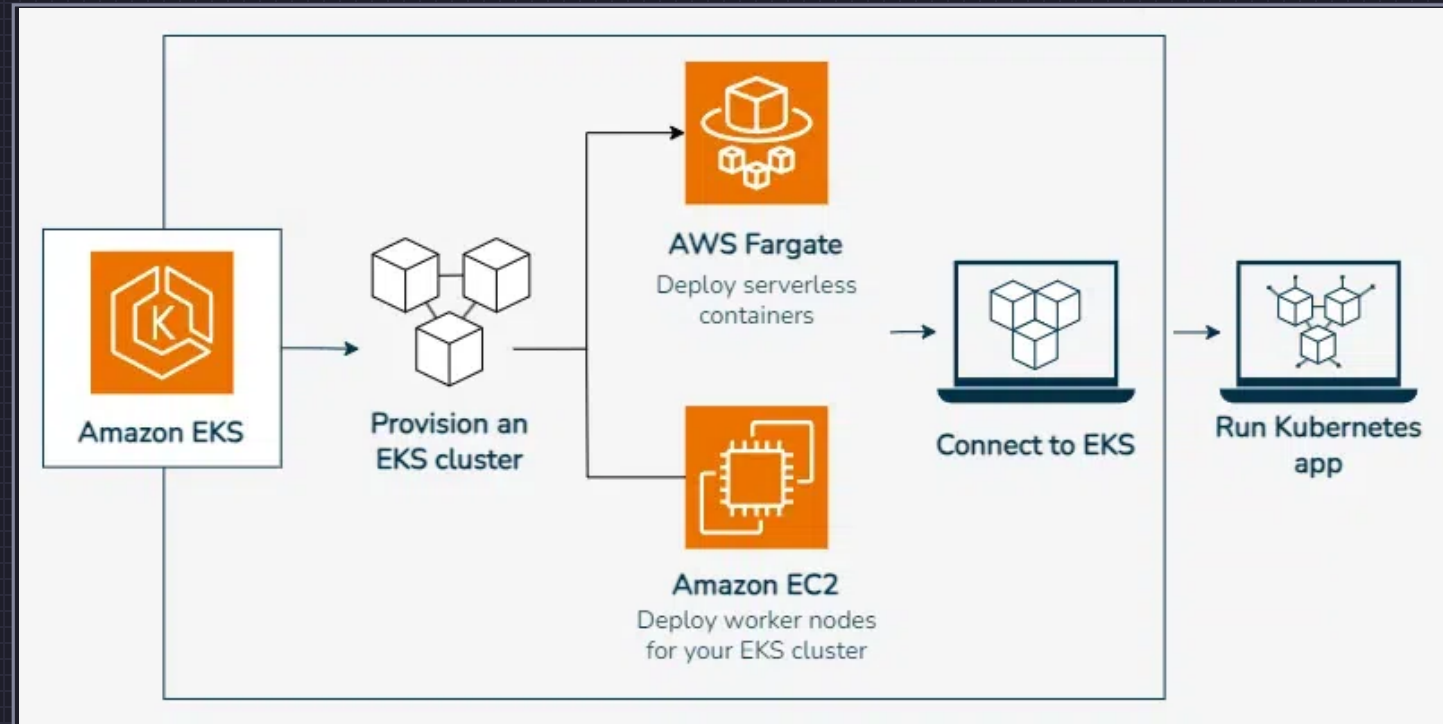


Amazon EKS

FRONT - 김선혁 박채령 임영철 정세환

목차

EKS



AWS 제공 관리형 K8S 서비스
직접 클러스터 설정 없이 EC2에서 컨테이너 실행
강점 - 확장성 , 보안성, AWS 서비스 연동

목차

1. AWS 환경 구성

1-1. VPC 설정

1-2. EC2 Instance

2. AWS 프로파일 등록

3. EKS 구성

3-1. EKS 생성

3-2. 생성 확인

4. EKS 서비스 배포

4-1. 웹 애플리케이션 배포

4-2. 서비스 적용

1. AWS 환경 구성

1-1. VPC 설정 - Default vpc

vpc-04f27a9d77b170b05 / front-default-vpc

[세부 정보](#) | [리소스 맵](#) | [CIDR](#) | [플로우 로그](#) | [태그](#) | [통합](#)

세부 정보

VPC ID  vpc-04f27a9d77b170b05	상태  Available	퍼블릭 액세스 차단  비활성	DNS 호스트 이름 활성화됨
DNS 확인 활성화됨	테넌시 default	DHCP 옵션 세트 dopt-070caeeae501569389	기본 라우팅 테이블 rtb-058adc6b6e17a58fa
기본 네트워크 ACL acl-0b581ecb41ca8efaf	기본 VPC 예	IPv4 CIDR 172.31.0.0/16	IPv6 풀 -
IPv6 CIDR(네트워크 경계 그룹) -	네트워크 주소 사용 지표 비활성화됨	Route 53 Resolver DNS 방화벽 규칙 그룹 -	소유자 ID  590184140480

VPC 이름 : IPv4_cidr

1-1. VPC 설정 – subnet

서브넷 (4) 정보 Last updated 5 minutes ago 작업 서브넷

Find resources by attribute or tag

<input type="checkbox"/>	Name	서브넷 ID	상태	VPC	퍼블릭 ...	IPv4 CIDR	가용 영역	가용 영역 ID	퍼블릭 IPv4 주소 할당
<input type="checkbox"/>	default1	subnet-0c6076c052e865f30	Available	vpc-04f27a9d77b170b05 front-default-vpc	비활성	172.31.0.0/20	ap-northeast-2a	apne2-az1	예
<input type="checkbox"/>	default2	subnet-0e4ceb3f5e51fba2	Available	vpc-04f27a9d77b170b05 front-default-vpc	비활성	172.31.16.0/20	ap-northeast-2b	apne2-az2	예
<input type="checkbox"/>	default3	subnet-04f98c8311bfa2186c	Available	vpc-04f27a9d77b170b05 front-default-vpc	비활성	172.31.32.0/20	ap-northeast-2c	apne2-az3	예
<input type="checkbox"/>	default4	subnet-0ef92e63ee8bdab3e	Available	vpc-04f27a9d77b170b05 front-default-vpc	비활성	172.31.48.0/20	ap-northeast-2d	apne2-az4	예

Subnet name, vpc 연결여부, IPv4_cidr, AZ(a~d), public 주소 할당 여부

1-1. VPC 설정 – Route Table

rtb-058adc6b6e17a58fa / front-default-rt

세부 정보 라우팅 서버넷 연결 엣지 연결 라우팅 전파 태그

세부 정보

라우팅 테이블 ID rtb-058adc6b6e17a58fa	기본 예	명시적 서버넷 연결 -	엣지 -
VPC vpc-04f27a9d77b170b05 front-default-vpc	소유자 ID 590184140480		

rtb-058adc6b6e17a58fa / front-default-rt

세부 정보 라우팅 서버넷 연결 엣지 연결 라우팅 전파 태그


라우팅 (2)

라우팅 필터링

대상	대상
0.0.0.0/0	igw-0f72512faa4bf190a
172.31.0.0/16	local

RT name, rt vpc 연결여부 및 라우팅 테이블, subnet 연결은 따로 없음

1-1. VPC 설정 – Internet Gateway

<input checked="" type="checkbox"/>	Name	인터...	상태	VPC ID
<input checked="" type="checkbox"/>	front-default-igw	igw-0f7...	 Attached	vpc-04f27a9d77b170b05 front-default-vpc

IGW name, igw-vpc attach 여부

1-2. EC2 Instance - Security

EC2 > 보안 그룹 > 보안 그룹 생성

보안 그룹 생성

보안 그룹은 인터넷 및 아웃바운드 트래픽을 관리하는 인스턴스의 가상 방화벽 역할을 합니다. 새 보안 그룹을 생성하려면 아래의 필드를 작성하십시오.

