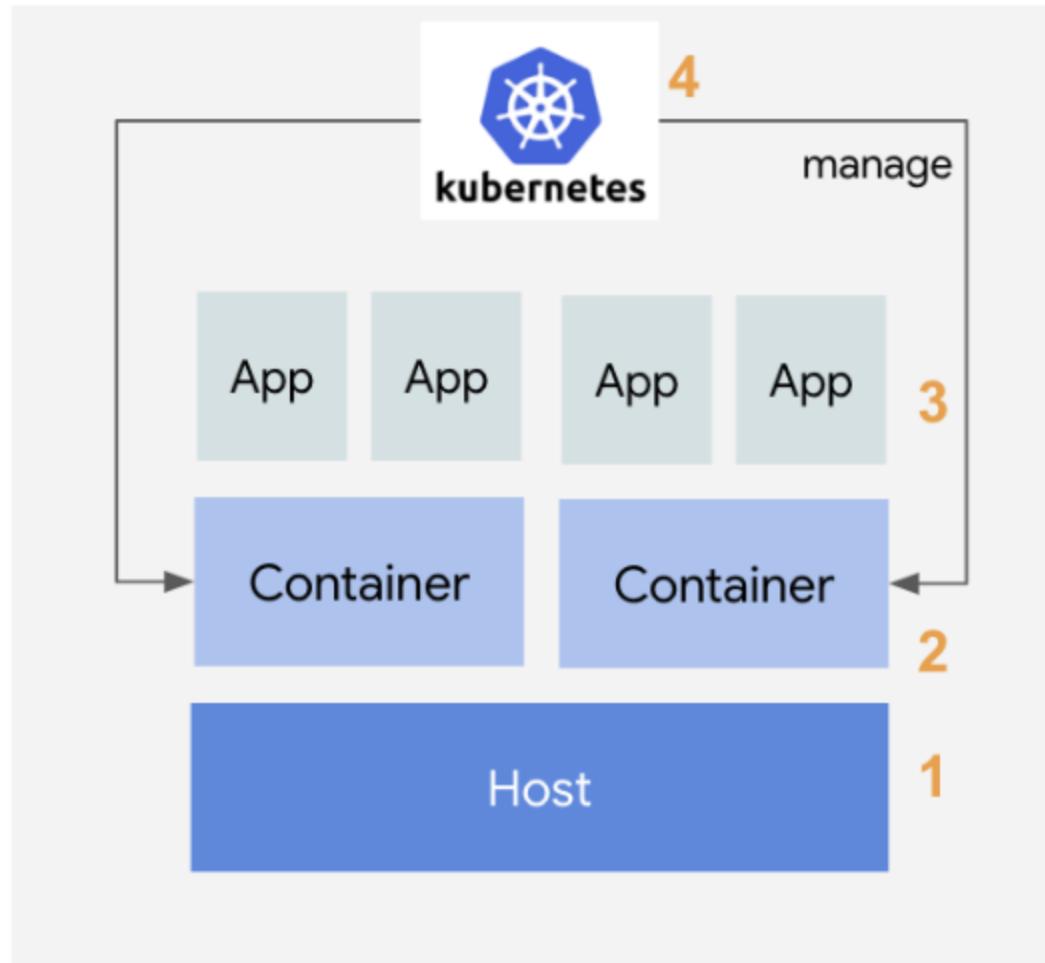
