

개발환경 설정

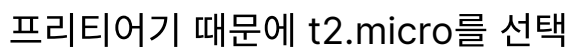
EC2 인스턴스 대여하기

1. 빌릴 인스턴스의 위치를 서비스할 지역 근처로 설정 후 대여할 인스턴스 운영체제 선택

The screenshot shows the AWS Management Console interface for the EC2 Instance Wizard. The '1. AMI 선택' (Select AMI) step is active. The 'AWS Marketplace' tab is selected, and the 'Ubuntu Server 18.04 LTS (HVM), SSD Volume Type' AMI is chosen. The region is set to 'us-east-1' (US East (N. Virginia)). The instance type is 't2.micro'.

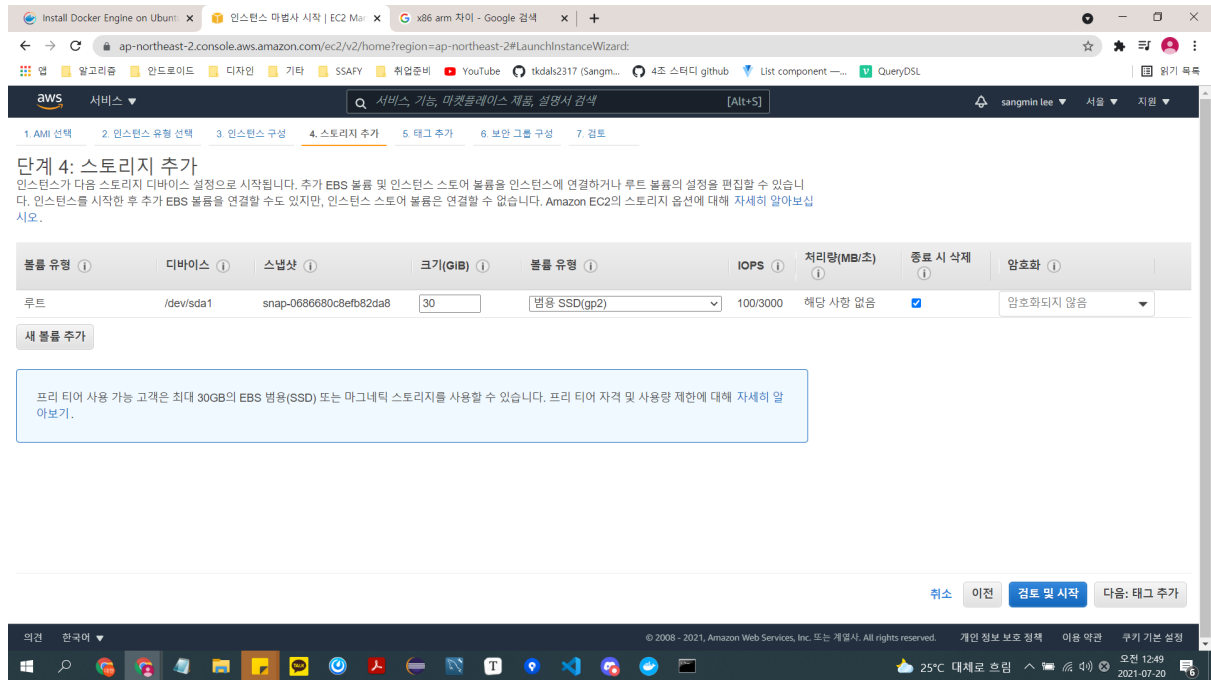
Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0ba5cd124d7a79612 (64비트 x86) / ami-08b051fc14e6c551e (64 비트 Arm)

