05

AWS에서 테라폼을 이용해 EKS 클러스터에 Argo CD를 부트스트랩하기 위한 보충자료

5장에서 테라폼을 사용할 시 독자들이 이해하고 실습하기에는 설명이 다소 부족하다고 판단돼 추가적인 안내를 드리고자 보충자료를 공유드립니다. 이 책의 목적이 AWS나 테라폼이 아니기 때문에 실습을 최대한 가능하게 하는 것을 목표로 보충자료를 작성한 점 참고 부탁드립니다.

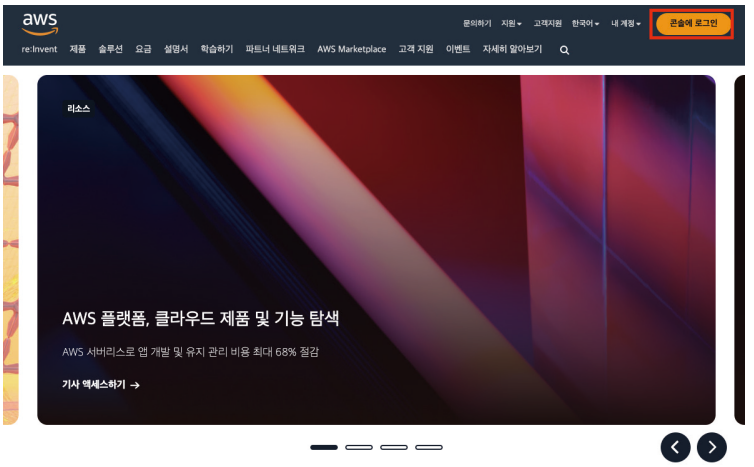
5장에서는 테라폼을 통해 EKS[Elastic Kubernetes Service][1]를 만들고, 해당 클러스터 내에서 Argo CD를 설치해 부트스트랩을 하는 실습을 진행합니다.

따라서 이 실습을 진행하기에 앞서 AWS 계정을 먼저 생성해야 합니다.

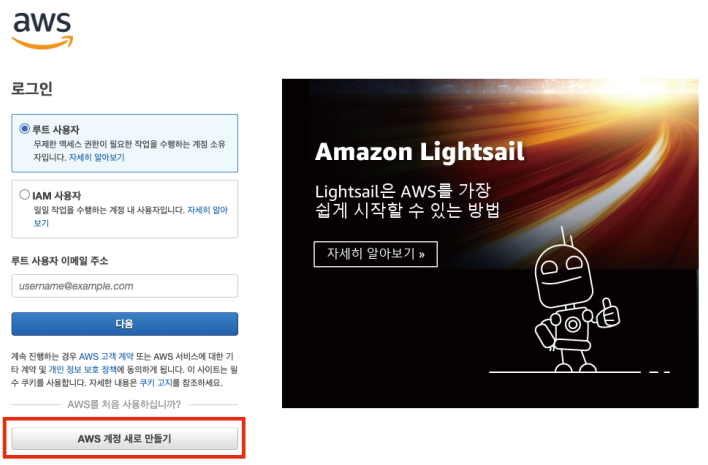
AWS 계정 생성하기

다음 링크(https://aws.amazon.com)로 접속합니다. **콘솔에 로그인** 버튼을 클릭합니다.

1 AWS에서 제공하는 쿠버네티스 클러스터



AWS 계정 새로 만들기를 통해 새로운 계정을 만듭니다.



AWS에서 필요로 하는 정보를 입력하고, 결제할 카드까지 등록하면 계정을 생성할 수 있습니다.

실습 시에는 비용이 청구되니 반드시 실습 후 리소스를 꼼꼼하게 지워야 함을 강조드립니다.

IAM 계정 생성하기

테라폼을 사용하기에 앞서 IAM 사용자를 만들어야 합니다. 해당 사용자로 테라폼을 사용할 것이기 때문에 반드시 필요한 절차입니다.

IAM을 생성하고 액세스 키를 발급받는 방법은 책에 언급돼 있으므로 책을 참조하길 바랍니다.

책에서는 IAM 사용자에게 관리자 권한을 부여하는데, 원칙적으로는 올바르지 않지만 실습의 편의를 위해서 그렇게 했습니다. 이 사용자는 실습이 끝나면 반드시 삭제하길 바랍니다. 액세스 키가 노출될 경우 이 사용자를 통해서 큰 피해를 입을 수도 있습니다.

AWS CLI 설치 및 configure 등록

AWS CLI를 사용하기 위해 먼저 액세스 키를 등록해야 합니다. 해당 액세스 키에 등록된 정보를 통해서 테라폼이 동작합니다.

