kubernetes

KUBERNETES(K8S)

Orcsoft Internal Training #5

docker

ORCSOFT

AGENDA

Container concept

Introduction to Kubernetes

System Architecture

Fundamental of Kubernetes

FUNDAMENTAL OF KUBERNETES

Pods, Container and Services

Daemon Sets and Replication Controller (RC)

Deployment/Replica-Set (RS) and Rolling update

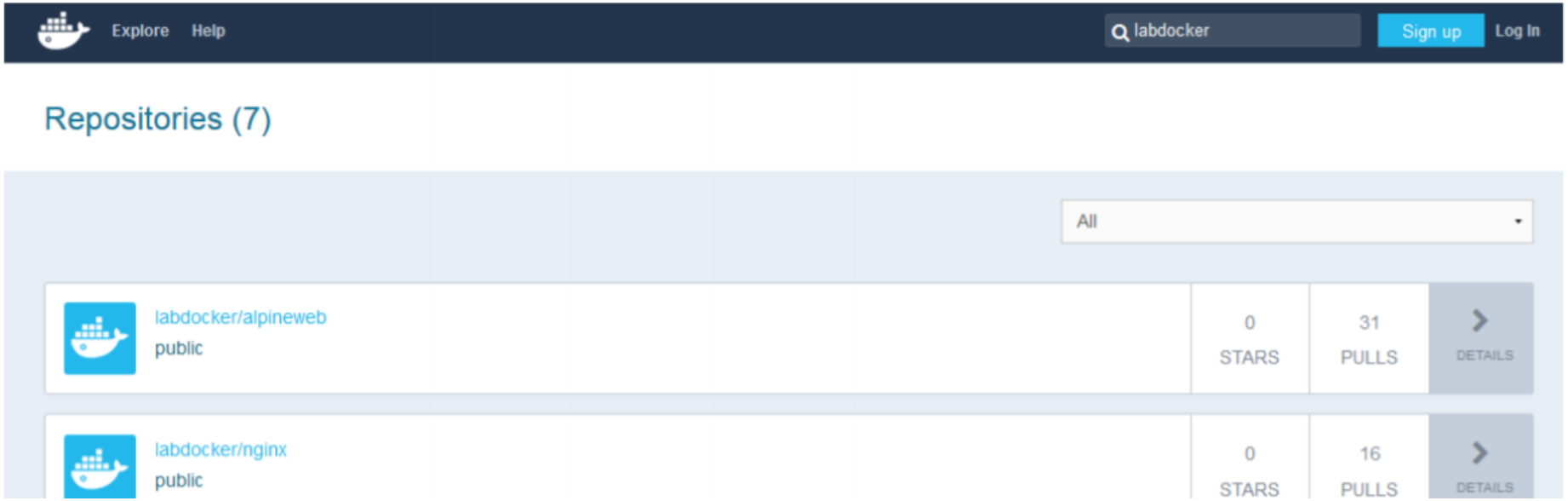
Liveness and Readiness

Config Map and Secret

Job And Cron Job

Horizontal Pods Autoscaling (HPA)

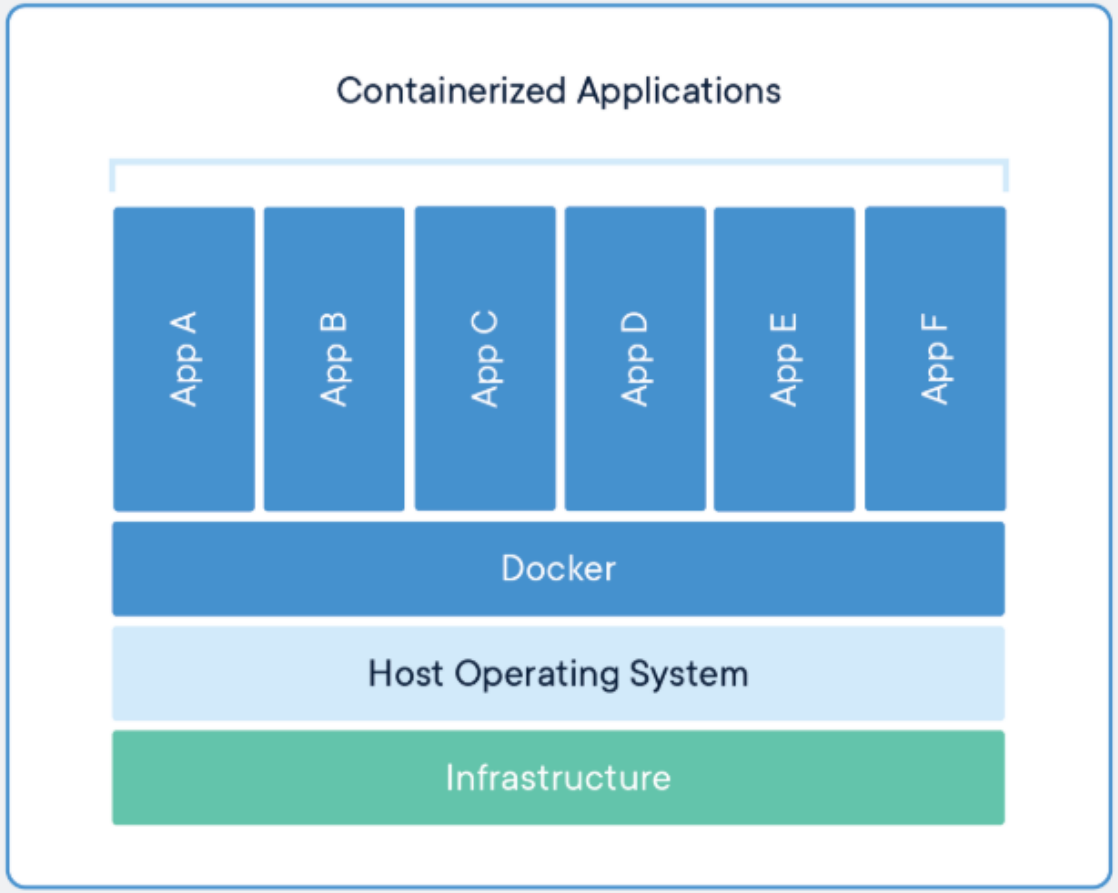
- Repository for lab

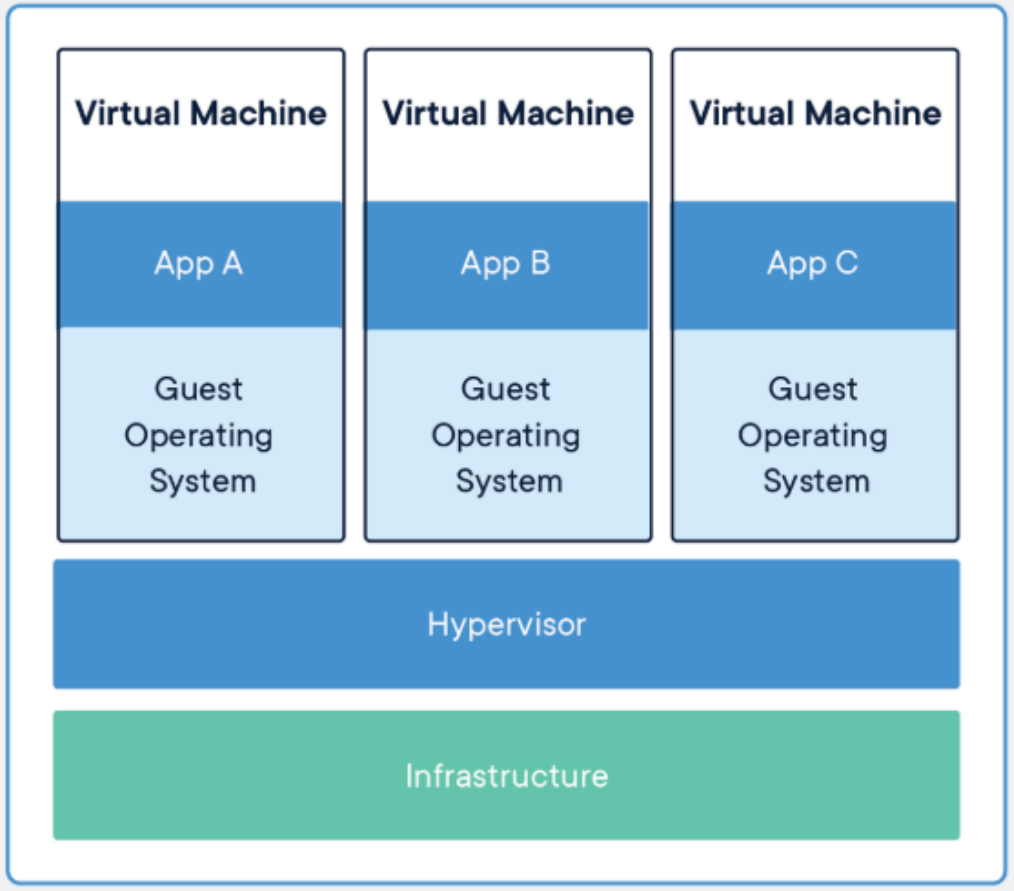


LAB RESOURCE

CONTAINER CONCEPT

VIRTUALIZATION มีข้อจำกัด





ภาพประกอบจาก https://www.docker.com/resources/what-container

ตัวอย่างการใช้งาน CONTAINER

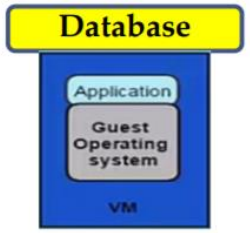
- Production Environment
- Day 1: Application 1: Implement

Web Server

- Apache 2.20 Web Server
- PHP 5.5 Engine
- Larevel 4.1 Framework

Web Service

- IIS 8
- .Net FrameWork 3.5

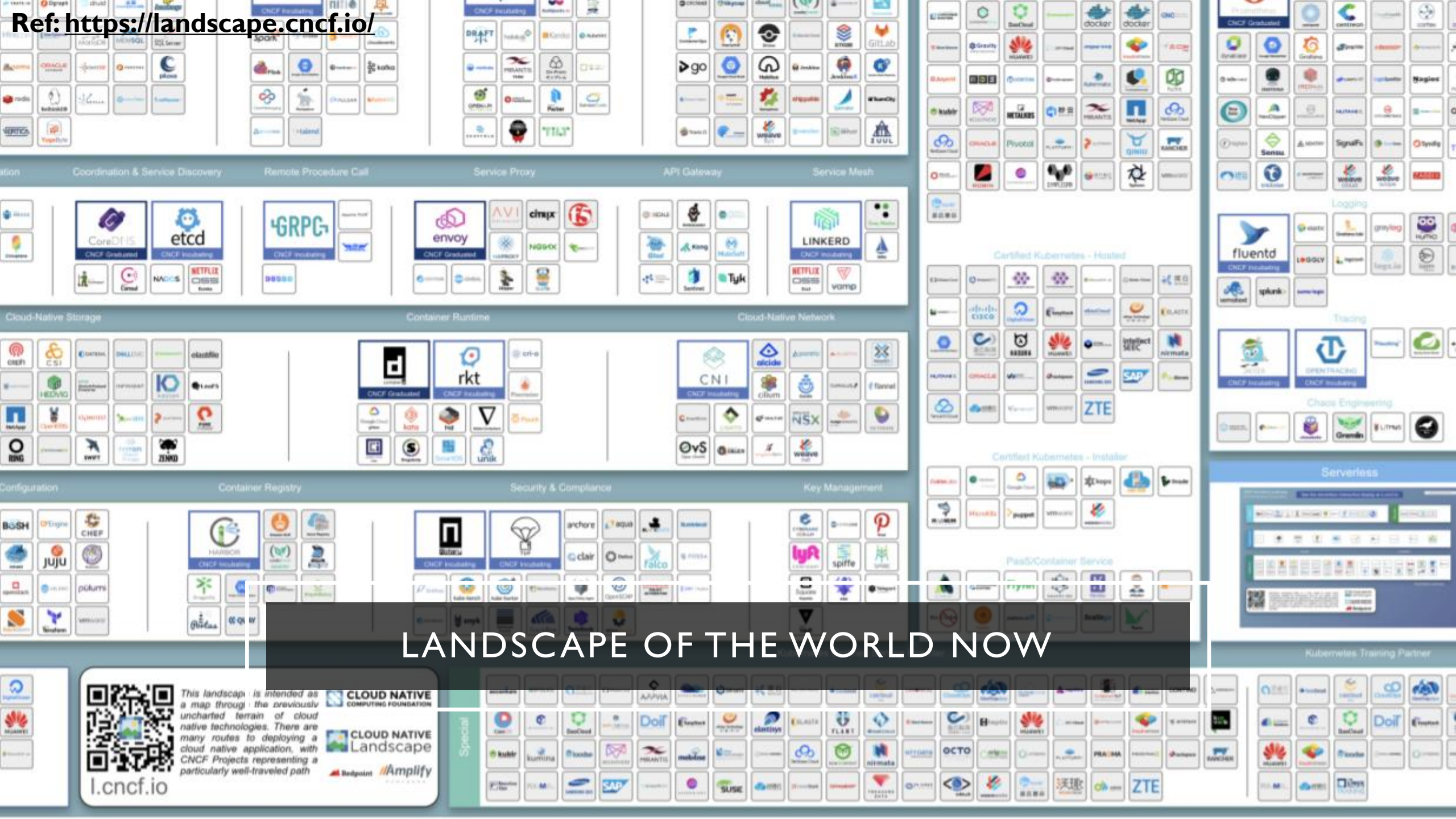


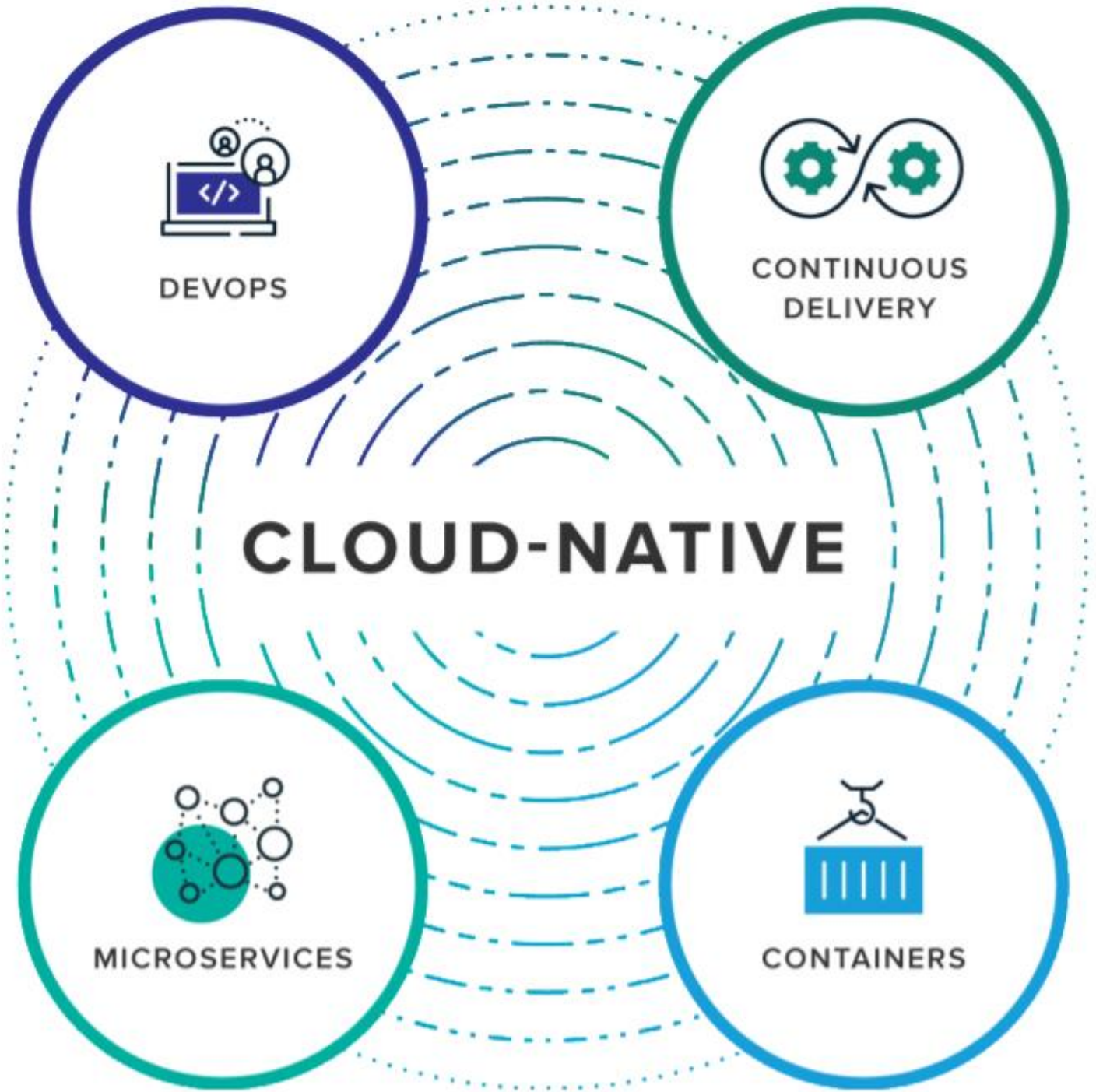
Database

- MariaDB 5.1

- Day 2: Application 2: Need to implement
  - Need PHP 7.0 ?
  - MariaDB 10.1.14 (Need search feature on 10.1)
- So… The problem still exist.

Ref: https://landscape.cncf.io/





LANDSCAPE OF THE WORLD NOW

INTRODUCTION TO KUBERNETES

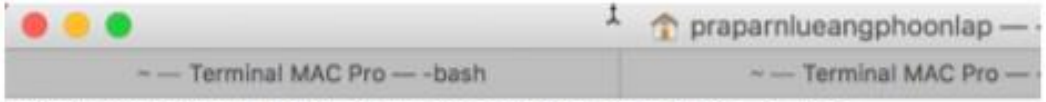
KUBERNETES(K8S) ?

Production-Grade Container Orchestration: *Automated container deployment, scaling, and management*

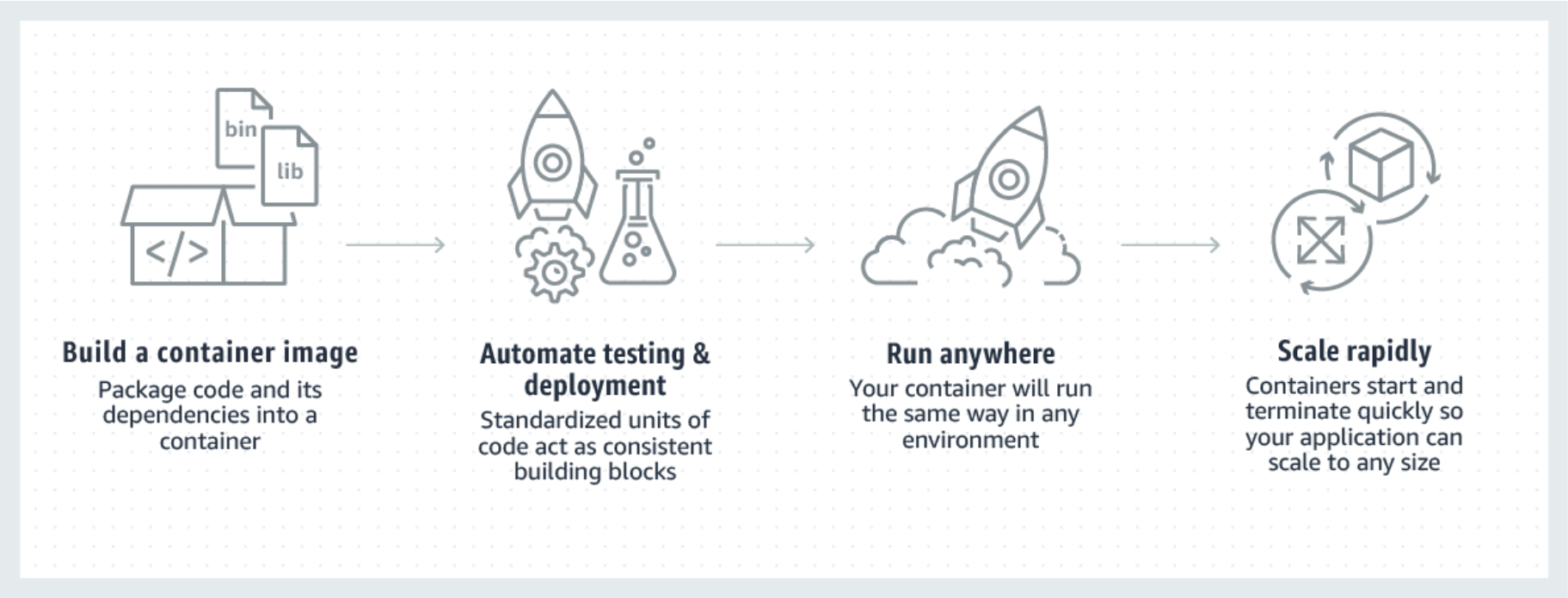
INTRODUCTION TO KUBERNETES

Check Kubernetes version

```
kubectl get nodes -o yaml
```




```
praparns-MacBook-Pro:~ praparn$ kubectl get nodes -o yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: Node
  metadata:
    annotations:
      node.alpha.kubernetes.io/ttl: "0"
      volumes.kubernetes.io/controller-managed-attach-detach: "true"
    creationTimestamp: 2017-06-24T09:03:07Z
    labels:
      beta.kubernetes.io/arch: amd64
      beta.kubernetes.io/os: linux
      kubernetes.io/hostname: minikube
    name: minikube
    namespace: ""
    resourceVersion: "15676"
    selfLink: /api/v1/nodesminikube
    uid: f03de16d-58bb-11e7-aae1-080027559511
  spec:
    externalID: minikube
  nodeInfo:
    architecture: amd64
    bootID: cdf78a28-072d-4707-8c7d-dbaa0b95fff7
    containerRuntimeVersion: docker://1.11.1
    kernelVersion: 4.9.13
    kubeProxyVersion: v1.6.0
    kubeletVersion: v1.6.0
    machineID: ef85c4ce9a23440e83266debd5cc9856
    operatingSystem: linux
    osImage: Buildroot 2017.02
    systemUUID: 801D91BA-D487-47D2-9C5D-C12F278B49C5
kind: List
metadata: {}
resourceVersion: ""
selfLink: ""
```



ORCHESTRATION = CONTAINER + CLUSTER

What is CNCF?

CNCF is an open source software foundation dedicated to making cloud native computing universal and sustainable. Cloud native computing uses an open source software stack to deploy applications as microservices, packaging each part into its own container, and dynamically orchestrating those containers to optimize resource utilization. Cloud native technologies enable software developers to build great products faster.

JOIN

Projects

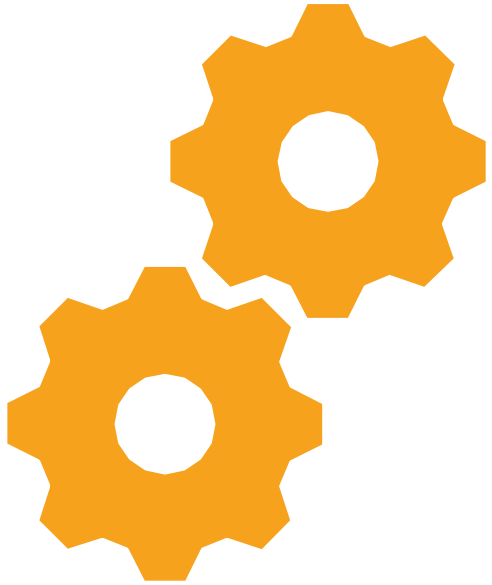
We host and nurture components of cloud native software stacks, including Kubernetes, Prometheus and Envoy. Kubernetes and other CNCF projects are some of the **highest velocity projects** in the history of open source. We are regularly adding new projects to better support a full stack cloud native environment.

INTRODUCTION TO KUBERNETES

- But why Container Orchestration ?
  - Container Platform is next generation that partner/beat existing virtualize technology
    - Easy to build-ship-run on dev and production
    - Zero Configure
    - Suitable for Microservice architecture
    - Good for application, Good for business (Win/Win !!!)

KEY FEATURE

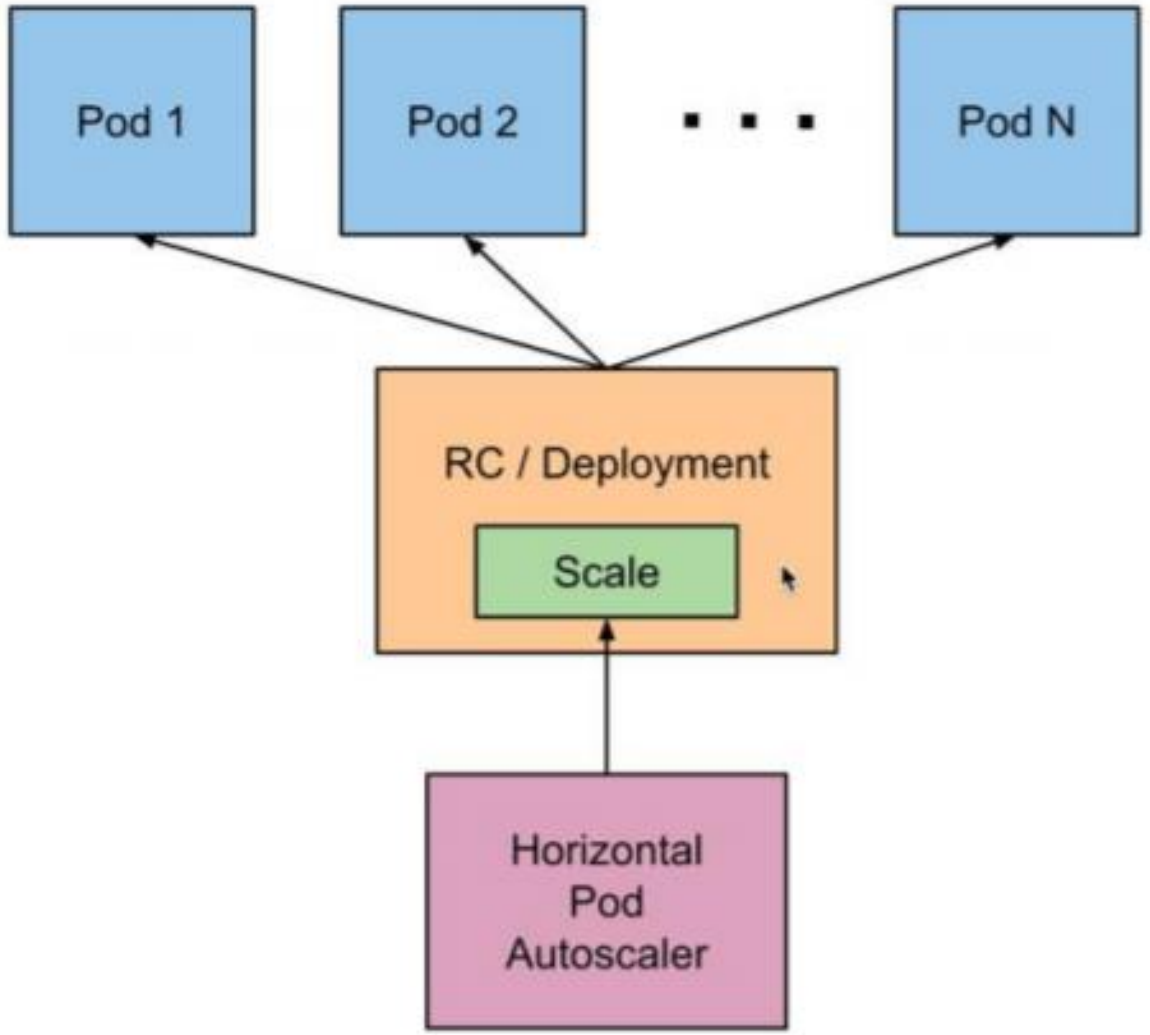
- Automatic binpacking
- Horizontal Pod Autoscaling (HPA)
- Automated rollouts and rollbacks
- Storage orchestration
- Self-healing
- Service discovery and load balancing
- Secret and configuration management
- Batch execution



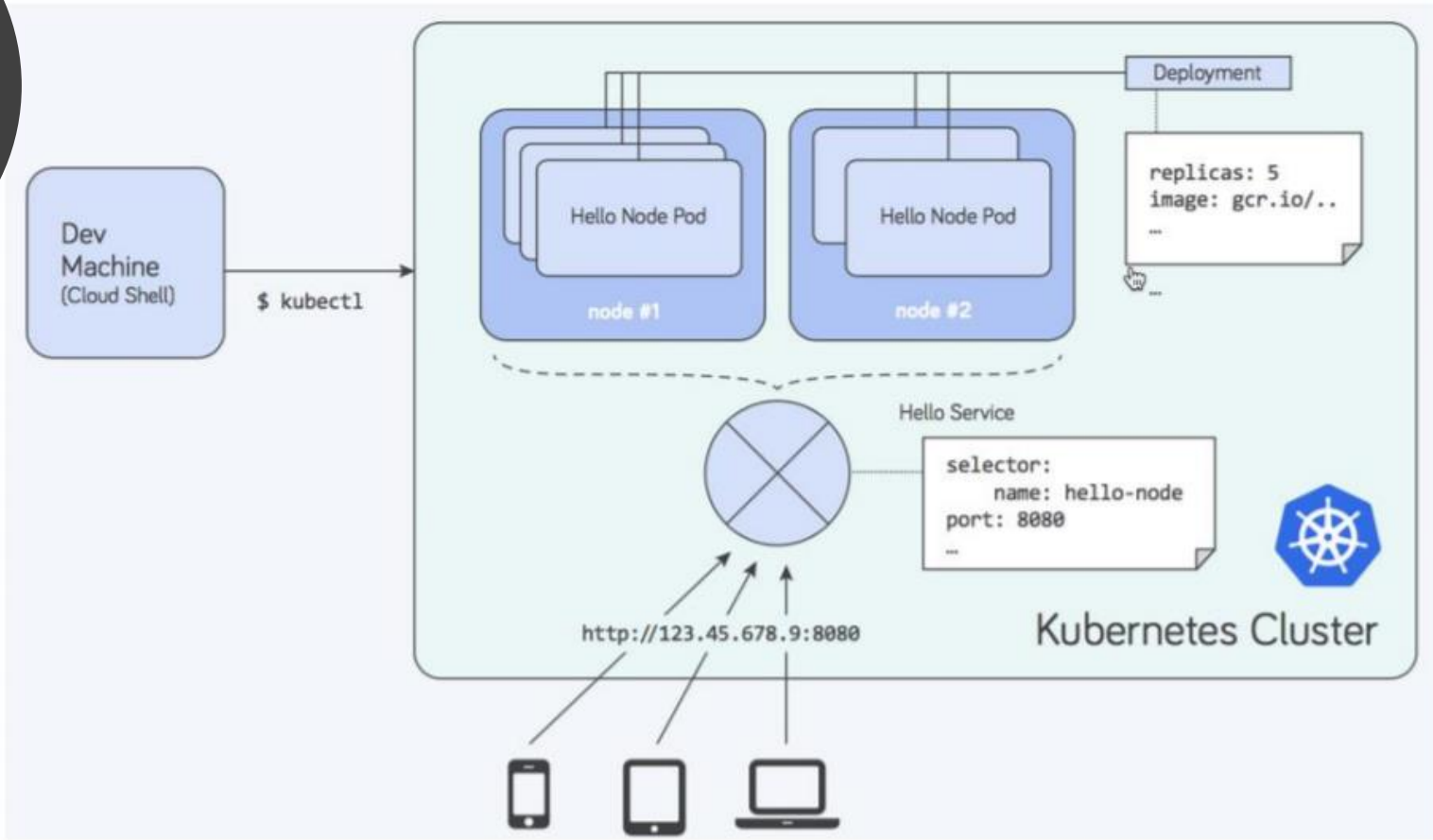
AUTOMATIC BINPACKING

Check available node resource by command: kubectl describe node

HORIZON PODS AUTOSCALING

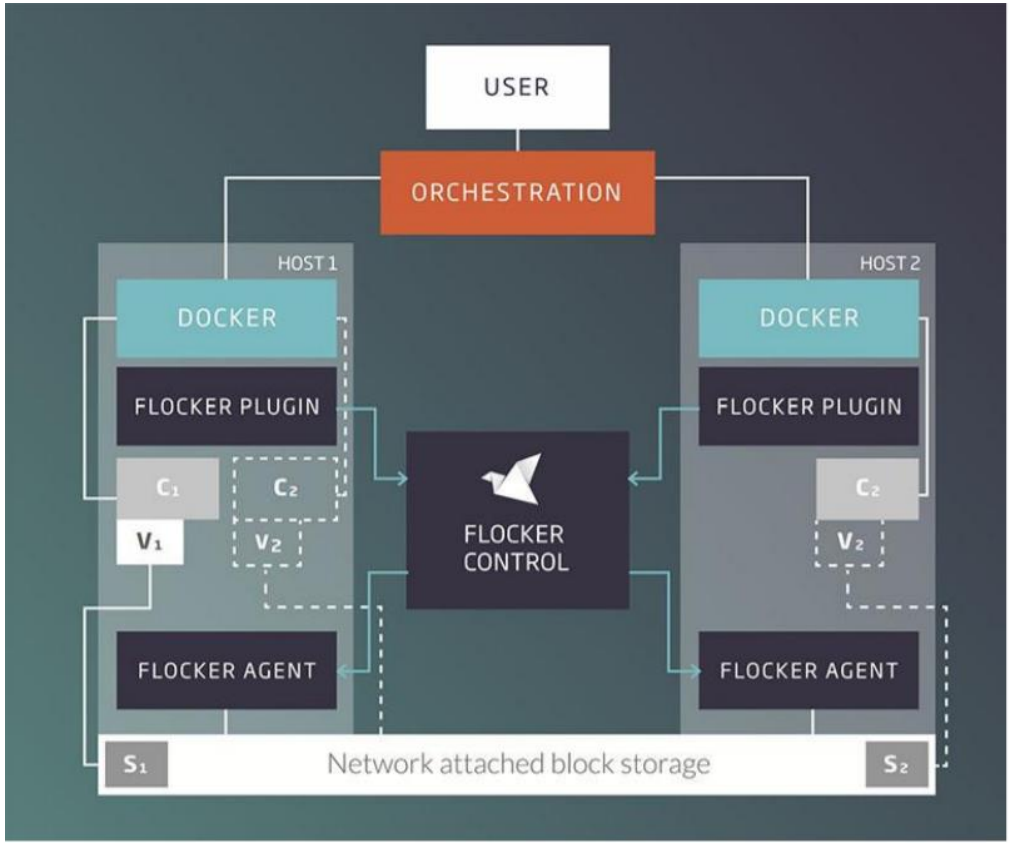


AUTOMATED ROLLOUTS AND ROLLBACKS



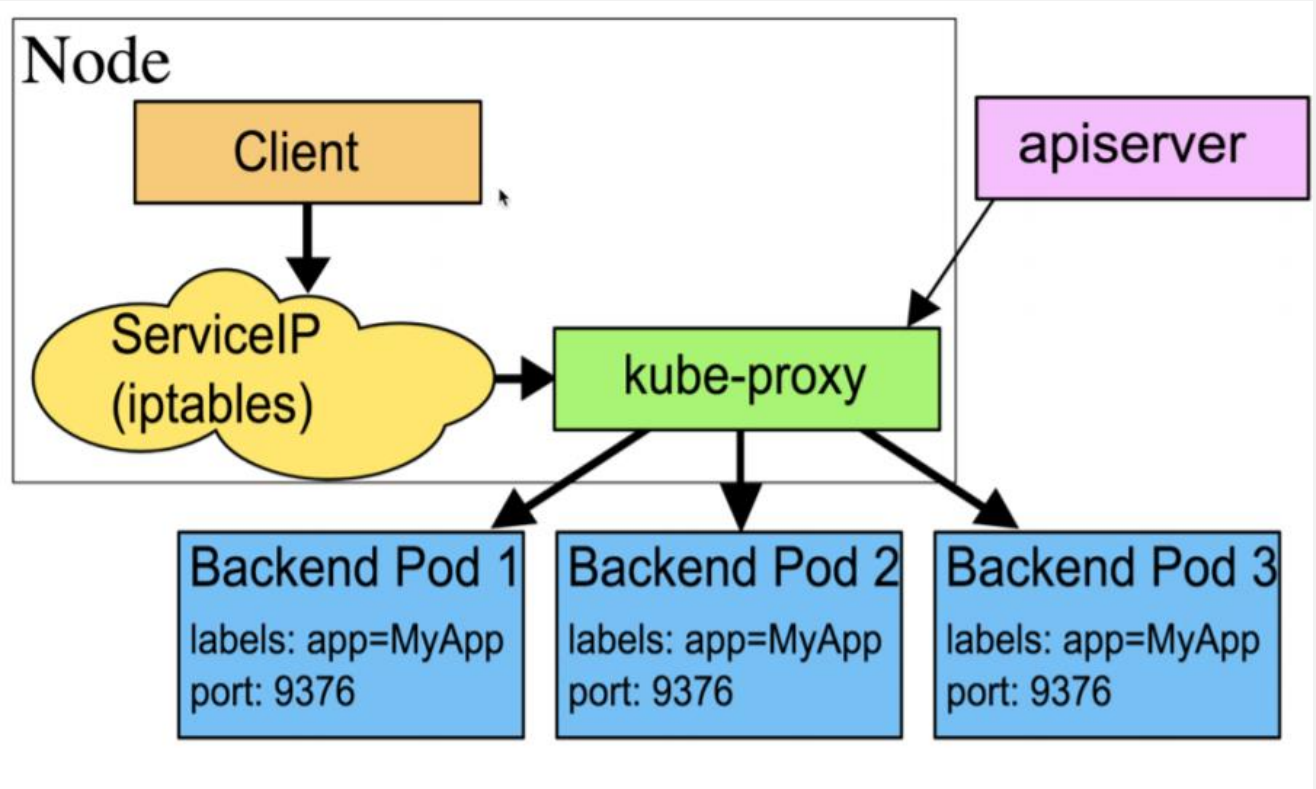
STORAGE ORCHESTRATION

- **Support several storage type:**
  - **Local Storage**
  - **Network Storage (NFS, iScsi, Gluster, Ceph, Cinder, Flocker)**
  - **Cloud Storage (AWS, GCE, Azue Disk etc)**



