Elastic Container Service 실습

우 여 명

Matholic

발표자 소개

- AWS를 좋아함
- 도커파일로 이미지를 여러개 만들어본 경험 있음
- 도커 스웜을 살짝 해본 경험 있음
- 쿠버네티스 경험 없음
- 회사에서 이번에 ECS를 도입

수학 교육 솔루션을 만드는 회사

Elastic Container Service - 도커 기반의 배포 도구

> Amazon Elastic Container Service(ECS)는 확장성이 뛰어난 고성능 컨테이너 오케스트레이션 서비스로서, Docker 컨테이너를 지원하며 AWS에서 컨테이너식 애플리케이션을 쉽게 실행하고 확장 및 축소할 수 있습니다. Amazon ECS를 사용하면 자체 컨테이너 오케스트레이션 소프트웨어를 설치하고 운영할 필요가 없으며, 가상 머신의 클러스터를 관리 및 확장하거나 해당 가상 머신에서 컨테이너를 예약하지 않아도 됩니다.

실습 목표

- ECR에 도커 이미지를 푸시해본다.
- ECS를 웹 기반 UI에서 실습해본다.
- ECS를 ecs-cli를 기반으로 실습해본다.
- travis-ci와 ecs-cli를 이용해서 CI/CD를 구축해본다.

실습 준비

- 우선 배포할 서비스 - 스프링부트 hello world!
  https://github.com/voyagerwoo/vw.demo.helloworld
- AWS 루트 권한 유저
- Let's go : https://ap-northeast-2.console.aws.amazon.com/ecs/home?region=ap-northeast-2

WEB console 실습

1. ECR(Elastic Container Registry) 만들기

좌측 메뉴바에서 ECR을 클릭한다. 한글로 `리포지토리` 라고 되어있다. 그리고 `리포지토리 생성` 을 누른다. 이름만 써서 `다음 단계` 를 클릭하면 생성된다. 생성되면 docker push 스크립트가 나온다.

Elastic Container Registry 시작하기

단계 1: 리포지토리 구성

단계 2: Docker 이미지 빌드, 태그 지정 및 푸시

Docker 이미지 빌드, 태그 지정 및 푸시

리포지토리가 존재하므로 다음 절차에 따라 Docker 이미지를 푸시할 수 있습니다.

리포지토리 생성 완료
957582603404.dkr.ecr.ap-northeast-2.amazonaws.com/hello-ecs

AWS CLI 및 Docker를 설치하고 아래 절차에 대한 자세한 내용을 보려면 ECR 설명서 페이지를 참조하십시오.

1) 다음과 같이 레지스트리에 대해 Docker 클라이언트를 인증하는 데 사용할 수 있는 `docker login` 명령을 조회합니다.

```
aws ecr get-login --no-include-email --region ap-northeast-2
```

2) 이전 단계에서 반환된 `docker login` 명령을 실행합니다.
참고:
Windows PowerShell을 사용 중인 경우, 다음 명령을 대신 실행합니다.

```
Invoke-Expression -Command (aws ecr get-login --no-include-email --region ap-northeast-2)
```

3) 다음 명령을 사용하여 Docker 이미지를 빌드합니다. Docker 파일을 처음부터 빌드하는 방법에 대한 정보를 보려면 여기에 있는 지침을 참조하십시오. 이미지를 이미 빌드한 경우에는 이 단계를 건너뛸 수 있습니다.

```
docker build -t hello-ecs .
```

4) 빌드가 완료되면 다음과 같이 이미지에 태그를 지정하여 리포지토리에 푸시할 수 있습니다.

```
docker tag hello-ecs:latest 957582603404.dkr.ecr.ap-northeast-2.amazonaws.com/hello-ecs:latest
```

5) 다음 명령을 실행하여 이 이미지를 새로 생성한 AWS 리포지토리로 푸시합니다.

```
docker push 957582603404.dkr.ecr.ap-northeast-2.amazonaws.com/hello-ecs:latest
```

*필수

완료

2. ECR에 푸시하기

만들어진 repo에는 보통 하나의 도커 이미지를 다른 태그 정보(버전)로 푸시한다. 푸시하는 스크립트는 만들어진 repo에 들어가서 `푸시 명령 보기` 를 누르면 확인할 수 있다.

나는 travis를 활용하여 github에 코드가 푸시되면 자동으로 도커이미지가 빌드되고 ECR에 푸시되도록 설정했다.

[참고]

https://github.com/voyagerwoo/vw.demo.helloworld/blob/master/.travis.yml

3. 작업(Task) 정의

작업 정의 생성

작업 정의는 작업에 포함할 컨테이너와 그 컨테이너들이 상호 작용하는 방식을 지정합니다. 컨테이너가 사용할 데이터 볼륨을 지정할 수도 있습니다. 자세히 알아보기

To learn about which parameters are supported for Windows Containers, please see the ECS Documentation.,

작업 정의 이름*: hello-ecs

작업 역할: *역할 선택...*

인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. IAM 콘솔에서 Amazon Elastic Container Service 작업 역할을 생성합니다.

네트워크 모드: <default>

<default>을(를) 선택할 경우 ECS는 Docker의 기본 네트워킹 모드를 사용하여 컨테이너를 시작합니다. Docker의 기본 네트워킹 모드는 Windows의 Linux 브리지(Bridge) 및 NAT 브리지(Bridge)입니다. <default>은(는) Windows에서 유일하게 지원되는 모드입니다.

작업 크기

작업 크기를 통해 작업의 고정된 크기를 지정할 수 있습니다. 작업 크기는 Fargate 시작 유형을 사용하는 작업에 대해 필요하며, EC2 시작 유형에 대해서는 선택 사항입니다. 작업 크기가 설정되면 컨테이너 수준 메모리 설정은 선택 사항입니다. 작업 크기는 Windows 컨테이너에서 지원되지 않습니다.

작업 메모리(MiB): 2048

The amount of memory (in MiB) used by the task. It can be expressed as an integer using MiB, for example 1024, or as a string using GB, for example '1GB' or '1 gb'.

