



k8s - 쿠버네티스 아키텍처



k8s는 어떤 사상으로 움직이는지 알아보자

쿠버네티스 특징

1. 선언적

- 상태 확인 → 현재 상태가 설정 상태가 아니네? → 선언된 상태로 변환 (관리비용 감소)
but, 단순한 내부동작 제어 쉽지않음...(kubectl로 제어 기능 추가됨)
- 상태복구 (클러스터가 동작 중이라면 원 상태로 자동복구)
- 단일장애점이 없음

2. 워크로드 분리

운영체제처럼 프로세스의 관리를 추상화 해주는 레이어

3. 어디서나 설치 및 사용 가능, 성숙한 커뮤니티



※쿠버네티스의 원리를 알아보자(간단히)

공식문서에서는?



쿠버네티스 기능

자동화된 돌아옴과 롤백

쿠버네티스는 애플리케이션 또는 애플리케이션의 설정 변경시 점진적으로 돌아오는 동시에 애플리케이션을 모니터링해서 모든 인스턴스가 동시에 종료되지 않도록 보장한다. 만약 어떤 문제가 발생하면 쿠버네티스는 변경 사항을 롤백한다. 성장하는 디플로이먼트 솔루션 생태계를 이용한다.

스토리지 오케스트레이션

로컬 스토리지, AWS나 GCP와 같은 퍼블릭 클라우드 공급자 또는 NFS, iSCSI, Ceph, Cinder와 같은 네트워크 스토리지 시스템에서 원하는 스토리지 시스템을 자동으로 마운트한다.

자동 빈 패킹(bin packing)

리소스 요구 사항과 기타 제약 조건에 따라 컨테이너를 자동으로 배치하지만, 가용성은 그대로 유지한다. 활용도를 높이고 더 많은 리소스를 절약하기 위해 중요한(critical) 워크로드와 최선의(best-effort) 워크로드를 혼합한다.

IPv4/IPv6 이중 스택

파드와 서비스에 IPv4와 IPv6 주소 할당

자가 치유

오류가 발생한 컨테이너를 재시작하고, 노드가 죽었을 때 컨테이너를 교체하기 위해 다시 스케줄하고, 사용자 정의 상태 체크에 응답하지 않는 컨테이너를 제거하며, 서비스를 제공할 준비가 될 때까지 클라이언트에 해당 컨테이너를 알리지 않는다.

서비스 디스커버리와 로드 밸런싱

쿠버네티스를 사용하면 익숙하지 않은 서비스 디스커버리 메커니즘을 사용하기 위해 애플리케이션을 수정할 필요가 없다. 쿠버네티스는 파드에게 고유한 IP 주소와 파드 집합에 대한 단일 DNS 명을 부여하고, 그것들 간에 로드-밸런싱을 수행할 수 있다.

시크릿과 구성 관리

사용자의 이미지를 다시 빌드하거나 스택 구성의 시크릿을 노출하지 않고 시크릿과 애플리케이션 구성을 배포하고 업데이트한다.

배치 실행

쿠버네티스는 서비스 외에도 배치(batch)와 CI 워크로드를 관리할 수 있으며, 원하는 경우 실패한 컨테이너를 교체할 수 있다.

Horizontal 스케일링

간단한 명령어나 UI를 통해서 또는 CPU 사용량에 따라 자동으로 애플리케이션의 스케일을 업 또는 다운한다.

확장성을 고려하여 설계됨

쿠버네티스 업스트림 소스 코드 수정 없이 쿠버네티스 클러스터에 기능을 추가할 수 있다.

쿠버네티스가 아닌것도 알아야됩니다.

MSA환경과 유사한 환경에서 작업하시는 분 들은 작업 환경이 모두 쿠버네티스 어플리케이션 안에서 이루어지는 줄 알고있습니다. (ex. 대표적으로 이충호)

하드웨어보단 컨테이너 수준에서 운영되기 때문에 쿠버네티스 밖에서 운영되는 모놀리틱 솔루션을 선택해 추가하는 형태로도 많이 운영되고있습니다.

