Kubernetes in Rakuten CPD

Sept 26th, 2020

Kejun Huang

Cloud Platform Deparment

Rakuten, Inc.

Rakuten


Rakuten
Rakuten
R

About Rakuten

- One of largest Internet serivces companies in Japan. Officifal sponser of FC Barcelona.
  - E-Commerce
  - Rakuten Point
  - Fintech (Credit card, bank, payment, etc)
  - Travel
  - Many many others…

Self introduction

- Vice Manager of Application Platform Group, which manages the Kubernetes cluster in Cloud Platform Department(CPD)
- Responsible for traffic/network/mesh part
- Join Rakuten in 2015
- Start the career as a Python engineer in douban.com, China.

Team

- 1 PJM, 12 Engineers(Tokyo).
- 7 Nationalities, English is the primary communication language
- All Engineers are CKA/CKAD holders
- Most tools and software are written in Go.

Agenda

1. Private cloud initiative
2. Multi tenancy
3. Network and Traffic
4. CI/CD
5. Improving user experience
6. Challenges and future work

Private Cloud initiative

- Each deparment or team has their way of running services
- Private Cloud is intended to provide a standard way for internal teams to build, deploy services into multiple regions.
  - Avoid duplicate work
  - Improve engineers productivity
  - Provide better resillience during diasters (BCP)

Background

- We call our private cloud OneCloud, which we are building across multiple regions
- OneCloud is mostly built on top of Open Source solutions.
- Most of products in OneCloud run on baremetal
  - Baremetal is based on miroservers that are speficilly built for OneCloud
  - We call it Baremetal as a service, BMaaS
- OneCloud also has an in-house developed UI, called OneCloud portal
- We have abstracted common systems into services and provide to our internal tenants
  - Container as a service based on k8s (my team)
  - Monitoring as a service
  - Load balancing as a service
  - etc

Why Kubernetes

- Already have the team who has build a successful k8s platform.
  - We started to use k8s in 2014
- Kubernetes has great community support
- Many other Cloud Native products on top of k8s
  - Istio
  - Prometheus

