Custom Resource & Workflow

Hong, YunJay

Kubernetes Study

30 Aug, 2021

Kubernetes Resources

- k8s에는 아주 많은 종류의 리소스가 정의 되어 있고, 이를 이용해 컨테이너의 실행과 로드밸런서의 작성을 비동기로 실행함.
- Pod, ReplicaSet, Deployment, ClusterIP, NodePort, Secret 등의 리소스들을 알맞은 방법으로 실행해서 원하는 클러스터의 동작을 이루어냄.
- 거기에 더해 잘 선언된 k8s의 API 구조를 이용해서 사용자가 직접 리소스를 정의하는 CustomResource를 사용할 수 있게 되어있다.

CustomResource

- Pod, NodePort 등의 리소스처럼 사용자가 직접 리소스를 정의하고, kubectl과 같은 기본 명령어와 함께 사용하게 만들어주는 리소스.
- Custom Reource Definition이라는 Resource로 Custom Resource를 정의한다.
- Cluster Level과 Namespace Level에서 정의가 가능
- CustomController를 이용해 desired state로의 처리가 되도록 실행된다.

CustomResource

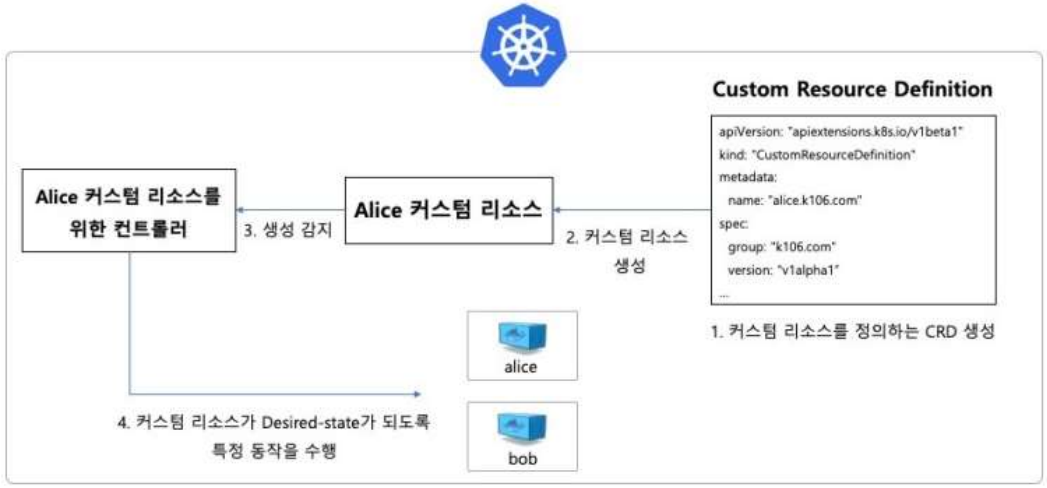
```yaml
# mypod-crd.yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: mypods.crd.example.com
spec:
  group: crd.example.com
  version: v1
  scope: Namespaced
  names:
    plural: mypods
    singular: mypod
    kind: MyPod
    shortNames:
    - mp
```

```yaml
# mypod-resource.yml
apiVersion: "crd.example.com/v1"
kind: MyPod
metadata:
  name: mypod-test
spec:
  uri: "any uri"
  customCommand: "custom command"
  image: nginx
```

```
master@master-jay:~/kubernetes-start-up-study/16_k8s_custom_resource$ kubectl apply -f mypod-crd.yaml
customresourcedefinition.apiextensions.k8s.io/mypods.crd.example.com created
master@master-jay:~/kubernetes-start-up-study/16_k8s_custom_resource$ kubectl get crd | grep mypods
mypods.crd.example.com                2021-08-29T13:57:18Z
master@master-jay:~/kubernetes-start-up-study/16_k8s_custom_resource$ kubectl apply -f mypod-resouce.yml
mypod.crd.example.com/mypod-test created
master@master-jay:~/kubernetes-start-up-study/16_k8s_custom_resource$ kubectl get mypod-
error: the server doesn't have a resource type "mypod-"
master@master-jay:~/kubernetes-start-up-study/16_k8s_custom_resource$ kubectl get mypod
NAME         AGE
mypod-test   11s
```

Custom Controller

- CustomResource는 구조화된 데이터 덩어리일 뿐이고, 이 데이터가 어떻게 행동해야 하는지에 대한 동작은 Custom Controller가 있어야 실행된다.
- CustomReource가 최종적으로 가져야하는 desired state를 정의하고, 이를 유지하기 위한 동작을 선언해 가지는 것이 Custom Controller의 목적이다.



Operator Pattern

- Custom Resource가 생성되고 삭제될 때의 패턴을 k8s native resource 처럼 자동화 할 때 사용된다.
- 쿠버네티스 컨트롤러를 포함하는 개념이고, 여러 오퍼레이터가 이미 구현되어 있으며 이를 활용하여 crd의 동작을 정의할 수 있다.

Argo Workflow

- Argo Project에서 만든 컨테이너 기반 워크플로우 엔진.
- 사용자가 원하는대로 job 간의 종속성을 만들고 순서대로, 혹은 병렬로 여러 job을 실행시킬 수 있다.
- 고급 워크플로우 관리 기능 / 웹 UI를 제공한다.

Argo Workflow - Pros N Cons

- 컨테이너 레벨에서 실행되기 때문에 각 실행 단위별로 고립된 환경을 이용할 수 있다.
- 각 Job에서는 독립된 일을 진행하기 때문에 재사용성이 높아진다. Job에서 생성되는/필요로하는 데이터 인터페이스를 잘 갖추어 놓을 경우 활용도가 높아진다.
- 하지만 Pod의 생성과 삭제에 드는 Overhead가 커 성능 저하가 발생할 가능성이 있다.
- Job마다 다른 컨테이너에서 실행되기 때문에 Volume Mount를 통해 데이터를 공유해야한다.