OpenStack and Kubernetes

Tomasz Paszkowski

Agenda

- Kubernetes and OpenStack - introduction
- Why OpenStack on Kubernetes?
- OpenStack on Kubernetes - Evolution of Technology
- Summary
- Q&A

Kubernetes

- Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.
- Kubernetes is a production-grade, open-source platform that orchestrates the placement (scheduling) and execution of application containers within and across computer clusters.

OpenStack

- OpenStack is an open-source system for IaaS.
- OpenStack is a production-grade, open-source platform that orchestrates the placement (scheduling) and execution of VMs and delivers necessary software defined infrastucture for them.

Why OpenStack on Kubernetes ?

- Solves multiple engineering challenges within datacenter
  - Service discovery
  - Simple and clean design for services, deployment, upgrades
  - Scale up/down, self-healing, unified, repeatable deployment with containers and manifests
  - Ease of installation
  - Maximizing infrastructure utilization, by workload collocation (containers and VMs)
- Excellent platform to take advantage of rolling upgrades within OpenStack (zero downtime upgrades)

Why OpenStack on Kubernetes ?

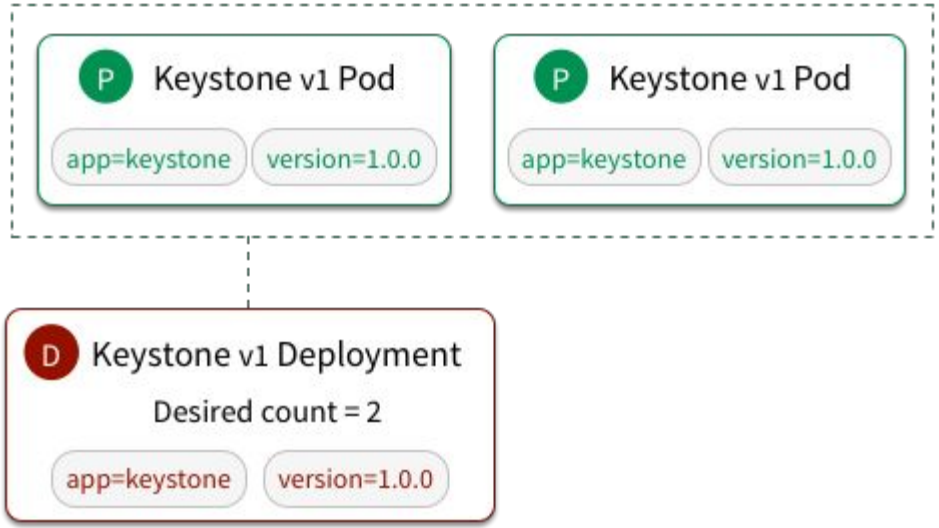
- OpenStack historically was hard to deploy and troubleshoot, maintain
- Kubernetes is an excellent orchestration platform to take advantage of rolling upgrades within OpenStack (zero downtime upgrades)

