openstack.

22.05.2018

Build Your Serverless Container Cloud with OpenStack and Kubernetes

Kevin Zhao
Senior Software Engineer on Arm.
OpenStack Zun Core Reviewer
kevin.zhao@arm.com

openstack.

Agenda

What is Serverless Container Cloud

Zun and Container Capsule

Build the Serverless Cloud

Demo

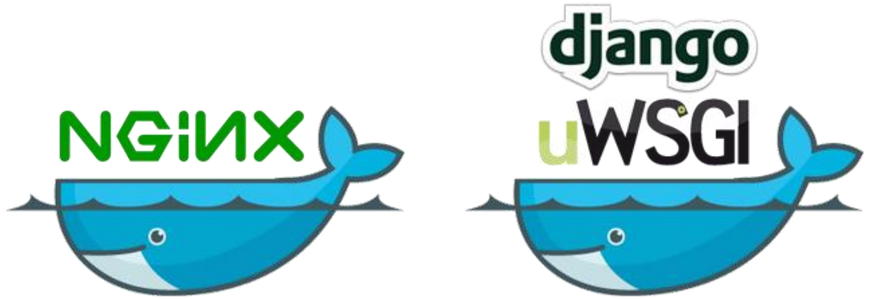
FAQ’s

openstack.

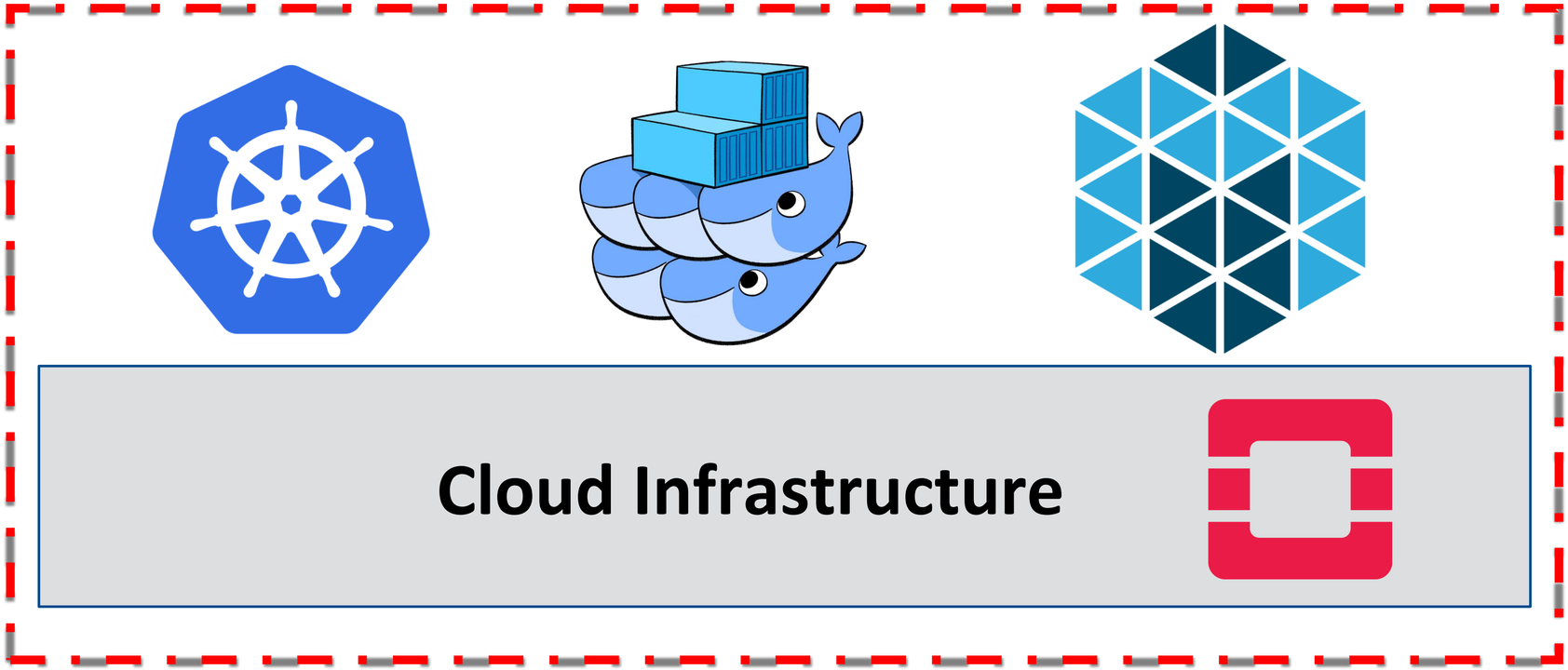
What is Serverless Container Cloud

openstack.

Traditional Container Cloud



- Provision the cluster first, pay much effort in cluster management
- Cluster level multi-tenant isolation
- Low resource utilization

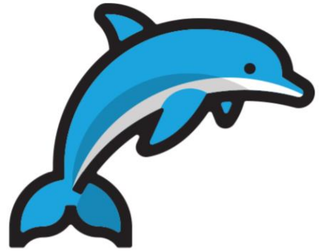


Serverless Means “Clusterless”

Run container without managing servers or clusters.

Ability

- Run container right way with one command
- Container level multiple tenant support
- Hypervisor level security isolation



