쿠버네티스 학습

kubctl이란?

- 쿠버네티스에 내가 원하는 것을 요청할 때 쓰이는 명령어

kubectl 명령어 구조

- kubectl [command] [TYPE] [NAME] [flags]

command: 자원(object)에 실행할 명령(create,get,delete,edit 등)

TYPE: 자원의 타입(node, pod, service)

NAME: 자원의 이름(원하는대로 붙일 수 있음 소문자, 숫자, 대시문자 사용가능)

flags: 부가적으로 설정할 옵션 ( --help, -o options 등)

kubectl get pod webserver -o wide

kubectl commands

- kubtetl –-help

- kubectl command –help

- kubectl run <자원이름> <옵션>

- kubectl create -f obj.yaml

- kubectl apply -f obj,yaml

- kubectl get <자원이름> <객체이름>

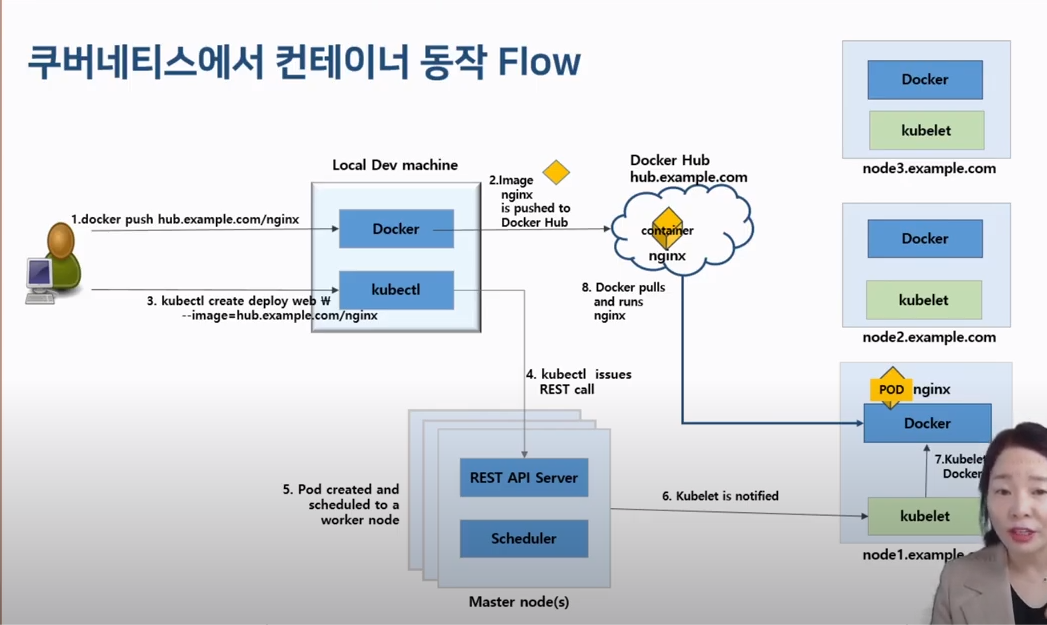
- kubectl edit <자원이름> <객체이름>

- kubectl describe <자원이름> <객체이름>

- kubectl delete pod main

쿠버네티스 아키텍처

1.Kubernetes 동작원리



컨테이너를 쿠버네티스 플랫폼에 올려서 사용하려는게 이 사용자의 목적이다. 먼저 컨테이너를 빌드 하겠죠. 예를들어 mainUI를 만들어 주는 컨테이너, login작업을 수행하는 컨테이너, 주문에 관련된 작업을 해주는 컨테이너 이런 컨테이너들을 쭉 만들어서 doker명령어로 저장로라고 불려지는 허브 어디에? 사내에 만들어져 있을 수 도 있고 또는 도커닷컴에서 만들어서 운영하는 퍼블릭 공간의 허브가 될 수 도 있다. 이렇게 허브가 있으면 만들어진 컨테이너를 도커 푸시 명령어를 이용해서 허브에 여러 컨테이너들을 저장해 놓는다. 이제 사용자는 쿠버네티스 명령어를 이용해서 쿠버네티스에게 이 컨테이너가 실행될 수 있도록 요청한다. 어떻게?? Yaml 파일이나 cli 명령어를 이용해서 가능 여기서는 cli명령어로 hube.example.com에 있는 nginx를 deploy로 실행해 주세요라고 kubectl명령어를 만들어서 마스터에게 보낸다. 이때 마스터라고 불리긴 하지만 control-plane이라고도 불린다. 마스터 시스템이 한대,두대, 여러대던지 어쨌든 마스터에게 보내면 api 서버가 존재해서 쿠버네티스 관련된 명령어 요청을 받아준다. 그런다음 이 api는 이 요청에 따라서 nginx라는 컨테이너를 node1,2,3 등 여러 대 중에 어디에다 배치하면 좋을지 scheduler에게 요청한다. 스케줄러는 현재 노드의 상태를 보고 어느 node가 가장 좋을지 선택한 다음 응답해 준다. 예를 들어 node1이 가장 좋을 때 이 api는 node` 시스템이 있는 kubelet에 요청한다. Node1의 kubelet아 너가 nginx하나를 실행해 줬으면 좋겠어 라고, 그러면 node1의 kubelet은 그 요청을 받아서 doker명령어로 바꿔서 도커 데몬에게 실제 컨테이너 실행 요청을 해주게 된다. 그러면 도커데몬은 정의가 되어 있는 hub.example.com에 nginx컨테이너가 있는지 검색한 다음 있으면 그것을 받아와서 컨테이너로 실행해 준다. 쿠버네티스는 이렇게 동작되는 컨테이너를 pod라는 단위로 관리하게 된다. 여기까지가 컨테이너를 빌드해서 저장하고 k8s 환경에서 실행 했을 때 동작되는 과정이다.

2. 쿠버네티스 컴포넌트

|  |  |