Deploying an application

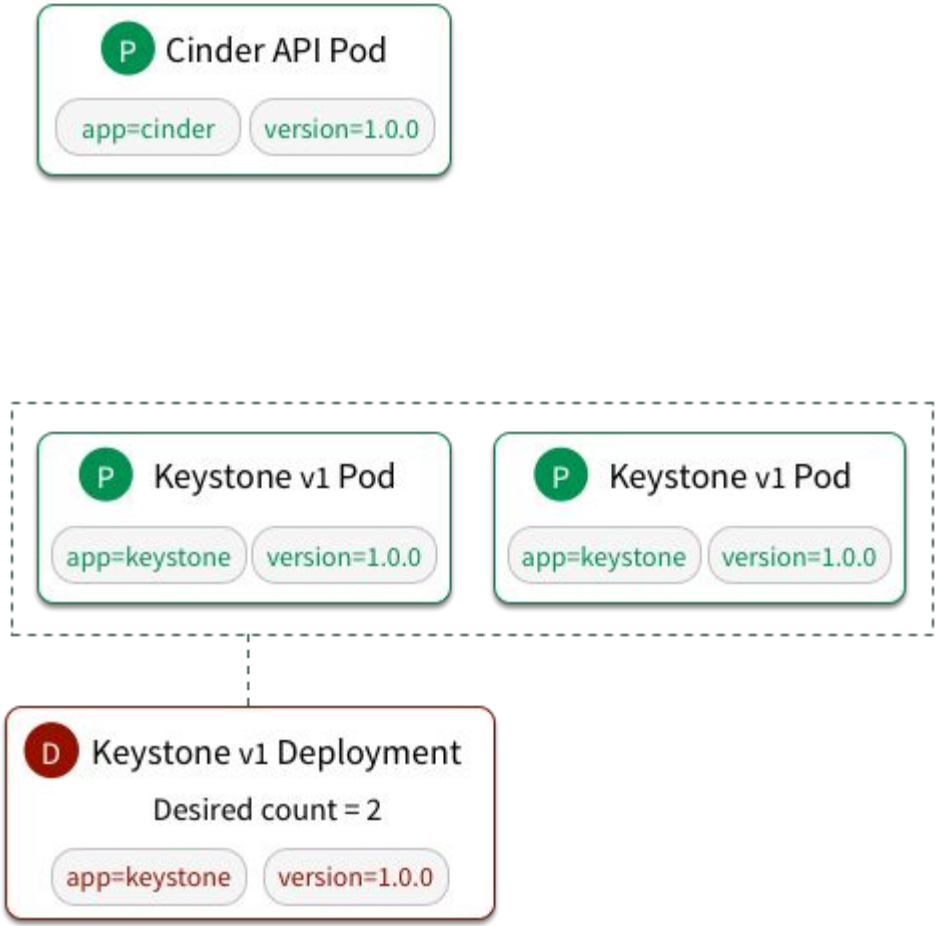
Initial declaration

Deploying an application

Automatic reconciliation

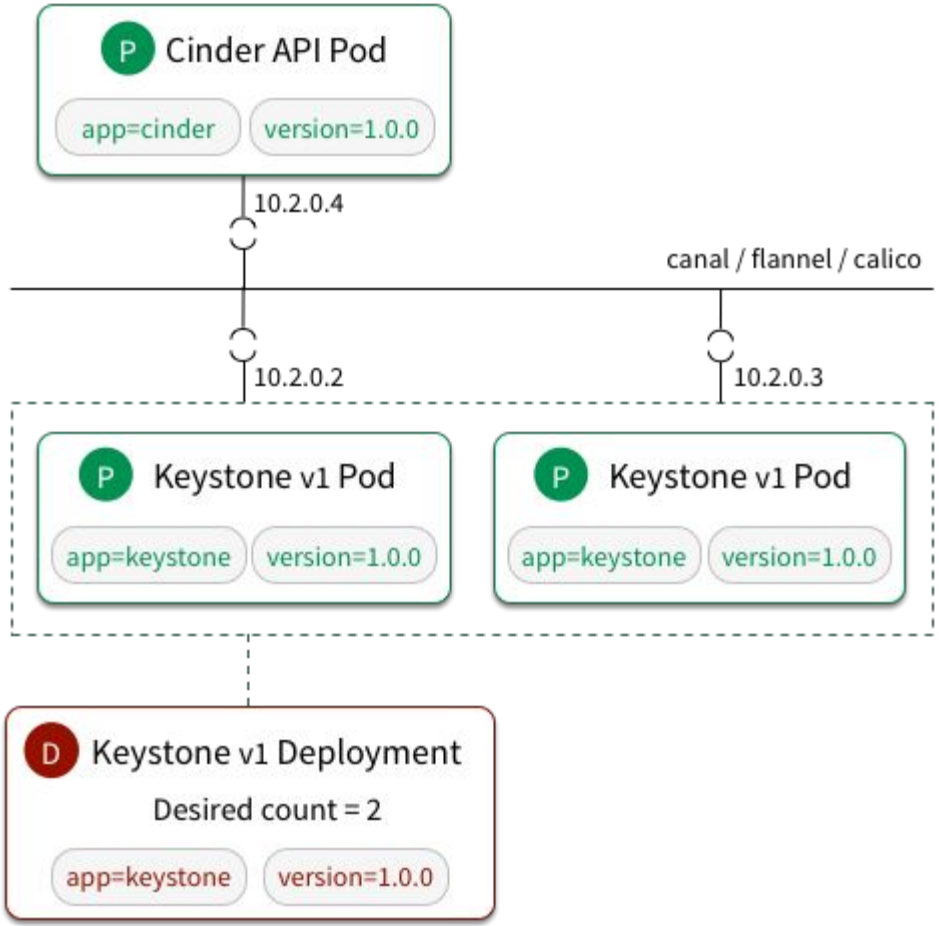


Contacting an application



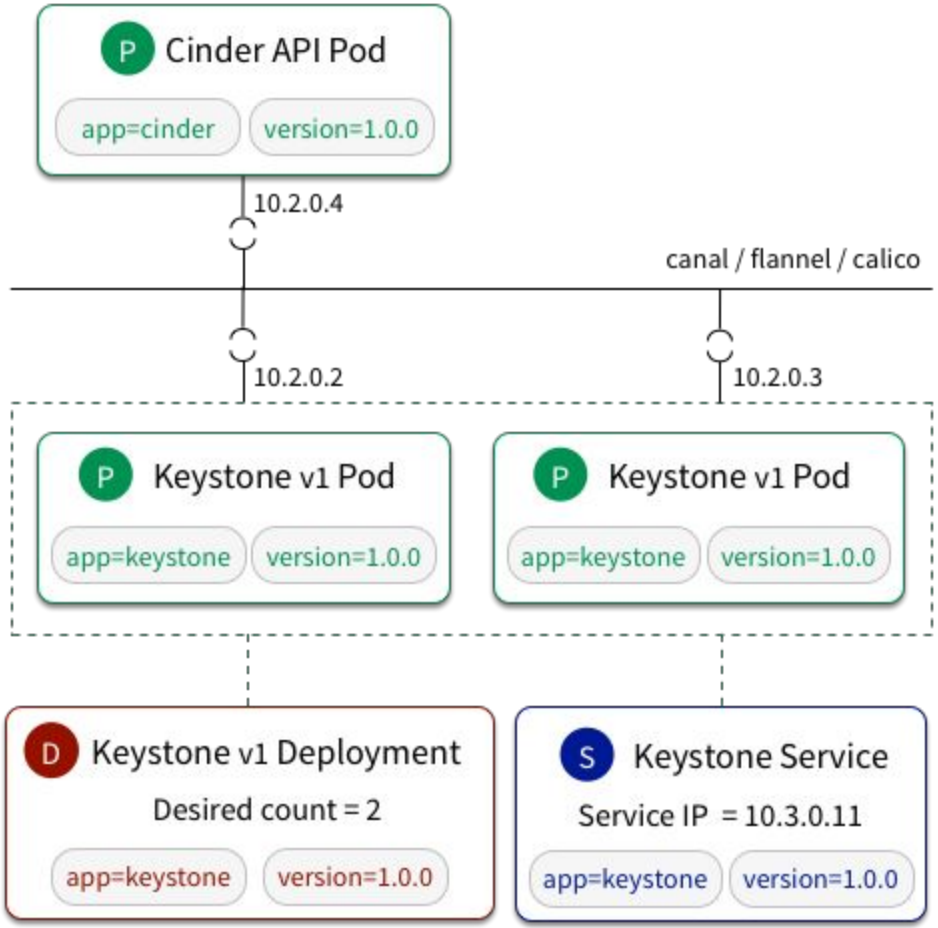
Contacting an application

Overlay network



