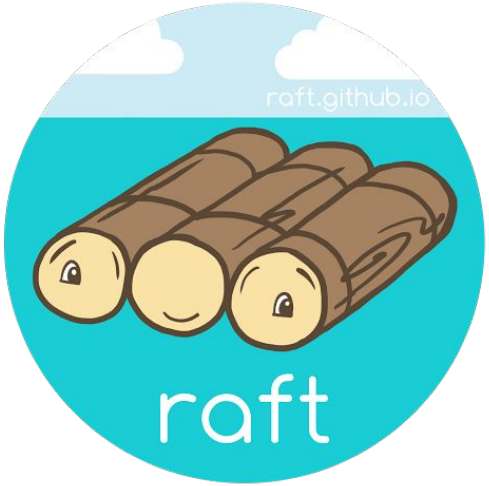


Raft: Introducción

Jessica Barbosa

Cloud Native México

11 de septiembre de 2018

Agenda

- Sistemas distribuidos y consenso
- Raft: animaciones felices
- Raft y Kubernetes (etcd)

Sistemas distribuidos

- ¿Sistema distribuido?
  - Muchas compus (nodos) trabajando juntas para lograr un objetivo común
- ¿Por qué es difícil?

Ejemplito de libro: Los dos generales

- Dos generales atacando una ciudad
- Sólo pueden ganar si atacan al mismo tiempo
- Tienen que ponerse de acuerdo sobre la hora

Sistemas distribuidos y consenso

- ¿Sistema distribuido?
  - Muchas compus (nodos) trabajando juntas para lograr un objetivo común
- ¿Consenso?
  - Todos los nodos del sistema coinciden en el valor que tiene cierta variable (Ej: hora de ataque)
- ¿Por qué es difícil?

happens!

Sistemas distribuidos y consenso

- Muchas cosas pueden salir mal:
  - Un nodo se desconecta y deja de responder
  - Problemas en la red aíslan a un conjunto de nodos
  - Gente maligna interceptando mensajes
- Podemos diseñar muchas soluciones que sacrifican unas cosas para garantizar otras.
  - Paxos
  - Raft

Raft

- Piezas del rompecabezas
- Reglas principales
- Escenarios (poquitos)
- Objetivo:

  Quiero un sistema distribuido en el que haya consenso y que se comporte bien a pesar de 💩

Raft: Las piezas

- Objetivo:

  Máquina de estados replicada en los nodos del sistema y que aguante si algunos nodos mueren

- Máquina de estados: Lista de pasos en un log que, si los aplico en el mismo orden, me llevan al mismo estado final
- Replicada: Hay consistencia en los nodos respecto del contenido del log; todos leemos lo último que se escribió
- Tolera particiones de red: Si se caen ciertos nodos, el cluster sigue respondiendo

Ojo: *Availability* muere si no hay un mínimo número de nodos vivos (CAP)

Raft: Las piezas

- Roles: Líder, candidatos y seguidores
- Todos empiezan como seguidores. Timeout, alguien se propone como candidato. Si consigue el voto de la mayoría de nodos, se vuelve líder y el cluster puede operar felizmente.
- Si nadie gana, otra elección. Si el líder muere, elección.
- Sólo vale un líder por mandato. Se confirma que el líder está vivo vía *heartbeats*

Raft: Las piezas

- Número de mandato. Se incrementa cada que alguien se propone como candidato.
- El tiempo que un nodo tarda en alcanzar el *timeout* es mucho mayor que lo que toma la comunicación entre nodos.
- Se vale no votar por un candidato: por ejemplo, si es más chafa que yo.
- Líder recibe peticiones, las escribe a su log, lo manda a los demás, ya que la mayoría lo escribió, avisa que esa entrada del log está *committeada*

Raft: Animaciones felices

- Clonar el repo
  - https://github.com/ongardie/raftscope
- Instalar dependencias
  - git submodule update --init --recursive
- Levantar un servidor web (caddy, nginx, apache, ...)
- http://localhost:<puerto>
  - 2015 para caddy

Raft: Animaciones felices

- Timeout y elección
- Elección normal
- Elección sin mayoría (matamos dos nodos y *timeouteamos* otros dos)
- Líder chafa que no recibe votos (matar al líder, revivir a uno muerto y proponerlo como candidato)


raft.github.io
raft

&

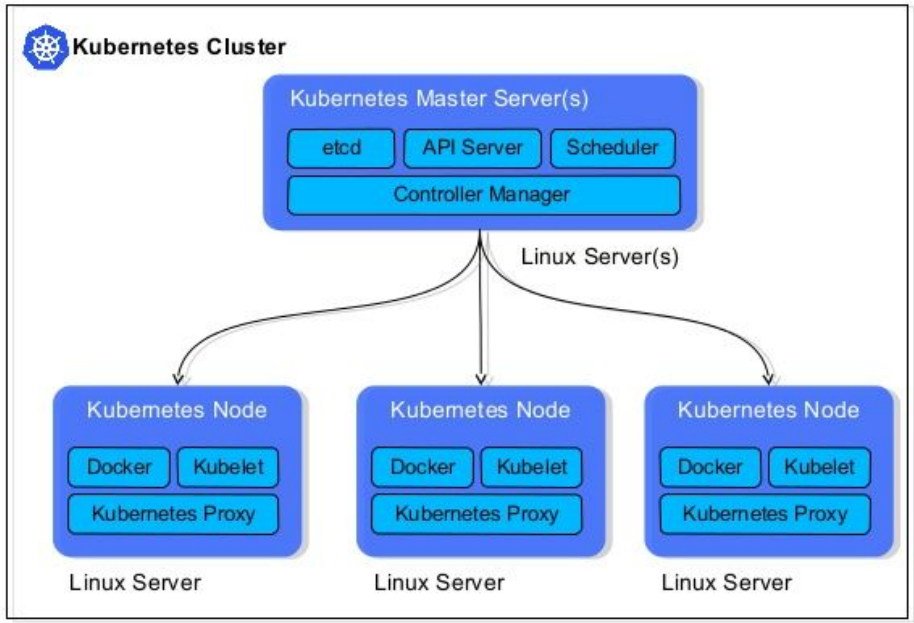
Kubernetes

Características

- Está formado de un administrador y un conjunto de nodos.
- Tiene un scheduler para colocar contenedores en un clúster.
- Tiene un API server y una capa de persistencia con etcd.
- Tiene un controller para conciliar estados.
- Se implementa en máquinas virtuales o máquinas bare metal, en nubes públicas o on-premise.
- Está escrito en Go Lang.