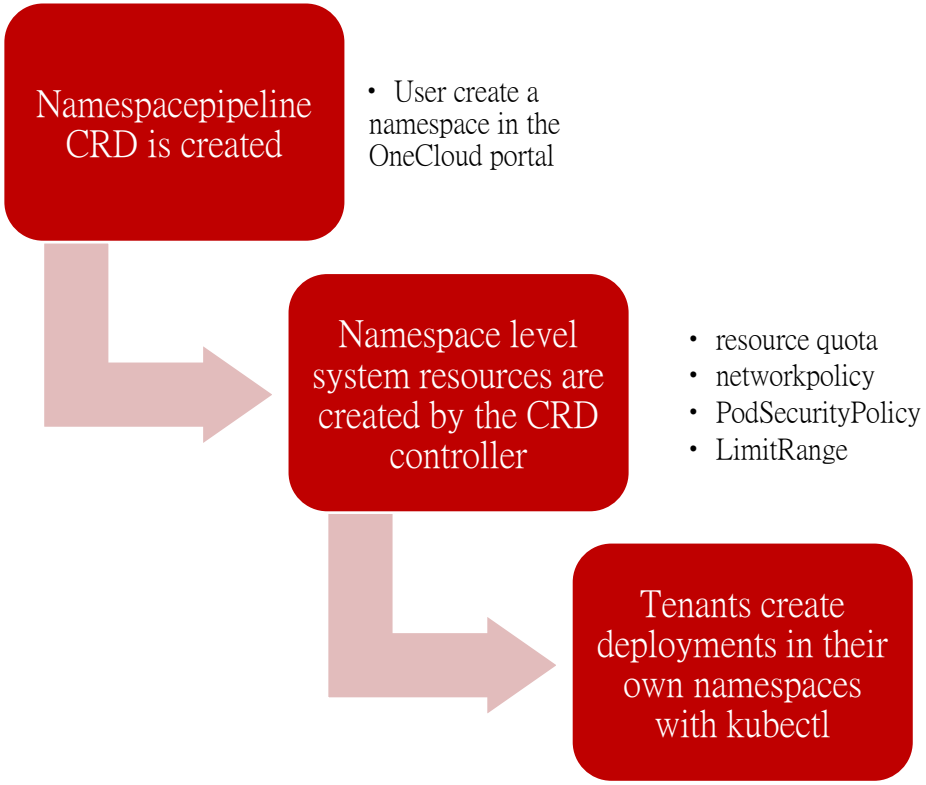
Our Current k8s setup

- Uses community version
- A Physical k8s cluster is shared by multiple tenants
  - Inside one region, we have multiple k8s clusters
- Runs on on-premise baremetal hardware
- provisioned using our internal tooling
- Cluster itself is stateless, does not support PV. Tenants are mostly web services.
- Full self-service and multi-tenant cluster integrated with internal IAM system and internal portal UI.
- Extending the k8s with other Cloud Native products

Why big cluster

- Standardise policy design of k8s
  - RBAC
  - Security
- Reduce cost of managing multiple k8s clusters
- Improve server utilisation
  - Some workloads do not require a k8s cluster to run
  - API servers require redundancy