대표적인 모놀리틱 솔루션

1. CI/CD 툴
2. Middle Ware
3. DB

4. Cash / Cluster Storage

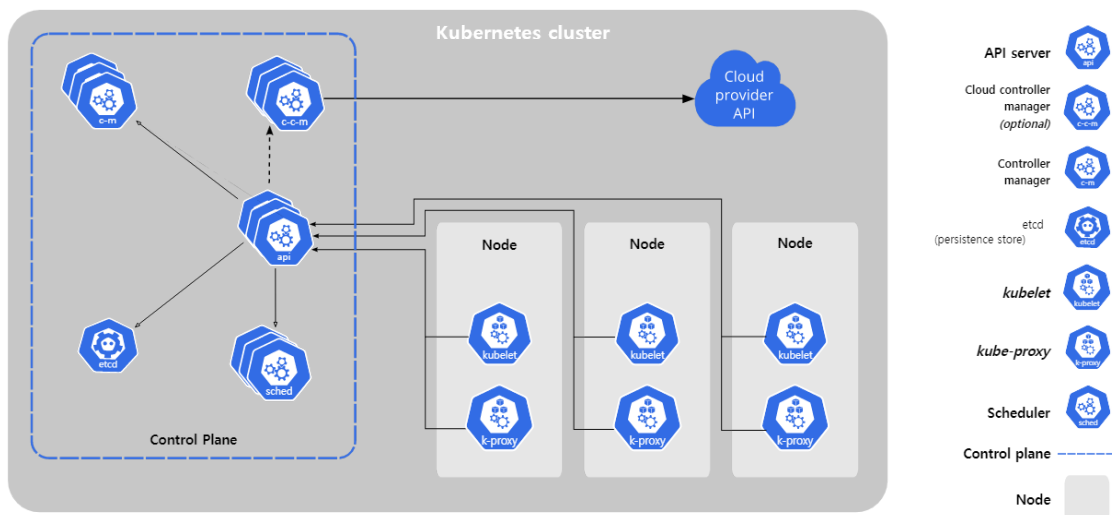
등은 쿠버네티스 환경에서 실행되는 추가적인 솔루션이라고 생각하면 됩니다.

단지, 정의된 워크플로우에 따라 입력된 상태를 유지하게 됩니다.

클러스터의 전체 구조

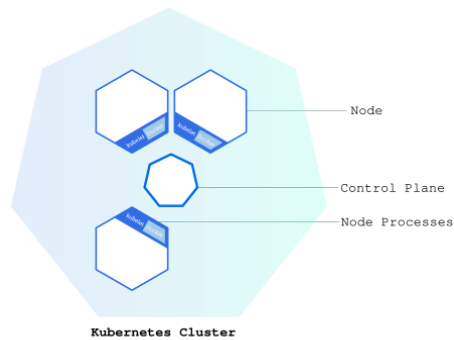
<https://hoing.io/archives/131> 및

<https://kubernetes.io/ko/docs/concepts/overview/components/>를 참조하였습니다.

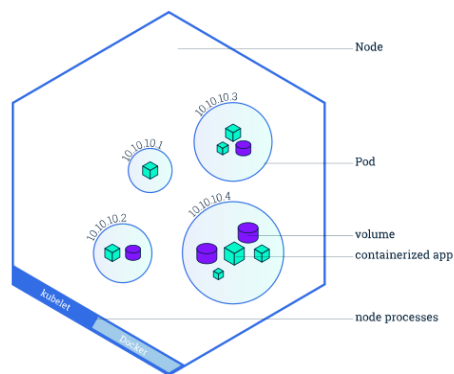


쿠버네티스를 배포하면

- **Cluster**를 얻게됩니다.
- 클러스터는 한개 이상의 **Worker Node**(컨테이너 애플리케이션)를 얻게 됩니다.
- 워커 노드는 구성요소인 **Pod**를 호스트합니다.



Cluster 다이어그램



Node 및 Pod 다이어그램

Cluster

- 그룹단위 입니다. 90가지 예제로 배우는 예제에서는 Master 3, Worker 2로 구성될 예정입니다.

Control Plane ≅ Master

- **Worker Node + 클러스터 내부 Pod**를 관리합니다.
- 컨트롤 플레인이 여러 컴퓨터에 걸쳐 실행되고, 일반적으로 여러 노드를 실행함으로 **내결함성**과 **고가용성**이 제공된다고 합니다.
- 실질적으로는 1개 운영되며, 마스터에 문제가 생기면, 다른 워커 노드와 달리 대신할 수 있는 대체자가 없습니다, 대신, 예비노드에서 관리 및 제어로 대체됩니다.

Worker ≡ Node

- Master의 명령을 수행하는 서버입니다. 실제 실행과 중지, 유지등 역할을 수행합니다.
- k8s에서는 워커머신 이지만, 클러스터에 따라 VM or physical Machine이 될 수 있다고 하네요(설치되는 환경에 따라 다른 듯 합니다.)
- Master에 의해 Pod를 관리합니다.

Pod

- Node에 배포된 1..N(일반적으로 1개)의 컨테이너 그룹입니다. 컨테이너를 묶음단위로 관리 가능하게 해줍니다.