

kubernetes

KUBERNETES(K8S)

Orcsoft Internal Training #5

By Chaiwat Klomchitcharoen(Jay)



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

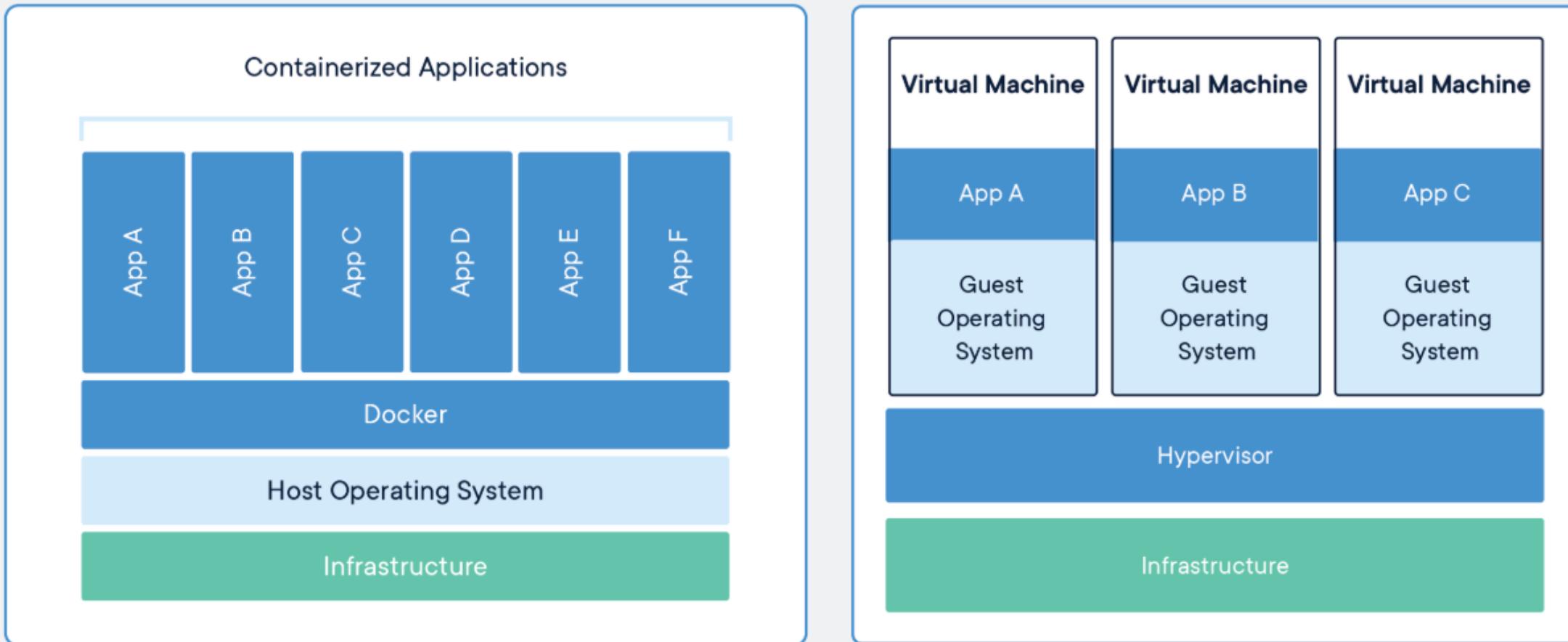
The screenshot shows a search results page for the query "labdocker". The top navigation bar includes a logo, "Explore", "Help", a search bar with the query "labdocker", and "Sign up" and "Log In" buttons. Below the search bar, the text "Repositories (7)" is displayed. A dropdown menu is open above the repository list, showing "All" selected. The repository list displays two entries:

Repository	Type	Stars	Pulls	Actions
labdocker/alpineweb	public	0	31	DETAILS
labdocker/nginx	public	0	16	DETAILS

LAB RESOURCE

CONTAINER CONCEPT

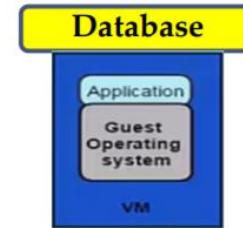
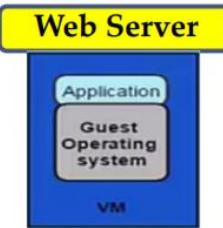
VIRTUALIZATION มีข้อจำกัด



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

ตัวอย่างการใช้งาน **CONTAINER**

- Production Environment
- Day 1: Application 1: Implement

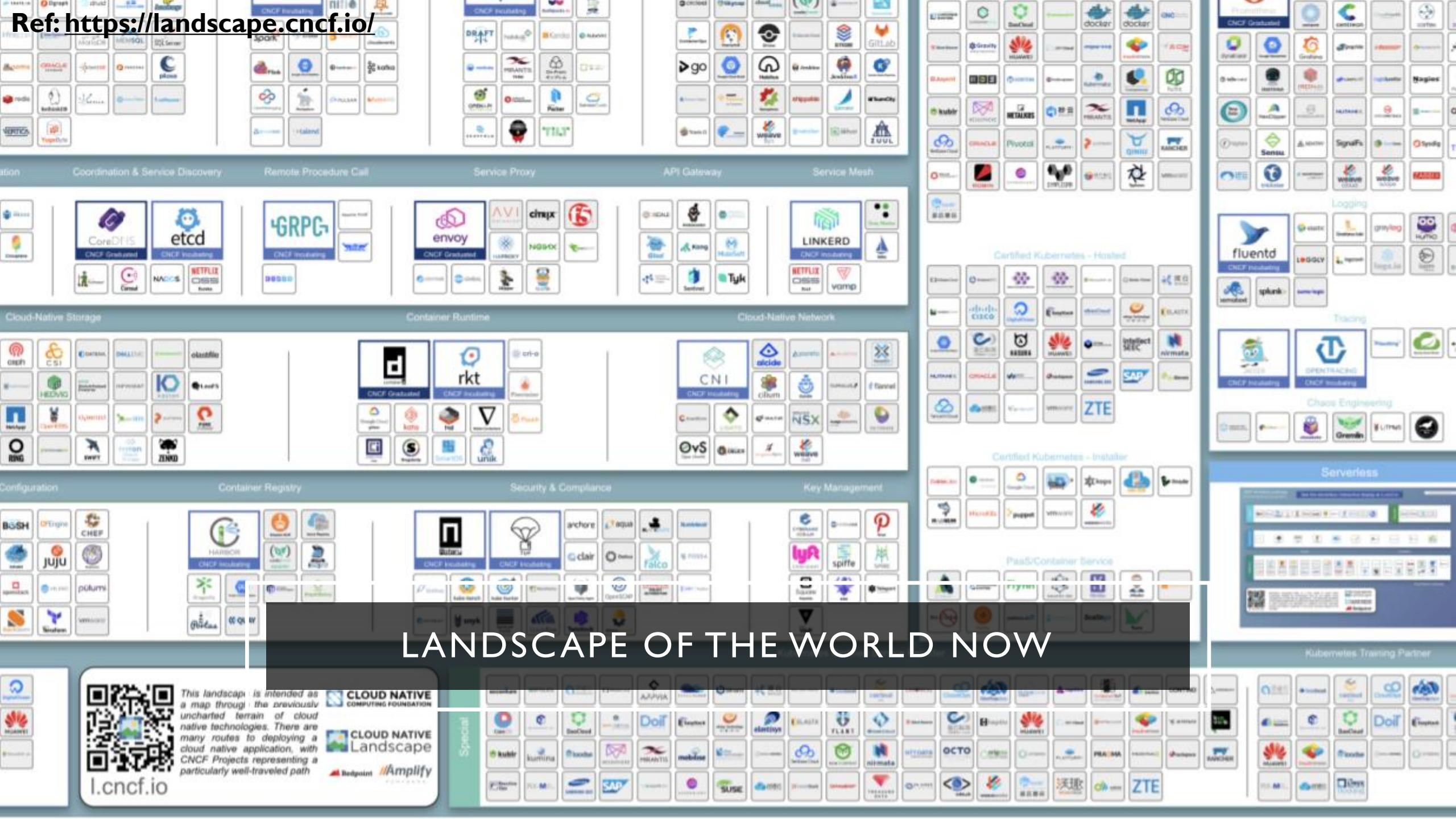


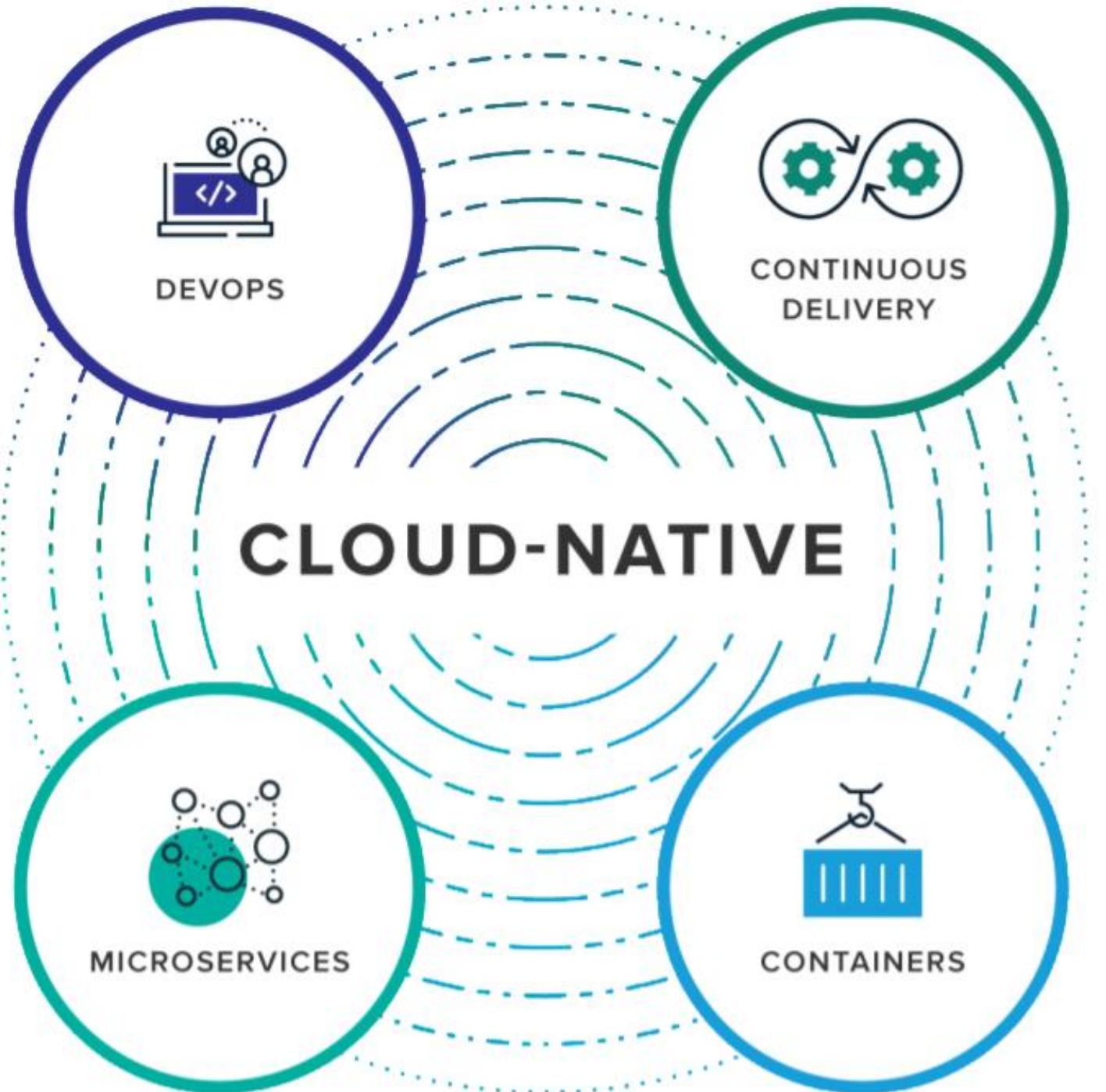
- Apache 2.20 Web Server
- PHP 5.5 Engine
- Laravel 4.1 Framework

- IIS 8
- .Net FrameWork 3.5

- 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.





LANDSCAPE OF
THE WORLD
NOW

INTRODUCTION TO KUBERNETES



KUBERNETES(K8S) ?

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

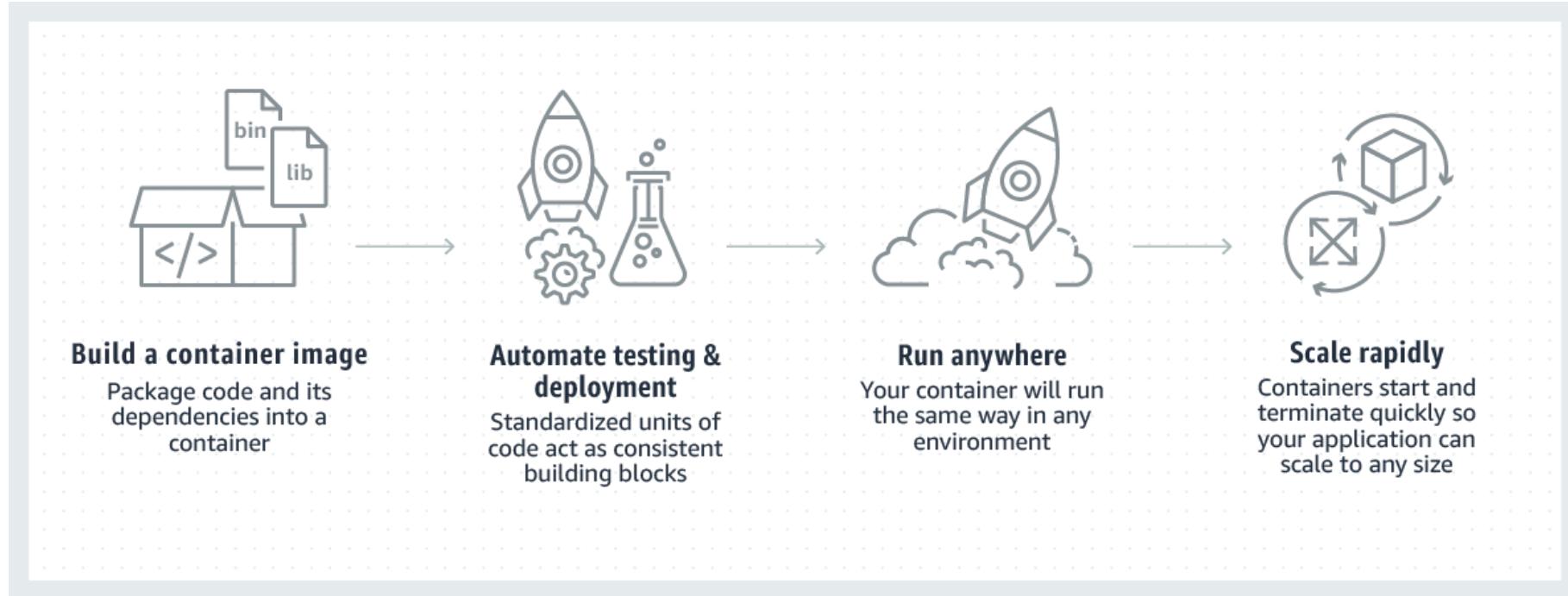
INTRODUCTION TO KUBERNETES

Check Kubernetes version

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

```
i  praparnlueangphoonlap —
— Terminal MAC Pro — bash | — Terminal MAC Pro —
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/nodes/minikube
  uid: f03de16d-58bb-11e7-aae1-080027559511
spec:
  externalID: minikube
```

```
nodeInfo:
  architecture: amd64
  bootID: cdf78a28-072d-4707-8cd-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.



GRPC



NATS

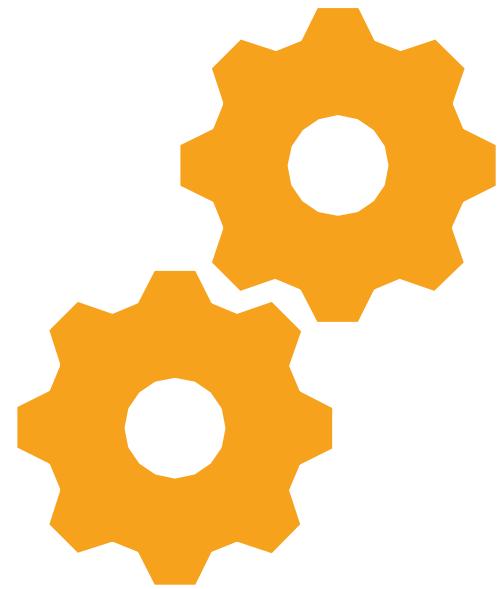


INTRODUCTION TO KUBERNETES

- 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

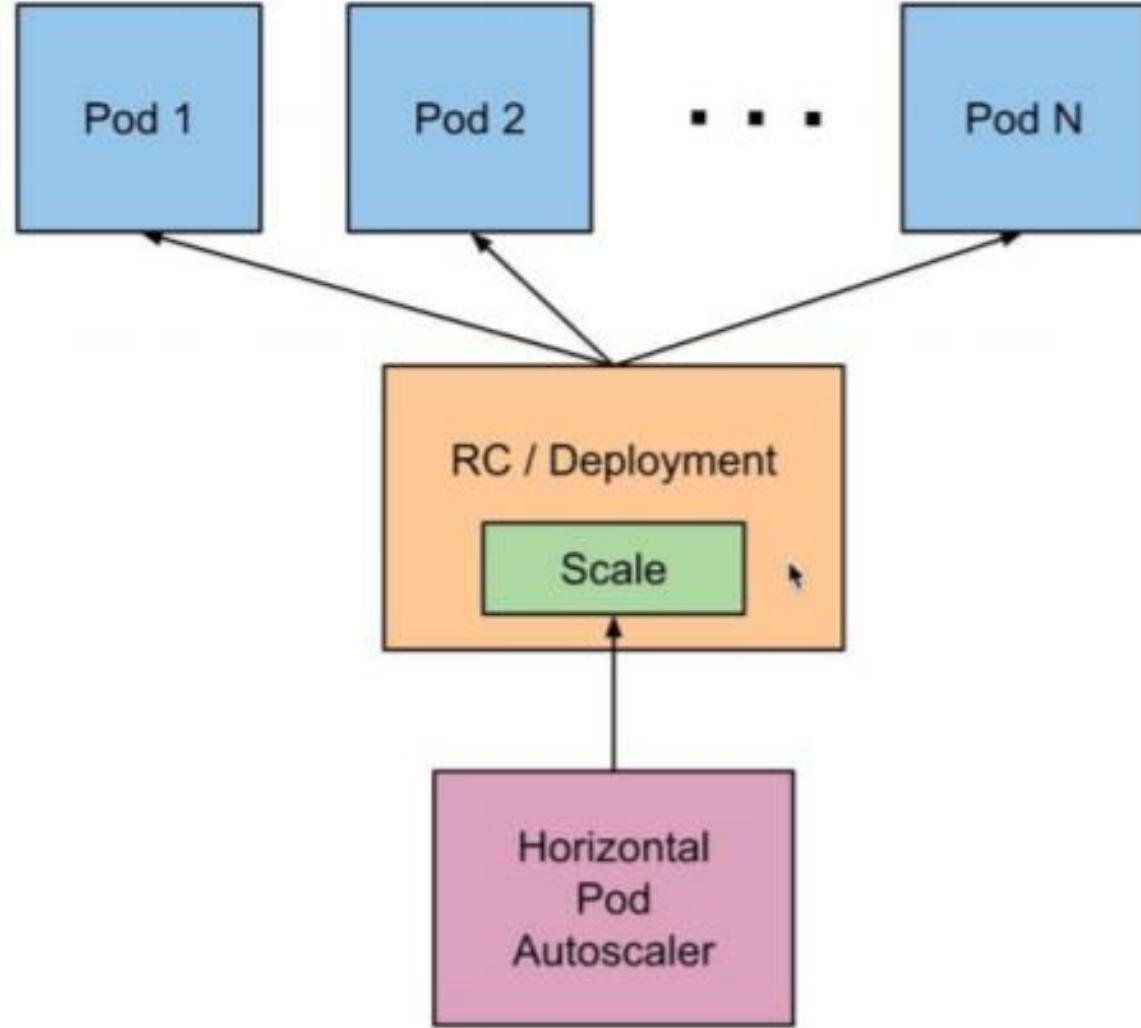
- Automatic binpacking
- Horizontal Pod Autoscaling (HPA)
- Automated rollouts and rollbacks
- Storage orchestration
- Self-healing
- Service discovery and load balancing
- Secret and configuration management
- Batch execution



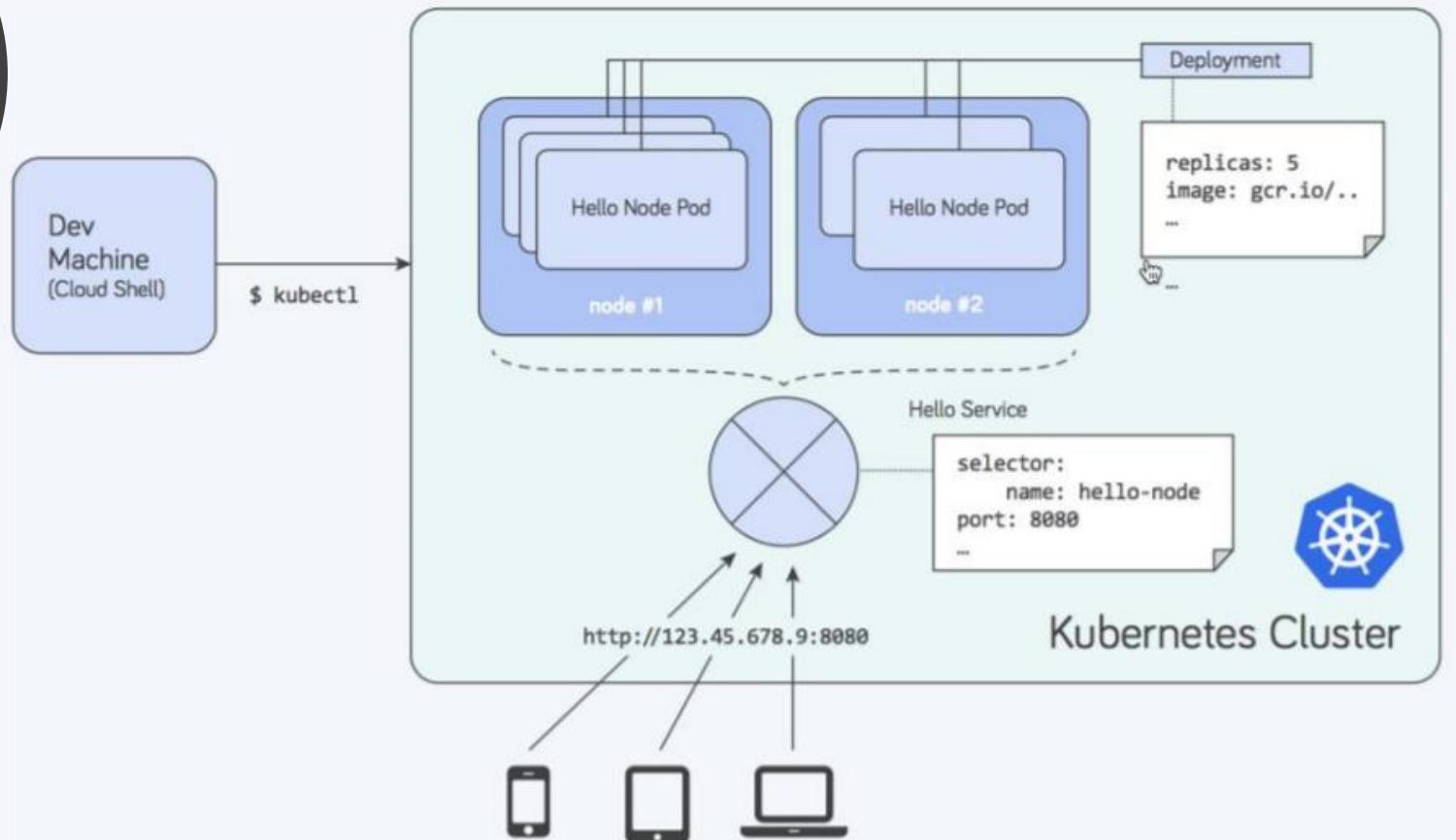
AUTOMATIC BINPACKING

Check available node resource by command: `kubectl describe node`

HORIZON PODS AUTOSCALING

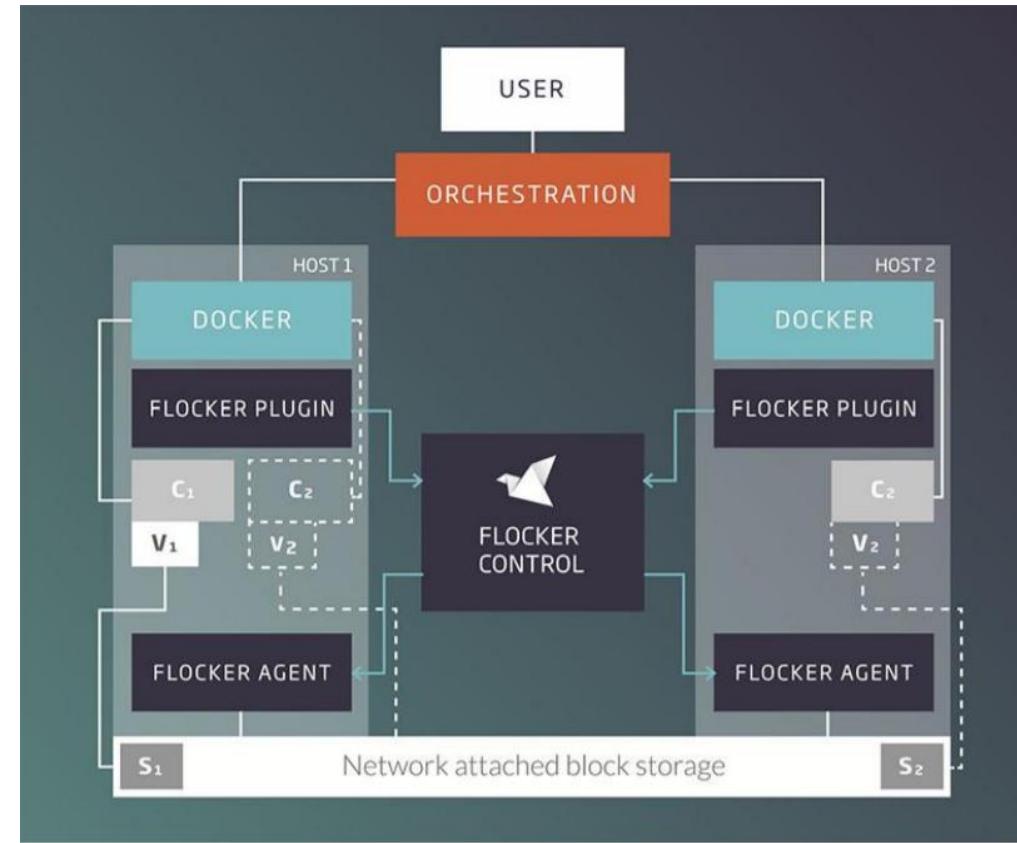


AUTOMATED ROLLOUTS AND ROLLBACKS



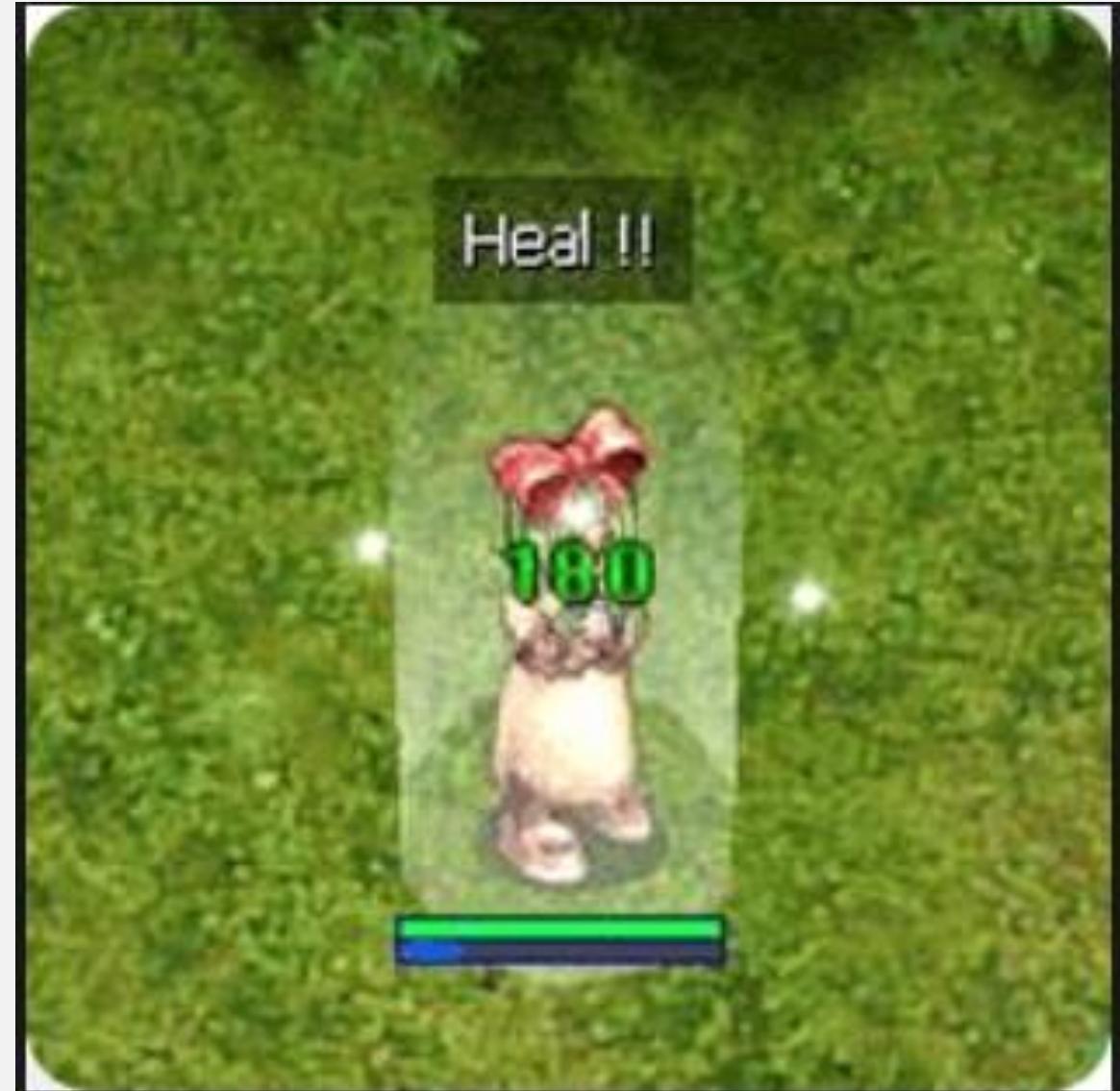
STORAGE ORCHESTRATION

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

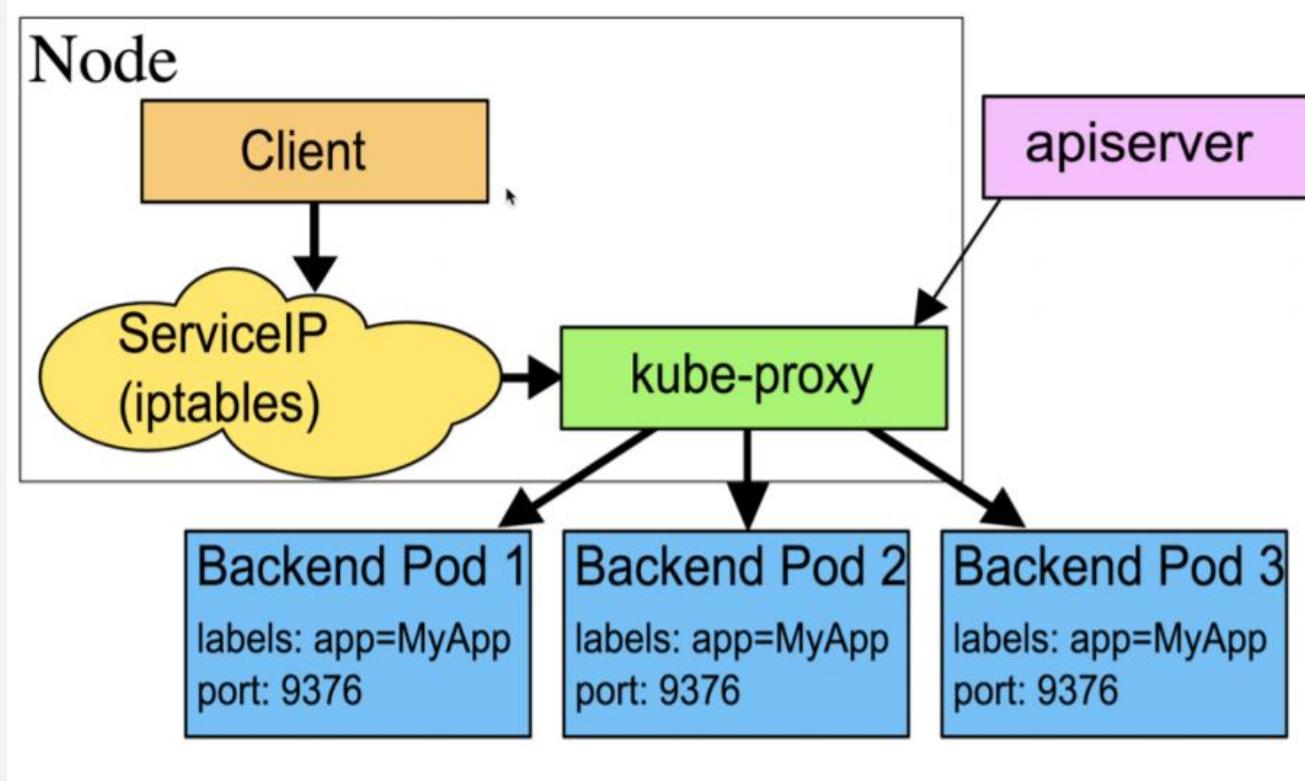


SELF-HEALING

- 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

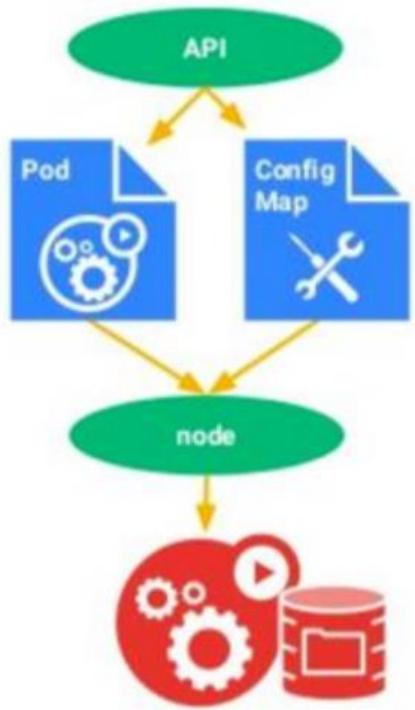