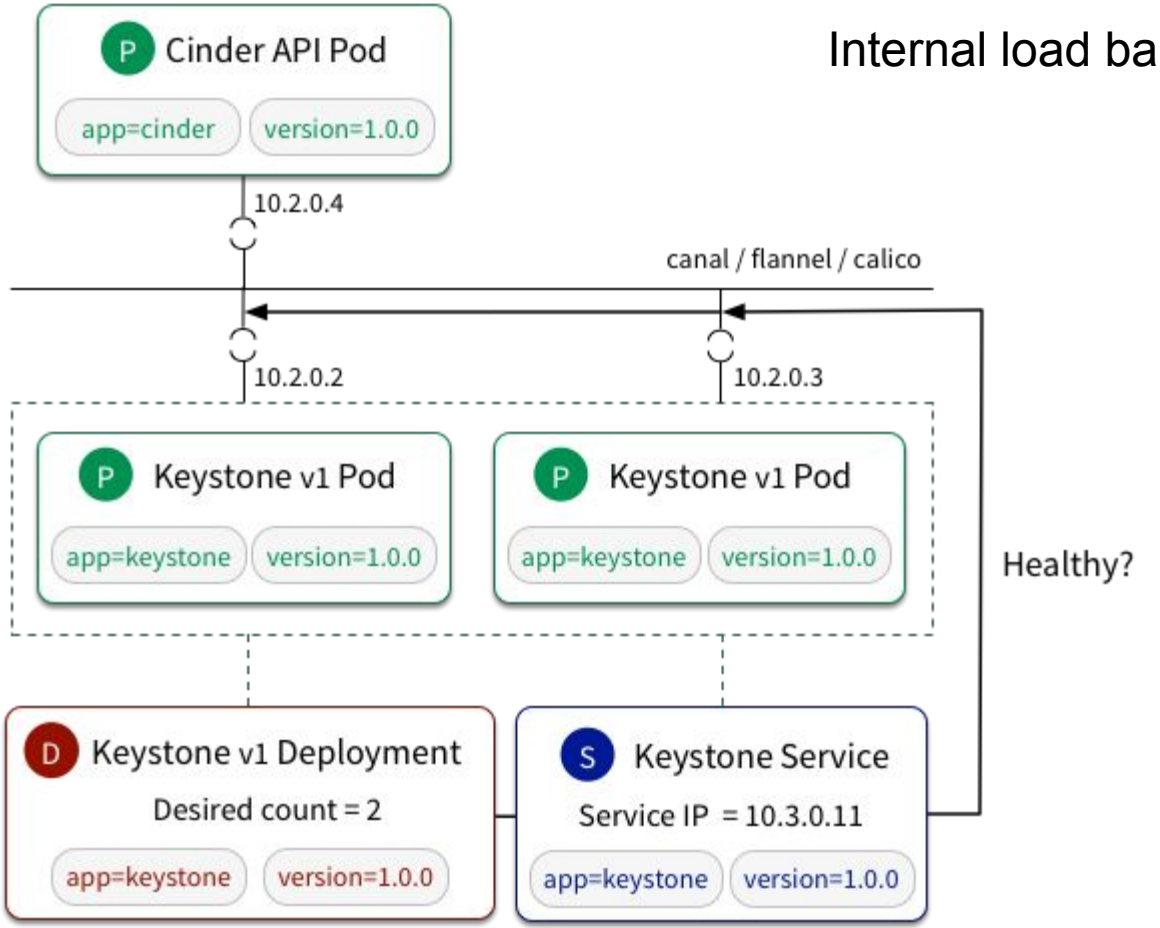
Contacting an application

Internal load balancing



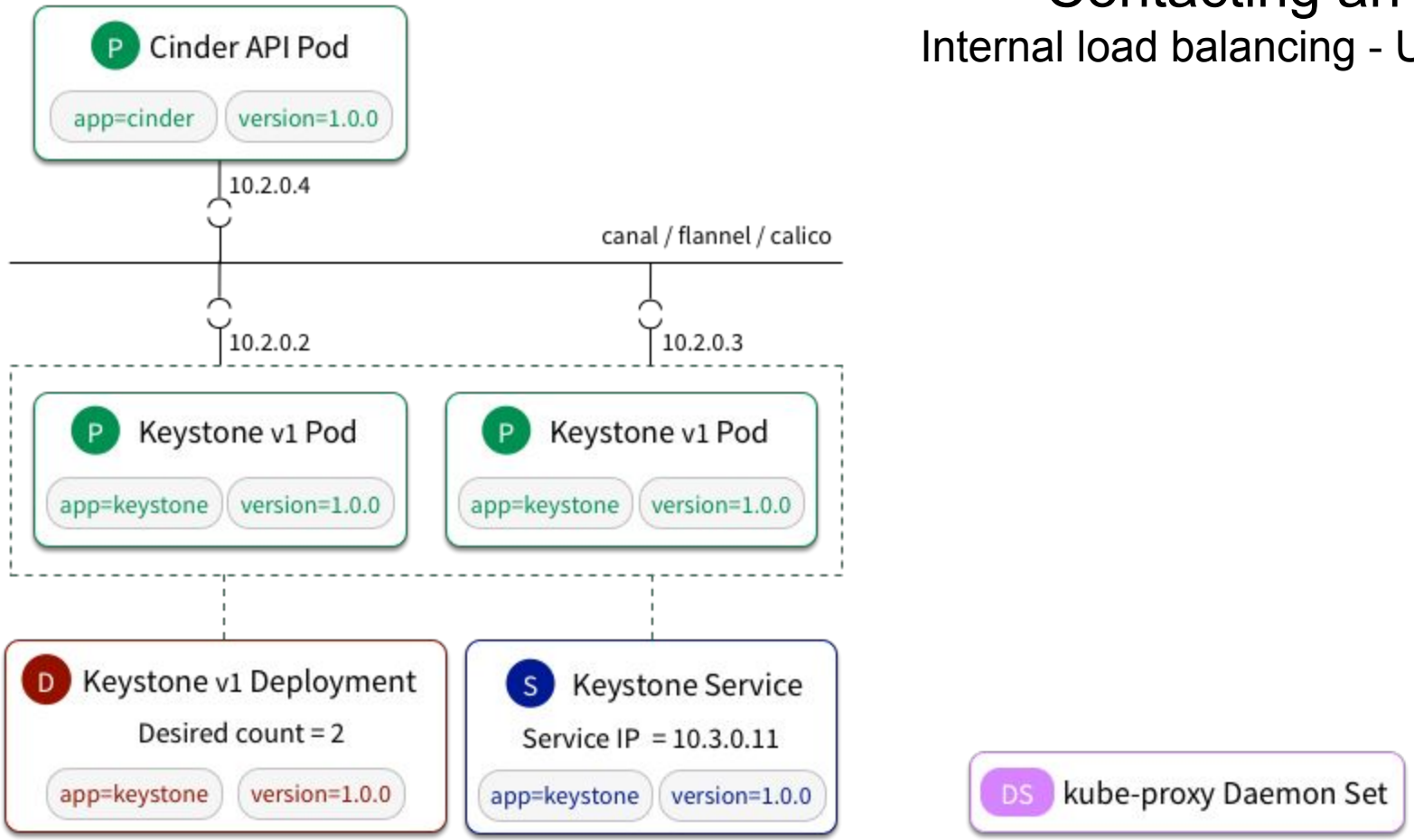
Contacting an application

Internal load balancing - Health checks & Healing



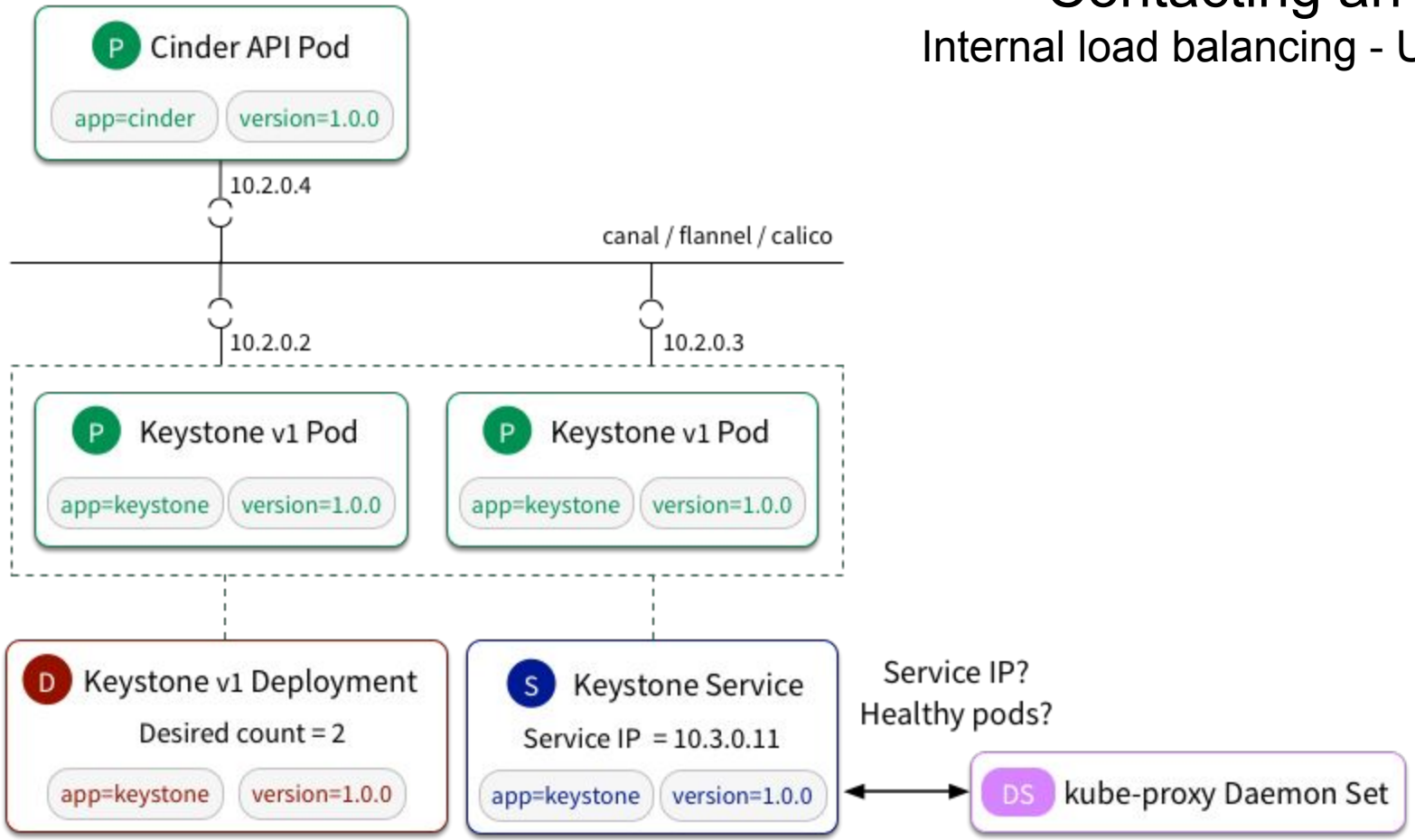
Contacting an application