SELF-HEALING

- Replication Controller (RC) will maintain unit of Pods as design (not to much (kill) and not to few (create)
- Full-fill Pods on every failure case with automatic by system



SERVICE DISCOVERY AND LOAD BALANCING

- Service will act like "connector" for client need to connect with Pods
- Discovery will use for service to look "Pods" by environment variable or dns service
- Support load balancing between multiple Pods (replica)

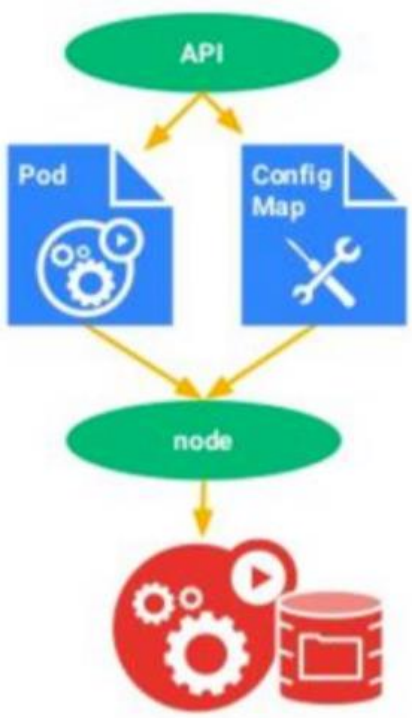
SECRET AND CONFIGURATION MANAGEMENT

```
$ kubectl get secrets
NAME                  TYPE                                  DATA      AGE
db-user-pass          Opaque                                2         51s

$ kubectl describe secrets/db-user-pass
Name:            db-user-pass
Namespace:       default
Labels:          <none>
Annotations:     <none>

Type:            Opaque

Data
====
password.txt:    12 bytes
username.txt:    5 bytes
```



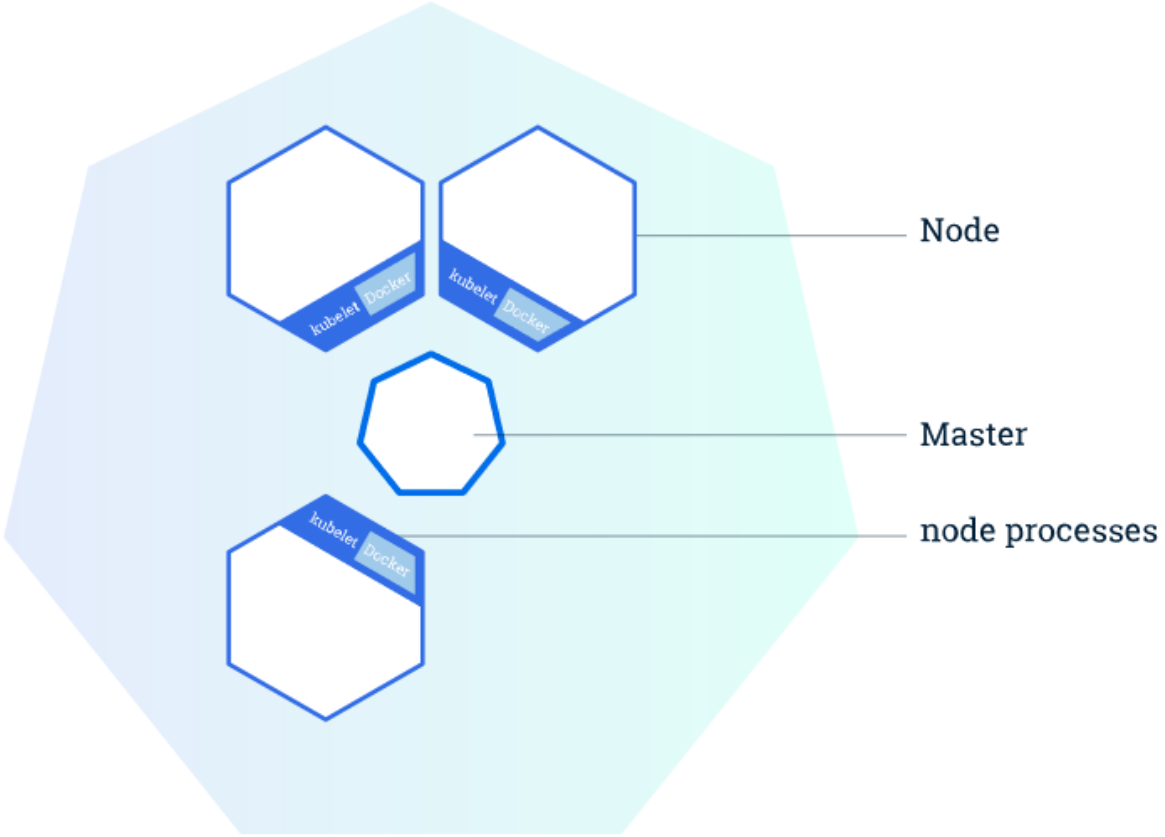
ConfigMap

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: dragon-config
labels:
  environment: non-prod
data:
  dragon.how.much: very
  dragon.type: fast
```

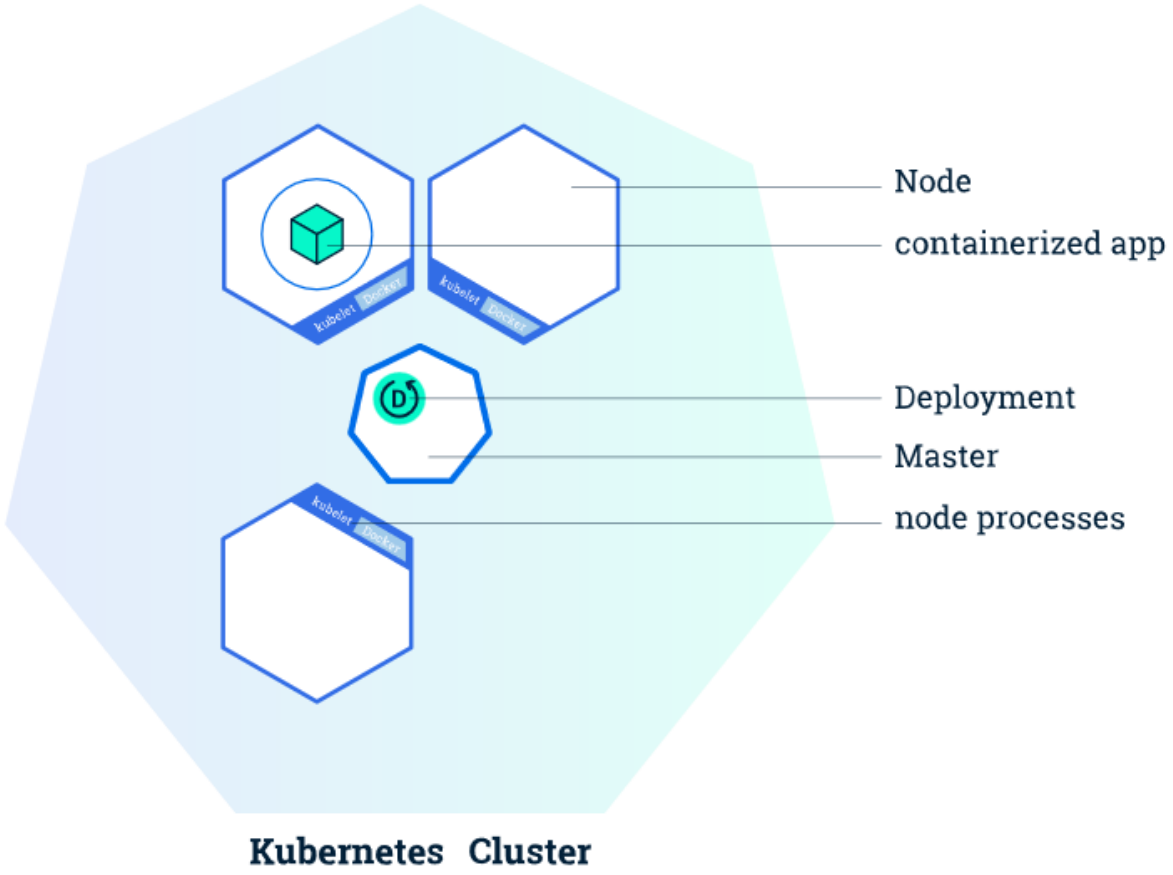
```yaml
apiVersion: v1
kind: Pod
metadata:
 name: dragon-pod
spec:
 containers:
  - name: dragon-container
    image: dragon-image
    env:
     - name: DRAGON_LEVEL
       valueFrom:
        configMapKeyRef:
         name: dragon-config
         key: dragon.how.much
     - name: DRAGON_TYPE
       valueFrom:
        configMapKeyRef:
         name: dragon-config
         key: dragon.type
```

BATCH EXECUTION

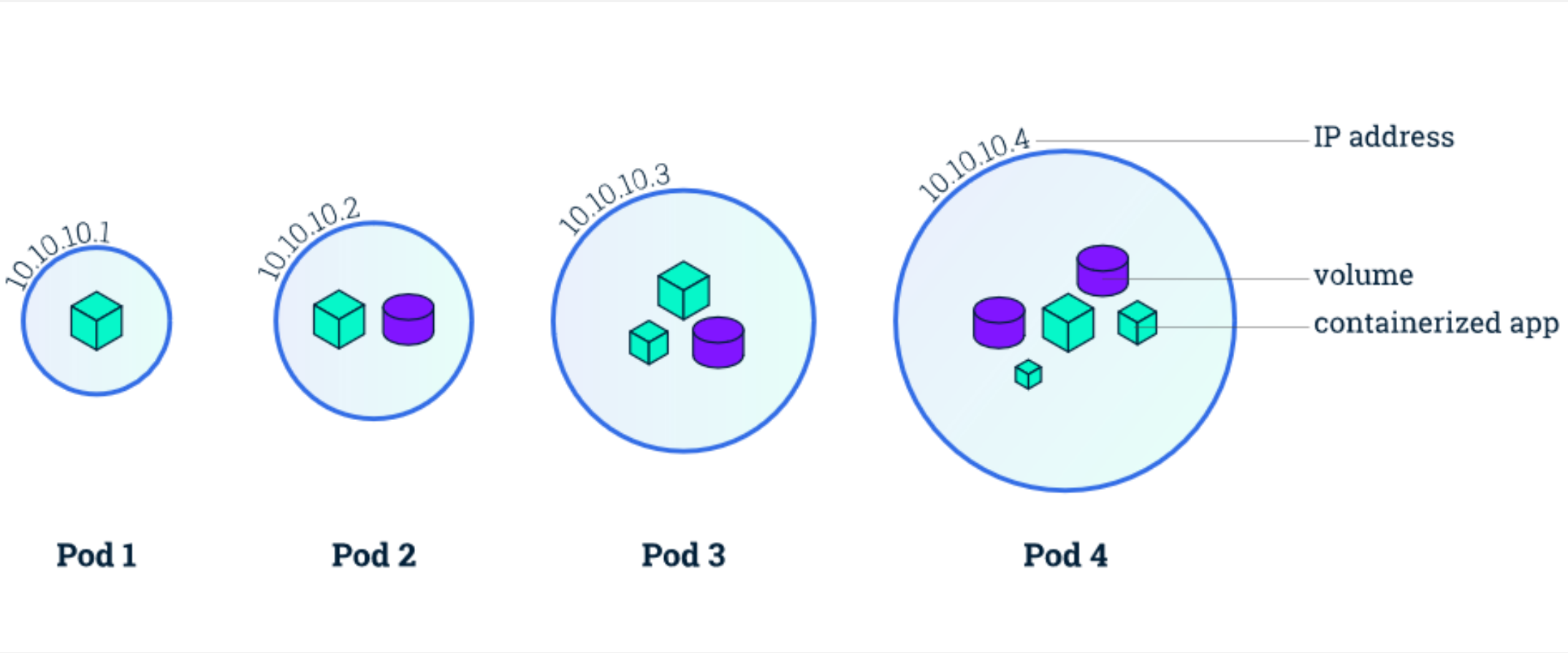
SYSTEM ARCHITECTURE

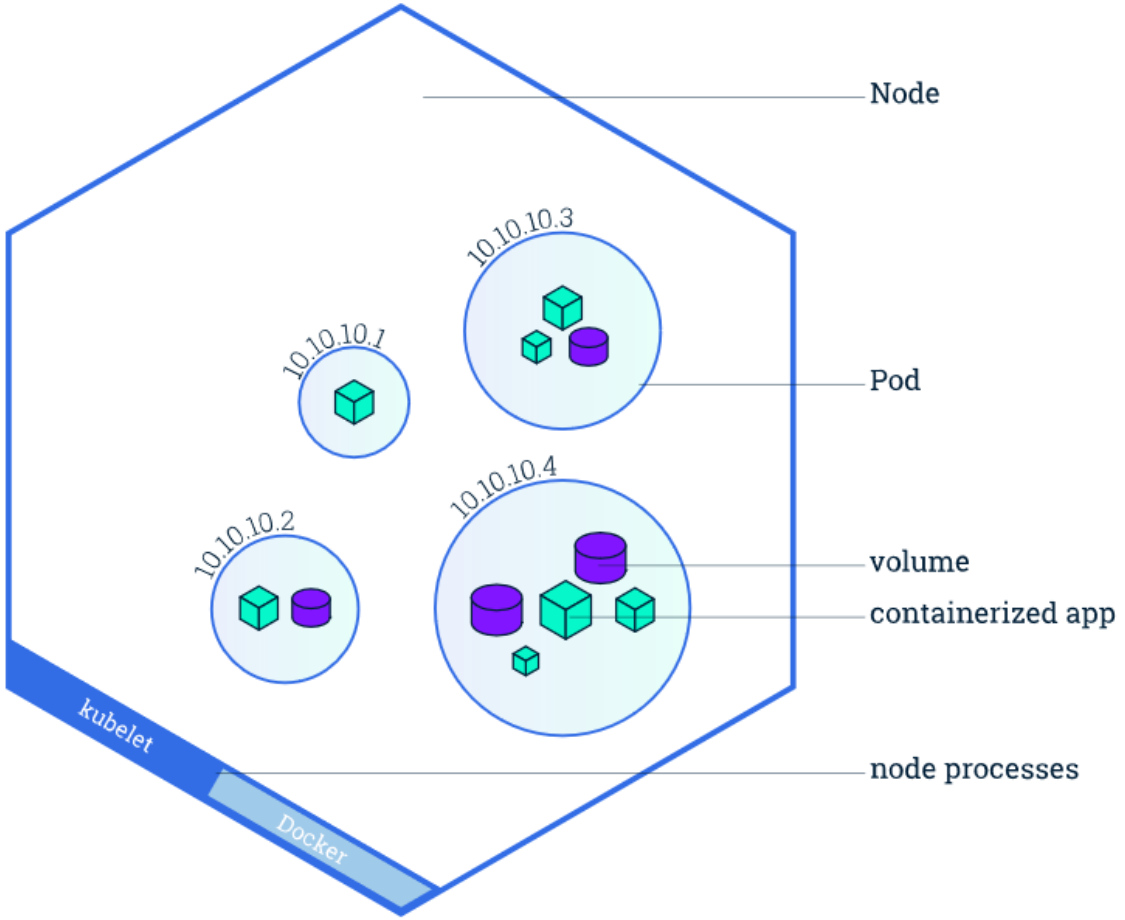


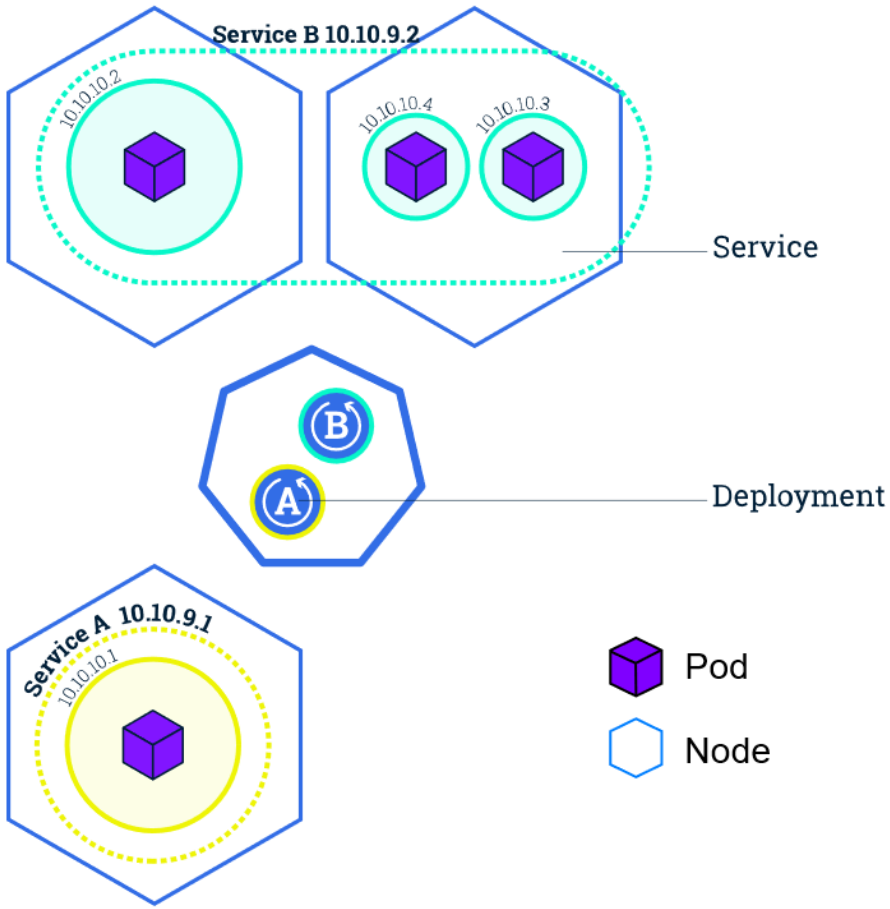
Kubernetes cluster

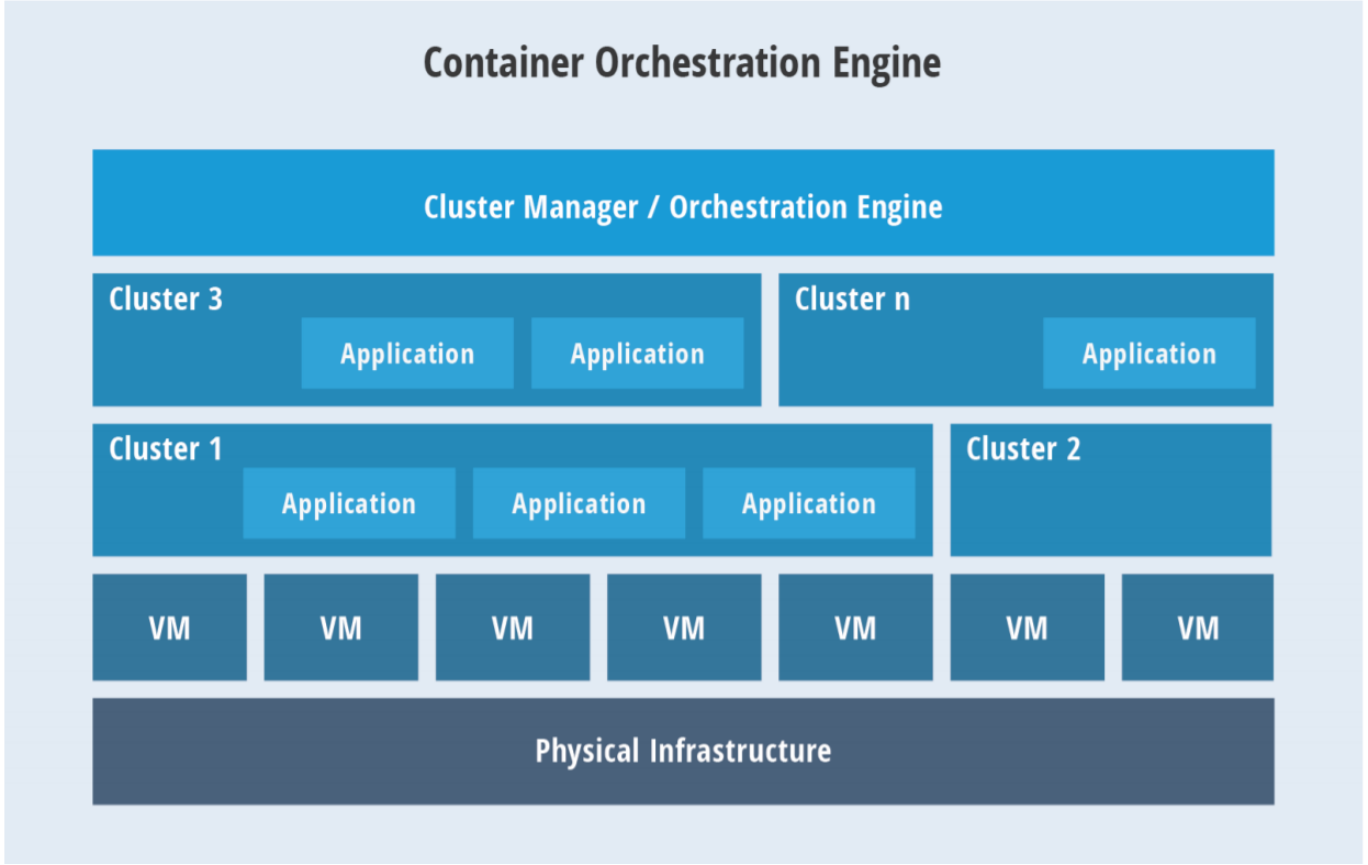


Kubernetes Cluster


10.10.10.1
10.10.10.2
10.10.10.3
10.10.10.4
IP address
volume
containerized app
Pod 1
Pod 2
Pod 3
Pod 4


Node
10.10.10.3
10.10.10.1
Pod
10.10.10.4
10.10.10.2
volume
containerized app
kubelet
Docker
node processes


Service B 10.10.9.2
10.10.10.2
10.10.10.4
10.10.10.3
Service
B
A
Deployment
Service A 10.10.9.1
10.10.10.1
Pod
Node


Container Orchestration Engine
Cluster Manager / Orchestration Engine
Cluster 3
Application
Application
Cluster n
Application
Cluster 1
Application
Application
Application
Cluster 2
VM
VM
VM
VM
VM
VM
VM
Physical Infrastructure

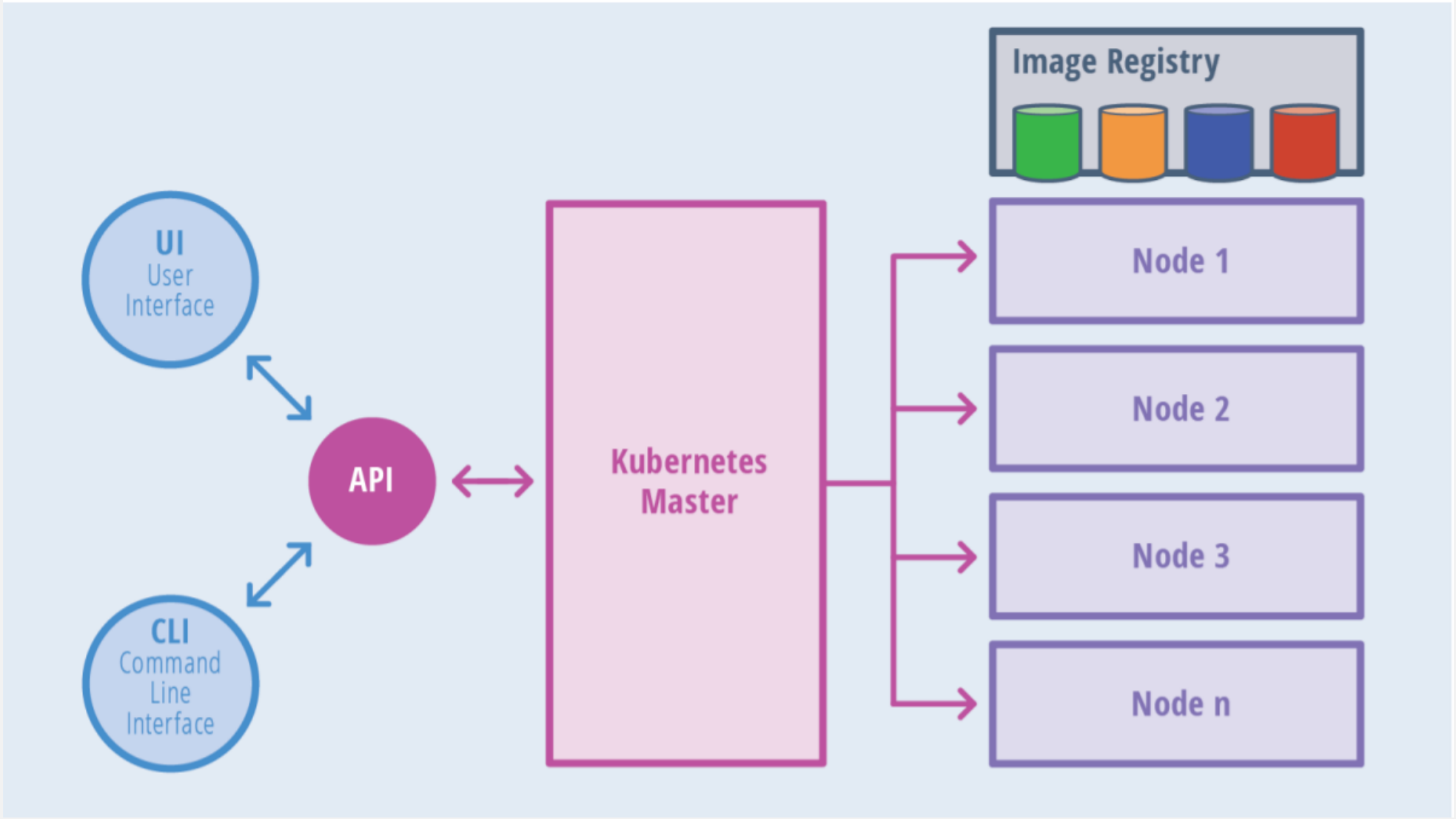
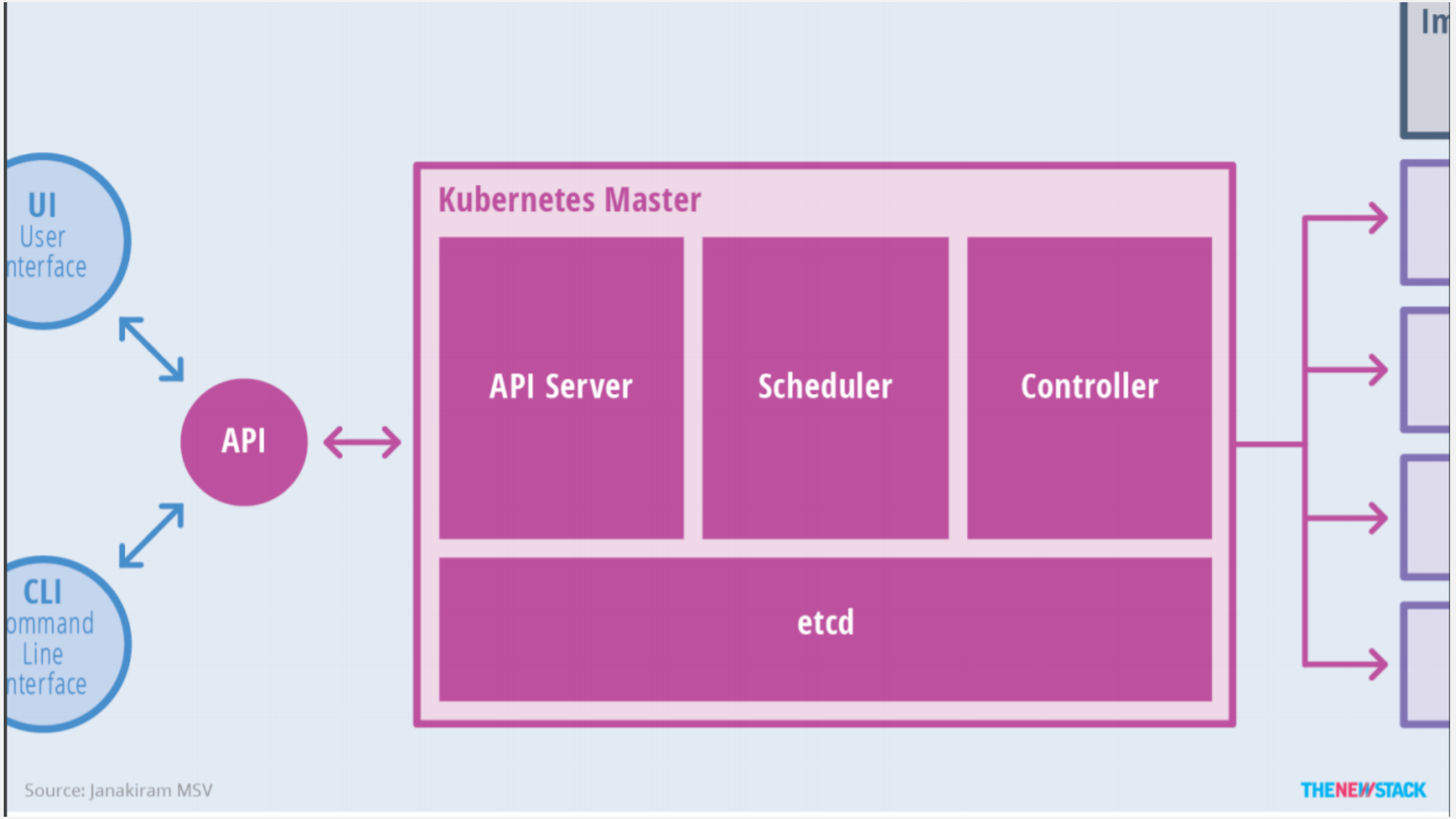
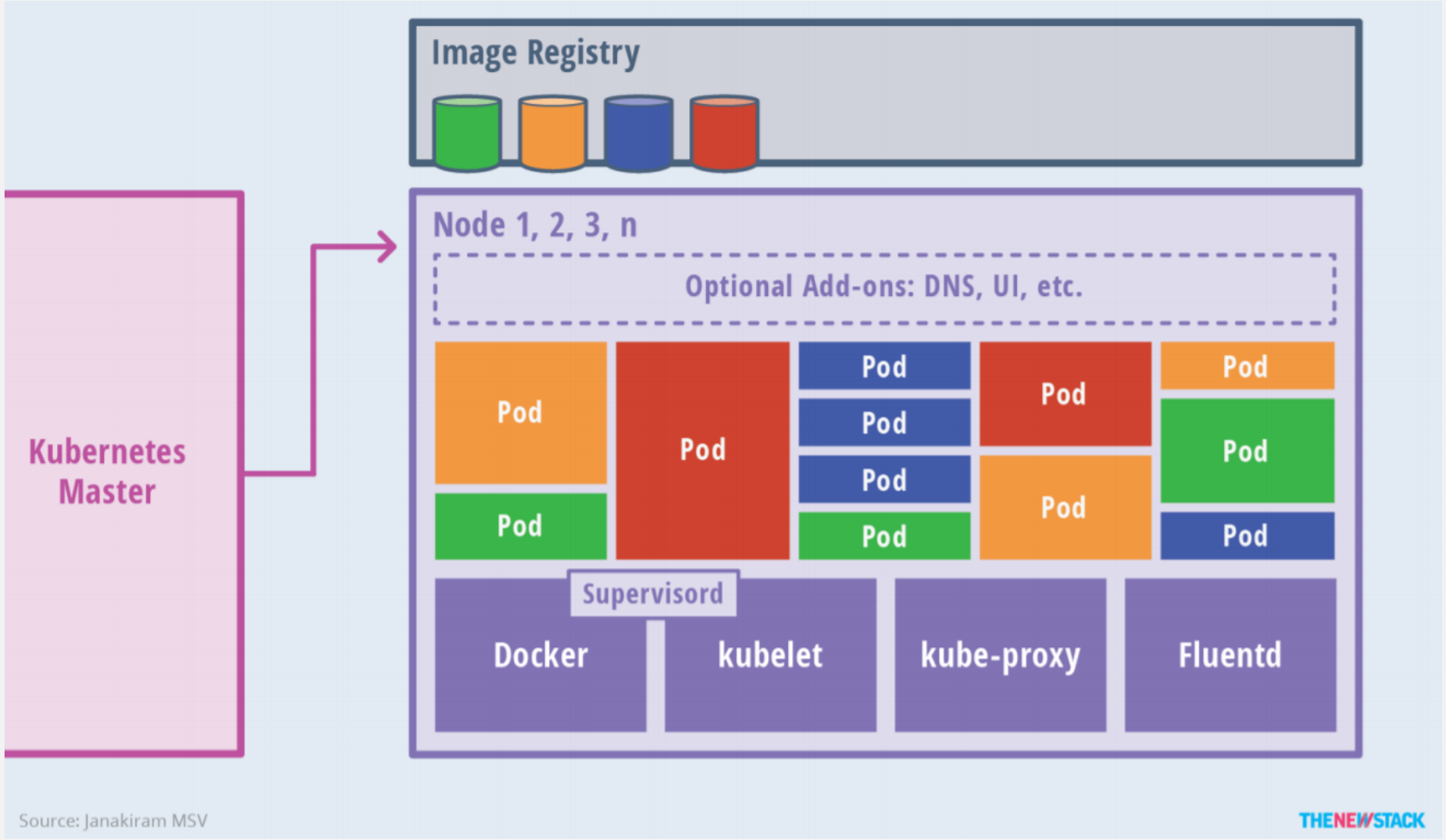


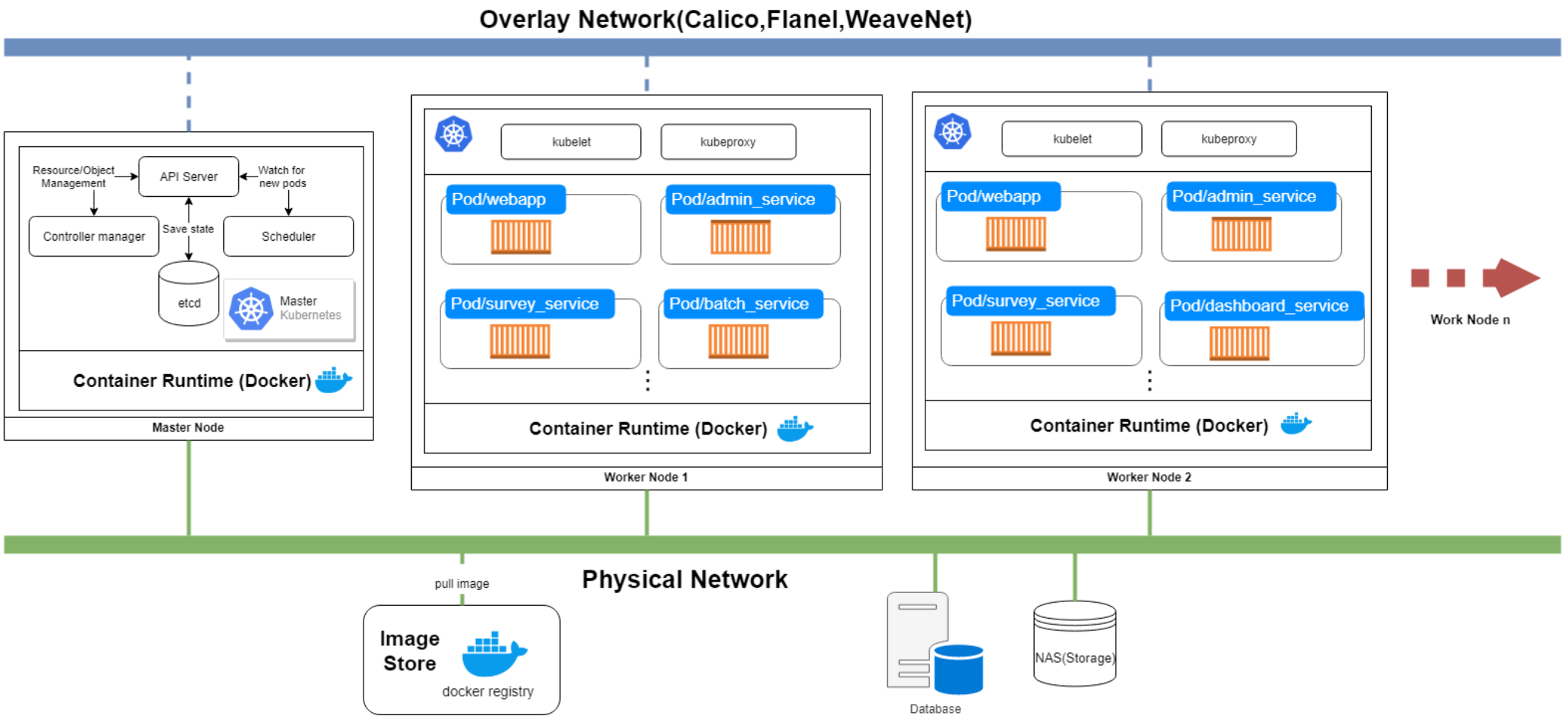
Image Registry
UI
User
Interface
API
CLI
Command
Line
Interface
Kubernetes
Master
Node 1
Node 2
Node 3
Node n


UI
User
nterface
API
CLI
ommand
Line
nterface
Kubernetes Master
API Server
Scheduler
Controller
etcd
Source: Janakiram MSV
THENEWSTACK

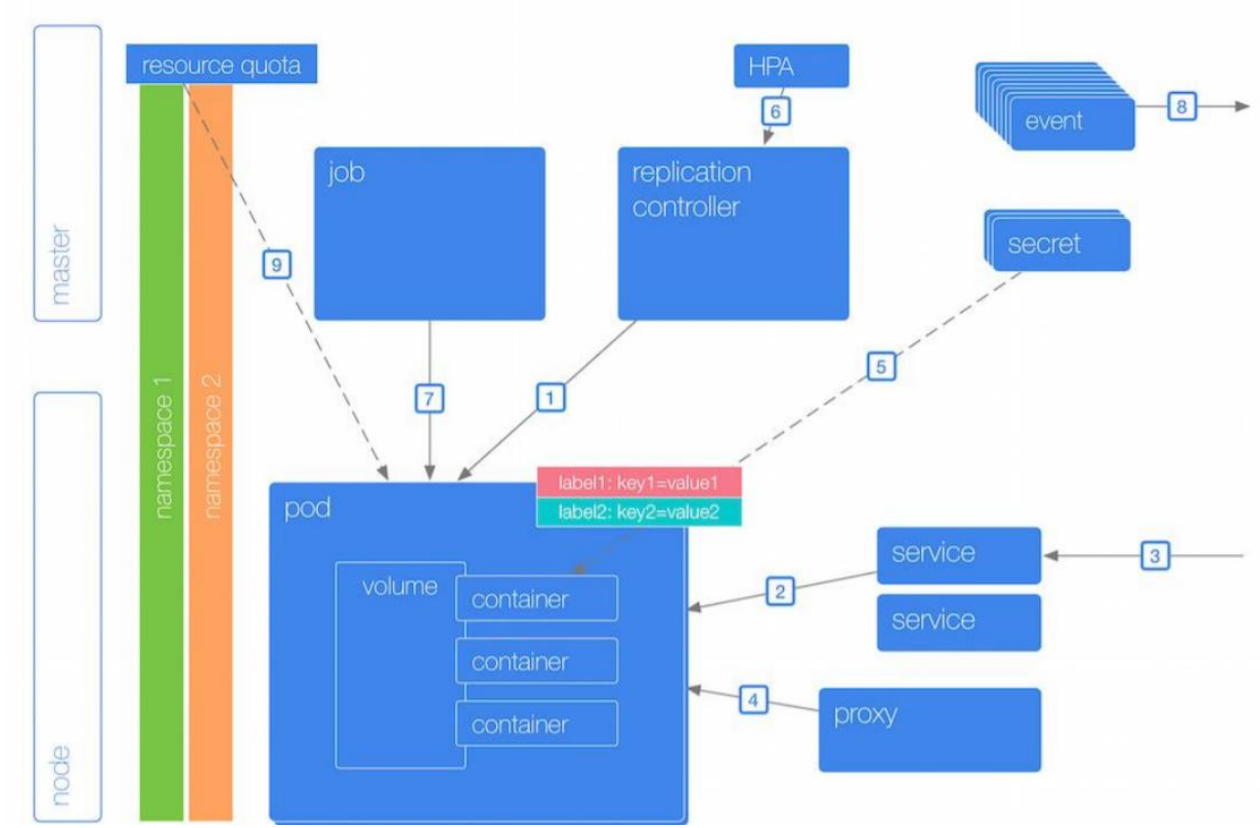


Source: Janakiram MSV

THENEWSTACK


Overlay Network(Calico,Flanel,WeaveNet)
Resource/Object Management
API Server
Watch for new pods
Controller manager
Save state
Scheduler
etcd
Master Kubernetes
Container Runtime (Docker)
Master Node
kubelet
kubeproxy
Pod/webapp
Pod/admin_service
Pod/survey_service
Pod/batch_service
Container Runtime (Docker)
Worker Node 1
kubelet
kubeproxy
Pod/webapp
Pod/admin_service
Pod/survey_service
Pod/dashboard_service
Container Runtime (Docker)
Worker Node 2
Work Node n
Physical Network
pull image
Image Store
docker registry
Database
NAS(Storage)

SYSTEM ARCHITECTURE