```
$ kubectl get secrets
NAME          TYPE      DATA   AGE
db-user-pass  Opaque    2      51s

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

Type:        Opaque

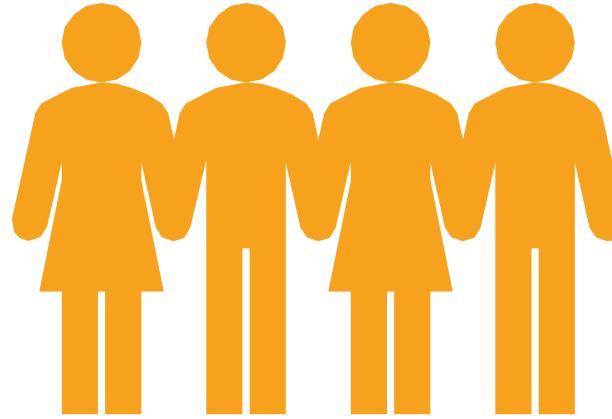
Data
=====
password.txt: 12 bytes
username.txt: 5 bytes
```



ConfigMap

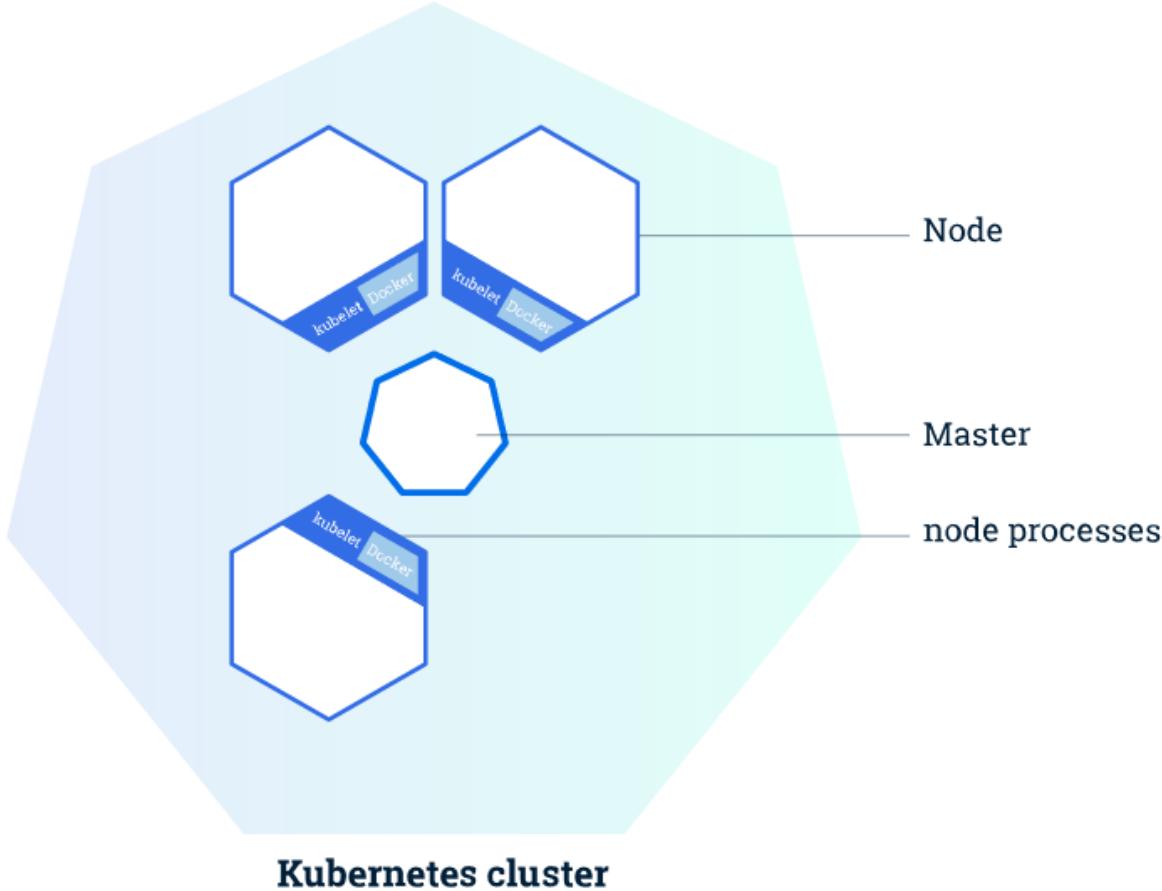
```
apiVersion: v1
kind: ConfigMap
metadata:
  name: dragon-config
labels:
  environment: non-prod
data:
  dragon.how.much: very
  dragon.type: fast
```

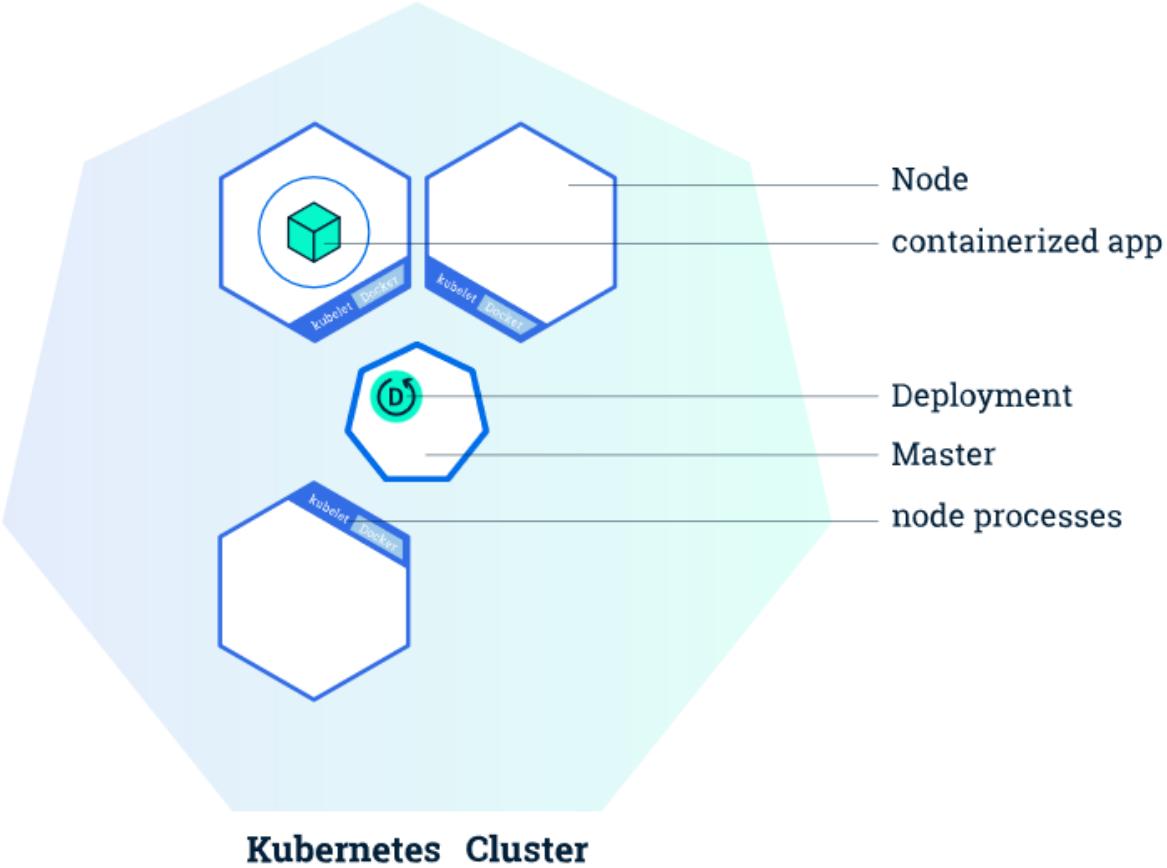
```
apiVersion: v1
kind: Pod
metadata:
  name: dragon-pod
spec:
  containers:
    - name: dragon-container
      image: dragon-image
      env:
        - name: DRAGON_LEVEL
          valueFrom:
            configMapKeyRef:
              name: dragon-config
              key: dragon.how.much
        -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