1. 빌릴 컴퓨터의 사양 설정



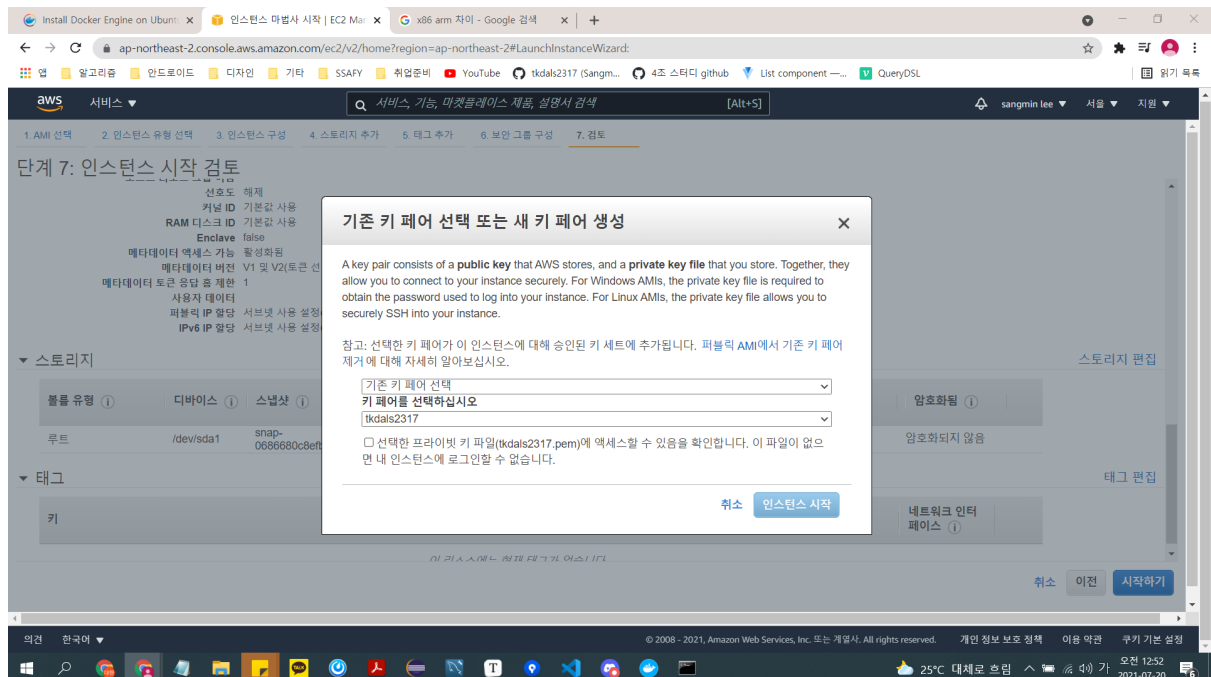
빌릴 인스턴스(컴퓨터)의 개수 1
그 외는 default로 설정

1. 스토리지 저장 용량 설정

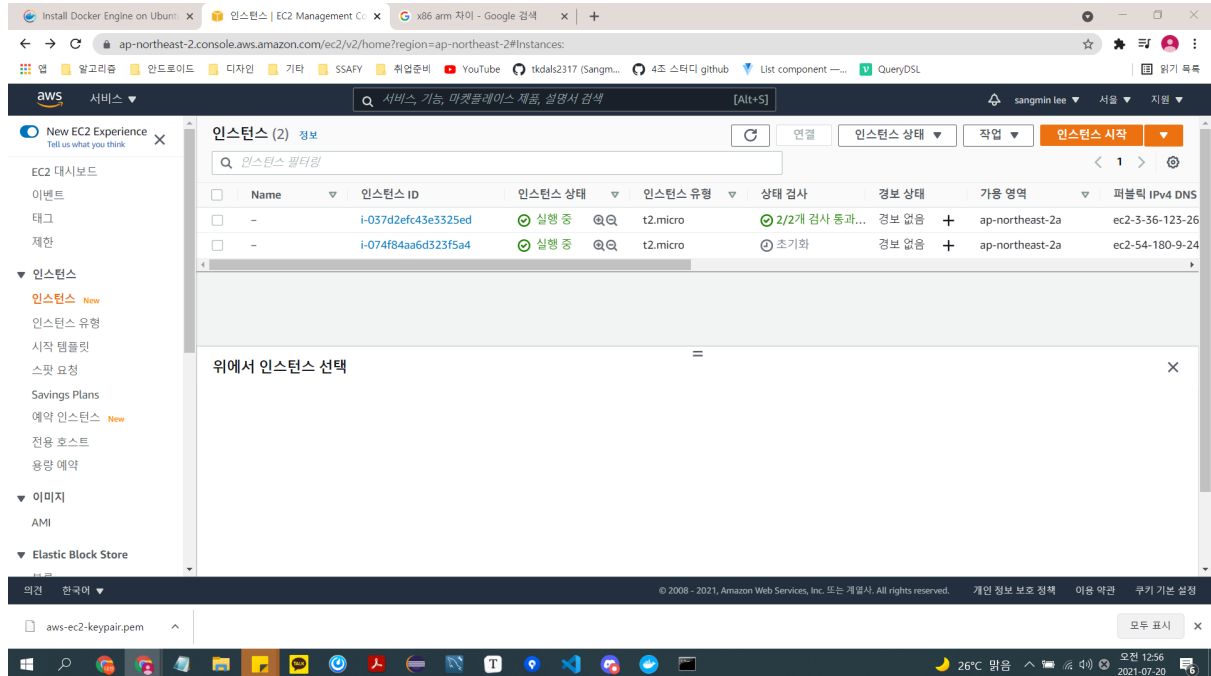


저장용량 30GiB로 설정 그외 default 값으로 설정

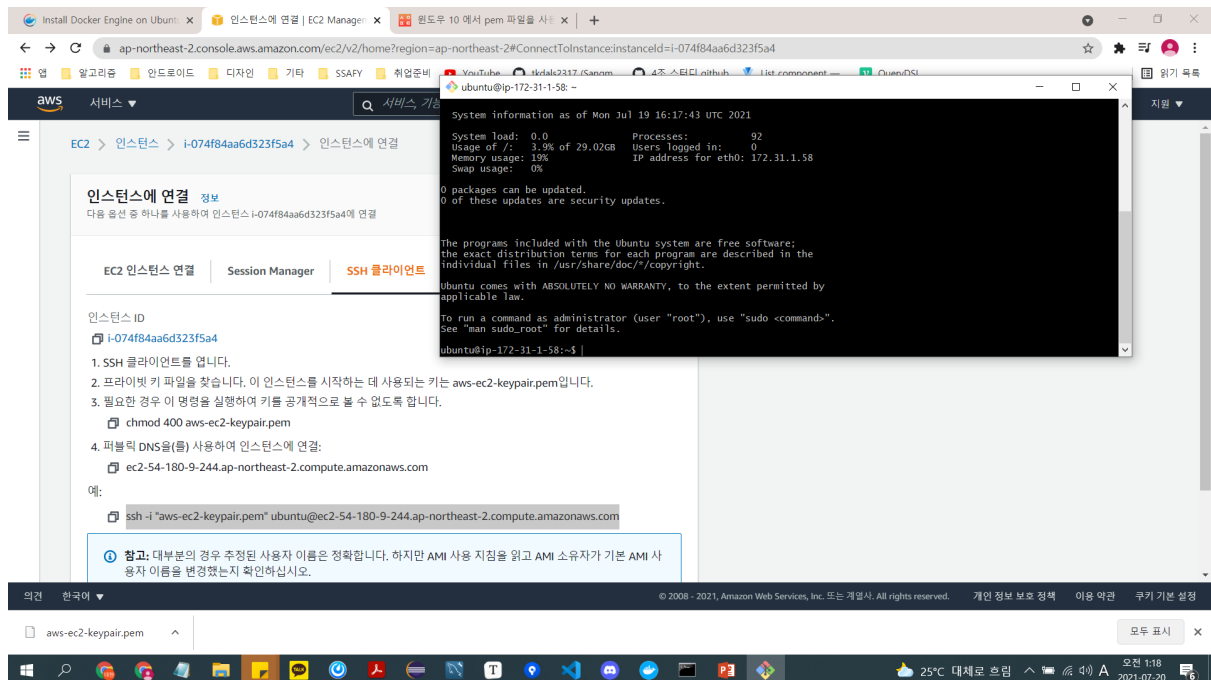
1. 태그 추가 및 보안 그룹 추후 설정
2. 인스턴스 시작 검토 후 시작하기



키 페어는 파일로 된 비밀번호라고 생각하면 된다
 잃어버릴 시 절대 인스턴스에 접근 하지 못하므로 관리를 잘해야한다
 pem 파일을 저장하고 인스턴스를 시작하면 된다



1. git bash를 통해 실행



pem파일 권한 설정

```
> chmod 400 aws-ec2-keypair.pem
```

실행

```
> ssh -i "aws-ec2-keypair.pem" ubuntu@ec2-54-180-9-244.ap-northeast-2.compute.amazonaws.com
```

