로드 밸런서 구성

IAM Policy 생성

AWS Load Balancer Controller를 위한 IAM Policy를 다운로드한다.

```
curl -o iam_policy.json https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.4.4/docs/install/iam_policy.
```

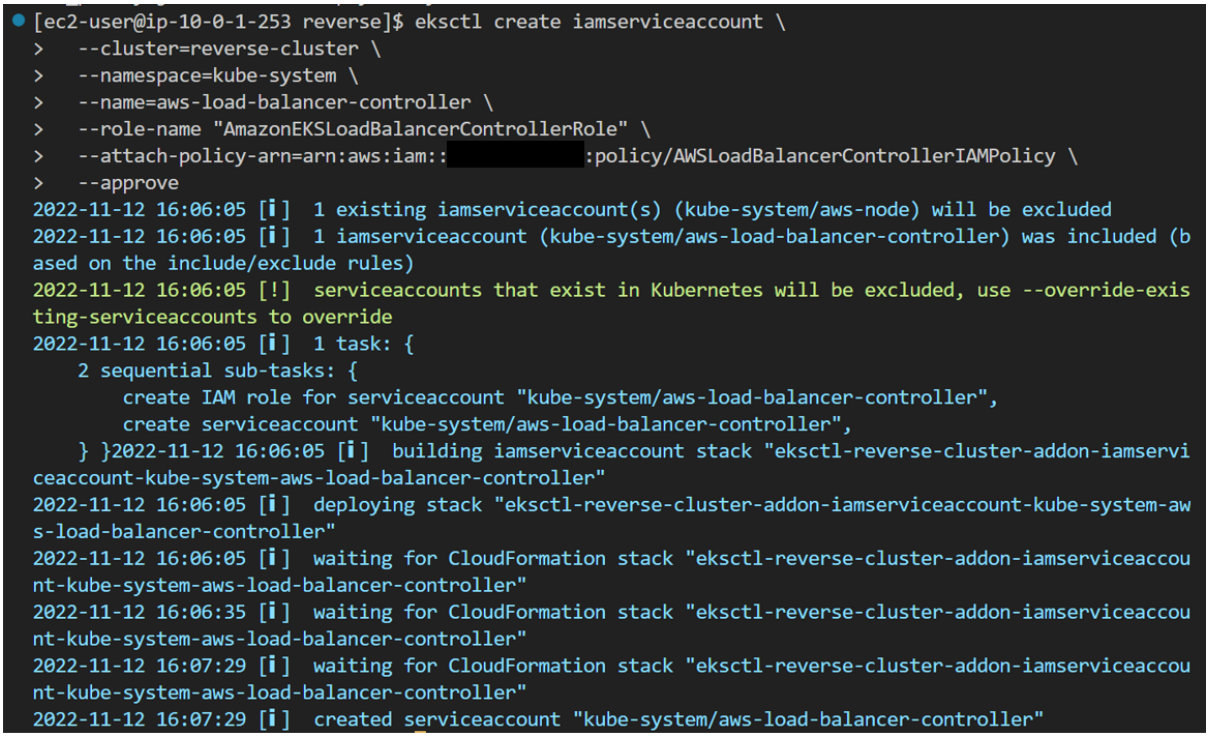
IAM policy를 생성한다.

```
aws iam create-policy \
    --policy-name AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://iam_policy.json
```

IAM Role 및 Kubernetes의 Service Account 생성

위에서 생성한 IAM Policy를 이용하여 아래와 같이 생성한다.

```
eksctl create iamserviceaccount \
  --cluster=reverse-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name "AmazonEKSLoadBalancerControllerRole" \
  --attach-policy-arn=arn:aws:iam::<계정넘버>:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```



Helm 설치 및 repository 추가

AWS Load Balancer Controller를 Helm을 통해 설치하기 때문에 Helm도 설치하고 eks-charts repository도 추가한다.