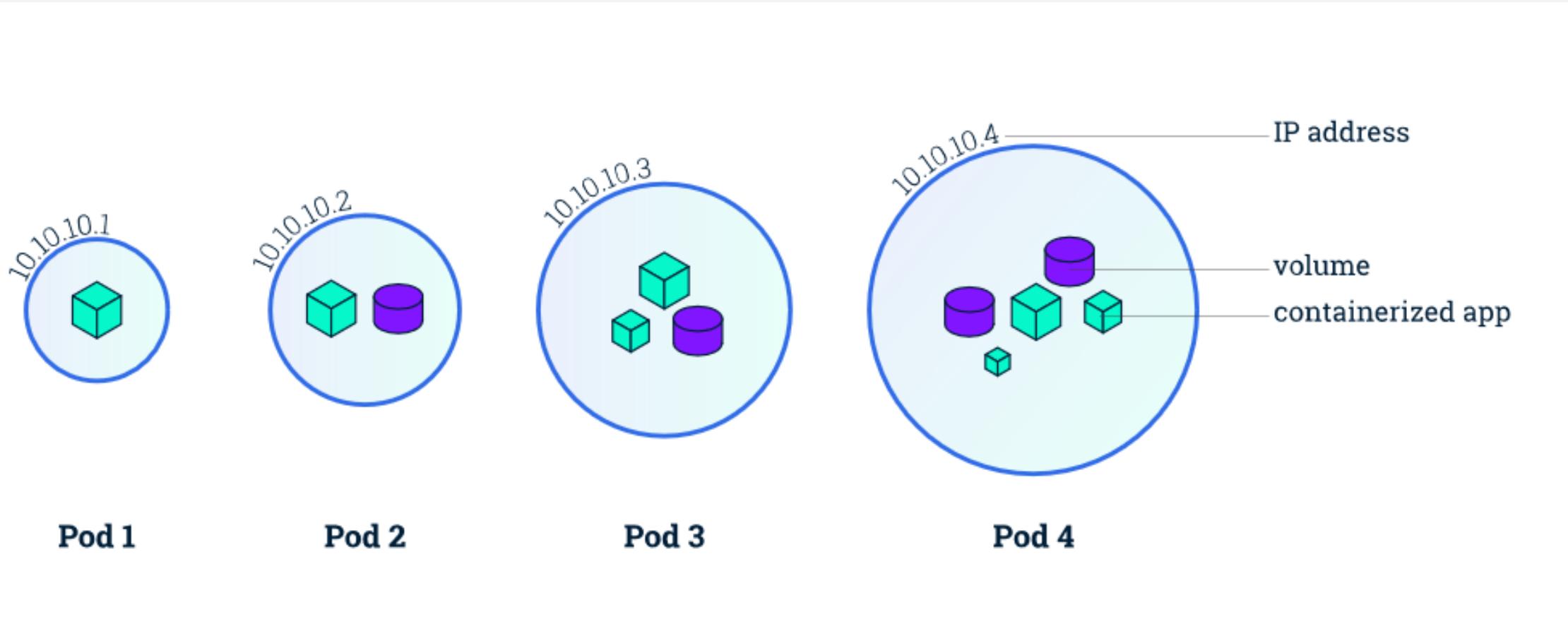


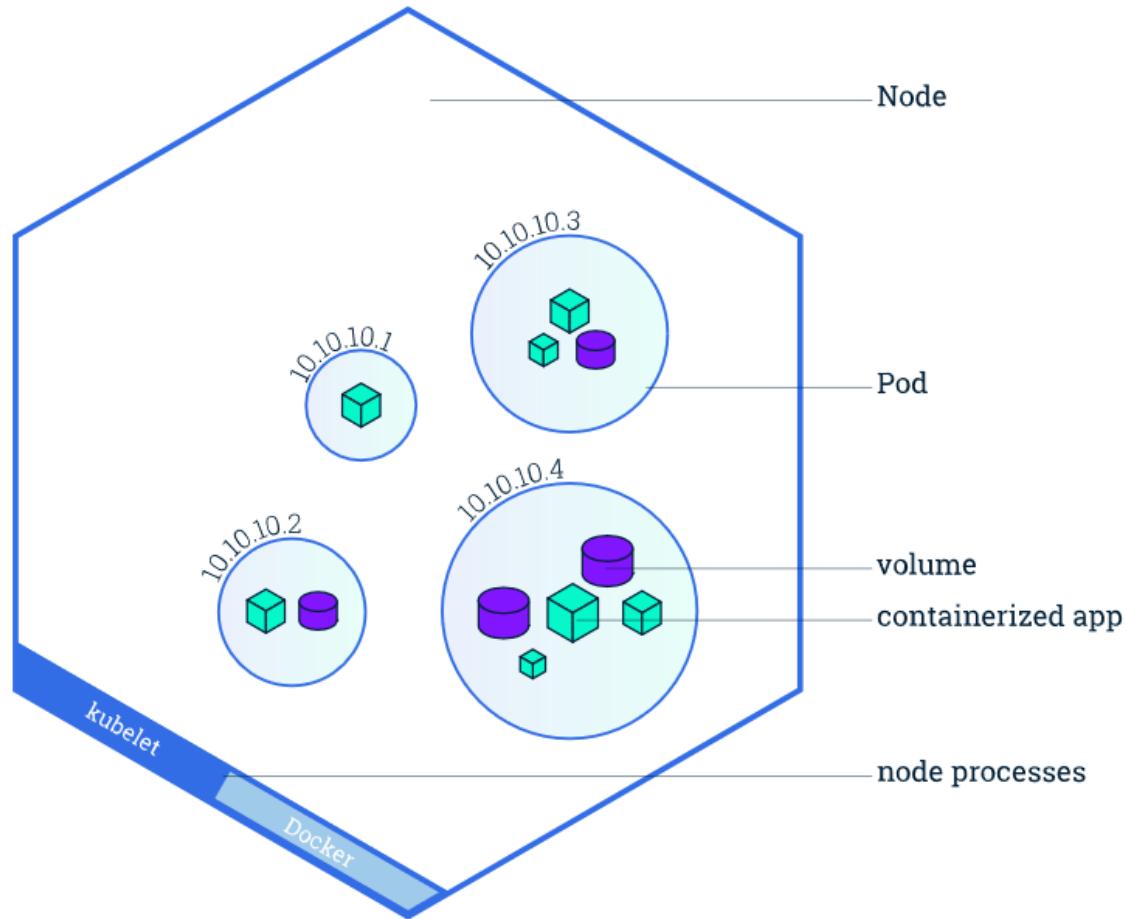
BATCH EXECUTION

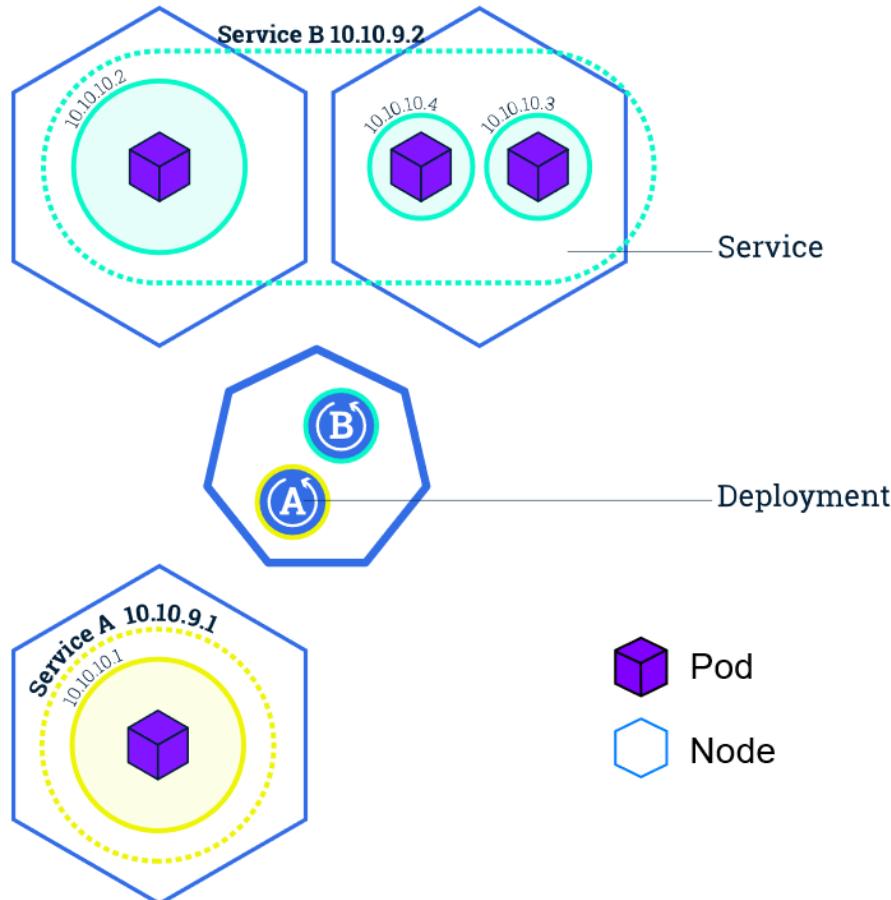
SYSTEM ARCHITECTURE

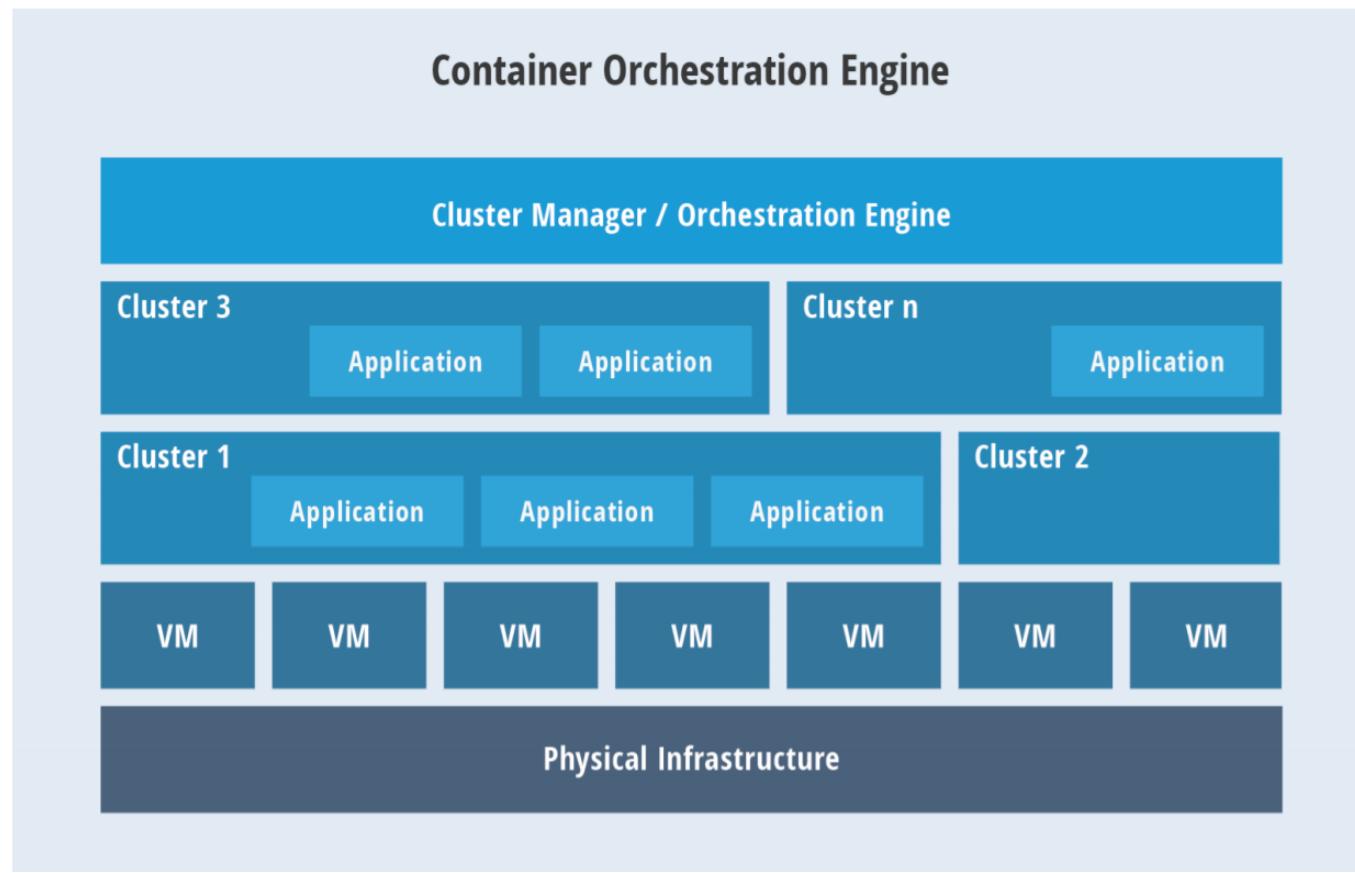


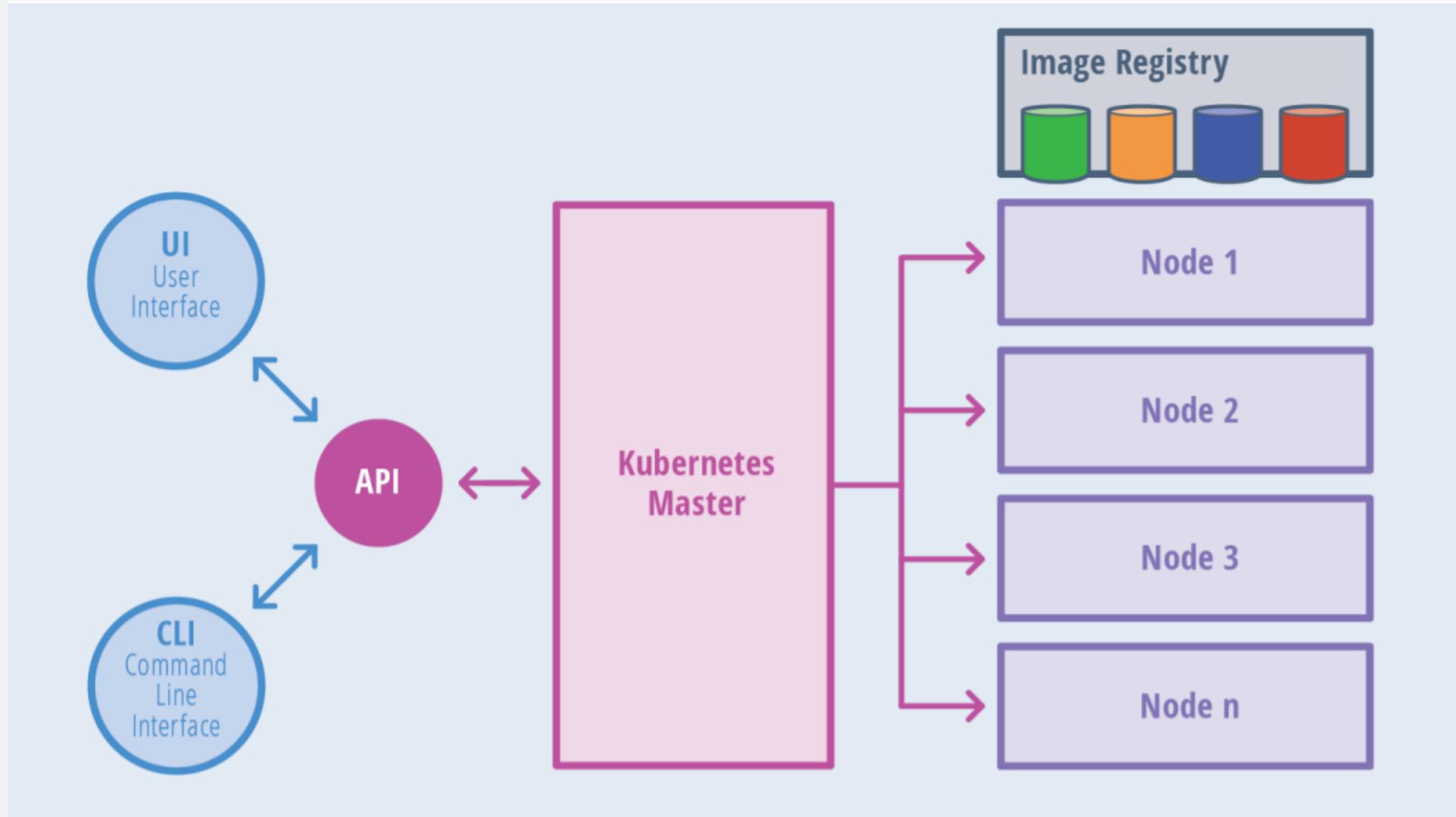


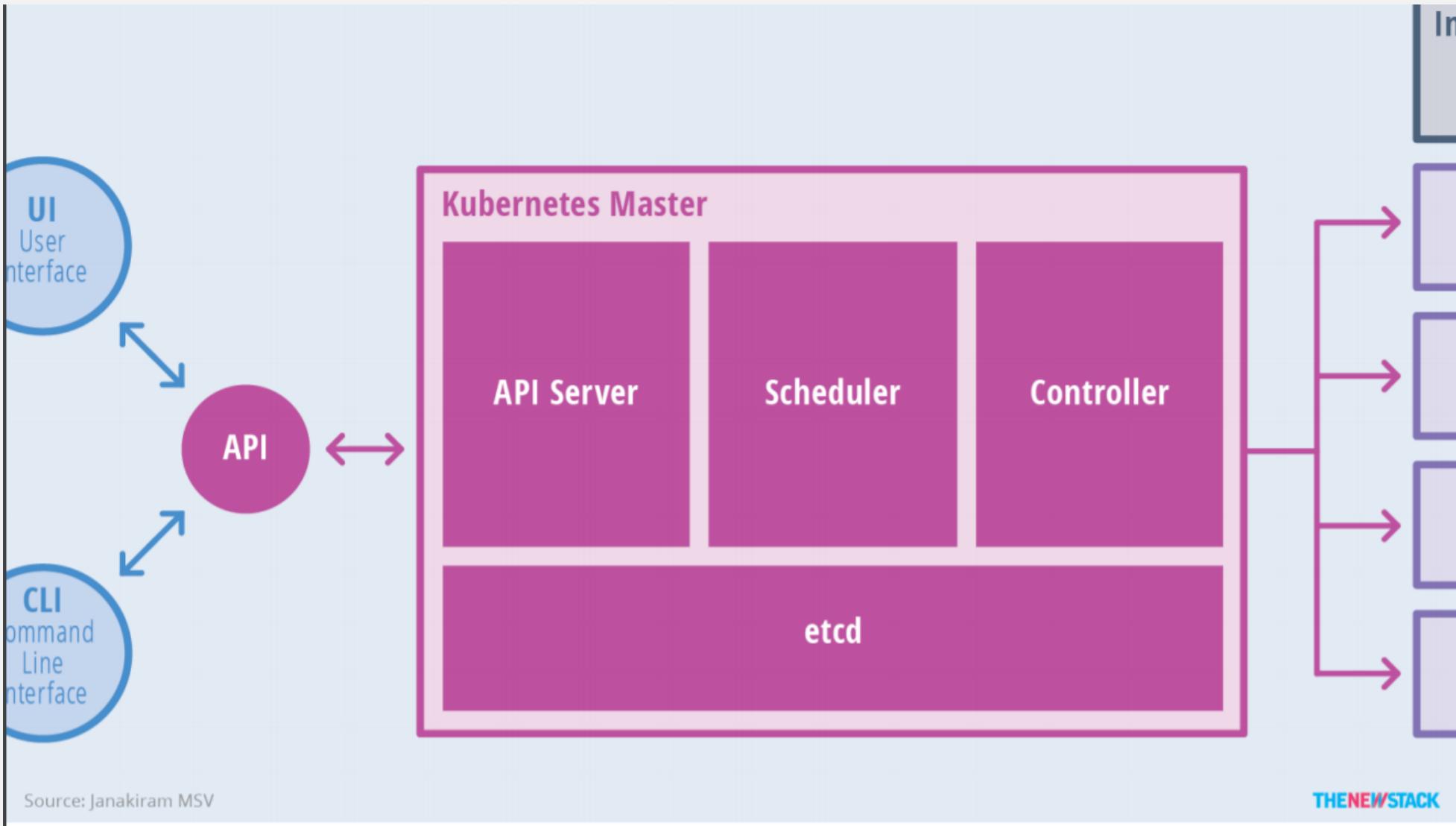


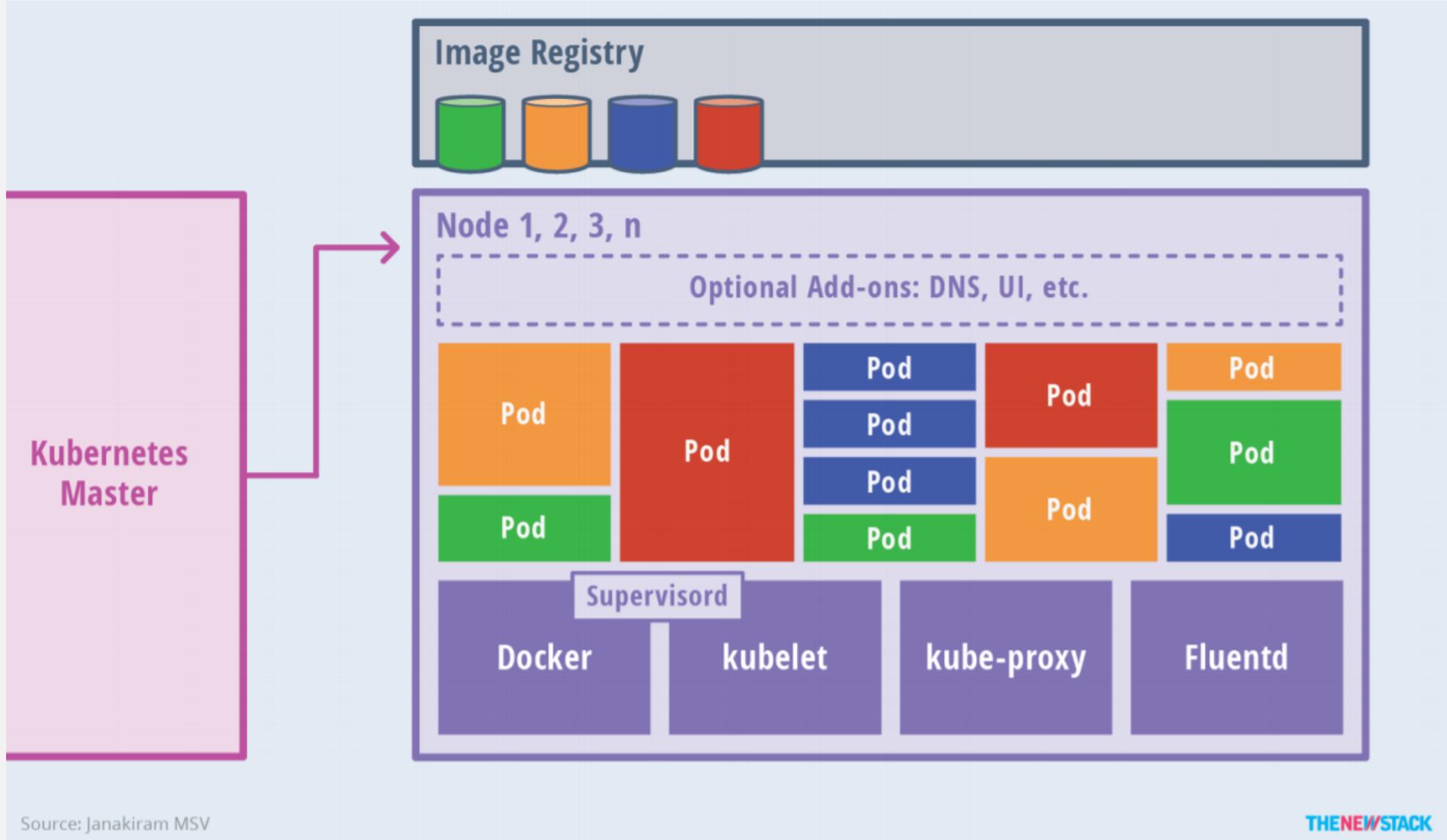




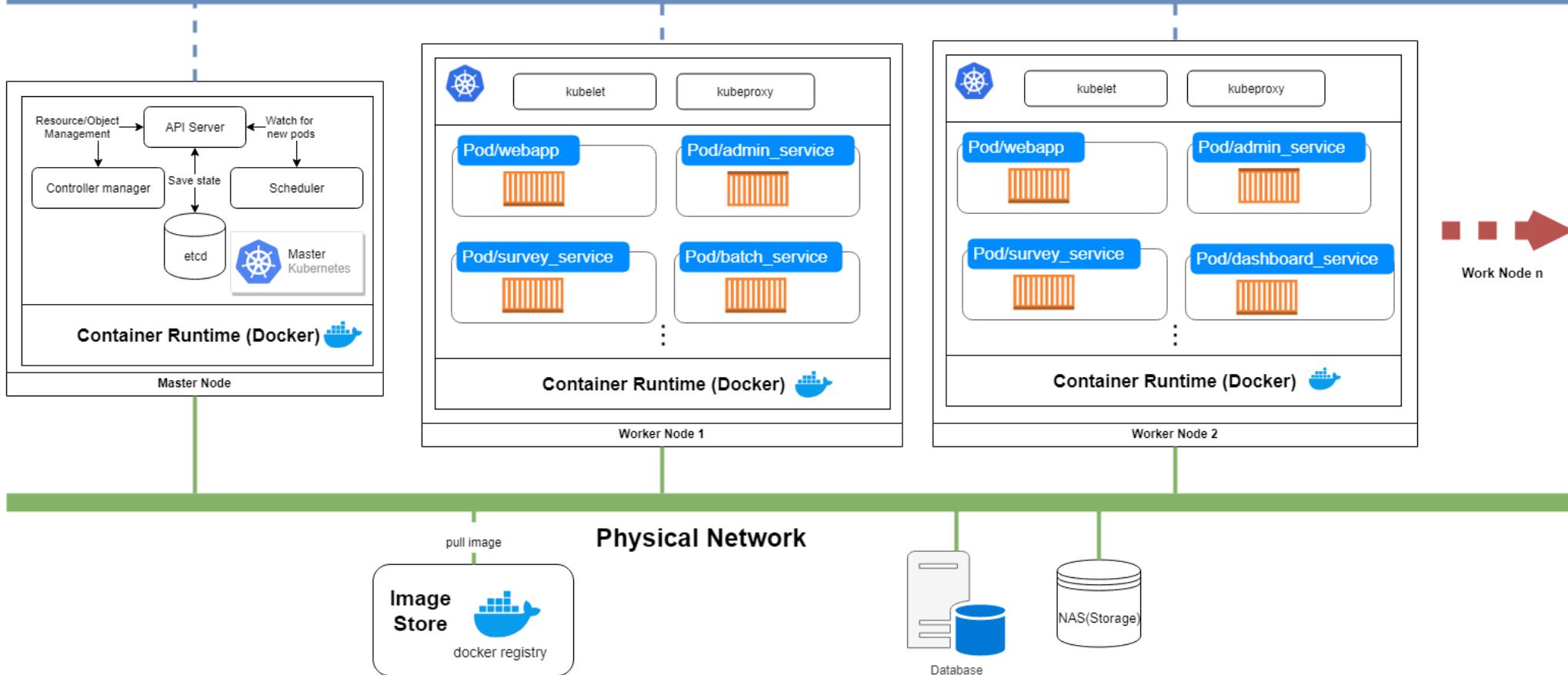




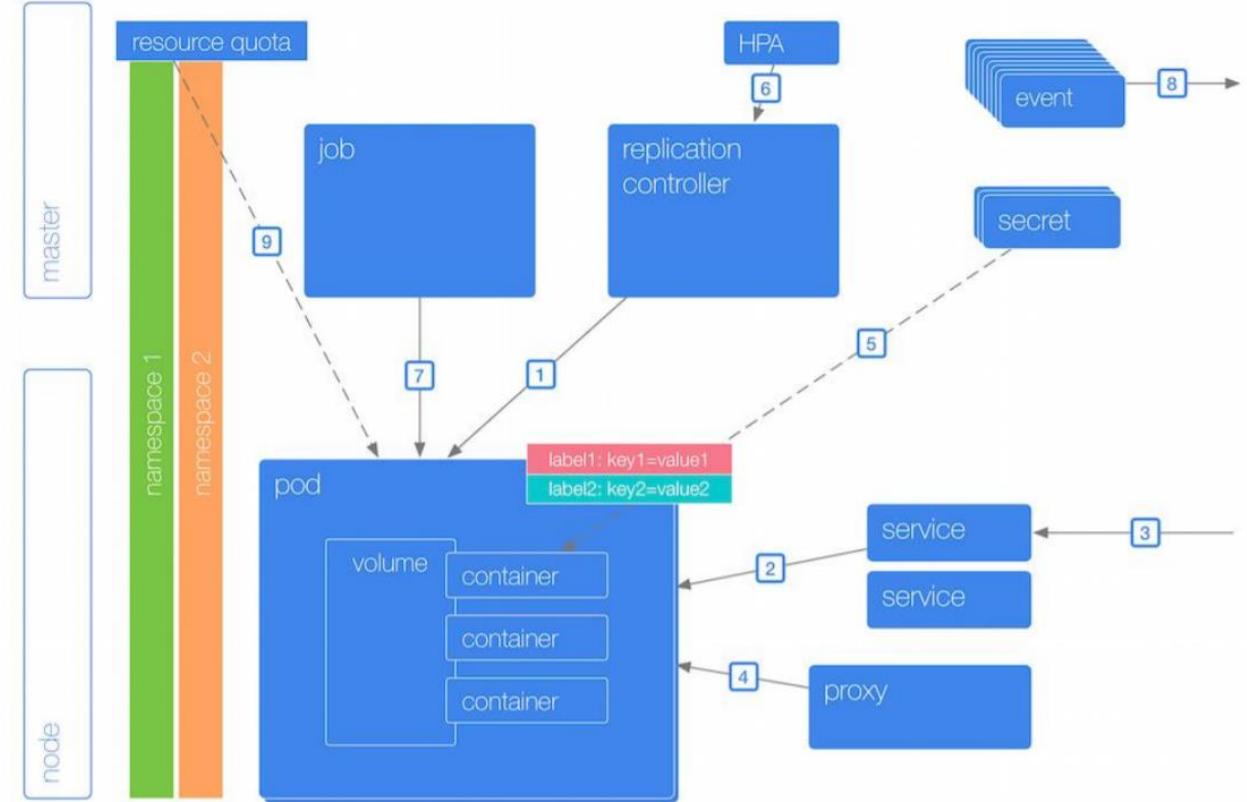




Overlay Network(Calico,Flannel,WeaveNet)



SYSTEM ARCHITECTURE





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 "YAML" 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

A photograph of a massive cargo ship sailing on a vast, blue ocean under a sky filled with wispy clouds. The ship's deck is completely covered with a dense stack of shipping containers in various colors, including blue, green, red, and white. A large white rectangular box with a thin black border is overlaid on the lower half of the image, containing the text.

PODS, CONTAINER AND SERVICES

PODS, CONTAINER AND SERVICES

- Pods vs Container

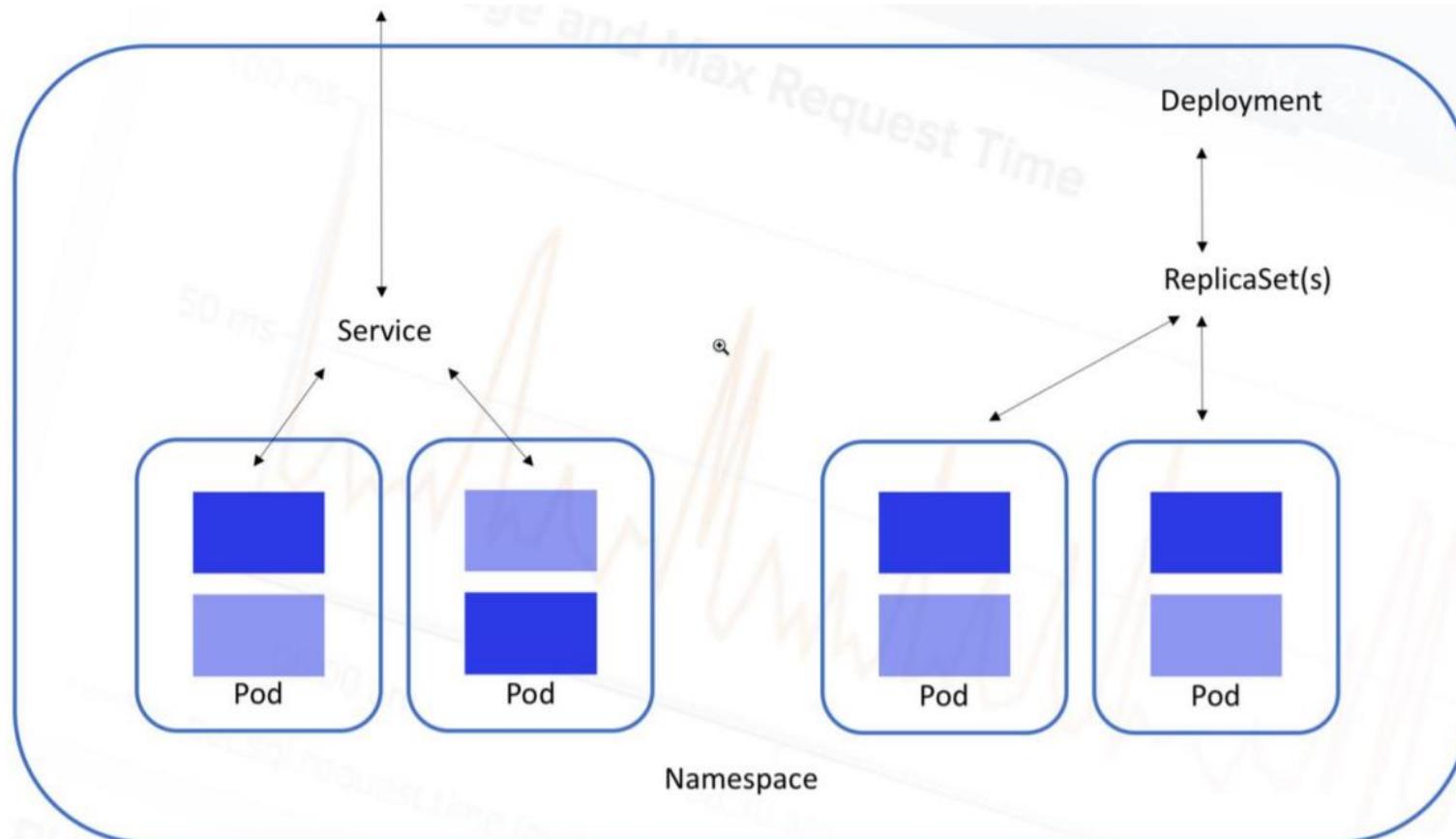
- Docker's view point:
 - 1 Container: 1 Application, 1 Component of Microservice
 - So for micro service we need multi container
 - Cache component
 - Web component
 - Database component
 - Etc
- KuberneTEST's view point:
 - 1 Pods = 1 Container
 - 1 Pods = N Container (Container on the same context,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



- **Pods 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 kubernetes 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>

```

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)

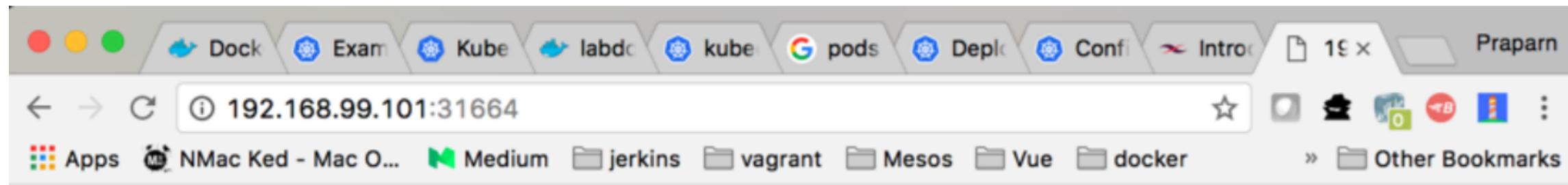
```
kubectl expose deployment <name> <option>
```

- 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)

```
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)

```
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
```

YAML Converter

Save & Share

YAML Input sample X

```
1 apiVersion: "v1"
2 kind: Pod
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    containers:
13      - name: webtest
14        image: labdocker/cluster:webservicelite
15        ports:
16          - containerPort: 5000
17            protocol: TCP
18        nodeSelector:
19          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