Internal load balancing - Under the hood



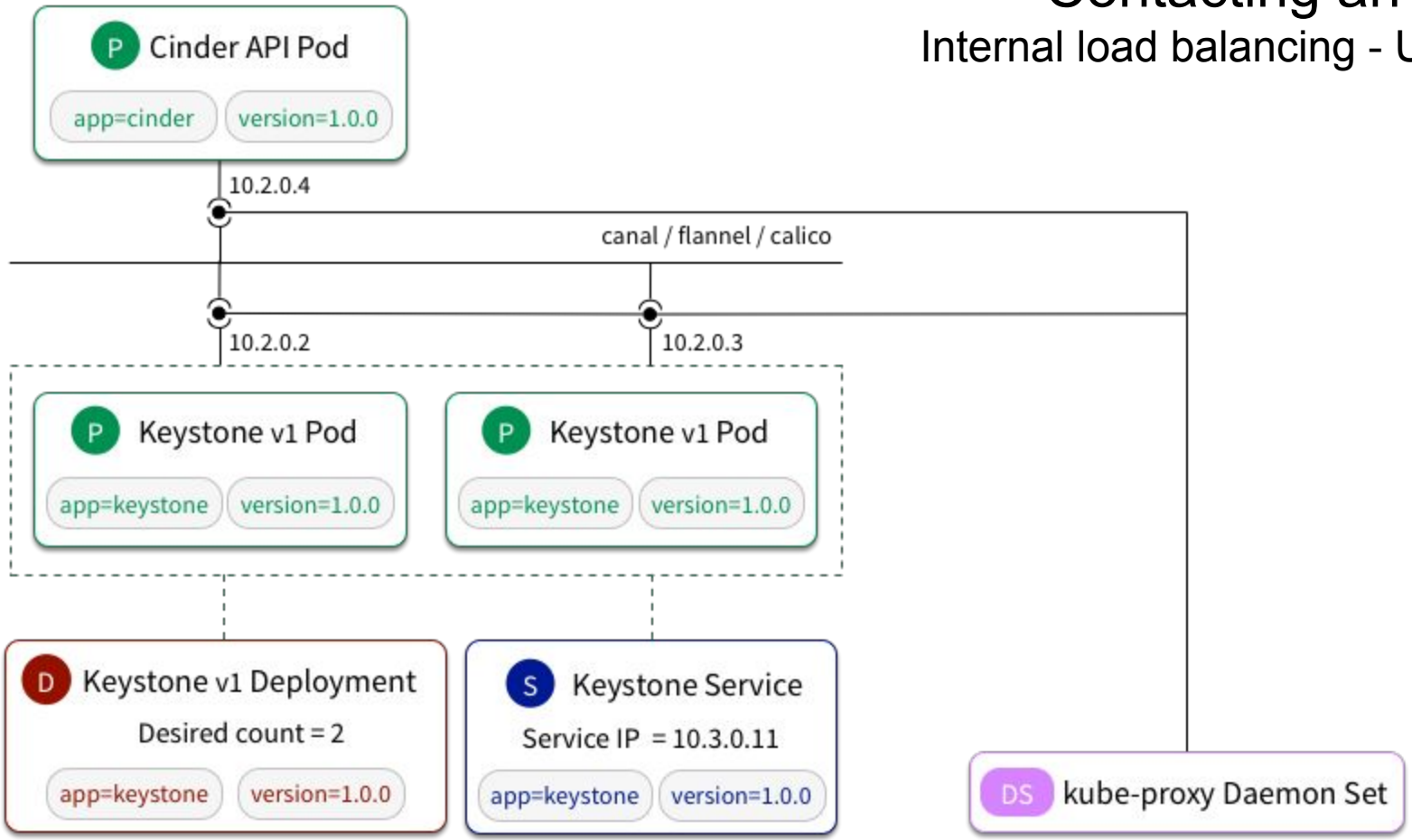
Contacting an application

Internal load balancing - Under the hood



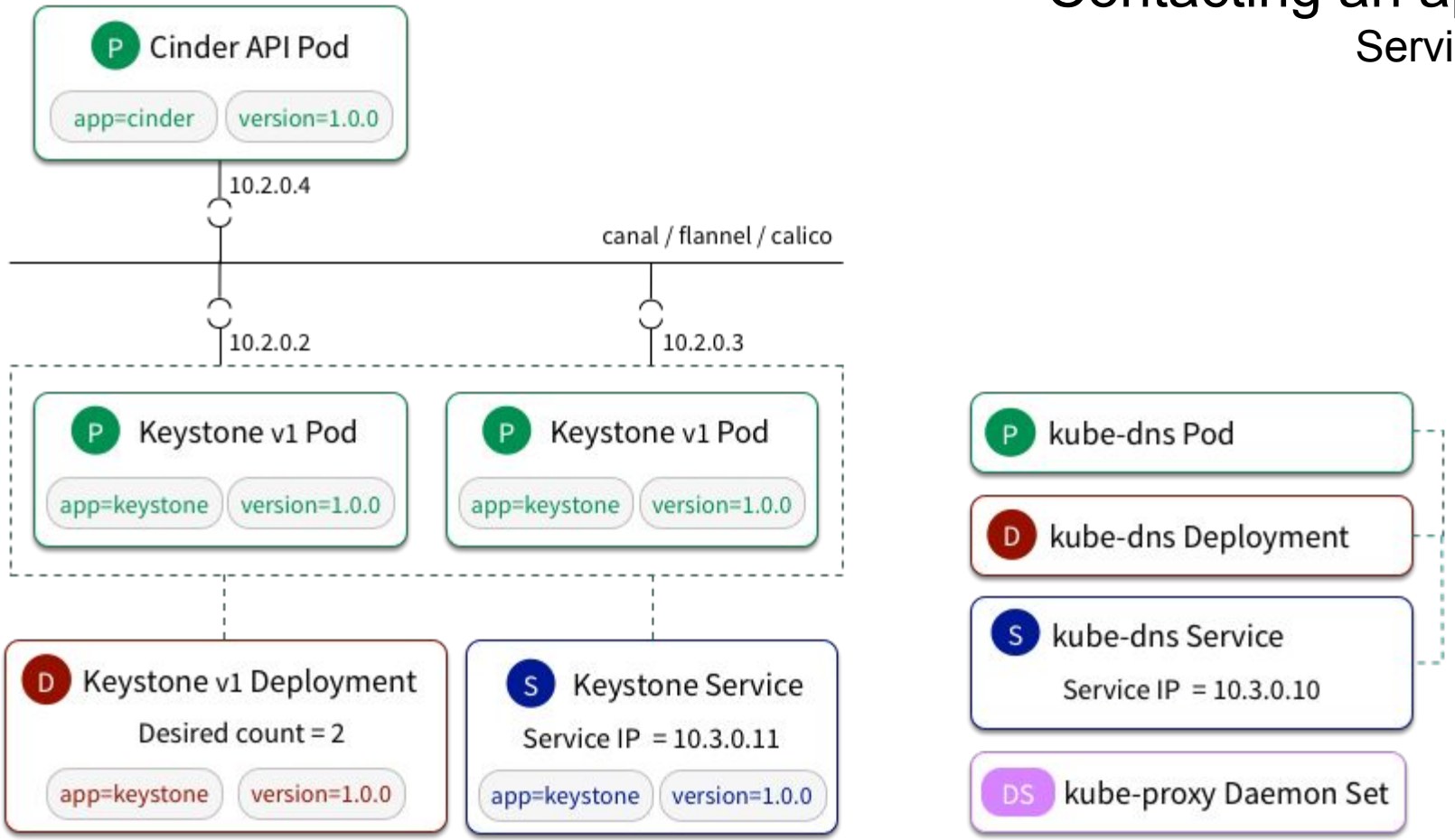
Contacting an application

Internal load balancing - Under the hood



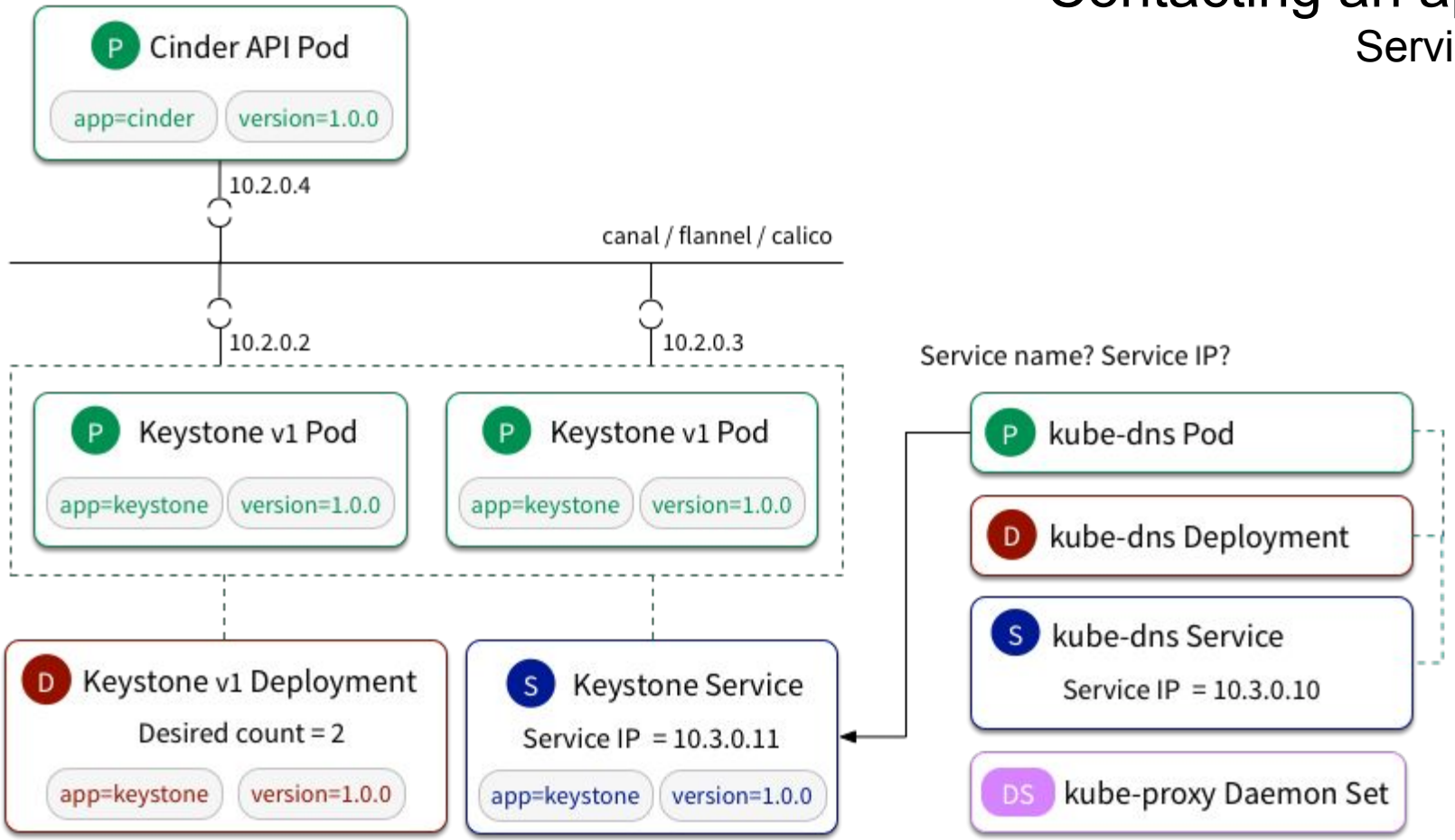