기본 세부 정보

보안 그룹 이름 정보

Front-bastion-sg

생성 후에는 이름을 변경할 수 없습니다.

설명 정보

Front-bastion-sg

VPC 정보

vpc-04f27a9d779b170605

인바운드 규칙 정보

유형	프로토콜	포트 범위	소스	설명 - 선택 사항
SSH	TCP	22	내부	

211.117.44.4/32

삭제

Sg name, inbound – 자신의 ip만 ssh허용 (bastion host)

EC2 > 키 페어 > 키 페어 생성

키 페어 생성 정보

키 페어

프라이빗 키와 퍼블릭 키로 구성되는 키 페어는 인스턴스에 연결할 때 자격 증명을 제공하는 데 사용됩니다. 보안 자격 증명 세트입니다.

이름

Front-key

이름에는 최대 255개의 ASCII 문자가 포함됩니다. 앞 또는 뒤에 공백을 포함할 수 없습니다.

키 페어 유형 | 정보

☒ RSA ☐ ED25519

프라이빗 키 파일 형식

☒ .pem
OpenSSH와 함께 사용

☐ .ppk
PuTTY와 함께 사용

Kp name, .pem 선택

1-2. EC2 Instance 생성

EC2 > 인스턴스 > 인스턴스 시작

인스턴스 시작 정보

Amazon EC2를 사용하면 AWS 클라우드에서 실행되는 가상 머신 또는 인스턴스를 생성할 수 있습니다. 아래의 간단한 단계에 따라 빠르게 시작할 수 있습니다.

이름 및 태그 정보

이름: [추가 태그 추가](#)

▼ 애플리케이션 및 OS 이미지(Amazon Machine Image) 정보

AMI는 인스턴스를 시작하는 데 필요한 소프트웨어 구성(운영 체제, 애플리케이션 서버 및 애플리케이션)이 포함된 템플릿입니다. 아래에서 찾고 있는 항목이 보이지 않으면 AMI를 검색하거나 찾아보세요.

Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Linux

Debian

더 많은 AMI 찾기
AWS, Marketplace 및 커뮤니티의 AMI 포함

Amazon Machine Image(AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type [프리 티어 사용 가능](#)
ami-024ea438ab0376a47 (64비트(x86)) / ami-0b5511d5304edfc79 (64비트(Arm))
가상화: hvm ENA 활성화됨: true 루트 디바이스 유형: ebs

설명
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

아키텍처: AMI ID: ami-024ea438ab0376a47 사용자 이름: ubuntu [확인된 공급 업체](#)

▼ 인스턴스 유형 정보 | 초연 보기

인스턴스 유형

t2.medium
레벨: t2 2 vCPU 4 GiB 메모리 현재 세대: true
온디맨드 Windows 기본 요금: 0.0756 USD per Hour 온디맨드 RHEL 기본 요금: 0.0864 USD per Hour
온디맨드 SUSE 기본 요금: 0.1576 USD per Hour
온디맨드 Ubuntu Pro 기본 요금: 0.0611 USD per Hour 온디맨드 Linux 기본 요금: 0.0576 USD per Hour

☒ 모든 세대 [인스턴스 유형 비교](#)

소프트웨어가 사전 설치된 AMI에는 추가 비용이 적용됩니다.

▼ 키 페어(로그인) 정보

키 페어를 사용하여 인스턴스에 안전하게 연결할 수 있습니다. 인스턴스를 시작하기 전에 선택한 키 페어에 대한 액세스 권한이 있는지 확인하세요.

키 페어 이름 - 필수
 [새 키 페어 생성](#)

▼ 네트워크 설정 정보

VPC - 필수 | 정보
 (기본값) [새 VPC 생성](#)

서브넷 | 정보
 default1 [새 서브넷 생성](#)

퍼블릭 IP 자동 할당 | 정보
 [프리 티어 허용 범위를 벗어나는 경우 추가 요금이 적용됩니다.](#)

방화벽(보안 그룹) | 정보
보안 그룹은 인스턴스에 대한 트래픽을 제어하는 방화벽 규칙 세트입니다. 특정 트래픽이 인스턴스에 도달하도록 허용하는 규칙을 추가합니다.

☐ 보안 그룹 생성 ☒ 기존 보안 그룹 선택

일반 보안 그룹 | 정보
 [보안 그룹 규칙 비교](#)

[X](#)
VPC: vpc-04f27a9d77b170b05
여기서 추가 또는 제거하는 보안 그룹은 모든 네트워크 엔티티에서 추가 또는 제거됩니다.

▶ 고급 네트워크 구성

▼ 스토리지 구성 정보 [고급](#)

1x GiB [루트 볼륨 3000IOPS \(알로화되지 않음\)](#)

사전작성 keypair(Front key)

네트워크 – VPC, subnet(AZ-a) public 주소 할당
기존 생성 SG선택(ssh허용), storage : 8GiB, gp3

Instance name, AMI, instance type(t2.micro)

2. AWS 프로파일 등록

2. AWS 프로 파일 등록 - Access Key 생성 (IAM)

IAM 대시보드 정보

보안 권장 사항 0

- ✔ 루트 사용자에게 MFA 있음
루트 사용자에 대해 멀티 팩터 인증(MFA)을 적용하면 이 계정의 보안이 강화됩니다.
- ✔ MFA가 있음
IAM 사용자에게 다중 인증(MFA)을 적용하면 이 계정의 보안이 강화됩니다.
- ✔ 사용자 kimsh에게 1년 이상 사용하지 않은 활성 액세스 키가 없습니다.
미사용 액세스 키를 비활성화하거나 삭제하면 보안이 향상됩니다.

AWS 계정

계정 ID
 590184140480

계정 별칭
daewoo01 편집 | 삭제

이 계정의 IAM 사용자를 위한 로그인 URL
 <https://daewoo01.signin.aws.amazon.com/console>

IAM 리소스

이 AWS 계정의 리소스

사용자 그룹	사용자	역할	정책	ID 제공업체
0	6	11	2	0

Quick Links

[내 보안 자격 증명](#)

액세스 키, 멀티 팩터 인증(MFA) 및 기타 자격 증명을 관리합니다.

액세스 키 모범 사례 및 대안 정보

보안 개선을 위해 액세스 키와 같은 장기 자격 증명을 사용하지 마세요. 다음과 같은 사용 사례와

사용 사례

- ☒ Command Line Interface(CLI)
AWS CLI를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☐ 로컬 코드
로컬 개발 환경의 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 액

설명 태그 설정 - 선택 사항

이 액세스 키에 대한 설명은 이 사용자에게 태그로 연결되고, 액세스 키와 함께 표시됩니다.

설명 태그 값

이 액세스 키의 용도와 사용 위치를 설명합니다. 좋은 설명은 나중에 이 액세스 키를 자신있게 교체하는 데 유용합니다.

Front-Access-Key

최대 256자까지 가능합니다. 허용되는 문자는 문자, 숫자, UTF-8로 표현할 수 있는 공백 및 _.: / = + - @입니다.

