K-Digital Training

| 결재 | 담당 | 원장 |
|---|---|---|
| | | |

클라우드데브옵스(DevOps) 엔지니어및관리자 양성과정(8기)

# 3차 프로젝트 완료 보고서

**- VPN을 활용한 하이브리드 클라우드 구축 -**

2023.10.04

**구성원 : 김경태, 김학남, 서희경**

고용노동부 한국정보교육원 구.경원직업전문학교

| 프로젝트 완료 보고서 | | |
|---|---|---|
| 프로젝트 주제 | VPN을 활용한 하이브리드 클라우드 구축 | |
| 단계 : 프로젝트 완료 | 작성자 : 김경태 | 작성일 : 2023.10.04 |

문서 개정 이력

| 개정번호 | 개정일자 | 시행일자 | 개정내용 | 담당자 |
|---|---|---|---|---|
| 1.0 | 2023.10.04. | | 최초 작성 | 김경태 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

**교 육 기 관 :** 한 국 정 보 교 육 원

**팀 명 :**

**팀 장 :**

**팀 원 :**

# 목차


1. 프로젝트 개요

- 프로젝트 명 ..... 4
- 프로젝트 기간 ..... 4
- 프로젝트 목표 ..... 4
- 프로젝트 시나리오 ..... 4
- 프로젝트 수행 요건 ..... 4

2. 프로젝트 추진 체계

- 프로젝트 참여 인력 총괄표 ..... 5
- 참여 인력 업무 분장 ..... 5

3. 세부 프로젝트 내용

- 전체 구성도 ..... 6
- 네트워크 구성도 ..... 6
- 서버(물리/가상) 구성 현황 ..... 7
- 네트워크 구성 현황 ..... 9
- 상세 구축 및 구성 내용 ..... 14
- 구축 결과 ..... 20

4. 프로젝트 일정 ..... 24

5. 피드백

- 미달성 목표 ..... 21
- 향후 계획 ..... 21

| 프로젝트 완료 보고서 | | |
|---|---|---|
| 프로젝트 주제 | VPN을 활용한 하이브리드 클라우드 구축 | |
| 단계 : 프로젝트 완료 | 작성자 : 김경태 | 작성일 : 2023.10.04 |

# 1. 프로젝트 개요

- 프로젝트 명

  **VPN**를 활용한 **하이브리드 클라우드** 구축

- 프로젝트 기간

  2023.08.21. ~ 2023.10.4.**(총 45일)**

- 프로젝트 목표

  가) **AWS**를 이용한 클라우드 컴퓨팅 서비스 구축
  나) 가용영역 이중화 및 **Public/Private Subnet** 구분으로 가용성과 보안성 확보
  다) Service **Load Balancing**, **Auto Scaling** 서비스 구축
  라) **Kubernetes**를 활용하여 컨테이너 자동 배포 시스템 구축 -> **PaaS** 제공
  마) 퍼블릭 클라우드와 온프레미스를 **VPN**을 활용하여 **하이브리드 클라우드** 구성

- 프로젝트 시나리오

  가) 대상
  - 00컴퍼니 (가상)

  나) 가정
  - 유튜브 동영상 시청이 가능한 학원 커뮤니티가 필요해짐.

  1. 보안 및 규정 준수 문제:
     고객사는 민감한 고객 데이터를 다루고 있으며, 관련 규정 및 규정 준수 요구 사항을 준수해야 합니다. 클라우드 서비스를 도입하지 않으면 보안 및 규정 준수 요구 사항을 충족하기 어렵습니다.

  2. 확장성 및 리소스 최적화 문제
     수시로 변하는 요구 사항에 대응하기 위한 유연한 IT 인프라가 필요합니다.
     현재의 온프레미스 환경은 확장이 제한되며, 리소스를 최적화할 수 없어서
     비용 효율적인 운영이 어렵습니다.

  3. 재해 복구 및 비상 대응 미비성
     현재의 온프레미스 인프라는 단일 지역에 집중되어 있어 자연 재해 또는 기타 재난 시에 업무 중단의 위험이 큽니다. 클라우드 기술을 활용하여 다중 지역에 데이터 백업 및 복구 옵션을 구축하려고 합니다.

다) 고객사의 요구사항

1. IAM 생성
 - 관리자 권한 IAM 생성
 - 해당 IAM으로 AWS 관리

2. AZ 이중화
- 가용성 확보
- 서브넷 분리 -> 사설 네트워크 보호

3. Bastion Host
- 사설 네트워크의 인스턴스 -> 외부 인터넷 통신

4. Auto Scaling
- 리소스 사용량에 따른 인스턴스, Pod 조절

5. Load Balancer
- 가용성 증가 및 부하 분산
- ALB, NLB
- (K8S) LoadBalancer Service

6. Template
- 시작 템플릿
- 도커 이미지 빌드

- 프로젝트 수행요건

가) 설계 및 개발 요건
- 시스템의 물리적/논리적 Scale out/up에 대비하여 설계되어야 한다.
- WEB, WAS, DB 각각 서비스를 이중화 해 서비스의 가용성 확보를 최우선으로 한다.
- 라우터의 NAT기능을 통해 내/외부 간 통신은 가능케 하되, 직접적인 연결은 차단한다.
- NFS의 운영 ↔ DR 간 데이터 동기화를 통해 DR로 전환 시, 최대한의 가용성을 확보한다.

## 2. 프로젝트 추진 체계

- 프로젝트 참여 인력 총괄표

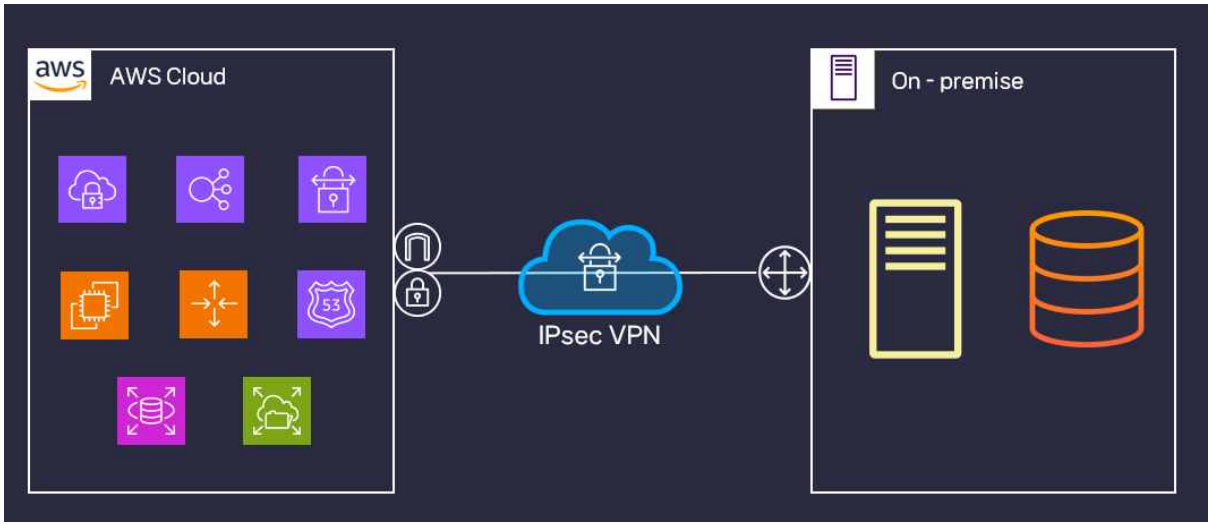
| 성 명 | 소 속 | 역할 | 담 당 업 무 |
|---|---|---|---|
| 김 학 남 | 한국정보교육원 | Project Leader | PM, VPN 구성 |
| 김 경 태 | 한국정보교육원 | Project Assistant | AWS 구축 |
| 사 희 경 | 한국정보교육원 | Project Assistant | On premise 구축 |

- 참여 인력 업무 분장

| 업무명 | 업무내용 |
|---|---|
| PM | - 프로젝트 수행 관리 및 책임<br>- 프로젝트 범위, 인원, 일정, 결과 보고<br>- 프로젝트 진행 상황에 따른 계획 조정<br>- 기타 서류, 보고서 작성 및 발표 |
| AWS 구축 | - 네트워크 토폴로지 구성<br>- IAM 계정 생성<br>- VPC 생성 및 네트워크 구축<br>- 3 Tier Architecture 구축 (EC2 Instance / EFS 설정)<br>- Auto Scaling / Load Balancer 구축 (health check 설정) |
| On premise 구축 | - 네트워크 토폴로지 구성<br>- Kubernetes 설치<br>- 3 Tier Architecture 구축<br>- Deployment / Service를 위한 Yaml 파일 작성 |
| VPN을 통한 연동 | - 네트워크 토폴로지 구성<br>- FRR 설치<br>- ipsec 설치 및 환경설정(libreswan) -> BGP 연결<br>- VPN 구성 및 연결 |

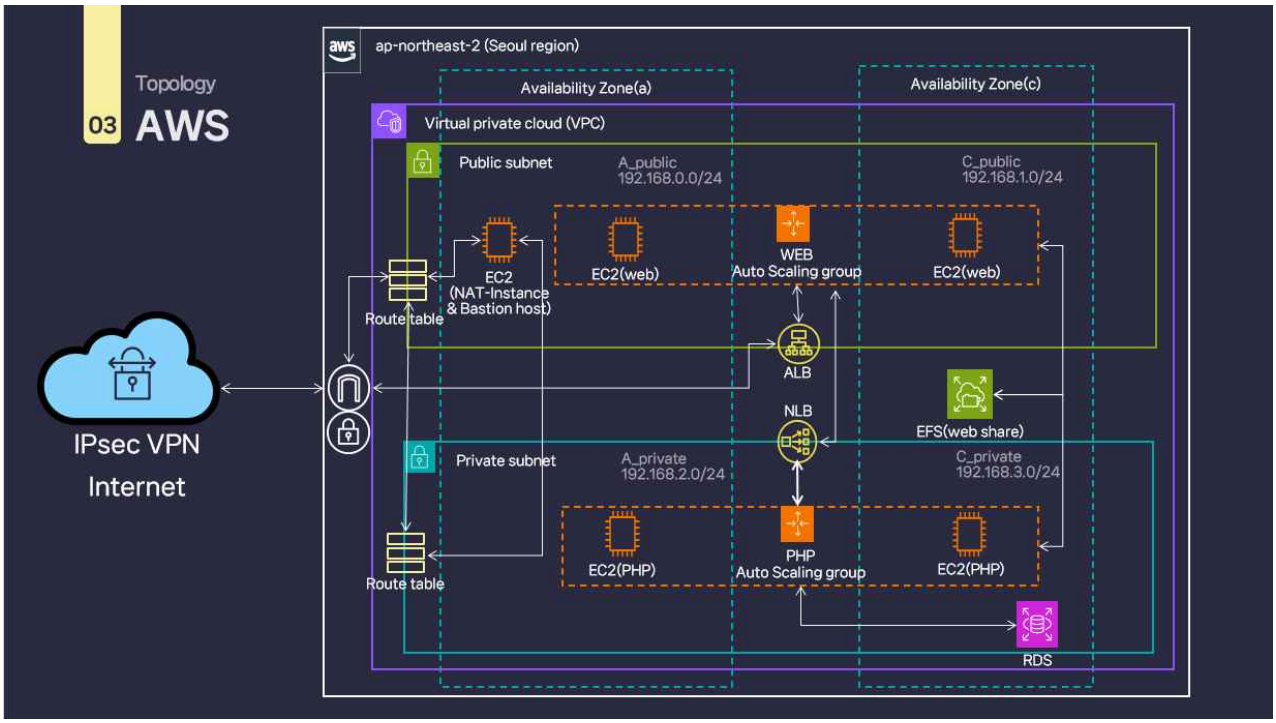
# 3. 세부 프로젝트 내용

- 전체 구성도




가) vlan, inter-vlan설정으로 운영 VMWARE 두 대의 IP를 다르게 구성(상호 통신 가능)
나) Prod#1, #2 중 한 Node에 장애가 발생해도 다른 한 Node에서 처리 가능하도록 구성
다) NAT를 통해 한국정보교육원에서 WEB 접속이 가능하도록 설정
라) DR용 전용회선을 사용해 운영서버 All Fault 시, DR Node를 통해 접속 가능하도록 처리

- AWS 구성도

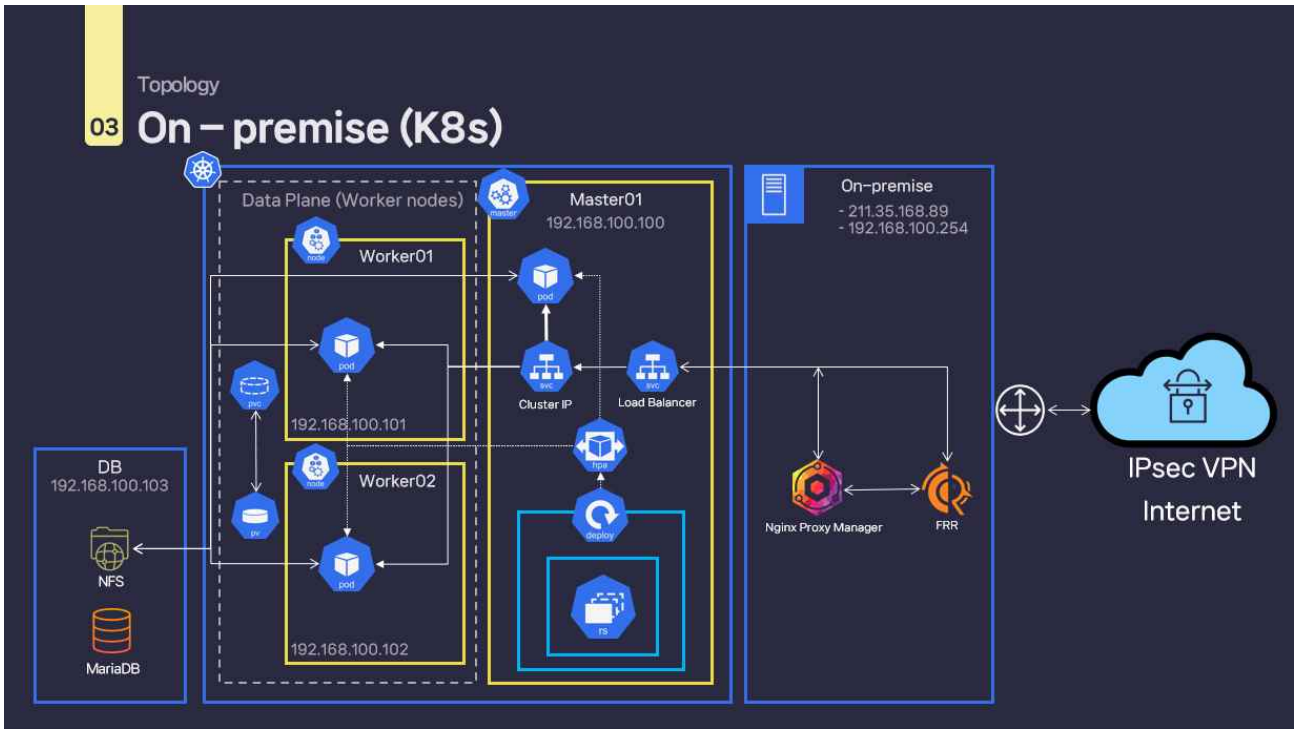



AWS 상세구축

1. 서울 리전에서 2개의 a와 c 프리티어 가용영역(지속적 서비스 운영을 위한 가용영역 분리)과 4개의 서브넷구성으로 퍼블릭과 프라이빗으로 구성
2. 퍼블릭에는 인터넷게이트웨이 생성으로 인터넷 연결 및 프라이빗에는 NAT 인스턴스를 생성하여 베스천호스트로 사용하여 인터넷 연결
3. 오토 스케일링을 통해 인스턴스 자동 생성 및 로드밸런싱을 통한 부하분산
4. RDS를 통해 데이터베이스를 구축
5. EFS를 통해 파일시스템 공유하여 서비스를 구축

- ON-Premise (K8s)

  가) 물리서버 구성 현황




온프레미스 상세구축

1. 마스터노드와 워커노드 두 대를 두어 파드 생성
2. 각 파드를 NFS로 분리된 DB서버와 연동시켜 서비스 구축 및 FRR를 통해 라우팅
3. 각 서비스들은 NGINX PROXY 매니저를 통해 외부에서 접근 가능

사용장비 및 소프트웨어

※ 설치정보

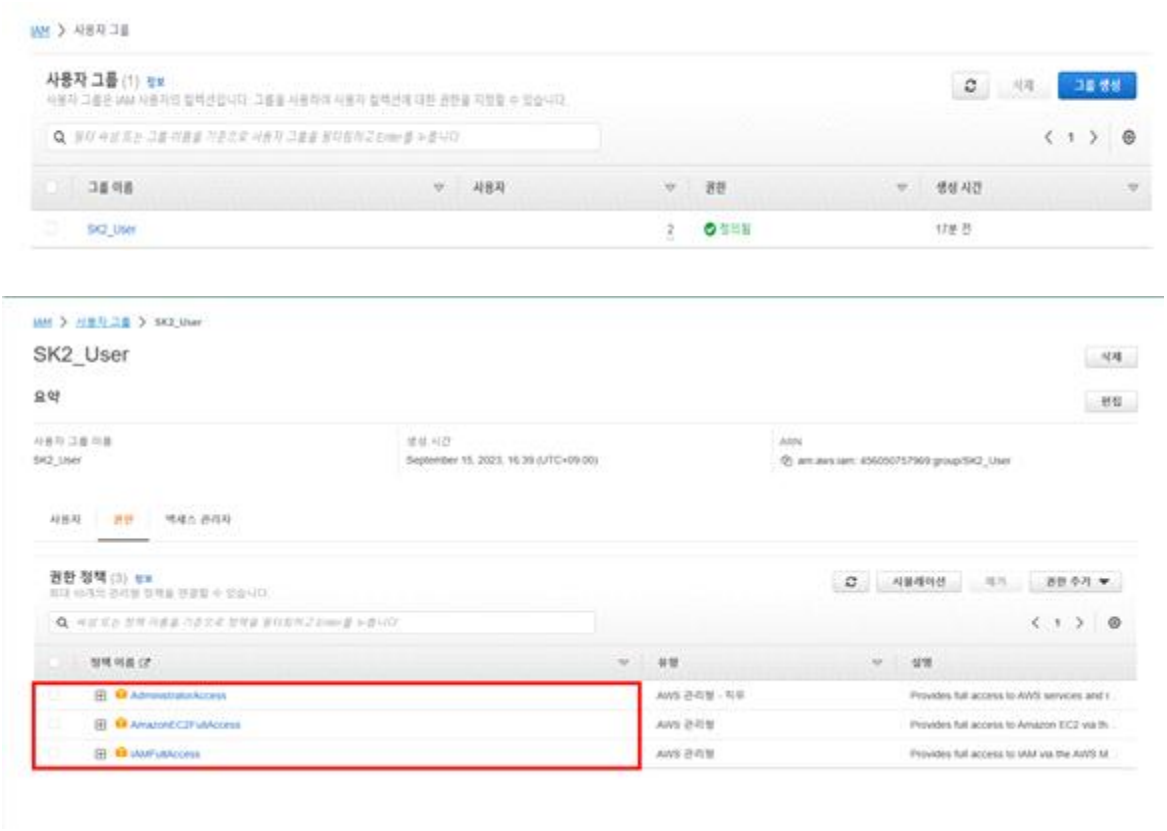
| | | |
|---|---|---|
| 서버구축 | Hypervisor OS | VMware v17.0.1 |
| | 가상 라우터 | FRR v7.5.1 |
| | Server OS | Rocky Linux 8.8 |
| WEB | NGINX Proxy Manager | NGINX Proxy Manager v.2.10.4 |
| | NGINX | NGINX v.1.25.2 |
| | PHP | PHP v8.2.0 |
| Service | DB | MariaDB v10.6.15 |
| | | Nextcloud |
| | | Gnuboard |
| | | Wordpress |
| | DNS | Gabia |
| | | CloudFlare |