제공하는 코드는 리눅스(x86 기반)에서 사용할 수 있는 설치 방법입니다.

```
$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
$ unzip awscliv2.zip
$ sudo ./aws/install
```

노트

> 다음 링크(https://docs.aws.amazon.com/ko_kr/cli/latest/userguide/getting-started-install.html)에서 본인의 운영체제에 맞는 설치 방법을 확인하길 바랍니다.

설치가 완료되면 다음 명령어를 통해서 AWS 정보를 등록합니다. 이는 책에 설명돼 있습니다.

```
aws configure
```

테라폼 설치하기

다음 명령어로 테라폼을 설치합니다. 이는 리눅스(우분투) 기반의 명령어입니다.

```
$ wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/
share/keyrings/hashicorp-archive-keyring.gpg
$ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg]
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/
sources.list.d/hashicorp.list
$ sudo apt update && sudo apt install terraform
```

노트

테라폼은 다음 링크(https://developer.hashicorp.com/terraform/install)를 통해서 본인의 운영체제에 맞는 설치 방법을 확인할 수 있습니다.

EKS 프로비저닝

테라폼 설치가 완료되면 테라폼의 워크스페이스를 만들기 위해 다음과 같이 진행합니다. 깃허브에서 제공하는 코드는 최종본이므로 필요한 만큼 참고해서 사용하길 바랍니다. 현재 실습 단계에서는 ch05/terraform 폴더의 파일을 전부 사용할 경우 문제가 생길 수 있기 때문에, 다음 6개 파일만 사용해서 인프라를 생성하길 권장합니다.

또한 저자의 깃허브는 과거 코드를 반영하고 있어 현재에는 변경된 파라미터가 많습니다. 따라서 아래 파일은 에이콘출판사 깃허브에서 제공하는 코드를 우선 사용하길 바랍니다.

eks.tf / iam.tf / network.tf / provider.tf / variables.tf / versions.tf

variables.tf에서 domain과 zone_id는 도메인을 구매한 사람에게만 적용되는 것이므로, 도메인을 사용하지 않는다면 31-39번 라인을 지우고, 사용한다면 해당 부분을 본인의 도메인에 맞게 수정하길 바랍니다.

EKS 클러스터 접근

기본적으로 쿠버네티스 클러스터에 접근하기 위해서는 kubectl 명령어를 사용하며, kubeconfig 정보를 등록해 대상 클러스터와의 인증을 실시합니다.

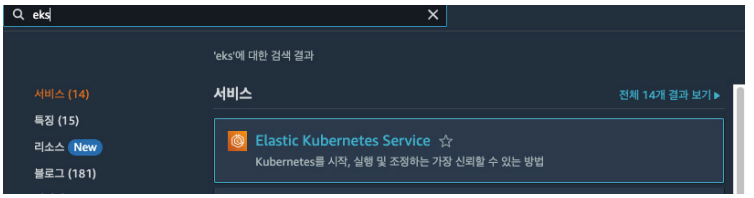
```
aws eks update-kubeconfig --region <region ID> --name <clustername>
```

해당 명령어를 입력하면 ~/.kube/config에 클러스터의 인증 정보를 저장하게 됩니다. 다만 EKS에서는 우리가 사용하는 AWS IAM 사용자를 기반으로 접근하기 때문에 해

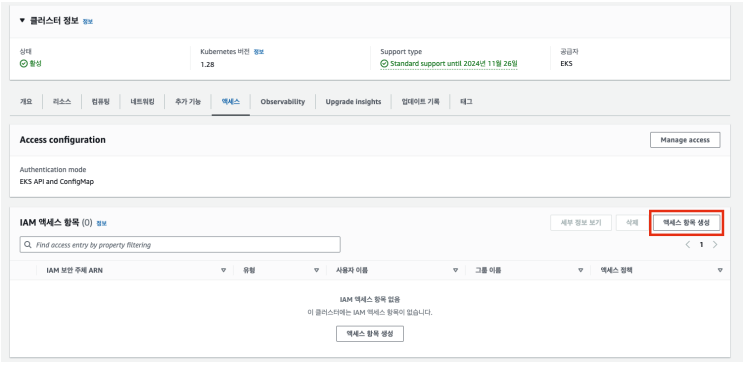
당 IAM 사용자를 EKS 클러스터에 접근할 수 있도록 별도로 허용하는 추가 절차가 필요합니다. 만약 해당 설정을 하지 않는다면 클러스터에 접근할 수 없을 것입니다.

EKS_API를 활용하거나 aws-auth라는 컨피그맵을 활용하는 두 가지 방법이 있는데, 조금 더 간편하고 빠른 EKS_API를 활용해 보겠습니다.