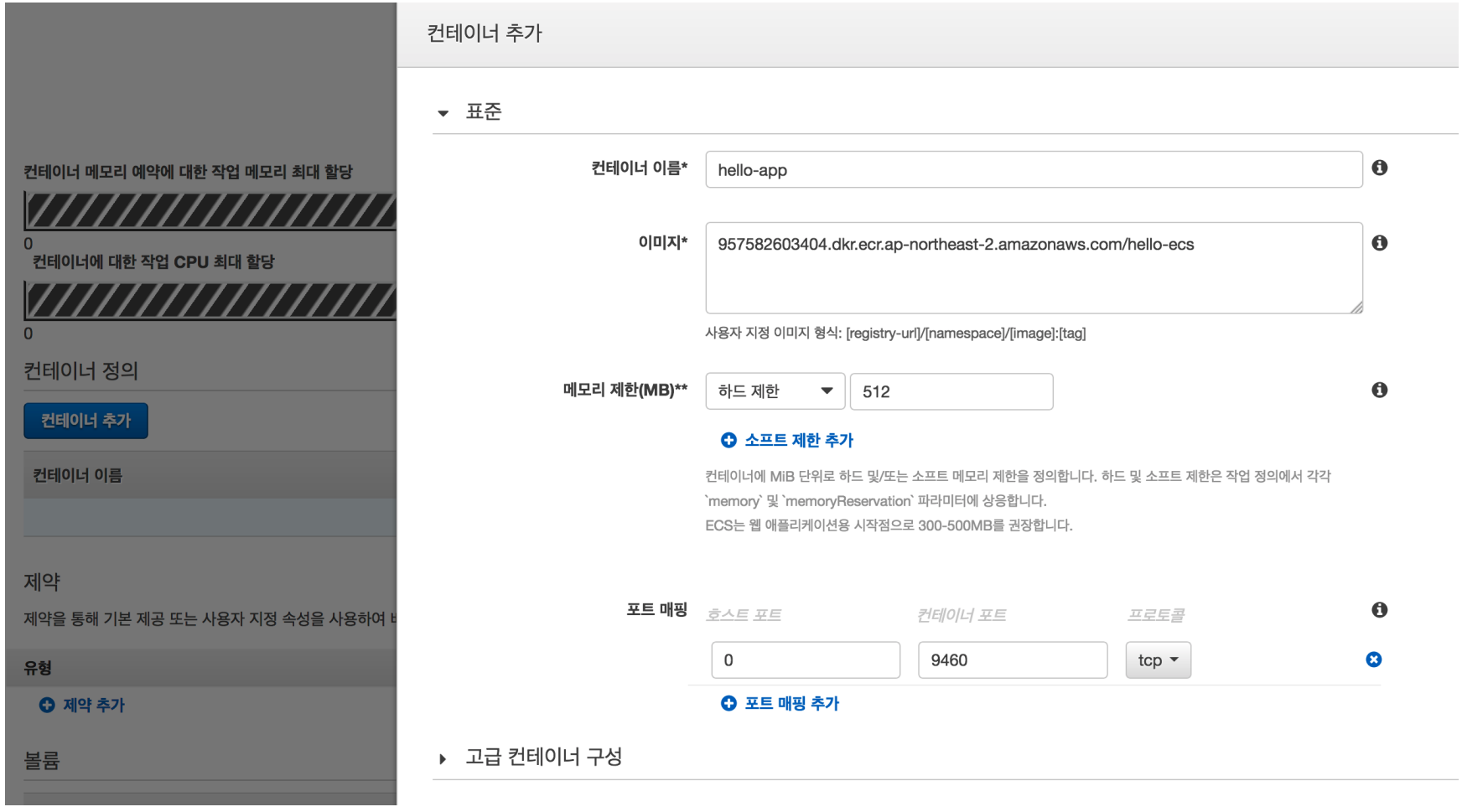
작업 CPU(유닛): 1024

The number of CPU units used by the task. It can be expressed as an integer using CPU units, for example 1024, or as a string using vCPUs, for example '1 vCPU' or '1 vcpu'.

컨테이너 추가

⚠️ 호스트 포트는 0을 지정한다. 이렇게 되면 호스트 포트는 랜덤한 포트로 지정이 되고 나중에 ELB가 알아서 그 포트로 라우팅 해주게 된다.

만약 포트를 그냥 지정해버리면 업데이트할 때 포트가 충돌이 나서 컨테이너를 모두 stop 했다가 다시 올려야한다.



생성!

컨테이너 정의

컨테이너 추가

컨테이너 이름	이미지	하드/소프트 메모리 제한(MB)	CPU 단위	필수	
hello-app	957582603404.dkr.ecr.ap-northeast-2.amazonaws.co…	512/--		true	

제약

제약을 통해 기본 제공 또는 사용자 지정 속성을 사용하여 배치 전략에 사용되는 인스턴스를 필터링할 수 있습니다. 스케줄러는 먼저 제약과 일치하는 인스턴스를 필터링한 후 배치 전략을 적용하여 작업을 배치합니다.