ZUN

an OpenStack Community Project

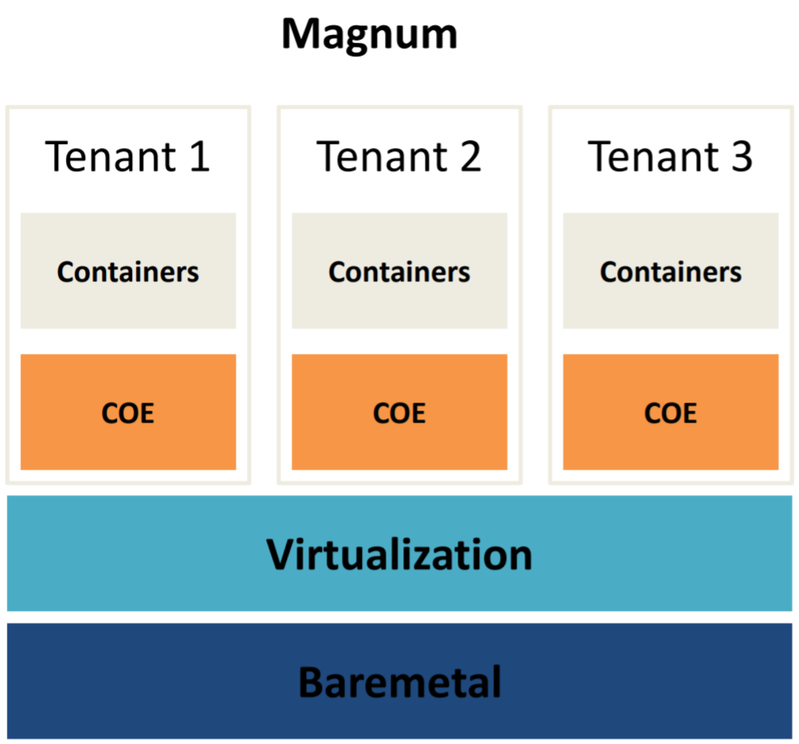
Azure Container Instance

AWS Fargate

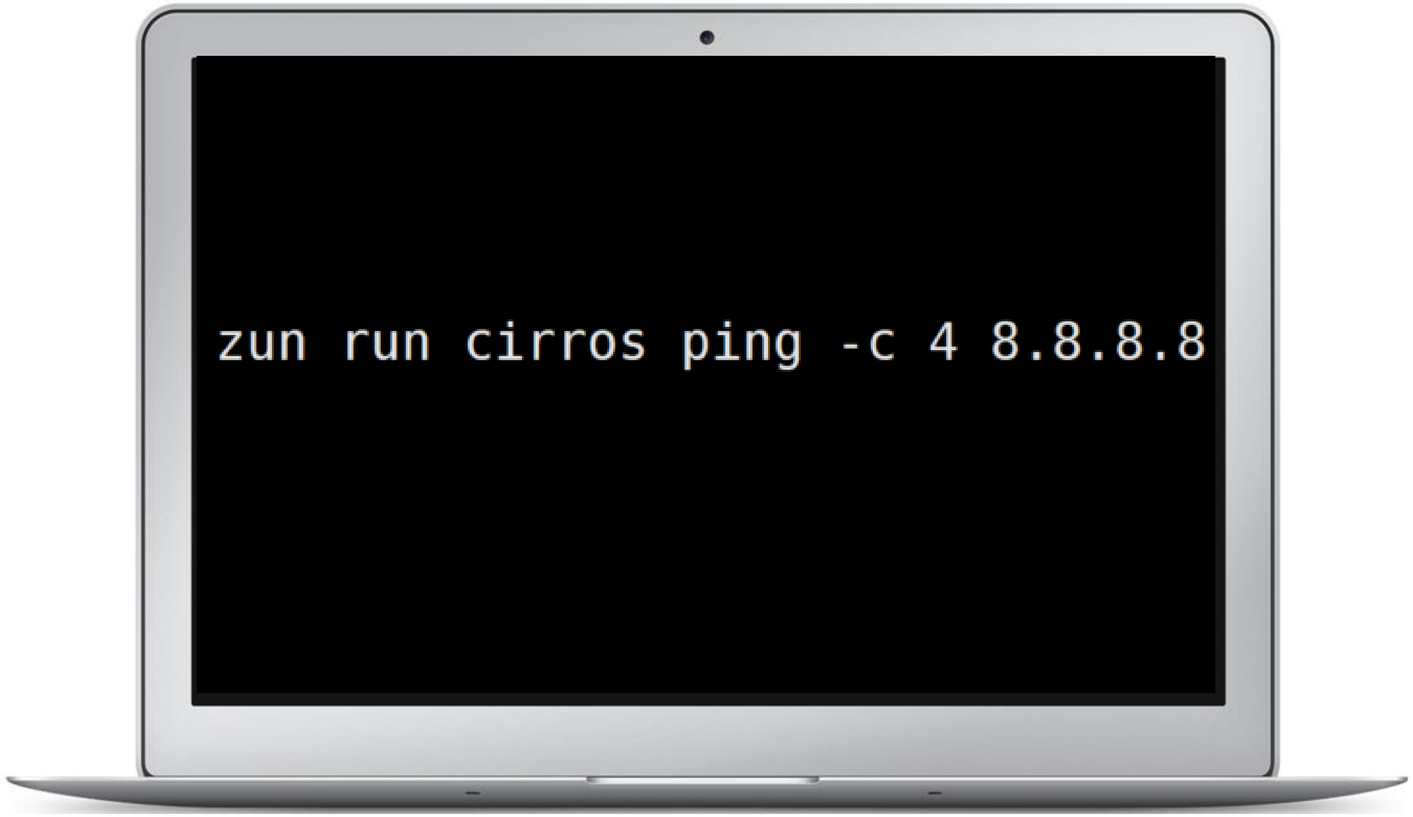
HYPER.SH

openstack.

Build a cluster

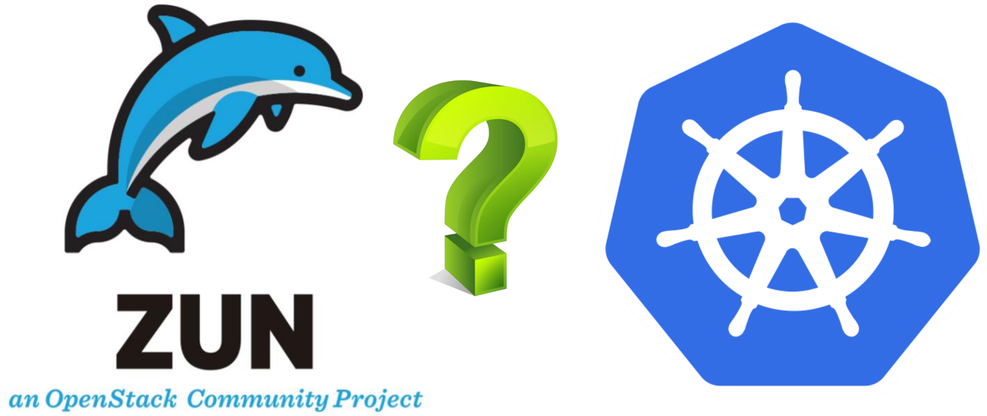


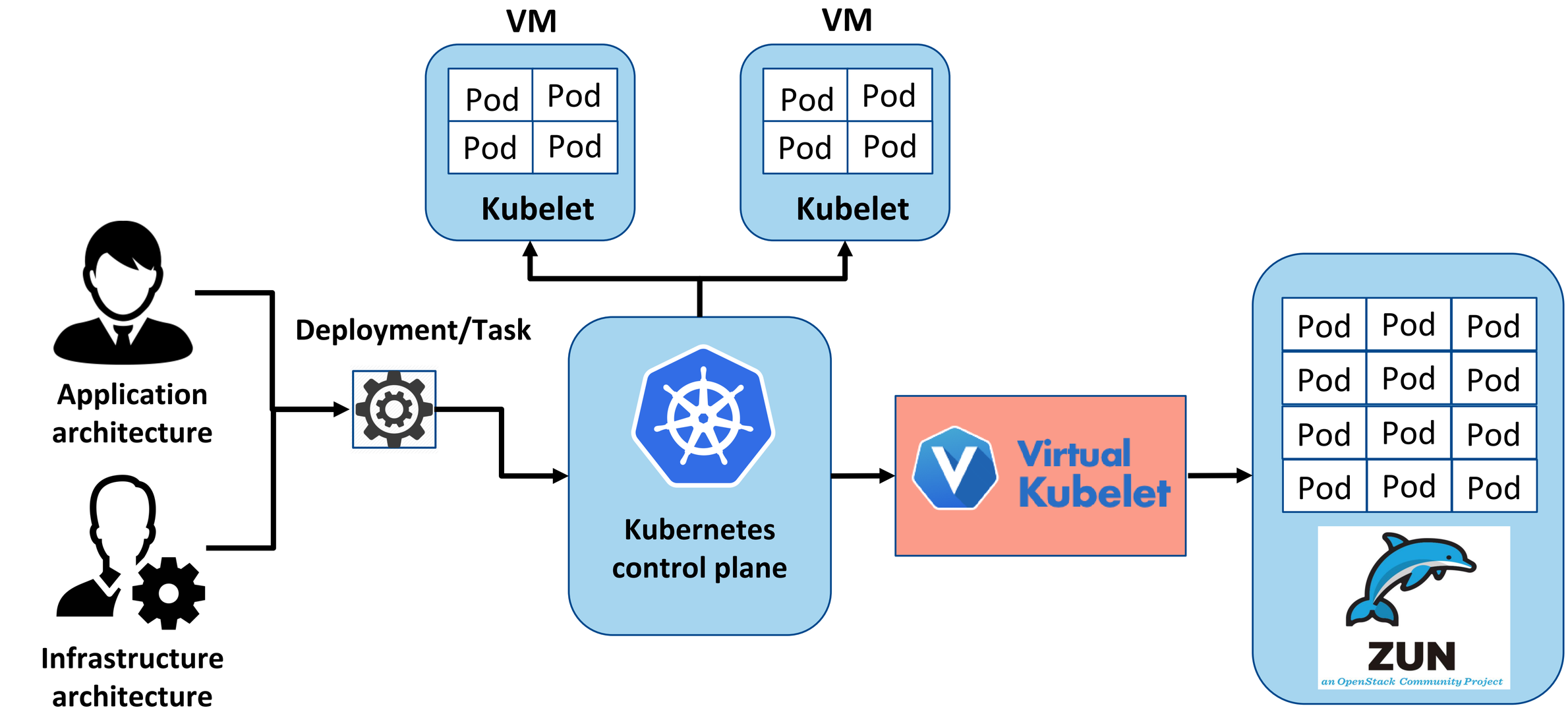
Just one command



openstack.