```
curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 > get_helm.sh
chmod 700 get_helm.sh
./get_helm.sh
```

```
[ec2-user@ip-10-0-1-253 reverse]$ curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 > get_helm.sh
et_helm.sh
./get_helm.sh
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11156  100 11156    0     0  35562      0 --:--:-- --:--:-- --:--:-- 35528
[ec2-user@ip-10-0-1-253 reverse]$ chmod 700 get_helm.sh
[ec2-user@ip-10-0-1-253 reverse]$ ./get_helm.sh
Downloading https://get.helm.sh/helm-v3.10.2-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
```

설치한 Helm 버전을 확인한다.

```
helm version --short | cut -d + -f 1
```

```
[ec2-user@ip-10-0-1-253 reverse]$ helm version --short | cut -d + -f 1
v3.10.2
```

이제 Helm 명령을 실행하여 클러스터에서 Helm 차트를 설치, 수정, 삭제 또는 쿼리할 수 있다.

Helm을 사용하여 AWS Load Balancer Controller를 설치한다.

1. eks-charts 레포지토리를 추가한다.

   ```
   helm repo add eks https://aws.github.io/eks-charts
   ```

   ```
   [ec2-user@ip-10-0-1-253 reverse]$ helm repo add eks https://aws.github.io/eks-charts
   "eks" has been added to your repositories
   ```

2. 최신 차트가 적용되도록 로컬 레포지토리를 업데이트한다.

   ```
   helm repo update
   ```

   ```
   [ec2-user@ip-10-0-1-253 reverse]$ helm repo update
   Hang tight while we grab the latest from your chart repositories...
   ...Successfully got an update from the "eks" chart repository
   Update Complete. ⎈Happy Helming!⎈
   ```

3. AWS ECR 생성 및 ECR에 이미지 푸시

   ▼ ECR

노드가 Amazon EKS Amazon ECR 이미지 레포지토리에 액세스할 수 없는 경우 다음 컨테이너 이미지를 가져와서 노드가 액세스할 수 있는 레포지토리로 푸시해야한다.

```
602401143452.dkr.ecr.ap-northeast-2.amazonaws.com/amazon/aws-load-balancer-controller:v2.4.4
```

4. AWS Load Balancer Controller를 설치한다. Amazon EC2 인스턴스 메타데이터 서비스(IMDS)에 대해 제한적인 액세스 권한이 있는 Amazon EC2 노드에 컨트롤러를 배포하거나 Fargate에 배포하는 경우, 다음 `helm` 명령에 다음 플래그를 추가한다.

   - `--set region=region-code`
   - `--set vpcId=vpc-xxxxxxxx`

```
helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=reverse-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller \
  --set image.repository=602401143452.dkr.ecr.ap-northeast-2.amazonaws.com/amazon/aws-load-balancer-controller
```

```
[ec2-user@ip-10-0-1-253 reverse]$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
>   -n kube-system \
>   --set clusterName=reverse-cluster \
>   --set serviceAccount.create=false \
>   --set serviceAccount.name=aws-load-balancer-controller \
>   --set image.repository=602401143452.dkr.ecr.ap-northeast-2.amazonaws.com/amazon/aws-load-balancer-controller
NAME: aws-load-balancer-controller
LAST DEPLOYED: Sat Nov 12 16:34:35 2022
NAMESPACE: kube-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
AWS Load Balancer controller installed!
```

controller가 잘 설치되었는지 확인

