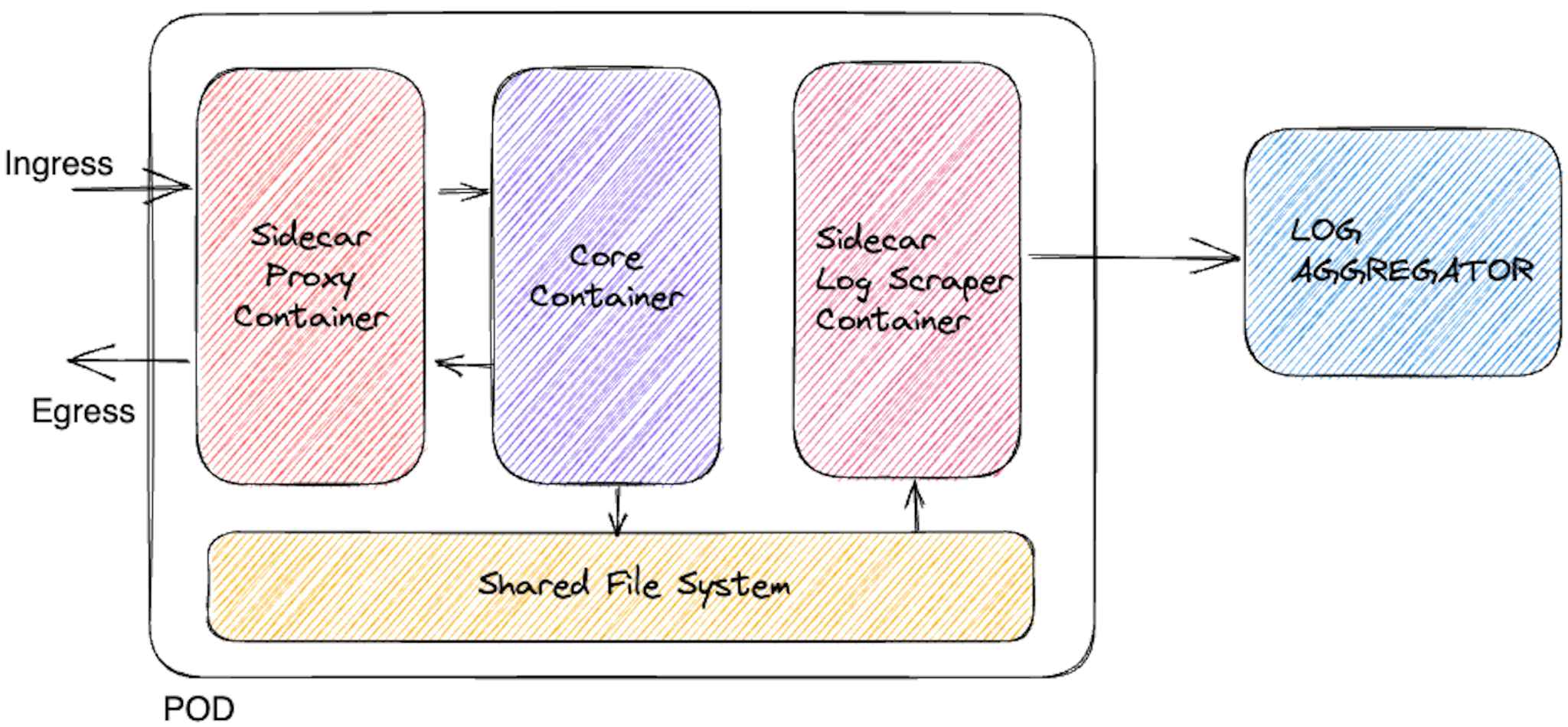
# Workloads

Kubernetes Objects - apiVersion, Kind, metadata, and Spec

## Pod



alias k=kubectl

k api-resources

k explain pod.spec.containers

k explain pod.spec.containers --recursive

### 

### One-container-per-Pod

Create a file - vi pod.yaml

apiVersion: v1

kind: Pod

metadata:

name: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

kubectl apply -f pod.yaml

kubectl get pods

kubectl describe pod nginx

# Looks for events and lifecyle

### Multi-Container-Pod

apiVersion: v1

kind: Pod

metadata:

name: multicontainer-pods

spec:

containers:

#Container 01

- name: web

image: httpd

ports:

- containerPort: 80

#Container 02

- name: rexdis

image: redis

### Init-Container



apiVersion: v1

kind: Pod

metadata:

name: purple

spec:

containers:

- command:

- sh

- -c

- echo The app is running! && sleep 3600

image: busybox:1.28

name: purple-container

# Adding 2 init containers to execute sleep commands

initContainers:

- command:

- sh

- -c

- sleep 60

image: busybox:1.28

name: warm-up-1

- command: ["sh", "-c", "sleep 120"]

image: busybox:1.28

name: warm-up-2

kubectl get pods purple -w

# Both the init containers will get executed before the main container is started

# NAME READY STATUS RESTARTS AGE

# purple 0/1 Init:1/2 0 2m41s

# After 3mins(60+120seconds), the output will be

# NAME READY STATUS RESTARTS AGE

# purple 1/1 Running 0 3m7s

### Static-Pod

SSH into worker-node-1,

sudo ls /etc/kubernetes/

sudo mkdir /etc/kubernetes/manifests

sudo ls /etc/kubernetes/

sudo vi /etc/kubernetes/manifests/pod1

Create a yaml file in

apiVersion: v1

kind: Pod

metadata:

name: static-web

spec:

containers:

- name: web

image: nginx

ports:

- name: web

containerPort: 80

protocol: TCP

In the master node,

k get pods | grep static-web

The pod will appear in default ns

Delete the static pod file in worker01

sudo rm /etc/kubernetes/manifests/pod1

In the master node,

k get pods -A

The pod will Disappear in default ns

### Resource Limits

apiVersion: v1

kind: Pod

metadata:

name: rl-pod

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

resources:

requests: # Minimum Value

memory: "100Mi"

cpu: "250m" # 1 core = 1000m

limits: # Maximum Value

memory: "128Mi"

cpu: "300m"

