KubeCon

North America 2017

decco: Secure, Multitenant Cluster for Internet Facing Services

Bich Le, Chief Architect @Platform9

Background & Motivation

PLATFORM9

deploy, manage & maintain { kubernetes | openstack | fission } as a service

on the infrastructure of your choice

Public Cloud ⟷ Private Cloud

amazon web services   Microsoft Azure   Google Cloud Platform   vmware   Bare Metal

Objectives

- Run control plane services on shared Kubernetes cluster(s)
- Secure enough to host multiple customers
- Simplify Kubernetes without hiding its power and API
- Useful to community

Deployement Cluster Config & Ops

- Public repo: https://github.com/platform9/decco
- Follows the Kubernetes controller/operator pattern
- Runs inside or outside cluster
- Automates provisioning and teardown of
  - DNS records
  - K8s resources: namespaces, deployments, services, ingress, netpolicies
- Introduces 2 custom resources: Space, App

DNS conventions for service endpoints

For hypothetical customer “foo”

- Customer-wide fqdn: foo.platform9.net
- Region-specific fqdn: foo-regionname.platform9.net
- Services using reverse proxy (HTTP only): foo-regionname.platform9.net/servicename
- Services not using reverse proxy (TCP and HTTP): servicename.foo-regionname.platform9.net

Network security

General observations about control plane services:

- TLS encryption for all traffic
- HTTP services
  - don’t use client certificate authentication
  - use app-level authentication (Keystone)
- TCP services
  - use client certificate auth
  - optionally use app-level auth (e.g. rabbitmq / mysql password)

Network security (cont'd)

Internally, decco secures the network using a combo of

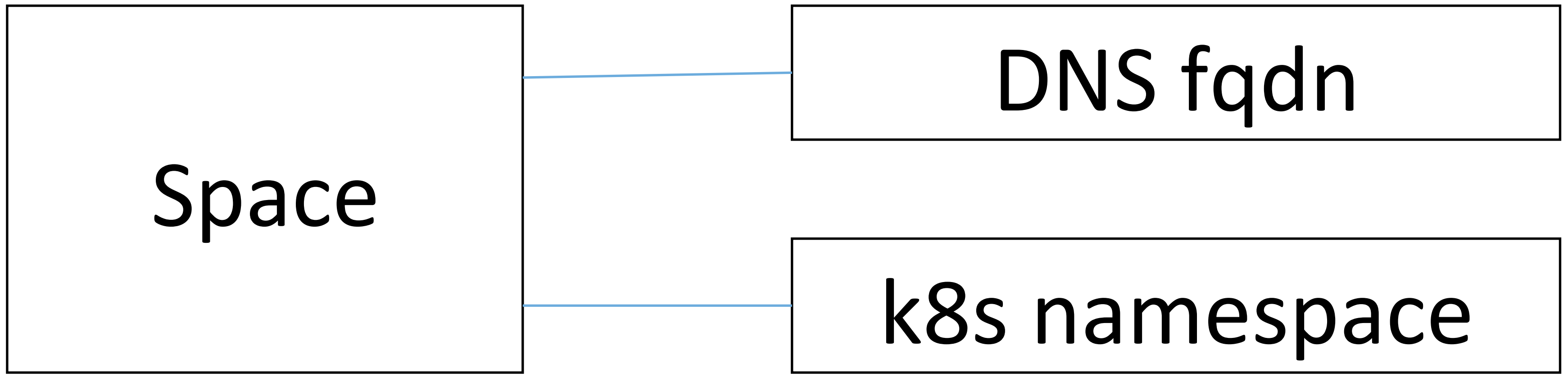
- TLS
  - encryption
  - mutual certificate verification
- Network Policy resources