취소

이전

엑세스 키 만들기

엑세스 키 (0) [엑세스 키 만들기](#)

엑세스 키를 사용하여 AWS CLI, AWS Tools for PowerShell, AWS SDK 또는 직접 AWS API 호출을 통해 AWS에 프로그래밍 방식 호출을 전송합니다. 한 번에 최대 두 개의 액세스 키(활성 또는 비활성)를 가질 수 있습니다. [자세히 알아보기](#)

엑세스 키가 없습니다. 액세스 키와 같은 장기 보안 인증을 사용하지 않는 것이 모범 사례입니다. 대신 단기 보안 인증을 제공하는 도구를 사용하세요. [자세히 알아보기](#)

[엑세스 키 만들기](#)


엑세스 키 검색 정보


엑세스 키

본식하거나 잊어버린 비밀번호 엑세스 키는 검색할 수 없습니다. 대신 새 엑세스 키를 생성하고 이전 키를 비활성화합니다.

엑세스 키

비밀 엑세스 키

 AKIAY5ZNXJ3AHYHQDZSL

 ***** 표시

엑세스 키 모범 사례

- 엑세스 키를 일반 텍스트 코드 리포지토리 또는 코드로 저장해서는 안 됩니다.
- 더 이상 필요 없는 경우 엑세스 키를 비활성화하거나 삭제합니다.
- 최소 권한을 활성화합니다.
- 엑세스 키를 정기적으로 교체합니다.

엑세스 키 관리에 대한 자세한 내용은 [AWS 액세스 키 관리 모범 사례](#)를 참조하세요.

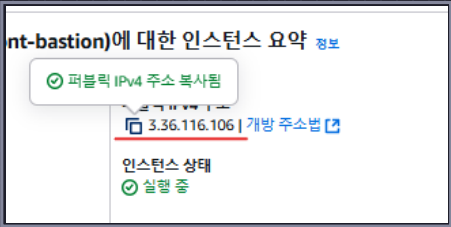
csv 파일 다운로드

완료

Instance name, AMI, instance type(t2.micro)

Access key 자격 증명 – CLI, name지정, csv를 통한 accesskey download

2. AWS 프로 파일 등록 – Instance 내부 접속



```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ whoami
ubuntu
ubuntu@ip-172-31-15-102:~$ cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.1 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
```

연결

일반

이름(N): Front-bastion

프로토콜(P): SSH

호스트(H): 3.36.116.106

포트 번호(O): 22

아이콘:

연결 > 사용자 인증

인증 방법과 기타 관련 매개 변수를 선택하십시오.

이 섹션은 로그인 할 때 시간을 절약하기 위해 사용할 수 있습니다. 그러나 보안을 중요시하는 경우 이 섹션을 비워 두는 것이 좋습니다.

☐ 인증 프로필 사용(A)

인증 프로필(F): <지정하지 않음> 찾아보기(B)...

사용자 이름(U): ubuntu

암호(P):

방법(M):

- ☐ Password
- ☒ Public Key
- ☐ Keyboard Interactive
- ☐ GSSAPI

설정(S)... 위로(U) 아래로(D)

확인 연결 취소

사용자 인증

로그인 프로토포트 인증 방법과 기타 관련 매개 변수를 선택하십시오.

로그인 스크

SSH

보안

터널링

SFTP

TELNET

RLOGIN

SER

RDP

프

연

터미널

키

VT

고급

모양

창

하

고급

추

별

로그

파일 전

X/Y

ZM

Public Key 설정

키 파일(F)

사용자 키(U): <없음>

암호(P):

사용자 키

이름	종류	길이	설명
Front-key	RSA	2048	PEM-rsa-import-20250211

생성(G)... 등록 정보(P) 삭제(D) 가져오기(I)... 내보내기(E)...

확인 취소

Xshell 연결 – ip(사전복사한 instance public ip) :22 / 연결 – 사용자 인증
user : ubuntu(default create user) / 인증방식 : public key(사전 생성 key pair 파일)

2. AWS 프로 파일 등록 - AWS CLI 접속

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ sudo apt-get install -y unzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 174 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 un
Fetched 174 kB in 0s (8874 kB/s)
Selecting previously unselected package unzip.
(Reading database ... 70610 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4.1_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4.1) ...
Setting up unzip (6.0-28ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
```

```
ubuntu@ip-172-31-15-102:~$ unzip awscliv2.zip
Archive:  awscliv2.zip
  creating: aws/
  creating: aws/dist/
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
  inflating: aws/README.md
```

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
ubuntu@ip-172-31-15-102:~$ aws --version
aws-cli/2.24.1 Python/3.12.6 Linux/6.8.0-1021-aws exe/x86_64.ubuntu.24
ubuntu@ip-172-31-15-102:~$
```

```
ubuntu@ip-172-31-15-102:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed
100 64.7M  100 64.7M    0     0  86.9M      0 --:--:-- --:--:-- --:--:-- 86.8M
ubuntu@ip-172-31-15-102:~$
```

https://docs.aws.amazon.com/ko_kr/cli/latest/userguide/getting-started-install.html

위의 링크를 통한 Ubuntu내 AWS CLI 설치 진행

3. EKS 환경 구성

3-1. EKS 생성 – Kubectl, Ekstcl 설치

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.7/2024-12-12/bin/linux/amd64/kubectl
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 49.0M 100 49.0M 0 0 4612k 0 0:00:10 0:00:10 --:--:-- 5251k
ubuntu@ip-172-31-15-102:~$ ls
aws awscli2.zip kubectl
ubuntu@ip-172-31-15-102:~$ chmod +x ./kubectl
ubuntu@ip-172-31-15-102:~$ ls
aws awscli2.zip kubectl
```

```
ubuntu@ip-172-31-15-102:~$ mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
ubuntu@ip-172-31-15-102:~$ kubectl version
Client Version: v1.30.7-eks-59bf375
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@ip-172-31-15-102:~$
```

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ ARCH=amd64
ubuntu@ip-172-31-15-102:~$ PLATFORM=$(uname -s)_$ARCH
ubuntu@ip-172-31-15-102:~$ curl -sLO "https://github.com/eksctl-io/eksctl/releases/latest/download/eksctl_${PLATFORM}.tar.gz"
ubuntu@ip-172-31-15-102:~$ ls
aws awscli2.zip bin eksctl_linux_amd64.tar.gz kubectl
ubuntu@ip-172-31-15-102:~$ tar -xzf eksctl_${PLATFORM}.tar.gz -C /tmp && rm eksctl_${PLATFORM}.tar.gz
ubuntu@ip-172-31-15-102:~$ sudo mv /tmp/eksctl /usr/local/bin
ubuntu@ip-172-31-15-102:~$ eksctl version
0.203.0
ubuntu@ip-172-31-15-102:~$
```

<https://eksctl.io/installation/> - for unix 부분참조(eksctl)

https://docs.aws.amazon.com/ko_kr/eks/latest/userguide/install-kubectl.html

linux(amd64) – 최신버전 설치(kubectl)

