Hewlett Packard
Enterprise

Building a Kubernetes Cluster with Ansible

kubernetes

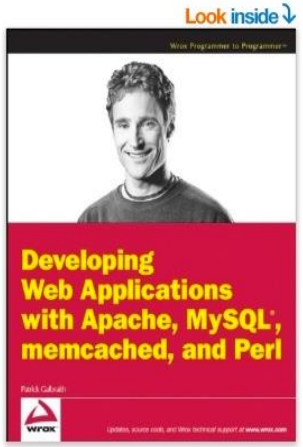
Patrick Galbraith, ATG
Cloud Computing Expo, NYC, May 2016

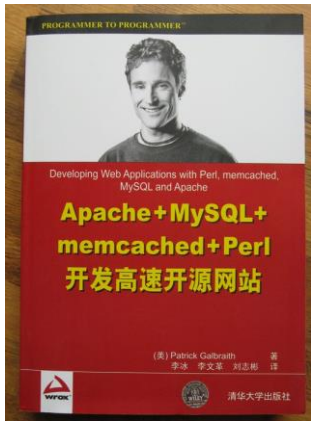
HPE ATG

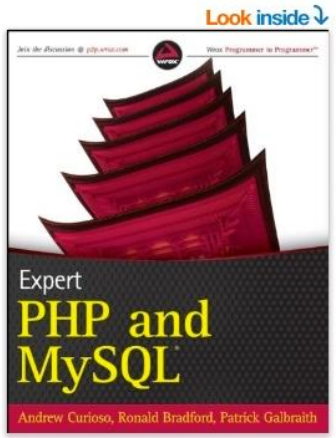
HPE's (HP Enterprise) Advanced Technology Group for Open Source and Cloud embraces a vision that is two steps ahead of today's solutions. We use this vision to drive product adoption and incubate technologies to advance HPE. Through Open Source initiatives we foster collaboration across HPE and beyond.

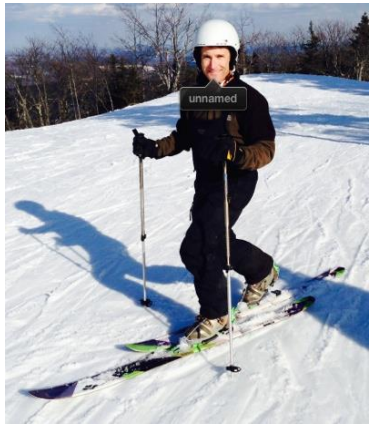
About the speaker

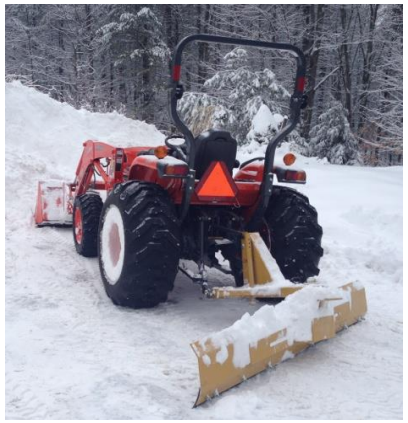
- Patrick Galbraith
- HP Advanced Technology Group
- Has worked at Blue Gecko, MySQL AB, Classmates, Slashdot, Cobalt Group, US Navy, K-mart
- MySQL projects: memcached UDFs, DBD::mysql, federated storage engine
- Family
- Outdoors











Purpose of this talk – why are you here?

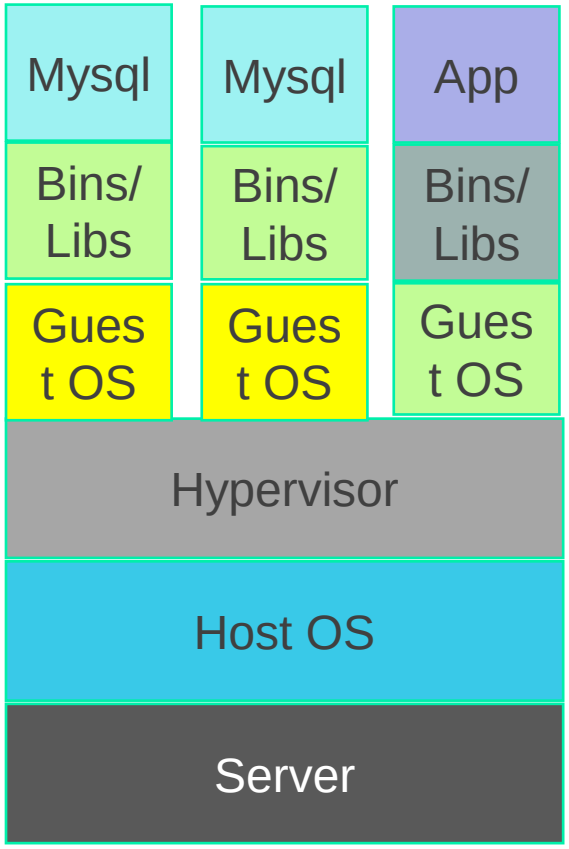
- Docker
  - Containers vs. Virtualization
  - Simple Docker usage
  - Clustered Docker
- Kubernetes
  - Understand what Kubernetes is
  - How to set up Kubernetes
- Ansible
  - What is Ansible?
  - How can Ansible build Kubernetes?