| --- | --- |
| 마스터 컴포넌트 | 워커 노드 컴포넌트 |
| etcd  - key-value 타입의 저장소 | kubelet  - 모든 노드에서 실행되는 k8s 에이전트  - 데몬 형태로 동작 |
| kube-apiserver  - k8s API를 사용하도록 요청을 받고 요청이 유효한지 검사 | kube-proxy  - k8s의 network 동작을 관리  - iptables rule을 구성 |
| kube-scheduler  - 파드를 실행할 노드 선택 | 컨테이너 런타임  - 컨테이너를 실행하는 엔진  - docker, containerd, runc |
| kube-contorller-manager  - 파드를 관찰하며 개수를 보장 |  |

3. 애드온 (출처: https://ikcoo.tistory.com/3 [이쿠의 슬기로운 개발생활])

|  |  |
| --- | --- |
| 네트워크 애드온 | CNI – weave,calico, flaneld, kube-route 등  컨테이너 간의 네트워킹을 제어할 수 있는 플러그인을 만들기 위한 표준  Kubernetes Cluster 내부는 Master Node에 의해 여러 컨테이너가 생성 삭제 복구를 반복하고 있음  그에 따라 각 컨테이너의 고정적이지 않고 재할당이 빈번함  이러한 특징을 해결하기 위해 Kubernetes Cluster는 가상 네트워크가 구성되어 있는데  기본적으로는 Worker Node의 kube-proxy 가 네트워크를 관리하지만  보다 효율적인 네트워크 환경을 구성하기 위해  다양한 네트워크 관련 Addon 이 제공됨 |
| DNS 애드온 | coreDNS  다른 Addon 은 필수적이지 않지만 DNS Addon 만큼은 필수적임.  실제로 Kubernetes Cluster 내에서 작동하는 DNS Server.  Kubernetes Service Object에게 DNS 레코드를 제공하는 역할을 수행함.  kubernetes 내부에서 실행된 컨테이너들은 자동으로 DNS 서버에 등록되어 자동 탐색이 가능해짐 |
| 대시보드 애드온 | kubernetes Cluster를 위한 Web 기반 UI.  일반적으로 kubernetse Cluster에 명령을 내릴 때 CLI에서 kubectl을 사용하는데  가시화하여 관리의 편의성을 제공하기 위해 대시보드를 제공함 |
| 컨테이너 애드온 | cAdvisor  어떤 시스템이던 가장 중요한 부분 중 하나는 바로 모니터링.  Kubernetes 환경에서도 모니터링을 제공하는 Addon이 존재함.  Cluster 내부에서 실행 중인 Container의 CPU, 메모리와 같은 리소스 데이터를  저장하고 볼 수 있는 방법을 제공하는 Addon. |
| 클러스터 로깅 | 컨테이너 로그, k8s 운영 로그들을 수집해서 중앙화  Kubernetes Cluster 내부에서 생성되는 모든 Log를 중앙화 하여 관리할 수 있음.  Master Node, Worker Node, Worker Node 내부의 Container 등 모든 로그를 중앙화 된  로그 수집 시스템에 모아서 볼 수 있도록 해주는 Addon.  클라우드 서비스를 이용 중이라면 각 클라우드 벤더에서 제공해주는 로깅 서비스를 사용하면 되지만  직접 kubernetes를 설치한 경우에는 Log 부분도 고려해야함  해당 addon 은 직접 kubernetes 를 설치한 경우에 사용하는 Addon.  ELK(ElasticSearch, Logstash, Kibana), EFK(ElasticSearch, Fluentd, Kibana),DataDog |