3-1. EKS 생성 – Ekstcl을 이용한 EKS 환경 구축

```
ubuntu@ip-172-31-15-102:~$ eksctl create cluster \
--name Front-eks \
--region ap-northeast-2 \
--with-oidc \
--nodegroup-name Front-ng \
--zones ap-northeast-2a,ap-northeast-2c \
--nodes 2 \
--node-type t3.medium \
--node-volume-size=20 \
--managed
2025-02-11 03:31:05 [i] eksctl version 0.203.0
2025-02-11 03:31:05 [i] using region ap-northeast-2
2025-02-11 03:31:05 [i] subnets for ap-northeast-2a - public:192.168.0.0/19 private:
2025-02-11 03:31:05 [i] subnets for ap-northeast-2c - public:192.168.32.0/19 private:
2025-02-11 03:31:05 [i] nodegroup "Front-ng" will use "" [AmazonLinux2/1.30]
2025-02-11 03:31:05 [i] using Kubernetes version 1.30
2025-02-11 03:31:05 [i] creating EKS cluster "Front-eks" in "ap-northeast-2" region
2025-02-11 03:31:05 [i] will create 2 separate CloudFormation stacks for cluster it
2025-02-11 03:31:05 [i] if you encounter any issues, check CloudFormation console or
```

Name 지정

Region : 아시아 – 서울

Zone : AG

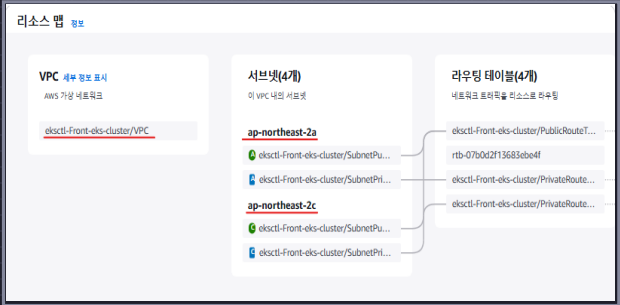
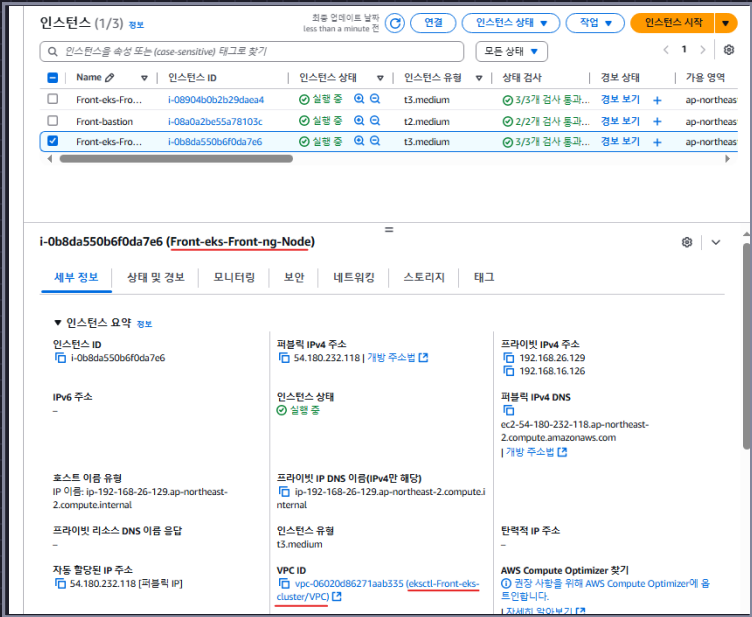
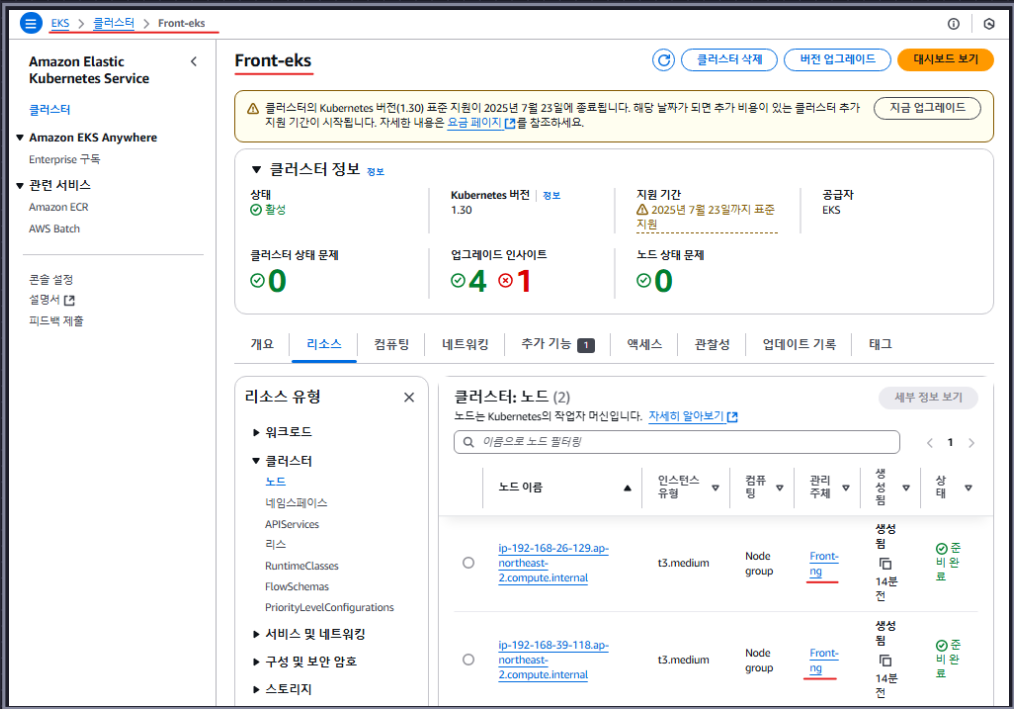
Node 수 : 2개

The screenshot shows the AWS CloudFormation console. On the left, the 'Stacks' list shows two stacks: 'eksctl-Front-eks-cluster' and 'eksctl-beautiful-gopher-1739244574-cluster'. The main panel displays the details for 'eksctl-Front-eks-cluster', which is in the 'CREATE_IN_PROGRESS' state. The 'Events' tab shows a list of events, including the creation of a NATGateway and several PublicSubnetRoutes.

타임스탬프	논리적 ID	상태	세부
2025-02-11 12:31:35 UTC+0900	NATGateway	CREATE_IN_PROGRESS	...
2025-02-11 12:31:34 UTC+0900	PublicSubnetRoute	CREATE_COMPLETE	-
2025-02-11 12:31:34 UTC+0900	PublicSubnetRoute	CREATE_IN_PROGRESS	-
2025-02-11 12:31:33 UTC+0900	PublicSubnetRoute	CREATE_IN_PROGRESS	-
2025-02-11 12:31:32 UTC+0900	PublicRouteTable	CREATE_COMPLETE	-

Eksctl 명령어에 의해 노드 instanc를 생성하기 위해
CloudFormation-stack이 자동 생성됨을 확인

3-2. 생성 확인 - Ekstcl을 이용한 EKS 환경 구축



Eksctl 명령어에 의해 controle-plane인 Front-ng에 의한 Node 2개가 새로 생성됨을 EKS 서비스에서 확인

Eksctl 명령어에 의해 생성된 CloudFormation으로 Node Instance가 2개 생성 됨을 확인

3-2. 생성 확인 - Eksctl을 이용한 EKS 환경 구축

VPC (3) 정보

<input type="checkbox"/>	Name	VPC ID	상태	퍼블릭
<input type="checkbox"/>	front-default-vpc	vpc-04f27a9d7...	Available	
<input type="checkbox"/>	eksctl-beautiful-gop...	vpc-00f1fe19f...	Available	
<input type="checkbox"/>	eksctl-Front-eks-clus...	vpc-06020d86...	Available	