What are containers?

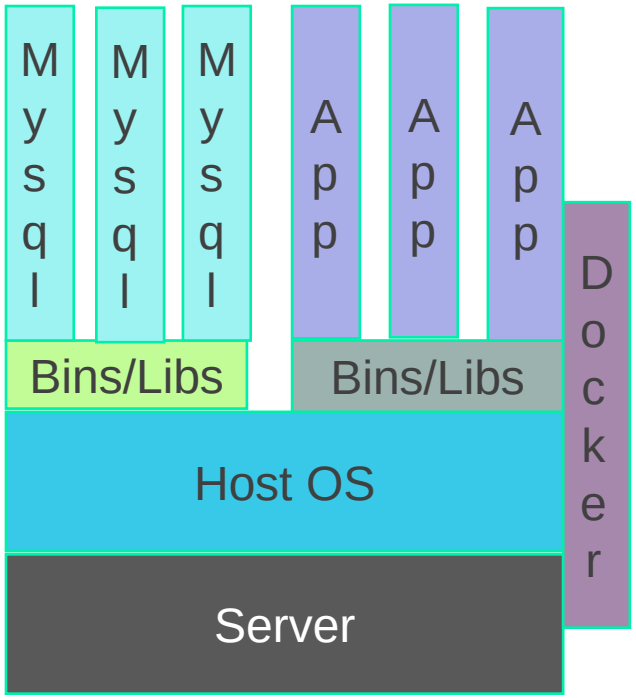
- Operating-system-level isolation
- Encapsulated, hermetically sealed applications
- Portable. And did I mention, portable?!
- Small footprint of container images
- Fast to launch!
- Use of host OS and Kernel
- Execution consists of time to startup application in question
- LXC, Docker, Solaris Zones, BSD Jails, Parallels Virtuozzo, OpenVZ, …

VM vs. Containers

VM



Containers



What is Docker?

- Set of tools for managing containers
- Command line tool that doubles as a daemon
- Kernel namespaces – the core ingredient to containers working
  - PID
  - IPC
  - uts (what will be seen by a group of processes)
  - Mount
  - Network
  - User
  - Cgroups (control groups) -- limit, account and isolate resource usage (CPU, memory, disk I/O, etc.) of process groups

- Originally used lxc, now defaults to Libcontainer but meant for any containerization mechanism
- Much more light weight than VMs
- Encapsulated application containers in a relatively isolated but lightweight operating environment
- Written in Go

Docker – common terms and usage

- Dockerfile
- EXPOSE ports
- Entrypoints and CMD
- docker build
- docker push
- docker run
- docker inspect
- docker exec
- docker commit

Dockerfile

- https://github.com/CaptTofu/percona_xtradb_cluster_docker

Running a docker container

```
$ docker run \
    --name mybox  \
    -e MYSQL_ROOT_PASSWORD=secret \
    -d \
    mysql/mysql-server --log-bin --server-id=100$

$ cat minimal.cnf
[mysqld]
user=mysql
log-bin=mysql-bin
server-id=100

$ docker run \
    --name mybox  \
    -e MYSQL_ROOT_PASSWORD=secret \
    -d --hostname mybox \
    -v $PWD/minimal.cnf:/etc/my.cnf
    mysql/mysql-server
```

Hewlett Packard Enterprise

Clustered Docker

- Kubernetes -- http://kubernetes.io
- CoreOS -- https://coreos.com/
- Mesos + Marathon -- http://mesos.apache.org/ Apache project, Zookeeper, etc
- Project Atomic -- http://www.projectatomic.io/-- RH/Fedora/Centos designed for ru
- Docker Openstack -- https://wiki.openstack.org/wiki/Docker Hypervisor Driver for
- Swarm/Compose/Machine
- RancherOS http://rancher.com/rancher-os Minimalist Linux, Docker daemon runs
- Flocker -- https://clusterhq.com
- Spotify Helios -- https://github.com/spotify/helios -- Zookeeper
- Deis (http://deis.io)
- Maestro (https://github.com/toscanini/maestro)
- Shipyard (http://shipyard-project.com)
- … others to come!