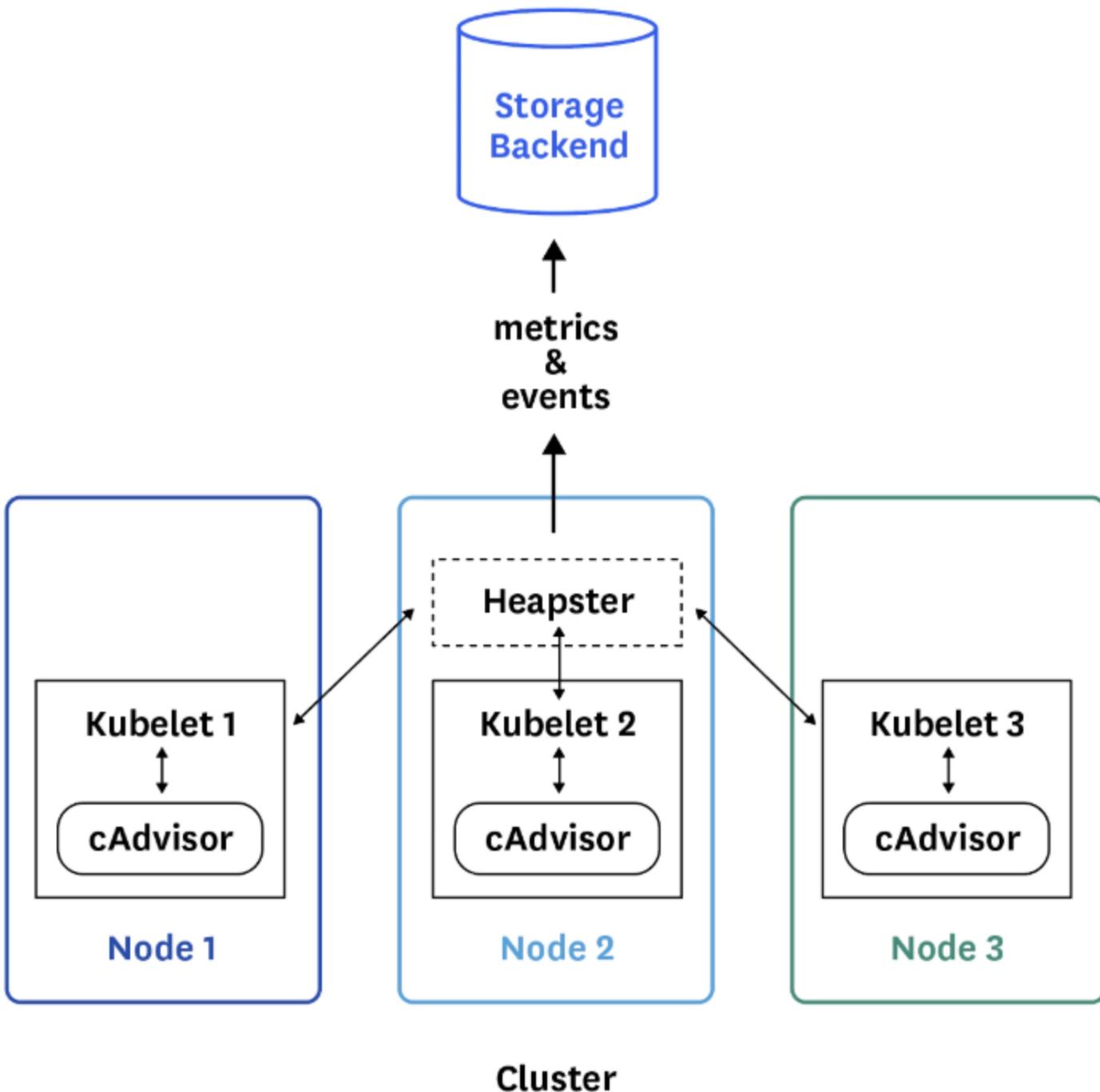