```
1 {
2   "apiVersion": "v1",
3   "kind": "Pod",
4   "metadata": {
5     "name": "webtest",
6     "labels": {
7       "name": "web",
8       "owner": "Praparn_L",
9       "version": "1.0",
10      "module": "WebServer",
11      "environment": "development"
12    }
13  },
14  "spec": {
15    "containers": [
16      {
17        "name": "webtest",
18        "image": "labdocker/cluster:webservicelite",
19        "ports": [
20          {
21            "containerPort": 5000,
22            "protocol": "TCP"
23          }
24        ]
25      }
26    ],
27    "nodeSelector": {
28      "kubernetes.io/hostname": "kubernetes-1"
29    }
30  }
31 }
```

PODS, CONTAINER AND SERVICES

Imperative object configuration

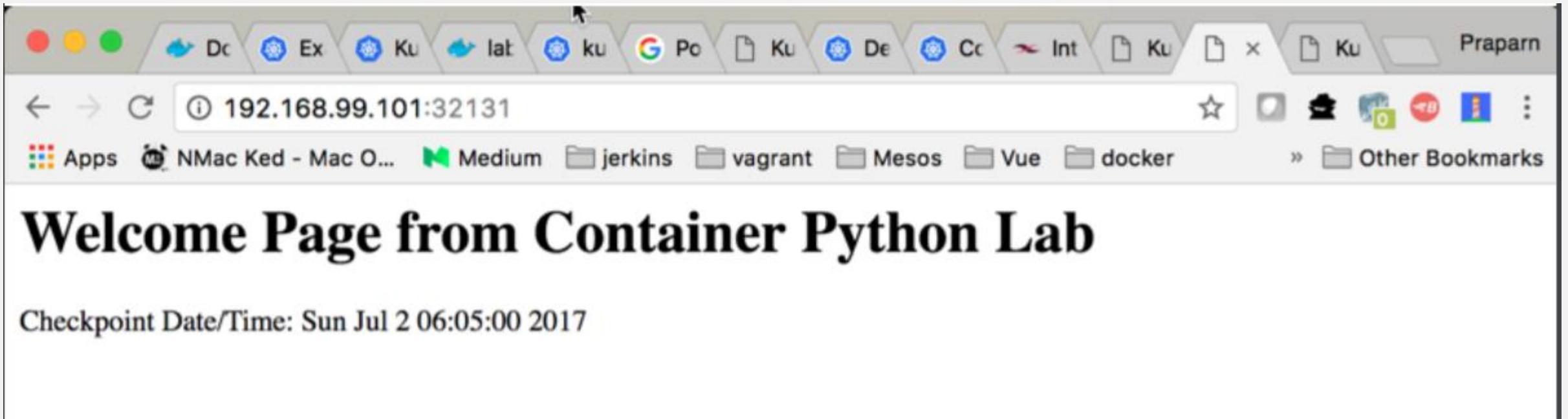
```
kubectl create -f <Filename>
```

```
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://04b9cdfdd6451a4e78c873f704fb4d35de7bba082343d0773c7ea0ebbb3f03a
    Image:          labdocker/cluster:webservicelite
    Image ID:       docker://sha256:837c8f41c918ede06f85f9b554b6120fdb654cc2a8eb7eec74bc383b09865f2b
    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  spec.containers{webtest}  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
5de7bba082343d0773c7ea0ebbb3f03a
14m        14m       1  kubelet, minikube  spec.containers{webtest}  Normal  Started   Started container with id 04b9cdfdd6451a4e78c873f704fb4d3
5de7bba082343d0773c7ea0ebbb3f03a
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:

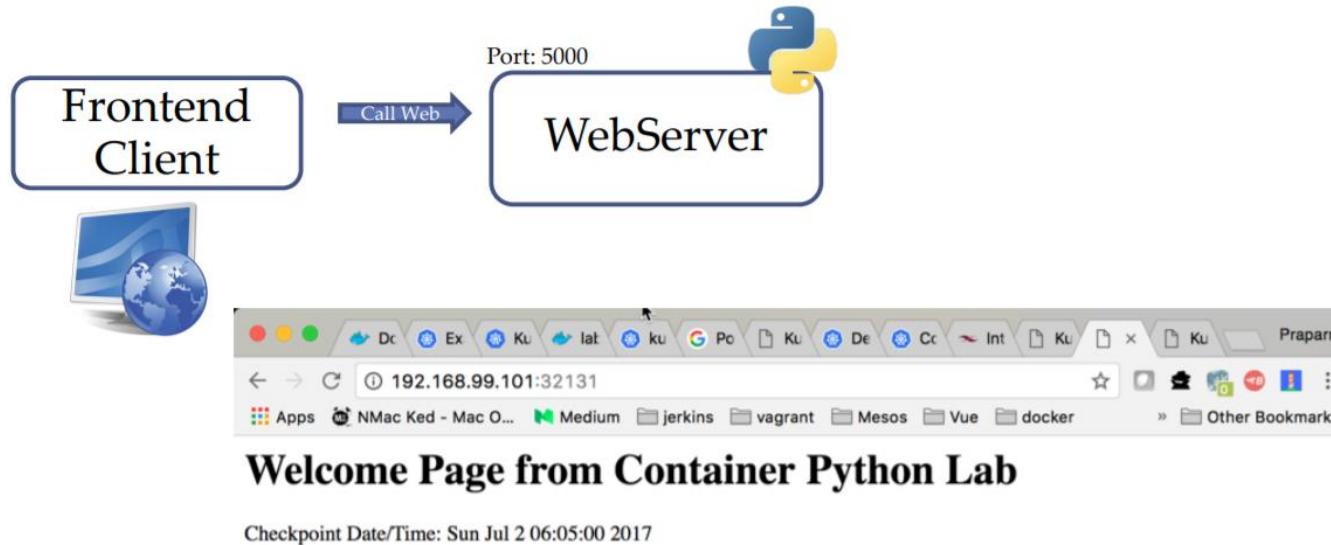
- Coverage all change in single command (Entire folder)
 - Fully support live-object configuration
 - Keep track/Merge all change in configuration file
 - Update default value in configuration file

- 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.

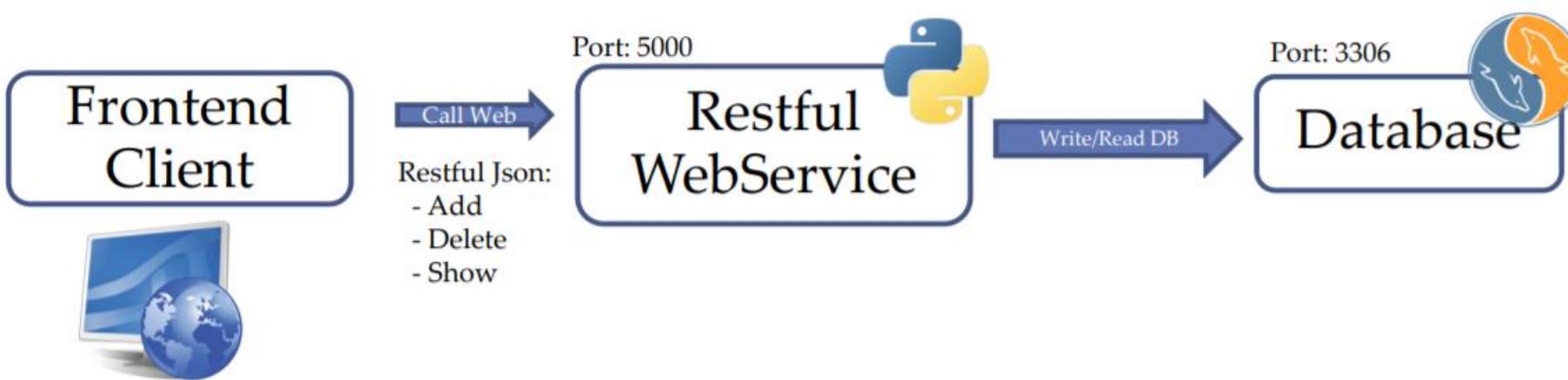
- Part 1: Deploy simple web pods with single container



WORKSHOP I.I: PODS & SERVICE

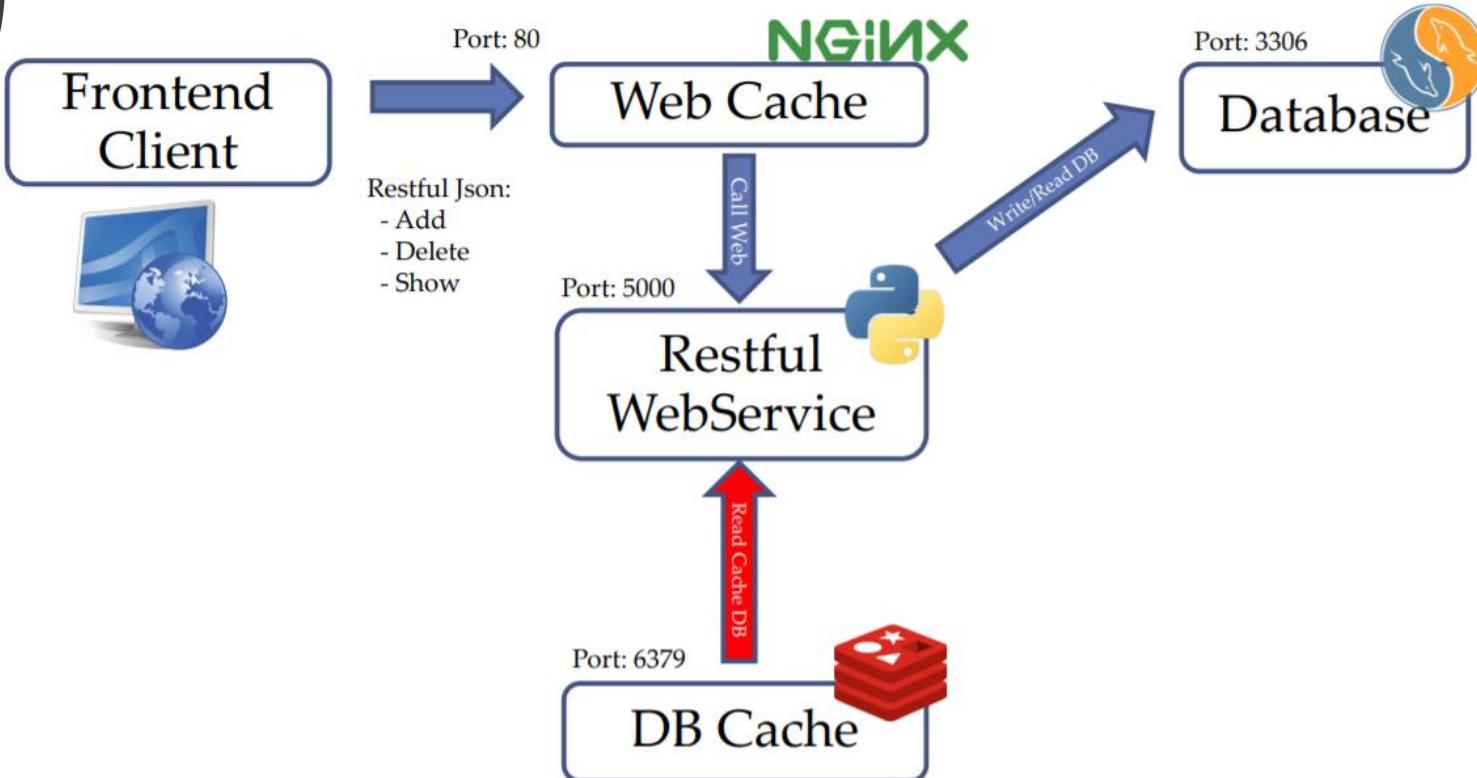
WORKSHOP I.I: PODS & SERVICE

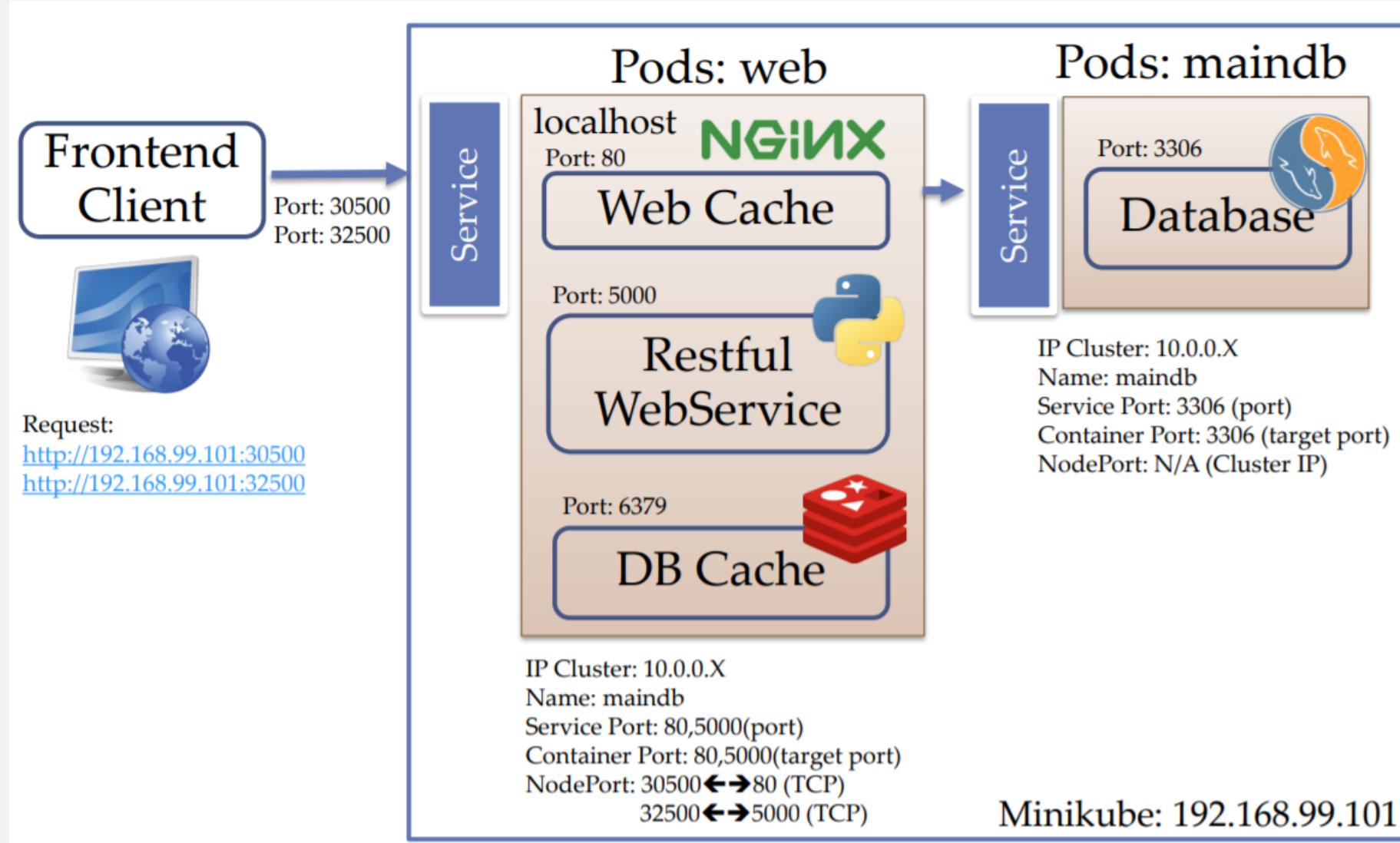
- Example Case: Basic Restful API

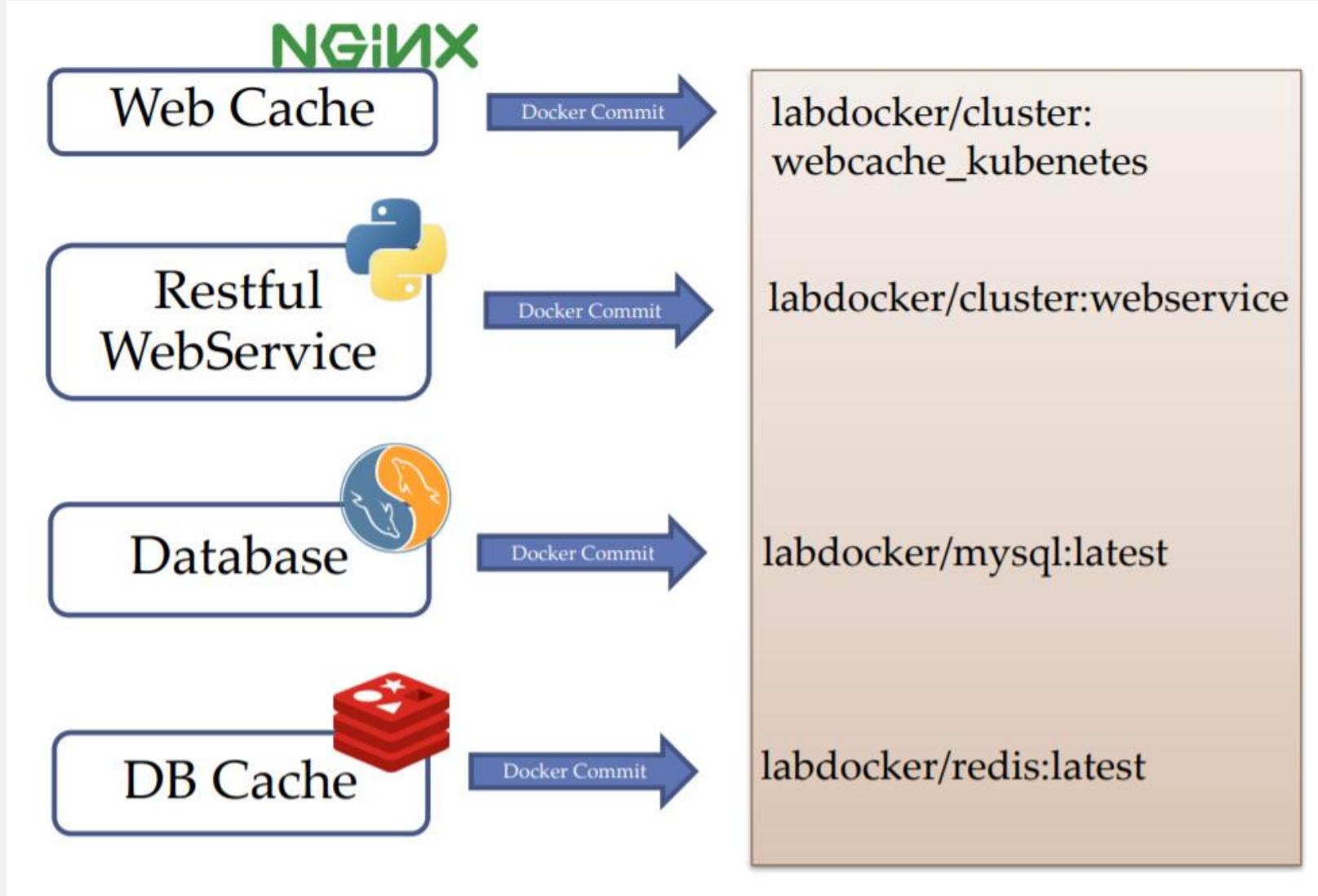


WORKSHOP
1.1: PODS
& SERVICE

- Example Case: High I/O Restful API







- Pods: maindb(YML)

```
1  apiVersion: "v1"
2  kind: Pod
3  metadata:
4    name: maindb
5    labels:
6      name: "maindb"
7      owner: "Praparn_L"
8      version: "1.0"
9      module: "maindb"
10     environment: "development"
11 spec:
12   containers:
13     - name: maindb
14       image: labdocker/mariadb:latest
15     ports:
16       - containerPort: 3306
17         protocol: TCP
18     env:
19       -
20         name: "MYSQL_ROOT_PASSWORD"
21         value: "password"
22
```

- Service: maindb(YML)

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: maindb
5    labels:
6      name: "maindb"
7      owner: "Praparn_L"
8      version: "1.0"
9      module: "maindb"
10     environment: "development"
11 spec:
12   ports:
13     - port: 3306
14       targetPort: 3306
15   selector:
16     name: "maindb"
17     owner: "Praparn_L"
18     version: "1.0"
19     module: "maindb"
20     environment: "development"
21
```

- Pods: web(YML)

```
 1 apiVersion: "v1"
 2 kind: Pod
 3 metadata:
 4   name: web
 5   labels:
 6     name: "web"
 7     owner: "Praparn_L"
 8     version: "1.0"
 9     module: "web"
10     environment: "development"
11 spec:
12   containers:
13     - name: cachedb
14       image: labdocker/redis:latest
15       ports:
16         - containerPort: 6379
17           protocol: TCP
18     - name: webservice
19       image: labdocker/cluster:webservice
20       env:
21         - name: "REDIS_HOST"
22           value: "localhost"
23       ports:
24         - containerPort: 5000
25           protocol: TCP
26     - name: webcache
27       image: labdocker/cluster:webcache_kubernetes
28       ports:
29         - containerPort: 80
30           protocol: TCP
```

Service: web (YML)

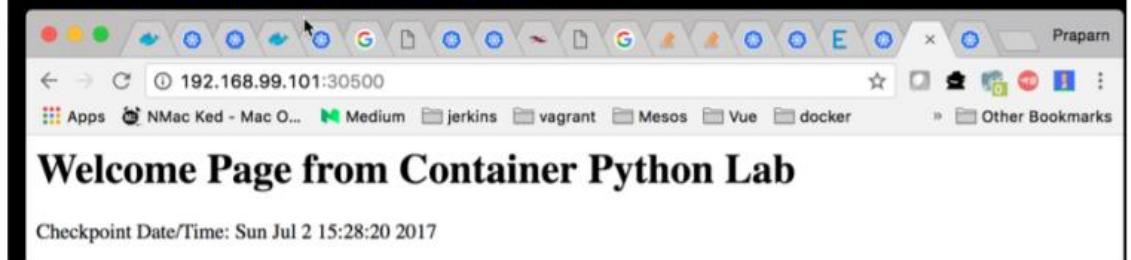
```
 1 apiVersion: v1
 2 kind: Service
 3 metadata:
 4   name: web
 5   labels:
 6     name: "web"
 7     owner: "Praparn_L"
 8     version: "1.0"
 9     module: "Web"
10     environment: "development"
11 spec:
12   selector:
13     name: "web"
14     owner: "Praparn_L"
15     version: "1.0"
16     module: "web"
17     environment: "development"
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
```

PODS, CONTAINER AND SERVICES

- Services

- Existing Pods “web”

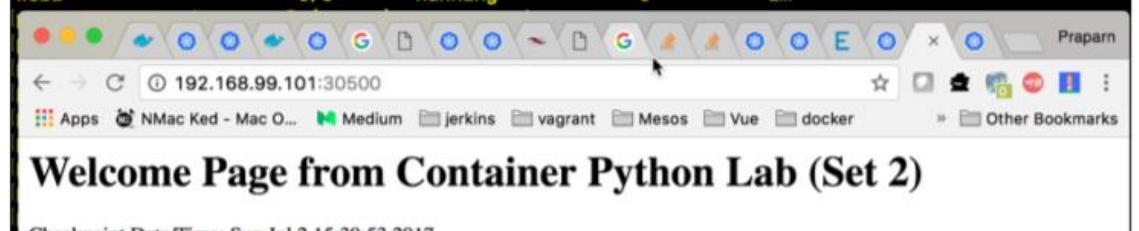
```
praparns-MacBook-Pro:multicontainer praparn$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
curl-1580724602-gjihq  0/1     CrashLoopBackOff  27      1h
maindb        1/1     Running   0         56m
web           3/3     Running   0         7m
praparns-MacBook-Pro:multicontainer praparn$ 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


Welcome Page from Container Python Lab (Set 2)



Checkpoint Date/Time: Sun Jul 2 15:30:53 2017


```
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

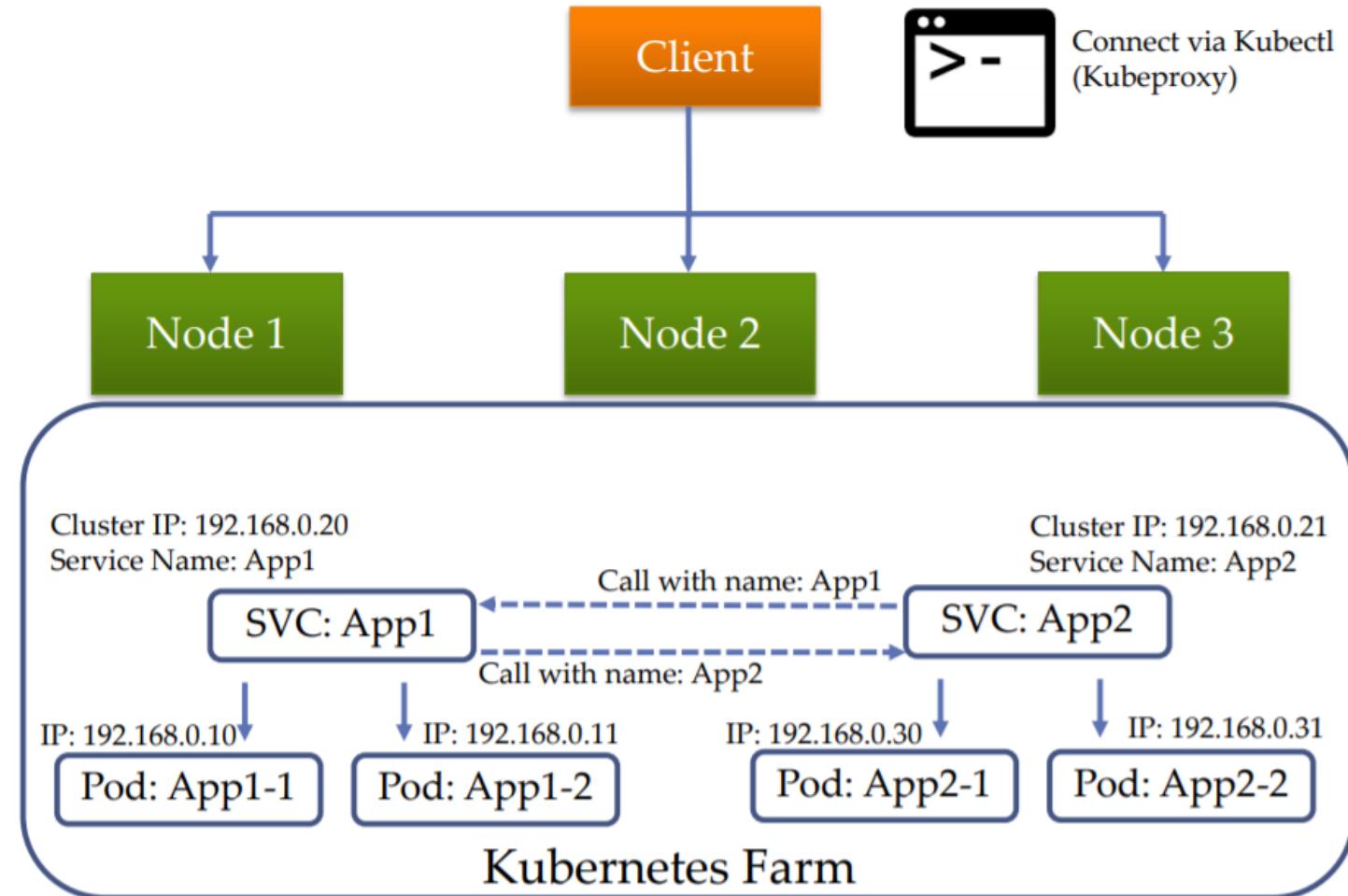
- Services:
 - Publish Service Type:
 - CLUSTER-IP (Default): blind port with ip address of cluster (Accessible from internal cluster system)
 - NodePort: blind port with ip address of node. By default kubernetes will random port (30000-32757). If we need to specify set the option: “nodePort: XXXXX”