쿠버네티스 namespace

|  |
| --- |
| - Namespace는 쿠버네티스 API 종류 중 하나이다.  - 클러스터 하나를 여러 개의 논리적인 단위로 나눠서 사용  - 쿠버네티스 클러스터 하나를 여러 팀이나 사용자가 함께 공유  - 용도에 따라 실행해야 하는 앱을 구분할 때 사용 |
| namespace를 사용하면 좋은점  - 수많은 파드들이 존재하는데 필요한 것들을 namespace로 묶을 수 있어서 관리하기가 용이  - 여러 개의 버전을 namespace로 분류 가능 |
| namespace 생성  - CLI  $kubectl create namespace blue  $kubectl get namesapces  - Yaml  $kubectl create namespace green –dry-run -o yaml > green-ns.yaml  $vim green-ns.yaml  $kubectl create -f green-ns.yaml |
| namespace 관리  $kubectl get namespace  $kubectl delete namespace |
| namespace 생성 실습 내용  $kubectl get pod -n [namespace이름]  $kubectl create -f [오브젝트명]: 파일을 불러들여와 오브젝트를 생성한다.  $kubectl get pods -n kube-system: kube-system에서 동작중인 pod리스트를 볼 때 사용  $kubectl create namespace orange –dry-run -o yaml > orange-ns.yaml  $kubectl create -f orange-ns.yaml  --dry-run : 실제로 실행하지 않고 실행 가능 여부만 확인  -o yaml : 실행 결과를 yaml 파일로 출력  $kubectl create -f nginx.yaml -n blue : blue namepsace에 yaml파일 설정내용 |
| 사용할 네임스페이스 switch  기본으로 사용하는 namespace를 default가 아닌 다른 이름의 namespace로 switch  방법  1) namespace를 포함한 context 등록  $kubectl config --help  $kubectl config set-context NAME --cluster=Kubernetes …  $kubectl config view  2) 등록된 namespace로 context 변경  $kubectl config use-context NAME |
| 사용할 네임스페이스 switch 실습 내용  $kubectl config --help  $kubectl config view  $kubectl config set-context blue@kubernetes --cluster=kubernetes --user=kubernetes-admin –namespace=blue    $kubectl config current-context  $kubectl config use-context blue@kubernetes    $kubectl delete pods mypod -n default  $kubectl get pods -n default  $kubectl get pods -n blue  kubectl delete [오브젝트] : 오브젝트를 삭제  $kubectl delete namespaces blue |

yaml 템플릿

|  |
| --- |
| 사람이 쉽게 읽을 수 있는 데이터 직렬화 양식  기본문법  - 구조화된 데이터를 표현하기 위한 데이터 포맷  - python처럼 들여쓰기로 데이터 계층을 표기  - 들여쓰기를 할 때에는 Tab이 아닌 Space Bar를 사용  - 가독성이 좋아 설정 파일에 적합한 형식  - Scalar 문법 : ‘:’을 기준으로 key: value 를 설정  - 배열 문법: ‘-‘문자로 여러 개를 나열  - 공식 사이트: http:/yaml.org/  - Kubernetes yam example  $ cat nginx-pod.yaml |

API version

|  |  |
| --- | --- |
| alpha -> beta -> stable  kubernetes Object 정의 시 apiVersion이 필요  kubernetes가 update하는 API가 있으면 새로운 API가 생성됨 | |
| API Object 종류 및 버전 | |
| Deployment  Pod  ReplicaSet  ReplicationController  Service  PersistentVolume | apps/v1  v1  apps/v1  v1  v1  v1 |
| $kubectl explain pod 이용해서 version확인 가능 | |