```
"labels": {
  "key1" : "value1",
  "key2" : "value2"
}
```

```
"release" : "stable", "release" : "canary"
```

```
"environment" : "dev", "environment" : "qa", "environment" : "production"
```

```
"tier" : "frontend", "tier" : "backend", "tier" : "cache"
```

```
"partition" : "customerA", "partition" : "customerB"
```

```
"track" : "daily", "track" : "weekly"
```

SYSTEM ARCHITECTURE

Topic	K8S	Docker/Swarm
Architecture	Open-system (Base on cluster manager "Borg" for support complex workload)	**Swarm**: Proprietary of Docker product, "Easy to use", "Extend capability of Docker in cluster"
Operation command	Almost operate by "YMAL" file (Declarative Command)	Almost operate by "command" (Imperative Command)
Unit of Work	Pods (Pods >= Container)	Container
How to Identify Work	"Label operation"	**Docker**: By container name **Swarm**: By service/stack name
Level of workload management	Service Level: (Simple) Replication Level: (Auto healing) Deployment Level: (Auto healing + Roll Update)	**Docker**: N/A **Swarm**: Service Level (Snag with service/stack)
Auto scaling	HPA (Horizontal Pods Scaling) base on CPU	No
Health check	Liveness & Readiness (Multi option to check application health)	Service health only

FUNDAMENTAL OF KUBERNETES

PODS, CONTAINER AND SERVICES

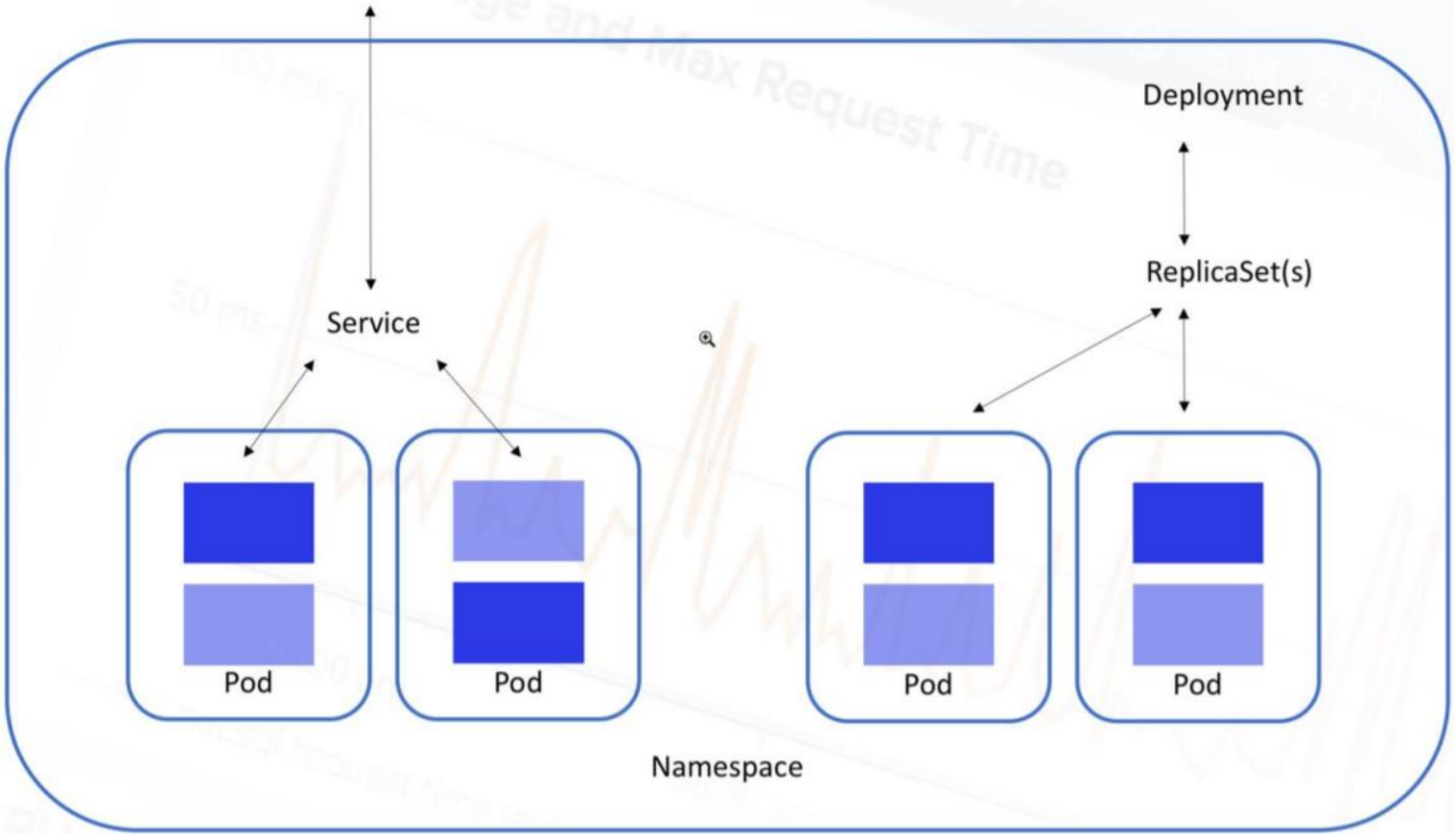
PODS, CONTAINER AND SERVICES

- Pods vs Container
  - Docker's view point:
    - 1 Container: 1 Application, 1 Component of Microservice
    - So for micro service we need multi container
      - Cache component
      - Web component
      - Database component
      - Etc
  - Kubernetest's view point:
    - 1 Pods = 1 Container
    - 1 Pods = N Container (Container on the same context, Work closely)
    - So we can have 1 Pods for container more than 1 container

PODS, CONTAINER AND SERVICES

- Pods vs Container
  - All container on same Pods will share:
    - Process ID (PID)
    - Network access (Communicate to each other via "localhost")
    - Internal Process Command (IPC)
    - Unix Time-Sharing (UTS)
    - Hostname
    - IP Address/Ports
  - Use Case for Multiple Pods:
    - Apache (1 Container) +Tomcat (1 Container)
    - Apache(1 Container) + PHP (1 Container)
    - Nginx (Cache: 1 Container) + Apache/PHP (1 Container)
    - Web Server (1 Container) + Data Volume(Cache: 1 Container)
  - Pods will can create replicas of 1000+ set on cluster system

- Container/ Pods / Service / Deployment/ RS



- Way to deploy object in kubenetes by kubectl

Management technique	Operates on	Recommended environment	Supported writers	Learning curve
Imperative commands	Live objects	Development projects	1+	Lowest
Imperative object configuration	Individual files	Production projects	1	Moderate
Declarative object configuration	Directories of files	Production projects	1+	Highest

  - Imperative commands:
    - Ex: kubectl <action> <type/name> <option>
    - Ex: kubectl run webtest --image labdocker/nginx:latest

  - Imperative object configuration
    - Ex: kubectl <action> –f <YAML file>
    - Ex: kubectl create –f nginx.yml
    - Ex: kubectl replace –f nginx_update.yml

  - Declarative object configuration
    - Ex: kubectl apply –f <directory>

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite
      ports:
        - containerPort: 5000
          protocol: TCP
```

- Imperative commands:
  - "kubectl run" (Pods + Deployment+ RC (Replicas))

kubectl run –image=<image name> <option>

    - Option:
      - --env="key=value"
      - --port=port
      - --replicas=<number of Pods replicas>
      - --overrides=<json>
      - --labels=<label>
      - Etc

    - Example:

```
praparns-MacBook-Pro:~ praparn$ kubectl run webtest --image=labdocker/cluster:webservicelite --port=5000
deployment "webtest" created
```

    - **DEPRECATED !!!**

- Imperative commands:
  - "kubectl run" (Pods + Deployment+ RC (Replicas))

kubectl run –image=<image name> <option>

    - Option:
      - --env="key=value"
      - --port=port
      - --replicas=<number of Pods replicas>
      - --overrides=<json>
      - --labels=<label>
      - Etc

    - Example:

```
praparns-MacBook-Pro:~ praparn$ kubectl run webtest --image=labdocker/cluster:webservicelite --port=5000
deployment "webtest" created
```

    - **DEPRECATED !!!**

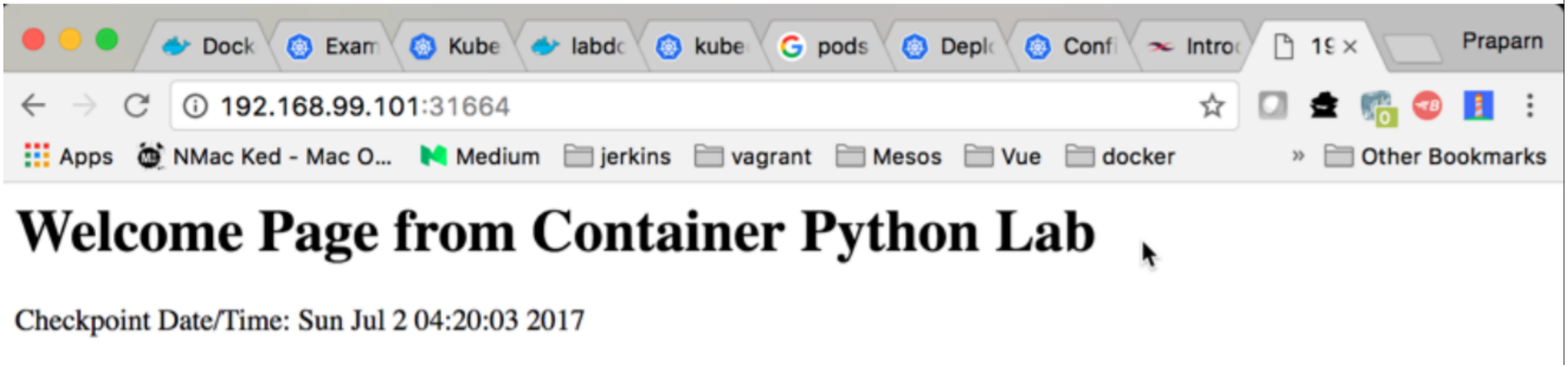
- Imperative commands:
  - kubectl expose (Service)

kubectl expose deployment <name> <option>

- Option:
  - --name=name
  - --port=port for service
  - --target-port=port of deployment set
  - --type=<NodePort/ClusterIP/LoadBalance>
  - --protocol=<TCP/UAT etc>
  - Etc

```
praparns-MacBook-Pro:~ praparn$ kubectl expose deployment webtest --target-port=5000 --type=NodePort
service "webtest" exposed
praparns-MacBook-Pro:~ praparn$ kubectl get svc webtest
NAME      CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
webtest   10.0.0.96    <nodes>       5000:31664/TCP   8s
```

- Imperative commands:



PODS, CONTAINER AND SERVICES

Imperative object configuration

kubectl create –f <Filename>

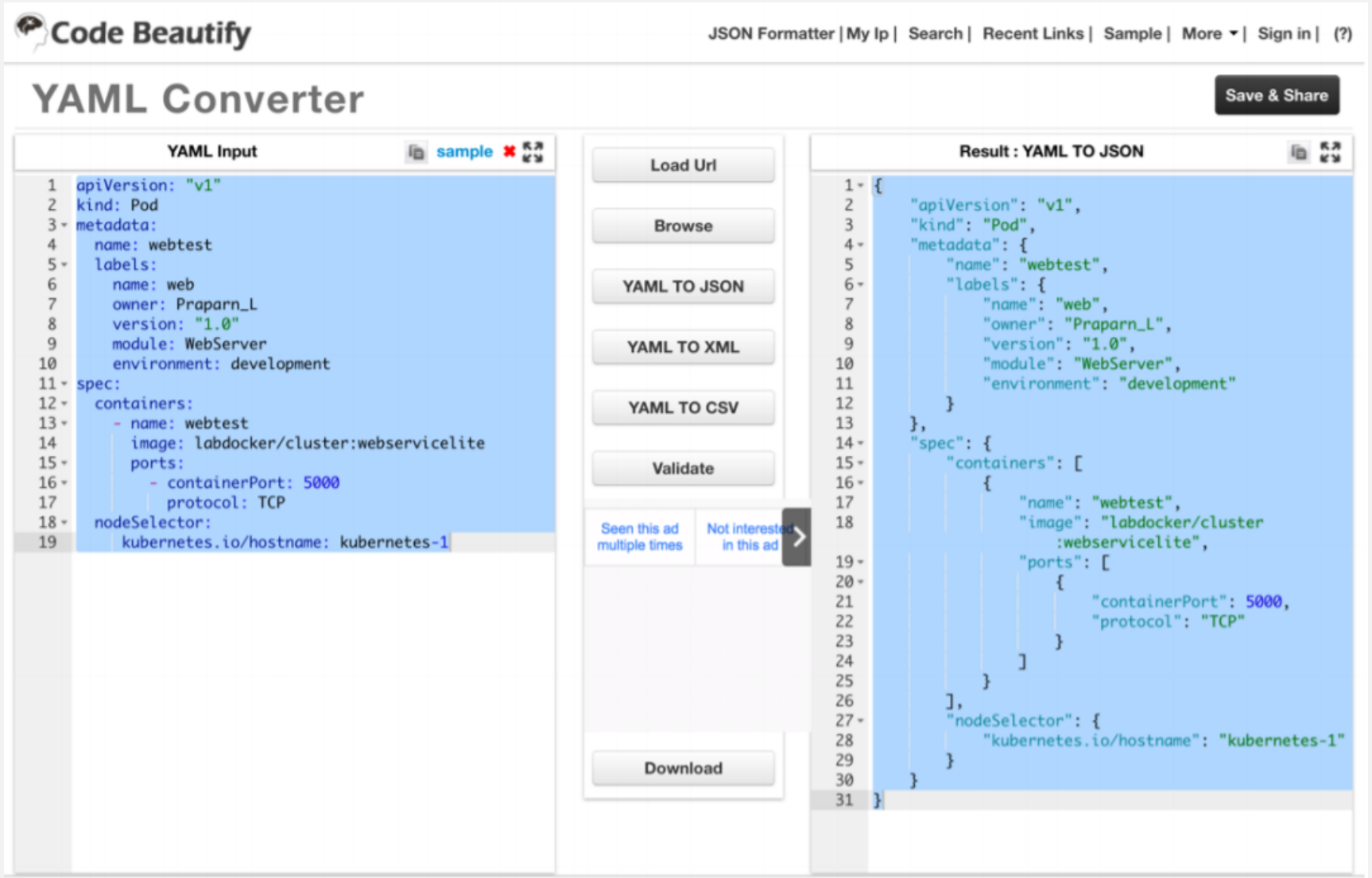
- Create Pods (YAML)

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite
      ports:
        - containerPort: 5000
          protocol: TCP
```

Create Service (YAML)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  type: NodePort
  ports:
   - port: 5000
     name: http
     targetPort: 5000
     protocol: TCP
```


Code Beautify
JSON Formatter | My Ip | Search | Recent Links | Sample | More | Sign in | (?)
YAML Converter
Save & Share
YAML Input
sample
apiVersion: "v1"
kind: Pod
metadata:
name: webtest
labels:
name: web
owner: Praparn_L
version: "1.0"
module: WebServer
environment: development
spec:
containers:
- name: webtest
image: labdocker/cluster:webservicelite
ports:
- containerPort: 5000
protocol: TCP
nodeSelector:
kubernetes.io/hostname: kubernetes-1
Load Url
Browse
YAML TO JSON
YAML TO XML
YAML TO CSV
Validate
Seen this ad multiple times
Not interested in this ad
Download
Result : YAML TO JSON
{
"apiVersion": "v1",
"kind": "Pod",
"metadata": {
"name": "webtest",
"labels": {
"name": "web",
"owner": "Praparn_L",
"version": "1.0",
"module": "WebServer",
"environment": "development"
}
},
"spec": {
"containers": [
{
"name": "webtest",
"image": "labdocker/cluster
:webservicelite",
"ports": [
{
"containerPort": 5000,
"protocol": "TCP"
}
]
}
],
"nodeSelector": {
"kubernetes.io/hostname": "kubernetes-1"
}
}
}

PODS, CONTAINER AND SERVICES

Imperative object configuration

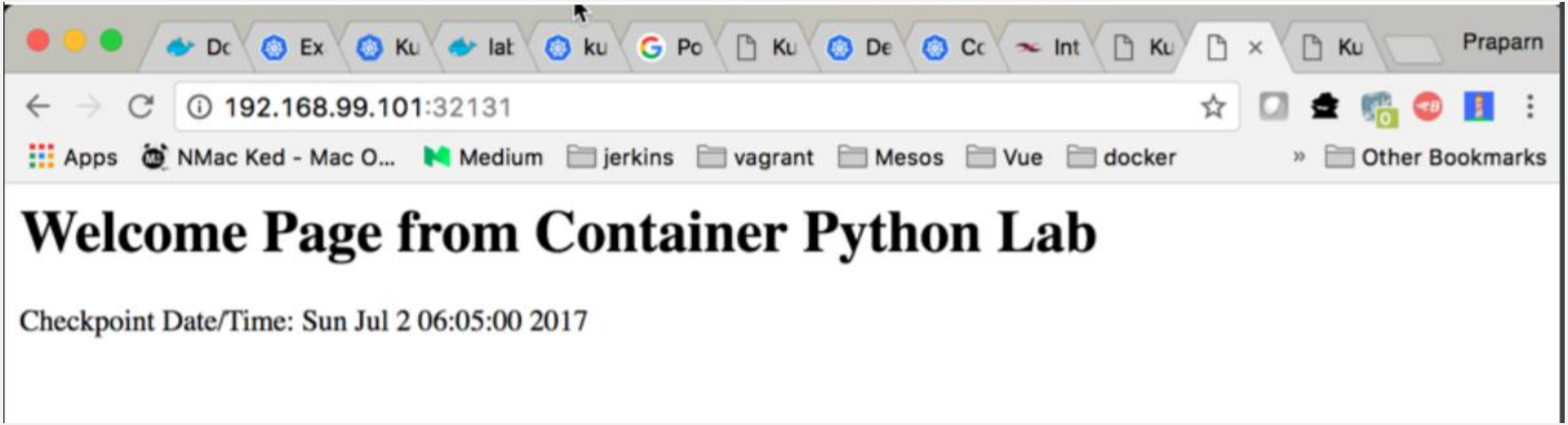
kubectl create –f <Filename>

```
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ ls -lh
total 24
-rw-r--r--@ 1 praparn  staff   550B Jul  2 09:29 instruction.txt
-rw-r--r--@ 1 praparn  staff   321B Jul  2 12:52 webtest_pod.yml
-rw-r--r--@ 1 praparn  staff   393B Jul  2 12:52 webtest_svc.yml
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl create -f webtest_pod.yml
pod "webtest" created
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
webtest   1/1       Running   0          4m
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl create -f webtest_svc.yml
service "webtest" created
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl get svc
NAME         CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
kubernetes   10.0.0.1     <none>        443/TCP          4d
webtest      10.0.0.87    <nodes>       5000:32131/TCP   4m
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ curl http://192.168.99.101:32131
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Sun Jul  2 06:04:08 2017
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$
```

PODS, CONTAINER AND SERVICES

Imperative object configuration

kubectl create –f <Filename>



PODS, CONTAINER AND SERVICES

Check log on container:

kubectl logs <Pods name> –c <container name>

```
praparns-MacBook-Pro:singlecontainer praparn$ kubectl logs webtest -c webtest
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 985-709-866
```

Shell inside container:

kubectl exec –it <Pods name> –c <container name> sh

```
praparns-MacBook-Pro:singlecontainer praparn$ kubectl exec -it webtest -c webtest sh
/usr/src/app # ls -lh
total 36
-rw-r--r--    1 root     root         482 Jul  1 03:42 docker-compose.yml
-rw-r--r--    1 root     root         345 Jul  1 02:10 dockerfile_nginx
-rw-r--r--    1 root     root          80 Jun 30 17:32 dockerfile_python
-rw-r--r--    1 root     root          88 Jul  2 03:29 dockerfile_python_lite
-rw-r--r--    1 root     root        2.7K Jul  2 03:01 main.py
-rw-r--r--    1 root     root         291 Jul  2 03:46 mainlite.py
-rw-r--r--    1 root     root        1.2K Jul  1 03:51 nginx.conf
-rw-r--r--    1 root     root          24 Jun 18 04:33 requirements.txt
-rw-r--r--    1 root     root           6 Jul  2 03:32 requirementslite.txt
/usr/src/app # hostname
webtest
/usr/src/app #
```

PODS, CONTAINER AND SERVICES

Check detail property of Pods/Service:

kubectl describe <Pods/SVC/etc> <Name>

```
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$ kubectl describe pods webtest
Name:           webtest
Namespace:      default
Node:           minikube/192.168.99.101
Start Time:     Sun, 02 Jul 2017 12:56:50 +0700
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Status:         Running
IP:             172.17.0.4
Controllers:    <none>
Containers:
  webtest:
    Container ID:       docker://04b9cdfdd6451a4e78c873f704fb4d35de7bba082343d0773c7ea0ebbbd3f03a
    Image:              labdocker/cluster:webservicelite
    Image ID:           docker://sha256:837c8f41c918ede06f05f9b554b6120fdb654cc2a8eb7eec74bc383b09865f2b
    Port:               5000/TCP
    State:              Running
      Started:          Sun, 02 Jul 2017 12:56:51 +0700
    Ready:              True
    Restart Count:      0
    Environment:        <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-vxm58 (ro)
Conditions:
  Type          Status
  Initialized   True
  Ready         True
  PodScheduled  True
Volumes:
  default-token-vxm58:
    Type:       Secret (a volume populated by a Secret)
    SecretName: default-token-vxm58
    Optional:   false
QoS Class:      BestEffort
Node-Selectors: <none>
Tolerations:    <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type            Reason          Message
  ---------     --------        -----   ----                    -------------                   --------        ------          -------
  14m           14m             1       default-scheduler                                       Normal          Scheduled       Successfully assigned webtest to minikube
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Pulled          Container image "labdocker/cluster:webservicelite" already
y present on machine
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Created         Created container with id 04b9cdfdd6451a4e78c873f704fb4d3
5de7bba082343d0773c7ea0ebbbd3f03a
  14m           14m             1       kubelet, minikube       spec.containers{webtest}        Normal          Started         Started container with id 04b9cdfdd6451a4e78c873f704fb4d3
5de7bba082343d0773c7ea0ebbbd3f03a
praparns-MacBook-Pro:WorkShop_1.2_Pods_Service_Deployment praparn$
```

PODS, CONTAINER AND SERVICES

Check overall of Pods/Service

kubectl get <Pods/SVC/etc>

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME      READY     STATUS    RESTARTS   AGE
maindb    1/1       Running   0          18m
web       3/3       Running   0          16m
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                      AGE
kubernetes   10.0.0.1      <none>        443/TCP                      4d
maindb       10.0.0.134    <none>        3306/TCP                     17m
web          10.0.0.69     <nodes>       5000:30661/TCP,80:30500/TCP  16m
praparns-MacBook-Pro:multicontainer praparn$
```

Remove Pod/Service

kubectl delete –f <Filename>