사용장비 및 소프트웨어

※ 설치정보

| | |
|---|---|
| 컨테이너 | Docker v24.0.6 |
| | Docker compose v2.21.0 |
| | Kubernetes v1.28.2 |
| AWS | IAM |
| | VPC VPN |
| | EC2 |
| | Route53 |
| | Auto Scaling Group EC2 |
| | RDS |
| | EFS |
| | Application Load Balancer EC2 |
| | Network Load Balancer EC2E |
| | Cloudwatch |
| | VPN |

| AWS | IAM |
|---|---|




IAM 사용자 그룹을 생성하여 관리자 권한을 추가

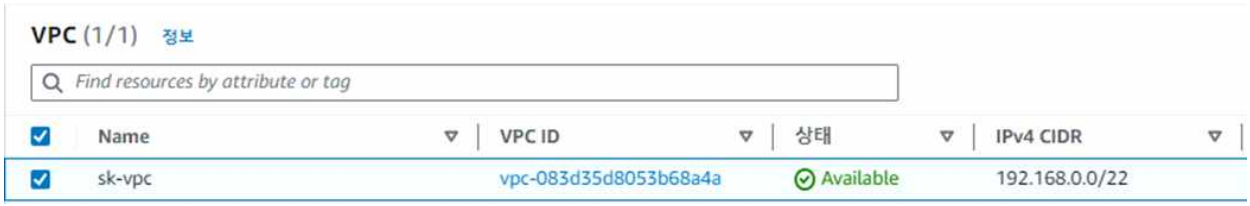
- 그룹 생성 및 권한 추가

- 시스템 관리 및 생성을 하기 위해 관리자 권한 부여

| AWS | IAM |
|---|---|

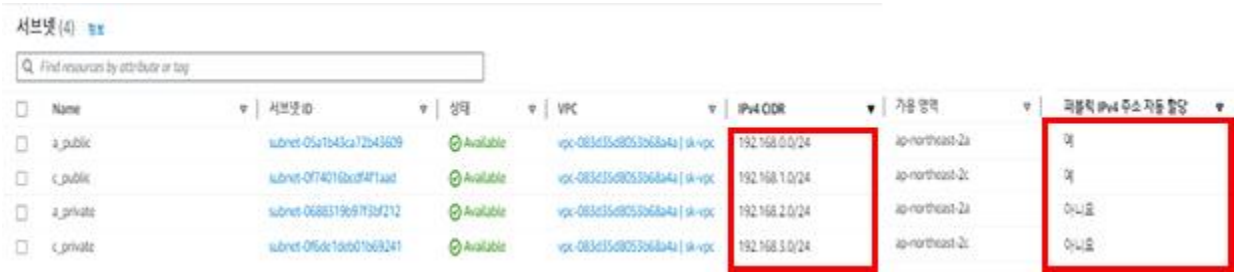
해당 그룹에 사용자 계정 두 개를 추가하여 작업
- 2명의 사용자 계정 생성 후 그룹 추가
- 권한이 추가 된 그룹에 사용자 추가
- 각각의 사용자 계정에 MFA 설정을 통한 보안 강화

| AWS | VPC |
|---|---|




네트워크 격리 및 보안을 위해 VPC 생성
- VPC 생성
- 네트워크 격리 및 보안
- 고가용성 및 확장성

| AWS | VPC |
|---|---|




가용영역을 퍼블릭 및 프라이빗으로 이중화하여 가용성을 확보
- 서브넷을 4개로 분리하여 사설 네트워크 보호

Availability Zone
- Public / Private 각각의 2개의 가용영역을 사용한 가용영역 이중화로 가용성 확보

Subnet
- Public / Private 서브넷을 분리하여 사설 네트워크 보호

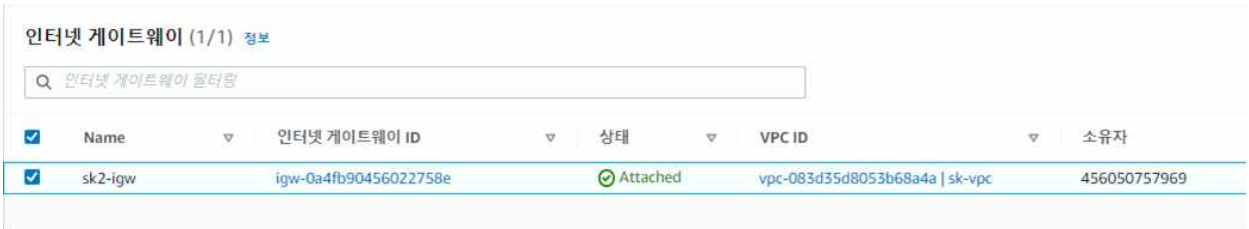
- 네트워크 구성 현황

  가) Router 구성

| AWS | VPC |
|---|---|

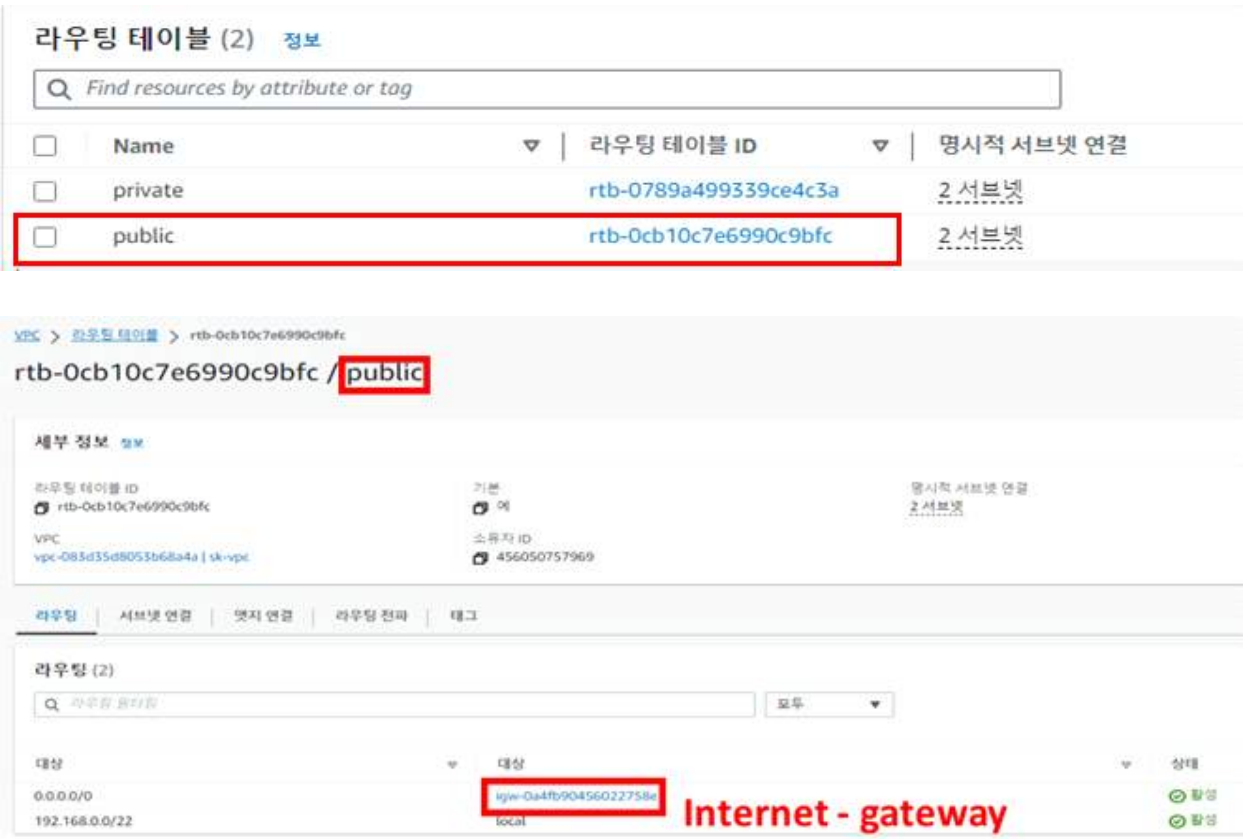
퍼블릭 가용 영역에 공인 IP 할당
- 인터넷 액세스 및 외부 서비스 통합

| AWS | 인터넷 게이트웨이 |
|---|---|



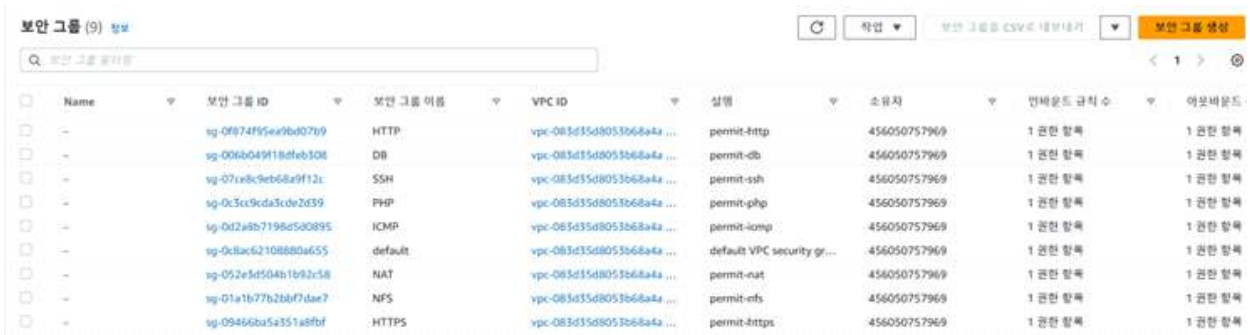

인터넷 게이트웨이 생성
- 인터넷 디폴트 게이트웨이 역할
- 인터넷 액세스 제공
- 퍼블릭 서비스 제공

Router




퍼블릭 라우팅 테이블에 등록하여 인터넷과 연결되도록 설정
- Public 서브넷들은 인터넷게이트웨이를 통해 외부와 연결

| AWS | 보안 그룹 |
|---|---|

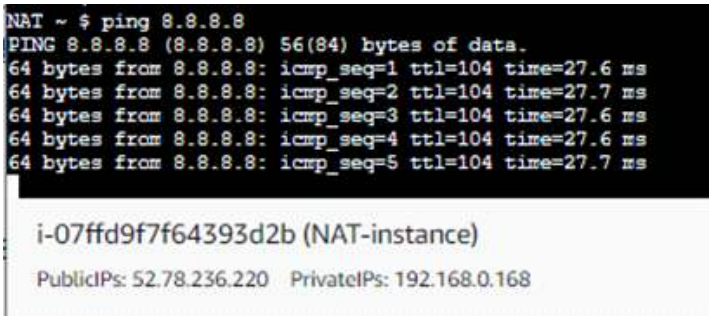



인스턴스 생성에 필요한 보안그룹을 등록 , 키페어 생성

보안그룹 생성
1. 네트워크 보안 제어
2. 방화벽 역할
3. 로그 및 감사

키페어 생성
1. 인스턴스 액세스 제어
2. SSH & RDP 액세스
3. 암호화 키 관리

| AWS | EC2 |
|---|---|




Public 인스턴스는 인터넷 게이트웨이 라우팅으로 인해 외부와 연결이 됨

| AWS | EC2 |
|---|---|

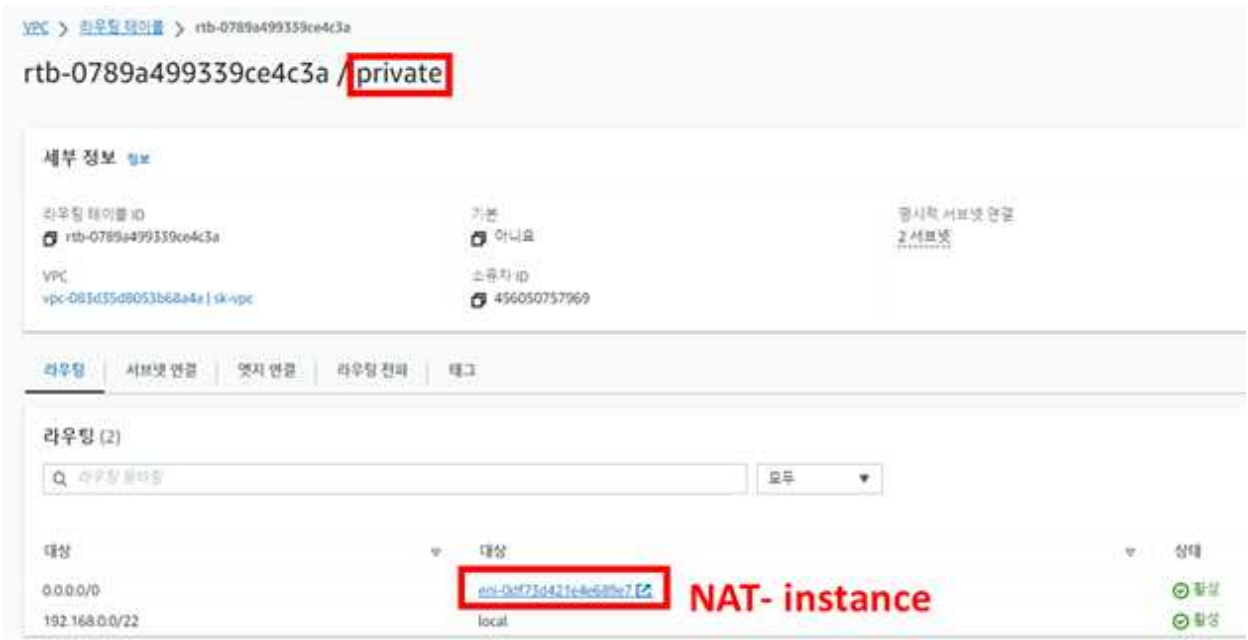
Private 인스턴스는 퍼블릭 IP를 할당받지 못 했기 때문에
직접 접속이 되지 않고 인터넷 연결 또한 되지 못 함

| AWS | EC2 – NAT Instance |
|---|---|

NAT 인스턴스를 생성하여 Private 인스턴스에서도 인터넷 연결 뿐 만 아니라
Bastion host로 접속이 가능하도록 설정

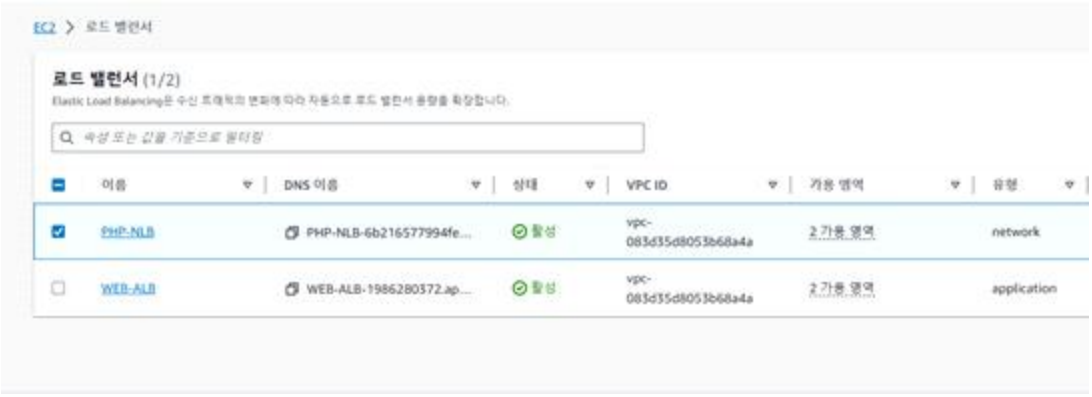
NAT instance
- NAT instance를 Bastion host로 사용
- Private cloud의 인터넷 연결 가능
- 사설 네트워크 보안 강화

| AWS | 라우팅 테이블 |
|---|---|



NAT 인스턴스를 Private Routing Table에 등록하여 인터넷 연결

| AWS | 로드 밸런서 |
|---|---|

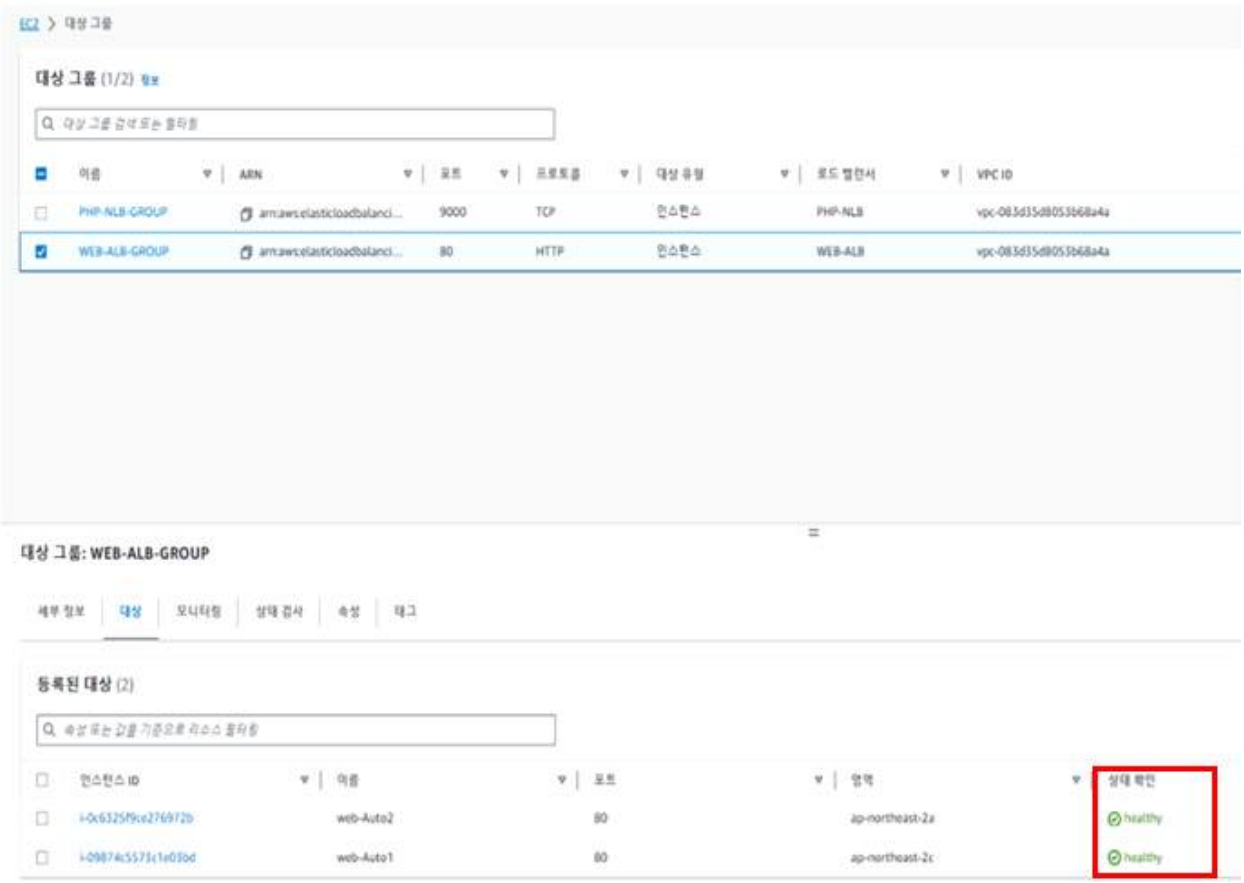



웹은 퍼블릭 가용영역에서 어플리케이션 로드밸런서로 생성하여 트래픽 분산,
PHP는 프라이빗 가용영역에서 네트워크 로드밸런서를 생성하여 트래픽을 분산하여 가용성 증가

| AWS | 대상 그룹, 상태 검사 |
|---|---|

- 대상 그룹을 사용하여 애플리케이션의 신뢰성과 가용성 향상
- 트래픽을 효과적으로 관리
- 상태 검사 경로 : /health.html

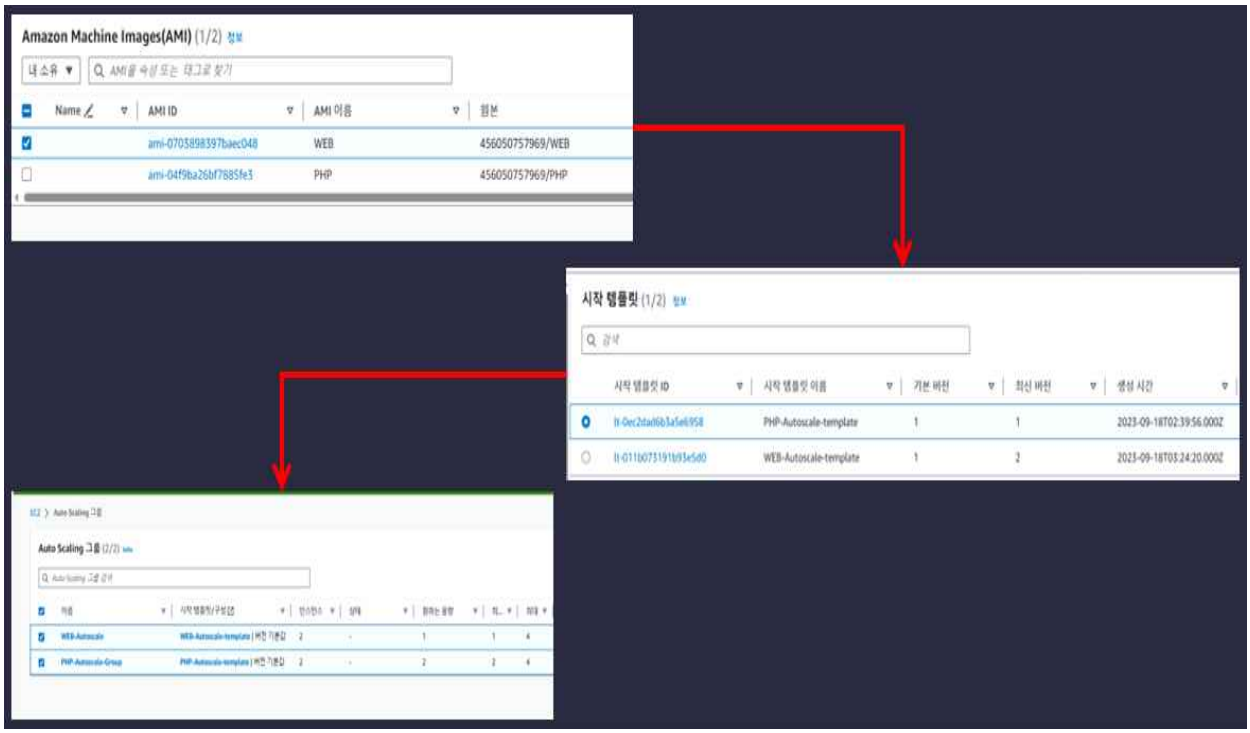
## 대상 그룹, 상태 검사




health check를 통해서 정상 작동되지 않는 인스턴스로는 트래픽을 보내지 않도록 설정

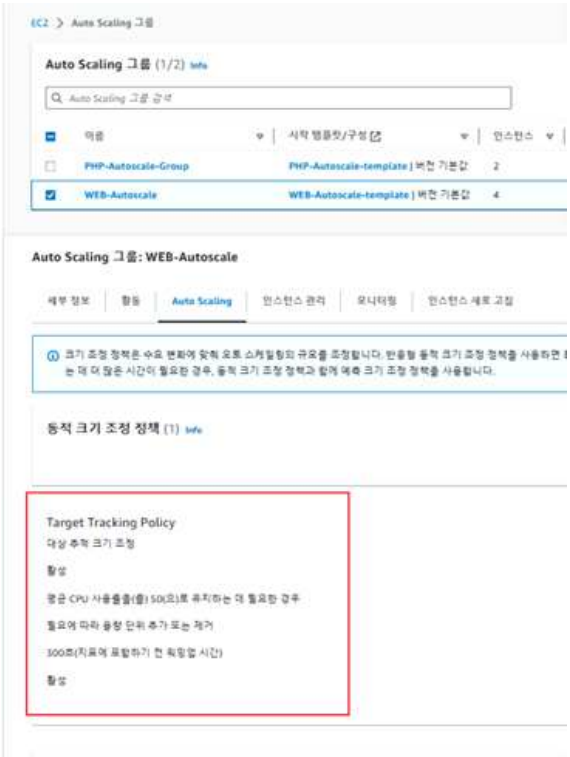
WEB-ALB-GROUP 성공 코드
- 대상으로부터 응답 성공 확인 코드
- 200-399번 코드 지정 -> healthy