decco concepts

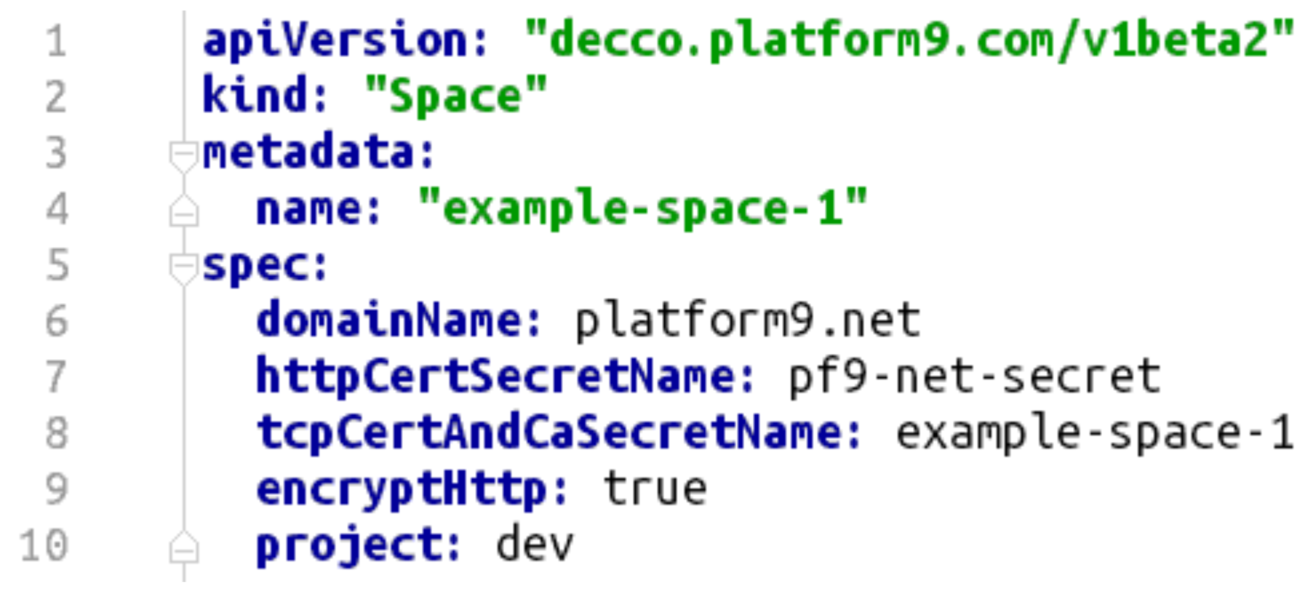
- Space
- Project
- App

Space

- A naming and isolation boundary for a related set of Internet-facing services



Space – example yaml



```yaml
apiVersion: "decco.platform9.com/v1beta2"
kind: "Space"
metadata:
  name: "example-space-1"
spec:
  domainName: platform9.net
  httpCertSecretName: pf9-net-secret
  tcpCertAndCaSecretName: example-space-1
  encryptHttp: true
  project: dev
```

Space – what happens upon creation

Decco creates:

- Namespace
- DNS record
- Optional network policy resource (see *Project* slide)
- Resources needed for network routing and TLS processing