```
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$
```

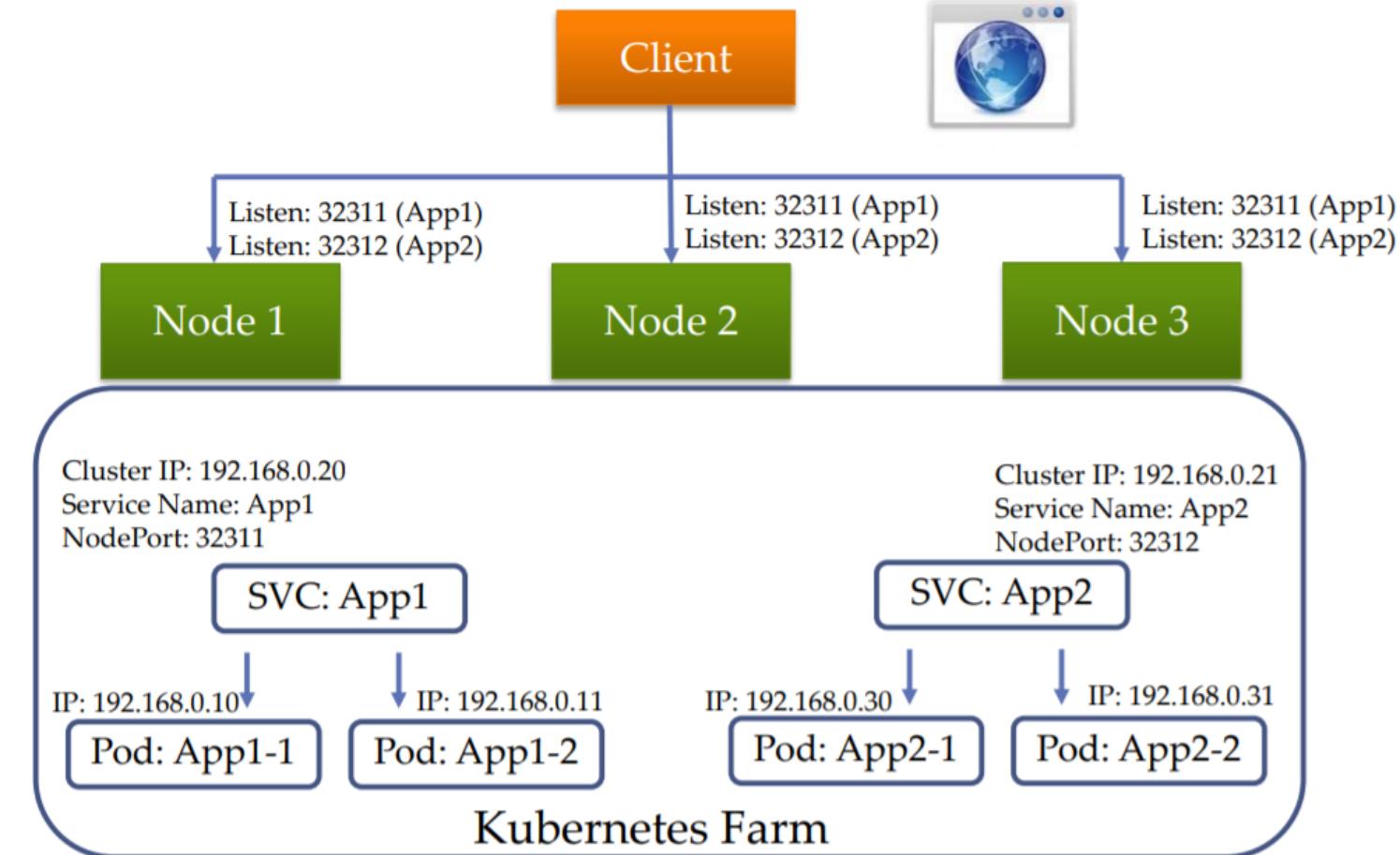
PODS, CONTAINER AND SERVICES

- Services: ClusterIP



PODS, CONTAINER AND SERVICES

- Services: NodePort



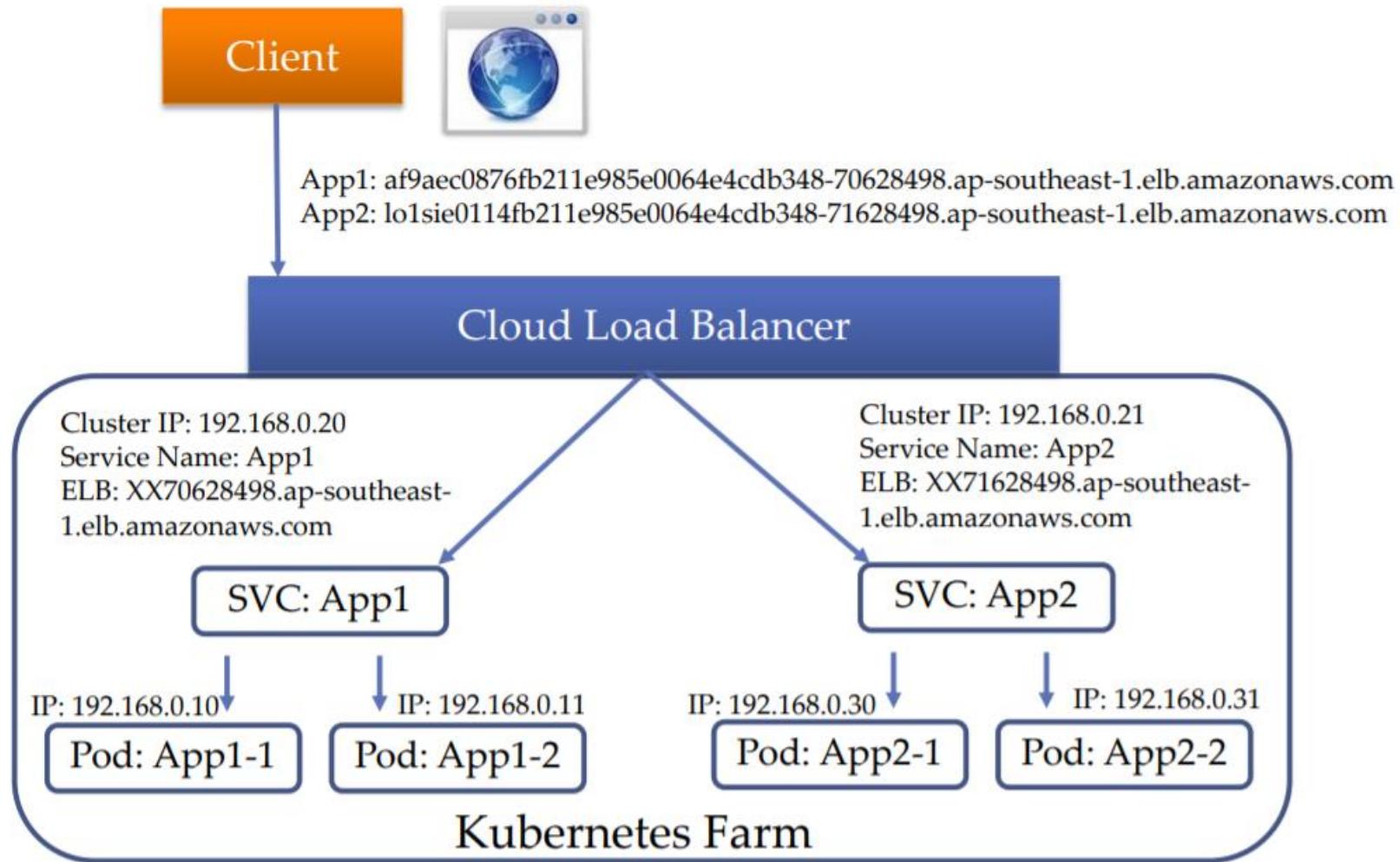
PODS, CONTAINER AND SERVICES

- 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



```
1 kind: Service
2 apiVersion: v1
3 metadata:
4   name: my-service
5 spec:
6   selector:
7     app: MyApp
8   ports:
9     - protocol: TCP
10    port: 80
11    targetPort: 9376
12    nodePort: 30061
13   clusterIP: 10.0.171.239
14   loadBalancerIP: 78.11.24.19
15   type: LoadBalancer
16 status:
17   loadBalancer:
18     ingress:
19       - ip: 146.148.47.155
20
```

- 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

```
1 kind: Service
2 apiVersion: v1
3 metadata:
4   name: my-service
5 spec:
6   selector:
7     app: MyApp
8   ports:
9     - name: http
10    protocol: TCP
11    port: 80
12    targetPort: 9376
13  externalIPs:
14    - 80.11.12.10
```

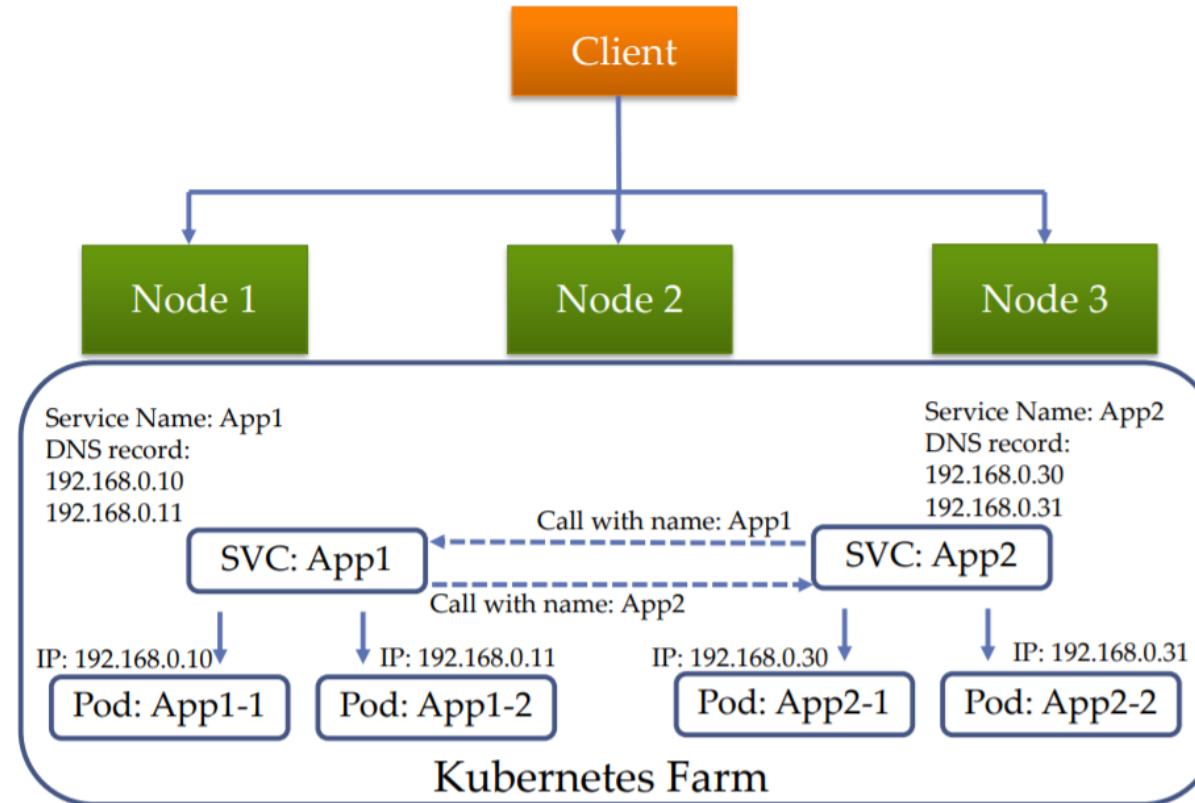
Services

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

```
! 20dns.yml ✘  
1  # A headless service to create DNS records  
2  ---  
3  apiVersion: v1  
4  kind: Service  
5  metadata:  
6    name: kafka  
7    namespace: sansiri-unityapi  
8  spec:  
9    ports:  
10   - port: 9092  
11     # [podname].broker.kafka.svc.cluster.local  
12     clusterIP: None  
13     selector:  
14       app: kafka  
15
```

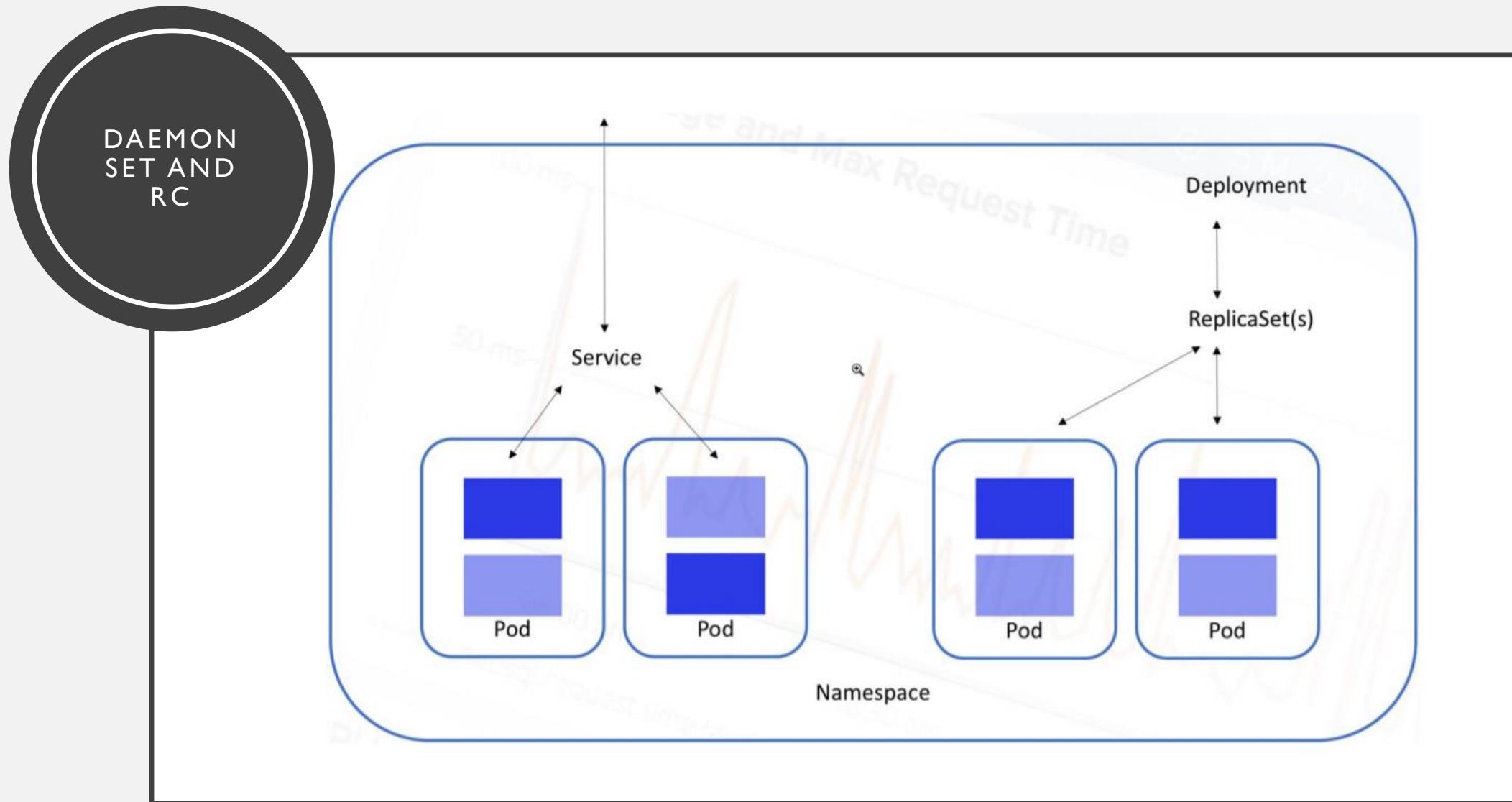
PODS,
CONTAINER
AND
SERVICES

- Services: Headless





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 work similarly.
 - RC (Replication Controller) will respond:
 - Create/Maintain Pods as "Replication Controller" (Pods Farm)
 - Keep copy of Pods (Replicas) as designed
 - Ensure that Pods are up and running with amount like design
 - If too much, it will kill some Pods
 - If too few, it will create another replicas of Pods
 - Auto healing if some Pods crash for 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”

```
1 apiVersion: "v1"
2 kind: Pod
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    containers:
13      - name: webtest
14        image: labdocker/cluster:webservicelite
15        ports:
16          - containerPort: 5000
17            protocol: TCP
```

RC “webtest”

```
1 apiVersion: v1
2 kind: ReplicationController
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    replicas: 3
13    template:
14      metadata:
15        labels:
16          name: web
17          owner: Praparn_L
18          version: "1.0"
19          module: WebServer
20          environment: development
21    spec:
22      containers:
23        - name: webtest
24          image: labdocker/cluster:webservicelite
25          ports:
26            - containerPort: 5000
27              protocol: TCP
```

SVC “webtest”

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    selector:
13      name: web
14      owner: Praparn_L
15      version: "1.0"
16      module: WebServer
17      environment: development
18
19    type: NodePort
20    ports:
21      - port: 5000
22        name: http
23        targetPort: 5000
24        protocol: TCP
25        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$ █
```

← → ⏪ ⓘ 192.168.99.100:32500

_apps NMac Ked - Mac O... Medium jenkins vagrant Mesos Vue doc

Welcome Page from Container Python Lab

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

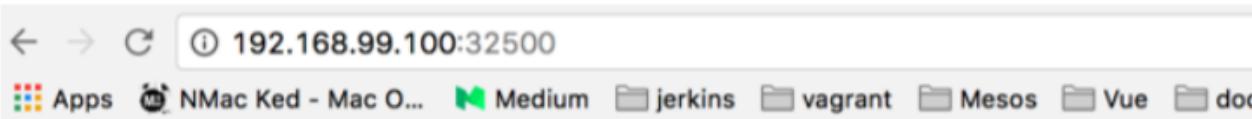
REPLICATION CONTROLLER (RC)

- 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
```

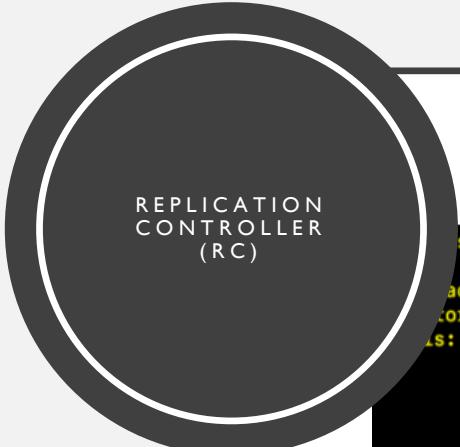
- Recheck Pods unit and available

```
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

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl describe rc webtest
  name:          webtest
  namespace:    default
  selector:     environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
  replicas:      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$
```

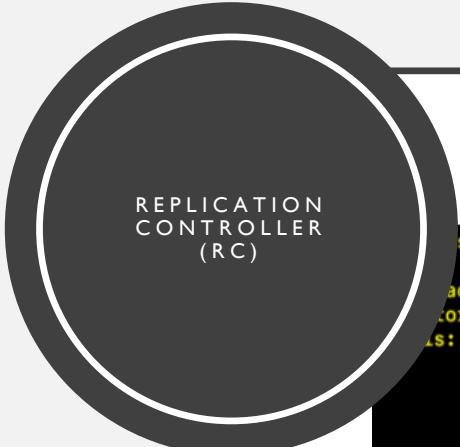
REPLICATION CONTROLLER (RC)

- 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-2hgddq  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-thhx f  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

```
praparns-MacBook-Pro:ReplicationController praparn$ kubectl describe rc webtest
  name:          webtest
  namespace:    default
  selector:     environment=development,module=WebServer,name=web,owner=Praparn_L,version=1.0
  replicas:      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$
```

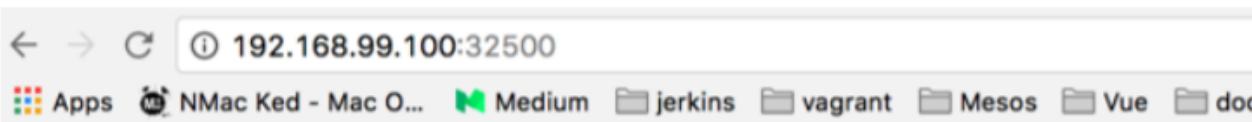
REPLICATION CONTROLLER (RC)

- 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

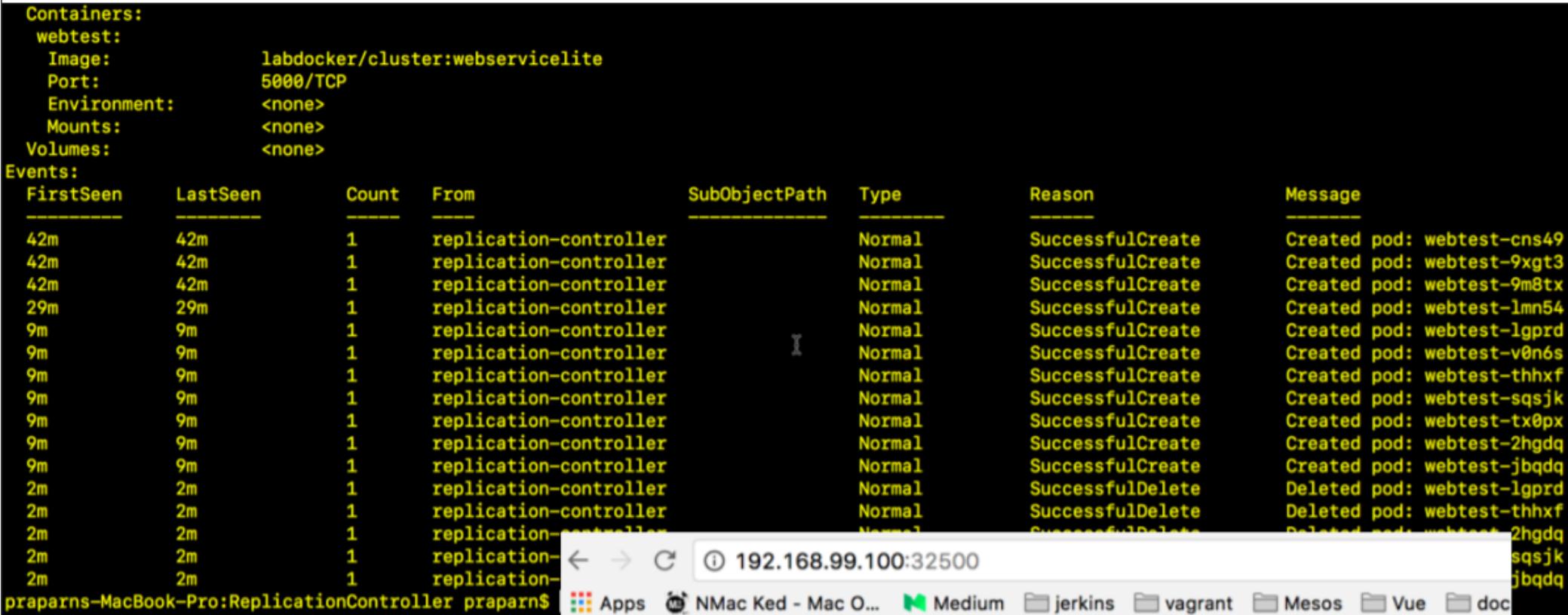
Checkpoint Date/Time: Wed Jul 5 15:55:31 2017

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-thhx
  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-thhx
  2m            2m        1      replication-controller          Normal  SuccessfulDelete  Deleted pod: webtest-2hgdq
  2m            2m        1      replication-controller          Normal  SuccessfulDelete  Deleted pod: webtest-sqsjk
  2m            2m        1      replication-controller          Normal  SuccessfulDelete  Deleted pod: webtest-jbqdq
praparn-MacBook-Pro:ReplicationController praparn$
```



The screenshot shows a terminal window on a Mac OS X desktop. The title bar says "Terminal". The command entered was "kubectl get rc". The output shows the configuration of the "webtest" container and a detailed log of events. The events log lists numerous "SuccessfulCreate" entries for various pod names (e.g., webtest-cns49, webtest-9xgt3) and one "SuccessfulDelete" entry for pod "webtest-lgprd". The terminal window is part of a larger desktop environment with other application icons visible in the dock.