보안 그룹 (11) 정보

<input type="checkbox"/>	Name	보안 그룹 ID
<input type="checkbox"/>	-	sg-0c190e17f06d7132f
<input type="checkbox"/>	eksctl-Front-eks-clus...	sg-069aaad99541bc711
<input type="checkbox"/>	-	sg-05c758f6605b2c805
<input type="checkbox"/>	eks-cluster-sg-beauti...	sg-05c4686f541ed858a
<input type="checkbox"/>	-	sg-0898d24ab37bc277f
<input type="checkbox"/>	-	sg-01ccfb2042568ded5
<input type="checkbox"/>	-	sg-072f968df1d91639c
<input type="checkbox"/>	eks-cluster-sg-Front-...	sg-02ec7fe4e421e3c38
<input type="checkbox"/>	eksctl-beautiful-gop...	sg-0734ddfa8e8c20779
<input type="checkbox"/>	eksctl-beautiful-gop...	sg-0dc4d5a351809b82a
<input type="checkbox"/>	eksctl-Front-eks-clus...	sg-01af0093a5c482c3d

서브넷 (10) 정보

<input type="checkbox"/>	Name	서브넷 ID	상태
<input type="checkbox"/>	eksctl-beauti...	subnet-01142cd2e6500ed07	Available
<input type="checkbox"/>	eksctl-beauti...	subnet-03492ad949e3d874b	Available
<input type="checkbox"/>	eksctl-beauti...	subnet-04605aa9c9d53cd4e	Available
<input type="checkbox"/>	eksctl-beauti...	subnet-0c83417c9c80405fd	Available
<input type="checkbox"/>	eksctl-beauti...	subnet-08b6d703ed36a76a1	Available
<input type="checkbox"/>	eksctl-beauti...	subnet-07f78dc59b3ec919b	Available
<input type="checkbox"/>	eksctl-Front-...	subnet-0fd9545b1c72c997c	Available
<input type="checkbox"/>	eksctl-Front-...	subnet-031858109644f94c5	Available
<input type="checkbox"/>	eksctl-Front-...	subnet-030442de82f99ab5a	Available
<input type="checkbox"/>	eksctl-Front-...	subnet-07330eae6add34d10	Available

라우팅 테이블 (1/10) 정보

<input type="checkbox"/>	Name	라우팅 테이블
<input checked="" type="checkbox"/>	eksctl-Front-eks-cluster/PublicRouteTable	rtb-00abaf134
<input type="checkbox"/>	front-default-rt	rtb-058adc6b8
<input type="checkbox"/>	-	rtb-0a1dd928
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	rtb-0c79a918
<input type="checkbox"/>	-	rtb-07b0d2f13
<input type="checkbox"/>	eksctl-Front-eks-cluster/PrivateRouteTa...	rtb-0e663215
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	rtb-00ed1125
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	rtb-0195686f
<input type="checkbox"/>	eksctl-Front-eks-cluster/PrivateRouteTa...	rtb-088253b8
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	rtb-06b88c6c

인터넷 게이트웨이 (3) 정보

<input type="checkbox"/>	Name	인터...	상태
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	igw-081...	Attached
<input type="checkbox"/>	eksctl-Front-eks-cluster/InternetGateway	igw-0c3...	Attached
<input type="checkbox"/>	front-default-igw	igw-0f7...	Attached

탄력적 IP 주소 (2)

<input type="checkbox"/>	Name	할당된 IPv4 주소	유형
<input type="checkbox"/>	eksctl-beautiful-gopher-1739244574-cl...	3.34.163.149	퍼블릭
<input type="checkbox"/>	eksctl-Front-eks-cluster/NATIP	3.35.189.107	퍼블릭

NAT 게이트웨이 (2) 정보

<input type="radio"/>	Name	NAT 게이트웨이 ID	연결 유
<input type="radio"/>	eksctl-beautiful-gop...	nat-0fa07ef51e3f196c2	Public
<input type="radio"/>	eksctl-Front-eks-clus...	nat-0fc49fd9784aa3e39	Public

Eksctl 명령어에 의해 새로 VPC, subnet, SG, RT, ACL, GW, ElasticIP, NAT_GW가 설치됨을 확인

4. EKS 서비스 배포

4-1. 웹 애플리케이션 배포 - 임의의 Deployment, SVC 생성

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl get nodes
NAME                                STATUS    ROLES    AGE   VERSION
ip-192-168-26-129.ap-northeast-2.compute.internal Ready    <none>   18m   v1.30.8-eks-aec579
ip-192-168-39-118.ap-northeast-2.compute.internal Ready    <none>   18m   v1.30.8-eks-aec579
ubuntu@ip-172-31-15-102:~$
```

Kubectl을 통한 Controlplane과 타 node간의 접속 확인

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl expose deployment webtest --port=80 --type=LoadBalancer
service/webtest exposed
ubuntu@ip-172-31-15-102:~$ kubectl get services
error: the server doesn't have a resource type "services"
ubuntu@ip-172-31-15-102:~$ kubectl get svc
NAME      TYPE      CLUSTER-IP    EXTERNAL-IP
kubernetes ClusterIP  10.100.0.1     <none>
webtest   LoadBalancer  10.100.165.59  af7cbc7cd275f416c8e605ade164beee-570115867.ap-northeast-2.elb.amazonaws.com
ubuntu@ip-172-31-15-102:~$
```

사전 작성한 Deployment를 연결해줄 svc를 생성 (type : LoadBalancer)

Get svc를 통한 확인 작업

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl create deployment webtest --image=nginx:1.14 --port=80 --replicas=5
deployment.apps/webtest created
ubuntu@ip-172-31-15-102:~$ kubectl get pods -o wide
NAME                                READY    STATUS              RESTARTS   AGE   IP                                NODE
webtest-6d754887d7-6zld8            0/1     ContainerCreating   0           6s    <none>                           ip-192-168-39-118.ap-northeast-2.elb.amazonaws.com
webtest-6d754887d7-88rbb            0/1     ContainerCreating   0           6s    <none>                           ip-192-168-26-129.ap-northeast-2.elb.amazonaws.com
webtest-6d754887d7-1d4wf            0/1     ContainerCreating   0           6s    <none>                           ip-192-168-26-129.ap-northeast-2.elb.amazonaws.com
webtest-6d754887d7-tvthh            0/1     ContainerCreating   0           6s    <none>                           ip-192-168-26-129.ap-northeast-2.elb.amazonaws.com
webtest-6d754887d7-zndwc            0/1     ContainerCreating   0           6s    <none>                           ip-192-168-39-118.ap-northeast-2.elb.amazonaws.com
ubuntu@ip-172-31-15-102:~$
```

오브젝트가 정상 작성되는지 확인하기 위한 임의의 object 작성

Deployment

Image : nginx

Port : 80

Replicas : 5

Get을 통한 Deployment 생성 상태 및 pod 5개 생성여부 확인

4-1. 웹 애플리케이션 배포 - SVC type 변경 확인

```
1 Front-bastion x +
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: "2025-02-11T04:03:13Z"
  labels:
    app: webtest
  name: webtest
  namespace: default
  resourceVersion: "5472"
  uid: f7cbc7cd-275f-416c-8e60-5ade164beee4
spec:
  clusterIP: 10.100.165.59
  clusterIPs:
  - 10.100.165.59
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 30601
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: webtest
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer: {}
```

Edit 명령어를 통한 SVC의 설정 변경 (type -> LB로)

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl edit svc webtest
service/webtest edited
ubuntu@ip-172-31-15-102:~$ kubectl get svc
NAME         TYPE          CLUSTER-IP    EXTERNAL-IP
kubernetes   ClusterIP     10.100.0.1     <none>
webtest      LoadBalancer  10.100.165.59  af7cbc7cd275f416c8e605ade164beee-1836680444.ap...
```

Get을 통한 타입 변경 확인

Ec2의 loadbalancer에 svc의 external-ip를 검색해서 LB가 정상 작동됨을 확인
대상 인스턴스를 통해 instance node들에 정상적으로 연결됨을 확인

로드 밸런서 (1/1)

Elastic Load Balancing은 수신 트래픽의 변화에 따라 자동으로 로드 밸런서 용량을 확장합니다.