https://www.slideshare.net/domingo.suarez/cloudnativemexicok8s101

Kubernetes

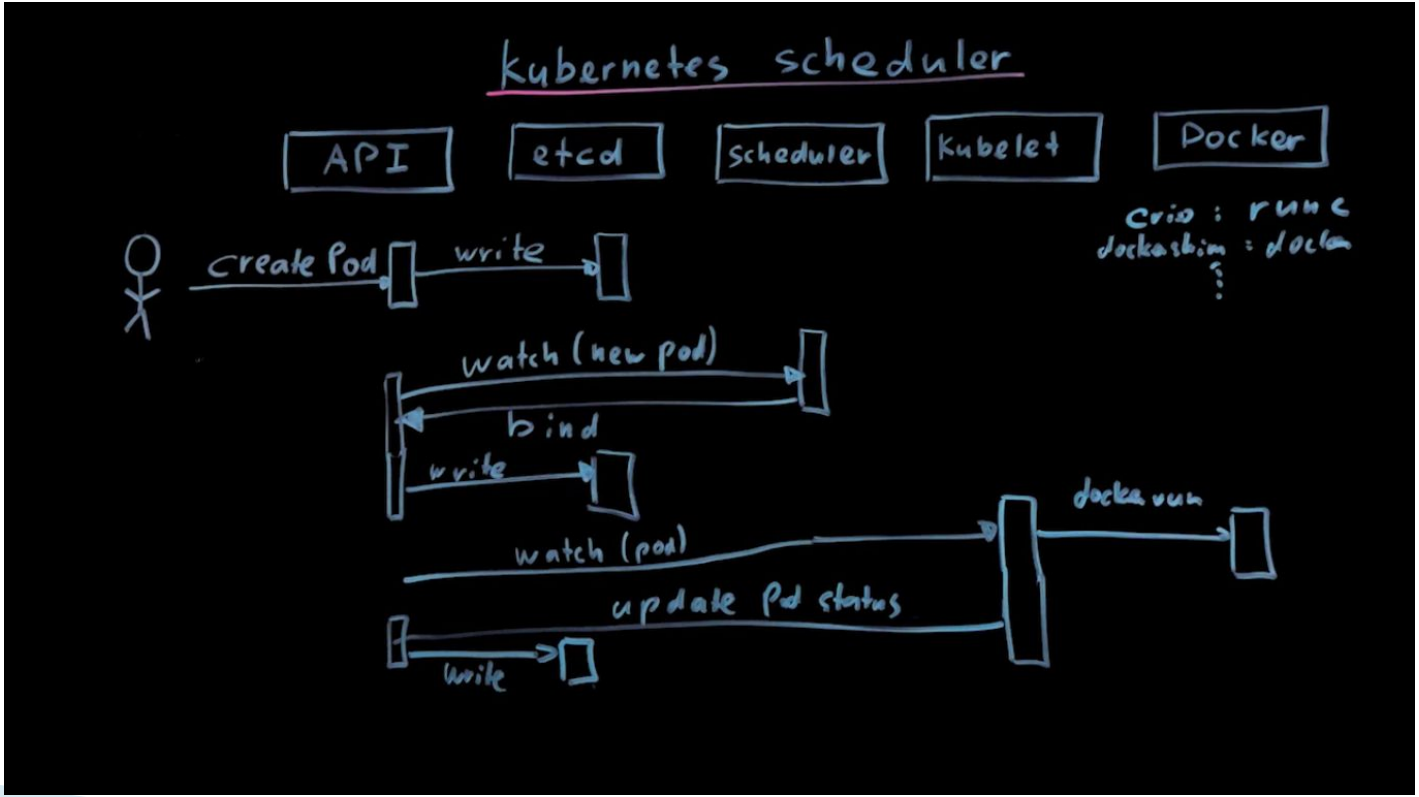


https://www.cbronline.com/enterprise-it/software/how-to-make-containers-enterprise-ready/attachment/kubernetes-architecture/

etcd

- Base de datos de tipo (llave, valor)
- Distribuida: Corre en un cluster de nodos
- Tiene almacenamiento *persistente*
- ¿Qué cosas guarda Kubernetes en etcd?
  - Información de los pods: ¿en qué nodo corre cada pod?
  - Estado del cluster: ¿qué nodos están vivos?
  - Eventos: ¿tiene o no memoria suficiente un nodo?
  - Red: ¿cómo se llama cada nodo (DNS)?

Raft y Kubernetes: etcd



https://www.youtube.com/watch?v=E3ExWruji7g

Referencias

- Esta presentación
  - https://github.com/jbarbosat/mis-desmadres/blob/master/presentaciones/cloud_native_mexico/raft_introduccion.pdf
- Animaciones Raft
  - https://github.com/ongardie/raftscope
- Raft y Kubernetes
  - https://www.youtube.com/watch?v=oosXgmF-77U
- Raft, el paper
  - https://raft.github.io/raft.pdf