```
praparns-MacBook-Pro:singlecontainer praparn$ kubectl delete  -f webtest_svc.yml
service "webtest" deleted
praparns-MacBook-Pro:singlecontainer praparn$ kubectl delete  -f webtest_pod.yml
pod "webtest" deleted
```

- Trade-offs
  - Imperative commands:
    - Advantage:
      - Simple & Easy
      - Single Step for All
      - Effective immediate real-time
    - Disadvantage:
      - Hard to review on complete deployment
      - Unable to track change of them/Store source of deployment
      - No template for reiteration
      - Very long syntax when facing with complex configuration

- Trade-offs
  - Imperative object configuration
    - Advantage:
      - Deployment source can keep and review process (svn, git etc)
      - Provide formal template for deployment
      - No skill set required deployment process
    - Disadvantage:
      - Need to understand of kubernetes before use
      - Writing YAML file before operate
      - Lost configuration update when it not appear on file (Next replacement will lost)
      - Design for manage via file not directory

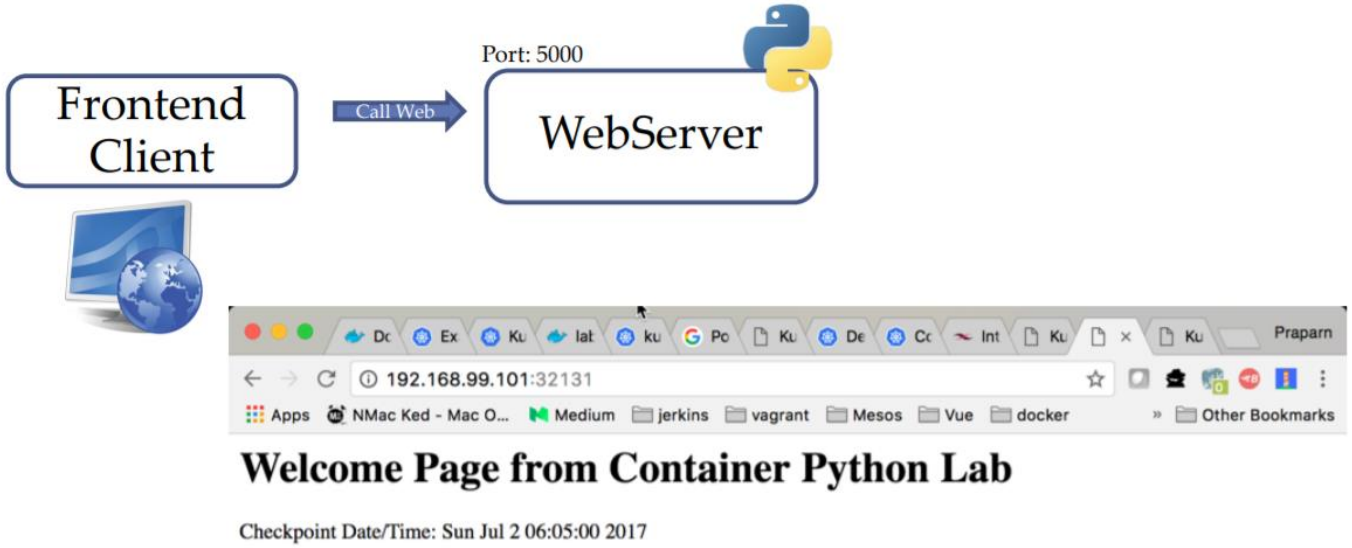
- Trade-offs
  - Declarative object configuration
    - Advantage:
      - Coverage all change in single command (Entire folder)
      - Fully support live-object configuration
      - Keep track/Merge all change in configuration file
      - Update default value in configuration file
    - Disadvantage:
      - More complicate on configuration file

Field in object configuration file	Field in live object configuration	Field in last-applied-configuration	Action
Yes	Yes	-	Set live to configuration file value.
Yes	No	-	Set live to local configuration.
No	-	Yes	Clear from live configuration.
No	-	No	Do nothing. Keep live value.

- Trade-offs
  - Declarative object configuration
    - Advantage:
      - Coverage all change in single command (Entire folder)
      - Fully support live-object configuration
      - Keep track/Merge all change in configuration file
      - Update default value in configuration file
    - Disadvantage:
      - More complicate on configuration file

Field in object configuration file	Field in live object configuration	Field in last-applied-configuration	Action
Yes	Yes	-	Set live to configuration file value.
Yes	No	-	Set live to local configuration.
No	-	Yes	Clear from live configuration.
No	-	No	Do nothing. Keep live value.

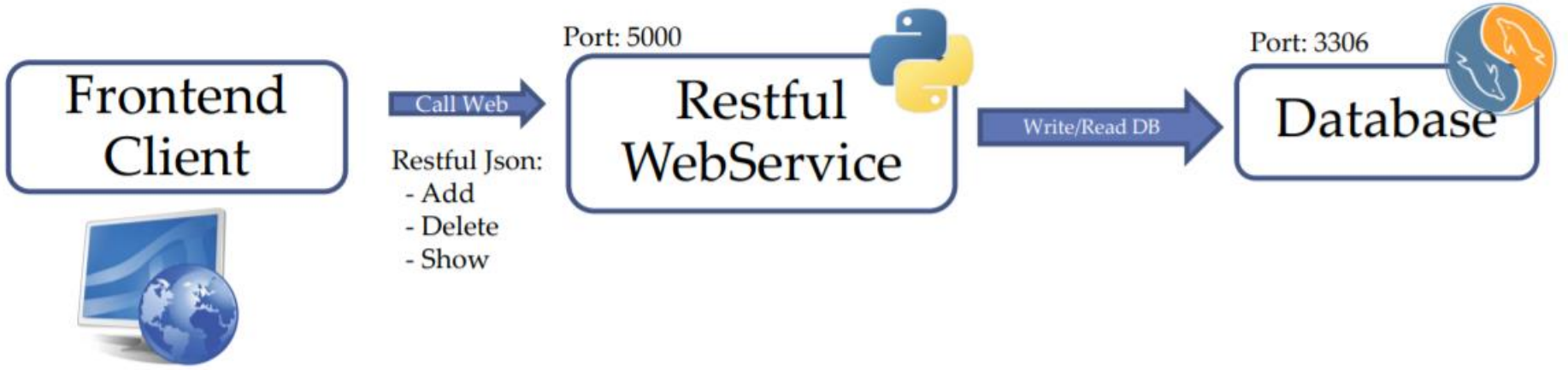
- Part 1: Deploy simple web pods with single container



WORKSHOP 1.1: PODS & SERVICE

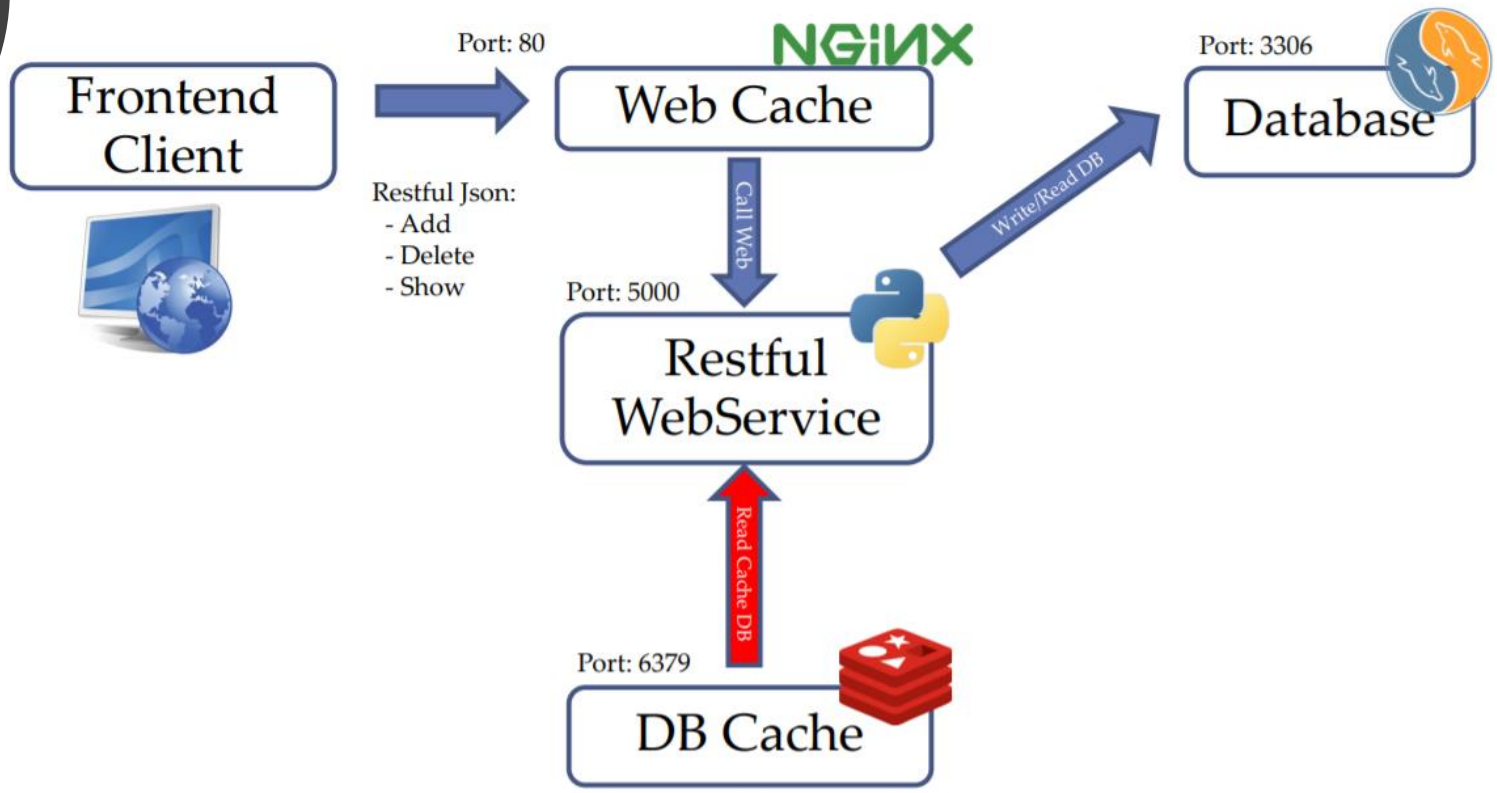
WORKSHOP 1.1: PODS & SERVICE

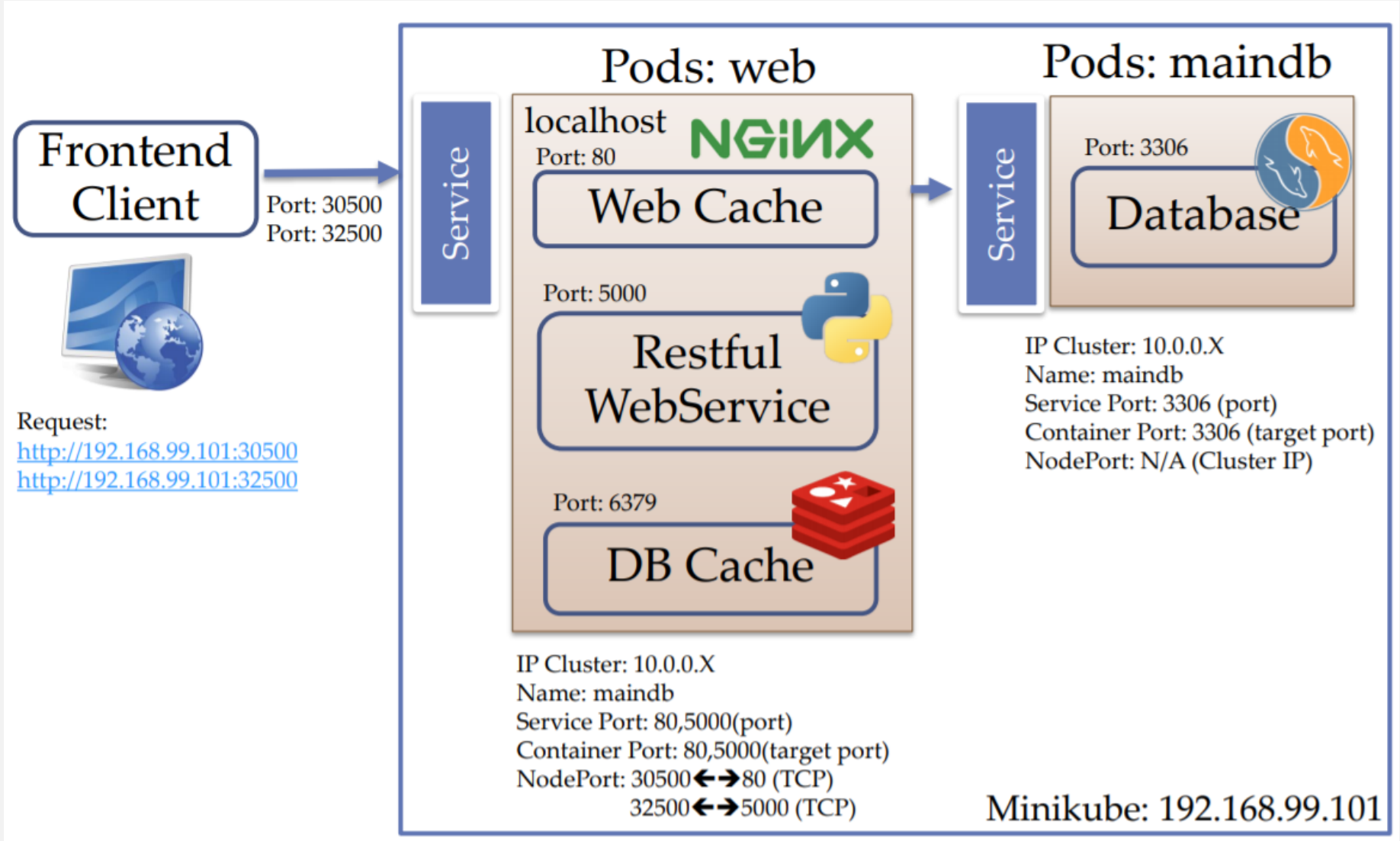
- Example Case: Basic Restful API

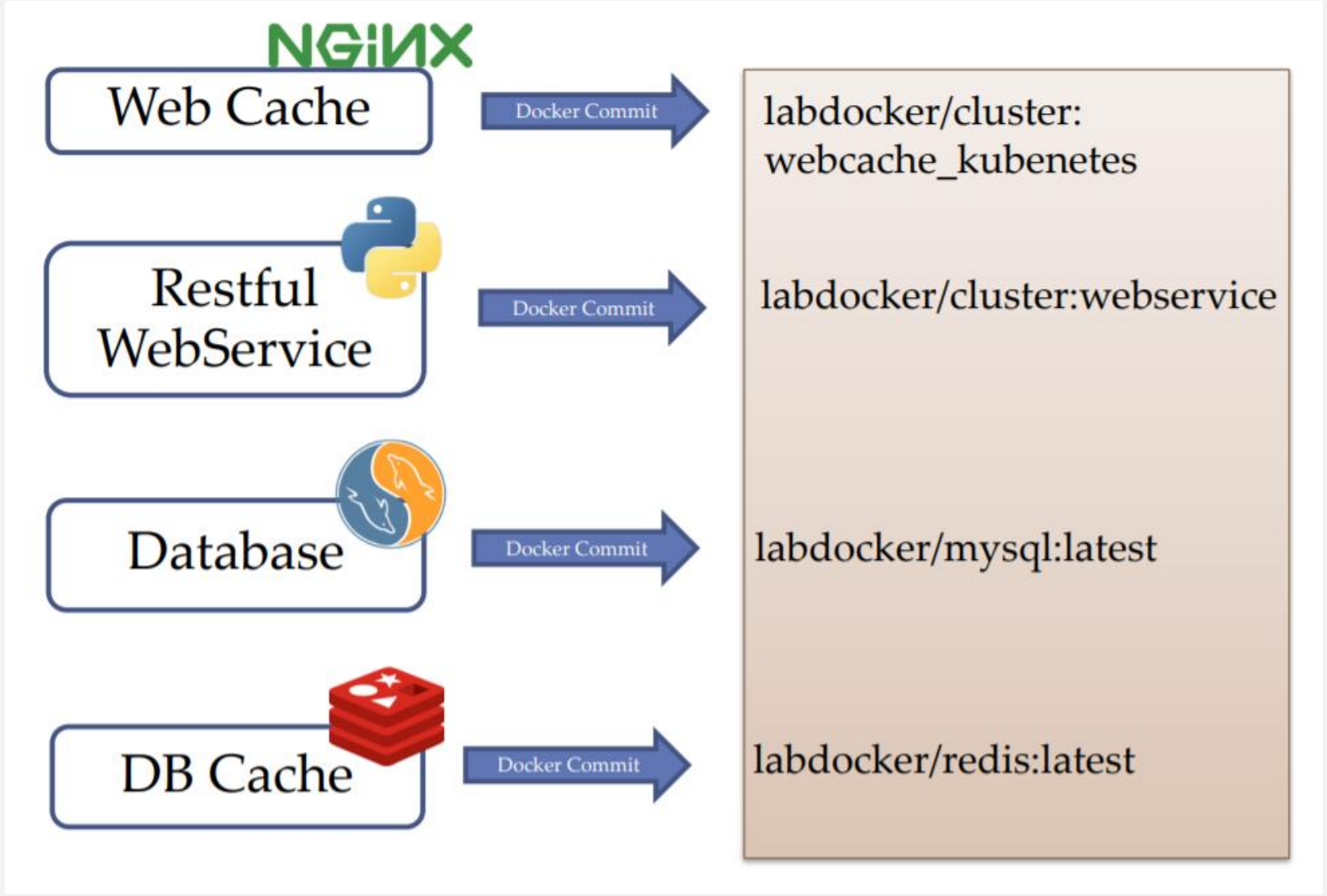


WORKSHOP
1.1: PODS
& SERVICE

- Example Case: High I/O Restful API




Frontend Client
Port: 30500
Port: 32500
Request:
http://192.168.99.101:30500
http://192.168.99.101:32500
Service
Pods: web
localhost
Port: 80
NGINX
Web Cache
Port: 5000
Restful WebService
Port: 6379
DB Cache
IP Cluster: 10.0.0.X
Name: maindb
Service Port: 80,5000(port)
Container Port: 80,5000(target port)
NodePort: 30500←→80 (TCP)
32500←→5000 (TCP)
Service
Pods: maindb
Port: 3306
Database
IP Cluster: 10.0.0.X
Name: maindb
Service Port: 3306 (port)
Container Port: 3306 (target port)
NodePort: N/A (Cluster IP)
Minikube: 192.168.99.101


NGINX
Web Cache
Docker Commit
labdocker/cluster:
webcache_kubenetes
Restful
WebService
Docker Commit
labdocker/cluster:webservice
Database
Docker Commit
labdocker/mysql:latest
DB Cache
Docker Commit
labdocker/redis:latest

- Pods: maindb(YML)

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: maindb
  labels:
    name: "maindb"
    owner: "Praparn_L"
    version: "1.0"
    module: "maindb"
    environment: "development"
spec:
  containers:
    - name: maindb
      image: labdocker/mariadb:latest
      ports:
        - containerPort: 3306
          protocol: TCP
      env:
        -
          name: "MYSQL_ROOT_PASSWORD"
          value: "password"

```

Service: maindb(YML)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: maindb
  labels:
    name: "maindb"
    owner: "Praparn_L"
    version: "1.0"
    module: "maindb"
    environment: "development"
spec:
  ports:
  - port: 3306
    targetPort: 3306
  selector:
    name: "maindb"
    owner: "Praparn_L"
    version: "1.0"
    module: "maindb"
    environment: "development"

```

- Pods: web(YML)

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: web
  labels:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "web"
    environment: "development"
spec:
  containers:
    - name: cachedb
      image: labdocker/redis:latest
      ports:
        - containerPort: 6379
          protocol: TCP
    - name: webservice
      image: labdocker/cluster:webservice
      env:
        - name: "REDIS_HOST"
          value: "localhost"
      ports:
        - containerPort: 5000
          protocol: TCP
    - name: webcache
      image: labdocker/cluster:webcache_kubernetes
      ports:
        - containerPort: 80
          protocol: TCP
```

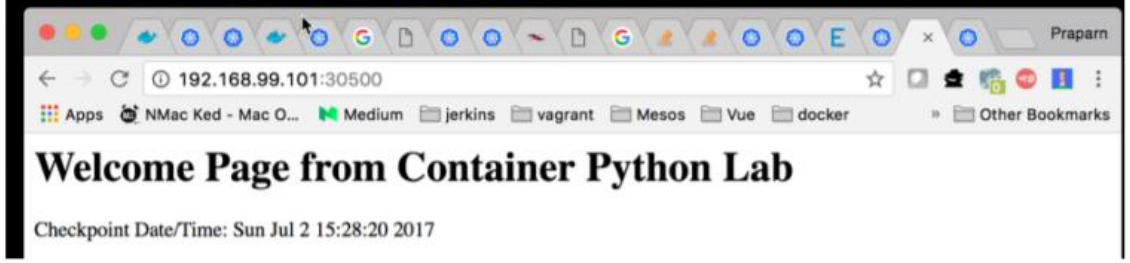
Service: web (YML)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web
  labels:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "Web"
    environment: "development"
spec:
  selector:
    name: "web"
    owner: "Praparn_L"
    version: "1.0"
    module: "web"
    environment: "development"
  type: NodePort
  ports:
   - port: 5000
     name: webservice
     targetPort: 5000
     protocol: TCP
     nodePort: 32500
   - port: 80
     name: webcache
     targetPort: 80
     protocol: TCP
     nodePort: 30500
```

PODS, CONTAINER AND SERVICES

- Services
  - Existing Pods "web"

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME                    READY     STATUS             RESTARTS   AGE
curl-1580724602-gjjhq   0/1       CrashLoopBackOff   27         1h
maindb                  1/1       Running            0          56m
web                     3/3       Running            0          7m
praparns-MacBook-Pro:multicontainer praparn$ curl http://$Server_IP:$Server_Port
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Sun Jul  2 15:28:13 2017
praparns-MacBook-Pro:multicontainer praparn$
```

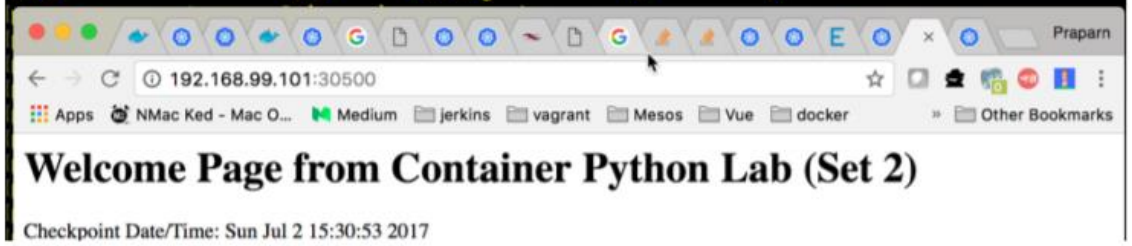



```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                      AGE
kubernetes   10.0.0.1      <none>        443/TCP                      5d
maindb       10.0.0.236    <none>        3306/TCP                     1m
web          10.0.0.212    <nodes>       5000:32500/TCP,80:30500/TCP  1m
praparns-MacBook-Pro:multicontainer praparn$
```

PODS, CONTAINER AND SERVICES

- Services
  - Replace new Pods “web2”

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl create -f webmodule_pod2.yml
pod "web2" created
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME                    READY     STATUS             RESTARTS   AGE
curl-1580724602-gjjhq   0/1       CrashLoopBackOff   27         1h
maindb                  1/1       Running            0          58m
web                     3/3       Running            0          10m
web2                    3/3       Running            0          1m
praparns-MacBook-Pro:multicontainer praparn$ kubectl delete pods web
pod "web" deleted
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME                    READY     STATUS             RESTARTS   AGE
curl-1580724602-gjjhq   0/1       CrashLoopBackOff   27         1h
maindb                  1/1       Running            0          59m
web2                    3/3       Running            0          2m
```




```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                      AGE
kubernetes   10.0.0.1      <none>        443/TCP                      5d
maindb       10.0.0.236    <none>        3306/TCP                     1m
web          10.0.0.212    <nodes>       5000:32500/TCP,80:30500/TCP  1m
praparns-MacBook-Pro:multicontainer praparn$
```

PODS, CONTAINER AND SERVICES

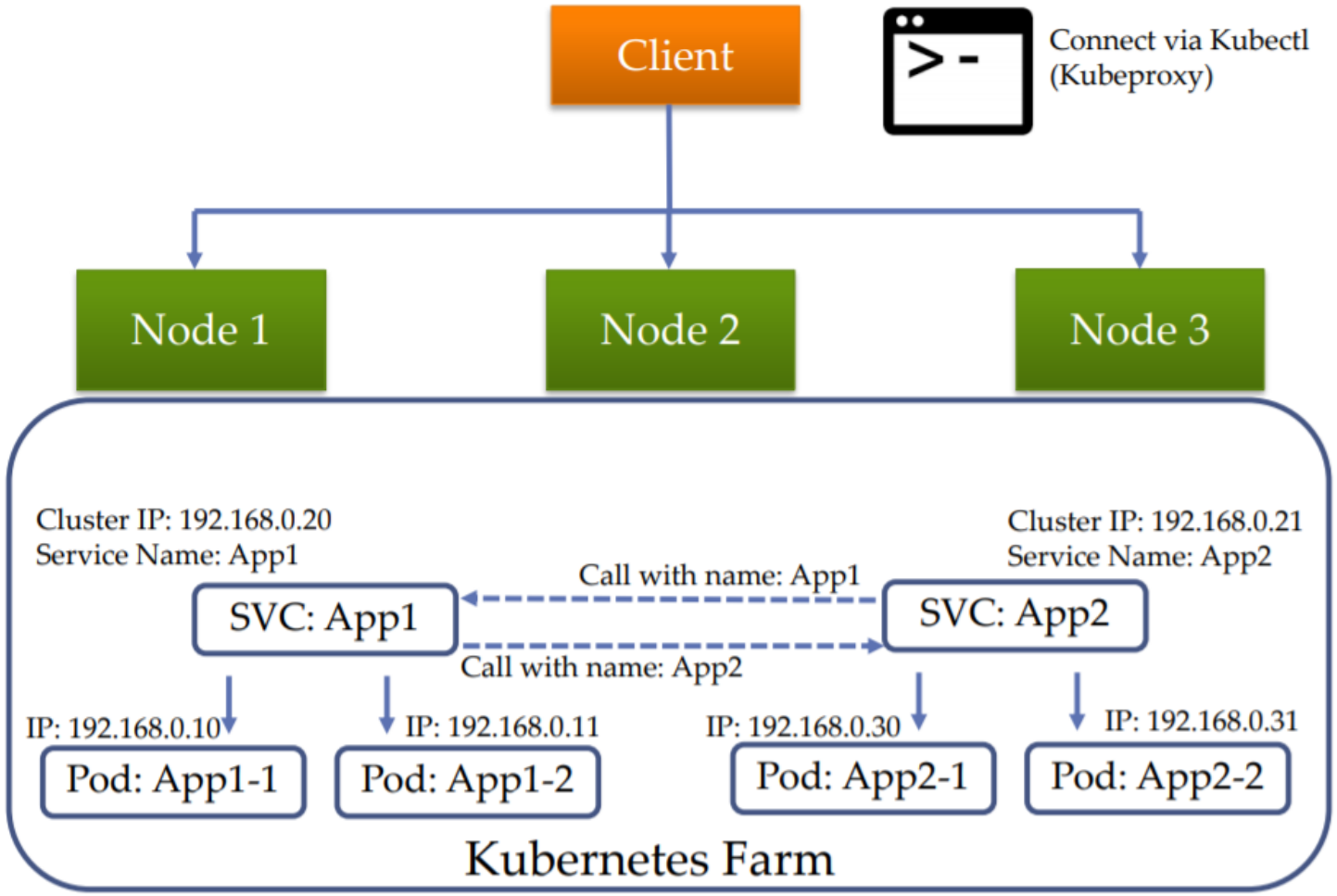
- Services:
  - Publish Service Type:
    - CLUSTER-IP (Default): blind port with ip addess of cluster (Accessible from internal cluster system)
    - NodePort: blind port with ip address of node. By default kubernetes will random port (30000-32757). If we need to specify set the option: "nodePort: XXXXX"

```
18    type: NodePort
19    ports:
20     - port: 5000
21       name: webservice
22       targetPort: 5000
23       protocol: TCP
24       nodePort: 32500
25     - port: 80
26       name: webcache
27       targetPort: 80
28       protocol: TCP
29       nodePort: 30500
```

```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)                      AGE
kubernetes   10.0.0.1      <none>        443/TCP                      5d
maindb       10.0.0.236    <none>        3306/TCP                     1m
web          10.0.0.212    <nodes>       5000:32500/TCP,80:30500/TCP  1m
praparns-MacBook-Pro:multicontainer praparn$
```

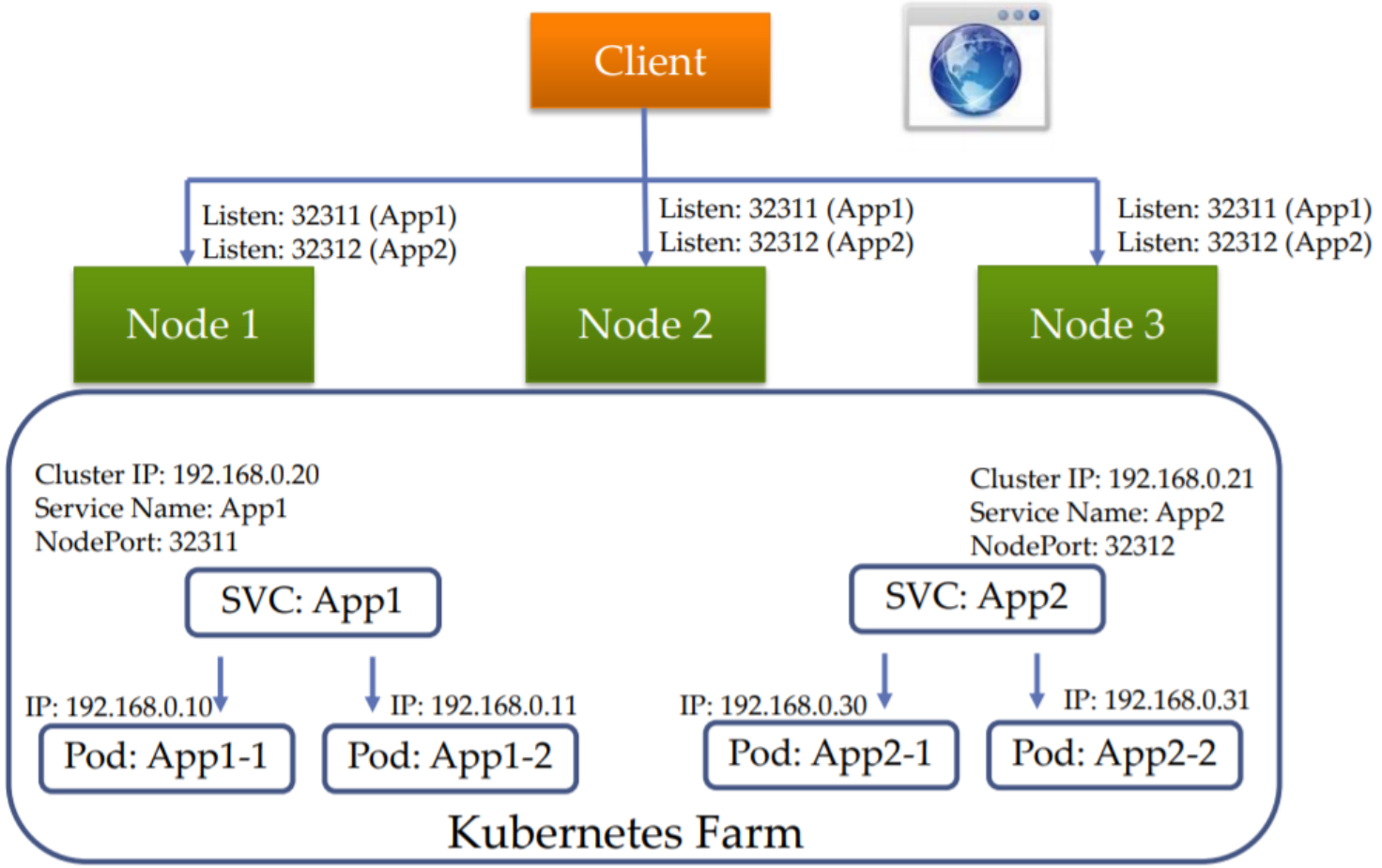
PODS, CONTAINER AND SERVICES

- Services: ClusterIP



PODS, CONTAINER AND SERVICES

- Services: NodePort

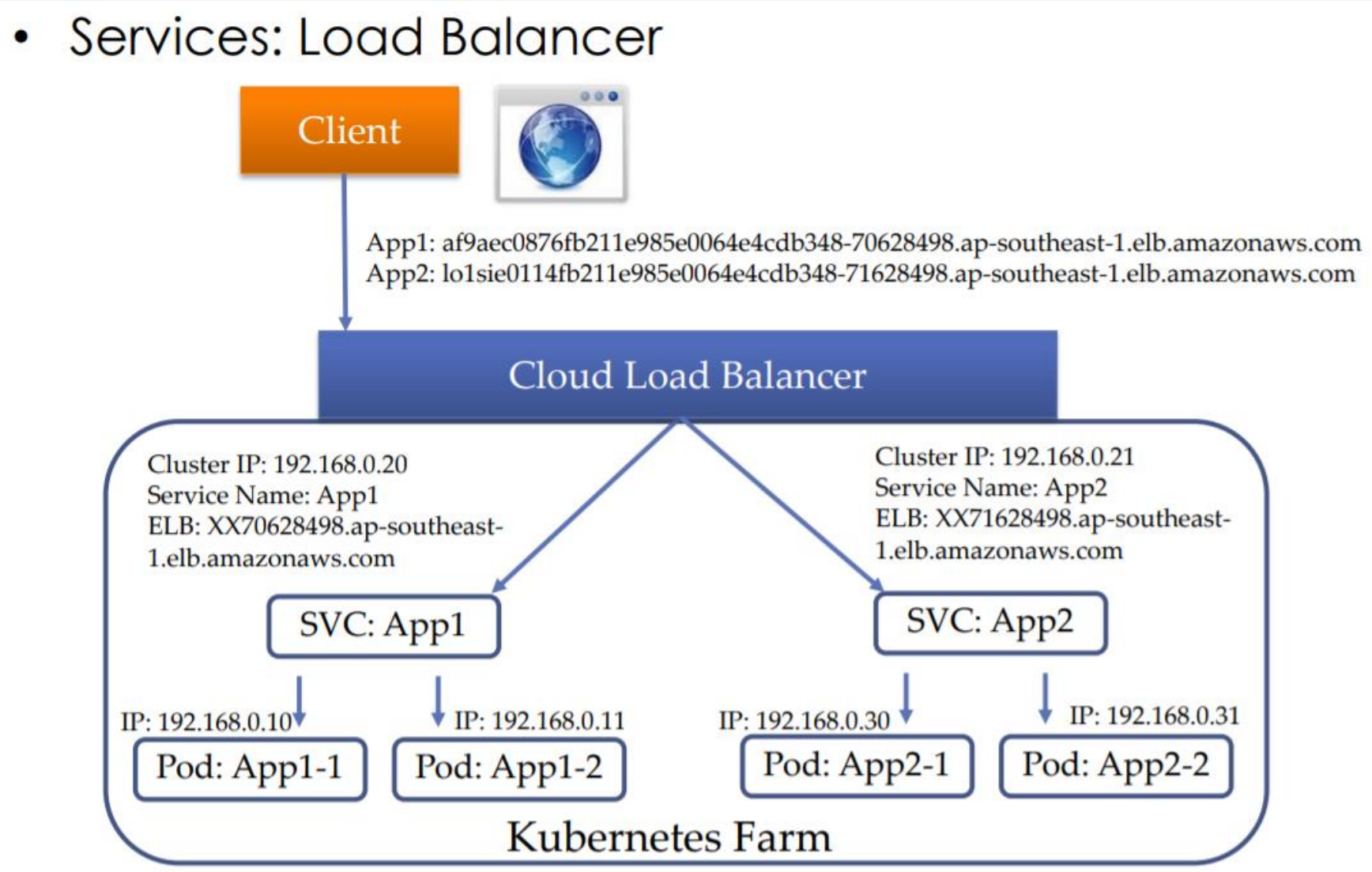


PODS, CONTAINER AND SERVICES

- Services
  - Publish Service Type:
    - LoadBalance:
      - Use load balancer from external cloud provider for intercept traffic
      - Manage by Kubernetes Itself
      - Flexible for use facilities on cloud