유형	표현식

제약 추가

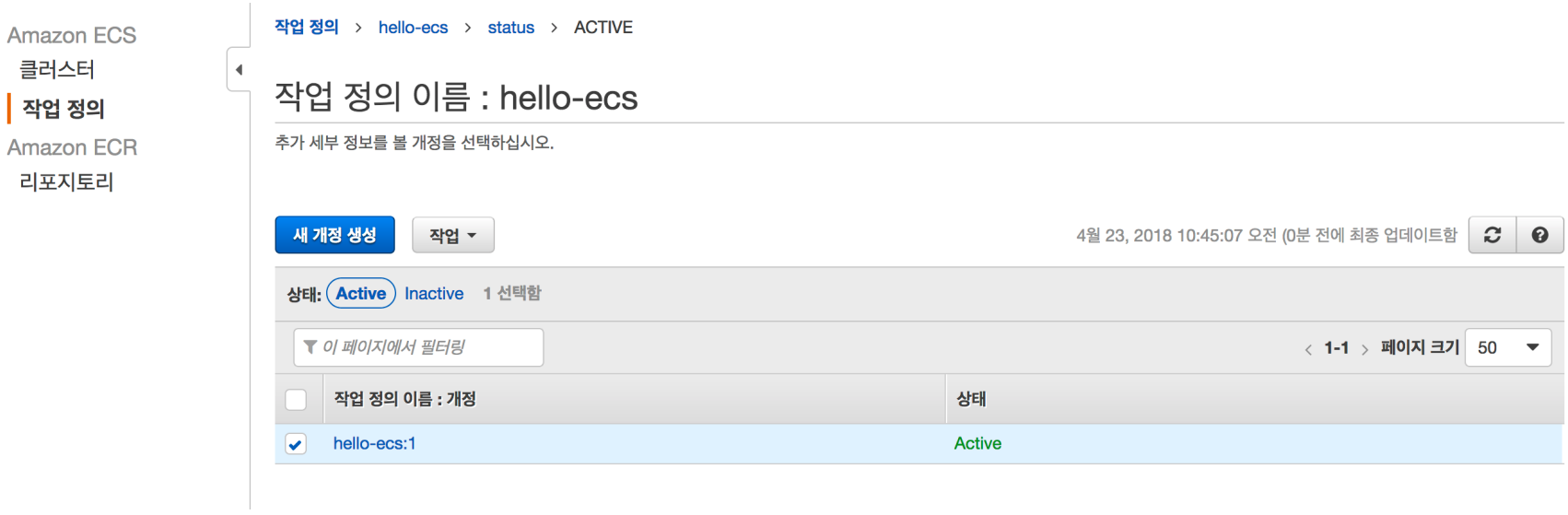
볼륨

이름	소스 경로	
결과 없음		

볼륨 추가

JSON을 통한 구성

취소 생성

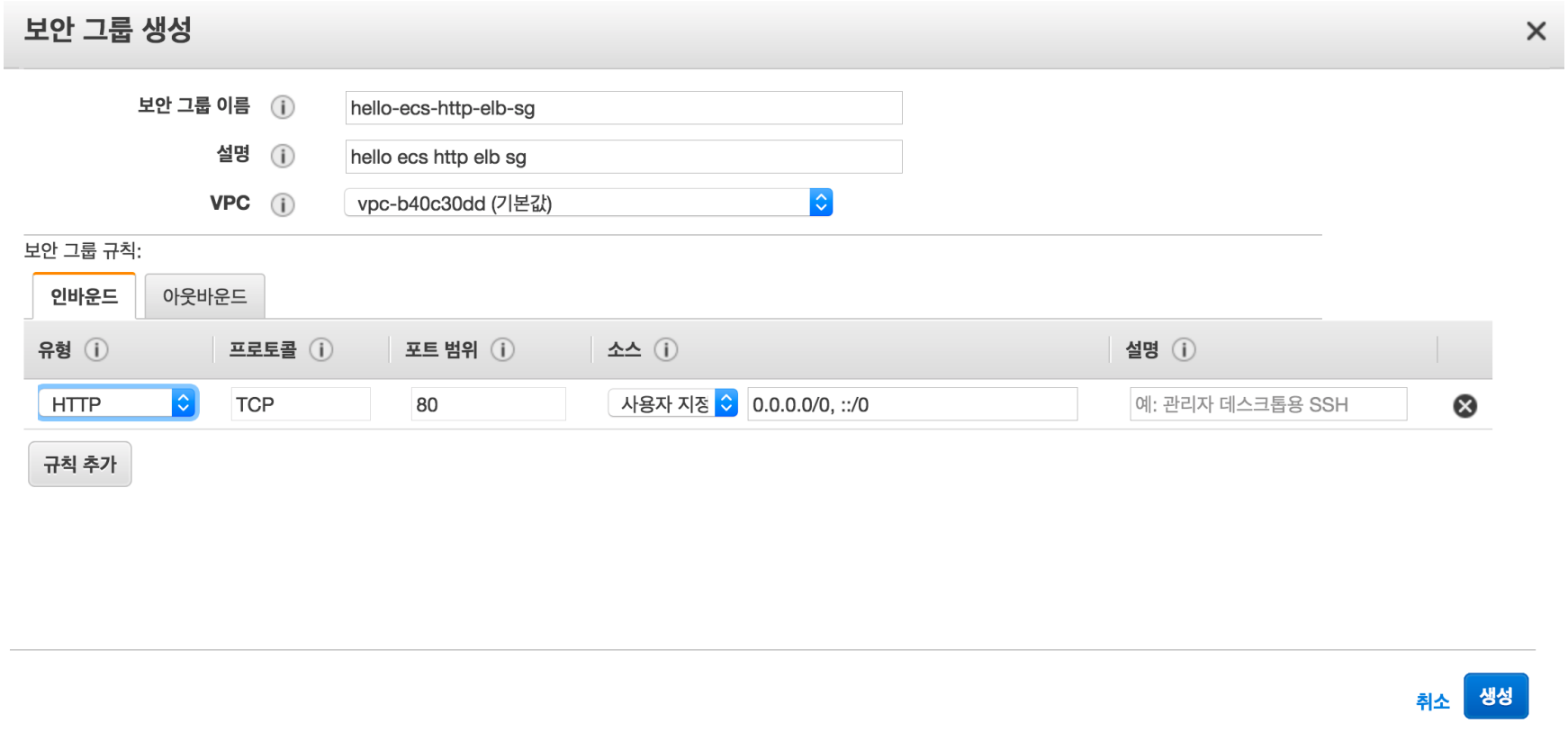

Amazon ECS
클러스터
작업 정의
Amazon ECR
리포지토리
작업 정의 > hello-ecs > status > ACTIVE
작업 정의 이름 : hello-ecs
추가 세부 정보를 볼 개정을 선택하십시오.
새 개정 생성
작업
4월 23, 2018 10:45:07 오전 (0분 전에 최종 업데이트함
상태: Active Inactive 1 선택함
이 페이지에서 필터링
1-1
페이지 크기 50
작업 정의 이름 : 개정
상태
hello-ecs:1
Active

3. 보안그룹(Security Group) 생성

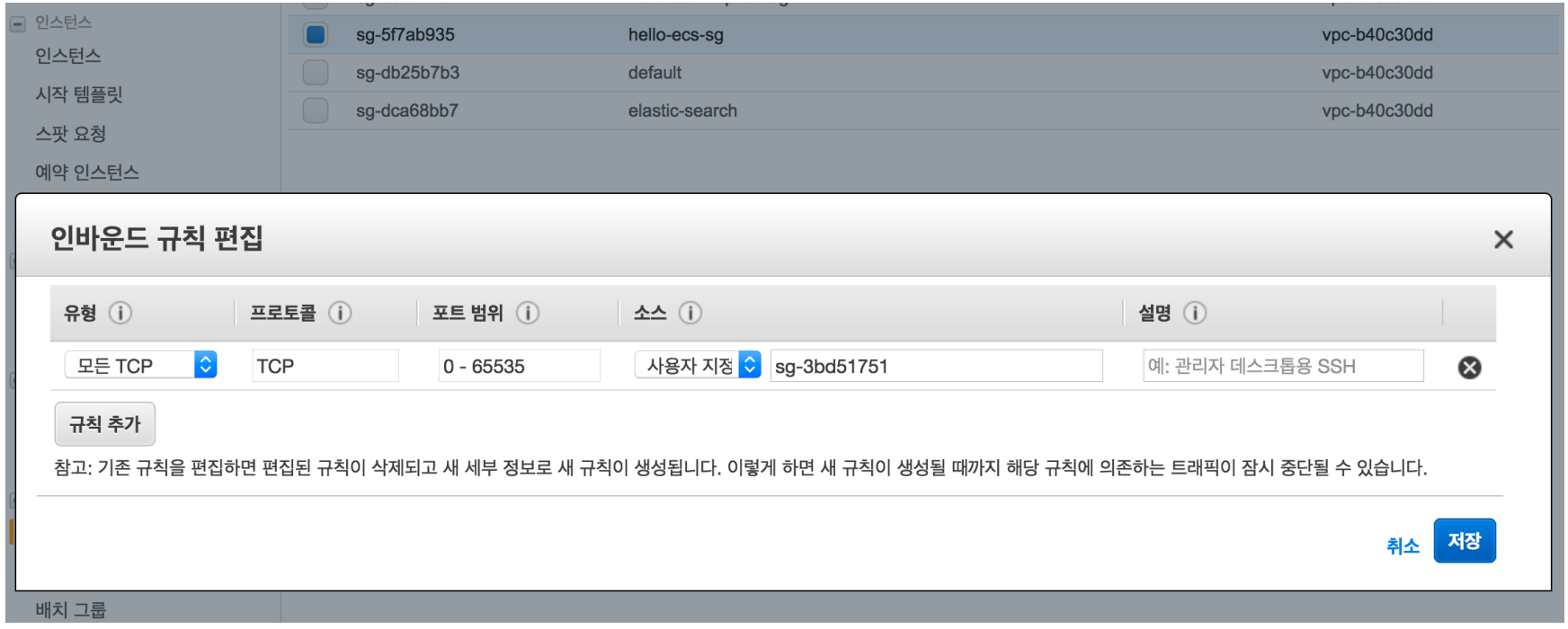
https://ap-northeast-2.console.aws.amazon.com/ec2/v2/home?region=ap-northeast-2#SecurityGroups:sort=vpcId

미리 보안 그룹을 만들어 둔다. 나중에 ELB를 붙일때 굉장히 편하다.