DNS update details

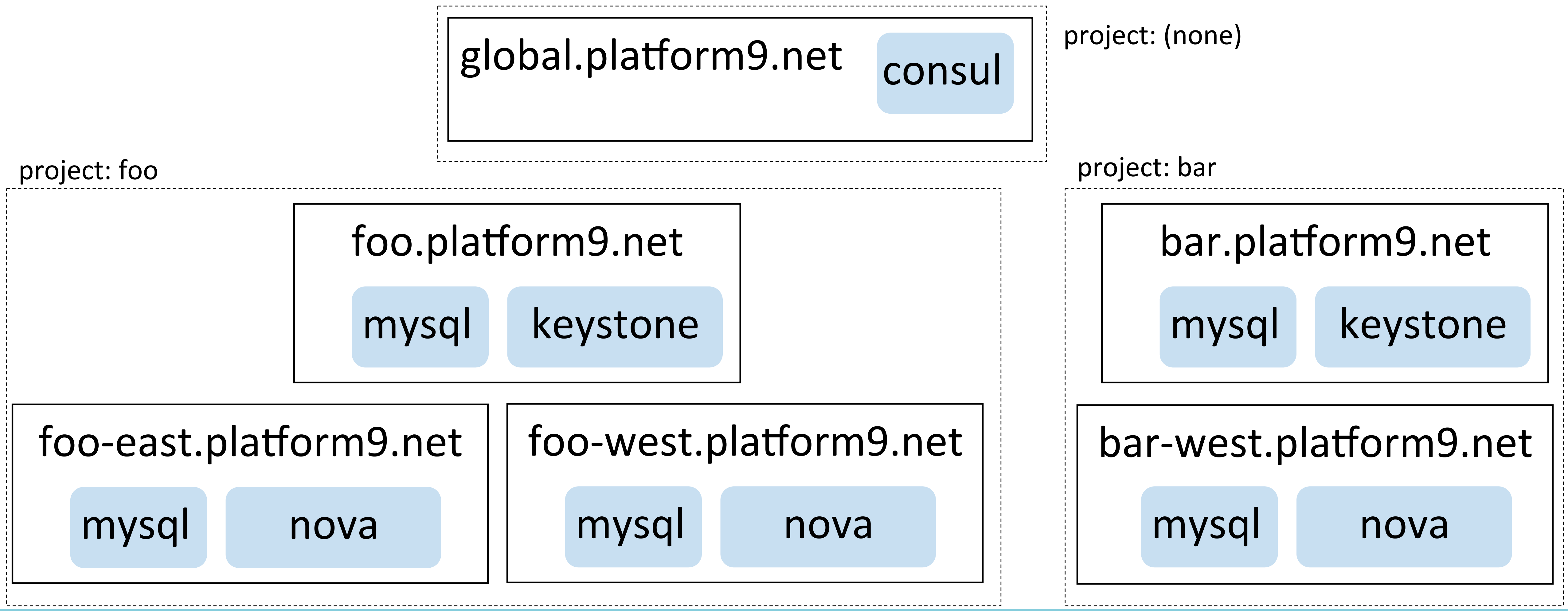
Decco uses same DNS library as Kubernetes Federation

1. Compute FQDN
   - Space: ${spaceName}.${domainName}
   - TCP Service: ${svcName}.${spaceName}.${domainName}
2. Lookup external endpoint (IP/fqdn) of cluster’s ingress service
3. Using configured DNS provider
   - Lookup hosted zone corresponding to the space’s domain name
   - If found, create A or CNAME record for the FQDN in the hosted zone, otherwise fail

Project

- An optional property to restrict incoming connections to those from spaces with the same project name
- Internally enforced using a Network Policy with namespaceSelector
- Tested on GKE “Alpha” 1.8.x cluster with Calico network plugin