Contacting an application

Service discovery

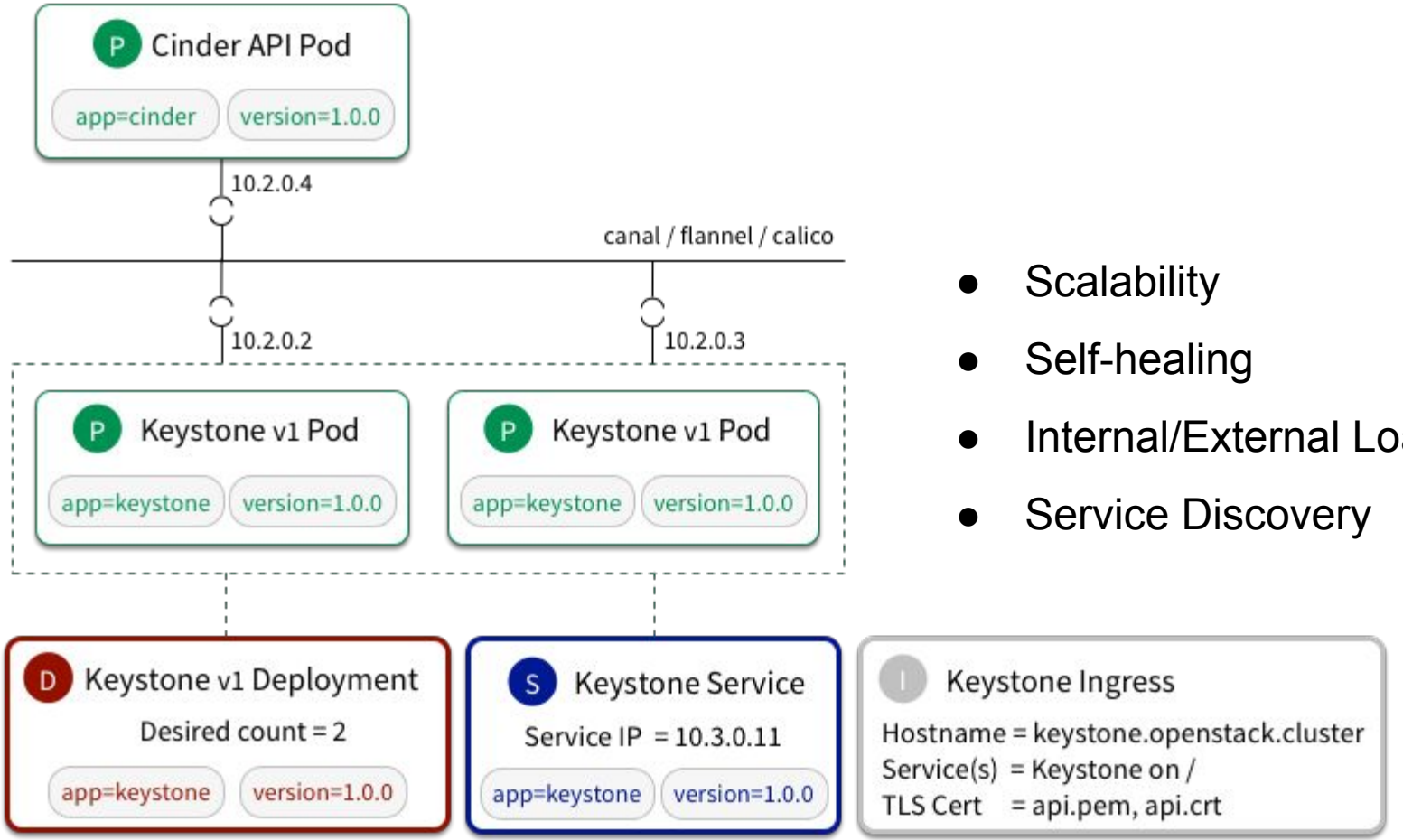


Contacting an application

Service discovery



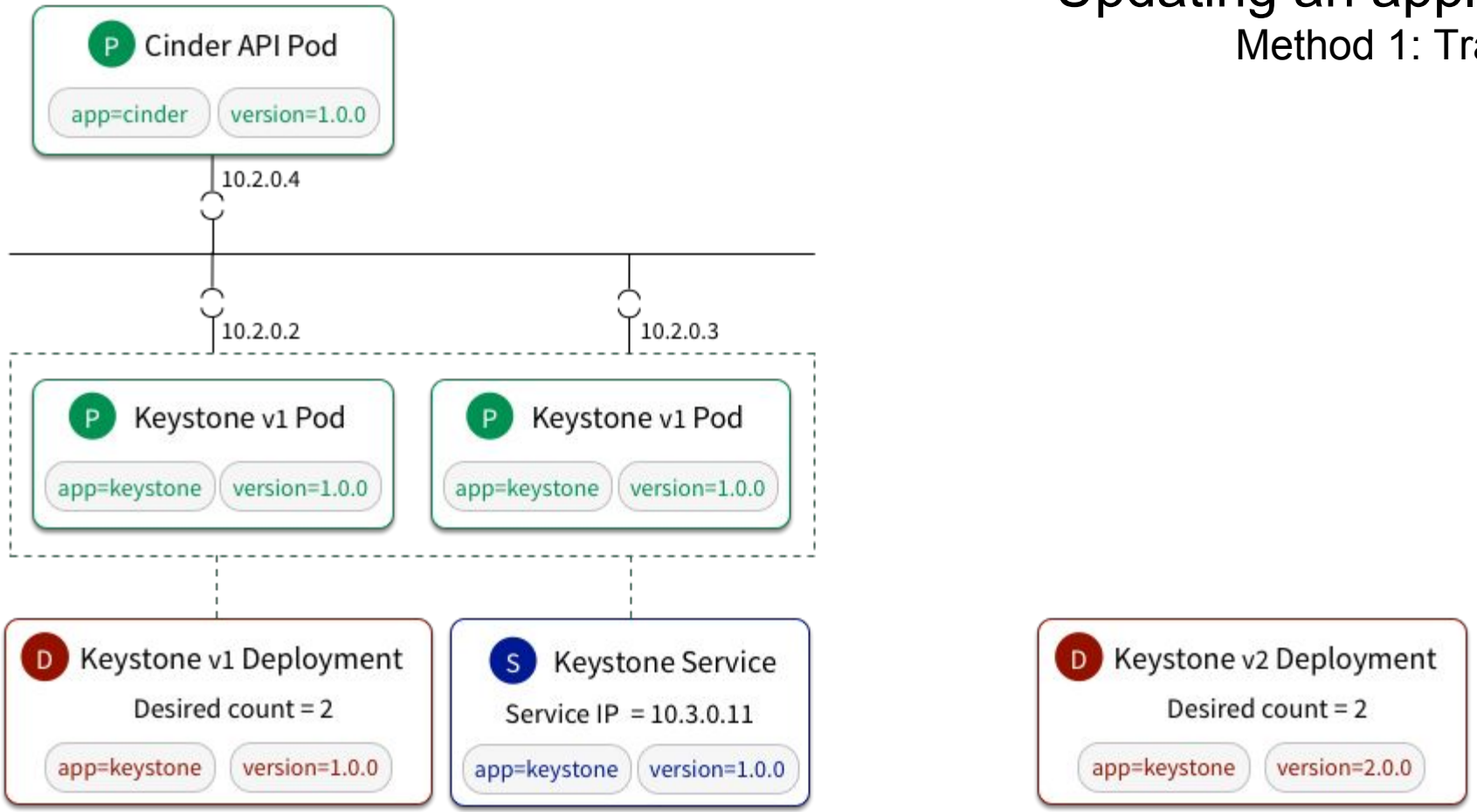
Summary



- Scalability
- Self-healing
- Internal/External Load-Balancing
- Service Discovery