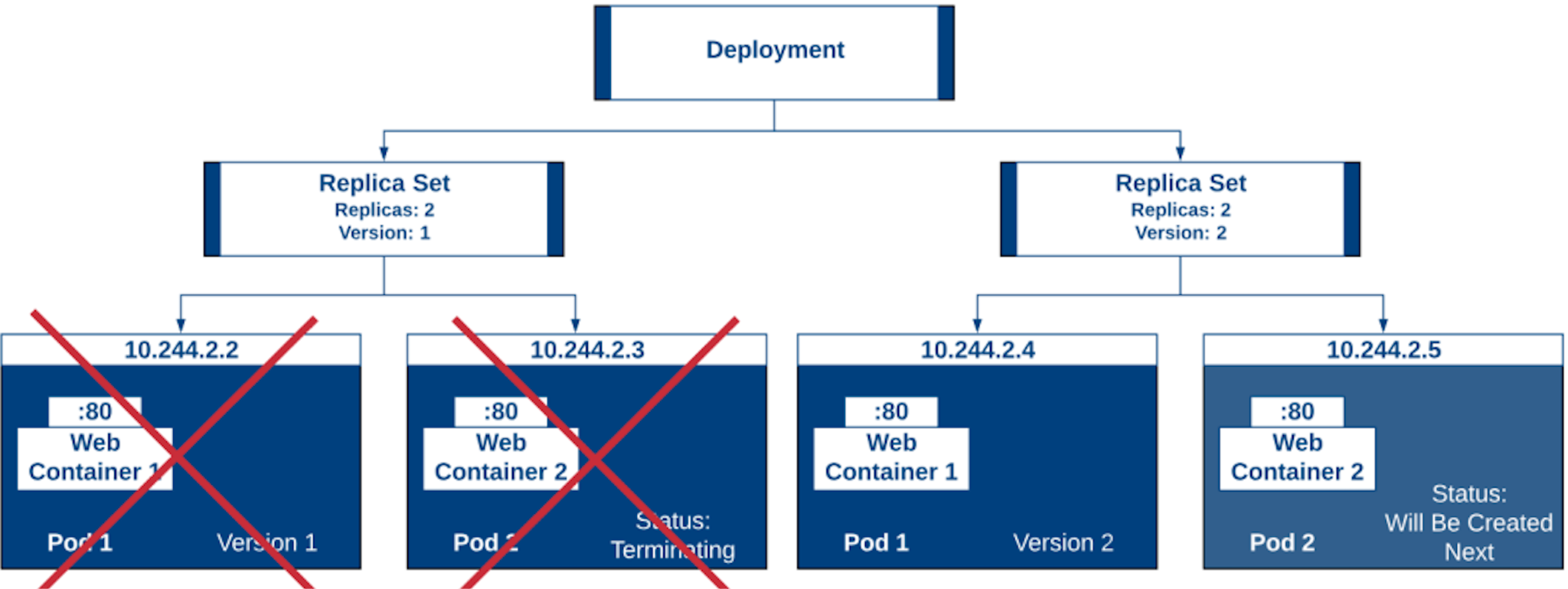
K apply -f resource-limits.yaml

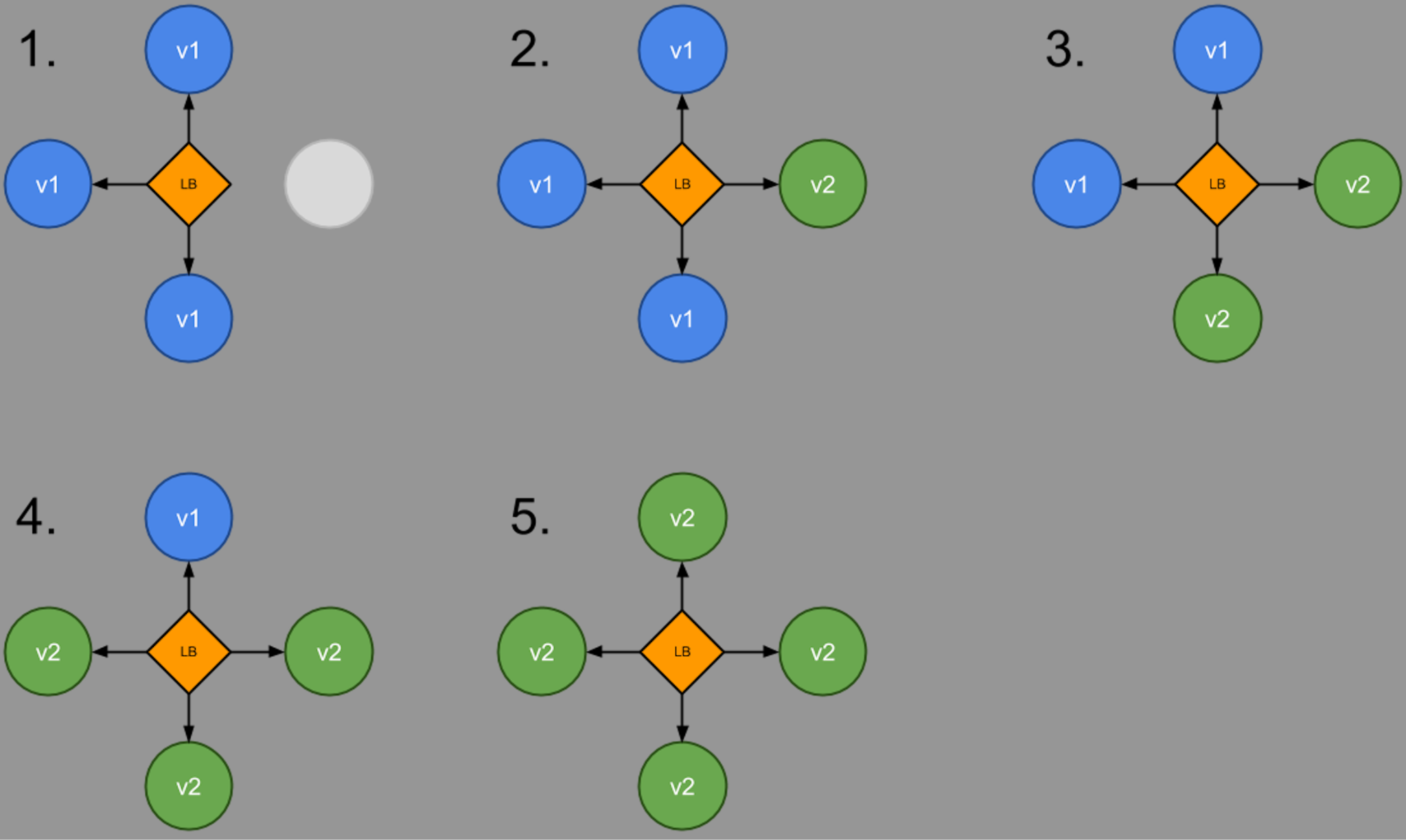
k describe pod rl-pod

## Deployment

k explain deploy

k explain deploy.spec.strategy --recursive





apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

labels:

app: nginx

spec:

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:1.14.2

ports:

- containerPort: 80

kubectl apply -f deploy.yaml

kubectl get deployments

kubectl rollout status deployment nginx-deployment

kubectl get rs

### Update Deployment

k explain deploy.spec.strategy.rollingUpdate

Syntax: [Not to be executed]

kubectl set image deployment <deployment-name> container-name=image:tag

kubectl set image deployment nginx-deployment nginx=nginx:1.16.1

kubectl rollout status deployment nginx-deployment

kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment nginx-deployment

**Setting wrong image**

kubectl set image deployment nginx-deployment nginx=nginx:xxxxxx

kubectl rollout status deployment nginx-deployment

Waiting for rollout to finish: 1 out of 3 new replicas has been updated...

kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment

kubectl rollout history deployment nginx-deployment

kubectl rollout history deployment nginx-deployment --revision=2

### Rolling Back to a Previous Revision

kubectl rollout undo deployment nginx-deployment

kubectl rollout history deployment nginx-deployment

kubectl rollout history deployment nginx-deployment --revision=4

kubectl get deployment nginx-deployment

kubectl describe deployment nginx-deployment

# Check container image version/tag

k rollout undo deployment nginx-deployment --to-revision=1

1 -> nginx:1.14.2

2 -> nginx:1.16.1

3 -> nginx:xxxx

undo = 3-1 = 2

1 -> nginx:1.14.2

3 -> nginx:xxxx

4 -> nginx:1.16.1

undo = 1

3 -> nginx:xxxx

4 -> nginx:1.16.1

5 -> nginx:1.14.2

### Scaling deployment

kubectl scale deployment nginx-deployment --replicas=10

kubectl scale deployment nginx-deployment --replicas=1

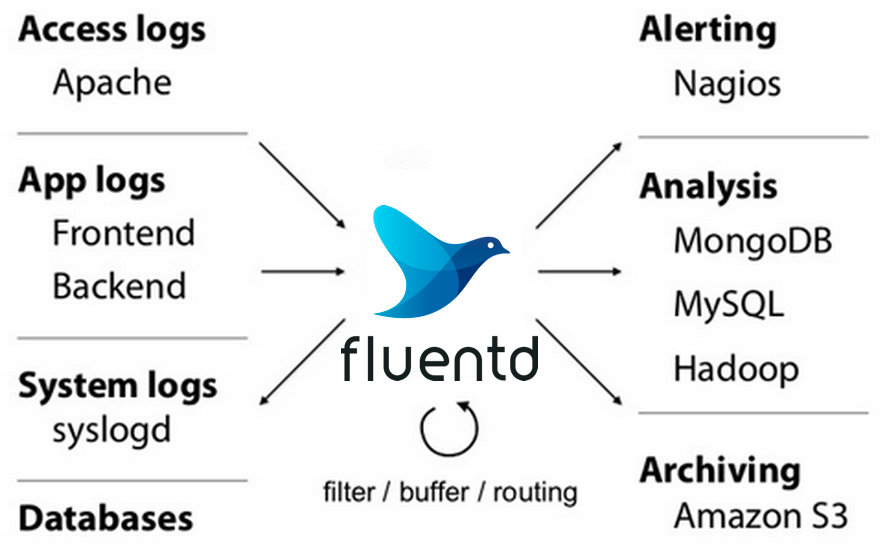
kubectl get deployment nginx-deployment

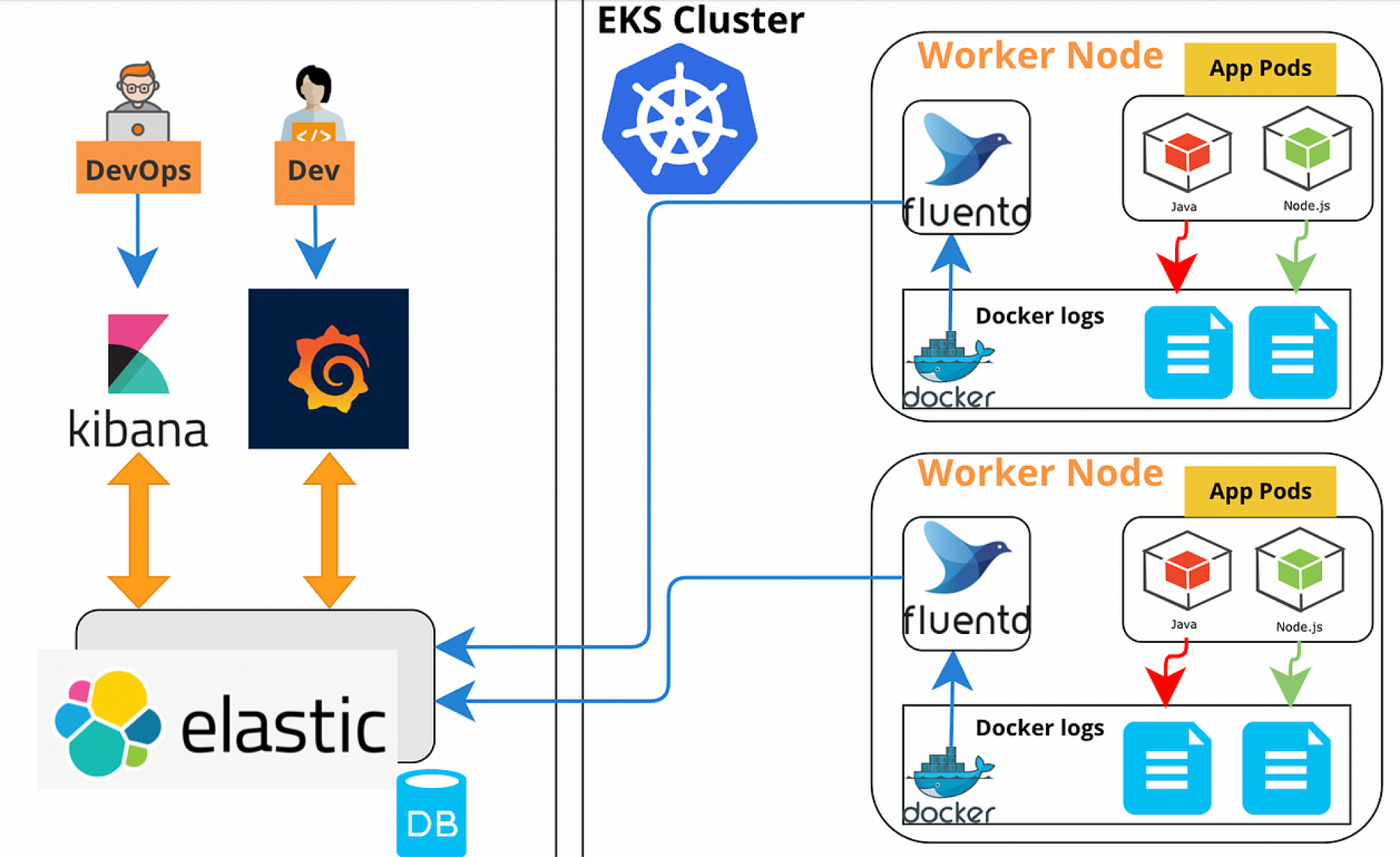
kubectl get rs

kubectl get pods | grep nginx-deployment

kubectl describe deployment nginx-deployment

## Daemon Set





wget https://k8s.io/examples/controllers/daemonset.yaml

kubectl apply -f https://k8s.io/examples/controllers/daemonset.yaml

kubectl get ds -n kube-system

kubectl describe ds fluentd-elasticsearch -n kube-system

kubectl get pods -o wide -n kube-system | grep fluentd

## Jobs

apiVersion: batch/v1

kind: Job

metadata:

name: pi

spec:

template:

spec:

containers:

- name: pi

image: busybox:1.28

imagePullPolicy: IfNotPresent

command:

- /bin/sh

- -c

- date; echo Hello from the Kubernetes cluster

restartPolicy: Never

backoffLimit: 4