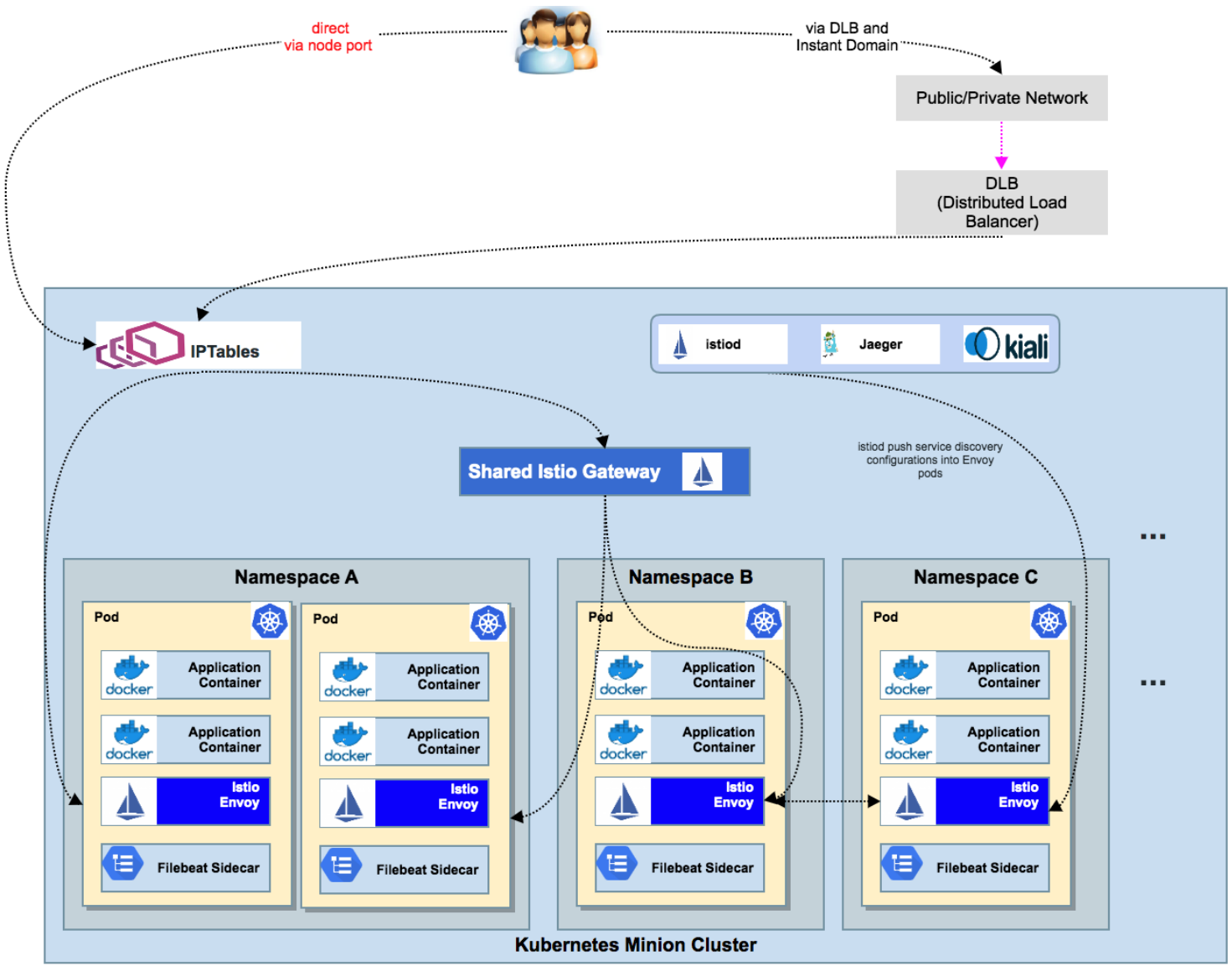
Multi tenancy in Kubernetes in CPD



We use logical separation(namespaces) to provide multi tenancy

- One namespace will not use all the resources in the cluster (ResourceQuota)
- One deployment will not use all the resources on one machine(LimitRange)
- Pods cannot be run as root (PSP)
- Pods by default cannot receive ingress and send egress traffic from inside or outside of the cluster except in-house system namespaces. (Netpol)
- Tenant can choose to change above default policies if it’ s required.

Networking and Traffic



- DLB is our internal load balancer solution for exposing the service
- We use Istio as our netwoking and mesh solution
  - By default, all tenants got injected with Envoy pods as sidecar container
  - Tenant pods use sidecar Envoy to take ingress and send egress traffic
- By default, all tenants use shared istio-ingressgateway
  - Tenant can choose to provision their own gateway if it’ s required

InstantDomain

Users Create k8s service object

InstantDomain controller creates VirtualService after getting the events

Users can access its workload from outside of the cluster with a configured domain

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: some-instant-domain-rule
spec:
  hosts:
  - some.instantdomain.com
  http:
  - timeout: 5s
    route:
    - destination:
        host: productpage.prod.svc.cluster.local
```

curl some.instantdomain.com

Advantages of using InstantDomain

- Users do not need to setup DNS and HTTPs termination
- Users do not need to learn Istio concepts

In case tenants need

- Custom domain
- Other protocols other than HTTP
- Finer control over their traffic

In-House Cloud Control Manager

Users Create k8s service object

CCM creates the actual load balancer in DLB

Users can access its workload from outside of the cluster.

```yaml
apiVersion: "v1"
kind: "Service"
metadata:
  name: "my-service"
spec:
  type: "LoadBalancer"
  ports:
  - name: http-test
    port: 80
    protocol: "TCP"
    targetPort: 80
  selector:
    mypods: mypods
```

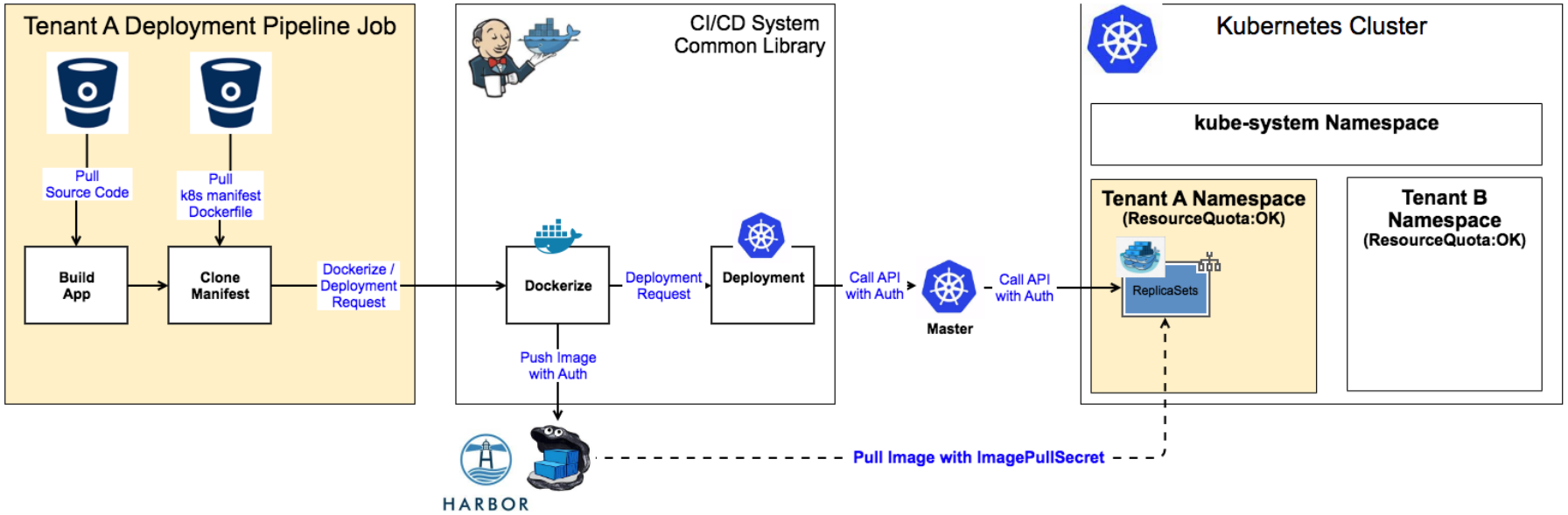
```
$ kubectl get svc