3.1 ELB 용 보안그룹



3.2 ECS instance용 보안그룹

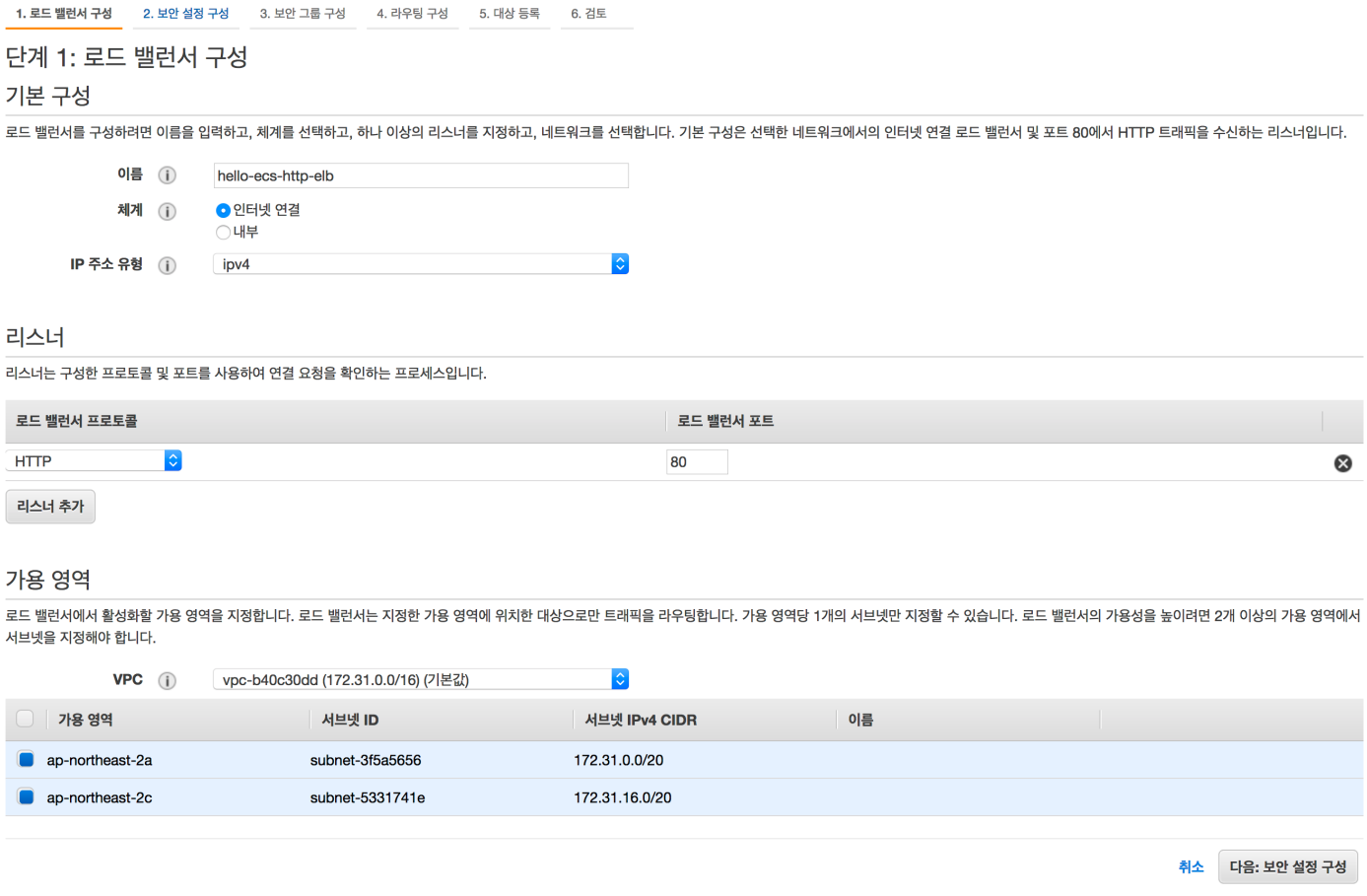


- 소스는 elb 보안 그룹

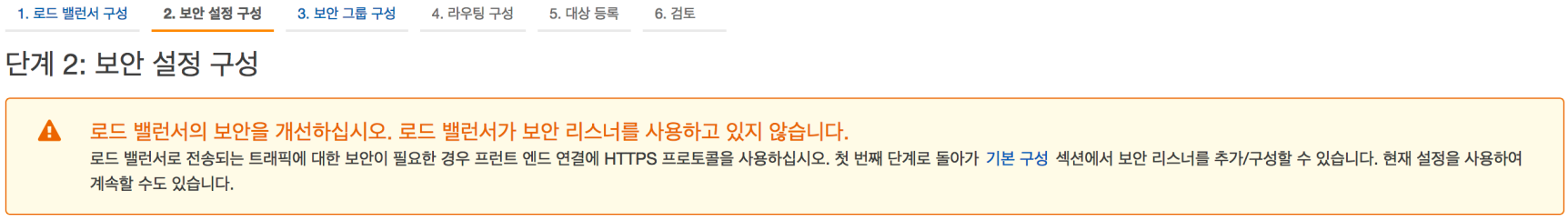
4. ELB와 대상 그룹 생성

https://ap-northeast-2.console.aws.amazon.com/ec2/v2/home?region=ap-northeast-2#LoadBalancers:sort=loadBalancerName

applicaton Load Balancer, HTTP



Skip!


1. 로드 밸런서 구성
2. 보안 설정 구성
3. 보안 그룹 구성
4. 라우팅 구성
5. 대상 등록
6. 검토
단계 2: 보안 설정 구성
로드 밸런서의 보안을 개선하십시오. 로드 밸런서가 보안 리스너를 사용하고 있지 않습니다.
로드 밸런서로 전송되는 트래픽에 대한 보안이 필요한 경우 프런트 엔드 연결에 HTTPS 프로토콜을 사용하십시오. 첫 번째 단계로 돌아가 기본 구성 섹션에서 보안 리스너를 추가/구성할 수 있습니다. 현재 설정을 사용하여 계속할 수도 있습니다.

만들어둔 보안 그룹 선택

1. 로드 밸런서 구성 | 2. 보안 설정 구성 | **3. 보안 그룹 구성** | 4. 라우팅 구성 | 5. 대상 등록 | 6. 검토

단계 3: 보안 그룹 구성

보안 그룹은 로드 밸런서에 대한 트래픽을 제어하는 방화벽 규칙 세트입니다. 이 페이지에서는 특정 트래픽을 로드 밸런서에 도달하도록 허용할 규칙을 추가할 수 있습니다. 먼저 새 보안 그룹을 생성할지 아니면 기존 보안 그룹을 선택할지 결정합니다.