```
kubectl get deployment -n kube-system aws-load-balancer-controller
```

```
[ec2-user@ip-10-0-1-253 reverse]$ kubectl get deployment -n kube-system aws-load-balancer-controller
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           2m30s
```

도메인 구입

가비아에서 도메인 re-verse.kr로 구입했음.

퍼블릭 인증서 요청

AWS Route53 으로 도메인(https) 연결하기

기존에 AWS ElasticBeanstalk으로 배포한 웹앱에 구입한 도메인을 연결할 필요가 있어 AWS Route53으로 도메인을 연결하였습니다. AWS Route53에서도 도메인 구매가 가능하나 제가 사용하려는 도메인이 AWS에서는 사용이 불가하여 가비아에서 구매했습니다. 다른 곳에서 구매하셨어도 전반적인 진행은 유사할 것입니다.

https://omty.tistory.com/47

퍼블릭 인증서 요청

다음 섹션에서는 ACM 콘솔 또는 AWS CLI를 사용하여 퍼블릭 ACM 인증서를 요청하는 방법을 설명합니다. 공인 인증서를 요청한 후에는 도메인 소유권 확인 에 설명된 절차 중 하나를 완료해야 합니다. 인증서를 요청할 때 문제가 발생하면 인증서 요청 문제 해결 단원을 참조하세요. AWS Private CA를 사용하여 프라이빗 PKI에 대한 인증서를 요청하려면 프라이빗 PKI 인증서 요청 섹션을 참조하세요.

https://docs.aws.amazon.com/ko_kr/acm/latest/userguide/gs-acm-request-public.html

우리의 경우 re-verse.kr 과 argocd.re-verse.kr을 모두 인증서를 발급할 것이기 때문에
위의 퍼블릭 인증서 요청에서 `이 인증서에 다른 이름 추가` 라는 항목에 *.re-verse.kr을 추가해주면 된다.

인증서를 도메인과 ACM을 사용해서 연결한다.

ALB 생성 및 서비스 , 디플로이먼트 연결

Workshop Studio

Discover and participate in AWS workshops and GameDays

https://catalog.us-east-1.prod.workshops.aws/workshops/9c0aa9ab-90a9-44a6-abe1-8dff360ae428/ko-KR/70-deploy-service/300-frontend

Amazon EKS의 애플리케이션 로드 밸런싱

Kubernetes ingress 를 생성할 때 애플리케이션 트래픽을 로드 밸런싱하는 AWS Application Load Balancer(ALB)가 프로비저닝됩니다. 자세한 내용은 Application Load Balancer 사용 설명서 의 Application Load Balancer란 무엇인가요? 및 Kubernetes 설명서의 수신 을 참조하세요. ALB는 노드 또는 AWS Fargate에 배포되는 pods 와 함께 사용됩니다. ALB를 퍼블릭 또는 프라이빗 서브넷에 배포할 수 있습니다.

https://docs.aws.amazon.com/ko_kr/eks/latest/userguide/alb-ingress.html

Annotations - AWS Load Balancer Controller

You can add annotations to kubernetes Ingress and Service objects to customize their behavior. Annotation keys and values can only be strings. Advanced format should be encoded as below: boolean: 'true' integer: '42' stringList: s1,s2,s3 stringMap: k1=v1,k2=v2 json: 'jsonContent' Annotations applied to Service have higher priority over annotations applied to Ingress.

https://kubernetes-sigs.github.io/aws-load-balancer-controller/v2.2/guide/ingress/annotations/#group.name