3. 모니터링, 로깅, 문제해결

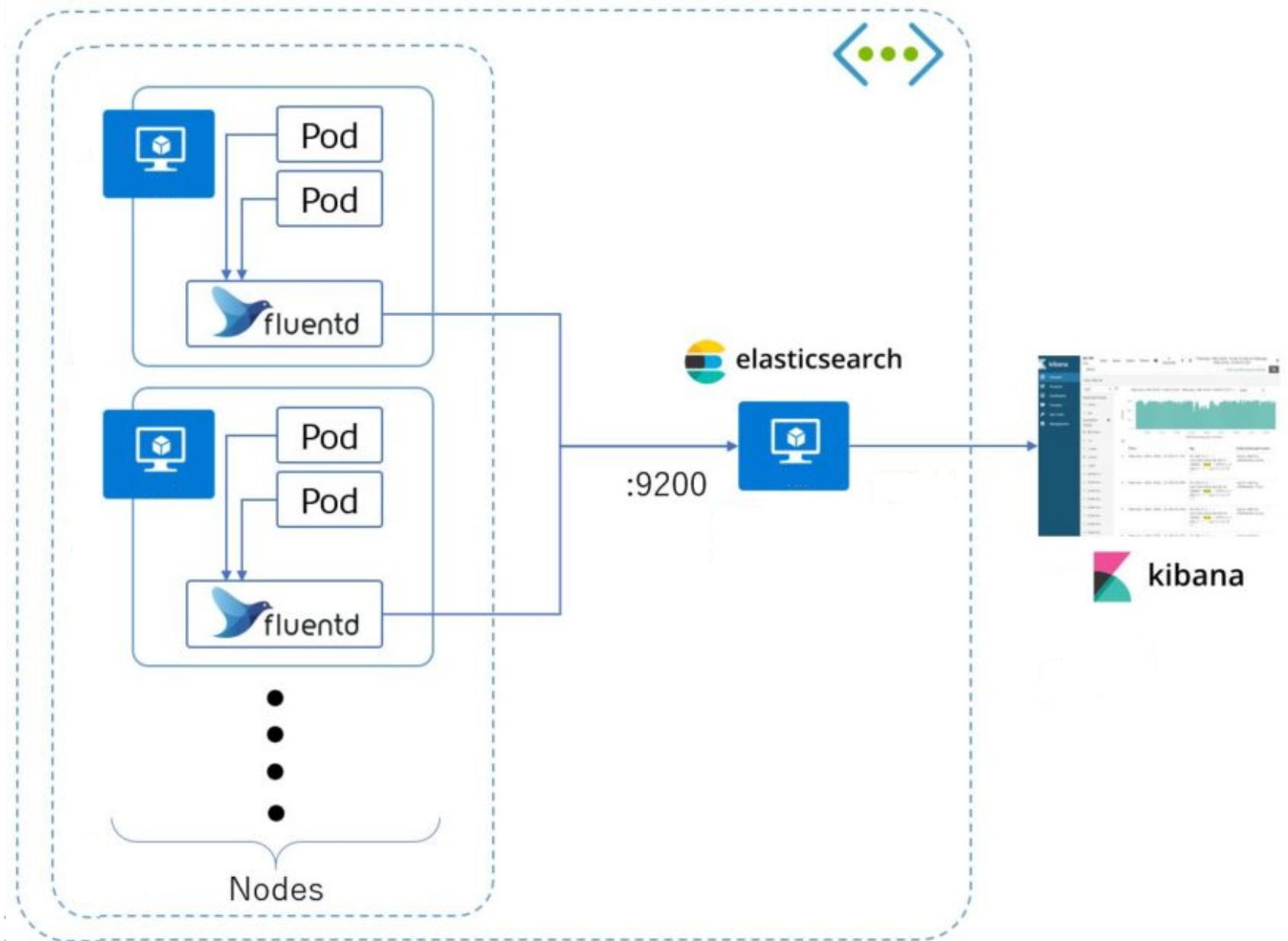
Kubernetes Monitoring



- Node, Container, App, Kubernetes 크게 4가지 계층을 모니터링
- Collecting Metrics, Storing Metrics, Visualize Metrics