보안 그룹 할당: ○ **새** 보안 그룹 생성

● **기존** 보안 그룹 선택

필터 VPC 보안 그룹

	보안 그룹 ID	이름	설명	작업
☐	sg-db25b7b3	default	default VPC security group	새로 복사
☐	sg-dca68bb7	elastic-search	elastic-search	새로 복사
☑	sg-3bd51751	hello-ecs-http-elb-sg	hello ecs http elb sg	새로 복사
☐	sg-5f7ab935	hello-ecs-sg	hello ecs security group	새로 복사

상태 검사 경로에는 상태 체크용 경로 지정

대상 등록은 나중에

1. 로드 밸런서 구성 | 2. 보안 설정 구성 | 3. 보안 그룹 구성 | 4. 라우팅 구성 | 5. 대상 등록 | 6. 검토

단계 5: 대상 등록

대상 그룹에 대상을 등록합니다. 활성화된 가용 영역에서 대상을 등록할 경우, 로드 밸런서는 등록 프로세스가 완료되고 대상이 초기 상태 검사를 통과하자마자 해당 대상으로 요청을 라우팅하기 시작합니다.

등록된 대상

인스턴스를 등록 취소하려면 등록된 인스턴스를 하나 이상 선택한 다음 제거를 클릭합니다.

제거

인스턴스	이름	포트	상태	보안 그룹	영역

사용 가능한 인스턴스가 없습니다.

인스턴스

추가 인스턴스를 등록하려면 하나 이상의 실행 인스턴스를 선택하고 포트를 지정한 다음 추가를 클릭하십시오. 기본 포트는 대상 그룹에 지정된 포트입니다. 인스턴스가 지정된 포트에 이미 등록되어 있는 경우 다른 포트를 지정해야 합니다.

등록된 항목에 추가 | 포트: 80

인스턴스 검색

인스턴스	이름	상태	보안 그룹	영역	서브넷 ID	서브넷 CIDR
i-044eb0441d4dbe...	ECS Instance - EC...	running	hello-ecs-sg	ap-northeast-2a	subnet-3f5a5656	172.31.0.0/20
i-061f7e81630973...	ECS Instance - EC...	running	hello-ecs-sg	ap-northeast-2c	subnet-5331741e	172.31.16.0/20

1. 로드 밸런서 구성 | 2. 보안 설정 구성 | 3. 보안 그룹 구성 | 4. 라우팅 구성 | 5. 대상 등록 | 6. 검토

단계 6: 검토

계속하기 전에 로드 밸런서 세부 정보를 검토하십시오.

로드 밸런서 편집

이름	hello-ecs-http-elb
체계	internet-facing
리스너	포트:80 - 프로토콜:HTTP
IP 주소 유형	ipv4
VPC	vpc-b40c30dd
서브넷	subnet-3f5a5656, subnet-5331741e
태그	

보안 그룹 편집

보안 그룹	sg-3bd51751

라우팅 편집

대상 그룹	새 대상 그룹
대상 그룹 이름	hello-service
포트	80
대상 유형	instance
프로토콜	HTTP
상태 검사 프로토콜	HTTP
경로	/
상태 검사 포트	traffic port
정상 임계 값	5
비정상 임계값	2
제한 시간	5
간격	30
성공 코드	200

대상 편집

인스턴스	

취소 | 이전 | 생성

로드 밸런서 생성 상태

성공적으로 로드 밸런서 생성 완료

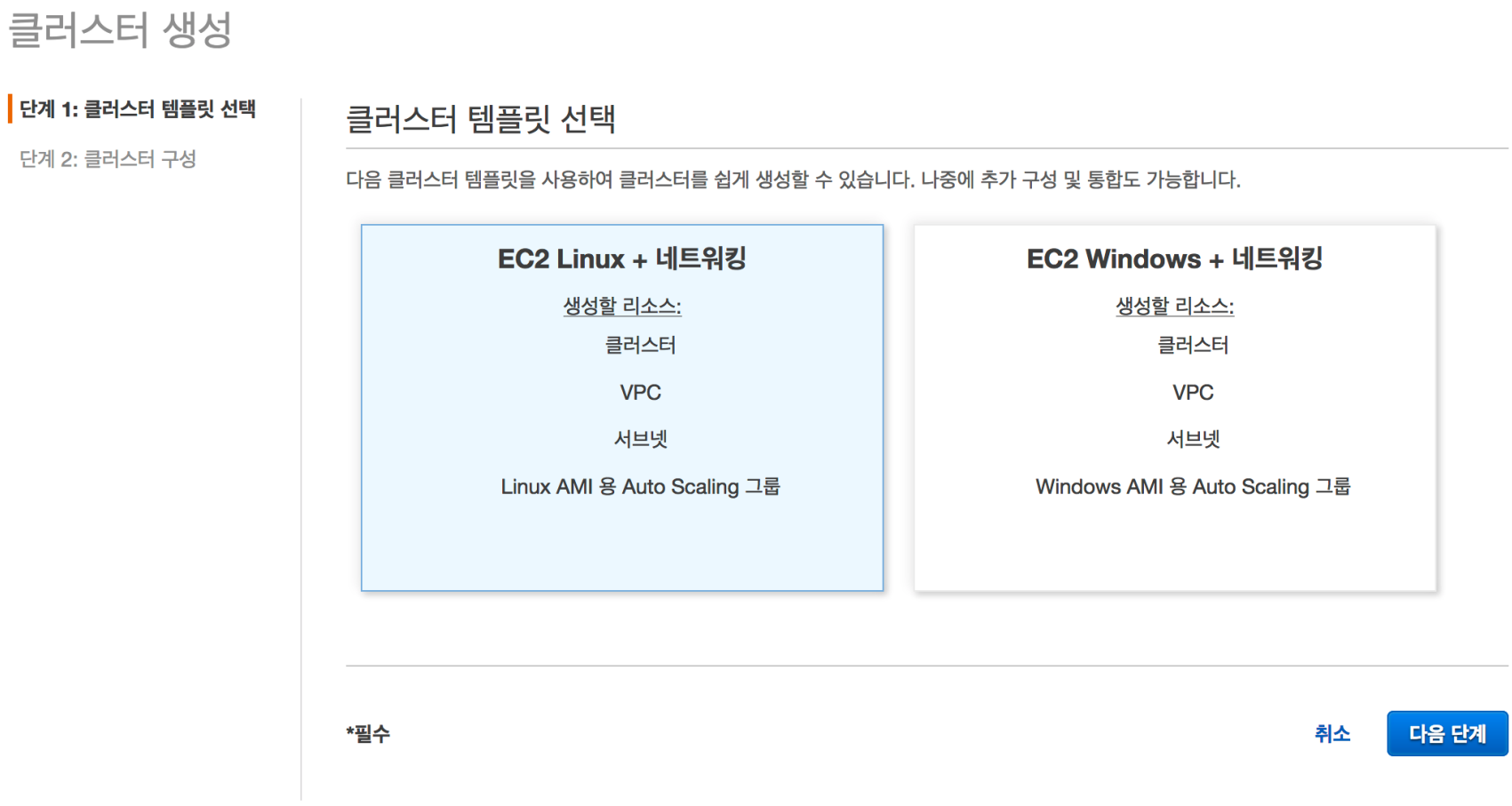
성공적으로 로드 밸런서 hello-ecs-http-elb 을(를) 생성했습니다.