```yaml
kind: Service
apiVersion: v1
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
      nodePort: 30061
  clusterIP: 10.0.171.239
  loadBalancerIP: 78.11.24.19
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
      - ip: 146.148.47.155
```

- Services: Load Balancer



PODS, CONTAINER AND SERVICES

PODS, CONTAINER AND SERVICES

- Services:
  - Publish Service Type:
    - ExternalIP:
      - Similar with Load Balancer
      - Reference to external load balance by IP Address
      - Not manage by kubernetes itself

```yaml
kind: Service
apiVersion: v1
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
  - name: http
    protocol: TCP
    port: 80
    targetPort: 9376
  externalIPs:
  - 80.11.12.10
```

PODS, CONTAINER AND SERVICES

Services

- Publish Service Type:
  - Headless Service (StatefulSet): service will just dns round-robin for all statefulset pods's ip address

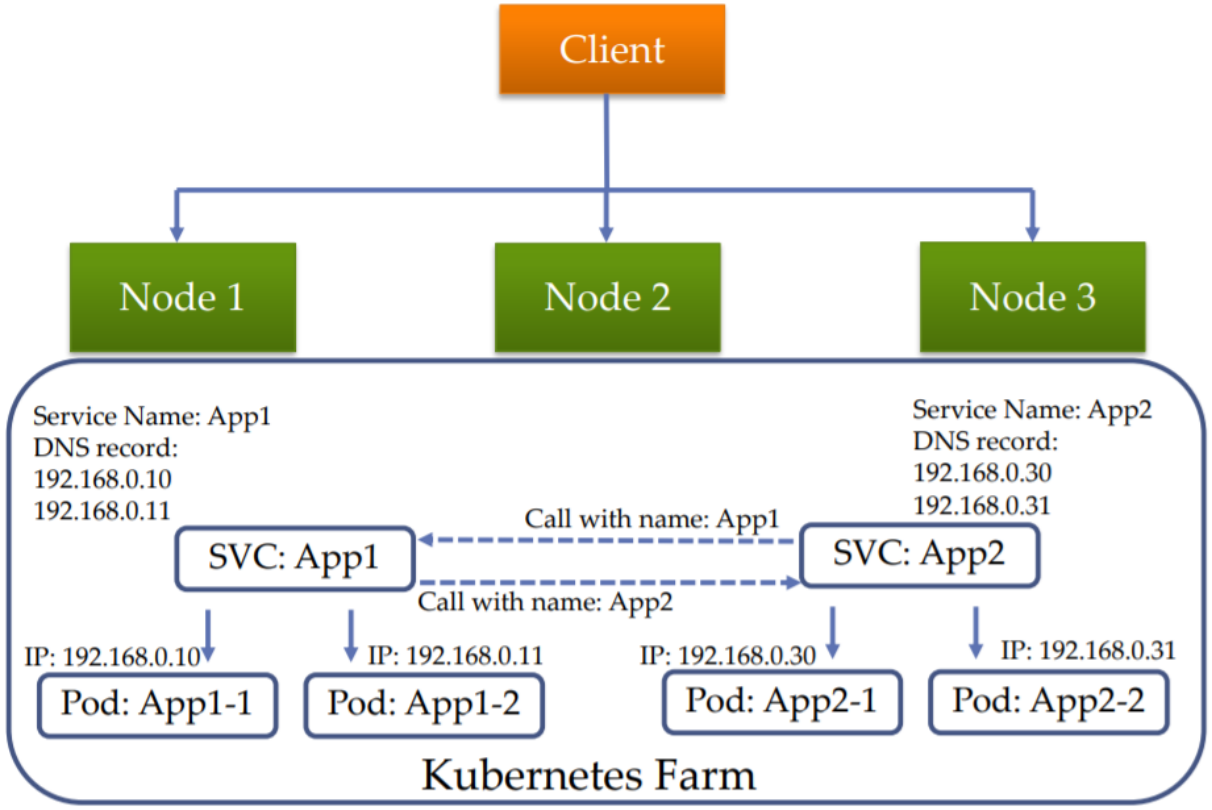
20dns.yml

```yaml
# A headless service to create DNS records
---
apiVersion: v1
kind: Service
metadata:
  name: kafka
  namespace: sansiri-unityapi
spec:
  ports:
  - port: 9092
  # [podname].broker.kafka.svc.cluster.local
  clusterIP: None
  selector:
    app: kafka

```

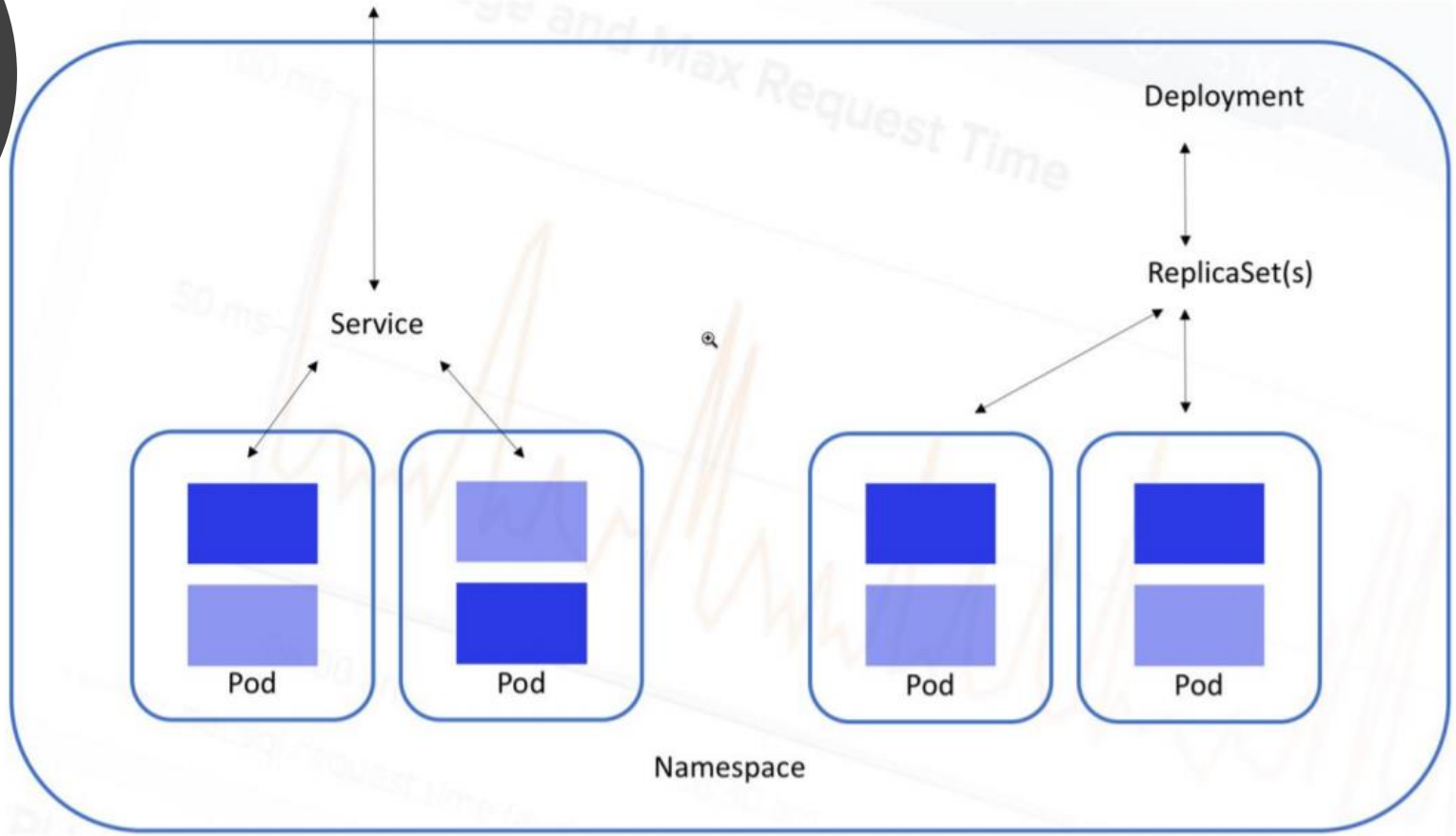
PODS, CONTAINER AND SERVICES

- Services: Headless



DAEMON SET AND RC

DAEMON SET AND RC



Daemon Set and RC

- What is Daemon Set
  - On basic pods and service can make microservice up and run!
  - But…
    - No maintain about available of Pods
    - Not response Pods scaling
    - Etc
  - Daemon Set will response:
    - Make sure that Pods run on all (or some) node in cluster
    - When node add to cluster. Pods will deploy automatic
    - When node remove, GC will automatic remove Pod
    - When remove Daemon Set, Pods will automatic remove
  - General Use Case
    - Storage Daemon (Ceph etc)
    - Node Monitor Daemon (Prometheus Agent etc)

Replication Controller (RC)

- What is RC
  - Daemon Set and Replication Controller is work similar.
  - RC (Replication Controller) will response:
    - Create/Maintain Pods as "Replication Controller" (Pods Farm)
    - Keep copy of Pods (Replicas) as design
    - Ensure that Pods is up and run with amount like design
      - If too much, It will kill some Pods
      - If too kill, It will create another replicas of Pods
    - Auto healing if some Pods crash with any reason
    - Maintenance on cluster-level not node level

Replication Controller (RC)

- Replication Controller (RC)
  - RC is first version of module to maintain Pods available in cluster
  - Use Case…
    - Rescheduling
    - Scaling
    - Rolling updates (Complicate)
    - Map with service for manage release
  - RC is running base on label type: “Equality-based requirement”
  - Ex:

```
selector:
  name: web
  owner: Praparn_L
  version: "1.0"
  module: WebServer
  environment: development
```

REPLICATION CONTROLLER (RC)

- Example
  - Pods “webtest”

```yaml
apiVersion: "v1"
kind: Pod
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite
      ports:
        - containerPort: 5000
          protocol: TCP
```

RC “webtest”

```yaml
apiVersion: v1
kind: ReplicationController
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 3
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite
          ports:
            - containerPort: 5000
              protocol: TCP
```

SVC “webtest”

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  type: NodePort
  ports:
   - port: 5000
     name: http
     targetPort: 5000
     protocol: TCP
     nodePort: 32500
```

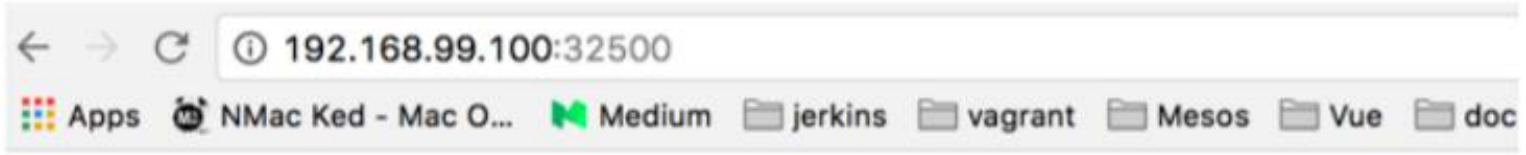
REPLICATION CONTROLLER (RC)

- Create rc "webtest"

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl create -f webtest_rc.yml
replicationcontroller "webtest" created
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get rc
NAME      DESIRED   CURRENT   READY     AGE
webtest   3         3         3         2m
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9m8tx   1/1       Running   0          3m
webtest-9xgt3   1/1       Running   0          3m
webtest-cns49   1/1       Running   0          3m
```

- Create svc "webtest"

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl create -f webtest_svc.yml
service "webtest" created
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
kubernetes   10.0.0.1      <none>        443/TCP          8d
webtest      10.0.0.6      <nodes>       5000:32500/TCP   6s
praparns-MacBook-Pro:ReplicationController praparn$
```



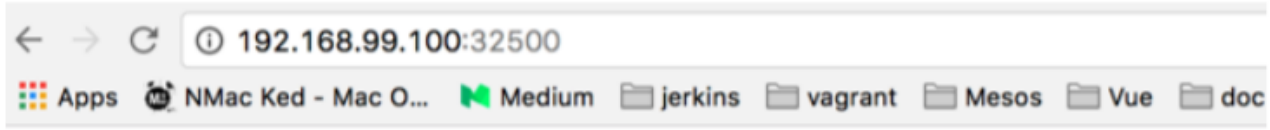
REPLICATION CONTROLLER (RC)

- Test delete some Pods from command line

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9m8tx   1/1       Running   0          11m
webtest-9xgt3   1/1       Running   0          11m
webtest-cns49   1/1       Running   0          11m
praparns-MacBook-Pro:ReplicationController praparn$ kubectl delete pods webtest-9m8tx
pod "webtest-9m8tx" deleted
```

- Recheck Pods unit and available

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9xgt3   1/1       Running   0          21m
webtest-cns49   1/1       Running   0          21m
webtest-lmn54   1/1       Running   0          8m
praparns-MacBook-Pro:ReplicationController praparn$ curl http://192.168.99.100:32500
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Wed Jul  5 15:51:59 2017
praparns-MacBook-Pro:ReplicationController praparn$
```



Welcome Page from Container Python Lab

Checkpoint Date/Time: Wed Jul 5 15:55:31 2017

REPLICATION CONTROLLER (RC)

- Check detail of create process on RC

```
s-MacBook-Pro:ReplicationController praparn$ kubectl describe rc webtest
Name:           webtest
ace:            default
tor:            environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
s:              environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Replicas:       3 current / 3 desired
Pods Status:    3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:       environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
  Containers:
   webtest:
    Image:              labdocker/cluster:webservicelite
    Port:               5000/TCP
    Environment:        <none>
    Mounts:             <none>
  Volumes:              <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type            Reason          Message
  ---------     --------        -----   ----                    -------------   --------        ------          -------
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-9m8tx
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-cns49
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-9xgt3
  13m           13m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-lmn54
praparns-MacBook-Pro:ReplicationController praparn$
```

REPLICATION CONTROLLER (RC)

- Scale up replicas on RC

kubectl scale <option> --replicas <Type/Name>

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl scale --replicas=10 rc/webtest
replicationcontroller "webtest" scaled
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get rc
NAME      DESIRED   CURRENT   READY     AGE
webtest   10        10        5         32m
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-2hgdq   1/1       Running   0          18s
webtest-9xgt3   1/1       Running   0          32m
webtest-cns49   1/1       Running   0          32m
webtest-jbqdq   1/1       Running   0          18s
webtest-lgprd   1/1       Running   0          18s
webtest-lmn54   1/1       Running   0          19m
webtest-sqsjk   1/1       Running   0          18s
webtest-thhxf   1/1       Running   0          18s
webtest-tx0px   1/1       Running   0          18s
webtest-v0n6s   1/1       Running   0          18s
```

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl scale --replicas=5 -f webtest_rc.yml
replicationcontroller "webtest" scaled
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get rc webtest
NAME      DESIRED   CURRENT   READY     AGE
webtest   5         5         5         40m
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9xgt3   1/1       Running   0          40m
webtest-cns49   1/1       Running   0          40m
webtest-lmn54   1/1       Running   0          27m
webtest-tx0px   1/1       Running   0          7m
webtest-v0n6s   1/1       Running   0          7m
praparns-MacBook-Pro:ReplicationController praparn$
```

REPLICATION CONTROLLER (RC)

- Check detail of create process on RC

```
s-MacBook-Pro:ReplicationController praparn$ kubectl describe rc webtest
Name:           webtest
ace:            default
tor:            environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
s:              environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
Annotations:    <none>
Replicas:       3 current / 3 desired
Pods Status:    3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:       environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
  Containers:
   webtest:
    Image:              labdocker/cluster:webservicelite
    Port:               5000/TCP
    Environment:        <none>
    Mounts:             <none>
  Volumes:              <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type            Reason          Message
  ---------     --------        -----   ----                    -------------   --------        ------          -------
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-9m8tx
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-cns49
  26m           26m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-9xgt3
  13m           13m             1       replication-controller                  Normal          SuccessfulCreate        Created pod: webtest-lmn54
praparns-MacBook-Pro:ReplicationController praparn$
```

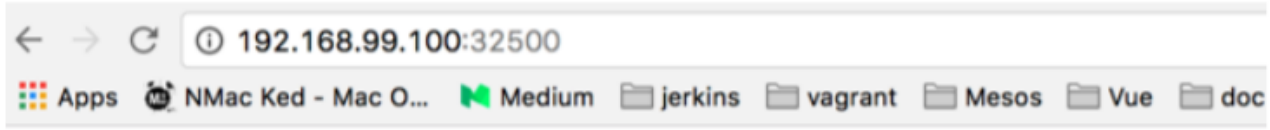
REPLICATION CONTROLLER (RC)

- Test delete some Pods from command line

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9m8tx   1/1       Running   0          11m
webtest-9xgt3   1/1       Running   0          11m
webtest-cns49   1/1       Running   0          11m
praparns-MacBook-Pro:ReplicationController praparn$ kubectl delete pods webtest-9m8tx
pod "webtest-9m8tx" deleted
```

- Recheck Pods unit and available

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
NAME            READY     STATUS    RESTARTS   AGE
webtest-9xgt3   1/1       Running   0          21m
webtest-cns49   1/1       Running   0          21m
webtest-lmn54   1/1       Running   0          8m
praparns-MacBook-Pro:ReplicationController praparn$ curl http://192.168.99.100:32500
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Wed Jul  5 15:51:59 2017
praparns-MacBook-Pro:ReplicationController praparn$
```



Welcome Page from Container Python Lab

Checkpoint Date/Time: Wed Jul 5 15:55:31 2017

WORKSHOP 1.2: RC

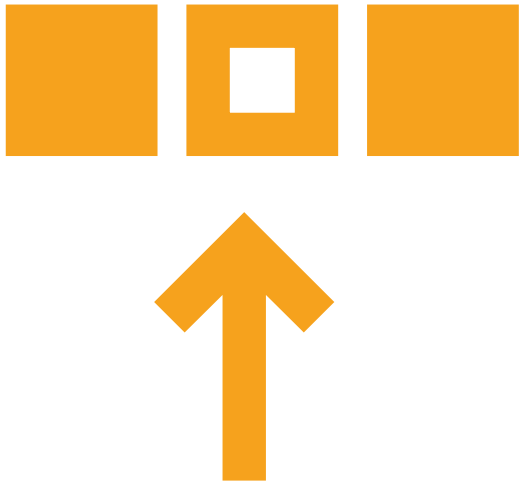
- Deploy replication controller

```
  Containers:
   webtest:
    Image:              labdocker/cluster:webservicelite
    Port:               5000/TCP
    Environment:        <none>
    Mounts:             <none>
  Volumes:              <none>
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type        Reason          Message
  ---------     --------        -----   ----                    -------------   --------    ------          -------
  42m           42m             1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-cns49
  42m           42m             1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-9xgt3
  42m           42m             1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-9m8tx
  29m           29m             1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-lmn54
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-lgprd
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-v0n6s
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-thhxf
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-sqsjk
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-tx0px
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-2hgdq
  9m            9m              1       replication-controller                  Normal      SuccessfulCreate  Created pod: webtest-jbqdq
  2m            2m              1       replication-controller                  Normal      SuccessfulDelete  Deleted pod: webtest-lgprd
  2m            2m              1       replication-controller                  Normal      SuccessfulDelete  Deleted pod: webtest-thhxf
  2m            2m              1       replication-
  2m            2m              1       replication-
  2m            2m              1       replication-
praparns-MacBook-Pro:ReplicationController praparn$
```

Replication Controller

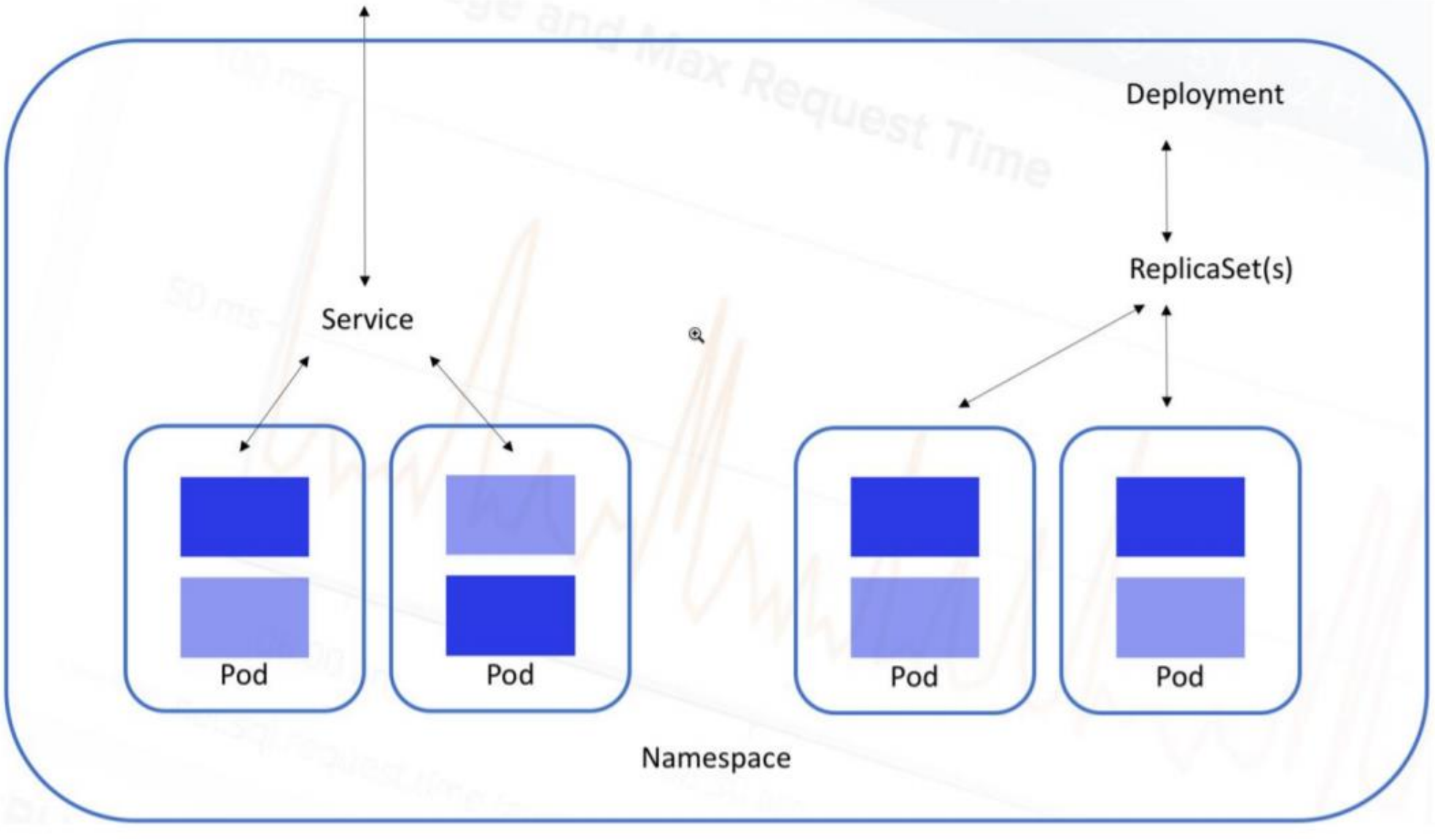
- Cleanup Lab

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl delete -f webtest_svc.yml
service "webtest" deleted
praparns-MacBook-Pro:ReplicationController praparn$ kubectl delete -f webtest_rc.yml
replicationcontroller "webtest" deleted
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get rc
No resources found.
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get pods
No resources found.
praparns-MacBook-Pro:ReplicationController praparn$ kubectl get svc
NAME         CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   10.0.0.1     <none>        443/TCP   8d
praparns-MacBook-Pro:ReplicationController praparn$
```



DEPLOYMENT/RS AND UPDATE

DEPLOYMENT/RS AND UPDATE



DEPLOYMENT/RS AND UPDATE

- What is deployment/RS?
  - Deployment and RS (ReplicaSet) is set "next-generation of RC" by provide full function to maintain versioning of Pods in production (No downtime: On-the-fly)
    - Update new version (Rollout)
    - Revert old version (Rollback)
    - Pause/Resume process
    - Check status
  - By design deployment will order job to ReplicaSet(RS) for create Pods as design for up and run
  - When new version was rollout from deployment (Automatic)
    - Create new RS and start to scale as desired
    - Scale down existing RS to 0
    - Delete existing RS

DEPLOYMENT/RS AND UPDATE

ReplicaSet(RS) vs Replication Controller (RC)

- RS is generation that evolution from RC with capability more dynamic
- RC support label with method "Equality-based requirement"

```yaml
selector:
  name: web
  owner: Praparn_L
  version: "1.0"
  module: WebServer
  environment: development
```

- RS support label with method "Equality-based requirement" and "Set-based requirement"

```yaml
selector:
  matchLabels:
    environment: development
  matchExpressions:
    - {key: environment, operator: In, values: [development]}
```

Deployment/RS and Update

- Example
  - labdocker/cluster:webservicelite_v1

  - labdocker/cluster:webservicelite_v1.51rc

  - labdocker/cluster:webservicelite_v1.8ga

- Example
  - Deployment “webtest”

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
```

SVC “webtest”

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development

  type: NodePort
  ports:
   - port: 5000
     name: http
     targetPort: 5000
     protocol: TCP
     nodePort: 32500
```

Deployment/RS and Update

- Create deployment “webtest” and ReplicaSet (RS)

```
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl create -f webtest_deploy.yml --record
deployment "webtest" created
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl get deployment webtest
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   3         3         3            3           10s
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl get rs
NAME                 DESIRED   CURRENT   READY     AGE
webtest-4261491039   3         3         3         16s
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl get pods
NAME                       READY     STATUS    RESTARTS   AGE
webtest-4261491039-b7gkv   1/1       Running   0          20s
webtest-4261491039-p231t   1/1       Running   0          20s
webtest-4261491039-rjzk4   1/1       Running   0          20s
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$
```

- Create svc “webtest”

```
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl create -f webtest_svc.yml --record
service "webtest" created
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl get svc
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
kubernetes   10.0.0.1      <none>        443/TCP          9d
webtest      10.0.0.97     <nodes>       5000:32500/TCP   4s
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$
```

Deployment/RS and Update

- Procedure for deployment operate
  - Check existing RS (0 when create new)
  - Create new RS
  - Scale RS to designed replicas

```
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$ kubectl describe deployment/webtest
Name:                   webtest
Namespace:              default
CreationTimestamp:      Thu, 06 Jul 2017 23:17:25 +0700
Labels:                 environment=development
                        module=WebServer
                        name=web
                        owner=Praparn_L
                        version=1.0
Annotations:            deployment.kubernetes.io/revision=1
Selector:               environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  1 max unavailable, 1 max surge
Pod Template:
  Labels:       environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                version=1.0
```

```
Conditions:
  Type          Status  Reason
  ----          ------  ------
  Available     True    MinimumReplicasAvailable
OldReplicaSets: <none>
NewReplicaSet:  webtest-4261491039 (3/3 replicas created)
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath   Type            Reason                  Message
  ---------     --------        -----   ----                    -------------   --------        ------                  -------
  10m           10m             1       deployment-controller                   Normal          ScalingReplicaSet       Scaled up replica set webtest-4
261491039 to 3
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$
```

Deployment/RS and Update

- Deployment update strategy (RollingUpdate)
  - Update will start trigger when some of "spec:template" is changed

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 3
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
```

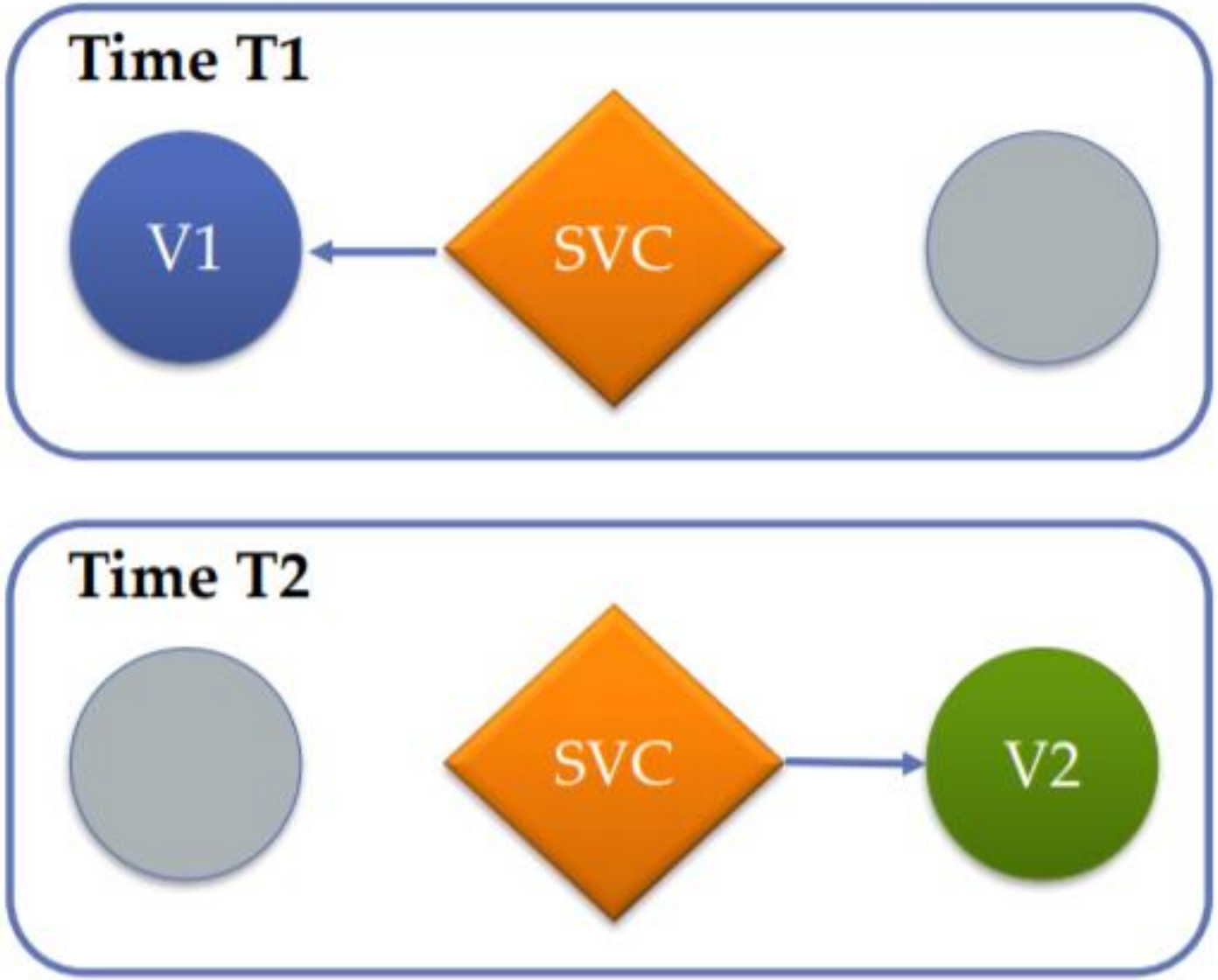
  - Step for produce update (rollout) with Each Change (Default)
    - Create new RS for template change
    - Scale down existing RS to 1 (1 max unavailable)
    - Scale new RS as designed (No more than design +1: 1 max surge)
    - Delete existing RS
  - Default deployment will allow 25% for lack (maxUnavaliable) and 25% for over design (maxSurge)

kubernetes

Deployment/RS and Update

- Let's Talk about deployment's strategies !!!
  - On deployment we have many procedure for apply rollout strategy with deployment
  - Recreate (For Development): Destroy all pods and recreate it again.
  - RollingUpdate (Default): Create new version of pods and wait for it ready before terminate existing version
  - Blue/Green: Create new deployment for "green" version and reconfigure service for point to new deployment
  - Canary: Create new deployment with same "Label" but less replicas in begin and increase as need
  - A/B: Create new deployment and let's use service mesh for select some traffic to new deployment