Serverless container technology is cool
But I need to work with Kubernetes

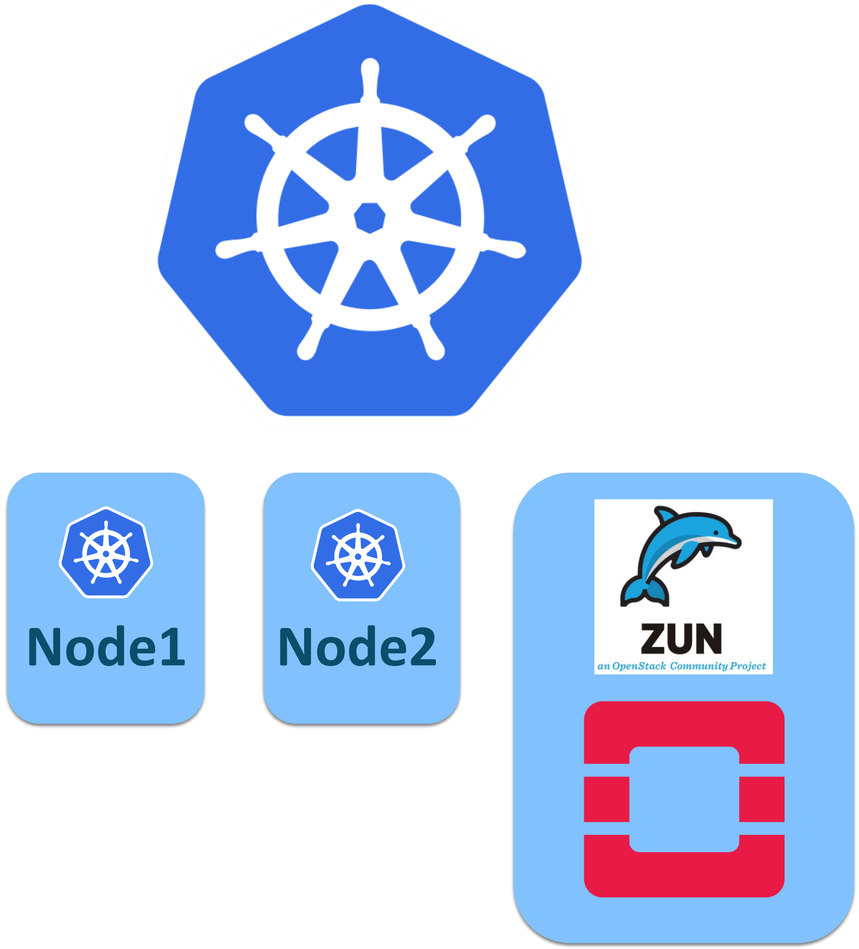



VM
VM
Pod
Pod
Pod
Pod
Kubelet
Pod
Pod
Pod
Pod
Kubelet
Application architecture
Infrastructure architecture
Deployment/Task
Kubernetes control plane
Virtual Kubelet
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
ZUN
an OpenStack Community Project

OpenStack as a Virtual Kubelet Node

- For user, Kubernetes on top
- OpenStack as a Virtual Kubelet node

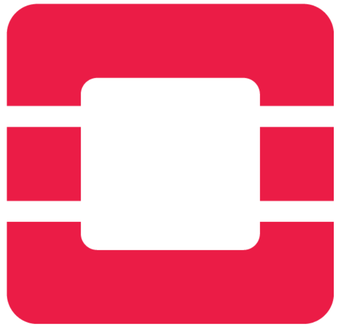
- Deploy workloads on this virtual node use kubectl and the backend realization is Zun.
- For user, nothing different. User only needs to focus on the containers.



openstack.

For building serverless container cloud, what do you need initially

Standalone OpenStack and Kubernetes
Network connected



OpenStack provisioned
Kubernetes

openstack.

Zun and Container Capsule

Zun Introduction

Wine container from Ancient China

Zun – Container Service

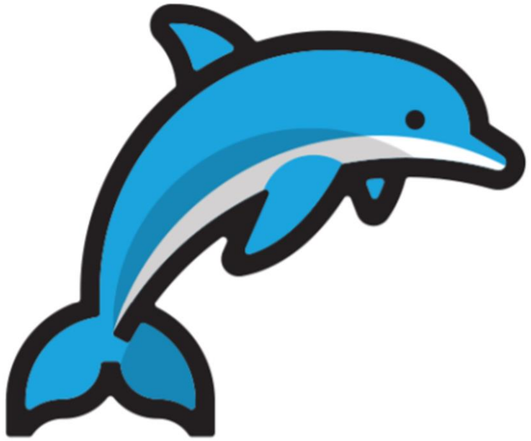
- Container Service of OpenStack
- Provide the ability of **provisioning and managing containers** without caring underlying infrastructur

Characteristics

- Container as the first class resource
- Individual IP Address/vCPU/Memory

Goal of Zun

- Make users focus on their application
- Pay just what they need(Clusterless)