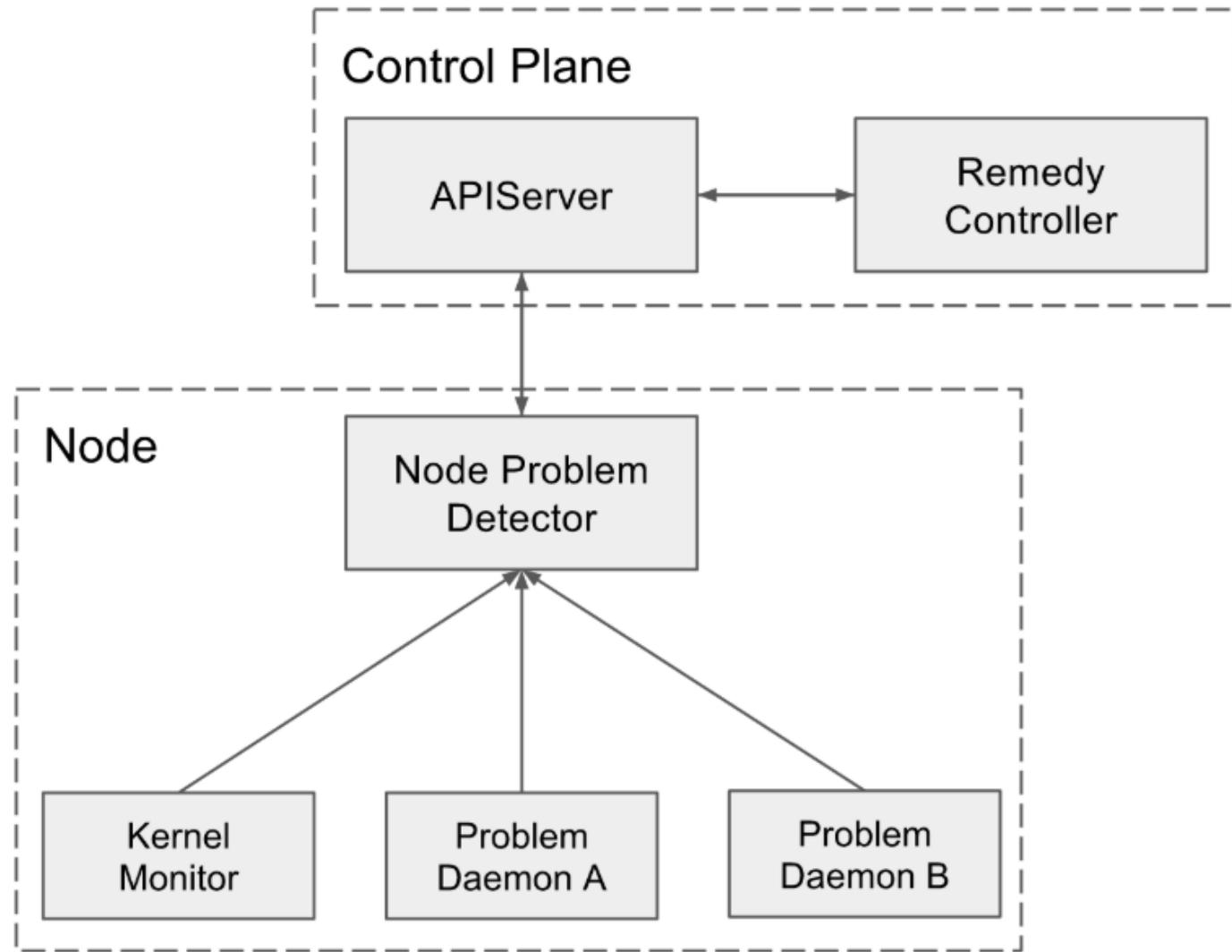
Kubernetes Logging



node-problem-detector

- 각 노드에 CPU, 메모리, 디스크 손상 등 여러 문제가 발생할 수 있음
- <https://github.com/kubernetes/node-problem-detector>
- DaemonSet (클러스터 전체에 포드를 띄울 때 사용하는 컨트롤러)
- 마스터에게 문제를 보고하는 API (NodeCondition, Event)
- 특정 문제를 탐지하는 역할 (Problem Daemons, ex: KernelMonitor)
- 일부 문제를 자동으로 해결할 수 있는 컨트롤러 (Remedy Controller)
- Kubernetes Addon enabled by default in GCE
- `kubectl create -f node-problem-detector.yaml`

System Diagram



하드웨어 장애 (노드의 응답이 없는 경우)