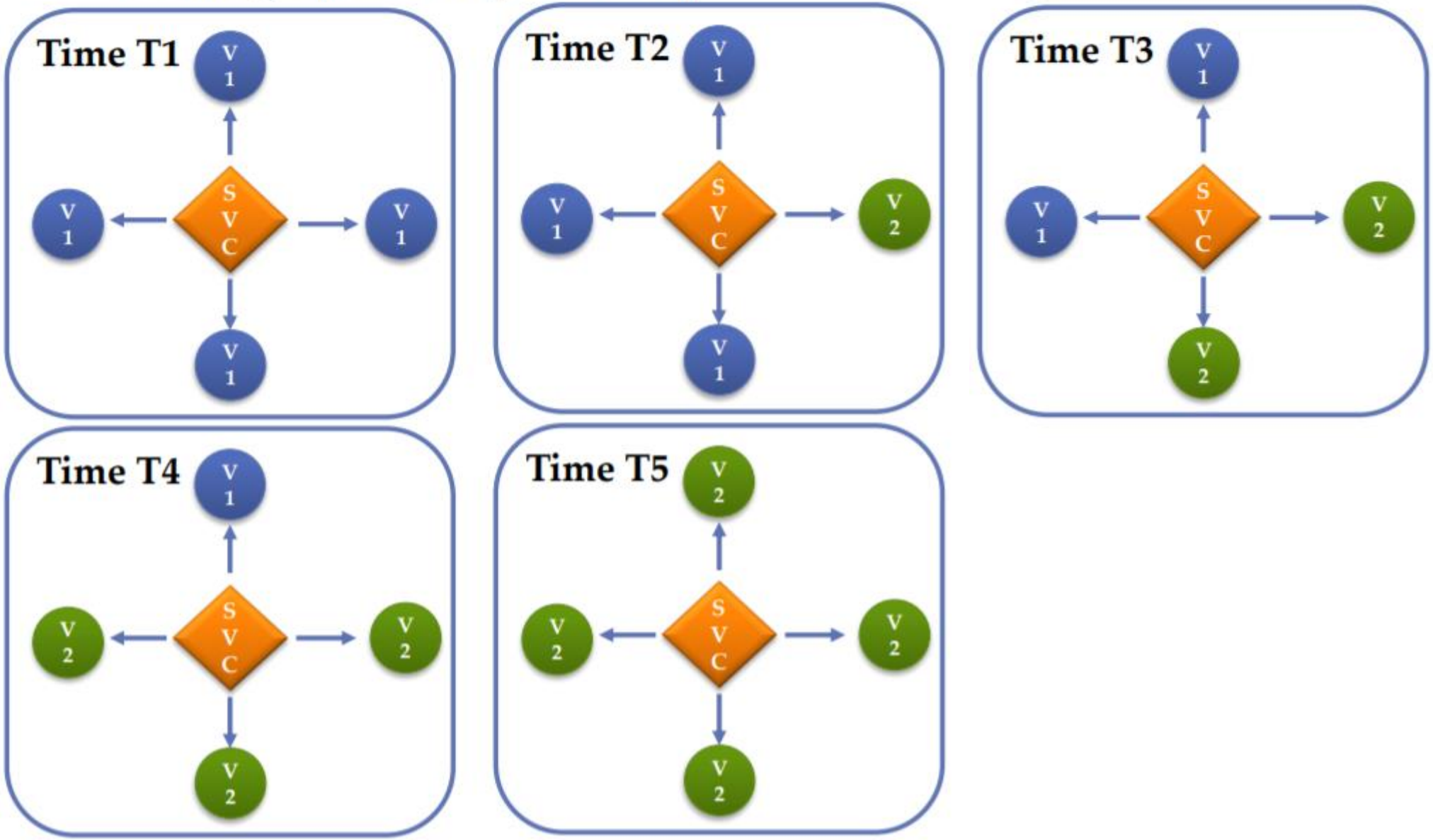
Recreate (For Development)



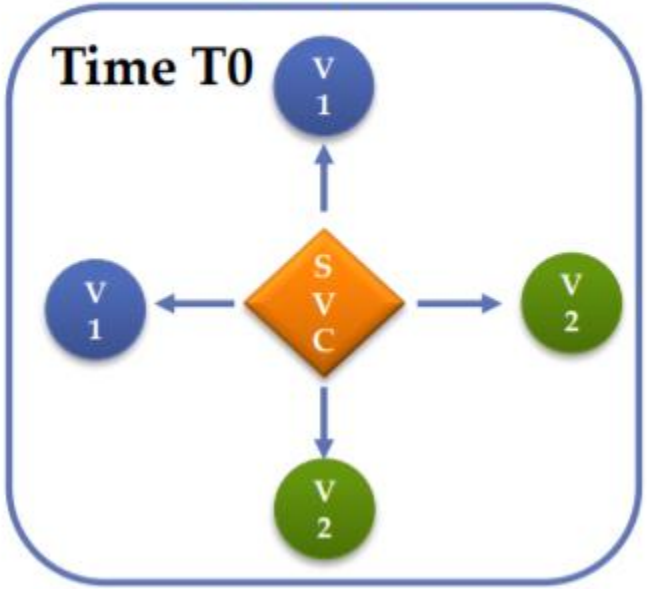
webtest_deploy.yml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  strategy:
    type: Recreate
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
```

- RollingUpdate (Default)



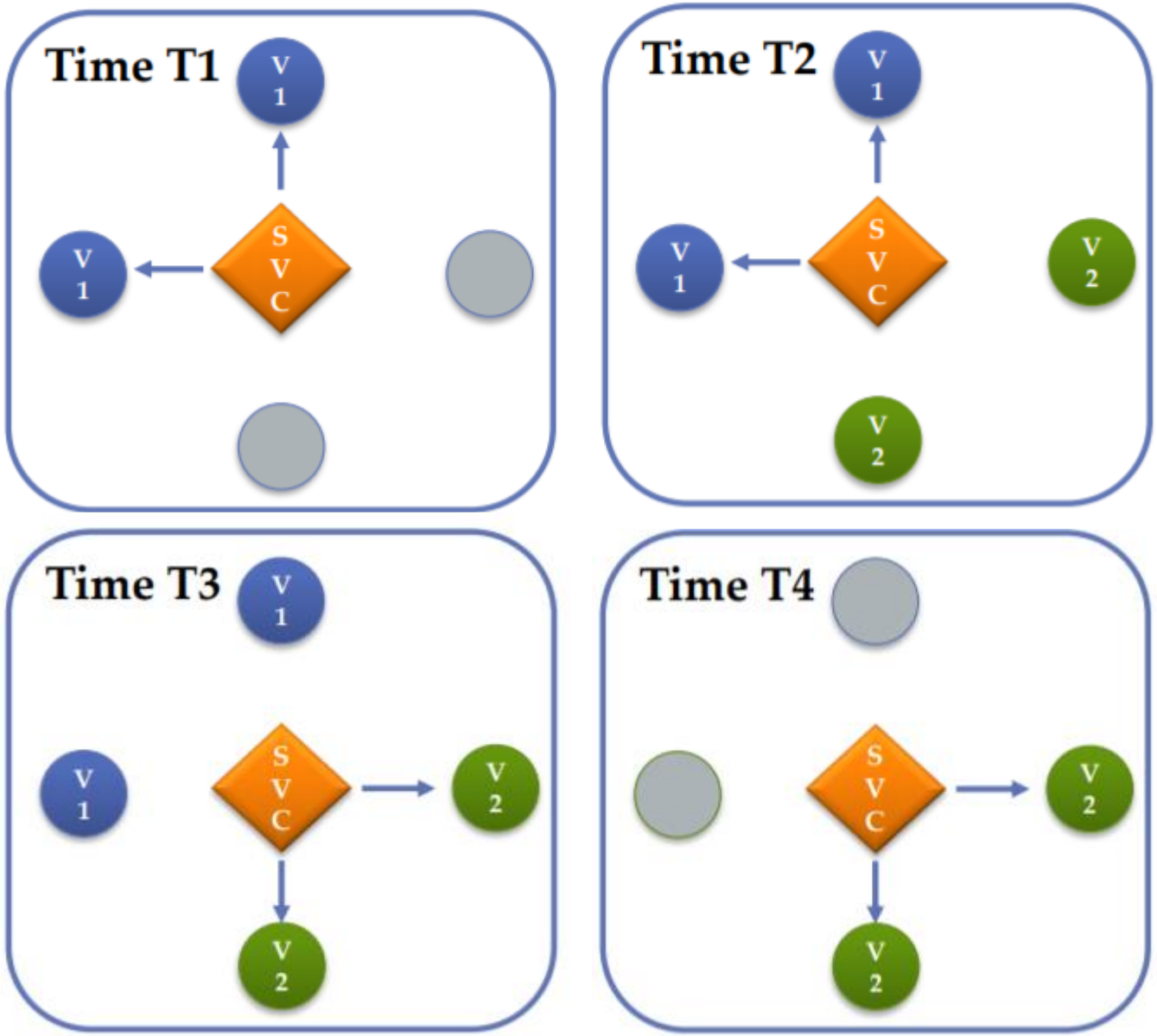
- RollingUpdate (Default)



webtest_deploy.yml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 4
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavaible: 1
      maxSurge: 1
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
```

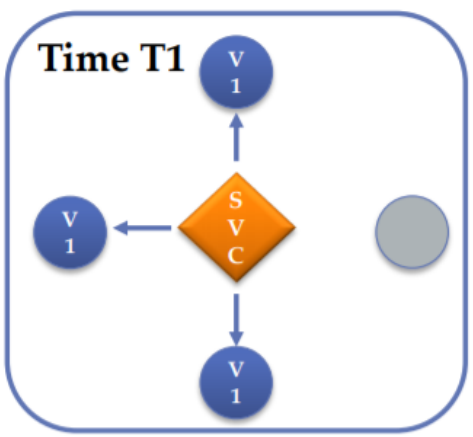
- Blue/Green

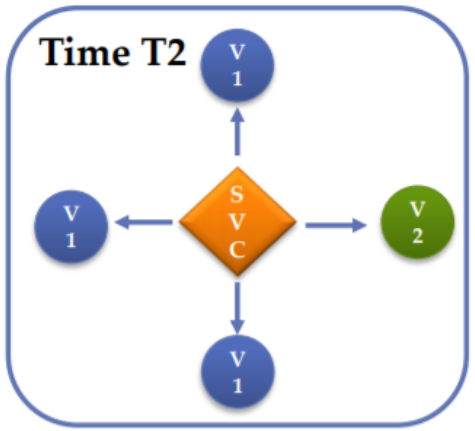


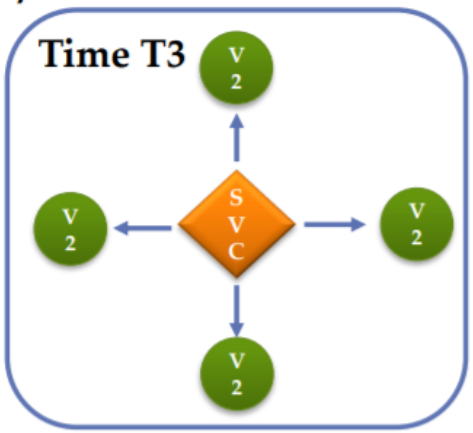
webtest_deploy.yml ● webtest_svc.yml ×

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
```

- Canary: (2 Deployment same label)







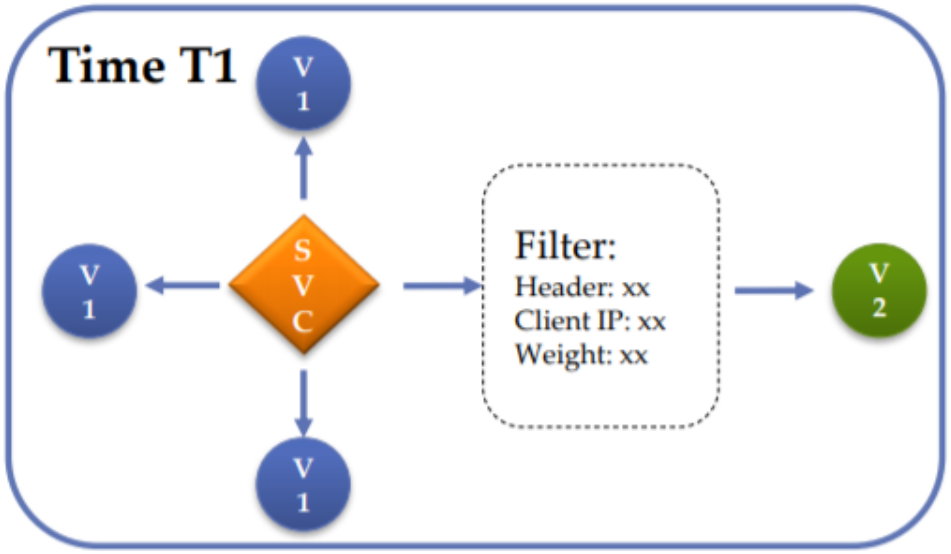

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest_V1
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 3
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest_V2
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
```

webtest_deploy.yml ● | webtest_svc.yml ×

```yaml
apiVersion: v1
kind: Service
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  selector:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
```

- A/B: (2 Deployment with Service Mesh)



AWS App Mesh

Application-level networking for all your services

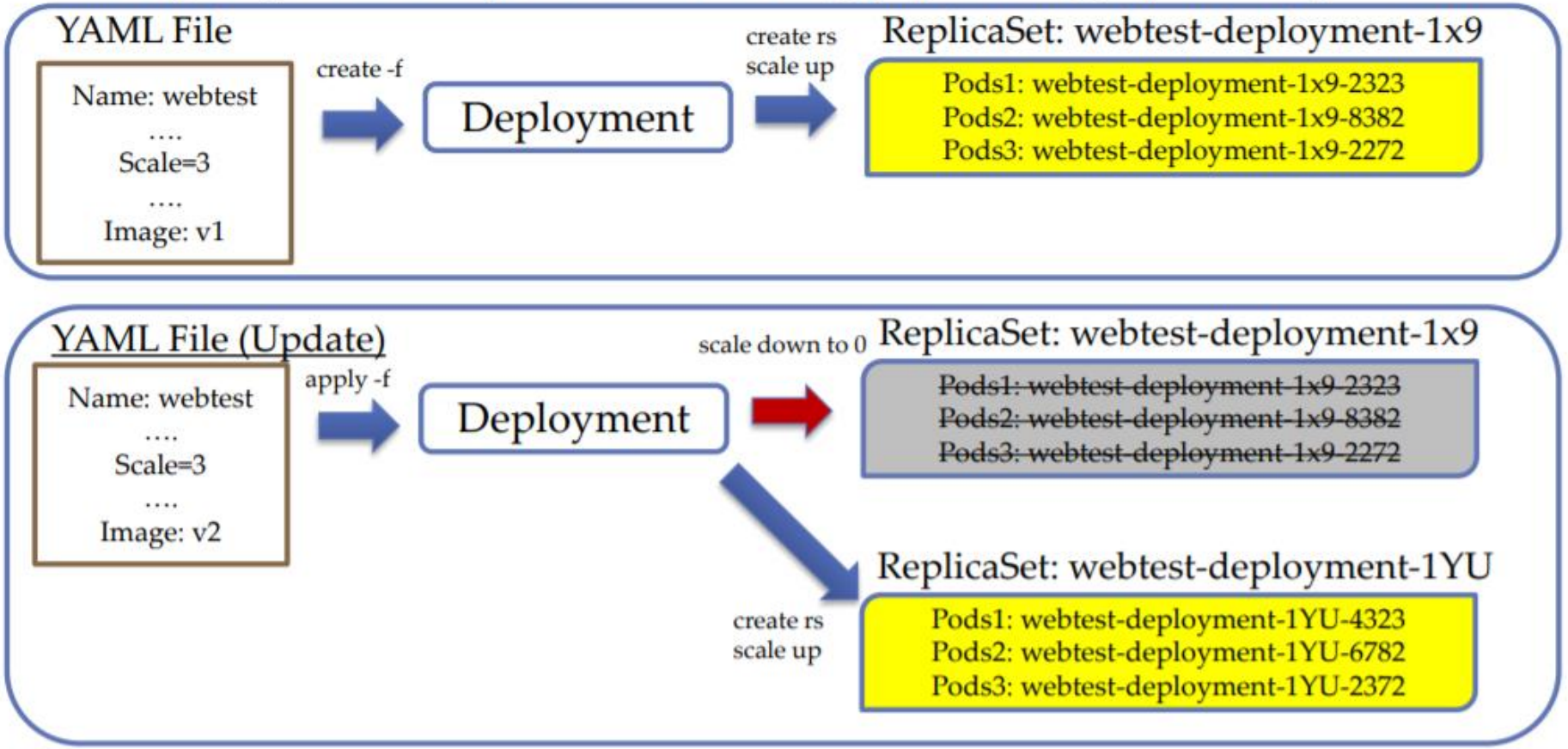
Istio

Connect, secure, control, and observe services.

LINKERD

vamp

- Deployment update strategy (RollingUpdate)



Deployment/RS and Update

- How to be grateful rollout/rollback
  - Set the rollingupdate's strategy

```
replicas: 3
strategy:
  type: RollingUpdate
rollingUpdate:
  maxUnavailable: 50%
  maxSurge: 100%
```

  - Setup "terminationGracePeriodSeconds" (After SIGTERM)

```
spec:
  terminationGracePeriodSeconds: 30
```

  - Setup "readiness" for indicate application is ready (Next chapter) and coding for set readiness fail when get "SIGTERM"
  - Setup "readinessGates" for additional healthcheck for external hookback before start open pods

Deployment/RS and Update

- Deployment update strategy (Rollout)
  - Trigger Change on Deployment
  - **Method1: Set online**

kubectl set <Property> <Type/Name> Variable=Value

    - Ex: kubectl set image deployment/webtest webtest:betaversion

  - **Method2: Edit online**

kubectl edit <Type/Name>

    - Ex: kubectl edit deployment/webtest

  - **Method3: Edit YAML File and Apply**

Edit file <FileName> .YML

kubectl apply –f <FileName>

- Check status of deployment's update (rollout)
  - Use "rollout" utility

```
kubectl rollout <Option> <Type/Name>
```

  - Ex:
    - Check status of rollout process:
      - kubectl rollout status deployment/webtest
    - Check history of rollout process:
      - Kubectl rollout history deployment/webtest
    - Rollback rollout process:
      - kubectl rollout undo deployment/webtest --to-revision=2
    - Pause/Resume process:
      - kubectl rollout pause/resume deployment/webtest

Deployment/RS and Update

- Deployment update strategy
  - Set new image on deployment

```
ubuntu@ip-10-0-1-211:~$ kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.51rc --record=true
deployment.extensions/webtest image updated
ubuntu@ip-10-0-1-211:~$ kubectl rollout status deployment/webtest
Waiting for deployment "webtest" rollout to finish: 1 old replicas are pending termination...
Waiting for deployment "webtest" rollout to finish: 1 old replicas are pending termination...
deployment "webtest" successfully rolled out
ubuntu@ip-10-0-1-211:~$ kubectl get deployment
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   3         3         3            3           98s
ubuntu@ip-10-0-1-211:~$ kubectl get rs
NAME                 DESIRED   CURRENT   READY   AGE
webtest-594bdc7f5d   3         3         3       17s
webtest-7dc5bdcdd7   0         0         0       102s
ubuntu@ip-10-0-1-211:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
webtest-594bdc7f5d-2pqwf   1/1     Running   0          22s
webtest-594bdc7f5d-fngzv   1/1     Running   0          19s
webtest-594bdc7f5d-wwkqj   1/1     Running   0          20s
ubuntu@ip-10-0-1-211:~$ kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.8ga --record=true
deployment.extensions/webtest image updated
ubuntu@ip-10-0-1-211:~$ kubectl rollout status deployment/webtest
deployment "webtest" successfully rolled out
ubuntu@ip-10-0-1-211:~$
```

  - Rollout History

Deployment/RS and Update

- Deployment update strategy
  - Rollout History

```
ubuntu@ip-10-0-1-211:~$ kubectl rollout history deployment/webtest
deployment.extensions/webtest
REVISION  CHANGE-CAUSE
1         kubectl create --filename=https://raw.githubusercontent.com/praparn/kubernetes_20180701/master/WorkShop_1.4_Deployment/webtest_deploy.yml --record=true
2         kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.51rc --record=true
3         kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.8ga --record=true

ubuntu@ip-10-0-1-211:~$
```

Deployment/RS and Update

- Deployment update strategy
  - Rollback process

```
ubuntu@ip-10-0-1-211:~$ kubectl rollout undo deployment/webtest --to-revision=2
deployment.extensions/webtest
ubuntu@ip-10-0-1-211:~$ kubectl rollout status deployment/webtest
deployment "webtest" successfully rolled out
ubuntu@ip-10-0-1-211:~$ curl http://10.0.1.211:32500
<H1> Welcome Page from Container Python Lab Web Version 1.51 RC </H1>Checkpoint Date/Time: Mon Oct 29 08:10:17 2018
ubuntu@ip-10-0-1-211:~$
```

  - Rollout History

```
ubuntu@ip-10-0-1-211:~$ kubectl rollout history deployment/webtest
deployment.extensions/webtest
REVISION  CHANGE-CAUSE
1         kubectl create --filename=https://raw.githubusercontent.com/praparn/kubernetes_20180701/master/WorkShop_1.4_Deployment/webtest_deploy.yml --record=true
3         kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.8ga --record=true
4         kubectl set image deployment/webtest webtest=labdocker/cluster:webservicelite_v1.51rc --record=true

ubuntu@ip-10-0-1-211:~$
```

LIVENESS AND READINESS

LIVENESS AND READINESS

Liveness and Readiness Probe

Pods are you still alright ?

- Normally kubernetes will check Pods status and process criteria for orchestrator following "restartPolicy"
  - Always (Default)
  - On-Failure
  - None

- Pods status is depend on "ContainerState" in that Pods (1-M)
  - Waiting (ContainerStateWaiting)
  - Running (ContainerStateRunning)
  - Terminated (ContainerStateTerminated)

LIVENESS AND READINESS

- Pods are you still alright ?
  - When Pods receive status from container. It will set Pods's status to inform kubelet for operate with Pods (Remain / Terminated / Restart)

  - Pods Phase (Phase Value)
    - Pending: When initial Pods and loading images for container
    - Running: Some container running but not all
    - Successed: All container fullfill running
    - Failed: All container exit fail
    - Unknow: Can not monitor state

- Do we need more specific health-check?
  - Depend on container (application) fail condition.
  - If container fail condition will make process "crash" or effect to container down/unhealth. This no need to operate more check.
  - If container fail condition may occur inside application that possible process still online. This need more health check

LIVENESS AND READINESS

- Container probe process
  - Kubernetes have function for make properly make sure that container (inside Pods) still healthy by three options
    - ExecAction: Execute some command inside container (expect result: return 0)
    - TCPSocketAction: Start TCP communication on specific port of container (expect result: port open)
    - HTTPGetAction: Send http/https request to specific port and path (expect result: return 200 for OK)
  - Probe's result
    - Success
    - Failed
    - Unknown
  - livenessProbe and readinessProbe Difference
    - livenessProbe: check status of container is running properly or not ?
    - readinessProbe: check readiness for process service or not ?

LIVENESS AND READINESS

Liveness and Readiness Probe

- Container Lifecycle Hooks
  - On kubernetes we can provide some process to hook on container Life cycle that kubernetes support in 2 event
  - PostStart:
    - Run some activity immediate when container is created (No guarantee to execute before "ENTRYPOINT" of container)
  - PreStop:
    - Run some activity immediate before container terminate (Delete, Liveness Fail) This is will blocking to do this task before terminate

LIVENESS AND READINESS

```
41            - name: ES_JAVA_OPTS
42              value: -Xms5g -Xmx5g
43            - name: bootstrap.memory_lock
44              value: "false"
45            - name: cluster.name
46              value: pam-es-docker-cluster
47            - name: xpack.security.enabled
48              value: "false"
49            image: "3dsinteractive/elasticsearch-std:6.6"
50            lifecycle:
51              postStart:
52                exec:
53                  command: ["/bin/sh", "-c", "chown -R 1000:1000 /usr/share/elasticsearch/data && chmod -R 777 /usr/share/elasticsearch/data"]
54
55  #           imagePullPolicy: Always
56            ports:
57              - name: "elastic9200"
58                containerPort: 9200
59              - name: "elastic9300"
60                containerPort: 9300
61            resources:
```

```
1   apiVersion: app/v1
2   kind: Deployment
3   metadata:
4     name: nginx
5   spec:
6     template:
7       metadata:
8         labels:
9           app: nginx
10      spec:
11        containers:
12        - name: nginx
13          image: nginx
14          ports:
15          - containerPort: 80
16          lifecycle:
17            preStop:
18              exec:
19                command: ["nginx","-s","quit"]
```

LIVENESS AND READINESS

ExecAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1
          ports:
          - containerPort: 5000
            protocol: TCP
          readinessProbe:
            exec:
              command:
                - cat
                - /usr/src/app/main.py
            initialDelaySeconds: 15
            periodSeconds: 5
          livenessProbe:
            exec:
              command:
                - cat
                - /usr/src/app/main.py
            initialDelaySeconds: 15
            periodSeconds: 15
```

```
deployment "webtest" created
service "webtest" created
praparns-MacBook-Pro% kubectl get svc/webtest
NAME      CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
webtest   10.0.0.127    <nodes>       5000:32500/TCP   20s
praparns-MacBook-Pro% kubectl get deployment/webtest
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         1         1            1           33s
praparns-MacBook-Pro% kubectl describe deployment/webtest
Name:                   webtest
Namespace:              default
CreationTimestamp:      Sat, 16 Sep 2017 22:04:07 +0700
Labels:                 environment=development
                        module=WebServer
                        name=web
                        owner=Praparn_L
                        version=1.0
Annotations:            deployment.kubernetes.io/revision=1
Selector:               environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
Replicas:               1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  1 max unavailable, 1 max surge
Pod Template:
  Labels:
```

192.168.99.100:32500

Apps · NMac Ked - Mac OS... · Medium · jerkins · vagrant · Mesos · Vue · docker · NGINX · Taiwan · Other Bookmarks

Welcome Page from Container Python Lab Web Version 1.00

Checkpoint Date/Time: Sat Sep 16 15:06:15 2017

LIVENESS AND READINESS

ExecAction:

```
praparns-MacBook-Pro% kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
webtest-336910061-26pdb     1/1       Running   0          4m
praparns-MacBook-Pro% kubectl exec -it webtest-336910061-26pdb sh
/usr/src/app # ls
docker-compose.yml      dockerfile_python_lite  nginx.conf
dockerfile_nginx        main.py                 requirements.txt
dockerfile_python       mainlite.py             requirementslite.txt
/usr/src/app # rm main.py
/usr/src/app # exit
praparns-MacBook-Pro%
```

ExecAction:

```
praparns-MacBook-Pro% kubectl get pods
NAME                        READY     STATUS    RESTARTS   AGE
webtest-336910061-26pdb     1/1       Running   1          7m
praparns-MacBook-Pro% kubectl describe pods/webtest-336910061-26pdb
Name:           webtest-336910061-26pdb
Namespace:      default
Node:           minikube/192.168.99.100
Start Time:     Sat, 16 Sep 2017 22:04:07 +0700
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                pod-template-hash=336910061
                version=1.0
Annotations:    kubernetes.io/created-by={"kind":"SerializedReference","apiVersion":"v1","reference":{"kind":"Repli
me":"webtest-336910061","uid":"4992f987-9af0-11e7-bc1a-080027fb95be",...
Status:         Running
```

LIVENESS AND READINESS

LIVENESS AND READINESS

ExecAction:

LIVENESS AND READINESS

ExecAction:

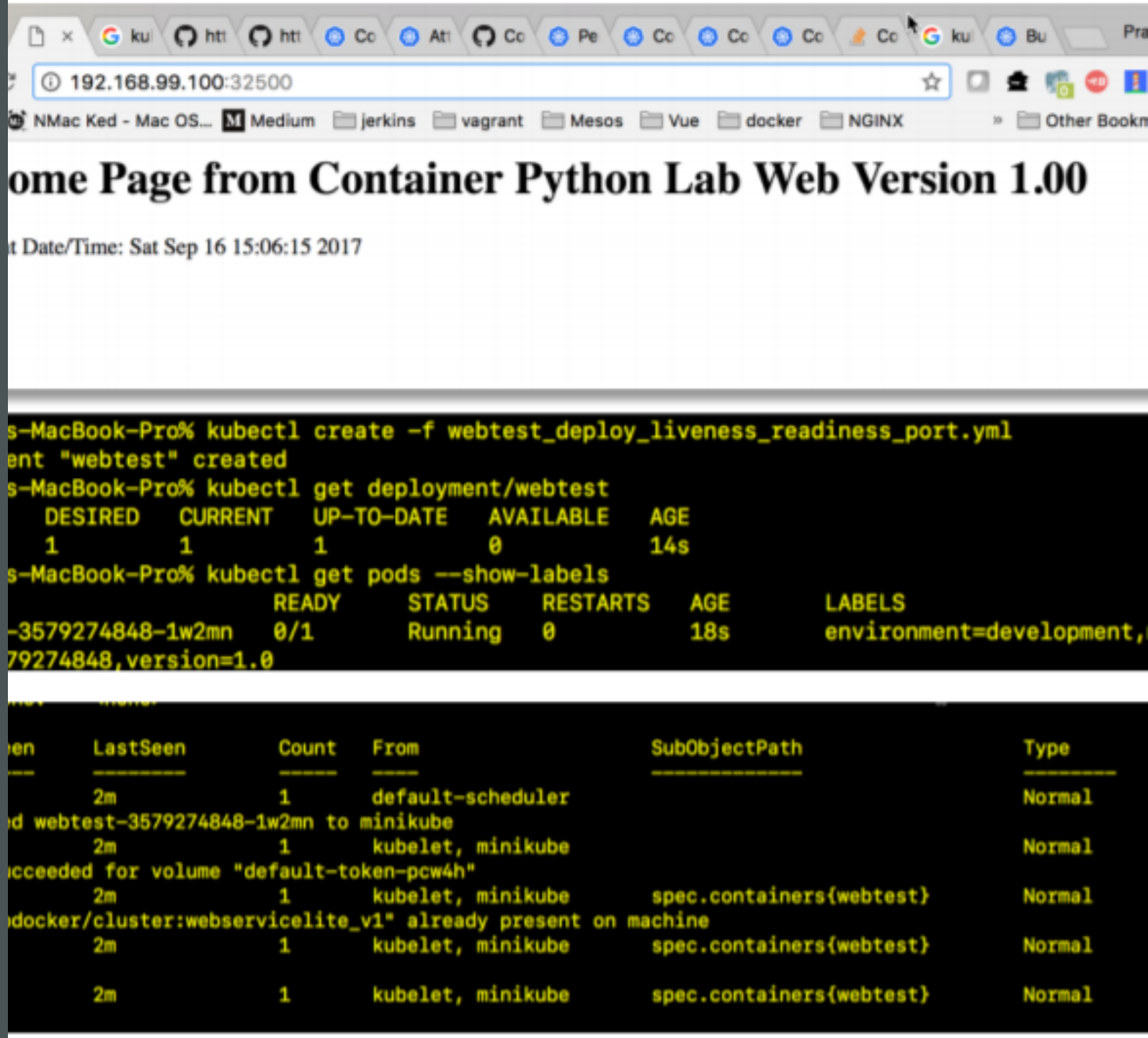
LIVENESS AND READINESS

- TCPSocketAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      containers:
      - name: webtest
        image: labdocker/cluster:webservicelite_v1
        ports:
        - name: webservice
          containerPort: 5000
          protocol: TCP
        readinessProbe:
          tcpSocket:
            port: 5000
          initialDelaySeconds: 15
          periodSeconds: 5
        livenessProbe:
          tcpSocket:
            port: 5000
          initialDelaySeconds: 15
          periodSeconds: 15
```

LIVENESS AND READINESS

- TCPSocketAction:



LIVENESS AND READINESS

- TCPSocketAction:

instruction.txt | webtest_deploy_liveness_readiness_port.yml

```
21      spec:
22        containers:
23          - name: webtest
24            image: labdocker/cluster:webservicelite_v1
25            ports:
26            - name: webservice
27              containerPort: 5000
28              protocol: TCP
29            readinessProbe:
30              tcpSocket:
31                port: 3000
32              initialDelaySeconds: 15
33              periodSeconds: 5
34            livenessProbe:
35              tcpSocket:
36                port: 3000
37              initialDelaySeconds: 15
38              periodSeconds: 15
```

```
praparns-MacBook-Pro% kubectl apply -f webtest_deploy_liveness_readiness_port.yml
deployment "webtest" configured
praparns-MacBook-Pro% kubectl get pods
NAME                       READY     STATUS    RESTARTS   AGE
webtest-1810247028-x5xdf   0/1       Running   0          13s
praparns-MacBook-Pro% kubectl describe pods/webtest-1810247028-x5xdf
Name:           webtest-1810247028-x5xdf
Namespace:      default
Node:           minikube/192.168.99.100
Start Time:     Sat, 16 Sep 2017 22:32:18 +0700
Labels:         environment=development
                module=WebServer
                name=web
                owner=Praparn_L
                pod-template-hash=1810247028
                version=1.0
Annotations:    kubernetes.io/created-by={"kind":"SerializedReference","apiVersion":"v1","reference":{"kind":"ReplicaSet","namespace":"default","na
me":"webtest-1810247028","uid":"38ea7253-9af4-11e7-bc1a-080027fb95be"...
```