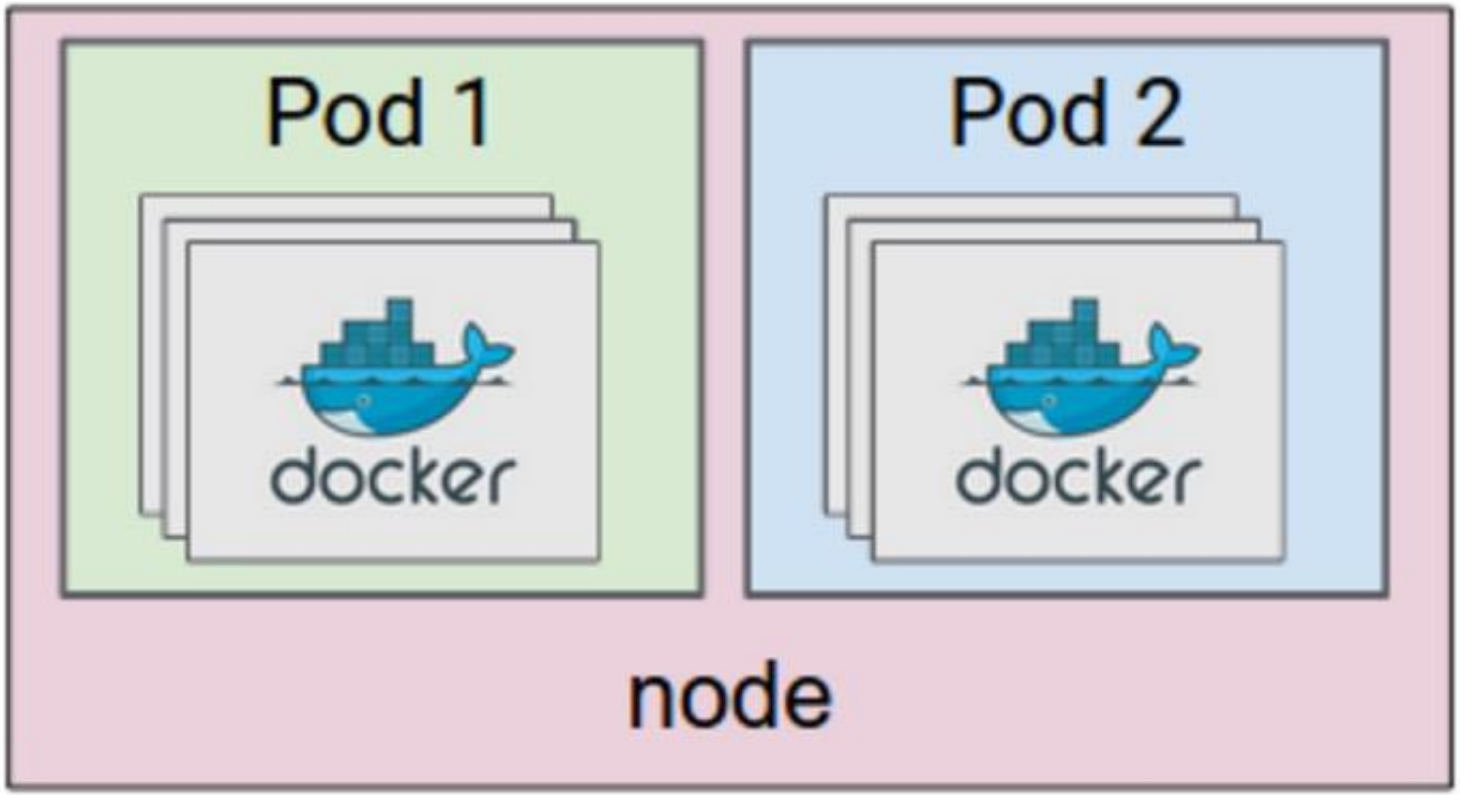
Kubernetes

- Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications”
- Designed on the same principles that allows Google to run billions of containers a week – Borg paper: https://research.google.com/pubs/pub43438.html
- Scale without increasing your ops team.
- Lean
- Portable – will run cloud, bare metal, hybrid, etc
- Extensible – using modular design allowing for plug-ability and hooks
- Automatic inpacking – ensures container placement per resource requirements
- Self-healing – auto-placement, auto-restart, auto-replication
- Batch execution – can manage batch in CI workloads
- Google engineering bring good work to the Open-source world

Kubernetes concepts

- Pod
  - Group of closely-related containers on the same host
- Service
  - Virtual abstraction
  - Basic load-balancer
  - Single consistent access point to a pod
- Replication controller
  - Defines pods to be horizontally scaled
  - Uses a label query for identifying what containers to run
  - Maintains specified number of replicas of a particular thing to run
  - Dynamic resizing
- Label
  - Key/value tag to mark work units a part of group
  - Management and action targeting
- Definition file – YAML/json describing a pod, service, or replication controller