| AWS | AMI |
|---|---|




오토 스케일링 생성으로 이미지를 생성하고 시작 템플릿을 생성함으로서 이루어 짐

| AWS | Auto Scaling 그룹 |
|---|---|

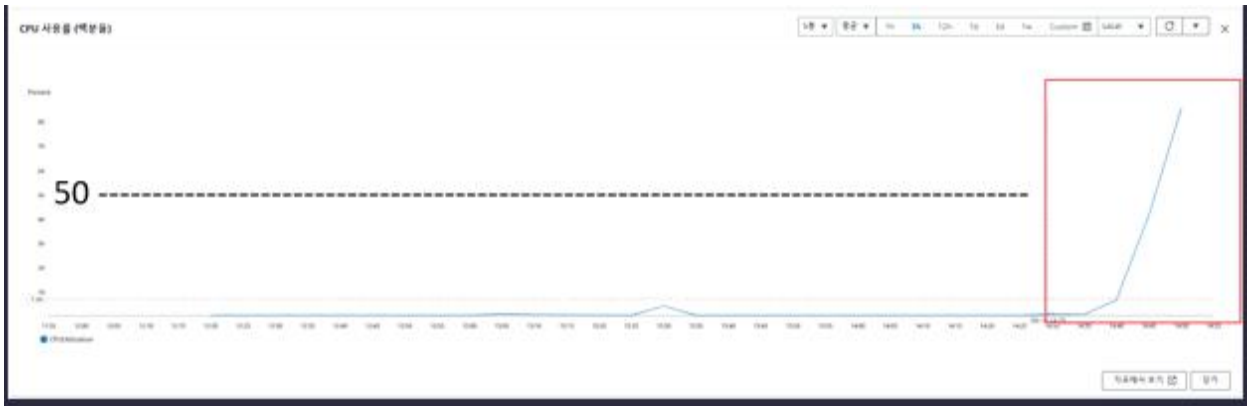


- 오토 스케일링 조절 정책으로 CPU 사용률에 따라 인스턴스 자동 추가 및 제거가 되도록 설정
- 조절 정책으로 평균 CPU 사용률을 50% 이상 초과 할 시 2개에서 최대 4개까지 늘어나도록 설정

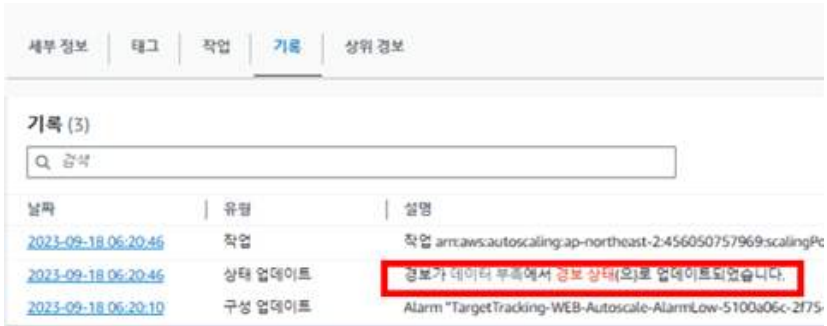
| AWS | IP : WEB 서버 공통 |
|---|---|

```
WEB1 ~ $ stress --cpu 1 --timeout 1800s
stress: info: [3795] dispatching hogs: 1 cpu, 0 io, 0 vm, 0 hdd
```

평균 CPU 사용률을 50% 이상 트래픽 증가 시키기 위해 Stress 명령어 사용



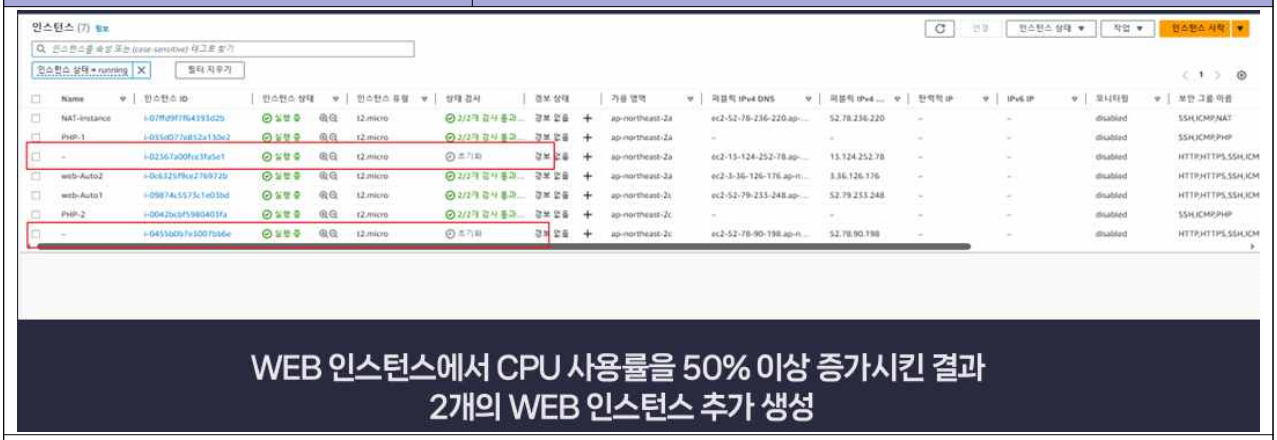

CPU 사용률 50% 초과




데이터 부족으로 인한 경보 상태 확인
(Cloudwatch)

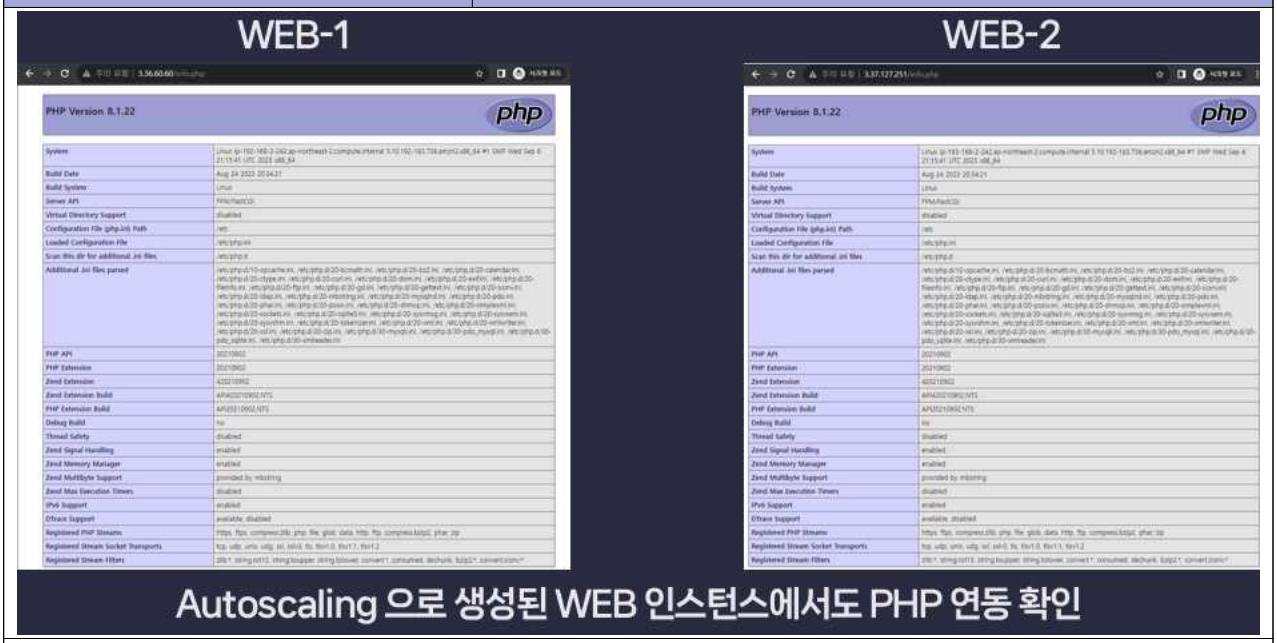
- 스트레스 부하 명령어를 통해 CPU 사용률을 일시적으로 증가 시켰음
- 모니터링을 통해 CPU 사용률이 50% 이상 증가 한 것을 확인

| AWS | CPU 부하 테스트 |
|---|---|




- 인스턴스가 2개가 초가 생성되어 총 4개의 인스턴스가 된 것을 확인

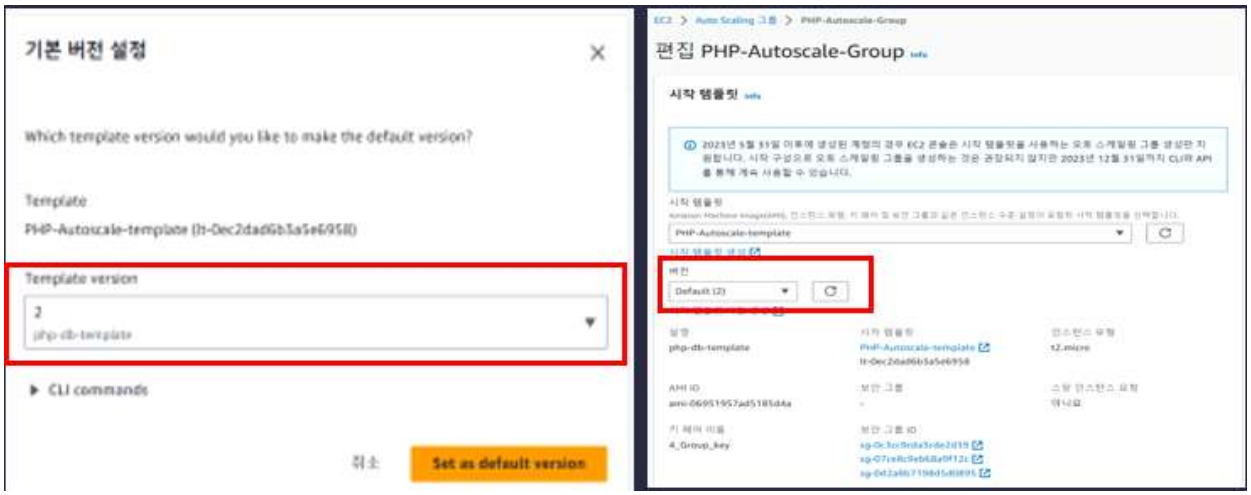
| AWS | PHP 연동 확인 |
|---|---|




- 추가로 생성된 웹 인스턴스에서도 PHP 연동 확인으로 인해
- 기존의 인스터스와 동일하여 서비스의 지속성을 높임

| AWS | AMI version |
|---|---|

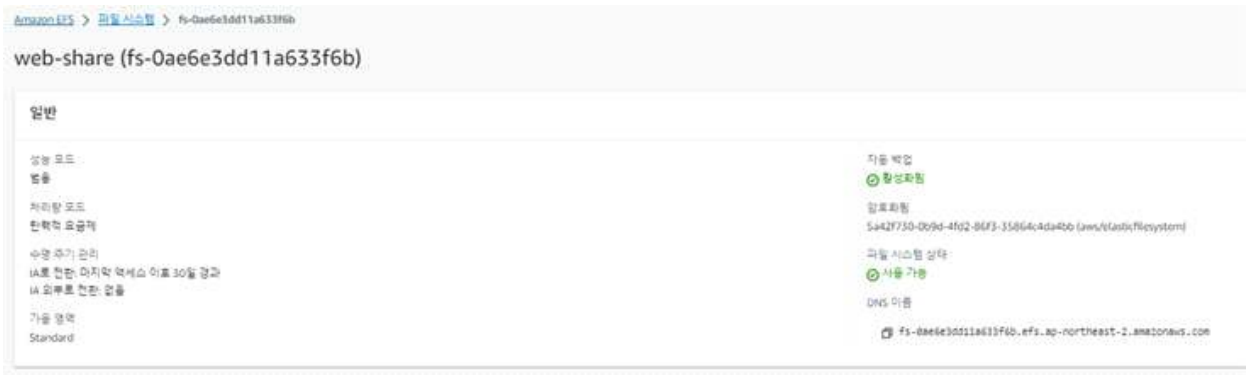
Autoscaling 버전 생성




성능 개선 및 최적화를 시키기 위해 버전을 변경하여 오토 스케일링에 적용

Auto scaling 버전 변경
- 새로운 기능과 성능 개선 및 최적화
- 버그 수정과 안정성 향상
- 호환성 유지

| AWS | File System |
|---|---|




```
WEB1 ~ $ df -Th
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  732K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  1.9G  6.2G  24% /
127.0.0.1:/    nfs4      8.0E  106M  8.0E   1% /usr/share/nginx/html
tmpfs          tmpfs      96M     0   96M   0% /run/user/0
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
```

```
PHP-1 ~ $ df -Th
Filesystem     Type      Size  Used Avail Use% Mounted on
devtmpfs       devtmpfs  468M     0  468M   0% /dev
tmpfs          tmpfs     477M     0  477M   0% /dev/shm
tmpfs          tmpfs     477M  612K  476M   1% /run
tmpfs          tmpfs     477M     0  477M   0% /sys/fs/cgroup
/dev/xvda1     xfs       8.0G  2.0G  6.1G  24% /
127.0.0.1:/    nfs4      8.0E  106M  8.0E   1% /usr/share/nginx/html
tmpfs          tmpfs      96M     0   96M   0% /run/user/1000
```

파일시스템 공유를 위해 EFS를 생성 및 연동

EFS
- EFS를 이용해 NFS 사용
- 추후 로드밸런싱을 이용한 웹서버를 구축하기 위해
  NGINX의 루트 디렉터리 /usr/share/nginx/html을 NFS mount

| AWS | RDS, 파라미터 그룹 |
|---|---|

고가용성 및 자동 백업을 위해 RDS 생성

RDS 생성
- 고가용성 및 자동 백업을 위해 RDS 사용
- 분리된 DB 서버 구축으로 DB 안정성 및 보안 강화

| AWS | MariaDB |
|---|---|

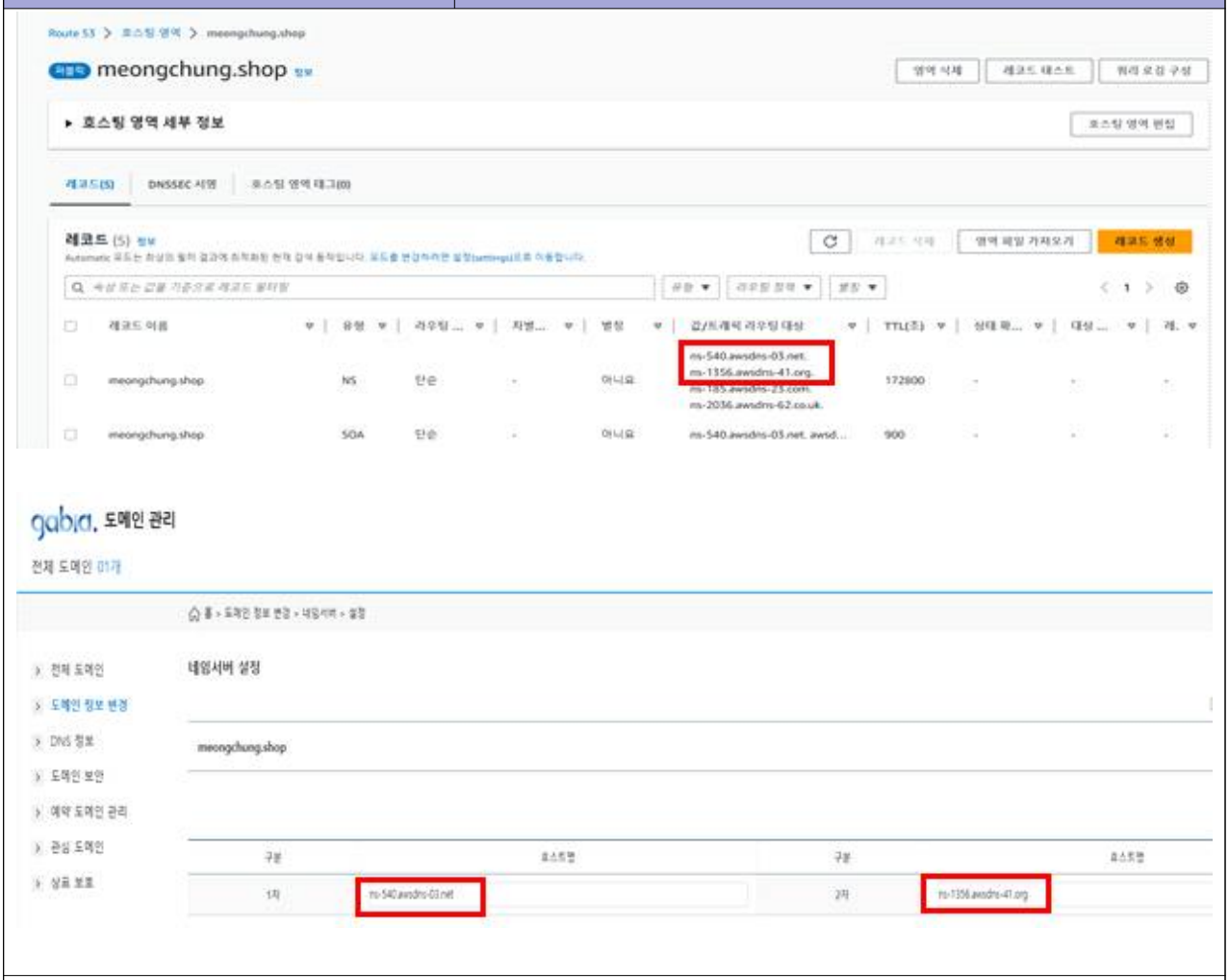
```
52  tomcat9                 available    [ =stable ]
53  unbound1.13             available    [ =stable ]
54 †mariadb10.5=latest      enabled      [ =stable ]
```

```
[ec2-user@ip-192-168-3-178 ~]$ mysql -u admin -p -h db.meongchung.shop
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 206
Server version: 10.6.14-MariaDB managed by https://aws.amazon.com/rds/

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statemen
t.
```

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| gnuboard           |
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
| sys                |
| wordpress          |
+--------------------+
7 rows in set (0.001 sec)