Pod 학습내용

pod 개념 및 사용하기

livenessProbe를 사용한 self-healing Pod

init container

infra container(pause) 이해하기

static pod 만들기

Pod에 resource 할당하기

환경변수를 이용해 컨테이너에 데이터 전달하기

pod구성 패턴의 종류

1. Pod 개념 및 사용하기

|  |
| --- |
| Container 정리      container는 하나의 application 처럼 사용  쿠버네티스에서는 Pod라고하는 단위를 통해서 container를 동작시킴 |
| Pod란?  - 컨테이너를 표현하는 k8s API의 최소 단위    Pod에는 하나 또는 여러 개의 컨테이너가 포함될 수 있음 |
| pod 생성 하기 |
| 실습  $kubectl run web1 –image=nginx:1.14 --port=80    $cat pod-nginx.yaml  $kubectl create -f pod-nginx.yaml  $kubectl get pods  $kubectl get pods -o wide  $kubectl get pods web1  $kubectl get pods web1 -o yaml : 동작중인 web1 pod를 yaml파일로 보여주는 명령어  $ watch kubectl get pods -o wide  \*watch [명령어] : watch뒤의 명령어를 2초마다 한번씩 실행  $ curl 10.40.0.1    $kubectl get pods web1 -o json | grep -I podip : podip 내용 추출 |
| multi-container pod 생성하기 |
| 실습  $kubectl describe [오브젝트타입] [오브젝트명]  - 특정 또는 그룹의 오브젝트를 더욱 자세하게 보여주는 명령어  $kubectl dscribe pod web1  $kubectl create -f pod-multi.yaml    $kubectl describe pod multipod  $kubectl exec multipod -c nginx-container -it -- /bin/bash  $cd /usr/share/nginx/html/  $cat index.html  $echo “TEST web” > index.html  $exit  $curl 10.46.0.1  $kubectl exec multipod -c centos-container -it -- /bin/bash  $ps -ef  $curl localhost:80 🡪 동일한 ip를 가지고 web-server가 nginx를 송출하기 때문에 centos-container에도 웹접속됨  - multi-container pod에서 container들의 pod명과 ip는 동일  $kubectl logs multipod -c nginx-container  $kubectl logs [pod명] -c [container명] : [pod명]안의 [container명]의 로그를 출력  - 멀티 컨테이너는 container명까지 입력해야 하지만 싱글 컨테이너는 pod명까지만 입력하면됨 |
| Pod 동작 flow |
| Pod 관리하기  동작중인 파드 정보 보기  $kubectl get pods  $kubectl get pods -o wide  $kubectl describe pod webserver  동작중인 파드 수정  $kubectl edit pod webserver  동작중인 파드 삭제  $kubectl delete pod webserver  $kubectl delete pod --all |

|  |
| --- |
| Question & Answer  1. 현재 namespace에서 동작중인 Pod는 몇 개인가?  $kubectl get pods  2. 현재 시스템에서 동작중인 Pod 수는?  $kubectl get pods --all-namespaces  3. 컨테이너 nginx를 실행하는 nginx-pod라는 이름의 Pod를 생성하시오.  $kubectl run nginx-pod --image=nginx  4. 앞에서 생성한 Pod의 image정보를 확인하는 명령은 무엇인가?  $kubectl describe pod nginx-pod  5. 앞에서 생성한 nginx-pod는 어느 node에 배치되었나?  6. 앞에서 생성한 Pod에는 몇 개의 컨테이너가 포함되어 있나?  $kubectl get pods 의 Ready상태를 보면 알 수 있다. 또는  $kubectl describe pod [pod명]으로 일일이 확인  7. 앞에서 생성한 Pod의 현재 상태는 어떠한가?  $kubectl get pods 의 Ready상태를 보면 알 수 있다. 또는  $kubectl describe pod [pod명]으로 일일이 확인  8. 새 Pod의 컨테이너 상태는 어떻습니까?  describe 명령어로 확인 가능  9. ‘kubectl get pods’명령의 출력에서 READY 열은 무엇을 의미하나?  현재 준비완료인 것 / 전체 Pod 수  10. 생성한 pod를 삭제하시오.  $kubectl delete pod nginx-pod  11. 컨테이너 image ‘redis123’을 실행하는 pod ‘redis’를 redix.yaml 을 이용해서 생성하시오.  $kubectl run redis --image=redis123 –dry-run o yaml > redis.yaml  redis.yaml 수정  $kubectl create -f redis.yaml  12. 앞서 만든 redis pod의 image를 redis로 수정하여 동작시키시오.  $kubectl edit pod redis -> image를 redis로 수정 후 저장 |