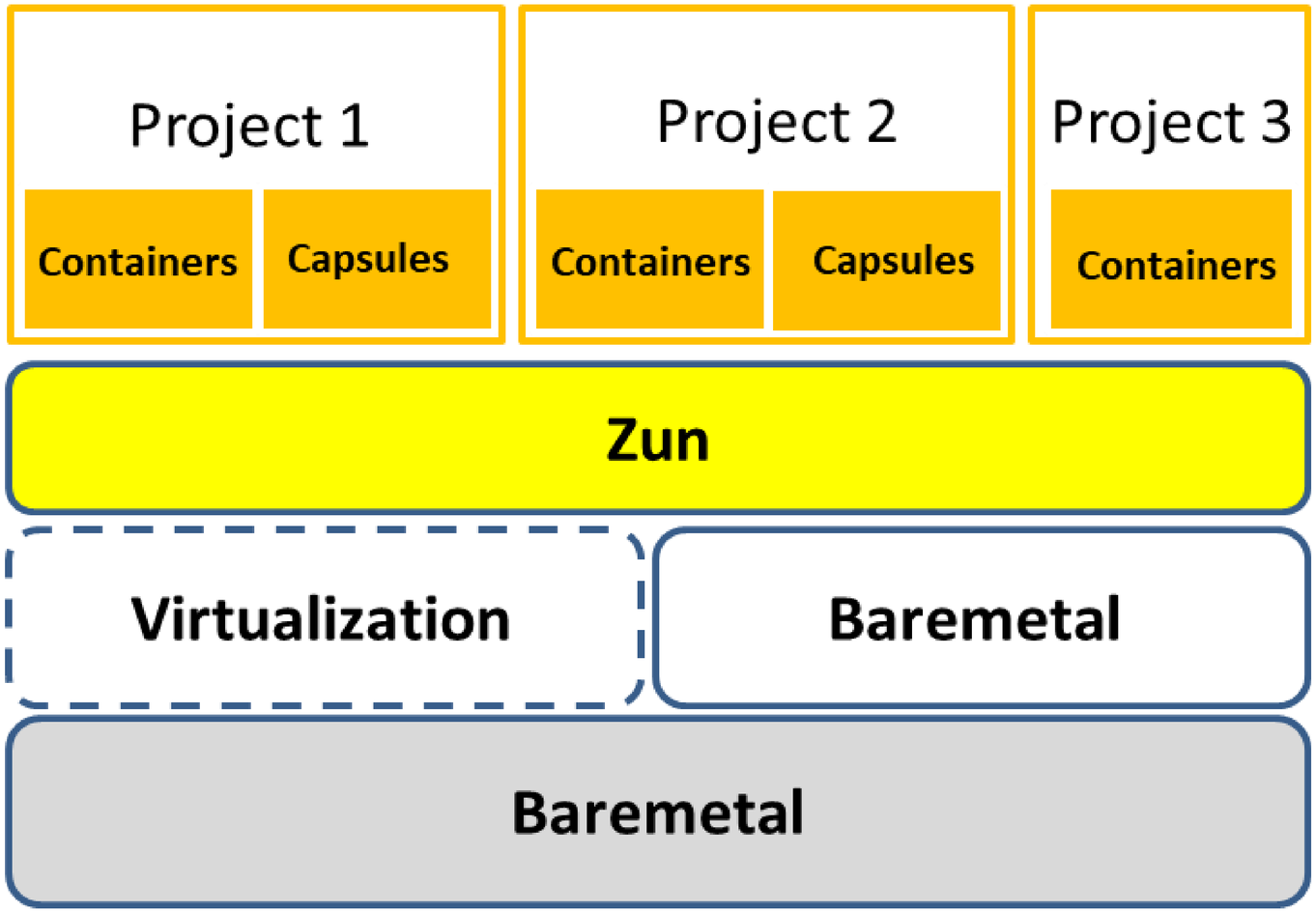


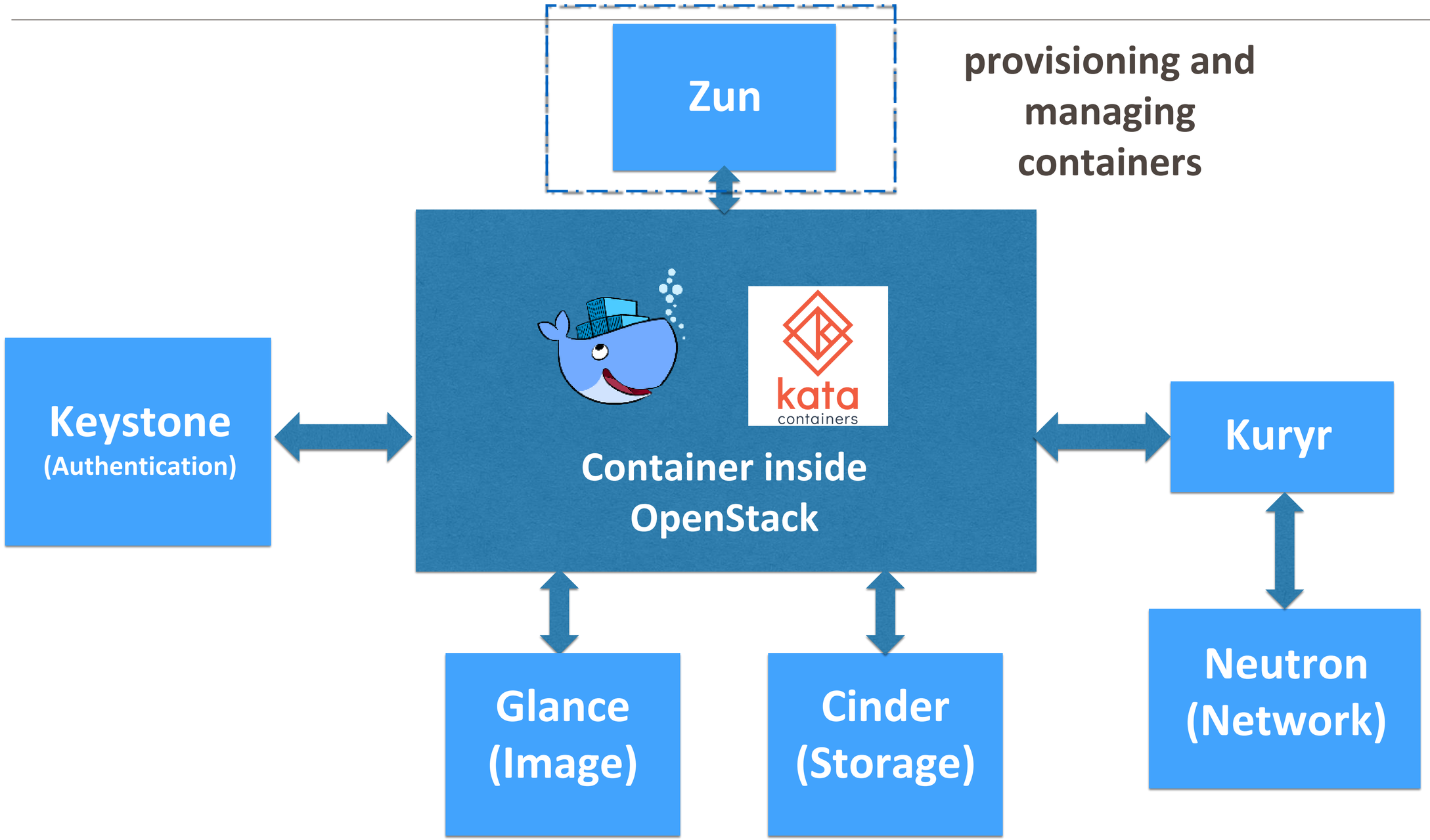
ZUN

an OpenStack Community Project

kata containers

Zun – Container Service




Zun
provisioning and managing containers
Keystone
(Authentication)
kata
containers
Container inside OpenStack
Kuryr
Glance
(Image)
Cinder
(Storage)
Neutron
(Network)

openstack.

Try Zun

```
$ # Run a container
$ docker search cirros
$ zun run --name test-container cirros ping 8.8.8.8

$ # Retrieve the log of the container
$ zun logs test-container
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=40 time=25.513 ms
64 bytes from 8.8.8.8: seq=1 ttl=40 time=25.348 ms
64 bytes from 8.8.8.8: seq=2 ttl=40 time=25.226 ms
64 bytes from 8.8.8.8: seq=3 ttl=40 time=25.275 ms

$ # Execute another command
$ zun exec test-container ls -a
...
```

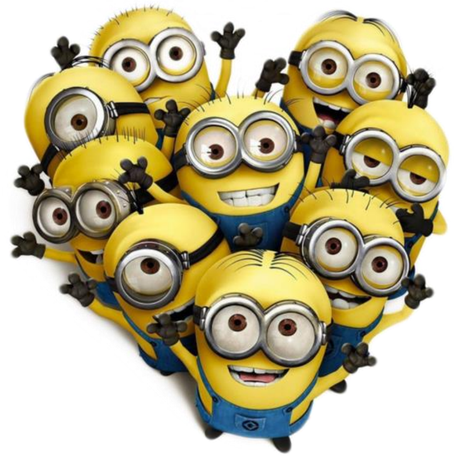
openstack.

How to implement Pod in Zun

openstack.