Kubernetes pod



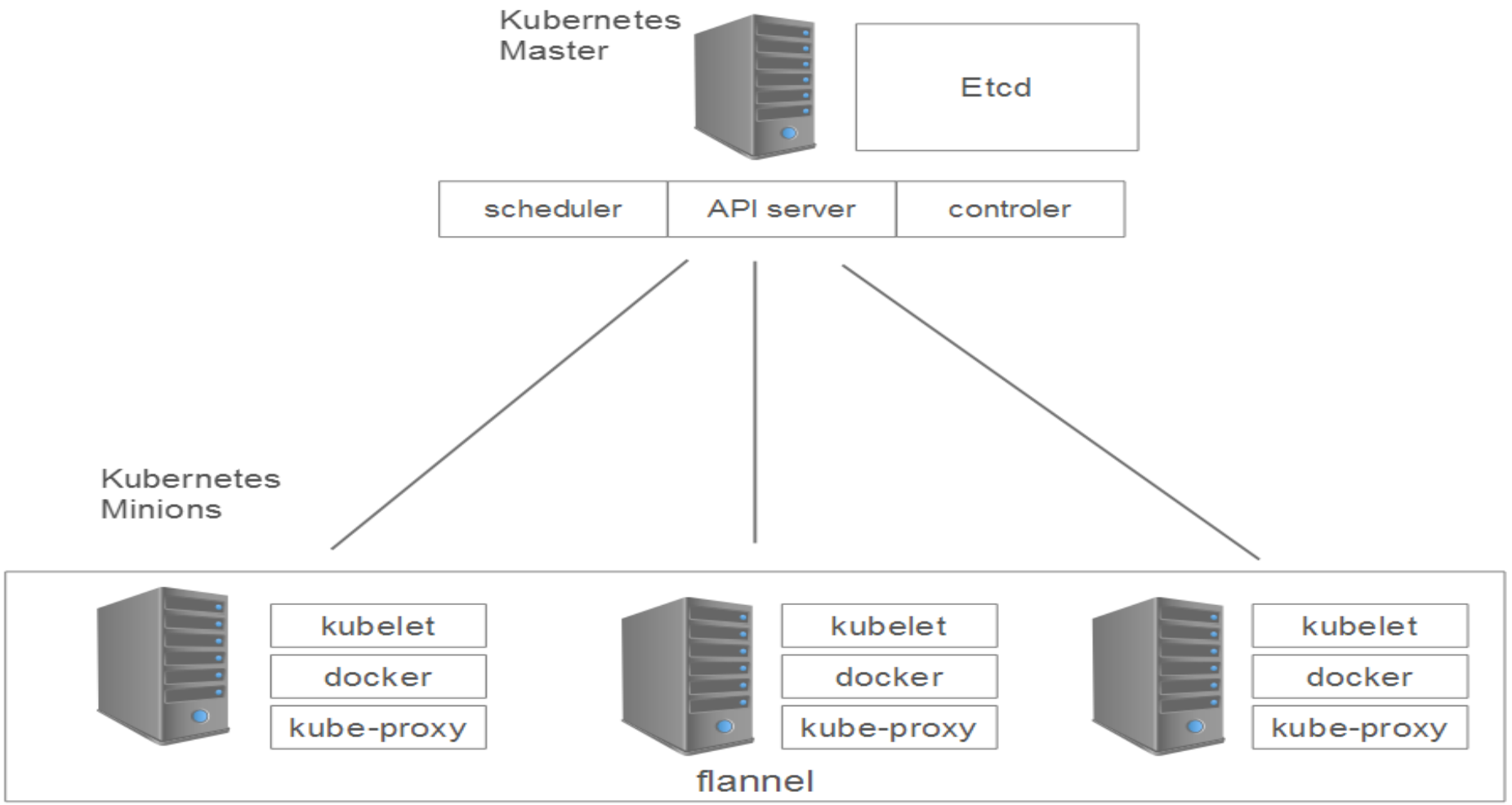
Kubernetes Master

- kube-apiserver – API Server (RESTful)
  - primary management for cluster
  - reconciles etcd entries with deployed containers
- kube-controller-manager — Controller Manager Server
  - Handle replication processes defined by replication tasks
  - Writes details to etcd
  - Monitors changes and implements procedure to reflect the change
- kube-scheduler -- Scheduler Server
  - Assigns workloads to specific minions in cluster taking into account service's operating requirements and infrastructure environment
- kube-register -- Register Server