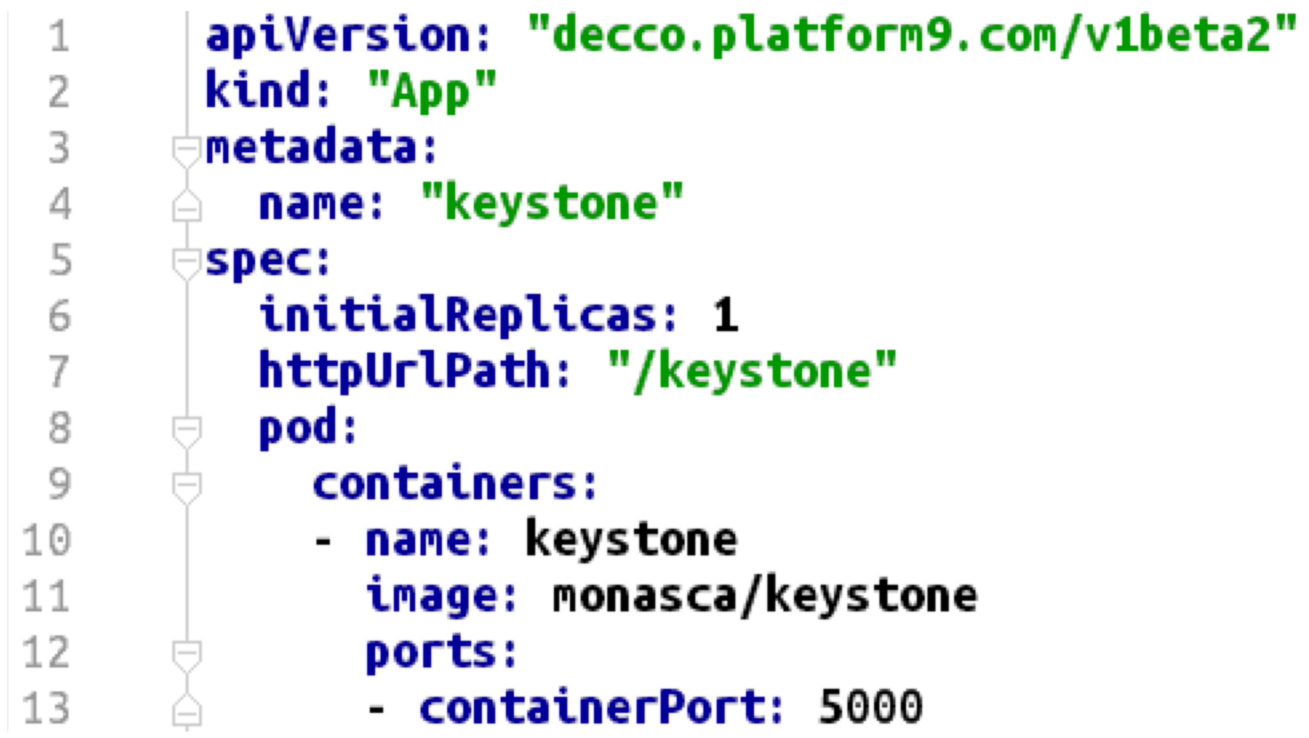
Example


global.platform9.net
consul
project: (none)
project: foo
foo.platform9.net
mysql
keystone
foo-east.platform9.net
mysql
nova
foo-west.platform9.net
mysql
nova
project: bar
bar.platform9.net
mysql
keystone
bar-west.platform9.net
mysql
nova

App

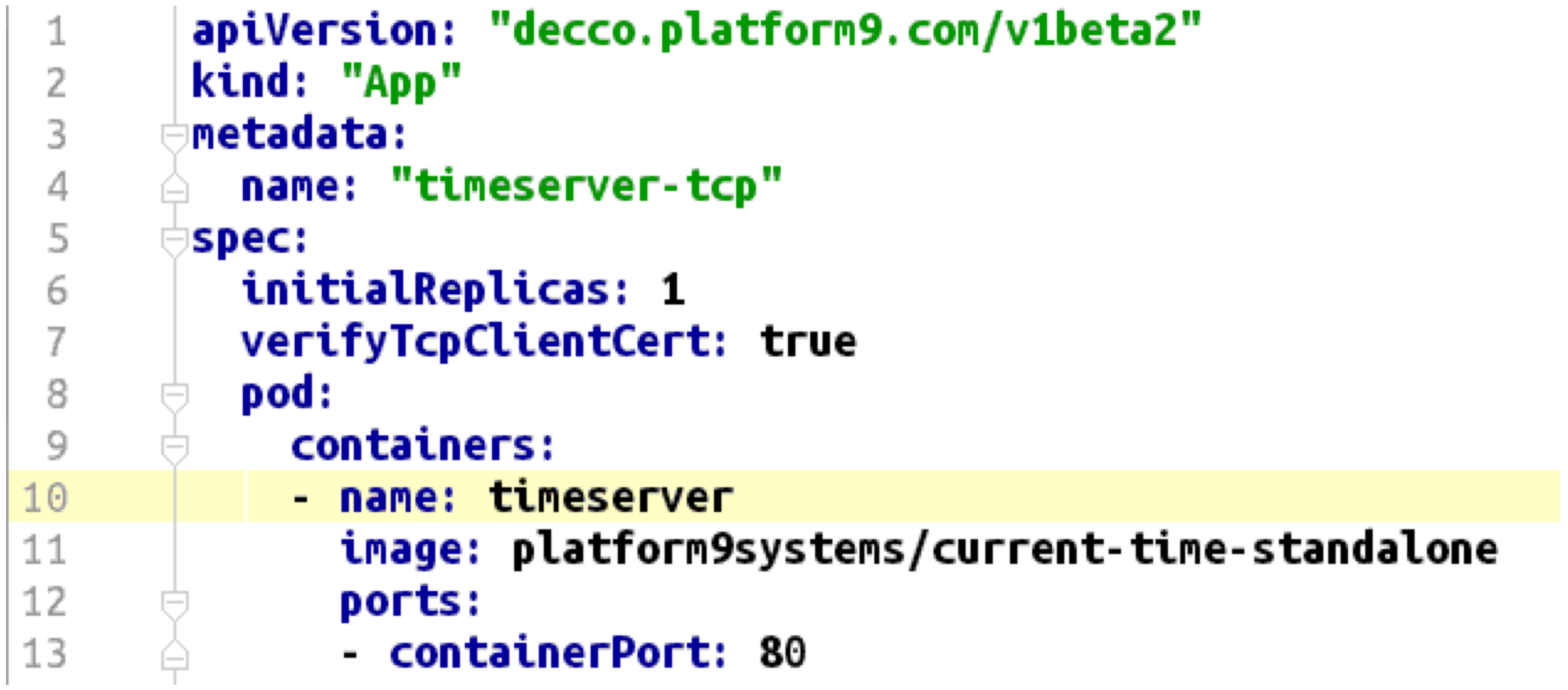
- A thin wrapper around a PodSpec, specifying a service listening on a given port (*Planned: support for multiple service ports)*
- Service can be externally exposed via
  - reverse proxy (httpUrlPath not empty)
  - direct TCP (httpUrlPath empty)
- TCP specific options
  - tlsCaAndCertSecretName (use space default secret if not specified)
  - verifyTcpClientCert
  - createDnsRecord
- Other options: initialReplicas, tlsEgresses