MariaDB [(none)]> select user,host,password from mysql.user;
+-------------+-----------+-------------------------------------------+
| User        | Host      | Password                                  |
+-------------+-----------+-------------------------------------------+
| mariadb.sys | localhost |                                           |
| rdsadmin    | localhost | *A6258D0A818748EED2F480CF4F45971C24A13E40 |
| admin       | %         | *EA8D87B5453F22FC060E38F7F759A2A37DC325E1 |
| wordpress   | %         | *A4B6157319038724E3560894F7F932C8886EBFCF |
| gnuuser     | localhost | *A4B6157319038724E3560894F7F932C8886EBFCF |
| gnuuser     | %         | *A4B6157319038724E3560894F7F932C8886EBFCF |
+-------------+-----------+-------------------------------------------+
6 rows in set (0.001 sec)
```

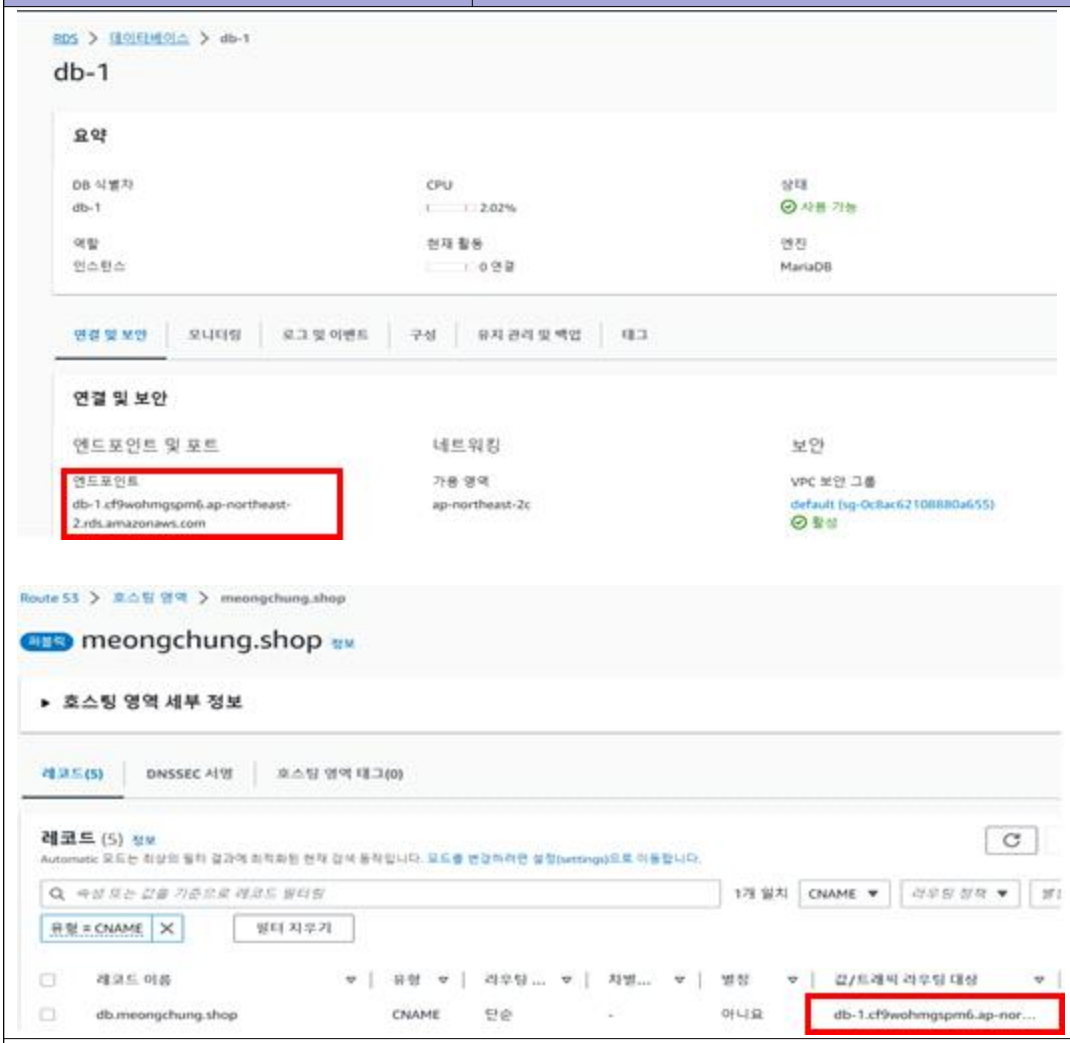
- Private 가용영역에 DB를 설치해서 계정 생성 및 권한 설정을 작업을 할 수 있게 함

| AWS | Route53 |
|---|---|



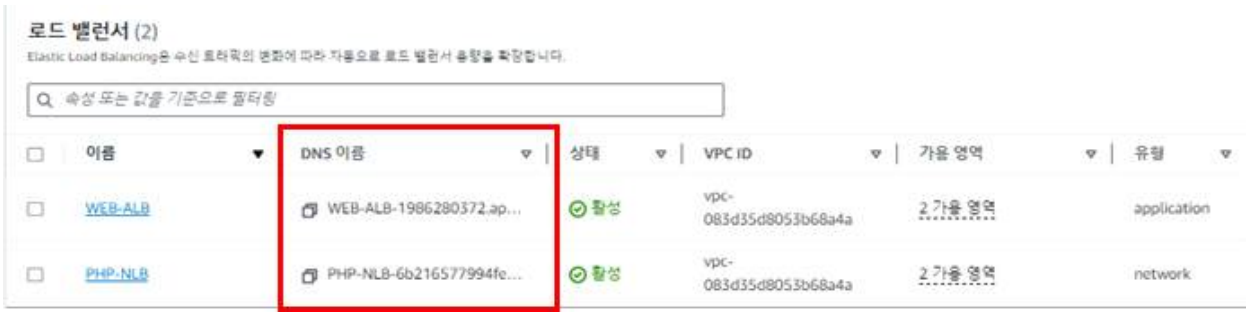

- 가비아에서 구매한 도메인을 Route53에서 관리하기 위해 네임서버로 설정

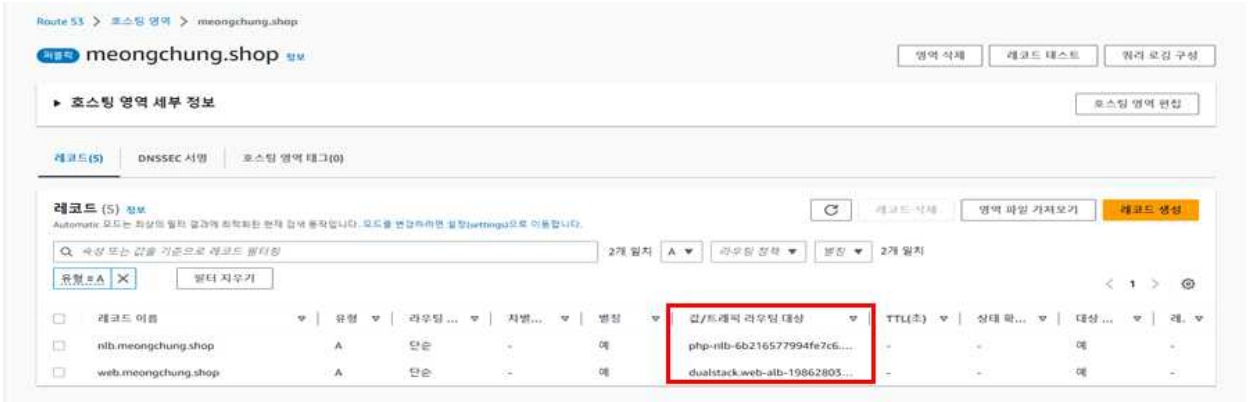
| AWS | RDS 엔드포인트 |
|---|---|




- RDS 엔드포인트를 Route53 가상호스트에 등록하여 사용

| AWS | 로드 밸런서 (ALB, NLB) |
|---|---|




- 각 로드밸런서 DNS를 Route53 가상호스트에 등록하여 사용

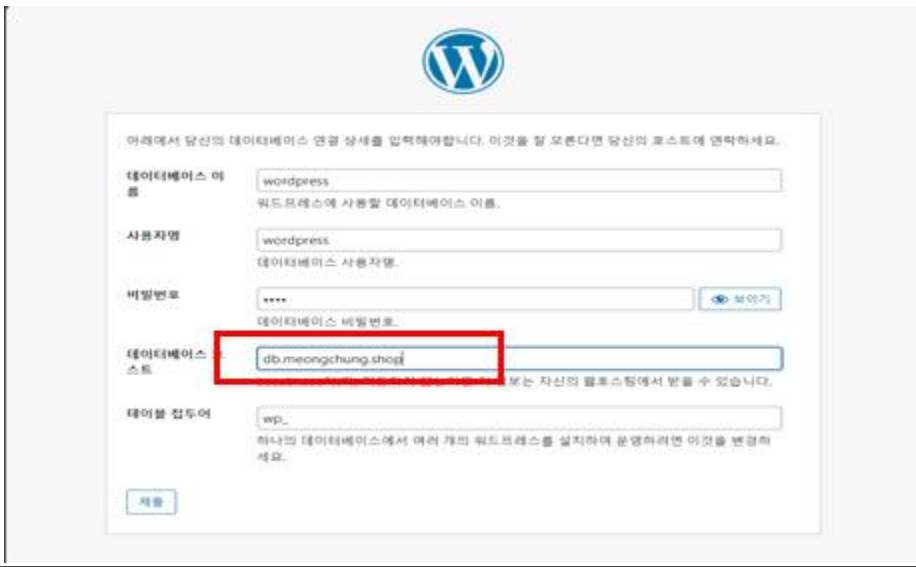
Load Balancer : 로드밸런싱을 통해 가용성 증가 및 부하 분산

Application LB를 이용해
-> HTTP 로드밸런싱

Network LB를 이용해
-> PHP용 로드밸런싱

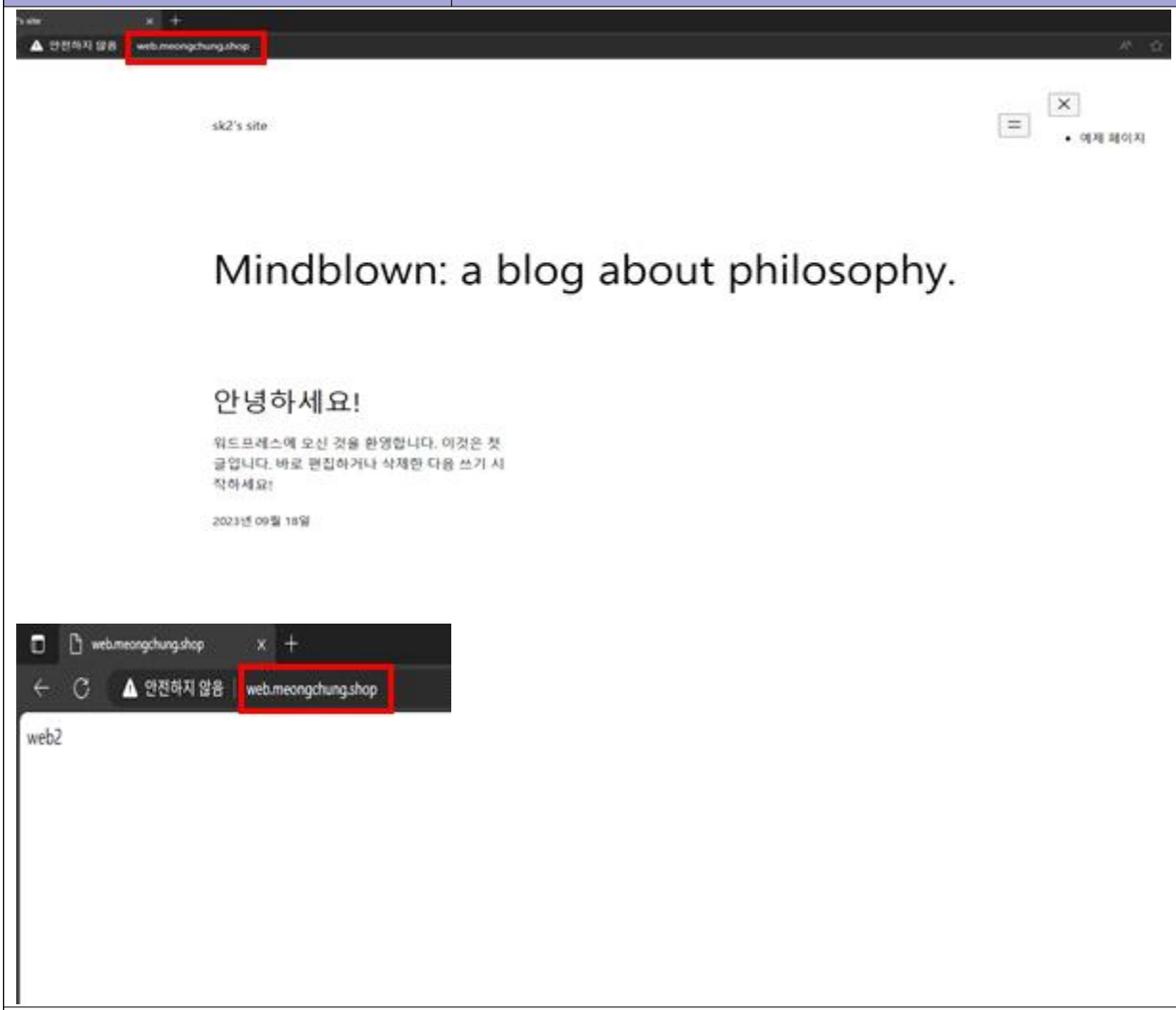
| AWS | Nginx – PHP 연동 |
|---|---|

```
location ~ [^/].php(/|$) {
    fastcgi_split_path_info ^(.+?\.php)(/.*)$;
    set $path_info $fastcgi_path_info;
    fastcgi_index index.php;
    include fastcgi_params;
    fastcgi_pass nlb.meongchung.shop:9000;
    fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
}
```




- NLB 가상호스트를 적용하여 NGINX와 PHP를 연동
- 워드프레스 설치 시 데이터 호스트를 RDS 가상호스트에 적용

| AWS | Wordpress |
|---|---|

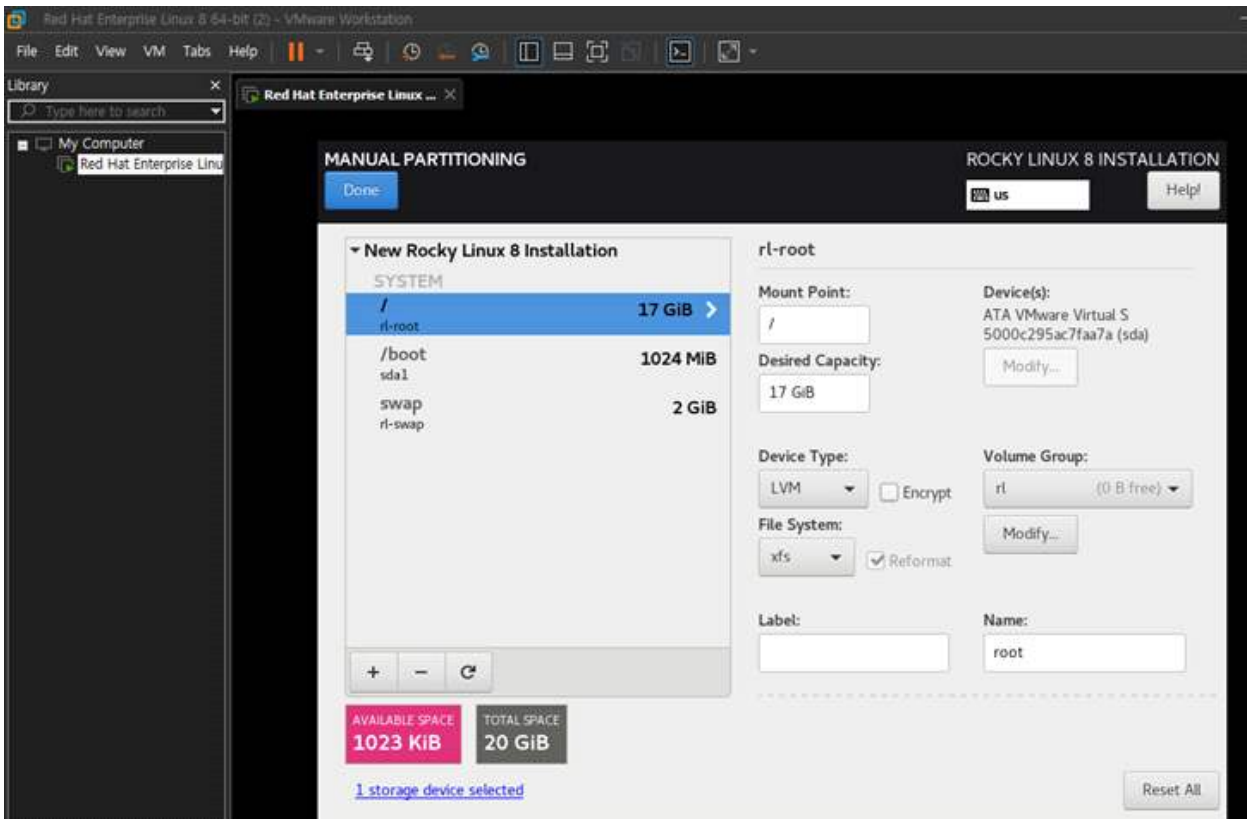



- ALB 가상호스트로 로드밸런싱하여 웹 접속

ALB 접속하는 Domain name
- Route53에 web.meongchung.shop 생성
- 해당 도메인으로 ALB 접근(WEB)
- NGINX Load Balancing
- 두 인스턴스로 부하 분산

## ● On Premise 상세 기술

| ON-PREMISE | VM Partitioning, Resource |
|---|---|




Partitioning
20GB 용량기준
- /root: 17GB
- /boot: 1GB
- /swap: 2GB

Resource
- CPU core: 2
- Memory: 4096MB
- Disk (SATA): 20GB

| ON-PREMISE | VM 생성 |
|---|---|

```
[root@Onpremise ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq s
    link/ether 00:0c:29:e0:ab:a6 brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    inet 211.35.168.89/24 brd 211.35.168.255 scope global dynami
       valid_lft 6967sec preferred_lft 6967sec
    inet6 fe80::20c:29ff:fee0:aba6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq s
    link/ether 00:0c:29:e0:ab:b0 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 192.168.100.254/24 brd 192.168.100.255 scope global nop
       valid_lft forever preferred_lft forever
    inet6 fe80::5e14:1a20:a977:64c6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

```
[root@master01 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqu
    link/loopback 00:00:00:00:00:00 brd 00:00:00:0
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1
    link/ether 00:0c:29:52:85:74 brd ff:ff:ff:ff:f
    altname enp3s0
    inet 192.168.100.100/24 brd 192.168.100.255 sc
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe52:8574/64 scope link r
       valid_lft forever preferred_lft forever
```

```
[root@worker01 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc no
    link/loopback 00:00:00:00:00:00 brd 00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
    link/ether 00:0c:29:a0:b9:bd brd ff:ff:ff:ff
    altname enp3s0
    inet 192.168.100.101/24 brd 192.168.100.255
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fea0:b9bd/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@worker02 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc no
    link/loopback 00:00:00:00:00:00 brd 00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
    link/ether 00:0c:29:ab:6c:5f brd ff:ff:ff:ff
    altname enp3s0
    inet 192.168.100.102/24 brd 192.168.100.255
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feab:6c5f/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@DB ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noq
    link/loopback 00:00:00:00:00:00 brd 00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
    link/ether 00:0c:29:ed:cc:b8 brd ff:ff:ff:ff
    altname enp3s0
    inet 192.168.100.103/24 brd 192.168.100.255
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:feed:ccb8/64 scope link
       valid_lft forever preferred_lft forever
```

VM 생성
- 기존 Linux 템플릿을 Cloning하여 5대의 리눅스 가상머신을 생성
- ON-PREMISE 가상머신은 Bridge, Host-only 2개의 네트워크 인터페이스를 생성
- 그 외 나머지 VM은 host-only VMnet1 네트워크만 사용

VM IP고정
- ON-PREMISE VM : Bridged, host-only(VMnet1)
- host-only(VMnet1): Static IP 설정

| ON-PREMISE | Docker, Docker-compose 설치 |
|---|---|

```
[root@Onpremise ~]# docker compose version
Docker Compose version v2.21.0
```

```
[root@Onpremise ~]# docker version
Client: Docker Engine - Community
 Version:           24.0.6
 API version:       1.43
 Go version:        go1.20.7
 Git commit:        ed223bc
 Built:             Mon Sep  4 12:33:07 2023
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          24.0.6
  API version:      1.43 (minimum version 1.12)
  Go version:       go1.20.7
  Git commit:       1a79695
  Built:            Mon Sep  4 12:32:10 2023
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.24
  GitCommit:        61f9fd88f79f081d64d6fa3bb1a0dc71ec870523
 runc:
  Version:          1.1.9
  GitCommit:        v1.1.9-0-gccaecfc
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

- ON-PREMISE 가상머신에 도커 v24.0.6와 도커컴포즈 v2.21.0 설치

주요 목적
- Docker Build를 통해 서비스 이미지 생성
- NGINX Proxy Manager 설치

| ON-PREMISE | FRRouting |
|---|---|