참고: 로드 밸런서가 완전히 설정되어 트래픽을 라우팅할 준비가 완료되고 대상이 등록 프로세스를 완료한 후 초기 상태 검사를 통과하려면 몇 분이 걸릴 수 있습니다.

닫기

5. Cluster 생성

잘 모르겠지만 오른쪽은 하면 안될것 같음...



클러스터 구성

클러스터 이름* hello-ecs

☐ 빈 클러스터 생성

인스턴스 구성

프로비저닝 모델

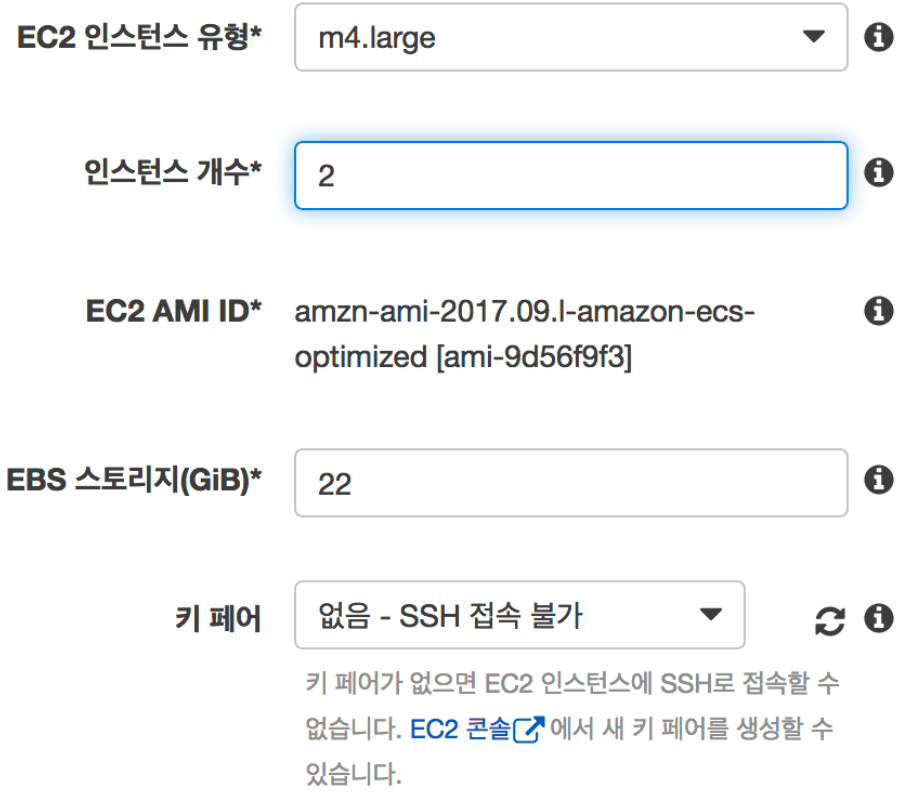
◉ 온디맨드 인스턴스

온디맨드 인스턴스를 사용하면 장기 약정이나 선결제 금액 없이 시간당 컴퓨팅 파워에 대한 요금을 지불할 수 있습니다.

○ 스팟

Amazon EC2 스팟 인스턴스를 사용하면 온디맨드 가격에서 최대 90% 할인된 금액으로 예비 Amazon EC2 컴퓨팅 용량에 입찰할 수 있습니다. 자세히 알아보기

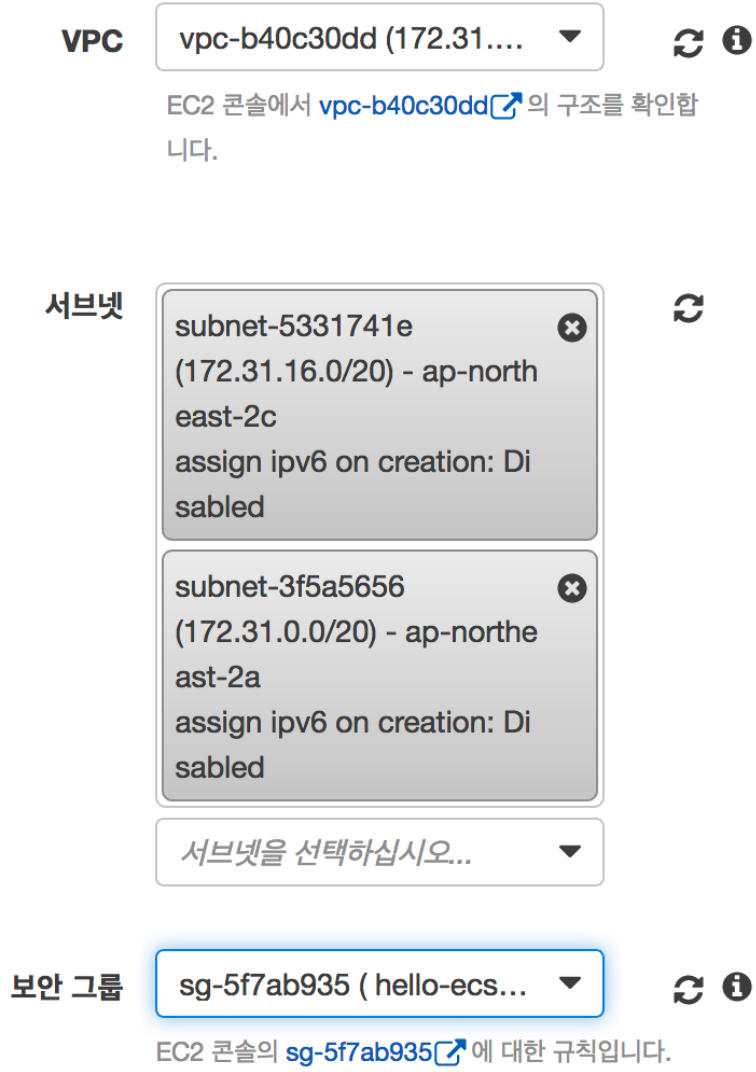
인스턴스는 두개!


EC2 인스턴스 유형*
m4.large
인스턴스 개수*
2
EC2 AMI ID*
amzn-ami-2017.09.l-amazon-ecs-optimized [ami-9d56f9f3]
EBS 스토리지(GiB)*
22
키 페어
없음 - SSH 접속 불가
키 페어가 없으면 EC2 인스턴스에 SSH로 접속할 수 없습니다. EC2 콘솔에서 새 키 페어를 생성할 수 있습니다.