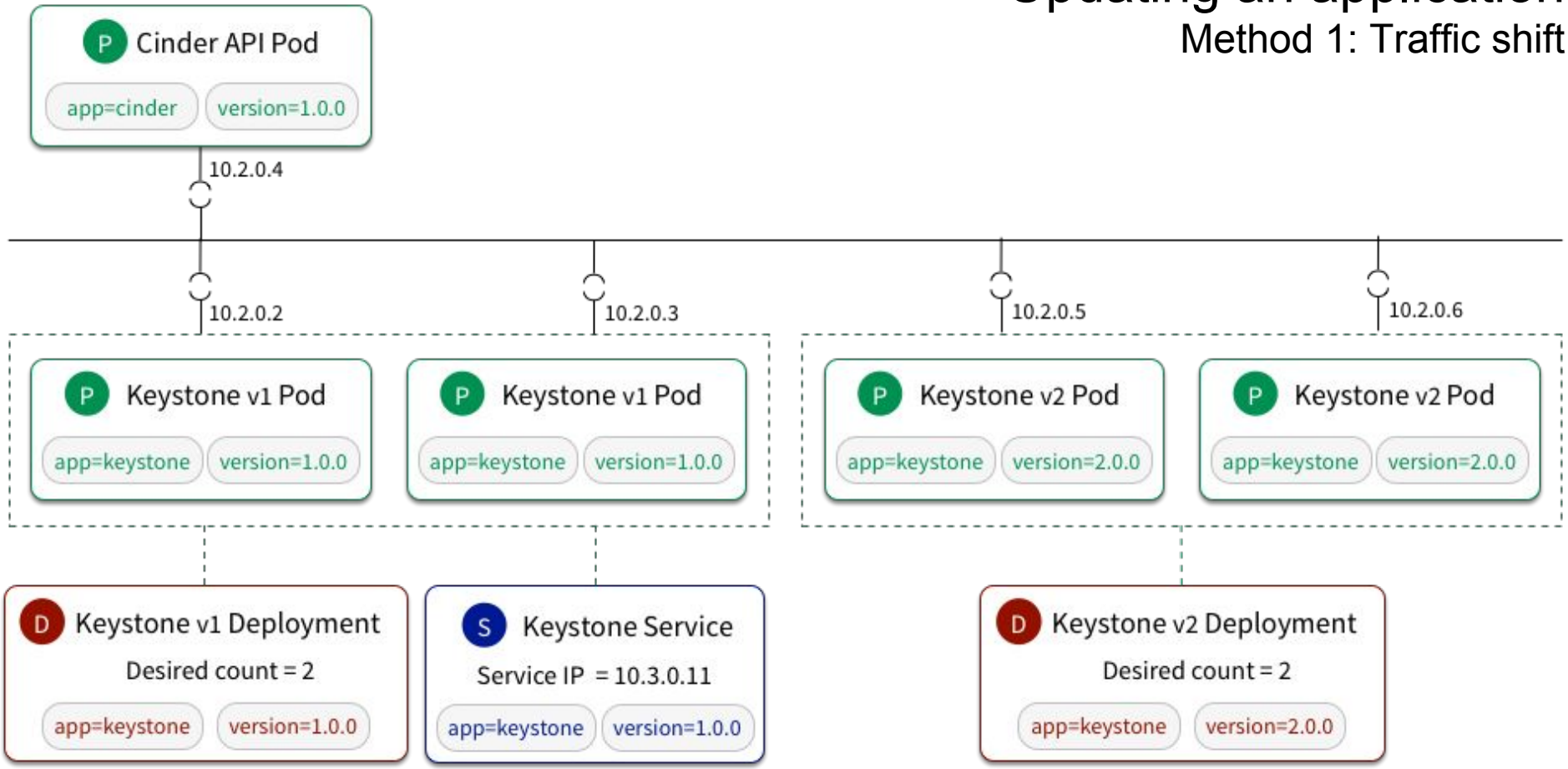
Updating an application

Method 1: Traffic shift



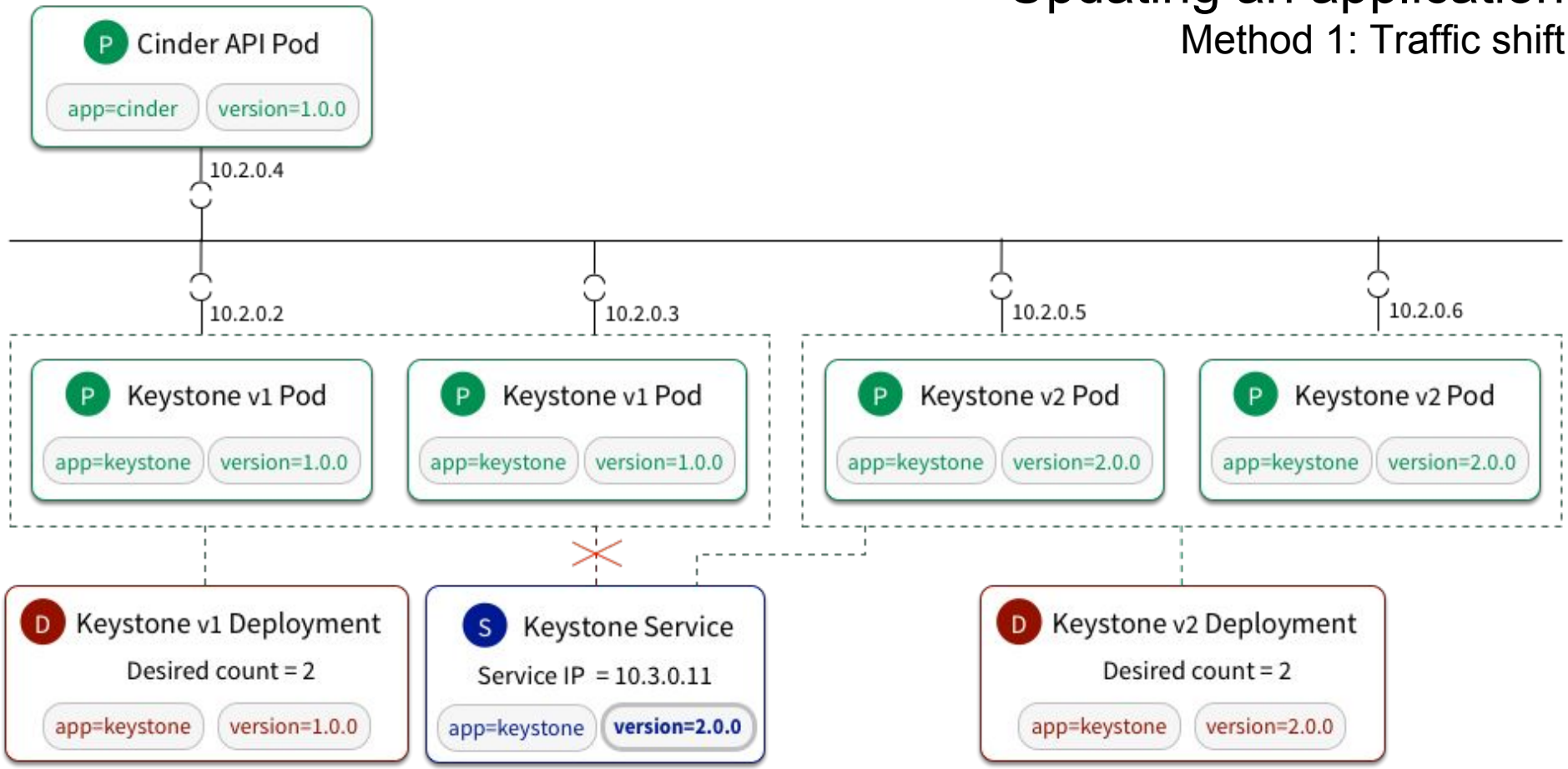
Updating an application

Method 1: Traffic shift



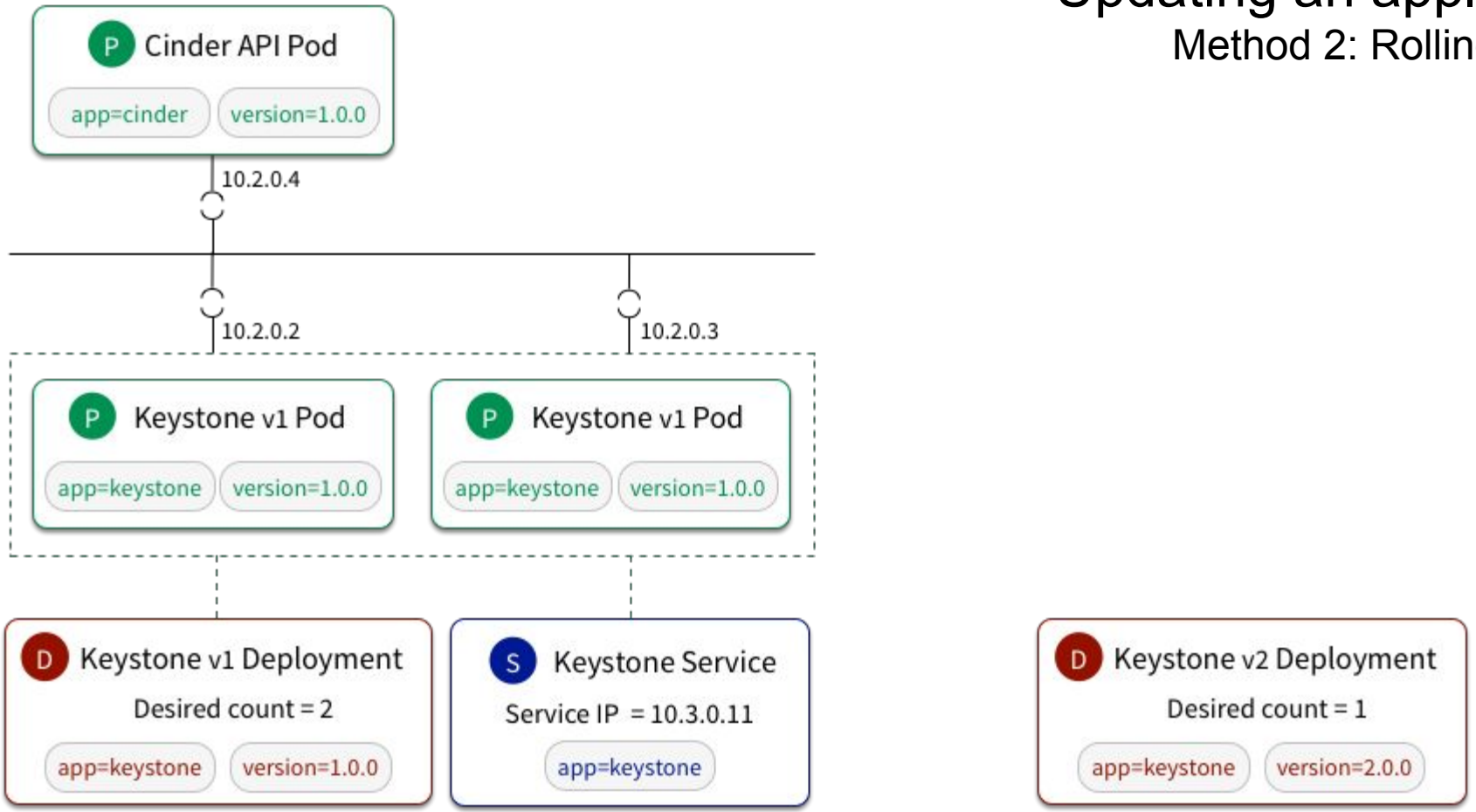
Updating an application

Method 1: Traffic shift



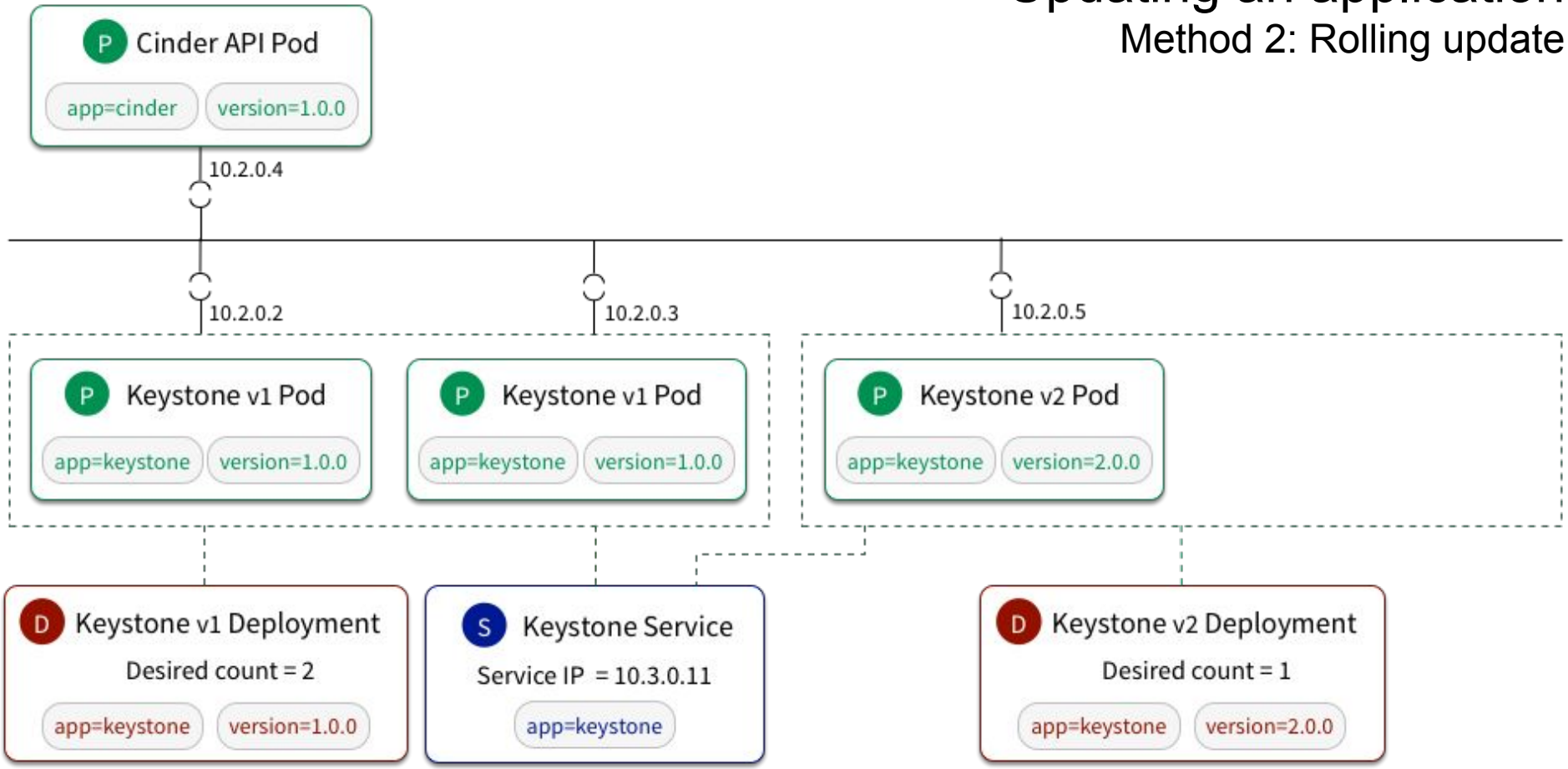
Updating an application

Method 2: Rolling update



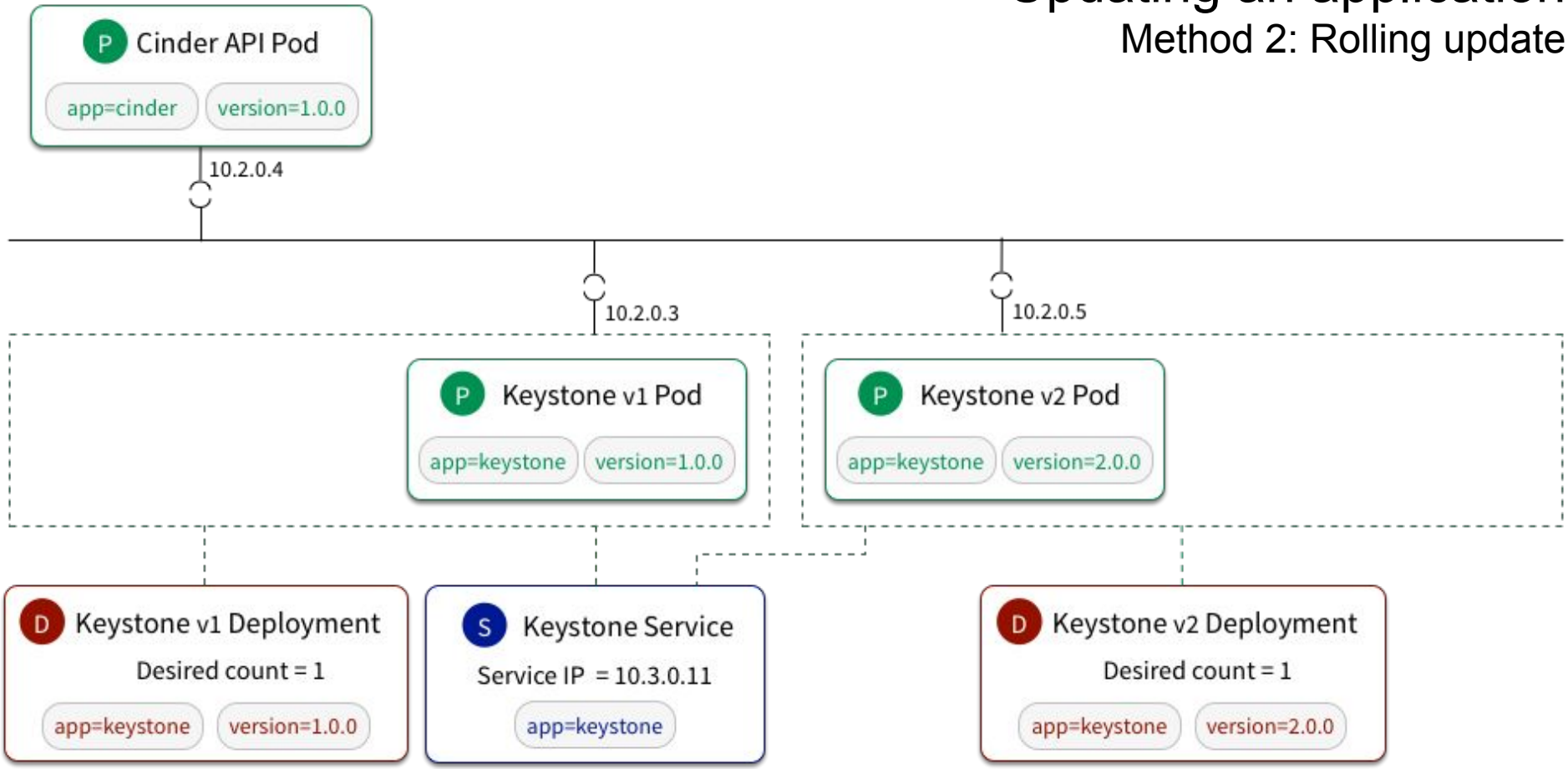
Updating an application

Method 2: Rolling update



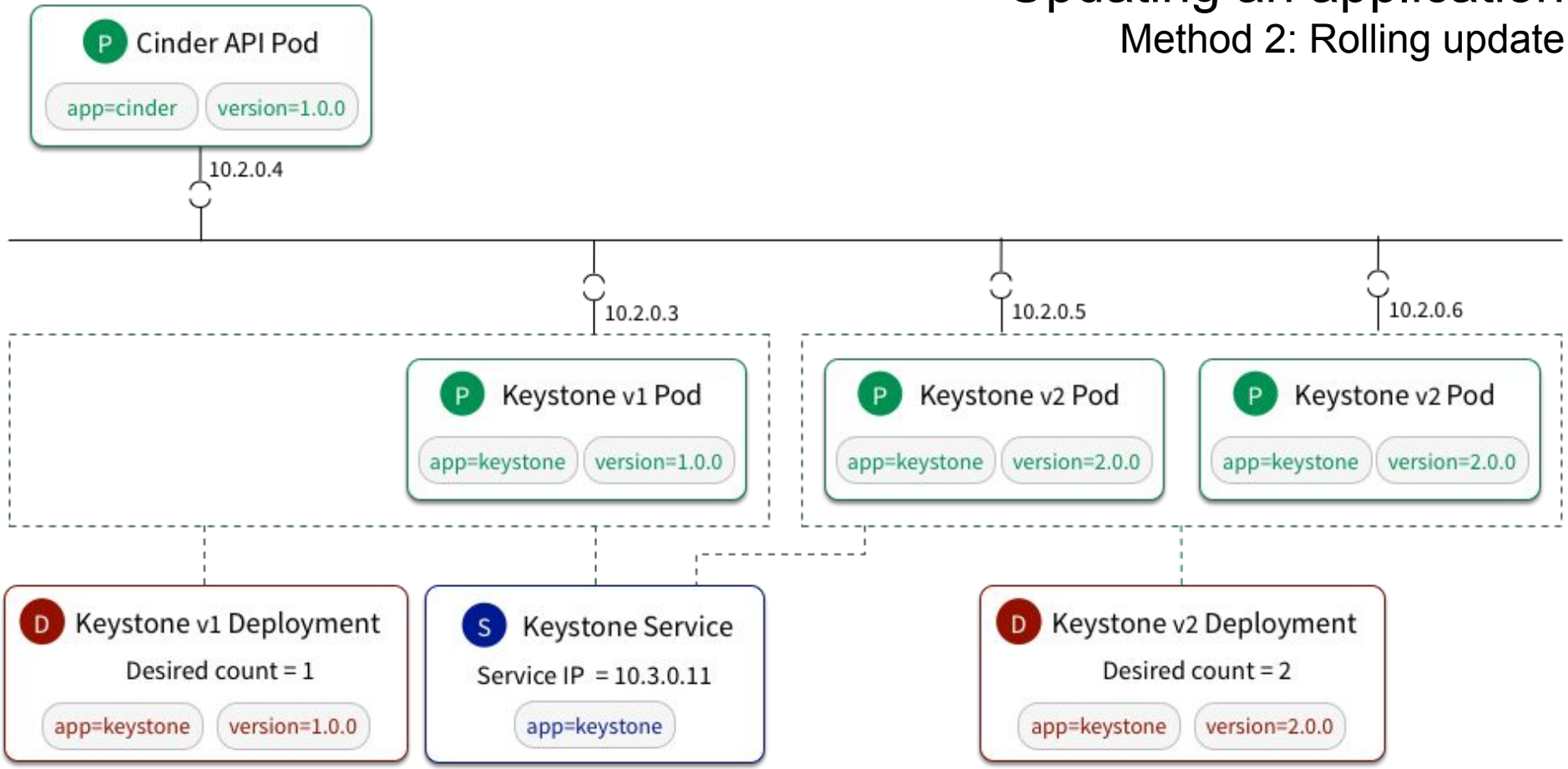
Updating an application

Method 2: Rolling update



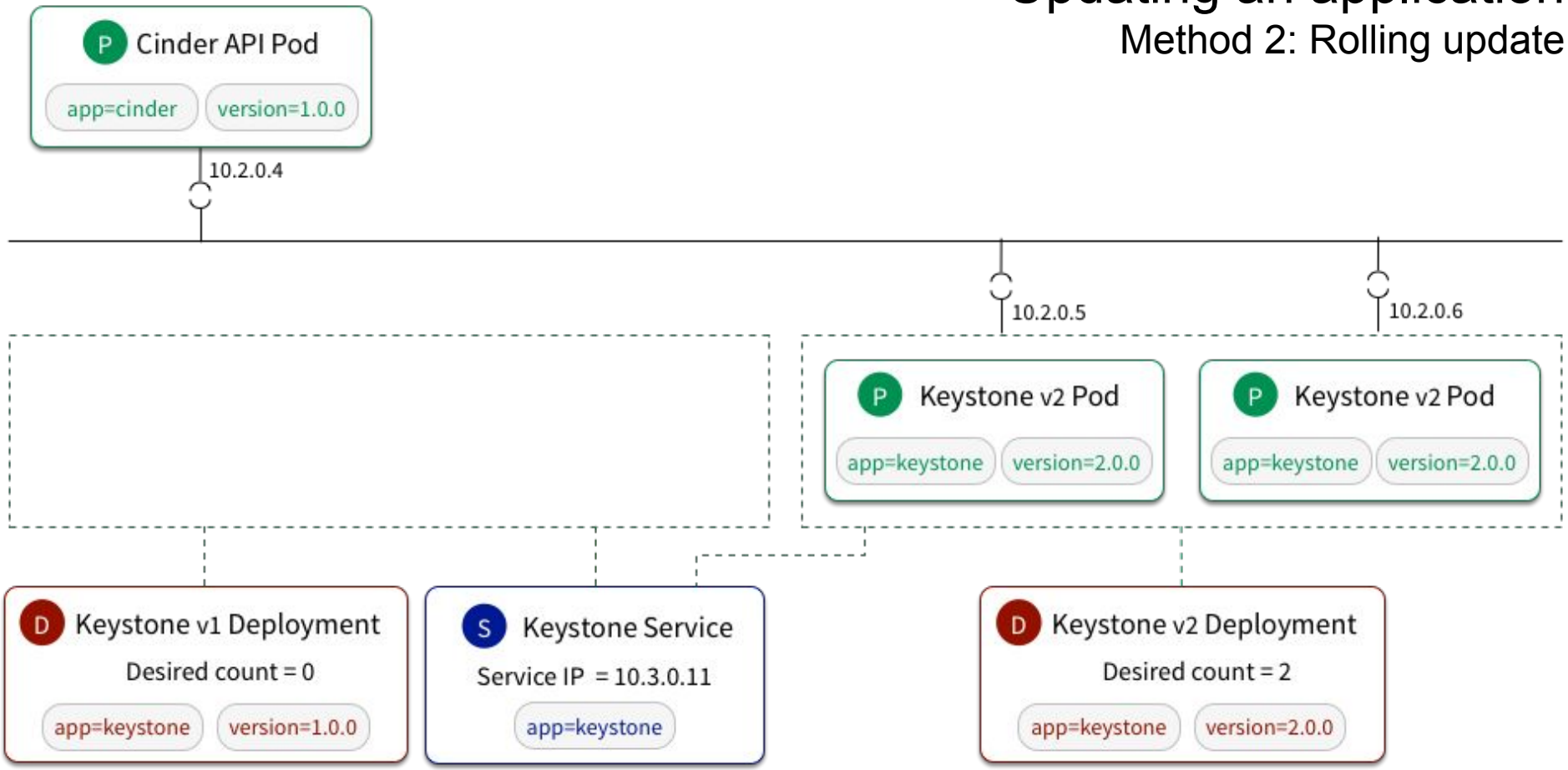
Updating an application

Method 2: Rolling update



Updating an application

Method 2: Rolling update



File templating

- configMap as a native Kubernetes object
- Only root can access configMap files on k8s < 1.3
- Configmap shadow entire mount directory on k8s < 1.4 (subPath)
- Configmap was not writable

Kubernetes Jobs

- Implementation of batch jobs in Kubernetes world
- Perfect match for oneshoot actions:
  - Creating DB
  - Making users, projects, roles
  - Registering Endpoints
- States of jobs are kept in Kubernetes itself rather than in external component

Dependencies management in Kubernetes

- Installation of OpenStack is not a trivial task
  - One has to follow a certain order of deploying services
  - Register endpoints, create databases etc.
- Looks challenging on Kubernetes
  - Dynamic hostnames and IPs
  - Designed for stateless applications with no native support for inter-pod dependencies