로드 밸런서 필터링 1개 일치

af7cbc7cd275f416c8e605ade164beee-1836680444.ap-northeast-2.elb.amazonaws.com

<input checked="" type="checkbox"/>	이름	DNS 이름	상태	VPC ID	가용 영역
<input checked="" type="checkbox"/>	af7cbc7cd275f416c8e6...	af7cbc7cd275f416c8e605a...	-	vpc-06020d86271aab335	2가용 영역

로드 밸런서: af7cbc7cd275f416c8e605ade164beee

세부 정보 | 리스너 | 네트워크 매핑 | 보안 | 상태 검사 | **대상 인스턴스** | 모니터링 | 속성 | 태그

대상 인스턴스 (2)

현재 로드 밸런서에 등록된 인스턴스가 표시됩니다. 인스턴스를 등록 취소하려면 인스턴스를 선택한 다음 등록 취소를 선택합니다. 여러 인스턴스를 취소하려면 인스턴스 관리를 선택합니다.

대상 인스턴스 필터링

<input type="checkbox"/>	인스턴스 ID	이름	상태 확인	상태 확인 설명
<input type="checkbox"/>	i-08904b0b2b29daea4	Front-eks-Front-ng-Node	✓ 서비스 중	해당되지 않음
<input type="checkbox"/>	i-0b8da550b6f0da7e6	Front-eks-Front-ng-Node	✓ 서비스 중	해당되지 않음

4-1. 웹 애플리케이션 배포 – 정상 배포 확인

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

생성된 LB의 dns를 브라우저에 복붙 시 nginx 서버가 정상 작동함을 확인

4-2. 서비스 적용 - HELM 설치 및 LB_IAM 생성

```
ubuntu@ip-172-31-15-102:~$ curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
chmod 700 get_helm.sh
./get_helm.sh
```

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.11.0/docs/install/iam_policy.json
```

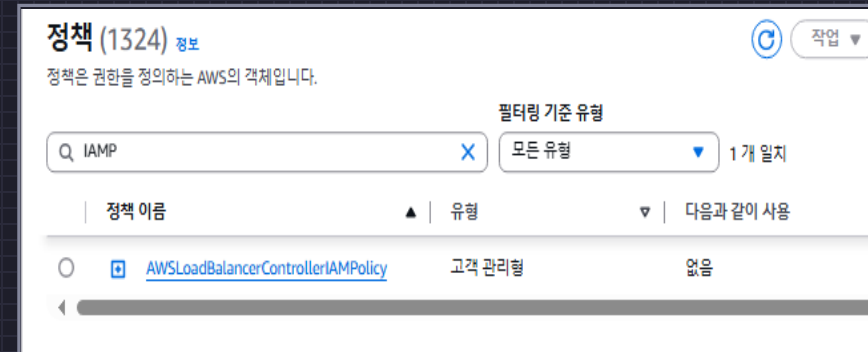
HELM 설치 참조 : <https://helm.sh/docs/intro/install/>

참조 : https://docs.aws.amazon.com/ko_kr/eks/latest/userguide/lbc-helm.html

```
ubuntu@ip-172-31-15-102:~$ aws iam create-policy \
--policy-name AWSLoadBalancerControllerIAMPolicy \
--policy-document file://iam_policy.json
{
  "Policy": {
    "PolicyName": "AWSLoadBalancerControllerIAMPolicy",
    "PolicyId": "ANPAYS2NXJ3AGFGMXJ5OZ",
    "Arn": "arn:aws:iam::590184140480:policy/AWSLoadBalancerControllerIAMPolicy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2025-02-11T04:56:05+00:00",
    "UpdateDate": "2025-02-11T04:56:05+00:00"
  }
}
```

Curl을 통한 IAM-policy json 파일 다운로드

AWS CLI를 이용한 IAM 정책 생성 (LB허가 - AWSLoadBalancerControllerIAMPolicy)



```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ export cluster_name=front-eks
oidc_id=$(aws eks describe-cluster --name $cluster_name --query "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
echo $oidc_id
2074E862685A24EA36F9C1344AC8C31C
ubuntu@ip-172-31-15-102:~$
```

Cluster name, oidc id 등의 환경 변수 지정

4-2. 서비스 적용 - Helm으로 LBC 설치

```
ubuntu@ip-172-31-15-102:~$ helm repo add eks https://aws.github.io/eks-charts
"eks" has been added to your repositories
```

```
ubuntu@ip-172-31-15-102:~$ helm repo update eks
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "eks" chart repository
Update Complete. ☺Happy Helming!☺
ubuntu@ip-172-31-15-102:~$
```

패키지를 이용하기 위해 사전에 helm repository 추가 및 업데이트

```
ubuntu@ip-172-31-15-102:~$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
-n kube-system \
--set clusterName=Front-eks \
--set serviceAccount.create=false \
--set serviceAccount.name=aws-load-balancer-controller
NAME: aws-load-balancer-controller
LAST DEPLOYED: Tue Feb 11 05:40:44 2025
NAMESPACE: kube-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
AWS Load Balancer controller installed!
ubuntu@ip-172-31-15-102:~$
```

```
ubuntu@ip-172-31-15-102:~$ kubectl get pod -A
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	aws-load-balancer-controller-688db64d49-nq5mg	1/1	Running	0	71s
kube-system	aws-load-balancer-controller-688db64d49-wkvj4	1/1	Running	0	71s
kube-system	aws-node-f25kb	2/2	Running	0	118m
kube-system	aws-node-qj72p	2/2	Running	0	118m
kube-system	coredns-5b9dfbf96-5qds8	1/1	Running	0	122m
kube-system	coredns-5b9dfbf96-ncr55	1/1	Running	0	122m
kube-system	kube-proxy-m6d7r	1/1	Running	0	118m
kube-system	kube-proxy-q9zrh	1/1	Running	0	118m
kube-system	metrics-server-598987b95f-cbnkg	1/1	Running	0	122m
kube-system	metrics-server-598987b95f-cpnt4	1/1	Running	0	122m

```
ubuntu@ip-172-31-15-102:~$
```

설치된 eks repository를 이용하고 clustername 및 sa 지정

LBC pod가 성공적으로 생성됨을 확인

4-2. 서비스 적용 - *Yaml*을 이용한 *Deployoy* 생성

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl create namespace nlb-sample-app
namespace/nlb-sample-app created
ubuntu@ip-172-31-15-102:~$
```

Ns nlb-sample app 생성