보안 그룹은 만들어둔 걸 사용

네트워킹

컨테이너 인스턴스가 사용할 VPC를 구성합니다. VPC는 AWS 클라우드의 격리된 부분으로서, Amazon EC2 인스턴스와 같은 AWS 객체로 채워집니다. 기존 VPC를 선택하거나 이 마법사로 새 VPC를 만들 수 있습니다.



컨테이너 인스턴스 IAM 역할

Amazon ECS 컨테이너 에이전트는 사용자를 대신하여 Amazon ECS API 작업을 호출하므로 에이전트가 사용자에게 속한다는 것을 서비스가 알기 위해서는 에이전트를 실행하는 컨테이너 인스턴스에 ecsInstanceRole IAM 정책과 역할이 필요합니다. ecsInstanceRole이 아직 없는 경우, AWS가 생성할 수 있습니다.

컨테이너 인스턴스 IAM 역할 ecsInstanceRole

***필수**

취소 이전 생성

시작 상태

컨테이너 인스턴스를 시작하고 있으며 실행 상태가 되어 액세스할 준비가 완료될 때까지 몇 분이 걸릴 수 있습니다. 새 컨테이너 인스턴스에서는 사용 시간이 즉시 시작되어 인스턴스를 중지 또는 종료할 때까지 계속 누적됩니다.

뒤로 **클러스터 보기**

ECS 상태 - 3 중 3 완료 **hello-ecs**

ECS 클러스터
hello-ecs ECS 클러스터 생성 완료

IAM 정책
ecsInstanceRole 역할에 대한 IAM 정책 연결 완료

CloudFormation 스택
EC2ContainerService-hello-ecs CloudFormation 스택과 그 리소스 생성 완료

클러스터 리소스

인스턴스 유형	m4.large
원하는 인스턴스 개수	2
키 페어	
ECS AMI ID	ami-9d56f9f3
VPC	vpc-b40c30dd
서브넷	subnet-5331741e, subnet-3f5a5656
VPC 가용 영역	ap-northeast-2a, ap-northeast-2c
보안 그룹	sg-5f7ab935
실행 구성	EC2ContainerService-hello-ecs-EcsInstanceLc-1PP0S5UKOT2PC
Auto Scaling 그룹	EC2ContainerService-hello-ecs-EcsInstanceAsg-1OH4N7X7GD2T6

서비스는 아직 없음

클러스터 : hello-ecs

클러스터 삭제

클러스터에 있는 리소스의 현황을 상세히 볼 수 있습니다.

상태	ACTIVE
등록된 컨테이너 인스턴스	2
대기 중인 작업 개수	0
실행 중인 작업 개수	0

서비스 | 작업 | ECS 인스턴스 | 측정치 | 예약된 작업

생성 | 업데이트 | 삭제

4월 23, 2018 12:04:24 오전 (0분 전에 최종 업데이트함

이 페이지에서 필터링

서비스 이름	상태	작업 정의	원하는 작업	실행 중인 작업
결과 없음				

6. Service 생성

서비스 생성

- **단계 1: 서비스 구성**
- 단계 2: 네트워크 구성
- 단계 3: Auto Scaling (선택사항)
- 단계 4: 서비스 검토

서비스 구성

서비스를 통해 클러스터에서 실행하고 유지 관리할 작업 정의의 사본 개수를 지정할 수 있습니다. Elastic Load Balancing 로드 밸런서를 옵션으로 사용하여 들어오는 트래픽을 서비스 내 컨테이너에 분산할 수 있습니다. Amazon ECS는 로드 밸런서를 통해 작업의 개수를 유지하고 작업 일정을 조정합니다. 서비스 Auto Scaling을 옵션으로 사용하여 서비스 내 작업의 개수를 조정할 수도 있습니다.

작업 정의 hello-ecs:1

클러스터 hello-ecs

서비스 이름 hello-service

작업 개수 2

최소 정상 상태 백분율 50

최대 백분율 200

작업 배치

클러스터 내 인스턴스에 작업이 배치되는 방식을 사용자 지정할 수 있게 해줍니다. 다양한 배치 전략을 사용하여 최적화함으로써 가용성 및 효율성을 높일 수 있습니다.

배치 템플릿 AZ 균형 분산 편집

이 템플릿은 가용 영역 전반에 걸쳐 작업을 분산하고 가용 영역 내에서는 인스턴스에 두루 작업을 분산합니다. 자세히 알아보기.

전략: spread(attribute:ecs.availability-zone), spread(instanceId)

*필수 취소 다음 단계

서비스 생성

단계 1: 서비스 구성

단계 2: 네트워크 구성

단계 3: Auto Scaling (선택사항)

단계 4: 서비스 검토

네트워크 구성

VPC 및 보안 그룹

VPC 및 보안 그룹은 작업 정의에 awsvpc 네트워크 모드가 있는 경우 구성 가능합니다.

상태 검사 유예 기간

서비스 작업을 시작하고 ELB 상태 검사에 응답하는 데 시간이 다소 걸린다면 최대 1,800초의 상태 검사 유예 기간을 지정할 수 있습니다. 이 기간에는 ECS 서비스 스케줄러가 ELB 상태 검사의 상태를 무시합니다. 이 유예 기간은 ECS 서비스 스케줄러가 작업을 비정상으로 표시하여 작업 처리 전에 이를 중단시키는 일이 없도록 해줍니다. 이는 서비스가 로드 밸런서를 사용하도록 구성된 경우에만 유효합니다.

상태 검사 유예 기간 30

Elastic Load Balancing(선택 사항)

Elastic Load Balancing 로드 밸런서는 들어오는 트래픽을 서비스에서 실행 중인 작업에 두루 분산합니다. 기존 로드 밸런서를 선택하거나 Amazon EC2 콘솔에서 새 로드 밸런서를 생성합니다. 로드 밸런서 구성을 마쳤으면 **저장**을 선택하여 작업을 계속합니다.

ELB 유형:

- ○ 없음
  해당 서비스에서는 로드 밸런서를 사용하지 않습니다.
- ◉ Application Load Balancer
  컨테이너가 동적 호스트 포트 매핑을 사용하도록 허용합니다(컨테이너 인스턴스마다 다중 작업이 허용됨). 여러 서비스가 규칙 기반 라우팅 및 경로를 사용하여 단일 로드 밸런서에서 동일한 리스너 포트를 사용할 수 있습니다.
- ○ Network Load Balancer
  A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model. After the load balancer receives a request, it selects a target from the target group for the default rule using a flow hash routing algorithm.
- ○ Classic Load Balancer
  정적 호스트 포트 매핑을 필요로 합니다(컨테이너 인스턴스당 1개의 작업만 허용됨). 규칙 기반 라우팅 및 경로가 지원되지 않습니다.

서비스의 IAM 역할 선택

Task definitions that use the awsvpc network mode use the AWSServiceRoleForECS service-linked role, which is created for you automatically. Learn more.