Dependencies management in Kubernetes

- We made OpenStack services self-aware of their dependencies:
- Each pod ensures its dependencies are resolved before starting exact application at a container-level:
  - Defined as the container's entrypoint
- Kubernetes-entrypoint:
  - Talks directly to the Kubernetes API to check the state of the dependencies
  - Two ways of using entrypoint:
    - as init-container (introduced in Kubernetes 1.3)
    - inside proper pod

Nova-kubernetes-drain

- Making OpenStack aware of Kubernetes deployment
- Operating Kubernetes cluster without treating OpenStack as a special workload
- Running kubectl cordon will disable compute-node
- Running kubectl drain is triggering auto-evacuation
- Implemented as lifecycle hooks

Inter-pod affinity and anti-affinity

- Allow you to constrain which nodes your pod is eligible to schedule on based on labels of pods that are already running on the node rather than based on labels on nodes.
- My pod should (or should not in case of anti-affinity) run only when pod with specific label is already running on that node
- Thanks to anti-affinity we were able to replace daemonsets with deployments (upgradeable)

Nova-compute instances alongside Kubernetes workloads

- Ability to place containers and VMs on the same physical nodes
- Implementing new driver in Nova:
  - K8sLibvirtDriver
- Compute node reports resources specified in Kubernetes deployment manifest:
  - Memory limits
  - Cpu limits

OpenStack enhancements

- Cinder HA
- Neutron VRRP HA split brain improvements
- Nova live migration without DNS
- oslo.config reading from etcd, configmaps - WIP

Fuel-CCP

- Not using Kolla images
  - Image builder integrated with CI/CD pipeline
- Node-affinity / labels
- Calico as a default network backend
- Custom installer specific for this solution

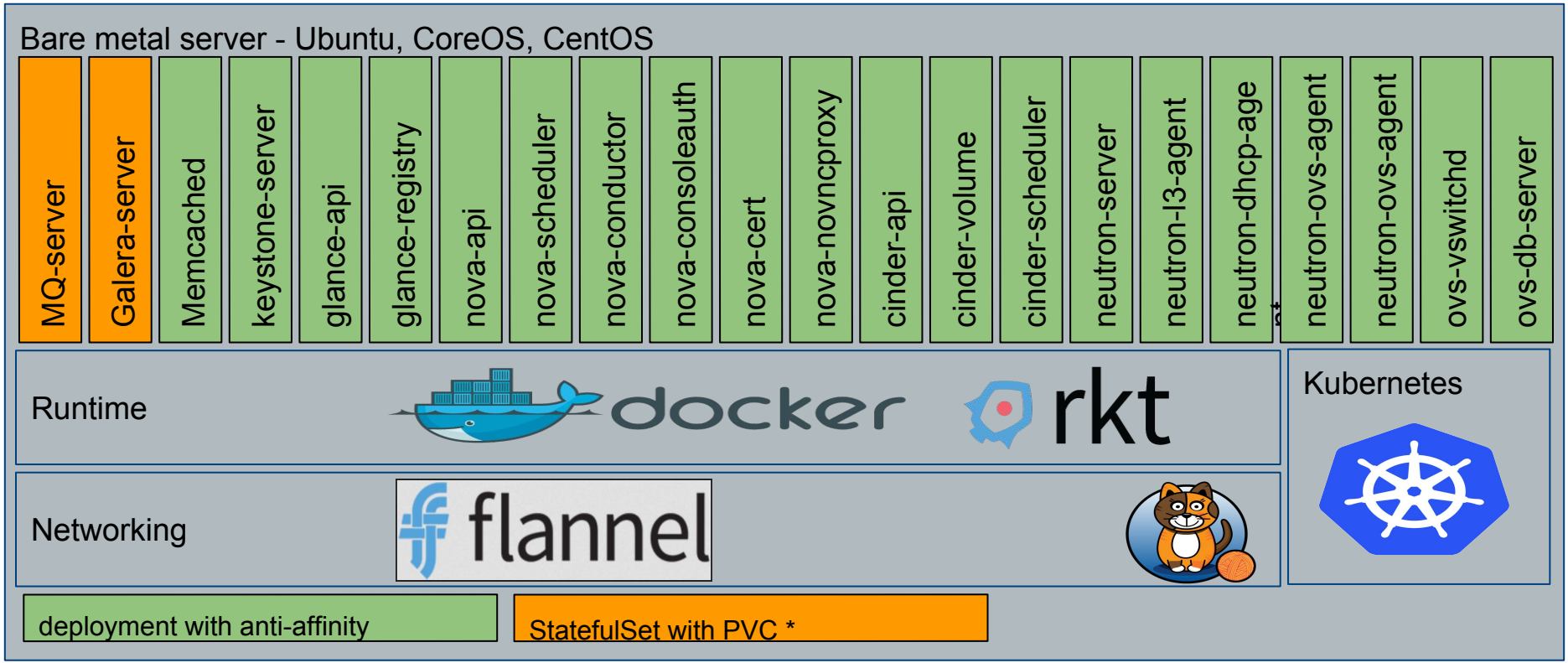
Helm

- Kubernetes packet manager
- Official Kubernetes tool
- Charts are packages of pre-configured Kubernetes resources
- Create reproducible builds of your Kubernetes applications
- Intelligently manage your Kubernetes manifest files
- Share your own applications as Kubernetes charts
- Manage releases of Helm packages

Kolla-kubernetes

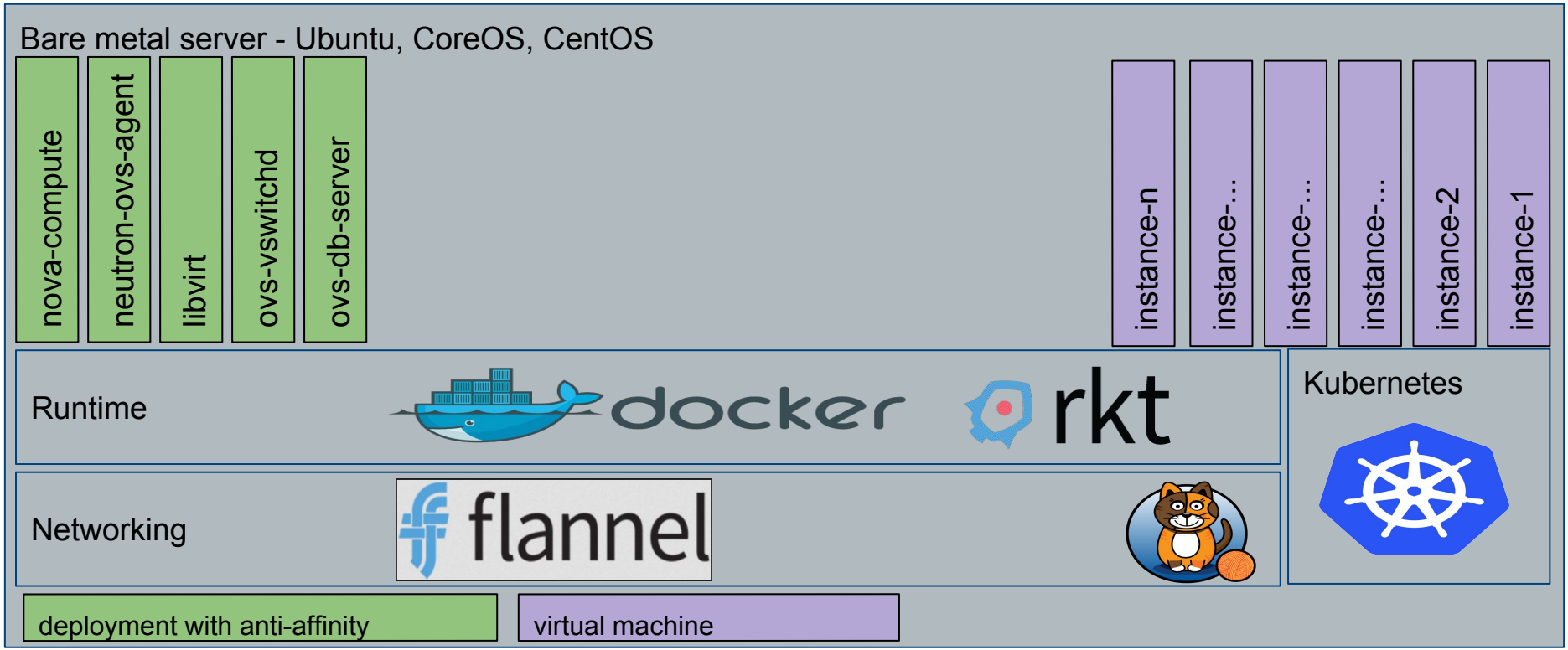
- Driven by OpenStack foundation
- Helm used as an installer
- Kolla-images + kubernetes-entrypoint in init-containers
- Nested templating

Reference Design - control plane



*PVC - persistent volume claim

Reference Design - compute node



OpenStack-Helm

- Heavily based on Stackanetes
- Kolla-images + kubernetes-entrypoint in init-containers
- Targeting image agnostic model (ie: loci images)
- Release 0.1.0
    - All Stackanetes services + MaaS (Ubuntu backed bare metal provisioning)

Kubernetes on OpenStack

- Kubernetes multitenancy still requires underlay cloud
- Not all workloads are suitable for containers (ie: VDI)
- OpenStack and Kubernetes self awareness, to maximize infrastructure utilization (VM vs container workloads collocation)
- Kubernetes federation fits into hybrid cloud requirements
- Easy deployment with magnum, or kube-now

SWEET! LET'S SUMMARIZE

Kubernetes and OpenStack

- Always use right solutions which work best for particular use-case
- To maximize infrastructure utilization collocate containers and VMs
- Hyperscale your datacenter

Legal Notice and disclaimers

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.
- No computer system can be absolutely secure.
- Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.
- Intel, the Intel logo and others are trademarks of Intel Corporation in the U.S. and/or other countries. *Other names and brands may be claimed as the property of others.

Q&A