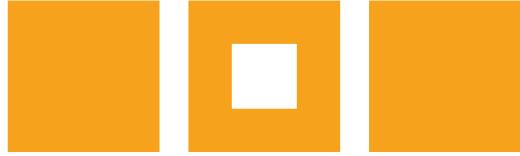
Welcome Page from Container Python Lab

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

Replication Controller

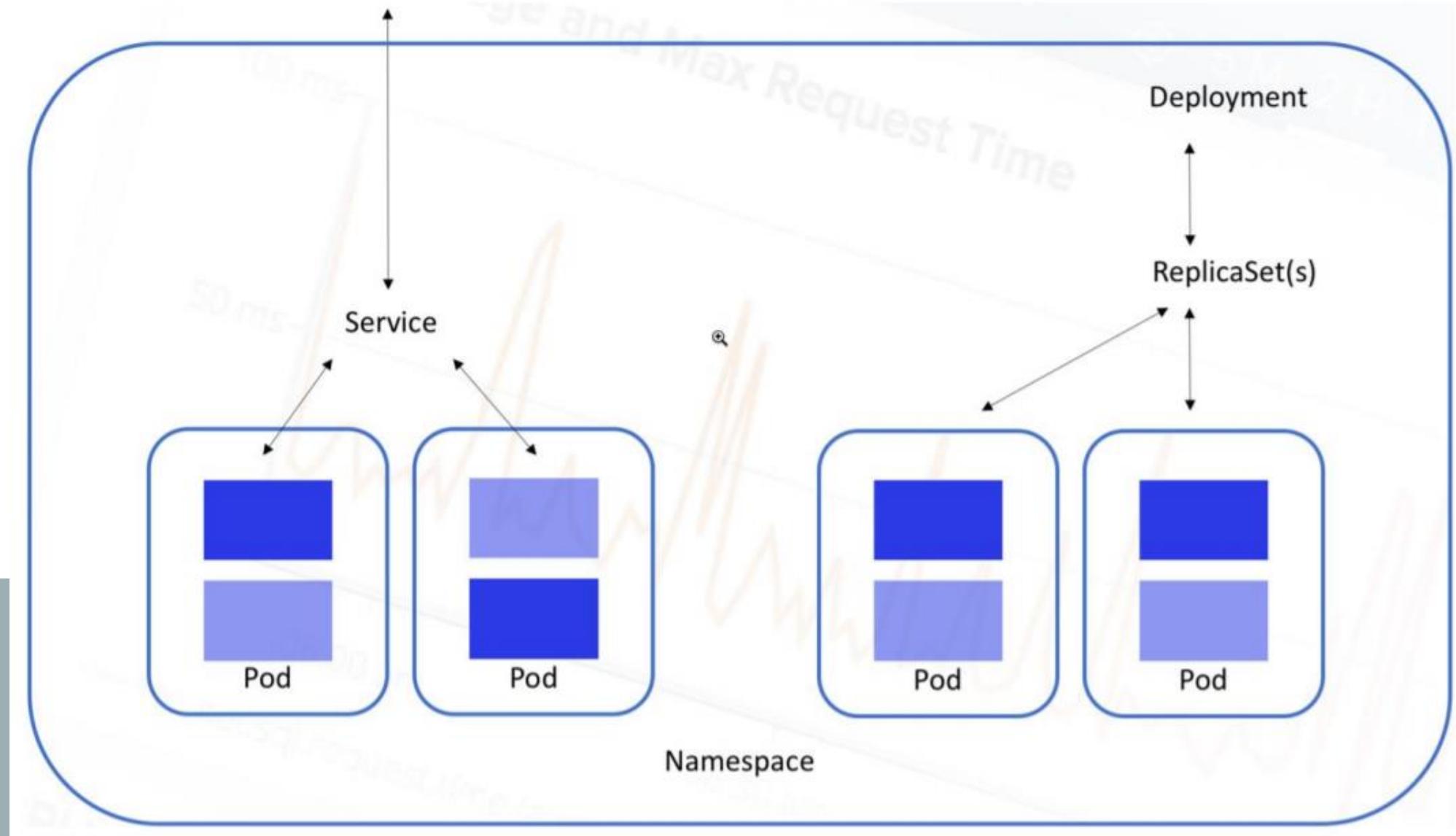
- Cleanup Lab

```
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

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”

```
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”

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

Deployment/RS and Update

- Example

- labdocker/cluster:webservicelite_v1



- Welcome Page from Container Python Lab Web Version 1.00**

Checkpoint Date/Time: Thu Jul 6 15:19:27 2017

- labdocker/cluster:webservicelite_v1.51rc



- Welcome Page from Container Python Lab Web Version 1.51 RC**

Checkpoint Date/Time: Thu Jul 6 15:39:05 2017

- labdocker/cluster:webservicelite_v1.8ga



- Welcome Page from Container Python Lab Web Version 1.80 GA**

Checkpoint Date/Time: Thu Jul 6 15:36:54 2017

- Example

- Deployment “webtest”

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    replicas: 3
13    selector:
14      matchLabels:
15        name: web
16        owner: Praparn_L
17        version: "1.0"
18        module: WebServer
19        environment: development
20    template:
21      metadata:
22        labels:
23          name: web
24          owner: Praparn_L
25          version: "1.0"
26          module: WebServer
27          environment: development
28    spec:
29      containers:
30        - name: webtest
31          image: labdocker/cluster:webservicelite_v1
32          ports:
33            - containerPort: 5000
34              protocol: TCP
```

- SVC “webtest”

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    selector:
13      name: web
14      owner: Praparn_L
15      version: "1.0"
16      module: WebServer
17      environment: development
18
19    type: NodePort
20    ports:
21      - port: 5000
22        name: http
23        targetPort: 5000
24        protocol: TCP
25        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  deployment-controller  Normal  ScalingReplicaSet  Scaled up replica set webtest-4
261491039 to 3
praparns-MacBook-Pro:WorkShop_1.4_Deployment praparn$
```

```
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  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
 - 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 (maxUnavailable) and 25% for over design (maxSurge)



kubernetes

Deployment/RS and Update

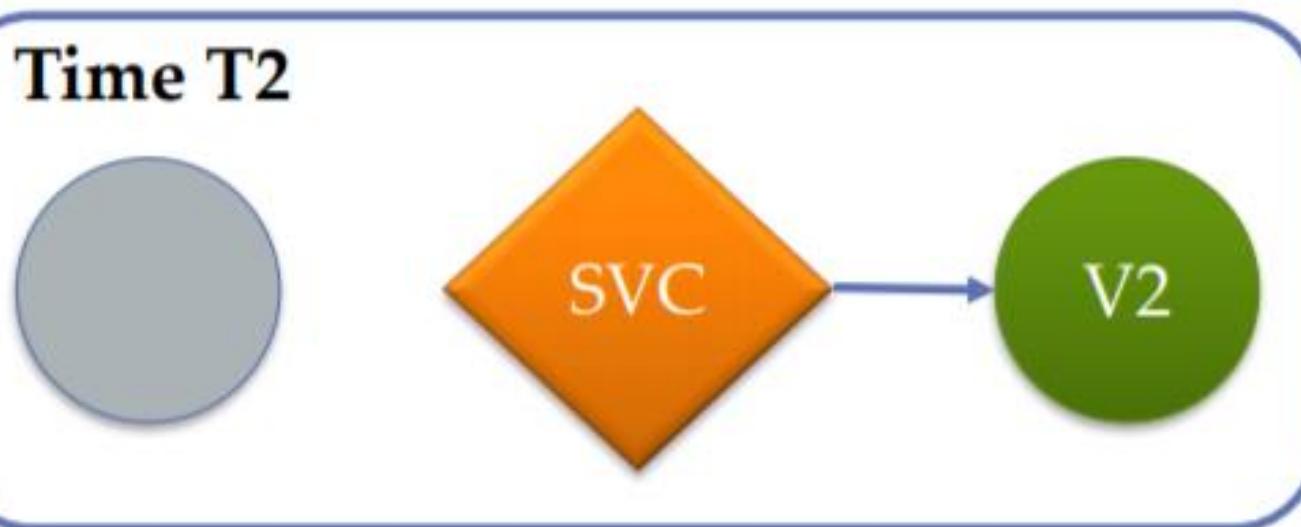
- Let's Talk about deployment's strategies !!!
 - On deployment we have many procedure for apply rollout strategy with deployment
 - Recreate (For Development): Destroy all pods and recreate it again.
 - RollingUpdate (Default): Create new version of pods and wait for it ready before terminate existing 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)

Time T1

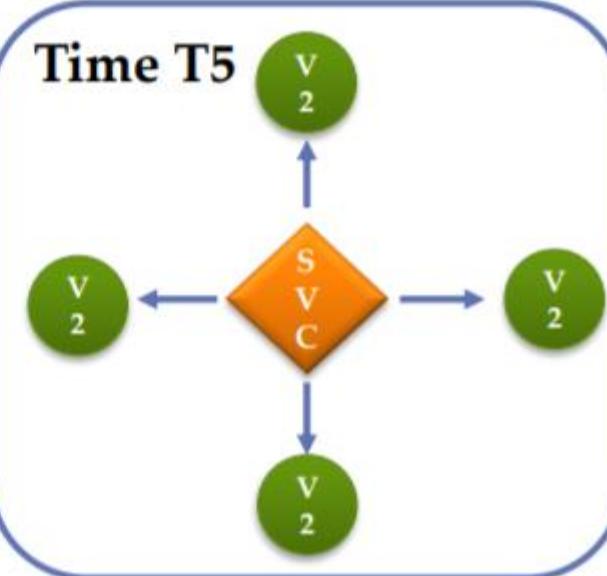
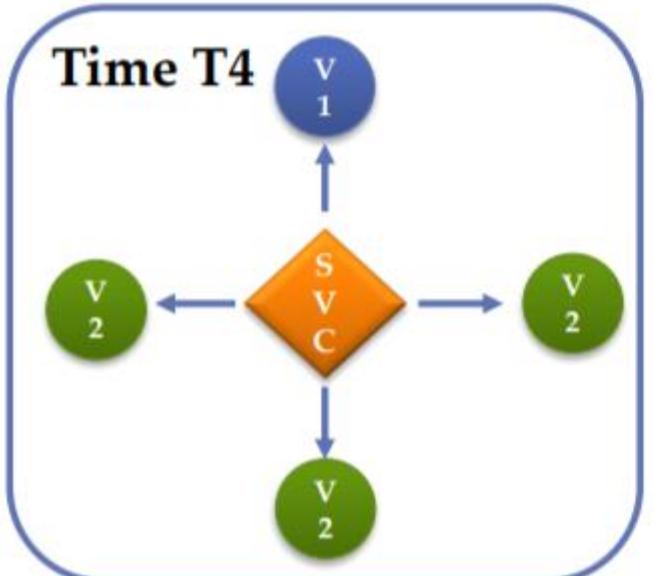
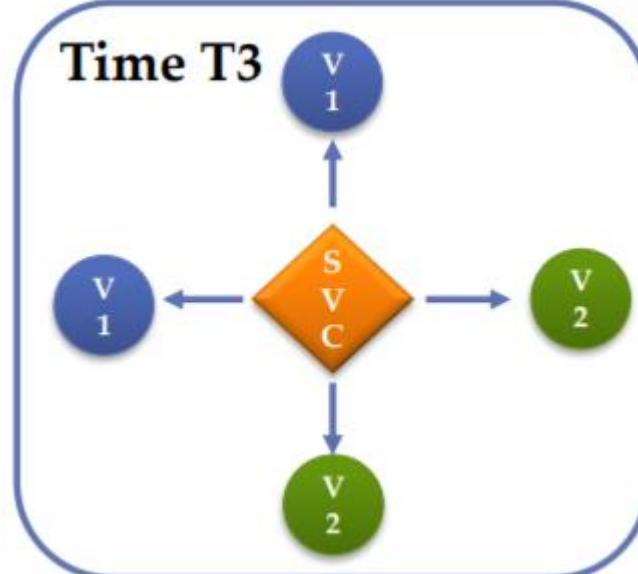
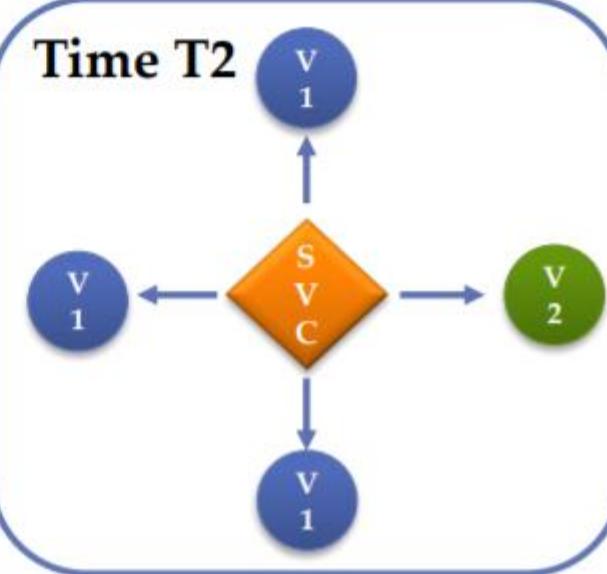
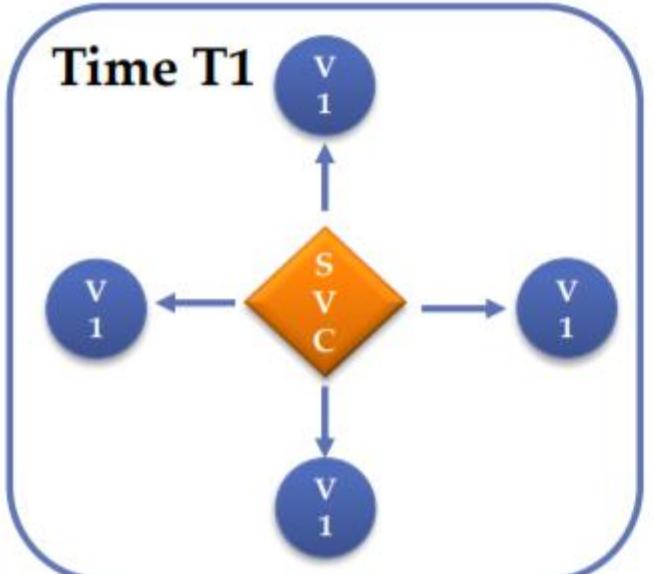


Time T2

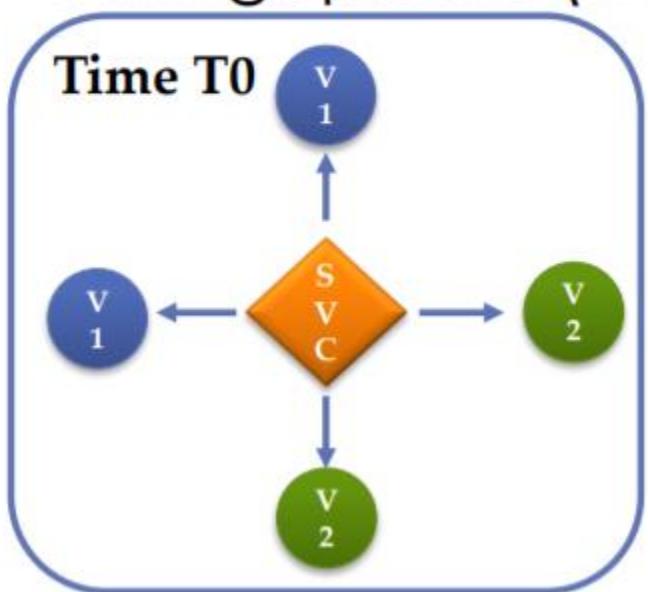


```
! webtest_deploy.yml •  
1 apiVersion: apps/v1  
2 kind: Deployment  
3 metadata:  
4   name: webtest  
5   labels:  
6     name: web  
7     owner: Praparn_L  
8   version: "1.0"  
9   module: WebServer  
10  environment: development  
11 spec:  
12   replicas: 1  
13   strategy:  
14     type: Recreate  
15   selector:  
16     matchLabels:  
17       name: web  
18       owner: Praparn_L
```

- RollingUpdate (Default)

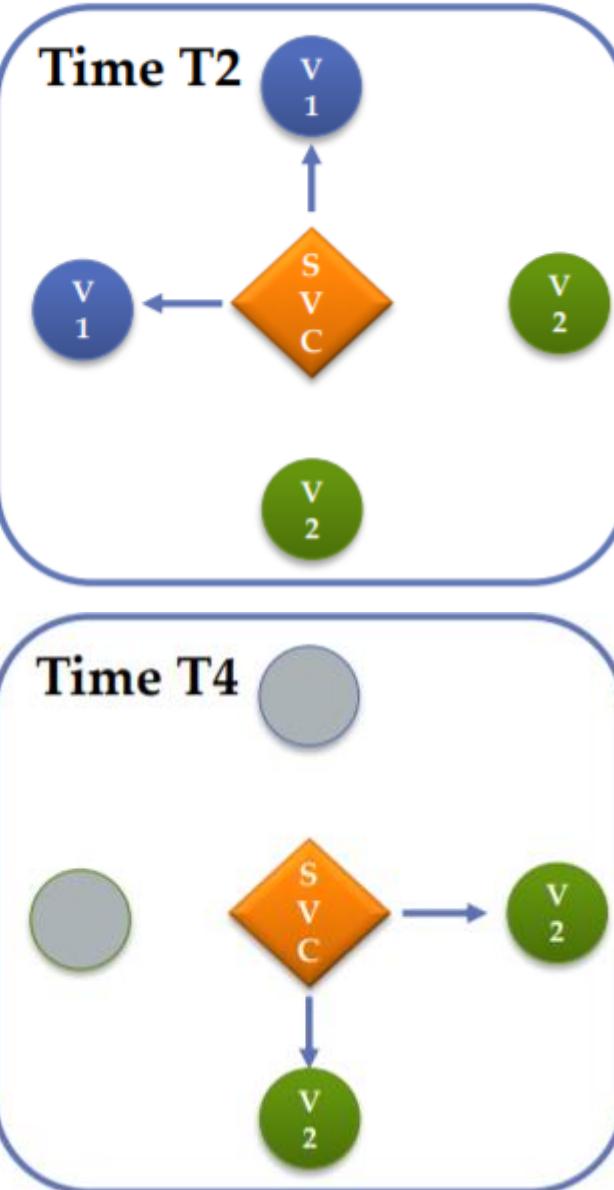
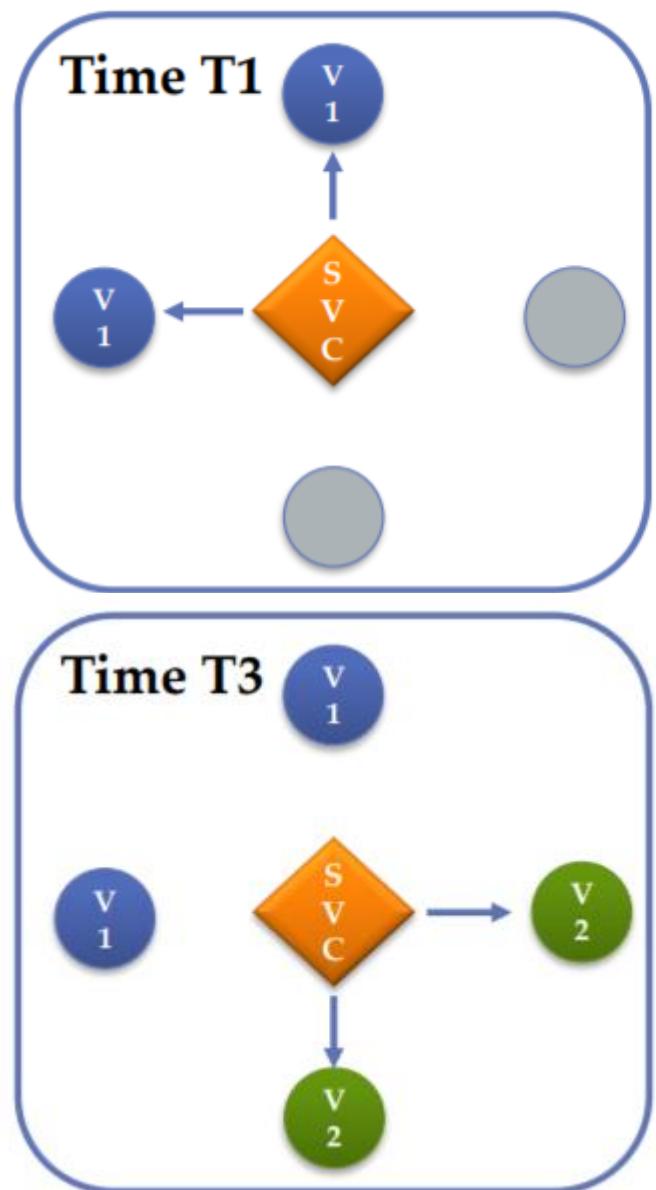


- RollingUpdate (Default)



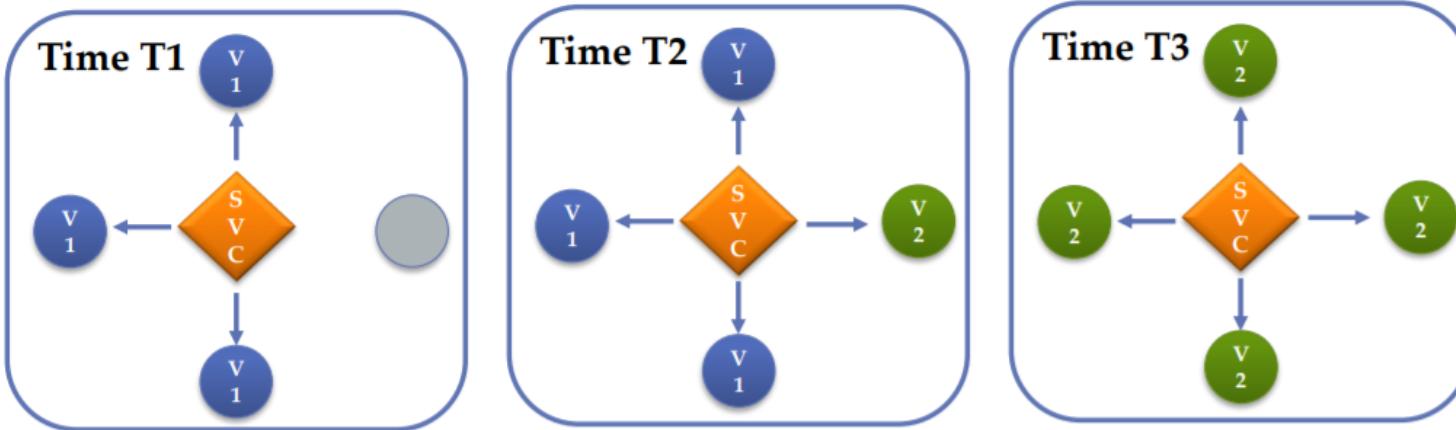
```
! webtest_deploy.yml ●  
1  apiVersion: apps/v1  
2  kind: Deployment  
3  metadata:  
4    name: webtest  
5    labels:  
6      name: web  
7    owner: Praparn_L  
8    version: "1.0"  
9    module: WebServer  
10   environment: development  
11 spec:  
12   replicas: 4  
13   strategy:  
14     type: RollingUpdate  
15     rollingUpdate:  
16       maxUnavailable: 1  
17       maxSurge: 1  
18   selector:  
19     matchLabels:  
20       name: web  
21       owner: Praparn_L
```

- Blue/Green



```
! webtest_deploy.yml • ! webtest_svc.yml x
1   apiVersion: v1
2     kind: Service
3       metadata:
4         name: webtest
5           labels:
6             name: web
7             owner: Praparn_L
8             version: "1.0"
9             module: WebServer
10            environment: development
11      spec:
12        selector:
13          name: web
14          owner: Praparn_L
15          version: "1.0"
16          module: WebServer
17          environment: development
```

- Canary: (2 Deployment same label)



```

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4  |   name: webtest_V1
5  |   labels:
6  |     name: web
7  |     owner: Praparn_L
8  |     version: "1.0"
9  |     module: WebServer
10 |     environment: development
11 spec:
12 |   replicas: 3

```

```

1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4  |   name: webtest_V2
5  |   labels:
6  |     name: web
7  |     owner: Praparn_L
8  |     version: "1.0"
9  |     module: WebServer
10 |     environment: development
11 spec:
12 |   replicas: 1

```

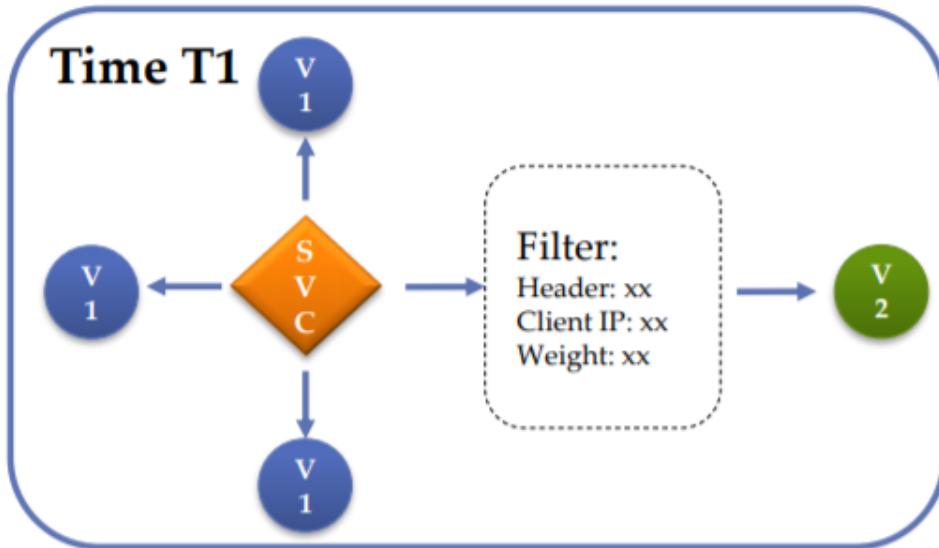
```

