

# 죽이기
docker-compose -f es.yml down
```

4-1. 보안설정

```
# read_me_to_restore_data 인덱스
We delete all databases, but download a copy to our server. The only way of recovery is you must send 0.01 BTC to bc1qugzxdcudxm9uaafjadmhe8pllr36s5z3jqywu5. You have until 48 hours to pay or data will be inac
```

- 갑자기 모든 인덱스가 사라지더니 read_me_to_restore_data라는 인덱스가 생겨서 봤더니 위와 같은 메세지가 있었음… 해킹당한것!!!

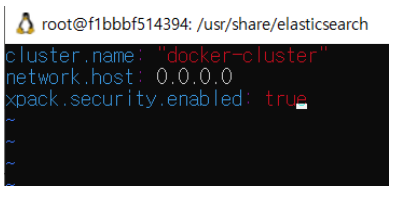
  ⇒ 아이디랑 비번 설정해줘야함… elastic search에서 진행…

- 아이디와 비번 설정
  - https://javas.tistory.com/11
  - xpack.security.transport.ssl.enabled: true 이거는 하면 안됨!!!
  - 하지만 elasticsearch.yml파일은 config/elasticsearch.yml에 위치함
  - 마찬가지로 kibana.yml파일도 config/kibana.yml에 위치함
- elastic search 설정파일 바꾸는 법

```
docker exec -u 0 -it es /bin/bash

# vi 다운로드
apt-get update
apt-get upgrade
apt-get install vim

cd config/
vi elasticsearch.yml
```

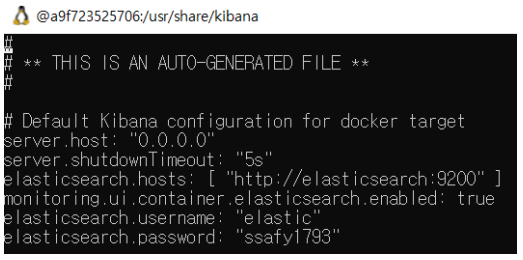


- kibana의 설정파일 확인

```
docker exec -u 0 -it kibana /bin/bash

# vi 다운로드
apt-get update
apt-get upgrade
apt-get install vim
```

```
cd config/
vi kibana.yml
```



- http://k9e104a.p.ssafy.io:5601/ 으로 kibana 접속 확인
  - **Kibana server is not ready yet** 라는 에러가 뜨면 설정이 뭔가 잘못된 것임
    - docker logs <kibana CONTAINER ID>
    - 위의 명령어로 뭐가 잘못됐는지 확인하자

[번외]

- https://velog.io/@eeheaven/ElasticSearch-한글-검색-자동완성
  - korean jaso 분석기를 사용하기 위해 elastic version을 7.16.2로 설정함!!
- 추가로 플러그인(분석기)를 설치하고 싶을 때

```
# elastic search 컨테이너 내부로 들어가자
$ docker exec -u 0 -it es /bin/bash
$ bin/elasticsearch-plugin install analysis-nori
$ bin/elasticsearch-plugin install --batch https://github.com/netcrazy/elasticsearch-jaso-analyzer/releases/download/v7.16.2/jaso-analyzer-plugin-7.16.2-plugin.zip
```

4-2. logstash 설정

- 1분마다 mysql에 접근해서 수정된 데이터 가져와서 인덱스 수정
- springboot의 로그 가져와서 인덱스 생성

```
mkdir logstash
cd logstash
sudo vi logstash.conf
```

```
# /logstash/logstash.conf
input {
tcp {
                port => 5000
                codec => json_lines
         add_field => { "source" => "tcp" } # source 필드 추가
        }
jdbc{
        jdbc_validate_connection => true
        jdbc_driver_class => "com.mysql.jdbc.Driver"
        jdbc_driver_library => "/usr/share/java/mysql-connector-j-8.0.33.jar"
        jdbc_connection_string => "jdbc:mysql://cozytrain.ckacazfcovre.ap-northeast-2.rds.amazonaws.com:3306/test_train?characterEncoding=utf8"
        jdbc_user => "admin"
        jdbc_password => "ssafy1793"
        jdbc_paging_enabled => true
    jdbc_page_size => 50
        schedule => "* * * * *"
            statement => "SELECT * FROM member WHERE member_id > :sql_last_value ORDER BY member_id ASC"

         record_last_run => true
    clean_run => true
    tracking_column_type => "numeric"
    tracking_column => "member_id"
    use_column_value => true
   last_run_metadata_path => "/etc/logstash/data/member"
         add_field => { "source" => "jdbc" } # source 필드 추가
    }
}


output {
  if [source] == "tcp" {
    elasticsearch {
      hosts => ["http://es:9200"]
      index => "logstash-%{+YYYY.MM.dd}"
         user => "elastic"
    password => "ssafy1793"
    }
  } else if [source] == "jdbc" {
            elasticsearch {
      hosts => ["http://es:9200"]
      index => "logstash-member"
         user => "elastic"
    password => "ssafy1793"
        manage_template => true
        template => "/etc/logstash/data/template.json"
        template_name => "logstash-member"
        template_overwrite => true
    }
stdout {
         }
        }
}
```

1. RDS의 인바운드규칙에서 elk서버 열어주기
2. logstash 컨테이너로 들어가기 & 마지막 접근 위치 저장할 폴더 만들기