Docker

- 서비스의 배포와 Kurento 미디어 서버 실행을 위해 사용
- 로컬 개발 환경 설치 : <https://www.docker.com/get-started>
- 배포환경(AWS EC2) 설치 : <https://docs.docker.com/engine/install/ubuntu/>

```
# Ubuntu 환경에 접속하여 Docker와 Docker-Compose 설치하기
# 1. HTTPS를 통해 리포지토리를 사용할 apt수 있도록 패키지 인덱스를 업데이트하고 패키지를 설치합니다
sudo apt-get update
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
# 2. Docker와 암호화된 통신을 하기 위해 도커 공식 GPG 키 추가
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/
keyrings/docker-archive-keyring.gpg

# 3. apt로 도커를 설치하기 위해서는 도커 stable 레파지토리를 추가해주어야 함(amd64 : Version 확인)
echo \
    "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://downlo
ad.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# 4. 패키지 목록 업데이트 후 목록에서 도커 설치 패키지가 존재하는 지 확인
sudo apt-get update
sudo apt list | grep docker-ce
sudo apt list | grep containerd.io
sudo apt list | grep containerd.io

# 5. apt install 명령으로 docker-ce, docker-ce-cli, containerd.io 설치
sudo apt-get install docker-ce docker-ce-cli containerd.io

# 6. 깃허브 주소로 docker compose 설치
sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(u
name -s)-$(uname -m)" -o /usr/local/bin/docker-compose

# 7. 권한 설정
sudo chmod +x /usr/local/bin/docker-compose
```

```
# 8. 설치 확인
docker-compose --version
```



Docker Compose는 다중 컨테이너 애플리케이션을 정의하고 공유할 수 있도록 개발된 도구입니다. Compose에서 서비스를 정의하는 YAML 파일을 만들고, 단일 명령을 사용하여 모두 실행하거나 모두 종료

Compose 파일을 사용하여 애플리케이션을 로컬로 배포 하거나 Docker CLI를 사용하여 Amazon ECS 또는 Microsoft ACI 의 클라우드에 배포할 수 있습니다 .

Kurento(미디어 서버)

- WebRTC를 구현하기 위해 사용
- 도커 환경에서 Kurento 미디어 서버 Docker 이미지를 실행
- 참고자료 : <https://doc-kurento.readthedocs.io/en/latest/user/installation.html#installation-guide>

```
# Kurento 실행
# 1. 도커에서 Kurento Media Server 이미지 실행(미디어 서버의 기본 포트는 8888)
sudo docker pull kurento/kurento-media-server:latest

# 2. 도커 실행
sudo docker run -d --name kms --network host \
    kurento/kurento-media-server:latest
```

STUN/TURN 서버 설치

Coturn을 WebRTC 시그널링(Signaling)을 위한 STUN/TURN 서버로 활용

STUN/TURN 서버는 로컬 환경에서 동작하지 않으며, Public으로 동작 가능한 AWS EC2 환경에 설치되어야 한다

참고자료 : <https://doc-kurento.readthedocs.io/en/latest/user/faq.html#faq-coturn-install>