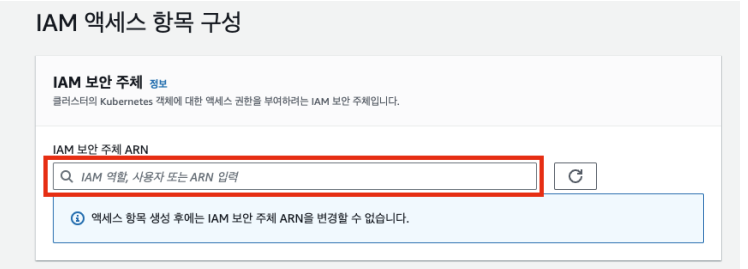
먼저 AWS 콘솔에 접근해서 EKS 서비스로 들어갑니다.



액세스 탭으로 접근한 뒤 **IAM 액세스 항목**에서 **액세스 항목 생성** 버튼을 클릭합니다.

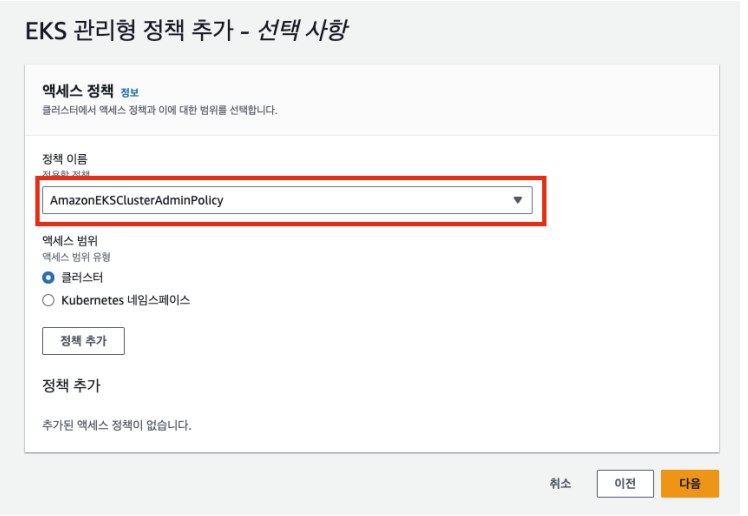


IAM 보안 주체 ARN에 IAM 사용자 정보를 입력해 등록하고 맨 아래 **다음** 버튼을 클릭합니다.

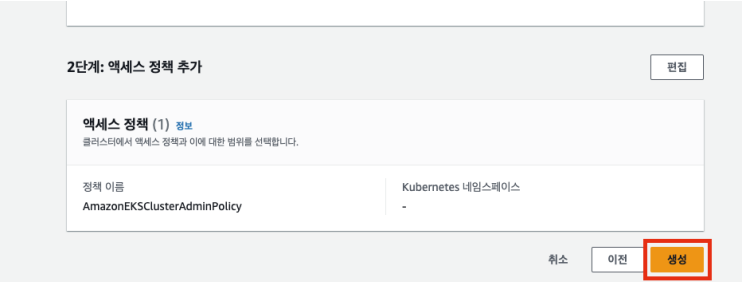


EKS 관리형 정책으로 넘어가서 액세스 정책 이름으로 'AmazonEKSCluster AdminPolicy'를 선택하고, 액세스 범위를 클러스터로 둔 뒤 **정책 추가** 버튼을 클릭합니다.

정책이 제대로 추가됐다면 **다음** 버튼을 클릭합니다.



최종 확인 화면에서 **생성** 버튼을 클릭합니다.



이제 kubectl 명령어로 클러스터에 제대로 접근할 수 있는지 확인해 보겠습니다. 해당 명령어를 입력했을 때 인증 관련 오류가 발생하지 않고 정상적으로 리소스 조회가 된다면 잘 설정된 것입니다.

```
$ kubectl get pod

No resources found in default namespace.
```

테라폼으로 돌아와서 argocd.tf 파일을 추가하고, 테라폼을 다시 작동합니다.

이때 kustomize에서 argocd 네임스페이스를 생성하면서 argocd 네임스페이스 안에 리소스를 만들게 돼 있었는데, 동시에 실행할 경우 실행 순서가 정상적으로 진행되지 않는 경우가 있습니다. 실습 편의상 kubectl 명령어를 통해서 네임스페이스를 먼저 생성하고 테라폼을 작동시키는 걸 추천합니다.

만약 위에서 variables.tf의 도메인을 사용하지 않았다면 argocd.tf에서도 삭제해야 할 게 있습니다. 아래 라인의 externalDNS 부분은 제거하고 실행하길 바랍니다.