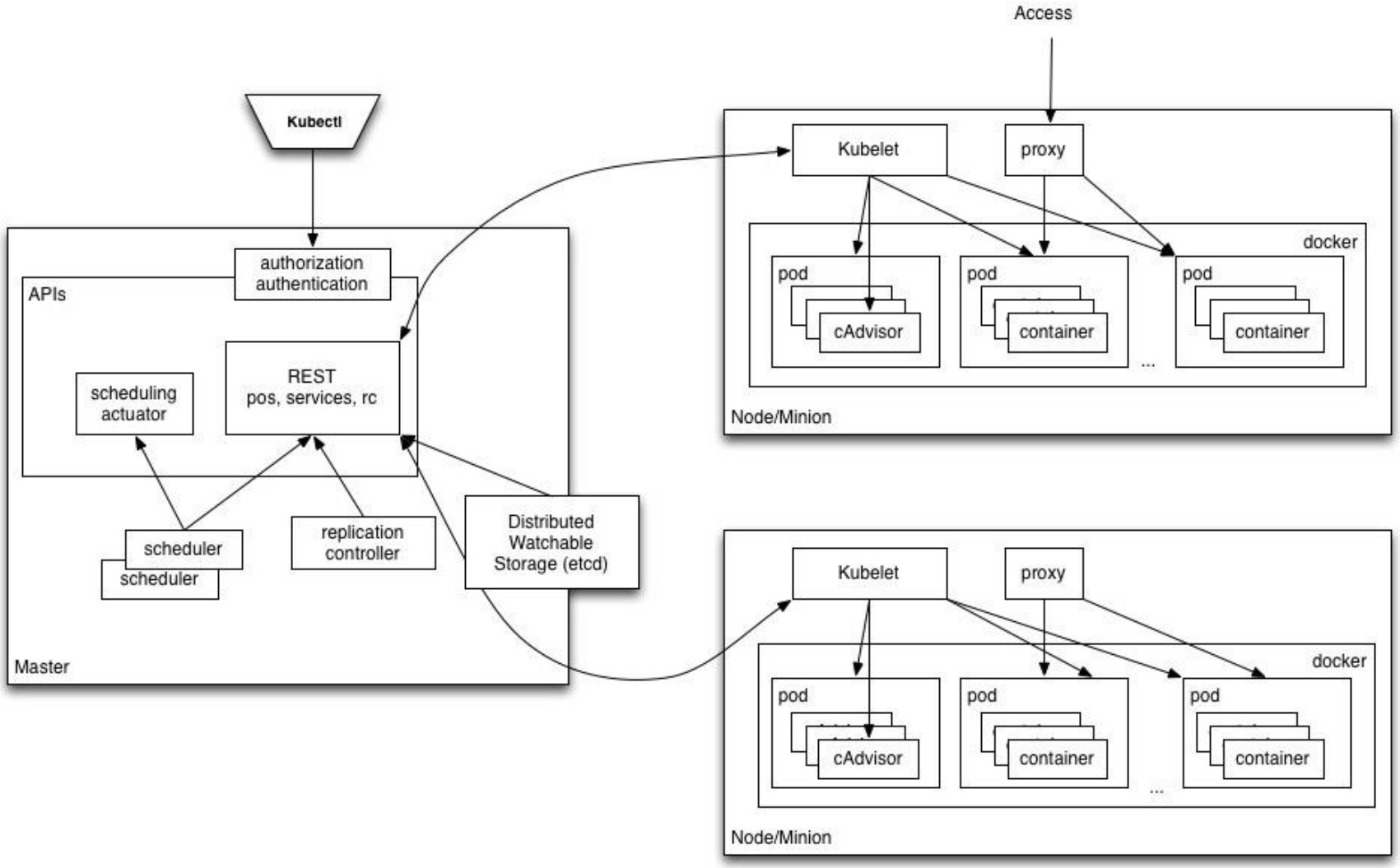
Kubernetes minion

- kubelet
  - Communicates with the master, relaying information to/from
  - Reads and updates etcd
  - Takes a set of PodSpecs and ensures that the containers described in those PodSpecs are running and healthy.
  - Receives work in a manifest that defines the workload and operating parameters.
  - Assumes responsibility for the state of work on minion
- kube-proxy
  - Ensures network environment is accessible but isolated.
  - Makes services available externally by forwarding requests to containers.
  - Can perform rudimentary load balancing.

Kubernetes Basic Setup



Kubernetes diagram



Kubernetes usage

- Pod configuration file – YAML or JSON
- Service configuration file
- Replication controller configuration file
- export KUBERNETES_API=http://kube-master:8080
- kubectl create –f mysql_master.json
- kubectl create –f mysql_master_service.json
- …

How can I run Kubernetes?

- https://github.com/CaptTofu/vagrant-kubernetes-cluster.git (this talk!)
- https://github.com/Samsung-AG/kraken.git
- http://kubernetes.io/docs/hellonode/
- https://github.com/pires/kubernetes-vagrant-coreos-cluster
- https://github.com/TheNewNormal/kube-solo-osx
- …

Other deployment Strategies for Kubernetes

- Helm -- http://helm.sh/ -- Package Manager for k8s
- Deis v2 – Builds, deploys, 12-factor PaaS
- Kupak
- RedSpread
- KPM

More info, projects

- A curated list for awesome kubernetes sources -- https://github.com/ramitsurana/awesome-kubernetes

How do I build Kubernetes?