서비스의 IAM 역할 선택 ecsServiceRole

ELB 이름 hello-ecs-http-elb

로드를 밸런싱할 컨테이너

hello-app : 9460

제거

리스너 포트 80:HTTP

리스너 프로토콜 HTTP

대상 그룹 이름 hello-service

대상 그룹 프로토콜 HTTP

대상 유형 instance

경로 패턴 /

평가 순서 default

상태 확인 경로 /hello

서비스 생성 후 ELB 콘솔에 추가 상태 확인 옵션을 구성할 수 있습니다.

*필수

취소 이전 다음 단계

Auto Scaling (선택사항)

CloudWatch 경보에 대응하여 지정한 범위 내에서 원하는 서비스 개수를 자동으로 늘리거나 줄입니다. 언제든지 서비스 Auto Scaling 구성을 수정하여 애플리케이션의 요구 사항을 충족할 수 있습니다.

서비스 Auto Scaling

- (●) 원하는 서비스 개수를 조정하지 마십시오.
- ( ) 서비스 Auto Scaling을 구성하여 원하는 서비스 개수를 조정합니다.

***필수**

취소 | 이전 | 다음 단계

클러스터	hello-ecs
작업 정의	hello-ecs:1
서비스 이름	hello-service
작업 개수	2
최소 정상 상태 백분율	50
최대 백분율	200

네트워크 구성

편집

상태 검사 유예 기간	30
컨테이너 이름:	hello-app
컨테이너 포트:	9460
ELB 이름:	hello-ecs-http-elb
대상 그룹:	hello-service
상태 확인 경로:	/hello
리스너 포트:	80
경로-패턴:	/
서비스 역할:	ecsServiceRole

Auto Scaling (선택사항)

편집

시작 상태

ECS 서비스 상태 - 2 중 2 완료

로드 밸런서 생성

IAM 정책

IAM 서비스 정책 연결 완료

IAM 서비스 정책 arn:aws:iam::957582603404:role/ecsServiceRole 역할에 연결 완료. 정책 보기: AmazonEC2ContainerServiceRole

서비스 생성

서비스 생성: hello-service

서비스 생성 완료

서비스 생성 완료. 잠시 후 작업을 시작합니다. 보기: hello-service

ECS 서비스에 연결할 수 있는 추가 통합

코드 파이프라인

서비스에서 CI/CD 프로세스를 설정합니다. 소스를 통해 빌드하거나 ECR 리포지토리를 배포용 소스로 사용할 수 있습니다.

파이프라인 생성

뒤로 서비스 보기

세부 정보 | 작업 | 이벤트 | Auto Scaling | 배포 | 측정치

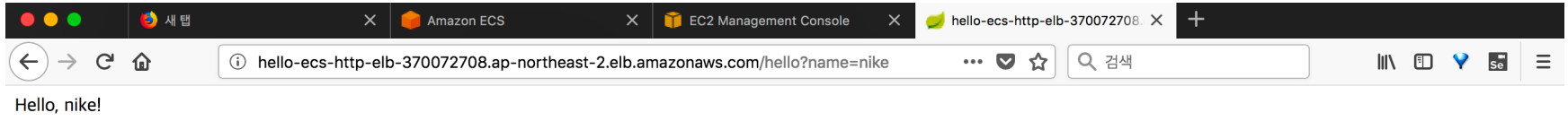
4월 23, 2018 10:43:19 오전 (0분 전에 최종 업데이트함

이 페이지에서 필터링

1-2

이벤트 ID	이벤트 시간	메시지
a15018f9-e56d-4469-a997-358f874ebcd3	2018-04-23 10:43:05 +0900	service hello-service registered 2 targets in target-group hello-service
8f79c5a4-532e-42ae-a1af-374e9473c2d5	2018-04-23 10:42:54 +0900	service hello-service has started 2 tasks: task 80d183f1-c2b4-4e26-8a3b-e5dee904b8c2 task fc026861-502c-489a-b364-c06609f622ad.

7. 반영 확인


새 탭
Amazon ECS
EC2 Management Console
hello-ecs-http-elb-370072708.
hello-ecs-http-elb-370072708.ap-northeast-2.elb.amazonaws.com/hello?name=nike
검색
Hello, nike!

8. 웹 콘솔 결론

- 복잡하다.
- 역시 개발자는 코드로
- 그래도 컨셉은 대충 이해할 수 있었다.

ECS CLI 실습

준비

- credentials
- ECS CLI 설치
- ELB와 대상그룹은 아까처럼 하나더 만들어 둔다. (보안 그룹은 그대로)

1. configure

```
ecs-cli configure --cluster hello-ecs-cli \
  --region ap-northeast-2 \
  --default-launch-type EC2 \
  --config-name hello-ecs-cli

ecs-cli configure profile \
  --access-key ${AWS_ACCESS_KEY_ID} \
  --secret-key ${AWS_SECRET_ACCESS_KEY} \
  --profile-name  hello-ecs-cli
```

2. Cluster 생성

```
ecs-cli up --keypair voyager.woo \
  --security-group sg-5f7ab935 \
  --cluster hello-ecs-cli \
  --vpc vpc-b40c30dd \
  --subnets subnet-5331741e,subnet-3f5a5656 \
  --capability-iam --size 2 \
  --instance-type t2.medium
```

⚠️ security group은 이름이 아닌 id로 넣어야한다! 이름을 넣으면 클러스터는 생기는데 인스턴스가 안생김...

3. Task를 정의하는 docker compose 작성

```
version: '2'
services:
  hello-ecs-app:
    image: ${ECR_HOST}/hello-ecs:latest
    cpu_shares: 100
    mem_limit: 524288000
    ports:
      - "0:9460"
    logging:
      driver: awslogs
      options:
        awslogs-group: hello-ecs-cli
        awslogs-region: ap-northeast-2
        awslogs-stream-prefix: wordpress
```

4. compose test

```
# test compose container up
ecs-cli compose --file hello-compose.yml up \
  --create-log-groups --cluster hello-ecs-cli

# test compose scale up
ecs-cli compose --file hello-compose.yml scale 2 \
  --cluster hello-ecs-cli

# shutdown compose container
ecs-cli compose --file hello-compose.yml down \
  --cluster hello-ecs-cli
```

5. Service 생성

만들어둔 대상 그룹의 ARN 으로 Load balancer에 연결한다.

```
# 서비스 생성
ecs-cli compose --file hello-compose.yml service \
  create --cluster hello-ecs-cli \
  --deployment-max-percent 200 \
  --deployment-min-healthy-percent 50 \
  --target-group-arn "${TARGET_GROUP_ARN}" \
  --health-check-grace-period 30 \
  --container-name hello-ecs-app \
  --container-port 9460 \
  --create-log-groups
```

6. Service 실행

서비스를 띄우고 컨테이너 개수를 2개로 늘린다.

```
ecs-cli compose --file hello-compose.yml service up \
  --cluster hello-ecs-cli

ecs-cli compose --file hello-compose.yml service scale 2 \
  --cluster hello-ecs-cli
```

7. Service 업데이트

```
ecs-cli compose --file hello-compose.yml service up \
  --cluster hello-ecs-cli --force-deployment
```