```
ubuntu@ip-172-31-15-102:~$ cat > sample-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nlb-sample-app
  namespace: nlb-sample-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: public.ecr.aws/nginx/nginx:1.23
          ports:
            - name: tcp
              containerPort: 80
```

홈페이지에서 샘플 deployment의 yaml을 복사

생성 pod name : nlb-sample-app

Ns : nlb-sample-app

Replicas=3

```
1 Front-bastion x +
ubuntu@ip-172-31-15-102:~$ kubectl apply -f sample-deployment.yaml
deployment.apps/nlb-sample-app created
ubuntu@ip-172-31-15-102:~$
```

Yaml을 이용하여 Deployment 생성

```
ubuntu@ip-172-31-15-102:~$ kubectl get pods -n nlb-sample-app
NAME                                READY   STATUS    RESTARTS   AGE
nlb-sample-app-fccbb75cd-kbccf      1/1     Running   0           53s
nlb-sample-app-fccbb75cd-pxkcq      1/1     Running   0           53s
nlb-sample-app-fccbb75cd-wvhjs      1/1     Running   0           53s
ubuntu@ip-172-31-15-102:~$
```

Deployment replicas=3에 의한 pod 3개의 생성 확인

4-2. 서비스 적용 - Yaml을 이용한 SVC(LB) 생성

1 Front-bastion x +

```
ubuntu@ip-172-31-15-102:~$ vi sample-service.yaml
```

1 Front-bastion x +

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

역시 동일하게 웹 페이지 내의 yaml 복사
Service type : LoadBalancer

1 Front-bastion x +

```
ubuntu@ip-172-31-15-102:~$ kubectl apply -f sample-service.yaml
service/nlb-sample-service created
ubuntu@ip-172-31-15-102:~$
```

```
ubuntu@ip-172-31-15-102:~$ kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.100.0.1	<none>	443/TCP	139m

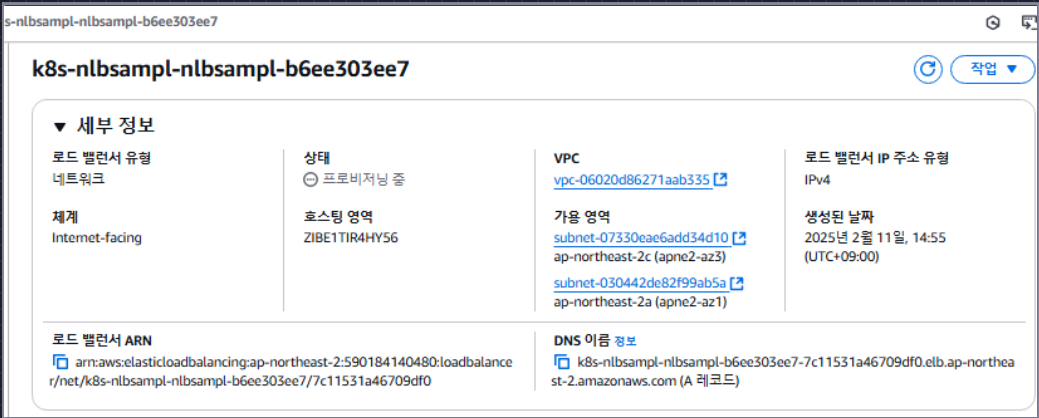
```
ubuntu@ip-172-31-15-102:~$ kubectl get svc -n nlb-sample-app
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
nlb-sample-service	LoadBalancer	10.100.254.245	k8s-nlbsamp1-nlbsamp1-b6ee303ee7-7c11531a46709df0.elb.ap-northeast-2.amazonaws.com	80:30138/TCP

```
42s
ubuntu@ip-172-31-15-102:~$
```

SVC 이름 및 Type(LB) External IP에서 LoadBalancer DNS 확인

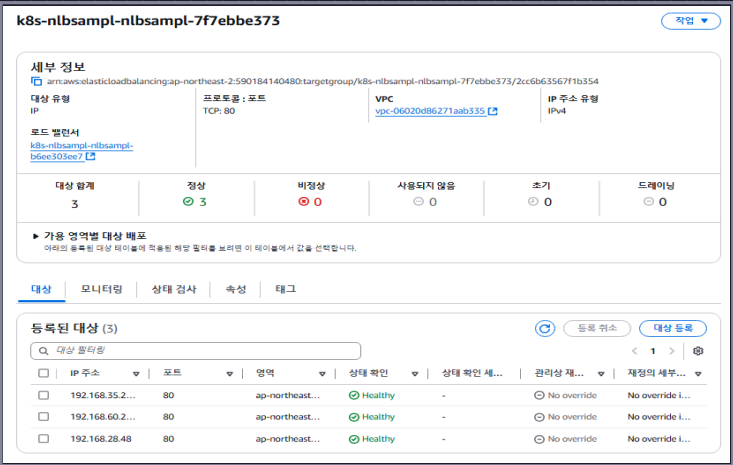
4-2. 서비스 적용 - NLB(SVC) AWS 홈페이지 확인



방금 확인한 External IP와 위의 DNS 이름이 동일한 것으로 보아
AWS 페이지 내에 NLB가 성공적으로 생성된 것을 확인

```
ubuntu@ip-172-31-15-102:~$ kubectl get pods -n nlb-sample-app -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP
NODE   READINESS GATES
nlb-sample-app-fccbb75cd-kbccf      1/1     Running   0           11m   192.168.60.251
      <none>
nlb-sample-app-fccbb75cd-pxkcq      1/1     Running   0           11m   192.168.35.208
      <none>
nlb-sample-app-fccbb75cd-wvhjs      1/1     Running   0           11m   192.168.28.48
      <none>
ubuntu@ip-172-31-15-102:~$
```

SVC(NLB)와 연동된 pod들의 IP 확인



IP가 동일한 것으로 보아 Pod 들이 NLB의
TargetGroup으로 등록된 것을 확인



NLB의 DNS 이름 복사 후 브라우저에 붙여넣기 시 성공
적으로 NLB가 작동함을 확인

4-2. 서비스 적용 - 홈페이지를 참조하여 ALB 생성

```
ubuntu@ip-172-31-15-102:~$ vi 2048_full.yaml
```

```
ubuntu@ip-172-31-15-102:~$ cat 2048_full.yaml
---
apiVersion: v1
kind: Namespace
metadata:
  name: game-2048
---
apiVersion: apps/v1
kind: Deployment
metadata:
  namespace: game-2048
  name: deployment-2048
spec:
  selector:
    matchLabels:
      app.kubernetes.io/name: app-2048
  replicas: 2
  template:
    metadata:
      labels:
        app.kubernetes.io/name: app-2048
    spec:
      containers:
        - image: public.ecr.aws/16m2t8p7/docker-2048:latest
          imagePullPolicy: Always
          name: app-2048
          ports:
            - containerPort: 80
```

```
---
apiVersion: v1
kind: Service
metadata:
  namespace: game-2048
  name: service-2048
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: NodePort
  selector:
    app.kubernetes.io/name: app-2048
---
```

Deployment와 연동되는 SVC(type-Nodeport) 생성

NS, ALB targetgroup의 pod를 생성하는
Deployment 생성 (image game - 2048)

4-2. 서비스 적용 - 홈페이지를 참조하여 ALB 생성

```
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  namespace: game-2048
  name: ingress-2048
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/target-type: ip
spec:
  ingressClassName: alb
  rules:
  - http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: service-2048
            port:
              number: 80
ubuntu@ip-172-31-15-102:~$
```