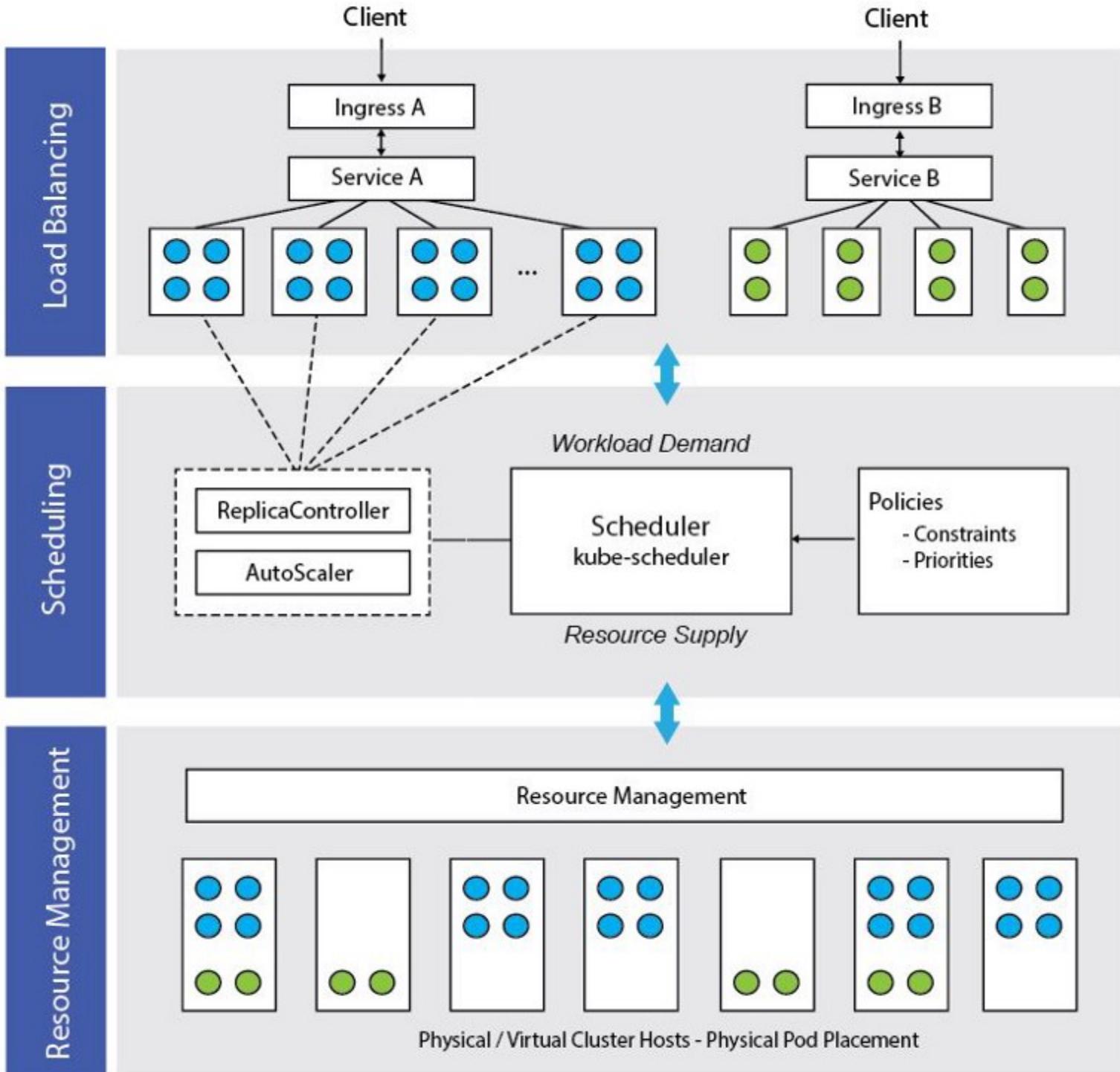
- 로그 정보를 활용하기가 어려움
- 클러스터에 노드가 새로 추가된 경우, 설정 문제일 가능성
- 힙스터의 노드나 중앙 집중 로깅에서 에러가 포함된 로그로 원인을 찾기
- 힙스터는 1.11 버전부터 deprecated, 1.13 버전부터 완전히 제거
- metric-server로 대체