NAME                      TYPE           CLUSTER-IP       EXTERNAL-IP      PORT(S)        AGE
gateway-example-gateway   LoadBalancer   100.102.136.127  [redacted]       80:62399/TCP   16h
```

How does user create resources into our K8s cluster

- OneCloud portal
- kubectl
- CI/CD

CI/CD Pipeline



- Harbor has replication enabled. Images pushed to one registry will be replicated into other regions
- imagePullSecret are set by namespacepipelineCRD
- Kustomize is to use to populate the YAML manifests.

Improving User Experience

- Kubernetes is complicated
- Kubernetes + Istio is even more complicated
- Kubernetes + Istio + Jenkins + Tekton + Harbor + Flux + Prometheus + ⋯ = Explosion

What we are trying to do

1. Providing good defaults
2. Avoid duplicate efforts

Improving UX 1: Knative


Users create ksvc object
Knative creates service, virtualservices and other resources
Users workloads are deployed and accessible.
Autoscaler kicks in when it’ s required

Improving UX 2: webhooks

A WebHook is an HTTP callback: an HTTP POST that occurs when something happens; a simple event-notification via HTTP POST. A web application implementing WebHooks will POST a message to a URL when certain things happen.

- ValidatingWebhook for doing extra validation work for tenants.
- MutatingWebhook for injecting some common logic
  - Sidecar containers, e.g. filebeat container
  - preStop hook

Improving UX 3: roc-cli

```
➜ ~ roc login -c jpw1-caas1-lab1 -n istio-system
I0914 16:49:40.170277   11608 login.go:41] access to Rakuten One Cloud IAM
I0914 16:49:41.157192   11608 oidc.go:367] user "Kejun Huang" is authenticated by Rakuten One Cloud IAM
I0914 16:49:41.430108   11608 login.go:90] cluster-id specified. will try to authorize by CaaS
I0914 16:49:41.876044   11608 login.go:119] succeeded to validate token via this cluster
I0914 16:49:41.899329   11608 login.go:136] successfully authorized by CaaS cluster: jpw1-caas1-lab1
now you can access CaaS cluster (jpw1-caas1-lab1) by kubectl
if you don't have kubectl, you can get with 'roc get kubectl'
you can change your context with 'roc get ctx' and 'roc use ctx <context number>'
➜ ~
```

- Use k8s with OpenIDConnect for authentication and authorisation
- No need to manually configure kubeconfig
- Switch context between clusters quickly

Some lesson we have learnt

- Multi-tenanacy and security with k8s is lots of work
- You need to understand the pod lifecycle
  - How to achieve true gracefull rolling restart?
- Istio has a fast release cycle and sometimes is not stable
  - Having your own tests in some lab environments to verify the upgrade
- Read the iptables rules created by kube-proxy
  - If you have a LoadBalancer type service and access this service inside same k8s cluster, k8s will directly forward the traffic to target service

Future work

- Container native load balancing. No iptables!
- K8s upgrade automation
- Knative user adoption
- Integration with Tekton, Flux
- VM in k8s
- GPU and PV support
- Dedicate nodes for tenants
- Abstract common logic into sidecar using WebAssembly or Lua
- …

If you are willing to joining in and work on these topics,
We are hiring!
kejun.huang@rakuten.com
https://japan-job-en.rakuten.careers/search-jobs/cpd/31066/1


Rakuten