! webtest_deploy.yml • ! webtest_svc.yml ✘
1  apiVersion: v1
2  kind: Service
3  metadata:
4  |   name: webtest
5  |   labels:
6  |     name: web
7  |     owner: Praparn_L
8  |     version: "1.0"
9  |     module: WebServer
10 |     environment: development
11 spec:
12 |   selector:
13 |     name: web
14 |     owner: Praparn_L
15 |     version: "1.0"
16 |     module: WebServer
17 |     environment: development

```

Kubernetes: Production Workload Orchestration

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



AWS App Mesh
Application-level networking for all your services



Istio
Connect, secure, control, and observe services.

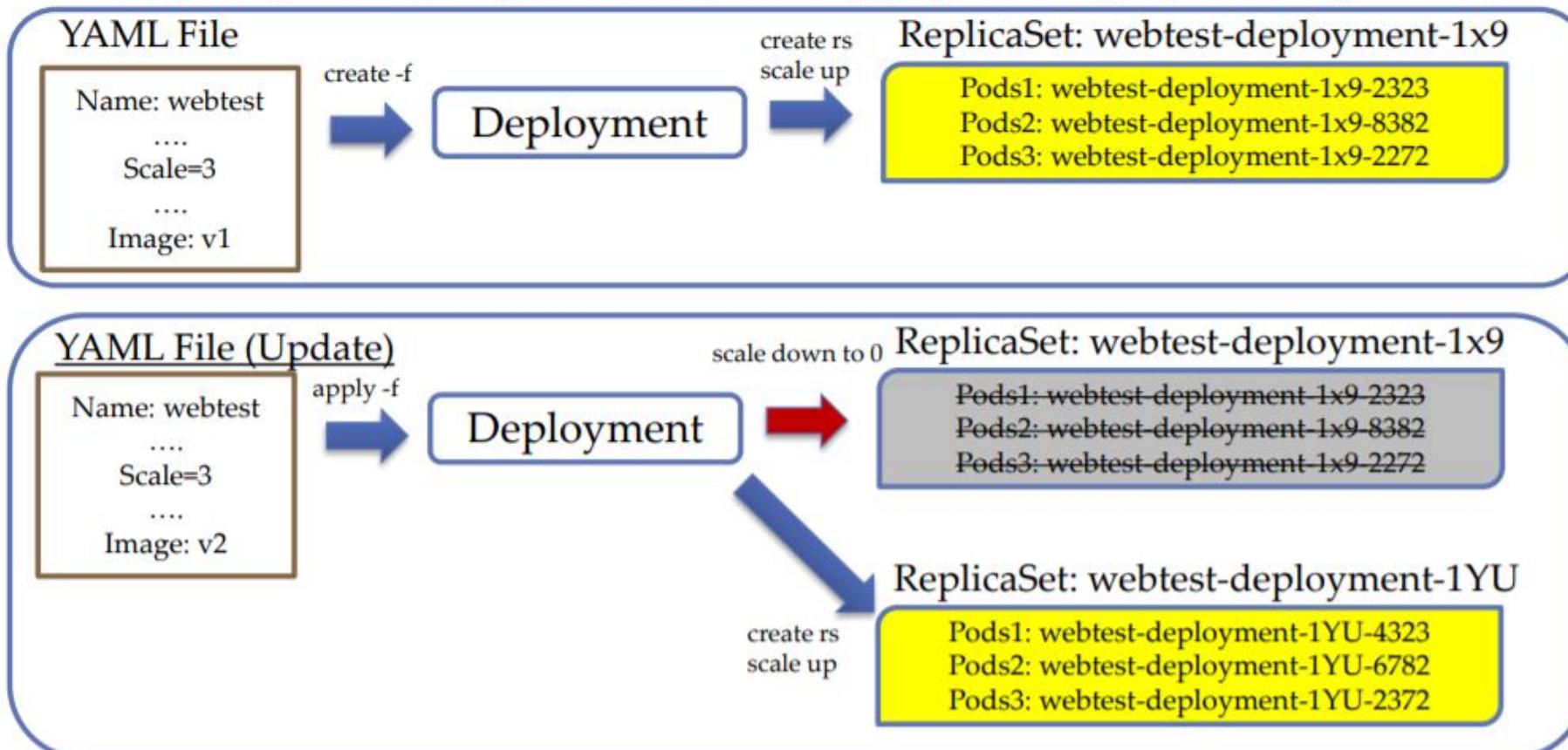


LINKERD



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-7dc5bdcd7   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.yaml --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 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 Pod'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

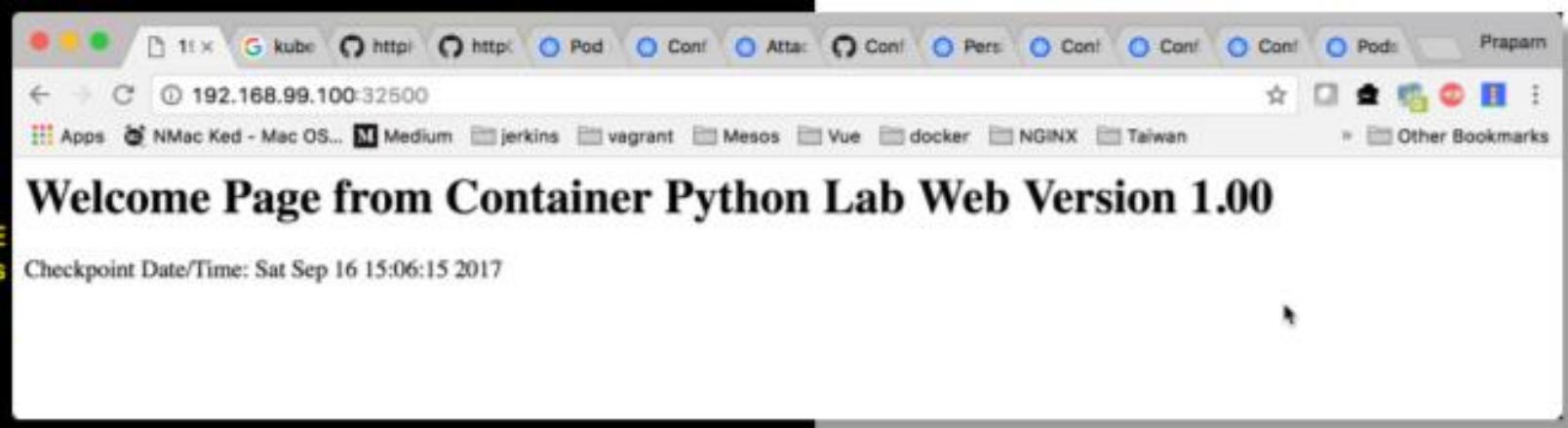
```
41      - name: ES_JAVA_OPTS
42      |   value: -Xms5g -Xmx5g
43      - name: bootstrap.memory_lock
44      |   value: "false"
45      - name: cluster.name
46      |   value: pam-es-docker-cluster
47      - name: xpack.security.enabled
48      |   value: "false"
49      image: "3dsinteractive/elasticsearch-std:6.6"
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:
62
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:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11  spec:
12    replicas: 1
13    selector:
14      matchLabels:
15        name: web
16        owner: Praparn_L
17        version: "1.0"
18        module: WebServer
19        environment: development
20    template:
21      metadata:
22        labels:
23          name: web
24          owner: Praparn_L
25          version: "1.0"
26          module: WebServer
27          environment: development
28      spec:
29        containers:
30          - name: webtest
31            image: labdocker/cluster:webservicelite_v1
32        ports:
33          - containerPort: 5000
34            protocol: TCP
35        readinessProbe:
36          exec:
37            command:
38              - cat
39              - /usr/src/app/main.py
40            initialDelaySeconds: 15
41            periodSeconds: 5
42        livenessProbe:
43          exec:
44            command:
45              - cat
46              - /usr/src/app/main.py
47            initialDelaySeconds: 15
48            periodSeconds: 15
```

```
kubectl create -f webtest_svc.yaml
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:      environment=development
                 module=WebServer
                 name=web
                 owner=Praparn_L
                 version=1.0
```



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":"ReplicaSet","name":"webtest-336910061","uid":"4992f987-9af0-11e7-bc1a-080027fb95be"},...}
Status:               Running
```



LIVENESS AND
READINESS

Events:	FirstSeen	LastSeen	Count	From	SubObjectPath	Type	Reason
↳ signed webtest-336910061-26pdb to minikube	7m	7m	1	default-scheduler		Normal	Scheduled
↳ Up succeeded for volume "default-token-pcw4h"	7m	7m	1	kubelet, minikube		Normal	Success
↳ failed: cat: can't open '/usr/src/app/main.py': No such file or directory	2m	2m	8	kubelet, minikube	spec.containers{webtest}	Warning	Unhealthy
↳ cc/app/main.py': No such file or directory	2m	2m	3	kubelet, minikube	spec.containers{webtest}	Warning Unhealthy	Liveness probe
↳ already present on machine	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Pulled	Container image "labdoc"
↳	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Created	Created container
↳	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Started	Started container
↳ -336910061-26pdb_default(499cef14-9af0-11e7-bc1a-080027fb95be)" container "webtest" is unhealthy, it will be killed	2m	2m	1	kubelet, minikube	spec.containers{webtest}	Normal Killing	Killing container with
↳ barns-MacBook-Pro%							

LIVENESS AND READINESS

ExecAction:

Events:	FirstSeen	LastSeen	Count	From	SubObjectPath	Type	Reason
↳ signed webtest-336910061-26pdb to minikube	7m	7m	1	default-scheduler		Normal	Scheduled
↳ Up succeeded for volume "default-token-pcw4h"	7m	7m	1	kubelet, minikube		Normal	Success
↳ failed: cat: can't open '/usr/src/app/main.py': No such file or directory	2m	2m	8	kubelet, minikube	spec.containers{webtest}	Warning	Unhealthy
↳ cc/app/main.py': No such file or directory	2m	2m	3	kubelet, minikube	spec.containers{webtest}	Warning Unhealthy	Liveness probe
↳ already present on machine	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Pulled	Container image "labdoc"
↳	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Created	Created container
↳	2m	2m	2	kubelet, minikube	spec.containers{webtest}	Normal Started	Started container
↳ -336910061-26pdb_default(499cef14-9af0-11e7-bc1a-080027fb95be)" container "webtest" is unhealthy, it will be killed	2m	2m	1	kubelet, minikube	spec.containers{webtest}	Normal Killing	Killing container with
↳ barns-MacBook-Pro%							

LIVENESS AND READINESS

ExecAction:

LIVENESS AND READINESS

- TCPSocketAction:

```
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9     module: WebServer
10    environment: development
11 spec:
12   replicas: 1
13   selector:
14     matchLabels:
15       name: web
16       owner: Praparn_L
17       version: "1.0"
18       module: WebServer
19       environment: development
20   template:
21     metadata:
22       labels:
23         name: web
24         owner: Praparn_L
25         version: "1.0"
26         module: WebServer
27         environment: development
28     spec:
29       containers:
30         - name: webtest
31           image: labdocker/cluster:webservicelite_v1
32         ports:
33           - name: webservice
34             containerPort: 5000
35             protocol: TCP
36             readinessProbe:
37               tcpSocket:
38                 port: 5000
39             initialDelaySeconds: 15
40             periodSeconds: 5
41             livenessProbe:
42               tcpSocket:
43                 port: 5000
44             initialDelaySeconds: 15
45             periodSeconds: 15
```

LIVENESS AND READINESS

- **TCPSocketAction:**

① 192.168.99.100:32500

NMac Ked - Mac OS... M Medium jenkins vagrant Mesos Vue docker NGINX Other Bookmarks

ome Page from Container Python Lab Web Version 1.00

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

```
s-MacBook-Pro% kubectl create -f webtest_deploy_liveness_readiness_port.yml
deployment "webtest" created
s-MacBook-Pro% kubectl get deployment/webtest
  DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
    1         1         1           0          14s
s-MacBook-Pro% kubectl get pods --show-labels
                    READY  STATUS  RESTARTS  AGE      LABELS
-3579274848-1w2mn   0/1    Running   0          18s  environment=development,
79274848,version=1.0

events
  Type    Reason     LastSeen   Count   From          SubObjectPath
  ----  -----     ----       ---   ----          -----
  Normal  Scheduled  2m        1      default-scheduler
  Normal  PodScheduled  2m        1      kubelet, minikube
  Normal  PullSuccess  2m        1      kubelet, minikube
  Normal  PullImage    2m        1      kubelet, minikube  spec.containers{webtest}
  Normal  PodScheduled  2m        1      kubelet, minikube
  Normal  PullImage    2m        1      kubelet, minikube  spec.containers{webtest}
  Normal  PodScheduled  2m        1      kubelet, minikube
  Normal  PullImage    2m        1      kubelet, minikube  spec.containers{webtest}
```

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","name":"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 "labdockerc/cluster:webserviceelite_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:

The screenshot shows a Mac OS X desktop environment. In the top-left corner, a browser window displays the "Welcome Page from Container Python Lab Web Version 1.00". The page includes a timestamp: "Checkpoint Date/Time: Sat Sep 16 15:54:10 2017". In the top-right corner, a Postman application window is open, showing a list of API requests in the "History" tab and detailed headers for a recent GET request to "http://192.168.99.100:32500".

Postman History:

- GET http://192.168.99.100:32500
- GET http://192.168.99.100/users/removeUser/99
- GET http://192.168.99.100/users/1
- POST http://192.168.99.100/users/insertUser
- GET http://192.168.99.100/init
- GET http://192.168.99.100/

Postman Request Details:

- Method: GET
- URL: http://192.168.99.100:32500
- Authorization: No Auth
- Headers:
 - content-length → 113
 - content-type → text/html; charset=utf-8
 - date → Sat, 16 Sep 2017 15:56:21 GMT
 - server → Werkzeug/0.12.3 Python/2.7.13
- Status: 200 OK
- Time: 46 ms

LIVENESS AND READINESS

- HTTPGetAction:

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: webtest
5    labels:
6      name: web
7      owner: Praparn_L
8      version: "1.0"
9      module: WebServer
10     environment: development
11   spec:
12     replicas: 1
13     selector:
14       matchLabels:
15         name: web
16         owner: Praparn_L
17         version: "1.0"
18         module: WebServer
19         environment: development
20     template:
21       metadata:
22         labels:
23           name: web
24           owner: Praparn_L
25           version: "1.0"
26           module: WebServer
27           environment: development
```

```
28   spec:
29     containers:
30       - name: webtest
31         image: labdocker/cluster:webserviceelite_v1
32         ports:
33           - name: webservice
34             containerPort: 5000
35             protocol: TCP
36         readinessProbe:
37           httpGet:
38             # Optional "host: my-host" for set specific ip of Pods
39             # Optional "scheme: HTTPS"
40             #schema: HTTPS
41             path: /
42             port: webservice
43             httpHeaders:
44               - name: server
45                 value: "Werkzeug/0.12.2 Python/2.7.13"
46             initialDelaySeconds: 15
47             periodSeconds: 5
48             timeoutSeconds: 10
49             successThreshold: 1
50             failureThreshold: 3
51     livenessProbe:
52       httpGet:
53         # Optional "host: my-host" for set specific ip of Pods
54         # Optional "scheme: HTTPS"
55         #schema: HTTPS
56         path: /
57         port: webservice
58         httpHeaders:
59           - name: server
60             value: "Werkzeug/0.12.2 Python/2.7.13"
61         initialDelaySeconds: 15
62         periodSeconds: 5
63         timeoutSeconds: 10
64         successThreshold: 1
65         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   <none>           Normal  Scheduled  Successful
y assigned webtest-1098172621-zvd22 to minikube
9m         9m        1   kubelet, minikube   <none>           Normal  SuccessfulMountVolume  MountVolume
 SetUp succeeded for volume "default-token-pcw4h"
9m         9m        1   kubelet, minikube   spec.containers{webtest}  Normal  Pulled    Container i
mage "labdockerc/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:

```
49      resources:
50      livenessProbe:
51        httpGet:
52          # Optional "host: my-host" for set specific ip of Pods
53          # Optional "scheme: HTTPS"
54          #schema: HTTPS
55          path: /init
56          port: webservice
57          httpHeaders:
58            - name: server
59              value: "Werkzeug/0.12.2 Python/2.7.13"
60        initialDelaySeconds: 15
61        periodSeconds: 5
62        timeoutSeconds: 5
63        successThreshold: 1
64        failureThreshold: 3
65      livenessProbe:
66        httpGet:
67          # Optional "host: my-host" for set specific ip of Pods
68          # Optional "scheme: HTTPS"
69          #schema: HTTPS
70          path: /init
71          port: webservice
72          httpHeaders:
73            - name: server
74              value: "Werkzeug/0.12.2 Python/2.7.13"
75        initialDelaySeconds: 15
76        periodSeconds: 5
77        timeoutSeconds: 5
78        successThreshold: 1
79        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-fpxw4 to minikube							MountVolume	
1m	1m	1	kubelet, minikube		Normal	SuccessfulMountVolume	MountVolume	
.SetUp succeeded for volume "default-token-pcw4h"								
1m	13s	5	kubelet, minikube	spec.containers{webtest}	Normal	Pulled	Container	
image "labdocker/cluster:webserviceelite_v1" already present on machine								
1m	13s	9	kubelet, minikube	spec.containers{webtest}	Warning	Unhealthy	Readiness	
probe 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 c	
tainer with id docker://webtest:pod "webtest-692878837-fpxw4_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 c	
tainer								
1m	12s	5	kubelet, minikube	spec.containers{webtest}	Normal	Started	Started c	
tainer								
praparns-MacBook-Pro% [

LIVENESS AND READINESS

WORKSHOP : LIVENESS AND READINESS

The image consists of two side-by-side screenshots. The left screenshot shows a web browser window with the URL `http://192.168.99.100:32500`. The page title is "Welcome Page from Container Python Lab Web Version 1.00". Below the title, it says "Checkpoint Date/Time: Sat Sep 16 15:54:10 2017". The right screenshot shows the Postman application interface. It displays a history of API requests made to `http://192.168.99.100:32500`. One specific request is highlighted: a GET request to `http://192.168.99.100:32500` with the status 200 OK and a time of 46 ms. The Headers tab of the request details shows the following:

Name	Value
content-length	113
content-type	text/html; charset=utf-8
date	Sat, 16 Sep 2017 15:56:21 GMT
server	Werkzeug/0.12.2 Python/2.7.13



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

```
1 apiVersion: v1
2 kind: ConfigMap
3 metadata:
4   name: webmodule_configmap
5   namespace: webmicroservice
6   labels:
7     name: "webmodule_configmap"
8     owner: "Praparn_L"
9     version: "1.0"
10    module: "ConfigMap"
11    environment: "development"
12 data:
13   REDIS_HOST: localhost| I
```

```
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

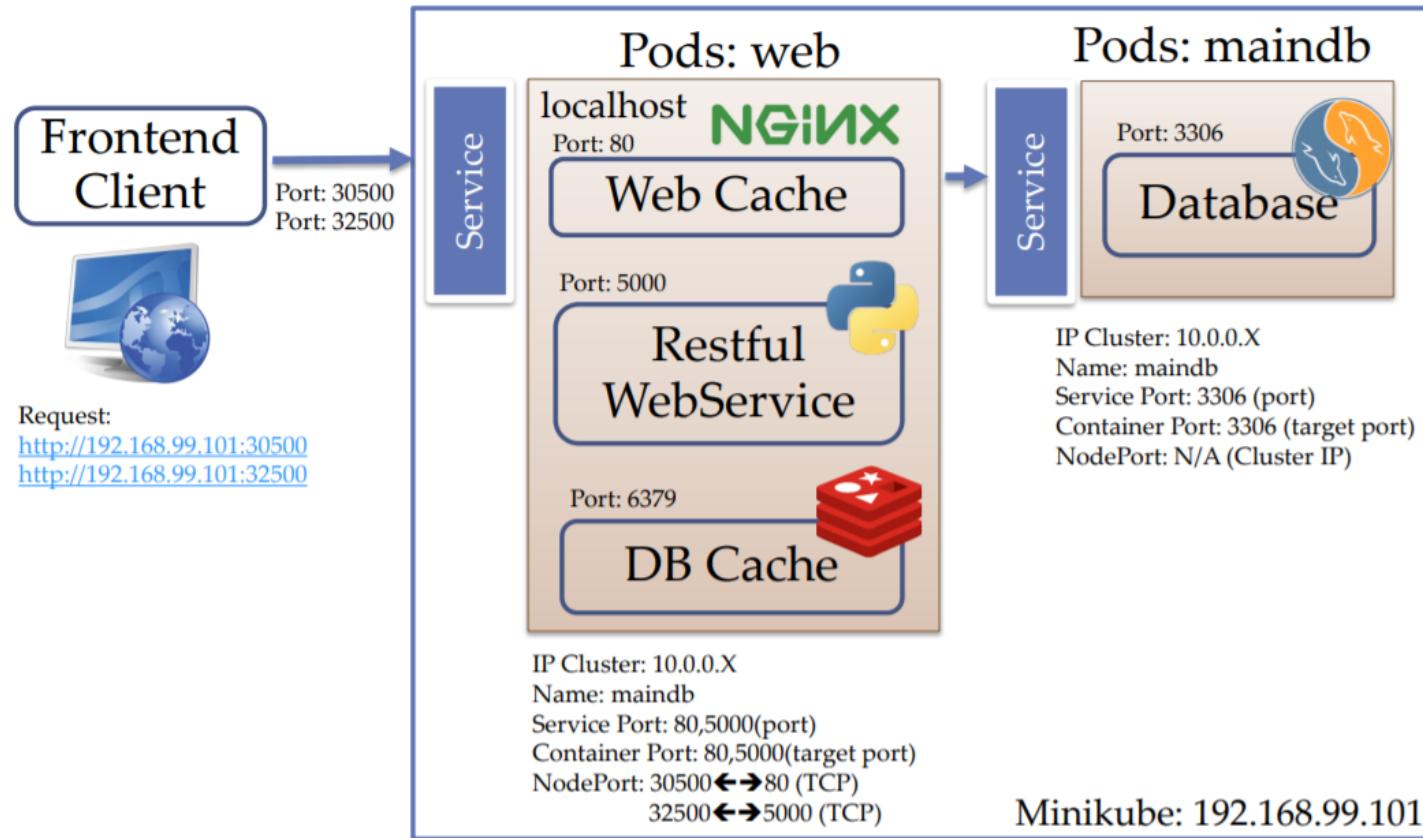
```
kubectl create secret generic <name> <option>
```

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

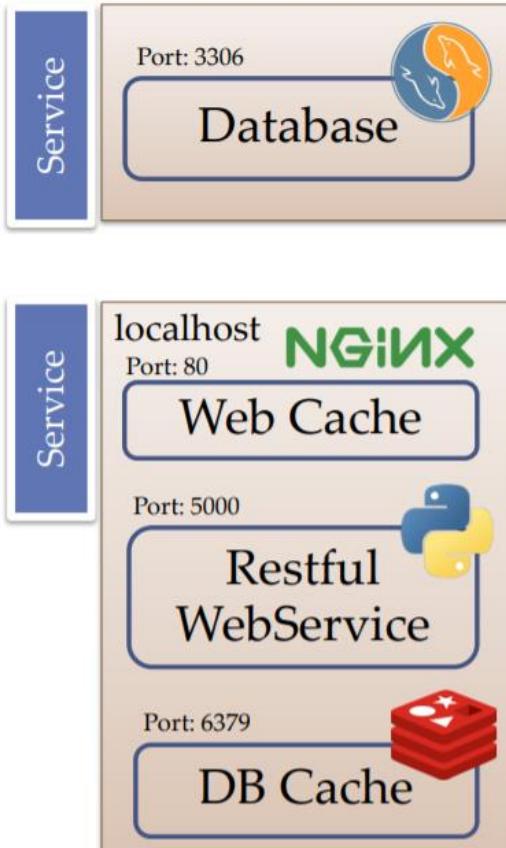
```
praparns-MBP:~ praparn$ echo -n "root"|base64  
cm9vdA==  
praparns-MBP:~ praparn$ echo -n "password"|base64  
cGFzc3dvcmQ=  
praparns-MBP:~ praparn$
```

```
1  apiVersion: v1  
2  kind: Secret  
3  metadata:  
4    name: databasemodule_secret  
5    namespace: webmicroservice  
6    labels:  
7      name: "databasemodule_secret"  
8      owner: "Praparn_L"  
9      version: "1.0"  
10     module: "Secret"  
11     environment: "development"  
12   type: Opaque  
13   data:  
14     username: cm9vdA==  
15     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"
```

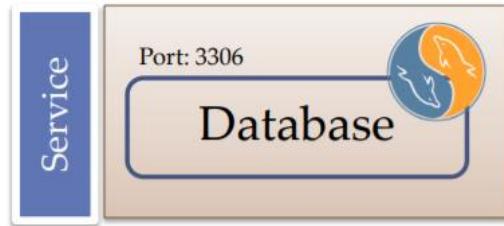
Sourcecode: main.py

```
redis = 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.yaml: databasemodule_secret.yaml

```
1 apiVersion: v1
2 kind: Secret
3 metadata:
4   name: databasemodule-secret
5   namespace: webmicroservice
6   labels:
7     name: "databasemodule-secret"
8     owner: "Praparn_L"
9     version: "1.0"
10    module: "Secret"
11    environment: "development"
12 type: Opaque
13 data:
14   username: cm9vdA==
15   password: cGFzc3dvcmQ=
```

Deploy.yaml: databasemodule_deploy_config.yaml

```
22 apiVersion: "v1"
23 kind: Deployment
24 metadata:
25   name: maindb
26   labels:
27     name: "maindb"
28     owner: "Praparn_L"
29     version: "1.0"
30     module: "maindb"
31     environment: "development"
32 spec:
33   replicas: 1
34   template:
35     metadata:
36       labels:
37     spec:
38       containers:
39         - name: maindb
40           image: labdocker/mysql:latest
41           ports:
42             - containerPort: 3306
43               protocol: TCP
44           env:
45             - name: username
46               valueFrom:
47                 secretKeyRef:
48                   name: databasemodule-secret
49                   key: username
50             - name: password
51               valueFrom:
52                 secretKeyRef:
53                   name: databasemodule-secret
54                   key: password
```

ConfigMap and Secret

Secret.yml:

databasemodule_secret.yml

```
1 apiVersion: v1
2 kind: Secret
3 metadata:
4   name: databasemodule-secret
5   namespace: webmicroservice
6   labels:
7     name: "databasemodule-secret"
8     owner: "Praparn_L"
9     version: "1.0"
10    module: "Secret"
11    environment: "development"
12   type: Opaque
13  data:
14    username: cm9vdA==
15    password: cGFzc3dvcnQ=
```

Deploy.yml:

webmodule_deploy_config.yml

```
46 spec:
47   containers:
48     - name: cachedb
49       image: labdocker/redis:latest
50     ports:
51       - containerPort: 6379
52         protocol: TCP
53     - name: webservice
54       image: labdocker/cluster:webservice
55       env:
56         - name: REDIS_HOST
57           valueFrom:
58             configMapKeyRef:
59               name: webmodule_configmap
60               key: REDIS_HOST
61         - name: username
62           valueFrom:
63             secretKeyRef:
64               name: databasemodule-secret
65               key: username
66         - name: password
67           valueFrom:
68             secretKeyRef:
69               name: databasemodule-secret
70               key: password
71       ports:
72         - containerPort: 5000
73           protocol: TCP
```

ConfigMap.yml:

webmodule_configmap.yml

```
1 apiVersion: v1
2 kind: ConfigMap
3 metadata:
4   name: webmodule-configmap
5   namespace: webmicroservice
6   labels:
7     name: "webmodule-configmap"
8     owner: "Praparn_L"
9     version: "1.0"
10    module: "ConfigMap"
11    environment: "development"
12   data:
13     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