15-20번 라인

```
        helm:
          values: |
            externalDNS:
              iamRole: ${aws_iam_role.external_dns.arn}
              domain: ${var.domain}
              txtOwnerID: ${var.zone_id}
```

38-41번 라인

```
  externalDNS:
     iamRole: ${aws_iam_role.external_dns.arn}
     domain: ${var.domain}
     txtOwnerID: ${var.zone_id}
```

```
$ kubectl create namespace argocd
$ terraform apply
```

GUI로 접속하기

Argo CD Server의 GUI 화면으로 접근하는 방법은 다양하지만, 가장 쉬운 방법은 다음 명령어를 입력하는 것입니다. 다음 명령어를 통해서 AWS 로드 밸런서를 이용해 외부로 노출시킬 수 있습니다.

```
kubectl patch service argocd-server -p '{"spec": {"type": "LoadBalancer"}}' -n argocd
```

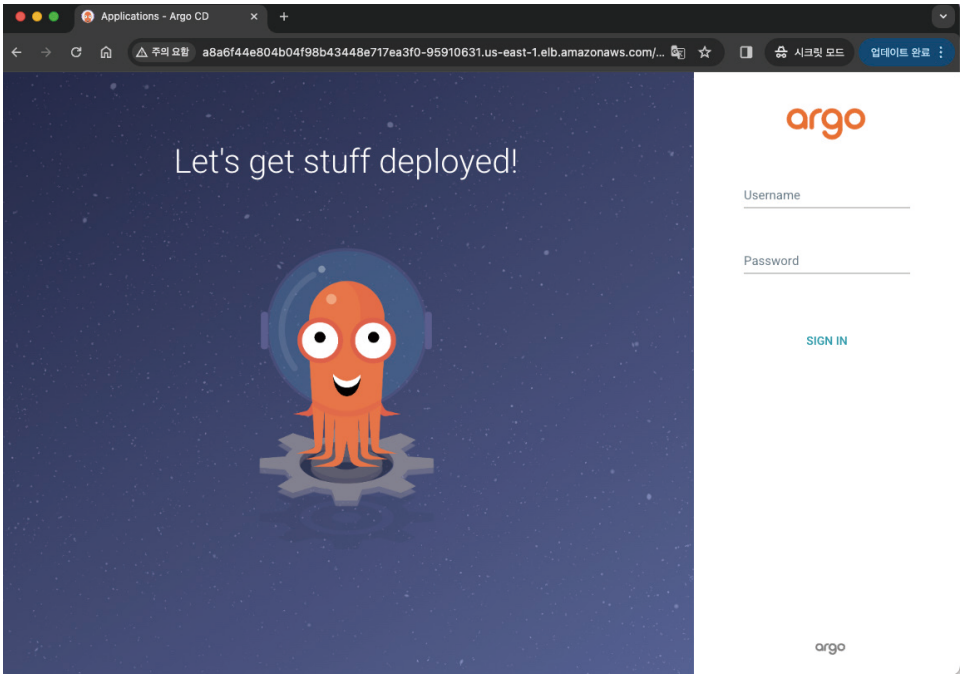
일반적으로는 CLB[Classic Load Balancer]로 만들어지며, 해당 로드 밸런서 주소로 들어가서 확인할 수 있습니다. ALB[Application Load Balancer]를 사용하고 싶다면 external DNS를 별도로 사용해야 합니다. 만약 개인이 도메인을 가지고 있다면 Route 53을 통해서 도메인을 등록해 사용할 수 있으나, 해당 부분을 설명하면 AWS 사용 방법에 대한 내용이 너무 길어지므로 생략합니다.

잠시 후 서비스를 조회해보면 로드 밸런서의 주소를 확인할 수 있습니다.

```
kubectl get service -n argocd
```

```
NAME                    TYPE           CLUSTER-IP       EXTERNAL-IP                                                              PORT(S)                      AGE
argocd-dex-server       ClusterIP      172.20.216.77    <none>                                                                   5556/TCP,5557/TCP,5558/TCP   101s
argocd-metrics          ClusterIP      172.20.115.37    <none>                                                                   8082/TCP                     95s
argocd-redis            ClusterIP      172.20.240.188   <none>                                                                   6379/TCP                     86s
argocd-repo-server      ClusterIP      172.20.36.85     <none>                                                                   8081/TCP,8084/TCP            103s
argocd-server           LoadBalancer   172.20.37.44     a8a6f44e804b04f98b43448e717ea3f0-95910631.us-east-1.elb.amazonaws.com   80:30430/TCP,443:31064/TCP   98s
argocd-server-metrics   ClusterIP      172.20.143.72    <none>                                                                   8083/TCP                     98s
```

해당 주소로 접속하면 화면이 나타납니다.



만약 로드 밸런서를 확인하고 싶다면 AWS 콘솔에서 EC2[Elastic Compute Cloud]에 접속한 후 **로드 밸런서** 탭에서 확인할 수 있습니다.