하드웨어 장애 (노드의 응답이 있는 경우)

- 디스크나 하드웨어에서 장애가 발생할 수 있음
- node-problem-detector를 최대한 활용 (장애 감지)
- 클라우드 환경일 경우, 장애가 의심되는 노드를 교체

할당량, 공유, 제한

- 자원을 할당할 때 Pod와 Container의 ResourceQuota를 확인함
- 아래는 몇 가지 잘못된 결과의 예시
- **부족한 자원**: Pod가 일정량의 CPU나 메모리를 요구하지만 용량에 맞는 노드가 없을 경우, 포드는 스케줄링 되지 않음
- **낮은 사용률**: Pod가 요청한 리소스의 일부만 사용하는 경우, 리소스 낭비
- **노드 구성의 불일치**: 높은 CPU와 적은 메모리를 요구하는 Pod가 대용량 메모리 노드에서 스케줄링 될 경우, 다른 Pod는 해당 노드에 스케줄링 될 수 없음, 미사용 메모리 낭비가 발생
- 대시보드 리소스 모니터링으로 확인, `describe`로 자세한 내용을 확인
- <https://kubernetes.io/docs/concepts/policy/resource-quotas/>



잘못된 설정과 비용 관리

- 부정확한 노드나 포드, 라벨
- 복제 컨트롤러 없는 포드 스케줄링
- 부정확한 서비스 포트 명세, ConfigMap
- 필요에 따라 자원이 할당되고 해제되기 때문에 탄력적인 비용

CoreOS



- 도커로 모든 어플리케이션을 컨테이너로 만든다면
- 서버는 도커를 돌리기 위해 존재할 뿐

CoreOS

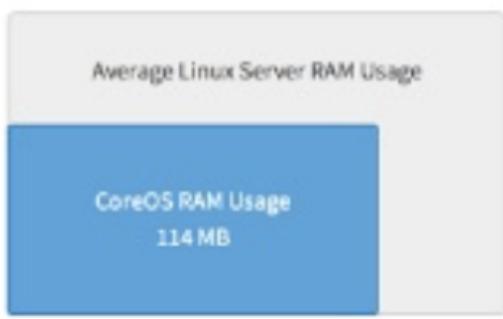


Linux for Massive Server Deployments

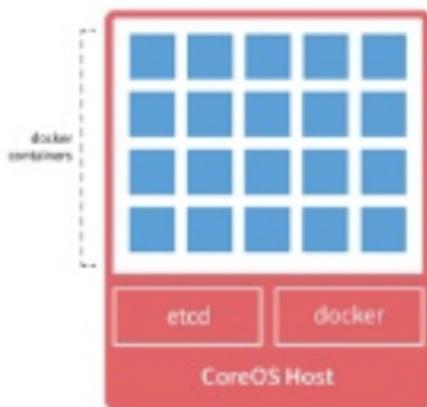
CoreOS enables warehouse-scale computing on top of a minimal, modern operating system.

- Docker 구동에 최적화된 가볍고 최소화된 모던 OS (2013~, RedHat)
- no package manager (apt / yum)
- Chrome OS를 기반으로 몇 가지 기능을 추가하고
- 서버로 사용할 수 있도록 커스터 마이징

CoreOS



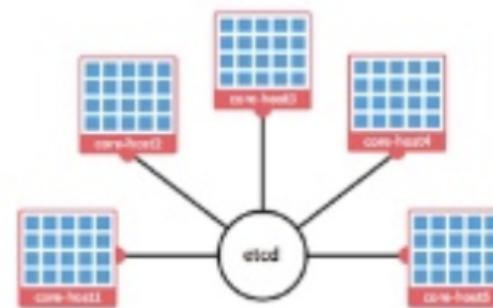
A Minimal Operating System



Docker Containers



Painless Updating



Clustered By Default

CoreOS - etcd, systemd

- distributed key value store (Raft consensus)
- `etcdctl set /message Hello`
- `etcdctl get /message`
- Application IP, PORT를 매핑시키는 용도로도 사용
- 개선된 systemd

CoreOS - Prometheus