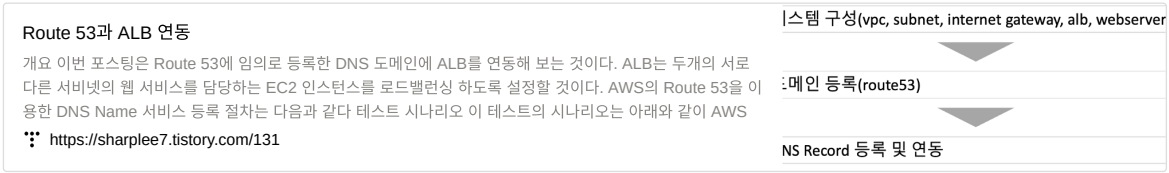
아직 도메인과 SSL 적용하지 않은 상태

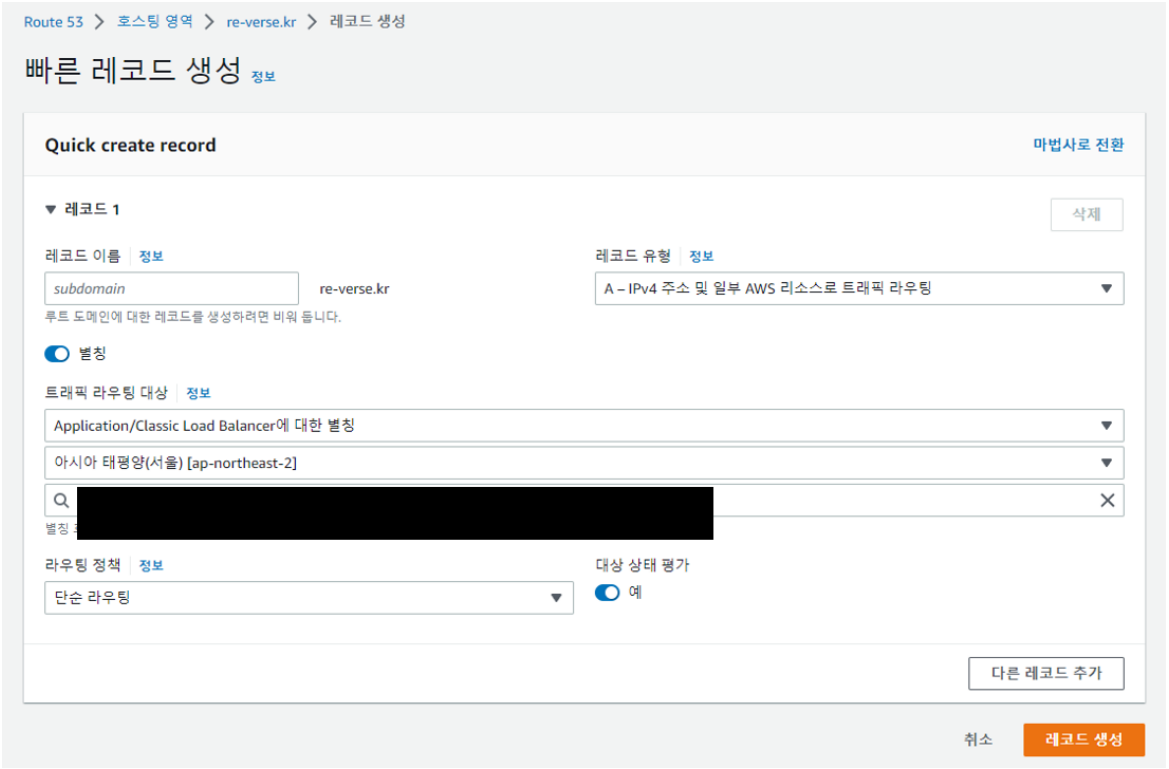
```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: reverse-ingress
  namespace: reverse
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/certificate-arn: "arn:aws:acm:ap-northeast-2:<계정 넘버>:certificate/<인증번>"
    alb.ingress.kubernetes.io/target-type: ip
    # alb.ingress.kubernetes.io/group.name: "reverse-group"
    # alb.ingress.kubernetes.io/group.order: '1'
spec:
  rules:
    - http:
        paths:
          - pathType: Prefix
            path: "/"
            backend:
              service:
                name: reverse-service
                port:
                  number: 80
          - pathType: Prefix
            path: "/api/v1"
            backend:
              service:
                name: reverse-archive-service
                port:
                  number: 80
          - pathType: Prefix
            path: "/api/v1/auth"
            backend:
              service:
                name: reverse-auth-service
                port:
                  number: 80
          - pathType: Prefix
            path: "/socket"
            backend:
              service:
                name: reverse-signal-service
                port:
                  number: 80
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: reverse
  namespace: default
spec:
  selector:
    matchLabels:
      app: reverse
  replicas: 2

  template:
    metadata:
      labels:
        app: reverse

    spec:
      containers:
        - name: reverse
          image: nginx
          imagePullPolicy: Always
          resources:
            limits:
              memory: "128Mi"
              cpu: "500m"
          ports:
```

```
            - containerPort: 80

---
apiVersion: v1
kind: Service
metadata:
  name: reverse-service
spec:
  type: NodePort
  selector:
    app: reverse
  ports:
    - port: 80
      protocol: TCP
      # nodePort: 31000
      targetPort: 80
```

ALB와 도메인 연결





가비아에서 산 도메인의 네임서버를 aws의 것으로 바꾸고, 가비아에서 레코드를 acm 것이랑 연동하면 aws ACM의 인증서를 발급받는다. 이제 위와 같이 route 53을 사용하여 내가 만든 로드밸런서랑 연결을 해주면 이제 알아서 연동이 된다.