- Basic control plane
  - Docker
  - Flannel
- Set up containerized apt-repository
- Set up private docker registry
- Install master components: api server, controller manager, scheduler
- Start master components
- Install minion components: kubelet and kube-proxy
- Start up minion components

Ansible

- Automation Engine
  - Application deployment, configuration management, provisioning, orchestration

- Agentless / SSH connections
  - Push model – programs (modules) pushed to nodes and executed over SSH
  - Copies files to remote location being configured, executes, wipes

- Inventory described and managed in a text file
  - Inventory can be static or dynamic

- Playbooks: the Ansible orchestration language
  - YAML file, designed to be human readable

Ansible: Inventory file

```
[southwest:children]
arizona
new-mexico

[arizona]
phoenix
tuscon

[new-mexico]
albuquerque
santa-fe
```

- List of hosts being managed
- Grouped into categories (master/minion, regions, type)
- Hierarchical

Ansible: Inventory file

```
[kubernetes]
[kubernetes:children]
kubernetes-master
kubernetes-minions

[kubernetes-master]
kubernetes-master-001

[kubernetes-minions]
kubernetes-minion—000
kubernetes-minion-001
kubernetes-minion-002
```

Ansible: Playbook, example top level

```
---

- hosts: "{{ target|default('kubernetes') }}"
  roles:
  - common
  - docker-registry
  - docker-private-registry

- hosts: "{{ target|default('kubernetes-master') }}"
  roles:
  - kuberneter-master
```

- Language of Configuration, deployment and orchestration
- Describe configuration you want to enforce
- Contains "plays" (steps of process being executed)
  - Map to specific groups of hosts
  - Include roles (pre-packaged units of work)

Ansible: Playbook

```
- name: Install the apt key for ubuntu
  apt_key: id=7F0CEB10 keyserver="keyserver.ubuntu.com" state=present

- name: Install the repository for Ubuntu mongodb  apt_repository: repo="deb http://repo.mongodb.org/apt/{{
  ansible_os_family|lower }} {{ debian_version.stdout }}/mongodb-org/3.0 main" state=present
- name: pre-create ntp group, system  group: name=ntp system=yes state=present

- name: pre-create ntp user, system  user: name=ntp group=ntp system=yes state=present

- name: install NTP
  apt: name=ntp state=present update_cache=yes
  notify: restart ntp

- name: install various packages
  apt: name={{ item }} state=present update_cache=yes
  with_items: common_packages  when: ansible_os_family == "Debian"
```

Ansible modules

- Module library
- Usually Written in Python
- Use a common API for returning json to Ansible to indicate failure or success
- For just about everything you would need!
- Divided into core and extras

Using Ansible to build Kubernetes

- Roles: (https://github.com/kubernetes-cluster-automation)
  - docker-apt-repository
  - docker-registry
  - docker-private-registry
  - etcd
  - flanneld
  - kubernetes-master
  - kubernetes-minion

Docker Registry setup

- Two containers
  - Docker registry, uses volume for images
  - Docker registry proxy (nginx)
    - Linked to docker-registry container for certs and .htpasswd
    - Generation and setup of self-signed SSL cert
    - Generation and setup of htpassword
    - Set up Docker on all hosts
      - /etc/docker/certs.d for all hosts
      - Set up of /root/.docker/config.json to automatically be logged in for all hosts