App – http example



```yaml
apiVersion: "decco.platform9.com/v1beta2"
kind: "App"
metadata:
  name: "keystone"
spec:
  initialReplicas: 1
  httpUrlPath: "/keystone"
  pod:
    containers:
    - name: keystone
      image: monasca/keystone
      ports:
      - containerPort: 5000
```

App – tcp example



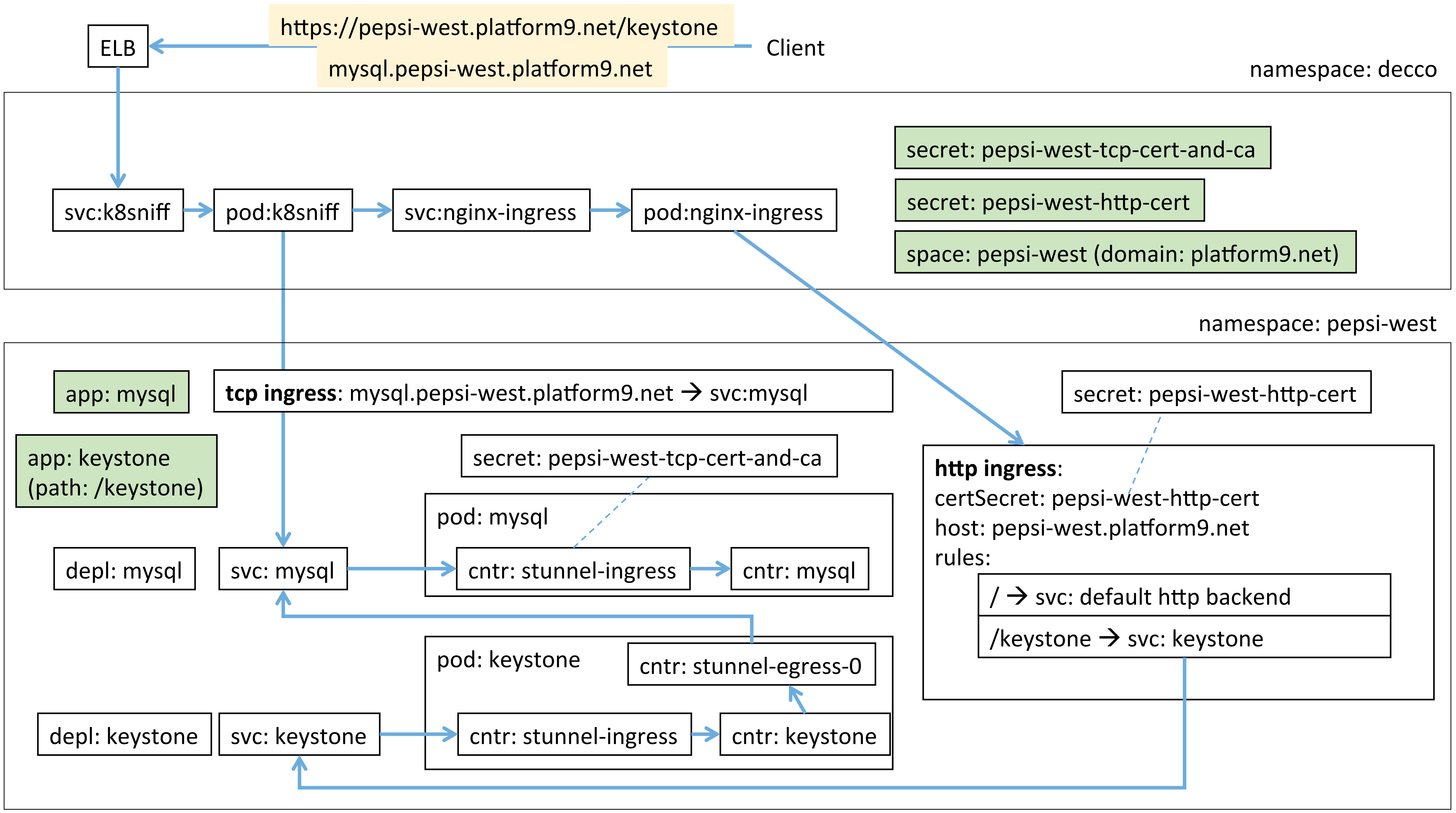
```yaml
apiVersion: "decco.platform9.com/v1beta2"
kind: "App"
metadata:
  name: "timeserver-tcp"
spec:
  initialReplicas: 1
  verifyTcpClientCert: true
  pod:
    containers:
    - name: timeserver
      image: platform9systems/current-time-standalone
      ports:
      - containerPort: 80
```