Zun and Container Capsule

Container Capsule



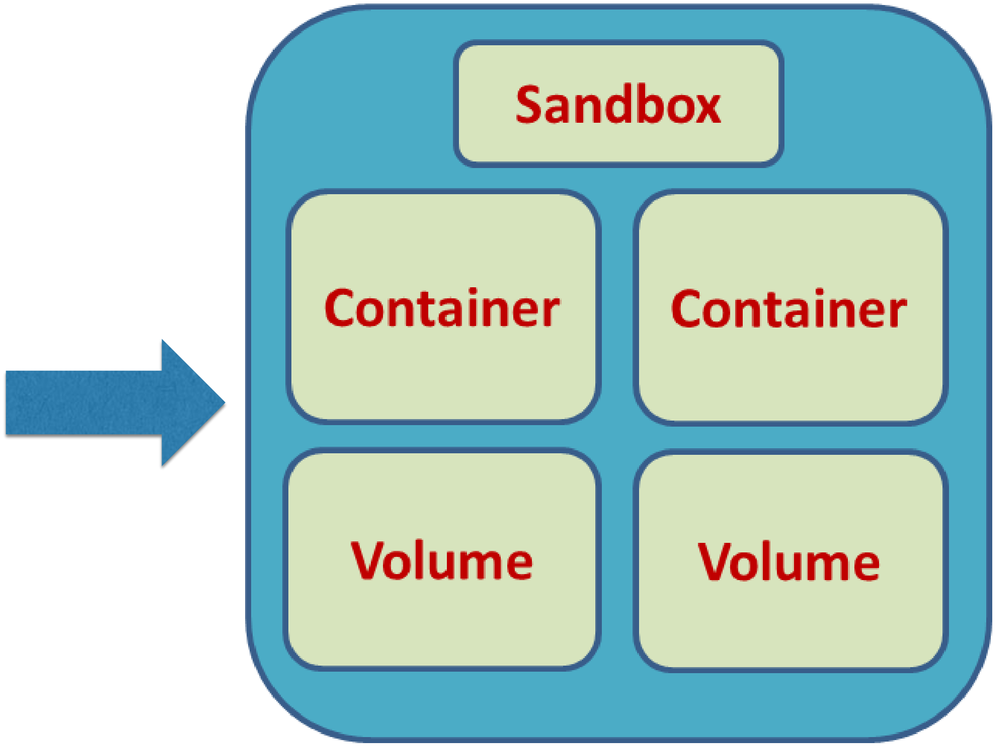
Container Capsule

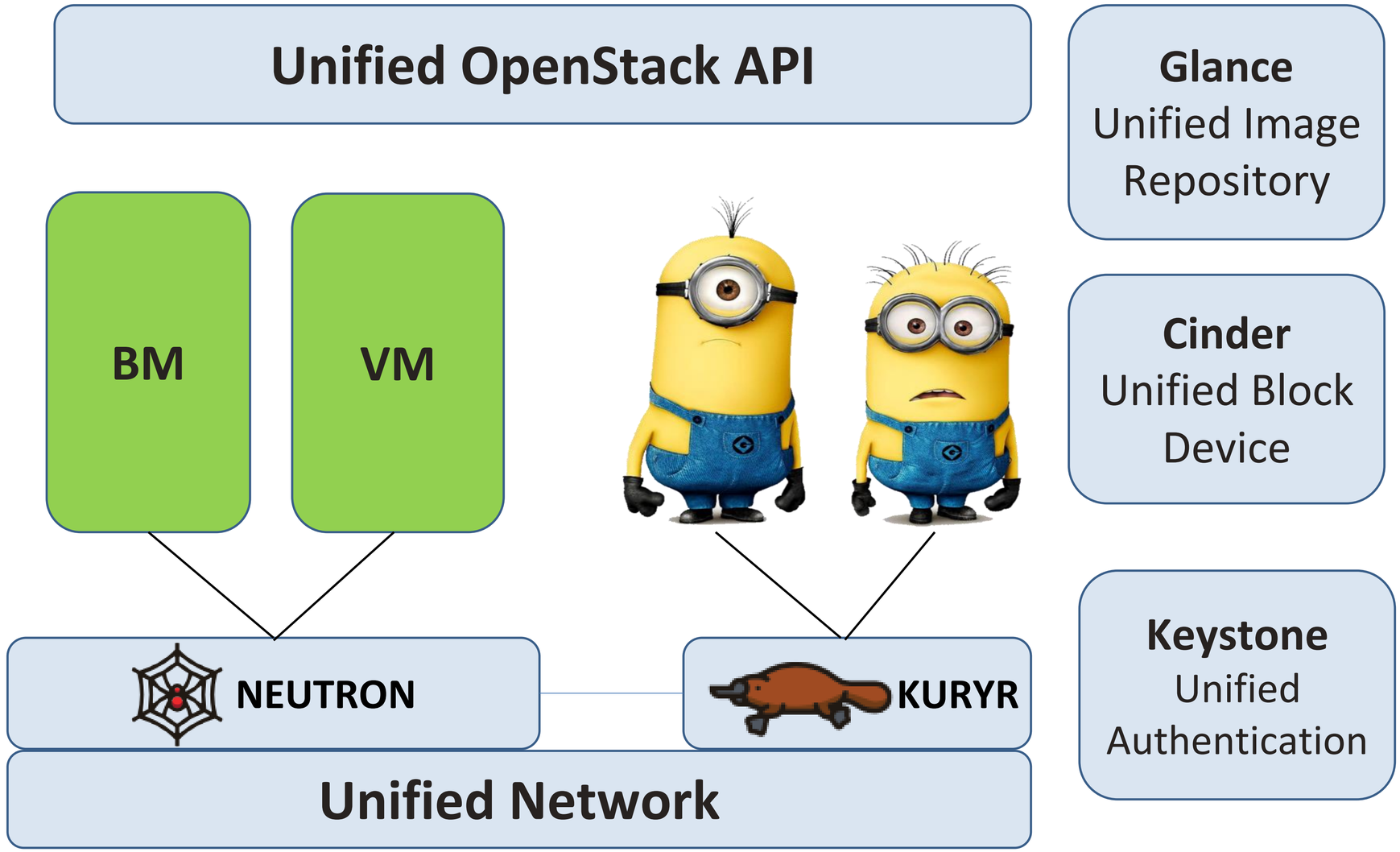
Component

- One Sandbox container
- Multiple containers
- Multiple volumes

Characteristic

- Basic unit in Zun
- Co-Scheduled/Co-located
- Share the network namespace
- Share the resource limits





Glance
Unified Image Repository

Cinder
Unified Block Device

Keystone
Unified Authentication

openstack.

Capsule Template

Support type:

- Yaml
- Json

Kubernetes friendly

```yaml
capsuleVersion: beta
kind: capsule
metadata:
  name: template
  labels:
    app: web
    app1: web1
spec:
  restartPolicy: Always
  containers:
  volumes:
```

```yaml
volumes:
- name: volume01
  cinder:
    size: 5
    autoRemove: True
- name: volume02
  cinder:
    size: 5
    autoRemove: True
```

```yaml
spec:
  containers:
  - image: ubuntu
    command:
      - "/bin/bash"
    imagePullPolicy: ifnotpresent
    workDir: /root
    ports:
      - name: nginx-port
        containerPort: 80
        hostPort: 80
        protocol: TCP
    resources:
      requests:
        cpu: 1
        memory: 1024
    env:
      ENV1: /usr/local/bin
      ENV2: /usr/bin
    volumeMounts:
    - name: volume01
      mountPath: /data1
      readOnly: True
```

openstack.

Capsule API

POST /v1/capsules

- zun capsule-create –f demo.yaml

GET /v1/capsules

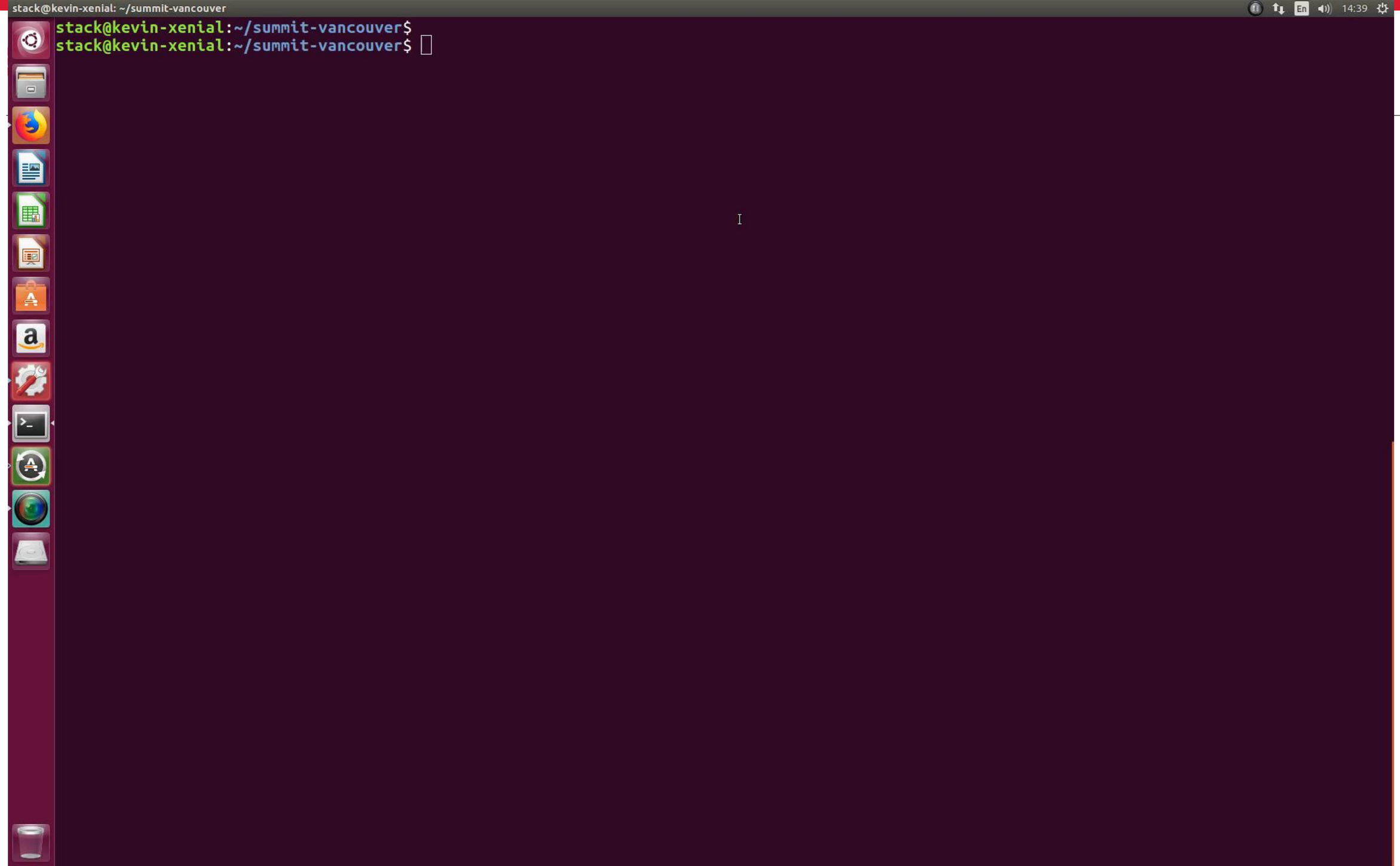
- Zun capsule-list

GET /v1/capsules/{uuid}

- Zun capsule-describe <uuid>/<name>

DELETE /v1/capsules/{uuid}

- Zun capsule-delete <uuid>/<name>


stack@kevin-xenial: ~/summit-vancouver
14:39
stack@kevin-xenial:~/summit-vancouver$
stack@kevin-xenial:~/summit-vancouver$

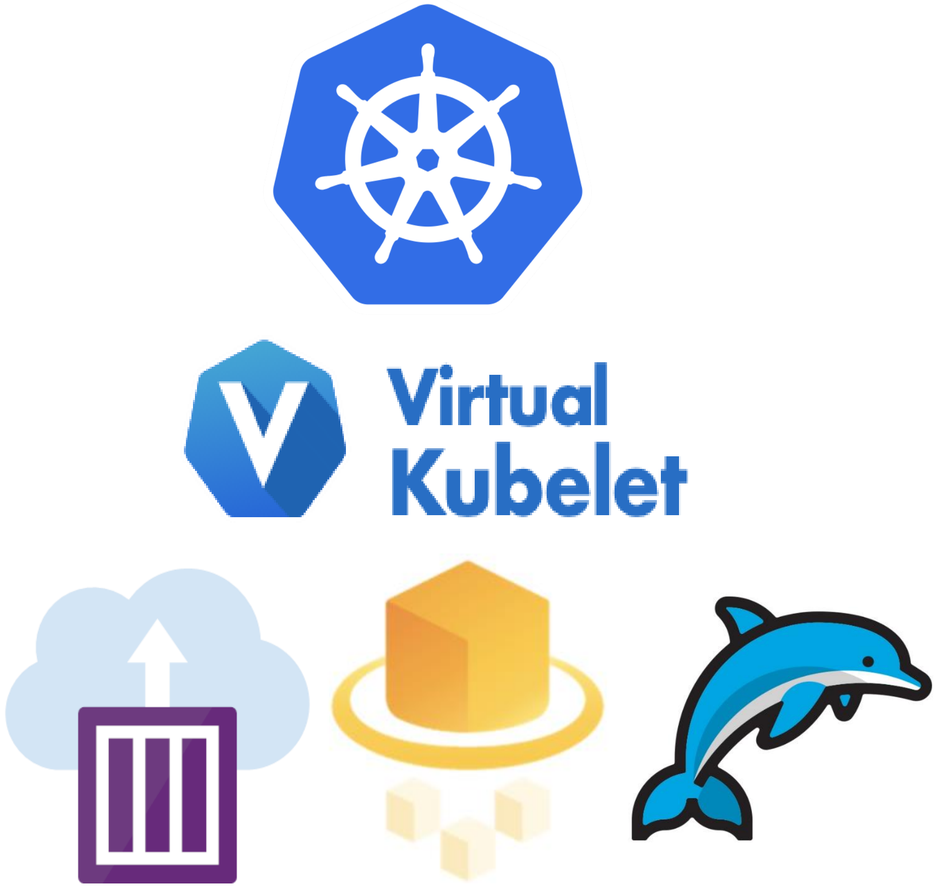
openstack.