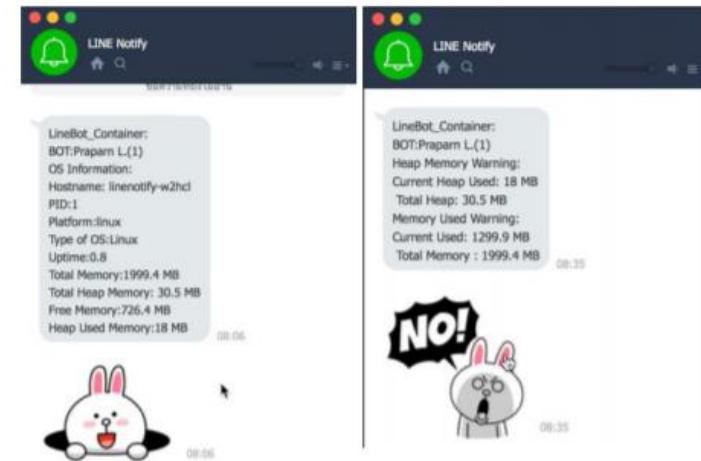
```
kubectl create -f <YAML File>
```

- “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

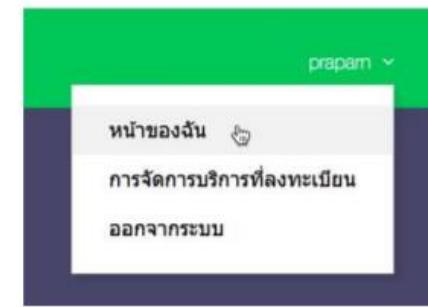
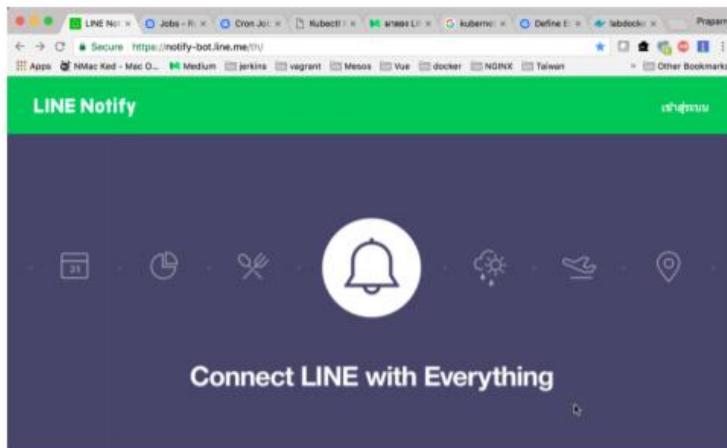
```
1  apiVersion: batch/v1
2  kind: Job
3  metadata:
4    name: linenotify
5    labels:
6      name: linenotify
7      owner: Praparn_L
8      version: "1.0"
9      module: Job
10     environment: development
11   spec:
12     template:
13       metadata:
14         name: linenotify
15       spec:
16         containers:
17           - name: linenotify
18             image: labdocker/linenotify
19             env:
20               - name: TITLE
21                 value: "BOT:Praparn L."
22               - name: INTERVAL
23                 value: "10000"
24               - name: HEAP_HIGH
25                 value: "40"
26               - name: MEM_HIGH
27                 value: "20"
28               - name: SH_OS
29                 value: "Y"
30               - name: TOKEN
31                 value: "EIEqyillzkzpaCo1R0rebZTcGbIyzDZhP0jcdd0t6CX"
```

```
praparns-MacBook-Pro% kubectl create -f job.yml
job "linenotify" created
praparns-MacBook-Pro% kubectl get jobs
NAME      DESIRED   SUCCESSFUL   AGE
linenotify 1/1        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/>



ออก Access Token (สำหรับผู้พัฒนา)

เมื่อใช้ Access Token แบบบุคคล จะสามารถตั้งค่าการแจ้งเตือนได้โดยไม่ต้องลงทะเบียนกับเว็บไซต์

ออก Token

LINE Notify API Document

WORKSHOP: JOB AND CRONJOB

ออก Token

โปรดใส่ชื่อ Token (จะแสดงเมื่อมีการแจ้งเตือน)

LINEBOT

โปรดเลือกห้องแชทที่ต้องการส่งข้อความแจ้งเตือน

Search by group name

รับการแจ้งเตือนแบบล่าสุดจาก LINE Notify

Token ที่ออก

zHOIcJCcpIIS8mEedn [REDACTED]

ถ้าออกจากหน้านี้ ระบบจะไปตรวจสอบ Token ที่ออกใหม่แล้วไป โปรดคลิก Token ก่อนออกจากหน้านี้

ตัดออก ปิด

2017.07.19 23:42

From: LineBot_Container

To: praparn

ยกเลิก

WORKSHOP: JOB AND CRONJOB

- Task1: Create Jobs for Monitor System via LINE

```
1  apiVersion: batch/v1
2  kind: Job
3  metadata:
4    name: linenotify
5    labels:
6      name: linenotify
7      owner: Praparn_L
8      version: "1.0"
9      module: Job
10     environment: development
11
12    spec:
13      template:
14        metadata:
15          name: linenotify
16        spec:
17          containers:
18            - name: linenotify
19              image: labdocker/linenotify
20              env:
21                - name: TITLE
22                  value: "BOT:Praparn L."
23                - name: INTERVAL
24                  value: "10000"
25                - name: HEAP_HIGH
26                  value: "20"
27                - name: MEM_HIGH
28                  value: "20"
29                - name: SH_OS
30                  value: "N"
31                - name: TOKEN
32                  value: "EIEeqyillzkpaCo1R0rebZTcGbIyzDZhP0jcdd0t6CX"
33
34    restartPolicy: Never
```



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)
```



[Documentation](#) [Blog](#) [Partners](#) [Community](#) [Case Studies](#)

Cron Job Limitations

A cron job creates a job object *about* once per execution time of its schedule. We say “about” because there are certain circumstances where two jobs might be created, or no job might be created. We attempt to make these rare, but do not completely prevent them. Therefore, jobs should be *idempotent*. The job is responsible for retrying pods, parallelism among pods it creates, and determining the success or failure of the set of pods. A cron job does not examine pods at all.

Job and Cron Job

- AutoCleanup Job

The screenshot shows a GitHub repository page for `lwolf / kube-cleanup-operator`. The repository has 18 commits, 1 branch, 2 releases, and 3 contributors. The README file contains the following text:

```
Experimental Kubernetes Operator to automatically delete completed Jobs and their Pods. Controller listens for changes in Pods created by Jobs and deletes it on Completion.
```

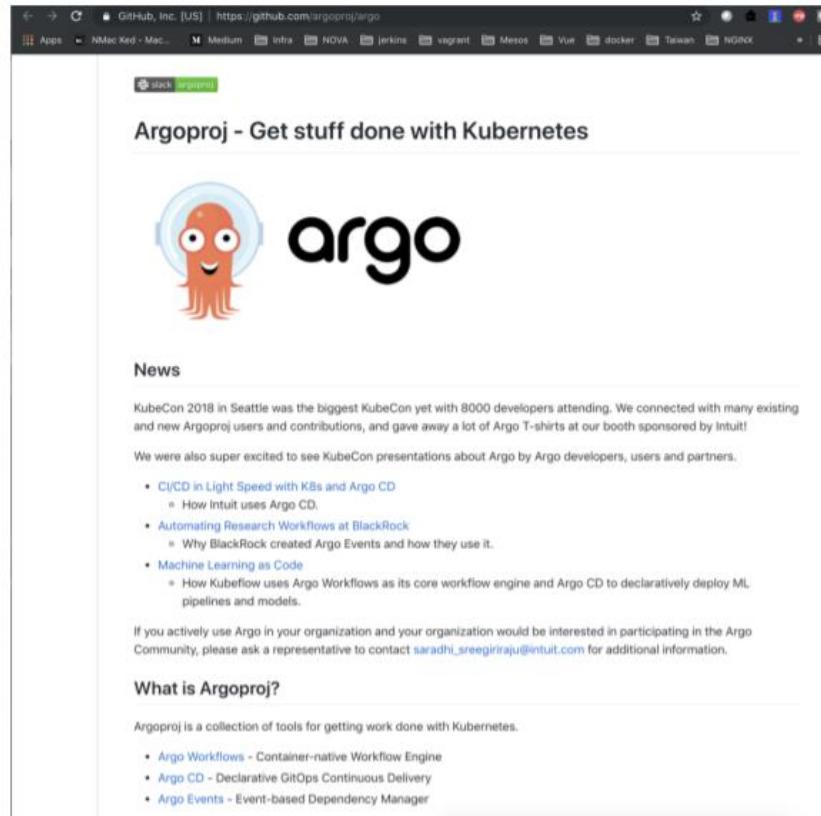
Some defaults:

- All Namespaces are monitored by default
- Only Pods created by Jobs are monitored

The URL of the repository is <https://github.com/lwolf/kube-cleanup-operator>.

Job and Cron Job

- WorkFlow Job

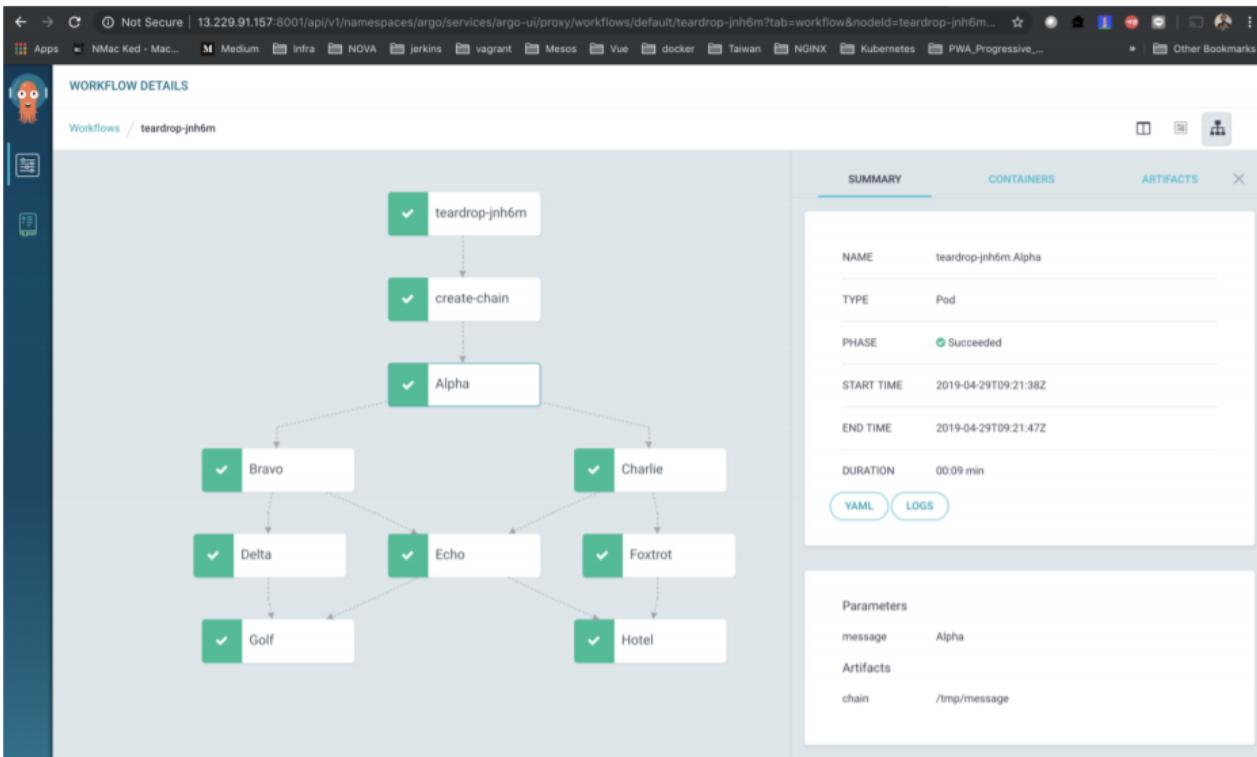


The screenshot shows the GitHub repository page for argoproj/argo. The page features a large orange octopus logo with the word "argo" next to it. Below the logo is a "News" section with two paragraphs about KubeCon 2018 and KubeCon presentations. There is a bulleted list of topics related to Argo CD and KubeFlow. At the bottom, there is a "What is ArgoProj?" section with a bulleted list of tools: Argo Workflows, Argo CD, and Argo Events.

<https://github.com/argoproj/argo>

Job and Cron Job

- WorkFlow Job



HORIZONTAL POD AUTOSCALING(HPA)

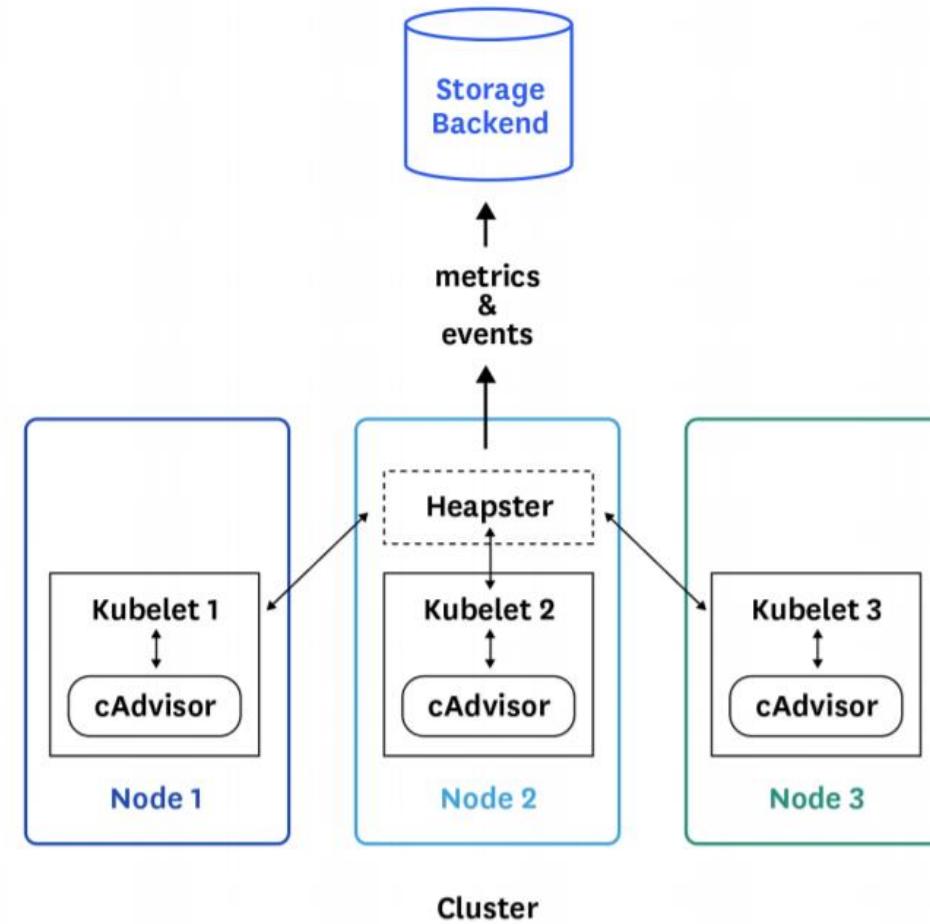
- Horizontal Pod Autoscaling (HPA)
 - The scale is easy to operate with "kubectl scale --replicas=XXX deployment/<name>" with single command
 - But..
 - How can you scale meet up/down actual required ?
 - HPA will response for monitor workload on Pods (Now base on CPU) and automatic trigger deployment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

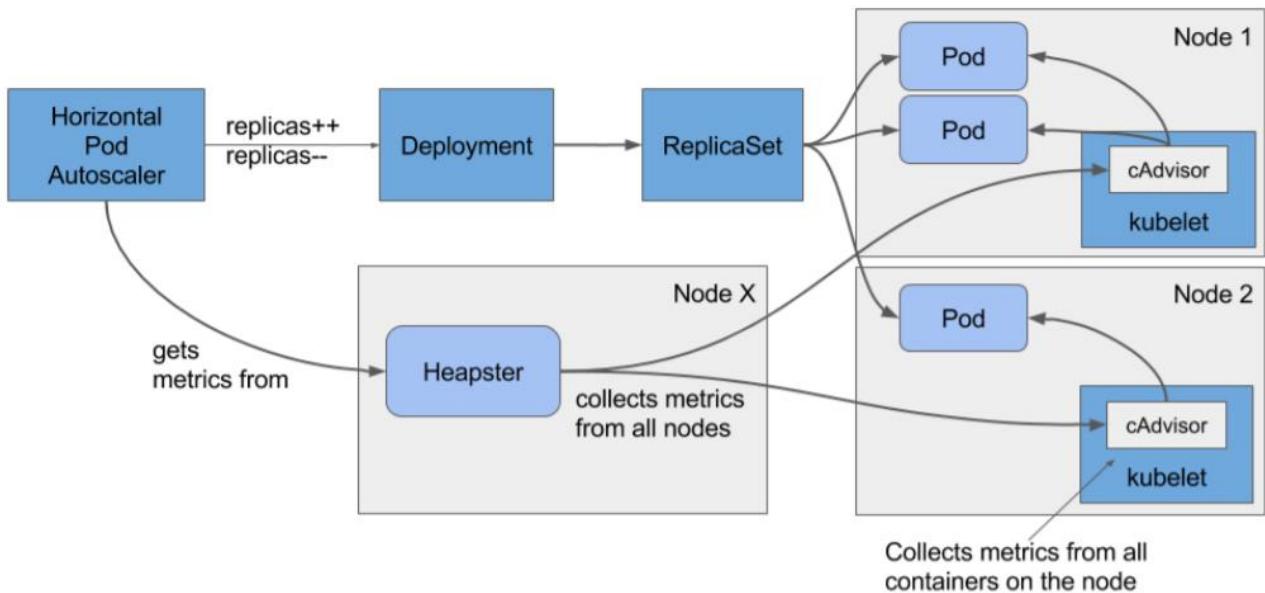
```
kubectl autoscale <type/name> <option:>
```

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

HORIZONTAL POD AUTOSCALING(HPA)

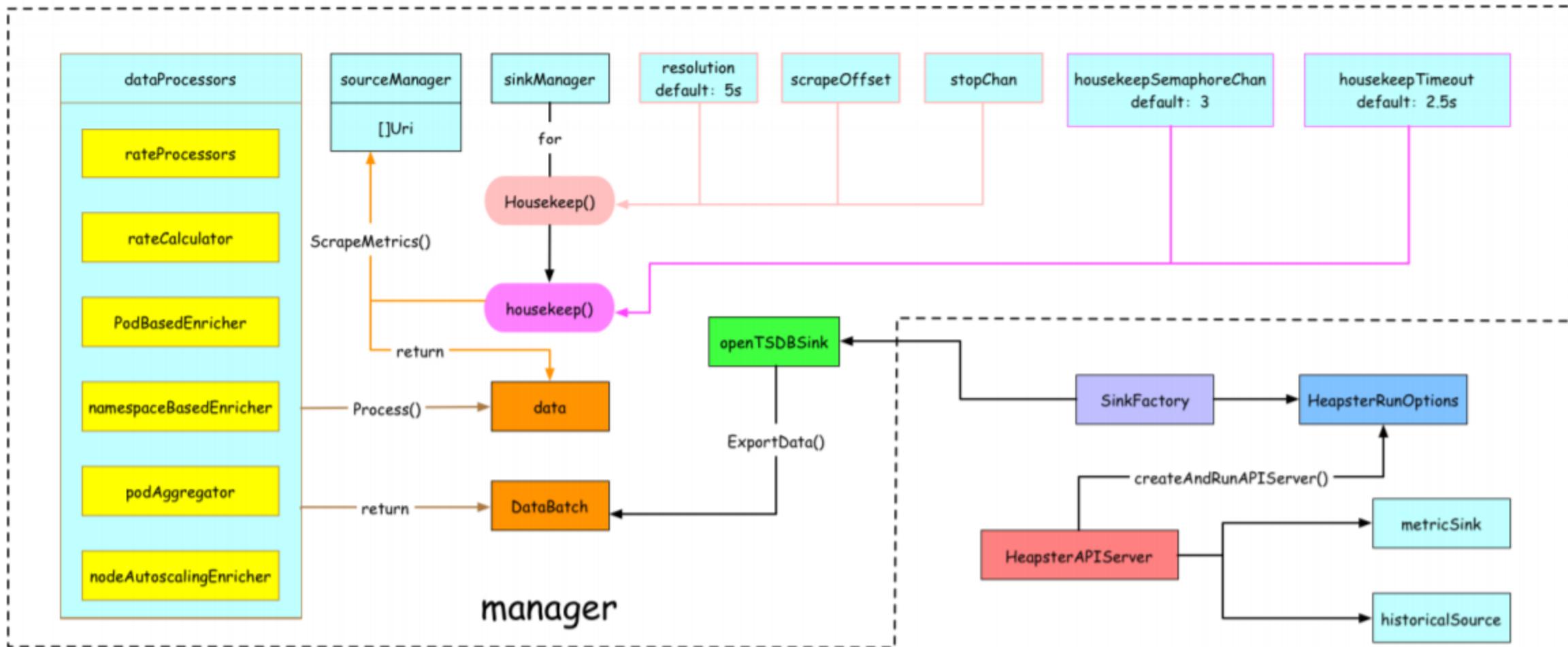
HORIZONTAL POD AUTOSCALING(HPA)





HORIZONTAL POD AUTOSCALING(HPA)

Heapster



HORIZONTAL POD AUTOSCALING(HPA)

the cadvisor works well, because the dashboard shows workload metrics.

gjmuzj commented on Dec 28, 2017

I found a solution in my case:
`kube-controller-manager`'s parameter `--horizontal-pod-autoscaler-use-rest-clients` default value is `true`, while in k8s 1.8.x is `false`. change it to `false` and it works.

MaciekPtytel commented on Dec 28, 2017

With `--horizontal-pod-autoscaler-use-rest-clients=true` HPA uses new instead of old way of getting metrics. Setting it to `false` as you did works long term solution is to run metrics server as part of your cluster set up. To https://kubernetes.io/docs/tasks/debug-application-cluster/core-metrics-pipeline/

Starting with v1.9 HPA uses new resource metrics API that is not available in CDK out of the box. If you want to use autoscaler you should do it:

```
tee-master controller-manager-extra-args="--horizontal-pod-autoscaler-use-rest-clients=false"
```

Based on the discussion here kubernetes/kubernetes#57673 we can either set the --horizontal-pod-autoscaler-use-rest-clients=false or deploy this Metrics server: https://kubernetes.io/docs/tasks/debug-application-cluster/core-metrics-pipeline/ Issue reported here as well: kubernetes/kubernetes#57998

itsakalosse commented 2 days ago

Hyperbolic2348 commented 2 days ago

The metrics server is deployed by default in kube-up. I would think we should deploy it as well.

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.

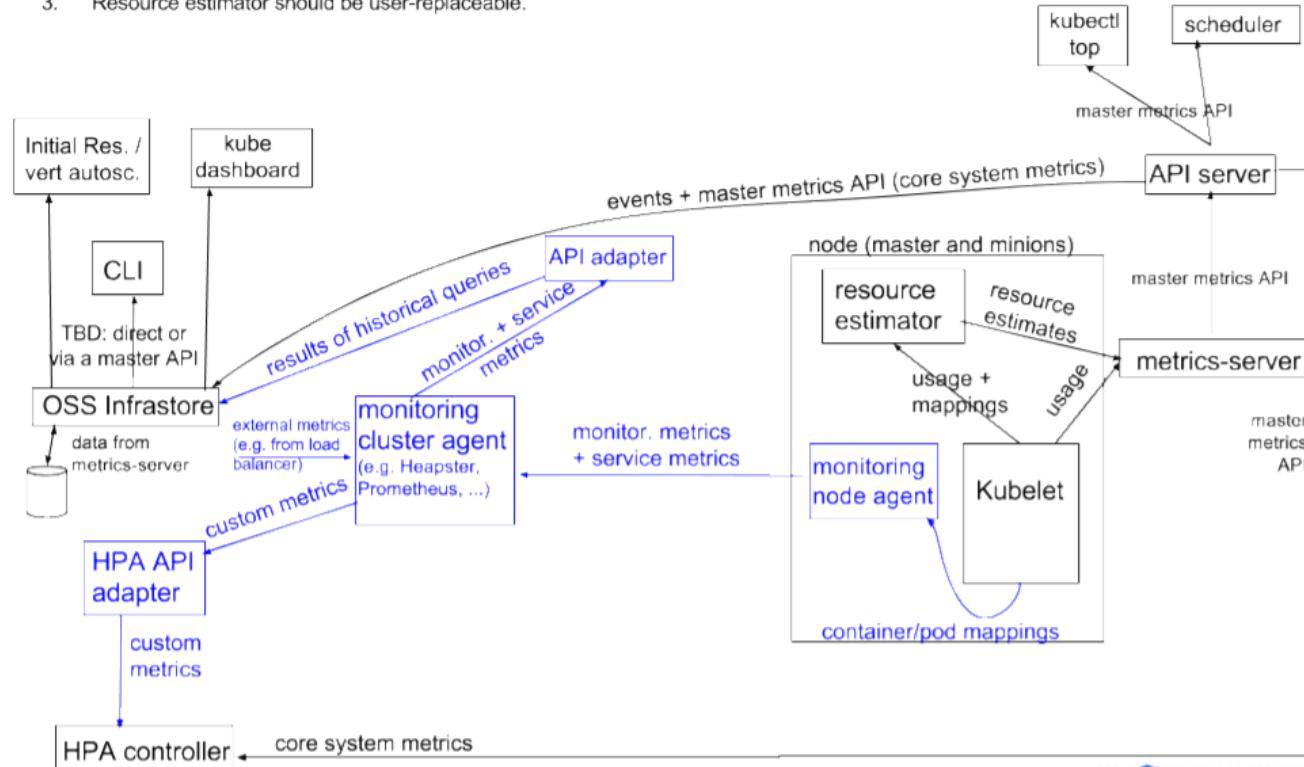
Note: The API requires metrics server to be deployed in the cluster. Otherwise it will be not available.

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

Code Issues 28 Pull requests 7 Projects 0 Wiki Insights

Branch: master · heapster / docs / deprecation.md Find file Copy path

AdamDang Typo fix: encorgaged->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

Code Issues 2,196 Pull requests 942 Projects 12 Insights

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

DirectXMan12 Bump metrics-server to v0.3.1

Latest commit 13d59fd on Sep 10, 2017 by DirectXMan12

... OWNERS Add kawych to Metrics Server owners 10 months ago

README.md Add Troubleshooting sections to Heapster and Metrics Server addons do... 8 months ago

auth-delegator.yaml Made metrics-server critical service managed by addon-manager a year ago

auth-reader.yaml Made metrics-server critical service managed by addon-manager a year ago

metrics-apiservice.yaml Bumped Metrics Server to v0.2.0 a year ago

metrics-server-deployment.yaml Bump metrics-server to v0.3.1 2 months ago

metrics-server-service.yaml Made metrics-server critical service managed by addon-manager a year ago

resource-reader.yaml Autoscaler metrics-server with pod-nanny a year ago

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