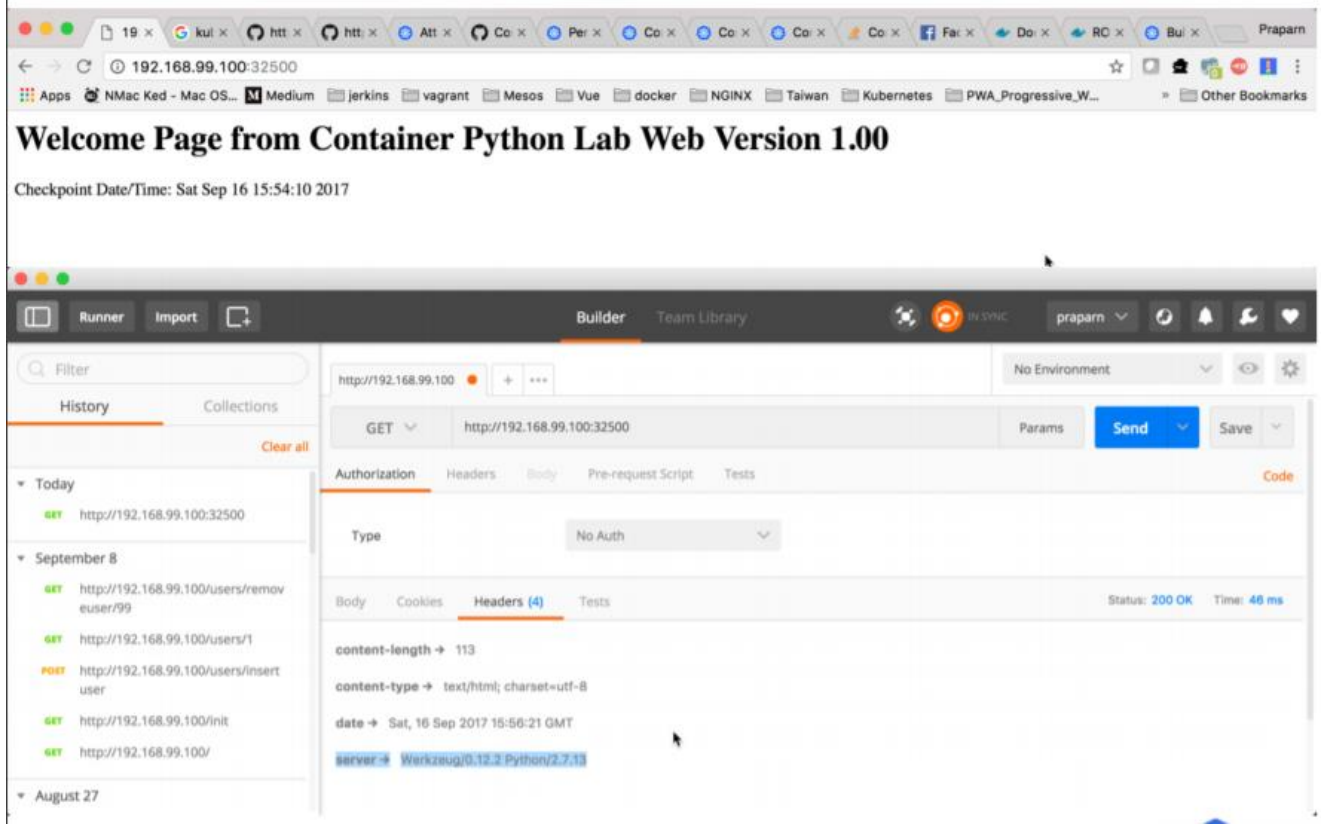
LIVENESS AND READINESS

- TCPSocketAction:

```
Events:
  FirstSeen     LastSeen        Count   From                SubObjectPath                   Type          Reason                  Message
  ---------     --------        -----   ----                -------------                   --------      ------                  -------
  4m            4m              1       default-scheduler                                   Normal        Scheduled               Successfull
y assigned webtest-1810247028-x5xdf to minikube
  4m            4m              1       kubelet, minikube                                   Normal        SuccessfulMountVolume   MountVolume
.SetUp succeeded for volume "default-token-pcw4h"
  4m            38s             6       kubelet, minikube   spec.containers{webtest}        Normal        Pulled                  Container i
mage "labdocker/cluster:webservicelite_v1" already present on machine
  4m            38s             6       kubelet, minikube   spec.containers{webtest}        Normal        Created                 Created con
tainer
  4m            38s             6       kubelet, minikube   spec.containers{webtest}        Normal        Started                 Started con
tainer
  3m            38s             5       kubelet, minikube   spec.containers{webtest}        Normal        Killing                 Killing con
tainer with id docker://webtest:pod "webtest-1810247028-x5xdf_default(38f840f4-9af4-11e7-bc1a-080027fb95be)" container "webtest" is unhealthy, it w
ill be killed and re-created.
  4m            9s              12      kubelet, minikube   spec.containers{webtest}        Warning       Unhealthy               Liveness pr
obe failed: dial tcp 172.17.0.6:3000: getsockopt: connection refused
  4m            4s              37      kubelet, minikube   spec.containers{webtest}        Warning       Unhealthy               Readiness p
robe failed: dial tcp 172.17.0.6:3000: getsockopt: connection refused
```

LIVENESS AND READINESS

- HTTPGetAction:



LIVENESS AND READINESS

- HTTPGetAction:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
```

```yaml
spec:
  containers:
    - name: webtest
      image: labdocker/cluster:webservicelite_v1
      ports:
      - name: webservice
        containerPort: 5000
        protocol: TCP
      readinessProbe:
        httpGet:
          # Optional "host: my-host" for set specific ip of Pods
          # Optional "scheme: HTTPS"
          #schema: HTTPS
          path: /
          port: webservice
          httpHeaders:
            - name: server
              value: "Werkzeug/0.12.2 Python/2.7.13"
        initialDelaySeconds: 15
        periodSeconds: 5
        timeoutSeconds: 10
        successThreshold: 1
        failureThreshold: 3
      livenessProbe:
        httpGet:
          # Optional "host: my-host" for set specific ip of Pods
          # Optional "scheme: HTTPS"
          #schema: HTTPS
          path: /
          port: webservice
          httpHeaders:
            - name: server
              value: "Werkzeug/0.12.2 Python/2.7.13"
        initialDelaySeconds: 15
        periodSeconds: 5
        timeoutSeconds: 10
        successThreshold: 1
        failureThreshold: 3
```

- HTTPGetAction:

```
praparns-MacBook-Pro% kubectl create -f webtest_deploy_liveness_readiness_http.yml
        kubectl get deployment/webtest
        kubectl get pods --show-labels
deployment "webtest" created
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         0         0            0           0s
NAME                       READY     STATUS              RESTARTS   AGE       LABELS
webtest-1098172621-zvd22   0/1       ContainerCreating   0          0s        environment=development,module=WebServer,name=web,owner=Praparn_L,pod
-template-hash=1098172621,version=1.0
```

```
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type            Reason                  Message
  ---------     --------        -----   ----                    -------------                   --------        ------                  -------
  9m            9m              1       default-scheduler                                       Normal          Scheduled               Successfull
y assigned webtest-1098172621-zvd22 to minikube
  9m            9m              1       kubelet, minikube                                       Normal          SuccessfulMountVolume   MountVolume
.SetUp succeeded for volume "default-token-pcw4h"
  9m            9m              1       kubelet, minikube       spec.containers{webtest}        Normal          Pulled                  Container i
mage "labdocker/cluster:webservicelite_v1" already present on machine
  9m            9m              1       kubelet, minikube       spec.containers{webtest}        Normal          Created                 Created con
tainer
  9m            9m              1       kubelet, minikube       spec.containers{webtest}        Normal          Started                 Started con
tainer
```

LIVENESS AND READINESS

LIVENESS AND READINESS

- HTTPGetAction:

```
        readinessProbe:
          httpGet:
            # Optional "host: my-host" for set specific ip of Pods
            # Optional "scheme: HTTPS"
            #schema: HTTPS
            path: /init
            port: webservice
            httpHeaders:
              - name: server
                value: "Werkzeug/0.12.2 Python/2.7.13"
          initialDelaySeconds: 15
          periodSeconds: 5
          timeoutSeconds: 5
          successThreshold: 1
          failureThreshold: 3
        livenessProbe:
          httpGet:
            # Optional "host: my-host" for set specific ip of Pods
            # Optional "scheme: HTTPS"
            #schema: HTTPS
            path: /init
            port: webservice
            httpHeaders:
              - name: server
                value: "Werkzeug/0.12.2 Python/2.7.13"
          initialDelaySeconds: 15
          periodSeconds: 5
          timeoutSeconds: 5
          successThreshold: 1
          failureThreshold: 3
```

```
praparns-MacBook-Pro% kubectl apply -f webtest_deploy_liveness_readiness_http.yml
deployment "webtest" configured
praparns-MacBook-Pro% kubectl get deployment/webtest
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         1         1            0           14m
praparns-MacBook-Pro% kubectl get deployment/webtest
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
webtest   1         1         1            0           15m
praparns-MacBook-Pro% kubectl get pods
NAME                       READY     STATUS    RESTARTS   AGE
webtest-692878837-fxpw4    0/1       Running   3          1m
```

- HTTPGetAction:

```
Events:
  FirstSeen     LastSeen        Count   From                    SubObjectPath                   Type            Reason                  Message
  ---------     --------        -----   ----                    -------------                   --------        ------                  -------
  1m            1m              1       default-scheduler                                       Normal          Scheduled               Successful
y assigned webtest-692878837-fxpw4 to minikube
  1m            1m              1       kubelet, minikube                                       Normal          SuccessfulMountVolume   MountVolu
.SetUp succeeded for volume "default-token-pcw4h"
  1m            13s             5       kubelet, minikube       spec.containers{webtest}        Normal          Pulled                  Container
mage "labdocker/cluster:webservicelite_v1" already present on machine
  1m            13s             9       kubelet, minikube       spec.containers{webtest}        Warning         Unhealthy               Readiness
robe failed: HTTP probe failed with statuscode: 404
  1m            13s             9       kubelet, minikube       spec.containers{webtest}        Warning         Unhealthy               Liveness p
obe failed: HTTP probe failed with statuscode: 404
  1m            13s             4       kubelet, minikube       spec.containers{webtest}        Normal          Killing                 Killing co
tainer with id docker://webtest:pod "webtest-692878837-fxpw4_default(4f6f205f-9b00-11e7-bc1a-080027fb95be)" container "webtest" is unhealthy, it w
ll be killed and re-created.
  1m            12s             5       kubelet, minikube       spec.containers{webtest}        Normal          Created                 Created co
tainer
  1m            12s             5       kubelet, minikube       spec.containers{webtest}        Normal          Started                 Started co
tainer
praparns-MacBook-Pro%
```

LIVENESS AND READINESS

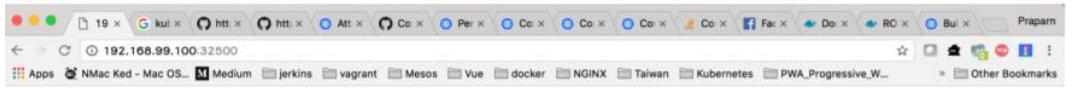
LIVENESS AND READINESS

- ReadinessGate: (New feature 1.14)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: webtest2
  labels:
    name: web
    owner: Praparn_L
    version: "1.0"
    module: WebServer
    environment: development
spec:
  replicas: 1
  selector:
    matchLabels:
      name: web
      owner: Praparn_L
      version: "1.0"
      module: WebServer
      environment: development
  template:
    metadata:
      labels:
        name: web
        owner: Praparn_L
        version: "1.0"
        module: WebServer
        environment: development
    spec:
      readinessGates:
      - conditionType: webtest1.kubernetes labthailand.com/python1
      containers:
        - name: webtest
          image: labdocker/cluster:webservicelite_v1.51rc
          ports:
          - containerPort: 5000
            protocol: TCP
          readinessProbe:
            exec:
              command:
                - cat
                - /usr/src/app/main.py
            initialDelaySeconds: 15
            periodSeconds: 5
```

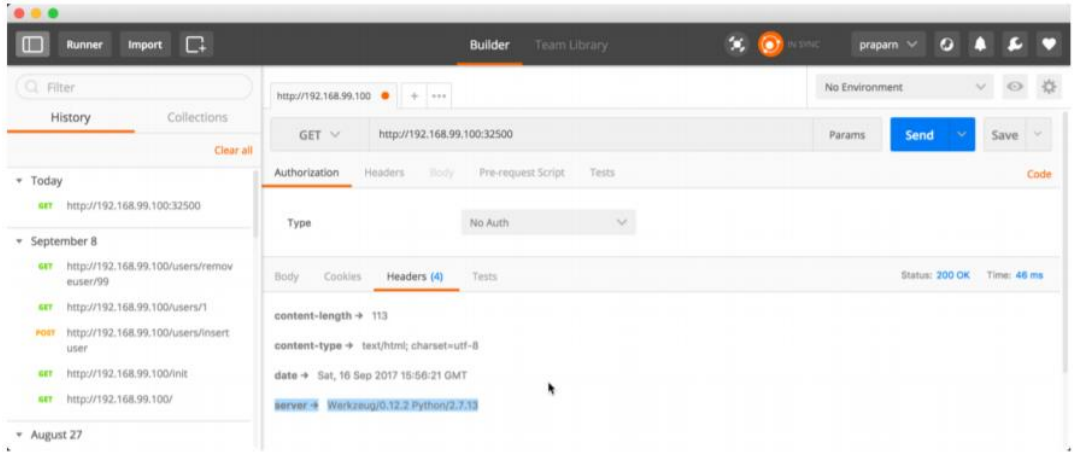
```yaml
Kind: Pod
…
spec:
  readinessGates:
  - webtest1.kubernetes labthailand.com/python1
…
status:
  conditions:
  - lastProbeTime: null
    lastTransitionTime: 2019-04-13T00:00:00Z
    status: "False"
    type: Ready
  - lastProbeTime: null
    lastTransitionTime: 2019-04-13T00:00:00Z
    status: "False"
    type: webtest1.kubernetes labthailand.com/python1
  containerStatuses:
  - containerID: docker://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
    ready : true
…
```

WORKSHOP : LIVENESS AND READINESS



Welcome Page from Container Python Lab Web Version 1.00

Checkpoint Date/Time: Sat Sep 16 15:54:10 2017




CONFIGMAP
AND
SECRET

CONFIGMAP AND SECRET

ConfigMap and Secret

- Make secret data and configuration great again !
- Many container need some configuration/potential data for make it work. But is it should store in image/configuration (Container,Pods, Deployment, RC etc)?
  - Root password of database
  - Environment variable
  - Custom variable
  - Path of mount volume data
  - Etc
- ConfigMap will provide central configuration for Pods operate
- Secret will encode sensitive data for keep secret

ConfigMap and Secret

- ConfigMap
  - ConfigMap belong to namespace scope
  - Option to create:
    - literal values
    - From file or folder
    - YAML file

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: webmodule_configmap
  namespace: webmicroservice
  labels:
    name: "webmodule_configmap"
    owner: "Praparn_L"
    version: "1.0"
    module: "ConfigMap"
    environment: "development"
data:
  REDIS_HOST: localhost
```

kubectl create configmap <name> <option>

- Ex: “kubectl create configmap webmodule_configmap --from-literal=REDIS_HOST=localhost”
- Ex: “kubectl create –f webmodule_configmap.yml”

ConfigMap and Secret

- Secret
  - ConfigMap will encode64 algorithm
  - Option to create:
    - From file <store confidential value>
    - YAML file

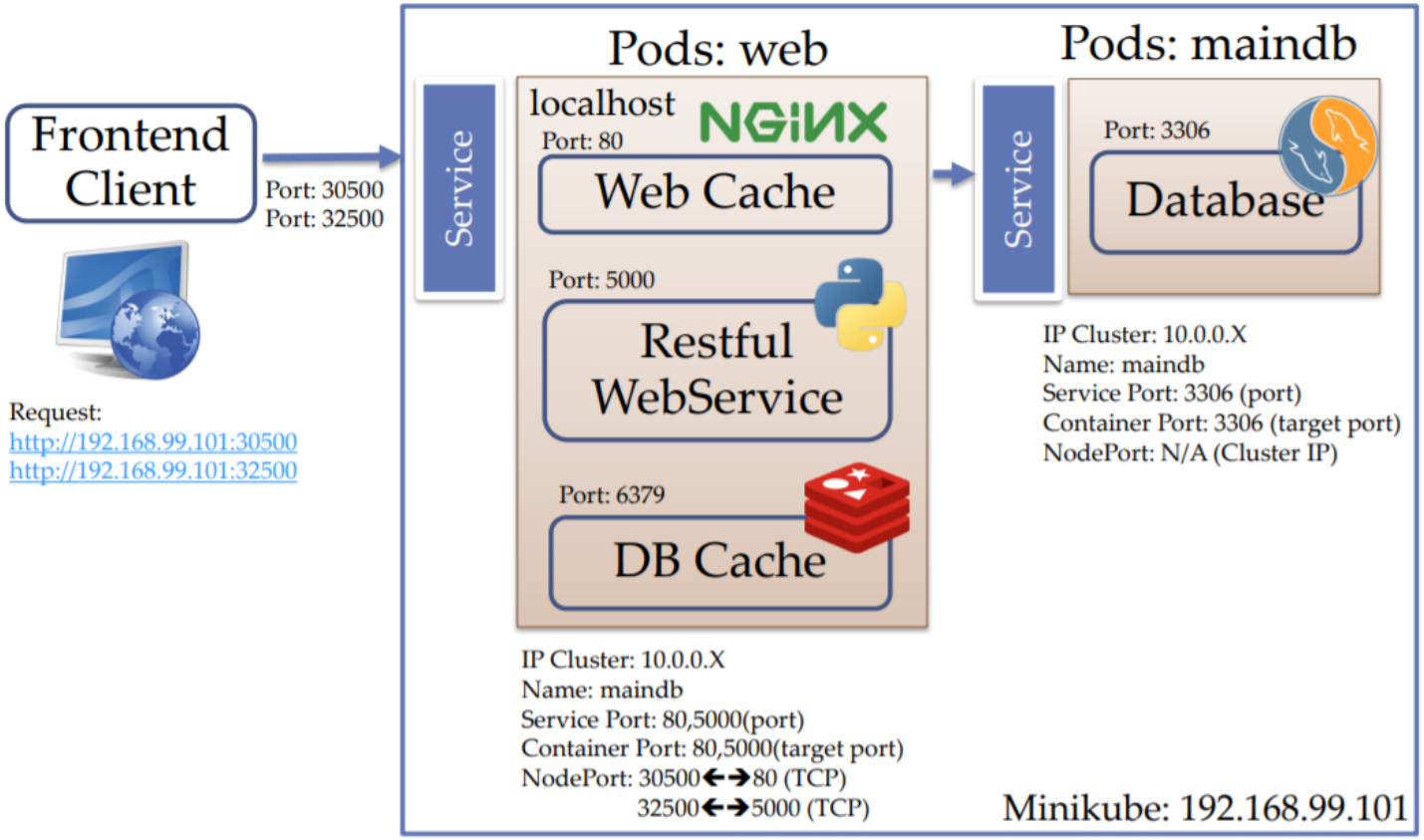
kubectl create secret generic <name> <option>

- Ex: "kubectl create secret generic databasemodule_secret --from-file=./username.txt --from-file=./password.txt"
- Ex: "kubectl create -f databasemodule_secret.yml"

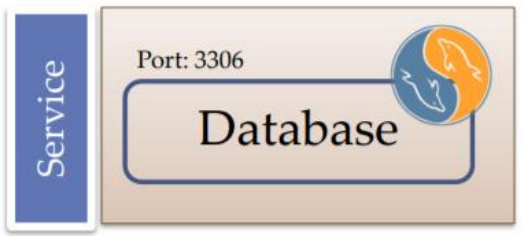
```
praparns-MBP:~ praparn$ echo -n "root"|base64
cm9vdA==
praparns-MBP:~ praparn$ echo -n "password"|base64
cGFzc3dvcmQ=
praparns-MBP:~ praparn$
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: databasemodule_secret
  namespace: webmicroservice
  labels:
    name: "databasemodule_secret"
    owner: "Praparn_L"
    version: "1.0"
    module: "Secret"
    environment: "development"
type: Opaque
data:
  username: cm9vdA==
  password: cGFzc3dvcmQ=
```

ConfigMap and Secret

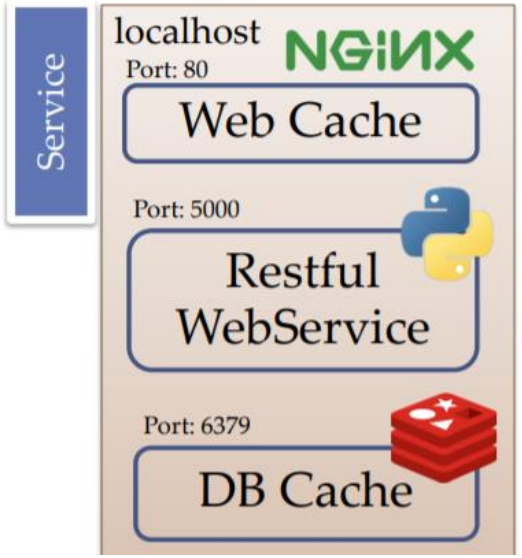


ConfigMap and Secret



Pods.yml: databasemodule_pod.yml

```
containers:
  - name: maindb
    image: labdocker/mysql:latest
    ports:
      - containerPort: 3306
        protocol: TCP
    env:
      -
        name: "MYSQL_ROOT_PASSWORD"
        value: "password"
```



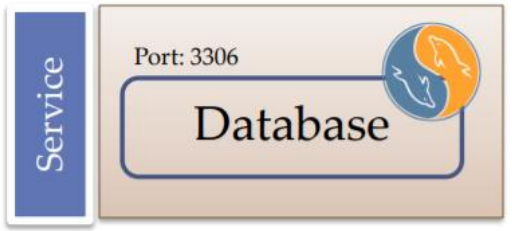
Sourcecode: main.py

```
CACHE_DB = redis.Redis(host=os.environ.get('REDIS_HOST', 'cachedb'), port=6379)
db = MySQLdb.connect("maindb","root","password")
MAIN_DB = db.cursor()
```

Pods.yml: webmodule_pod.yml

```
  - name: webservice
    image: labdocker/cluster:webservice
    env:
      - name: "REDIS_HOST"
        value: "localhost"
```

ConfigMap and Secret



Secret.yml: databasemodule_secret.yml

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: databasemodule-secret
  namespace: webmicroservice
  labels:
    name: "databasemodule-secret"
    owner: "Praparn_L"
    version: "1.0"
    module: "Secret"
    environment: "development"
type: Opaque
data:
  username: cm9vdA==
  password: cGFzc3dvcmQ=
```

Deploy.yml: databasemodule_deploy_config.yml

```yaml
apiVersion: "v1"
kind: Deployment
metadata:
  name: maindb
  labels:
    name: "maindb"
    owner: "Praparn_L"
    version: "1.0"
    module: "maindb"
    environment: "development"
spec:
  replicas: 1
  template:
    metadata:
      labels:
    spec:
      containers:
      - name: maindb
        image: labdocker/mysql:latest
        ports:
          - containerPort: 3306
            protocol: TCP
        env:
          - name: username
            valueFrom:
              secretKeyRef:
                name: databasemodule-secret
                key: username
          - name: password
            valueFrom:
              secretKeyRef:
                name: databasemodule-secret
                key: password
```

ConfigMap and Secret

Secret.yml:
databasemodule_secret.yml

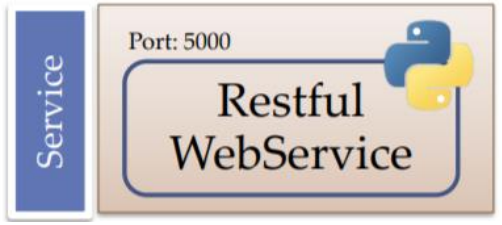
```yaml
apiVersion: v1
kind: Secret
metadata:
  name: databasemodule-secret
  namespace: webmicroservice
  labels:
    name: "databasemodule-secret"
    owner: "Praparn_L"
    version: "1.0"
    module: "Secret"
    environment: "development"
type: Opaque
data:
  username: cm9vdA==
  password: cGFzc3dvcmQ=
```

Deploy.yml:
webmodule_deploy_config.yml

```yaml
    spec:
      containers:
      - name: cachedb
        image: labdocker/redis:latest
        ports:
        - containerPort: 6379
          protocol: TCP
      - name: webservice
        image: labdocker/cluster:webservice
        env:
        - name: REDIS_HOST
          valueFrom:
            configMapKeyRef:
              name: webmodule_configmap
              key: REDIS_HOST
        - name: username
          valueFrom:
            secretKeyRef:
              name: databasemodule-secret
              key: username
        - name: password
          valueFrom:
            secretKeyRef:
              name: databasemodule-secret
              key: password
        ports:
          - containerPort: 5000
            protocol: TCP
```

ConfigMap.yml:
webmodule_configmap.yml

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: webmodule-configmap
  namespace: webmicroservice
  labels:
    name: "webmodule-configmap"
    owner: "Praparn_L"
    version: "1.0"
    module: "ConfigMap"
    environment: "development"
data:
  REDIS_HOST: localhost
```



WORKSHOP: CONFIGMAP AND SECRET

JOB AND CRON JOBS

Job and Cron Job

- Some task on kubernetes will batch process or non-interactive job
  - Update EOD Process
  - Monitor System Health
  - Calculate Balance
  - Run reindexing file/database
  - etc

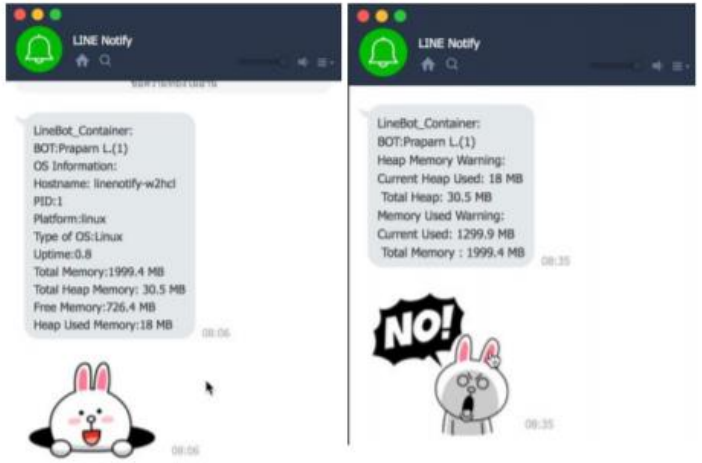
```
kubectl create –f <YAML File>
```

- "Job" on kubernetes was design to operate special purpose
  - Job will track status of complete job
  - Job will autostart new Pods when it failed or deleted
  - Job will delete Pods when job was deleted

Job.yml: job.yml

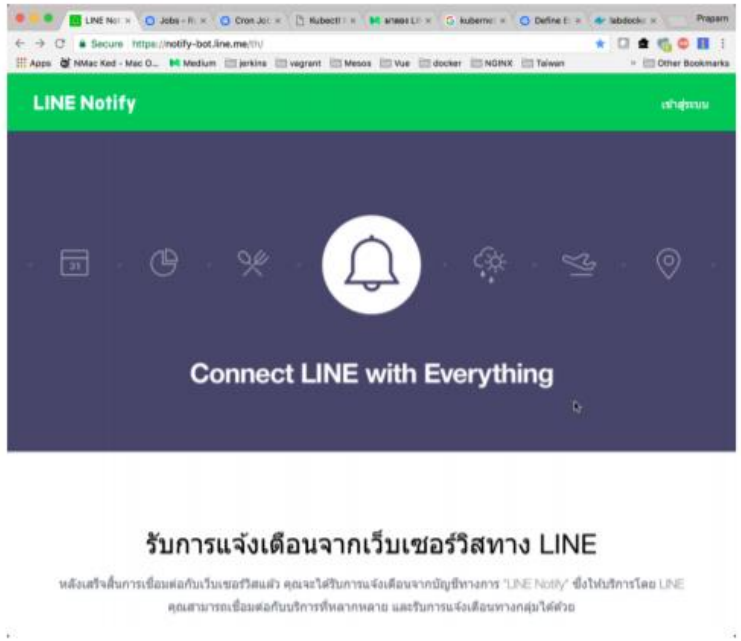
```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: linenotify
  labels:
    name: linenotify
    owner: Praparn_L
    version: "1.0"
    module: Job
    environment: development
spec:
  template:
    metadata:
      name: linenotify
    spec:
      containers:
      - name: linenotify
        image: labdocker/linenotify
        env:
        - name: TITLE
          value: "BOT:Praparn L."
        - name: INTERVAL
          value: "10000"
        - name: HEAP_HIGH
          value: "40"
        - name: MEM_HIGH
          value: "20"
        - name: SH_OS
          value: "Y"
        - name: TOKEN
          value: "EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX"
```

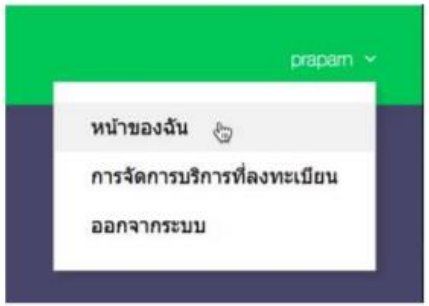
```
praparns-MacBook-Pro% kubectl create -f job.yml
job "linenotify" created
praparns-MacBook-Pro% kubectl get jobs
NAME         DESIRED   SUCCESSFUL   AGE
linenotify   1         0            1m
praparns-MacBook-Pro% kubectl get pods
NAME               READY     STATUS    RESTARTS   AGE
linenotify-w2hcl   1/1       Running   0          2m
praparns-MacBook-Pro% kubectl describe jobs/linenotify
Name:           linenotify
Namespace:      default
Selector:       controller-uid=e816fc3f-6e79-11e7-9aa8-08002763e747
Labels:         environment=development
                module=Job
                name=linenotify
                owner=Praparn_L
                version=1.0
Annotations:    <none>
```



WORKSHOP: JOB AND CRONJOB

- Task0: Generate LINE token
  - https://notify-bot.line.me





WORKSHOP: JOB AND CRONJOB

WORKSHOP: JOB AND CRONJOB

- Task1: Create Jobs for Monitor System via LINE

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: linenotify
  labels:
    name: linenotify
    owner: Praparn_L
    version: "1.0"
    module: Job
    environment: development
spec:
  template:
    metadata:
      name: linenotify
    spec:
      containers:
      - name: linenotify
        image: labdocker/linenotify
        env:
        - name: TITLE
          value: "BOT:Praparn L."
        - name: INTERVAL
          value: "10000"
        - name: HEAP_HIGH
          value: "20"
        - name: MEM_HIGH
          value: "20"
        - name: SH_OS
          value: "N"
        - name: TOKEN
          value: "EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX"
      restartPolicy: Never
```

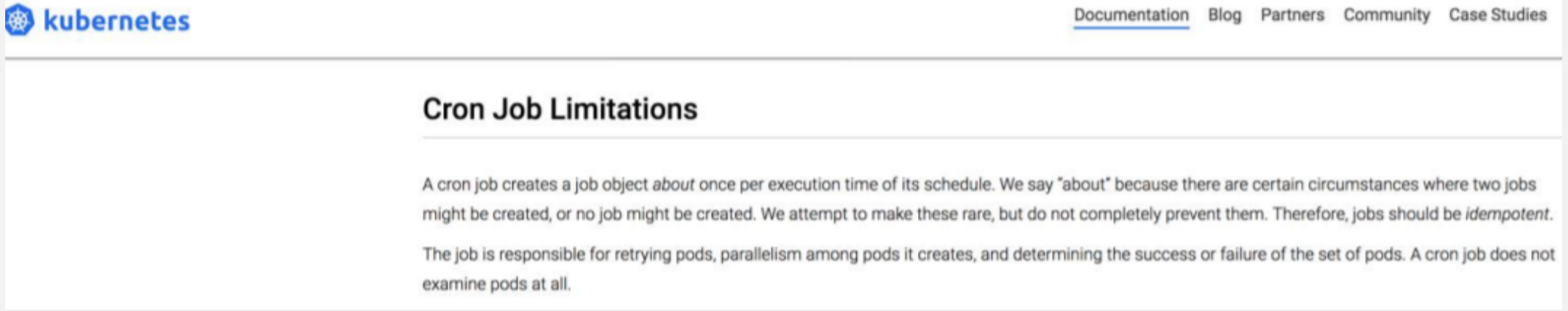
LINE

TITLE: ➔ Input Name
INTERVAL : ➔ Input Check Interval (ms)
HEAP_HIGH: ➔ % of Heap Memory Warn
MEM_HIGH: ➔ % of Memory Used Warn
SH_OS: ➔ Mode
(Show information:Y, Show only Warning Reach:N)
TOKEN: ➔ Input LINE TOKEN

Job and Cron Job

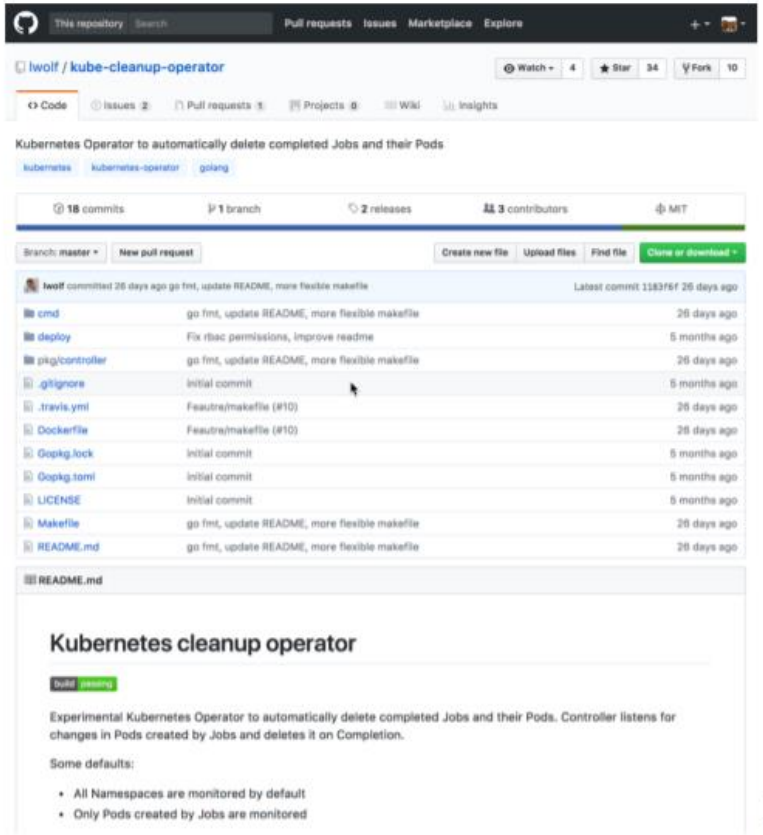
- "CronJob" is time base "Jobs" with
  - Run on specific point-in-time
  - Repeat point in time
  - etc

```
* * * * * command to be executed
- - - - -
| | | | |
| | | | ----- Day of week (0 - 7) (Sunday=0 or 7)
| | | ------- Month (1 - 12)
| | --------- Day of month (1 - 31)
| ----------- Hour (0 - 23)
------------- Minute (0 - 59)
```