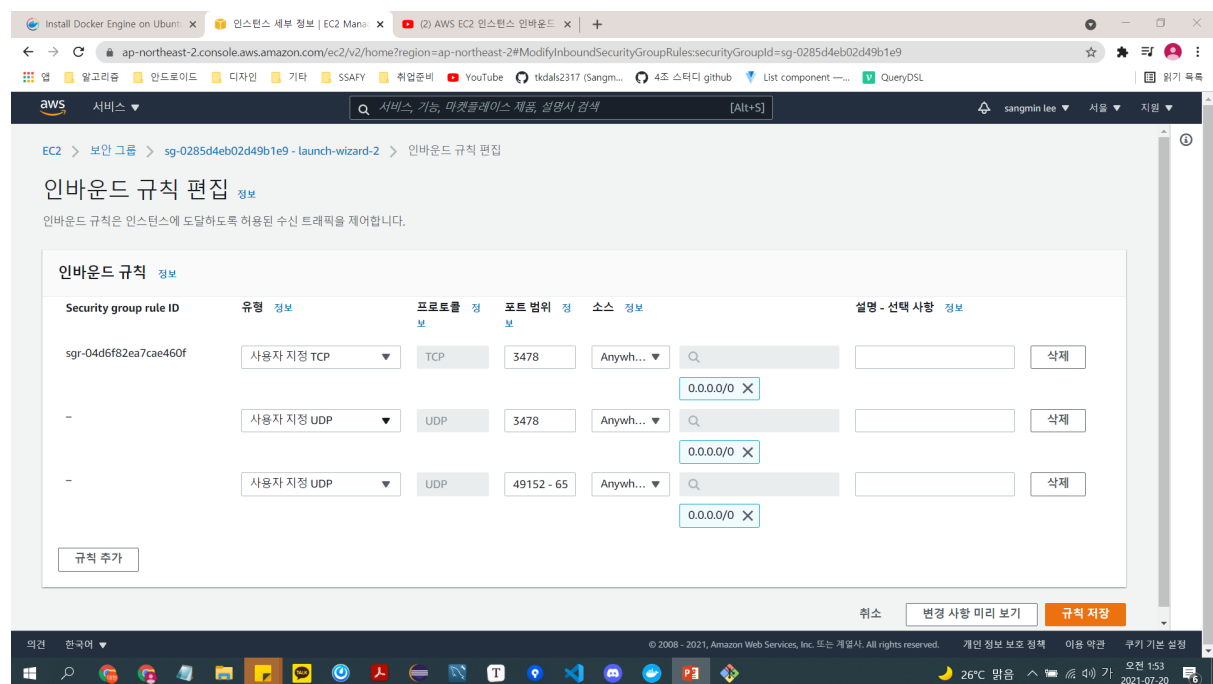
EC2 보안 그룹 설정 및 IP 주소 확인

AWS에서 EC2 → 보안 그룹에 STUN/TURN 서버로 활용할 EC2의 보안 그룹을 설정

인바운드 규칙을 아래와 같이 프로토콜 및 포트 번호를 허용하도록 추가하고 EC2 퍼블릭 IPv4 주소와 프라이빗 IPv4 주소를 미리 확인

```
3478 : UDP
3478 : TCP
49152-65535 :UDP
```

인바운드 규칙



ip주소 확인

▼ 네트워킹 세부 정보 정보

퍼블릭 IPv4 주소

54.180.9.244 | [개방 주소법](#)

퍼블릭 IPv4 DNS

ec2-54-180-9-244.ap-northeast-2.compute.amazonaws.com | [개방 주소법](#)

프라이빗 IPv4 주소

172.31.1.58

프라이빗 IPv4 DNS

ip-172-31-1-58.ap-northeast-2.compute.internal

VPC ID

vpc-fd41d096

서브넷 ID

subnet-c726a5ac

퍼블릭 IPv4 주소 : 54.180.9.244

프라이빗 IPv4 주소 : 172.31.1.58

Coturn 설치

```
# 도커 환경에서 설치
sudo apt-get update && sudo apt-get install --no-install-recommends --yes \
coturn
```

Coturn 설정

Ubuntu의 cddefault/coturn 파일을 아래와 같이 수정

```
TURN_SERVER_ENABLED=1
```

Ubuntu의 /etc/turnserver.conf 파일을 아래와 같이 수정

```
listening-port=3478
tls-listening-port=5349
listening-ip=<EC2의 프라이빗 IPv4 주소>
external-ip=<EC2의 퍼블릭 IPv4 주소>/<EC2의 프라이빗 IPv4 주소>
relay-ip=<EC2의 프라이빗 IPv4 주소>
fingerprint
lt-cred-mech
user=myuser:mypassword
realm=myrealm
log-file=/var/log/turn.log
simple-log
```