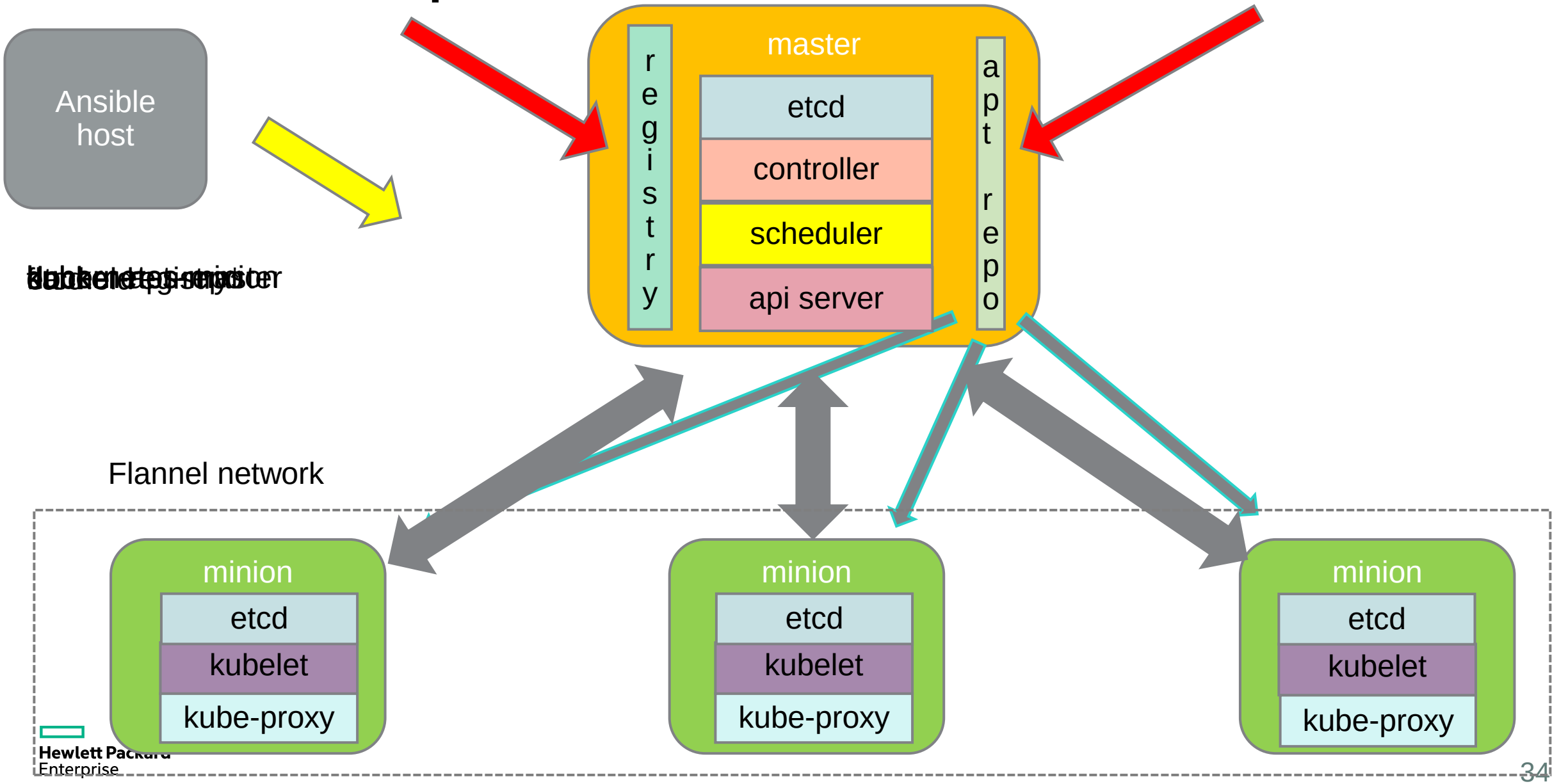
Etcd and flannel

- Install etcd and flannel apt packages
- Generate systemd unit files
- Configure etcd cluster
- Modify docker systemd unit file to use flannel
- Start up flannel and restart docker
- End of a run, flannel network is set up and containers on all machines can ping each other

Kubernetes

- kubernetes-master
  - Install apt package
  - Generate systemd unit files
  - Start api server, controller manager, and scheduler
- kubernetes-minion
  - Install apt package
  - Generate systemd unit files
  - Start kubelete and kube-proxy
  - Load SkyDNS pod and service files