```
[root@onpremise ~]# systemctl status frr
● frr.service - FRRouting
   Loaded: loaded (/usr/lib/systemd/system/frr.service; enabled; vendor pr
   Active: active (running) since Mon 2023-09-18 11:13:30 KST; 4s ago
     Docs: https://frrouting.readthedocs.io/en/latest/setup.html
  Process: 5322 ExecStart=/usr/libexec/frr/frrinit.sh start (code=exited,
   Status: "FRR Operational"
    Tasks: 15 (limit: 24660)
   Memory: 20.0M
   CGroup: /system.slice/frr.service
           ├─5339 /usr/libexec/frr/watchfrr -d -F traditional zebra bgpd o
           ├─5357 /usr/libexec/frr/zebra -d -F traditional -A 127.0.0.1 -s
           ├─5362 /usr/libexec/frr/bgpd -d -F traditional -A 127.0.0.1
           ├─5369 /usr/libexec/frr/ospfd -d -F traditional -A 127.0.0.1
           └─5372 /usr/libexec/frr/staticd -d -F traditional -A 127.0.0.1
```

```
[root@onpremise ~]# vtysh

Hello, this is FRRouting (version 7.5.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr#
```

- ON-PREMISE 가상머신의 서로 다른 네트워크 대역인 Bridged와 VMnet1을 통신하기 위해, 오픈소스 네트워크 라우팅 소프트웨어인 FRR 설치
- 네트워크의 라우팅 및 경로 선택을 관리
- Bridged와 Host-only (VMnet1) 네트워크 통신 가능
- vtyos 명령어로 FRR 접근

| ON-PREMISE | FRRouting table |
|---|---|

```
Frr# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

K>* 0.0.0.0/0 [0/100] via 211.35.168.254, ens160, src 211.35.168.89, 1d07h19m
C>* 172.18.0.0/16 is directly connected, br-1040b26c52e8, 1d07h19m
C>* 192.168.100.0/24 is directly connected, ens192, 1d07h19m
C>* 211.35.168.0/24 is directly connected, ens160, 1d07h19m
```

Public IP (Bridged)

Private IP (VMnet1)

```
[root@Onpremise ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:e0:ab:a6 brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    inet 211.35.168.89/24 brd 211.35.168.255 scope global dynamic noprefixroute ens160
       valid_lft 5956sec preferred_lft 5956sec
    inet6 fe80::20c:29ff:fee0:aba6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:e0:ab:b0 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 192.168.100.254/24 brd 192.168.100.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::5e14:1a20:a977:64c6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
4: br-1040b26c52e8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group defaul
t
    link/ether 02:42:b0:98:00:10 brd ff:ff:ff:ff:ff:ff
    inet 172.18.0.1/16 brd 172.18.255.255 scope global br-1040b26c52e8
       valid_lft forever preferred_lft forever
    inet6 fe80::42:b0ff:fe98:10/64 scope link
       valid_lft forever preferred_lft forever
```

- FRR 라우팅 테이블에 두 네트워크가 connected로 통신이 가능한 것 확인 가능
- Bridged Network : 211.35.168.89/24
- VMnet1 :192.168.100.254/24

| ON-PREMISE | DDNS, Crontab |
|---|---|

```
[root@Onpremise ~]# cat ddns.sh
#!/bin/bash

### dnf install bind-utils -y 를 해서 host command 설치 되어야 함
### cloudflare information

dns_record="kkangtae.store"
zoneid="c60e10dd3670eab98219707b69c16ff4"
cloudflare_zone_api_token="zuZ5npTACE08w39czf9X3hvNHdat0kPLM-8TYZ_T"
proxied="false"
ttl=120
# internal로 변경 시 서버의 interface에 있는 IP가 입력됨
what_ip=external

### Check validity of "ttl" parameter
if [ "${ttl}" -lt 120 ] || [ "${ttl}" -gt 7200 ] && [ "${ttl}" -ne 1
  echo "Error! ttl out of range (120-7200) or not set to 1"
```

```
[root@Onpremise ~]# crontab -l
#cloudfalre-ddns
*/5 * * * * ~/ddns.sh
@reboot ~/ddns.sh
```

- Bridged 네트워크의 동적 IP 주소가 바뀌어도 도메인 이름으로 웹 서비스에 접근이 가능하도록 DDNS 쉘 스크립트를 만들어 실행
- 위 작업을 재부팅, 그리고 5분마다 자동 실행을 위한 Crontab 설정

| ON-PREMISE | K8s 설치 및 연결 |
|---|---|

```
[root@master1 ~]# kubectl version --client
Client Version: v1.28.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

```
[root@worker1 ~]# kubectl version --client
Client Version: v1.28.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

```
[root@worker2 ~]# kubectl version --client
Client Version: v1.28.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
```

```
[root@master01 ~]# k get node -o wide
NAME       STATUS   ROLES           AGE   VERSION   INTERNAL-IP       EXTERNAL-IP   OS-IMAGE
                     KERNEL-VERSION                  CONTAINER-RUNTIME
master01   Ready    control-plane   3d    v1.28.2   192.168.100.100   <none>        Rocky Linux 8.
8 (Green Obsidian)   4.18.0-477.21.1.el8_8.x86_64   containerd://1.6.24
worker01   Ready    <none>          3d    v1.28.2   192.168.100.101   <none>        Rocky Linux 8.
8 (Green Obsidian)   4.18.0-477.21.1.el8_8.x86_64   containerd://1.6.24
worker02   Ready    <none>          3d    v1.28.2   192.168.100.102   <none>        Rocky Linux 8.
8 (Green Obsidian)   4.18.0-477.21.1.el8_8.x86_64   containerd://1.6.24
```

- 컨테이너 어플리케이션 배포를 위한 Kubernetes v1.28.2 설치
- Master, worker node로 사용 할 가상머신에 Kubernetes 설치 및 클러스터 구성
- master01, worker01, worker02 node에 K8s 설치
- Master node 및 Worker node 지정
- Master, Worker node 연동하여 Cluster 구축

| ON-PREMISE | Taint 설정 |
|---|---|

```
[root@master1 ~]# kubectl taint nodes master1 node-role.kubernetes.io/control-plane:NoSchedule-
node/master1 untainted
```

```
[root@master1 ~]# kubectl describe node master1 | grep -i taint
Taints:             <none>
```

- 클러스터 내 자원이 부족 할 것을 대비하여 Master node에 taint (테인트) 옵션을 제거
- untainted 상태의 Master node에도 Worker node처럼 Pod을 스케쥴링 가능
- default : taint 상태

| ON-PREMISE | Calico 배포 |
|---|---|

```
[root@master01 ~]# cat calico.yaml
                name: calico-config
                key: veth_mtu
            # The default IPv4 pool to create on startup if none exists. Pod IPs will be
            # chosen from this range. Changing this value after installation will have
            # no effect. This should fall within `--cluster-cidr`.
            - name: CALICO_IPV4POOL_CIDR
              value: "192.168.101.0/24"
            # Disable file logging so `kubectl logs` works.
            - name: CALICO_DISABLE_FILE_LOGGING
              value: "true"
```

```
[root@master1 ~]# kubectl get pods -n kube-system
NAME                                       READY   STATUS    RESTARTS   AGE
calico-kube-controllers-7ddc4f45bc-d27wr   1/1     Running   0          43s
calico-node-44mfc                          1/1     Running   0          43s
calico-node-k8g2f                          1/1     Running   0          43s
calico-node-x8w9f                          1/1     Running   0          43s
coredns-5dd5756b68-68s25                   1/1     Running   0          22m
coredns-5dd5756b68-jn9b4                   1/1     Running   0          22m
etcd-master1                               1/1     Running   0          23m
kube-apiserver-master1                     1/1     Running   0          23m
kube-controller-manager-master1            1/1     Running   0          23m
kube-proxy-52fq8                           1/1     Running   0          21m
kube-proxy-fvcwx                           1/1     Running   0          22m
kube-proxy-m8gt7                           1/1     Running   0          21m
kube-scheduler-master1                     1/1     Running   0          23m
[root@master1 ~]#
[root@master1 ~]# k get nodes
NAME      STATUS   ROLES           AGE   VERSION
master1   Ready    control-plane   23m   v1.28.2
worker1   Ready    <none>          21m   v1.28.2
worker2   Ready    <none>          21m   v1.28.2
```

- 컨테이너를 위한 네트워킹, IP 관리, 접근 제어 등의 기능 제공을 위해 Calico CNI 설치
- 각 Node에 설치하여 각 Pod 간 네트워크 통신 가능
- Calico 설치를 위한 yaml 파일을 생성 및 실행
- 이때, 새로 생성될 Pod의 IPv4 Pool : 192.168.101.0/24
- Calico Pod가 생성 및 실행되는 것을 확인

| ON-PREMISE | MetalLB 설치 |
|---|---|

```
[root@master01 ~]# cat metallb-ip.yaml
# Loadbalancer가 사용할 IP 주소 대역 환경 설정
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: first-pool
  namespace: metallb-system
spec:
  addresses:
  - 192.168.100.200-192.168.100.250 #k8s가 사용하는 네트워크 대역 중 서비스에 줄 IP 대역
---
# IP가 L2 계층에서 ARP protocol에 반응할 수 있는 환경 설정
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: default
  namespace: metallb-system
spec:
  ipAddressPools:
  - first-pool
[root@master1 ~]# kubectl get all -n metallb-system
NAME                                  READY   STATUS    RESTARTS   AGE
pod/controller-5c6b6c8447-p6gft       1/1     Running   0          94s
pod/speaker-56rdj                     1/1     Running   0          94s
pod/speaker-ntjrk                     1/1     Running   0          94s
pod/speaker-zp4qc                     1/1     Running   0          94s

NAME                      TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S
service/webhook-service   ClusterIP   10.106.13.77   <none>        443/TC

NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILAB
daemonset.apps/speaker    3         3         3       3            3

NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/controller   1/1     1            1           94s

NAME                                    DESIRED   CURRENT   READY   AGE
replicaset.apps/controller-5c6b6c8447   1         1         1       94s
```

- Kubernetes Cluster에서 사용할 수 있는 오픈소스 로드밸런서
- Metallb 설치를 위한 yaml 작성 및 실행
- 이때, 로드밸런서 Service의 EXTERNAL-IP 대역을 192.168.100.200 ~ 192.168.100.250/24 로 지정

| ON-PREMISE | Metric-Server |
|---|---|

```
[root@master01 ~]# kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

```
[root@master01 ~]#  kubectl edit deploy metrics-server -n kube-system
```

```
        - args:
          - --cert-dir=/tmp
          - --secure-port=4443
          - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
          - --kubelet-use-node-status-port
          - --metric-resolution=15s
          - --kubelet-insecure-tls
```

```
          volumeMounts:
          - mountPath: /tmp
            name: tmp-dir
        dnsPolicy: ClusterFirst
        hostNetwork: true
        nodeSelector:
          kubernetes.io/os: linux
```

```
Every 1.0s: kubectl top node

NAME       CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
master1    210m         10%    1760Mi          46%
worker1    44m          2%     1147Mi          62%
worker2    46m          2%     1160Mi          63%
```

- Kubernetes Cluster 내에서 리소스 사용량 체크 및 모니터링을 위해 모든 Node에 Metric-Server 설치
- top 명령어를 통해 CPU, 메모리 사용량 등 리소스 사용량 확인 가능

| ON-PREMISE | MariaDB |
|---|---|

```
[root@DB ~]# systemctl status mariadb
● mariadb.service - MariaDB 10.6.15 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor preset: disabled)
  Drop-In: /etc/systemd/system/mariadb.service.d
           └─migrated-from-my.cnf-settings.conf
   Active: active (running) since Wed 2023-09-20 18:41:06 KST; 1 day 22h ago
     Docs: man:mariadbd(8)
           https://mariadb.com/kb/en/library/systemd/
```

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| gnuboard           |
| information_schema |
| mysql              |
| nextcloud          |
| performance_schema |
| sys                |
| xeboard            |
+--------------------+
7 rows in set (0.001 sec)
```

```
MariaDB [(none)]> select user from mysql.user;
+-------------+
| User        |
+-------------+
| gnuuser     |
| nextuser    |
| root        |
| xeuser      |
| gnuuser     |
| mariadb.sys |
| mysql       |
| nextuser    |
| root        |
| xeuser      |
+-------------+
10 rows in set (0.005 sec)
```

- 웹서비스 Gnuboard와 Nextcloud의 DB를 별도의 VM을 생성하여 지정
- MariaDB v10.6.15 설치
- 데이터베이스 생성 : 'gnuboard', 'nextcloud'
- 사용자 생성 : 'gnuuser', 'nextuser'
- 사용자 권한 부여

| ON-PREMISE | NFS |
|---|---|

```
[root@DB ~]# systemctl status nfs-utils
● nfs-utils.service - NFS server and client services
   Loaded: loaded (/usr/lib/systemd/system/nfs-utils.service; static; vendor preset: disabled)
   Active: active (exited) since Fri 2023-09-22 17:38:07 KST; 4s ago
  Process: 102088 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
 Main PID: 102088 (code=exited, status=0/SUCCESS)
    Tasks: 0 (limit: 24660)
   Memory: 0B
   CGroup: /system.slice/nfs-utils.service
```

```
[root@DB ~]# showmount -e
Export list for DB:
/web *
```

```
[root@DB ~]# ls -l /web
total 12
drwxr-xr-x. 16   33 tape 4096 Sep 21 09:36 gnu
-rw-r--r--.  1 root root    5 Sep 21 11:06 index.html
-rw-r--r--.  1 root root   20 Sep 20 12:34 info.php
drwxr-xr-x.  2   33 tape    6 Sep 21 11:09 nextcloud
```

- DB VM에 NFS 서버를 설치
- /web 공유디렉토리 지정
- DB VM의 IP : 192.168.100.103/24

| ON-PREMISE | PV, PVC |
|---|---|

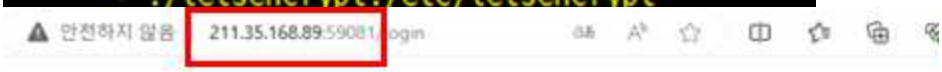
```
[root@master01 ~]# cat << 'EOF' > nfs-pv.yaml
> apiVersion: v1
> kind: PersistentVolume
> metadata:
>   name: nfs-pv
> spec:
>   capacity:
>     storage: 10Gi
>   volumeMode: Filesystem
>   accessModes:
>     - ReadWriteMany
>   mountOptions:
>     - nfsvers=4.2
>   nfs:
>     path: /web
>     server: 192.168.100.103
>
> apiVersion: v1
> kind: PersistentVolumeClaim
> metadata:
>   name: nfs-pvc
> spec:
>   accessModes:
>     - ReadWriteMany
>   resources:
>     requests:
>       storage: 10Gi
> EOF
[root@master01 ~]# ls -l
total 256
-rw-------. 1 root root   1364 Sep 18 10:04 anaconda-ks.cfg
-rw-r--r--. 1 root root 244732 Sep 19 15:39 calico.yaml
-rwxr-xr-x. 1 root root   1781 Sep 19 14:57 default_config.sh
-rw-r--r--. 1 root root    519 Sep 19 16:11 metallb-ip.yaml
-rw-r--r--. 1 root root    404 Sep 19 17:45 nfs-pv.yaml
drwxr-xr-x. 2 root root     81 Sep 19 16:32 yaml
[root@master01 ~]# k apply -f nfs-pv.yaml
persistentvolume/nfs-pv created
persistentvolumeclaim/nfs-pvc created
```

```
[root@master01 ~]# k get pv,pvc
NAME                      CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM
N   AGE
persistentvolume/nfs-pv   10Gi       RWX            Retain           Bound    default/nfs-pvc
    24s

NAME                          STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE
persistentvolumeclaim/nfs-pvc Bound    nfs-pv   10Gi       RWX                           24s
```

- PV, PVC 생성을 위한 yaml 작성 및 실행
- 이때, nfs의 path와 NFS 서버의 IP를 지정 (DB VM)
- PVC Bound 상태 확인

| ON-PREMISE | Nginx Proxy Manager |
|---|---|

```
[root@Onpremise ~]# cat docker-compose.yaml
version: '3.8'
services:
  app:
    image: 'jc21/nginx-proxy-manager:latest'
    restart: unless-stopped
    ports:
      - '80:80'
      - '59081:81'
      - '443:443'
    volumes:
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
```



Nginx Proxy Manager 설치
- 웹 기반 관리
- 도메인 및 서브도메인 관리
- SSL/TLS 인증서 관리
- 웹서버 및 프록시 서버 설정을 더욱 쉽게 관리하고 보안 강화
- docker-compose.yaml 작성 및 실행하여 설치
- 관리모드 접속포트는 59081/tcp로 지정

| ON-PREMISE | Nextcloud - yaml |
|---|---|

```
[root@master01 ~]# ls -l nextcloud-yaml/
total 12
-rw-r--r--. 1 root root  302 Sep 20 16:10 hpa.yaml
-rw-r--r--. 1 root root 1700 Sep 20 15:47 nextcloud.yaml
-rw-r--r--. 1 root root  645 Sep 20 14:07 pv-pvc.yaml
```

```
[root@master01 nextcloud-yaml]# cat pv-pvc.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-nextcloud
  namespace: ns-nextcloud
spec:
  capacity:
    storage: 30Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: nextcloud
  mountOptions:
    - nfsvers=4.2
  nfs:
    path: /web/nextcloud
    server: 192.168.100.103
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-nextcloud
  namespace: ns-nextcloud
spec:
  accessModes:
    - ReadWriteOnce
    - ReadWriteMany
  volumeMode: Filesystem
  resources:
    requests:
      storage: 30Gi
  storageClassName: nextcloud
```

```
[root@master01 nextcloud-yaml]# cat nextcloud.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nextcloud-deployment
  namespace: ns-nextcloud
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nextcloud
  template:
    metadata:
      labels:
        app: nextcloud
    spec:
      containers:
        - env:
            - name: APC_SHM_SIZE
              value: 256M
            - name: CRON_PERIOD
              value: 10m
            - name: NEXTCLOUD_MAX_TIME
              value: "3600"
            - name: NEXTCLOUD_MEMORY_LIMIT
              value: 1024M
            - name: NEXTCLOUD_TRUSTED_DOMAINS
              value: cloud.kkangtae.store
            #- name: OVERWRITEHOST
            #  value: cloud.ddomung.site
            #- name: OVERWRITEPROTOCOL
            #  value: https
            - name: PHP_MEMORY_LIMIT
              value: 1024M
            - name: PHP_UPLOAD_LIMIT
              value: 10G
            - name: TZ
              value: Asia/Seoul
            - name: UPLOAD_MAX_SIZE
              value: 10G
          name: nextcloud
          image: nextcloud
          ports:
            - containerPort: 80
          volumeMounts:
            - name: nextcloud-storage-volume
              mountPath: /var/www/html
          resources:
            limits:
              cpu: 1000m
            requests:
              cpu: 500m
      volumes:
        - name: nextcloud-storage-volume
          persistentVolumeClaim:
            claimName: pvc-nextcloud
```

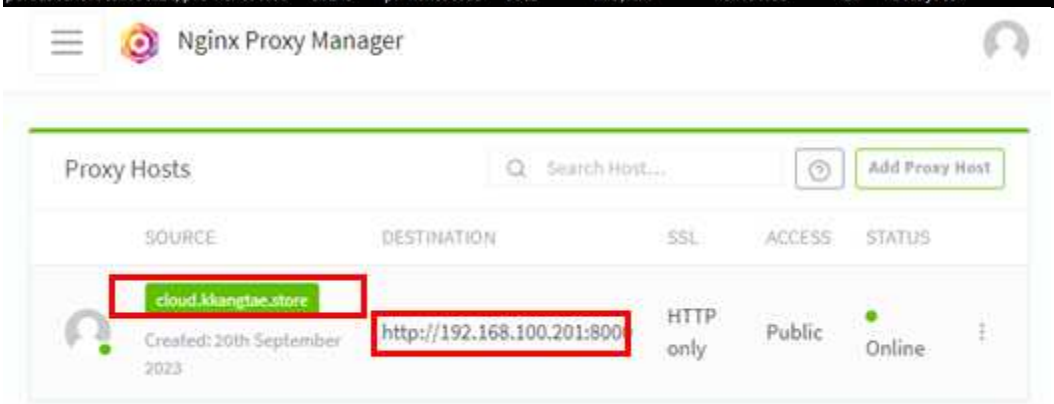
```
[root@master01 nextcloud-yaml]# cat hpa.yaml
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: autoscale-nextcloud
  namespace: ns-nextcloud
spec:
  maxReplicas: 10
  minReplicas: 1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nextcloud-deployment
  targetCPUUtilizationPercentage: 50
```