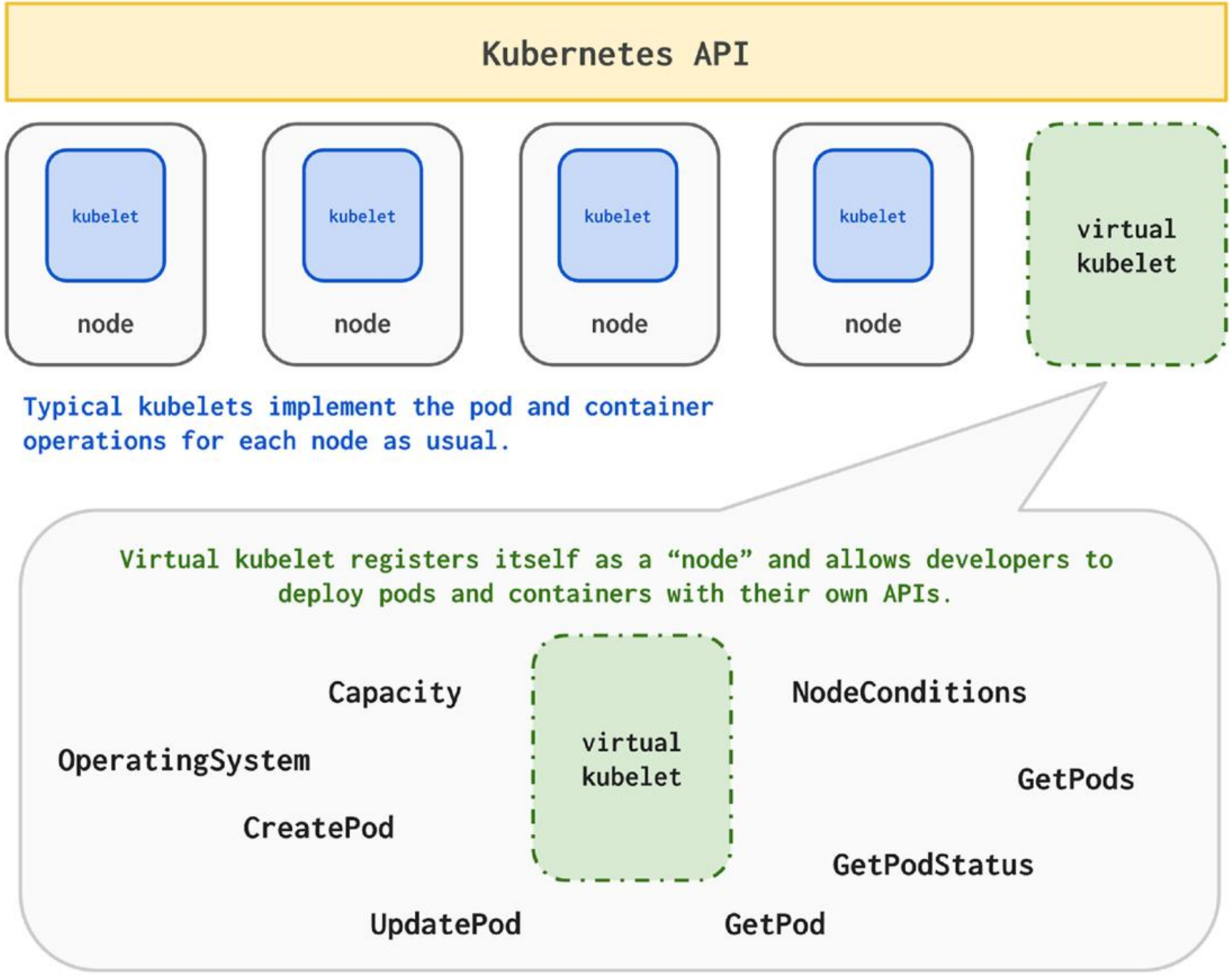
Build Serverless Container Cloud

Kubernetes on top, Zun backend

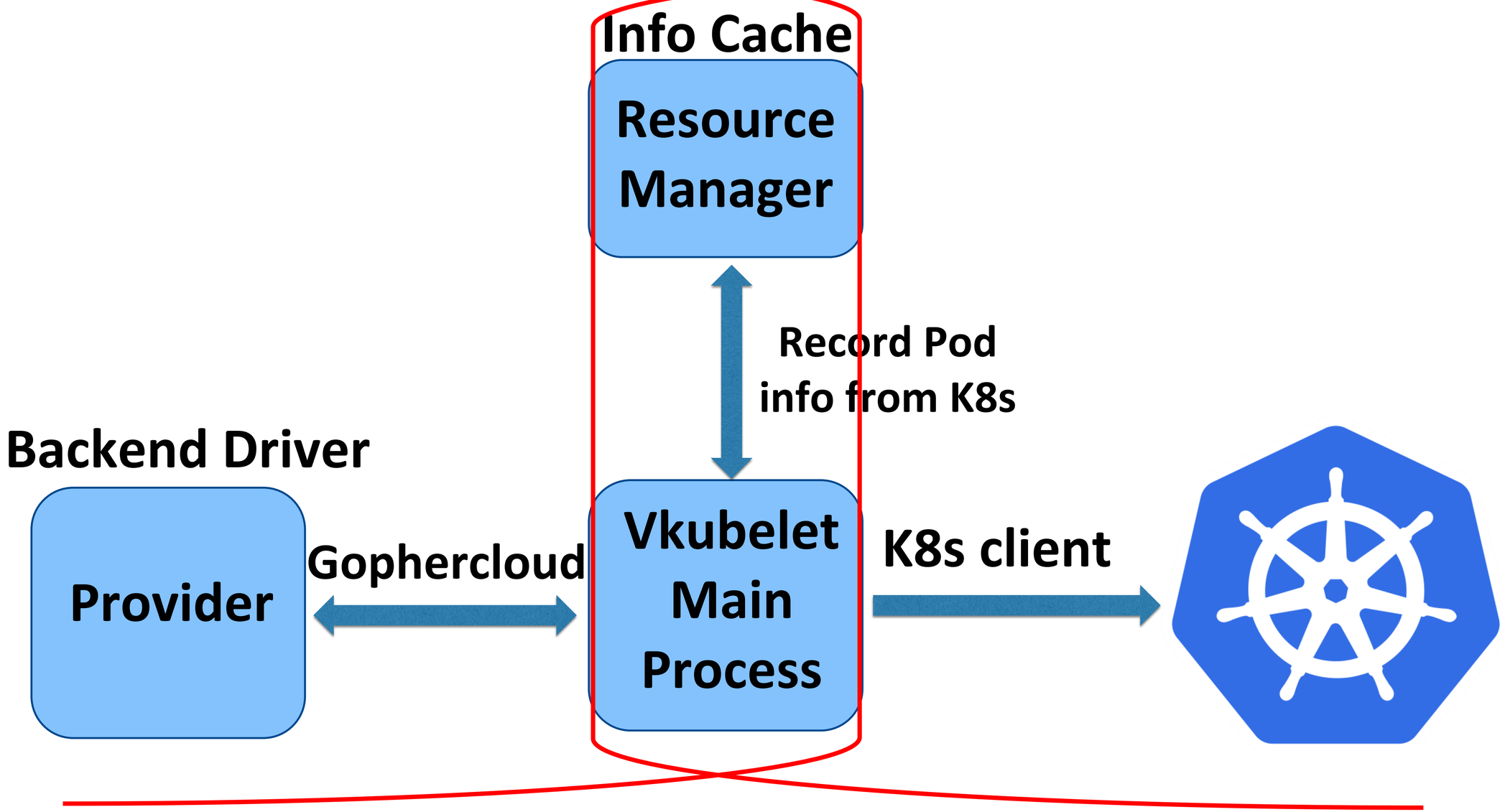
Virtual-Kubelet

- Kubelet implementation, masquerades container service as Kubelet node.
- **Kubernetes on top, programmed back.**
- **Intermediary** to map Kubernetes requests and resource to container service