App – what happens at creation time

- Create service resource(s)
- Create DNS record if createDnsRecord is true (example fqdn: `myservice.myspace.platform9.net`)
- Create resources for network routing and TLS
  - Ingress resources / path rules
  - Service mesh rules
- Create deployment resource with modified pod spec
  - Inject side cars needed for network routing

Network routing and TLS

- Planned: service mesh integration (e.g. Istio)
- Out of the box: cascaded ingress controllers
  - k8sniff inspects SNI headers and routes based on SNI server name
  - If no match, forwards to nginx ingress controller
  - Decco manages ingress resources for both (TCP and HTTP)
  - TLS
    - Certs need to be externally signed and supplied via secrets
    - Stunnel side cars



https://pepsi-west.platform9.net/keystone
mysql.pepsi-west.platform9.net
Client
ELB
namespace: decco
svc:k8sniff
pod:k8sniff
svc:nginx-ingress
pod:nginx-ingress
secret: pepsi-west-tcp-cert-and-ca
secret: pepsi-west-http-cert
space: pepsi-west (domain: platform9.net)
namespace: pepsi-west
app: mysql
app: keystone (path: /keystone)
tcp ingress: mysql.pepsi-west.platform9.net → svc:mysql
secret: pepsi-west-tcp-cert-and-ca
pod: mysql
depl: mysql
svc: mysql
cntr: stunnel-ingress
cntr: mysql
pod: keystone
cntr: stunnel-egress-0
depl: keystone
svc: keystone
cntr: stunnel-ingress
cntr: keystone
secret: pepsi-west-http-cert
http ingress:
certSecret: pepsi-west-http-cert
host: pepsi-west.platform9.net
rules:
/ → svc: default http backend
/keystone → svc: keystone

Roadmap

- Service mesh integration
- Integration with Availability Zones and Kubernetes Federation
- Support multiple service endpoints per app
- Open source project
  - Docs
  - Automated test
- Explore better container isolation via VM-based solutions (e.g. Frakti / runV)

Demo and Thanks!

- For more info
  - https://github.com/platform9/decco
  - http://www.platform9.com
- Other talks
  - Fission (serverless on Kubernetes) (Thursday 2:45pm)
  - Cost-effective Compute Clusters using Spot and Preemptible Instances (Friday 2pm)