```
---
apiVersion: v1
kind: Service
metadata:
  name: nextcloud-service
  namespace: ns-nextcloud
spec:
  type: LoadBalancer
  selector:
    app: nextcloud
  ports:
  - name: nextcloud-service-port
    protocol: TCP
    port: 8000
    targetPort: 80
```

- Nextcloud 웹서비스를 위한 3개의 yaml 생성
- hpa.yaml : 리소스 사용량에 따라 최소 1개 ~ 최대 10개의 Pod 생성 설정
- nextcloud.yaml : Deployment, LoadBalancer Service 등 지정
- pv-pvc.yaml : 용량 및 NFS mount 지정(DB)
- Namespace : ns-nextcloud

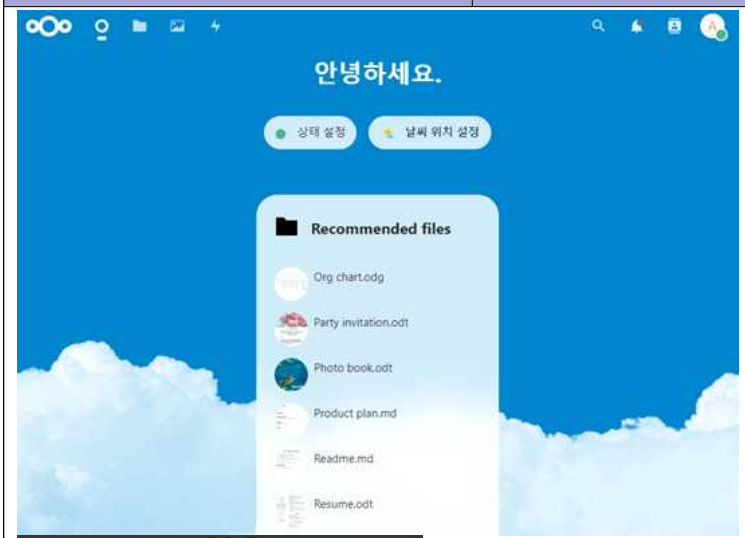
| ON-PREMISE | Nextcloud – Proxy Hosts |
|---|---|

```
Every 1.0s: kubectl get po,rs,deploy,svc,pv,pvc -n ns-nextcloud -o wide                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                
master01: Wed Sep

NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES
pod/nextcloud-deployment-77d5954b4d-l2vsx 1/1 Running 0 42m 192.168.101.208 worker01 <none> <none>

NAME DESIRED CURRENT READY AGE CONTAINERS IMAGES SELECTOR
replicaset.apps/nextcloud-deployment-77d5954b4d 1 1 1 42m nextcloud nextcloud app=nextcloud,pod-template-hash=77d5954b4d

NAME READY UP-TO-DATE AVAILABLE AGE CONTAINERS IMAGES SELECTOR
deployment.apps/nextcloud-deployment 1/1 1 1 42m nextcloud nextcloud app=nextcloud

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE SELECTOR
service/nextcloud-service LoadBalancer 10.110.249.243 192.168.100.201 8000:30131/TCP 42m app=nextcloud

NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE VOLUMEMODE
persistentvolume/pv-nextcloud 30Gi RWO,RWX Recycle Bound ns-nextcloud/pvc-nextcloud nextcloud 42m Filesystem

NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE VOLUMEMODE
persistentvolumeclaim/pvc-nextcloud Bound pv-nextcloud 30Gi RWO,RWX nextcloud 42m Filesystem
```



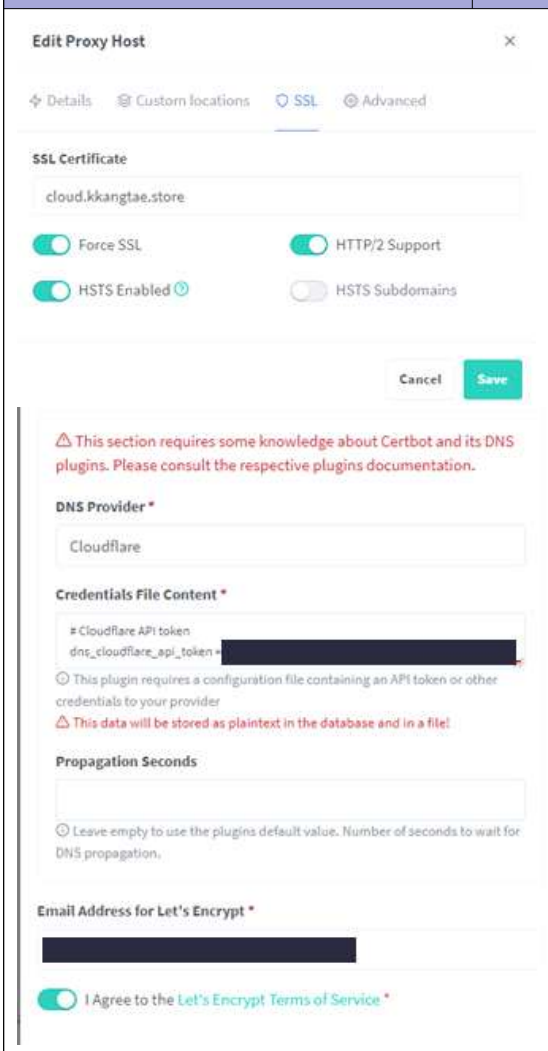
- Nginx Proxy Manager에 접속하여 Nextcloud 로드밸런서 서비스의 외부 IP를 목적지로 지정한 Proxy Host를 생성 (cloud.kkangtae.store)
- Service port인 8000/tcp를 사용하여 Nextcloud 접속

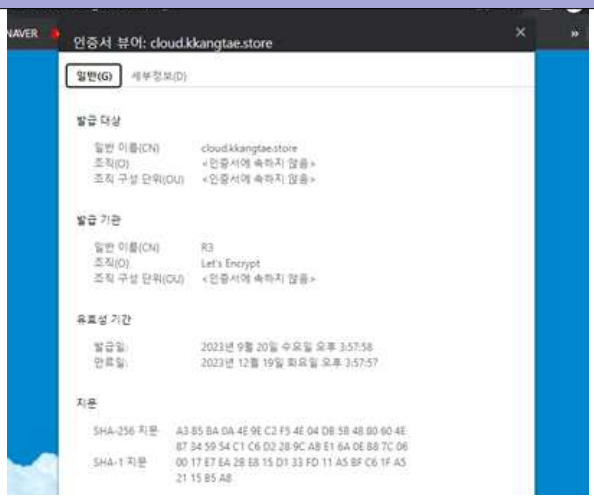
| ON-PREMISE | Nextcloud – 접속 및 구축 |
|---|---|



- 데이터베이스 호스트 : DB VM의 IP
- DB VM에서 생성한 데이터베이스(nextcloud)와 사용자(nextuser) 지정

| ON-PREMISE | Nextcloud – HTTPS 접속 |
|---|---|




- Let's Encrypt를 이용하여 와일드카드 인증서 발급
- 이때, Cloudflare에 등록된 도메인(kkangtae.store)의 API token을 입력
- HTTPS 보안 웹접속 가능

| ON-PREMISE | Nextcloud – Auto Scaling |
|---|---|

```
root@nextcloud-deployment-77d5954b4d-l2vsx:/var/www/html# stress --vm 2 -t 60s
stress: info: [214] dispatching hogs: 0 cpu, 0 io, 2 vm, 0 hdd
stress: info: [214] successful run completed in 60s
Every 1.0s: kubectl get po,rs,deploy,svc,pv,pvc -n ns-nextcloud -o wide                                                    master01: Wed Sep 2

NAME                                                   READY   STATUS    RESTARTS   AGE   IP               NODE       NOMINATED NODE   READINESS GATES
pod/nextcloud-deployment-77d5954b4d-282t7              1/1     Running   0          31s   192.168.101.90   worker02   <none>           <none>
pod/nextcloud-deployment-77d5954b4d-5g96f              1/1     Running   0          16s   192.168.101.209  worker01   <none>           <none>
pod/nextcloud-deployment-77d5954b4d-6xhnm              1/1     Running   0          31s   192.168.101.134  master01   <none>           <none>
pod/nextcloud-deployment-77d5954b4d-l2vsx              1/1     Running   0          96m   192.168.101.208  worker01   <none>           <none>

NAME                                              DESIRED   CURRENT   READY   AGE   CONTAINERS   IMAGES      SELECTOR
replicaset.apps/nextcloud-deployment-77d5954b4d   4         4         4       96m   nextcloud    nextcloud   app=nextcloud,pod-template-hash=77d5954b4d

NAME                                 READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES      SELECTOR
deployment.apps/nextcloud-deployment   4/4     4            4           96m   nextcloud    nextcloud   app=nextcloud

NAME                        TYPE           CLUSTER-IP       EXTERNAL-IP       PORT(S)          AGE   SELECTOR
service/nextcloud-service   LoadBalancer   10.110.249.243   192.168.100.201   8000:30131/TCP   96m   app=nextcloud

NAME                                   CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                          STORAGECLASS   REASON   AGE   VOLUMEMODE
persistentvolume/pv-nextcloud          30Gi       RWO,RWX        Recycle          Bound    ns-nextcloud/pvc-nextcloud     nextcloud               96m   Filesystem

NAME                                     STATUS   VOLUME         CAPACITY   ACCESS MODES   STORAGECLASS   AGE   VOLUMEMODE
persistentvolumeclaim/pvc-nextcloud      Bound    pv-nextcloud   30Gi       RWO,RWX        nextcloud      96m   Filesystem
Every 1.0s: kubectl get po,rs,deploy,svc,pv,pvc -n ns-nextcloud -o wide                                                    master01: Wed Sep 2

NAME                                                   READY   STATUS    RESTARTS   AGE     IP               NODE       NOMINATED NODE   READINESS GATES
pod/nextcloud-deployment-77d5954b4d-6xhnm              1/1     Running   0          8m16s   192.168.101.134  master01   <none>           <none>

NAME                                              DESIRED   CURRENT   READY   AGE    CONTAINERS   IMAGES      SELECTOR
replicaset.apps/nextcloud-deployment-77d5954b4d   1         1         1       104m   nextcloud    nextcloud   app=nextcloud,pod-template-hash=77d5954b4d

NAME                                 READY   UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES      SELECTOR
deployment.apps/nextcloud-deployment   1/1     1            1           104m   nextcloud    nextcloud   app=nextcloud

NAME                        TYPE           CLUSTER-IP       EXTERNAL-IP       PORT(S)          AGE    SELECTOR
service/nextcloud-service   LoadBalancer   10.110.249.243   192.168.100.201   8000:30131/TCP   104m   app=nextcloud

NAME                                   CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                          STORAGECLASS   REASON   AGE    VOLUMEMODE
persistentvolume/pv-nextcloud          30Gi       RWO,RWX        Recycle          Bound    ns-nextcloud/pvc-nextcloud     nextcloud               104m   Filesystem

NAME                                     STATUS   VOLUME         CAPACITY   ACCESS MODES   STORAGECLASS   AGE    VOLUMEMODE
persistentvolumeclaim/pvc-nextcloud      Bound    pv-nextcloud   30Gi       RWO,RWX        nextcloud      104m   Filesystem
```

- 임의의 Pod에서 CPU 부하테스트 진행 (2개 Process, 60초간)
- 부하 증가 : Pod 1개에서 4개로 증가
- 부하테스트 종료 : 기존 Pod 1개로 복구

| ON-PREMISE | Docker Build GNU – PHP config |
|---|---|

```
[root@DB ~]# cat web/conf/php/www.conf
[www]
user = www-data
group = www-data

listen = 9000
listen.owner = www-data
listen.group = www-data

pm = dynamic
pm.max_children = 30
pm.start_servers = 5
pm.min_spare_servers = 5
pm.max_spare_servers = 10
pm.max_requests = 500
```

```
[root@DB ~]# cat web/conf/php/php.ini
[PHP]
engine = On
short_open_tag = On
precision = 14
output_buffering = 4096
zlib.output_compression = Off
implicit_flush = Off
unserialize_callback_func =
serialize_precision = -1
disable_functions =
```

```
[root@DB ~]# tree web
web
└── conf
    ├── nginx
    │   ├── conf.d
    │   │   └── default.conf
    │   └── nginx.conf
    └── php
        ├── php.ini
        └── www.conf
```

- DB VM의 공유디렉토리 '/web'에 PHP conf 파일 생성

| ON-PREMISE | Docker Build GNU – PHP Dockerfile |
|---|---|

```
[root@DB ~]# cat Dockerfile
# FROM image이름:tag 해당 이미지를 사용
FROM php:8.2-fpm
# LABEL은 docker inspect 또는 docker info를 통해 해당이미지의 간략한 정보 작성
LABEL maintainer="meongchung@meonchung.shop"
LABEL issue_date=""
LABEL description="php-fpm module add"
# RUN은 이미지에서 특정 명령을 수행하는 것으로 가능한 하나로 합쳐서 사용하는 것이 좋음
# RUN은 Layer가 되어서 여러 명령을 RUN으로 구분하면 Layer가 너무 많아지는 문제 발생
RUN apt update \
    && apt install -y \
    libmagickwand-dev --no-install-recommends \
    libicu-dev \
    libonig-dev \
    && printf "\n" | pecl install imagick \
    && docker-php-ext-enable imagick \
    && docker-php-ext-install mysqli \
    && docker-php-ext-install pdo_mysql \
    && docker-php-ext-install iconv \
    && docker-php-ext-install intl \
    && docker-php-ext-install opcache \
    && docker-php-ext-install mbstring \
    && apt update -y

# COPY는 php.ini와 www.conf 같은 환경설정 파일을 host에 작성한 것을 build 시 추가
COPY ./web/conf/php/php.ini /usr/local/etc/php/php.ini
COPY ./web/conf/php/www.conf /usr/local/etc/php-fpm.d/www.conf
# EXPOSE 명령은 컨테이너 port를 open
EXPOSE 9000
# WORKDIR은 docker 접속 시 해당 경로로 바로 접근하고 컨테이너에서 명령어가 실행될 때의 default 위치
WORKDIR /root
# 컨테이너 실행 시 실행될 명령어
CMD ["php-fpm"]
```

- PHP 이미지를 실행하는 Dockerfile 생성
- 이때, PHP conf 파일을 복사하는 스크립트 내용 추가

| ON-PREMISE | Docker Build GNU – PHP 이미지 Build & Push |
|---|---|

```
[root@DB ~]# docker images
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
php-8/php           8.2-fpm   fe87a163fd38   15 minutes ago   752MB
meongchung/php-8.2  8.2-fpm   fe87a163fd38   15 minutes ago   752MB
```

```
root@DB ~]# docker push  meongchung/php-8.2:8.2-fpm
The push refers to repository [docker.io/meongchung/php-8.2]
5f70bf18a086: Mounted from meongchung/php-8
a509caf6cfef: Pushed
4b5b0de77fe6: Pushed
100ff102f9d8: Pushed
c925048cfc5b: Mounted from meongchung/php-8
9d2c2f7ef22b: Mounted from meongchung/php-8
ffb257a6a684: Mounted from meongchung/php-8
bfe0089ee16d: Mounted from meongchung/php-8
1775f7f20de4: Mounted from meongchung/php-8
1283d7cc1b75: Mounted from meongchung/php-8
be1b5a500198: Mounted from meongchung/php-8
e2de06477579: Mounted from meongchung/php-8
1ff185c2e955: Mounted from meongchung/php-8
a2d7501dfb35: Mounted from meongchung/php-8
8.2-fpm: digest: sha256:7a3b4458c67a34abb0c36e343109bb170c52c6c222eafded639373c4445b46fe size: 3244
```

- 생성된 PHP 이미지를 Docker hub의 개인 Repository로 Push (meongchung)

| ON-PREMISE | Docker Build GNU – Nginx config 생성 |
|---|---|

```
[root@DB ~]# cat web/conf/nginx/nginx.conf
user www-data;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    #access_log  /var/log/nginx/access.log  main;

    sendfile            on;
    tcp_nopush          on;
    keepalive_timeout   65;
    types_hash_max_size 4096;

    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;
}
[root@DB ~]# cat web/conf/nginx/conf.d/default.conf
server {
    listen       80;
    listen       [::]:80;
    server_name  _;
    root         /usr/share/nginx/html;

    location / {
        index index.php index.html index.htm;
        try_files $uri $uri/ /index.php$uri?$query_string;
    }

    # Load configuration files for the default server block.
    include /etc/nginx/default.d/*.conf;
    error_page 404 /404.html;
    location = /404.html {
    }

    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    }

    location ~ [^/].php(/|$) {
        fastcgi_split_path_info ^(.+?\.php)(/.*)$;
        set $path_info $fastcgi_path_info;
        fastcgi_index index.php;
        include fastcgi_params;
        fastcgi_pass 10.97.112.94:9000;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
    }

    # Deny access to .htaccess files
    location ~ /.ht {
        deny all;
    }
}
```

- DB VM의 '/web' 공유디렉토리에 Nginx conf 파일 생성
- 이때, default.conf에서 연동할 PHP cluster 고정 IP와 9000 포트 지정

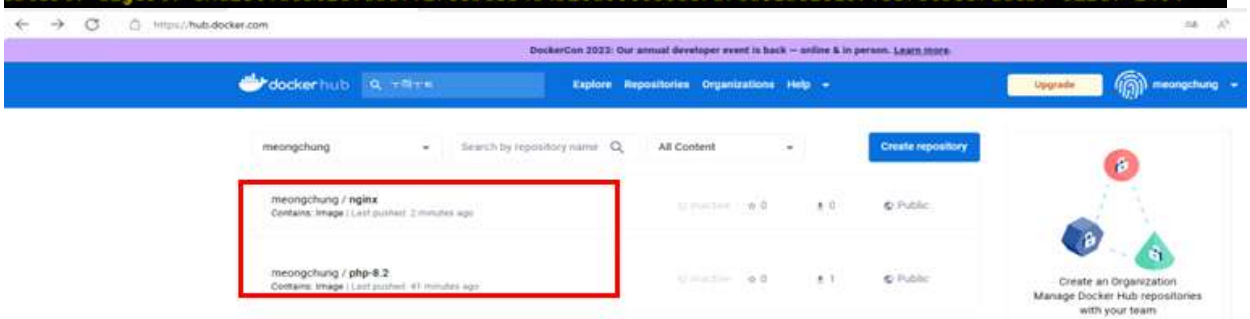
| ON-PREMISE | Docker Build GNU – Nginx Dockerfile |
|---|---|

```
[root@DB ~]# cat Dockerfile
FROM nginx:latest
LABEL maintainer="meongchung@meonchung.shop"
LABEL description="nginx with config"
RUN apt update \
    && apt install vim -y
COPY ./web/conf/nginx/nginx.conf /etc/nginx/nginx.conf
COPY ./web/conf/nginx/conf.d/default.conf /etc/nginx/conf.d/default.conf
VOLUME [/etc/nginx/conf.d]
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

- Nginx 이미지를 실행하는 Dockerfile 생성
- PHP와 연동하는 conf 적용
- 이때, Nginx conf 파일을 복사하는 스크립트 내용 추가 --> 해당 이미지 안에 conf 파일 포함하려는 목적

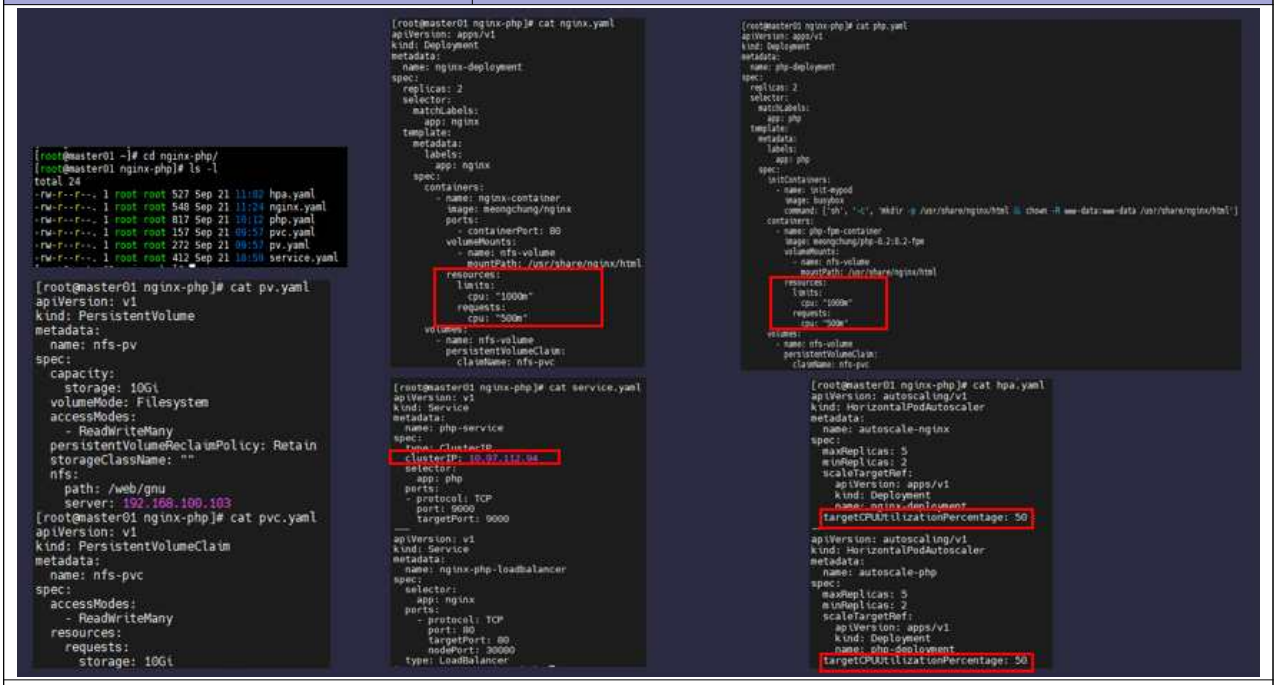
| ON-PREMISE | Docker Build GNU – 이미지 Build & Push |
|---|---|

```
[root@DB ~]# docker images
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
meongchung/nginx    latest    2ecb26d578b4   39 seconds ago   248MB
[root@DB ~]# docker push  meongchung/nginx:latest
The push refers to repository [docker.io/meongchung/nginx]
dc14c0659a2d: Pushed
39a447d40cff: Pushed
64af2b7d18bc: Pushed
aae231785348: Mounted from library/nginx
e48f2ce44b27: Mounted from library/nginx
3bfd54ea739a: Mounted from library/nginx
0f4045499bea: Mounted from library/nginx
1b34d645672f: Mounted from library/nginx
c74e4ebd2844: Mounted from library/nginx
a2d7501dfb35: Mounted from meongchung/php-8.2
latest: digest: sha256:ad5c2efda8ff2783be354d4b2ca00e6cc6d7a7dd81ae18197f53730b557dd6b7 size: 2404
```