kubectl apply -f job.yaml

kubectl describe jobs/pi

## CronJobs

apiVersion: batch/v1

kind: CronJob

metadata:

name: hello

spec:

schedule: "\* \* \* \* \*"

jobTemplate:

spec:

template:

spec:

containers:

- name: hello

image: busybox:1.28

imagePullPolicy: IfNotPresent

command:

- /bin/sh

- -c

- date; echo Hello from the Kubernetes cluster

restartPolicy: OnFailure

kubectl apply -f cronjob.yaml

kubectl get cronjob hello

kubectl get jobs -w

kubectl delete -f cronjob.yaml

Reference: [https://crontab.guru/#\*\_\*\_\*\_\*\_\*](https://crontab.guru/#*_*_*_*_*)

## Configuration basics

Introduction:



<https://12factor.net/config>

[All JAVA Spring Configuration item](https://docs.spring.io/spring-boot/docs/current/reference/html/application-properties.html#application-properties.data.spring.datasource.username)

[MySQL DB Configuration](https://github.com/spring-guides/gs-accessing-data-mysql/blob/main/complete/src/main/resources/application.properties)

a[pplication.properties](https://github.com/spring-guides/gs-accessing-data-mysql/blob/main/complete/src/main/resources/application.properties)

spring.datasource.url=jdbc:mysql:// \

${MYSQL\_HOST:localhost}:3306/db\_example

spring.datasource.username=springuser

spring.datasource.password=ThePassword

spring.datasource.driver-class-name =com.mysql.jdbc.Driver

For each environment, we define the individual application.properties



## Env

env.yaml  
apiVersion: v1

kind: Pod

metadata:

name: envar-demo

labels:

purpose: demonstrate-envars

spec:

containers:

- name: envar-demo-container

image: gcr.io/google-samples/node-hello:1.0

env:

- name: DEMO\_GREETING

value: "Hello from the environment"