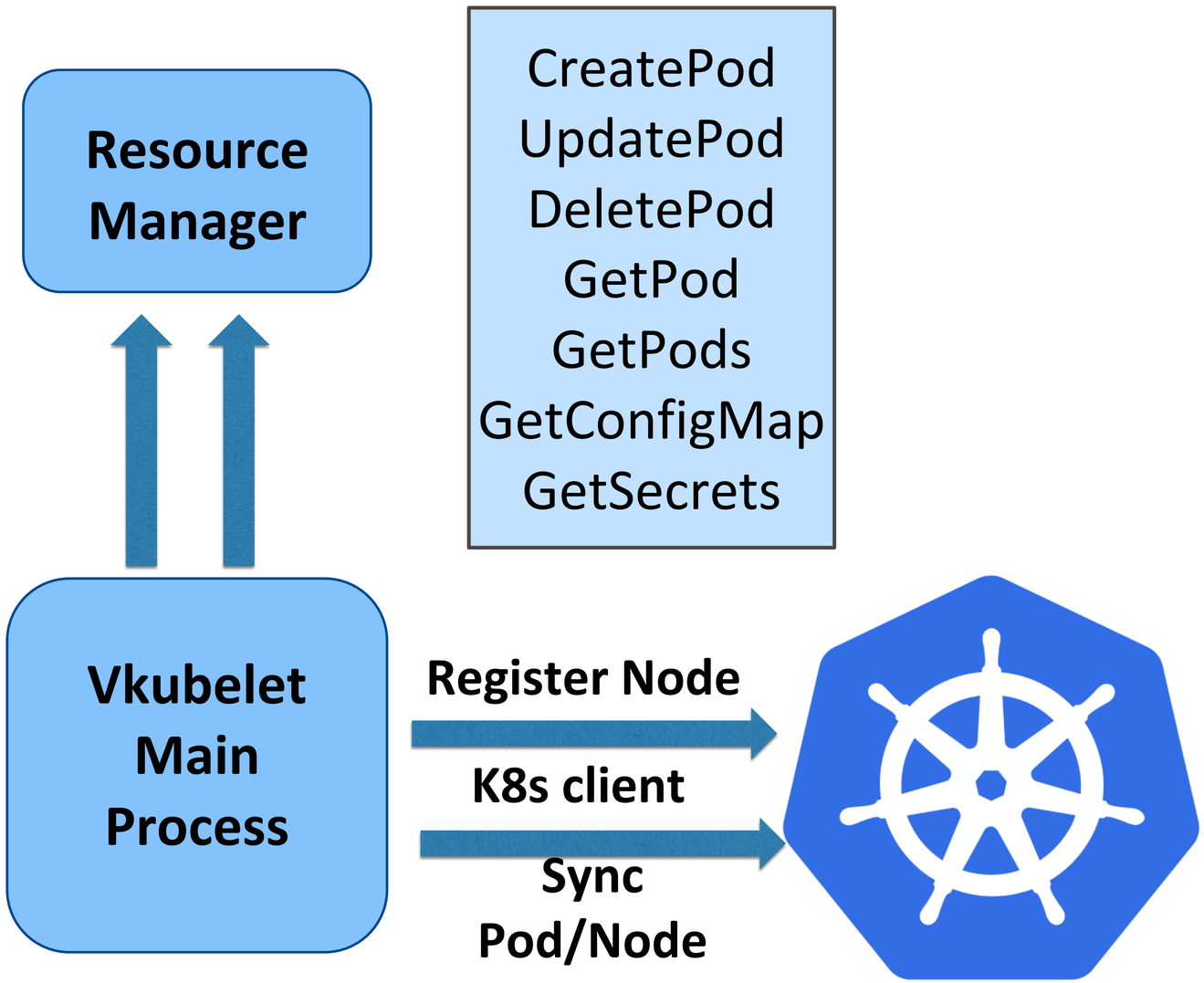



Kubernetes API
kubelet
node
kubelet
node
kubelet
node
kubelet
node
virtual kubelet
Typical kubelets implement the pod and container operations for each node as usual.
Virtual kubelet registers itself as a “node” and allows developers to deploy pods and containers with their own APIs.
Capacity
NodeConditions
virtual kubelet
OperatingSystem
GetPods
CreatePod
GetPodStatus
UpdatePod
GetPod

Virtual-kubelet structure



Virtual-kubelet Structure



Register Node:

1. Capacity(), get virtual node info. Fill into k8s node object.
2. Call: k8sClient.CoreV1().Nodes().Create (node)

Sync Pod and Node:

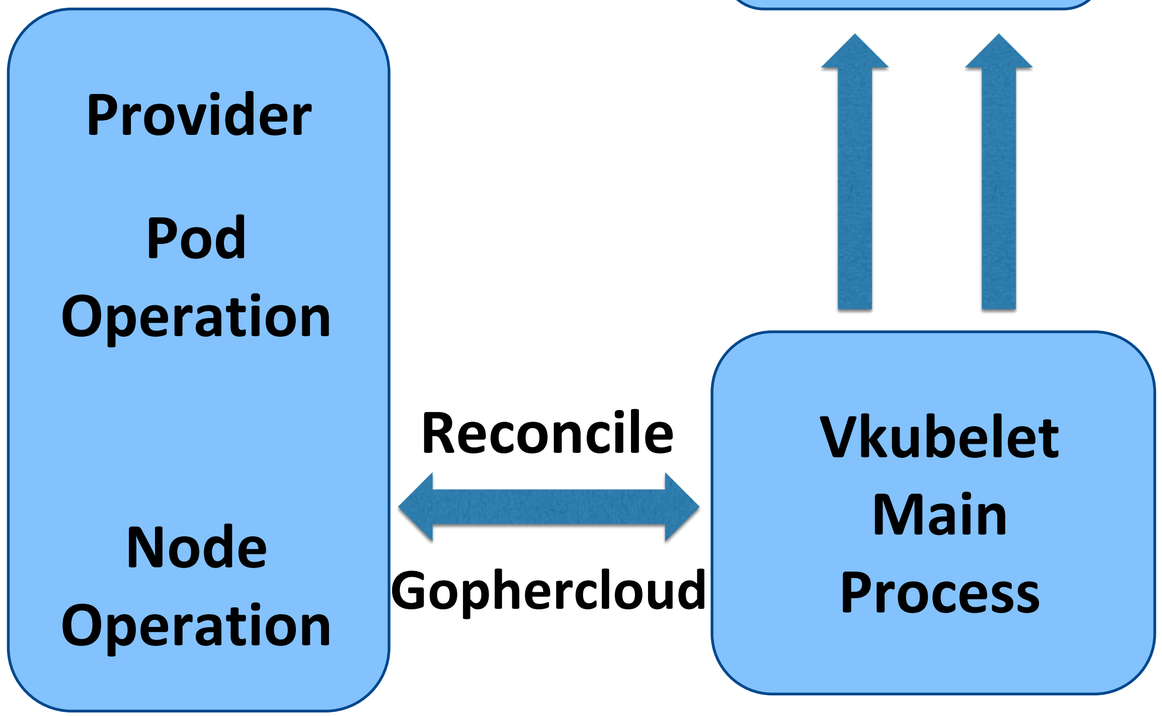
1. Watch pod requests from K8s. Update resource manager.
2. Periodic task to update Node and Pod status from provider to K8s.

openstack.

Virtual-kubelet Structure

Pod:
CreatePod
UpdatePod
DeletePod
GetPod
GetPods
GetPodStatus

Node:
Capacity
NodeCondition



Reconcile:
1. Create:
GetPods from RM,
GetPods from Provider,
CreatePod if no DeletionTimeStamp
2. Delete:
GetPods from provider,
Check into RM, if not exist, DeletePod from Provider

openstack.