- 생성된 Nginx 이미지를 Docker hub의 개인 Repository로 Push

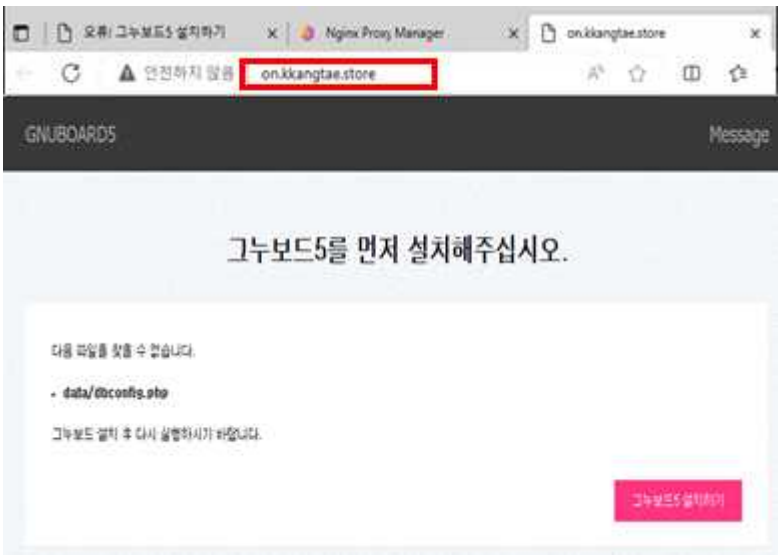
| ON-PREMISE | Gnuboard - yaml |
|---|---|




- Kubernetes로 Nginx와 PHP Cluster를 활용한 Gnuboard 구축
- 6개의 yaml 생성
- PHP Cluster IP 고정, 리소스 제한, AutoScaling 기능 등 추가 및 구현
- 이때, CPU 사용률이 50% 이상이 되면 Pod는 최대 5개까지 증가할 수 있도록 지정 (부하분산)

| ON-PREMISE | Gnuboard – Proxy Hosts |
|---|---|

```
[root@master01 nginx-php]# kubectl get po,rs,deploy,svc,pv,pvc -o wide
NAME                                          READY   STATUS    RESTARTS   AGE     IP               NODE       NOMINATED NODE   READINESS GATES
pod/nginx-deployment-74668f8c88-4f7mw         1/1     Running   0          9m49s   192.168.101.218  worker01   <none>           <none>
pod/nginx-deployment-74668f8c88-ftkcg         1/1     Running   0          9m34s   192.168.101.137  master01   <none>           <none>
pod/php-deployment-64dd6c496f-9nhsc           1/1     Running   0          9m49s   192.168.101.98   worker02   <none>           <none>
pod/php-deployment-64dd6c496f-r865k           1/1     Running   0          9m34s   192.168.101.219  worker01   <none>           <none>

NAME                                    DESIRED   CURRENT   READY   AGE     CONTAINERS          IMAGES                          SELECTOR
replicaset.apps/nginx-deployment-74668f8c88   2         2         2       9m49s   nginx-container     meongchung/nginx                app=nginx,pod-template-hash=74668f8c88
replicaset.apps/php-deployment-64dd6c496f     2         2         2       9m49s   php-fpm-container   meongchung/php-8.2:0.2-fpm      app=php,pod-template-hash=64dd6c496f

NAME                               READY   UP-TO-DATE   AVAILABLE   AGE     CONTAINERS          IMAGES                          SELECTOR
deployment.apps/nginx-deployment   2/2     2            2           9m49s   nginx-container     meongchung/nginx                app=nginx
deployment.apps/php-deployment     2/2     2            2           9m49s   php-fpm-container   meongchung/php-8.2:0.2-fpm      app=php

NAME                 TYPE           CLUSTER-IP     EXTERNAL-IP      PORT(S)        AGE     SELECTOR
service/kubernetes   ClusterIP      10.96.0.1      <none>           443/TCP        44h     <none>
service/nginx-php-loadbalancer   LoadBalancer   10.106.96.30   192.168.100.200   80:30080/TCP   9m49s   app=nginx
service/php-service  ClusterIP      10.97.112.94   <none>           9000/TCP       9m49s   app=php

NAME                   CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE     VOLUMEMODE
persistentvolume/nfs-pv   10Gi       RWX            Retain           Bound    default/nfs-pvc                            9m49s   Filesystem

NAME                      STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE     VOLUMEMODE
persistentvolumeclaim/nfs-pvc   Bound    nfs-pv   10Gi       RWX                           9m49s   Filesystem
```




- PHP와 연동된 Nginx 웹서비스 실행
- 이때, Nginx Proxy Manager에 접속하여 로드밸런서 외부 IP 192.168.100.200으로 Proxy Host 추가
- on.kkangtae.store 라는 도메인 이름으로 Gnuboard 접속 확인

| ON-PREMISE | Gnuboard – AutoScaling |
|---|---|

```
root@nginx-deployment-76db77d77f-9w6zm:/# stress --vm 1 -t 1800s
stress: info: [63] dispatching hogs: 0 cpu, 0 io, 1 vm, 0 hdd

Every 1.0s: kubectl top node

NAME        CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
master01    374m         18%    2410Mi          63%
worker01    1039m        51%    2118Mi          55%
worker02    43m          2%     1819Mi          47%
```

```
Every 1.0s: kubectl get po,svc,rs,deploy,pv,pvc -o wide

NAME                                      READY   STATUS    RESTARTS   AGE     IP                NODE       NOMINATED NODE   READINESS GATES
pod/nginx-deployment-76db77d77f-9w6zm     1/1     Running   0          3m54s   192.168.101.129   worker01   <none>           <none>
pod/nginx-deployment-76db77d77f-j4kmp     1/1     Running   0          3m58s   192.168.101.96    worker02   <none>           <none>
pod/nginx-deployment-76db77d77f-td8lw     1/1     Running   0          36s     192.168.101.130   worker01   <none>           <none>
pod/nginx-deployment-76db77d77f-v2tcr     1/1     Running   0          36s     192.168.101.193   master01   <none>           <none>
                                          1/1     Running   0          3h19m   192.168.101.98    worker02   <none>           <none>
pod/php-deployment-64dd6c496f-r865k       1/1     Running   0          3h19m   192.168.101.219   worker01   <none>           <none>

NAME                          TYPE           CLUSTER-IP     EXTERNAL-IP       PORT(S)        AGE    SELECTOR
service/kubernetes            ClusterIP      10.96.0.1      <none>            443/TCP        47h    <none>
service/nginx-php-loadbalancer LoadBalancer  10.106.96.30   192.168.100.200   80:30080/TCP   3h19m  app=nginx
service/php-service           ClusterIP      10.97.112.94   <none>            9000/TCP       3h19m  app=php

NAME                                       DESIRED   CURRENT   READY   AGE     CONTAINERS          IMAGES                          SELECTOR
replicaset.apps/nginx-deployment-74668f8c88   0         0         0       3h19m   nginx-container     meongchung/nginx                app=nginx,
replicaset.apps/nginx-deployment-76db77d77f   4         4         4       3m58s   nginx-container     meongchung/nginx                app=nginx,
replicaset.apps/php-deployment-64dd6c496f     2         2         2       3h19m   php-fpm-container   meongchung/php-8.2:8.2-fpm      app=php,pc

NAME                               READY   UP-TO-DATE   AVAILABLE   AGE     CONTAINERS          IMAGES                          SELECTOR
deployment.apps/nginx-deployment   4/4     4            4           3h19m   nginx-container     meongchung/nginx                app=nginx
deployment.apps/php-deployment     2/2     2            2           3h19m   php-fpm-container   meongchung/php-8.2:8.2-fpm      app=php

NAME                   CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                 STORAGECLASS   REASON   AGE     VOLUMEMODE
persistentvolume/nfs-pv   10Gi       RWX            Retain           Bound    default/nfs-pvc                              3h19m   Filesystem

NAME                      STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE     VOLUMEMODE
persistentvolumeclaim/nfs-pvc   Bound    nfs-pv   10Gi       RWX                           3h19m   Filesystem
```

```
Every 1.0s: kubectl get po,svc,rs,deploy,pv,pvc -o wide

NAME                                      READY   STATUS    RESTARTS   AGE     IP                NODE       NOMINATED NODE   READINESS GATES
pod/nginx-deployment-76db77d77f-j4kmp     1/1     Running   0          11m     192.168.101.96    worker02   <none>           <none>
pod/nginx-deployment-76db77d77f-v2tcr     1/1     Running   0          7m52s   192.168.101.193   master01   <none>           <none>
                                          1/1     Running   0          3h26m   192.168.101.98    worker02   <none>           <none>
pod/php-deployment-64dd6c496f-r865k       1/1     Running   0          3h26m   192.168.101.219   worker01   <none>           <none>

NAME                          TYPE           CLUSTER-IP     EXTERNAL-IP       PORT(S)        AGE    SELECTOR
service/kubernetes            ClusterIP      10.96.0.1      <none>            443/TCP        47h    <none>
service/nginx-php-loadbalancer LoadBalancer  10.106.96.30   192.168.100.200   80:30080/TCP   3h26m  app=nginx
service/php-service           ClusterIP      10.97.112.94   <none>            9000/TCP       3h26m  app=php

NAME                                       DESIRED   CURRENT   READY   AGE     CONTAINERS          IMAGES                          SELECTOR
replicaset.apps/nginx-deployment-74668f8c88   0         0         0       3h26m   nginx-container     meongchung/nginx                app=nginx
replicaset.apps/nginx-deployment-76db77d77f   2         2         2       11m     nginx-container     meongchung/nginx                app=nginx
replicaset.apps/php-deployment-64dd6c496f     2         2         2       3h26m   php-fpm-container   meongchung/php-8.2:8.2-fpm      app=php,p

NAME                               READY   UP-TO-DATE   AVAILABLE   AGE     CONTAINERS          IMAGES                          SELECTOR
deployment.apps/nginx-deployment   2/2     2            2           3h26m   nginx-container     meongchung/nginx                app=nginx
deployment.apps/php-deployment     2/2     2            2           3h26m   php-fpm-container   meongchung/php-8.2:8.2-fpm      app=php

NAME                   CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                 STORAGECLASS   REASON   AGE     VOLUMEMODE
persistentvolume/nfs-pv   10Gi       RWX            Retain           Bound    default/nfs-pvc                              3h26m   Filesystem

NAME                      STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   AGE     VOLUMEMODE
persistentvolumeclaim/nfs-pvc   Bound    nfs-pv   10Gi       RWX                           3h26m   Filesystem
```

- 임의의 Nginx Pod에 Stress 명령어를 이용하여 CPU 부하 테스트를 진행
- 부하가 증가함에 따라 Pod가 4개까지 증가함
- 부하테스트 종료 후 기존의 Pod 2개로 복구됨

| ON-PREMISE | Gnuboard – Rolling Update |
|---|---|

```
[root@master01 nginx-php]# kubectl exec -it pod/nginx-deployment-76db77d77f-j4kmp -- nginx -v
nginx version: nginx/1.25.2
[root@master01 nginx-php]#
[root@master01 nginx-php]# kubectl exec -it pod/nginx-deployment-76db77d77f-v2tcr -- nginx -v
nginx version: nginx/1.25.2
```

```
[root@master01 nginx-php]# kubectl set image deploy nginx-deployment nginx-container=nginx:1.16
deployment.apps/nginx-deployment image updated
```

```
[root@master01 nginx-php]# kubectl get po,rs,deploy,svc --show-labels -o wide
NAME                                        READY   STATUS    RESTARTS   AGE     IP               NODE       NOMINATED NODE
pod/nginx-deployment-5b8b95488c-tbpdb       1/1     Running   0          9s      192.168.101.67   worker02   <none>
pod/nginx-deployment-5b8b95488c-xndfh       1/1     Running   0          16s     192.168.101.131  worker01   <none>
pod/php-deployment-64dd6c496f-9nnsc         1/1     Running   0          5h41m   192.168.101.98   worker02   <none>
pod/php-deployment-64dd6c496f-r865k         1/1     Running   0          5h41m   192.168.101.219  worker01   <none>

NAME                                        DESIRED   CURRENT   READY   AGE     CONTAINERS        IMAGES
replicaset.apps/nginx-deployment-5b8b95488c 2         2         2       16s     nginx-container   nginx:1.16
-hash=5b8b95488c
replicaset.apps/nginx-deployment-74668f8c88 0         0         0       5h41m   nginx-container   meongchung/nginx
-hash=74668f8c88
replicaset.apps/nginx-deployment-76db77d77f 0         0         0       146m    nginx-container   meongchung/nginx
-hash=76db77d77f
replicaset.apps/php-deployment-64dd6c496f   2         2         2       5h41m   php-fpm-container meongchung/php-8.
ash=64dd6c496f
```

- 기존의 Nginx 버전은 v1.25.2
- Nginx v1.16으로 Rolling Update를 진행하여 Pod 하나씩 순차적으로 버전 Update 진행

| ON-PREMISE | Gnuboard – RollBack |
|---|---|

```
[root@master01 nginx-php]# kubectl rollout undo deploy nginx-deployment
deployment.apps/nginx-deployment rolled back
```

```
[root@master01 nginx-php]# kubectl get po,rs,deploy,svc --show-labels -o wide
NAME                                        READY   STATUS        RESTARTS   AGE     IP               NODE       NOMINATED NODE   RE
pod/nginx-deployment-5b8b95488c-xndfh       1/1     Terminating   0          3m15s   192.168.101.131  worker01   <none>           <n
pod/nginx-deployment-76db77d77f-2lgpq       1/1     Running       0          4s      192.168.101.132  worker01   <none>           <n
pod/nginx-deployment-76db77d77f-vmlc7       1/1     Running       0          7s      192.168.101.194  master01   <none>           <n
pod/php-deployment-64dd6c496f-9nnsc         1/1     Running       0          5h44m   192.168.101.98   worker02   <none>           <n
pod/php-deployment-64dd6c496f-r865k         1/1     Running       0          5h44m   192.168.101.219  worker01   <none>           <n

[root@master01 nginx-php]# kubectl exec -it pod/nginx-deployment-76db77d77f-2lgpq -- nginx -v
nginx version: nginx/1.25.2
[root@master01 nginx-php]#
[root@master01 nginx-php]# kubectl exec -it pod/nginx-deployment-76db77d77f-vmlc7 -- nginx -v
nginx version: nginx/1.25.2
```

- 반대로 원래 버전(v1.25.2)로 복구하기 위한 RollBack 진행
- Pod 하나씩 순차적으로 Rollback 진행
- 모든 Nginx Pod의 버전이 다시 v1.25.2로 복구된 것을 확인