```
ubuntu@ip-172-31-15-102:~$ kubectl apply -f 2048_full.yaml
namespace/game-2048 created
deployment.apps/deployment-2048 created
service/service-2048 created
ingress.networking.k8s.io/ingress-2048 created
ubuntu@ip-172-31-15-102:~$
```

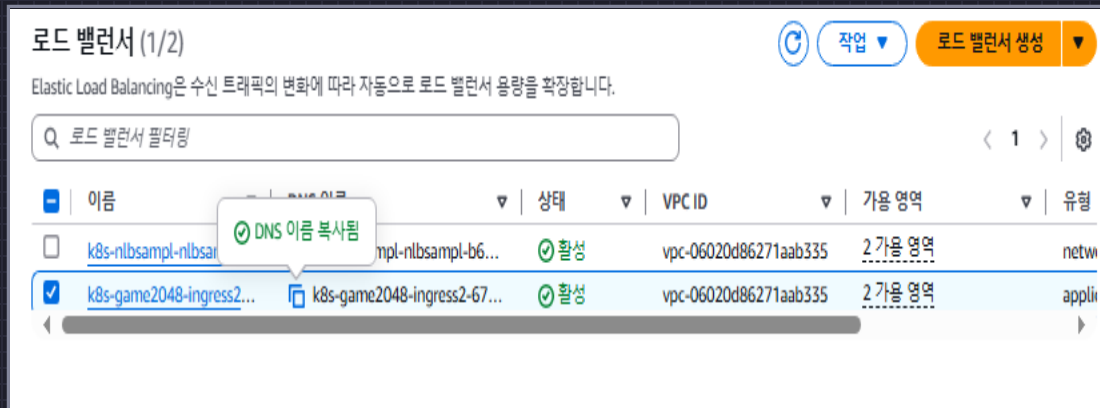
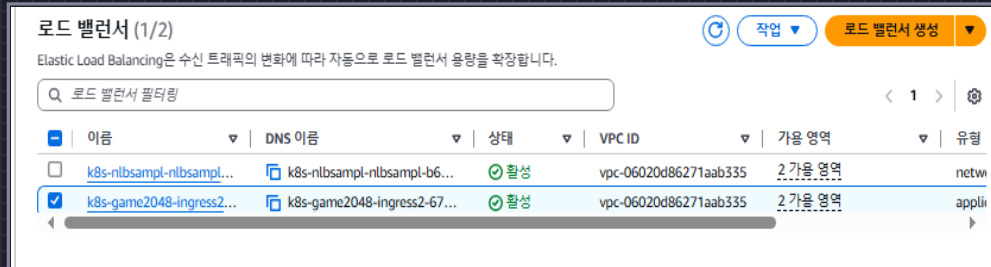
보안규칙을 적용하기 위한 Ingress적용
(http 허용 및 경로 및 SVC 한정)

```
ubuntu@ip-172-31-15-102:~$ kubectl get pods -n game-2048
NAME                                READY   STATUS    RESTARTS   AGE
deployment-2048-85f8c7d69-jpg15    1/1     Running   0           59s
deployment-2048-85f8c7d69-p5j8m    1/1     Running   0           59s
ubuntu@ip-172-31-15-102:~$ kubectl get services -n game-2048
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service-2048  NodePort    10.100.239.171   <none>        80:31598/TCP     79s
```

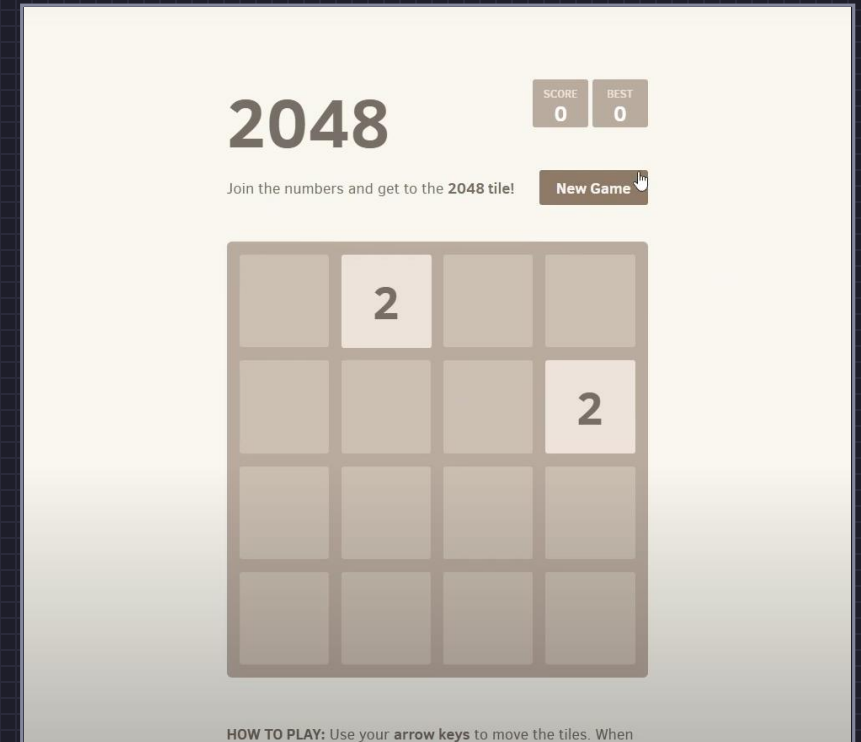
```
ubuntu@ip-172-31-15-102:~$ kubectl get ingress -n game-2048
NAME      CLASS  HOSTS                                                                 PORTS   AGE
ingress-2048  alb    *      k8s-game2048-ingress2-67770ffaab-11440059.ap-northeast-2.elb.amazonaws.com  80      117s
ubuntu@ip-172-31-15-102:~$ kubectl describe -n game-2048 ingress ingress-2048
Name:      ingress-2048
Labels:    <none>
Namespace: game-2048
Address:   k8s-game2048-ingress2-67770ffaab-11440059.ap-northeast-2.elb.amazonaws.com
Ingress Class:  alb
Default backend: <default>
Rules:
  Host      Path  Backends
  ----      -
  *          /     service-2048:80 (192.168.3.135:80,192.168.49.48:80)
Annotations: alb.ingress.kubernetes.io/scheme: internet-facing
              alb.ingress.kubernetes.io/target-type: ip
Events:
  Type      Reason              Age   From      Message
  ----      -
  Normal    SuccessfullyReconciled 2m35s  ingress   Successfully reconciled
ubuntu@ip-172-31-15-102:~$
```

Pod 및 SVC(nodeport) Ingress가 성공적으로 작성됨을 확인
Ingress의 DNS 확인 (Address)

4-2. 서비스 적용 - AWS의 LoadBalancer에서 ALB 확인 및 검증



Ingress address와 동일한 것으로 보아 Ingress에
의한 ALB가 생성 됨을 확인



ALB의 DNS를 이름을 브라우저에 복사 후 정상적으로 작동함을 확인

프로젝트 기대 효과

프로젝트 학습 효과

- K8S 클러스터 구축 및 관리 역량 강화
- AWS 환경에서의 K8S 운영 경험 습득
- 효율적 클라우드 네이티브 APP 배포 및 관리
- 보안이 강화된 클러스터 환경 구축

프로젝트 활용 방안

- 기업 환경에서 K8S 기반 APP 운영 및 확장
- DevOps 및 CI/CD 환경에서 Amazon EKS를 활용한 배포 시스템 구축
- MSA 기반 클라우드 네이티브 APP 운영

Thank you 😊