Virtual-kubelet Zun support

Communication:

1. Gophercloud for Zun

   Capsule API support in Gophercloud

2. Virtual Kubelet Zun client

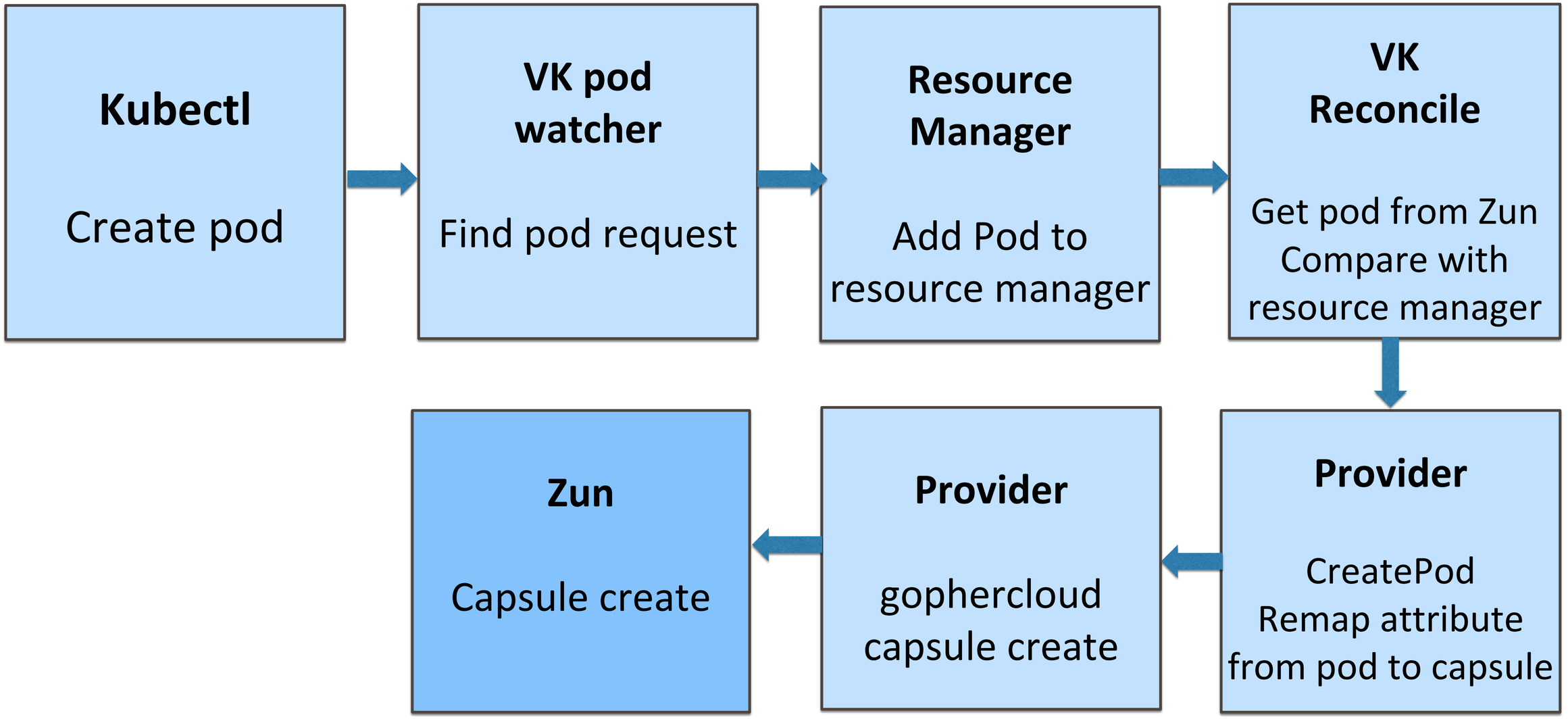
   Connect Zun by Gophercloud

Resource Providing:

1. Capsule will be the backend realization of Pod
2. Provider essential functions for pod and node management

openstack.

Pod Create Process



openstack.

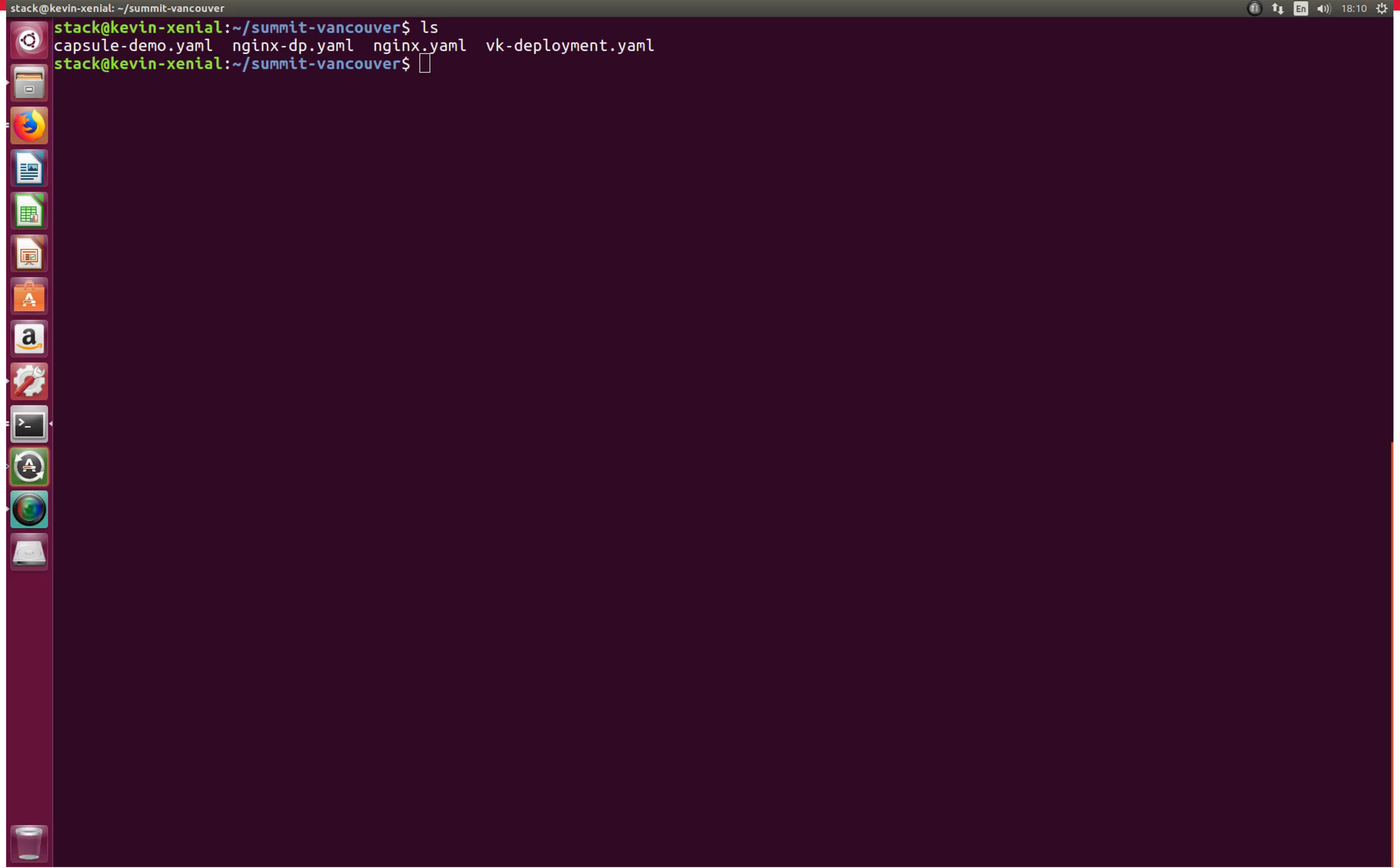
Future

- Enhanced the virtual kubelet support for configmap and secret
- Enhanced Capsule implementation and operation
- Aligned with Kubernetes related attribute
- Cinder multiple attach for container

openstack.

Talk is cheap

Show me the demo

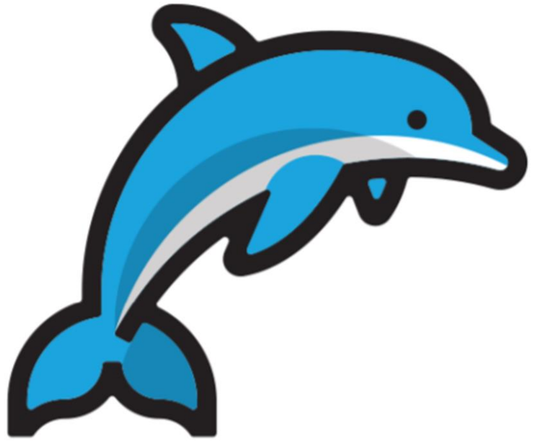

stack@kevin-xenial: ~/summit-vancouver
18:10
stack@kevin-xenial:~/summit-vancouver$ ls
capsule-demo.yaml nginx-dp.yaml nginx.yaml vk-deployment.yaml
stack@kevin-xenial:~/summit-vancouver$

openstack.

You are welcome to join us

Wiki: https://wiki.openstack.org/wiki/Zun

IRC: #openstack-zun



ZUN

an OpenStack Community Project

Integration of Openstack Zun with Kata containers

May 23th, 2:40pm-3:20pm, Room 109

Zun Project Update

May 24th, 3:30pm-3:50pm, Room 212

openstack.

THANKS.

Questions?

 openstack

 @OpenStack

 openstack

 OpenStackFoundation