```
docker exec -u 0 -it {logstash 컨테이너 이름} /bin/bash
wget https://repo1.maven.org/maven2/com/mysql/mysql-connector-j/8.0.33/mysql-connector-j-8.0.33.jar
```

```
mkdir /etc/logstash/data
chown -R logstash:logstash /etc/logstash/data
```

mysql의 값이 중복으로 elastic search에 들어가는 문제점

https://m.blog.naver.com/koys007/221883846169

▼ (수정전) logstash.conf

```
input {
  tcp {
        port => 5000
        codec => json_lines
        add_field => { "source" => "tcp" } # source 필드 추가
  }
  jdbc{
         jdbc_validate_connection => true
         jdbc_driver_class => "com.mysql.jdbc.Driver"
         jdbc_driver_library => "/usr/share/java/mysql-connector-j-8.0.33.jar"
         jdbc_connection_string => "jdbc:mysql://cozytrain.ckacazfcovre.ap-northeast-2.rds.amazonaws.com:3306/test_train?characterEncoding=utf8"
         jdbc_user => "admin"
         jdbc_password => "ssafy1793"
         jdbc_paging_enabled => true
         jdbc_page_size => 50
         schedule => "* * * * *"
         statement => "SELECT * FROM member ORDER BY member_id"
         add_field => { "source" => "jdbc" } # source 필드 추가
     }
}


output {
  if [source] == "tcp" {
    elasticsearch {
      hosts => ["http://es:9200"]
      index => "logstash-%{+YYYY.MM.dd}"
    }
  } else if [source] == "jdbc" {
           elasticsearch {
      hosts => ["http://es:9200"]
      index => "logstash-member-%{+YYYY.MM.dd}"
    }

        stdout {
         }
        }
}
```

- 단순히 member의 모든 필드값을 조회해서 logstash-member에 저장함

  ⇒ 계속 똑같은 값들을 조회하고 별도의 처리 없이 기존의 index에 추가됨

▼ (수정후) logstash.conf

- 위와 같음

```
statement => "SELECT * FROM member WHERE member_id > :sql_last_value ORDER BY member_id ASC"
          record_last_run => true
          #tracking_column 값 초기화, 같은 로그로 반복 테스트할 때 필수
          clean_run => true
          tracking_column_type => "numeric"
          tracking_column => "member_id"
          use_column_value => true
          last_run_metadata_path => "/etc/logstash/data/member"
```

- tracking_column : 로그 중복 전송을 막기 위한 기준 필드로 timestamp가 아닌 필드 사용

  ⇒ 여기서는 member_id가 pk값이고 오름차순으로 올라가는 데이터이므로 선택

- last_run_metadata_path :
- sql_last_value는 가장 마지막에 저장된 member_id값을 가짐

  ⇒ 이 값보다 큰 값이 들어오면 데이터가 새로 들어온 것이므로 해당 데이터만 index에 추가

실행중인 도커 목록

```
ubuntu@ip-172-26-6-116:~$ docker ps -a
CONTAINER ID   IMAGE                                        COMMAND                  CREATED       STATUS      PORTS
          NAMES
a9f723525706   docker.elastic.co/kibana/kibana:7.16.2        "/bin/tini -- /usr/l…"   11 days ago   Up 5 days   0.0.0.0:5601->5601/tcp, :::5601->5601/tcp
          kibana
d5a00d849257   docker.elastic.co/logstash/logstash:7.16.2    "/usr/local/bin/dock…"   11 days ago   Up 5 days   5044/tcp, 0.0.0.0:5000->5000/tcp, :::5000->5000/tcp, 9600/tcp
          ubuntu-logstash-1
f1bbbf514394   ubuntu-es                                    "/bin/tini -- /usr/l…"   11 days ago   Up 5 days   0.0.0.0:9200->9200/tcp, :::9200->9200/tcp, 0.0.0.0:9300->9300/tcp, :::9300->9
300/tcp   es
```

방화벽 포트 목록

```
22 (v6)                    ALLOW IN    Anywhere (v6)
5601 (v6)                  ALLOW IN    Anywhere (v6)
```

UFW(방화벽) 포트 허용 설정

- **ufw allow [port] [protocol]**
  - ufw allow 5601