Coturn 재기동

Coturn 서버를 재기동하여 변경된 설정을 반영

```
sudo service coturn restart
```

STUN/TURN 서버 설정

Kurento 미디어 서버가 STUN/TURN 서버와 함께 동작하기 위해 STUN/TURN 서버의 위치를 알려주는 설정 작업을 수행해야 함

참고자료 : <https://doc->

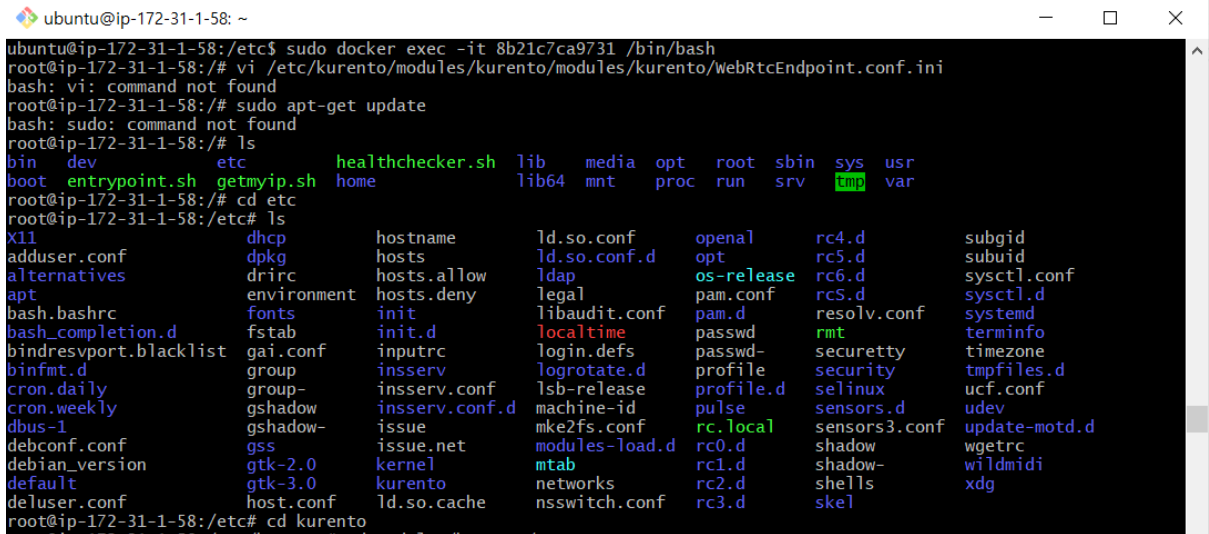
kurento.readthedocs.io/en/latest/user/configuration.html#stun-turn-server

Kurento Docker 컨테이너 접속

Kurento 미디어 서버가 기동되는 도커 컨테이너에 접속

```
# 실행중인 컨테이너 ID를 확인합니다 <CONTAINER ID>
docker ps -a

#컨테이너 터미널에 접속
docker exec -it <CONTAINER ID> /bin/bash
```



Kurento WebRtcEndpoint.ini 파일 수정

명령 프롬프트나 터미널에서 아래 명령어를 통해 WebRtcEndpoint.ini 파일을 수정

```
# vi로 파일 편집
vi /etc/kurento/modules/kurento/modules/kurento/WebRtcEndpoint.conf.ini
# 수정 내용
stunServerAddress=<EC2의 퍼블릭 IPv4 주소>
stunServerPort=3478
turnURL=myuser:mypassword@<EC2의 퍼블릭 IPv4 주소>?transport=udp
```

```
root@ip-172-31-1-58: /
;; using TURN is the remaining alternative.
;;
;; You don't need to configure both STUN and TURN, because TURN already includes
;; STUN functionality.
;;
;; The provided URL should follow one of these formats:
;;
;; * user:password@ipaddress:port
;; * user:password@ipaddress:port?transport=[udp|tcp|tls]
;;
;; <ipaddress> MUST be an IP address; domain names are NOT supported.
;; <transport> is OPTIONAL. Possible values: udp, tcp, tls. Default: udp.
;;
;; You need to use a well-working TURN server. Use this to check if it works:
;; https://webrtc.github.io/samples/src/content/peerconnection/trickle-ice/
;;
;; From that check, you should get at least one Server-Reflexive Candidate
;; (type "srflx") AND one Relay Candidate (type "relay").
;;
turnURL=myuser:mypassword@54.180.9.244?transport=udp
;; Certificate used for DTLS authentication.
```

Docker MySQL 컨테이너 생성

```
# 1.mysql 이미지 가져오기(버전 확인)
# 최신 버전 : sudo docker pull mysql
sudo docker pull mysql:5.7.33

# 2.도커 내려받은 이미지 확인
sudo docker images

# 3. mysql 컨테이너 실행
sudo docker run --name mysqlcontainer -e MYSQL_ROOT_PASSWORD=root -d -p 3306:3306 mysql:5.7.33
## -name : 컨테이너 이름 지정
## -e : mysql password 설정
## -d : daemon(백그라운드 실행)
## -p : 포트번호 설정

# 4. 현재 동작중인 컨테이너 확인
sudo docker ps

# 5. 컨테이너 접속
sudo docker exec -it mysqlcontainer bash
## -it : 컨테이너를 종료하지 않은채로, 터미널의 입력을 계속해서 컨테이너로 전달하기 위해서 사용
## mysql 환경으로 변경되어야 함

# 6. mysql 아이디 비밀번호 입력
mysql -u root -p

# 7. 확인
show databases;
```

ubuntu@ip-172-31-1-58: ~

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases

-> ;

Database
information_schema
mysql
performance_schema
sys

4 rows in set (0.00 sec)

앞으로 해야할 일 DB 이관, 설정 파일 변경, 데이터베이스 공유 방법