## ● Site-to-Site VPN

| ON-PREMISE | Libreswan |
|---|---|

```
[root@Onpremise ~]# ipsec verify
Verifying installed system and configuration files

Version check and ipsec on-path                         [OK]
Libreswan 4.9
Checking for IPsec support in kernel                    [OK]
 NETKEY: Testing XFRM related proc values
         ICMP default/send_redirects                    [OK]
         ICMP default/accept_redirects                  [OK]
         XFRM larval drop                               [OK]
Pluto ipsec.conf syntax                                 [OK]
Checking rp_filter                                      [OK]
Checking that pluto is running                          [OK]
 Pluto listening for IKE on udp 500                     [OK]
 Pluto listening for IKE/NAT-T on udp 4500              [OK]
 Pluto ipsec.secret syntax                              [OK]
Checking 'ip' command                                   [OK]
Checking 'iptables' command                             [OK]
Checking 'prelink' command does not interfere with FIPS [OK]
Checking for obsolete ipsec.conf options                [OK]
```

```
[root@Onpremise ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:e0:ab:a6 brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    inet 211.35.168.89/24 brd 211.35.168.255 scope global dynamic noprefixroute ens160
       valid_lft 6030sec preferred_lft 6030sec
    inet6 fe80::20c:29ff:fee0:aba6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:0c:29:e0:ab:b0 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 192.168.100.254/24 brd 192.168.100.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::5e14:1a20:a977:64c6/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

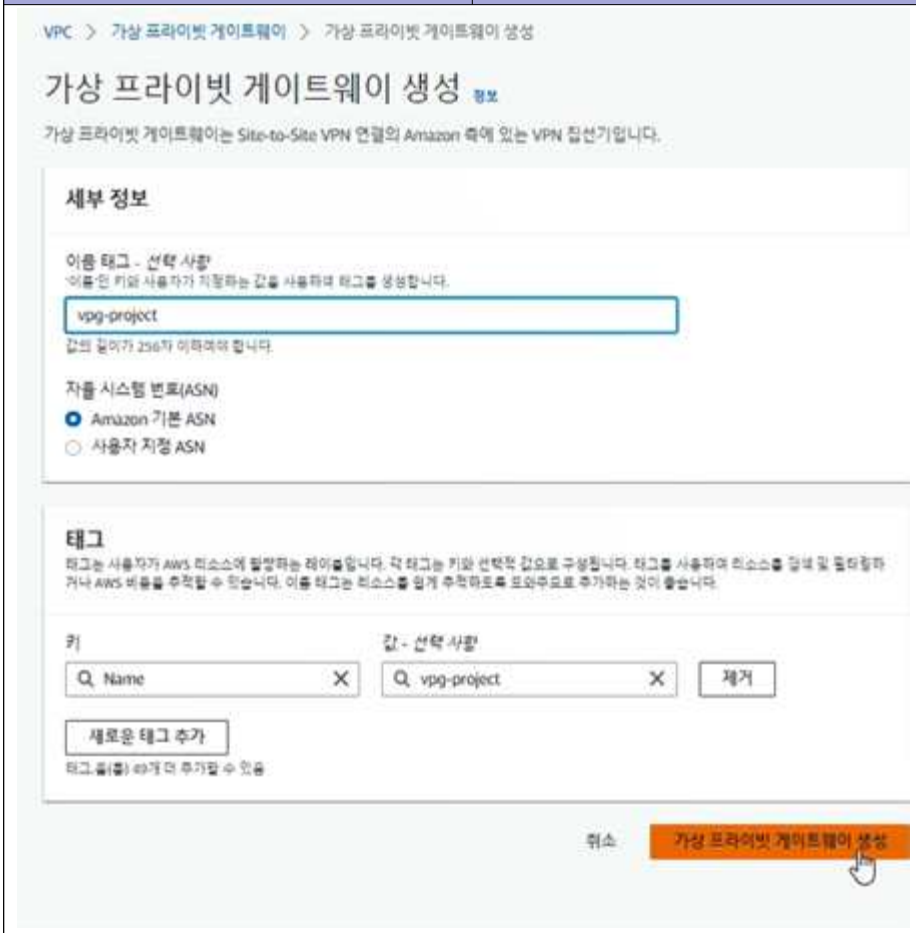
```
000 "aws-tunnel-1": 0.0.0.0/0===61.82.182.200---61.82.182.254...15.164.105.233<15.164.105.233>===0.0.0.0/0; erouted;
000 "aws-tunnel-1":     oriented; my_ip=unset; their_ip=unset; my_updown=ipsec _updown;
000 "aws-tunnel-1":   xauth us:none, xauth them:none,  my_username=[any]; their_username=[any]
000 "aws-tunnel-1":   our auth:secret, their auth:secret
000 "aws-tunnel-1":   modecfg info: us:none, them:none, modecfg policy:push, dns:unset, domains:unset, cat:unset;
000 "aws-tunnel-1":   sec_label:unset;
000 "aws-tunnel-1":   ike_life: 28800s; ipsec_life: 3600s; replay_window: 32; rekey_margin: 540s; rekey_fuzz: 100%;
000 "aws-tunnel-1":   retransmit-interval: 500ms; retransmit-timeout: 60s; iketcp:no; iketcp-port:4500;
000 "aws-tunnel-1":   initial-contact:no; cisco-unity:no; fake-strongswan:no; send-vendorid:no; send-no-esp-tfc:no;
000 "aws-tunnel-1":   policy: IKEv2+PSK+ENCRYPT+TUNNEL+PFS+UP+IKE_FRAG_ALLOW+ESN_NO;
000 "aws-tunnel-1":   v2-auth-hash-policy: none;
000 "aws-tunnel-1":   conn_prio: 0,0; interface: ens160; metric: 0; mtu: unset; sa_prio:auto; sa_tfc:none;
000 "aws-tunnel-1":   nflog-group: unset; mark: 5/0xffffffff, 5/0xffffffff; vti-iface:vti1; vti-routing:no; vti-shar
000 "aws-tunnel-1":   our idtype: ID_IPV4_ADDR; our id=61.82.182.200; their idtype: ID_IPV4_ADDR; their id=15.164.10
000 "aws-tunnel-1":   dpd: action:restart; delay:10; timeout:30; nat-t: encaps:yes; nat_keepalive:yes; ikev1_natt:bo
000 "aws-tunnel-1":   newest ISAKMP SA: #1; newest IPsec SA: #2; conn serial: $1;
000 "aws-tunnel-1":   IKE algorithms: AES_CBC_256-HMAC_SHA2_256-MODP2048
000 "aws-tunnel-1":   IKEv2 algorithm newest: AES_CBC_256-HMAC_SHA2_256-MODP2048
000 "aws-tunnel-1":   ESP algorithms: AES_GCM_16-NONE
000 "aws-tunnel-1":   ESP algorithm newest: AES_GCM_16_128-NONE; pfsgroup=<Phase1>
000
000 Total IPsec connections: loaded 1, active 1
000
000 State Information: DDoS cookies not required, Accepting new IKE connections
000 IKE SAs: total(1), half-open(0), open(0), authenticated(1), anonymous(0)
000 IPsec SAs: total(1), authenticated(1), anonymous(0)
000
000 #1: "aws-tunnel-1":4500 STATE_V2_ESTABLISHED_IKE_SA (established IKE SA); REKEY in 28042s; newest ISAKMP; idle;
000 #2: "aws-tunnel-1":4500 STATE_V2_ESTABLISHED_CHILD_SA (established Child SA); REKEY in 2601s; newest IPSEC; erou
000 #2: "aws-tunnel-1" esp.c3947aa1@15.164.105.233 esp.7d36b08a@61.82.182.200 tun.0@15.164.105.233 tun.0@61.82.182.2
000
000 Bare Shunt list:
000
[root@localhost ~]#
```

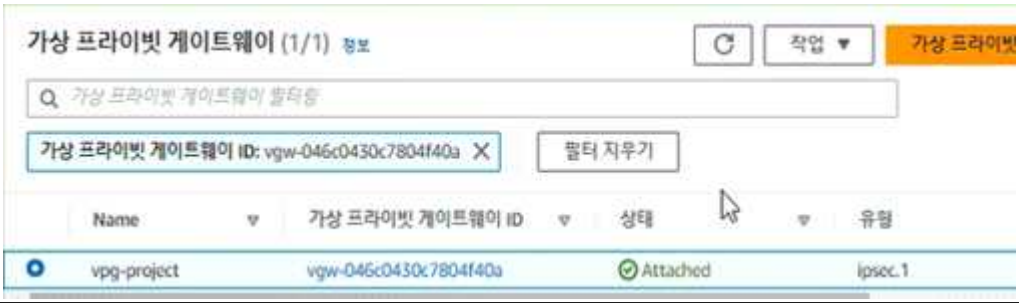
- IPsec 및 가상 사설 네트워크 VPN 연결을 위해 Libreswan 설정
- 'Onpremise' VM에서 진행

| ON-PREMISE | AWS - CGW |
|---|---|

- AWS에서 고객 게이트웨이 생성
- 이때 IP주소는 'Onpremise' VM의 Public IP를 지정

| ON-PREMISE | AWS - VPG |
|---|---|



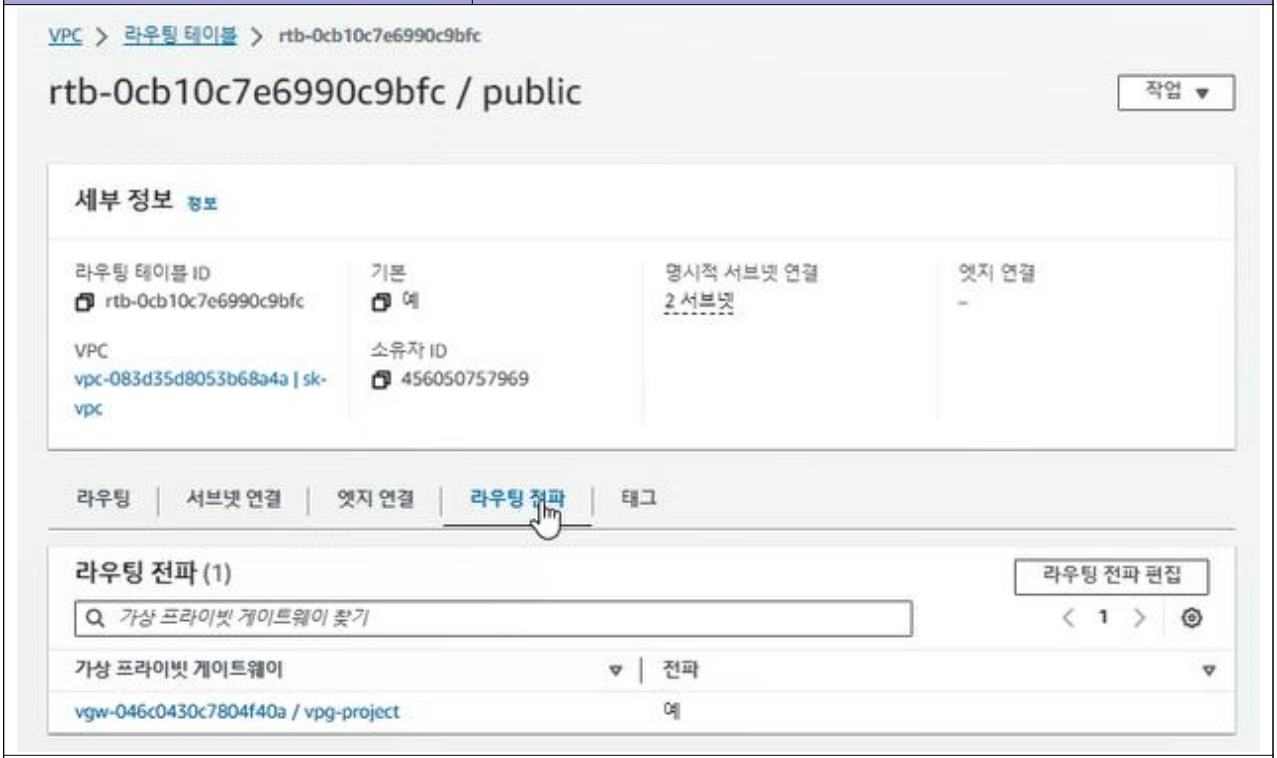




- 가상 사설 게이트웨이 생성
- AWS의 VPC와 연결하는 것이 목적

| ON-PREMISE | AWS – VPN 연결 |
|---|---|

- 온프레미스 네트워크와 AWS VPC 간의 안전한 통신을 위한 VPN 연결 생성
- 생성했던 VPG와 CGW를 지정
- 라우팅 옵션 : 동적(BGP)

| ON-PREMISE | AWS – 라우팅 전파 |
|---|---|




- 해당 라우팅 테이블에서 라우팅 전파 기능 활성화
- Public Cloud AWS와 온프레미스 네트워크 간의 원활한 통신
- VPG가 자동으로 라우팅 테이블에게 라우팅 정보를 전파
- 즉, 라우팅 테이블에 수동으로 VPN 라우팅 정보를 입력할 필요가 없음

| ON-PREMISE | BGP |
|---|---|

```
[root@Onpremise ~]# vtysh

Hello, this is FRRouting (version 7.5.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

Frr# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup

K>* 0.0.0.0/0 [0/100] via 211.35.168.254, ens160, src 211.35.168.89, 02:40:36
C>* 169.254.210.84/30 is directly connected, vti1, 02:05:32
C>* 172.18.0.0/16 is directly connected, br-1040b26c52e8, 02:40:34
K>* 192.168.0.0/22 [0/0] via 169.254.210.85, vti1, 01:52:35
B   192.168.0.0/22 [20/100] via 169.254.210.85, vti1, weight 1, 02:00:32
C>* 192.168.100.0/24 is directly connected, ens192, 02:40:36
C>* 211.35.168.0/24 is directly connected, ens160, 02:40:36
Frr# exit
```

- 'Onpremise' VM에서 FRR로 접근 (vtyos)
- BGP 라우팅 프로토콜 선언
- AWS Public IP 대역(169.254.210.84/30)과 Connected 연결로 통신 가능
- AWS Private IP 대역(192.168.0.0/22)과 BGP로 통신 가능

| ON-PREMISE | Test 1 |
|---|---|

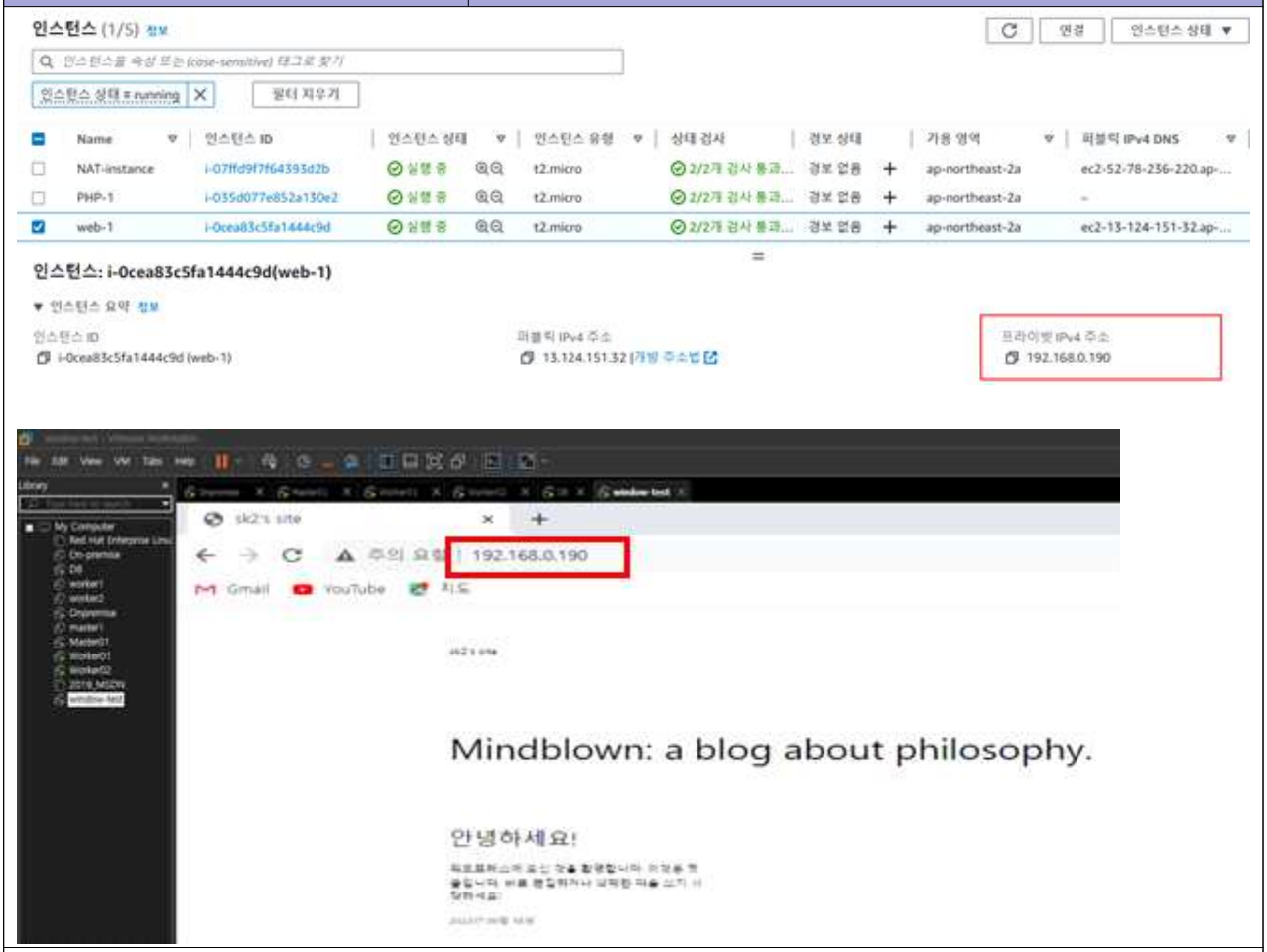
```
[root@Onpremise ~]# ping 192.168.1.97
PING 192.168.1.97 (192.168.1.97) 56(84) bytes of data.
64 bytes from 192.168.1.97: icmp_seq=1 ttl=254 time=6.34 ms
64 bytes from 192.168.1.97: icmp_seq=2 ttl=254 time=6.00 ms
64 bytes from 192.168.1.97: icmp_seq=3 ttl=254 time=6.64 ms
64 bytes from 192.168.1.97: icmp_seq=4 ttl=254 time=6.17 ms
```

```
[root@Onpremise ~]# ping 192.168.2.242
PING 192.168.2.242 (192.168.2.242) 56(84) bytes of data.
64 bytes from 192.168.2.242: icmp_seq=1 ttl=254 time=6.100 ms
64 bytes from 192.168.2.242: icmp_seq=2 ttl=254 time=6.47 ms
64 bytes from 192.168.2.242: icmp_seq=3 ttl=254 time=10.1 ms
64 bytes from 192.168.2.242: icmp_seq=4 ttl=254 time=8.77 ms
64 bytes from 192.168.2.242: icmp_seq=5 ttl=254 time=7.08 ms
```

```
[root@DB .ssh]# ls -l
total 8
-rw-------. 1 root root 1674 Sep 21 10:42 4_Group_key.pem
-rw-r--r--. 1 root root  175 Sep 21 10:45 known_hosts
[root@DB .ssh]# ssh -i 4_Group_key.pem ec2-user@192.168.0.190
Last login: Thu Sep 21 01:52:01 2023 from 169.254.210.86

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
No packages needed for security; 2 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-192-168-0-190 ~]$
```

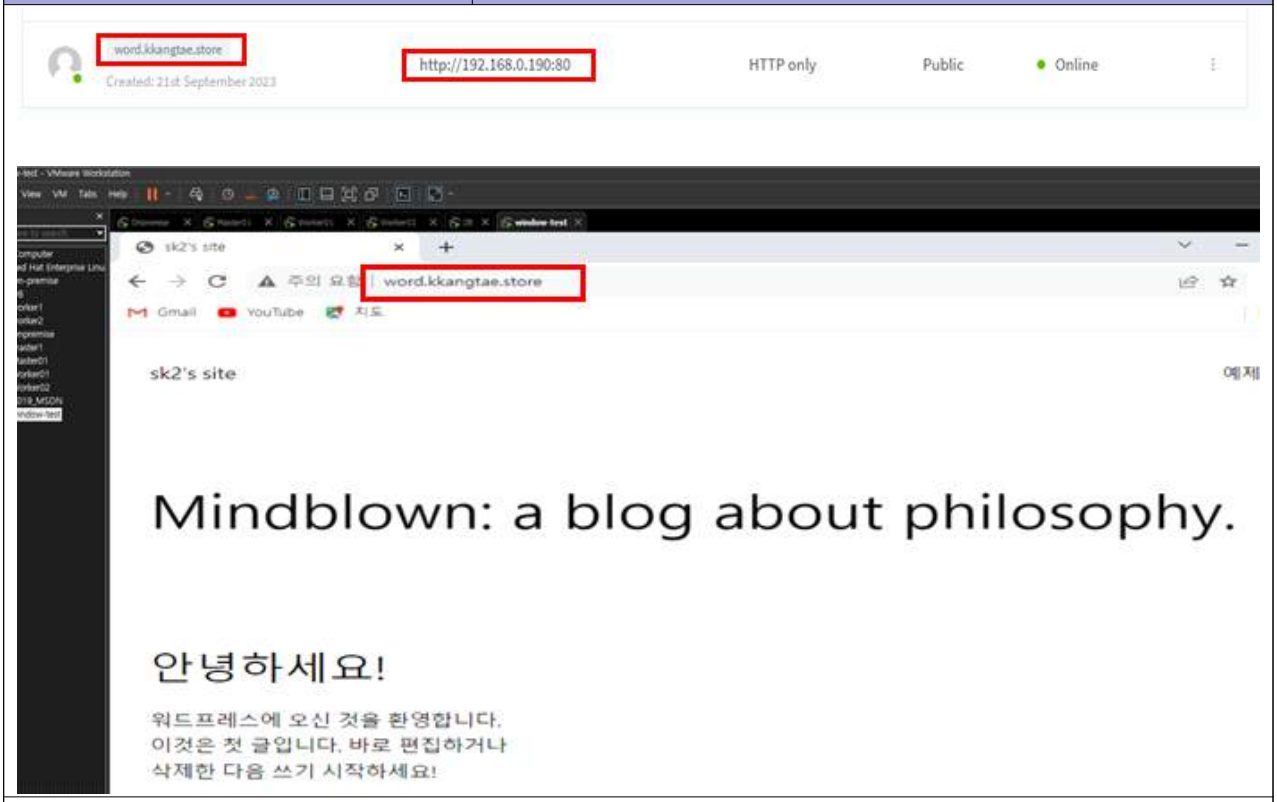
- ICMP ping test 진행
- 'Onpremise' VM에서 AWS 인스턴스의 Private IP로 통신 가능
- 동일한 key pair를 가지고 있는 상태에서 DB(VMnet1) VM에서 AWS 인스턴스(web-1)의 Private IP로 SSH 접속 test 진행 및 성공

| ON-PREMISE | Test 2 |
|---|---|




- 'web-1' AWS 인스턴스의 Private IP로 웹접속 테스트를 위해 온프레미스 환경의 VMnet1 네트워크만을 가진 임의의 Windows VM 생성 후 Test
- 정상적으로 Wordpress에 접속

| ON-PREMISE | Test 3 |
|---|---|




- Nginx Proxy Manager에 해당 Private IP로 Proxy Host를 생성 (word.kkangtae.store)
- 해당 도메인 이름으로 Wordpress 접속 가능

# 4. 프로젝트 일정

| 항목 | 세부 내용 | 어플리케이션 | 일정 | 담당 | 비고 |
|---|---|---|---|---|---|
| 요구 분석 | 요구사항 분석 | | 2023-08-21 ~ 2023-08-22 | 서희경 | |
| | 프로젝트 분석 및 설계 | Excel | 2023-08-21 ~ 2023-08-22 | 김학남 | |
| 설계 | AWS 구성 | Powerpoint | 2023-08-22 ~2023-08-23 | 김경태 | |
| | On-premise 구성 | Powerpoint | 2023-08-24 ~2023-08-25 | 서희경 | |
| 하이브리드 클라우드 구축 | AWS Network 구축 | VPC, EC2, RDS, Route53, EFS 등 | 2023-08-26 ~2023-09-06 | 김경태 | |
| | On-premise 환경 구축 | K8s, Docker, MariaDB, NFS, VMware, Linux 등 | 2023-09-07 ~ 2023-09-26 | 서희경 | |
| | VPN 구성 및 연결 | Site-to-Site VPN, FRR, BGP, IPsec, VMware, Linux 등 | 2023-09-27 ~ 2023-10-01 | 김학남 | |
| 테스트 | 웹서비스 접속 및 디버깅 | | 2023-10-01 ~ 2023-10-02 | 김학남 | |
| 보고서 작성, 발표 | | Powerpoint, Excel, hwp | 2023-10-03 ~ 2023-10-04 | 김경태 | |

# 5. 피드백

- 미달성 목표

가) HAproxy를 통한 WEB 이중화 구성 실패

- HAproxy 설치 및 테스트는 VMWARE Workstation Local Network에서 완료
- 이후, 인프라에 적용 시 134.100.X.X 대역과 연동 실패(원인 분석 중)

나) Youtube OPEN API를 활용한 Youtube 실시간 인기동영상 크롤링 기능 개발 실패

- 개발 스킬 부족으로 인한 기능 구현 실패

다) 커뮤니티 기능 중 게시판 기능 구현 실패

- 개발 스킬 및 백엔드 구성에 대한 이해 부족으로 기능 구현 실패

라) 운영 ↔ DR간 데이터 실시간 데이터 동기화 구현 실패

- Third Party App 없이 스케줄링으로 구현하기는 한계가 있어 대책 강구 필요.

- 향후 계획

가) Trouble Shooting을 통해 HAproxy Fault에 대한 원인 분석 완료 후,
재구성 하여 WEB 서버 이중화 구현(WEB 이중화 구현 시, 서비스 단 이중화 구성 완료)

나, 다) 개발에 대한 기술적인 보완 및 백엔드 인프라에 대한 전반적인 이해 필요.
이후, 개발을 통한 초기 기획 서비스 구축 완료

라) CDC나 Veritas 같은 상용 솔루션 외에 방식