Ansible set up of Kubernetes



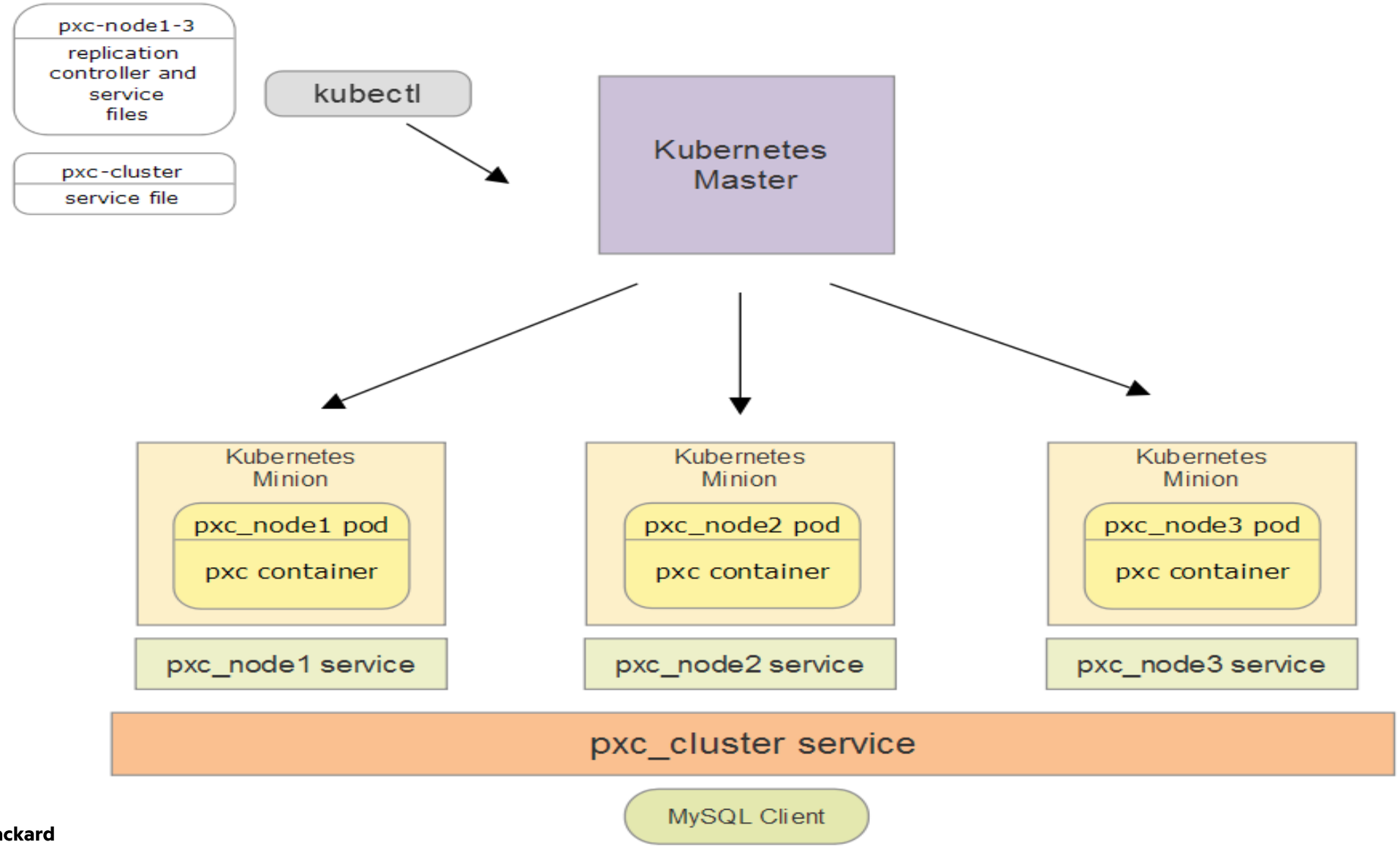
Using Kubernetes

https://github.com/kubernetes/kubernetes/tree/master/examples/mysql-galera

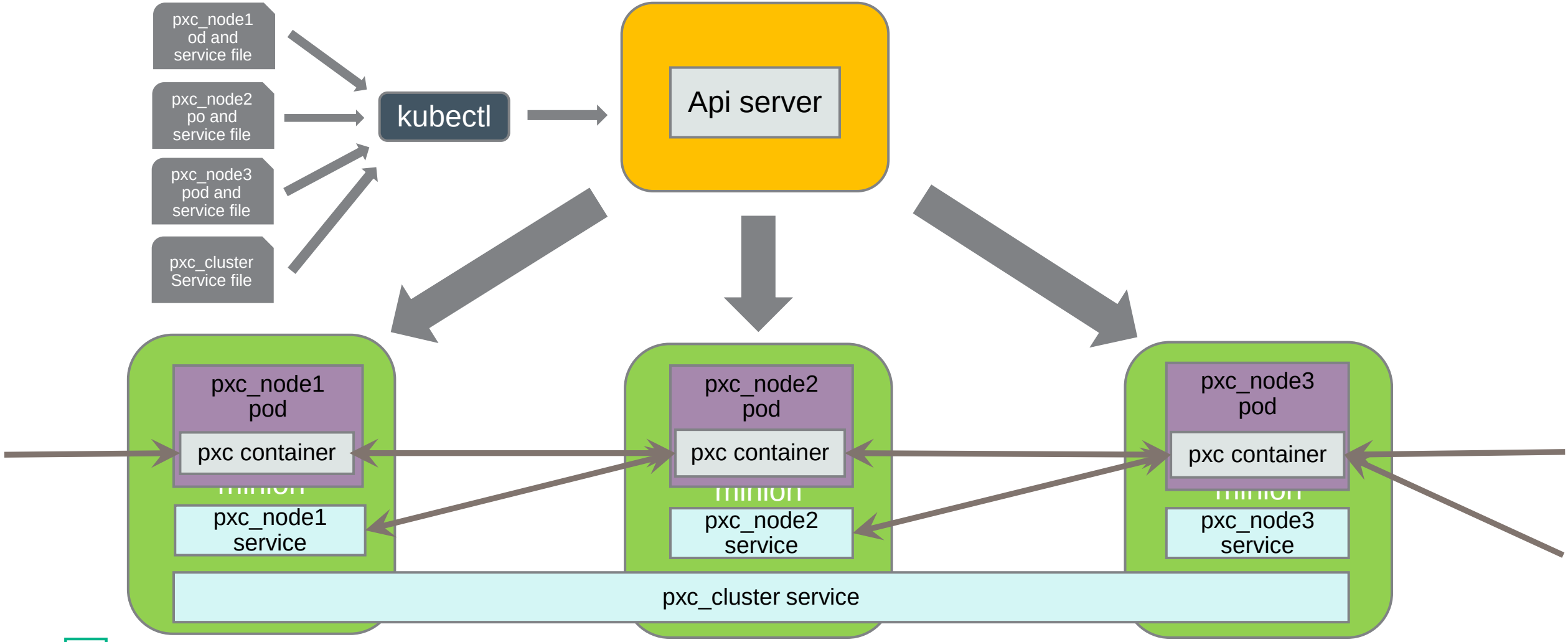
Galera replication on Kubernetes

https://github.com/kubernetes/kubernetes/tree/master/examples/mysql-galera

Galera on Kubernetes



Galera on Kubernetes Process



Thank you! Questions?

Demo

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