- name: DEMO\_FAREWELL

value: "Such a sweet sorrow"

kubectl apply -f env.yaml

kubectl exec envar-demo -- printenv

## 

## ConfigMaps

config-map.yaml

apiVersion: v1

kind: ConfigMap

metadata:

name: game-demo

data:

*# property-like keys; each key maps to a simple value*

player\_initial\_lives: "3"

ui\_properties\_file\_name: "user-interface.properties"

*# file-like keys*

game.properties: |

*enemy.types=aliens,monsters*

*player.maximum-lives=5*

user-interface.properties: |

*color.good=purple*

*color.bad=yellow*

*allow.textmode=true*

k apply -f config-map.yaml

k describe cm game-demo

**Using EnvFrom**

cm-pod.yaml

apiVersion: v1

kind: Pod

metadata:

name: dapi-test-pod02

spec:

containers:

- name: test-container

image: gcr.io/google-samples/node-hello:1.0

envFrom:

- configMapRef:

name: game-demo

k apply -f cm-pod.yaml

k exec dapi-test-pod02 -- printenv

## Secrets

apiVersion: v1

kind: Secret

metadata:

name: mysecret

type: kubernetes.io/basic-auth

stringData:

username: admin

password: t0p-Secret

k apply -f secret

k describe secret mysecret

apiVersion: v1

kind: Pod

metadata:

name: dapi-test-pod03

spec:

containers:

- name: test-container

image: gcr.io/google-samples/node-hello:1.0

envFrom:

- secretRef:

name: mysecret

k apply -f secret-pod

k exec dapi-test-pod03 -- printenv

## Monitoring Pods/Nodes: Metrics Server

kubectl apply -f<https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml>

kubectl get pods -n kube-system

​​wget -c https://gist.githubusercontent.com/initcron/1a2bd25353e1faa22a0ad41ad1c01b62/raw/008e23f9fbf4d7e2cf79df1dd008de2f1db62a10/k8s-metrics-server.patch.yaml

kubectl patch deploy metrics-server -p "$(cat k8s-metrics-server.patch.yaml)" -n kube-system

kubectl get pods -n kube-system

### Metrics Server Commands

kubectl top pods --all-namespaces

kubectl top node

kubectl top node master

## Horizontal Pod Autoscaling(HPA)

kubectl autoscale deploy nginx-deployment --min=3 --max=5 --cpu-percent=40

kubectl get hpa

kubectl delete hpa nginx-deployment

apiVersion: autoscaling/v2

kind: HorizontalPodAutoscaler

metadata:

name: nginx-deployment

spec:

maxReplicas: 5

metrics:

- resource:

name: cpu

target:

averageUtilization: 40

type: Utilization

type: Resource

minReplicas: 3

scaleTargetRef:

apiVersion: apps/v1

kind: Deployment

name: nginx-deployment

## Namespaces

kubectl get namespace

kubectl get ns

kubectl create ns test

kubectl create -f deploy.yaml -n test

k get all -n test

k delete ns test

## Helm

QuickStart:<https://helm.sh/docs/intro/quickstart/>

sudo snap install helm --classic

helm version

### Tools

helm repo add bitnami https://charts.bitnami.com/bitnami

helm repo update

helm search repo bitnami

#Can view many big tools are available

Charts:

<https://github.com/bitnami/charts/tree/master/bitnami>

Nginx -<https://github.com/bitnami/charts/tree/master/bitnami/nginx/templates>

helm install test-nginx bitnami/nginx

k get deploy test-nginx

k get pod

helm uninstall test-nginx

k get deploy

### Application

Traditional Method: without using Helm

<https://github.com/IBM/guestbook/tree/master/v2>

<https://github.com/IBM/guestbook/blob/master/v2/guestbook-deployment.yaml>

Helm way:

<https://github.com/IBM/helm101/tree/master/charts/guestbook>

[https://github.com/IBM/helm101/blob/masthttps://github.com/IBM/helm101/blob/master/charts/guestbook/templates/guestbook-deployment.yamler/charts/guestbook/templates/guestbook-deployment.yaml](https://github.com/IBM/helm101/blob/master/charts/guestbook/templates/guestbook-deployment.yaml)

git clone https://github.com/IBM/helm101.git

cd helm101/charts

helm install guestbook-demo ./guestbook/

k get deploy

k get svc

k get pod

To Learn - Helm chart development:

<https://helm.sh/docs/chart_template_guide/getting_started/>

Helm History

1637 cd workloads/

1638 cat live.yaml

1639 k explain pod.spec.containers.livenessProbe

1640 cat live.yaml

1641 k apply -f live.yaml

1642 k get pod liveness-exec -w

1643 cat live.yaml

1644 k explain pod.spec.containers.livenessProbe

1645 sudo ls /etc/kubernetes/manifests/

1646 sudo cat /etc/kubernetes/manifests/kube-apiserver.yaml

1647 cat readiness.yaml

1648 k apply -f readiness.yaml

1649 k get pod readiness-exec -w

1650 sudo cat /etc/kubernetes/manifests/kube-apiserver.yaml

1651 k explain pod.spec.containers.livenessProbe

1652 kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml

1653 k get pods -n kube-system

1654 wget -c https://gist.githubusercontent.com/initcron/1a2bd25353e1faa22a0ad41ad1c01b62/raw/008e23f9fbf4d7e2cf79df1dd008de2f1db62a10/k8s-metrics-server.patch.yaml

1655 cat k8s-metrics-server.patch.yaml

1656 kubectl patch deploy metrics-server -p "$(cat k8s-metrics-server.patch.yaml)" -n kube-system

1657 k get rs -n kube-system

1658 k get pods -n kube-system

1659 k top pods

1660 k top --help

1661 k top pod --help

1662 k top pod --containers=true

1663 k top pods

1664 k top pods --sort-by cpu

1665 k top pods -A --sort-by cpu

1666 k top pods -A --sort-by memory

1667 k top nodes

1668 k top nodes --sort-by cpu

1669 k edit deploy metrics-server -n kube-system

1670 k get deploy

1671 kubectl autoscale deploy nginx-deployment --min=3 --max=5 --cpu-percent=40

1672 kubectl autoscale --help

1673 k get deploy

1674 k get hpa nginx-deployment -o yaml

1675 history

1676 helm list

1677 helm repo ls

1678 helm repo --help

1679 helm repo list

1680 helm repo remove bitnami

1681 helm

1682 helm repo ls

1683 helm repo list

1684 helm repo add bitnami https://charts.bitnami.com/bitnami

1685 helm repo list

1686 helm repo update

1687 helm search repo bitnami

1688 helm install dev-nginx bitnami/nginx

1689 k get deploy

1690 alias k=kubectl

1691 k get deploy

1692 k get svc

1693 cd workloads/

1694 vi stag.yaml

1695 vi prod.yaml

1696 cat stag.yaml

1697 cat prod.yaml

1698 helm install stag-nginx bitnami/nginx --values stag.yaml

1699 k get deploy

1700 helm install prod-nginx bitnami/nginx --values prod.yaml

1701 k get deploy

1702 helm list

1703 cp stag.yaml dev.yaml

1704 cat dev.yaml

1705 helm list

1706 helm upgrade dev-nginx bitnami/nginx --values dev.yaml

1707 k get deploy

1708 helm list

1709 helm rollback dev-nginx 1

1710 helm list

1711 k get deploy

1712 helm uninstall dev-nginx

1713 helm list

1714 git clone https://github.com/IBM/helm101.git

1715 cd helm101/charts/guestbook/

1716 ls

1717 cd ..

1718 ls

1719 cat dev.yaml

1720 vi prod.yaml

1721 ls

1722 helm install prod-guesbook ./guestbook/

1723 k get deploy

1724 helm install prod-guesbook ./guestbook/ -n test

1725 k get deploy -n test

1726 helm list

1727 helm list -A