Job and Cron Job

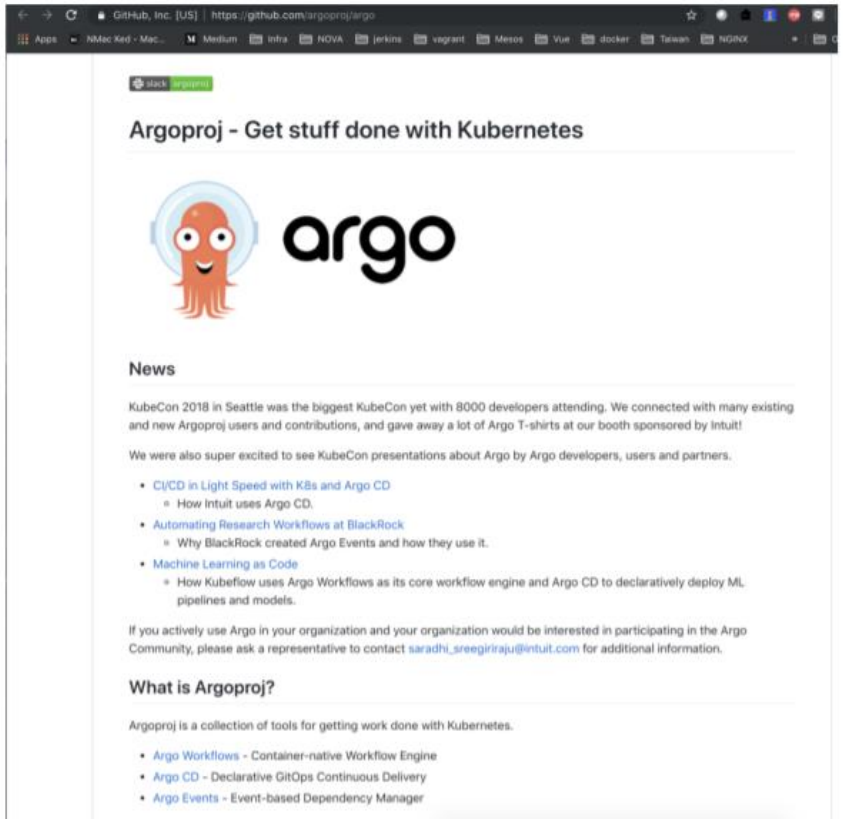
- AutoCleanup Job



https://github.com/lwolf/kube-cleanup-operator

Job and Cron Job

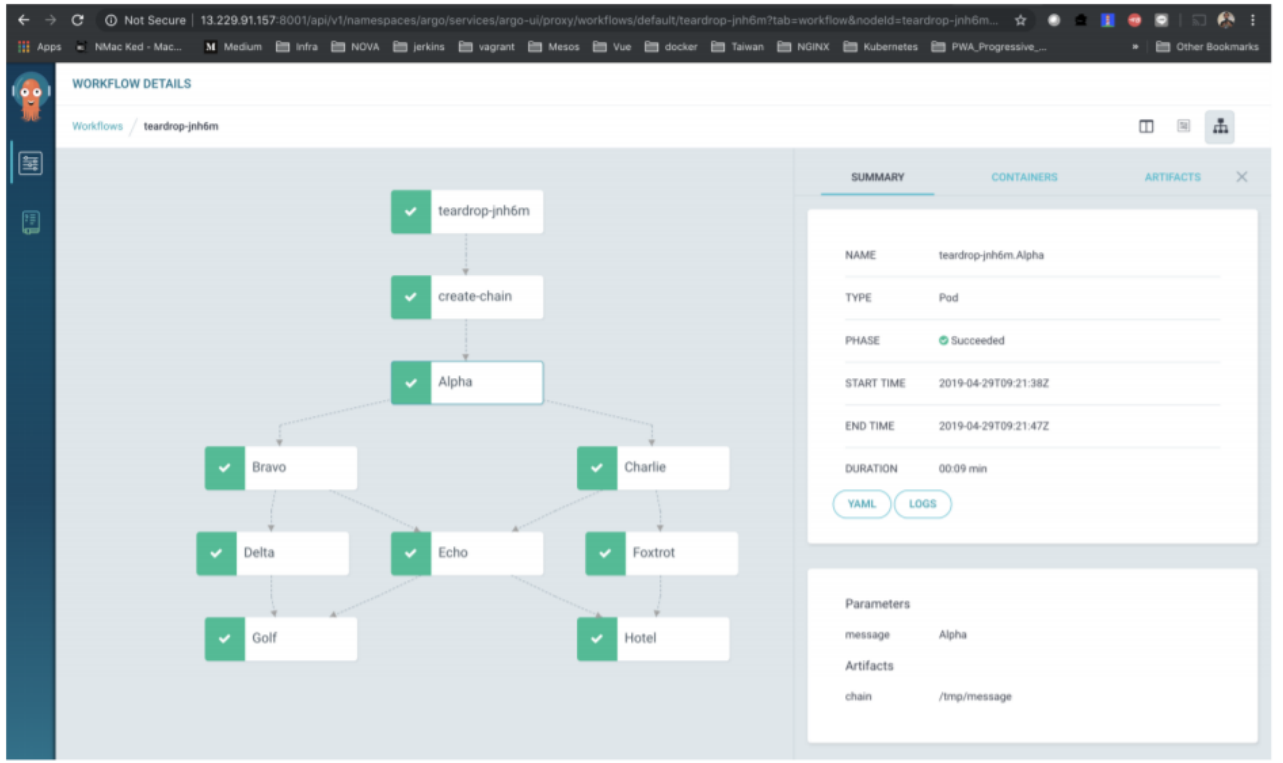
- WorkFlow Job



https://github.com/argoproj/argo

Job and Cron Job

- WorkFlow Job



HORIZONTAL POD AUTOSCALING(HPA)

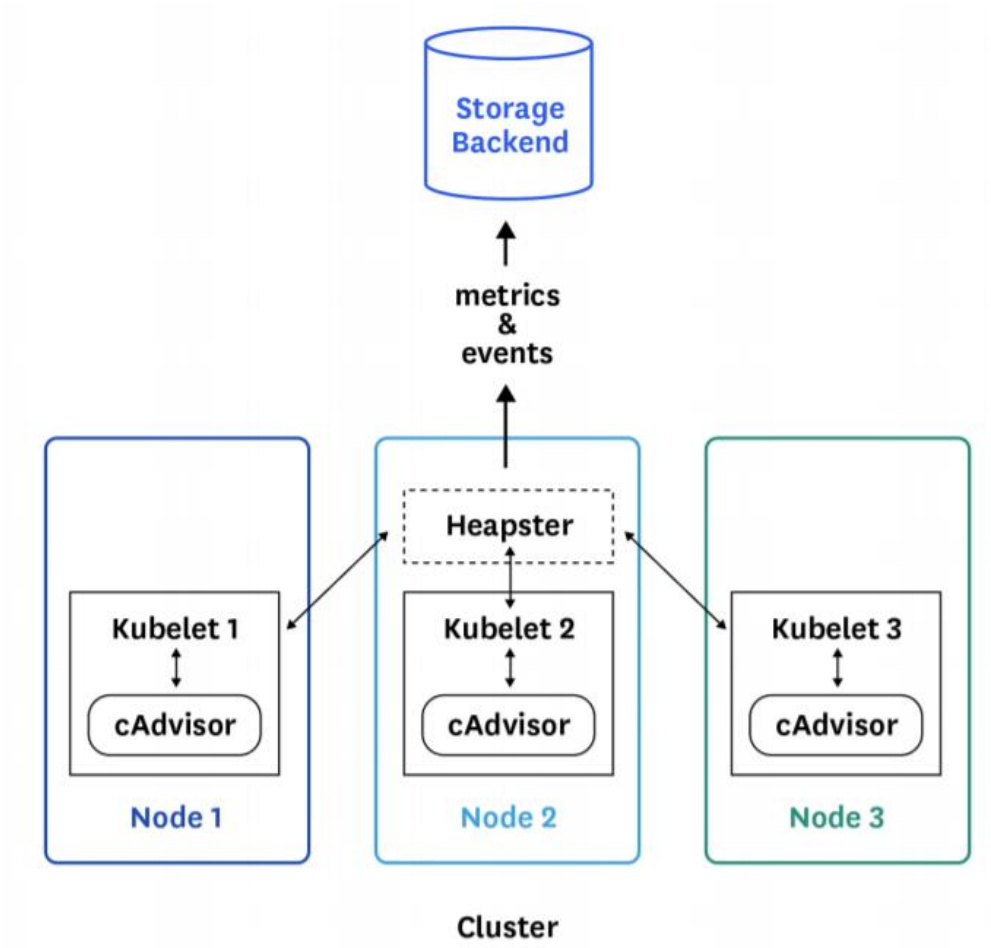
- Horizontal Pod Autoscaling (HPA)
  - The scale is easy to operate with "kubectl scale --replicas=XXX deployment/<name>" with single command
  - But..
    - How can you scale meet up/down actual required ?
  - HPA will response for monitor workload on Pods (Now base on CPU) and automatic trigger deployment to scale-up application

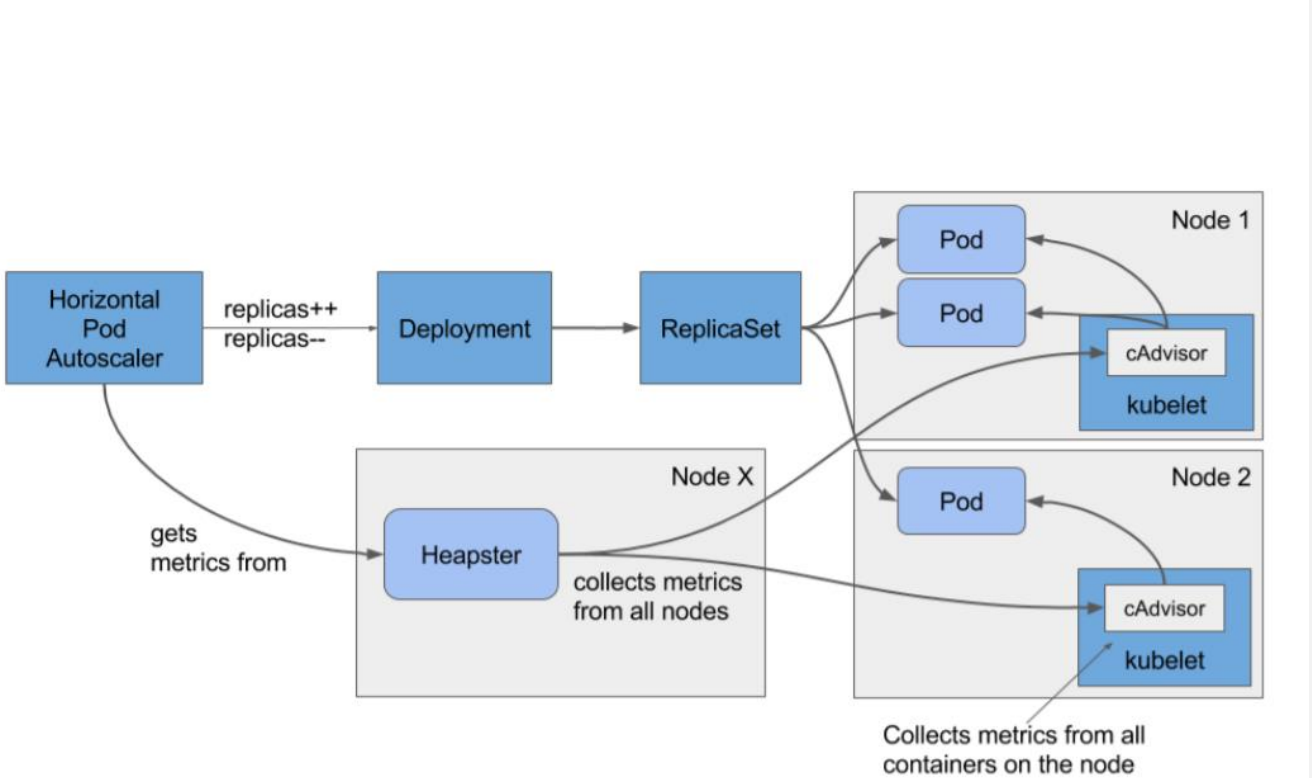
kubectl autoscale <type/name> <option:>

- Ex: kubectl autoscale deployment/webtest --min=1--max=10 --cpu-percent=80

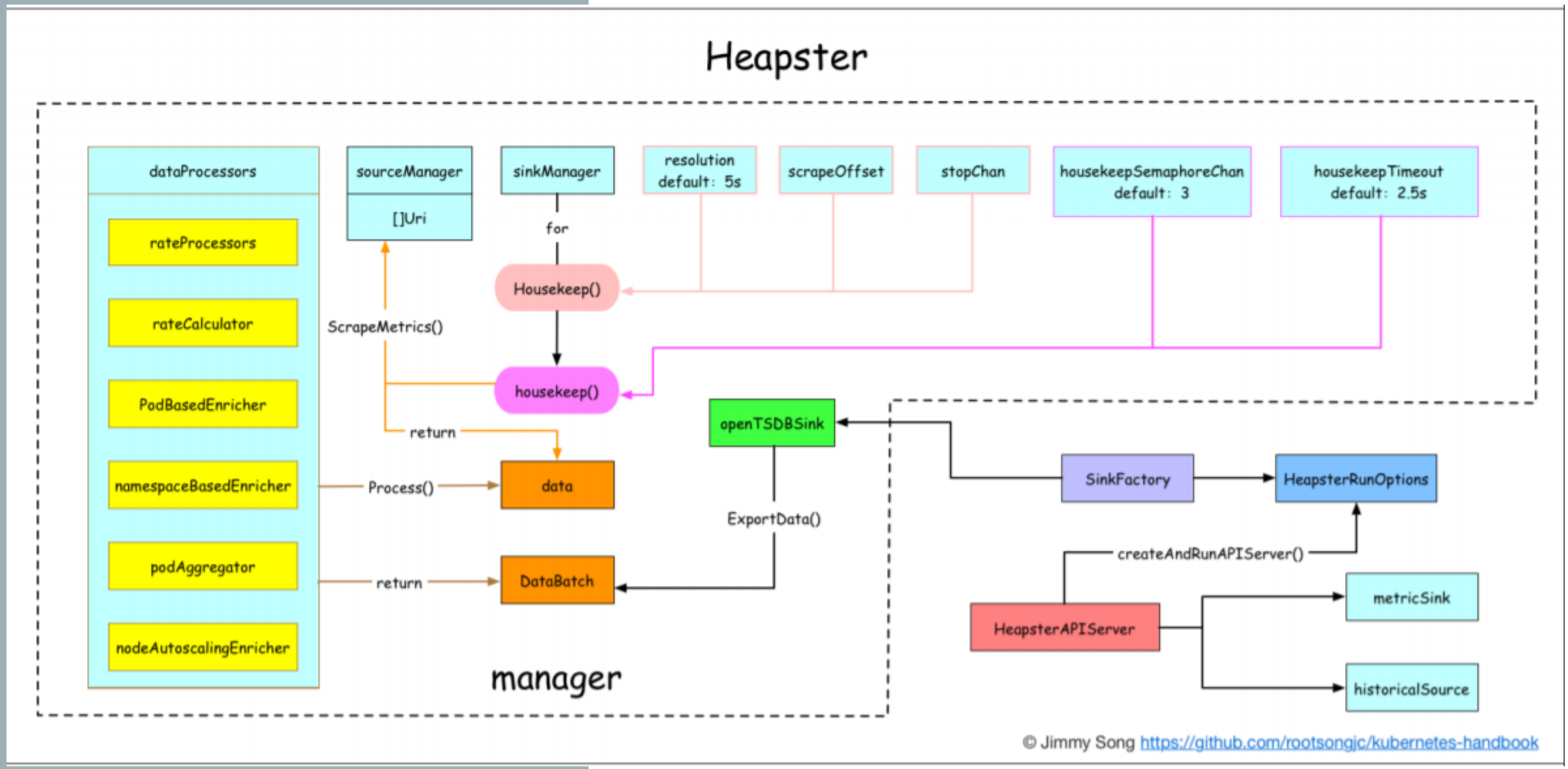
HORIZONTAL POD AUTOSCALING(HPA)

HORIZONTAL POD AUTOSCALING(HPA)

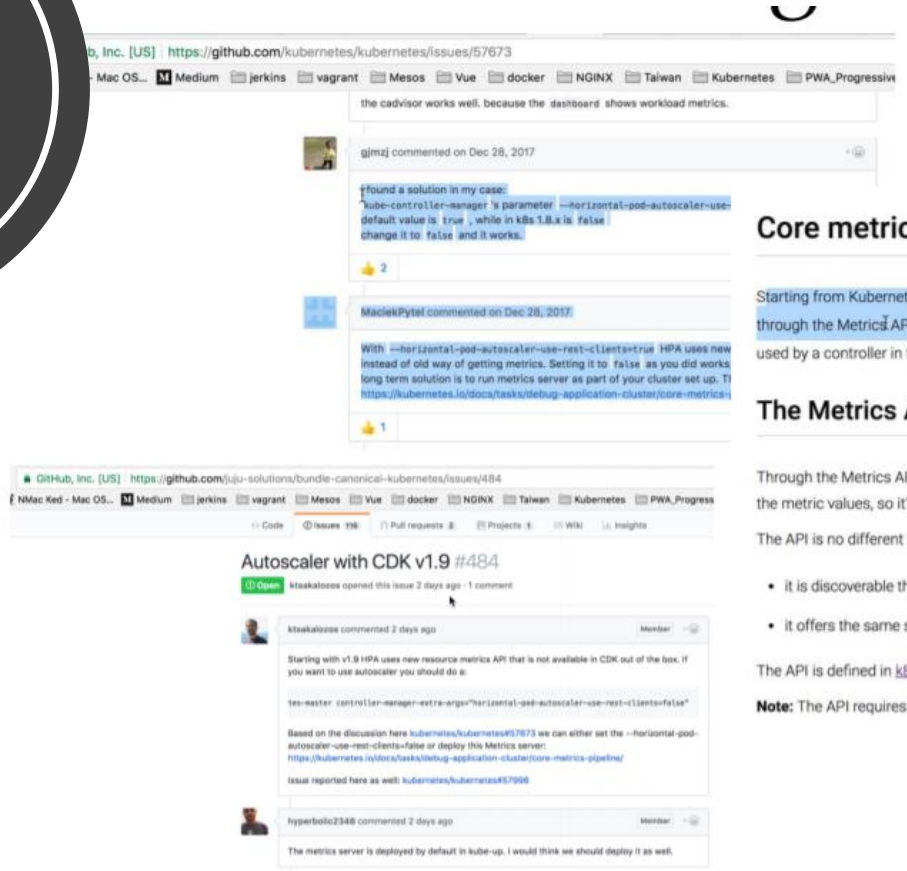




HORIZONTAL POD AUTOSCALING(HPA)


Heapster
dataProcessors
rateProcessors
rateCalculator
PodBasedEnricher
namespaceBasedEnricher
podAggregator
nodeAutoscalingEnricher
sourceManager
[]Uri
sinkManager
for
resolution
default: 5s
scrapeOffset
stopChan
housekeepSemaphoreChan
default: 3
housekeepTimeout
default: 2.5s
Housekeep()
housekeep()
ScrapeMetrics()
return
Process()
data
return
DataBatch
openTSDBSink
ExportData()
SinkFactory
HeapsterRunOptions
createAndRunAPIServer()
HeapsterAPIServer
metricSink
historicalSource
manager
© Jimmy Song https://github.com/rootsongjc/kubernetes-handbook

HORIZONTAL POD AUTOSCALING(HPA)



Core metrics pipeline

Starting from Kubernetes 1.8, resource usage metrics, such as container CPU and memory usage, are available in Kubernetes through the Metrics API. These metrics can be either accessed directly by user, for example by using `kubectl top` command, or used by a controller in the cluster, e.g. Horizontal Pod Autoscaler, to make decisions.

The Metrics API

Through the Metrics API you can get the amount of resource currently used by a given node or a given pod. This API doesn't store the metric values, so it's not possible for example to get the amount of resources used by a given node 10 minutes ago.

The API is no different from any other API:

- it is discoverable through the same endpoint as the other Kubernetes APIs under `/apis/metrics.k8s.io/` path
- it offers the same security, scalability and reliability guarantees

The API is defined in k8s.io/metrics repository. You can find more information about the API there.

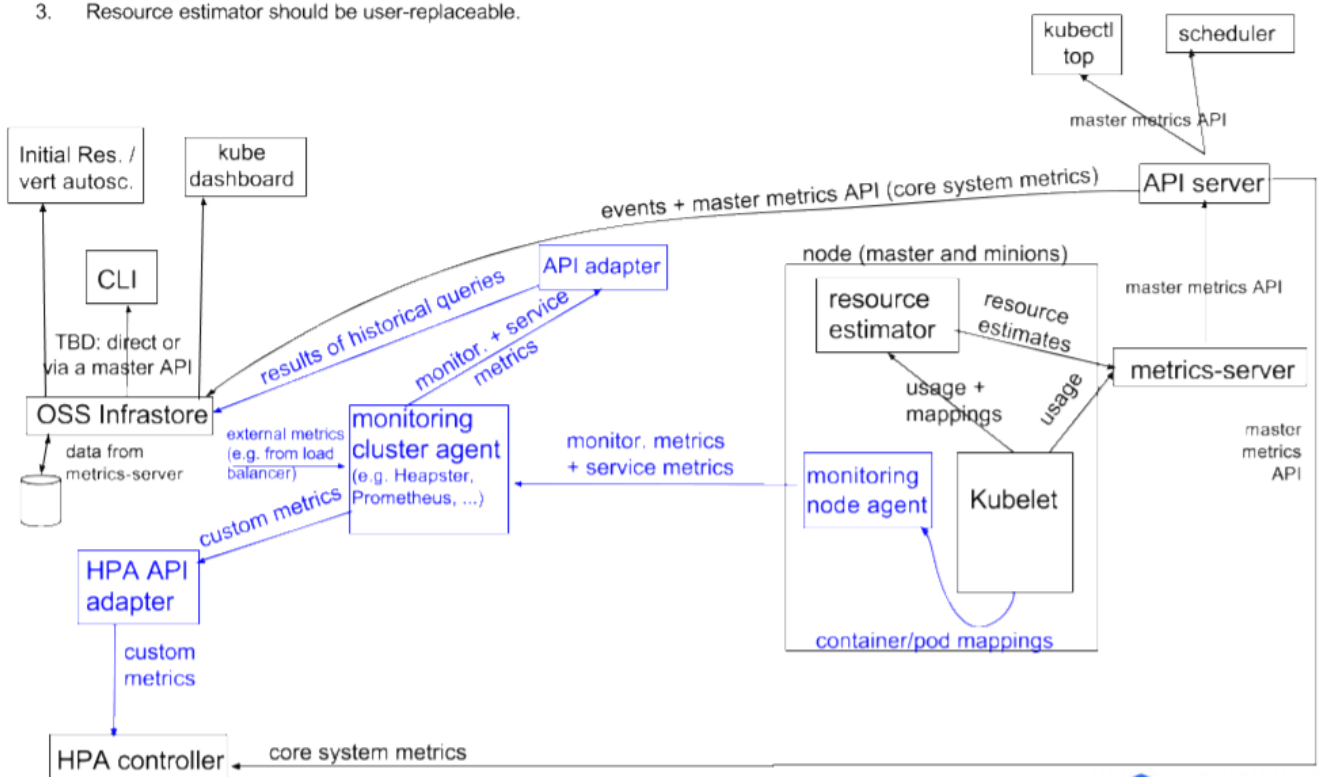
Note: The API requires metrics server to be deployed in the cluster. Otherwise it will be not available.

Monitoring architecture proposal: OSS

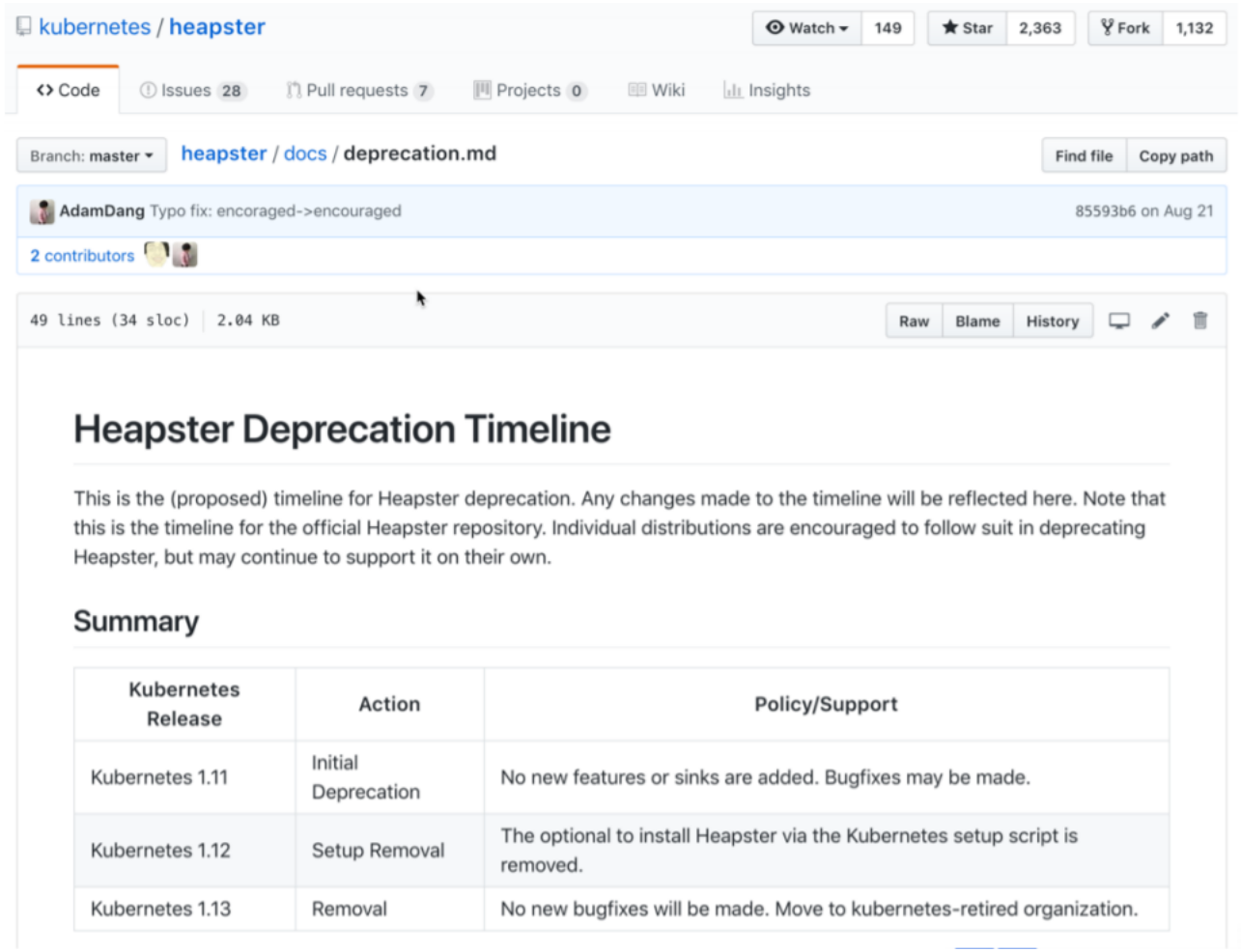
(arrows show direction of metrics flow)

Notes

1. Arrows show direction of metrics flow.
2. Monitoring pipeline is in blue. It is user-supplied and optional.
3. Resource estimator should be user-replaceable.



HORIZONTAL POD AUTOSCALING(HPA)



kubernetes / heapster

Watch 149 | Star 2,363 | Fork 1,132

Code | Issues 28 | Pull requests 7 | Projects 0 | Wiki | Insights

Branch: master ▾ heapster / docs / deprecation.md — Find file | Copy path

AdamDang Typo fix: encoraged->encouraged — 85593b6 on Aug 21

2 contributors

49 lines (34 sloc) | 2.04 KB — Raw | Blame | History

Heapster Deprecation Timeline

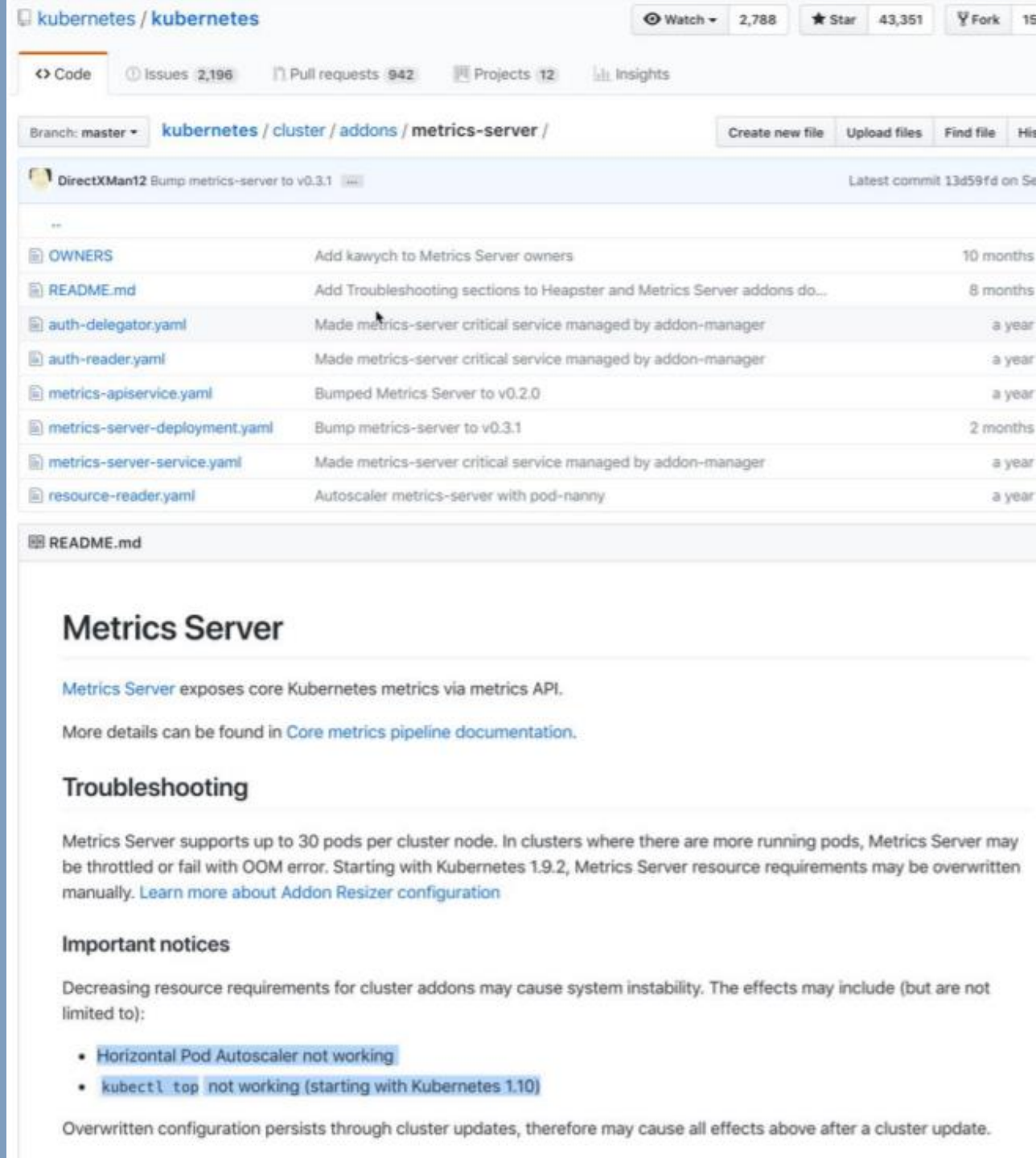
This is the (proposed) timeline for Heapster deprecation. Any changes made to the timeline will be reflected here. Note that this is the timeline for the official Heapster repository. Individual distributions are encouraged to follow suit in deprecating Heapster, but may continue to support it on their own.

Summary

Kubernetes Release	Action	Policy/Support
Kubernetes 1.11	Initial Deprecation	No new features or sinks are added. Bugfixes may be made.
Kubernetes 1.12	Setup Removal	The optional to install Heapster via the Kubernetes setup script is removed.
Kubernetes 1.13	Removal	No new bugfixes will be made. Move to kubernetes-retired organization.

“HORIZONTAL POD AUTOSCALING(HPA)

WORKSHOP



kubernetes / kubernetes

Watch 2,788 | Star 43,351 | Fork 15

Code | Issues 2,196 | Pull requests 942 | Projects 12 | Insights

Branch: master | kubernetes / cluster / addons / metrics-server / | Create new file | Upload files | Find file | His

DirectXMan12 Bump metrics-server to v0.3.1 ... Latest commit 13d59fd on Se

OWNERS	Add kawych to Metrics Server owners	10 months
README.md	Add Troubleshooting sections to Heapster and Metrics Server addons do...	8 months
auth-delegator.yaml	Made metrics-server critical service managed by addon-manager	a year
auth-reader.yaml	Made metrics-server critical service managed by addon-manager	a year
metrics-apiservice.yaml	Bumped Metrics Server to v0.2.0	a year
metrics-server-deployment.yaml	Bump metrics-server to v0.3.1	2 months
metrics-server-service.yaml	Made metrics-server critical service managed by addon-manager	a year
resource-reader.yaml	Autoscaler metrics-server with pod-nanny	a year

README.md

Metrics Server

Metrics Server exposes core Kubernetes metrics via metrics API.

More details can be found in Core metrics pipeline documentation.

Troubleshooting

Metrics Server supports up to 30 pods per cluster node. In clusters where there are more running pods, Metrics Server may be throttled or fail with OOM error. Starting with Kubernetes 1.9.2, Metrics Server resource requirements may be overwritten manually. Learn more about Addon Resizer configuration

Important notices

Decreasing resource requirements for cluster addons may cause system instability. The effects may include (but are not limited to):

- Horizontal Pod Autoscaler not working
- `kubectl top` not working (starting with Kubernetes 1.10)

Overwritten configuration persists through cluster updates, therefore may cause all effects above after a cluster update.

Q&A

REFERENCES

- https://www.blognone.com/node/106492
- http://bit.ly/2Y7DQ4U
- Slide Kubernetes_Training_v9_201904.pdf